DOI 10.15622/ia.2022.21.7

O. FUENTES, J. SAVAGE, L. CONTRERAS

A SLAM SYSTEM BASED ON HIDDEN MARKOV MODELS

Fuentes O., Savage J., Contreras L. **A SLAM System Based on Hidden Markov Models.**

Abstract. Methods of simultaneous localization and mapping (SLAM) are a solution for the navigation problem of service robots. We present a graph SLAM system based on Hidden Markov Models (HMM) where the sensor readings are represented with different symbols using a number of clustering techniques; then, the symbols are fused as a single prediction, to improve the accuracy rate, using a Dual HMM. Our system's versatility allows to work with different types of sensors or fusion of sensors, and to implement, either active or passive, graph SLAM. A graph-SLAM approach proposed by the International's Karto Robotics in Cartographer, the nodes represent the pose of the robot and the edges the constraints between them. Nodes are usually defined according to contiguous nodes except when loop closures are detected where constraints for non-contiguous nodes are introduced, which corrects the whole graph. Detecting loop closure is not trivial; in the ROS implementation, scan matching is performed by Sparse Pose Adjustment (SPA). Cartographer uses an occupancy map in order to estimate the position where the map representation is done via Gmapping. The Toyota HSR (Human Support Robot) robot was used to generate the data set in both real and simulated competition environments. In our SLAM representation, we have wheel odometry estimate according to initial position of the robot, a Hokuyo 2D Lidar scan for observations, and a signal control and a world representation is estimated. We tested our system in the kidnapped robot problem by training a representation, improving it online, and, finally, solving the SLAM problem.

Keywords: localization, SLAM, robot navigation, mapping, Hidden Markov Model, sensor fusion, service robot

1. Introduction. Service robots, such as the Toyota HSR [1], are increasingly becoming a part of our everyday life, so the ability to explore, map, and navigate its surroundings is of the utmost importance. SLAM or simultaneous localization and mapping is a solution for this problem.

There are many accepted and well studied methods for solving SLAM, a brief overview of the main paradigms used for solving SLAM is presented on section 2. Depending on the application, one approach might be better suited than other. The sensors information available is also an important factor to decide which approach to use, e.g. Wheel information is an efficient way of estimating small changes in position, however a drone would not have this valuable information. Taking into consideration these differences may favour a SLAM method or type of sensor in a specific environment or even a specific region in an environment, e.g., a dark corner may be a terrible place to use image-based methods, on the other hand, a colored flat wall would render little information to a LIDAR-based one.

We propose a graph SLAM system based on HMM's (Hidden Markov Models). Our method can take advantage of several types of sensors measurements (or sensor fusion) while estimating a graph topological representation. Dual-HMM allows us to use two different quantizers-simultaneously, effectively fusing data at a symbol level. Furthermore, new nodes can be added without modifying the graph (modularity). They can be grown or altered without training an entirely new model.

In summary, our main contributions are:

– A graph SLAM system based on Hidden Markov Models.

– A modular system capable of using a wide variety of sensors and features.

– An autonomous training method.

– A navigation method capable of obstacle avoidance.

– A robust localization method.

The remaining of the paper is divided as follows. In Section 2 we present a summary of the SLAM problem and the HMM-based approaches. Then, in Section 3 we introduce our probabilistic approach to the graph SLAM problem and in Section 5 we describe the experimental results. Finally, the main findings are discussed in Section 6.

2. Related Work. The three main SLAM paradigms are Kalman Filter, Particle Filter and Pose graph based implementations, and all of these can be found in the most commonly used open-source libraries.

2.1. EKF SLAM. The Extended Kalman Filter SLAM [2, 3] is one of the most accepted SLAM solutions; it consists of three basic operations:

2.1.1. Robot Movement. The agent moves increasing its position uncertainty due to odometry errors.

$$S_t \longleftarrow f(s_{t-1}, u, n),$$

where:

– f - motion model,

– S_t - state of the robot at time t,

– u - control signal,

– n - noise.

2.1.2. Discovery. The agent finds new interesting landmarks, which need to be referenced. The position uncertainty and sensor error readings are modeled using an inverse observation model i.e. where a landmark is in the map, given the scene seen by the robot.

$$L_i = g(S_t, \vec{O}_t, y_i),$$

where:

- g , Direct Observation model,
- L_i - i-th landmark,
- S_t - robot state at time t,
- y_i - measure of i-th landmark.

2.1.3. Re-Discovery. The agent finds a previously mapped landmark and re-estimates both its position and landmark position.

The extended Kalman filter has a "stage" for each of the above operations, making it a useful estimator for propagating the uncertainty related to the three mentioned actions.

$$y_i = h(S_t, \vec{O}_t, L_i),$$

where:

- h - Indirect Observation model.

The map representation itself is a matrix that stacks vectors of all mapped landmarks on any given robot state

$$map = \begin{bmatrix} S \\ L_1 \\ . \\ . \\ . \\ L_I \end{bmatrix},$$

where:

- S - robot state.

2.2. Particle Filtering. One of the most important characteristics of this SLAM approach is the building of an occupancy grid map. This map is later used to achieve localization with solutions similar to de EKF filtering, hence the name Filtering in Particle Filtering. Localization is achieved with Adaptative Monte Carlo Localization AMCL, a member of Markov localization algorithms. Murphy [4] et al. proposed Rao-Blackewllized Particle Filters as a SLAM solution; the key idea of the Rao-Blackwellized particle filter for SLAM is to estimate the joint posteriori $P(\vec{x}_t, m \mid \vec{z}_t, \vec{u}_{t-1})$. The Rao-Blackwellized particle filter for SLAM makes use of the following factorization:

$$P(x_{1:t}, m \mid z_{1:t}, u_{1:t-1}) = P(m \mid x_{1:t}, z_{1:t}) P(x_{1:t} \mid z_{1:t}, u_{1:-t-1}),$$

where x is the pose of the robot, m is the map, Z_t are the observations at time t, and U_t is the control signal.

Importance Sampling Filters (such as Sequential Importance Resampling or SIR) are employed to estimate the posterior $P(x_{1:t} \mid z_{1:t}, u_{1:-t-1})$, as mentioned by Grisetti et al. in [4], where "each particle represents a potential trajectory of the robot. Furthermore, an individual map is associated with each sample. The maps are built from the observations, and the trajectory represented by the corresponding particle".

2.3. Graph SLAM. According to Grisseti et al., "one intuitive way of formulating SLAM is to use a graph whose nodes correspond to the poses of the robot at different points in time and whose edges represent constraints between the poses" [6].

From a probabilistic point of view, SLAM can be represented with a sequence of random variables as X (robot pose), M (map features), Y (sensor readings), and U (robot motion). To solve the SLAM problem, we simply use the maximum a posteriori probability:

$$P(X_t, M \mid Y_t, U_t, x_0),$$

for each time step t.

There has been a lot of work on solving this estimation problem [6] [7]. However, special attention has been given to [8], where the authors apply variable elimination techniques to reduce the dimensionality of the optimization problem. In the work presented in [9], a solution to the active SLAM problem is proposed "in scenarios in which some prior information about the environment is available in the form of a topo-metric graph".

2.4. ROS SLAM. As mentioned before, many state-of-the-art solutions can be found as open-source libraries in popular frameworks such as the Robot Operating System (ROS). Here, we present some of the most popular implementations.

2.4.1. Gmapping. Gmapping [10] is based on Rao-Blackwelized Particle Filters proposed by Grisetti et al. in [11]. Particle Filters are a known application of Bayesian Filters in which a large number of importance weighted particles represent the a-posteriori probability; a probabilistic occupancy grid is used as a map representation, and AMCL, Adaptative Montecarlo Localization, is used for localization in this map – AMCL is a member of the Markov localization algorithms family.

2.4.2. HECTOR SLAM. HECTOR SLAM does not use odometry information for the localization; it rather uses high update rates and low distance measurements to estimate the robot movement. A version of ICP is used to estimate the pose between samples and maintain the robot pose estimate.

2.4.3. CARTOGRAPHER. A graph-SLAM approach proposed by the International's Karto Robotics in [12] – in Cartographer, the nodes represent the pose of the robot and the edges the contraints between them. Nodes are usually defined according to contiguous nodes except when loop closures are detected where contraints for non-contiguous nodes are introduced, which corrects the whole graph. Detecting loop closure is not trivial; in the ROS implementation, scan matching is performed by Sparse Pose Adjustment (SPA) [13]. Cartographer uses an occupancy map in order to estimate the position where the map representation is done via Gmapping.

As we can see from this brief review, some methods are better suited for specific tasks or robot architectures. Our research is based on the idea that each method has its strengths, and we propose a method that, focusing on the versatility of application, tries to take some of the advantages inherent to these methods.

3. HMM-based graph-SLAM. We propose a versatile graph-slam system based on Hidden Markov Models; the goal is to estimate a topographic graph given noisy sensor measurements and pose estimates. In this section we will briefly describe the core concepts in our implementation.

3.1. Hidden Markov Models. A Hidden Markov Model (HMM) is a two random variable stochastic process in which only one of the random variables is directly observable. In its discrete version, and provided the Markov property [14] is fulfilled, the system dynamics is fully defined by a transition matrix A, an emission matrix B, and, optionally, an initial conditions vector $\vec{\pi}$.

$$a_{ij} = P\left(S_t = s_j \mid S_{t-1} = s_i\right),$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ a_{31} & a_{32} & \dots & a_{3N} \\ \cdot & \cdot & \cdot & \cdot \\ a_{N1} & a_{N1} & \dots & a_{NN} \end{bmatrix},$$

$$b_j(k) = P(V_k \mid q_t = S_j),$$

$$B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1M} \\ b_{21} & b_{22} & \dots & b_{2M} \\ b_{31} & b_{32} & \dots & b_{3M} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ b_{N1} & b_{N1} & \dots & b_{NM} \end{bmatrix},$$

$$\lambda = [A, B, \vec{\pi}].$$

We propose the hidden random variable to represent the pose of the robot, and the observable variable will be a discrete representation (quantization) of the sensor measurements. The dynamic programming based algorithms' family related to HMM's [15] provide efficient estimations to probabilities related to the localization problem.

Our approach on Markov localization uses a discrete code-book to represent the sensor's readings. The symbolic representation of the readings allows for any sensor or fusion of sensors to be used as observations. Localization is achieved, as any other Markov localization approach, by estimating the probability of being on state $S_t = s_n$ given a last known state $S_{t-1} = s_n$, a sensor(s) observation symbol $O_t = V_k$, and a control signal $U(t)$. The 2D pose state space is defined by the tuple $\langle x, y, \theta \rangle$.

For each time step t, random variable S can take the value of a discrete set of N states; similarly, the observations of each time step will be a member of the discrete code-book observation set of K symbols V. Control signal $U(t)$ is assumed constant, and only an on-off signal for the potential fields autonomous navigation system is necessary, as follows:

$$S_{t-1} = s_n, O_t = V_k, U(t) = u.$$

An action set is used for the active slam pose belief exploration. The planning path is then obtained using a search algorithm like Dijkstra or A*. Pose traces are proposed, so given a state belief and an action the most likely future state is estimated.

The algorithms typically associated with HMM's can efficiently estimate this probability using dynamic programming. Even though a number of states in the models are reported as fixed, it was done to have uniformity between models, since models are modular and can easily be connected to form a global model, similar to sub-maps in Cartographer. The balance between computer costs and accuracy can be fine-tuned with the number of states, and the observations code-book size.

The resulting representation of the environment is a symbolic pose graph, where each node or state is the centroid of the free region surrounding it; orientation is also accounted for with a scaling factor. The HMM model is completed by recording all the observations in the code-book sensed while the agent is at any given state. As the agent acquires more information by exploration, new states are formed, or the existing ones are moved to account for new data, eventually including all the explored areas. The number of total states is a hyperparameter of the model, and it can be seen as a scale factor to be used. The more states in which free space is divided, the more accurate the correction is at the expense of more computational expenses.

One of the main advantages of the method is adaptability, since it can use any sensor or fusion of sensors. As briefly mentioned above, observations are not used directly. A symbol, part of a discrete data set, is used to represent sensor readings. This quantization of readings allows the method to be used with various sensors, different sensor coding, or even sensor fusion. In HMM terms, we have various emission matrices for each transition matrix. We perform estimation on various models, each related to each sensor, or fusion of sensors. The model likelihood given a set of observations is easy to obtain with HMM algorithms like Forward Algorithm, (eq. 2) and its Backward counterpart,(eq. 3); and it is used as a metric to decide which model to use on a specific area. A lidar will gather more information from a dark corner, while a camera would do better on a long hallway with distinctive visual landmarks. Computational cost is in the order of N^2T (eq. 4). However, the number of states rapidly increases the computational cost, "the curse of dimensionality", where the direct calculations without dynamic programming algorithms are in the order of N^T (eq. 1).

$$P(\vec{O} \mid \lambda) = \sum_{j=1}^{S} \left(\pi_{s1} \prod_{t=1}^{i=T} a_{s_t . s_{t+1}} b_{s_t - s_{t+1}}(o_i) \right), \tag{1}$$

$$\alpha_{t+1}(i) = \sum_{i=1}^{N} (\alpha_t(i) a_{i,j}) b_j(o_{t+1}), \tag{2}$$

$$\beta_t(i) = \sum_{i=1}^{N} (a_{i,j} b_j(o_{t+1}) \beta_{t+1}(j), \tag{3}$$

$$P(\vec{O} \mid \lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i). \qquad (4)$$

Finally, localization is performed using the Viterbi algorithm; this algorithm estimates the most probable sequence of states given a sequence of observation symbols. The model with maximum likelihood is obtained with the Forward Algorithm. The most likely model given the readings is used in the Viterbi algorithm, with $max\left[P(\vec{S} \mid \lambda, \vec{O})\right]$ being the discrete representation of the space in which each state is the closest centroid to the robot state at the time of sensor capture. The size of the free region represented by each state is variable, but it is kept to a small enough area to make wheel odometry reliable. The transitions between states happen on well-defined regions normally distributed around a transition point. The most frequent transition point between states is used as a "reset" wheel odometry according to that point. Similarly to cartographers' sub-maps, wheel odometry is reliable in small regions.

4. Implementation. We have presented all the components of the system we propose in a very general way. Now, we will show some interesting implementations using those components. It is important to mention that a map is not necessary for our method but, if one is available or needed, given that our method deals with the same probabilities as the particle filter used in AMCL [16], it is possible to use Rao-Blackwellization. Furthermore, as our training methods create a decent grid estimation, whether using a map or not is greatly dependant on the application and error scale needed.

In our implementation, a global reference frame is created. However, global mapping based on iterative closest point (ICP) methods is also a possibility we explored in some regions, especially when a high.

In our SLAM representation, we have wheel odometry estimate according to initial position of the robot, a Hokuyo 2D Lidar scan for observations, and a signal control and a world representation is estimated. The HMM Model is trained as follows.

4.1. Training. Short exploration runs are conducted, gradually restarting to a known value, like an entrance or a recharging station; this episodic learning approach makes the wheel odometry reliable. The duration of the episodes is such that wheel odometry is reliable. The control signal in exploration mode is an on/off signal enabling a reactive potential fields behavior [17]. Some additional constraints are added to the behavior, as mentioned in our previous work [18], like an exploratory turn every given time sample, or an artificial attraction towards doors.

Model training is done with a labeled training set. This training set is formed by a vector $\vec{O}_t$ that contains all the 720 laser readings from a laser sensor and an odometry-based pose vector or hidden state S_k.

Baum-Welch [15] is the most commonly used method for the HMM model estimation. It is an elegant approach that calculates the optimal model parameters given the observations; it is also an Expectation-Maximization (EM) algorithm [19]. The optimization is done by maximizing the Likelihood of the Model given some readings. The most common training method is used offline and will serve as an initial estimation for our method. We maximize the likelihood of the model given some observations (training set). An auxiliary variable ξ is defined.

$$\xi_t(i,j) = P(s_t = i, s_{t+1} = j | o_t, \lambda),$$

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{T+1})\beta_{t+1}(j)}{P(o|\lambda)}$$
$$= \frac{\alpha_t(i)a_{ij}b_j(o_{T+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{T+1})\beta_{t+1}(j)},$$

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j),$$

$$\sum_{t=1}^{T-1}\gamma_t(i) = S_i^T,$$

where S_i^T is the number of transitions from S_i over T time steps.

$$\sum_{t=1}^{T-1}\xi_t(i,j) = S_{ij}^T,$$

where S_{ij}^T is the number of transitions from S_i to S_j over T time steps.

$$\hat{a_{i,j}} = \frac{S_{ij}^T}{S_i^t} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(j)},$$

$$\hat{b_i(k)} = \frac{\sum_{t=1}^{T}1_{o_t=v_k}\gamma_t(j)}{\sum_{t=1}^{T}\gamma_t(j)}.$$

This variable can be easily represented with the Backward and Forward variables, as follows [15] uses $\sum_{t=1}^{T} 1_{o_t=v_k}$ to represent number of times V_k is seen in state j.

We also use a gradient descent implementation of the Baum-Welch algorithm [20] to keep optimizing the model online. This can introduce subtle changes when the environment changes or when a new area is mapped and added to the global model. A mini-batch approach can also be used to re-estimate the model every time a batch buffer is filled. Each new reading contributes to an infinitesimal change in its respective row and column in the transition matrix. Note from eqs. 5 and 6 that changes only happen on the row and column related to the reading. The least frequent poses will be absorbed by the more frequent (desired) readings. This online process allows aligning the new readings and reinforcing the optimal values given a space and obstacles configuration into the matrix.

$$\frac{\partial P}{\partial a_{ij}} = \sum_{t=1}^{T} \alpha_t(i) b_j(O_{t+1}) \beta_{t+1}(j), \tag{5}$$

$$\frac{\partial P}{\partial b_j(O_t)} = \left\{ \begin{array}{cc} \delta_{j1}\beta_1(j) & t=1 \\ \sum\limits_{i=1}^{N} \alpha_{t-1}(i) b_{ij}(O_{t+1}) \beta_t(j) & t \neq 1 \end{array} \right. . \tag{6}$$

Due to the low complexity for estimating a model and, later on, navigating the model to obtain information, it is possible to use several models simultaneously; this allows us to add new nodes (i.e. states) to the model without the need to re-estimate the whole model.

In case autonomous mapping is not required, and human interaction is used to "explore" the scene, the model will benefit from different strategies, and active policy estimate can be obtained. Again, versatility is an important characteristic of our method. Regardless of how the training set is obtained and the model estimated, that is, either offline, online , batches, etc., an HMM is obtained with the Baum-Welch algorithm. Figure 1 shows a topological node graph obtained from the transition matrix of the proposed HMM after online navigation. Results found in Section 5 and Figure 10 and Figure 11 show this process in a standard competition arena where an apartment is explored, and an initial graph is estimated; the green arrows represent the state <x,y,θ>. The background image is just a reference for illustrative purposes since the map is not being used.

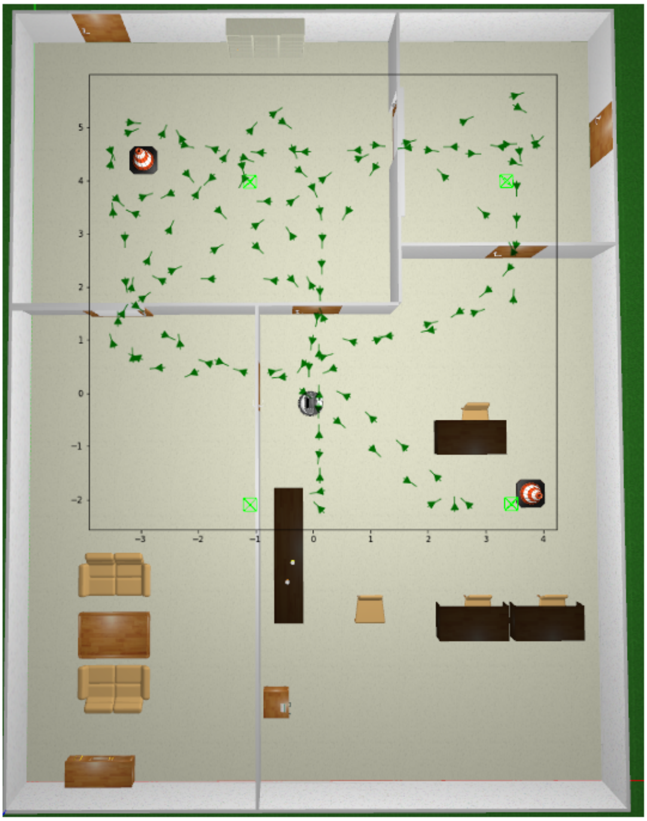

Fig. 1. Topographical graph representation, obtained from the transition matrix of the HMM trained. Graph is created with the states corpus (2D poses) shown as green arrows

4.2. Observations. Observations used for the estimation of the HMM are quantized (and finite) in nature, so the method used for the quantization or the kind of the sensor itself is not important as long as we use a finite code-book to represent them. We explored various observation alphabets for the code-book where Lidar readings were treated using different clustering techniques, namely K-means and affinity propagation (Figure 2).

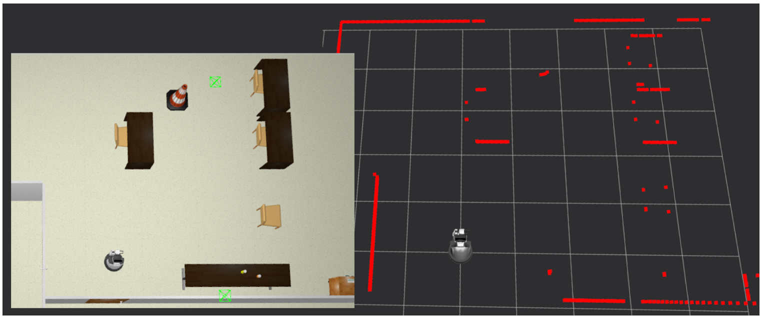

Fig. 2. Apartment scene with its corresponding Hokuyo Lidar reading

Some experiments were also done by fusing LIDAR and 2D images features obtained with Resnet [21] architecture. The final pose estimator efficiency greatly depends on good readings. "A good reading is that which can be re-observed, and one easily differentiates from other reads" [22].

4.2.1. Lidar. K-means (or mini K-means in this case) works well when a large amount of data is used. The low occurrence of outliers takes them out of the estimation; however, highly repeated values will "skew" the means resulting in some information loss. A different algorithm is proposed (Affinity Propagation) [23] to quantize the laser readings based on their similarity, rather than their frequency of appearance. A dual HMM is introduced, which estimates independently using code-books from each clustering algorithm, and trusting only matching estimates obtained independently by each model. A block diagram of the proposed DUAL HMM architecture can be seen in Figure 3 . The statistical characteristics of most readings are quite similar. This similarity enables the usage of affinity propagation clustering techniques to generate more heterogeneous clusters; however, the drawback is the lack of scalability when there are many samples. On the other hand, K-means works well with large data sets and being a Euclidean distance approach, it can be used online (with the previous centroids).

In sum, we use measurement symbols as anchors of information and depending on the sensor's nature, on the environment conditions, and even on the route, one kind of symbol generation might result in a better performance.

A robust SLAM system should benefit from all symbol representations; Sensor Fusion [24] is used in this way, yet we propose a much simpler approach:

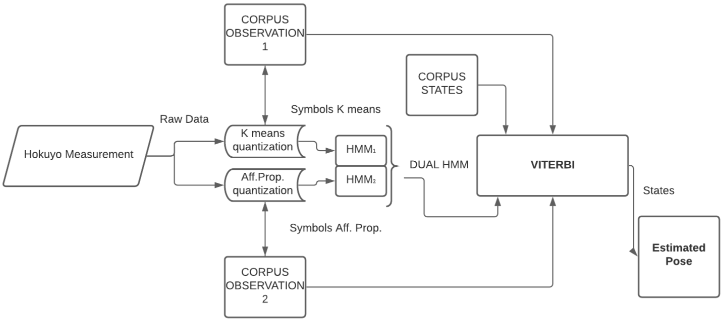

Fig. 3. Block diagram of the proposed DUAL-HMM system

Dual-HMM. In our experiments, we use two different features extracted from the same observations vector, but this applies to any kind of observation.

Figure 4 shows a common scene of a service robot, and on the right side is an example of a typical lidar measurement in green; the centroid obtained with K means in orange and the Affinity Propagation exemplar in green.

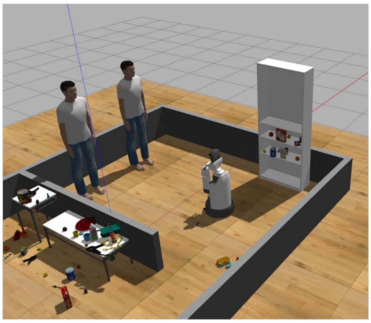
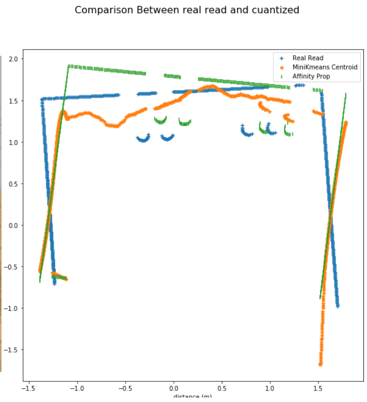

Fig. 4. Example of a Hokuyo typical reading (blue) with its K-means centroid (orange), and affinity propagation exemplar (green)

The final pose estimator's efficiency greatly depends on good readings "a good reading is that which can be re-observed, and one easily differentiates from other readings" [22]. K-means (or mini K-mean, in this case) works well when a large amount of data is used since the low occurrence of outliers takes them out of the estimation; however, highly repeated values will skew the means. Affinity Propagation [23] is another clustering algorithm, and it was used to quantize the Hokuyo readings based on their similarity rather than their frequency of appearance. Figure 5 shows K-means information loss in the most observed symbol. In comparison, Figure 6 shows the symbols assigned by the more computationally expensive Affinity Propagation to a series of new readings.

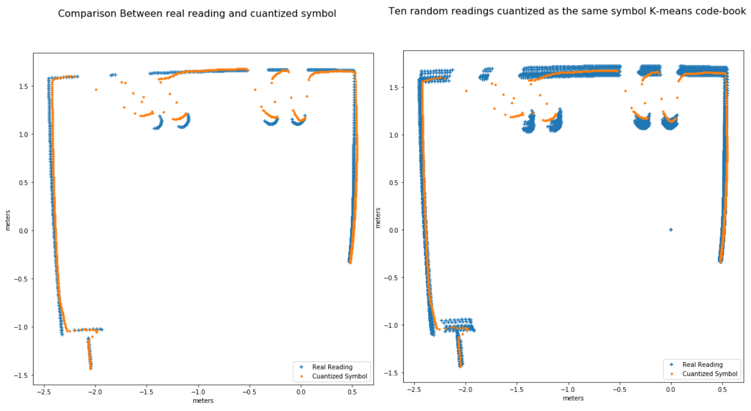

Fig. 5. Most common read and 10 samples clustered in the same K-means alphabet symbol

In our approach, a dual HMM makes independent estimates using symbols from each clustering algorithm. Although there are other methods to achieve sensor fusion, in our proposal, we use a simple symbol level data fusion.

4.2.2. Resnet feature extraction. Another observations alphabet was quantized by getting an observation vector $\vec{O}_t$ from the last layer of a Resnet 50 CNN (Convolutional Neural Network). Such vectors were quantized into an alphabet and their respective emission matrix estimated. All models share the same transition matrix. Results reported in section 5 show models using the fusion of a Resnet symbol with a lidar K-means symbol.

4.3. Localization. Once a model, or several models, are found, it is possible to estimate the robot's pose with a set of past observation symbols $\vec{O} = V_t, V_{t-1}, \ldots, V_{t-M}$. The most direct approach is to the Viterbi algorithm [25] that takes a sequence of quantized observations $\vec{O}$, an HMM model λ, and an initial conditions vector $\vec{\pi}$, and yields the most probable state sequence traversed by the system given the observations and HMM model.

$$P_1 = \pi_i \cdot b_i(o_1),$$

$$P_t(j) = \max_{1<i<N} [P_{t-1}(i) \cdot a_{ij}] \cdot b_j(o_T),$$

$$S_t(j) = arg \max_{1<i<N} [P_{t-1}(i) \cdot a_{ij}],$$

$$P\left(\vec{S} \mid \vec{O}, \lambda\right).$$

The length M of this observations vector is called the Viterbi buffer, and it is a hyperparameter that should be trained to optimize the models. Forward and Backward algorithms are used to find the most common state sequence and the probabilities of being in a specific state given the observations.

4.4. Pose Correction. Wheel odometry is an easy way to estimate a pose with the information from the wheels and the control signals. This estimation is reliable in the short term because a small error – due to slips and control noise – is accumulated over time, i.e. a reference point must be maintained. The accumulated error makes the estimation unreliable in the long run; this error is normally distributed [26] so, it is possible to characterize a particular floor-wheel interaction odometry error.

In our proposal, wheel odometry is corrected by resetting the reference point where the correction is made when a state transition is detected via the

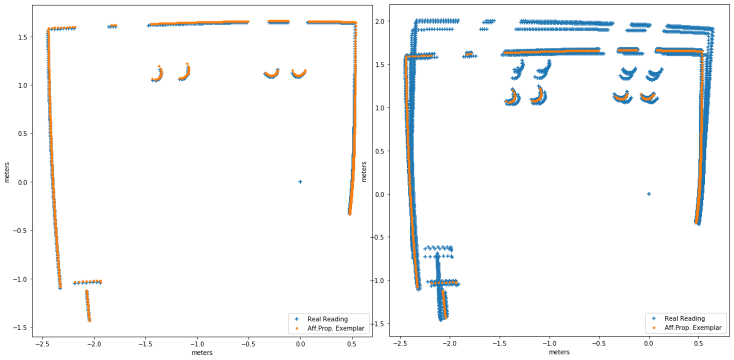

Fig. 6. Most common read and 10 samples clustered in the same affinity propagation alphabet symbol

Viterbi algorithm. It is easy to observe from the training set that the transitions from one state to another usually happen approximately in the same pose value, i.e. transitions happen in a normally distributed point with the same distribution as the wheel odometry error; this value is used as the correction value on trusted transitions. Wheel odometry can be trusted inside the small region related to a specific state or centroid or time interval.

Figure 7 shows in green the position estimate of the Wheel odometry in a run traversing through two states; black points show the real pose of the robot. Two independent HMM's correct the position according to their own estimates – it can be seen in the upper part both models estimate the same pose, and correction is the same.

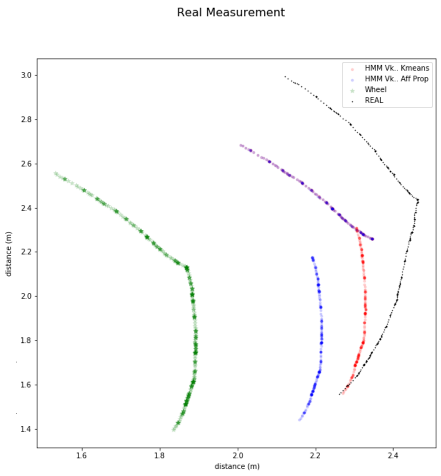

Fig. 7. Real pose in black and three different pose estimation methods. Wheel odometry in green, a single HMM (K-means) in blue and single HMM (aff.prop.) in blue

4.5. Navigation. After the training process, the possible states of the system are represented with the pose centroids corpus; the topological map used by the Dijkstra algorithm is a version of the HMM's transition matrix. We propose different kinds of nodes; some of them are bidirectional, others one-directional, some others only landmarks used to re-estimate the position. This is called *labeling* in some literature relating to HMM and must not be confused with the labeling of the training set. Again, the graph nodes (or pose centroids) are re-estimated online with each reading, and they tend to align with the optimal route as the robot navigates by being attracted to a single centroid at a time (virtual attractor). Dijkstra route will be then a path containing the sequence of centroids to visit to get from a current position to a goal. Potential fields reactive behavior is used to sequentially visit these virtual attractors while avoiding unmapped or dynamic obstacles. Figure 8 shows a sequence of images of the robot traversing a path made of such attractors – the control signal is simply an on/off signal, and the robot is either trying to get to the next attractor or it is static on a fixed position.

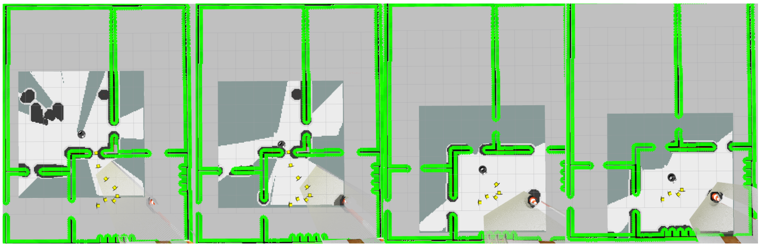

Fig. 8. Different moments of reactive navigation through a route formed by virtual attractors. Route is produced after planning using Dijkstra to navigate the joint transition matrix of the Dual-HMM

5. Experiments. We present results using a Gazebo simulated home environment, a typical indoor setting for service robots [27]; even though a map is provided with the simulated environments, we do not use it in our SLAM tests, except for Figure clarity.

5.1. Benchmark. We use a benchmarking method proposed in [32] to compare our models. Dieter et al. propose "a metric for measuring the performance of a SLAM algorithm not by comparing the map itself but by considering the poses of the robot during data acquisition". As mentioned, a map might be useful for human interaction, and feature comparison, but there are many other representations that can work as a "map" and, although our method could be used to build an occupancy grid with gmapping, the map itself is not required for our SLAM method, and is only presented for clarity.

The proposed metric also allows to compare methods that use a map with others that do not since it only relies on estimates of the trajectory of the robot given by a set of poses where the observations are taken.

5.2. Training. First, the robot roams the environment reactively and, at the same time, registers wheel odometry and laser measurements at every time step. In order to make wheel odometry reliable, episodes are run among known fixed locations like doors or furniture, or in short periods of time – this serves as ground truth comparisons for the localization task and odometry correction metric.

Once enough training data is obtained (around 15k samples in our experiments), two HMM's are calculated, both HMM's share the same transition matrix and, as a consequence, the same topological representation; each HMM, however, uses a different code-book to represent the observations (K-Means and affinity propagation, respectively). Each code-book favors different world areas, so dual representation or sensor fusion can be applied; however, we opted for a dual HMM. The dual approach ensures that only correct estimates are used for correction; as a consequence, wheel odometry is corrected only when both HMM's yield the same estimation given the observations.

The pose can come either from regular uncorrected wheel odometry or, ideally, corrected odometry. In the simulations, Gaussian noise with the parameters obtained from the real robot was added to the ideal training set as a data augmentation pre-processing technique.

Additionally, to test the results of the different models we later perform around 100 sample runs in the arena with no initial information and random routes.

In Figure 9, an example run is shown. Pink points represent older real poses (still unreliable since no initial conditions are used for estimation) and yellow points are poses with a big enough buffer to be reliable. The Figure shows an example of a correct estimation, this problem is commonly known as Kidnapped Robot [30], meaning no initial conditions information is used for the estimates; however, once a correct estimation is made initial, state probabilities are available for further estimates, greatly improving accuracy. Green dots represent the poses centroids, i.e. the corpus of the HMM hidden variable X, and blue and red dots represent real and estimated hidden states, respectively. Wheel odometry correction is not applied at this stage, and accuracy is tested with the quantized states, not the real odometry, i.e. correct estimations on the HMM's with no initial conditions $\vec{\pi}$.

5.3. DUAL-HMM. The simple HMM using K-means Lidar alphabet was enough for the WRS tasks but, in order to test various data fusion schemes, a dual HMM is trained with the same training set – dual HMM uses both Lidar alphabets where the K-means and the Aff. Prop. symbols create each an emission matrix for the same transition matrix. Finally, a fused data model is proposed where the alphabet code-book is the fusion of 2D Lidar readings with the normalized features obtained from the Resnet CNN. Even though some information can be obtained this way, it is highly correlated with the random starting point because, by nature, there are regions with better features than others and, even using initial information, no significant improvement to the model is found.

In our experimental setup, the considerable computer expenses related to CNN's do not make it worth it to solve the robot relocalization problem, since accuracy does not improve significantly. This fusion may favour a different application, robot configuration or environment, and, since we show data fusion using different code-books, it is included in this work as a possible configuration for visual SLAM.

5.4. Modularity and Online Improvement. To show modularity, we use a multiple room environment. New graphs (or states) can be added after an initial HMM was found, or even a completely new HMM for each room; there is no need to retrain the whole model, as our previous research suggests [18]. Figure 10 and Figure 11 show the centroids aligning closer once enough online training had taken place. Online training yields a more structured and organized topological representation of the explored environment (shown in Figure 11), i.e. the topological representation keeps improving as the robot successfully navigates it thanks to on-line Baum Welch algorithm. Figure 10 and Figure 11 show the topological graph representation before and after online improvement.

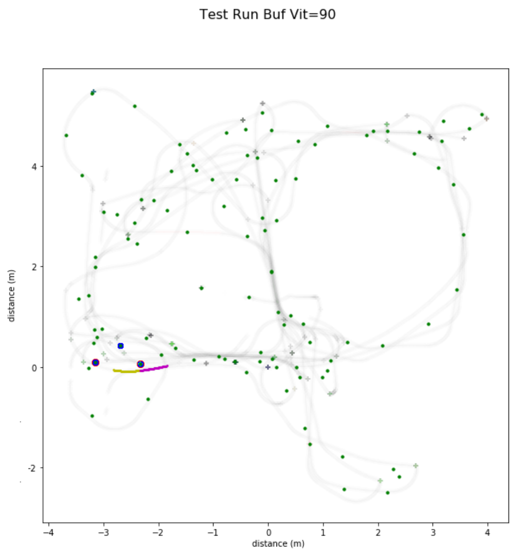

Fig. 9. An example run with a correct estimate and no initial conditions information traversing three states. State corpus in green, estimates on red and blue, beginning of the test run in pink

5.5. Pose Correction. A different virtual environment was used to validate wheel odometry correction. The reality gap is broken since the training data obtained from the model was used to navigate the real world arena, both used in the World Robot Summit 2018 and 2020 (WRS) competitions. Figure 12 and Figure 13 show the arena and the topological graph found.

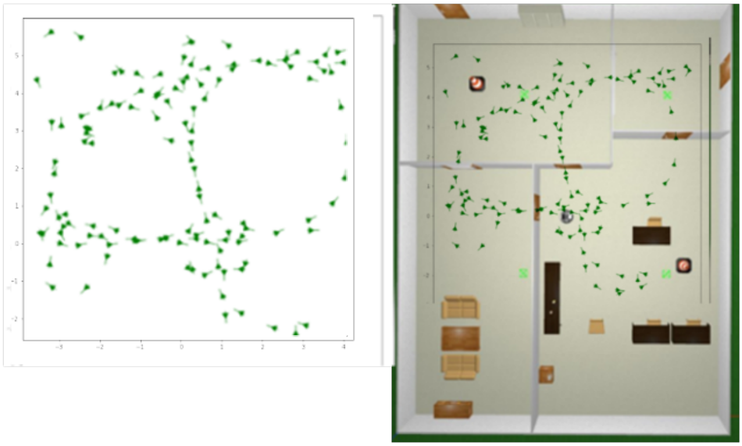
Fig. 10. Initial topological representation of the environment as obtained from off-line training

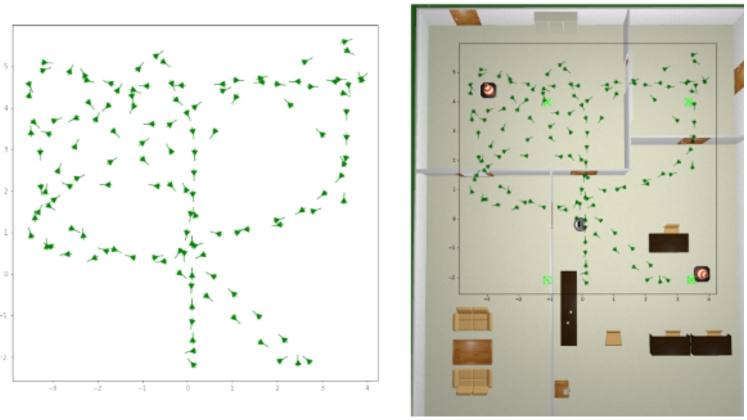
Fig. 11. A more structured and organized topological representation of the explored environment, thanks to on-line training

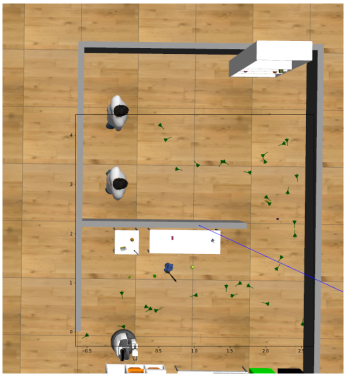

Fig. 12. World Robot Summit arena with the proposed topological graph representation

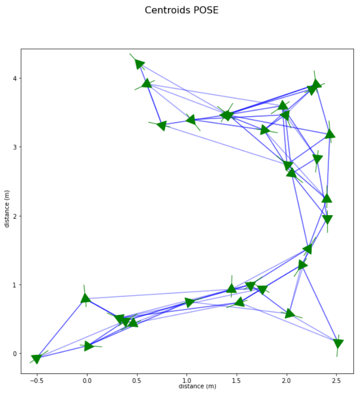

Fig. 13. WRS arena HMM representation Topological Map graph

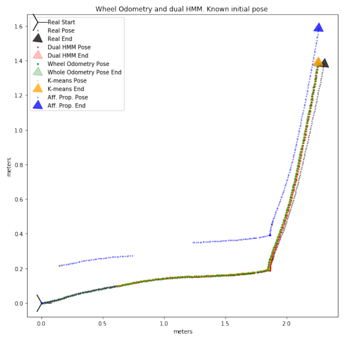

Fig. 14. Comparison with known initial location of Wheel Odometry estimate with known initial conditions in green , Dual-HMM pose estimate in red. Each of the dual single components observations K means in orange and Affinity Propagation in blue. Real trajectory shown in black

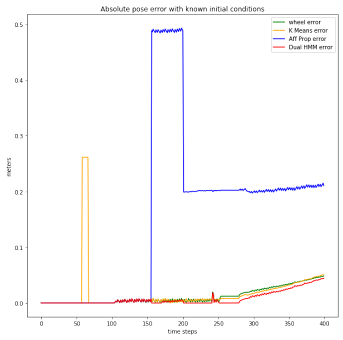

Fig. 15. Absolute pose error of Figure 14

Again, the robot was trained by acquiring the training set on a potential fields autonomous exploration run. Since the dimensions of the arena made wheel odometry reliable enough, the whole arena was modeled in the same HMM transition matrix (Figure 13).

The test runs shown do not use initial state information (kidnapped robot); however, this initial information is available after a good estimate, and could be used for maintaining the agent’s pose estimate.

Such a case is shown in Figure 14 where a comparison between the real trajectory and the components (K-means in orange and Affinity Propagation in blue) of dual HMM, as well as dual HMM in red; the real trajectory is displayed in black.

In case a good enough current initial position is known, wheel odometry is a very good way to estimate pose; once error starts growing, it can be seen that Dual HMM keeps pose estimates better aligned on the long run. It is important to note that HMM has a small quantization error on the first estimation because there is no initial information, and as such no initial transition, just the centroid of the hidden state is available in the first kidnapped robot estimate. A much longer run is shown in Figure 16 to further illustrate this idea. Figure 15 and Figure 17 report the absolute error in pose estimate over time.

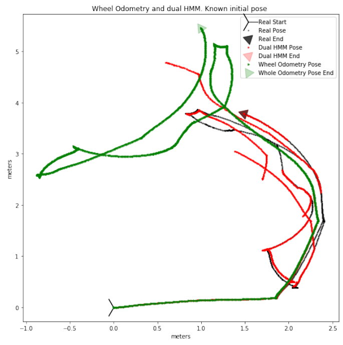

Fig. 16. Same initial conditions than Figure 14 but a much longer run. Wheel odometry in green clearly diverges while Dual-HMM in red remains operational trough the whole run

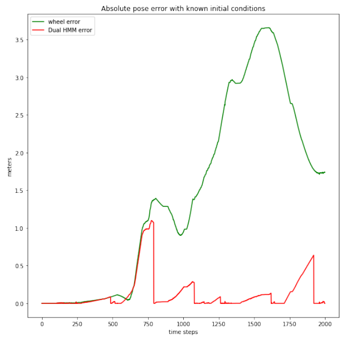

Fig. 17. Absolute pose error of a long run

5.5.1. Kidnapped robot. The wheel odometry has no initial information, and no way to re-estimate its real position, so the error keeps growing as time goes on, unlike the dual HMM odometry; as the agent continues to move it keeps traversing reliable states and correcting its pose. Results shown in Figure 18 and Figure 19 suggest that the kidnapped robot problem was solved once enough time to traverse a trusted transition had elapsed; however, if wheel odometry pose has initial conditions, it is a better estimate for the first few time steps.

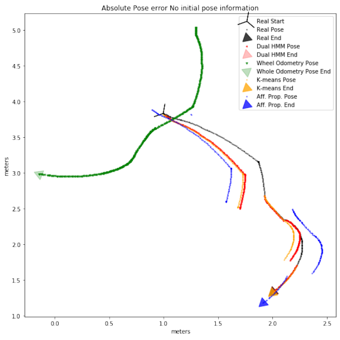

Fig. 18. In case wheel odometry has no initial information

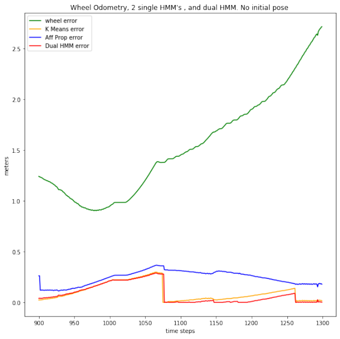

Fig. 19. Absolute pose error of run with unknown initial conditions

5.5.2. Catastrophic Event. An example run before such an event is shown in Figure 20. It can be seen in Figure 21 that the wheel odometry error is smaller than the quantization error. Then, in Figure 22, the beginning of a catastrophic event that greatly increases wheel odometry error is shown. Such error is can be seen in Figure 23; however, dual HMM SLAM can re-estimate its pose after traversing a reliable transition, since the system resets its odometry to a known transition value, remaining operational on the long run as shown in Figure 24 and Figure 25.

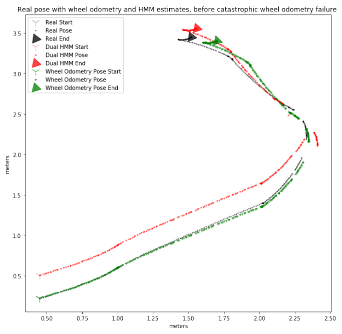

Fig. 20. Wheel Odometry and Dual-HMM error. Event start. In black the real pose sequence, green is wheel odometry with initial conditions, in red the Dual-HMM, which has a quantization error at the beginning of the run before a catastrophic event

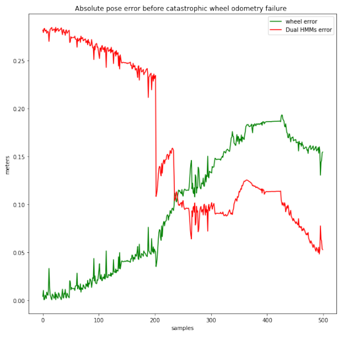

Fig. 21. Absolute pose error of wheel odometry and Dual-HMM. Both errors are relatively small, it is interesting to see how after a few time steps Dual HMM error is similar

6. Conclusions and Future Work. We proposed a SLAM method based on Hidden Markov Models which generated a modular graph representation of the environment; once the model was calculated, the kidnapped robot problem was solved in a number of environments using 2D sensor readings as observations.

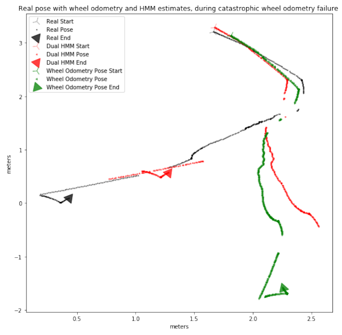

Fig. 22. Wheel Odometry in green and Dual-HMM in red during Event

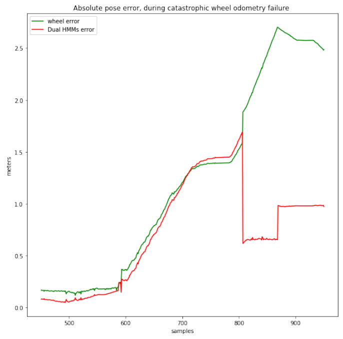

Fig. 23. Wheel Odometry and Dual-HMM error during Event

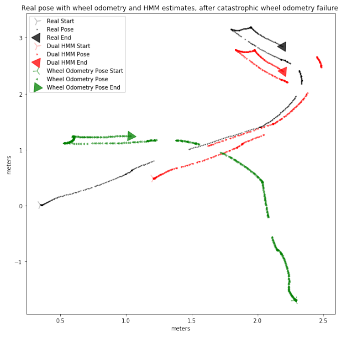

Fig. 24. Wheel Odometry and Dual-HMM estimates after Event

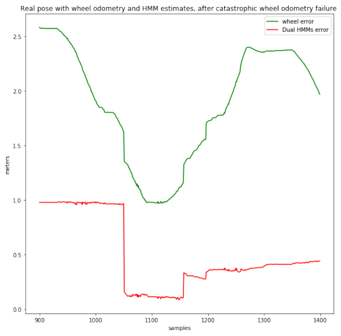

Fig. 25. Wheel Odometry and Dual-HMM error after Event

Also, although it was not shown here, the localization system remained operational during different home service tasks that required robot navigation in crowded environments. The experiments suggest that SLAM was performed successfully; it is important to notice that the online implementation of the algorithms made it possible for real-time optimization of the graph representation.

The same measurements were quantized combining different symbols from the same source using two clustering methods.

The same principle could also be applied to a wide variety of sensors or different features extracted from the same sensor measurements. This makes our system not only robust but also versatile, as it can be easily adapted to use a wide variety of sensors and action policies.

Future research is being conducted with aims in an active SLAM implementation using sensor fusion of laser, 2D image features, CNN extracted features, and 3D map features (octomaps [31]). At the same time, the optimal action policy to navigate the environment will be found using Q-learning or other appropriate reinforcement learning techniques.

References

1. Takashi Yamamoto, Tamaki Nishino, Hideki Kajima, Mitsunori Ohta, and Koichi Ikeda. Human {Support} {Robot} ({HSR}). In *{ACM} {SIGGRAPH} 2018 {Emerging} {Technologies}*, {SIGGRAPH} '18, pages 11:1—-11:2, New York, NY, USA, 2018. ACM.
2. P. Zarchan and H. Musoff. *Fundamentals of Kalman Filtering: A Practical Approach*. Number v. 190 in Fundamentals of Kalman filtering: a practical approach. American Institute of Aeronautics and Astronautics, Incorporated.
3. R. Smith, M. Self, and P. Cheeseman. Autonomous Robot Vehicles. chapter Estimating, pages 167–193. Springer-Verlag, Berlin, Heidelberg, 1990.
4. Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*,

23(1):34–46, 2007.

5. D. Fox, W. Burgard, F. Dellaert, S. Thrun AAAI/IAAI, and undefined 1999.
6. G. Grisetti, R. Kuemmerle, C. Stachniss, and W. Burgard. A Tutorial on Graph-Based {SLAM}. *Intelligent Transportation Systems Magazine, IEEE*, 2(4):31–43, 2010.
7. M. Zaffar, S. Ehsan, R. Stolkin, and K.M. Maier. Sensors, slam and long-term autonomy: A review. In *2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pages 285–290, 2018.

Monte carlo localization: Efficient position estimation for mobile robots. *aaai.org*, 1999.

8. Sebastian Thrun and Michael Montemerlo. The graph SLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures. *The International Journal of Robotics Research*, 25(5-6):403–429, 2006.
9. A. Soragna, M. Baldini, D. Joho, R. Kümmerle, and G. Grisetti. Active slam using connectivity graphs as priors. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 340–346, 2019.
10. João Machado Santos, David Portugal, and Rui P. Rocha. An evaluation of 2d slam techniques available in robot operating system. In *SSRR*, pages 1–6. IEEE, 2013.
11. Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.
12. Regis Vincent, Benson Limketkai, and Michael Eriksen. Comparison of indoor robot localization techniques in the absence of GPS. In Russell S Harmon, John H Holloway Jr., and J Thomas Broach, editors, *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XV*, volume 7664, pages 606–610. International Society for Optics and Photonics, SPIE, 2010.
13. Kurt Konolige, Giorgio Grisetti, Rainer Kümmerle, Wolfram Burgard, Benson Limketkai, and Regis Vincent. Efficient sparse pose adjustment for 2d mapping. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 22–29, 2010.
14. G.O.S Ekhaguere. On notions of Markov property. *Journal of Mathematical Physics*, 18(11):2104–2107, 1977.
15. Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *PROCEEDINGS OF THE IEEE*, pages 257–286, 1989.
16. D. Fox, W. Burgard, F. Dellaert, S. Thrun AAAI/IAAI, and undefined 1999. Monte carlo localization: Efficient position estimation for mobile robots. *aaai.org*, 1999.
17. Randall C. Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4), 1986.
18. Savage J., Fuentes O. Map representation using hidden markov models for mobile robot localization. volume 161, 2018.
19. A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.
20. Pierre Baldi and Yves Chauvin. Smooth {On}-{Line} {Learning} {Algorithms} for {Hidden} {Markov} {Models}. *Neural Comput.*, 6, 1993.
21. Long Wen, Yang Zhao, Shuguang Li, Hong Cheng, and Chen Zhang. {MST}-{ResNet}: {A} {Multiscale} {Spatial} {Temporal} {ResNet} for {Steering} {Prediction}. pages 246–251, 2019.
22. Jianbo Shi and Carlo Tomasi. Good Features to Track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.
23. Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315:972–977, 2007.

24. Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
25. G.D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.
26. Agostino Martinelli. Modeling and {Estimating} the {Odometry} {Error} of a {Mobile} {Robot}. *IFAC Proceedings Volumes*, 34(6):407–412, 2001.
27. Yasuyoshi Yokokohji, Yoshihiro Kawai, Mizuho Shibata, Yasumichi Aiyama, Shinya Kotosaka, Wataru Uemura, Akio Noda, Hiroki Dobashi, Takeshi Sakaguchi, and Kazuhito Yokoi. World robot summit – summary of the pre-competition in 2018. *Advanced Robotics*, pages 1–24, 09 2019.
28. ROS. ROS (Robot Operating System, 2018.
29. Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2149–2154, Sendai, Japan, Sep 2004.
30. Imam Bukhori and Zool Ismail. Detection of kidnapped robot problem in monte carlo localization based on the natural displacement of the robot. *International Journal of Advanced Robotic Systems*, 14:172988141771746, 07 2017.
31. Kai Wurm, A Hornung, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems. volume 2, 01 2010.
32. Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner. On Measuring the Accuracy of SLAM Algorithms. Technical report.

Fuentes Oscar — Team leader, Bio robotics laboratory, National Autonomous University of Mexico (UNAM). Research interests: machine learning, deep learning, computer vision, artificial intelligence, SLAM, navigation. The number of publications — 3. oscarfuentes-foto@gmail.com; 3000, Sirkuito Eskolar, Av. Universidad, 04510, Mexico City, Mexico; office phone: +52(55)56223041.

Savage Jesus — Ph.D., Professor, National Autonomous University of Mexico (UNAM). Research interests: autonomous mobile robots, digital signal processing, computer architectures. The number of publications — 0. robotssavage@gmail.com; 3000, Sirkuito Eskolar, Av. Universidad, 04510, Mexico City, Mexico; office phone: +52(55)56223041.

Contreras Luis — Ph.D., Research fellow, Advanced intelligence and robotics research center, Tamagawa University. Research interests: computer science, visual information, computer vision and robotics. The number of publications — 0. luis@lab.tamagawa.ac.jp; 6, Tamagawagakuen, 194-8610, Tokyo, Japan; office phone: +81 42-739-8111.

УДК 006.72 **DOI** 10.15622/ia.2022.21.7

О. Фуэнтес, Х. Саваж, Л. Контрерас

СИСТЕМА SLAM, ОСНОВАННАЯ НА СКРЫТЫХ МАРКОВСКИХ МОДЕЛЯХ

Фуэнтес О., Саваж Х., Контрерас Л. **Система SLAM, основанная на скрытых марковских моделях.**

Аннотация. Методы одновременной локализации и картографирования (SLAM) являются решением проблемы навигации сервисных роботов. Мы представляем графовую систему SLAM, основанную на скрытых марковских моделях (HMM), где показания датчиков представлены различными символами с использованием ряда методов кластеризации; затем символы объединяются в один для повышения точности с использованием двойных HMM. Универсальность нашей системы позволяет работать с датчиками разных типов или комбинировать датчики, а также реализовать активную или пассивную графовую систему SLAM. В подходе Graph-SLAM, предложенном Karto Robotics International в Cartographer, узлы представляют положение робота, а ребра представляют ограничения между ними. Узлы обычно задаются по непрерывным узлам, за исключением случаев обнаружения замыкания цикла, когда вводятся ограничения на несмежные узлы, что корректирует весь граф. Обнаружение цикливания не является тривиальным; в реализации ROS сопоставление сканирования выполняется с использованием регулировки положения разреженности (SPA). Картограф использует карту занятости, чтобы оценить положение, в котором карта отображается с помощью Gmapping. Робот Toyota HSR (Human Support Robot) использовался для создания набора данных как в реальных, так и в смоделированных условиях соревнований. В нашем представлении SLAM есть оценка одометрии колес в соответствии с начальным положением робота, 2D-лидарное сканирование Hokuyo для наблюдений, а также контроль сигналов и оценка карты окружающего пространства. Мы протестировали нашу систему в задаче о похищенном роботе, обучили начальную модель, затем улучшили ее в онлайн режиме и, наконец, решили задачу SLAM.

Ключевые слова: локализация, SLAM, навигация робота, картографирование, скрытая марковская модель, датчик.

Фуэнтес Оскар — руководитель группы, лаборатория биоробототехники, Национальный автономный университет Мексики (НАУМ). Область научных интересов: машинное обучение, глубокое обучение, компьютерное зрение, искусственный интеллект, SLAM, навигация. Число научных публикаций — 3. oscarfuentesfoto@gmail.com; Сиркуито Эсколар, проспект Университедад, 3000, 04510, Мехико, Мексика; р.т.: +52(55)56223041.

Саваж Хесус — Ph.D., профессор, Национальный автономный университет Мексики (НАУМ). Область научных интересов: автономные мобильные роботы, цифровая обработка сигналов, компьютерная архитектура. Число научных публикаций — 0. robotssavage@gmail.com; Сиркуито Эсколар, проспект Университедад, 3000, 04510, Мехико, Мексика; р.т.: +52(55)56223041.

Контрерас Луи — Ph.D., научный сотрудник, исследовательский центр передового интеллекта и робототехники, Университет Тамагава. Область научных интересов: информатика, визуальная информация, компьютерное зрение и робототехника. Число научных публикаций — 0. luis@lab.tamagawa.ac.jp; Тамагава Гакуэн, 6, 194-8610, Токио, Япония; р.т.: +81 42-739-8111.

Литература

1. Такаши Ямамото, Тамаки Нишино, Хидеки Кадзима, Мицунори Охта и Коичи Икеда.
2. П. Зарчан и Х. Мусофф. *Основы фильтрации Калмана: практический подход*. Number v. 190 в Основах калмановской фильтрации: практический подход. Американский институт аэронавтики и астронавтики, Incorporated.
3. Р. Смит, М. Селф и П. Чизмен. Автономные автомобили-роботы. глава Оценка, страницы 167–193. Шпрингер-Верлаг, Берлин, Гейдельберг, 1990.
4. Джорджио Гризетти, Сирилл Стахнисс и Вольфрам Бургард. Улучшенные методы построения сеток с помощью частиц Рао-Блэквелла Фильтры. *IEEE Transactions on Robotics*, 23 (1): 34–46, 2007.
5. D. Fox, W. Burgard, F. Dellaert, S. Thrun AAAI / IAAI и undefined 1999.
6. G. Grisetti, R. Kuemmerle, C. Stachniss и W. Burgard. Учебник по графическому SLAM . *Intelligent Transportation Systems Magazine, IEEE*, 2 (4): 31–43, 2010 г.
7. M. Заффар, С. Эхсан, Р. Столкин и К.М. Майер. Датчики, шум и длительная автономность: обзор. In *Конференция NASA / ESA 2018 по адаптивному оборудованию и системам (AHS)*, страницы 285-290, 2018. Локализация Монте-Карло: эффективное определение местоположения для мобильных устройств роботы. *aaai.org*, 1999.
8. Себастьян Трун и Майкл Монтемерло. Алгоритм graph SLAM с приложениями к крупномасштабному отображению Городские сооружения. *Международный журнал исследований робототехники*, 25 (5-6): 403-429, 2006.
9. А. Сорагна, М. Балдини, Д. Йохо, Р. Кюммерле и Г. Гризетти. Активный слэм с использованием графов связности в качестве априорных значений. In *Международная конференция IEEE / RSJ по интеллектуальным роботам, 2019 г. and Systems (IROS)*, страницы 340–346, 2019.
10. Жуан Мачадо Сантос, Д. Португалия и Руи П. Роча. Оценка методов двумерного удара, доступных в работе робота. В *SSRR*, страницы 1–6. IEEE, 2013.
11. Джорджио Гризетти, Сирилл Стахнисс и Вольфрам Бургард. Улучшенные методы построения сеток с помощью частиц Рао-Блэквелла Фильтры. *IEEE Transactions on Robotics*, 23 (1): 34–46, 2007.
12. Регис Винсент, Бенсон Лимкеткай и Майкл Эриксен. Сравнение методов локализации комнатных роботов при отсутствии GPS. In Russell S Harmon, John H Holloway Jr. и J Thomas Broach, редакторы, *Обнаружение и обнаружение мин, взрывоопасных предметов и скрытых объектов Targets XV*, том 7664, страницы 606-610. Международное общество оптики и фотоника, SPIE, 2010.
13. Курт Конолиге, Джорджио Гризетти, Райнер Кюммерле, Вольфрам Бургард, Бенсон Лимкеткай и Регис Винсент. Эффективная корректировка разреженной позы для 2d маппинга. На *Международной конференции IEEE / RSJ 2010 по интеллектуальным роботам and Systems*, страницы 22–29, 2010 г.
14. G.O.S. Ekhaguere. О понятиях марковского свойства. *Journal of Mathematical Physics*, 18 (11): 2104–2107, 1977.
15. Лоуренс Р. Рабинер. Учебное пособие по скрытым марковским моделям и избранным приложениям в распознавание речи. В *PROCEEDIES OF THE IEEE*, страницы 257–286, 1989.
16. D. Fox, W. Burgard, F. Dellaert, S. Thrun AAAI / IAAI и undefined 1999. Локализация Монте-Карло: эффективное определение местоположения для мобильных устройств роботы. *aaai.org*, 1999.

17. Рэндалл С. Смит и Питер Чизмен. О представлении и оценке пространственной неопределенности. *Международный журнал исследований робототехники*, 5 (4), 1986.
18. Сэвидж, Фуэнтес Представление карты с использованием скрытых марковских моделей для локализации мобильного робота. , том 161, 2018.
19. А.П. Демпстер, Н.М. Лэрд и Д.Б. Рубин. Максимальная вероятность получения неполных данных с помощью алгоритма EM. *ЖУРНАЛ КОРОЛЕВСКОГО СТАТИСТИЧЕСКОГО ОБЩЕСТВА, СЕРИЯ B*, 39 (1): 1–38, 1977.
20. Пьер Бальди и Ив Шовен. Smooth On-Line Learning Algorithms для Скрытых Марковских моделей .
Neural Comput., 6, 1993.
21. Лун Вэнь, Ян Чжао, Шугуан Ли, Хун Чэн и Чэнь Чжан. MST-ResNet : A Мультимасштаб Spatial Temporal ResNet для рулевого управления
22. Джианбо Ши и Карло Томази. Хорошие возможности для отслеживания. em Конференция IEEE по компьютерному зрению и распознаванию образов, страницы 593-600, 1994.
23. Брендан Дж. Фрей и Делберт Дук. Кластеризация путем передачи сообщений между точками данных. em Science, 315: 972–977, 2007.
24. Мин Лян, Бинь Ян, Юн Чен, Жуй Ху и Ракель Уртасун. Многозадачная мультисенсорная комбинация для обнаружения трехмерных объектов. В em Proceedings of the IEEE / CVF Conference on Computer Vision and Распознавание образов (CVPR), июнь 2019 г.
25. Б-г Форни. Алгоритм Витерби. em Proceedings of the IEEE, 61 (3): 268–278, 1973.
26. Агостино Мартинелли. Моделирование и оценка ошибки одометрии мобильного робота }. em IFAC Proceedings Volumes, 34 (6): 407–412, 2001.
27. Ясуёси Ёкокоджи, Ёсихиро Каваи, Мидзухо Сибата, Ясумичи Айяма, Шинья Котосака, Ватару Уэмура, Акио Нода, Хироки Добаши, Такеши Сакагути и Кадзухито Ёкои. World robot Summit - итоги предсоревнований 2018. em Advanced Robotics, страницы 1–24, 09, 2019.
28. РОС. ROS (операционная система роботов, 2018.
29. Натан Кениг и Эндрю Ховард. Разработка и использование парадигм для беседки, мульти-робота с открытым исходным кодом симулятор. In em Международная конференция IEEE / RSJ по интеллектуальным роботам и Systems, страницы 2149-2154, Сендай, Япония, сентябрь 2004 г.
30. Имам Бухори и Зоол Исмаил. Проблема обнаружения похищенного робота в локализации монте-карло на основе естественного перемещения робота. em Международный журнал передовых робототехнических систем, 14: 172988141771746, 07 2017.
31. Кай Вурм, А. Хорнунг, Марен Бенневиц, Сирилл Стахнисс и Вольфрам Бургард. Octomap: вероятностное, гибкое и компактное представление трехмерной карты для робототехнических систем. volume 2, 01 2010.
32. Бастиан Стедер, Кристиан Дорнхеге, Майкл Рунке, Джорджио Гризетти, Сирилл Стахнисс и Александр Клейнер. Об измерении точности алгоритмов SLAM. Технический отчет.