# Multi-Hypothesis SLAM for Non-Static Environments with Reoccurring Landmarks

Kristin Nielsen and Gustaf Hendeby


Nielsen, K., Hendeby, G., (2022), Multi-Hypothesis SLAM for Non-Static Environments with Reoccurring Landmarks, *IEEE Transactions on Intelligent Vehicles*, 1-13.
https://doi.org/10.1109/tiv.2022.3214978

# Multi-Hypothesis SLAM for Non-Static Environments with Reoccurring Landmarks

Kristin Nielsen*† and Gustaf Hendeby†
*Automation; Epiroc Rock Drills AB; Örebro, Sweden
kristin.nielsen@epiroc.com
†Dept. Electrical Engineering; Linköping University; Linköping, Sweden
gustaf.hendeby@liu.se

***Abstract*—A static world assumption is often used when considering the *simultaneous localization and mapping* (SLAM) problem. In reality, especially when long-term autonomy is the objective, this is not valid. This paper studies a scenario where uniquely identifiable landmarks can attend multiple discrete positions, not known *a priori*. Based on a feature based multi-hypothesis map representation, a multi-hypothesis SLAM algorithm is developed inspired by target tracking theory. The creation of such a map is merged into the SLAM framework allowing any available SLAM method to solve the underlying mapping and localization problem for each hypothesis. A recursively updated hypothesis score allows for hypothesis rejection and prevents exponential growth in the number of hypotheses. The developed method is evaluated in an underground mine application, where physical barriers can be moved in between multiple distinct positions. Simulations are conducted in this environment showing the benefits of the multi-hypothesis approach compared to executing a standard SLAM algorithm. Practical considerations as well as suitable approximations are elaborated upon and experiments on real data further validates the simulated results and show that the multi-hypothesis approach has similar performance in reality as in simulation.**

***Index Terms*—SLAM, multi-hypothesis, non-static environment**

## I. Introduction

Real-time, accurate and robust localization is fundamental for safe navigation of autonomous vehicles. When a map of the environment is not available, or the accuracy of the given map is not sufficient, the problem can be formulated within the framework of *simultaneous localization and mapping* (SLAM). Theoretically and conceptually SLAM can be considered a solved problem, but issues remain when applied to long-term, real-life problems [1]. Many of the wide-spread SLAM methods such as FastSLAM 2.0 [2] or the graph based solvers, iSAM2 [3] and g$^2$o [4], assume a rigid and static environment. The real world is both dynamic and nonrigid due to movements and deformability of objects. The problem of building life-long maps that can take changes into account is often mentioned as an important remaining challenge in this field [1, 5], and recently various aspect of this issue have been addressed within the automated vehicle community [6–12].

An environment subject to change can be categorized into two main categories,

- a *dynamic* environment, with objects moving relatively fast so that the perception of the object changes while the object is in field-of-view (FOV), or,
- a *non-static* environment, where changes in the environment are much slower and typically not captured by sensors in real time, but rather discovered when an area is revisited.

This paper focuses on a non-static environment where changes can be represented by a sequence of discrete modes defining the current state of the map. Specifically, re-occuring changes are targeted, where a change is likely to later be reverted. Such situation typically occurs in indoor environments where, for example, doors can be opened or closed [13], or, in public parking spaces where parking lots can be occupied or not [14]. Another application is autonomous vehicles in an underground mine. For specific mining methods a typical operation area for an underground loader consists of a long tunnel with cross-sections leading to drawpoints where material is loaded and transported to a crusher. To manage the ore body and to prevent mistakes, tunnels leading to currently inactive drawpoints can be closed with physical barriers that later are reopened when the drawpoint is activated again. Due to the special nature of an underground mine, with the area being in total control of the mine operators, it is possible to make such moving barriers easily identifiable by the vehicle's perception system.

In [15] we presented a feature based multi-hypothesis map representation to be used for localization in non-static environments where uniquely identifiable landmarks can have multiple discrete modes. This paper extends the work by considering the creation and updating of such maps. The major contributions of this paper are:

- The theoretical incorporation of the feature based multi-hypothesis map representation into the SLAM framework, still allowing the underlying static SLAM problem to be solved by any state-of-the-art SLAM solver of choice.
- A tree structure for efficient hypothesis management and decision making is introduced. Practical considerations and possible approximations are discussed and elaborated.
- A formalization of a multi-hypothesis SLAM algorithm reinforced with ingredients from target tracking theory,

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

allowing missed detections and false alarm observations to be modeled.
- The proposed algorithm is demonstrated in experiments using both simulated and real data from an underground mine.

The structure of this paper is as follows: Section II presents related work in the literature. Section III presents the multi-hypothesis map representation and incorporates it into the SLAM theory. Section IV defines a hypothesis score used for decision making, and Section V suggests how the hypothesis creation and decision making can be formalized in a tree structure. Section VI highlights practical considerations and Section VII suggests possible approximations. In Section VIII the multi-hypothesis SLAM algorithm is presented followed by experiments using both simulated, and real data, from an underground mine in Section IX. Finally, concluding remarks are given in Section X.

## II. Related Work

A dynamic environment is often handled by trying to filter out moving objects in the input data [16, 17], or more sophisticated solutions where the motion of the objects are tracked and classified [18, 19]. Work more focused on non-static environments often introduces some sort of a memory decay [7]. In [20] a persistence filter is presented, designed to run in parallel with a SLAM solver. Each landmark in the map has an associated probability of existence that decays over time when the landmark is not observed. In [8] each feature is scored as a function of times it has been observed. Low score features can then be disregarded not considered robust for localization. Others have incorporated multiple hypotheses more directly into the factor graph representation in a graph-based SLAM system. In [21–24] such systems are designed primarily with the goal of identifying and removing outliers or false loop-closures from the factor graph.

A switching model solution for the state transition is presented in [25] where the discrete modes are modeled as a Markov chain. Hypotheses are represented in a tree structure that can efficiently be solved by graph-based SLAM solvers. However, it is not explored how this method would perform with ambiguities in the observation model. A similar solution is the multi-hypothesis algorithm based on iSAM2 (MH-iSAM2) developed in [26], which is more general in the types of ambiguities that can be modeled but lacks the possibility to model the modes as a Markov chain.

All of these methods completely deletes a rejected hypothesis and the information is permanently forgotten or never added to the persistent map. This paper proposes a solution where rejected hypotheses are saved to later be re-activated if the environment is changed back to its previous state. The solution suggested in this paper is similar in spirit to the work done in [27], where map hypotheses connected to different time scales are kept in memory. However, in [27] a sample based map representation is used which does not provide a sensor agnostic solution, also the updating of the map hypotheses are complicated beyond the SLAM framework. Our feature based solution can be used with any type of sensor where an acceptable feature extraction method is available, and the hypotheses evaluation is performed in combination with any SLAM solver responsible of updating each hypothesis.

A combination of SLAM and target tracking theory has been proposed earlier, in the multi-hypothesis FastSLAM presented in [28] where a *multi hypothesis tracker* (MHT) is used to define the particles in a FastSLAM implementation, or in *random finite set* (RFS) versions of SLAM [29–33]. Such multi-hypothesis approaches theoretically result in an exponential growth in the number of hypotheses. In this paper hypotheses are evaluated with a recursively updated score connected to specific landmark modes, similar to the work done in [34]. This allows for a decision making where unlikely hypotheses can be de-activated at certain points in times, and an up-to-date version of the map is maintained.

## III. Non-Static Environment

Throughout this paper it is assumed that true landmarks have a unique signature, identifiable by the robot's perception system. However, spurious observations not originating from any true landmark, but erroneously given a valid signature, may appear. It is also assumed that the environment is non-static, *i.e.*, landmarks are moved between discrete positions only when not in the robot's FOV. Under these assumptions, the static formulation of the SLAM problem fails to give good estimates. A discrete movement of a landmark can cause jumps in the state estimation and inconsistencies in the map.

### A. Multi-Hypothesis Map

This section contains a short description of the feature based multi-hypothesis map representation presented in [15].

Landmarks in the map identified by the same signature are grouped together and an observation is by the unique identifier associated to a group in the map. Probabilities of existence of the landmarks in a group can be estimated according to incoming observations and a valid up-to-date version of the map is provided by selecting the landmark with highest probability in each group as currently active.

The multi-hypothesis map is defined by

$$\mathcal{M} = \left[ m_1^{s_1}, \ldots, m_{n_{s_1}}^{s_1}, \ldots, m_1^{s_k}, \ldots, m_{n_{s_k}}^{s_k} \right]^\top, \qquad (1)$$

where $s_j$ implies that the landmark belongs to group $j$, with a total number of $k$ groups. A version of a specific landmark will be referred to as a mode and the number of possible modes of the landmark with signature $s_j$ is thus given by $n_{s_j}$. Let $\delta^j = q$, $q = 1, \ldots, n_{s_j}$ define a mode indicator for the landmarks in the group $s_j$, stating that landmark $m_q^{s_j}$ is active, implying that the data association is done with this particular landmark. A mode indicator vector can then be constructed for each time instance $t$,

$$\delta_t = \begin{bmatrix} \delta_t^1 & \ldots & \delta_t^k \end{bmatrix}^\top, \qquad (2)$$

indicating which mode is active in each landmark group at a particular point in time. This yields a sequence of mode vectors in time,

$$\delta_{1:T} = \{\delta_0, \ldots, \delta_T\}. \qquad (3)$$

The non-static map can now be represented by a static part $\mathcal{M}$ and a sequence of modes $\delta_{1:T}$, stating which landmarks contained in $\mathcal{M}$ are active at each time instance.

The mode sequence is modeled by a discrete valued Markov chain with a time dependent transition matrix. If the modes are independent across different groups the Markov chain factorizes by

$$\Pr(\delta_t|\delta_{t-1}) = \prod_{j=1}^{k} \Pr(\delta_t^j|\delta_{t-1}^j). \tag{4}$$

For group $j$ the elements in the transition matrix $\Pi_{t-1}^j$ is defined by,

$$\Pr(\delta_t^j = k|\delta_{t-1}^j = q) = \left[\Pi_{t-1}^j\right]_{qk}, \tag{5}$$

where $[\cdot]_{qk}$ is the element in the $q$th row and $k$th column.

### B. Multi-Hypothesis SLAM

With the multi-hypothesis map representation, an observation model can be expressed as,

$$p(z_t|x_t, \mathcal{M}, \delta_t), \tag{6}$$

where $z_t$ are observations at time $t$ and $x_t$ are the state vector of the vehicle at time $t$. The filter SLAM problem can be formulated as estimating

$$p(x_T, \mathcal{M}, \delta_T|z_{1:T}, u_{1:T}), \tag{7}$$

where $u_{1:T}$ represents exogenous input. This can be split into two factors (omitting $u_{1:T}$ to simplify notation)

$$p(x_T, \mathcal{M}, \delta_T|z_{1:T}) = p(x_T, \mathcal{M}|\delta_T, z_{1:T})p(\delta_T|z_{1:T}). \tag{8}$$

The first factor constitutes the static SLAM problem conditioned on a specific mode vector. Different techniques to get approximate solutions to the static SLAM problem are available in the literature, *e.g.*, EKF-SLAM [35], FastSLAM [2, 36], and more modern graph-based SLAM solvers such as iSAM2 [3] or g2o [4]. From all of these methods it is possible to recursively get an estimate of $x_T$ and $\mathcal{M}$, and also of the likelihood $p(z_{1:T}|x_T, \mathcal{M}, u_{1:T}, x_0)$. The second factor in (8) is the likelihood of a mode vector conditioned on the sequence of measurements.

In theory, it is possible to estimate (8) by creating one hypothesis for each possible mode sequence and associate one SLAM solver to each hypothesis. However, this quickly becomes computationally intractable since the number of mode sequences grows exponentially with time. In [15] it is shown how the number of hypotheses are reduced in the pure localization problem by utilizing the knowledge that landmarks do not move while in FOV. By deducing which mode is currently active, the number of concurrent hypotheses can often be reduced to only one. The same idea of recursively updating a hypothesis score to allow for a decision to be made is here adopted also for building the multi-hypothesis map within the SLAM framework.

## IV. Hypothesis Score

Consider a SLAM system during a time window $t \in [t_1, t_2]$ where a specific landmark is in FOV. Assuming that multiple modes for this landmark exists, the hypothesis that the landmark is located in one of the positions is formulated as

$$\mathcal{H}_k^j : \delta_t^j = k, \quad \forall\, t \in [t_1, t_2]. \tag{9}$$

Consider the log-likelihood of the hypothesis conditioned on observations and apply Bayes' rule,

$$L_t^{\mathcal{H}_k^j} = \log \Pr(\mathcal{H}_k^j|z_{t_1:t}) = \log \frac{p(z_{t_1:t}|\mathcal{H}_k^j)\Pr(\mathcal{H}_k^j)}{p(z_{t_1:t})}. \tag{10}$$

A *hypothesis score* to compare two hypotheses can then be defined as

$$L_t^{\mathcal{H}_{ki}^j} = \log \frac{\Pr(\mathcal{H}_k^j|z_{t_1:t})}{\Pr(\mathcal{H}_i^j|z_{t_1:t})} = \log \frac{p(z_{t_1:t}|\mathcal{H}_k^j)\Pr(\mathcal{H}_k^j)}{p(z_{t_1:t}|\mathcal{H}_i^j)\Pr(\mathcal{H}_i^j)}, \tag{11}$$

where the normalizing constant $p(z_{t_1:t})$ cancels. The marginal probability of a hypothesis $\Pr(\mathcal{H}_k^j)$ can be deduced from the Markov chain model. First note that,

$$\Pr(\mathcal{H}_k^j) = \Pr(\delta_{t_1}^j = k), \tag{12}$$

since the landmark cannot change mode while in FOV. In the Markov chain model this implies an identity transition matrix. For $t < t_1$ the transition matrix is generally not identity and

$$\Pr(\delta_{t_1}^j = k) = \sum_i \Pr(\delta_{t_1}^j = k|\delta_{t_1-1}^j = i)\Pr(\delta_{t_1-1}^j = i), \tag{13}$$

where $\Pr(\delta_{t_1-1}^j = i)$ is known from previous visits or initialized when a landmark is first observed.

Assuming the measurements in the sequence $z_{t_1:t}$ are mutually independent, the unnormalized log-likelihood $\hat{L}_t^{\mathcal{H}_k^j} \propto L_t^{\mathcal{H}_k^j}$ can be updated recursively as new measurements become available,

$$\hat{L}_t^{\mathcal{H}_k^j} = \hat{L}_{t-1}^{\mathcal{H}_k^j} + \log p(z_t|\mathcal{H}_k^j), \tag{14}$$

with the initial value

$$\hat{L}_{t_1}^{\mathcal{H}_k^j} = \log p(z_{t_1}|\mathcal{H}_k^j)\Pr(\delta_{t_1}^j = k). \tag{15}$$

*Hypothesis scores* can thus be computed by taking differences of unnormalized log-likelihoods,

$$L_t^{\mathcal{H}_{ki}^j} = \hat{L}_t^{\mathcal{H}_k^j} - \hat{L}_t^{\mathcal{H}_i^j}. \tag{16}$$

The recursive update of the score allows for a mode decision to be made before $t = t_2$, by applying a multi-hypothesis *sequential probability ratio test* (SPRT). All pair-wise scores, $L_t^{\mathcal{H}_{ki}^j}$ are computed and a decision is made in favor of $\mathcal{H}_k^j$ if,

$$L_t^{\mathcal{H}_{ki}^j} > \frac{1-\alpha}{\alpha} \quad \text{for all } i \neq k, \tag{17}$$

where $\alpha$ is the accepted probability of selecting the wrong hypothesis [37]. Since executing many SLAM solvers in parallel can be computationally demanding it is desirable to make decisions and remove unlikely hypotheses as quickly as possible. Therefore an extra decision criterion to disregard

unlikely hypotheses may be added where a hypothesis $\mathcal{H}_k^j$ is disregarded if

$$L_t^{\mathcal{H}_{ki}^j} < -\frac{1-\alpha}{\alpha} \quad \text{for all } i \neq k. \tag{18}$$

Note that the probability of a hypothesis can be acquired from the score directly if $n_{s_j} = 2$ or by solving a system of equations containing the scores if $n_{s_j} > 2$.

### A. Measurement Likelihood

In an ideal world, estimates of the likelihoods $p(z_t|\mathcal{H}_k^j)$ are directly obtained from the SLAM solver. However, this likelihood estimate only considers observations that are matched to landmarks in the map, and when different hypotheses are evaluated also missed detections and false alarms should be considered. In the state estimator based on the multi-hypothesis map representation, presented in [15], the likelihood is computed from equations inspired by target tracking applications. In principal the same computations can be used for the SLAM application.

Assume that each landmark $m_k^{s_j}$ in the map has a probability of detection given by,

$$P_{\mathrm{D}}^{jk}(x_t) = P_{\mathrm{D0}} P_{\mathrm{FOV}}^{jk}(x_t), \tag{19}$$

where $P_{\mathrm{D0}}$ is a sensor specific part, modeled by sensor type specific models or as a constant [38]. The uncertainty from the landmark extraction method of choice is included in $P_{\mathrm{D0}}$, and in case of a mixed dynamic/non-static environment occlusions caused by dynamic objects not included in the map can also be modeled in $P_{\mathrm{D0}}$. The probability that a landmark is in FOV is given by $P_{\mathrm{FOV}}^{jk}(x_t)$, as a function of the state vector. With a filtering approach, the state vector and its uncertainty is estimated, which in combination with the uncertainty of the landmarks, $R_{\mathcal{M}}$, determines $P_{\mathrm{FOV}}^{jk}(x_t)$.

Any method that extracts landmarks from raw sensor data will due to noise and imperfections in the method itself, eventually produce false measurements. Clutter, consisting of spurious measurements not originating from any known landmark, is assumed Poisson distributed, with a false alarm rate $\beta_{\mathrm{FA}}$. Also the distribution of not yet discovered landmarks are assumed Poisson distributed with rate $\beta_{\mathrm{NT}}$.

In target tracking, the hypothesis likelihood is factorized with the assumption that different targets/landmarks are not correlated. This is not valid if the map covariance $R_{\mathcal{M}}$ contains known landmark correlations. However, a reasonable assumption is that clutter, new landmarks, and missed detections are independent from observations matched to landmarks in the map. The likelihood can then be factorized as,

$$\begin{aligned} p(z_t|\mathcal{H}_k^j) = P_{D0}^{|\mathcal{J}|} \left( \prod_{j \in \mathcal{J}} P_{\mathrm{FOV}}^{jk} \right) p(z_t^{\mathcal{J}}|\mathcal{H}_k^j) \\ \times \beta_{\mathrm{FA}}^{m_t^{\mathrm{FA}}} \beta_{\mathrm{NT}}^{m_t^{\mathrm{NT}}} \prod_{j \in \mathcal{M}_D} \left(1 - P_{D0} P_{\mathrm{FOV}}^{jk}\right), \end{aligned} \tag{20}$$

where $z_T^{\mathcal{J}}$ is the sub-set of the measurements in the scan $z_t$ that are matched to a landmark in the map, $|\mathcal{J}|$ is the cardinality of $z_T^{\mathcal{J}}$, $\mathcal{M}_D$ is the set of missed detections (*i.e.*, landmarks with $P_{\mathrm{FOV}} > 0$ that are not represented in the observations). The number of false alarms and newly discovered landmarks are denoted by $m_t^{\mathrm{FA}}$ and $m_t^{\mathrm{NT}}$, respectively.

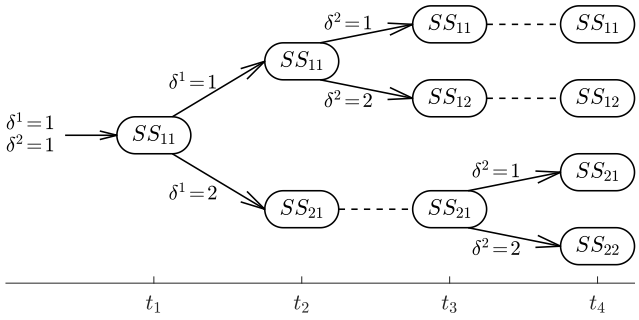


Fig. 1. The hypothesis tree structure are growing when new landmark positions are observed. Each node is associated with one hypothesis but only the leafs are connected to a complete mode vector and an associated *SLAM solver* (SS). The two indices to the SLAM solvers are representing the mode indicator value for the first and second landmark group respectively.

## V. Hypothesis Score for SLAM

For a state estimation application, the hypothesis scores and a multi-hypothesis map known *a priori* is enough to provide up-to-date knowledge of current positions of landmarks while at the same time localize the robot in the map. Each time a landmark comes into FOV a hypothesis test is initialized with associated state estimators for all possible modes of that landmark. As soon as a decision is made, all state estimators associated with disregarded hypotheses may be removed from further processing. For the SLAM application, special care has to be taken with observations not matching the current map since all possible hypotheses are not known *a priori*. Also, the hypothesis score update gets slightly more involved since the normalizing factor in the hypothesis score only cancels if the time interval is equal for the hypotheses. This section first presents a tree structure that simplifies the bookkeeping of hypotheses currently under evaluation. Then different aspects of not knowing all possible hypotheses up-front are considered.

### A. Hypothesis Tree

Hypotheses currently under evaluation can be represented by a tree structure where each node represents one hypothesis associated with a prior probability of that hypothesis and a recursively updated score. The tree can be dynamically updated as the robot is moving and nodes can be added when a new hypothesis evaluation is instantiated. Tree growth is typically triggered when an observation of a landmark is not gated with any known position, a landmark with known multiple modes enters FOV, or, a missed detection is noticed. Nodes are removed when decisions are made or hypotheses merged.

Each node in the tree is associated with a certain value of one specific mode indicator and values for the other mode indicators are given by ancestor nodes. This means that only the leaf nodes in the tree has access to a completely defined

mode indicator vector. To update a static SLAM solver according to (8), a global hypothesis, *i.e.*, a complete mode vector, is needed. Only the leaves on the tree are therefore associated with a SLAM solver. All landmark modes are always included in the map for all SLAM solvers even though they are not marked as active by the mode vector. This allows the positions and covariances of these landmarks to be implicitly updated by known correlations to the robot pose or other landmarks, even though they are never explicitly updated by an observation.

Fig. 1 shows an example of how a hypothesis tree, comprising two landmark groups, is built up over time as new hypotheses evaluations are initialized. The depth of the tree reflects the number of landmark groups that are currently evaluated, and the width reflects the number of concurrent SLAM solvers that need to be updated. Note how the third level in the tree in Fig. 1 is not divided at the same point in time for the different branches. This is perfectly valid behavior since different slam solvers might discover and initiate a hypotheses evaluations at different times.

### B. Hypotheses Branching

There are four different types of observations or measurements that can trigger branching of the hypothesis tree;

1) *negative information* in form of missed detections,
2) *new landmark* observations,
3) *false alarm* measurements, or,
4) a *landmark enters FOV*.

None of these observations will under all circumstances trigger a new branch in the tree. If appropriate nodes are already present in the tree, the observation is merely used in the, for each hypothesis, corresponding factor in (20) to update the hypothesis score, and for updating the SLAM solver. Below follows a description of when each of the four situations will trigger branching. Note that if a complex tree structure already exists, the same observation might be classified and treated differently in the different sub-trees due to discrepancies in the state and map estimates of the different SLAM solvers.

*1) Negative Information:* When the multi-hypothesis map representation is used in a pure state estimation scenario, as in [15], negative information can be utilized in the sense that if a landmark expected to be observed is not, the mode probabilities in that landmark group will shift towards other modes. This usage of negative information does not directly transfer to the SLAM application since not all modes are known *a priori*. If an expected observation is missing, there might not yet exist an alternative mode whose probability of existence could be increased. To reflect this scenario the *alternative hypothesis* is introduced. This hypothesis is created when an active mode of a landmark enters FOV but there is no observation of it. The alternative hypothesis constitutes the option that the landmark is not in any known position. The landmark could have been removed completely from the operation area or it is placed in a position where it has not yet been observed. This hypothesis can be treated as a null hypothesis and can therefore be defined as,

$$\mathcal{H}_0^j : \delta_t^j = 0. \tag{21}$$

For $\mathcal{H}_0^j$ the probability of the landmark being in FOV is always zero, $P_{\text{FOV}}^{j0}(x_t) = 0$.

*2) New Landmark:* If an observation of a landmark is made, not gating with any known mode of that landmark, a new hypothesis containing this observed position should be started. A hypothesis where $m_t^{\text{NT}} = 1$ is created and the landmark is added to the map according to the SLAM solvers internal procedure of adding new landmarks. If multiple modes of that landmark already exists, but are currently inactive, hypotheses for all inactive modes should also be started and evaluated. In that case a tree structure where a parent node has multiple children is created. If on the other hand, no landmark with the observed signature already exists in the map representation, no new hypothesis should be created. The observation is merely left to the SLAM solver to add to the map.

*3) False Alarm:* Clutter measurements are by the SLAM solver seen as observations not gated with any known mode and are impossible to distinguish from an actual new landmark mode. Hence, a new hypothesis should be created reflecting this new observation. Call this newly created hypothesis, $\mathcal{H}_B^j$, and the original hypothesis $\mathcal{H}_A^j$. The problem is that none of these hypotheses reflects the truth, that the observation is in fact a false alarm. Therefore, a third hypothesis also has to be created $\mathcal{H}_C^j$, where the map and mode indicator vector is equal to that of $\mathcal{H}_A^j$ but with this particular observation treated as a false alarm in (20) and not used in the measurement update of the SLAM solver. Note that this third hypothesis always has to be started also for the *new landmark* case since the SLAM solver cannot distinguish between the two events.

*4) Landmark Enters FOV:* This branching trigger applies when multiple modes are present in the map and any of them enters FOV. A tree node is created for each existing mode and a score evaluation is started. This is analogous to the hypothesis management for the pure state estimation problem described in [15]. For the SLAM problem though, this situation might also start an alternative hypothesis, according to the *negative information* trigger, if no observations are made of the landmark when it enters FOV.

## VI. Practical Considerations

Updating multiple SLAM solvers in parallel can for many systems be computationally demanding. There are some practical considerations that can be taken into account to limit the computations or the number of SLAM solvers that has to be executed. The branching triggers described in Section V-B are general for all Markov chains. The total number of hypotheses created can be reduced by exploiting the idea that transitions are not allowed while a landmark is in FOV. Further computations can be saved by noting that the alternative hypothesis does not need its own SLAM solver.

### A. The Alternative Hypothesis

Since the alternative hypothesis carries no different information about the state vector or the landmark positions than the original hypothesis that it is an alternative to, the SLAM solver does not have to be forked for this hypothesis and a

lot of the factors in the likelihood ratio $L_t^{\mathcal{H}^j_{k0}}$ cancels. For hypothesis with equal state estimates it is enough to compute,

$$p(z_t|\mathcal{H}^j_k) \propto \begin{cases} P_{D0}P^{jk}_{\text{FOV}}p(z_t^{\mathcal{J}}|\mathcal{H}^j_k) & \text{if } z_t^j \neq \emptyset \\ \left(1 - P_{D0}P^{jk}_{\text{FOV}}\right)p(z_t^{\mathcal{E}}|\mathcal{H}^j_k) & \text{if } z_t^j = \emptyset \end{cases}, \quad (22)$$

since all other factors cancel when the score is formed. Here, $z_t^j$ is the observation associated with the landmark that the hypothesis is formed with, and $z_t^{\mathcal{E}}$ is the subset of matched measurements such that $\left\{z_t^j, z_t^{\mathcal{E}}\right\} = z_t^{\mathcal{J}}$. The likelihood of the alternative hypothesis is given by,

$$p(z_t|\mathcal{H}^j_0) \propto \begin{cases} \beta_{\text{FA}}p(z_t^{\mathcal{E}}|\mathcal{H}^j_0) & \text{if } z_t^j \neq \emptyset \\ p(z_t^{\mathcal{E}}|\mathcal{H}^j_0) & \text{if } z_t^j = \emptyset, \end{cases} \quad (23)$$

since $P^{j0}_{\text{FOV}} = 0$. This yields a score update according to

$$L_t^{\mathcal{H}^j_{k0}} = L_{t-1}^{\mathcal{H}^j_{k0}} + \begin{cases} \log\left(\frac{P_{D0}P^{jk}_{\text{FOV}}p(z_t^{\mathcal{J}}|\mathcal{H}^j_k)}{\beta_{\text{NT}}p(z_t^{\mathcal{E}}|\mathcal{H}^j_k)}\right) & \text{if } z_t^j \neq \emptyset \\ \log\left(1 - P_{D0}P^{jk}_{\text{FOV}}\right) & \text{if } z_t^j = \emptyset, \end{cases} \quad (24)$$

where it is used that $p(z_t^{\mathcal{E}}|\mathcal{H}^j_k) = p(z_t^{\mathcal{E}}|\mathcal{H}^j_0)$.

A potential dilemma occurs if there are already two active hypotheses with different modes of landmark $s_j$, and then negative information triggers an alternative hypothesis branching. Which of the two SLAM solvers should the alternative hypothesis use? Theoretically, two different alternative hypotheses should be created. In practice, however, when this situation occurs the landmark is often in FOV and the no transition Markov property defer any mode change and no alternative hypothesis has to be created at all.

### B. Back-tracking

A practical complication with the definition of the hypothesis score in (11) is that all likelihoods that are to be compared has to be started at the same point in time for the normalizing factor to cancel. If an evaluation of a landmark is ongoing and an observation is made that is not gated with any of the already evaluated hypotheses. A third hypothesis should be added to the tree, either by backtracking the probability of that hypothesis to when the already existing evaluation was started, or by adding it as a child to the existing branches. Fig.2 shows the tree at $t = t_0$ when only two modes of a landmark exists, and the resulting tree at $t = t_1$ when the third hypothesis is created, for the two options. If the backtracking strategy is used the unnormalized log-likelihood of the third hypothesis has to be corrected by adding the log-likelihood for the time steps in between $t_0$ and $t_1$ with the assumption that the landmark was in the $\mathcal{M}_D$ set. Also the priors of the existing hypotheses, $\Pr(\delta^j_{t_0} = k)$, and $\Pr(\delta^j_{t_0} = m)$ has to be adjusted in this back-tracking procedure to have the probabilities at $t = t_0$ to sum to 1. This requires data to always be saved when an evaluation is present for potential future branching.

The alternative of adding all new hypothesis as children to existing nodes does not require old data to be saved. However, this solution might cause duplication of the same global hypothesis in the tree leaves, resulting in computational overhead of executing unnecessary SLAM solvers.

### C. Score Recursion

It would be possible to only evaluate hypothesis scores for the leaf nodes and compare them to each other according to (17) and (18), but that might delay decisions. In some situations a decision can be made at a higher level in the tree and large sub-trees can be removed at once. However, this requires the scores at the higher levels to be computed and updated through marginalization. For a non-leaf node the likelihood is determined by

$$p(z_t|\mathcal{H}^j_k) = \sum_{\mathcal{H}^i_m} p(z_t|\mathcal{H}^j_k, \mathcal{H}^i_m)\Pr(\mathcal{H}^i_m|\mathcal{H}^j_k), \quad (25)$$

where the summation is over all hypotheses that are a branch to the node. With a deep tree this of course has to be done recursively until a leaf is reached, and then backtracked to the higher levels. Note that $\Pr(\mathcal{H}^i_m|\mathcal{H}^j_k) = \Pr(\mathcal{H}^i_m)$ since the mode of each landmark is assumed independent and $\Pr(\mathcal{H}^i_m)$ can be obtained from the scores at the branched level in the tree. Note that this strategy is not applicable if $i = j$, which is the case if multi-hypothesis are added as children as discussed in section VI-B.

### D. Observation of Inactive Modes

Theoretically it should not be possible to obtain measurements from landmarks that are not in FOV. However, in practice $P^{jk}_{\text{FOV}}$ is not exactly known and the decision of when a landmark is considered to be in FOV is decided by a threshold parameter that has to be tuned. This may lead to situations where an inactive landmark is not considered to be in FOV yet, but still observations are obtained. Sometimes this could be solved by lowering the threshold, but then you risk to falsely decide for an alternative hypothesis before measurements are obtained in other situations. A practical solution is therefore to start a new hypothesis according to the *landmark enters FOV*-trigger when this situation occurs and simply treat the landmark as if in FOV.

## VII. Possible Approximations

To further limit computations and the need of executing SLAM solvers in parallel some approximations can be made.

### A. Transform Alternative Hypothesis

Consider a situation where an alternative hypothesis exists, created due to a negative information trigger, and an observation is made that is not gating with any of the current modes. Then the alternative hypothesis can be transformed into a regular hypothesis with the landmark mode positioned according to the incoming observation. This limits the overhead of having to create a new hypothesis and backtrack the score if instead the score from the alternative hypothesis is adopted. As soon as measurements are starting to come in, the alternative hypothesis is unlikely to be true and would probably be rejected soon anyway. A risk with this approximation is that the incoming observation is in fact a false alarm and the currently true position of the landmark might not be in FOV at the moment. In that case two SLAM solvers will be executed

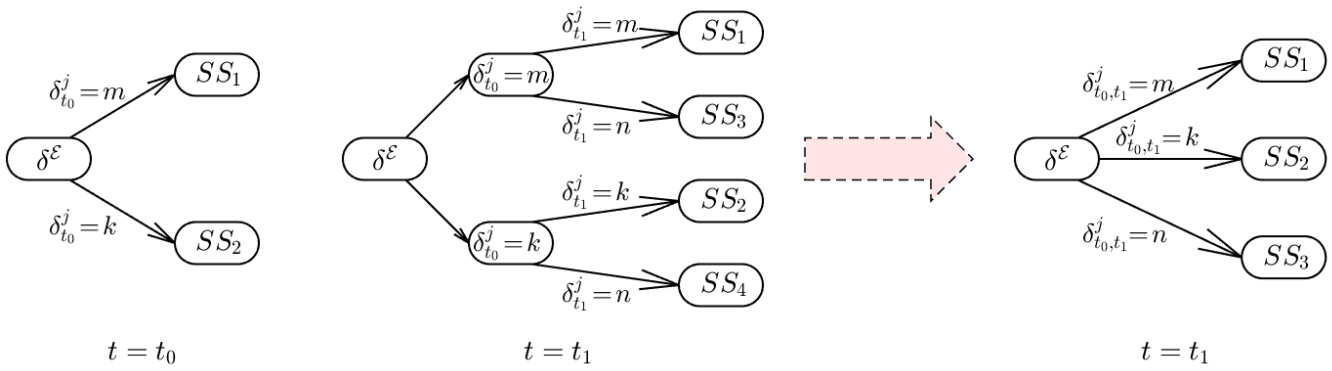


Fig. 2. The left picture show the hypothesis tree at $t = t_0$, $\delta^{\mathcal{E}}$ denotes the subset of the mode indicator vector for all landmark groups except the $j$th one. At $t = t_1$ an additional hypothesis of landmark $j$ is created that can be added to the tree directly as children (middle) or be re-arranged by back-tracking (right).

in parallel with equally decreasing score until observations are made of the true position or an alternative hypothesis is triggered again by the landmarks re-entering the FOV.

The transformation of alternative hypothesis implicates further simplifications in the situation when there are two existing modes of one landmark and only the inactive one comes into FOV while no observations are made. In this case, an evaluation should be triggered and an alternative hypothesis should be created. However, the alternative hypothesis will in this case completely follow the hypothesis where the landmark is not in FOV and as soon as an observation is made, the alternative hypothesis is transformed into that hypothesis. Therefore, it is unnecessary to create the alternative hypothesis to begin with since hypothesis would anyhow be triggered when an ungated observation is made or the active mode enters FOV.

### B. False Alarm

In case of a false alarm triggered branching, three hypotheses are evaluated; the original hypothesis $\mathcal{H}^j_A$, the hypothesis reflecting the measurement $\mathcal{H}^j_B$, and, the hypothesis $\mathcal{H}^j_C$ with state according to $\mathcal{H}^j_A$ but with this particular measurement considered as false alarm. Since data association is assumed trivial the SLAM solver associated with $\mathcal{H}^j_A$ might take a huge leap when updated with the observation not gating with the active mode in this hypothesis. Depending on other landmarks in the vicinity and correlations to them, this update might locally adapt to the observed situation causing the succeeding updates to result in relatively large likelihoods, even though the hypothesis is in fact false. Also, $\mathcal{H}^j_C$, which in this case is the true hypothesis, might get a low likelihood if the false alarm rate is low. It is therefore important to tune the thresholds so that such hypotheses are not immediately rejected, but still allow for a quick decision in the coming updates to avoid running SLAM solvers in parallel.

A solution to this is to always treat a non-gated observation as a false alarm, in all hypotheses. In the case of spurious false alarms this can be interpreted as an immediate reject of $\mathcal{H}^j_A$ and instead only proceed with $\mathcal{H}^j_C$, which for a false alarm is the correct hypothesis. However, with this approach $\mathcal{H}^j_A$ is also rejected in situations where a landmark has actually changed mode and $\mathcal{H}^j_B$ is the true one. This might affect the time it takes to make a decision. However, the false alarm rate is for many applications typically low and the score for $\mathcal{H}^j_C$ might even decrease faster than the score for $\mathcal{H}^j_A$ where the solution gradually adapts to the observations. Fig. 3 show a simple example where this approximation is beneficial for the result, since $\mathcal{H}^j_A$ is the wrong hypothesis but adapts to observations.

### C. Merge Hypotheses

In the SLAM application where landmark and state estimates are constantly updated and adjusted to incoming measurements, situations might occur where the hypotheses are not well separated. To saved computations the hypotheses can then be merged to one. Merge candidates can be discovered by evaluating the Mahalanobis distance between the map estimates. Let hypothesis $\mathcal{H}_A$ and $\mathcal{M}_B$, have map estimate $\mathcal{M}_A$ and $\mathcal{M}_B$, and associated covariance matrices $R_{\mathcal{M}_A}$ and $R_{\mathcal{M}_B}$, respectively. The hypotheses are considered similar if the Mahalanobis distance is smaller than a threshold,

$$(\mathcal{M}_A - \mathcal{M}_B)^\top (R_{\mathcal{M}_A} + R_{\mathcal{M}_B})^{-1} (\mathcal{M}_A - \mathcal{M}_B) < \kappa^2. \tag{26}$$

The hypotheses can be merged by linear combination weighted by the probability of each of the hypothesis.

## VIII. Multi-Hypothesis SLAM

This section describes how the multi-hypothesis map representation can be used together with a static SLAM solver of your choice.

### A. Implementation

The multi-hypothesis map representation can be used in the SLAM framework together with any SLAM solver that can provide estimates of $p(x_t, \mathcal{M}|\delta_t, z_{1:t})$ and of the measurement likelihood $p(z^{\mathcal{J}}_t|\mathcal{H}^j_k)$. The workflow of multi-hypothesis SLAM is described in Algorithm 1. First all active SLAM solvers, which corresponds to leafs in the hypothesis tree, are predicted to the current time step, then possibly new hypotheses are created and added to the hypothesis tree. All unnormalized log-likelihoods are updated for all nodes in the hypothesis tree according to (14) and then the pair-wise scores are computed and compared to the decision thresholds. If decisions can be made the hypothesis tree is pruned accordingly.

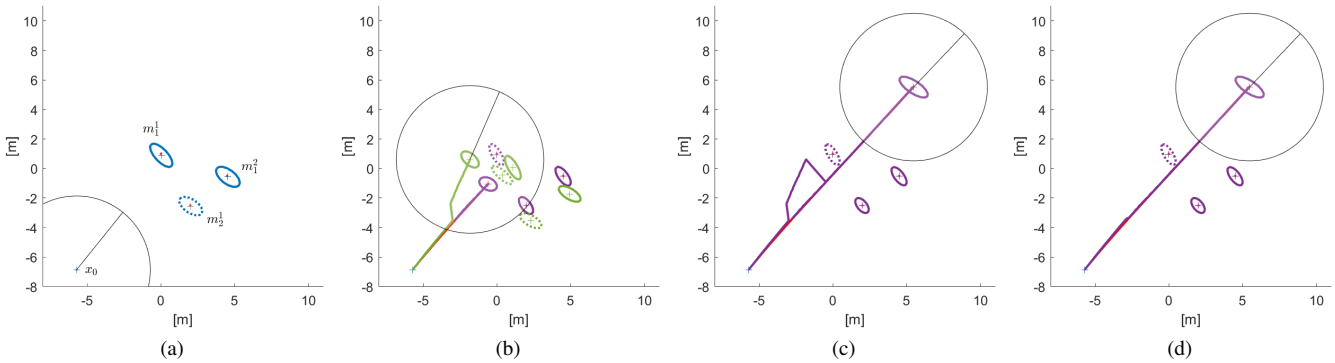


Fig. 3. A prior map is given where the landmark $m^1$ has a mode indicator not corresponding with observations to come. In (a) the prior configuration is shown. Landmark and state estimate are marked with position and uncertainty ellipse. The FOV of the robot is visualized with a black circle and a line reflecting the heading. Dashed landmarks mark inactive modes in the landmark group associated with that color. In (b) the green graphics corresponds to the original hypothesis $\mathcal{H}_A^1$ where the landmark is still positioned at $m_1^1$, but slightly adapted to observation originating from $m_2^1$. The purple graphic represents $\mathcal{H}_B^1$. Note that the erroneous hypothesis is chosen according to maximum likelihood. In (c) when $m_1^2$ enters FOV, the correct hypothesis is eventually selected. If instead only gated observations are included in the correction step, the incorrect hypothesis is quickly rejected when consecutive false alarms are observed. This is shown in (d).

**Algorithm 1** Multi-Hypothesis SLAM

**Input:** $x_0$, $P_0$, $u_{1:T}$, $z_{1:T}$
Initialize root SLAM solver ($x_0$, $P_0$)
**for** t = 1:T **do**
  **for all** leafs in the hypothesis tree **do**
    $p(x_t, \mathcal{M}|\delta_t, z_{1:t-1}, u_{1:t}) \leftarrow$ SLAM time update
    Branch hypothesis if needed
  **end for**
  Update all scores, $\hat{L}_t^{\mathcal{H}_k^j}$
  Prune hypothesis tree by decisions or merging
  **for all** leafs in the hypothesis tree **do**
    $p(x_t, \mathcal{M}|\delta_t, z_{1:t}, u_{1:t}) \leftarrow$ SLAM measurement update
  **end for**
  $p(x_t, \mathcal{M}, \delta_t|z_{1:t}) \leftarrow$ global estimate
**end for**

The hypotheses are also checked for merging and the tree is pruned if necessary. Finally, the SLAM measurement update is performed for all leafs in the hypothesis tree, and a global estimate is obtained from the hypothesis tree.

To obtain stability due to noise and uncertainties in the estimates, a list of landmarks currently in FOV for each SLAM solver is managed through:

- $P_{\text{FOV}} > \gamma_1$: add to the list of landmarks in FOV,
- $P_{\text{FOV}} < \gamma_2$: remove from the list of landmarks in FOV,

where $\gamma_1 > \gamma 2$. The thresholds should be high enough not to initiate alternative hypotheses when they should not, and, low enough for truly valid observations not to be miss-classified as false alarms.

### B. The Global Hypothesis

To retrieve a global hypothesis at all times during execution, a *maximum likelihood* (ML) approach can be taken. In cases with a complex tree structure, this can be obtained by starting at the root and selecting the branch with highest score all the way down to a leaf. That leaf provides the global hypothesis.

## IX. Underground Mine Experiments

The multi-hypothesis SLAM algorithm is evaluated in an underground mine application using a mid-articulated loader

Fig. 4. A *Scooptram18* loader from Epiroc, equipped with odometry and two lidar sensors, is used for data collection. [Asset: Epiroc]

equipped with odometry and two 2D lidar sensors, see Fig. 4. The operation area is a long tunnel with crossroads leading to drawpoints where material is picked up, see Fig. 5. To manage the ore body, drawpoints are closed or opened with physical barriers to prevent loaders from entering the wrong aisle. If the physical barriers can be uniquely identified by the perception system of the vehicle, a situation suitable for multi-hypothesis SLAM is achieved. This situation is evaluated with both simulated data and with real data recorded in an underground mine in the area covered by the map in Fig. 5.

### A. SLAM Solver

The SLAM solvers implement *extended Kalman filter* (EKF) SLAM [35] and estimates the 2D position of the vehicle, *i.e.*, the Cartesian coordinates of the vehicle in the global frame together with the heading, $\left[x^1, x^2, \theta\right]^\top$. The state vector is extended by the Cartesian coordinates of discovered landmarks, which yields the full state vector

$$x_t = \left[x_t^1, x_t^2, \theta_t, \mathcal{M}_t^\top\right]^\top. \tag{27}$$

The state transition model is nonlinear and given by

$$x_{t+1} = x_t + T \begin{bmatrix} u_t^v \cos\theta_t \\ u_t^v \sin\theta_t \\ u_t^w \end{bmatrix} + v_t, \tag{28}$$

where $T = 50\,\text{ms}$ is the sampling rate, $u_t^v$ and $u_t^w$ are the linear and angular velocity, respectively and is obtained from the odometry. The process noise $v_t \sim \mathcal{N}(0, Q)$ is

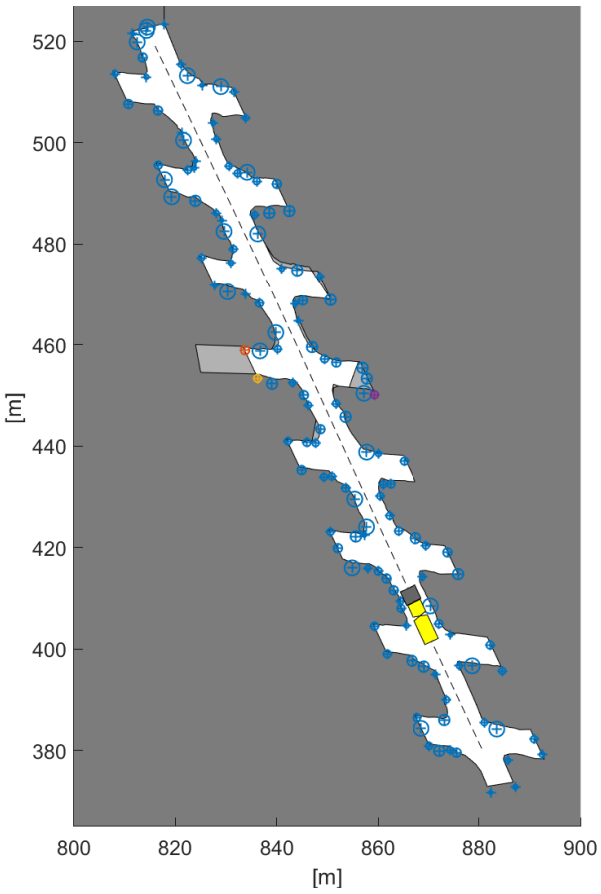


Fig. 5. The operation area with the two polygon-based metric maps overlayed. Lidar data is simulated while the loader travels along the dashed line to obtain a prior feature-based map. The landmarks in the prior map is marked with a cross and covariance ellipses. Different colors is associated with the different signatures of the landmarks, and blue landmarks are considered not uniquely identifiable.

modeled as time-invariant with diagonal covariance $Q = 0.05 \ \mathrm{diag}\left(0.05, 0.05, (5^\circ)^2\right)$, chosen according to previous knowledge about the application.

The measurement model is a transformation of the landmarks in the map to the vehicles coordinate frame. For each landmark currently in field-of-view this yields,

$$z_t^i = \mathcal{R}(x_t)\left(\mathcal{M}_t^{\mathrm{FOV},i} - \left[x_t^1, x_t^2\right]^\top\right) + e_t, \qquad (29)$$

where $\mathcal{R}(x_t)$ is a rotation matrix and $\mathcal{M}_t^{\mathrm{FOV},i}$ is the $i$th landmark in the subset of visible landmarks in the complete map $\mathcal{M}_t$ selected by the mode vector. The measurement noise is assumed white Gaussian and equal for all $i$ landmarks, $e_t \sim \mathcal{N}(0, R)$, with $R = 0.25I$ where $I$ is the identity matrix.

### B. Data Generation

Lidar data is obtained as 2D point clouds either directly from recorded log-files or simulated by ray-casting in a polygon defined metric map, available for this particular area. Measurement errors added to the simulated data is created utilizing a *support vector machine* (SVM) model, trained on real data from an underground application, where the measurement error depends on range and inclination angle of a laser ray to the wall [39]. This is the same model as used in [40] where more details and internal parameter values can be found. This method of producing measurement errors gives rather realistic sensor data.

From the point clouds, features are extracted with the *fast laser interest region transform* (FLIRT) algorithm presented in [41] and evaluated with comparatively good results for the underground application in [40]. The method searches for high-curvature regions and identifies keypoints from peaks of an evaluation function $F$. The minimum value of $F$ to be considered a peak is set to $F_{\min} = 0.367$ and the minimum separation of peaks is set to $F_{\mathrm{dist}} = 0.12$, to identify enough keypoints to allow for good positioning while keeping them few to be well separated. All other parameters associated with the FLIRT algorithm are set according to [40].

Since the feature-based map representation is sensor agnostic, it is assumed that the unique identifiers are in place and what actual device that in practice is used to obtain this can be application dependent. In these experiments the identifiers are added by manually assigning a signature to landmarks discovered within bounding boxes where changes are applied. All other landmarks are not considered uniquely identifiable and are therefore not considered for branching. When giving landmarks their signatures the ground truth position of the loader is used. For the real data, no ground truth is available and the estimated position from an internal positioning system, executed on the loader and based on matching raw lidar data to the polygon defined metric map, is used instead.

### C. Prior Map Generation

Two versions of the polygon defined metric map is available of the operation area, see Fig. 5, the original one and a modified one where a tunnel is extended and a corner is moved closer to the main aisle. A prior map and associated covariance matrix is generated by executing the EKF SLAM solver with simulated lidar data from the ground truth trajectory defined in Fig. 5, where also the obtained map is presented. Three uniquely identified landmarks are present in the map and all other landmarks are not considered for branching.

### D. Probability of FOV

The probability that a landmark is in FOV is in this implementation estimated by considering the raw point cloud obtained from the lidar sensor. Since the map $\mathcal{M}$ is completely defined by landmark points it is not possible to find occlusions directly from the map, at least not without further processing of the map. However, by using the actual point cloud from the sensors it is possible to estimate the probability that a particular landmark is within the area covered by the point cloud.

100 samples of each landmark position and vehicle position are created given their estimated distributions. The point cloud is transformed to each sampled vehicle position and a ratio is obtained on how often the landmark is withing the area covered by the point cloud. This ratio is used as an estimate of $P_{\mathrm{FOV}}$. In the ideal case, the landmark would be on the exact border of the point cloud area, the ratio approximation

will therefore vastly underestimate the $P_{\text{FOV}}$. To compensate for this, the point cloud areas are extended with $0.5\,\text{m}$ in the radial direction.

### E. Simulated Data Experiment

A simulated data experiment is conducted where a trajectory is created in the modified map. White Gaussian noise is added to the odometry data by

$$u_t = \begin{bmatrix} u_t^v \\ u_t^w \end{bmatrix} + \begin{bmatrix} e_t^v \\ e_t^w \end{bmatrix}, \tag{30}$$

where $e_t^v \sim \mathcal{N}(0, 0.1^2)$ and $e_t^w \sim \mathcal{N}(0, (5^\circ)^2)$. Landmark measurements are simulated according to the process described in Section IX-B, and signatures are manually assigned to match the scenario that the unique landmarks in the prior map have moved.

The multi-hypothesis SLAM algorithm is given the prior map in Fig. 5, clutter and new landmark rates are set to $\beta_{\text{FA}} = \beta_{\text{NT}} = 3.5 \cdot 10^{-7}$, the FOV thresholds are $\gamma_1 = 0.8$, $\gamma_2 = 0.1$, and, the score threshold for decision is set to $\alpha = 10^{-8}$. An elliptic gating is used for matching observations with landmarks in the map, with the significance level set to $99\%$.

The transition matrix is assumed equal for all landmarks and is set to identity when a landmark is in FOV. During a time period when a landmark is not in FOV it could possibly change mode in any time step during the period. However, if it is assumed that the transition matrix is constant when a landmark is not in FOV a final transition probability at the end of the time period yields,

$$\begin{bmatrix} \Pr(\delta_{t+i}^j = k) \\ \Pr(\delta_{t+i}^j = q) \end{bmatrix} = \left(\Pi^j\right)^i \begin{bmatrix} \Pr(\delta_t^j = k) \\ \Pr(\delta_t^j = q) \end{bmatrix}. \tag{31}$$

The values of $\Pi$ is highly application dependent and should be tuned to match the frequency of changes in the environment. Due to how the experiments is setup here, the time $i$ since a landmark was previously in FOV is not known. Using a transition probability of 0.5 would correspond to a stationary state, while letting the diagonal values be larger than the off-diagonal ones considers the last observed mode more probable. In these experiments $\left(\Pi^j\right)^i = \widetilde{\Pi} = \left[\begin{smallmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{smallmatrix}\right]$ is used, which results in a more conservative algorithm not advantageous in experiments where the landmarks have truly changed mode.

When running this simulation with a static SLAM solver the vehicle position is shifted to the right when the unique landmarks comes into FOV. The distances to observation of the landmarks in the extended drawpoint is much longer than expected and the SLAM solver compensates by correcting both the landmark positions and the vehicle position. If instead the multi-hypothesis approach is used, new hypotheses of all three unique landmarks are started and decisions are made according to incoming observations.

Fig. 6 shows the *root mean square error* (RMSE) and the *normalized estiamted error squared* (NEES) of the estimated vehicle position, relative the known ground truth. The NEES is computed by $(x_t - \hat{x}_t)^\top \hat{P}_t^{-1} (x_t - \hat{x}_t)$, where $x_t$ are the true state vector, $\hat{x}_t$ and $\hat{P}_t$ are the estimated state and

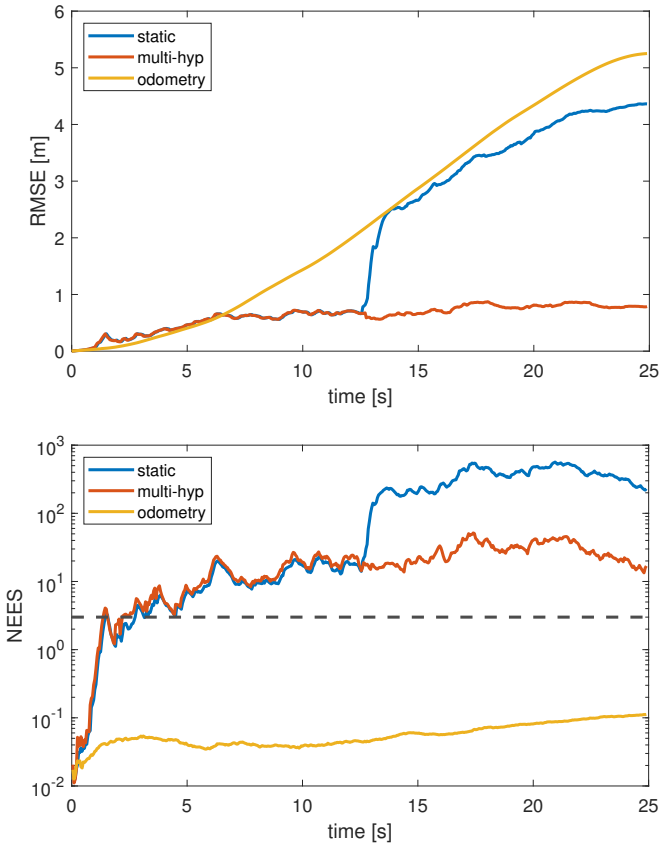


Fig. 6. Results for the simulated data experiment with 100 Monte Carlo realizations. Orange line is the multi-hypothesis SLAM algorithm, blue is when static SLAM is used and yellow is dead-reckoning with only odometer data. The black dashed line in the NEES plot marks the optimal value, 3.

covariance, respectively. For Gaussian distribution of the state, this quantity is $\chi^2$-distributed and the optimal value is equal to the dimension of the state vector [42]. If only the elements of the state corresponding to the vehicle position is considered, the optimal NEES value is in this case 3. As seen in Fig. 6 a dead-reckoning solution gives conservative estimates since the NEES values is below three. The multi-hypothesis and static SLAM approaches are similar until half-way when the unique landmarks come into FOV. The static SLAM solver then diverges while the multi-hypothesis approach remains close to the true solution. The multi-hypothesis approach also gives a correct posterior version of the map where the prior positions of the landmarks are marked as inactive.

### F. State-of-the-art Comparison

State-of-the-art methods for SLAM in dynamic or non-static environment often adopts either an outlier rejection method where observations not coherent with the map is filtered out, or, some sort of memory decay where non-robust landmarks are eventually deleted. A simulation is therefore performed comparing the suggested multi-hypothesis approach with an ideal outlier rejection and an ideal memory decay.

The prior map used in this experiment is the output from the previous one. That is, both modes of the uniquely identifiable landmarks are present in the map. Data is simulated in the

original map but the multi-hypothesis SLAM algorithm is given the mode indicator vector corresponding to landmark positions in the modified map. Assuming it was a long time since this particular aisle was changed, a perfect memory decay would completely have forgotten the original version of the map. Therefore this SLAM solver is executed as a static EKF SLAM solver given a prior map with the original modes of the landmarks removed. This solver allows multiple landmarks with the same signature to be added if observations not gated with the known position is made and the original position of the landmark will be re-added. If a perfect outlier rejection is assumed, all observations not matching the known position of a landmark would be considered an outlier. This would be the case when a landmark is moved. A outlier rejection SLAM solver is therefore executed ignoring all observations from the moving landmarks, only using the not uniquely identifiable ones.

The simulation constitutes 100 Monte Carlo realizations on the subpart of the trajectory in Fig. 5 where the moving landmarks are visible. Each run is started with a prior uncertainty in the starting position given by the covariance $\mathrm{diag}([0.04, 0.04, 0.0023])$, which is estimated by running the trajectory once from the very beginning.

Using the ML hypothesis in each time step as the global state estimate, might cause the multi-hypothesis algorithm to, at specific time steps, use the wrong hypothesis, even though the correct one is still active and updated and decided upon at a later stage. Therefore the methods are evaluated by the RMSE of the best active hypothesis at only the last time step when the changing landmarks have left field-of-view. In this experiment the moving landmarks are few compared to the number of static landmark visible at the same time. To evaluate how the algorithms perform when a larger ratio of observations belong to moving landmarks, a percentage of the static landmarks in the prior map is removed and also the same percentage of observations of the static landmarks are removed. Fig. 7 show the RMSE of the vehicle position at the final time step for the different algorithms and for different percentages of removed landmarks. When no landmark is removed the differences between the algorithms are small, but as landmarks are removed, the moving landmarks have a higher impact on the estimates and the multi-hypothesis algorithm outperforms the others. This clearly shows the benefits of saving a rejected hypothesis if it is likely to re-occure in the future. The drop in RMSE when all static landmarks are removed seen in Fig. 7 can be explained by erroneous data associations. Since the static landmarks are not uniquely identifiable observations might be gated with the wrong landmark, causing erroneous input to the SLAM solver updates.

### G. Real Data Experiment

An experiment with real data collected by a loader in this particular operation area, gives similar results as the simulated experiment. The real data is more coherent with the original map than the modified. Therefore a prior map is created by simulation in the modified map and then the multi-hypothesis SLAM solver is executed with the same parameter setup as

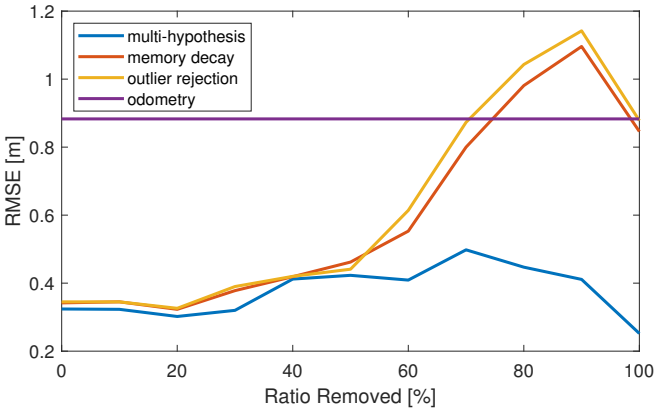


Fig. 7. Comparison of the multi-hypothesis SLAM algorithm with a perfect memory decay and a perfect outlier rejection. Landmarks in the map and in observations are randomly removed with varying ratios. Running with pure odometry is unaffected by the ratio since no landmarks are used in the dead-reckoning state estimate.

presented in Section IX-E. Since no ground truth is available for this data RMSE and NEES cannot be computed but Fig. 8 show the resulting trajectory when the loader is traveling from the top to the bottom in the map, creating new hypotheses for the uniquely identifiable landmarks and make decisions accordingly. This result is similar to the simulated results, which further validates the simulations and the benefits of the multi-hypothesis approach.

Worth noting is that for one of the landmarks (the purple one in Fig. 8) multiple hypotheses are never started. By inspecting an actual laser scan from this part of the area this is not surprising. Fig. 9 shows that the sensor data is not more coherent with the modified map than the original one. This highlights the flexibility and robustness of the multi-hypothesis approach, where the end result reflects the actual observations well.

## X. Conclusion

The assumption of a static world is a prerequisite for SLAM to be considered a solved problem. In reality this is almost never the case. This paper have considered a scenario where uniquely identifiable landmarks are present and can be moved between multiple discrete positions, not known *a priori*. A multi-hypothesis SLAM algorithm have been developed based on the feature based multi-hypothesis map representation introduced in [15], where also a hypothesis score enables decision making to prevent the number of hypotheses to grow unbounded. This map representation has theoretically been incorporated into the SLAM framework and a tree structure has been formalized to manage the hypotheses. Many practical considerations have been elaborated upon and suitable approximations are suggested. This have resulted in a multi-hypothesis SLAM algorithm that works in combination with any available SLAM solver designed for a static environment.

The developed algorithm have been evaluated in experiments in the underground mine application, conducted with both simulated and real data. In this non-static environment, simulations show the benefits of the multi-hypothesis approach

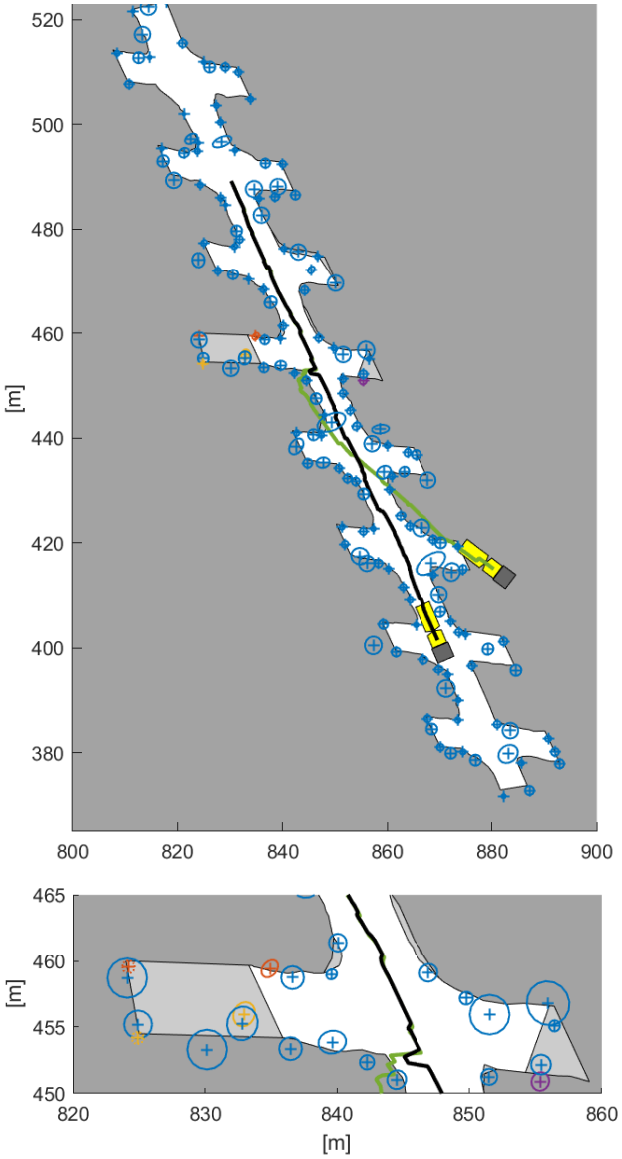


Fig. 8. The top picture show the black trajectory estimated when the multi-hypothesis SLAM algorithm is used. The landmarks shown are the estimates at the final time step. The green trajectory is the outcome from a static SLAM approach applied to this problem. Note that the estimated landmark positions for this approach is not shown in the figure. The bottom picture is zoomed in on the modified part of the map, and shows landmarks that are inactive according to the estimated mode indicator vector with dashed covariance ellipses.

compared to executing a standard SLAM solver. It also performs better compared to ideal outlier rejection and ideal memory decay, which are methods that many state-of-the-art SLAM solutions for non-static environments are based upon. The experiments on real data further validates the simulated results and show the potential of this approach to perform well in real world applications.

Future work will focus on not being reliant on the assumption that landmark are uniquely identifiable, which will allow uncertainty also in the data association.

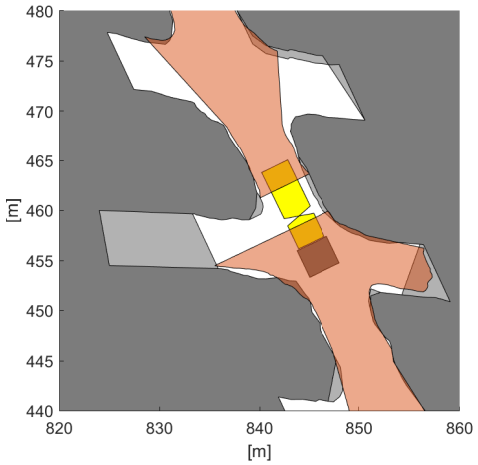


Fig. 9. An example of a real lidar scan, shown as an orange area. For the drawpoint to the right in the picture where one of the corners are changed in the two map versions, the real observations are somewhere in between the two map versions.

## References


[1] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 2, no. 3, pp. 194–220, sep 2017.

[2] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proc. of the 18th Int. joint Conf. on Artificial intelligence*, 2003, pp. 1151–1156.

[3] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[4] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, 2011, pp. 3607–3613.

[5] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[6] S. Yi, S. Worrall, and E. Nebot, "Metrics for the evaluation of localisation robustness," in *Intelligent Vehicles Symposium (IV)*. Paris, France: IEEE, jun 2019, pp. 1247–1253.

[7] J. S. Berrio, J. Ward, S. Worrall, and E. Nebot, "Updating the visibility of a feature-based map for long-term maintenance," in *IEEE Intelligent Vehicles Symposium (IV)*, Paris, France, 2019, pp. 1173–1179.

[8] ——, "Identifying robust landmarks in feature-based maps," in *IEEE Intelligent Vehicles Symposium (IV)*, Paris, France, 2019, pp. 1166–1172.

[9] J. S. Berrio, S. Worrall, M. Shan, and E. Nebot, "Long-term map maintenance pipeline for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10 427–10 440, aug 2022.

[10] A. Fabris, L. Parolini, S. Schneider, and A. Cenedese, "Use of probabilistic graphical methods for online map validation," in *Proc. IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*. Nagoya, Japan: IEEE, jul 2021, pp. 43–48.

[11] S. Cho, C. Kim, M. Sunwoo, and K. Jo, "Robust localization in map changing environments based on hierarchical approach of sliding window optimization and filtering," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 4, pp. 3783–3789, apr 2022.

[12] L. Delobel, R. Aufrere, C. Debain, R. Chapuis, and T. Chateau, "A real-time map refinement method using a multi-sensor localization framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1644–1658, may 2019.

[13] C. Stachniss and W. Burgard, "Mobile robot mapping and localization in non-static environments," in *Proc. of the 20th National Conference on Artificial Intelligence - Volume 3*, ser. AAAI'05. Pittsburgh, Pennsylvania, USA: AAAI Press, 2005, pp. 1324–1329.

[14] F. Schuster, M. Worner, C. G. Keller, M. Haueis, and C. Curio, "Robust localization based on radar signal clustering," in *IEEE Intelligent*

*Vehicles Symposium (IV)*. Gothenburg, Sweden: IEEE, jun 2016, pp. 839–844.
[15] K. Nielsen and G. Hendeby, "Feature based multi-hypothesis map representation for localization in non-static environments," in *Proc. 24th IEEE Int. Conf. on Information Fusion (FUSION)*, Linköping, Sweden, 2022, pp. 1–8.
[16] D. Hahnel, R. Triebel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, Taipei, Taiwan, 2003, pp. 1557–1563.
[17] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas," in *Proc. Int. Conf. on Robotics and Automation*, vol. 1, Taipei, Taiwan, 2003, pp. 842–849.
[18] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889–916, sep 2007.
[19] W. Dai, Y. Zhang, Y. Zheng, D. Sun, and P. Li, "RGB-D SLAM with moving object tracking in dynamic environments," *IET Cyber-Systems and Robotics*, vol. 3, no. 4, pp. 281–291, may 2021.
[20] D. M. Rosen, J. Mason, and J. J. Leonard, "Towards lifelong feature-based mapping in semi-static environments," in *Proc. Int. Conf. on Robotics and Automation (ICRA)*, Stockholm, Sweden, 2016, pp. 1063–1070.
[21] N. Sünderhauf and P. Protzel, "Switchable constraints for robust pose graph SLAM," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Vilamoura-Algarve, Portugal, 2012, pp. 1879–1884.
[22] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard, "Robust map optimization using dynamic covariance scaling," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 62–69.
[23] E. Olson and P. Agarwal, "Inference on networks of mixtures for robust robot mapping," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 826–840, 2013.
[24] D. M. Rosen, M. Kaess, and J. J. Leonard, "Robust incremental online inference over sparse factor graphs: Beyond the Gaussian case," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 1025–1032.
[25] F. Jiang, V. Agrawal, R. Buchanan, M. Fallon, and F. Dellaert, "iMHS: An Incremental Multi-Hypothesis Smoother," *ArXiv*, vol. abs/2103.13178, 2021.
[26] M. Hsiao and M. Kaess, "MH-iSAM2: Multi-hypothesis iSAM using Bayes Tree and Hypo-tree," in *Proc. Int. Conf. on Robotics and Automation (ICRA)*, 2019, pp. 1274–1280.
[27] P. Biber and T. Duckett, "Experimental analysis of sample-based maps for long-term SLAM," *The International Journal of Robotics Research*, vol. 28, no. 1, pp. 20–33, jan 2009.
[28] J. Nieto, J. Guivant, E. Nebot, and S. Thrun, "Real time data association for FastSLAM," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 1, Taipei, Taiwan, 2003, pp. 412–418.
[29] M. Adams, B.-N. Vo, R. Mahler, and J. Mullane, "SLAM gets a PHD: new concepts in map estimation," *IEEE Robot. Autom. Mag.*, vol. 21, no. 2, pp. 26–37, jun 2014.
[30] H. Deusch, S. Reuter, and K. Dietmayer, "The labeled multi-Bernoulli SLAM filter," *IEEE Signal Process. Lett.*, vol. 22, no. 10, pp. 1561–1565, oct 2015.
[31] A. Falchetti and M. Adams, "Probability hypothesis density filter visual simultaneous localization and mapping," in *Int. Conf. on Control, Automation and Information Sciences (ICCAIS)*. Xi'an, China: IEEE, oct 2021, pp. 879–886.
[32] M. Stubler, S. Reuter, and K. Dietmayer, "A continuously learning feature-based map using a bernoulli filtering approach," in *Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. Bonn, Germany: IEEE, oct 2017.
[33] M. Fatemi, K. Granstrom, L. Svensson, F. J. R. Ruiz, and L. Hammarstrand, "Poisson multi-Bernoulli mapping using Gibbs sampling," *IEEE Trans. Signal Process.*, vol. 65, no. 11, pp. 2814–2827, jun 2017.
[34] N. M. Kwok and G. Dissanayake, "An efficient multiple hypothesis filter for bearing-only SLAM," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 1, Sendai, Japan, 2004, pp. 736–741.
[35] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT Press, 2005.
[36] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. 18th National Conference on Artificial Intelligence*. USA: American Association for Artificial Intelligence, 2002, pp. 593—-598.
[37] V. P. Dragalin, A. G. Tartakovsky, and V. V. Veeravalli, "Multihypothesis sequential probability ratio tests .i. asymptotic optimality," *IEEE Trans. Inf. Theory*, vol. 45, no. 7, pp. 2448–2461, 1999.
[38] Y. Bar-Shalom, P. K. Willet, and X. Tian, *Tracking and data fusion : a handbook of algorithms*. YBS Publishing, 2011.
[39] A. Tallavajhula, "Lidar simulation for robotic application development: modeling and evaluation," Ph.D. dissertation, The Robotics Institute, Carnegie Mellon University Pittsburgh, May 2018.
[40] K. Nielsen and G. Hendeby, "Survey on 2D lidar feature extraction for underground mine usage," *IEEE Trans. Autom. Sci. Eng.*, pp. 1–14, 2022.
[41] G. D. Tipaldi and K. O. Arras, "FLIRT - interest regions for 2D range data," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Anchorage, AK, USA, 2010, pp. 3616–3622.
[42] Z. Chen, C. Heckman, S. Julier, and N. Ahmed, "Weak in the NEES?: auto-tuning Kalman filters with Bayesian optimization," in *Proc. 21st Int. Conf. on Information Fusion (FUSION)*, Cambridge, UK, 2018, pp. 1072–1079.

**Kristin Nielsen** is a Phd student at the division of Automatic Control, Department of Electrical Engineering, Linköping University. She received her MSc in Engineering Physics in 2011 from Uppsala University and has since been employed as Automation Software Developer at Epiroc Rock Drills AB. Epiroc is a company manufacturing vehicles and equipment, as well as offering automation solutions, for the mining industry. Kristin's Phd studies are conducted in collaboration with Epiroc and is partially sponsored by the Wallenberg AI, Autonomous Systems and Software Program (WASP).

**Gustaf Hendeby** (S'04-M'09-SM'17) received the M.Sc. degree in applied physics and electrical engineering in 2002 and the Ph.D. degree in automatic control in 2008, both from Linköping University, Linköping, Sweden.

He is Associate Professor and Docent in the division of Automatic Control, Department of Electrical Engineering, Linköping University. He worked as Senior Researcher at the German Research Center for Artificial Intelligence (DFKI) 2009–2011, and Senior Scientist at Swedish Defense Research Agency (FOI) and held an adjunct Associate Professor position at Linköping University 2011–2015. His main research interests are stochastic signal processing and sensor fusion with applications to nonlinear problems, target tracking, and simultaneous localization and mapping (SLAM), and is the author of several published articles and conference papers in the area. He has experience of both theoretical analysis as well as implementation aspects.

Dr. Hendeby is since 2018 an Associate Editor for IEEE Transactions on Aerospace and Electronic Systems in the area of target tracking and multisensor systems. In 2022 he served as general chair for the 25th IEEE International Conference on Information Fusion (FUSION) in Linköping, Sweden.