Confidence-aware Occupancy Grids

Ali-akbar Agha-mohammadi[1], Eric Heiden[2], Karol Hausman[2], Gaurav S. Sukhatme[2]

I. Introduction

Occupancy grid maps are among the most common representations of the environment when dealing with range sensors [1]. Typically, each grid cell stores the information about whether it is occupied or free in a binary form. In a different, slightly richer representation, each voxel contains the probability of being occupied. In the main body of literature occupancy grids are used to store binary occupancies updated by the log-odds method [2, 3]. Even though the log-odds-based occupancy grids have enjoyed success in a variety of applications, these methods, especially when coping with noisy sensors, suffer from three main issues:

1) The occupancy of each voxel is updated independently of the rest of the map. This is a well-known problem [2] which has been shown to lead to conflicts between map and measurement data. In particular, when the sensor is noisy or has a large field of view there is a clear coupling between voxels that fall into the field of view of the sensor.
2) The log-odds methods rely on the *inverse sensor model* (ISM), which needs to be hand-engineered for each sensor and a given environment.
3) In order to represent the voxel occupancy, each voxel stores a single number. As a result, there is no consistent confidence or trust value associated with it to help the planner decide how reliable the estimated occupancy is.

In this paper, we propose a method that partially relaxes these assumptions, generates more accurate maps, and provides a more consistent filtering mechanism than prior approaches. The highlights and contributions of this work are as follows:

1) The main assumption in traditional occupancy grid mapping is (partially) relaxed. We take into account the dependence between voxels in the measurement cone at every step. Further, the proposed method relaxes the binary assumption on the occupancy level and it is capable of coping with maps where each voxel is only partially occupied by obstacles.
2) We replace the ad-hoc inverse sensor model by a novel "sensor cause model", which is computed based on the forward sensor model in a principled manner.
3) In addition to the most likely occupancy value for each voxel, the proposed map representation contains confidence values (e.g. variance) of voxel occupancies. The confidence information is crucial for planning over grid maps. We incorporate the sensor model and its uncertainties in characterizing the map accuracy.
4) While the majority of approaches that relax the voxel-independence assumption are batch methods, our method does not require logging of the data in an offline phase. Instead, the map can be updated online as the sensory data are received.

[1]Ali-akbar Agha-mohammadi is with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, USA aliagha@jpl.nasa.gov

[2]Eric Heiden, Karol Hausman and Gaurav S. Sukhatme are with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA heiden@usc.edu

II. Confidence-rich Representation

For each voxel i we store the probability distribution of occupancy m^i. Here, the variable m^i can be interpreted in two ways:

1) In the more general setting, $m^i \in [0, 1]$ directly represents the occupancy level (the percentage of voxel i that is occupied by obstacles.). The proposed method can model continuous occupancy and relaxes the binary occupancy assumption in traditional occupancy mapping.
2) If the underlying true map is assumed to be a binary map (denoted by ${}^{bin}m$), the occupancy of the i-th voxel ${}^{bin}m^i \in \{0, 1\}$ is distributed as Bernoulli distribution ${}^{bin}m^i \sim Bernoulli(m^i)$. Hence, m^i refers to the parameter of the Bernoulli distribution. While inverse sensor-based mapping methods store m^i as a deterministic value, we estimate m^i probabilistically based on measurements and store its pdf in each voxel. Note that in this setting, ${}^{bin}m^i \in \{0, 1\}$, where $m^i \in [0, 1]$ represents the occupancy probability, i.e., $m^i_k = p({}^{bin}m^i = 1 | z_{0:k}, x_{0:k})$.

Problem description: Given the above-mentioned representation, we aim at estimating m based on noisy measurements by computing its posterior distribution. We define three beliefs over the map: (1) full map belief $\bar{b}^m_k = p(m|z_{0:k}, x_{0:k})$, (2) marginal cell beliefs $b^{m^i}_k = p(m^i|z_{0:k}, x_{0:k})$, and (3) the collection of marginals $b^m_k = (b^{m^i}_k)^{M}_{i=1}$[1]. Similar to ISM-based methods, for mapping we maintain and update the collection of marginals b^m_k. To do so, we derive the following items:

1) **Ranging sensor model:** Given the obstacles are described by a stochastic map, we derive a ranging sensor model, i.e., the probability of obtaining measurement z given a stochastic map and robot location: $p(z_k|x_k, b^m_k)$. This model will be utilized in the map update module.
2) **Recursive density mapping:** We derive a recursive mapping scheme τ that updates the current density map based on the last measurements

$$b^{m^i}_{k+1} = \tau^{m^i}(b^m_k, z_{k+1}, x_{k+1}). \quad (1)$$

[1]More precisely, in these definitions the variable b refers to the set of parameters that characterize the probability distributions. So, we will treat b as a vector (deterministic or random depending on the context) in the rest of the paper.

The fundamental difference with ISM-mapping is that the evolution of the i-th voxel depends on other voxels as well. Note that the input argument to τ^{m^i} is the collection of all voxel beliefs b^m, not just the i-th voxel map b^{m^i}.

Overall, this method relaxes the voxel independence and the binary occupancy assumptions of the ISM-based mapping.

III. Range-sensor Modeling

In this section, we model a range sensor when the environment representation is a stochastic map. We focus on passive ranging sensors like stereo cameras, but the discussion can easily be extended to active sensors too.

Ranging pixel: Let us consider an array of ranging sensors (e.g., disparity pixels). We denote the camera center by x_{cam}, the 3D location of the i-th pixel by v, and the ray emanating from x_{cam} and passing through v by $x = (x_{cam}, v)$. Let r denote the distance between the camera center and the closest obstacle to the camera along ray x. In stereo camera range r is related to the measured disparity z as:

$$z = r^{-1} f d_b \tag{2}$$

where, f is camera's focal length and d_b is the baseline between two cameras on the stereo rig. In the following, we focus on a single pixel v and derive the forward sensor model $p(z|x, b^m)$.

Pixel cone: Consider the field of view of pixel v. Precisely speaking, it is a narrow 3D cone with apex at x and boundaries defined by pixel v. Also, for simplicity one can consider just a ray x going through camera center x and the center of pixel v. Pixel cone $\texttt{Cone}(x)$ refers to the set of voxels in map m that fall into this cone (or lie on ray x). We denote this set by $\mathbb{C} = \texttt{Cone}(x)$.

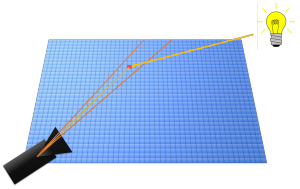

Fig. 1. The measurement cone formed by two red lines represents the field of view of pixel v. The disparity measurement on pixel v can be caused by light bouncing off from any of the voxels (here the red voxel) in the pixel cone and reaching the image plane.

Local vs global indices: For a given ray x, we order the voxels along the ray from the closest to the camera to the farthest from the camera. Notation-wise, $i^l \in \{1, \cdots, \|\mathbb{C}\|\}$ denote the local index of a voxel on ray x. Function $i^g = g(i^l, x)$ returns the global index i^g of this voxel in the map.

Cause variables: The disparity measurement on pixel v could be the result of light bouncing off any of voxels in the cone $\mathbb{C} = \texttt{Cone}(x)$ (see Fig. 1). Therefore, any of these voxels are a potential cause for a given measurement. In the case that the environment map is perfectly known, one can pinpoint the exact cause by finding the closest obstacle to the camera center. But, when the knowledge about the environment is partial and probabilistic, the best one can deduce about causes is a probability distribution over all possible causes in the pixel cone $\mathbb{C} = \texttt{Cone}(x)$. These causes will play an important role (as hidden variables) in deriving sensor model for stochastic maps.

Cause probability: To derive the full sensor model, we need to reason about which voxel was the cause for a given measurement. For a voxel $c \in \mathbb{C}(x)$ to be the cause, two events need to happen: *(i)* B^c, which indicates the event of light bouncing off voxel c and *(ii)* R^c, which indicates the event of light reaching the camera from voxel c.

$$p(c|b^m) = \Pr(B^c, R^c|b^m) = \Pr(R^c|B^c, b^m)\Pr(B^c|b^m) \tag{3}$$

Bouncing probability: To compute the bouncing probability, we rely on the fact that $\Pr(B^c|m^c) = m^c$ (by the definition). Note that $\Pr(B^c|m^c, b^m) = \Pr(B^c|m^c)$.

$$\begin{aligned}\Pr(B^c|b^m) &= \int_0^1 \Pr(B^c|m^c, b^m) p(m^c|b^m) dm^c \\ &= \int_0^1 m^c b^{m^c} dm^c = \mathbb{E}m^c = \widehat{m}^c\end{aligned}$$

Reaching probability: For the ray emanating from voxel c to reach the image plane, it has to go through all voxels on ray x between c and sensor. Let c^l denotes local index of voxel c along the ray x, i.e., $c^l = g^{-1}(c, x)$, then we have:

$$\begin{aligned}\Pr(R^c|B^c, b^m) &= (1 - \Pr(B^{g(c^l-1,x)}|b^m)) \\ &\quad \Pr(R^{g(c^l-1,x)}|B^{g(c^l-1,x)}, b^m) \\ &= \prod_{l=1}^{c^l-1} (1 - \Pr(B^{g(l,x)}|b^m)) \\ &= \prod_{l=1}^{c^l-1} (1 - \widehat{m}^{g(l,x)})\end{aligned} \tag{4}$$

IV. Confidence-Augmented Grid Map

In this section, we derive the recursive mapping algorithm described in Eq. (1). We derive the mapping algorithm τ that can reason not only about the occupancy at each cell, but also about the confidence level of this value.

To compute the belief of the i-th voxel, denoted by $b_k^{m^i} = p(m^i|z_{0:k}, x_{0:k})$, we bring the cause variables into formulation. As shown in [4], we can compute the belief as follows:

$$p(m^i|z_{0:k}, x_{0:k}) = (\alpha^i m^i + \beta^i) p(m^i|z_{0:k-1}, x_{0:k-1}) \tag{5}$$

where

$$\begin{aligned}\alpha^i = &\sum_{c_k^l=1}^{i^l-1} p(c_k|b_{k-1}^m, z_k, x_k) \\ &+ (1 - \widehat{m}^i)^{-1} \sum_{c_k^l=i^l+1}^{|\mathbb{C}(x)|} p(c_k|b_{k-1}^m, z_k, x_k)\end{aligned} \tag{6}$$

$$\begin{aligned}\beta^i = &(\widehat{m}^i)^{-1} p(c_k|b_{k-1}^m, z_k, x_k) \\ &- (1 - \widehat{m}^i)^{-1} \sum_{c_k^l=i^l+1}^{|\mathbb{C}(x)|} p(c_k|b_{k-1}^m, z_k, x_k)\end{aligned} \tag{7}$$

In a more compact form, we can rewrite Eq. (5) as:

$$b_{k+1}^{m^i} = \tau^i(b_k^m, z_{k+1}, x_{k+1}). \tag{8}$$

Sensor cause model: The proposed machinery gives rise to the term
$p(c_k|z_{0:k}, x_{0:k}) = p(c_k|b_{k-1}^m, z_k, x_k)$, which is referred to as *Sensor Cause Model* (SCM) in this paper. As opposed to the inverse sensor model in traditional mapping that needs to be hand-engineered, the SCM can be derived from the forward sensor model in a principled way as follows. η' is the normalization constant.

$$\begin{aligned} &p(c_k|z_{0:k}, x_{0:k}) = p(c_k|b_{k-1}^m, z_k, x_k) \\ &= \frac{p(z_k|c_k, x_k)p(c_k|b_{k-1}^m, x_k)}{p(z_k|b_{k-1}^m, x_k)} \\ &= \eta' p(z_k|c_k, x_k)p(c_k|b_{k-1}^m, x_k) \\ &= \eta' p(z_k|c_k, x_k)\widehat{m}_{k-1}^{c_k}\prod_{j=1}^{c_k^l-1}(1-\widehat{m}_{k-1}^{g(j,x)}), \forall c_k \in \mathbb{C}(x_k) \end{aligned} \tag{9}$$

The complete mapping algorithm is recapped in Alg. 1. For more details on these results, see [4].

Algorithm 1: Confidence-rich grid mapping

input : Current map belief b_k, Current observation z_k, current robot measurement ray x_k
output : Updated map belief b_{k+1}
Procedure : $b_{k+1} =$`Update`(b_k, z_k, x_k)

Find $\mathbf{V_{ray}}$ voxels on the ray x_k;
Compute SCM using Eq. (9);
foreach $v_i \in \mathbf{V_{ray}}$ **do**
    Compute α^i, β^i (Eqs. (6), (7));
    Update voxel belief (Eq. (5));
return b_{k+1};

V. Results

In this section, we demonstrate the performance of the proposed method and compare it with commonly used mapping methods. More details of the simulation setup can be found in [4].

sensor std dev	$0.25s$	$0.5s$	$1s$	$2s$	$3s$
log-odds MAE	0.346	0.348	0.351	0.358	0.376
GPOM MAE	0.421	0.468	0.468	0.468	0.468
CRM MAE	**0.289**	**0.299**	**0.323**	**0.355**	**0.365**
log-odds I_c	17.340	18.609	17.665	18.862	19.243
GPOM I_c	10.167	10.172	12.964	10.212	10.490
CRM I_c	**0.056**	**2.040**	**6.100**	**5.418**	**4.575**

Fig. 2. Mean Absolute Error (MAE) and inconsistency I_c with $\gamma = 1$ (Eq. 10) of log-odds mapping, GPOM and the proposed method under different sensor model noise levels (right), where s is the voxel size (0.06m).

For the sensing system, we simulate a ranging sensor with 60 omnidirectional depth sensors spanning over a field-of-view of 360° and reaching up to a range of 1m. Measurements are corrupted by zero-mean Gaussian noise with 0.06m std. deviation, i.e. one voxel length.

The occupancy maps resulting from the log-odds mapping, GPOM and the proposed method are shown in the upper row of Fig. 3, respectively.

First, we study the sensitivity of our method to different sensor noises. The top rows of Table 2 show the map mean absolute error (MAE) over time for different sensor noise std deviations. The MAE is averaged over all the voxels that were updated throughout the mapping process to best show the improvement of the affected parts of the map. For the same noise intensity, the proposed CRM method shows a smaller error compared to the log-odds and Gaussian Processes method.

It is worth emphasizing that reducing the map error is only an ancillary benefit of our method. The main objective of CRM is to provide a consistent confidence measure. This consistency is particularly important for planning purposes. A planner might be able to handle large errors as long as the filter indicates that the estimates are unreliable (e.g. via their variance). However, if the filter returns a wrong estimate with low variance (i.e. it is confident that its estimate is correct, while it is not), then the planner is prone to yield unsafe results.

To quantify the inconsistency between the error and reported variances, we utilize the following measure:

$$I_c = \sum_c \mathrm{ramp}(|e_c| - \gamma\sigma_c) \tag{10}$$

where, e_c and σ_c denote the estimation error and the std. deviation of voxel c, respectively. γ decides what level of error is acceptable. The ramp function $\mathrm{ramp}(x) := \max(0, x)$ ensures that only inconsistent voxels (with respect to $\gamma\sigma$) contribute to the summation. Accordingly, I_c indicates how much of the error signal is out of bound (i.e., how unreliable the estimate is) over the whole map. As can be seen in the middle row of Fig. 3, where voxels of high inconsistency appear brighter, the log-odds based approach tends to be overly confident in false estimates, compared to the other methods. GP-based maps appear less clearer compared to traditional occupancy grids with the advantage of a higher consistency. However, as can be seen in the bottom right map of Fig. 3, smaller obstacles are not mapped accurately and pose dangerous areas of inconsistency.

Fig. 4(a) compares the inconsistency measure (Eq. 10) over different γ thresholds for the presented mapping algorithms. The log-odds approach overestimates the confidence significantly more while GPOM and our method yield much safer estimates.

While GPOM and the proposed approach yield similar results in the ramp inconsistency comparison, it should be noted that GPOM is computationally much more expensive than occupancy grids [5]. Gaussian Processes rely on the full set of measurement samples in order to predict the map occupancy.

As an alternative measure of consistency, we compute the Pearson correlation coefficient between the error and std. deviation (see Fig. 4(b)), which indicates that the std. deviation generated by the proposed method is highly correlated with the error, hence, it reliably describes the error's behavior.

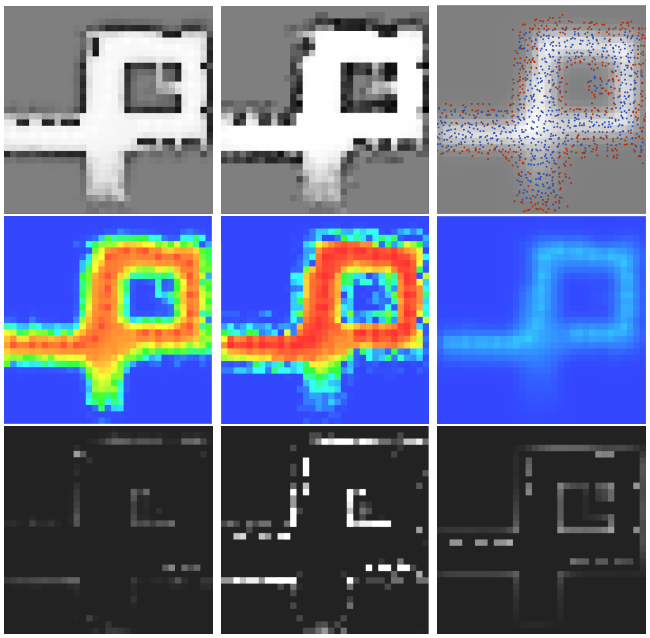

Fig. 3. Mapping results: CRM (left column), Log Odds (center column), GPOM (right column). Mean occupancies (top row), estimated standard deviation (middle row), inconsistencies $|e_c| > \sigma_c$ for absolute voxel error $|e_c|$ and std σ_c (lower row). The mean occupancy GP map (top right) shows the GP sample set where blue dots represent samples of free space and red dots represent occupied space. Measurement rays are discretized on the voxel grid for GPOM, and appear with a random offset only for visualization purposes.

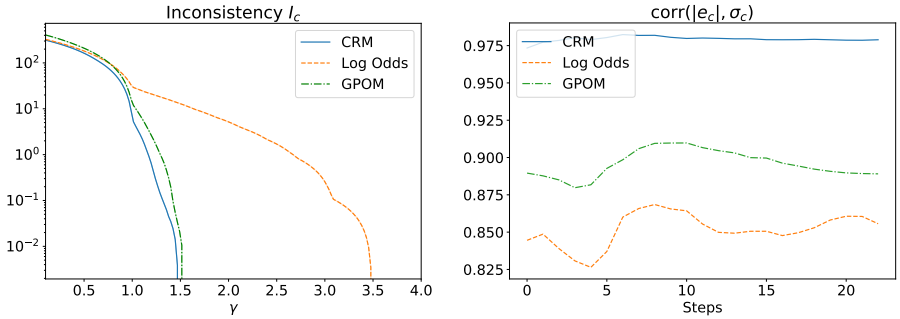

Fig. 4. Left: evaluation of the ramp measure for inconsistency (Eq. 10) over different threshold values γ. Right: Pearson correlation coefficient between the reported std deviation and the absolute error over all voxels at a given step.

VI. Conclusion

This paper proposes an alternative algorithm for grid mapping, featuring a richer representation of the voxel occupancy. The method runs online as measurements are received and it enables mapping environments where voxels might be partially occupied. The results show that the mapping accuracy is approximately 10% better than the ISM-based method and 30% better than Gaussian Processes maps, and more importantly, according to the proposed consistency measure, the confidence values are up to two orders of magnitude more reliable.

References

[1] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems," in *ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, vol. 2, 2010.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005.

[3] S. Thrun *et al.*, "Robotic mapping: A survey," *Exploring artificial intelligence in the new millennium*, vol. 1, pp. 1–35, 2002.

[4] Ali-akbar Agha-mohammadi, E. Heiden, K. Hausman, and G. Sukhatme, "Confidence-rich grid mapping: Extended version," *Technical Report*, 2017. [Online]. Available: http://people.lids.mit.edu/aliagha/Web/pubpdfs/CRM.pdf

[5] S. T. OCallaghan and F. T. Ramos, "Gaussian process occupancy maps," *The International Journal of Robotics Research*, vol. 31, no. 1, pp. 42–62, 2012.