Indoor UAV Height Estimation with Multiple Model-Detecting Particle Filters

Hechuan Wang, Xiaokun Zhao, Mónica F. Bugallo
Department of Electrical and Computer Engineering
Stony Brook University, Stony Brook, NY 11794
Email: {hechuan.wang, xiaokun.zhao, monica.bugallo}@stonybrook.edu,

***Abstract*—The precision of indoor localization, especially height estimation, is critical to unmanned aerial vehicle (UAV) navigation to avoid crashes because indoor environments are narrow and complex. The lack of satellite-based navigation signals makes this task very challenging. Moreover, objects in indoor environments could be randomly shaped and in motion, making map-based navigation unreliable. There exist solutions utilizing advanced sensor arrays such as laser scanners or multiple cameras, but the UAVs' weight load and computational resources are limited. In this paper, we propose a filtering-based method that allows for estimation of the height of the UAV by stand-alone range finders. Model-detecting particle filters are used to detect changes in objects while estimating the height of the UAV simultaneously. Multiple filters are utilized to speed up the computation. Numerical experiments show that the proposed method is more accurate than other methods.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) have recently gained increasing popularity in public and civil applications, including search and rescue missions [1], package delivery [2], precision agriculture [3], and topographic mapping [4]. With the wide range of applications of UAVs, accurate localization is becoming a critical requirement for navigation and collision avoidance. This requirement is relatively easy to meet in outdoor environments, where the Global Navigation Satellite Systems (GNSS) signals are accessible to the UAVs. However, in indoor environments, those signals are not available. Thus, indoor UAVs rely on measurements obtained by onboard sensors, such as cameras or laser scanners, which are heavy and need a lot of computational resources. Furthermore, the presence of irregular obstacles makes indoor UAV localization even more challenging.

Due to the existence of obstacles above and below UAVs in indoor environments, altitude estimation is critical for reliable UAV operation in indoor settings [5]. There have been various methods proposed in the literature to solve this problem. In [6], a sensor fusion algorithm is proposed to estimate the UAV altitude by combining the measurements from an ultrasonic proximity finder and an atmospheric pressure sensor with a weighting function after filtering the measurements. Kalman filters (KFs) and extended Kalman filters (EKFs) are also popular for data fusion based on multiple sensors. In [7], KF is chosen to fuse the signals from sonar and an accelerometer. The method in [8] estimates the UAV's 3D state by collecting data from a variety of sensors and fusing the measurements with an EKF. Besides traditional sensors, computer vision can be used to estimate the UAV altitude with a single onboard camera and machine learning approach [7], or two cameras that build stereoscopic system [9]. In another work [10], the altitude is estimated with only two IR sensors by employing multiple model adaptive estimation (MMAE), which gives the average of a bank of KFs. However, the shape of the obstacles is not taken into consideration in that solution.

In this paper, we propose a novel approach based on Bayesian filtering for indoor UAV altitude estimation. This method utilizes particle filtering with Bayesian model averaging to detect the shape-changing of the obstacles on the ceiling and/or the floor. At the same time, multiple filters are used to estimate the height in parallel, which reduce the dimension of the state space and the computational cost.

The paper is organized as follows. In Section II, we provide the mathematical formulation of the problem. Section III presents the details of the proposed methods. The results of numerical experiments are given in Section IV. And we draw the conclusion in Section V.

II. Mathematical Problem Formulation

In this work, the UAV is operated in an indoor environment shown in Fig. 1, where the ceiling and the floor are both flat and level. There are unknown obstacles on the ceiling and/or the floor, which can be of any shape. The UAV has two IR sensors onboard: one pointing straight to the ceiling and the other one pointing straight to the floor. We assume that the two sensors are at the same level, hence there is no vertical distance between the two sensors.

Our goal is to estimate the altitude of the UAV by obtaining only the two measurements from the IR sensors. The problem is challenging because of the noise that the sensors can bring, as well as the lack of knowledge in the unpredictable height of the obstacles. We formulate the problem as a state-space model (SSM), where the dynamics of the UAV and the vertical dimension of obstacles can be considered as latent states, and the measurements from the two sensors are observations, which are shown in Fig. 1.

The transition model can be built as the state vector $\mathbf{x}_t$. We define $x_{1,t}$, $x_{2,t}$, and $x_{3,t}$ as the height, vertical speed and vertical acceleration of the UAV at time t respectively.

ISBN: 978-1-6654-6798-8

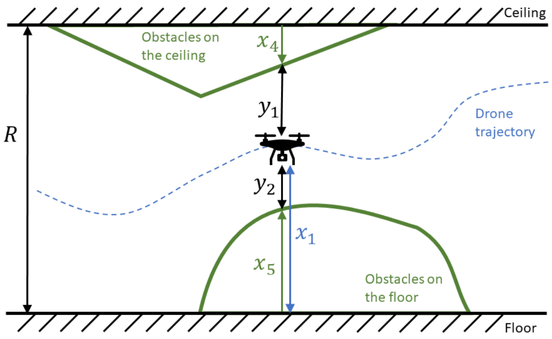

Fig. 1: State and measurement definition

According to the kinematics of UAV, their transition model can be given by:

$$x_{1,t} = x_{1,t-1} + T_s x_{2,t-1} + 0.5 T_s^2 x_{3,t-1}, \tag{1a}$$
$$x_{2,t} = x_{2,t-1} + T_s x_{3,t-1}, \tag{1b}$$
$$x_{3,t} = a_t,\ a_t \sim \mathcal{N}(0, \sigma_a^2), \tag{1c}$$

where T_s is the sampling period. Because the transition model of acceleration $x_{3,t}$ depends on many unknown factors, we simplify the model as a Gaussian distribution with variance σ_a^2.

The states $x_{4,t}$ and $x_{5,t}$ are defined as the height of the obstacles on the ceiling and the floor, respectively. They are unknown and not easily represented by a fixed model, and we will address this issue in Section III.

The observation model is given by the following equations:

$$y_{1,t} = R - x_{4,t} - x_{1,t} + v_{1,t}, \qquad v_{1,t} \sim \mathcal{N}(0, \sigma_o^2), \tag{2a}$$
$$y_{2,t} = x_{1,t} - x_{5,t} + v_{2,t}, \qquad v_{2,t} \sim \mathcal{N}(0, \sigma_o^2), \tag{2b}$$

where the height of the ceiling R is known.

III. Proposed Method

A. Proposed transition models for the objects

The transition model of the obstacle height states $x_{4,t}$ and $x_{5,t}$ depends on their shapes. In most cases, the obstacles have positive heights. However, in [10], Gaussian distributions, which have half of their support in the negative domain, are used to model the obstacle heights. In this paper, we propose some transition models that represent the states more appropriately:

$$x_{4,t} = 0, \tag{3}$$
$$x_{4,t} \sim \mathcal{U}(a, b), \tag{4}$$
$$x_{4,t} \sim \mathcal{E}xp(\mu_e), \tag{5}$$
$$x_{4,t} \sim \frac{1}{M} \sum_{m=1}^{M} \mathcal{N}(x_{4,t} | h_m, b^2), \tag{6}$$

where (3) represents no obstacles; (4) represents objects of uniformly distributed heights; (5) represents exponentially distributed heights which implies most of the objects are low in height; and (6) is the kernel density estimated distribution given recorded obstacle heights $h_{1:M}$ and the bandwidth of the smoothing kernel b. The transition of state $x_{5,t}$ can also be modeled in the same way.

B. Model-detecting particle filter (MDPF)

We propose a model-detecting particle filtering method that estimates the model belief along with the states.

Suppose that there are N SSM models for a time sequence. Each model can be expressed as follows:

$$\begin{aligned} \mathbf{x}_t &\sim f_n(\mathbf{x}_t | \mathbf{x}_{t-1}), \\ \mathbf{y}_t &\sim g_n(\mathbf{y}_t | \mathbf{x}_t), \end{aligned} \tag{7}$$

where $n \in 1:N$ is the index of the nth model.

At time step t, we assume equally weighted samples $\tilde{\mathbf{x}}_{n,t}^{(1:M_n)}$ are given, where n is the model index, and M_n is the number of samples dedicated to the nth model.

The samples are proposed by each model n individually, i.e.,

$$\mathbf{x}_{n,t}^{(m)} \sim f_n(\mathbf{x}_{n,t} | \tilde{\mathbf{x}}_{n,t-1}^{(m)}),\ m = 1 : M_n. \tag{8}$$

The non-normalized weights of the samples can be calculated by the observation distribution $g_n(\cdot)$, i.e.,

$$\tilde{w}_{n,t}^{(m)} = g_n(\mathbf{y}_t | \mathbf{x}_{n,t}^{(m)}). \tag{9}$$

The proposing and weighting steps are intrinsically parallel and can take advantage of modern computational devices such as multi-thread CPU or GPU. The weights are normalized within each model, i.e.,

$$w_{n,t}^{(m)} = \frac{\tilde{w}_{n,t}^{(m)}}{\sum_{i=1}^{M_n} \tilde{w}_{n,t}^{(i)}}. \tag{10}$$

The nth model's likelihood $L_{n,t}$ can be obtained by the mean of the non-normalized weights

$$L_{n,t} = \frac{1}{M_n} \sum_{m=1}^{M_n} \tilde{w}_{n,t}^{(m)}, \tag{11}$$

and the belief of the nth model, $P_{n,t}$, is updated in a Bayesian way, that is,

$$\tilde{P}_{n,t} = L_{n,t} P_{n,t-1}^{\rho}, \tag{12a}$$
$$P_{n,t} = \frac{\tilde{P}_{n,t}}{\sum_{n=1}^{N} \tilde{P}_{n,t}}, \tag{12b}$$

where ρ is the forgetting parameter. The value of ρ depends on the setting of the problem. If the model is known and fixed through time, we set $\rho = 1$ which means no forgetting is done. If the models are alternating and we want to detect the switching of models, we need to set $0 \leq \rho < 1$. A smaller ρ value represents sharper transition between the models. In the extreme case that $\rho = 0$, no model beliefs are remembered and the models are assumed to be changing suddenly without transition. In the experiments described in Section IV, we found that the forgetting parameter ρ will affect the smoothness of the belief curve: the curve will not be as smooth when ρ is close to 0, and will have a large latency when ρ is close to 1.

When estimating states, we first do the local estimation in each model n,

$$\hat{\mathbf{x}}_{n,t} = \sum_{m=1}^{M_n} w_{n,t}^{(m)} \mathbf{x}_{n,t}^{(m)}, \tag{13}$$

and then, the combined estimation can be acquired by Bayesian model averaging

$$\hat{\mathbf{x}}_t = \sum_{n=1}^{N} P_{n,t} \hat{\mathbf{x}}_{n,t}. \tag{14}$$

The samples from all models are aggregated by

$$\left(\mathbf{x}_t^{(1:M^*)}, w_t^{(1:M^*)}\right) = \bigcup_{n=1}^{N} \left(\mathbf{x}_{n,t}^{(1:M_n)}, P_{n,t} w_{n,t}^{(1:M_n)}\right), \tag{15}$$

where $M^* = \sum_{n=1}^{N} M_n$. Theoretically, these samples can be directly used by all models in the next iteration. However, that will cause the number of samples to grow exponentially. Instead, we perform downsampling to keep a stationary number of samples, i.e.,

$$\tilde{\mathbf{x}}_{n,t}^{(1:M_n)} \overset{iid}{\sim} \mathcal{C}\left(\mathbf{x}_t^{(1:M^*)}, w_t^{(1:M^*)}\right), \ n = 1:N, \tag{16}$$

where $\mathcal{C}(\text{items}, \text{weights})$ denotes a categorical distribution.

The sample size of each model can be constant and is the same for all the models, i.e.,

$$M_n = M, \ n = 1:N. \tag{17}$$

However, because some models may not represent the observations well, assigning a large number of samples to these models is a waste of computational resources. As the model beliefs represent the reliability of the models, we can assign different numbers of samples to different models accordingly.

The first strategy is that the number of samples is proportional to the model belief

$$\begin{aligned} M_{n,t} &= \lceil M^* P_{n,t-1} \rceil, \\ M_t^* &= \sum_{n=1}^{N} M_{n,t}, \end{aligned} \tag{18}$$

where $\lceil \cdot \rceil$ denotes the ceiling operation; because of this operation, the total number of samples M_t^* is not a constant number but a changing integer around a fixed value M^*. The problem of this strategy is that the number of samples dedicated to some models will decrease to 0, and therefore will not be able to recover [11]. This phenomenon is called “degeneracy”. Therefore, we adjust the previous strategy to keep a minimum number of samples M' for each model,

$$\begin{aligned} M_{n,t} &= M' + \lceil (M^* - NM') P_{n,t-1} \rceil, \\ M_t^* &= \sum_{n=1}^{N} M_{n,t}. \end{aligned} \tag{19}$$

The complete algorithm is summarized in Algorithm 1, and its block diagram is shown in Fig. 2.

Algorithm 1: One iteration of MDPF in time t

```
Given:
    $\tilde{\mathbf{x}}_{n,t}^{(1:M_n)}, \ n = 1:N$
Prediction phase:
    For $n = 1:N$
        Propose the samples by (8).
    end For
Update phase:
    For $n = 1:N$
        Weight the samples of filter $n$ by (9), (10).
        Calculate model likelihood by (11).
    end For
    Update the model probabilities by (12).
    Resample by (16).
Result:
    Combined estimation (14)
```

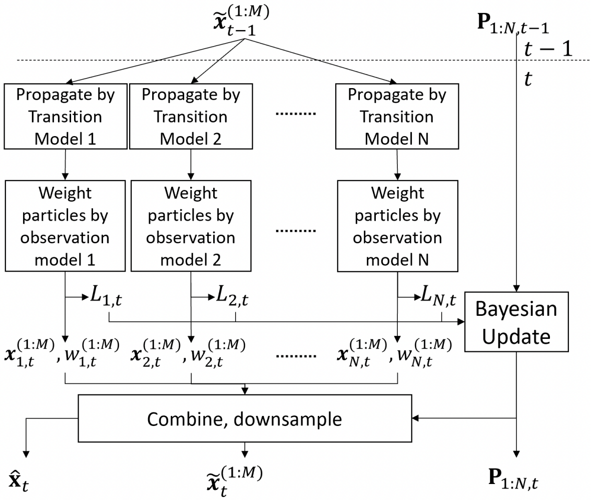
$\tilde{\boldsymbol{x}}_{t-1}^{(1:M)}$
$\mathbf{P}_{1:N,t-1}$
$t-1$
t
Propagate by Transition Model 1
Propagate by Transition Model 2
....
Propagate by Transition Model N
Weight particles by observation model 1
Weight particles by observation model 2
....
Weight particles by observation model N
$L_{1,t}$
$L_{2,t}$
$L_{N,t}$
Bayesian Update
$\boldsymbol{x}_{1,t}^{(1:M)}, w_{1,t}^{(1:M)}$
$\boldsymbol{x}_{2,t}^{(1:M)}, w_{2,t}^{(1:M)}$
....
$\boldsymbol{x}_{N,t}^{(1:M)}, w_{N,t}^{(1:M)}$
Combine, downsample
$\hat{\mathbf{x}}_t$
$\tilde{\boldsymbol{x}}_t^{(1:M)}$
$\mathbf{P}_{1:N,t}$

Fig. 2: Block diagram of the proposed MDPF algorithm.

C. Multiple filters implementation for UAV height estimation

In the UAV transition models, the states can be divided into three independent groups: drone states, floor obstacle height, and ceiling obstacle height. The transition of the states in each group does not depend on states from other groups. This allows us to divide the state space into smaller state groups and use multiple cooperative particle filters of smaller dimensions to do the estimation. Because particle filtering is a Monte Carlo based method, reducing the dimension of state space will help increase the accuracy with less samples, and thus reduce the computational cost. In this paper, we propose to use the concept of multiple particle filter (MPF) based on [12]. We add one step that shares the estimation results across filters and then re-estimates. This iterative implementation of MPF can improve the estimation accuracy. The algorithm is summarized in Algorithm 2, and the block diagram of the algorithm is shown in Fig. 3.

Algorithm 2: One iteration of MPF in time t

```
Given:
    $\tilde{\mathbf{x}}_{[k],t}^{(1:M_{t-1}^*)}, \ k = 1:K$    (20)
Prediction phase:
    For $k = 1:K$
        Run prediction phase of state group $k$'s filter.
        Calculate the public state $\check{\mathbf{x}}_{[k],t}$ by
            $\check{\mathbf{x}}_{[k],t} = \hat{\mathbf{x}}_{[k],t|t-1} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{x}_{[k],t}^{(m)}.$    (21)
    end For
Update phase:
    For $i = 1:I$
        For $k = 1:K$
            Run update phase of state group $k$'s filter using
            observation model
                $q(\mathbf{y}_t|\mathbf{x}_{[k],t}, \check{\mathbf{x}}_{[-k],t}).$    (22)
            Update the public states by
                $\check{\mathbf{x}}_t \leftarrow \hat{\mathbf{x}}_t = \cup_{k=1}^{K}\hat{\mathbf{x}}_{[k],t}.$    (23)
        end For
    end For
Result:
    Combined estimation $\hat{\mathbf{x}}_t$
```

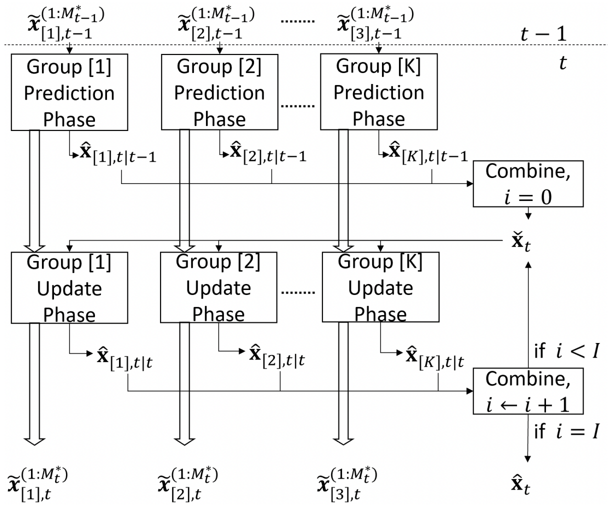
$\tilde{\mathbf{x}}_{[1],t-1}^{(1:M_{t-1}^*)}$
$\tilde{\mathbf{x}}_{[2],t-1}^{(1:M_{t-1}^*)}$
....
$\tilde{\mathbf{x}}_{[3],t-1}^{(1:M_{t-1}^*)}$
$t-1$
t
Group [1] Prediction Phase
Group [2] Prediction Phase
....
Group [K] Prediction Phase
$\hat{\mathbf{x}}_{[1],t|t-1}$
$\hat{\mathbf{x}}_{[2],t|t-1}$
$\hat{\mathbf{x}}_{[K],t|t-1}$
Combine, $i=0$
$\check{\mathbf{x}}_t$
Group [1] Update Phase
Group [2] Update Phase
....
Group [K] Update Phase
if $i<I$
$\hat{\mathbf{x}}_{[1],t|t}$
$\hat{\mathbf{x}}_{[2],t|t}$
$\hat{\mathbf{x}}_{[K],t|t}$
Combine, $i \leftarrow i+1$
if $i=I$
$\tilde{\mathbf{x}}_{[1],t}^{(1:M_t^*)}$
$\tilde{\mathbf{x}}_{[2],t}^{(1:M_t^*)}$
$\tilde{\mathbf{x}}_{[3],t}^{(1:M_t^*)}$
$\hat{\mathbf{x}}_t$

Fig. 3: Block diagram of the proposed MDPF algorithm.

In (20), K is the total number of groups. Subscript $[k]$ denotes the states in the kth group. Similarly, subscript $[-k]$ in (22) denotes all states that are not in the kth group. I is the number of iterations for using the estimated result as public states. In practice, a very small number of I, such as $I = 2$, will give a good result.

IV. Numerical Experiments

In our simulation, we flew the drone in an indoor environment with constant horizontal speed. The vertical movement followed the transition model (1) with parameter $\sigma_a^2 = 0.001(m^2/s)^2$. The ceiling height was $R = 3$ meters. Obstacles of different shapes were located on the ceiling and the floor. All the obstacles were of positive heights. Observations were acquired according to (2) with sensor variance parameter $\sigma_o^2 = 0.001m^2$. The numerical experiment duration was $T = 40$ seconds and the sampling period was $T_s = 0.02$ seconds. One realization of the simulation is shown in Fig. 4.

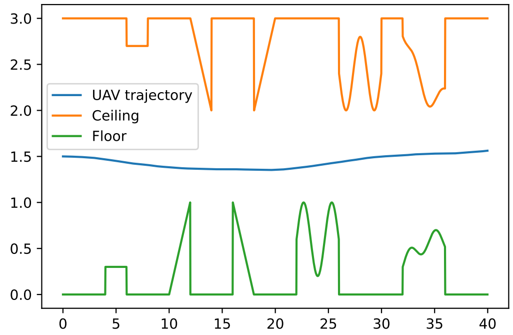

Fig. 4: Data of one simulation realization, where UAV trajectory is $\mathbf{x}_1$, ceiling is $R - \mathbf{x}_4$ and Floor is $\mathbf{x}_5$

We used the proposed MPFs to estimate the states of the UAV system. The states were divided into three groups: $\mathbf{x}_{[1],t} \triangleq \mathbf{x}_{1:3,t}$, $\mathbf{x}_{[2],t} \triangleq \mathbf{x}_{4,t}$, and $\mathbf{x}_{[3],t} \triangleq \mathbf{x}_{5,t}$. The first group of states was estimated by a PF that used the UAV transition model (1). For the second and third groups, we used the proposed MDPF with $M_n = 1000$ samples for each model and forgetting parameter $\rho = 0.6$. Here, the value of ρ is hand selected. To test the performance of different transition models, we did the experiment with three specific filter implementations, which are called F2∼4, with different model combinations. Each filter implementation used two proposed obstacle transition models, and the transition models for $\mathbf{x}_{[2]}$ and $\mathbf{x}_{[3]}$ were the same. Please see Table I for detailed model assignments and model parameters.

We tested the proposed methods by numerical experiments, and compared the results with the MMAE method proposed in [10], which is called F1 in the presented simulation results. The performance was measured by root mean square error (RMSE) of the UAV's height estimation defined as:

$$\frac{1}{S}\sum_{s=1}^{S}\frac{1}{T}\sum_{t=0}^{T}(x_{1,t} - \hat{x}_{1,t|t})^2, \tag{24}$$

where S is the number of simulations. We also recorded the RMSE of the obstacles' height estimate for reference:

$$\frac{1}{S}\sum_{s=1}^{S}\frac{1}{T}\sum_{t=0}^{T}\frac{1}{2}\left((x_{4,t} - \hat{x}_{4,t|t})^2 + (x_{5,t} - \hat{x}_{5,t|t})^2\right). \tag{25}$$

State group / Filter name	$\mathbf{x}_{[1],t} \triangleq \mathbf{x}_{1:3,t}$	$\mathbf{x}_{[2],t} \triangleq \mathbf{x}_{4,t}$	$\mathbf{x}_{[3],t} \triangleq \mathbf{x}_{5,t}$
F2	Drone transition (1) $\sigma_a^2 = 0.001$	No obstacles (3) & Uniform (4) with $a = 0,\ b = R$	
F3		No obstacles (3) & Exponential (5) with $\mu_e = 0.5$	
F4		No obstacles (3) & KDE (6) with $b = 0.001$	

TABLE I: State groups of MPFs and transition model(s) for each PF and MDPF in numerical experiments.

Filter type	RMSE / Filter	Drone Height RMSE	Obstacle height RMSE	Over all RMSE
MMAE	F1: Kalman	0.0132	0.0231	0.0206
Proposed	F2: No \Uniform	0.0068	0.0200	0.0168
	F3: No \Exponential	0.0061	0.0197	0.0165
	F4: No \KDE	0.0055	0.0185	0.0154

TABLE II: RMSE of estimations by different methods averaged over 100 simulations (No is the abbreviation of no obstacles).

The Experiment results are shown in Table II.

In the results, we see that the proposed method with any model was superior to MMAE. We also see that the performance from F4 was better than F3, and F3 was better than F2. That was because the uniform transition model (4) we used in F2 was less descriptive than the exponential transition model (5) we used in F3. The later assumes that most of the obstacle heights were not very large. The KDE transition model (6) had the full knowledge of the distribution of obstacle heights, and thus had the best performance. From this comparison, we confirmed that the estimation result can be improved if more information is given.

We can also infer the existence of obstacles by the model beliefs. For example, in group $\mathbf{x}_{[3],t}$ of F2, the two transition models were (3) and (4), which represented "obstacle" and "bare floor" respectively. If (4) had dominating model belief, we can infer that an obstacle existed at that time instant. The model beliefs of F2 are shown in Fig. 5, which suggests that the proposed methods can infer the existence of obstacles well.

V. Conclusion

In this paper, we propose a sequential estimation method for indoor UAV height estimation. Multiple models are used in particle filters to detect different situations of indoor obstacles on the floor and ceiling. The implementation of multiple particle filters reduces the dimension of Monte Carlo estimation and thus saves computation resources and improves performance. Simulation results show that the proposed methods can estimate the UAV height with higher precision than previous methods.

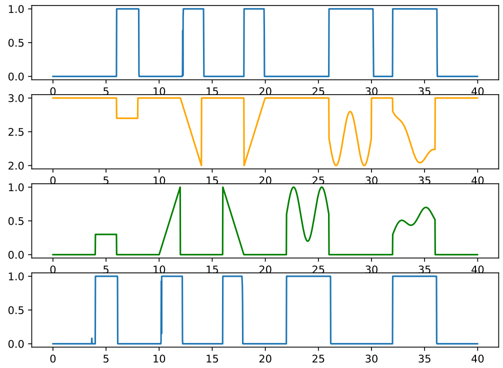

Fig. 5: Beliefs of obstacles' existence in one simulation of F2, where the first row is the belief that obstacles exist on the ceiling and the last row is the belief that obstacles exist on the floor. The second and the third rows are $R - \mathbf{x}_4$ and $\mathbf{x}_5$ respectively.

References

[1] P. Doherty and P. Rudol, "A UAV search and rescue scenario with human body detection and geolocalization," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2007, pp. 1–13.

[2] Y. Li, W. Yang, and B. Huang, "Impact of UAV delivery on sustainability and costs under traffic restrictions," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[3] T. Adão, J. Hruška, L. Pádua, J. Bessa, E. Peres, R. Morais, and J. J. Sousa, "Hyperspectral imaging: A review on UAV-based sensors, data processing and applications for agriculture and forestry," *Remote sensing*, vol. 9, no. 11, p. 1110, 2017.

[4] S. M. Azmi, B. Ahmad, and A. Ahmad, "Accuracy assessment of topographic mapping using UAV image integrated with satellite images," in *IOP Conference Series: Earth and Environmental Science*, vol. 18, no. 1. IOP Publishing, 2014, p. 012015.

[5] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *2009 IEEE international conference on Robotics and Automation*. IEEE, 2009, pp. 2878–2883.

[6] G. Szafranski, R. Czyba, W. Janusz, and W. Blotnicki, "Altitude estimation for the UAV's applications based on sensors fusion algorithm," in *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2013, pp. 508–515.

[7] A. Cherian, J. Andersh, V. Morellas, N. Papanikolopoulos, and B. Mettler, "Autonomous altitude estimation of a UAV using a single onboard camera," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 3900–3905.

[8] H. Du, W. Wang, C. Xu, R. Xiao, and C. Sun, "Real-time onboard 3d state estimation of an unmanned aerial vehicle in multi-environments using multi-sensor data fusion," *Sensors*, vol. 20, no. 3, p. 919, 2020.

[9] D. Eynard, P. Vasseur, C. Demonceaux, and V. Frémont, "UAV altitude estimation by mixed stereoscopic vision," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 646–651.

[10] L. Yang, H. Wang, Y. El-Laham, J. I. L. Fonte, D. T. Pérez, and M. F. Bugallo, "Indoor altitude estimation of unmanned aerial vehicles using a bank of Kalman filters," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 5455–5459.

[11] I. Urteaga, M. F. Bugallo, and P. M. Djurić, "Sequential Monte Carlo methods under model uncertainty," in *2016 IEEE Statistical Signal Processing Workshop (SSP)*. IEEE, 2016, pp. 1–5.

[12] P. M. Djuric, T. Lu, and M. F. Bugallo, "Multiple particle filtering," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 3. IEEE, 2007, pp. III–1181.