Rao-Blackwellized Particle Filtering in POMDP for Search-and-Rescue Autonomous Navigation*

Jiho Lee[1]

***Abstract*—Partially Observable Markov Decision Processes (POMDPs) provide a structured approach for decision-making under uncertainty, but their application requires efficient belief updates. Sequential Importance Resampling Particle Filters (SIRPF) are commonly employed as belief updaters in POMDPs but face challenges such as particle impoverishment and high computational costs as the system's effective dimensionality grows. This study explores the use of the hybrid computational strategy of Rao-Blackwellized Particle Filters (RBPF) as an alternative to reduce the dimensionality of the space that particles need to explore. We compare the performance of SIRPF and RBPF in a Unmanned Aerial Vehicles (UAV)-based search and rescue (SAR) scenario by incorporating Conformal Prediction to quantify the uncertainty of the system's belief estimates. Our results show that RBPF achieves lower variance and maintains stable performance over time, even with a smaller number of particles, while SIRPF requires a substantially larger number of particles for comparable accuracy. This study highlights the potential of RBPF as a belief updater in POMDP models, particularly when the system's effective dimensionality is high and computational efficiency is critical.**

***Index Terms*—Rao-Blackwellized Particle Filter, Partially Observable Markov Decision Processes, Conformal Prediction, Autonomous Systems, Search-and-Rescue**

I. Introduction

Unmanned Aerial Vehicles (UAVs) are transforming public safety by offering unique aerial viewpoints and quick support to emergency response teams, notably in roles such as search and rescue (SAR) [1]. As the advantages of UAV autonomy in supporting search operations are becoming increasingly evident [2], the need for computationally efficient models has become vital.

Partially Observable Markov Decision Processes (POMDPs) are well-suited for modeling decision-making under uncertainty and have been widely applied to various domains, including SAR missions with drones [3]. POMDPs present a structured framework for depicting an agent's beliefs about its environment, facilitating informed decisions in uncertain scenarios [9]. Particle filters are particularly effective for dynamically modeling these beliefs. For example, sequential importance resampling particle filters (SIRPF), also known as bootstrap particle filters (BPF), are widely used as belief updaters in POMDP models. However, while SIRPF is straightforward and commonly employed, the choice of transition density as the importance sampling probability density function can have significant implications for the filter's performance. Also, regular resampling is necessary to avoid particle impoverishment, and an exponential increase in the number of particles is required to enhance performance as the system's effective dimension grows, leading to high computational costs [4] [5].

Rao-Blackwellized Particle Filters (RBPF) offer a potential solution to some of the limitations of SIRPF. By analytically solving for the linear/Gaussian parts of the system, RBPF can reduce the dimensionality of the space that particles need to explore and lower the overall number of particles required. This filtering can reduce computational costs without compromising accuracy. However, while RBPF has been extensively studied in the Simultaneous Localization and Mapping (SLAM) domain [10] [11], its application in the POMDP context has not been explored.

To validate that RBPF maintains or enhances predictive accuracy with fewer particles, it is essential to quantify the uncertainty of the system's belief estimates. Conformal Prediction (CP), known for providing statistically valid prediction bounds with distribution-free and non-asymptotic coverage assurances [6], can address this challenge by offering reliable confidence intervals for the belief estimates of the POMDP model.

In summary, this project aims to explore utilizing the hybrid nature of RBPF for belief updaters in POMDP models. The potential reduction in computational costs will be crucial in real-time decision-making scenarios and can create a more robust framework optimized for online planning. Furthermore, the variance of the belief estimates will be quantified with CP to validate that RBPF does not indeed compromise the accuracy of the predictions with a smaller number of particles. By addressing these challenges, this study aims to contribute to the ongoing efforts to build more efficient autonomous systems.

II. Background and Related Work

A. Partially Observable Markov Decision Process (POMDP)

In Markov Decision Processes (MDPs), agents have complete and perfect knowledge of the current state of the environment. At any given time, the agent knows exactly where it is or what the situation is. In POMDPs, however, agents do not have full knowledge of the current state. The agent has to make decisions under uncertainty about the environment; thus, POMDPs are often more suitable for real-world scenarios where perfect information about the state is rarely available.

*This work is not supported by any organization or funding.

Author is with the Ann And H.J. Smead Aerospace Engineering Sciences Department, 429 UCB, University of Colorado Boulder, Boulder CO 80309, USA. Correspondence: jile6110@colorado.edu

[1]J. Lee is a first responder at Boulder Emergency Squad, 3532 Diagonal Highway Boulder, CO 80301.

Because the agent lacks full visibility of the state, it maintains a probabilistic *belief* over the possible states of the environment. This belief is updated as the agent takes actions and receives observations, following the following Bayesian equation:

$$b'(s') \propto P(o|s', a) \sum_{s \in S} P(s'|s, a) b(s) \tag{1}$$

- $b'(s')$ is the posterior belief of being in state s' after taking action a and receiving observation o.
- $P(o|s', a)$ is the probability of receiving observation o after taking action a and ending up in state s'.
- $P(s'|s, a)$ is the transition probability of moving from state s to state s' after taking action a.
- $b(s)$ is the prior belief of being in state s before taking action a.

Solving POMDPs involves finding optimal policies that specify the best action to take based on the current belief state.

B. Rao-Blackwellized Particle Filter (RBPF)

A Rao-Blackwellized Particle Filter (RBPF) is an advanced form of particle filtering that leverages the Rao-Blackwell theorem to enhance computational efficiency and estimation accuracy. This theorem is based on sufficient statistics to reduce the variance of estimators [4] as represented in (2).

$$\text{var}[\tau(x|Y_{1:N})] \geq \text{var}[\tau(x|Y_{1:N}, \Theta) \tag{2}$$

Each particle is associated with a conditional Gaussian distribution whose parameters are updated using standard Kalman filter equations. Hence, an RBPF can be effectively used in scenarios where part of the system dynamics or observations can be described by linear models with Gaussian noise, which is common in many practical applications such as navigation systems. This approach allows RBPF to focus the particle filter's computational resources only on the non-linear component, thereby reducing the number of particles required and increasing the update speed.

C. Conformal Prediction (CP)

Due to the partial observability in POMDP models, agents must consider not only the randomness of the system but also the uncertainty about the actual state of the system. Hence, the uncertainty of the agent's belief consists of both aleatory (intrinsic randomness of the system) and epistemic (uncertainty due to lack of knowledge) components that come from both transition and observation functions represented in (1). This uncertainty in the belief state can be quantified with Conformal Prediction (CP), a frequentist technique characterized by its approach to deriving distribution-free uncertainty quantification. This method is capable of producing confidence intervals with specified coverage using a finite number of independent and identically distributed (i.i.d.) samples [6].

To illustrate how CP works, consider a practical example of image classification. The process begins with a fitted predicted model, denoted as $\hat{f}$, which outputs estimated probabilities for each class: $\hat{f}(x) \in [0, 1]^K$, where K represents the number of classes. The learner then uses the classifier along with calibration data, which is distinct from the training set, to constructs a prediction set of possible labels $C(X_{\text{test}}) \subset \{1, \ldots, K\}$. This construction ensures validity as per the following condition:

$$1 - \alpha \leq P(Y_{\text{test}} \in C(X_{\text{test}})) \leq 1 - \alpha + \frac{1}{n+1}, \tag{3}$$

where $(X_{\text{test}}, Y_{\text{test}})$ is a new test point from the same distribution, and $\alpha \in [0, 1]$ is a predefined error rate. This approach ensures that the probability of the prediction set including the correct label is nearly $1 - \alpha$ [6].

For those interested in exploring further, the Awesome Conformal Prediction Git repository [8] offers a comprehensive list of up-to-date papers and resources, highlighting current trends and advancements in the field.

III. Problem Formulation

The goal of the planning module is to devise a navigational strategy for a drone to search an area for a missing hiker whose location is not precisely known but is estimated based on the last known point. Additionally, the drone operates in a GPS-denied environment, necessitating the prediction of its own position based on noisy observations from static landmarks. In this section, we detail the elements of the POMDP model $\mathcal{P} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{O}, \mathcal{R}, \mathcal{Z})$ that we utilize for our study.

State Space $\mathcal{S}$: The state space of our model consists of continuous variables representing the drone's position and orientation, as well as the estimated position of the hiker. Specifically, the state vector is defined as $s = \{\xi_d, \eta_d, \theta_d, \xi_h, \eta_h\}$, where ξ_d and η_d are the drone's coordinates, θ_d is its heading, and ξ_h and η_h are the coordinates of the estimated position of the hiker.

Actions $\mathcal{A}$: The action set available to the drone includes both movement and operational modes, discretized into five specific commands. Each action tuple (s_t, w_t, m) consists of the speed s_t (km/s), turning rate w_t (rad/s), and a binary search mode indicator m. The discrete actions are as follows:

$$\begin{aligned}
\alpha_1 &= (s_t, 0.0, 0) \quad \text{(Move Forward)} \\
\alpha_2 &= (-s_t, 0.0, 0) \quad \text{(Move Backward)} \\
\alpha_3 &= (0.0, w_t, 0) \quad \text{(Turn Left)} \\
\alpha_4 &= (0.0, -w_t, 0) \quad \text{(Turn Right)} \\
\alpha_5 &= (0.0, 0.0, 1) \quad \text{(Search)}
\end{aligned}$$

Observations $\mathcal{O}$: The observation space consists of noisy sensor readings that include the relative position of the hiker, measurements of known landmarks, and a boolean indicating whether the hiker has been found. The observation vector is defined as $\mathbf{o} = \{\Delta x_h, \Delta y_h, \{(\rho_j, \phi_j)\}_{j=1}^{J}, \xi_{\text{found}}\}$ where ρ_j and ϕ_j are relative range and bearing to each landmark j, respectively, and J is the total number of landmarks the drone uses for localization.

Transition Model $\mathcal{T}$: The dynamics of the drone are described by the following differential equations:

$$\dot{\xi} = s_t \cdot \tilde{w}_s \cdot \cos(\theta),$$
$$\dot{\eta} = s_t \cdot \tilde{w}_s \cdot \sin(\theta),$$
$$\dot{\theta} = \omega_t \cdot \tilde{w}_\omega.$$

Here, $\tilde{w}_s$ and $\tilde{w}_\omega$ represent the noise components affecting the speed and turning rate, respectively, and s_t and w_t are defined by the chosen action. These equations can be implemented using Runge-Kutta methods like ode45 with a fixed timestep Δt. We assume the location of the hiker is fixed.

Observation Model $\mathcal{Z}$: The observation model includes both noisy measurements and deterministic check of the hiker. Specifically, the observations include:

- Δx_h and Δy_h: Noisy measurements of the hiker's position relative to the drone, modeled as:

  $$\Delta x_h = \xi_h - \xi_\text{d} + v_x,$$
  $$\Delta y_h = \eta_h - \eta_\text{d} + v_y,$$

  where $v_x, v_y \sim \mathcal{N}(0, \sigma^2_\text{sensor})$ represent the noise in the hiker's location sensor.
- Relative range ρ_j and bearing ϕ_j to each landmark j, as:

  $$\rho_j(k) = \sqrt{(\xi_j - \xi_d(k))^2 + (\eta_j - \eta_d(k))^2} + v_r(k),$$
  $$\phi_j(k) = \text{atan2}(\eta_j - \eta_d(k), \xi_j - \xi_d(k)) - \theta_d(k) + v_b(k),$$

  where $v_r(k)$ and $v_b(k)$ are the noise components in the range and bearing measurements, respectively.
- ξ_found: A deterministic observation indicating whether the hiker has been found, calculated as:

  $$\xi_\text{found} = \begin{cases} 1 & \text{if } \sqrt{(\xi_d - \xi_h)^2 + (\eta_d - \eta_h)^2} < r_\text{d}, \\ 0 & \text{otherwise}, \end{cases}$$

  where r_d is the detection radius within which the drone can confirm the hiker's presence.

Reward Model $\mathcal{R}$: The reward function is designed to motivate the drone to efficiently locate the hiker. It is composed of three distinct components: $R = R_\text{mov} + R_\text{bearing} + R_\text{search}$.

- $\mathcal{R}_\text{mov}$: This component rewards the drone for reducing the distance to the estimated location of the hiker. Mathematically, it is given by:

  $$\mathcal{R}_\text{mov}(s, s') = c_{mov} \cdot (d(s) - d(s'))$$

  where $d(s)$ computes the Euclidean distance from the drone to the hiker's estimated location in state s and c_{mov} is a constant.
- $\mathcal{R}_\text{bearing}$: This reward improves the alignment of the drone's heading with the direction to the hiker. It is quantified as:

  $$\mathcal{R}_\text{bearing}(s, s') = c_{bearing} \cdot (\beta(s) - \beta(s'))$$

  where $\beta(s)$ calculates the absolute difference between the drone's heading and the bearing to the hiker's estimated location in state s and $c_{bearing}$ is a constant.
- $\mathcal{R}_\text{search}$: This component of the reward function is triggered when the drone is in search mode. It incentivizes the drone to operate within an effective detection range. The search reward is defined as:

  $$\mathcal{R}_\text{search}(s, a, s') = \begin{cases} 100 & \text{if } d(s') \leq r_d \\ -100 & \text{otherwise} \end{cases}$$

IV. Technical Approach

A. RBPF Setup in POMDP

To employ RBPF within POMDP models, several adjustments are necessary.

State Space $\mathcal{S}$: Because each particle is associated with a conditional Gaussian distribution in RBPF, the state space of our model now comprises non-linear components and a Gaussian belief. Specifically, the state vector is defined as $s = \{\theta_d, \xi_h, \eta_h, b_d\}$, where b_d is the Gaussian distribution $\mathcal{N}(\mu_d, \Sigma_d)$ for ξ_d and η_d.

Transition Model $\mathcal{T}$: The parameters of the conditional Gaussian distribution are updated using standard Kalman filter equations. Given a predicted θ, we can use Unscented Kalman Filter (UKF) prediction step to approximate Gaussian belief with sigma points and derive the predicted belief for ξ_d and η_d.

Observation Model $\mathcal{Z}$ and Reward Model $\mathcal{R}$: Instead of having specific values for ξ_d and η_d, we now have a Gaussian belief b_d. Hence, the Euclidean distance is computed based on μ_d.

B. POMCP Solver

Among the existing POMDP solvers, we used POMCP for online planning. POMCP, based on Monte Carlo Tree Search (MCTS) principles, is an online solver that effectively navigates the decision space in uncertain environments by building and exploring a search tree of possible actions and observations from the current belief state [7]. This is an approximate method that relies on random sampling to explore the decision space without computational overhead and offers real-time adaptability and scalability for complex environments in SAR scenarios. After completing a simulation, POMCP backpropagates the observed rewards through the tree, updating the values of states and actions using Bayesian inference. This process refines the decision-making strategy over time and selects the next best action.

C. Weight Computation for RBPF with UKF

The weights of the particles are updated based on the observation likelihood calculated using the innovation covariance matrix to account for the Gaussian components. This is different from the standard SIRPF where the weights are simply updated using the likelihood of the observation given the predicted state.

Expected Measurements: Given a particle $\mathbf{x}_k^{(i)} = (\theta_d^{(i)}, \xi_h^{(i)}, \eta_h^{(i)}, \mu_d^{(i)}, \Sigma_d^{(i)})$, we calculate the expected measurements as follows:

$$\hat{r}_{k,j}^{(i)} = \sqrt{(\xi_j - \mu_{d,\xi}^{(i)})^2 + (\eta_j - \mu_{d,\eta}^{(i)})^2},$$
$$\hat{\phi}_{k,j}^{(i)} = \text{atan2}(\eta_j - \mu_{d,\eta}^{(i)}, \xi_j - \mu_{d,\xi}^{(i)}) - \theta_d^{(i)},$$

where $\hat{r}_{k,j}^{(i)}$ and $\hat{\phi}_{k,j}^{(i)}$ represent the expected range and bearing, respectively, to the j-th landmark at timestep k, and ξ_j and η_j are the coordinates of the landmark.

Innovation Covariance: The innovation covariance matrix $\mathbf{S}_k^{(i)}$ is obtained as follows:

$$\mathbf{S}_k^{(i)} = \sum_{\ell=0}^{2n} w_c^{\ell} \left(\gamma_k^{\ell} - \hat{\mathbf{z}}_k^{(i)}\right) \left(\gamma_k^{(\ell)} - \hat{\mathbf{z}}_k^{(i)}\right)^{\top} + \mathbf{R_k},$$

where $\gamma_k^{(\ell)}$ represents the ℓ-th sigma point, w_c^{ℓ} is the corresponding weight, $\hat{\mathbf{z}}_k^{(i)}$ is the expected range and bearing measurement vector $\hat{\mathbf{z}}_k^{(i)} = [\hat{r}_{k,1}^{(i)}, \hat{\phi}_{k,1}^{(i)}, \ldots, \hat{r}_{k,J}^{(i)}, \hat{\phi}_{k,J}^{(i)}]$, and $\mathbf{R_k}$ is the measurement noise covariance. UKF uses sigma points to approximate the distribution of the measurement.

Likelihood: The likelihood $\mathbf{L}_k^{(i)}$ of the observation $\mathbf{z}_k$ given the predicted state is then calculated using a multivariate Gaussian distribution:

$$\mathbf{L}_k^{(i)} = \mathcal{N}\left(\mathbf{z}_k \mid \hat{\mathbf{z}}_k^{(i)}, \mathbf{S}_k^{(i)}\right), \tag{4}$$

D. Conformal Intervals Calculation

We utilize CP to quantify the inherent uncertainty in our belief estimates for θ. For a Bayesian model, the posterior predictive density is used as the conformal score [6]. Hence, we use Kernel Density Estimation (KDE), a non-parametric approach, to obtain the negative posterior predictive density from the computed weights and particle collections:

$$s(z, \theta) = \hat{f}(\theta|z) \tag{5}$$

Then $\hat{q}$ is computed as the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ quantile of the scores, where α is the error rate, and we form prediction intervals using the following equation:

$$C(z) = \{\theta : \hat{f}(\theta|z) > -\hat{q}\} \tag{6}$$

We present a generic sketch of our proposed algorithm in Algorithm 1.

V. Results

A. Conformal Intervals Analysis

We conducted 100 ensembles of simulations to analyze the conformal intervals and compare the performance of SIRPF and RBPF.

Variance Analysis: As shown in Fig. 2, the variance of the bounds of θ increases over time for SIRPF. However, the results suggest that RBPF is more stable over time, even with 100 particles, and exhibits lower variance than SIRPF with 1000 particles in the later time steps. This illustrates the stability and robustness of RBPF compared to SIRPF.

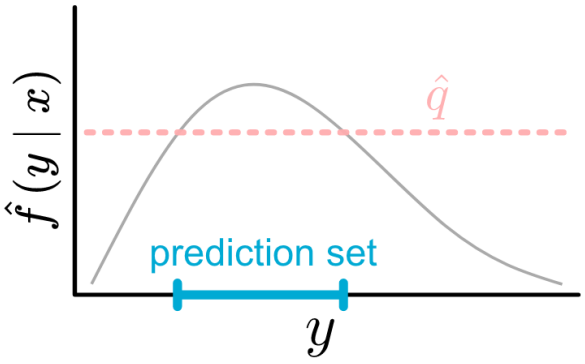

Fig. 1. A visual graph of Eq. 6 from [6] where y represents θ and x is the observation

Algorithm 1 Computation of Conformal Intervals for POMCP Belief Estimates

procedure ComputeConformalIntervals(*particles*, *weights*, α)
  Let $\theta_{estimates} \leftarrow$ Extract θ estimates from *particles*
  Let $kde \leftarrow$ Fit KDE to $\theta_{estimates}$ with *weights*
  Let $S \leftarrow$ Empty list to store scores for each θ
  for each θ in $\theta_{estimates}$ **do**
    Let $s \leftarrow -1 \times \text{pdf}(kde, \theta)$
    Append s to S
  end for
  Let $\hat{q} \leftarrow$ Compute $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$ quantile of S
  Let $\theta_{grid} \leftarrow$ linspace from θ_{min} to θ_{max}
  Let $P \leftarrow \theta$ in θ_{grid} where $pdf(kde, \theta) > -\hat{q}$
  return $(\min(P), \max(P))$

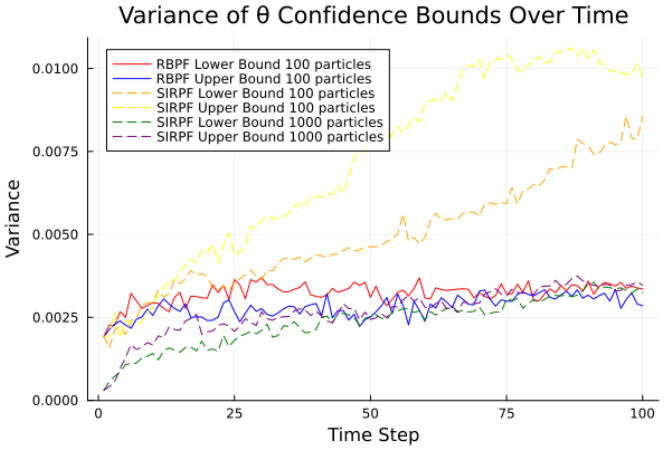

Fig. 2. Ensemble variance of θ over time for RBPF and SIRPF with different particle counts.

Average Width Analysis: Fig. 3 depicts the average width of the θ confidence bounds over time. Interestingly, SIRPF with 100 particles has the smallest width. However, this results from a loss of diversity in the particles (as detailed in the Effective Sample Size Analysis section) and implies an overly optimistic estimator. As time progresses, the width of SIRPF with 1000 particles gets narrower but is still wider than that of RBPF with 100 particles, indicating that RBPF exhibits better precision constantly over time.

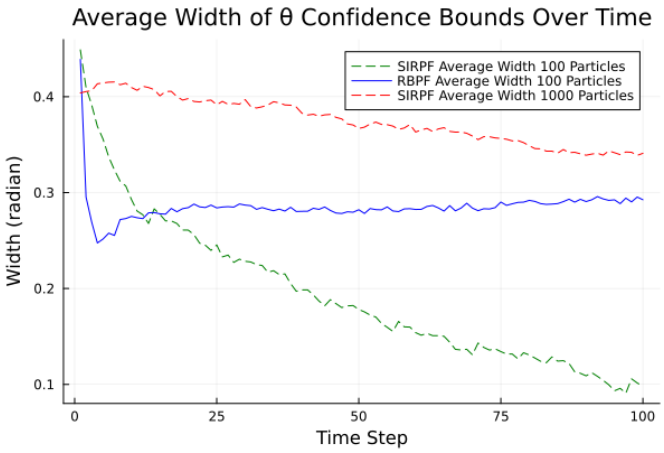

Fig. 3. Average width of θ over time for RBPF and SIRPF with different particle counts.

B. Effective Sample Size Analysis

The Effective Sample Size (ESS) comparison in Fig. 4 illustrates the efficiency of the particle filters. The ESS measures the number of effectively independent particles and is calculated as follows:

$$\text{ESS} = \frac{1}{\sum_{i=1}^{N} w_i^2}, \tag{7}$$

where w_i are the normalized particle weights.

The ESS for SIRPF is consistently below 50 percent and sometimes falls as low as 17 percent, indicating significant particle impoverishment, explaining the small width in Fig. 3. On the other hand, the ESS for RBPF remains relatively stable and is constantly above 50 percent.

C. Computational Costs Analysis

We focus on two key aspects for the computational costs: the cost of time for POMCP action selection and belief updates. From Table I, it is evident that RBPF is consistently slower than SIRPF for both POMCP action selection and belief updates when using the same number of particles. This is attributed to the increased computational complexity involved in UKF. Expectedly, the time taken for POMCP action selection remains relatively constant regardless of the number of particles. This is because the Monte Carlo sampling method employed in POMCP primarily depends on the depth of the search tree and the number of rollouts [7] rather than the number of particles in the belief. The depth and the number

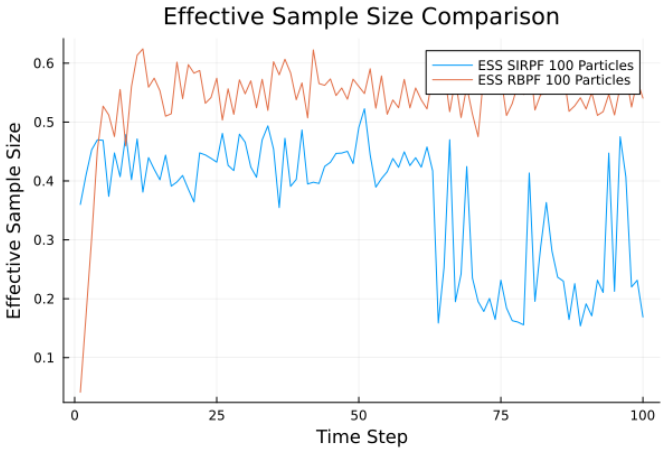

Fig. 4. Effective Sample Size (ESS) comparison for RBPF and SIRPF with 100 particles.

of rollouts were fixed in this comparison. In contrast, the time taken for belief updates increases linearly with the number of particles. The RBPF belief update is approximately ten times slower compared to SIRPF.

TABLE I
COMPARISON OF RBPF AND SIRPF ON COST OF TIME FOR POMCP ACTION SELECTION AND BELIEF UPDATES

Particle Count	POMCP Action Selection		Belief Updates	
	RBPF	SIRPF	RBPF	SIRPF
100	548.3 ms	3145 ms	24.99 ms	4.336 ms
1,000	593.1 ms	3266 ms	237.8 ms	32.14 ms
10,000	572.2 ms	3455 ms	2397 ms	315.4 ms
100,000	582.9 ms	3738 ms	25250 ms	3275 ms

VI. DISCUSSION

The lower variance observed in RBPF, even with fewer particles, demonstrates its efficiency for belief estimation in the given POMDP framework. In contrast, SIRPF requires a substantially larger number of particles to achieve comparable variance levels. Moreover, while the variance of the confidence bounds for SIRPF increases over time, RBPF maintains consistent performance, underscoring its potential in scenarios where long-term stability is crucial. However, the computational cost analysis showed that RBPF is slower than SIRPF for both POMCP action selection and belief updates due to the added complexity from UKF. In future work, we could explore using a simpler Extended Kalman Filter (EKF) to reduce computational time and also examine the performance of the belief updaters over longer horizons (time steps > 100). Overall, since the number of particles needed for SIRPF to avoid particle collapse grows "super-exponentially" with the effective dimension [5], we believe in the value of using RBPF in POMDP models. By reducing the dimensionality of the state space that particles need to explore, RBPF will effectively lower computational costs without compromising accuracy.

VII. Conclusion

In this study, we compared the performance of RBPF and SIRPF as belief updaters for a POMDP framework in SAR drone scenario. We used CP to quantify the uncertainty of the belief, providing confidence bounds for the state θ estimates. The results demonstrated that RBPF is more efficient at belief estimation in terms of particle numbers due to its ability to integrate out linear components by using UKF. However, the computational cost analysis highlighted that RBPF is slower than SIRPF for both POMCP selection and belief updates when using the same number of particles due to the added computational overhead. Future work should explore integrating simpler Kalman filters to reduce even more computational time and examine performance over longer horizons. In conclusion, our study indicates that RBPF offers a valuable approach for POMDP models as a belief updater, particularly when the system's effective dimensionality is high.

VIII. Release and Acknoledgment

The author grants permission for this report to be posted publicly. Also, the author extends his sincere thanks to Professor Zachary Sunberg[1], Professor Nisar Ahmed[2], and Professor Alireza Doostan[3] for their insightful input, guidance, and feedback.

References

[1] H. M. Ray, R. Singer, and N. Ahmed, "A Review of the Operational Use of UAS in Public Safety Emergency Incidents," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, Dubrovnik, Croatia: IEEE, Jun. 2022, pp. 922–931. doi: 10.1109/ICUAS54217.2022.9836061.

[2] D. Kingston, S. Rasmussen, and L. Humphrey, "Automated UAV tasks for search and surveillance," in *2016 IEEE Conference on Control Applications (CCA)*, IEEE, 2016, pp. 1–8.

[3] K. D. Julian and M. J. Kochenderfer, "Distributed wildfire surveillance with autonomous aircraft using deep reinforcement learning," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 8, pp. 1768-1778, 2019.

[4] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman filter: Particle filters for tracking applications*. Boston: Artech House, 2004.

[5] P. J. Bickel, B. Li, and T. Bengtsson, "Sharp failure rates for the bootstrap particle filter in high dimensions," in *Institute of Mathematical Statistics eBooks*, 2008, pp. 318–329. doi: 10.1214/074921708000000228.

[6] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," *arXiv preprint arXiv:2107.07511*, 2021.

[7] D. Silver and J. Veness, "Monte-Carlo Planning in large POMDPs," *Neural Information Processing Systems*, vol. 23, pp. 2164–2172, Dec. 2010. [Online]. Available: http://jveness.info/publications/nips2010%20-%20pomcp.pdf.

[8] V. Manokhin, "Awesome Conformal Prediction," version v1.0.0, Apr. 2022.

[9] M. J. Kochenderfer, T. A. Wheeler, and K. H. Wray, *Algorithms for Decision Making*. MIT Press, 2022.

[10] Y. K. Tee and Y. C. Han, "Lidar-Based 2D SLAM for Mobile Robot in an Indoor Environment: A Review," in *2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST)*, Miri, Malaysia, 2021, pp. 1–7, doi: 10.1109/GECOST52368.2021.9538731.

[11] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A Rao–Blackwellized particle filter for 6-D object pose tracking," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1328–1342, Oct. 2021, doi: 10.1109/tro.2021.3056043.

[1]Instructor for ASEN 5264: Decision Making under Uncertainty

[2]Instructor for ASEN 6044: Advanced State Estimation

[3]Instructor for ASEN 6412: Uncertainty Quantification

Jiho Lee is a current M.S. student in Aerospace Engineering Sciences at the University of Colorado Boulder. He graduated from the University of Pennsylvania with BS in Economics and BAS in Computer Science and Mathematics (Minor).