Object Permanence Filter for Robust Tracking with Interactive Robots

Shaoting Peng[1], Margaret X. Wang[2], Julie A. Shah[2] and Nadia Figueroa[1]

***Abstract*— Object permanence, which refers to the concept that objects continue to exist even when they are no longer perceivable through the senses, is a crucial aspect of human cognitive development. In this work, we seek to incorporate this understanding into interactive robots by proposing a set of assumptions and rules to represent object permanence in multi-object, multi-agent interactive scenarios. We integrate these rules into the particle filter, resulting in the Object Permanence Filter (OPF). For multi-object scenarios, we propose an ensemble of K interconnected OPFs, where each filter predicts plausible object tracks that are resilient to missing, noisy, and kinematically or dynamically infeasible measurements. Through several interactive scenarios, we demonstrate that the proposed OPF approach provides robust tracking in human-robot interactive tasks agnostic to measurement type, even in the presence of prolonged and complete occlusion. Project webpage: https://opfilter.github.io/.**

I. Introduction

As robots start escaping factories and coming into our homes and everyday lives, we must ensure that human-robot interaction (HRI) is safe, resilient, and robust. Much work in the HRI community focuses on developing safety-critical control schemes for interactive robots [1], [2]. However, offering resilience and robustness relies not only on the strengths of controllers but also on the weaknesses of perception systems. Unreliable perception is the primary bottleneck for deploying HRI systems in the real world.

In the context of multi-object multi-agent scenarios, the majority of modern tracking algorithms follow the 'tracking-by-detection' paradigm, i.e., an object detector is used to identify objects in a camera frame, and the position and orientation of these objects are determined and linked into tracks. This approach heavily relies on the accuracy of object detection and motion prediction. However, in many HRI scenarios, the presence of occlusions is inevitable. Even the most advanced vision-based algorithms and hardware systems can fail in simple HRI tasks, such as tracking or handing over objects that are being manipulated or occluded by a human, another agent, or an object. Objects such as bikers or hand movements are inherently difficult to predict, and their trajectories are further complicated by the presence of partial or complete occlusions. Perception failures resulting from such occlusions can cause missing, noisy, or kinematically and dynamically infeasible measurements. If not addressed, these failures can negatively impact the expected behavior of the robot and threaten the safety of humans interacting with it. Rather than expecting a perfect perception system to arise, we instead assume that perception will always be unreliable in HRI systems and should be actively mitigated. In this work, we propose an approach inspired by human cognition and development to alleviate these inevitable, unreliable perception issues. According to Piaget's theory of cognitive development [3], humans develop an understanding of object permanence at an early age, i.e., they understand that objects continue to exist even when they are not visible or cannot be sensed. Object permanence is crucial to perception and memory and is a defining feature of human intelligence.

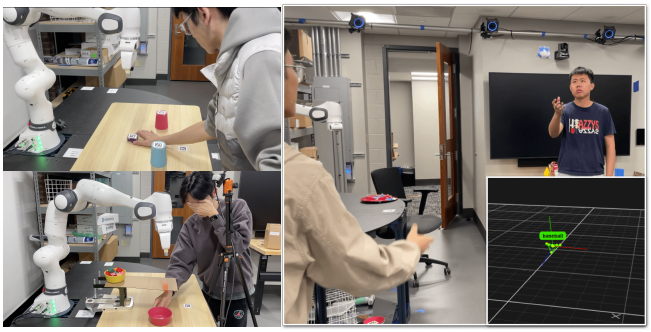

Fig. 1: Our Object Permanence Filter can be used for robust tracking of occluded objects or noisy measurements with different tracking systems at different frequencies. Apriltags (30Hz) used on the left for the cups game and sugar-dropping experiment and Optitrack (100Hz) used on the right for tracking a flying object.

Object permanence is evident in everyday life through various situations and experiences. A baby playing with a toy comprehends that the toy continues to exist even when it is concealed under a blanket; an adult who hides a car key in pockets understands that the key still exists. In addition, when a car drives behind a building, one can infer the car's location and expect it to reappear on the other side. Furthermore, object permanence plays a crucial role in more advanced forms of problem-solving, such as playing the cup game, during which an individual is aware of the ball being covered by one of the N cups and can easily track the cup. Humans unequivocally use object permanence to make predictions about hidden objects and plan their actions accordingly. This capability to comprehend and reason about the persistence of objects is imperative for a broad range of activities and experiences. As demonstrated by Saiki *et al.* [4], human beings are capable of maintaining multiple coherent object representations in dynamic scenarios, thereby enabling them to track multiple objects effectively. However, it becomes difficult as the number of objects increases. Hence, it is imperative for robots to develop their own robust models of object permanence. Developing such models can help robots understand the persistence of objects, leading to improved performance in various HRI tasks.

*Corresponding author: pengsht@seas.upenn.edu
[1]University of Pennsylvania, Philadelphia, USA
[2]Massachusetts Institute of Technology, Cambridge, USA

Contributions: We introduce the Object Permanence Filter (OPF) as a means to achieve resilient multi-object tracking. We modify the update step in particle filters with an Object Permanence Update (OP Update) that is robust to varying degrees of visual disruption. The OP update consists of three modules: *dynamics, occluder* and *uncertainty* modules, used to provide virtual measurements and scale covariance matrix when occlusions are detected (Fig. 2). By incorporating object permanence rules into these modules, a robot's perception system can maintain track of objects even when they are partially or completely occluded. In addition, we introduce a *feedback module* that monitors the uncertainty of the estimates and is used to modulate a robot's tracking behavior and inform the human operator if uncertainty explodes, allowing for a more safe, robust, and fluid HRI. We conduct comprehensive assessments of the OPF, employing both simulation and hardware experimentation. Our findings demonstrate the robust tracking capabilities of the OPF in heavy occlusion scenarios, alongside its capacity to adapt seamlessly to various measurement types.

II. Related Work

6-DoF Object Tracking: 6-DoF object tracking is an active area of research in computer vision (CV), encompassing the task of estimating the position and orientation of an object in 3D space. Various techniques have been proposed to address this problem, ranging from classical CV methods to learning-based approaches. As elucidated by Chen [5], the Kalman filter has emerged as a prevalent technique in object tracking due to its ability to combine noisy measurements with a dynamic model of the object's position and orientation. In more recent developments, researchers are using deep neural networks to learn intricate, high-dimensional representations from data for better tracking performance [6]. Nevertheless, the state-of-the-art approaches adhere to the tracking-by-detection paradigm, which can yield suboptimal results in occlusion scenarios. Though some work can alleviate it by memory mechanism [7][8] or sensor compensation [9][10], these methods primarily fall within the domain of learning-based approaches and may pose challenges when attempting to integrate them with other types of tracker and measurement. Besides, the rarity of depth and point cloud, as well as prolonged or extensive occlusion, invariably pose more challenges to the tracking performance.

Occlusion handling: Several research efforts have concentrated on addressing occlusions directly, including some modifications of the particle filter [11][12]. For instance, Kourosh *et al.* [12] introduced an occlusion-aware particle filter tracker. During object occlusion, their approach resorts to a stochastic particle motion mechanism, resembling a random walk pattern, causing particles to disperse widely from the last known object position in a broader search endeavor. While they handle occlusions in a heuristic way, by doing random searches they do not integrate the concept of object permanence. In contrast, Tokmakov *et al.* [13] improved CenterTrack with a spatiotemporal recurrent neural network, introducing object permanence for joint detection and tracking. Their approach uses deterministic pseudo-ground-truth during occlusions by extrapolating object positions based on the last velocity. While similar in nature to the role of our *dynamics module* described in Section III-B.1, the concept of object permanence is not exploited elsewhere, including the more important aspect as introduced in Section III-B.2, and the NN-based tracker may diverge with wrongly generated pseudo-ground-truth labels, causing undesirable behaviors. Methods that consider object permanence thoroughly in the context of diverse occlusion scenarios are still in need.

III. OPF: Object Permanence Filter

In this section, we introduce our proposed object permanence filter (OPF) framework. In section III-A, we introduce the prediction and update equations for the particle filter as well as our usages, followed by the update rules to estimate the virtual observation y_k^{occ} and covariance scaling function α_k for the PF updates in section III-B. Finally, in Section III-C, we show that by monitoring the posterior covariance matrix, we can create cautious closed-loop tracking controllers and convey to humans that certain objects have been occluded for a long time to ensure safety.

A. PF for Object Tracking

The particle filter works by generating and manipulating sets of particles to approximate the distribution of a stochastic process following a 'predict + update' cycle.

Importance Sampling The key idea behind the PF is called importance sampling [14], a technique that uses a known probability distribution $q(x)$ (proposal) to generate particles $x^{(i)}$ and approximate a given probability distribution $p(x)$ (target) by assigning weight to each of the particles:

$$x^{(i)} \sim q, \quad w^{(i)} = \frac{p(x^{(i)})}{q(x^{(i)})}, \quad \forall i = 1, ..., n \tag{1}$$

Predict Here we start with particles $x_{k|k}^{(i)}$:

$$P(x_k|y_1, ..., y_k) = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_{k|k}^{(i)}}(x) \tag{2}$$

where δ denotes the Dirac delta distribution, and all weights are equal, i.e. $w_{k|k}^{(i)} = \frac{1}{n}$. Suppose system dynamics f with Gaussian noise ϵ_k, then we can propagate each particle by one timestep $\forall i = 1, \ldots, n$:

$$x_{k+1|k}^{(i)} = f(x_{k|k}^{(i)}, u_k) + \epsilon_k, \ \epsilon_k \sim N(0, R) \tag{3}$$

Particles weights are unchanged, i.e., $w_{k+1|k}^{(i)} = w_{k|k}^{(i)} = \frac{1}{n}$.

Update Then we update the weight of each particle using the likelihood of receiving a new observation y_{k+1}:

$$w_{k+1|k+1}^{(i)} = \eta P(y_{k+1}|x_{k+1|k}^{(i)}) w_{k+1|k}^{(i)} \tag{4}$$

where η is a normalization factor.

Given a measurement function g, $P(y_{k+1}|x_{k+1|k}^{(i)})$ is a Gaussian with mean $g(x_{k+1|k}^{(i)})$, variance Q, and depends on Gaussian observation noise ν_k:

$$\begin{aligned} P(y_{k+1}|x_{k+1|k}^{(i)}) &= P\left(\nu_{k+1} \equiv y_{k+1} - g\left(x_{k+1|k}^{(i)}\right)\right) \\ &= \frac{1}{\sqrt{(2\pi)^p \det(Q)}} \exp\left(-\frac{\nu_{k+1}^T Q^{-1} \nu_{k+1}}{2}\right) \end{aligned} \tag{5}$$

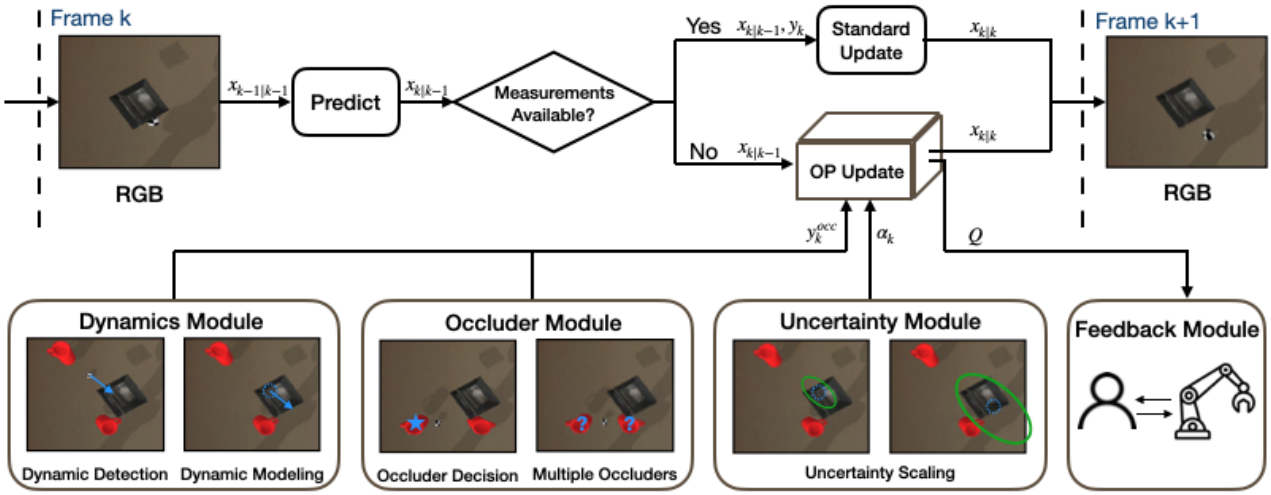

Fig. 2: **Object Permanence Filter (OPF):** Following the predict-update cycle, the OPF introduces **OP update** when no measurements are available for the object being tracked. **OP update** consists of i) **dynamics module** to detect and model the dynamics of the object, ii) **occluder module** to help decide the occluder and deal with multiple occluders, and iii) **uncertainty module** to update the covariance matrix. The **feedback module** monitors the uncertainty of the updates by tracking the trace of the update covariance matrix, which can be used to change the behavior of the robot or indicate to the human operator when the uncertainty is above a safety threshold ϵ_{safe}.

Resampling To avoid particle degeneracy problem [15], resampling is proposed to remove unlikely particles with very low weights and effectively split the particles with very large weights into multiple particles:

$$p(x) = \sum_{i=1}^{n} w^{(i)} \delta_{x^{(i)}}(x) \Rightarrow \frac{1}{n} \sum_{i=1}^{n} \delta_{x'^{(i)}}(x) \tag{6}$$

For our 6-Dof usage, an object's state is represented as

$$x_k = (\xi_k^x, \xi_k^y, \xi_k^z, \theta_k, \phi_k, \psi_k) \in \mathbb{R}^6 \tag{7}$$

$(\xi_k^x, \xi_k^y, \xi_k^z)$ is the translation of the object in the current frame (in meters), and (θ, ϕ, ψ) is the rotation, represented as Euler angles (in radians). Due to the high dimensionality and mismatch in the scales of translation and rotation, sampling the entire state can be inaccurate. Instead, we create two portions of particles to sample each, which improves sampling performance while maintaining decent time complexity.

B. Object Permanence Update

When a measurement from an object is missing due to occlusion or sensor failure at t_{occ}, then the state will remain the same for $k > t_{\text{occ}}$, which is $x_{k>t_{\text{occ}}|k>t_{\text{occ}}} \equiv x_{t_{\text{occ}}|t_{\text{occ}}}$. Due to object permanence, humans can understand that such missing measurements may be due to occlusions and can even predict the motion of that occluded object. To embed such a concept into the PF, in this work, we propose the Object Permanence (OP) Update that i) detects when occlusions happen, ii) infers the occluder from state estimates of neighboring objects, iii) estimates the occluded object dynamics, iv) updates state-estimate uncertainty.

OP Update Overview: Let $\mathcal{O}_i$ be the i-th object being tracked in a set of K objects/agents in the robot's workspace.

We start with the dynamics module (Section III-B.1), which is used to detect whether object $\mathcal{O}_i$ was moving before occlusion. If so, we model the object dynamics to get future predictions as virtual measurements, which are continuously passed to the update stage until a new measurement appears. If the object is static, we move to the occluder module (Section III-B.2), which is used to decide which object $\mathcal{O}_j, \forall j \neq i$ is the occluder by calculating and comparing the Bhattacharyya distances [16]. The occluder's observation is used to update the measurement of the occluded object ${y_k^i}^{\text{occ}} \leftarrow y_k^j$, depicted in Fig. 2 and listed in Alg. 1. This virtual observation ${y_k^i}^{\text{occ}}$ is fed to Eq. 4. Finally, we scale the covariance matrix Q in the update step of Eq. 5 by a scalar function α_k^i (to increase uncertainty) as described in Section III-B.3.

1) Dynamics Module: Humans can not only realize the existence of an object that has been occluded via object permanence, but can also approximately guess the position of the occluded object [17]. In this work, we embed this property by maintaining a history of H object states and analyzing the trajectory of the object before occlusion. Two trajectories of object $\mathcal{O}_i$'s translation T and rotation R before frame k are defined as:

$${}^{T}\text{Tr}_{k-1}^{\mathcal{O}_i} = [(\xi_0^{x_i}, \xi_0^{y_i}, \xi_0^{z_i}), ..., (\xi_{k-1}^{x_i}, \xi_{k-1}^{y_i}, \xi_{k-1}^{z_i})] \tag{8}$$

$${}^{R}\text{Tr}_{k-1}^{\mathcal{O}_i} = [(\theta_0^i, \phi_0^i, \psi_0^i), ..., (\theta_{k-1}^i, \phi_{k-1}^i, \psi_{k-1}^i)] \tag{9}$$

If $\mathcal{O}_i$ is occluded at frame k, then we use the state of object $\mathcal{O}_i$ in past H frames (i.e. if $H = 50$ the last 50 elements in $\text{Tr}_{k-1}^{\mathcal{O}_i}$) to decide for the dynamics of object $\mathcal{O}_i$:

- ${}^{T}\delta$ is a threshold to detect translation and ${}^{R}\delta$ is a threshold to detect rotation[1], an object is regarded as **static** if,

$$\begin{aligned} &\max(\|{}^{T}\text{Tr}_{k-1}^{\mathcal{O}_i}[p] - {}^{T}\text{Tr}_{k-1}^{\mathcal{O}_i}[q]\|_2) \leq {}^{T}\delta \quad \text{AND} \\ &\max(\|{}^{R}\text{Tr}_{k-1}^{\mathcal{O}_i}[p] - {}^{R}\text{Tr}_{k-1}^{\mathcal{O}_i}[q]\|_2) \leq {}^{R}\delta, \\ &\forall p, q \in \{k-H, ..., k-1\} \end{aligned} \tag{10}$$

Then we move to occluder module (Section III-B.2) and use the detected occluder's $\bar{\mathcal{O}}_j, \forall j \neq i$ observation y_k^j as the object $\mathcal{O}_i$'s virtual measurement ${y_k^i}^{\text{occ}}$.

[1]For the current experiment setup, ${}^{T}\delta$ is set to 0.01 and ${}^{R}\delta$ is set to 0.5

- Correspondingly, an object is regarded as **moving** if,

$$\begin{aligned} &\max(\|{}^{T}\mathrm{Tr}_{k-1}^{\mathcal{O}_i}[p] - {}^{T}\mathrm{Tr}_{k-1}^{\mathcal{O}_i}[q]\|_2) > {}^{T}\delta \quad \text{OR} \\ &\max(\|{}^{R}\mathrm{Tr}_{k-1}^{\mathcal{O}_i}[p] - {}^{R}\mathrm{Tr}_{k-1}^{\mathcal{O}_i}[q]\|_2) > {}^{R}\delta, \\ &\exists p, q \in \{k-H, ..., k-1\} \end{aligned} \tag{11}$$

  Then we predict the next state of object $\mathcal{O}_i$ by fitting a first-order polynomial $\mathrm{poly}(t) = at + b$ with $a, b \in \mathbb{R}^4$ for translation T and rotation R, which is determined by minimizing the squared error:

$$a, b = \underset{a,b}{\operatorname{argmin}} \sum_{p=k-H}^{k-1} \left|\mathrm{poly}(p) - Y_p^i\right|^2 \tag{12}$$

  Here $Y_p^i \in \mathbb{R}^4$ consists of a three-dimensional translation vector $(\xi_p^{x,i}, \xi_p^{y,i}, \xi_p^{z,i})$ and one rotation scalar ω^i. To get ω^i from $(\theta^i, \phi^i, \psi^i)$, a transformation is executed whereby the Euler angles are converted into an axis-angle representation $([e_x^i, e_y^i, e_z^i], \omega^i)$ with the primary objectives of enhancing tracking efficiency by only fitting one variable ω^i around the fixed axis, and mitigating the computational complexity associated with the fitting of all three Euler angles. This transformation serves to avoid the sine waves of Euler angles which are less efficient to fit. The outcome of this fitting is well-predicted rotation and the achievement of real-time (100Hz) object permanence tracking. Once a, b are determined, we don't update them by fitting a new model using the newly predicted positions to avoid the accumulation of errors. Dynamic predictions stop once the measurement of this object appears again.

2) Occluder Module: We introduce the Bhattacharyya distance [16], which measures the similarity of two objects $\mathcal{O}_p, \mathcal{O}_q$'s probability distributions $P, Q \in \chi$:

$$\begin{aligned} D_B(\mathcal{O}_p, \mathcal{O}_q) &= -\ln\left(BC(P, Q)\right) \\ BC(P, Q) &= \int_{\chi} \sqrt{p(x)q(x)}dx \end{aligned} \tag{13}$$

If $\mathcal{O}_i$ is occluded, Eq. 13 between $\mathcal{O}_i$ and all other $K-1$ objects is used to obtain two potential occluders $\mathcal{O}_p, \mathcal{O}_q$ with smallest distances $(D_B(\mathcal{O}_i, \mathcal{O}_p) < D_B(\mathcal{O}_i, \mathcal{O}_q))$. A hyperparameter $\epsilon_{\text{occ}} = 0.01$ based on the size of the tested object is set to distinguish single/multiple occluder:

- Single Occluder: if $D_B(\mathcal{O}_i, \mathcal{O}_q) - D_B(\mathcal{O}_i, \mathcal{O}_p) > \epsilon_{\text{occ}}$, then $\mathcal{O}_p$ is regarded as occluder and provides virtual observation ${y_k^i}^{\text{occ}} \leftarrow y_k^p$.
- Multiple Occluders: if $D_B(\mathcal{O}_i, \mathcal{O}_q) - D_B(\mathcal{O}_i, \mathcal{O}_p) \leq \epsilon_{\text{occ}}$, then a virtual object $\mathcal{O}_{i'}$ with state $x_k^{i'}$ copied from x_k^i is created, and two virtual observations are created: ${y_k^i}^{\text{occ}} \leftarrow y_k^p, {y_k^{i'}}^{\text{occ}} \leftarrow y_k^q$. Both $\mathcal{O}_i$ and $\mathcal{O}_{i'}$ will continue for the 'predict-update' cycle until the observation for it appears again. Hence, uncertainty is introduced by two sets of states, and the robot is aware of the multiple occluders.

We do not pre-set the objects as occluders (e.g. hands or end-effector) or occluded objects, because sometimes so-called 'occluders' can also be occluded by objects or other occluders. In this work, we treat every tracked object equally.

Algorithm 1: Object Permanence Update Step for $\mathcal{O}_i$

Input : Past H trajectory ${}^{T,R}Tr_{k-1}^{\mathcal{O}_i}$ (Eq. 8) of $\mathcal{O}_i$
Current state $x_{k|k-1}$ of object $\mathcal{O}_i$
Current state x_k^j and measurements y_k^j of all objects $\mathcal{O}_j$ $\forall j = 1, \ldots, K \setminus i$
Output : Virtual observation ${y_k^i}^{\text{occ}}$ and covariance matrix scaling factor α_k^i for $\mathcal{O}_i$
Parameters : $H, \kappa, \epsilon_{\text{occ}}$

/* Dynamics Module */
for $p \in [k-H, k-1]$ **do**
  for $q \in [k-H, k-1]$ **do**
    if $(\|{}^{T}Tr_{k-1}^{\mathcal{O}_i}[p] - {}^{T}Tr_{k-1}^{\mathcal{O}_i}[q]\|_2 > 0.01$ ***or*** $\|{}^{R}Tr_{k-1}^{\mathcal{O}_i}[p] - {}^{R}Tr_{k-1}^{\mathcal{O}_i}[q]\|_2 > 0.01)$ **then**
      /* Fit the polynomial */
      ${y_k^i}^{\text{occ}} \leftarrow \mathrm{poly}(x)$ with Eq. 12
      $\alpha_k^i \leftarrow$ Compute from x_k^p with Eq. 14
      return $({y_k^i}^{\text{occ}}, \alpha_k^i)$
/* Occluder Module */
Dists $\leftarrow$ []
for $j \in [1, K] \setminus i$ **do**
  $D_B(\mathcal{O}_i, \mathcal{O}_j) \leftarrow$ Compute with Eq. 13
  AddItem(Dists, $D_B(\mathcal{O}_i, \mathcal{O}_j)$)
sort(Dists) ascendingly
$D_B(\mathcal{O}_i, \mathcal{O}_p), D_B(\mathcal{O}_i, \mathcal{O}_q) \leftarrow$ Dists[0], Dists[1]
if $D_B(\mathcal{O}_i, \mathcal{O}_q) - D_B(\mathcal{O}_i, \mathcal{O}_p) > \epsilon_{occ}$ **then**
  ${y_k^i}^{\text{occ}} \leftarrow y_k^p$
  $\alpha_k^i \leftarrow$ Compute with Eq. 14
  return $({y_k^i}^{\text{occ}}, \alpha_k^i)$
else
  ${y_k^i}^{\text{occ}}, {y_k^{i'}}^{\text{occ}} \leftarrow y_k^p, y_k^q$
  $\alpha_k^i, \alpha_k^{i'} \leftarrow$ Compute from x_k^p, x_k^q with Eq. 14
  return $({y_k^i}^{\text{occ}}, \alpha_k^i, {y_k^{i'}}^{\text{occ}}, \alpha_k^{i'})$

3) Uncertainty Module: In the PF, one can measure the uncertainty of the filter through its covariance matrices. In the non-occluded condition, the covariance matrix updates at each step according to Eq. 5. However, when the object is occluded we use $y_k^i \leftarrow y_k^j$ with y_k^j being the observation of the occluder $\mathcal{O}_j$. Hence, we artificially increase the corresponding covariance matrices for each time step that measurement is missing as $\tilde{Q} \leftarrow \alpha_k Q$ in Eq. 5 with:

$$\alpha_k = \kappa^v \tag{14}$$

an exponentially increasing function dependent on the velocity v calculated from the object's trajectory and constant $\kappa > 1$. We empirically found $\kappa = 1.03$ suitable to showcase a moderate exponential increase in uncertainty.

C. Feedback Module

1) Human Intervention Feedback: Notice that when an object is occluded for a long time the scaled covariance matrices $\tilde{Q}$ for the PF will explode. This is by design as we can indicate the uncertainty as:

$$\mathcal{U}_{\text{PF}} = \mathrm{trace}(\tilde{Q}) \tag{15}$$

Hence, given an uncertainty safety threshold ϵ_{safe} if $\mathcal{U}_{PF} \geq \epsilon_{\text{safe}}$ the robot should either stop, fall back to a safety mode or send an alert signal to the human operator.

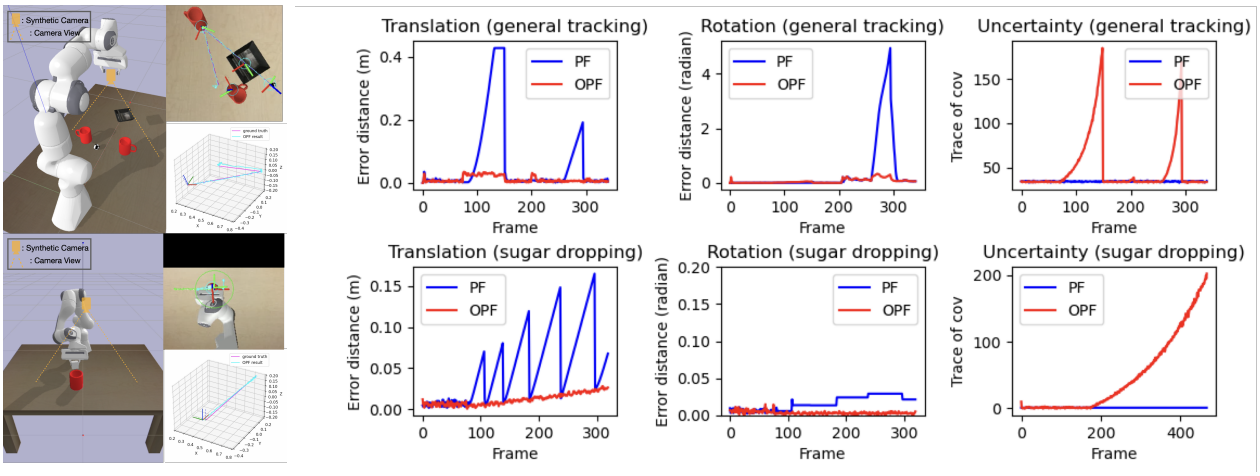

Fig. 3: **Comparative Results:** (top row) General object permanence (OP) tracking experiment, (bottom row) sugar-dropping experiment (inspired by [10]). The X-axis for all plots denotes k-th camera frame. (1st column) simulation snapshots, (2nd-3rd columns) tracking error distances of translation and rotation of the occluded object given by PF (blue) and OPF (red) (4th column) traces of Q (Eq. 5).

	General OP Tracking		Sugar-dropping	
Tracking Error	With OPF	With Standard PF	With OPF	With Standard PF
Translation error	0.01138 ± 7.253e-4	0.06289 ± 6.498e-4	0.02129 ± 5.872e-4	0.05018 ± 6.936e-4
Rotation error	0.06869 ± 2.947e-3	0.3870 ± 1.270e-3	0.003411 ± 4.629e-4	0.01456 ± 2.274e-3

TABLE I: Numerical results for general OP tracking (experiment 1) and sugar-dropping (experiment 2) over 5 runs.

2) Closed-loop Tracking Controller: Another use of $\mathcal{U}_{PF}$ defined in Eq. 15 can be to create a cautious tracking controller when used in closed-loop with the output of the filter. Let $\xi_r, \dot{\xi}_r, \xi_o, \dot{\xi}_o \in \mathbb{R}^3$ be positions and velocities of the robot's end-effector and object to track, respectively. Then one can define an object tracking control law as follows [18]:

$$\dot{\xi}_r = -k_p(\mathcal{U}_{PF})\,(\xi_r - \xi_o) + k_d(\mathcal{U}_{PF})\dot{\xi}_o \tag{16}$$

where $k_p(\mathcal{U}_{PF}) \in [0, k_p^{\text{nom}}]$ is a positive bounded tracking gain and $k_d(\mathcal{U}_{PF}) \in [0,1]$ a feedforward velocity damping term, formulated as decreasing sigmoid functions $k_{(\cdot)}(\mathcal{U}_{PF}) = k_{(\cdot)}^{\text{nom}} \cdot \frac{(\frac{1}{2}\epsilon_{\text{safe}})^n}{(\frac{1}{2}\epsilon_{\text{safe}})^n + \mathcal{U}_{PF}^n}$ with $n \geq 1$ controlling the steepness of the transition of tracking gain $k_{(\cdot)}^{\text{nom}} \to 0$ as $\mathcal{U}_{PF}$ changes from $0 \to \epsilon_{\text{safe}}$. Thus, nominal gains $(k_{(\cdot)}^{\text{nom}}, d_{(\cdot)}^{\text{nom}})$ are used when $\mathcal{U}_{PF} = 0$ and decrease to 0 as $\mathcal{U}_{PF} \to \epsilon_{\text{safe}}$.

IV. Experiments and Results

In this section, we first introduce our evaluation metrics, followed by tracking performance compared to the ground truth and the standard PF in simulated experiments (Fig. 3), as well as interesting hardware experiments designed to showcase the strength of our OPF (Fig. 1). Videos are provided in the multimedia attachment and project webpage.

A. Metrics and Evaluation Protocol

Following we describe the two metrics we use to evaluate the the PF and OPF on two occlusion-heavy simulated tasks:

- *Error distance:* Error distances for translation and rotation are defined by $\frac{\sum_{k=0}^{n} \|\hat{p}_k - p_k\|_2}{n}$, with $p_k = (\xi_k^x, \xi_k^y, \xi_k^z)$ for translation and $p_k = (\theta_k, \phi_k, \psi_k)$ for rotation of a tracked object at the k-th frame.
- *Confidence:* Another measure of the effectiveness of the OPF is the amount of confidence for each prediction. Following the feedback module (Section III-C) we evaluate confidence for each filter proportional to the uncertainty; i.e., trace$(\cdot)$ of covariance matrices via Eq. 15.

B. Comparative Simulation Experiments

We present two evaluation experiments implemented in PyBullet, (experiment 1) general OP tracking with both static and moving objects and (experiment 2) self-occluding sugar-dropping task as in [10]. Table I shows evaluation metrics on both tasks compared to the standard PF for 5 runs. Fig. 3 shows plots of the objects of interest predicted position vs. ground truth position and covariance traces w.r.t. time.

In all simulations, the origin of the coordinate system is placed on the base frame of the Franka Emika Panda arm. We assume a fixed external camera, see Fig. 3. All objects are tracked via color-based segmentation and blob detection on the RGB images from the synthetic camera. We use 5000 particles for translation and rotation per object for the PF.

1) General OP Tracking: This experiment is intended to showcase the performance of the standard PF vs. OPF while tracking multiple self-occluding dynamic and static objects.

Experiment Details: The object of interest in this experiment is a ball set initially at $[0.4, 0, 0]$. Two dynamic mugs (denoted as left/right wrt. the camera view) are set at $[0.4, \pm 0.15, 0.03]$. A static tray is at $[0.55, -0.09, 0.2]$. The camera is mounted above the table at $[0.55, 0, 0.49]$ and looks downward. We divide this experiment into two steps to test static objects and moving objects under occlusion scenarios:

Step one: The two mugs will move to the center of the table and cover the ball. The ball is closer to the left mug: $[0.4, 0.06, 0]$. Then the left mug is moved to $[0.8, 0.15, 0.03]$ and the ball moves with it. After the ball gets to the target position it is revealed and then occluded again with the same mug. This step tests the OPF tracking of a static object being occluded by multiple possible dynamic occluders.

Step two: The ball starts moving to $[0.3, -0.35, 0]$, leading it to pass under the tray. This step is designed to test the OPF with a moving object.

Analysis and Results The results are shown in Fig. 3 (top row) and Table I. Step 1 starts at frame 90, and Step 2 starts at frame 210. When using the standard PF one can see an obvious error distance in tracking. Such error arises from missing measurements, the PF will not propagate the state ahead, so the ball's predicted state is always close to where it was occluded, and abruptly updates to the position where it is revealed. The OPF on the other hand, is capable of predicting the motion of the occluded object very accurately (as shown in the plots and tracking error statistics), with an exponential increase in uncertainty by design (as shown in the trace plot). Further, during frames 250-300 in Step 2, the ball is moving under the tray. Without the OP update, the standard PF fails to predict the ball's position under the tray and performs a sudden update from the top to the bottom of the tray, while the OPF provides a much smoother approximation.

Regarding *uncertainty*, one can see in the last column of Fig. 3 the trace of the covariance matrix exhibiting 2 peaks increasing exponentially as the ball is occluded two times. For the first one it is occluded passively by the left mug, and the second one is caused by actively moving below the tray. The peaks exhibit faster and larger increases as they depend on the velocity and the timesteps that the object is occluded as per Eq. 14. When the measurement of the ball occurs again, the covariance drops to a normal scale.

2) Sugar-dropping Experiment: We showcase a dynamic sugar-dropping task, where a robot is tracking a mug with an external camera positioned prone to self-occlusions by the end-effector (inspired by [10]). The robot tracks the predicted state of the moving mug with a tracking controller as Eq. 16.

Experiment Details The robot grasps the sugar cube and then the mug is moved from $[0.55, 0.45, 0]$ to $[0.55, -0.45, 0]$. The robot will start tracking the mug as it moves to $[0.55, 0.25, 0]$ until the end. The camera is mounted at $[0.55, 0, 0.8]$ and looks downward. This experiment emulates a robot trying to drop a sugar cube into a cup of moving coffee, but the coffee cup is occluded by its own hand.

Analysis and Results In this case, the mug is occluded by the robot itself. With the OP rule, the robot is aware of the existence of the mug's dynamics and tries to approximate it, providing a smooth tracking trajectory under occlusion. In comparison, the performance of this task is erratic using the standard PF. The robot will stay still when self-occlusion exists, and sharply update to the newly observed state, resulting in a self-occlusion again and undesired jerky motion. As shown in Fig. 3 and Table I, the standard PF is laggy and jerky, whereas the OPF variants approximate the dynamics of the mug very close to the ground truth trajectory. Further, one can see the exponentially growing trace of the covariance matrix during the long self-occlusion conveying uncertainty.

C. Hardware Experiments

In this section, we conduct OPF-based tracking experiments with Franka Research 3 and test its generalizability with different measurement types. While these experiments have no ground truth they show the strength of the proposed OPF in different applications and scenarios.

1) Cups Game (Hardware): We validate the OPF on the cups game as shown in Fig. 1. We track the 2 occluding cups and the object of interest with AprilTags and human hands with Mediapipe [19]. An external camera is fixed with a 1.4m height and a downward view of the robot's workspace. Users are free to move around any of the objects and are incited to occlude the target object and move it around as in the original cups game. The output of the OPF is fed to a cautious tracking control law (Eq. 16) for the end-effector to express the tracking behavior of the OPF through motion. We also present an experiment with 3 occluding cups and multiple occluder conditions.

Analysis: Three users are invited to play the game with different occluding sequences and moving direction/speed. Results show the OPF can robustly track the object under occlusion of different kinds of occluders, and the robot is moving smoothly without jerky motions caused by sudden updates in the standard PF. Due to the feedback module, the robot stops in the end because of exploding uncertainty.

2) Sugar-dropping (Hardware): We validate the OPF on a variant of the sugar-dropping experiment from Section IV-B. To emulate self-occlusions we include a large static occluder that hides the bowl when it is moving. The robot tracks the bowl and releases the sugar as soon as its confidence of the bowl's state is high (low uncertainty) after occlusion.

Analysis: Three users are invited to generate different sugar-dropping trajectories and initial conditions. To show transparent results, we blindfold the human to avoid them actively catching the sugar cube. Results show the robot dropping the sugar in the bowl accurately with a high probability $8/10$ due to the dynamics module of the OPF.

3) Different Measurement Types: Finally, we validate that the OPF can be built on different measurement systems such as perception-based AprilTags detection and marker-based motion capture system for heavily occluded objects. More details can be found on our website and video.

V. Conclusion and Future Work

In this work, we propose a set of assumptions and rules to computationally embed object permanence into the particle filter to form the object permanence filter (OPF), which is an extended 6-DoF filter robust to heavy and prolonged occlusion scenarios in interactive tasks providing plausible tracking. We show that the OPF works well in simulation and hardware experiments, and due to its agnostic nature can be easily applied to multiple existing 6-DoF trackers achieving real-time performance. In the future, we plan to adopt physics-based object/human dynamics. Even by just considering the objects as freely moving 3D point masses and the human as a constrained 3D point mass, we can begin to diversify our modeling abilities. Finally, some parameters like δ and ϵ_{occ} can be dependent on the actual 3D shape of the object to get more generalizable thresholds for moving and occlusion.

REFERENCES

[1] P. A. Lasota, T. Fong, and J. A. Shah, "A survey of methods for safe human-robot interaction," *Foundations and Trends® in Robotics*, vol. 5, no. 4, pp. 261–349, 2017.

[2] S. Li, N. Figueroa, A. Shah, and J. A. Shah, "Provably Safe and Efficient Motion Planning with Uncertain Human Dynamics," in *Proceedings of Robotics: Science and Systems*, Virtual, July 2021.

[3] J. Piaget, "Part i: Cognitive development in children: Piaget development and learning," *Journal of Research in Science Teaching*, vol. 2, no. 3, pp. 176–186, 1964.

[4] J. Saiki, "Multiple-object permanence tracking: limitation in maintenance and transformation of perceptual objects," in *The Brain's eye: Neurobiological and clinical aspects of oculomotor research*, ser. Progress in Brain Research. Elsevier, 2002, vol. 140, pp. 133–148.

[5] S. Chen, "Kalman filter for robot vision: A survey," *IEEE Transactions on Industrial Electronics*, vol. 59, pp. 4409–4420, 2012.

[6] Y. Labb'e, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *European Conference on Computer Vision*, 2020.

[7] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Muller, A. Evans, D. Fox, J. Kautz, and S. Birchfield, "Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 606–617, 2023.

[8] B. V. Hoorick, P. Tendulkar, D. Surís, D. Park, S. Stent, and C. Vondrick, "Revealing occlusions with 4d neural fields," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3001–3011, 2022.

[9] M. Nagy, M. Khonji, J. Dias, and S. Javed, "Dfr-fastmot: Detection failure resistant tracker for fast multi-object tracking based on sensor fusion," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 827–833.

[10] N. Piga, F. Bottarel, C. Fantacci, and e. a. Vezzani, "Maskukf: An instance segmentation aided unscented kalman filter for 6d object pose and velocity tracking," *Frontiers in Robotics and AI*, vol. 8, 03 2021.

[11] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "Poserbpf: A rao-blackwellized particle filter for6d object pose estimation," *Robotics: Science and Systems XV*, 2019.

[12] K. Meshgi, S.-i. Maeda, S. Oba, H. Skibbe, Y.-z. Li, and S. Ishii, "An occlusion-aware particle filter tracker to handle complex and persistent occlusions," *CVIU*, vol. 150, 05 2016.

[13] P. Tokmakov, J. Li, W. Burgard, and A. Gaidon, "Learning to track with object permanence," in *ICCV*, oct 2021, pp. 10 840–10 849.

[14] T. Kloek and H. K. van Dijk, "Bayesian Estimates of Equation System Parameters: An Application of Integration by Monte Carlo," University Rotterdam, Econometric Institute Archives 272139, Nov. 1976.

[15] T. Li, S. Sun, T. Sattar, and J. Corchado Rodríguez, "Fight sample degeneracy and impoverishment in particle filters: A review," *Expert Systems with Applications*, vol. 41, p. 3944–3954, 06 2014.

[16] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *IEEE Transactions on Communication Technology*, vol. 15, no. 1, pp. 52–60, 1967.

[17] M. Moore and A. Meltzoff, "New findings on object permanence: A developmental difference between two types of occlusion," *British Journal of Developmental Psychology*, vol. 17, pp. 623 – 644, 11 1999.

[18] A. Billard, S. Mirrazavi, and N. Figueroa, *Learning for Adaptive and Reactive Robot Control: A Dynamical Systems Approach*, ser. Intelligent Robotics and Autonomous Agents series. MIT Press, 2022.

[19] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, and C.-L. C. et al., "Mediapipe: A framework for perceiving and processing reality," in *Workshop on CV for AR/VR at CVPR*, 2019.