Paparazzo: Active Mapping of Moving 3D Objects

Supplementary Material

A. Additional Details on Paparazzo

This section provides additional technical details on the Paparazzo framework. First, we present the complete formulation of the Extended Kalman Filter (EKF) used for motion prediction, together with the process/measurement noise parameters and the confidence criteria that drive the switch between Object Tracking Mode and Object Mapping Mode (Sec. A.1). Next, we describe the procedure used to accumulate the object point cloud over time (Sec. A.2). Finally, we detail how the dynamic object Gaussian model $\mathcal{G}_O$ is incrementally updated using the current observation and selected past keyframes (Sec. A.3).

A.1. EKF Formulation on $SE(3)$

The EKF used in Paparazzo estimates the object pose in the world reference frame and its body-frame twist:

$$(\, T_k,\ \omega_k \,), \qquad T_k \in SE(3),\ \ \omega_k \in \mathbb{R}^6 \,. \tag{6}$$

Since $SE(3)$ is a nonlinear manifold, the EKF operates on a minimal error state expressed in its tangent space:

$$\delta x_k = \begin{bmatrix} \delta\xi_k \\ \delta\omega_k \end{bmatrix} \in \mathbb{R}^{12}, \qquad \delta\xi_k \in \mathfrak{se}(3),\ \delta\omega_k \in \mathbb{R}^6 \,. \tag{7}$$

Here, $\delta\xi_k$ represents a 6-DoF perturbation of the pose expressed in the Lie algebra $\mathfrak{se}(3)$, while $\delta\omega_k$ encodes the deviation in the object twist. With this representation, the EKF proceeds through the standard prediction and update phases using Lie-group consistent linearizations.

Prediction. Under the constant-twist motion model adopted in Paparazzo, the pose evolves via the exponential map:

$$T_{k|k-1} = T_{k-1}\ \exp\big(\omega_{k-1}\,\Delta t\big)\,. \tag{8}$$

The covariance follows the linearized dynamics:

$$P_{k|k-1} = F_k P_{k-1} F_k^\top + Q\,, \tag{9}$$

where F_k is the Jacobian of the motion model and Q is the block-diagonal process noise matrix defined in Sec. A.1.1. Here, P_{k-1} denotes the posterior covariance at time step $k-1$ and $P_{k|k-1}$ the predicted covariance at time step k before incorporating the new measurement.

Update. Whenever the object is visible, Paparazzo provides an absolute pose estimate $T_{O_k}^{W,\text{meas}} \in SE(3)$, obtained from the pose initialization and the ICP refinement (as described in Sec. 3.4 and detailed in Sec. A.2). The innovation is computed on the Lie algebra:

$$y_k = \log(T_{\text{err}}) \in \mathfrak{se}(3)\,. \tag{10}$$

where $T_{\text{err}} = (T_{k|k-1})^{-1} T_{O_k}^{W,\text{meas}}$.

Since the measurement constrains only the object pose, the Jacobian is

$$H = \begin{bmatrix} I_6 & 0 \end{bmatrix}\,, \tag{11}$$

and the innovation covariance and Kalman gain are respectively

$$S_k = H P_{k|k-1} H^\top + R, \qquad K_k = P_{k|k-1} H^\top S_k^{-1}. \tag{12}$$

The error state and covariance update follow the standard EKF equations:

$$\delta x_k = K_k\, y_k, \qquad P_k = (I - K_k H)\, P_{k|k-1}\,, \tag{13}$$

where $P_k \equiv P_{k|k}$ denotes the posterior covariance after the update at time step k. The nominal state is then corrected by retracting the pose increment on $SE(3)$:

$$T_k = T_{k|k-1}\ \exp(\delta\xi_k)\,, \qquad \omega_k = \omega_{k|k-1} + \delta\omega_k\,. \tag{14}$$

This formulation explicitly shows how the process noise Q, measurement noise R, and innovation y_k contribute to the filtering process.

A.1.1. EKF Parameters and Confidence Criteria

The process noise Q is block–diagonal, separating uncertainty in pose and twist:

$$Q = \begin{bmatrix} Q_t & 0 \\ 0 & Q_\omega \end{bmatrix}, \qquad Q_t = 10^{-4} I_6\,, \tag{15}$$

$$Q_\omega = \text{diag}\big(10^{-4}, 10^{-4}, 10^{-4},\ 5{\times}10^{-4}, 5{\times}10^{-4}, 5{\times}10^{-4}\big).$$

These values are deliberately small: the object moves smoothly from frame to frame and receives frequent and accurate pose corrections from ICP. Slightly larger rotational entries in Q_ω keep the filter responsive to small angular variations while maintaining stability. All parameters were tuned empirically to balance prediction smoothness with robustness against occasional ICP imperfections (e.g., partial views or local registration jitter). The measurement noise covariance is set to $R = 10^{-3} I_6$, modeling the uncertainty of the absolute 6D pose provided by ICP. Since the filter starts with no prior knowledge of the object state, the initial covariance is set to $P_0 = I_{12}$, representing an uninformative prior over pose and twist.

Given these noise models, the key question becomes determining when the EKF is sufficiently reliable to switch from Object Tracking Mode to Object Mapping Mode. To this end, Paparazzo monitors two complementary quantities derived from the EKF state: (i) the uncertainty U_k, and (ii) the innovation consistency metric NIS_k. These values are compared against their respective thresholds, $\tau_u = 0.1$ and $\tau_n = 0.5$, as detailed in Sec. 3.3. In particular, the choice of the NIS threshold is guided by empirical observations. Although a 6D pose measurement would theoretically call for a $\chi^2(6)$ statistical test, in practice the innovations are orders of magnitude smaller. This is a direct consequence of three conditions in our setting: the object exhibits smooth frame-to-frame motion, the process noise is intentionally low to preserve rigid-motion coherence, and the ICP refinement provides accurate local pose corrections. Therefore, classical chi-squared thresholds (e.g., 12.59 at the 95th percentile) are too large to be meaningful. Instead, we adopt a data-driven threshold: during steady motion the NIS remains consistently below $0.1-0.2$, whereas genuine motion changes or ICP inconsistencies yield significantly larger values. Based on this empirical separation, we set $\tau_n = 0.5$. Thus, values above this threshold indicate that the observed motion is no longer compatible with the current state estimate. However, using these thresholds at a single time step is not sufficient, rather, they serve as indicators for assessing long-term EKF stability.

Confidence Condition. The EKF is considered confident when both the uncertainty and innovation remain below their thresholds for $N_s = 4$ consecutive frames:

$$U_k < \tau_u, \qquad \mathrm{NIS}_k < \tau_n \,. \tag{16}$$

This temporal stability requirement prevents spurious mode switches caused by short-term ICP mismatches or partial occlusions. Only after N_s stable frames the system transitions from the Object Tracking Mode to the Object Mapping Mode. Conversely, if innovation quantity exceeds its threshold for N_s consecutive frames, the system reliably detects a genuine motion change and switches back to the reactive Object Tracking Mode.

Table 3 summarizes all EKF parameters.

Table 3. EKF parameters used in all experiments.

Parameter	Value
τ_u (uncertainty threshold)	0.1
τ_n (NIS threshold)	0.5
N_s (stability window)	4
Q_t	$10^{-4} I_6$
Q_ω	$\mathrm{diag}(10^{-4}, 10^{-4}, 10^{-4}, 5{\cdot}10^{-4}, 5{\cdot}10^{-4}, 5{\cdot}10^{-4})$
R	$10^{-3} I_6$
P_0	I_{12}

A.2. Object Point Cloud Accumulation

The final object point cloud $\mathcal{P}$ is expressed in the reference frame of the object at its first detection time t_d. Whenever the object is visible and the mask $\mathcal{M}_k$ is available, we extract the corresponding object point cloud $\mathcal{P}_{O_k}^{C_k}$ in the current camera frame C_k. To bring this point cloud into the object reference frame O_{t_d}, we compute the following initial alignment:

$$\hat{\mathcal{P}}_{O_k}^{O_{t_d}} = T_W^{O_{t_d}} \, T_{C_k}^{W} \, \mathcal{P}_{O_k}^{C_k} \,, \tag{17}$$

where $T_W^{O_{t_d}}$, is the inverse of the rototranslation computed during the initialization phase (Sec. 3.2). However, the point cloud $\hat{\mathcal{P}}_{O_k}^{O_{t_d}}$ does not yet account for the motion that occurred between the detection time t_d and the current step k. As a result, it remains misaligned with the previously accumulated 3D model $\mathcal{P}$. To recover this missing relative motion and correctly integrate the new observations, we apply the alignment procedure described in Sec. 3.4. We first compute a coarse but globally consistent registration using KISS-Matcher, which is robust to outliers and large inter-frame displacements, estimating an initial transformation $\hat{T}_{\text{align},k}$. This estimate is then refined with Colored ICP to obtain the final transformation $T_{\text{align},k}$, that is applied to $\hat{\mathcal{P}}_{O_k}^{O_{t_d}}$ before merging it into the current object model:

$$\mathcal{P} \leftarrow \mathcal{P} \cup (T_{\text{align},k} \, \hat{\mathcal{P}}_{O_k}^{O_{t_d}}). \tag{18}$$

The same alignment also provides the absolute 6D pose measurement used by the EKF

$$T_{O_k}^{W,\text{meas}} = T_{O_{t_d}}^{W} \, T_{\text{align},k} \,. \tag{19}$$

A.3. Gaussian Optimization for the Dynamic Object

Whenever the object is detected and its mask $\mathcal{M}_k$ is available, we update the dynamic Gaussian model $\mathcal{G}_O$. At time k, all Gaussians are expressed in the current object reference frame O_k, which evolves over time according to the relative motions $T_{O_k}^{O_{t_d}}$ obtained during the alignment stage. Before optimizing the model, Paparazzo performs densification, which requires determining which regions of the object surface are actually visible at time k and already represented in $\mathcal{G}_O$. To do this, all Gaussian centers $g^{O_k} \in \mathbb{R}^3$ are first transformed into the current camera frame C_k using the latest estimated object pose:

$$g^{C_k} = T_W^{C_k} \, T_{O_k}^{W,\text{meas}} \, g^{O_k} \,. \tag{20}$$

The projection enables silhouette rendering: pixels in the mask $\mathcal{M}_k$ that are not covered by any projected Gaussian indicate unexplained regions of the object surface, where new Gaussians must be inserted. However, mask-based visibility alone is insufficient. Gaussian centers corresponding

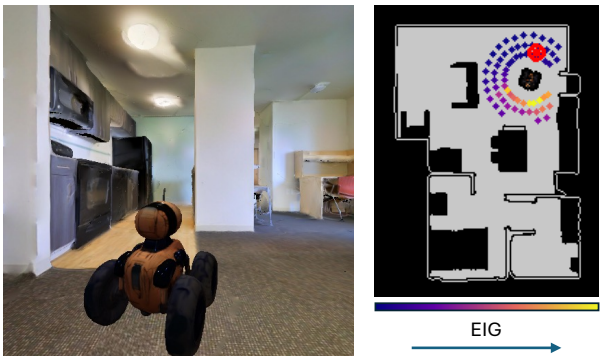

Figure 5. **Generation of candidate viewpoints for Object Mapping Mode.** When the EKF becomes confident, Paparazzo switches from tracking to mapping and evaluates a set of candidate viewpoints $\mathcal{V}$ distributed around the object. The expected information gain (EIG) of each pose, computed using the FisherRF criterion, is visualized here with a color gradient: darker tones correspond to low informativeness, while brighter tones highlight more informative poses for reconstructing the object.

to the back side of the object may still fall inside $\mathcal{M}_k$ when projected, even though they are not physically visible in the current view. To ensure geometric correctness, Paparazzo applies a depth-consistency filter. For each projected Gaussian at pixel u, let $z_{g_k}(u)$ be its depth and $z_k(u)$ the depth observed in the current frame. A Gaussian is considered visible and already accounted for in $\mathcal{G}_O$ only if:

$$\left|z_{g_k}(u) - z_k(u)\right| \leq \tau_d\,, \qquad \tau_d = 0.02\ \text{m}\,. \tag{21}$$

Here, τ_d defines a small but reasonably flexible depth tolerance, which can be relaxed if needed to accommodate larger discrepancies caused by sensor noise and slight registration errors. This constraint ensures that only the physically observable surface contributes to determining which parts of the object require densification.

Keyframe Selection and Optimization For the gaussian model optimization, Paparazzo leverages past keyframes that still observe the same surface region visible at time k. Each past keyframe $i < k$ stores an object point cloud $\mathcal{P}_{O_i}^{C_i}$ extracted at time i. During the accumulation procedure, as described in Sec. A.2, this point cloud is first expressed in the object reference frame O_{t_d} (using Eq. (17)), and then registered via the estimated alignment transformation $T_{\text{align},i}$:

$$\hat{\mathcal{P}}_i = T_{\text{align},i}\, \hat{\mathcal{P}}_{O_i}^{O_{t_d}}\,. \tag{22}$$

To determine whether keyframe i observes the same part of the object currently visible at time k, we transform $\hat{\mathcal{P}}_i$ into the current camera frame C_k:

$$\mathcal{P}_{O_i}^{C_k} = T_W^{C_k}\, T_{O_{t_d}}^{W}\, \hat{\mathcal{P}}_i\,. \tag{23}$$

Then, we reproject $\mathcal{P}_{O_i}^{C_k}$ into the current image plane, obtaining a set of pixel locations $\Omega_{k,i}$. We retain only the pixels that fall inside the current mask:

$$\Omega_{k,i} \leftarrow \Omega_{k,i} \cap \mathcal{M}_k\,. \tag{24}$$

Even in this case, mask agreement alone is insufficient, since points from the back side of the object may still project inside $\mathcal{M}_k$. Therefore, we apply the same depth-consistency filter used during densification: a reprojected point at pixel u is kept only if its depth is consistent with the current observation,

$$\left|\, z_{i\to k}(u) - z_k(u) \,\right| \;\leq\; \tau_d, \tag{25}$$

where $z_{i\to k}(u)$ is the depth of the reprojected 3D point in the camera frame C_k from keyframe i.

Let $\Omega'_{k,i}$ denote the set of points passing both mask and depth checks. We compute the visibility overlap

$$\eta_i = \frac{|\Omega'_{k,i}|}{|\mathcal{M}_k|}\,. \tag{26}$$

Only keyframes with $\eta_i \geq 0.5$ are retained, and at most the top 20 highest-overlap ones are selected for the gaussian optimization. Because the object is rigid, this 3D-aware visibility test ensures that only keyframes observing the same surface region as the current view meaningfully contribute to the update.

We then jointly optimize the Gaussians associated with the selected past keyframes and the current one, using RGB and SSIM losses [14]. This produces a spatially localized, temporally consistent, and computationally efficient refinement of the dynamic object model.

B. Additional Details on Experimental Results

To complement the evaluation presented in Sec. 4, this section provides additional technical details, further qualitative comparisons, and examples of representative agent and object trajectories. These materials are intended to give a deeper understanding of the behavior of Paparazzo and the baseline methods across different scenarios.

Experimental setup and runtime details. All experiments use a 512×512 RGB-D camera with a 90° field of view, consistent with prior active mapping work. Paparazzo runs online at approximately 8 FPS while using at most 4 GB of GPU memory. The EKF update requires at most 5 ms per step, while joint 3DGS refinement and FisherRF evaluation require about 0.1 s per step. The A* planner is not executed at every step; it is triggered intermittently, e.g., when switching from tracking to mapping mode or when a target viewpoint has been reached, and requires at most 0.5 s per invocation. All experiments were conducted on a single NVIDIA GeForce RTX 4090 GPU.

Figure 6. **EKF-based prediction of future object poses.** The EKF predicts the object pose, denoted in orange, over the next N_h steps. The figure shows three examples at steps k+10, k+20, and k+30. These predicted poses are then used to propagate all candidate viewpoints $\mathcal{V}$ into the future, producing a set of $|\mathcal{V}| \times N_h$ future-aligned viewpoints that are subsequently evaluated by the final cost function in Eq. (3).

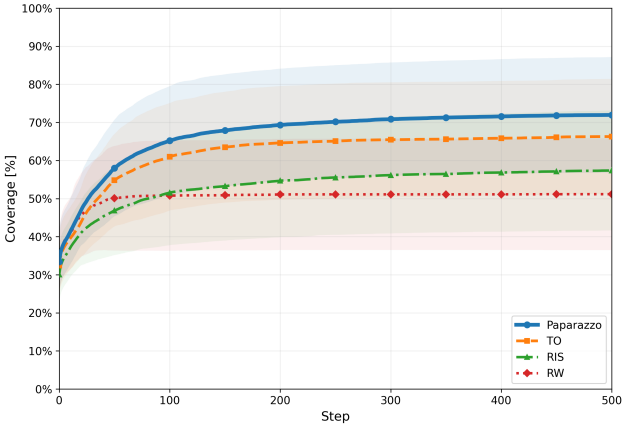

Figure 7. **Coverage over exploration steps.** Results are averaged across all scenes, motion patterns, and objects. Paparazzo consistently achieves higher coverage throughout the entire step budget. Shaded areas indicate the standard deviation across runs.

Paparazzo details. During Object Tracking Mode, the agent keeps the object within the camera's field of view by performing small corrective rotations of $\pm 10°$, as described in Sec. 4. The agent also maintains a distance of 1.5–2.5 m from the object. This emulates a realistic "follow-and-observe" behavior and allows reliable short-term tracking even in compact environments.

To construct the viewpoint candidates $\mathcal{V}$ for Object Mapping Mode, candidate camera poses are sampled around the object using a foveated, object-centered distribution. Camera centers lie on three concentric circular rings on a common horizontal plane, with radii between 1.2 and 1.8 m. Samples are placed every 12° in azimuth, and all camera orientations point toward the object (see Fig. 5). This design provides dense and uniform surface coverage while restricting viewpoints to geometrically feasible and navigable regions of the scene. Each candidate viewpoint is then evaluated using the FisherRF criterion, which quantifies its expected information gain (EIG).

Because the object is dynamic, Paparazzo must reason not only about the informativeness of a viewpoint at the current time but also about how informative it will remain as the object moves. To do this, Paparazzo propagates all candidate viewpoints $\mathcal{V}$ across the next $N_h = 60$ predicted object poses obtained from the confident EKF (see Fig. 6). This results in a set of $|\mathcal{V}| \times N_h$ future-consistent candidate viewpoints, each synchronized with the motion of the object and equipped with its predicted EIG.

Finally, Paparazzo selects the target viewpoint using the cost function defined in Eq. (3), which jointly accounts for information gain, reachability, and temporal synchronization between the agent trajectory and the object's predicted motion. In all experiments, the weights in Eq. (3) are set to $w_{\text{eig}} = 0.8$ and $w_{\text{sync}} = 1.2$, empirically determined on a validation set.

For each run, all methods start from the same object initialization. The object is randomly placed in front of the agent at a distance in the range [1.0, 2.5] m, with a lateral offset in [-0.5, 0.5] m relative to the agent, and a random orientation in [0, 360) around the z-axis of its own reference frame.

B.1. Additional Results

To further visualize the advantages of Paparazzo over the RW, RIS, and TO baselines, Fig. 7 reports the coverage as a function of the exploration steps, averaged across all scenes, motion patterns, and objects. This plot clearly illustrates the benefit of achieving a higher AUC: a method with a larger AUC reconstructs the object more efficiently throughout exploration, rather than only reaching a higher final coverage. As shown in the figure, Paparazzo consistently maintains higher coverage over the entire step budget, confirming its ability to acquire informative viewpoints earlier and more effectively than the baselines.

This quantitative advantage is also reflected in the final reconstruction quality. To complement the coverage-over-time analysis, additional qualitative reconstructions are shown in Fig. 8.

Across all tested objects, Paparazzo produces reconstructions that are both more complete and geometrically more consistent, particularly under challenging motion patterns. This is especially evident in the Stop & Go setting, where baseline methods frequently fail to recover large portions of the object due to motion discontinuities or prolonged static intervals.

In the examples of Object1 and Object2 reported in Fig. 8, Paparazzo achieves substantially higher completeness, reaching 81.23% and 74.45%, respectively. In contrast, all baseline methods exhibit severe missing regions.

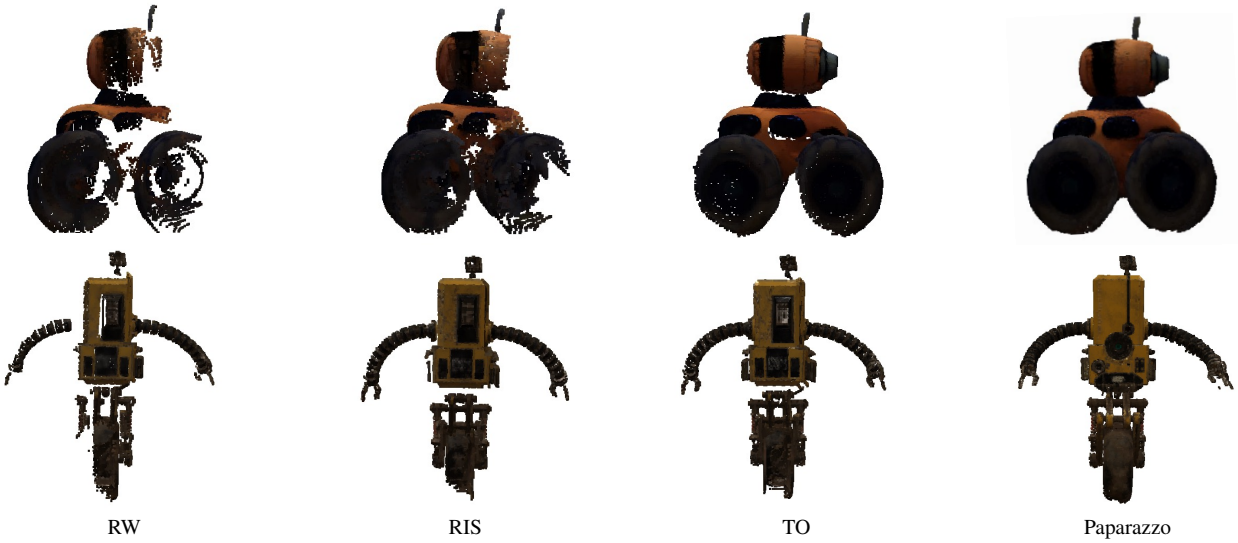

Figure 8. **Qualitative 3D reconstructions of Object 1 and Object 2 under Stop & Go motion.** Paparazzo consistently produces reconstructions that are significantly more complete, coherent, and stable across both objects. In contrast, the baselines, especially on Object 2, fail to recover the entire frontal surface, leaving substantial portions missing.

Notably, for Object 2 the RW, RIS, and TO baselines consistently fail to reconstruct the entire frontal surface of the object, demonstrating their difficulty in handling partial observations and non-smooth trajectories. Paparazzo, instead, successfully recovers these surfaces thanks to its dynamic viewpoint selection and continuous information-driven mapping strategy, which remain effective even when the object stops or follows irregular motions.

The coverage values obtained in this experiment for all methods are summarized in Tab. 4.

Table 4. Reconstruction coverage (%) for Object 1 and Object 2 under the *Stop & Go* motion pattern. Paparazzo significantly outperforms all baselines.

Method	**Object 1**	**Object 2**
RW	43.24	39.42
RIS	51.12	49.23
TO	68.25	58.24
Paparazzo	**81.23**	**74.45**

B.2. Additional Analysis

To provide a deeper understanding of Paparazzo's behavior during reconstruction, in Fig. 9 we show three examples where the object follows a Bouncing Ball (BB) motion across three distinct scenes—Denmark, Greigsville, and Ribera. The black frustum marks the initial camera pose, after which the agent alternates between Object Tracking Mode and Object Mapping Mode, continuously adapting its trajectory to maximize reconstruction quality. These examples highlight the robustness of our approach: Paparazzo consistently produces purposeful and informative motion across different environments and object types, without requiring any object-specific tuning or prior scene knowledge.

For completeness, Fig. 10 presents an example of a Stop & Go motion sequence in the Denmark scene, visualized over the first 300 steps for clarity. The blurred instance of the object marks its initial position, whereas the sharper instance indicates the location where it remained stationary for S=100 steps.

This qualitative comparison highlights a key limitation of the Tracking-Only (TO) baseline (Fig. 10a): although TO operates effectively when the object is continuously moving, it fails when the object stops. In these situations, TO simply remains static, illustrated by the grey camera frustum, waiting passively for motion to resume.

In contrast, Paparazzo (Fig. 10b) continues reconstructing autonomously even during long standstills. It keeps evaluating informative viewpoints, navigating toward them, and acquiring new observations. As a result, Paparazzo maintains reconstruction progress and achieves a more complete and temporally consistent model of the object, despite the absence of motion.

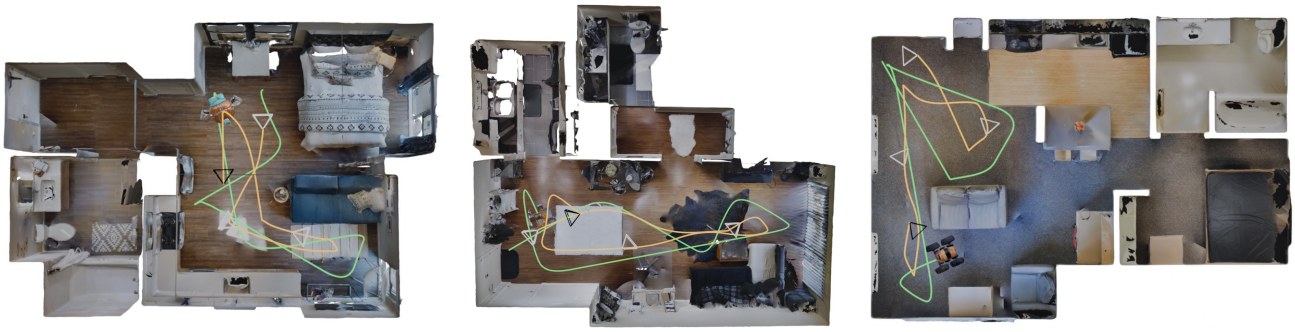

Figure 9. **Examples of object trajectories**. The moving target is performing the Bouncing Ball motion in the Denmark, Greigsville, and Ribera scenes (left to right). The agent executes the Paparazzo framework while continuously adapting its motion to track and map the moving object.

TO Paparazzo

Figure 10. **Illustration of Stop & Go motion.** Both panels show the same Stop & Go trajectory executed by the moving object. Across 300 steps, the object pauses once. (a) TO: the agent remains passive during the stop phase, losing valuable time and collecting no new viewpoints, which prevents further progress in the reconstruction. (b) Paparazzo: the agent continues to actively reposition and capture informative views even while the object is stationary, as visible from the additional exploratory camera frustums (in gray). This allows Paparazzo to maintain reconstruction progress during motion interruptions, unlike the TO baseline.