Supplementary Material

A As-Rigid-As-Possible Volume Tracking

Consider a time-varying mesh sequence, $S = \{M_0, M_1, \ldots, M_{N-1}\}$, where N is the total number of frames in the sequence. Let $M_t = (V_t, E_t, F_t)$ be a static mesh at time t. V_t, E_t, and F_t represent the sets of vertices, edges, and faces (triangles), respectively. To encode long-term structural consistency, we generate a set of K volume centers $C = \{c^1, c^2, \ldots, c^K\}$, where each $c^i \in \mathbb{R}^{3\times N}$ traces the spatial trajectory of a localized volume across all N frames. Here, each center represents a small volume surrounding it with positions that vary over time. To track the volume centers, we adopt As-Rigid-As-Possible Volume Tracking and set the mode to *max affinity based tracking*, as proposed in [Dvořák et al. 2023], to the time-varying mesh sequence S. The method identifies a configurable number of K points set C, referred to as centers.

To generate uniformly distributed centers, which ensure consistent spatial coverage of the mesh and enable stable motion tracking, we first set a resolution based on the longest axis of the mesh and convert each frame into a tightly bounded regular voxel grid. We then compute the fast winding number [Barill et al. 2018] to define an indicator function (IF), which identifies the voxels enclosed by the mesh as follows:

$$IF(x) = \begin{cases} 1 & \omega_f(x) > \mu \\ 0 & \omega_f(x) \le \mu \end{cases}, \tag{14}$$

where $\omega_f(x)$ represents the fast winding number calculated from point x, and μ is a threshold. In this paper, we set $\mu = 0.5$. The algorithm then samples K random voxels with $IF(x) = 1$ and designates them as centers. Each of the remaining voxels is then connected with its nearest center, and every center is moved to its surrounding neighbors' centroid iteratively. This process yields a uniformly distributed set of centers in the first frame. For subsequent frames, the previous frame's centers are linearly extrapolated, followed by energy-based optimization to maintain uniformity and temporal smoothness, following the approach in [Dvořák et al. 2023, 2022].

B Ablation study on the impact of volume centers

We conduct ablation experiments to evaluate the impact of the volume center counts, as shown in Figure 10. The quality of the reconstructed mesh does not always improve with a higher number of centers. More centers may introduce more challenges for MDS in generating a stable set of reference centers. When the number of centers increases, MDS must consider a larger number of pairwise distances, resulting in higher computational costs and potential errors in the optimization process. With an excessive number of centers, MDS-generated reference centers are more likely to contain abnormal outputs, as shown by the red arrow and dotted circle in Figure 10c and Figure 10d, which significantly reduces TSMC's overall performance.

C Comparison of SAM3-based Scene Decomposition

In the main paper, we introduce a SAM3-based scene decomposition pipeline. To further evaluate its effectiveness, we conduct an ablation study comparing our image-based approach with a geometry-based frame differencing method. The geometry-based baseline integrates block-wise vertex occupancy changes, normal-similarity checks, and polygon-count filtering to detect static versus dynamic components. A detailed description of this implementation is provided below.

Vertex occupancy. We first generate an axis-aligned bounding box of the mesh, then divide the mesh into blocks, B_t^j represents the j-th block in the t-th frame. Then for each mesh, an occupancy map is built, associating block indices with the vertices' positions they contain. For example, for meshes M_0 and M_t, the vertices in the corresponding blocks are compared. Given two sets of vertices V_0^j and V_t^j in block B^j at time 0 and t, the ratio r of unmatched vertices is computed using the following equations:

$$r = \frac{1}{|V_0^j|} \sum_{p \in V_0^j} \delta(p, V_t^j), \tag{15}$$

where $|V_0^j|$ is the number of vertices in block B_0^j and $\delta(p, V_t^j)$ is and indicator function:

$$\delta(p, V_t^j) = \begin{cases} 1 & \text{if} \min_{q \in V_t^j} |p - q| > \mu, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

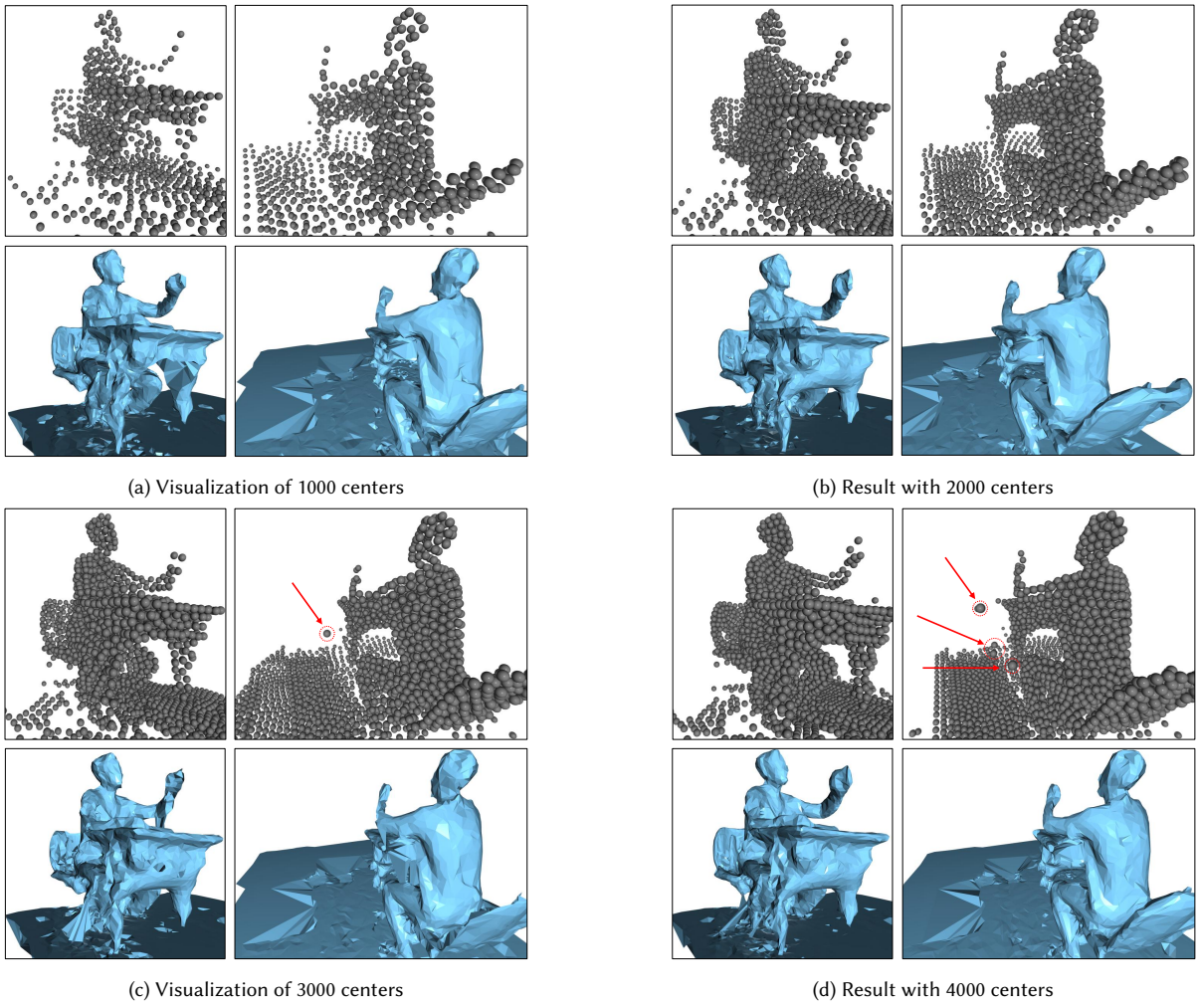
The block B^j is classified as a dynamic block if $r > \nu$, where μ and ν are two thresholds. The mesh is then divided into two parts, dynamic parts and static parts based on the information of dynamic and static blocks.

Normal-based reclassification. While vertex occupancy detects most motion, we observed that sensor noise in real-world scenes, particularly those captured using depth sensors, often causes large, static planar regions such as walls, floors, and ceilings to be incorrectly marked as dynamic. This misclassification degrades the performance of volume tracking by introducing unnecessary complexity. To correct this, we analyze the normal similarity of each connected facet (i.e., a set of adjacent triangles forming a locally connected surface patch) in the initially identified dynamic regions. For each connected facet $\hat{M}^{(m)} = (V^{(m)}, F^{(m)})$, where $V^{(m)}$ is the set of vertices and $F^{(m)}$ is the set of faces (triangles) of the facet, let $N^{(m)}$ denote the set of triangle normals associated with $F^{(m)}$. Let θ represent the largest angle between these normals:

$$\theta = \max_{\mathbf{N}_i, \mathbf{N}_j \in N^{(m)}} \arccos(|\mathbf{N}_i \cdot \mathbf{N}_j|), \tag{17}$$

where $\mathbf{N}_i$ and $\mathbf{N}_j$ are the normal vectors of corresponding triangles, and the dot product represents the cosine of the angle between these normals. If $\theta < \omega$, the connected facet is considered approximately planar and is reclassified as part of the static region. This step is crucial for eliminating flat surfaces with consistently oriented normals, which interfere with reliable volume tracking due to their lack of distinguishable motion.

Pruning spurious islands with poly count. While the previous steps isolate meaningful dynamic regions and exclude large planar surfaces, real-world meshes reconstructed from point clouds often include small, disconnected patches of geometry—commonly referred to as mesh islands. These islands typically arise from scattered vertices and may form curved surfaces or spherical fragments after Poisson reconstruction, contributing little semantic value and introducing compression artifacts. To mitigate this, we remove isolated



(a) Visualization of 1000 centers

(b) Result with 2000 centers

(c) Visualization of 3000 centers

(d) Result with 4000 centers

Fig. 10. Impact of different numbers of centers for volume tracking. Volume centers are represented with black spheres, and their corresponding reconstructed meshes are drawn in blue. Examples of abnormal reference centers are marked using the red arrow and dotted circle.

connected components whose polygon count falls below a predefined threshold.

Figure 11 shows a qualitative comparison of the two approaches. While the geometry-based method can produce reasonable decompositions, it inevitably inaccurately preserves extra parts such as floor surfaces (e.g., for "Answering" and "Arena") or captures only a subset of a dynamic object (e.g., for "Drinking" and "Sitting"), since portions of the object may remain static across frames. In contrast, our SAM3-based approach provides more semantic dynamic object boundaries over the sequence, while maintaining low computational cost.

D Additional Implementation Details

All the experiments are conducted using a Lambda Vector equipped with an AMD Ryzen Threadripper 7960X 24-core 4.20 GHz CPU and an RTX 4090 GPU with 24 GB of memory.

In both TSMC and TVMC*, we use Draco to compress the generated volume-tracked reference mesh at relatively high quality, with settings of $qp = 14, cl = 7$. We vary the number of basis vectors m from 3 to $3GoF$ to explore a range of bitrates in TSMC's displacement compression module. Draco uses the parameter qp to control the level of geometric information quantization. We vary qp from 2 to 17 in our Draco experiments. And store displacements in *PointClouds* and compress them with the setting of *pointcloud* and the qp from 7 to 17 to achieve the target bitrates. For the TSDF and KLT-based compression method [Tang et al. 2018], we vary the TSDF resolution of 64, 128, and 256, and use 8, 16, and 32 basis vectors. For NeCGS we also vary the TSDF resolution of 64, 128, and 256, along with embedded feature resolution of 2, 4, and 8. We do not further increase the TSDF or embedded feature resolution, as doing so would exceed the available 24 GB of GPU memory.

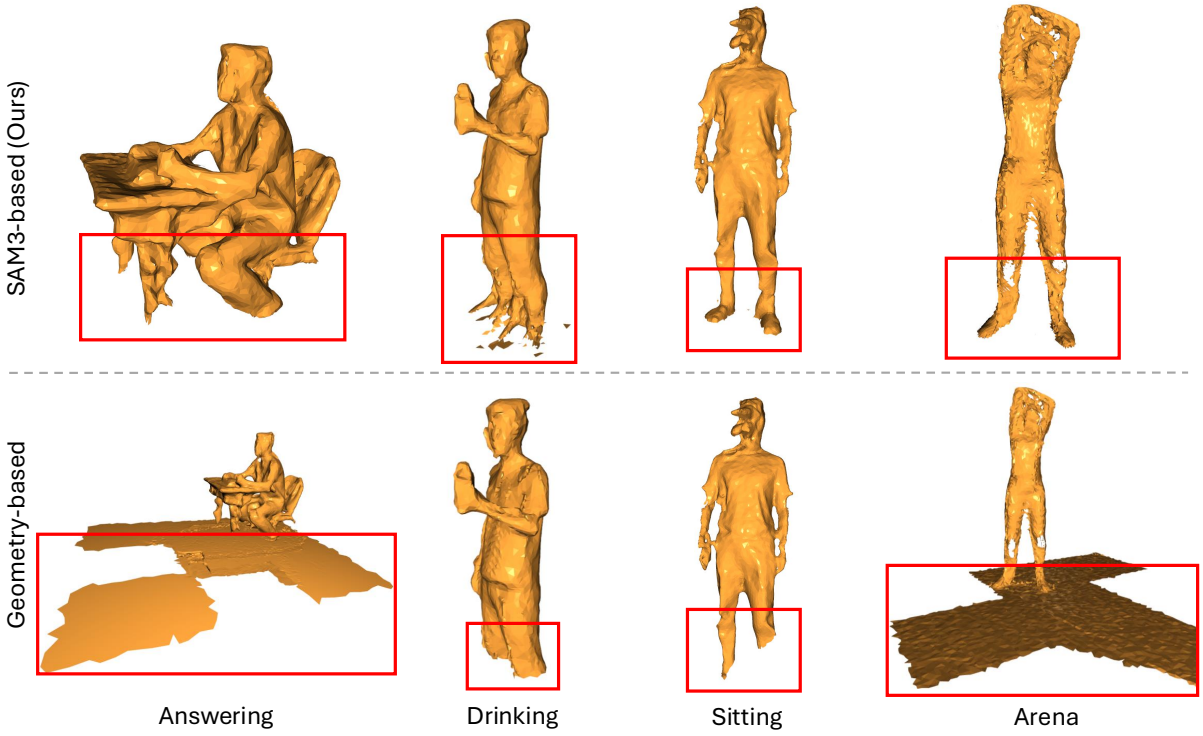


Fig. 11. Volume tracking results on the entire "Arena" scene mesh. Gray sphere dots represent the estimated volume centers, and the gray surface represents the scene mesh. Directly applying volume tracking to the scene mesh fails, as the reference centers cannot correctly represent the region enclosed by the mesh surface due to the winding number issue.

E Limitations and Future Work

TSMC relies on a reconstructed reference mesh obtained through volumetric fusion and tracking. This step inevitably introduces geometric smoothing and temporal inconsistencies, making the overall codec inherently lossy. Consequently, TSMC cannot support true lossless compression as achieved by methods such as Draco.

The volume-tracking stage requires several seconds per frame, which limits TSMC to real-time decoding but prevents real-time encoding. This makes TSMC suitable for on-demand volumetric video streaming where encoding is performed offline, but not for ultra-low-latency capture-and-stream or live streaming pipelines. One of our future works could be accelerating the volume tracking using GPUs to reduce the encoding time.

Currently, TSMC focuses on geometry compression, we leave extending TSMC to jointly encode geometry with textures for future work.