

Figure 6: Structure overview of Q-SLAM. 1) Tracking: initialize per-frame camera poses and depth prediction. Correct the noisy depth using our proposed depth correction module based on the segmentation results from monocular inputs. 2) NeRF: using the selected keyframes to supervise the optimization of NeRF network equipped with our proposed quadric-decomposed transformer. 3) Mapping: global bundle adjustment to jointly optimize the scene representation and camera poses taking rays sampled from all keyframes. Reconstruct the complete scene by fusing the rendered RGB images and depth maps with TSDF-fusion [33].

## 404 A Additional Methodology Details

### 405 A.1 Quadric surface fitting and depth correction

406 By setting  $\nabla \mathbb{C}_c = 0$  in Eq. 2 to obtain the optimal  $c^*$

$$c^* = \frac{1}{N} \sum_{i=1}^N (\mathcal{C}_q \cdot \mathbf{q}_i + \mathcal{C}_l \cdot \mathbf{x}_i) \triangleq \mathcal{C}_q \cdot \bar{\mathbf{q}} + \mathcal{C}_l \cdot \bar{\mathbf{x}} \quad (9)$$

407 The cost function in Eq. 2 becomes:

$$\mathbb{C} = \sum_{i=1}^N (\mathcal{C}_q \cdot (\mathbf{q}_i - \bar{\mathbf{q}}) + \mathcal{C}_l \cdot (\mathbf{x}_i - \bar{\mathbf{x}}))^2 \quad (10)$$

408 where  $\bar{\mathbf{q}} = \frac{1}{N} \sum_{i=1}^N \mathbf{q}_i$  are the quadric term averaged on points in a patch,  $\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$  is the  
 409 linear term.

410 The intermediate variables are defined as follows:

$$\begin{aligned} \mathbb{L} &\triangleq \sum_{i=1}^N (\mathbf{x}_i - \bar{\mathbf{x}}) (\mathbf{x}_i - \bar{\mathbf{x}})^T \\ \mathbb{M} &\triangleq \sum_{i=1}^N (\mathbf{q}_i - \bar{\mathbf{q}}) (\mathbf{q}_i - \bar{\mathbf{q}})^T \\ \mathbb{N} &\triangleq - \sum_{i=1}^N (\mathbf{q}_i - \bar{\mathbf{q}}) (\mathbf{x}_i - \bar{\mathbf{x}})^T \end{aligned} \quad (11)$$

411 Setting  $\nabla \mathbb{C}_{\mathcal{C}_l} = 0$  gives

$$\mathbb{L} \mathcal{C}_l^* = \mathbb{N}^T \mathcal{C}_q \quad (12)$$

412 By substituting  $\mathcal{C}_l^*$  back to Eq. 10, we can obtain

$$\begin{aligned}\mathbb{C} &\triangleq \sum_{i=1}^N \left\| \left( (\mathbf{q}_i - \bar{\mathbf{q}})^T + (\mathbf{x}_i - \bar{\mathbf{x}})^T \mathbb{L}^{-1} \mathbb{N}^T \right) \mathcal{C}_q \right\|^2 \\ &= \mathcal{C}_q^T \Psi \mathcal{C}_q, \text{ where } \Psi \triangleq \mathbb{M} - \mathbb{N} \mathbb{L}^{-1} \mathbb{N}^T\end{aligned}\quad (13)$$

413 Minimizing Eq. 13 over  $\mathcal{C}_q$  gives the eigenvector  $\mathcal{C}_q^*$  of  $\Psi$  corresponding to the minimum eigenvalue,  
414 and  $\mathcal{C}_l^*$  can be solved from Eq. 12, and  $c^*$  from Eq. 9.

415 As defined by Taubin *et al.* [44], the distance from a point  $\mathbf{x}$  to a quadric surface  $f$  is:

$$d(\mathbf{x}, f) \approx \frac{f^2(\mathcal{C}, \mathbf{x})}{|\nabla_{\mathbf{x}} f(\mathcal{C}, \mathbf{x})|^2} \quad (14)$$

416 For every fitted patch, we calculate the average distance between the original points to the fitted  
417 surface as the fitting error. Those patches with error exceeding the given threshold will be discarded.  
418 We only preserve the patches with relatively small fitting error, which implies a good fitting surface  
419 for the following depth correction.

## 420 A.2 Ray points sampling

421 Initially, rays are constructed from the provided image and calibration matrix, and  $N_s$  points are  
422 uniformly sampled along each ray within the range  $[d_{near}, d_{far}]$ . Utilizing the corrected depth  
423 values, denoted as  $d$ , we refine the sampling process by selecting additional  $N_d$  samples within the  
424 range of  $[0.95d, 1.05d]$ . We do not simply sample around  $d$  because potential errors in the corrected  
425 depth values might lead to an extended sampling distance away from the true surface.

## 426 A.3 Dynamic branch design

427 Following SUDS [45], we design a simplified dynamic branch and calculate the final radiance fields  
428 of moving objects as follows:

$$\sigma(\mathbf{x}, t) = \sigma_s(\mathbf{x}) + \sigma_d(\mathbf{x}, t), \quad \mathbf{c}(\mathbf{x}, \mathbf{d}, t) = \frac{\sigma_s}{\sigma} \mathbf{c}_s(\mathbf{x}, \mathbf{d}) + \frac{\sigma_d}{\sigma} \mathbf{c}_d(\mathbf{x}, \mathbf{d}, t) \quad (15)$$

429 where  $\mathbf{x}$  and  $\mathbf{d}$  represents the 3D coordinates and viewing direction respectively, and  $t$  is the times-  
430 tamp. The subscript  $s$  and  $d$  stand for static and dynamic branch respectively, and the outputs of  
431 dynamic branch depend on time, while the static branch does not.

432 The color  $\hat{\mathbf{C}}$  and depth  $\hat{D}$  are rendered as follows:

$$\hat{\mathbf{C}}(\mathbf{r}, t) = \int_0^{+\infty} T(s) \sigma(\mathbf{r}(s), t) \mathbf{c}(\mathbf{r}(s), \mathbf{d}, t) ds \quad (16)$$

$$\hat{D}(\mathbf{r}, t) = \int_0^{+\infty} T(s) \sigma(\mathbf{r}(s), t) ds \quad (17)$$

434 We do not incorporate the semantic head for outdoor scenes, because there are much more classes  
435 of objects compared to indoor scenes. To supervise the training on dynamic objects, we generate the  
436 static masks following [45], and apply a regularization loss of dynamic branch on the static regions.

$$\mathcal{L}_r = \sum_{\mathbf{x} \in \text{static}} |\sigma_d(\mathbf{x}, t)|_1 \quad (18)$$

## 437 B Additional Implementation Details

### 438 B.1 Data source

439 The data of other NeRF-SLAM methods in Tab. 1 is sourced from Nicer-SLAM [43], and the  
440 geometric reconstruction results (Acc., Comp., etc.) of SplatAM [14] comes from RTG-SLAM

441 since the original paper does not report these metrics. The results of other method in Tab. 2 are  
 442 mainly taken from GO-SLAM [2].

## 443 B.2 Dataset

444 Q-SLAM is evaluated on a variety of datasets, including Replica [36], ScanNet [37], and TUM  
 445 RGB-D [46] dataset. For evaluation of reconstruction quality, we test our method on 8 synthetic  
 446 scenes from Replica, which provides high-quality synthetic scenes, akin to the evaluation frame-  
 447 work adopted by NeRF-SLAM [4]. Following GO-SLAM [2], we evaluate the tracking accuracy  
 448 on ScanNet dataset which offers extensively annotated RGB-D scans of real-world scenarios, en-  
 449 compassing challenging short and long trajectories. Following Nice-SLAM [8], we also evaluate  
 450 on various scenes on indoor TUM RGB-D dataset, with ground truth poses provided by a motion  
 451 capture system. For camera tracking assessment, our approach is tested under two distinct modes:  
 452 one utilizing ground truth and the other utilizing estimated depth from monocular images as inputs.  
 453 The batch size of sampled rays to NeRF network is 8192.

## 454 B.3 Evaluation Metrics

455 We evaluate tracking accuracy by aligning the estimated trajectory with the ground truth trajectory  
 456 and computing the Root Mean Square Error (RMSE) of the Absolute Trajectory Error (ATE). This  
 457 metric quantifies the Euclidean distance between the estimated pose and the corresponding ground  
 458 truth pose. In line with the evaluation approach of NeRF-SLAM [4], we utilize Peak Signal-to-  
 459 Noise Ratio (PSNR), SSIM [39], and LPIPS [40] for image rendering evaluation, and Accuracy  
 460 [cm], Completion [cm], Completion Ratio [%] for 3D reconstruction assessment.

- 461 • Absolute Trajectory Error (ATE) (cm) ↓: Evaluates trajectory estimation accuracy by mea-  
 462 suring the average Euclidean translation distance between corresponding poses in estimated  
 463 and ground truth trajectories, often reported in terms of Root Mean Square Error (RMSE).
- 464 • Peak Signal to Noise Ratio (PSNR) ↑: Measures image quality by evaluating the ratio  
 465 between the maximum pixel value and the root mean squared error, usually expressed in  
 466 terms of the logarithmic decibel scale.
- 467 • Structural Similarity Index Measure (SSIM) ↑: Assesses image quality by examining the  
 468 similarities in luminance, contrast, and structural information among patches of pixels.
- 469 • Learned Perceptual Image Patch Similarity (LPIPS) ↓: Utilizes learned convolutional fea-  
 470 tures to assess image quality based on feature map mean squared error across layers.
- 471 • Accuracy (cm) ↓: Computes the average distance between sampled points from the recon-  
 472 structed mesh and the nearest ground-truth point.
- 473 • Completion (cm) ↓: Measures the average distance between sampled points from the  
 474 ground-truth mesh and the nearest reconstructed.
- 475 • Completion Ratio (%) ↑: the percentage of points in the reconstructed mesh with Comple-  
 476 tion under 5 cm.

## 477 B.4 Hyperparameters

478 All experiments are conducted on NVIDIA A6000 GPU with PyTorch 1.10.0. We use Adam as  
 479 our optimizer with  $\beta_1 = 0.9, \beta_2 = 0.999$ . The tracking backbone is Droid-SLAM, where we use  
 480 the pretrained weights to estimate depths and poses. We use TensorRF as the mapping backbone,  
 481 equipped with our proposed depth correction and quadric transformer. The threshold for motion  
 482 filter is 4.0 pixels, a tracked frame is considered as a keyframe only if the average optical flow is  
 483 greater than the threshold. The window size for local bundle adjustment is 25. During the joint  
 484 optimization process, camera poses are optimized for one epoch, and the NeRF network parameters  
 485 are optimized for five epochs.

## 486 B.5 Segmentation and Quadric Fitting

487 For the segmentation network, we use Segment Anything Model (SAM) [47], an off-the-shelf net-  
 488 work to produce the mask for quadric fitting. To prevent the negative effect of outliers, quadric  
 489 fitting only applies to segments with area larger than 200 pixels.

490 During the fitting process, we calculate the coefficient of determination to evaluate the fitting per-  
 491 formance. Let  $z_i$  be the predicted depth value and  $f_i$  be the corresponding corrected depth. The  
 492 coefficient of determination is calculated as follows:

$$\begin{aligned}\bar{z} &= \frac{1}{n} \sum_{i=1}^n z_i \\ SS_{\text{res}} &= \sum_i (z_i - f_i)^2 \\ SS_{\text{tot}} &= \sum_i (z_i - \bar{z})^2 \\ R^2 &= 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}\end{aligned}$$

493 We only perform depth correction on quadric surfaces with fitting coefficient greater than the given  
 494 threshold 0.85, otherwise, we just use the predicted depth for the supervision of NeRF training.

## 495 B.6 Mesh Reconstruction

496 Different from other approaches that reconstruct mesh of a scene by running marching cubes on the  
 497 Signed Distance Function (SDF) values of the queried points, we render images and depths for the  
 498 selected keyframes. The reason for the difference is that our rendering requires to correlate points  
 499 along and across rays, while other approaches process 3D points independently. We first render  
 500 RGB images and depth maps, and then use TSDF-fusion [33] to reconstruct the 3D volume mesh.

## 501 C Additional Experimental Results

502 **TUM-RGBD dataset.** We evaluate the tracking performance of our methods on the small-scale  
 503 indoor-scene dataset with two different inputs, monocular and RGBD images. As presented in Ta-  
 504 ble 5, our approach outperforms traditional SLAM, including ORB-SLAM2 [48] and ORB-SLAM3  
 505 [49], which exhibits failures in certain scenarios. In comparison to recent NeRF-based SLAM sys-  
 506 tems, our solution consistently achieves superior results across most scenes. We attribute the im-  
 507 provements to our proposed quadric representation and quadric transformer, especially for scenes  
 508 with well-segmented planes and surfaces such as desks, floors, and rooms.

Table 5: ATE RMSE [m] Results on TUM [46] dataset freiburg1 set (monocular setting). ORB-SLAM2 [48] and ORB-SLAM3 [49] fail on certain scenes.

	360	desk	desk2	floor	plant	room	rpy	teddy	xyz	avg
ORB-SLAM2 [48]	-	0.071	-	0.023	-	-	-	-	0.010	-
ORB-SLAM3 [49]	-	0.017	0.210	-	0.034	-	-	-	0.009	-
DeepV2D [50]	0.243	0.166	0.379	1.653	0.203	0.246	0.105	0.316	0.064	0.375
DeepFactors [51]	0.159	0.170	0.253	0.169	0.305	0.364	0.043	0.601	0.035	0.233
DROID-SLAM [16]	0.111	0.018	0.042	<b>0.021</b>	<b>0.016</b>	0.049	0.026	<b>0.048</b>	0.012	0.038
GO-SLAM[2]	0.089	0.016	0.028	0.025	0.026	0.052	<b>0.019</b>	<b>0.048</b>	0.010	0.035
Ours	<b>0.086</b>	<b>0.013</b>	<b>0.023</b>	0.026	0.027	<b>0.049</b>	0.021	0.049	<b>0.009</b>	<b>0.033</b>

509 Following GO-SLAM [2], we also test our solution with RGBD images as input, as indicated in  
 510 Table 6. While the quadric-based depth correction is not performed under this setting, our proposed  
 quadric ray transformer and semantic supervision also contribute to the performance improvement.

Table 6: ATE [m] Results on TUM dataset [46] with RGB-D inputs from freiburg1, freiburg2 and freiburg3 set.

Method	fr1/desk	fr2/xyz	fr3/office
Kintinuous [52]	0.037	0.029	0.030
BAD-SLAM [53]	0.017	0.011	0.017
ORB-SLAM2 [48]	0.016	<b>0.004</b>	<b>0.010</b>
iMAP [5]	0.049	0.020	0.058
NICE-SLAM [8]	0.027	0.018	0.030
Ours	<b>0.014</b>	0.005	0.011

511

## 512 D Visualization

513 In Fig. 7, we provide the qualitative results of the reconstruction. It can be observed that our method  
 514 outperforms GO-SLAM, especially on the boundaries of objects.

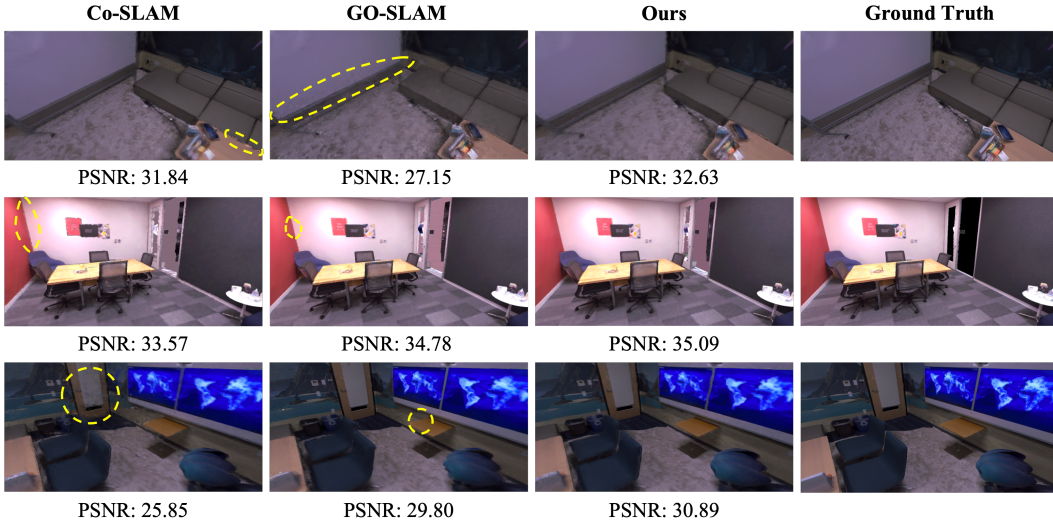


Figure 7: Qualitative reconstruction results on Replica dataset. We compare our solution with recent SOTA SLAM systems Co-SLAM [1] and GO-SLAM [2]. Our method can recover better texture features, especially on the boundary of instances.

515 We provide several selected visualization results for depth correction as shown in Fig. 8. Benefiting  
 516 from the segmentation mask, the depth correction improves the sharpness of the boundary of objects.

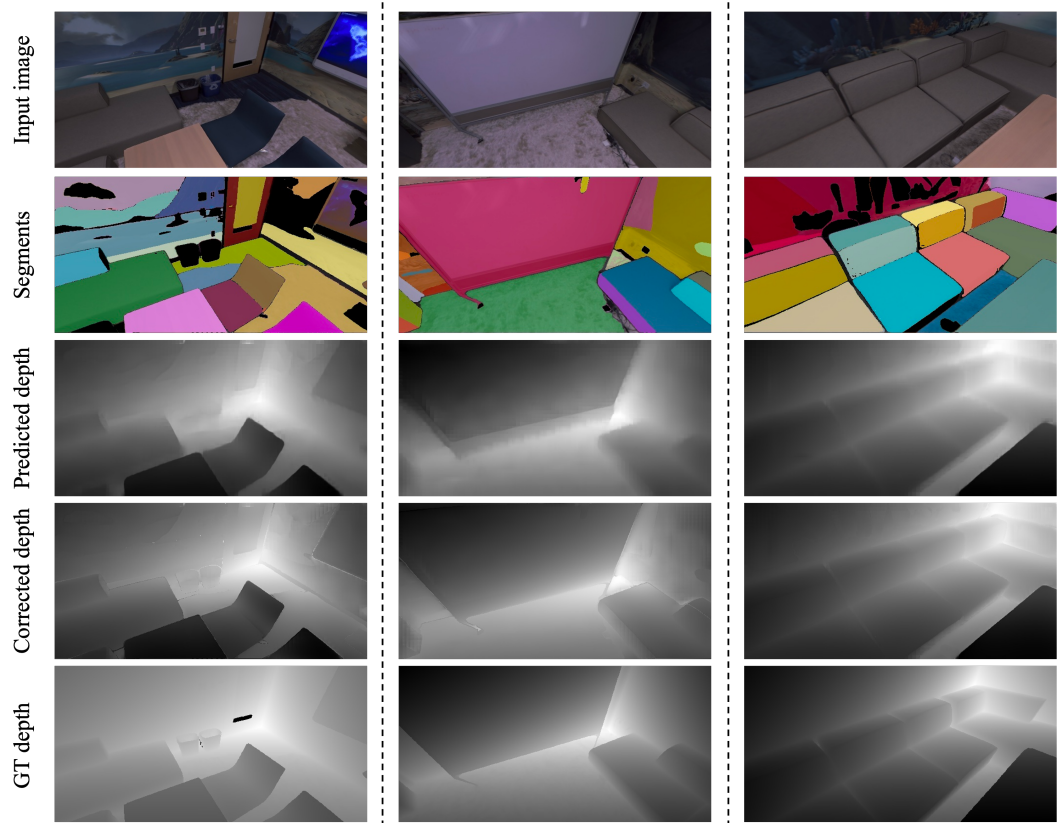


Figure 8: Qualitative results of depth correction.