

# Unleashing the Power of Data Synthesis in Visual Localization

Sihang Li\* Siqi Tan\* Bowen Chang Jing Zhang Chen Feng<sup>†</sup> Yiming Li<sup>†</sup>  
 New York University  
 {sihangli, tan.k, cfeng, yimingli}@nyu.edu  
<https://ai4ce.github.io/RAP>

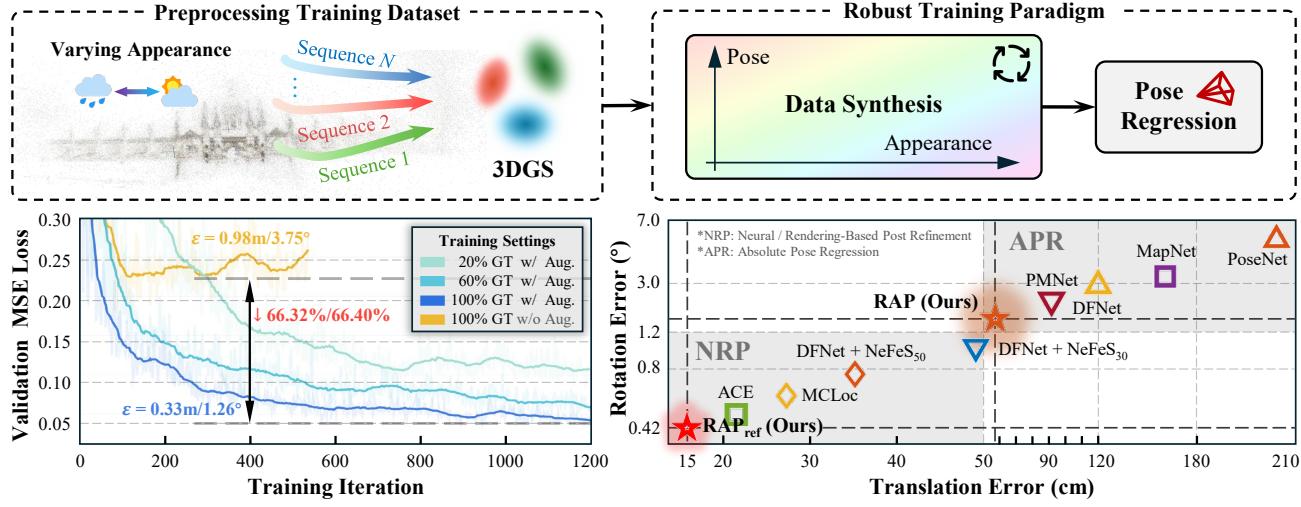


Figure 1. To achieve robust and generalizable pose regression, we lift real-world 2D images into 3D Gaussian Splats [23] as the data engine to synthesize many more posed images (**top-left**). A robust training paradigm is proposed to learn pose regression from real and synthetic data (**top-right**). Quantitative experiments validate the state-of-the-art performance of our method (**bottom-right**). With our data synthesis and robust training, the translation and rotation errors are respectively reduced by 66.32% and 66.40% (**bottom-left**).

## Abstract

Visual localization, which estimates a camera’s pose within a known scene, is a long-standing challenge in vision and robotics. Recent end-to-end methods that directly regress camera poses from query images have gained attention for fast inference. However, existing methods often struggle to generalize to unseen views. In this work, we aim to unleash the power of data synthesis to promote the generalizability of pose regression. Specifically, we lift real 2D images into 3D Gaussian Splats with varying appearance and deblurring abilities, which are then used as a data engine to synthesize more posed images. To fully leverage the synthetic data, we build a two-branch joint training pipeline, with an adversarial discriminator to bridge the syn-to-real gap. Experiments on established benchmarks show that our method outperforms state-of-the-art end-to-

end approaches, reducing translation and rotation errors by 50% and 21.6% on indoor datasets, and 35.56% and 38.7% on outdoor datasets. We also validate the effectiveness of our method in dynamic driving scenarios under varying weather conditions. Notably, as data synthesis scales up, our method exhibits a growing ability to interpolate and extrapolate training data for localizing unseen views.

## 1. Introduction

Visual localization, the task of calculating a 6-DoF camera pose—its translation and rotation—based on a query image within a given environment, is essential for wide applications, including robotics [1], autonomous vehicles [19], and virtual reality [11]. Besides traditional geometry-based approaches, recent learning-based visual localization methods either adopt scene coordinate regression (SCR) [4, 5, 40, 59] or absolute pose regression (APR) [6, 9, 22, 51]. SCR methods focus on learning-based 2D-3D correspondences followed by subsequent Perspective-n-Point (PnP) for pose

\*Equal contribution.

<sup>†</sup>Corresponding author.

estimation. In contrast, APR methods employ a supervised framework to train a regression neural network on image-pose pairs, enabling direct pose estimation during inference. APR offers the advantage of faster runtime and better performance in challenging conditions, such as images with low texture or repetitive patterns [30], making it a promising approach for advancing visual localization.

Despite the promises, a well-known pivotal work [47] discovered that existing APR seemed to perform image-based *memorization*, i.e., retrieving poses seen during training, more than *generalization*, i.e., interpolating between or extrapolating beyond training poses. Driven by this crucial finding, to improve such memorization while avoiding the need for denser real-world training samples, recent methods leverage Neural Radiance Field (NeRF) [37] to synthesize additional posed images for APR training [9, 30, 39]. However, a critical question remains: *why do previous APR methods struggle to generalize effectively?* Deep neural networks show impressive generalization in many other tasks [29, 60, 61, 63], and humans can infer poses from new viewpoints in known environments. So, what is missing? We hypothesize two key gaps: (1) *Data Gap*: Prior methods lacked sufficient diversity and quality in synthesized data to learn a robust feature space. (2) *Learning Gap*: Limitations in previous learning pipelines prevented effective use of large-scale data to enhance generalizability.

To address the two key gaps, we propose a novel robust training framework for **absolute pose regression (RAP)**, as shown in Fig. 1. We address the *data gap* by adopting and extending 3D Gaussian Splatting (3DGS) [23] to allow efficient and diverse data synthesis with controllable varying appearance. We address the *learning gap* by designing a two-branch joint training. The first branch coarsely trains our Transformer-based APR with both real data and data synthesized at the original real pose, together with an adversarial discriminator to reduce the syn-to-real domain gap. The second branch progressively unleashes the power of data synthesis with randomly perturbed poses and appearances, providing additional supervision to the same APR Transformer. We systematically evaluate each component’s effect and analyze how the synthesized data endows our APR model with better generalizability in pose regression.

Through extensive experiments, we demonstrate that scaling up the training data significantly enhances representation learning in APR. Using only 20% real training data with our data synthesis already leads to a representation with better APR performance than using all real data, as shown in Fig. 1. Meanwhile, our results indicate that APR consistently benefits from more diverse visual data, and *we observe clear signs of a generalizable APR emerging as the scale of data synthesis increases, and its localization performance cannot be explained merely by memorization*. Our contributions are summarized as follows:

- We identify research gaps in studying APR model generalizability and demonstrate through comprehensive experiments that large-scale data synthesis with effective learning strategies makes APR more generalizable.
- We leverage 3D Gaussian Splats as our data engine to efficiently synthesize novel views with new appearances.
- We propose a robust two-branch joint training pipeline with an adversarial discriminator to reduce the syn-to-real gap and fully unleash the power of synthetic data.
- Our method sets a new state-of-the-art in visual localization on real-world indoor, outdoor, and driving datasets.

## 2. Related Works

**Visual Localization.** Visual localization aims to estimate a camera’s translation and rotation within a 3D scene. Traditional geometry-based methods [7, 15, 31, 33, 42, 44–46, 55] accomplish this by using point clouds and a reference image database, relying on stored descriptor and image retrieval to establish 2D-3D correspondences. In contrast, scene coordinate regression (SCR) methods [3–5, 59] embed map information within neural networks to directly predict 2D-3D correspondences. Both approaches generally require PnP [17] and RANSAC [16] to output camera poses at test time, which adds additional computation cost. Alternatively, absolute pose regression (APR) [6, 8, 20, 22, 38, 50] aims to directly regress the camera pose from a query image using neural networks. Although the performance is suboptimal compared with geometry-based methods, APR remains a promising approach due to its fast inference.

**Data Augmentation for Pose Regression.** End-to-end pose regression methods rely heavily on the amount and diversity of training data. Previous work [47] shows that APR implicitly learns image *memorization* in the given environment, actually overfitting to the training set. Therefore, the following works LENS [39], DFNet [9] and PM-Net [30] enhance APR performance by enriching training views with NeRF. However, these approaches fail to address the generalizability of APR models and exhibit several limitations: (1) The efficiency of training and novel view synthesis (NVS) in NeRF is severely restricted, hindering scalability. (2) They limit NVS to geometric (pose) transformations while neglecting photometric (appearance) variations, thereby decreasing APR robustness to changes in visual appearance. (3) The augmented data is underutilized in their learning frameworks, leaving its potential for improving APR largely untapped. *Differently, our framework switches to 3DGS [23] as the scene representation to efficiently generate novel posed images with arbitrarily blended appearances and introduce an adversarial mechanism to unleash the power of synthetic data.*

**Handling and Synthesizing Challenging Scenarios.** Visual localization often encounters unstructured photo collections [54], where visual appearance varies due to mov-

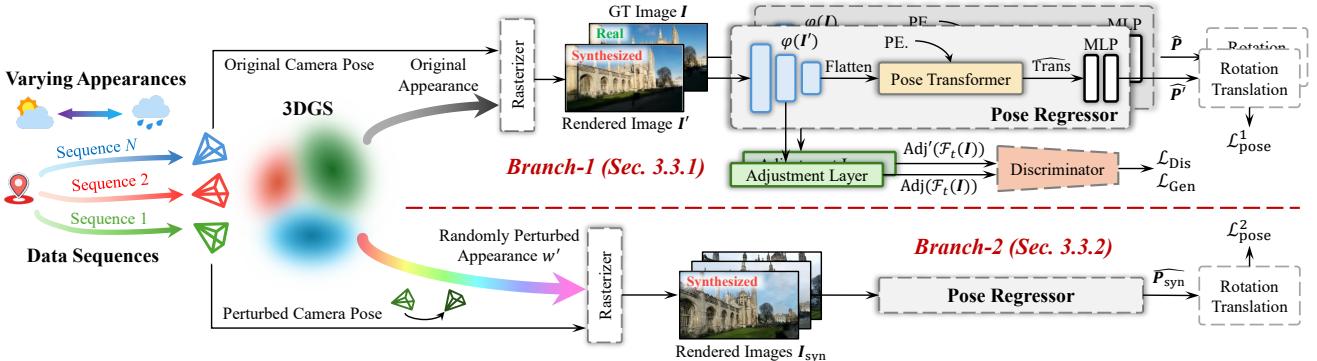


Figure 2. **Pipeline of RAP.** We lift multiple RGB video sequences into 3D Gaussian Splats, which serve as our data engine. The **branch-1** (see Sec. 3.3.1) inputs paired real and synthetic images to regress poses, with a discriminator to bridge the syn-to-real gap. The **branch-2** (see Sec. 3.3.2) generates views with novel poses and appearances, which are fed into the same pose regressor as additional supervision.

ing objects, lighting changes, and inconsistent camera exposure settings. To tackle these in-the-wild challenges, NeRF-W [36] uses per-image transient and appearance embeddings. In 3DGS [23], VastGaussian [28] applies a CNN to 3DGS outputs but still struggles with significant appearance variations. SWAG [12] mitigates this issue by storing appearance information in an external hash-grid-based implicit field, while GS-W [65] enhances flexibility by separating intrinsic and dynamic appearance features for each Gaussian point. 3DGM [27] leverages consensus across multiple sequences as the self-supervision signal to remove transient and moving objects without human annotations. Deblur-GS [62] addresses motion blur—another challenge in localization datasets—by modeling camera motion to yield sharper edges in rendered scenes. Our method incorporates appearance modeling and edge refinement to handle and synthesize diverse indoor, outdoor, and driving scenes.

### 3. Method

#### 3.1. Pre-Processing with 3DGS

A robust pose regressor should focus on intrinsic scene attributes, not appearance variations. Therefore, we first synthesize diverse visual data for training. We leverage 3DGS [23], representing scenes with explicit ellipsoids, to model diverse appearances (see Sec. B for more on 3DGS). Following GS-W [65], we assume the scene contains  $K$  Gaussians and represent the independent intrinsic material attributes using positions  $\mu \in \mathbb{R}^{K \times 3}$ , spherical harmonics  $\mathcal{Y} \in \mathbb{R}^{K \times 16 \times 3}$ , and other parameters  $\Theta$  including rotation  $q \in \mathbb{R}^{K \times 4}$ , scaling  $s \in \mathbb{R}^{K \times 3}$ , and opacity  $\alpha \in \mathbb{R}^K$ . To capture the dynamic appearance influenced by environmental factors, we extract features from the input image and assign each Gaussian its own feature using a learnable sampler  $\mathcal{S} \in \mathbb{R}^{K \times 2}$ , forming features  $\mathcal{E} \in \mathbb{R}^{K \times 16 \times 3}$ . We also incorporate the camera’s view direction  $\theta$  to account for viewpoint-dependent effects. The final color of Gaus-

sians  $\mathcal{C} \in \mathbb{R}^{K \times 3}$  is computed as follows:

$$\mathcal{C} = \text{MLP}(\mu, \mathcal{Y}, \omega \mathcal{E}, \theta), \quad (1)$$

where  $\omega$  is the blending weight that controls the dynamic appearance of the rendered image.

Another significant challenge in visual localization is motion blur, often caused by slow shutter speeds during video capture, leading to pose ambiguity and degraded rendering quality, further decreasing localization accuracy. Inspired by Deblur-GS [62], we model camera motion blur as the inverse of scene motion, i.e., the transformation in Gaussian position denoted by  $\mathcal{T} \in \text{SE}(3)$ . For each training image, we sample a certain time step along a linear trajectory with a sampling weight  $\phi \in \mathbb{R}^n$  and blend them to compute loss  $\mathcal{L}$  with the original blurry image  $\mathcal{I}_b \in \mathbb{R}^{H \times W \times 3}$ , optimizing  $\mathcal{T}, \phi, \mathcal{C}$  and other 3DGS parameters  $\Theta$ :

$$\underset{\phi, \mathcal{T}, \mu, \mathcal{Y}, \mathcal{E}, \Theta}{\operatorname{argmin}} \mathcal{L} \left( \mathcal{I}_b, \sum_{i=1}^n \phi_i \text{Render}(\mathcal{T}_i(\mu), \mathcal{C}, \Theta) \right), \quad (2)$$

where the details of  $\mathcal{L}$  are in Eq. II. After training, our 3DGS can efficiently render posed images given  $\theta$  and  $\omega$ .

#### 3.2. Architecture of Pose Regressor

Given a set of images and their associated camera poses  $\{(\mathbf{I}_i, \mathbf{P}_i)\}_{i=1}^n$ , our goal is to train a neural network to directly output a homogeneous camera pose  $\mathbf{P} \in \mathbb{R}^{3 \times 4}$  for a query image  $\mathbf{I} \in \mathbb{R}^{H \times W \times C}$ . Our network architecture is shown as the pose regressor in Fig. 2.

**Feature Extractor.** Pose regression networks typically extract features using a common backbone  $\varphi$ , such as VGG [53] or EfficientNet [56], leveraging multiple deeper layers for translation and rotation regression:

$$\varphi(\mathbf{I}) = \{\mathcal{F}_0(\mathbf{I}), \dots, \mathcal{F}_{N-1}(\mathbf{I}), \mathcal{F}_N(\mathbf{I})\}, \quad (3)$$

$\mathcal{F}_*(\cdot)$  denotes features extracted from the  $*$ -th layer of a backbone with  $N$  layers.  $\mathcal{F}_t(\mathbf{I})$  and  $\mathcal{F}_r(\mathbf{I})$  denote features for translation and rotation regression, respectively.

**Pose Transformer.** Unlike CNN-based regression models [9, 30], where fine-grained local features can introduce noise and harm performance, we propose *Pose Transformer* to leverage the strong capability of Vision Transformer (ViT) [14] for modeling long-range dependencies. Each Transformer generates a global token (Trans for translation and Rot for rotation) to provide a comprehensive context for pose regression, inspired by the CLS token in ViT. Given  $\mathcal{F}_r(\mathbf{I})$  and  $\mathcal{F}_t(\mathbf{I})$ , the translation token is then concatenated with the flattened input features<sup>\*</sup>:

$$\widetilde{\mathcal{F}}_t(\mathbf{I}) = \text{Cat}(\text{Flatten}(\mathcal{F}_t(\mathbf{I})), \text{Trans}) \in \mathbb{R}^{(H_t W_t + 1) \times C_t}. \quad (4)$$

The positional encodings are then added to the flattened feature  $(\text{PE} + \widetilde{\mathcal{F}}_t(\mathbf{I})) \in \mathbb{R}^{(H_t W_t + 1) \times C_t}$ . Multi-head Self-Attention (MSA) is then conducted through a stack of multiple layers with the post-processing as follows:

$$\begin{aligned} \widehat{\mathcal{F}}'_t(\mathbf{I}) &= \text{MSA}(\text{PE} + \widetilde{\mathcal{F}}_t(\mathbf{I})) + \text{PE} + \widetilde{\mathcal{F}}_t(\mathbf{I}), \\ \widehat{\mathcal{F}}_t(\mathbf{I}) &= \text{LN}(\text{FFN}(\text{LN}(\widehat{\mathcal{F}}'_t(\mathbf{I}))) + \widehat{\mathcal{F}}'_t(\mathbf{I})), \end{aligned} \quad (5)$$

where LN indicates layer normalization and FFN denotes the fully connected feed-forward network, consisting of two linear layers with a ReLU. The final output is flattened back to  $(H_t W_t + 1) \times c_t$ . More details can be found in Sec. D.2.

**Regression Head.** Only the processed translation token,  $\widehat{\text{Trans}}$ , capturing global features for regression, is fed into the regression head. This regression head consists of two MLPs, each with two hidden layers and GeLU activation:

$$\hat{\mathbf{t}} = \text{Linear}(\text{GeLU}(\text{Linear}(\widehat{\text{Trans}}))). \quad (6)$$

The  $\hat{\mathbf{t}}$  represents the final prediction for translation. Similarly, we obtain the rotation prediction denoted by  $\hat{\mathbf{r}}$ .

### 3.3. Two-Branch Joint Training Paradigm

#### 3.3.1 Branch-1: Aligning Features via Discriminator

Synthetic images from 3DGS provide novel viewpoints and appearances but often contain artifacts, leading to a syn-to-real domain gap. To align features from rendered and real images of the same pose, we introduce an adversarial training mechanism besides the basic pose regression training.

**Pose Regression Loss.** For basic training, we render the synthetic image  $\mathbf{I}'$  with the same pose label  $\mathbf{P}$  as the real image  $\mathbf{I}$ , both used as supervision for the pose regressor. The training objective consists of translation loss  $\mathcal{L}_t$  and the rotation loss  $\mathcal{L}_r$ , which are measured by the Euclidean distance between the ground truth pose  $\mathbf{P} = \{\mathbf{t}, \mathbf{r}\}$  and the estimated pose  $\widehat{\mathbf{P}} = \{\hat{\mathbf{t}}, \hat{\mathbf{r}}\}$ :

<sup>\*</sup>We only present the translation regression for simplicity.

$$\mathcal{L}_t = \|\mathbf{t} - \hat{\mathbf{t}}\|_2, \quad (7)$$

$$\mathcal{L}_r = \left\| \mathbf{r} - \frac{\hat{\mathbf{r}}}{\|\hat{\mathbf{r}}\|} \right\|_2, \quad (8)$$

$$\mathcal{L}_{\text{pose}}^1 = \mathcal{L}_t \exp(-s_t) + s_t + \mathcal{L}_r \exp(-s_r) + s_r, \quad (9)$$

where  $s_t$  and  $s_r$  are learned parameters for balancing the optimization between rotation and translation [21].

**Adversarial Loss.** The adversarial training mechanism optimizes the discriminator to distinguish real from rendered image features, while training the feature extractor to fool the discriminator, effectively bridging the domain gaps. To prevent vanishing gradients, we propose a novel adversarial objective for pose regression, inspired by LSGAN [35]:

$$\begin{aligned} \underset{D}{\text{argmin}} \mathcal{L}_{\text{Dis}}(D) &= \frac{1}{2} \mathbb{E}_{\mathbf{I} \sim p_{\text{data}}(\mathbf{I})} [(D(\text{Adj}(\mathcal{F}_t(\mathbf{I}))) - 1)^2] \\ &\quad + \frac{1}{2} \mathbb{E}_{\mathbf{I}'} [D(\text{Adj}'(\mathcal{F}_t(\mathbf{I}'))^2], \end{aligned} \quad (10)$$

$$\underset{G}{\text{argmin}} \mathcal{L}_{\text{Gen}}(G) = \frac{1}{2} \mathbb{E}_{\mathbf{I}'} [(D(\text{Adj}'(\mathcal{F}_t(\mathbf{I}')) - 1)^2]. \quad (11)$$

Here,  $\text{Adj}$  and  $\text{Adj}'$  are the adjustment layers, consisting of Conv-ReLU-BN layers. The feature extractor  $\varphi$  acts as the generator  $G$ , while  $D$  is the discriminator, composed of several convolutional layers with ReLU activations. More details are provided in Sec. D.3.

#### 3.3.2 Branch-2: Training while Synthesizing Data

With the proposed appearance-varying 3DGS, more posed images can be generated to enrich the training data for better generalizability. Specifically, our data synthesis is categorized into two dimensions: *pose augmentation* and *appearance augmentation*, as illustrated in Fig. 1. For pose augmentation, given a training pose  $\mathbf{P}$ , a perturbed pose  $\mathbf{P}_{\text{syn}}$  can be generated around  $\mathbf{P}$  by the translation noise of  $\delta t$  and rotation noise of  $\delta r$ . For appearance augmentation, we randomly adjust the appearance of rendered images using random blending weights  $\omega$ , and then render the synthetic image  $\mathbf{I}_{\text{syn}}$  using the Gaussian Splats trained in Sec. 3.1. The novel image-pose pair  $(\mathbf{I}_{\text{syn}}, \mathbf{P}_{\text{syn}})$ , online generated every 20 epochs during training until the validation loss ceases to decrease, serves as additional supervision for the training. Given the estimated pose of the synthesized image denoted by  $\widehat{\mathbf{P}}_{\text{syn}}$ , the loss function  $\mathcal{L}_{\text{pose}}^2(\widehat{\mathbf{P}}_{\text{syn}}, \mathbf{P}_{\text{syn}})$  is same as  $\mathcal{L}_{\text{pose}}^1$ .

#### 3.3.3 Overall Objective

The total loss for the pose regressor is:

$$\mathcal{L}_{\text{total}} = \beta_1 \mathcal{L}_{\text{pose}}^1 + \beta_2 \mathcal{L}_{\text{pose}}^2 + \beta_3 (\mathcal{L}_{\text{Gen}} + \mathcal{L}_{\text{Dis}}), \quad (12)$$

where  $\beta_1, \beta_2, \beta_3$  are loss weights. The total loss will optimize the pose regressor, adjustment layers, and discriminator. Only the pose regressor will be deployed in the inference phase, while the other two components are discarded.

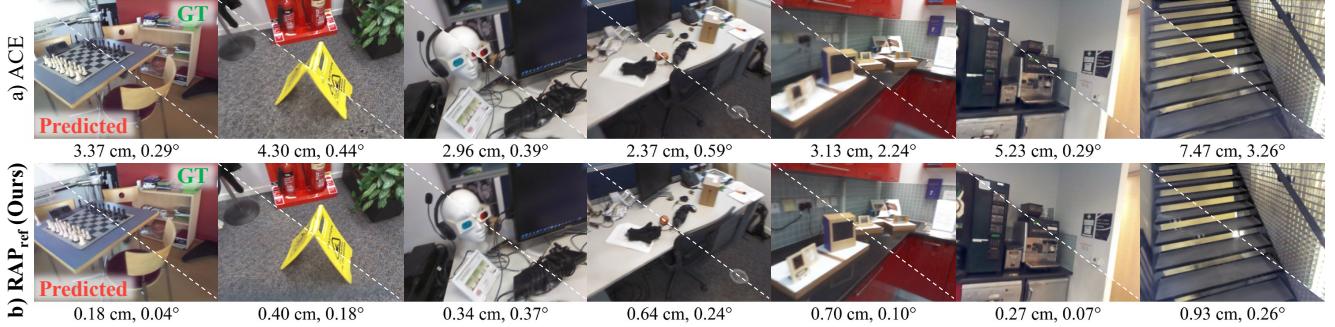


Figure 3. **Visualization of the localization errors on the 7-Scenes dataset [52].** Each subfigure is split by a diagonal line, with the **bottom-left** corner showing the image rendered from the estimated or refined pose and the **top-right** corner displaying the ground truth image. The continuity along the diagonal line demonstrates the improved accuracy of the poses obtained by our  $\text{RAP}_{\text{ref}}$ .

Table 1. **Quantitative results on the 7-Scenes dataset [52].** We compare our method against three categories of approaches (single-frame APR, SCR, and NRP) based on median translation (cm) and rotation ( $^{\circ}$ ) errors. The best results are highlighted in **bold**. More qualitative visualizations are in Fig. VII. **DSLAM GT** and **SfM GT** refer to different sets of ground truth. More details are provided in Sec. G.2.

Category	Methods	<i>Chess</i>	<i>Fire</i>	<i>Heads</i>	<i>Office</i>	<i>Pumpkin</i>	<i>Kitchen</i>	<i>Stairs</i>	Average
Single Frame APR	PoseNet (PN) [22]	32/8.12	47/14.4	29/1.20	48/7.68	47/8.42	59/8.64	47/13.80	44/10.4
	MapNet [6]	8/3.25	27/11.7	18/13.3	17/51.5	22/4.02	23/4.93	30/12.1	21/7.77
	AtLoc+ [58]	10/3.18	26/10.8	14/11.4	17/5.16	20/3.94	16/4.90	29/10.2	19/7.08
	MS-Transformer [51]	11/4.66	24/9.60	14/12.2	17/5.66	18/4.44	17/5.94	17/5.94	18/7.28
	PAE [49]	12/4.95	24/9.31	14/12.5	19/5.79	18/4.89	18/6.19	25/8.74	19/7.48
	DFNet [9]	5/1.88	17/6.45	<b>6/3.63</b>	8/2.48	10/2.78	22/5.45	16/3.29	12/3.71
	PMNet [30]	4/1.70	10/4.51	7/4.23	7/1.96	14/3.33	14/3.36	16/3.62	10/3.24
	<b>RAP (Ours, DSLAM GT)</b>	<b>2/1.15</b>	<b>7/4.44</b>	<b>5/5.94</b>	<b>7/1.95</b>	<b>7/2.23</b>	<b>7/2.12</b>	<b>12/3.07</b>	<b>7/2.98</b>
	<b>RAP (Ours, SfM GT)</b>	<b>1/0.84</b>	<b>6/3.44</b>	<b>4/5.48</b>	<b>5/1.92</b>	<b>4/1.71</b>	<b>7/2.11</b>	<b>9/2.10</b>	<b>5/2.51</b>
SCR	DSAC* [3]	0.5/0.17	0.8/0.28	0.5/0.34	1.2/0.34	1.2/0.28	0.7/0.21	2.7/0.78	1.1/0.34
	ACE [5]	0.5/0.18	0.8/0.33	0.5/0.33	1.0/0.29	1.0/0.22	0.8/0.20	2.9/0.81	1.1/0.34
	GLACE [59]	0.6/0.18	0.9/0.34	0.6/0.34	1.1/0.29	0.9/0.23	0.8/0.20	3.2/0.93	1.2/0.36
NRP	FQN-MN [18]	4.1/1.31	10.5/2.97	9.2/2.45	3.6/2.36	4.6/1.76	16.1/4.42	139.5/34.67	28/7.3
	CrossFire [40]	<b>1/0.4</b>	5/1.9	3/2.3	5/1.6	3/0.8	2/0.8	12/1.9	4.4/1.38
	DFNet + NeFeS <sub>50</sub> [10]	2/0.57	<b>2/0.74</b>	<b>2/1.28</b>	<b>2/0.56</b>	<b>2/0.55</b>	<b>2/0.57</b>	5/1.28	<b>2.4/0.79</b>
	HR-APR [32]	2/0.55	2/0.75	2/1.45	2/0.64	2/0.62	2/0.67	5/1.30	<b>2.4/0.85</b>
	MCLoc [57]	2/0.8	3/1.4	3/1.3	4/1.3	5/1.6	6/1.6	6/2.0	4.1/1.43
	<b>RAP<sub>ref</sub> (Ours, DSLAM GT)</b>	2.6/0.89	1.9/0.85	1.3/6.5	2.6/0.74	4.5/1.17	4/1.30	<b>2.8/0.78</b>	2.8/0.91
	<b>RAP<sub>ref</sub> (Ours, SfM GT)</b>	<b>0.33/0.11</b>	<b>0.57/0.22</b>	<b>0.39/0.28</b>	<b>0.60/0.18</b>	<b>0.82/0.20</b>	<b>0.53/0.15</b>	<b>1.12/0.32</b>	<b>0.62/0.21</b>

## 4. Experiments

### 4.1. Evaluation Setup

**Datasets.** For evaluation, we follow previous works [9, 30] to use four scenes in the Cambridge Landmarks dataset [22] with spatial extents from 875 m<sup>2</sup> to 5600 m<sup>2</sup>. We also employ the 7-Scenes dataset [52], which provides seven indoor scenes with volumes spanning 1 m<sup>3</sup>-18 m<sup>3</sup>, and follow the original training and testing splits with more accurate SfM pose annotations [5, 10]. Moreover, we evaluate our method on MARS [26], a self-driving dataset featuring challenges like moving objects, lighting changes, and motion blur.

**Baselines.** We first compare our proposed RAP against common single-frame APR approaches on the three datasets, where PMNet [30] and DFNet [9] are the most related and advanced methods based on data augmentation. We split the remaining methods into two categories

based on whether they rely on extra novel view synthesis in test time, including SCR [4, 5, 59] and NRP (Neural / Rendering-based Post refinement) [10, 18, 32, 40, 57, 66, 67], which involves rendering or querying features in novel views by the initial pose for the following one-shot or iterative refinement. MCLoc [57] is a pose refinement method using 3DGS, while others use NeRF.

**Implementation Details.** We first optimize our 3DGS for each scene without masking moving objects. We then train our RAP network, which uses an Efficient-B0 backbone [34] pre-trained on ImageNet [13], optimized with Adam [24] at a learning rate of  $1 \times 10^{-4}$ . Only the features from the third (reduction\_3) and fourth (reduction\_4) layers are used respectively for translation and rotation regression, and both layers are utilized for narrowing the domain gap via the discriminator, which is also optimized with Adam [24], using a learning rate of

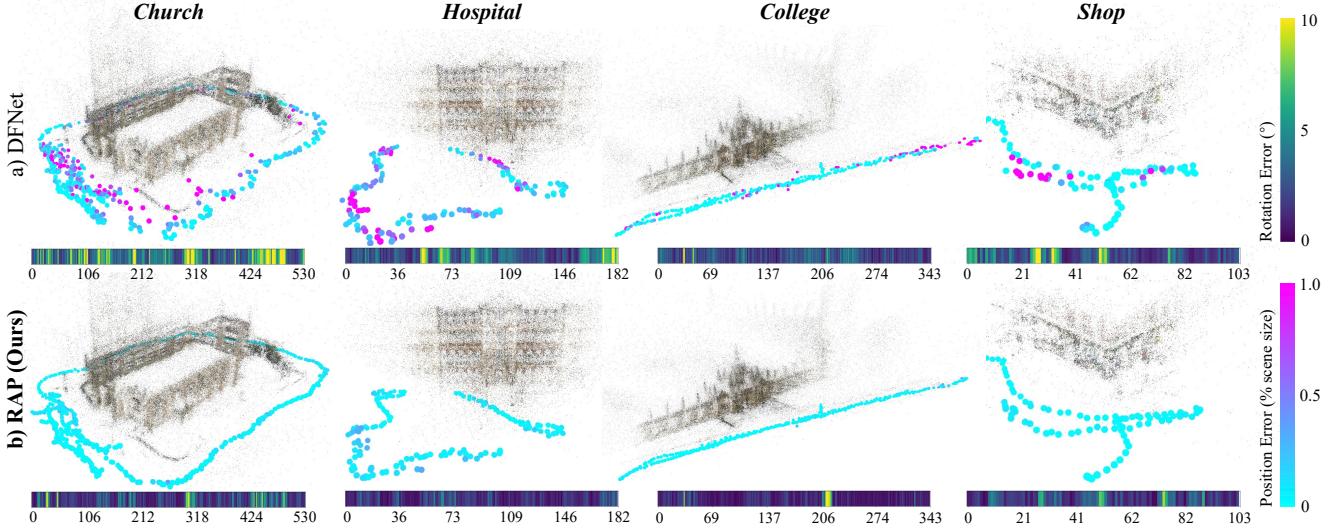


Figure 4. **Qualitative comparison of camera pose estimation errors between a) DFNet [9] and b) our RAP framework** across four scenes on the Cambridge Landmarks dataset [22]. The rotation errors are represented by the color bar on the upper right (in degrees), while position errors (as a percentage of scene size) are shown based on the lower right color bar. Our RAP framework estimates trajectories that more closely follow the ground truth, with significantly reduced rotation and position errors compared to DFNet [9].

Table 2. **Quantitative results on the Cambridge Landmarks dataset [22].** We compare our method with single-frame APR, SCR, and NRP methods in median translation (cm) and rotation ( $^{\circ}$ ) errors. The best results are highlighted in **bold**.

Category	Methods	College	Hospital	Shop	Church	Average
Single Frame APR	PoseNet (PN) [22]	166/4.86	262/4.90	141/7.18	245/7.95	204/6.23
	MapNet [6]	107/1.89	194/3.91	149/4.22	200/4.53	163/3.64
	MS-Transformer [51]	83/1.47	181/2.39	86/3.07	162/3.99	128/2.73
	PAE [49]	90/1.49	207/2.58	99/3.88	164/4.16	140/3.03
	DFNet [9]	73/2.37	200/2.98	67/2.21	137/4.03	119/2.90
	PMNet [30]	68/1.97	103/1.31	58/2.10	133/3.73	90/2.27
SCR	RAP (Ours)	<b>58/0.87</b>	<b>81/1.20</b>	<b>33/1.64</b>	<b>60/1.83</b>	<b>58/1.39</b>
	DSAC* [3]	<u>18/0.3</u>	<u>21/0.40</u>	<u>5/0.3</u>	<u>15/0.6</u>	<u>15/0.40</u>
	ACE [5]	29/0.38	31/0.61	<u>5/0.3</u>	19/0.6	21/0.47
NRP	GLACE [59]	19/0.32	<u>18/0.42</u>	<u>5/0.22</u>	<u>9/0.3</u>	<u>13/0.32</u>
	FQN-MN [18]	28/0.4	54/0.8	13/0.6	58/2	38/1
	CrossFire [40]	47/0.7	43/0.7	20/1.2	39/1.4	37/1
	DFNet + NeFeS <sub>30</sub> [10]	37/0.64	98/1.61	170/6.0	42/1.38	49/1.06
	DFNet + NeFeS <sub>50</sub> [10]	37/0.54	52/0.88	15/0.53	37/1.14	35/0.77
	HR-APR [32]	36/0.58	53/0.89	13/0.51	38/1.16	35/0.78
	MCLoc [57]	31/0.42	39/0.73	12/0.45	26/0.8	27/0.6
RAP <sub>ref</sub> (Ours)	RAP <sub>ref</sub> (Ours)	<b>18/0.38</b>	<b>22/0.42</b>	<b>8/0.39</b>	<b>16/0.47</b>	<b>16/0.42</b>

0.0002 and betas set to (0.5, 0.999). More details about training can be found in Sec. E.2. For generating random views, we apply random normalized perturbations to each training pose:  $\delta t = 20$  cm and  $\delta r = 10^{\circ}$  for indoor scenes, and  $\delta t = 150$  cm and  $\delta r = 4^{\circ}$  for outdoor scenes. To allow for comparison with SCR methods and leverage 3DGS’s efficient rendering for NRP, we extend the APR pipeline with match-based refinement, denoted as RAP<sub>ref</sub>. At test time, RAP’s initial predicted pose will be used to render an RGB-D image via 3DGS. Together with MAST3R [25], we can obtain 2D-3D correspondences to perform RANSAC-

PnP [16, 17], resulting in a refined pose. More details are provided in Sec. E and Sec. F.

## 4.2. Benchmark Results

**7-Scenes** [52]. As shown in Table 1, our RAP achieves superior performance, with a 50% reduction in translation error ( $0.10 \rightarrow 0.05$ ) and a 21.91% reduction in rotation error ( $3.24^{\circ} \rightarrow 2.51^{\circ}$ ) averagely compared with previous state-of-the-art single-frame APR methods. Only in *heads* the rotation error is suboptimal, which consists of only two sequences, one for training and the other for testing. This restriction may limit the ability of our augmentation method to fully capture the scene’s variability. Meanwhile, RAP<sub>ref</sub> achieves a substantial improvement, significantly reducing pose estimation errors with one-shot refinement using our 3DGS. Notably, our RAP<sub>ref</sub> is the first to achieve an average translation error below 1 cm and a rotation error below  $0.25^{\circ}$ . Qualitative examples are displayed in Fig. 3.

**Cambridge Landmarks** [22]. In the more challenging outdoor Cambridge Landmarks dataset, as shown in Table 2, our RAP demonstrates a significant performance advantage across all scenes, achieving over 30% improvements in both translation and rotation error compared to other single-frame APR methods. The visualization in Fig. 4 shows that our method produces fewer outliers than DFNet. In the two larger-scale scenes, i.e., *College* and *Church*, the improvement in rotation error even doubled. Table 2 also highlights the effectiveness of our RAP<sub>ref</sub> in further reducing pose errors through the refinement phase. Specifically, RAP<sub>ref</sub> with one-shot optimization significantly outperforms CrossFire [40] and DFNet + NeFeS [10], which require 30 and

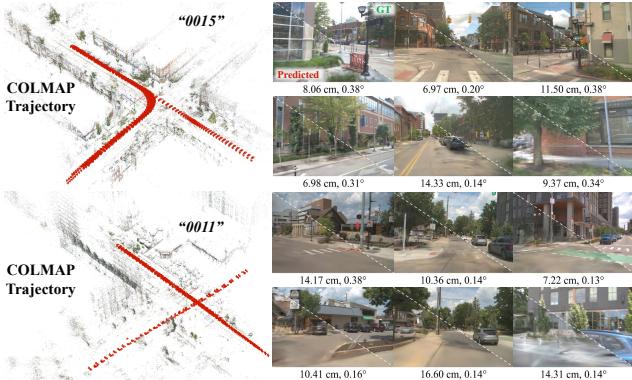


Figure 5. **Qualitative results of our RAP<sub>ref</sub> on the MARS dataset [26].** Our method achieves precise localization even in highly dynamic scenarios with varying weather conditions.

even 50 optimization steps. Compared to ACE [5], our model achieves competitive localization performance, with a slight drop only in *Shop*. This shortfall is likely due to the rolling shutter effect in the images, causing distortion and edge ambiguity in the rendered views, which can weaken the performance of 2D-2D matching in MAST3R [25].

**MARS [26].** Autonomous driving scenarios present unique challenges, including moving objects and frequent changes in lighting conditions, as illustrated in Fig. 5. Our RAP demonstrates effective and robust performance across four challenging scenes, as shown in Table 3, achieving an average 45 cm / 0.78° localization error. This significantly outperforms the baseline PoseNet [22] and DFNet [9], which struggles to model such large-scale driving scenarios using NeRF due to noise introduced during the learning process. With one-shot refinement, our RAP<sub>ref</sub> further reduces outdoor localization errors to below 10 cm.

### 4.3. Ablation Study

We conduct ablation studies on the validation set of *Shop* in the Cambridge Landmarks dataset to investigate the impact of all the components in our RAP. Setup I, our baseline, consists of the same components as in PoseNet [22] and has been retrained for our experiments. In Setup II, we replace the feature extraction from VGG16 [53] to Efficient-B0 [34], which enhances performance due to its superior feature representation, while they both exhibit poor performances due to the lack of data synthesis. In Setup III and IV, we explore the effectiveness of the designed pose augmentation and appearance augmentation, which bring notable improvements: translation error reduces from 103 cm to 75 cm, and rotation error from 3.52° to 3.14°. In Setup V and VI, we add regular convolutional layers and Pose Transformer between feature extraction and pose regression. Both improve performance due to the increasing parameters, but the Transformer achieves superior results by effectively handling long-term dependencies through atten-

Table 3. **Quantitative results on the MARS dataset [26].** We compare our pipeline with DFNet [9] and PoseNet [22] in translation error (cm) and rotation (°) errors.

Methods	“0011”	“0015”	“0037”	“0041”	Average
PoseNet [22]	149/1.80	136/2.34	123/1.60	75/0.92	121/1.67
DFNet [9]	298/4.70	528/10.25	535/7.18	642/4.43	530/6.64
<b>RAP (Ours)</b>	<b>44/0.68</b>	<b>59/1.55</b>	<b>24/0.40</b>	<b>51/0.50</b>	<b>45/0.78</b>
<b>RAP<sub>ref</sub> (Ours)</b>	<b>9.0/0.12</b>	<b>8.6/0.24</b>	<b>8.8/0.09</b>	<b>7.9/0.14</b>	<b>8.6/0.15</b>

Table 4. **Ablation study.** We systematically investigate the impact of the proposed components; see Sec. 4.3 for detailed analysis.

Setups	on <i>Shop</i>	Trans. (cm) ↓	Rot. (°) ↓
I (Baseline): $\varphi = \text{VGG16}$	174	5.45	
II: $\varphi = \text{Efficient-B0}$	103	4.64	
III: II + Pose Aug.	75	3.52	
IV: III + Appearance Aug.	60	3.14	
V: IV + Decoder (ConvNet)	52	2.51	
VI: V + Decoder (Transformer)	40	1.98	
<b>VII (Ours): VI + Discriminator</b>	<b>33</b>	<b>1.64</b>	

tion mechanisms. Finally, in Setup VII, our adversarial discriminator effectively reduces the syn-to-real domain gap, allowing the model to learn better pose regression features from synthetic data and further reduce localization error.

### 4.4. Discussion on Data Synthesis

**Emerging Interpolation and Extrapolation Capability.** Previously, APR has been understood to implicitly learn image retrieval [47], lacking the ability to successfully interpolate between training samples and generalize beyond them. To investigate how APR training is affected by increasing synthetic data, we experimented on *Shop* using only 20% of the real training set with our synthetic data, as shown in Fig. 6. We can observe that the training set with synthetic data, represented by the red hollow spheres, does not fully cover the test set spatially. Despite this, the model still closely predicts the test camera poses, demonstrating interpolation capability between training positions and even extrapolation capability beyond the original spheres.

**Analyzing Generalization Boundaries.** To evaluate the model’s generalization capability, we designed an experiment introducing a “void zone” centered on the test camera, where all real and synthetic data within this zone were excluded. The void zone was progressively expanded to determine the critical threshold at which the localization performance declines most significantly. Specifically, for *Shop*, we used 100% of the training set to ensure complete scene coverage, with void zone ranges set as [10/0.5, 20/1, 30/1.5, 50/2, 80/2.5, 100/3] (cm/°). The results in Table 5 demonstrate a stepwise decline in performance. Initially, expanding the void zone has minimal impact on localization accuracy. However, at 30 cm / 1.5°, a sharp decrease in performance marks the model’s generalization boundary.

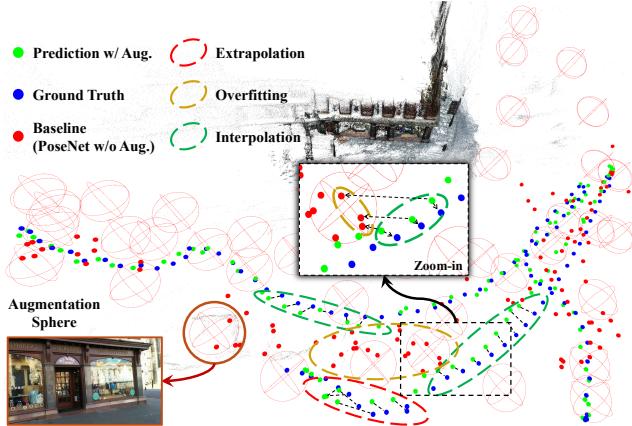


Figure 6. **Visualization of camera locations.** The red hollow spheres, centered at the real images in the training set, represent the potential locations of all synthetic images during the training phase. The blue dots, green dots, and red dots represent the ground truth, prediction by our RAP, and by the baseline, respectively.

**Density of Training Data.** As shown in Table 6, our method with the proposed augmentation significantly reduces errors as the density of real training data increases from 20% to 80%. However, the localization accuracy remains almost unchanged from 80% to 100%, as the scene is already sufficiently covered. Notably, using 100% of the training data without augmentation can result in a significantly higher maximum error in translation, nearly double that with only 20% of the training data with augmentation, despite its limited spatial coverage. This suggests that our augmentation method successfully prevents overfitting to the training data, improving generalization to the test set.

**Quality of Training Data.** We evaluate the impact of synthetic image quality on model performance in Table 7, using 20% and 50% of the real data for pose regression. For *Shop*, it is evident that fewer training samples in 3DGS result in lower-quality rendered views (as indicated by lower PSNR), leading to suboptimal localization performance, particularly for rotation. Surprisingly, localization performance using only 20% of the data for training suboptimal 3DGS and pose regression surpasses the results obtained with 100% of the data without augmentation, as shown in Table 6. This experiment confirms the need for a robust NVS model and the proposed augmentation method in APR training.

## 5. Conclusion

**Summary.** We address absolute pose regression with a robust two-branch joint training framework based on Transformer, coupled with an efficient data synthesis pipeline leveraging 3D Gaussian Splats (3DGS) to synthesize numerous posed images with diverse appearances as additional supervision. Our RAP achieves state-of-the-art localization performance, even under challenging appearance

Table 5. **Exploring the generalization boundaries of the model with synthetic data.** Green, blue, and red percentages indicate the relative change in localization error (**Med Err**) compared to the scenario without a void zone.

	w/o Void Zone	Void Zone (cm/ $^{\circ}$ )			
		10/0.5	20/1	30/1.5	50/2
Med Err $\downarrow$ (rel. change)	33/1.26 0%/0%	30/1.34 -9%/6.3%	32/1.32 -3%/4.7%	40/1.84 21%/46.0%	39/2.07 18%/64.3%
Avg Err $\downarrow$	41/1.51	38/1.75	41/1.63	51/2.40	48/2.40
Max Err $\downarrow$	155/4.52	147/6.56	219/6.01	192/8.38	246/9.80
Min Err $\downarrow$	3/0.17	8/0.20	4/0.22	7/0.30	4/0.16
					8/0.60

Table 6. **Impact of the density of real training data.** Our augmentation improves the model’s ability to generalize across the entire scene, although this effect has an upper limit.

Training Pose %	w/ Appearance & Pose Aug. (cm/ $^{\circ}$ )					w/o Aug. (cm/ $^{\circ}$ )	
	100%	80%	60%	40%	20%	100%	50%
Med Err $\downarrow$	33/1.26	32/1.27	37/1.90	57/2.23	87/3.65	98/3.75	104/4.17
Avg Err $\downarrow$	41/1.51	40/1.50	48/2.17	62/2.81	91/4.65	128/4.49	139/5.33
Max Err $\downarrow$	155/4.52	158/4.09	193/9.39	230/11.06	231/15.45	490/20.73	500/21.02
Min Err $\downarrow$	3/0.17	7/0.20	7/0.18	6/0.38	12/0.46	13/0.63	9/0.48

Table 7. **Impact of synthetic image quality.** Training with higher-quality synthetic images from advanced NVS models enhances localization performance.

3DGS Performance			Localization Performance (cm/ $^{\circ}$ )				
% Images (Train)	PSNR $\uparrow$ (Train)	PSNR $\uparrow$ (Test)	% Images (Train)	Med Err $\downarrow$	Avg Err $\downarrow$	Max Err $\downarrow$	Min Err $\downarrow$
20%	15.98	29.08	20%	58/3.59	68/4.31	211/12.19	14/0.51
20%	15.98	29.08	50%	43/2.47	55/3.26	196/21.11	7/0.40
50%	17.55	26.88	50%	37/1.88	48/2.37	184/10.17	9/0.52
100%	18.30	24.60	50%	33/1.64	41/2.12	130/11.07	4/0.38

variations. Moreover, we thoroughly investigate the impact of scaling synthetic data and present a novel perspective on APR: interpolation and extrapolation capabilities can emerge if the data and learning gaps in APR are effectively addressed. We believe our RAP could be a promising starting point, and the experiments presented in the paper can provide useful insights for future research in this field.

**Limitations and Future Works.** Similar to other APR approaches, our method does not yet achieve better accuracy than geometry-based techniques, and our per-scene training is relatively time-consuming. Future work includes efficiently training stronger APR models with geometric priors, leveraging temporal information, or powerful vision foundation models [43]. Furthermore, achieving generalization in dynamic environments with fewer training samples presents a promising research direction.

## Acknowledgment

This work was supported in part through NSF grants 2238968, 2121391, and 2024882, and the NYU IT High Performance Computing resources, services, and staff expertise. Yiming Li is supported by NVIDIA Graduate Fellowship (2024-2025).

## References

- [1] Joydeep Biswas and Manuela Veloso. Depth camera based indoor mobile robot localization and navigation. In *2012 IEEE International Conference on Robotics and Automation*, pages 1697–1702. IEEE, 2012. 1
- [2] Åke Björck. Solving linear least squares problems by gram-schmidt orthogonalization. *BIT Numerical Mathematics*, 7(1):1–21, 1967. 13
- [3] Eric Brachmann and Carsten Rother. Visual camera re-localization from rgb and rgb-d images using dsac. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5847–5865, 2021. 2, 5, 6
- [4] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6684–6692, 2017. 1, 5
- [5] Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5044–5053, 2023. 1, 2, 5, 6, 7, 17
- [6] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-aware learning of maps for camera localization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2616–2625, 2018. 1, 2, 5, 6
- [7] David M. Chen, Georges Baatz, Kevin Koser, Sam S. Tsai, Ramakrishna Vedantham, Timo Pylvanainen, Kimmo Roimela, Xin Chen, Jeff Bach, Marc Pollefeys, Bernd Girod, and Radek Grzeszczuk. City-scale landmark identification on mobile devices. In *CVPR 2011*, 2011. 2
- [8] Shuai Chen, Zirui Wang, and Victor Prisacariu. Directposenet: Absolute pose regression with photometric consistency. In *2021 International Conference on 3D Vision (3DV)*, 2021. 2
- [9] Shuai Chen, Xinghui Li, Zirui Wang, and Victor A Prisacariu. Dfnet: Enhance absolute pose regression with direct feature matching. In *European Conference on Computer Vision*, pages 1–17. Springer, 2022. 1, 2, 4, 5, 6, 7, 12, 14, 15, 16, 18, 23
- [10] Shuai Chen, Yash Bhalgat, Xinghui Li, Jia-Wang Bian, Kejie Li, Zirui Wang, and Victor Adrian Prisacariu. Neural refinement for absolute pose regression with feature synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20987–20996, 2024. 5, 6, 14
- [11] Manuela Chessa, Chiara Bassano, and Fabio Solari. Detection and localization of changes in immersive virtual reality. In *International Conference on Image Analysis and Processing*, pages 121–132. Springer, 2023. 1
- [12] Hiba Dahmani, Moussab Bennehar, Nathan Piasco, Luis Roldao, and Dzmitry Tsishkou. Swag: Splatting in the wild images with appearance-conditioned gaussians. In *European Conference on Computer Vision*, pages 325–340. Springer, 2025. 3
- [13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5
- [14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020. 4, 13
- [15] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 8092–8101, 2019. 2
- [16] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 6, 12, 14, 15, 17, 19, 21, 23
- [17] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003. 2, 6, 12, 14, 15, 17, 19, 21, 23
- [18] Hugo Germain, Daniel DeTone, Geoffrey Pascoe, Tanner Schmidt, David Novotny, Richard Newcombe, Chris Sweeney, Richard Szeliski, and Vasileios Balntas. Feature query networks: Neural surface description for camera pose refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5071–5081, 2022. 5, 6
- [19] Shahad S Ghintab and Mohammed Y Hassan. Cnn-based visual localization for autonomous vehicles under different weather conditions. *Engineering and Technology Journal*, 41(2):375–386, 2023. 1
- [20] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2
- [21] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5974–5983, 2017. 4
- [22] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Poseonet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015. 1, 2, 5, 6, 7, 12, 14, 15, 21, 22
- [23] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 3, 12
- [24] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015. 5

- [25] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024. [6](#), [7](#), [12](#), [15](#)
- [26] Yiming Li, Zhiheng Li, Nuo Chen, Moonjun Gong, Zonglin Lyu, Zehong Wang, Peili Jiang, and Chen Feng. Multiagent multitraversals multimodal self-driving: Open mars dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22041–22051, 2024. [5](#), [7](#), [14](#), [16](#), [23](#), [24](#)
- [27] Yiming Li, Zehong Wang, Yue Wang, Zhiding Yu, Zan Gojcic, Marco Pavone, Chen Feng, and Jose M. Alvarez. Memorize what matters: Emergent scene decomposition from multitraverse. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. [3](#)
- [28] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5721–5731, 2021. [3](#)
- [29] Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. *arXiv preprint arXiv:2410.18647*, 2024. [2](#)
- [30] Jingyu Lin, Jiaqi Gu, Bojian Wu, Lubin Fan, Renjie Chen, Ligang Liu, and Jieping Ye. Learning neural volumetric pose features for camera localization. *arXiv preprint arXiv:2403.12800*, 2024. [2](#), [4](#), [5](#), [6](#), [12](#)
- [31] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17627–17638, 2023. [2](#)
- [32] Changkun Liu, Shuai Chen, Yukun Zhao, Huajian Huang, Victor Prisacariu, and Tristan Braud. Hr-apr: Apr-agnostic framework with uncertainty estimation and hierarchical refinement for camera relocalisation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8544–8550, 2024. [5](#), [6](#)
- [33] Liu Liu, Hongdong Li, and Yuchao Dai. Efficient global 2d-3d matching for camera localization in a large-scale 3d map. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017. [2](#)
- [34] Liu Liu, Hongdong Li, and Yuchao Dai. Efficient global 2d-3d matching for camera localization in a large-scale 3d map. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2372–2381, 2017. [5](#), [7](#), [12](#)
- [35] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017. [4](#)
- [36] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. [3](#)
- [37] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, pages 405–421. Springer, 2020. [2](#)
- [38] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Coordinet: uncertainty-aware pose regressor for reliable vehicle localization. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022. [2](#)
- [39] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Lens: Localization enhanced by nerf synthesis. In *Conference on Robot Learning*, pages 1347–1356. PMLR, 2022. [2](#)
- [40] Arthur Moreau, Nathan Piasco, Moussab Bennehar, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Crossfire: Camera relocalization on self-supervised features from an implicit representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 252–262, 2023. [1](#), [5](#), [6](#)
- [41] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. [14](#)
- [42] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *Proceedings of the IEEE international conference on computer vision*, pages 3456–3465, 2017. [2](#)
- [43] Maxime Oquab, Timothée Darcret, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research Journal*, pages 1–31, 2024. [8](#)
- [44] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12716–12725, 2019. [2](#)
- [45] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020.
- [46] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1744–1756, 2016. [2](#)
- [47] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3302–3312, 2019. [2](#), [7](#), [16](#)
- [48] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. [14](#), [16](#)
- [49] Yoli Shavit and Yosi Keller. Camera pose auto-encoders

- for improving pose regression. In *European Conference on Computer Vision*, pages 140–157. Springer, 2022. 5, 6
- [50] Yoli Shavit, Ron Ferens, and Yosi Keller. Paying attention to activation maps in camera pose regression. *arXiv: Computer Vision and Pattern Recognition, arXiv: Computer Vision and Pattern Recognition*, 2021. 2
- [51] Yoli Shavit, Ron Ferens, and Yosi Keller. Learning multi-scene absolute pose regression with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2733–2742, 2021. 1, 5, 6
- [52] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene co-ordinate regression forests for camera relocalization in rgbd images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2937, 2013. 5, 6, 13, 14, 18, 19, 20
- [53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3, 7
- [54] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)*, pages 835–846. ACM, 2006. 2
- [55] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7199–7209, 2018. 2
- [56] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 3
- [57] Gabriele Trivigno, Carlo Masone, Barbara Caputo, and Torsten Sattler. The unreasonable effectiveness of pre-trained features for camera pose refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12798, 2024. 5, 6
- [58] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention guided camera localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10393–10401, 2020. 5
- [59] Fangjinhua Wang, Xudong Jiang, Silvano Galliani, Christoph Vogel, and Marc Pollefeys. Glace: Global local accelerated coordinate encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21562–21571, 2024. 1, 2, 5, 6
- [60] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024. 2, 12
- [61] Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. Scaling data generation in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12009–12020, 2023. 2
- [62] Chen Wenbo and Liu Ligang. Deblur-gs: 3d gaussian splatting from camera motion blurred images. *Proc. ACM Comput. Graph. Interact. Tech. (Proceedings of I3D 2024)*, 7(1), 2024. 3
- [63] Jiazhai Yang, Shenyuan Gao, Yihang Qiu, Li Chen, Tianyu Li, Bo Dai, Kashyap Chitta, Penghao Wu, Jia Zeng, Ping Luo, et al. Generalized predictive model for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14662–14672, 2024. 2
- [64] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for Gaussian splatting. *arXiv preprint arXiv:2409.06765*, 2024. 17
- [65] Dongbin Zhang, Chuming Wang, Weitao Wang, Peihao Li, Minghan Qin, and Haoqian Wang. Gaussian in the wild: 3d gaussian splatting for unconstrained image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 3
- [66] Boming Zhao, Luwei Yang, Mao Mao, Hujun Bao, and Zhaopeng Cui. Pnerfloc: Visual localization with point-based neural radiance fields. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7450–7459, 2024. 5
- [67] Qunjie Zhou, Maxim Maximov, Or Litany, and Laura Leal-Taixé. The nerfect match: Exploring nerf features for visual localization. *arXiv preprint arXiv:2403.09577*, 2024. 5
- [68] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5745–5753, 2019. 13

## Appendix

### A. Pipeline Workflow

- **Stage 1: Appearance-Varying 3DGS (Sec. 3.1)**
  - **Input:** Sequences of RGB images  $\mathcal{I}$  and corresponding camera poses  $\mathcal{P}$ .
  - **Output:** 3D appearance-varying Gaussians with de-blurring capability.
  - **Loss:**  $\mathcal{L}_1$ ,  $\mathcal{L}_{\text{D-SSIM}}$ ,  $\mathcal{L}_{\text{LPIPS}}$ ,  $\mathcal{L}_{\mathcal{S}}$ , and optionally,  $\mathcal{L}_{\text{depth}}$ .
- **Stage 2: Two-Branch Joint APR Training (Sec. 3.2)**
  - **Branch-1**
    - \* **Input:** Real image-pose pair  $(\mathcal{I}, \mathcal{P})$  and the corresponding synthesized image with the same pose  $(\mathcal{I}', \mathcal{P}')$ .
    - \* **Output:** Adjusted translation features ( $\text{Adj}(\mathcal{F}_t(\mathcal{I}))$ ,  $\text{Adj}'(\mathcal{F}_t(\mathcal{I}'))$ ), adjusted rotation features ( $\text{Adj}(\mathcal{F}_r(\mathcal{I}))$ ,  $\text{Adj}'(\mathcal{F}_r(\mathcal{I}'))$ ), and predicted poses  $(\widehat{\mathcal{P}}, \widehat{\mathcal{P}}')$
    - \* **Loss:** Pose loss  $\mathcal{L}_{\text{pose}}^1$ , generator loss  $\mathcal{L}_{\text{Gen}}$ , and adversarial loss  $\mathcal{L}_{\text{Dis}}$ .
  - **Branch-2**
    - \* **Input:** Synthesized images with randomly blended appearances and perturbed poses  $(\mathcal{I}_{\text{syn}}, \mathcal{P}_{\text{syn}})$ .
    - \* **Output:** Predicted poses  $\widehat{\mathcal{P}}_{\text{syn}}$ .
    - \* **Loss:** Pose loss  $\mathcal{L}_{\text{pose}}^2$ .
- **Stage-3: Post-Refinement (Sec. 4.1)**
  - **Input:** Rendered image from 3DGS using the initial pose predicted by the trained pose regressor above and the query image.
  - **Output:** Final refined pose.
  - **Loss:** RANSAC-PnP [16, 17] solver on pixel-level matching between the rendered and query images.

### B. 3D Gaussian Splatting Preliminary

Gaussian Splatting [23] is a promising real-time novel view synthesis approach. By representing scenes with a set of 3D Gaussians, this method preserves the differentiable properties of volumetric radiance fields while allowing efficient optimization and rendering. The scene is defined through parameters such as position  $\mu \in \mathbb{R}^{K \times 3}$ , covariance decomposed as rotation  $\mathcal{Q} \in \mathbb{R}^{K \times 4}$  and scaling  $s \in \mathbb{R}^{K \times 3}$ , anisotropic color  $c \in \mathbb{R}^{K \times 3}$  modeled by sphere harmonics  $\mathcal{Y} \in \mathbb{R}^{K \times 16 \times 3}$ , and opacity  $\alpha \in \mathbb{R}^K$ . During optimization, the scene representation is optimized by iteratively adjusting parameters through stochastic gradient descent, enabled by a differentiable rasterizer. This process is combined with adaptive density control to dynamically adds or removes Gaussians based on the gradient of screen-space points corresponding to the Gaussians and opacity reset to reduce overfitting caused by floaters. The rendering process involves projecting 3D Gaussians onto the 2D image plane, sorting them by depth, and then applying  $\alpha$ -blending

to generate the final image. The render equation is:

$$C = \sum_{i=1}^N T_i \alpha_i \mathbf{c}_i, \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (\text{I})$$

where  $C \in \mathbb{R}^3$  is each pixel's color and  $T_i$  is the transmittance. This approach significantly speeds up optimizing and rendering while achieving state-of-the-art visual quality.

### C. MASt3R Preliminary

This section further elaborates on the background knowledge of MASt3R [25] mentioned in Sec. 4.1. MASt3R grounds image matching tasks in 3D space to improve robustness and accuracy in challenging scenarios. Building on the DUSt3R [60] framework, MASt3R incorporates a new feature-matching head and a fast reciprocal matching algorithm, significantly enhancing performance for dense correspondences and camera pose estimation. It addresses the limitations of traditional 2D-based methods by leveraging dense 3D pointmaps and a coarse-to-fine matching strategy. Extensive evaluations demonstrate substantial gains in accuracy, computational efficiency, and generalizability, making MASt3R a robust solution for visual localization tasks.

### D. Architecture Details

This section provides additional details regarding the network structure mentioned in Sec 3.2 of the main text.

#### D.1. Feature Extraction

Our RAP pipeline first downsamples the input real images by a scale factor of 2 to enhance computational efficiency. The downsampled images are then normalized and passed through the backbone network. For feature extraction, we utilize EfficientNet-B0 [34] as the backbone for multi-scale feature extraction. Translation feature  $\mathcal{F}_t$  and rotation feature  $\mathcal{F}_r$  are extracted from the third ('reduction\_3') and fourth ('reduction\_4') layers, with the number of feature channels being  $C_t = 40$  and  $C_r = 112$ , which then are projected via  $1 \times 1$  convolutions to align with the input channel dimension  $D = 128$  of the proposed *Pose Transformer*.

#### D.2. Pose Transformer

Relying on fine-grained local features, as done in previous works [9, 30], can hinder invariant feature learning due to image noise caused by dynamic objects and illumination changes. To overcome this, we leverage Transformer's robust ability to capture long-range dependencies, as illustrated in Fig. II. Taking the Cambridge Landmarks dataset [22] as an example, the original image resolution is  $854 \times 480$ . After downsampling and feature extraction, the resulting *translation feature map* (identical for the rotation feature map) has a shape of  $[B, 112, H_t, W_t]$ , where

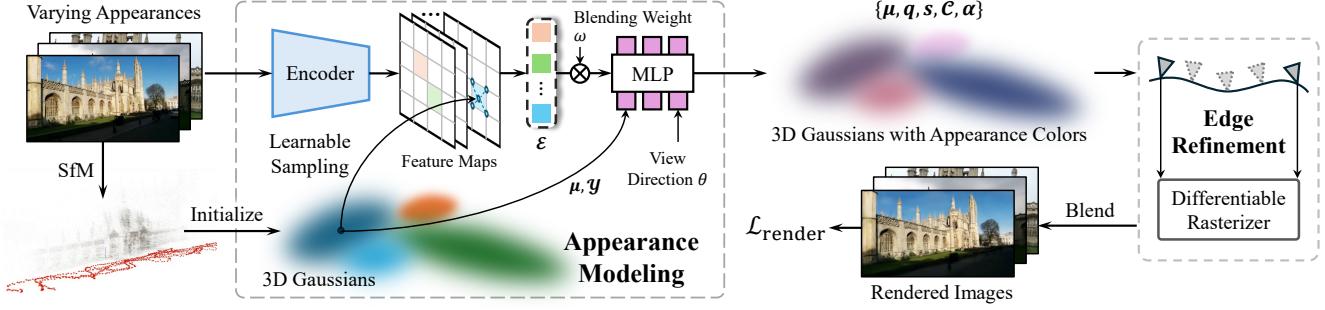


Figure I. **Overall illustration of appearance-varying 3DGS.** The framework models varying appearances using 3D Gaussians enhanced with appearance colors. It initializes 3D Gaussians from SfM data, refines their appearance by learnable sampling and blending weights computed via an encoder and MLP, and renders images by a differentiable rasterizer with edge refinement to minimize the rendering loss.

$H_t = 15$  and  $W_t = 27$ . A  $1 \times 1$  convolutional layer is then applied to adjust the number of feature channels to 128, aligning it with the input dimension of the Transformer architecture. The *translation token* is a learnable parameter with 128 dimensions. The translation feature map is flattened and concatenated with the translation token, forming a matrix of size  $[B, 128, H_t \times W_t + 1]$ . This matrix, combined with positional encodings, is fed into the Transformer decoder, which consists of six layers, each containing multi-head self-attention with eight heads. Finally, the dimension corresponding to the translation token is extracted and passed to the regression head to predict the translation. For clarity, we describe the process using translation as an example, but the same approach is applied to rotation.

### D.3. Adversarial Discriminator

We address the domain gap between synthetic and real images at the feature level by employing an adversarial discriminator. Specifically, the translation features  $\mathcal{F}_t(\mathbf{I})$  and rotation features  $\mathcal{F}_r(\mathbf{I})$  are first processed through the adjustment layers composed of two Conv-ReLU-BN layers, respectively, as  $\text{Adj}(\mathcal{F}_t(\mathbf{I}))$  and  $\text{Adj}'(\mathcal{F}_t(\mathbf{I}'))$ , which align the channel dimensions to a consistent size of 128. The discriminator, implemented as a sequence of four convolutional layers with LeakyReLU activations and dropout, progressively reduces the spatial dimensions. The output is flattened and passed through a fully connected layer tailored for different datasets. Meanwhile, the BCE loss is applied in the discriminator, ensuring effective adversarial learning and bridging the feature-level domain gap.

### D.4. Regression Head

The output features from the Transformer are fed into dedicated regression heads. For translation, the regressor outputs a 3-dimensional vector representing  $[x, y, z]$  coordinates. For rotation, the regressor outputs a 6-dimensional vector, which is a continuous representation of a rotation matrix [68], which is subsequently converted into a  $3 \times 3$  rotation matrix with Gram-Schmidt orthogonalization [2].

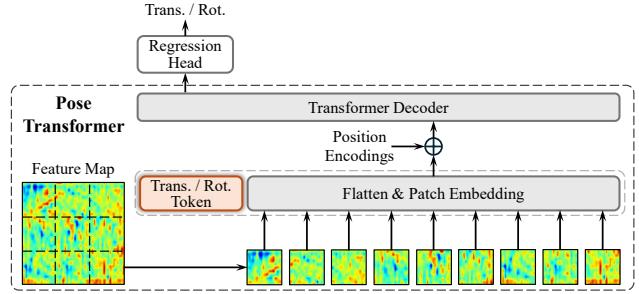


Figure II. **Structure of the Pose Transformer.** The feature map from the backbone is used as input, where we replace the `CLS` token in Vision Transformers (ViT) [14] with translation token `Trans` or rotation token `Rot` for the following regression head.

## E. Implementation Details

### E.1. Appearance-Varying 3DGS

The detailed pipeline of our Appearance-varying 3DGS is demonstrated in Fig. I. For challenging datasets with moving objects and camera motion blur, we extend the optimization iterations to 90,000 and adjust densification parameters and pruning behaviors according to the scene’s size and complexity. For 7-Scenes [52], we use the provided depth information to regularize 3DGS. We sample scenes that need edge refinement twice when optimizing each frame. The loss  $\mathcal{L}$  is implemented as:

$$\mathcal{L} = \gamma_1 \mathcal{L}_1 + \gamma_2 \mathcal{L}_{\text{D-SSIM}} + \gamma_3 \mathcal{L}_{\text{LPIPS}} + \gamma_4 \mathcal{L}_{\mathcal{S}} + \gamma_5 \mathcal{L}_{\text{depth}}, \quad (\text{II})$$

where  $\gamma_1 = 0.8$ ,  $\gamma_2 = 0.2$ ,  $\gamma_3 = 0.005$ ,  $\gamma_4 = 0.001$ , and  $\gamma_5$  is decayed from 1 to 0.01 if depth regularization is enabled; otherwise,  $\gamma_5 = 0$ .  $\mathcal{S}$  is the learnable sampler mentioned in Sec. 3.1.  $\mathcal{L}_{\mathcal{S}}$  is computed as:

$$\mathcal{L}_{\mathcal{S}} = \frac{1}{n} \sum \text{ReLU}(|\mathcal{S}| - 1). \quad (\text{III})$$

**Table I. Metadata showing the number of images in the training and test sets for each scene.** The number of image sequences in each scene is indicated in parentheses. Different appearances across image sequences collected at different times pose challenges to modeling the environment and performing visual localization. More visualization can be found in Fig. III, Fig. X, and Fig. XIII.

	Scenes	Chess (6)	Fire (4)	Heads (2)	Office (10)	Pumpkin (8)	Kitchen (14)	Stairs (6)	Total
7-Scenes [52]	Train	4000	2000	1000	6000	4000	7000	2000	26000
	Test	2000	2000	1000	4000	2000	5000	1000	17000
Cambridge [22]	Scenes	College (8)	Hospital (9)	Church (14)	Shop (3)	-	-	-	Total
	Train	1220	895	1487	231	-	-	-	3833
MARS [26]	Scenes	“0011” (9)	“0015” (5)	“0037” (5)	“0041” (5)	-	-	-	Total
	Train	792	788	771	819	-	-	-	3170
	Test	186	172	225	204	-	-	-	787

## E.2. Two-Branch Joint APR Training

The RAP is trained with a batch size of  $B = 12$  on NVIDIA RTX A6000 GPUs with 48 GB of memory. To optimize training and save time, we employ an early stopping mechanism with a patience value of 200. Additionally, the learning rate is reduced by a factor of 0.95 whenever the validation loss plateaus, with this adjustment made every 50 epochs. The loss weights used during training are  $\beta_1 = 1, \beta_2 = 1, \beta_3 = 0.7$ . Following DFNet [9], we also include an additional triplet loss with the same weight with  $\beta_1$ . For every  $N = 20$  epoch, we randomly generate the same number of views as the training sample size using our appearance and pose augmentations. The model generally converges after approximately 1000 epochs.

## E.3. Matching-Based Post Refinement

We only use MAST3R’s coarse mode to obtain 2D-2D matches between the rendered RGB image and the query image to save time. Then, we back-project the rendered depth map from 3DGS into 3D space. In the following RANSAC-PnP [16, 17], we set the projection error to be 2 pixels. All other settings follow the defaults provided in the MAST3R repository.

## F. Evaluation Metrics

We use a widely accepted metric to assess and compare the localization performance of various methods: the median error in translation and rotation, defined as  $a$  cm and  $b^\circ$ , respectively. In the main manuscript, we also report the mean, maximum, and minimum errors to statistically compare the performance distribution across different methods.

## G. 7-Scenes [52]

All the benchmark metadata, including 7-Scenes, are shown in Table I, where the training set consists of 26000 images, and the test set contains 17000 images, with volumes between  $1 \text{ m}^3$  and  $18 \text{ m}^3$ . These images include texture-less surfaces, object occlusions, and motion blur.

**Table II. Quantitative comparison of image quality between 3DGS using DSLAM [41] and SfM [48] poses.** SfM poses produce more realistic synthetic images with better consistency, as indicated by higher PSNR values that reflect higher image quality.

PSNR ↑	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
DSLAM	19.98	19.04	17.23	20.89	19.20	18.92	18.93
SfM	<b>26.52</b>	<b>24.79</b>	<b>20.51</b>	<b>26.35</b>	<b>24.87</b>	<b>24.66</b>	<b>22.98</b>

## G.1. Visualization of Rendering Quality

Figure III shows the image rendering results of our method compared to the DFNet [9] method across various scenes in the 7-Scenes dataset. DFNet consistently exhibits blurred edges and artifacts in all scenes, primarily due to the low resolution of voxel density sampling in NeRF. When the sampling points are insufficient, edge details become fuzzy. In contrast, our method leverages the explicit 3DGS approach, effectively addressing this issue. The image quality achieved by our method is significantly better than that of NeRF-based methods. Furthermore, our deblurring technique ensures that object edges are clear and sharp, further enhancing the overall rendering quality.

## G.2. Ground Truth Pose Details

Besides presenting the evaluation performance using SfM ground truth poses of the 7-Scenes dataset, which enable the generation of higher-quality images [10], we also provide quantitative and qualitative results based on DSLAM [41] ground truth poses of the same dataset. As shown in Fig. III, the data indicates that SfM poses yield more accurate results than DSLAM poses, which produce noticeable artifacts along object edges. The quantitative results in Table II compare the image quality metric, i.e., PSNR between the two sets of poses, showing a significant improvement in image quality with SfM poses, which further enhances the performance of APR. Moreover, Table I highlights that our RAP model based on SfM poses demonstrates stronger localization capability, although RAP based on DSLAM poses already achieving state-of-the-art performance.

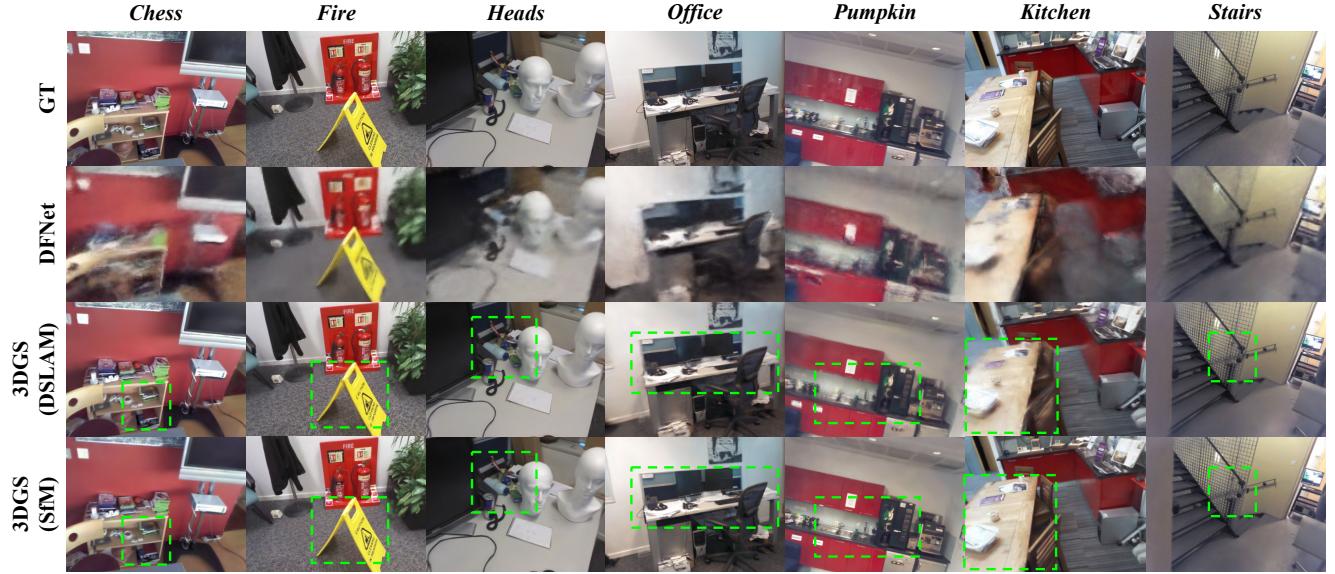


Figure III. Qualitative comparison of different NVS settings. Our 3DGS using SfM ground truth poses achieves the best visual quality, while using DSLAM poses results in blurry images, and the NeRF-based method produces the worst results.

### G.3. Additional Visualization of RAP

We present qualitative comparisons on the subsets of the 7-Scenes dataset in Fig. VII, comparing our method with DFNet [9]. Notably, in *Fire-seq-04*, *Pumpkin-seq-01*, and *Kitchen-seq-14*, our method avoids collapsing in certain regions, unlike DFNet, which generates a significant number of outliers. This demonstrates the strong generalization ability of our method.

### G.4. Additional Visualization of RAP<sub>ref</sub>

Figure VIII illustrates the intermediate steps involved in post-refinement. Specifically, after obtaining the initial pose estimation of the query image from RAP, we render the corresponding image and depth map through 3DGS. Then, we use MAST3R [25] to calculate the pixel correspondences between the two images. As shown in Fig. VIII, the matching lines before refinement are not sufficiently horizontal, indicating inaccuracies in the initial pose estimation. Next, we derive 2D-3D correspondences from the depth map and optimize the pose using a RANSAC-PnP [16, 17] solver. The final column of images demonstrates that the refined pose produces a rendered image almost indistinguishable from the original query image, with the matching lines now highly horizontal, highlighting the improved accuracy of the pose estimation. As shown in Fig. IX, the errors of our RAP<sub>ref</sub> are even less than 1 centimeter.

## H. Cambridge Landmarks [22]

### H.1. Effectiveness of Deblurring

Visual localization benchmarks are typically collected from video sequences, where motion blur between adjacent



Figure IV. Effectiveness of deblurring. The images in the second column, generated by 3DGS with deblurring capability, exhibit clearer and sharper edges than those produced without deblurring.

frames is inevitable. This negatively affects both the optimization of 3DGS and localization performance. To address this, we incorporate a deblurring module mentioned in Sec. 3.1, when optimizing 3DGS to mitigate these effects. As shown in Fig. IV, the deblurring module enhances the modeling of object edge details and removes artifacts, resulting in higher-quality data synthesis for APR training.

### H.2. 3DGS with Controllable Appearances

Figure X showcases images synthesized by our appearance-varying 3DGS on the Cambridge Landmarks dataset. The

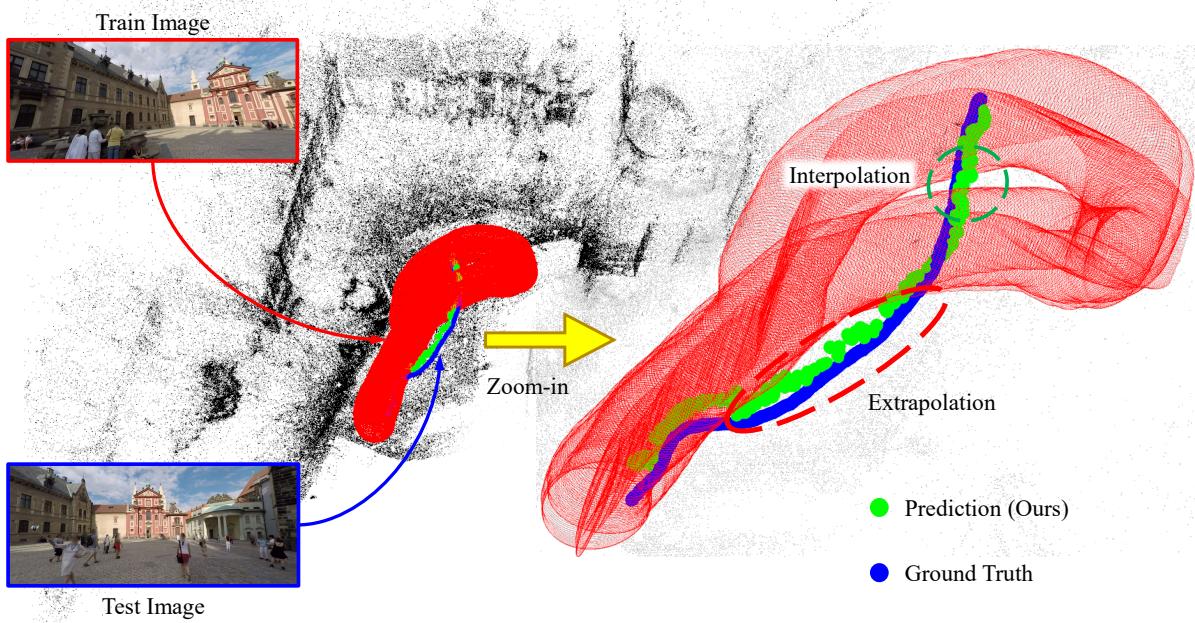


Figure V. Visualization of the training set distribution and results on *St. George’s Basilica Building* [47].

images rendered by our 3DGS exhibit finer details, such as sharper edges and textures, compared to those produced by DFNet [9], a NeRF-based method. These improvements contribute to better performance in both translation and rotation regression. Additionally, our 3DGS can model environments with varying weather conditions using multiple image sequences, enabling seamless interpolation between them. This allows for synthesizing more diverse images, aiding RAP in learning robust and invariant features and further enhancing pose regression performance.

### H.3. Additional Visualization of $\text{RAP}_{\text{ref}}$

The same process in  $\text{RAP}_{\text{ref}}$  on the Cambridge Landmarks dataset is shown in Fig. XI, with results in Fig. XII. Compared to indoor scenes, localization errors in outdoor scenes are significantly larger. This is attributed to inherent limitations in scene scale and image resolution. Even when the visual matching between the real and synthesized images appears nearly perfect to humans, as indicated by the image continuity near the diagonal in Fig. XII, errors can still occur within a single pixel in the image coordinate system. For fairness, we use the same resolution as DFNet [9].

## I. MARS [26]

### I.1. Ground Truth Pose Details

The GPS/IMU poses provided in the dataset are inaccurate, so we use COLMAP [48] poses as the ground truth and compute the scaling factor relative to the GPS locations to calculate the metric translation error.

### I.2. 3DGS with Controllable Appearances

The main challenges in driving scenarios include dynamic objects, such as vehicles and pedestrians, and dynamic environments with varying weather conditions. Fig. XIII shows that our appearance-varying 3DGS successfully models variations in ambient lighting. Notably, it can also capture dynamic elements in the scene, such as vehicles on the road.

### I.3. Additional Visualization of $\text{RAP}_{\text{ref}}$

Figure XIV and Fig. XV illustrate the same post-refinement process and results of  $\text{RAP}_{\text{ref}}$  on the MARS dataset. Our model successfully handles the challenges of varying appearances in autonomous driving scenarios. As shown in Fig. XV, although the ground truth images and rendered images along the diagonal often exhibit differences in appearance, this does not compromise localization accuracy, as evidenced by the continuity of the images.

## J. Emerging Generalization in APR

To better demonstrate the emergent generalization capability of the model under large-scale data training, we trained the model on the *St. George’s Basilica Building* [47] and visualized the results in Fig. V. Here, the translation perturbation was set to  $\delta t = 350$  cm and the rotation perturbation to  $\delta r = 60^\circ$ . Notably, the test set contained two regions entirely uncovered by the training set. With extensive synthetic data, our method successfully extrapolates beyond the training views and interpolates between them, as shown in the results. We also learn from our experiments that reduc-

Table III. Inference efficiency.

Method	PyTorch Mode	Avg FPS $\uparrow$
ACE [5]	With C++	50
	Eager	105
<b>RAP (Ours)</b>	Compiled	154
	Compiled reduce-overhead	187

ing the rotation perturbation, such that the overlap between test views and training views remains minimal, leads to poor localization performance. This is because the parameter space of translation and rotation is inherently a manifold in  $SE(3)$ . Even if the translation remains fixed, significant rotation changes result in entirely different visual content in the images, naturally preventing the model from estimating poses of such unseen views, which correspond to a large distance on the  $SE(3)$  manifold. Therefore, enabling generalization across a broader range of space is an important direction for future work.

## K. Inference Efficiency

As shown in Table III, our Python prototype of RAP achieves approximately 187 FPS with the `reduce-overhead` mode of `torch.compile`<sup>†</sup> on a laptop equipped with an NVIDIA RTX 4060 GPU running at 50 W and an Intel Core i9-13900H CPU, demonstrating real-time inference performance on compact devices. For  $RAP_{ref}$ , the post-refinement time per frame is 0.5 s on the same device, including RAP inference, 3DGS rendering with `gsplat` [64], MAST3R matching, and RANSAC-PnP [16, 17] solving using OpenCV. During this process, the GPU power consumption can reach 70–80 W. Additional implementation details will be provided in our code repository, which will be released soon.

## L. Failure Cases

Fig. VI presents several failure cases encountered during evaluation. Occlusions pose the most significant challenge for APR, particularly when dynamic objects are present. For example, in the second-row images, tree branches—absent in the 3DGS-synthesized image—appear during the inference stage, disrupting feature extraction. Additionally, textureless patterns in the image can degrade APR performance. For instance, in the third row, the stark contrast between the featureless sky and the building’s underexposed color creates ambiguities, posing challenges for feature extraction, potentially misleading the regression head, and impacting localization accuracy.

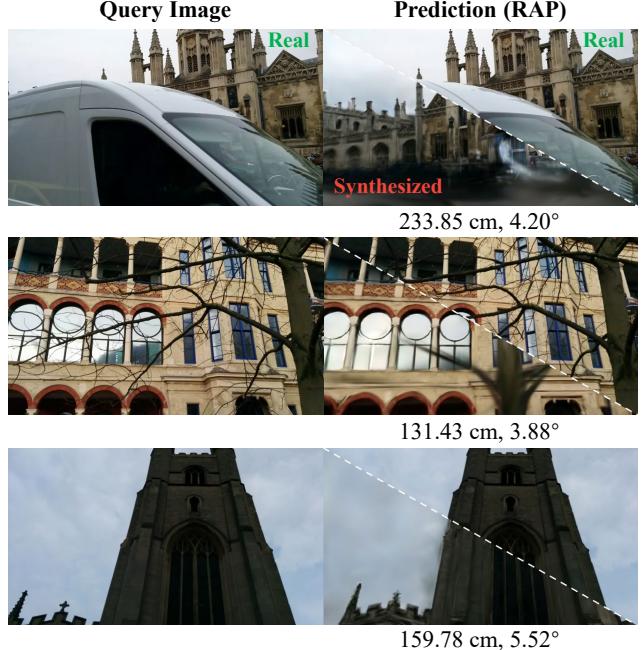


Figure VI. Failure cases. The primary reason for localization failure is occlusion, as shown in the first two rows. Additionally, textureless regions in the query image, such as the sky, can also result in significant errors.

<sup>†</sup>Timing measured using the function provided in [https://pytorch.org/tutorials/intermediate/torch\\_compile\\_tutorial.html](https://pytorch.org/tutorials/intermediate/torch_compile_tutorial.html); dataloader time is excluded.

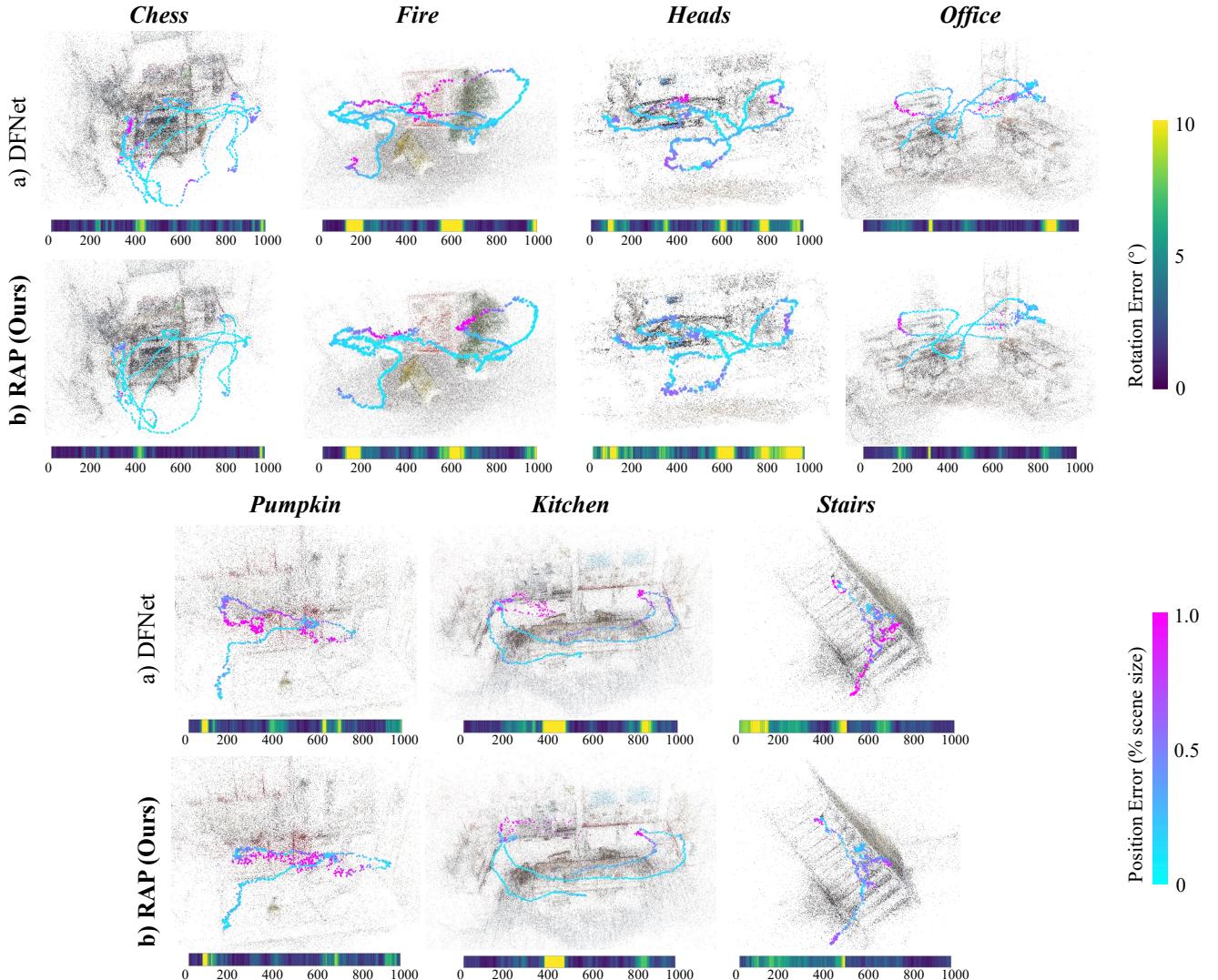
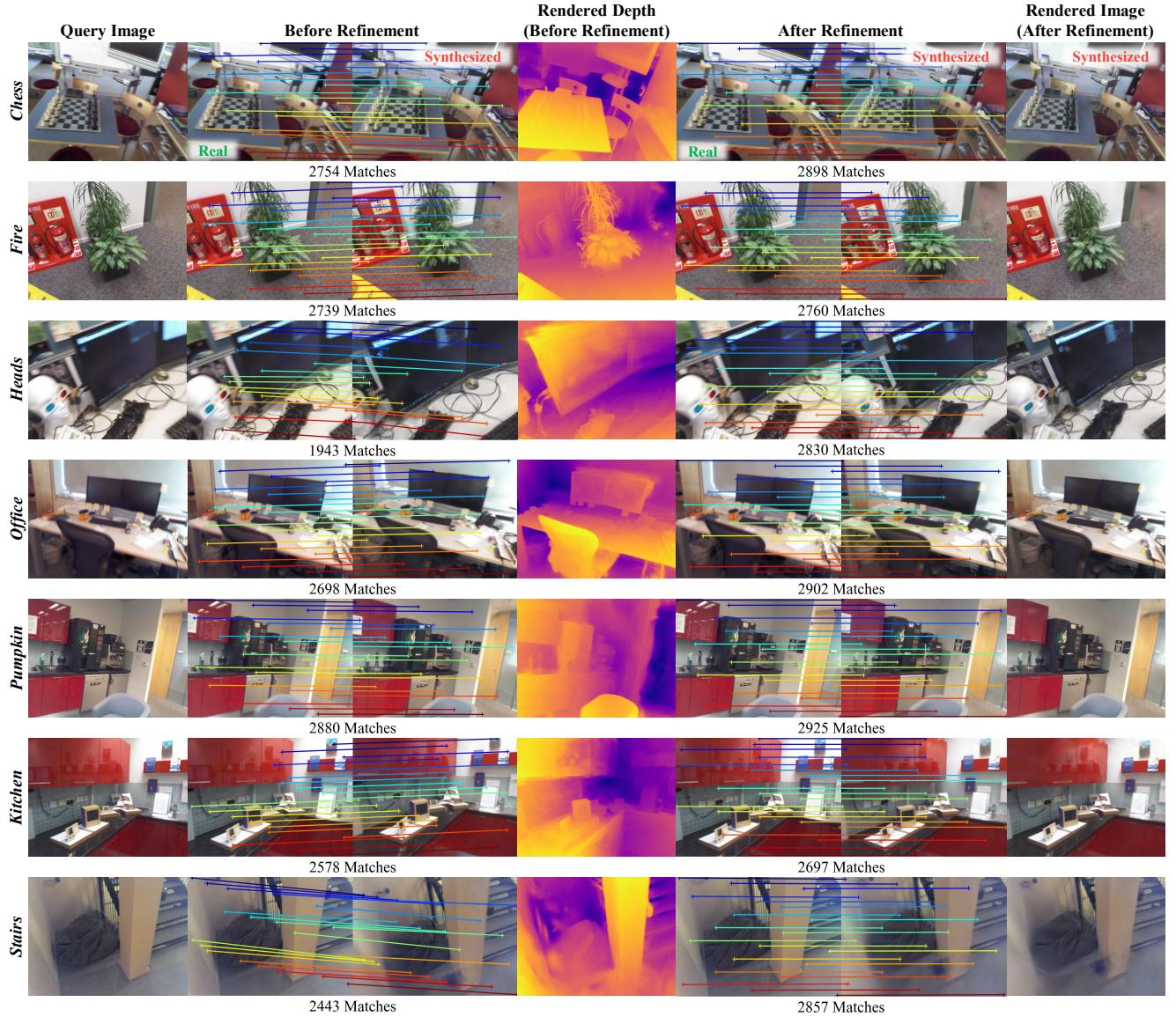
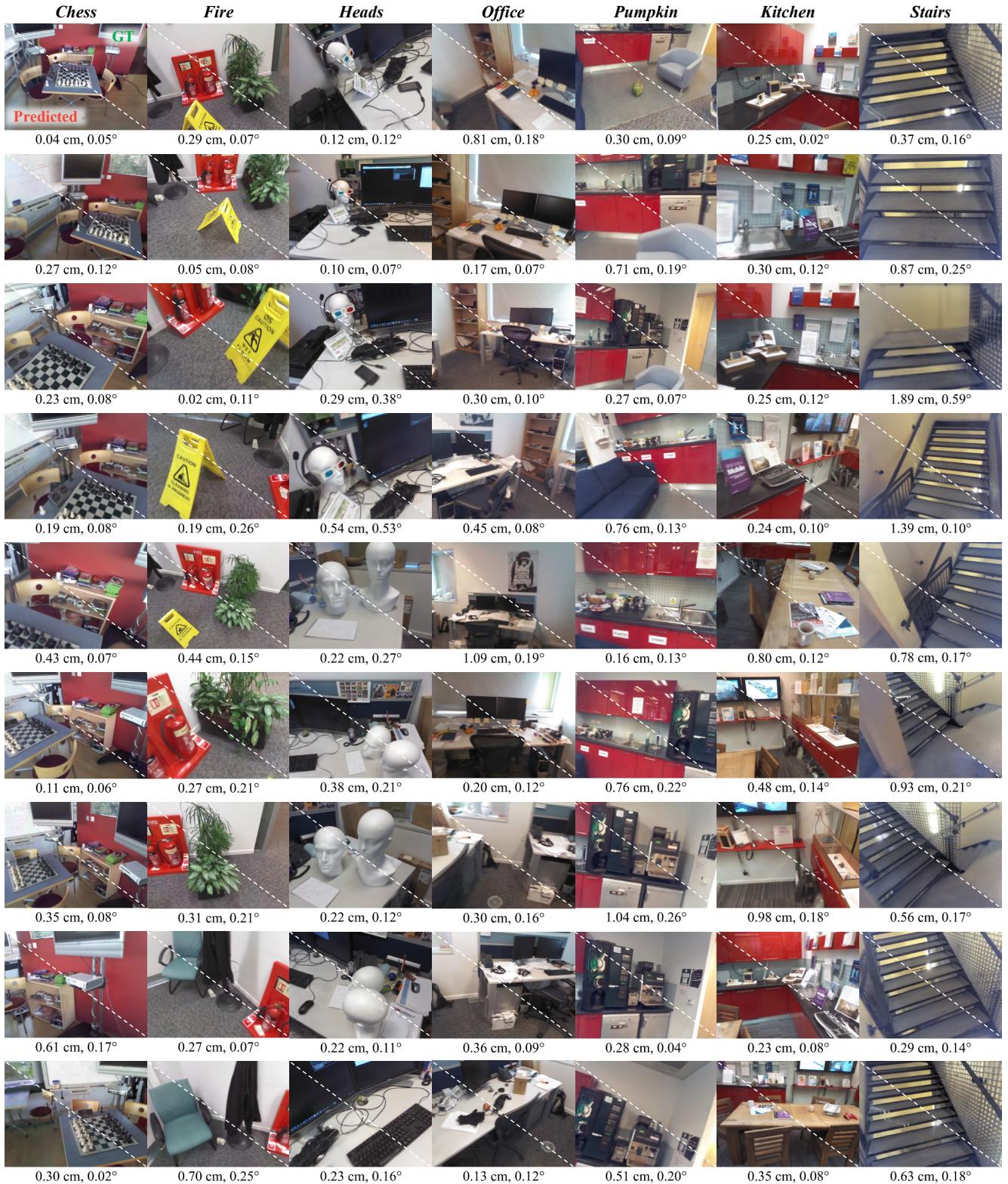


Figure VII. **Visualization of estimated camera poses on the 7-Scenes dataset [52].** Translation and rotation errors are indicated by the color of the error bars. Our RAP framework more closely follows the ground truth trajectory with fewer outliers compared to DFNet [9]. The sequences visualized are: *Chess-seq-03*, *Fire-seq-04*, *Heads-seq-01*, *Office-seq-07*, *Pumpkin-seq-01*, *Kitchen-seq-14*, and *Stairs-all*.



**Figure VIII. Visualization of the post-refinement pipeline on the 7-Scenes dataset [52].** Starting with the query image, we first obtain its initial pose from RAP, render it using 3DGS, and generate matches. The lines before refinement are not sufficiently horizontal due to inaccuracies in the initial pose. Next, we back-project the rendered depth to 3D and use RANSAC-PnP [16, 17] to compute a refined pose, which is then tested by rendering and matching again. The matches after refinement are horizontal, indicating that the refined poses are more accurate. Moreover, the rendered depth maps illustrate that our appearance-varying 3DGS successfully reconstructs the scene's geometric information, a critical factor in ensuring accurate 2D-3D correspondences.



**Figure IX. Visualization of localization errors on the 7-Scenes dataset [52].** Each subfigure is split by a diagonal line, with the **bottom-left** section showing the image rendered from the estimated or refined pose and the **top-right** section displaying the ground truth image. The continuity along the diagonal line demonstrates the satisfactory accuracy of the poses obtained by our RAP<sub>ref</sub>.

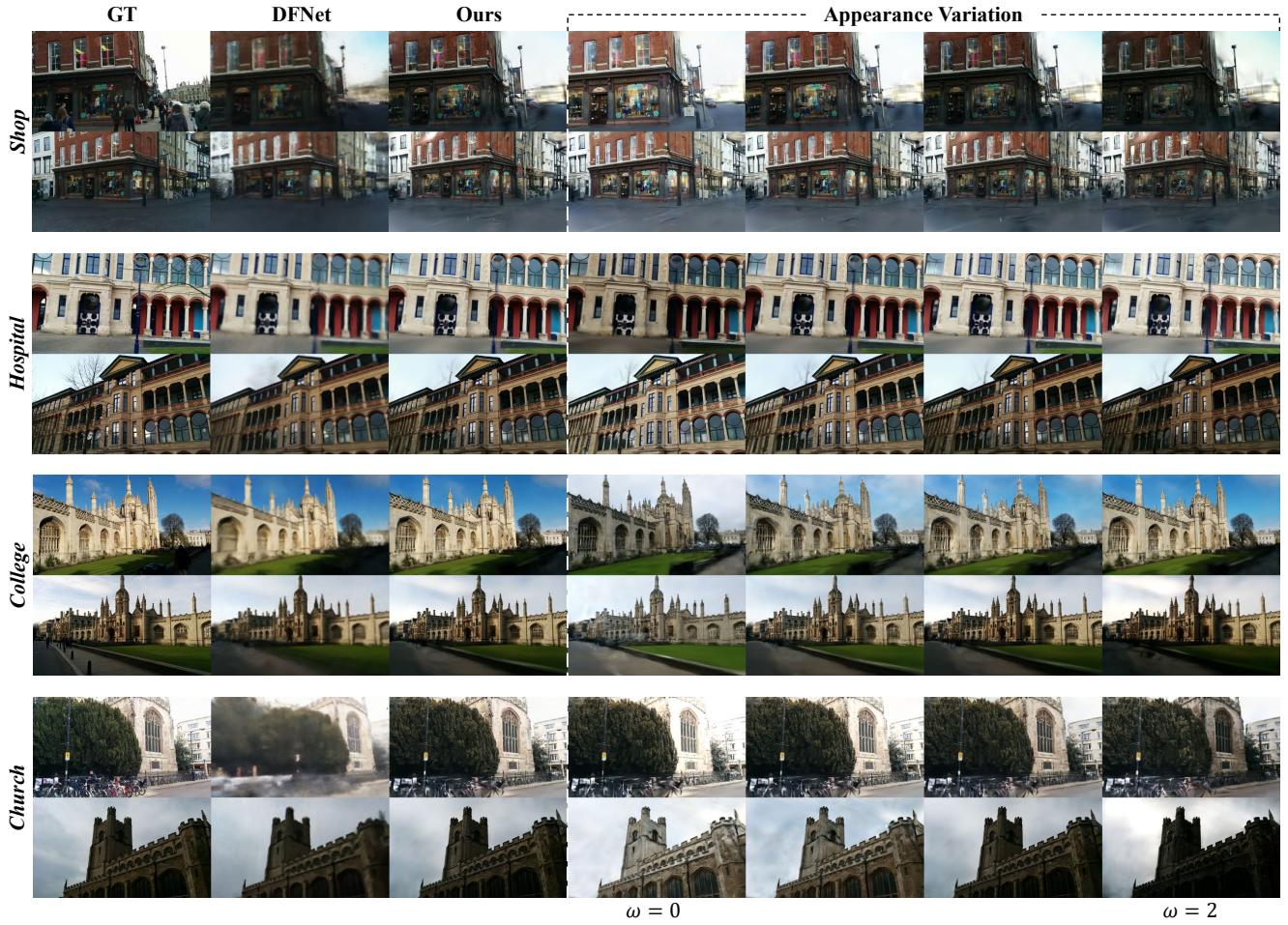


Figure X. **Synthetic images with varying appearances on the Cambridge Landmarks dataset [22]**. The appearances of synthetic images can be arbitrarily generated using different blending weights  $\omega$ , ranging from 0 to 2.

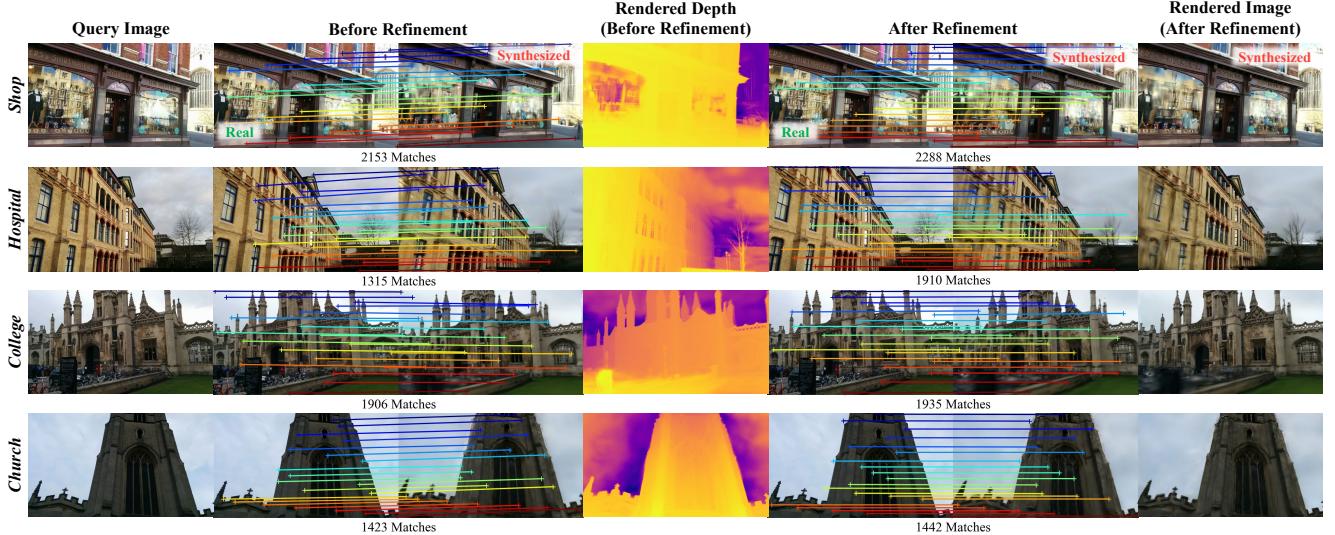


Figure XI. **Visualization of the post-refinement pipeline on the Cambridge Landmarks dataset [22]**. Starting with the query image, we first obtain its initial pose from RAP, render it using 3DGS, and generate matches. The lines before refinement are not sufficiently horizontal due to inaccuracies in the initial pose. Next, we back-project the rendered depth to 3D and use RANSAC-PnP [16, 17] to compute a refined pose, which is then tested by rendering and matching again. The matches after refinement are horizontal, indicating that the refined poses are more accurate.



Figure XII. **Visualization of localization errors on the Cambridge Landmarks dataset [22].** Each subfigure is split by a diagonal line, with the **bottom-left** section showing the image rendered from the estimated or refined pose and the **top-right** section displaying the ground truth image. The continuity along the diagonal line demonstrates the satisfactory accuracy of the poses obtained by our RAP<sub>ref</sub>.

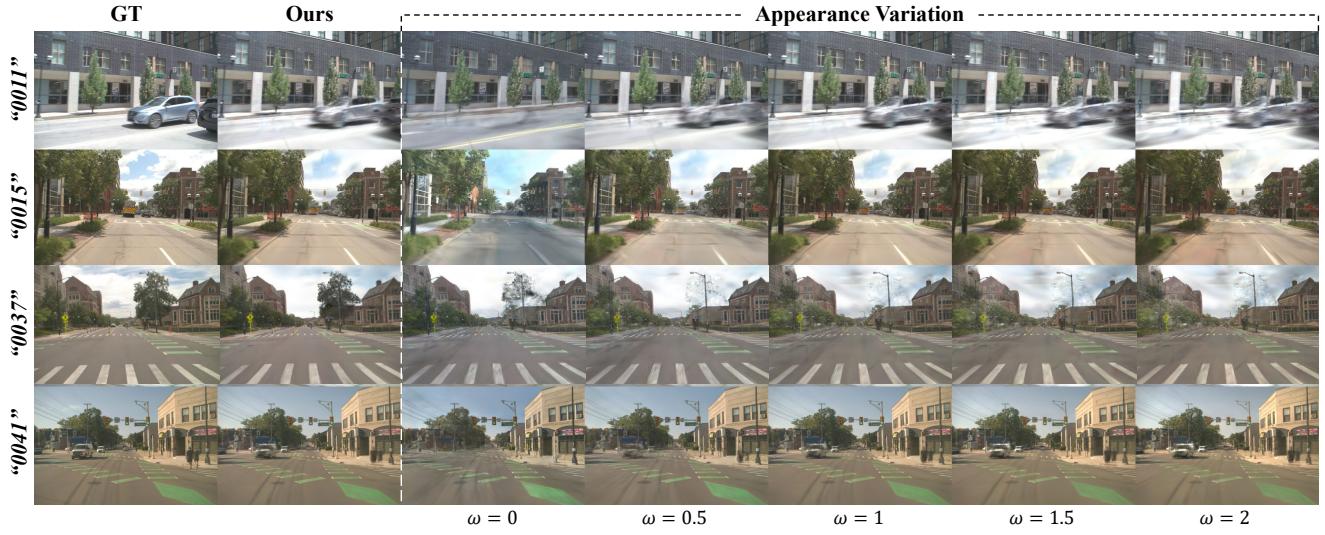


Figure XIII. **Synthetic images with varying appearances on the MARS dataset [26]**. The appearances of the synthetic images can be arbitrarily generated using different blending weights  $\omega$ , ranging from 0 to 2.

DFNet [9] results are not presented here because, despite our best efforts, we were unable to effectively train NeRF in DFNet. A likely reason is that NeRF requires the scene scale to be constrained within  $[-\pi, \pi]$ , which means the scaling factor must be manually aligned for each scene. This process can be labor-intensive and tedious, particularly for diverse outdoor scenes.

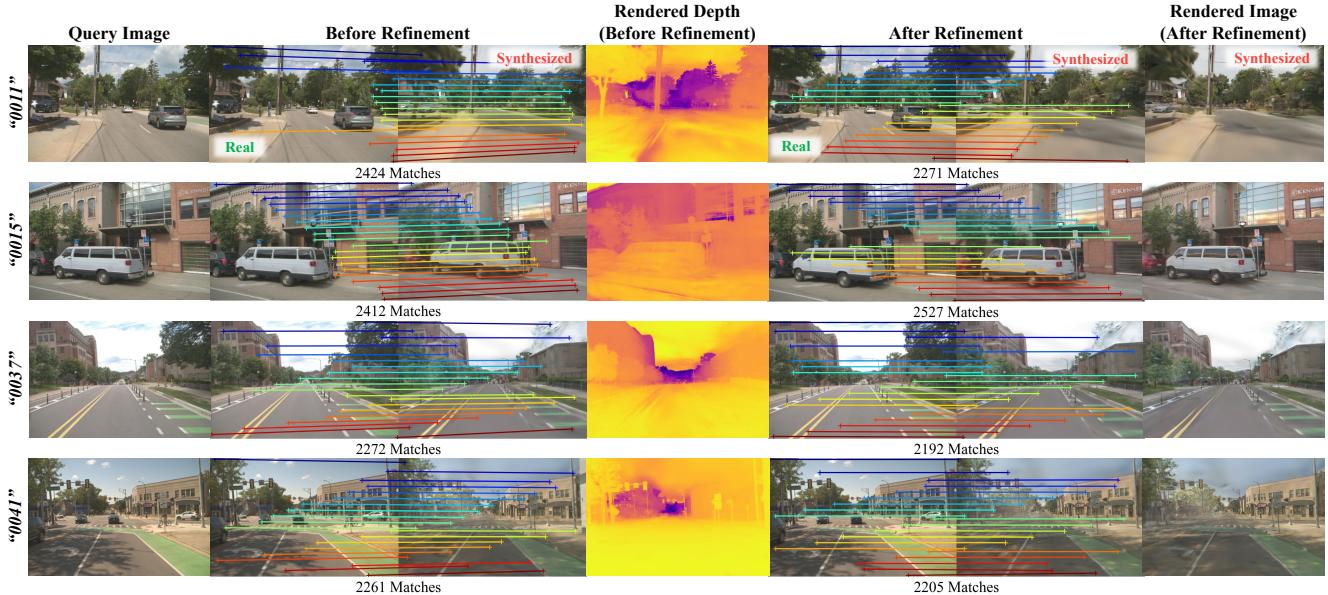


Figure XIV. **Visualization of the post-refinement pipeline on the MARS dataset [26]**. Starting with the query image, we first obtain its initial pose from RAP, render it using 3DGS, and generate matches. The lines before refinement are not sufficiently horizontal due to inaccuracies in the initial pose. Next, we back-project the rendered depth to 3D and use RANSAC-PnP [16, 17] to compute a refined pose, which is then tested by rendering and matching again. The matches after refinement are horizontal, indicating that the refined poses are more accurate. Moreover, the rendered depth maps illustrate that our appearance-varying 3DGS successfully reconstructs the scene's geometric information, a critical factor in ensuring accurate 2D-3D correspondences.



Figure XV. **Visualization of the localization errors on the MARS dataset [26].** Each subfigure is split by a diagonal line, with the **bottom-left** section showing the image rendered from the estimated or refined pose and the **top-right** section displaying the ground truth image. The continuity along the diagonal line demonstrates the improved accuracy of the poses obtained by our RAP<sub>ref</sub>.