GLACE: Global Local Accelerated Coordinate Encoding -Supplementary Material-

Fangjinhua Wang[1*] Xudong Jiang[1*] Silvano Galliani[2] Christoph Vogel[2] Marc Pollefeys[1,2]
[1]Department of Computer Science, ETH Zurich
[2]Microsoft Mixed Reality & AI Zurich Lab

1. Network Architecture

The detailed structure of our method is shown in Fig. 1. Our network has a fully-connected network architecture. By default, we set the hidden size h of all layers as 768, which is the size of the concatenation of local encoding (512) and global encoding (256).

In the beginning, the network takes as input the concatenation of local encoding and global encoding, and transforms it with a residual block which has 3 fully-connected layers. Then N residual blocks with 3 fully-connected layers are sequentially applied, where we choose N based on the scene scale. Specifically, we set $N = 1$ for all indoor scenes, and $N = 2$ for Cambridge Landmarks [4]. For Aachen Day [6, 7] dataset, we set $N = 3$ and double the hidden size h, *i.e.* 1536, for the second layer of each residual block. In addition, for evaluation of Aachen Day [6, 7] dataset with additional SuperPoint [3] feature, we set $N = 2$ and do not double the hidden size h in order to maintain a similar map size. Finally, we apply 3 fully-connected layers to get k logits $\{s_i\}$, one for each cluster center, and one homogeneous coordinate with parameters $\dot{d}, \hat{w}$ to define an offset. The final 3D coordinate $\hat{y}$ is estimated with Eq. 8 in the main paper.

To get the k cluster centers from training data, we cluster training camera positions with K-Means++ [1]. We set $k = 50$ for scenes that have a more multimodal distribution, including integrated rooms and Aachen Day dataset [6, 7], and $k = 1$ for scenes that have a more unimodal distribution, including individual scenes in the 7 Scenes [8], 12 Scenes [10] and Cambridge Landmarks [4].

2. Experiment Details

Following [2], we allocate a training buffer on the GPU, which stores local encodings and corresponding metadata, *i.e.* image indices and ground truth poses. This buffer is filled by iterating over the training images. Each image is first converted to grayscale and then subjected to a series of data augmentations: random scaling between $\frac{2}{3}$ and $\frac{3}{2}$, brightness and contrast jitter by 10%, and random rotations up to a maximum of $15°$. From each augmented image, we extract and uniformly sample 1024 local encodings. For the version using SuperPoint [3], importance sampling based on corner detection probability is employed instead. We also continuously update the training buffer during each training iteration when the number of training images is large.

Global features for each training image are extracted without any data augmentation and stored in a lookup table to avoid unnecessary duplication. During each training iteration, a batch of local encodings is randomly selected from the training buffer. Corresponding global encodings are then retrieved based on the image index. For these global encodings, we add Gaussian noise with a standard deviation of $\sigma = m = 0.1$, where m is the margin used in the triplet margin loss by the global feature extractor [11]. Subsequently, the global encodings are normalized back to the unit sphere.

We use AdamW [5] optimizer with a One Cycle learning rate scheduler [9] that increases the learning rate from $2 \cdot 10^{-4}$ to $5 \cdot 10^{-3}$ and then decreases to $2 \cdot 10^{-8}$. The detailed mapping times and number of GPUs for training is shown in Tab. 1.

Scene	Number of GPUs	Mapping Time
7 / 12 Scenes [8, 10]	1	6 min
Cambridge [4]	4	20 min
i12 / i19	8	1 h 50 min
Aachen Day [6, 7]	8	2 h 30 min

Table 1. Mapping Times of our method on different scenes. We use Nvidia Quadro RTX 6000 GPUs in experiments.

During evaluation, we use a 10px inlier threshold and 64 RANSAC hypotheses for all experiments, except that we use 3200 RANSAC hypotheses for Aachen Day [6, 7] dataset to match the number of RANSAC hypotheses of the ACE [2] $\times$ 50 baseline.

*Equal contribution.

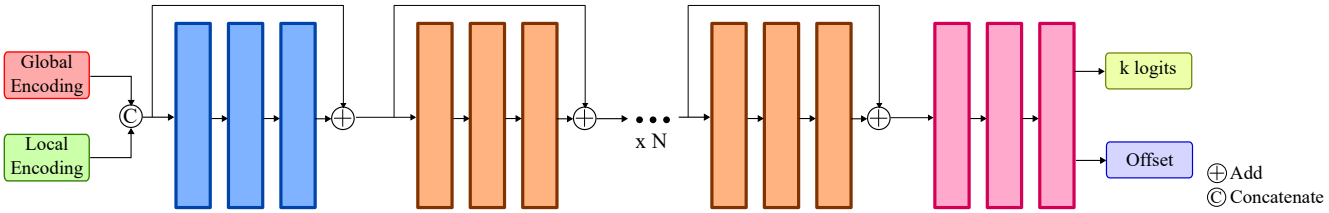

Figure 1. Detailed structure of our fully-connected network architecture for GLACE. In the beginning, a residual block (blue) transforms the concatenation of global and local encodings, which is followed by N sequential residual blocks (orange). Finally, three fully-connected layers (pink) are applied to get the k logits and offset for estimating the 3D position.

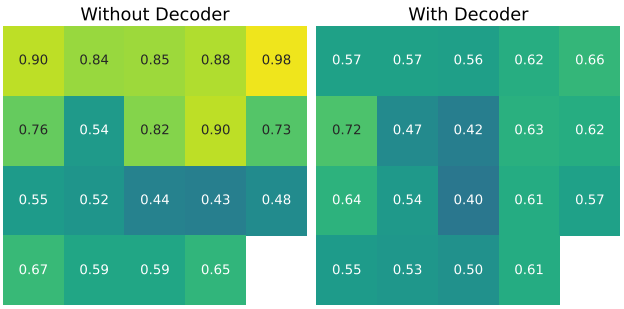

Figure 2. Comparison about mean absolute error of pixel location prediction in our 2D toy example.

3. Position Decoding in 2D Toy Example

We designed a simplified 2D toy example to show the effect of our position decoder. We randomly select 19 images from the 7 Scenes [8] and 12 Scenes [10] datasets and place them in a grid with a similar layout as the i19 scene. The images are resized and cropped to a size of 480 x 640 for convenient batch processing. We use the same pretrained ACE [2] encoder and train the MLP head with similar architecture, except that the output coordinate is now 2D instead of 3D. We use 19 decoder cluster centers, which are actually the centers of the 19 images. The output coordinate is directly supervised by the ground truth pixel location. Fig. 2 shows that, even for this simple example with strong supervision, our position decoder can allow the model to fit the training data with a multi-modal output distribution better.

4. Reconstruction Visualization

In Fig. 3, 4 and 5, we visualize the implicit reconstructions by accumulating the predicted 3D scene coordinates of the training images, and filter the outliers according to a 5px reprojection error threshold. The point cloud color is obtained from the center pixel of each image patch. As we can see, the implicit triangulation allows the model to learn meaningful 3D structures from reprojection loss only.

(a) i12 (b) i19

Figure 3. Reconstructions of integrated rooms.

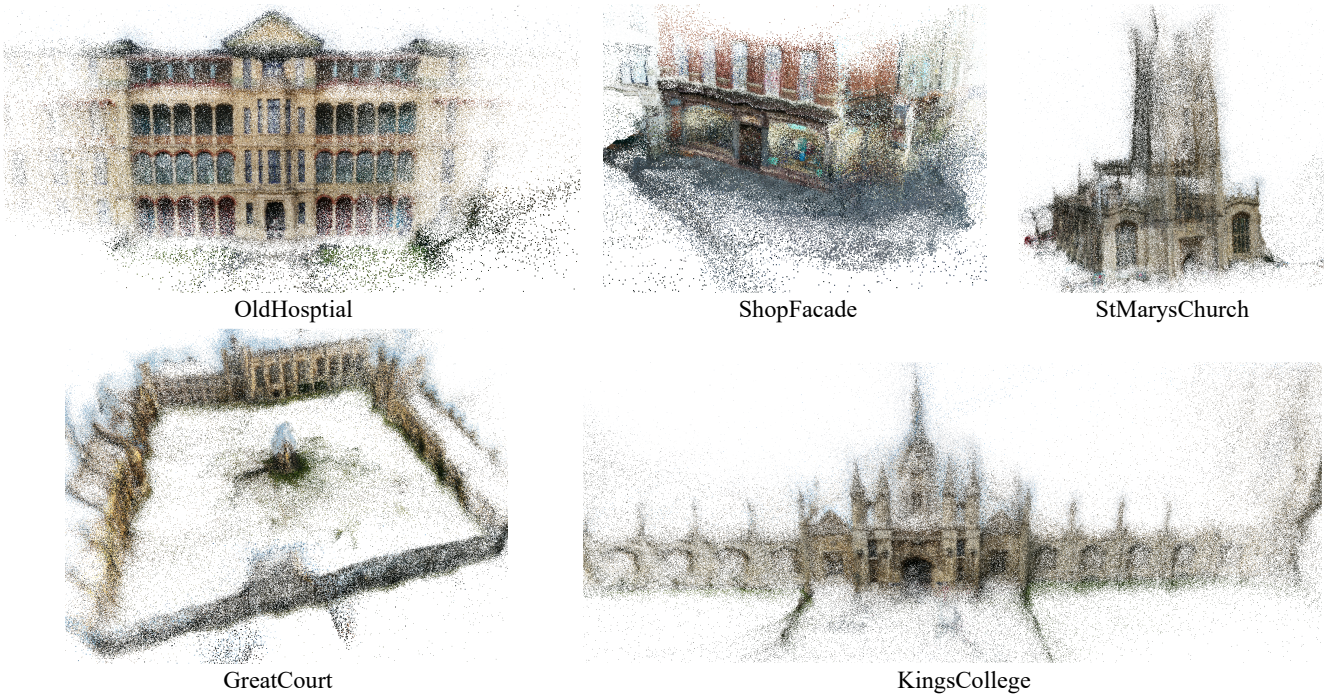

Figure 4. Reconstructions of Cambridge Landmarks [4].

Figure 5. Reconstrucion of Aachen Day dataset [6, 7].

References

[1] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. pages 1027–1035, 2007. 1

[2] 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

[3] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, 2018. 1

[4] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-DOF camera relocalization. In *CVPR*, 2015. 1, 4

[5] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 1

[6] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, 2012. 1, 4

[7] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *CVPR*, 2018. 1, 4

[8] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, 2013. 1, 2

[9] Leslie N. Smith and Nicholay Topin. Super-convergence: very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, 2019. 1

[10] Julien Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. In *3DV*, 2016. 1, 2

[11] Sijie Zhu, Linjie Yang, Chen Chen, Mubarak Shah, Xiaohui Shen, and Heng Wang. R^2former: Unified retrieval and reranking transformer for place recognition. *CoRR*, abs/2304.03410, 2023. 1