LoCUS: Learning Multiscale 3D-consistent Features from Posed Images Supplementary Material

Dominik A. Kloepfer[1], Dylan Campbell[2], João F. Henriques[1]
[1]Visual Geometry Group, University of Oxford
[2]Australian National University
{dominik, joao}@robots.ox.ac.uk, dylan.campbell@anu.edu.au

A. Memory Management

The training objective $\overrightarrow{\text{AP}}$ requires the calculation of the similarity differences $s_{ij} - s_{ik}$. As Brown et al. [1] point out, this can be computed efficiently in a single matrix of shape $\sum_{ij} y^{+}_{ij} \times \sum_{ij} y^{\Omega}_{ij}$. However, in our setting this matrix can quickly become very large. We found empirically that, for the Matterport3D dataset [2] and for a landmark radius $\rho_j = 0.2m$ and a multiplier for the "don't care" region $\kappa = 50$, so $\kappa\rho_j = 10.0m$, around 1% of all extracted pixel patches are positive patches for a given tentative landmark ($y^{+}_{ij} = 1$) and around 75% of all extracted pixel patches are not within a "don't-care"-region ($y^{\Omega}_{ij} = 1$). This means that, in the case of $|\mathcal{L}| = 64$ tentative landmarks, and 16 images in a batch from which 30×40 pixel patches are extracted, the similarity matrix has shape $(0.01 \cdot 64 \cdot 16 \cdot 30 \cdot 40 \times 0.75 \cdot 64 \cdot 16 \cdot 40 \cdot 40$, so a total of $1.1 \cdot 10^{1}0$ elements. Using single-precision floating point numbers, this results in a total memory consumption of 45 GB, significantly more than the 11 GB of memory available in the NVIDIA GeForce RTX 2080 Ti we are using for our experiments.

We reduce the size of this matrix to manageable levels by uniformly randomly discarding a fraction $1 - f$ of pixel patches. This effectively reduces the size of the matrix of similarity differences by a factor of f^2, as it reduces both the total number of patches with $y^{+}_{ij} = 1$ and the number of patches with y^{Ω}_{ij} by a factor f each. In expectation this does not change the shape of the loss landscape, though a smaller fraction f makes the training more noisy and therefore slightly slower.

B. Ablation Study

Table 1: Place recognition (retrieval) results for the proposed method with different parameters. Note that a higher sigmoid temperature τ will even for identical networks reduce the value of $\overrightarrow{\text{AP}}$.

	Objective ($\overrightarrow{\text{AP}}$)		Average Precision (AP)	
Model	Train	Val.	Train	Val.
64 dim, $\tau = 0.001, \rho_j = 0.2m$	0.58	0.57	0.56	0.58
64 dim, $\tau = 0.1, \rho_j = 0.2m$	0.29	0.29	0.51	0.48
64 dim, $\tau = 0.01, \rho_j = 0.6m$	0.59	0.58	**0.60**	**0.58**
64 dim, $\tau = 0.01, \rho_j = 1.0m$	0.59	0.56	0.58	0.58
64 dim, $\tau = 0.01, \rho_j = 0.2m$ *	0.56	0.54	0.57	0.55
128 dim, $\tau = 0.01, \rho_j = 0.2m$	0.58	0.54	0.56	0.55

* Values used in the main paper

Table 2: Semantic and instance segmentation (respectively "stuff" and "things") results, with object re-identification, for the proposed method with different parameters. We bold the best results for models that extract 64 feature dimensions, and where the model extracting 128 feature dimensions outperforms all others.

	Semantic			Instance			Overall		
Model	mAP	mIoU	Jac	mAP	mIoU	Jac	mAP	mIoU	Jac
64 dim, $\tau = 0.001, \rho_j = 0.2m$	0.41	0.22	0.54	0.41	0.24	0.33	0.41	0.25	0.53
64 dim, $\tau = 0.1, \rho_j = 0.2m$	0.33	0.18	0.47	0.40	0.27	0.24	0.40	0.28	0.48
64 dim, $\tau = 0.01, \rho_j = 0.6m$	0.52	0.30	0.60	**0.56**	**0.40**	**0.50**	**0.56**	**0.40**	**0.62**
64 dim, $\tau = 0.01, \rho_j = 1.0m$	0.49	0.28	0.58	0.54	0.37	0.49	0.54	0.37	0.61
64 dim, $\tau = 0.01, \rho_j = 0.2m$*	**0.53**	**0.37**	**0.67**	0.54	**0.40**	0.42	0.54	0.39	0.59
128 dim, $\tau = 0.01, \rho_j = 0.2m$	0.53	0.32	0.61	**0.60**	**0.46**	0.49	**0.61**	**0.46**	**0.63**

* Values used in the main paper

Table 3: Relative pose estimation results for the proposed method with different parameters. We report translation errors in meters and rotation errors in degrees.

	Translation			Rotation		
Model	Med.	Avg.	$\leq$ 1m	Med.	Avg.	$\leq 30°$
64 dim, $\tau = 0.001, \rho_j = 0.2m$	1.03	1.85	0.49	26.9	37.8	0.53
64 dim, $\tau = 0.1, \rho_j = 0.2m$	1.62	2.22	0.33	46.1	54.8	0.37
64 dim, $\tau = 0.01, \rho_j = 0.6m$	1.02	1.69	0.49	24.5	36.4	0.56
64 dim, $\tau = 0.01, \rho_j = 1.0m$	1.19	1.85	0.44	33.0	43.4	0.47
64 dim, $\tau = 0.01, \rho_j = 0.2m$*	**0.92**	**1.69**	**0.53**	**22.1**	**34.5**	**0.58**
128 dim, $\tau = 0.01, \rho_j = 0.2m$	0.94	1.76	0.52	23.6	35.4	0.57

* Values used in the main paper

We report the results of two additional values for the radius of each landmark $\rho_j \in \{0.6m, 1.0m\}$ and two additional values for the temperature of the sigmoid function $\tau \in \{0.001, 0.1\}$. The method used in the paper used a landmark radius of $\rho_j = 0.2m$ and a sigmoid temperature of $\tau = 0.01$. Aside from changing these two values, we keep all other setting the same, including the number of training epochs after which we evaluate the model performance (20 epochs).

We also report the results of a version of our method that uses 128 feature dimensions (rather than the 64 dimensions used in the paper) but otherwise keeps the network architecture constant.

B.1. Results

Increasing the ρ_j increases the number of positive pixel patches (increases $\sum_{ij} y_{ij}^{+}$), and thus increases the size of the matrix of similarity differences (see Sec. A for details). In an effort to keep the overall memory consumption during training constant, this required us to reduce the fraction of pixel patches that are retained from 8.6% for the method used in the paper to 6.3% for $\rho_j = 0.6m$ and to 0.51% for $\rho_j = 1.0m$. Similarly, when increasing the feature dimension, we had to decrease the fraction of retained pixel patches to While this random subsampling does not change the shape, one would expect it to lead to more noisy and/or slower training, accounting for some of the reduction in performance.

The results of our ablation study are listed in tables 1, 2, and 3. As one can see, the settings of $\rho_j = 0.2m$ and $\tau = 0.01$ outperform the other investigated values on the relative pose estimation task, though a landmark radius of $\rho_j = 0.6m$ slightly outperforms them on the object instance segmentation task.

This points to the different tasks benefiting from different landmark radii – a landmark size that is more similar to the size of objects in the environment improves performance on the segmentation tasks, while the reliance of our relative pose estimation algorithm on accurate pixel matches means that smaller landmarks (resulting in more accurate matches) yield better results there. Depending on the downstream task in question, one can therefore easily adjust the scale on which extracted features vary by adjusting the landmark radius ρ_j, which is an advantage of our proposed method.

We also report the results of using the same model (frozen DINO [**?**] backbone with two ViT blocks with an internal feature dimension of 128) but extracting feature vectors with a dimension of 128 for each pixel patch. This increases the expressiveness of the features, though it does not increase the network capacity. Other than increasing the feature dimension, we keep all other training settings constant, and again train this model for 20 epochs. The increased feature dimension does not increase the model's performance in the place recognition (Tab. 1) or relative pose estimation (Tab. 3) tasks, though there is a noticeable jump in performance in the object segmentation task (Tab. 2). This suggests that the network capacity (or perhaps the information retained by the pre-trained and frozen DINO feature extractor) is the limiting factor in the network learning to infer the relative locations in 3D of different pixel patches. In contrast, even when the larger feature vectors do not contain any more information about pixel patches' 3D locations than those with 64 dimensions, the additional dimensions will generally make the segmentation task with a linear probe easier (as long as the larger feature vectors span more than 64 dimensions).

C. Pixel Match Filtering for Relative Pose Estimation

To calculate the essential matrix from two RGB images A and B, from which the relative pose can be derived (as in **??**), the 5-point RANSAC algorithm we use [5] requires a set of pixel matches between the two images, that is, a set of pairs of pixel-coordinates (p_A, p_B) that are projections of the same point in 3D. In the following we lay out our method of generating these pixel patches in more detail.

First, we use the network trained with our proposed method to extract a set of features θ for each pixel (or, strictly speaking, patch of pixels) in both A and B.

We then begin the process of generating pixel matches by matching each pixel $p_i \in A$ with the pixel $q_j \in B$ whose feature θ_j has the highest cosine-similarity with θ_i. More formally, $q_j = m(p_i) = \arg\max_{q_k \in B}(\theta_i \cdot \theta_k)$. The matching function $m : A \to B$ is neither injective nor surjective. This process on its own returns a lot of false matches – even aside from inaccuracies made by the network, there generally are a lot of pixels $p_i \in A$ outside of any overlap between images A and B, meaning that there exists no good match for them in image B. While the RANSAC element in the algorithm is robust to the inclusion of a limited number of outliers, too many wrong matches quickly lead to deteriorating results.

We therefore filter the set of matches $\{(p_i, m(p_i))\forall p_i \in A\}$ using three criteria, all of which need to be fulfilled for a pixel match to be included in the final set of pixel matches used for camera pose estimation.

1. We require that the similarity between the features of the matched pixels is greater than a threshold, $\theta(p_i) \cdot \theta(m(p_i)) \geq \theta^\star$. In our experiments, a value of $\theta^\star = 0.7$ yielded good results.

2. We require that the pixels in the neighbourhood $\mathcal{N}$ of p_i are matched to pixels in B that are not too far from $m(p_i)$ in image B. More formally, we require that $\mathbb{E}_{p_k \in \mathcal{N}}[|m(p_i) - m(p_k)|] \leq d^\star$, with $d^\star = 3$ pixel.

   This approach has a similar intuition to Sattler et al. [6], who reject a match if the fraction of the neighbours of p_i that are mapped to points within a certain distance of $m(p_i)$ is below a threshold. We found however that, in our case, the more aggressive sampling using the mean distance led to better localisation results.

3. We use the intuition that, for a good pixel match, the pixel in A that has the highest similarity with $m(p_i) \in B$ should be in the vicinity of p_i. That is, with $m' : B \to A$ and $m'(q_j) = \arg\max_{p_k \in A}(\theta_j \cdot \theta_k)$ the function that matches all

pixels in B to their most similar pixel in A, we require that $|p_i - m'(m(p_i))| \leq r^\star$. In our experiments we used a value of $r^\star = 12$ pixels.

To further increase robustness, we repeat the same process after swapping images A and B, leading us to a second set of filtered pixel matches that is then concatenated with the first set.

D. Qualitative Results for Consistent Object Segmentation

As part of the supplementary material, we also generate a short video snippet showing object segmentation for a camera moving through a previously unseen environment. We generate the underlying images by rendering RGB images at a number of camera poses from one of the Matterport3D test-scenes using the Habitat [7, 4] simulator. We then extract features from these images using our trained network (64 feature dimensions, $\tau = 0.01$, $\rho_j = 0.6m$) and classify them into individual object instance or 'stuff' classes using a linear probe trained using the images from that scene in the original Matterport3D dataset. To improve the quality of the visualisation, we refine the segmentation masks using a Conditional Random Field [3]. The segmentation of each frame are thus generated independently from each other.

One can clearly see the stability of the classification of the same objects (e.g., couch, table, chair, painting, etc.) even from very different view-points. This shows that the extracted features are stable under viewpoint changes for a given location in 3D. Rather than merely recognising the table as "a table", the network is able to recognise it as *the same* table as in different images. In the places where the segmentation does briefly confuse one object for another, it generally confuses two objects of the same type and appearance, such as two windows or two pictures on a wall.

References

[1] Andrew Brown, Weidi Xie, Vicky Kalogeiton, and Andrew Zisserman. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *Proc. ECCV*, 2020. 1

[2] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. 1

[3] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. 4

[4] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 4

[5] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004. 3

[6] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Scramsac: Improving ransac's efficiency with a spatial consistency filter. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2090–2097, 2009. 3

[7] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 4