Generic 3D Representation via Pose Estimation and Matching supplementary material

Amir R. Zamir[1,2] Tilman Wekel[1] Pulkit Agrawal[2] Colin Wei[1]
Jitendra Malik[2] Silvio Savarese[1]

[1] Stanford University [2] University of California, Berkeley

http://3Drepresentation.stanford.edu/

Abstract. In the supplementary material, we provide the following items:
1) Experiments on joint (vs single) feature learning,
2) Experiments on Brown el. and and Mikolajczyk&Schmid Benchmarks,
3) More details about the dataset, including pixel alignment procedure, sample data, and coverage,
4) Evaluation of surface normal estimation on NYUv2 dataset,
5) Joint embedding of synthetic cubes and images,
6) Illustration of pose induction via tSNE embeddings of more ImageNet classes and MIT Places.
7) More details on the training procedure,

1 Joint Feature Learning

We investigated different aspects of joint learning the representation and information sharing among the two supervised tasks in Tables 1 and 2. To quantify the amount of information shared among the matching and pose estimation tasks, we trained a single-task network dedicated to each problem; their error for their respective task is reported in "Direct" row of the Table 1. "Transduction" provides the error rate when a linear classifier was trained on the frozen representation of one task to solve the other task. The fact that the Transduction setup achieves a reasonable performance suggests the two problems have a great deal of shared information in their representations. Table 2 compares the performance of single vs multi-task networks. The multi-task network performs comparable to its dedicated counterparts showing it encoded both problems with no performance drop.

2 Brown et al. Benchmark.

We evaluated the performance of our representation on the benchmarks of Brown et al. [3] and Mikolajczyk & Schmid [9] (next subsection). Compared to our

Table 1: Information Sharing Among Supervised Tasks

	Matching (FPR)	Pose (Error)
Direct	23.54%	16.58°
Transduction	30.06%	24.50 °
Chance	95%	90 °

Table 2: Joint vs Individual Learning

	Matching (FPR)	Pose (Error)
Pose-Net	-	16.58°
Matching-Net	23.54%	-
Joint-Net	23.0%	17.78 °

dataset, these benchmarks mostly include narrower baselines (except for a subset of [9]), and therefore, do not pronounce wide baseline handling abilities; our method also sees more training data than the baselines. However, these benchmarks would reveal 1) if our representation was performing well only on street view scenery, and 2) if wide baseline handling capability was achieved at the expense of lower performance on small baselines.

We compared our results with six baselines, including Zagoruyko & Komodakis's [15] and MatchNet [6], against their most similar descriptor dimensionality (512) and network architecture to ours. For this experiment, we mixed our training dataset with the corresponding training split of Brown's (see Table 3). Our representation outperforms the baselines, except for two splits from the Yosemite National Park that are substantially foliage covered (for which we speculate that the foliage coverage is reason since our ConvNet is mostly agnostic with respect to foliage as trees are uninformative for matching and pose in street view scenery).

Table 3: Evaluations on Brown's Benchmark [3]. FPR@95 (↓) is the metric.

Train	**Test**	MatchNet [6]	Zagor. siam [15]	Simonyan [11]	Trzcinski [12]	Brown [3]	Root-SIFT [2]	**Ours**
Yos	ND	7.70	5.75	6.82	13.37	11.98	22.06	**4.17**
Yos	Lib	13.02	13.45	14.58	21.03	18.27	29.65	**11.66**
Lib	ND	4.75	4.33	7.22	14.15	N/A	22.06	**1.47**
ND	Lib	8.84	8.77	12.42	18.05	16.85	29.65	**7.39**
Lib	Yos	13.57	14.89	**11.18**	19.63	N/A	26.71	13.78
ND	Yos	11.00	13.23	**10.08**	15.86	13.55	26.71	12.30
mean		9.81	10.07	10.38	17.01	15.16	26.14	**8.46**

Table 4: Evaluation on Mikolajczyk & Schmid's [9]. The metric is mAP(↑).

Transf. Magnitude	**1**	**2**	**3**	**4**	**5**
SIFT [8]	40.1	28.0	24.3	29.0	17.1
Zagor. [15]	43.2	37.5	29.2	28.0	16.8
Fischer et al [5]	42.3	33.9	26.1	22.1	14.6
Ours-rectified	46.4	**41.3**	29.5	23.7	17.9
Ours-unrectified	**51.4**	37.8	**34.2**	**30.8**	**20.8**

Mikolajczyk & Schmid Benchmark. The evaluation results on the benchmark of Mikolajczyk & Schmid [9] are provided in Table 4 using the standard protocol [9,15]. Following [15,5], we performed the matching on MSER features. The last two rows show our results on the MSER patches with and without rectification (i.e., skipping MSER rectification). Our representation outperforms the baselines in both cases, while not performing the rectification actually improves the performance.

3 Dataset and Data Collection Details

Our dataset was collected from a broad geographical area spanning multiple cities. Some representative city locations from which data was collected are shown in Figure 1.

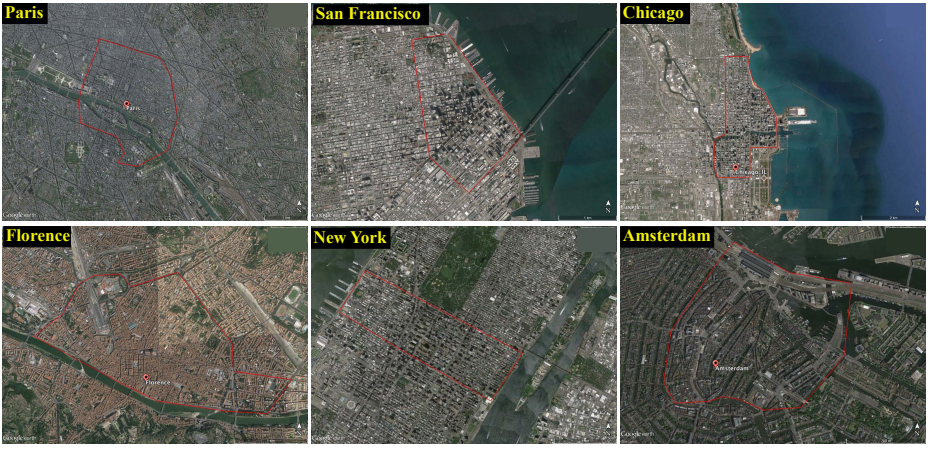

Fig. 1: Some of the areas from which we have collected our dataset.

Sample images from our dataset that were used for training the pose estimation and patch matching network are shown in Figures 2,3. Each row in the figure shows a single target location from different viewpoints. The target location is marked with a red dot.

3.1 Pixel Alignment and Pruning

The data collection system required integration of multiple resources, including GPS from street view, elevation maps, and 3D models. Any slight inaccuracy in the metadata or 3D models can cause a pixel misalignment in the collected images. Therefore, we performed the following pose-processing procedure to cancel some of these errors.

We wish to verify if the center of images in a bundle show the same physical target and adjust them if necessary. The collected image bundles can show large (often $> 100°$) viewpoint changes, while the existing registration methods do not handle such large angular changes in unconstrained scenes. To solve this issue, we utilize the metadata again: we extract the relative pose of the cameras to the desired physical point's surface normal (acquired from the 3D models). Then a homography transformation is applied to project one patch in the bundle onto another in a way that the local image planes on which the desired physical point lies are parallel. This cancels the perspective transformation caused by the

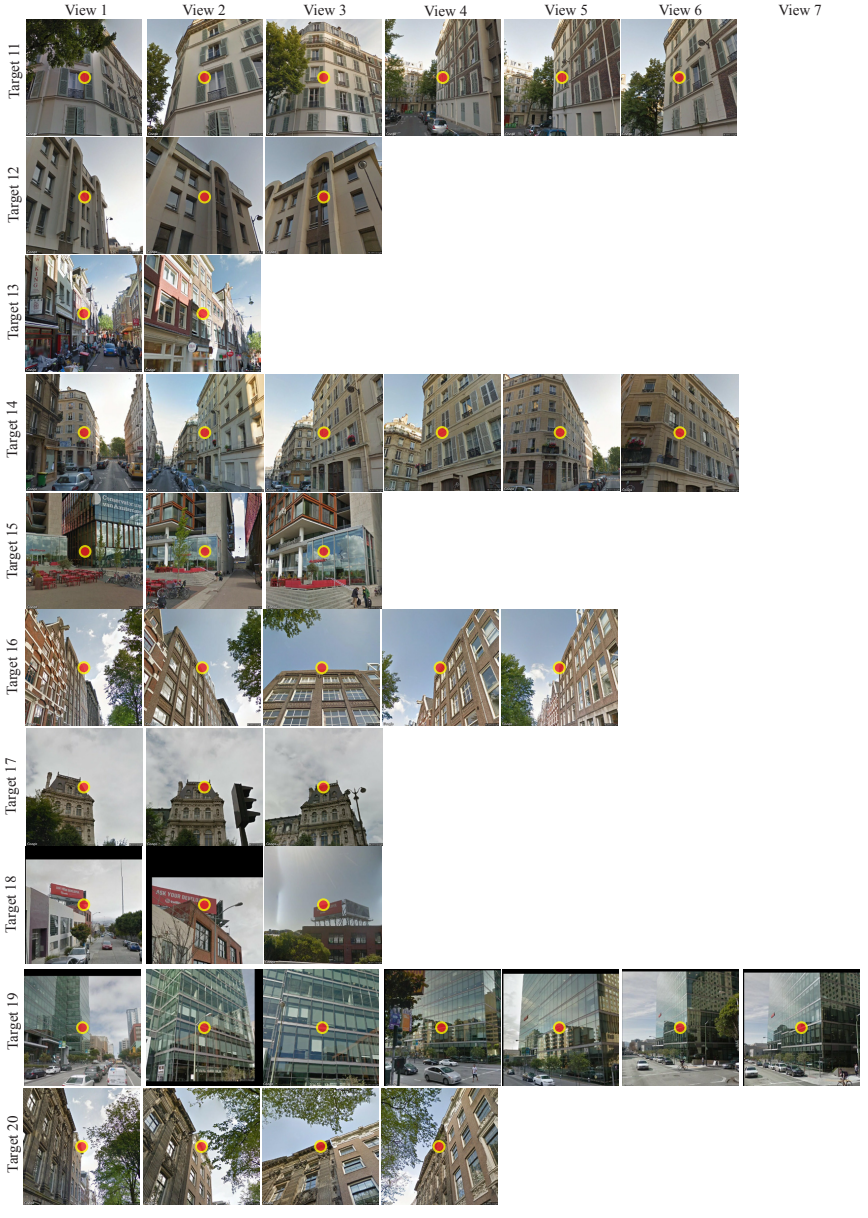

Fig. 2: Sample images from our dataset. Each row shows one image bundle showing one target. The marker is on the center pixel and should show the target point. The columns show different views.

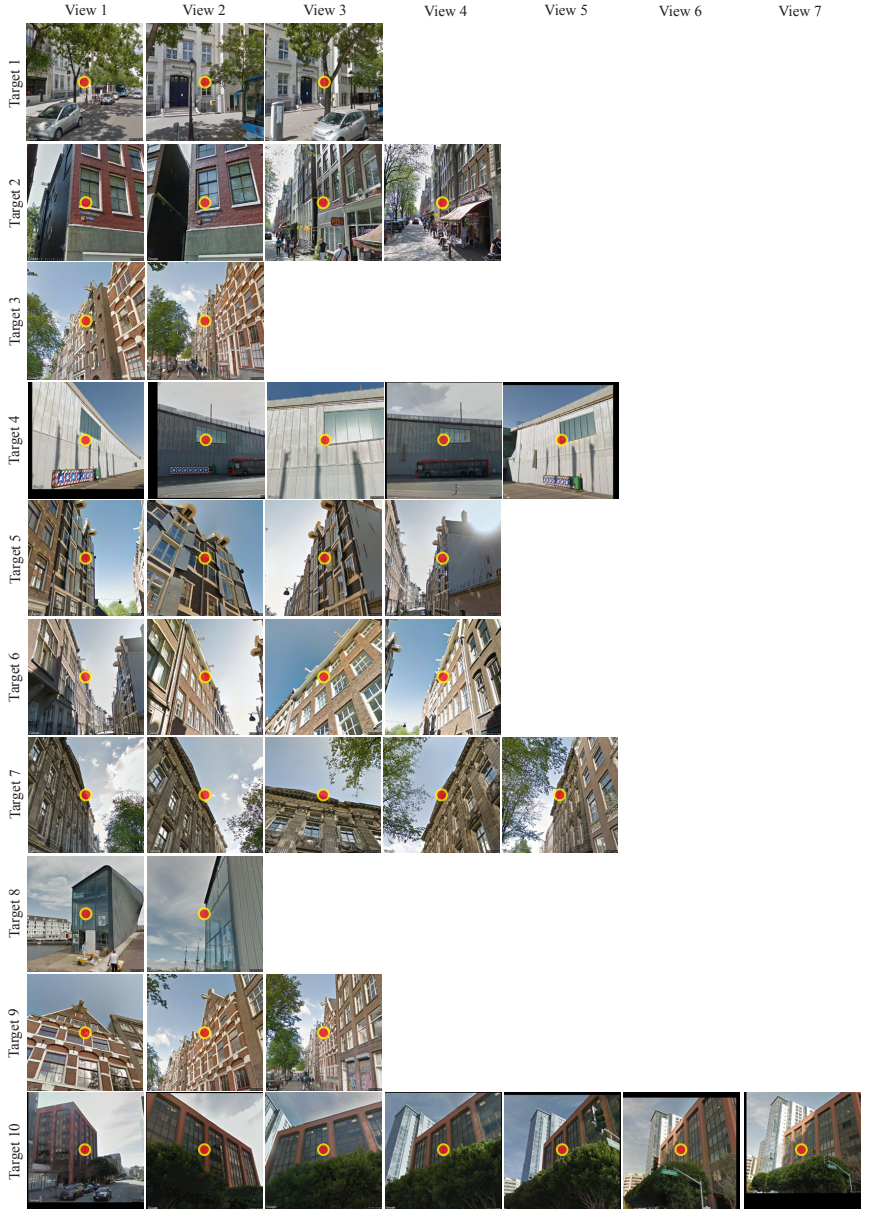

Fig. 3: Sample images from our dataset. Each row shows one image bundle showing one target. The marker is on the center pixel and should show the target point. The columns show different views.

baseline change (see 'Warped' in Figure 4). Thus, the registration transformation between the two rectified images can be effectively approximated by a similarity transformation. To avoid a quadratic complexity with respect to the number of images, we select the most frontal view for each target point and align the rest of the images of the target with respect to it.

Since the patches often show non-planarity, we found employing a nonrigid transformation (we used SIFT flow [7]) for registration of the rectified patches to be more robust. We use RANSAC to fit a similarity transformation to the resulting flow field and apply the inverse of the rectification transformation to find the translation vector which should be applied on the original image to perform the alignment (see 'Aligned' in Figure 4). We use only the translation component for performing the alignment since we want to preserve the original camera rotation information to use it in training. Finally, to remove the remaining unreliable and occluded patches, we extract two registration metrics (Structural Similarity Index which measures the pixel similarity among registered images, and magnitude of the estimated transformation) and threshold the dataset based on them (see 'Occluded' in Figure 4-b).

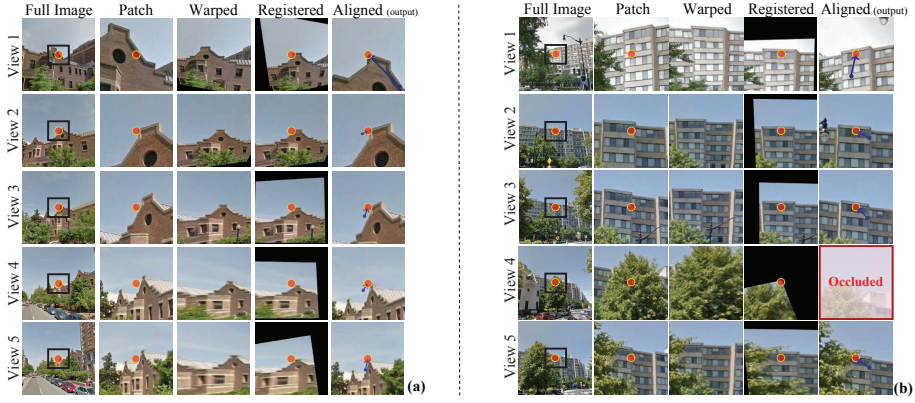

Fig. 4: **Pixel alignment process** for two sample targets. Blue arrow shows the adjustment vector between the initial and updated position of the center. (b) shows a case with occlusion (view 4).

3.2 Noise Statistics

A user study through Amazon Mechanical Turk was performed to quantify the amount of noise in the final dataset. 15124 random pairs of matching patches from the dataset were annotated by three Turkers to identify if they have at least 25% overlap in their content, and if not, what the reason was. Also, to quantify the magnitude of pixel misalignment, they were asked to click on the pixel in the 2^{nd} patch which corresponds to the center pixel of the 1^{st} patch.

About 68% of the pairs were found to have at least 25% overlap (by majority voting). For the subset with less than 25% overlap, the originating issues were

identified as: 49.5% due to inaccuracies in meta-data and 3D models, 27.7% due to tree occlusion, 16.5% due to other occlusions, and 6.1% other reasons. In heavily tree occluded areas (e.g., Washington D.C. suburb), the percentage of matching pairs reduced to 58%. Such areas can be easily avoided by geo-fencing (e.g., by collecting data from downtowns since they show more buildings than trees compared to suburbs).

The mean and standard deviation of the magnitude of misalignment (defined as the magnitude of the vector connecting the center of the 2^{nd} patch to the pixel location where the Turkers clicked) was 16.12 ($\approx 11\%$ of patch width) and 11.55 pixels, respectively.

In our training dataset, we did not perform any manual filtering or geo-fencing on top of the automatically collected data since the amount of noise appeared to be within the robustness tolerance of ConvNets as their training converged and well performed on the carefully filtered test set.

4 Surface Normal Estimation

Estimating surface normals is a fundamental problem in 3D computer vision. We evaluated the usefulness of the representation learned by our method for surface normal estimation on the standard NYU2 benchmark [10,4,13]. The dataset contains 795 training images and 654 testing images.

In this work, our goal is not to develop the best method for a specific task, but rather to investigate how useful are features learned by training for pose-estimation for various 3D tasks without the requirement for fine-tuning. For this task, we employed two methods to read the surface normal value out of the representation of an image/patch: a linear classifier and nearest neighbors. While running our experiments, we noticed that the surface normal estimation in the NYU dataset is heavily biased as most pixels belong to one of the three categories of ceiling, floor, and walls. This inevitably means that the prediction results would be dominated by performance on such categories. In order to account for this, we propose the use of reporting binned errors. The binned errors are calculated by first binning pixel wise surface normals into 20 bins (in a manner similar to [13]) and calculating the error within each bin. Then we calculate the average statistics such as the mean or the median across the bins. The idea of using binned error is similar to the idea of reporting mean of the class accuracies in classification tests. In addition to this setup, we also report numbers according to the standard benchmark evaluation of pixelwise median error and the percentage of pixels that were within $11.5^o, 22.5^o$ and 30^o respectively.

4.1 Prediction via Linear Classification

In previous works, such as [13,14], coarse surface normals were estimated by predicting normals on a 20x20 spatial grid. The normal of each grid block is predicted by solving a 20 way classification problem, where the 20 classes correspond to 20 pre-calculated surface normal clusters. The final estimate is provided by up sampling this 20x20 grid into the full image resolution.

Table 5: Evaluation of surface normal estimation on the NYU2 dataset using the 20-way classification setup. We trained a linear classifier on top of the representations to perform the estimation. Classification accuracy and angular error is reported on individual (20x20) grid locations. The angular error is calculated as the angular distance between the predicted and ground truth surface normal clusters. We report accuracies with and without binning.

	No Binning			Binned		
	(Higher Better)	(Error: Lower Better)		(Higher Better)	(Error: Lower Better)	
Net	Accuracy	Mean	Median	Accuracy	Mean	Median
Agrawal et al. [1]	23.3	18.9	15.9	14.0	22.7	19.1
Wang et al. [14]	18.0	21.3	18.9	10.5	23.6	24.7
AlexNet	25.4	18.8	14.2	**17.6**	21.6	**15.3**
Random (statistically informed)	9.2	25.0	24.9	5.0	27.2	25.2
Ours	**27.3**	**17.7**	**13.5**	17.4	**20.7**	**15.3**

Table 6: Evaluation of surface normal estimation on the NYU2 dataset using the common evaluation protocol affected by dominating wall, ceiling, and floor pixels. We used 1-NN technique on each representation to predict the normals. We report the pixelwise median error and the percentage of pixels that were within $11.5^o, 22.5^o$ and 30^o respectively. Our method outperforms the feature learning approaches of Agrawal et al [1] and of Wang et al. [14] and is comparable to layer 7 features of AlexNet trained for Imagenet classification using this evaluation setup.

	(Lower Better)	(Higher Better)		
Net	Median	11.5^o	22.5^o	30^o
Agrawal et al. [1]	20.2	29.1	54.5	67.9
Wang et al. [14]	20.4	28.9	54.0	67.7
AlexNet	19.7	30.3	55.7	69.5
Ours	19.6	30.7	55.6	68.8

Following a similar experimental setup, we first report the accuracy of predicting surface normals on 20x20 grids. We trained a linear classifier on top of our representation as well as the methods of [1,14] and AlexNet trained for ImageNet. We report the classification accuracy in individual grid locations and the mean/median angular error in the estimated surface normals. The angular error is calculated as the angular distance between the predicted and ground truth surface normal clusters. We report accuracies with and without binning in Table 5 indicating that our representation outperforms AlexNet and [1,14].

4.2 Prediction via Nearest Neighbors

In addition to results presented above, we also used 1-Nearest Neighbor to compute the pixel wise surface normal errors. For each image in the test set, we found the closest image in the training set using each representation. The surface normals of this closest image were taken as the predicted surface normals for the query image. The pixel wise error in surface normal estimation for various CNNs is reported in Table 6. The results are consistent with the experiments using a linear classifier.

5 Joint Embedding of Synthetic Cubes and Images

In order to further investigate the nature our representations, we performed a joint tSNE embedding of synthetic cubes along with a single car category from the EPFL dataset (see Figure 5). This plot is closely related to the Cube⇆Object nearest neighbor results provided in Figure 8 of the main paper (here we show all images but using a lossy 2D embedding; in Figure 8 of the main paper, the full dimensional representation was used for retrieving nearest neighbors, but only a few sample queries could be shown).

If a representation mostly encodes semantic (or low level appearance) information, then cars and cubes should cluster into two different parts of the feature space, whereas if it encodes more geometric information, the organization should be based on pose. Figure 5 shows that our method embeds cubes and cars according to their geometric pose. In addition, in our feature space, cars and cubes are close together and one is enclosed by the other, whereas the baselines linearly separate them and put them far apart. This indicates that, as compared to other representations, ours predominantly capture geometric information over semantics or low-level appearance. It should be noted that synthetic cubes were sampled uniformly in the pose space, whereas cars in the EPFL dataset are not sampled uniformly (no camera pitch). Due to this, many cubes do not have a corresponding car and the tSNE plots are slightly skewed.

tSNE on Affinity Matrix: the tSNE embeddings in Figure 5 are plotted in an affinity space. That is, given N car and M cube images, we compute the representation of each image and form a $(N + M) \times M$ affinity matrix where element (i, j) is the l_2 distance between the representations of the i^{th} image and j^{th} cube. In other words, the representation of all of the car and cube images are cast based on the cube collection as the reference. We then perform the tSNE embedding on this $(N + M) \times M$ affinity matrix with each row being the M dimensional representation of the image, rather than its original representation.

We adopted this approach in order to enable the tSNE plots to bring out the geometric encoding aspect and not be dominated by the appearance/semantic information in the representation. For performing a successful pose estimation, encoding of both geometric and appearance information is essential [1]. However, when a tSNE plot is formed, the user does not have any control over specifying whether the geometry or the appearance factor (or both) should govern the embedded space. In general, all of the manifolds existing in a representation contribute to the final 2D embedding. The workaround of finding the tSNE embedding on a unified affinity space lowers the impact of appearance as the appearance of all cubes are roughly the same while their pose covers a wide

[1] For instance, if the pose estimation is being performed based on vanishing points, then besides extraction of at least 3 vanishing points from the input images, the correspondences among vanishing points needs to be done as well. Therefore, the representation must encode some appearance information in order to enable finding the correspondences among vanishing points.

range. All of the plots in Figure 5 (including the baselines) are formed using this approach.

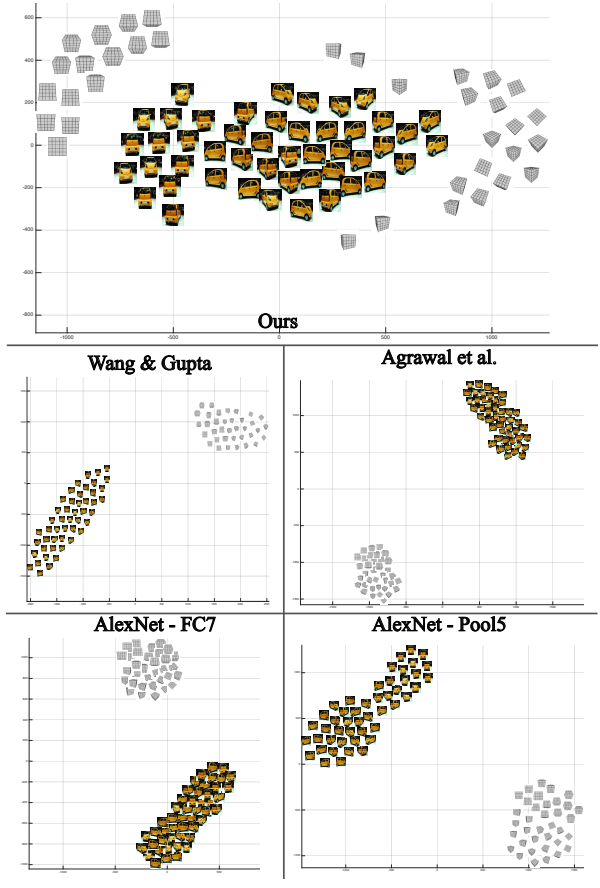

Fig. 5: tSNE embedding of the synthetic cubes and one category of EPFL dataset. Our representation shows a geometric organization while the baselines perform a clear semantic/appearance based separation.

6 Illustration of Pose Induction via tSNE Embeddings of more ImageNet Classes and MIT Places

We only trained our network for estimating 6-DOF pose on street view scenes. However, it appears that our representation learned something generic about the geometry of objects (see the discussion on unsupervised evaluations in section 4 of the main paper). This is elucidated by tSNE embeddings of objects in the Imagenet dataset (Figures 6,7). These embeddings show that the feature

Fig. 6: tSNE of more ImageNet categories of our representation vs various baselines.

space produced by our network performs pose induction for unseen object classes without any additional training.

However it should be noted that a purely geometric approach is sometimes insufficient for a full object pose estimation, such as, distinguishing between front and back the car. Such knowledge is dependent on semantics and not merely geometry. It is therefore not surprising that our method puts such poses close to each other. In addition, one more mode of confusion of our method is putting together poses that are apart by 90 degrees. If the pose is indeed estimated based on vanishing points (see section 4.2 of the main paper), it is to be expected that objects 90 degrees apart in azimuth would have the same

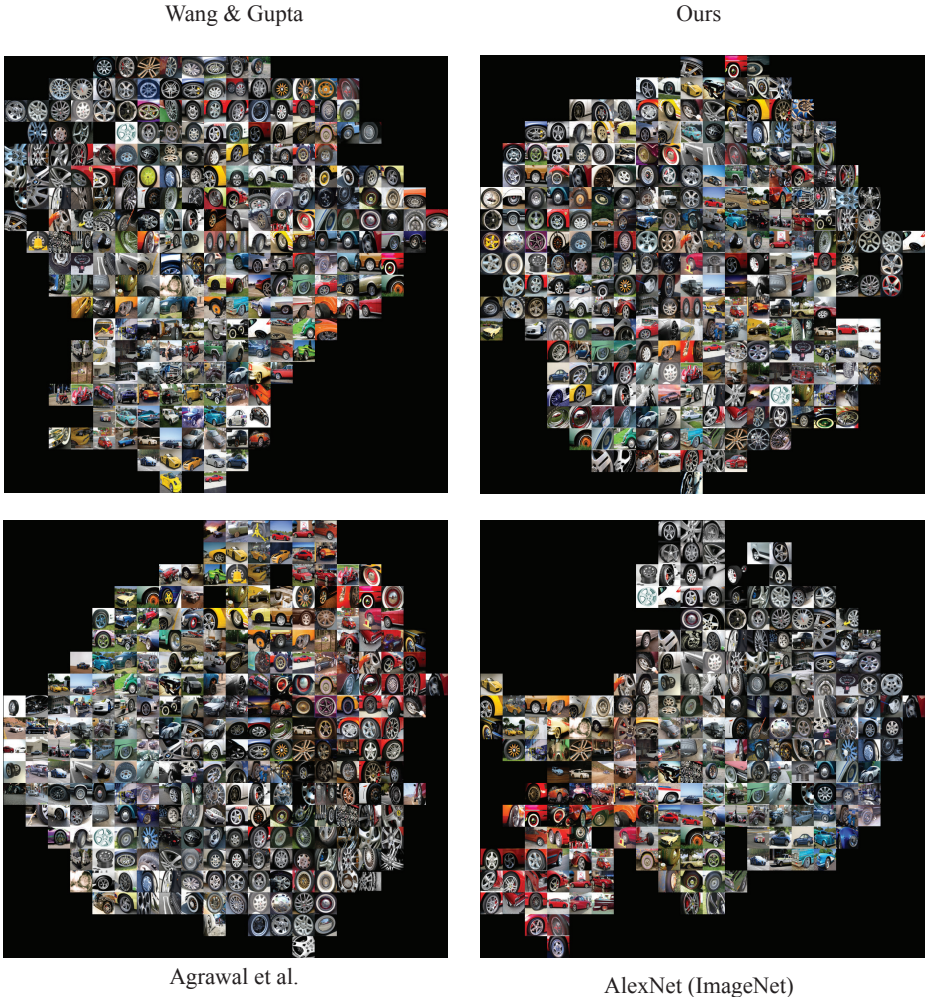

Fig. 7: tSNE of more ImageNet categories of our representation vs various baselines.

vanishing points and therefore it would require the knowledge of semantics to tease these poses apart. Since our method shows such confusion, it supplies further support that our method may performs pose estimation analogous to a method that would estimate pose based on vanishing points.

In addition to pose induction on imagenet, our method also performs pose induction on scene categories outside our street view dataset. For instance in Figure 8, the images of MIT places datasets (category library) are embedded according to pose. Finally, additional visualization of embedding of image patches from our dataset using our representation and AlexNet's (trained on Imagenet) are shown in Figure 9.

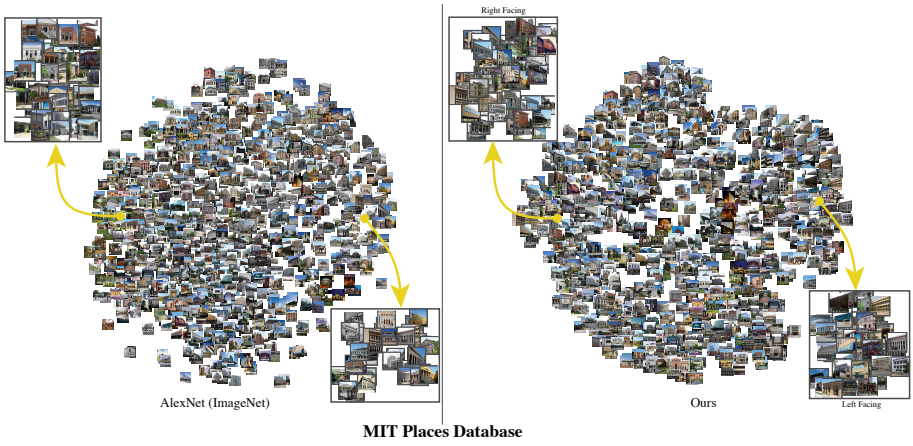

Fig. 8: tSNE of our representation vs AlexNet (ImageNet) on a non-streetview dataset: MIT Places Benchmark (class 'library' which is one of the pose rich categories). Our network shows a clear pose based embedding, unlike AlexNet trained on ImageNet. For both networks, the descriptor was computed over the entire image, and not patches.

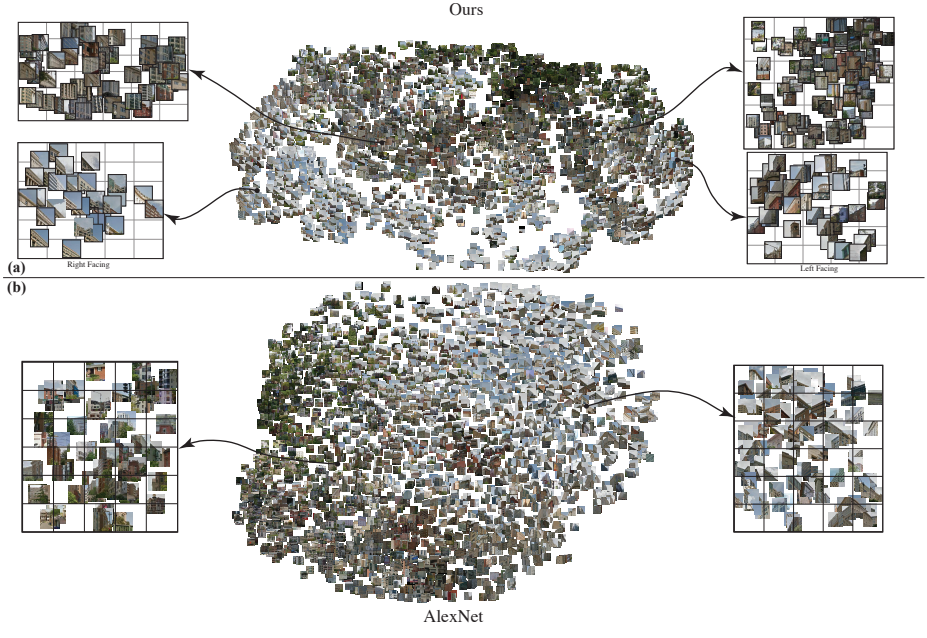

Fig. 9: This figure shows the same tSNE plot as the one in Figure 7 of the main paper, as well as AlexNet's tSNE for the sake of comparison. The AlexNet is trained on ImageNet and shows a non-geometric organization.

7 Additional Training Details

For pose training we followed a curriculum strategy where we first trained only for angles within $[-90°, 90°]$ and then extended to all the angles. We found that this performed slightly better than directly training for all angles. We also experimented with employing quaternions, but found that predicting Euler angles performed better (quantitative results in the main paper).

References

1. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving (2015)
2. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. pp. 2911–2918. IEEE (2012)
3. Brown, M., Hua, G., Winder, S.: Discriminative learning of local image descriptors. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33(1), 43–57 (2011)
4. Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2650–2658 (2015)
5. Fischer, P., Dosovitskiy, A., Brox, T.: Descriptor matching with convolutional neural networks: a comparison to sift (2014), arXiv preprint arXiv:1405.5769
6. Han, X., Leung, T., Jia, Y., Sukthankar, R., Berg, A.C.: Matchnet: Unifying feature and metric learning for patch-based matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3279–3286 (2015)
7. Liu, C., Yuen, J., Torralba, A.: Sift flow: Dense correspondence across scenes and its applications. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33(5), 978–994 (2011)
8. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision 60(2), 91–110 (2004)
9. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. Pattern Analysis and Machine Intelligence, IEEE Transactions on 27(10), 1615–1630 (2005)
10. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: European Conference on Computer Vision. pp. 746–760. Springer (2012)
11. Simonyan, K., Vedaldi, A., Zisserman, A.: Learning local feature descriptors using convex optimisation. Pattern Analysis and Machine Intelligence, IEEE Transactions on 36(8) (2014)
12. Trzcinski, T., Christoudias, M., Lepetit, V., Fua, P.: Learning image descriptors with the boosting-trick. In: Advances in neural information processing systems. pp. 269–277 (2012)
13. Wang, X., Fouhey, D.F., Gupta, A.: Designing deep networks for surface normal estimation. In: CVPR (2015)
14. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2794–2802 (2015)
15. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks (2015), arXiv preprint arXiv:1504.03641v1