3DInAction: Understanding human actions from 3D point clouds

Supplementary Material

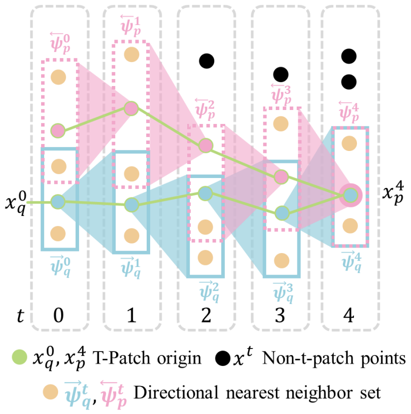
$\overleftarrow{\psi}_p^0$
$\overleftarrow{\psi}_p^1$
$\overleftarrow{\psi}_p^2$
$\overleftarrow{\psi}_p^3$
$\overleftarrow{\psi}_p^4$
x_q^0
x_p^4
$\overrightarrow{\psi}_q^0$
$\overrightarrow{\psi}_q^1$
$\overrightarrow{\psi}_q^2$
$\overrightarrow{\psi}_q^3$
$\overrightarrow{\psi}_q^4$
t 0 1 2 3 4
x_q^0, x_p^4 T-Patch origin
x^t Non-t-patch points
$\overrightarrow{\psi}_q^t, \overleftarrow{\psi}_p^t$ Directional nearest neighbor set

Figure 6. **Bidirectional t-patch illustration.** t-patches formed from start to finish are presented in light blue and the reverse-t-patches in pink. Note that the nearest neighbour in one direction is not necessarily the nearest neighbour in reverse (time step $t = 4$).

A. Bidirectional t-patches

In Section 3 we presented the t-patches, their construction and the t-patch collapse problem. To mitigate the collapse issue, we proposed a bidirectional t-patch formulation, given in Eq.(2). In Figure 6 we present an illustration of this process. The illustration depicts t-patches formed from start to finish in blue $\overrightarrow{\Psi}_q^t$ and in reverse in pink $\overleftarrow{\Psi}_p^t$. Note that the nearest neighbour in one direction is not necessarily the nearest neighbour in reverse, as can be seen in time step $t = 4$. The bidirectional t-patches are essential for long sequences in order to keep the coverage ratio and prevent t-patch collapse.

B. Pipeline architecture details

The t-Patch network computes a high-dimensional representation for each t-Patch. The architecture is composed of several MLP layers operating on the non-temporal dimensions (sharing weights across points) followed by a convolutional layer operating on both the temporal and feature dimensions. The network weights are shared across t-patches.

For the first t-Patch module, 512 t-Patches are extracted and fed into 3 MLPs with dimensions of $(64, 64, 128)$ followed by a 2D temporal convolution with a kernel of $(8, 128)$, *i.e.* operating over all feature channels and weighted averaging 8 consecutive frames. For the second module, 128 points (t-patch centers from the previous extractor) are sampled, the MLP is size is $(128, 128, 256)$ and the temporal kernel is 4. For the third module, no downsampling is performed, MLPs sizes are $(256, 512, 1024)$. All layers use ReLU activation and batch norm. The final classifier uses 3 fully connected layers of sizes $(512, 256, \#classes)$ with a drop out after the first and second layers with a drop probability of 0.4. We apply temporal smoothing as a convolutional kernel over the temporal domain before the last classifier layer with a kernel size of T (all frames).

C. DFAUST Dataset extension

We extend the DFAUST dataset for the task of action recognition. The DFAUST dataset [3] provides high-resolution 4D scans of human subjects in motion. It includes over 100 dynamic actions of 10 subjects (6:4 male-to-female ratio) with varying body shapes represented as registrations of aligned meshes. This dataset was not specifically designed for action understanding, however, it provides point cloud sequences with action labels per sequence. We extended it to our task by subdividing the dataset into clips of 64 frames of train and test human subjects. The trainingset is composed of three male and three female subjects and includes a total of 76 sequences, 395 clips, and ∼25K point cloud frames. The testset is composed of two male and two female subjects and includes a total of 53 sequences, 313 clips, and ∼20K frames. This split was chosen in order to guarantee no subject will appear in both training and test splits as well as to make sure that all actions appear both in the train and test splits. Note that not all actions are performed by all subjects.

The action instance occurrences and full action list are depicted in Figure 7. It shows that there are two dominant classes (*hips* and *knees*). Therefore, to mitigate this imbalance's effect on training we use a weighted sampler that uses a sampling probability that is inversely proportional to a class's occurrence in the training set.

D. IKEA ASM preprocessing

The IKEA ASM is a large scale dataset and each frame may contain a very large number of points (hundreds of thousands). Most of the points, however, lie on static background regions. In order to keep the training time reasonable, we downsampled each frame using farthest point sampling (FPS) to have a fixed number of 4096 points. We then saved the data into clips for training. This reduced the training time for all methods significantly since the data loading

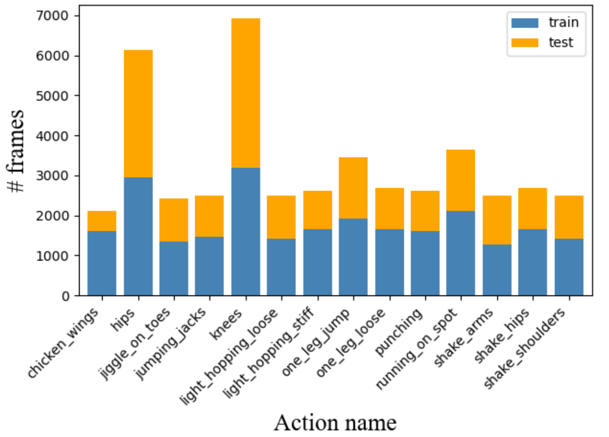

Figure 7. Our extension of the DFAUST dataset for action recognition. Number of frames per action in the training (blue) and test (orange) sets.

was a bottleneck.

E. Additional Experiments

Results on NTU RGB+D dataset. We conduct thorough experiments to evaluate the performance of our proposed approach compared to existing state-of-the-art methods on the NTU RGB+D 60 dataset [28]. Since the dataset is essentially saturated and previous methods show correlation between the different training splits (subject, view, and setup) we explore the performance as a function of the size of available data. Since the full NTU60 includes $57.6K$ videos and $\sim 1.1M$ clips we evaluate the performance in terms of small fractions of the original data, specifically $(2.5\%, 5\%, 7.5\%, 10\%)$ that amount to $(\sim 25.5K, \sim 54.3K, \sim 81.4, \sim 110.5K)$ clips respectively. The results are reported in Figure 8. The results show that while some methods thrive when a lot of data is available, the proposed method demonstrates superiority even when the dataset scale is limited. For completeness, the performance of all methods when all data is available are comparable and given here: 90.2, 91.0, 89.3 for P4Transformer [8], PSTTransformer [9] and the proposed approach respectively.

In this experiment we use the same training and test protocol as specified by P4Transformer [8] and retrain the models using the parameters reported in the papers. Since the proposed approach and architecture were designed for per-frame prediction we add a 3 layer GRU at the end, to aggregate the temporal domain into a single vector representation per clip. To fairly compare the quality of the representation we also replace our classifier with the same classifier architecture as in P4Transformer and PSTTransformer. This helps avoid changes in performance that are related to the classifier and focus the evaluation on the core representation ability. To generate the data fractions, we uniformly sample a fraction of the videos in both train and test sets. This way, the distribution of class occurrences is preserved. Note that while the number of videos in each class is equal in both trainingset and testset, the lengths are different and therefore the number of generated clips is different. This causes a significant class imbalance, as shown in Figure 9.

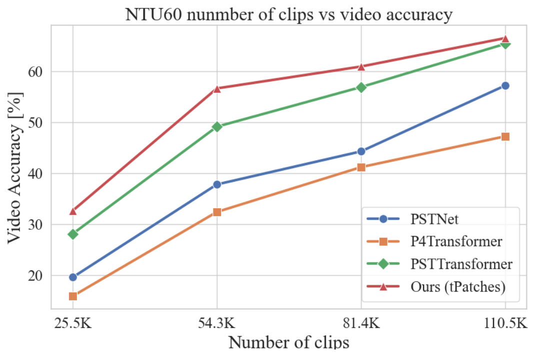

Figure 8. Video Accuracy as a function of the data fraction on the NTU 60 dataset. The results show that the proposed approach achieves better performance than PSTTransformer [8], P4Transformer [9] when the dataset scale is limited.

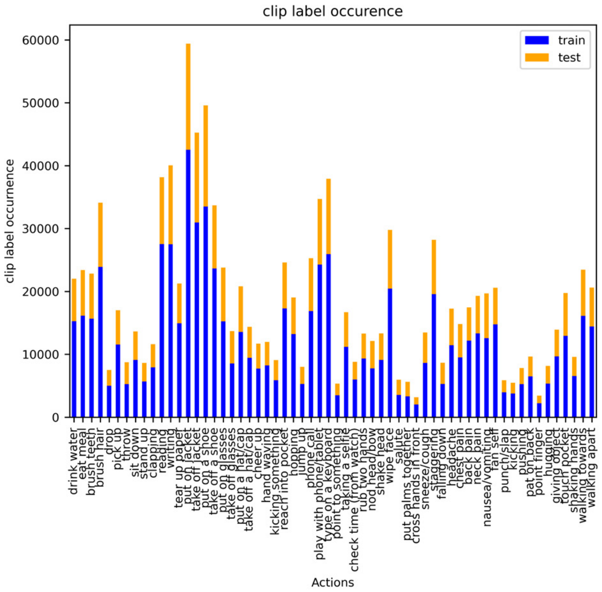

Figure 9. NTU60 clip class distribution. It shows an imbalance between different classes.

Absence of dynamic method's baselines on DFAUST and IKEA ASM. In Section 4 we report the performance of various baseline methods on existing datasets for 3D action recognition. For DFAUST and IKEA ASM we report static methods PointNet [23], PointNet^{++} [24], and

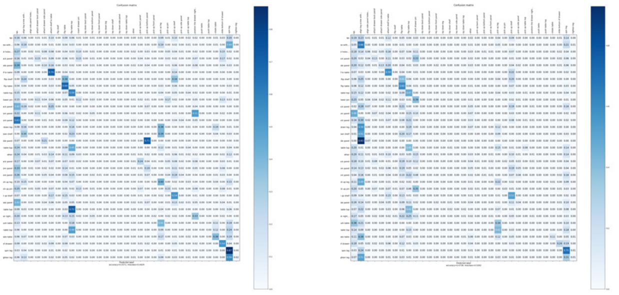

Figure 10. **Confusion matrices** for the proposed approach with (left) and without (right) bidirectional t-patches, evaluated on the IKEA ASM dataset.

Set Transformer [18] by applying them on each point cloud frame individually. Additionally, we report temporal methods like PSTNet [10] and also implemented a temporal smoothing version of each static method (PoinNet+TS, Pointnet^{++}+TS, and Set Transformer+TS respectively) by learning the weights of a convolutional layer over the temporal dimension. Some dynamic methods performance do not appear in the tables as those were not reported in the original papers. Since a code is publicly available for some, we corresponded with the authors of PSTNet [10], P4Transformer [8] and PST-Transformer [9] in order to get recommendations for testing these methods on our datasets, however, results were very poor for [8, 9] and therefore not reported to avoid unfair judgment on these methods.

IKEA ASM confusion matrices. In Table 2 of the main paper we presented quantitative results for action recognition on the IKEA ASM dataset [2].Here, in Figure 10, we present the confusion matrices for the proposed approach with and without bidirectional t-patches. Our analysis reveals varied impact on different action classes when Bidirectional t-patches (BD) are applied. Notably, the performance drops for classes like *align leg screw* and *align side panel*, while increasing for *spin leg* and *slide bottom drawer*. This variability stems from the data-dependence of BD, addressing temporal collapse, which can have a positive and negative effects by reintroducing informative and uninformative patches.

Occlusion ablation study. We conducted additional experiments to evaluate the effectiveness of the bidirectional t-patches in the context of occlusions. The results, reported here in Table 11 and visualized in Figure 12, show that the proposed approach provides a boost in performance which significantly benefits from the bidirectional sampling strategy. In this experiment, we simulated occlusions on the DFAUST dataset by randomly selecting a point from a random frame and removing all points within a 20% radius for 16 consecutive frames (64 frames per clip).

	Frame acc.		
Method	**top 1**	**top 3**	**mAP**
PointNet^{++} + TS	66.71	86.80	0.7985
Ours	76.75	95.54	**0.8770**
Ours + BD	**84.81**	**96.66**	0.8683
Ours+BD no occlusion	87.26	99.26	0.8616

Figure 11. **Occlusion ablation** on DFAUST. Exploring effects of occlusions with and without bidirectional t-patches. For reference we include our "no occlusion" result from the main paper.

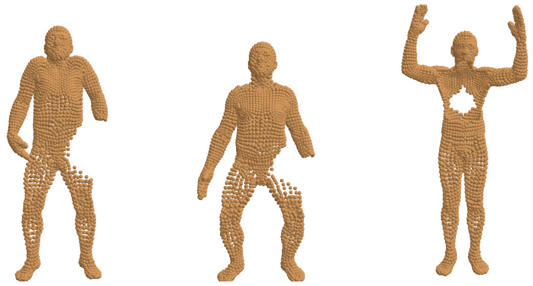

Figure 12. **Occlusion ablation**. Visualizing the point clouds used in the occlusion ablation experiment on the DFAUST dataset.

Additional details on accuracy and mAP inconsistency. Our models are trained using the cross entropy loss, which encourages high recall (i.e., Top 1 accuracy). The mAP metric roughly summarizes the trade-off between precision and recall and is important for cases where a clip contains multiple action labels. The reason for the small inconsistency between Top 1 and mAP is class imbalance. In the IKEA ASM dataset, there is a significant imbalance with many occurrences for classes such as "spin leg" (which appears four times in each of the three furniture types). Methods that perform well on dominant classes will be favored by the Top 1 metric. The DFAUST dataset is also affected by class imbalance, albeit less so than IKEA ASM. Importantly, when ranking by either Top 1 or mAP, our t-patch based method outperforms previous approaches.

Additional IKEA ASM Visualization. In Figure 5 we provided a visualization of the t-patches with grayscale point clouds. Here, in Figure 13 we provide an extended version of that figure that also includes colored point clouds.

Parameter ablation for t-patch extraction. In Section 4.4 we performed an ablation study and analyzed the effect of tuning the number of neighbors to extract (k) and the number of points to sub sample (n) on accuracy performance. The results were reported in Table 5. Here we complement this ablation by also reporting the time and number of network parameters for each parameter selection in Table 7. Timing experiment was done on an NVIDIA A5000 GPU (timing experiment for Figure 6 was done on an A100, hence the reported time differences).

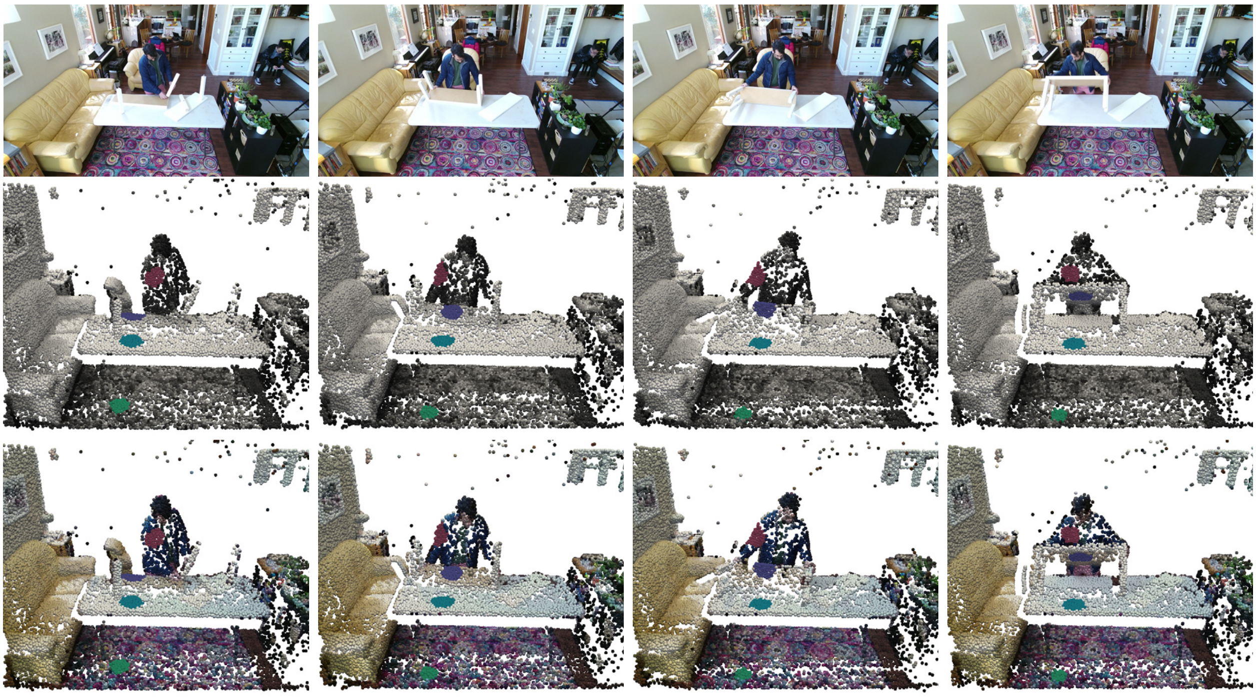

Figure 13. **IKEA ASM example with t-patches.** The flip table action for the TV Bench assembly is visualization including the RGB image (top), a grayscale 3D point cloud with colored t-patches (middle), and a RGB colored 3D point cloud with colored t-patches (bottom). t-patches are highlighted in color. The **blue** is on the moving TV Bench assembly, **maroon** is on the moving persons arm, **teal** is on the static table surface, and **green** is on the colorful static carpet.

		Frame acc.			
n	**k**	**top 1**	**top 3**	**mAP**	**Time [ms]**
256	16	76.96	97.54	0.8430	385
512	16	**80.03**	<u>97.57</u>	**0.8975**	415
1024	16	77.30	**97.88**	<u>0.8507</u>	448
512	8	76.87	96.21	0.7557	324
512	32	<u>77.91</u>	96.60	0.7453	608

Table 7. **t-patch parameters ablation**. Results for the number of neighboring points in a patch k and number of downsampled points n show that the method is robust.