Back to the Feature: Classical 3D Features are (Almost) All You Need for 3D Anomaly Detection - Supplementary Material

Eliahu Horwitz, Yedid Hoshen
School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel
http://www.vision.huji.ac.il/3d_ads/
{eliahu.horwitz, yedid.hoshen}@mail.huji.ac.il

Contents

A. Detailed *I-ROC* Results

Table 1. ***Detailed I-ROCAUC Results:*** Top half are current state-of-the-art, bottom half are methods investigated by us. A large number of our methods outperform all current methods by a wide margin. "iNet" indicates ImageNet pre-trained

	Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Previous Methods	Voxel GAN	0.383	0.623	0.474	0.639	0.564	0.409	0.617	0.427	0.663	0.577	0.537
	+ RGB	0.680	0.324	0.565	0.399	0.497	0.482	0.566	0.579	0.601	0.482	0.517
	Voxel AE	0.693	0.425	0.515	0.79	0.494	0.558	0.537	0.484	0.639	0.583	0.571
	+ RGB	0.510	0.540	0.384	0.693	0.446	0.632	0.550	0.494	0.721	0.413	0.538
	Voxel VM	0.75	0.747	0.613	0.738	0.823	0.693	0.679	0.652	0.609	0.69	0.699
	+ RGB	0.553	0.772	0.484	0.701	0.751	0.578	0.480	0.466	0.689	0.611	0.609
	Depth GAN	0.53	0.376	0.607	0.603	0.497	0.484	0.595	0.489	0.536	0.521	0.523
	+ RGB	0.538	0.372	0.580	0.603	0.430	0.534	0.642	0.601	0.443	0.577	0.532
	Depth AE	0.468	0.731	0.497	0.673	0.534	0.417	0.485	0.549	0.564	0.546	0.546
	+ RGB	0.648	0.502	0.650	0.488	0.805	0.522	0.712	0.529	0.540	0.552	0.595
	Depth VM	0.51	0.542	0.469	0.576	0.609	0.699	0.45	0.419	0.668	0.52	0.546
	+ RGB	0.513	0.551	0.477	0.581	0.617	0.716	0.450	0.421	0.598	0.623	0.555
Our Findings	RGB iNet	0.854	**0.840**	0.824	0.687	**0.974**	0.716	0.713	0.593	0.920	0.724	0.785
	Depth iNet	0.624	0.683	0.676	0.838	0.608	0.558	0.567	0.496	0.699	0.619	0.637
	NSA	0.841	0.494	0.776	0.913	0.636	0.616	0.795	0.597	0.856	0.438	0.696
	Raw	0.578	0.732	0.444	0.798	0.579	0.537	0.347	0.306	0.439	0.517	0.528
	HoG	0.560	0.615	0.676	0.491	0.598	0.489	0.542	0.553	0.655	0.423	0.560
	SIFT	0.696	0.553	0.824	0.696	0.795	**0.773**	0.573	0.746	0.936	0.553	0.714
	FPFH	0.820	0.533	0.877	0.769	0.718	0.574	0.774	0.895	**0.990**	0.582	0.753
	PointNext (area 1)	0.499	0.772	0.498	0.750	0.589	0.525	0.545	0.431	0.805	0.484	0.587
	SpinNet	0.535	0.413	0.568	0.662	0.472	0.480	0.367	0.494	0.722	0.527	0.524
	BTF	**0.938**	0.765	**0.972**	**0.888**	0.960	0.664	**0.904**	**0.929**	0.982	**0.726**	**0.873**

B. Detailed *P-ROC* Results

Table 2. ***Detailed P-ROC Results:*** Results not reported for previous methods, "iNet" indicates ImageNet pre-trained

Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
RGB iNet	0.983	0.984	0.980	0.974	0.985	0.836	0.976	0.982	0.989	0.975	0.966
Depth iNet	0.941	0.759	0.933	0.946	0.829	0.518	0.939	0.743	0.974	0.632	0.821
NSA	0.925	0.638	0.872	0.908	0.674	0.777	0.902	0.825	0.972	0.676	0.817
Raw Depth	0.404	0.306	0.772	0.457	0.641	0.478	0.354	0.602	0.905	0.558	0.548
HoG	0.782	0.846	0.965	0.684	0.848	0.741	0.779	0.973	0.926	0.903	0.845
SIFT	0.974	0.862	0.993	0.952	0.980	0.862	0.955	0.996	0.993	0.971	0.954
FPFH	0.995	0.955	**0.998**	0.971	0.993	0.911	0.995	**0.999**	**0.998**	0.988	0.980
Omnivore	0.936	0.840	0.776	0.901	0.919	0.850	0.894	0.911	0.981	0.958	0.896
PointNext	0.735	0.652	0.708	0.899	0.640	0.481	0.769	0.384	0.959	0.651	0.687
SpinNet	0.882	0.684	0.978	0.902	0.963	0.771	0.833	0.911	0.994	0.817	0.873
BTF	**0.996**	**0.991**	0.997	**0.995**	**0.995**	**0.972**	**0.996**	0.998	0.995	**0.994**	**0.993**

C. Detailed PointNext Results

Tab. 3, 4, and 5 contain the full breakdown of the PointNext [7] results on the S3DIS [1] dataset. All the results are based on the PointNext-XL model. PointNext trained a different model for each of the six areas in S3DIS.

Table 3. ***Detailed PointNext PRO Results***

Pretrained Area	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Area1	0.425	0.294	0.365	**0.772**	**0.227**	0.151	**0.408**	0.101	0.771	0.295	**0.380**
Area2	0.392	0.261	0.282	0.549	0.183	0.166	0.286	0.168	0.788	0.293	0.336
Area3	**0.484**	0.299	0.295	0.671	0.205	0.161	0.387	**0.173**	0.690	0.313	0.367
Area4	0.302	0.309	0.314	0.554	0.185	0.158	0.253	0.164	0.726	**0.343**	0.330
Area5	0.282	**0.315**	0.268	0.649	0.202	**0.173**	0.263	0.116	0.522	0.312	0.310
Area6	0.355	0.305	**0.410**	0.633	0.189	0.147	0.305	0.157	**0.799**	0.262	0.356

Table 4. ***Detailed PointNext I-ROC Results***

Pretrained Area	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Area1	0.499	**0.772**	0.498	0.750	**0.589**	**0.525**	**0.545**	**0.431**	0.805	0.484	**0.587**
Area2	0.565	0.600	0.516	0.486	0.541	0.379	0.476	0.308	0.760	0.432	0.506
Area3	**0.614**	0.697	0.489	0.588	0.571	0.446	0.476	0.318	0.763	**0.516**	0.547
Area4	0.536	0.698	0.496	0.491	0.538	0.459	0.487	0.275	0.776	0.501	0.525
Area5	0.585	0.641	**0.563**	**0.831**	0.555	0.412	0.413	0.342	**0.816**	0.459	0.561
Area6	0.536	0.674	0.547	0.620	0.517	0.406	0.497	0.319	0.793	0.493	0.540

Table 5. ***Detailed PointNext P-ROC Results***

Pretrained Area	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Area1	0.735	0.652	0.708	**0.899**	**0.640**	0.481	**0.769**	0.384	0.959	0.651	**0.687**
Area2	0.689	0.650	0.644	0.780	0.515	0.425	0.656	**0.478**	**0.964**	0.643	0.644
Area3	**0.762**	0.660	0.688	0.845	0.475	0.505	0.755	0.469	0.945	0.654	0.675
Area4	0.648	0.643	0.678	0.807	0.545	**0.512**	0.628	0.437	0.951	**0.715**	0.656
Area5	0.666	**0.681**	0.645	0.837	0.574	0.507	0.668	0.385	0.898	0.661	0.652
Area6	0.716	0.577	**0.726**	0.837	0.589	**0.512**	0.672	0.425	0.962	0.632	0.664

D. Additional Method Combinations

Tab. 6, 7, and 8 contain the results when combining depth and RGB from other methods, not shown in the main paper.

Table 6. ***Additional Method Combinations PRO Results***

Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Depth iNet+RGB	0.877	0.893	0.908	0.924	0.877	0.464	0.927	0.929	0.911	0.829	0.853
Raw+RGB	0.896	0.948	0.927	0.874	0.925	0.549	0.903	0.932	0.910	0.887	0.875
HoG+RGB	0.898	0.948	0.927	0.873	0.927	0.555	0.902	0.932	0.911	0.901	0.877
SIFT+RGB	0.895	0.947	0.927	0.875	0.929	0.555	0.904	0.932	0.910	0.895	0.876
SpinNet+RGB	0.897	0.948	0.929	0.878	0.928	0.550	0.904	0.931	0.911	0.899	0.877

Table 7. ***Additional Method Combinations I-ROC Results***

Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Depth iNet+RGB	0.808	0.707	0.739	0.836	0.882	0.547	0.731	0.667	0.825	0.648	0.739
Raw+RGB	0.877	0.876	0.785	0.718	0.960	0.699	0.742	0.581	0.895	0.623	0.775
HoG+RGB	0.887	0.891	0.791	0.716	0.972	0.676	0.714	0.576	0.862	0.649	0.773
SIFT+RGB	0.845	0.882	0.780	0.727	0.966	0.671	0.726	0.619	0.867	0.681	0.776
SpinNet+RGB	0.851	0.841	0.806	0.682	0.969	0.753	0.713	0.627	0.864	0.679	0.778

E. Method Specific Implementation Details

Bellow, we present additional implementation details for each of our investigated methods

E.1. RGB-only ImageNet Features

Utilizing PatchCore, we feed RGB images an ImageNet [2] pre-trained WideResNet50 [9] backbone as a feature extractor. To allow for localized segmentation, we extract patch-level features from the aggregated outputs of blocks 2 and 3, resulting in a feature dimension of 1536.

E.2. Depth-only ImageNet Features

As in the RGB-only case, we use PatchCore with a depth map normalized according to ImageNet statistics.

Table 8. ***Additional Method Combinations P-ROC Results***

Method	Bagel	Cable Gland	Carrot	Cookie	Dowel	Foam	Peach	Potato	Rope	Tire	Mean
Depth iNet+RGB	0.981	0.966	0.973	0.983	0.971	0.777	0.983	0.979	0.988	0.954	0.955
Raw+RGB	0.983	0.984	0.980	0.974	0.985	0.829	0.976	0.982	0.987	0.972	0.965
HoG+RGB	0.983	0.984	0.980	0.974	0.985	0.832	0.976	0.982	0.987	0.976	0.965
SIFT+RGB	0.983	0.984	0.980	0.974	0.985	0.829	0.976	0.982	0.987	0.975	0.965
SpinNet+RGB	0.983	0.984	0.980	0.975	0.985	0.830	0.976	0.982	0.987	0.976	0.965

E.3. Raw Depth Values

We divide the depth image into patches of 8×8 pixels, resulting in 28×28 patches. The descriptor is comprised of the 8×8 pixels of each patch, flattened to a 1D list of length 64.

E.4. NSA

We were not able to find an official implementation of CutPaste [5] and the public unofficial implementations lag behind the reported figures by up to 10%. Therefore, we compare our method to NSA [8], a follow-up to CutPaste that uses Poisson blending [6] to achieve more realistic augmentations. To test NSA on the new dataset, we modified their official implementation to handle depth images. The current implementation requires the images to be represented as integers, as such, the depth images are discretized to [0, 255]. Furthermore, NSA uses extensive, class-dependent hyperparameters. MVTec 3D-AD classes were assigned hyperparameters by visually comparing them with the original classes and assigning the values of the most similar class. We use depth images for running these experiments. It is possible that NSA performance can be improved by specific per-class augmentations, however, this requires prior knowledge of anomalies.

E.5. HoG

We use the depth image as input. To align with the feature map resolution, we use 8 pixels per cell, and 1 cell per block. We use 8 bins to arrive at a 32 dimensional representation.

E.6. D-SIFT

We use the depth image as input. We apply Dense SIFT to all the pixels, to reduce the resolution, we apply average pooling. Following standard SIFT practice, we use a 128 feature dimension.

E.7. FPFH

To speed up calculations of FPFH, we downsampled the point cloud prior to running the algorithm. The downsampling is performed on the organized point cloud (i.e. image downsampling); it is then flattened into an unorganized point cloud. Using the implementation from the open-source library Open3D [11], we extract a descriptor for each point. We then reshape these descriptors back into an organized point cloud and their resolution is lowered by average pooling them. FPFH requires normals to run, we estimate the normals using Open3D. The radius for the FPFH algorithm is 0.25 and the max_nn parameter is set to 100. The resulting feature is of dimension 33.

E.8. PointNext

We use tgethe PointNeXt-XL variant to extract features. To overcome the limitation of using a very small number of points (1024 or 2048), we use the S3DIS [1] pretrained model. Meaning, the model was pretrained for a segmentation task with RGB+XYZ data. Using these variants allows us to feed the model with many more points. Specifically, the samples are downsampled to 224×224, they are then reshaped into an unorganized point cloud (with the corresponding RGB values). For each point a 64 dimensional features is returned. It is then reshaped back into an organized feature point cloud. As with the other methods, we pooled the features into 28×28 patches of 8×8 pixels. We use the code from the official github repository.

Table 9. ***Numeric Preprocessing results:*** average metrics across all classes are reported (Fig. 7-top in the main paper)

	RGB		HoG		SIFT		FPFH	
	PRO	*I-ROC*	*PRO*	*I-ROC*	*PRO*	*I-ROC*	*PRO*	*I-ROC*
Raw	**0.876**	**0.788**	0.625	0.558	0.869	0.723	**0.930**	0.764
Pre	**0.876**	0.770	**0.771**	**0.559**	**0.910**	**0.727**	0.924	**0.782**

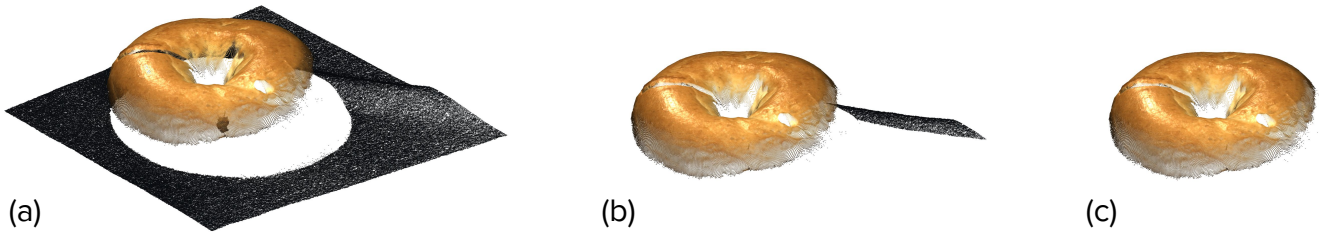

Figure 1. ***3D-Aware Preprocessing:*** In (a), a nuisance artifact in the fabric. Although a RANSAC-based plane removal step approximates the best fitting plane, in some cases, artifacts are too far away from it to be removed (b). We use a connected components algorithm to discard the remaining artifacts (c)

E.9. SpinNet

We use a model pretrained on 3DMatch [10]. Prior to feeding the points into the network, we downsampled to 224×224. We then divided each sample into 28×28 patches of 8×8 pixels. These patches are then fed into the model which outputs a 32 dimensional feature. We use the code from the official github repository.

F. Preprocessing Implementation Details

F.1. Plane Removal

By design, the objects in the dataset are centered within the image. We thus make a simplifying assumption that all the edges of the image lie on the same plane. To this end, we take a 10 pixels wide strip around the image boundary from the organized point cloud. After removing all NaNs (i.e. noise), we use RANSAC [4] to approximate the plane that best describes the boundary. The distance to this plane is calculated for each point in the point cloud, any point within 0.005 distance is removed. In practice, instead of removing the point, we zero the XYZ coordinates and RGB values for the point. This ensures the original resolution is kept. We use the Open3D [11] "Segment Plane" implementations for the RANSAC step with $ransac\_n = 50$ and $num\_iterations = 1000$, for the actual plane removal we use the returned plane equation and manually zero the values.

F.2. Clustering Based Outlier Removal

Although the plane removal step can identify and remove the majority of the planes, in some cases, the planes are not planer, see Fig 1. Points in those areas may therefore be flagged as anomalies. By running DB-Scan [3] as a connected-components approach, each cluster is treated as a connected component. We keep the largest component and remove all points from other components (as before, we zero the XYZ coordinates and RGB values of the point). We use the Open3D [11] DB-Scan implementation with $\epsilon = 0.006$ and $min\_points = 30$.

References

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3, 5

[2] 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. 4

[3] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996. 6

[4] 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. 6

[5] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9664–9674, 2021. 5

[6] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *SIGGRAPH*, 2003. 5

[7] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *arXiv:2206.04670*, 2022. 3

[8] Hannah M Schlüter, Jeremy Tan, Benjamin Hou, and Bernhard Kainz. Self-supervised out-of-distribution detection and localization with natural synthetic anomalies (nsa). *arXiv preprint arXiv:2109.15222*, 2021. 5

[9] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 4

[10] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017. 6

[11] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 5, 6