Supplementary material

A. Preservation of kernel asymmetry

Lemma 2.1 in the main paper ensures kernel asymmetry property. Let $\mathbf{x}_a$ and $\mathbf{x}_b$ be two points that exist in each others neighborhood. Consider $\mathbf{x}_a$ to be the first target point, i.e. we first perform convolution over its neighborhood with $\mathbf{x}_a$ at the kernel center. Say, this convolution results in $\mathbf{W}_a$ applied to $\mathbf{x}_b$. Now we perform a second convolution with $\mathbf{x}_b$ as the target point. This results in $\mathbf{W}_b$ applied to $\mathbf{x}_a$. If asymmetry is not preserved then $\mathbf{W}_a = \mathbf{W}_b$. This *symmetric* application of weights to point pairs leads to significant redundancy in the learned kernels. Asymmetry inhibits this problem and results in much more effective data modeling. In Fig. 7, we illustrate three simple scenarios that *do not* abide by Lemma 2.1, and hence violate the asymmetry property. We emphasize that these examples are for illustration only - justifying the rigour of mathematical analysis provided in the paper. Our method does not allow such cases following Lemma 2.1.

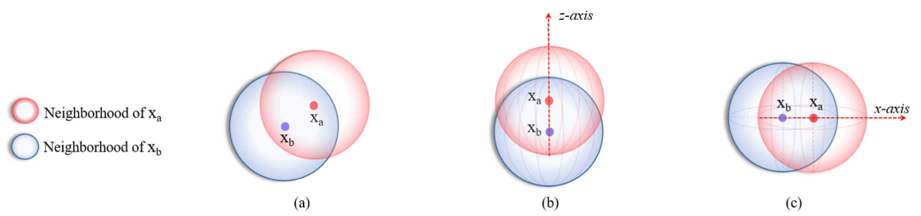


Figure 7. Examples of kernel bin divisions violating Lemma 2.1, and hence the asymmetry property. (a) The spherical space is not divided at all, which results in the same weight matrix applied to the point pair. (b) With $\Phi = [-\frac{\pi}{2}, \frac{\pi}{2}]$ and arbitrary Θ, there is no division along the ϕ dimension, resulting in symmetric application of weights to the pair $\mathbf{x}_a$-$\mathbf{x}_b$ on the $z\text{-}axis$ or lines parallel to it. (c) $\Theta = [-\pi, 0, \pi]$, i.e. $n = 2$ and arbitrary Φ. The kernel is divided into two bins along θ, which results in the point pair $\mathbf{x}_a$-$\mathbf{x}_b$ on the $x\text{-}axis$ or its parallel lines to share weights in a bin corresponding to a hemisphere.

B. Effective geometric exploration of the spherical kernel

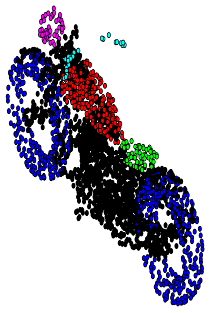
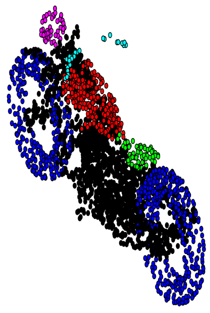
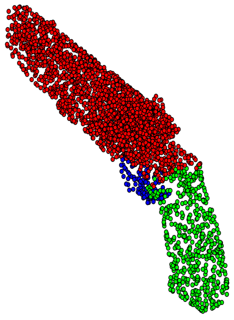
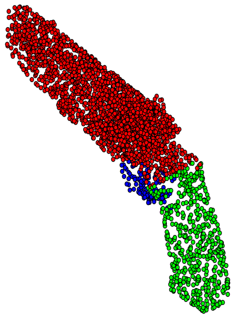
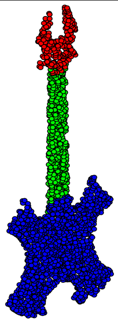
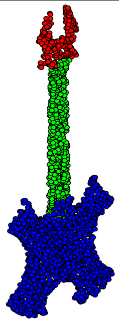
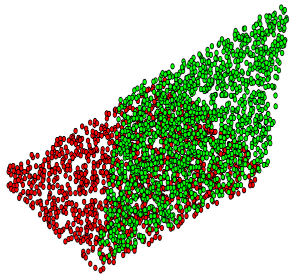
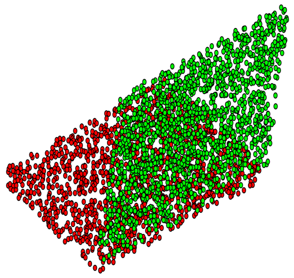
Among the compared methods, ECC [33] and OctNet [29] learn features using convolution operations. ECC uses dynamically generated filters, whereas OctNet employs 3D-CNN kernels. As compared to these methods, our spherical kernel is able to explore the geometric structure of point clouds more effectively. To demonstrate that, we remove the hierarchical concatenations of max-pooled features in our classification network, i.e. MLP(32)-Octree(64-64-64-128-128-128) and perform classification with ModelNet datasets. The resulting variant of our method matches ECC and OctNet in ignoring the intermediate layer features for the final classification. We report results of this variant in Table 5, which demonstrate a strong performance of the proposed Ψ-CNN even without the intermediate feature exploitation. This ascertains that the main strength of our method comes from the proposed spherical kernels.

Table 5. Classification results on ModelNets [39]. Ψ-CNN derives its main strenght from the spherical kernels.

Method	ModelNet10		ModelNet40	
	class	instance	class	instance
OctNet [29]	90.1	90.9	83.8	86.5
ECC [33]	90.0	90.8	83.2	87.4
Ψ-CNN (without intermediate features)	**94.0**	**94.0**	**87.8**	**91.1**
Ψ-CNN (with intermediate features)	**94.4**	**94.6**	**88.7**	**92.0**

C. Further examples of Part Segmentation

More examples of high- and low-quality segmentation resulting from our method are given below. As noted in the main paper, most of the low-quality results are normally caused by (reason-1): **confusing ground truth labelling** or (reason-2): **small parts without clear boundaries**. In Fig. 8, we can see that the high-quality segmentations are quite satisfactory. Among the low-quality segmentations in Fig. 9, the samples of *Laptop*, *Table*, *Earphone*, and *Knife* result in low mIoUs because of reason-1, whereas the samples of *Motorbike*, and *Pistol* have low mIoUs because of reason-2. Note that, although the obtained mIoUs are low in Fig. 9, the segmentations are generally visually acceptable. For instance, our method classifies the part of the cutting edge of *Knife* that is 'hidden in the handle', as its handle. Similar intuitive explanations can be found for nearly all the samples. Our 'Forward pass'/'Total' time for ShapeNet segmentation is 31.2ms/32.2ms.

High-Quality Segmentation			
GT	Ours	GT	Ours
			
Motorbike	90.3%	Pistol	94.6%
			
Guitar	90.9%	Laptop	97.2%

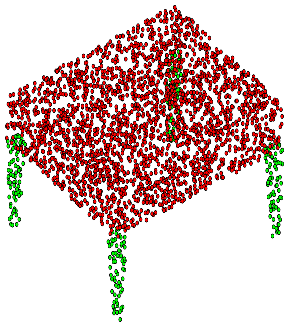
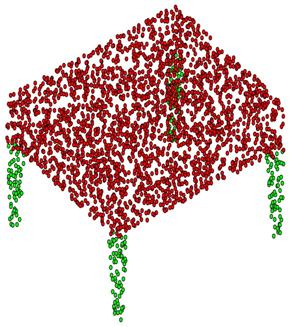
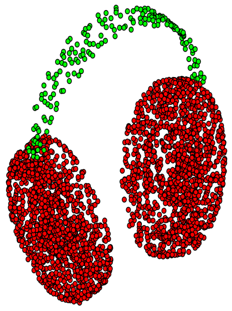
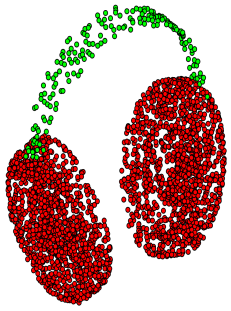

			

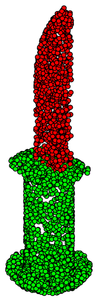
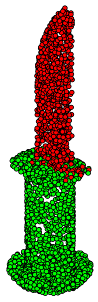
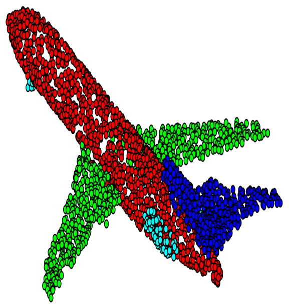
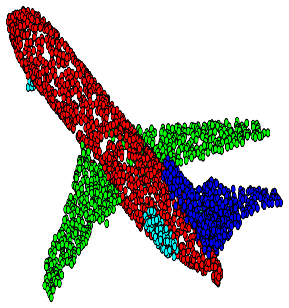
Table	92.2%	Earphone	99.4%
			
Knife	97.8%	Airplane	92.7%

Figure 8. Examples of high-quality segmentations resulting form Ψ-CNN. Computed mIoU value is also given with each sample. Color coding is within category (best viewed on screen).

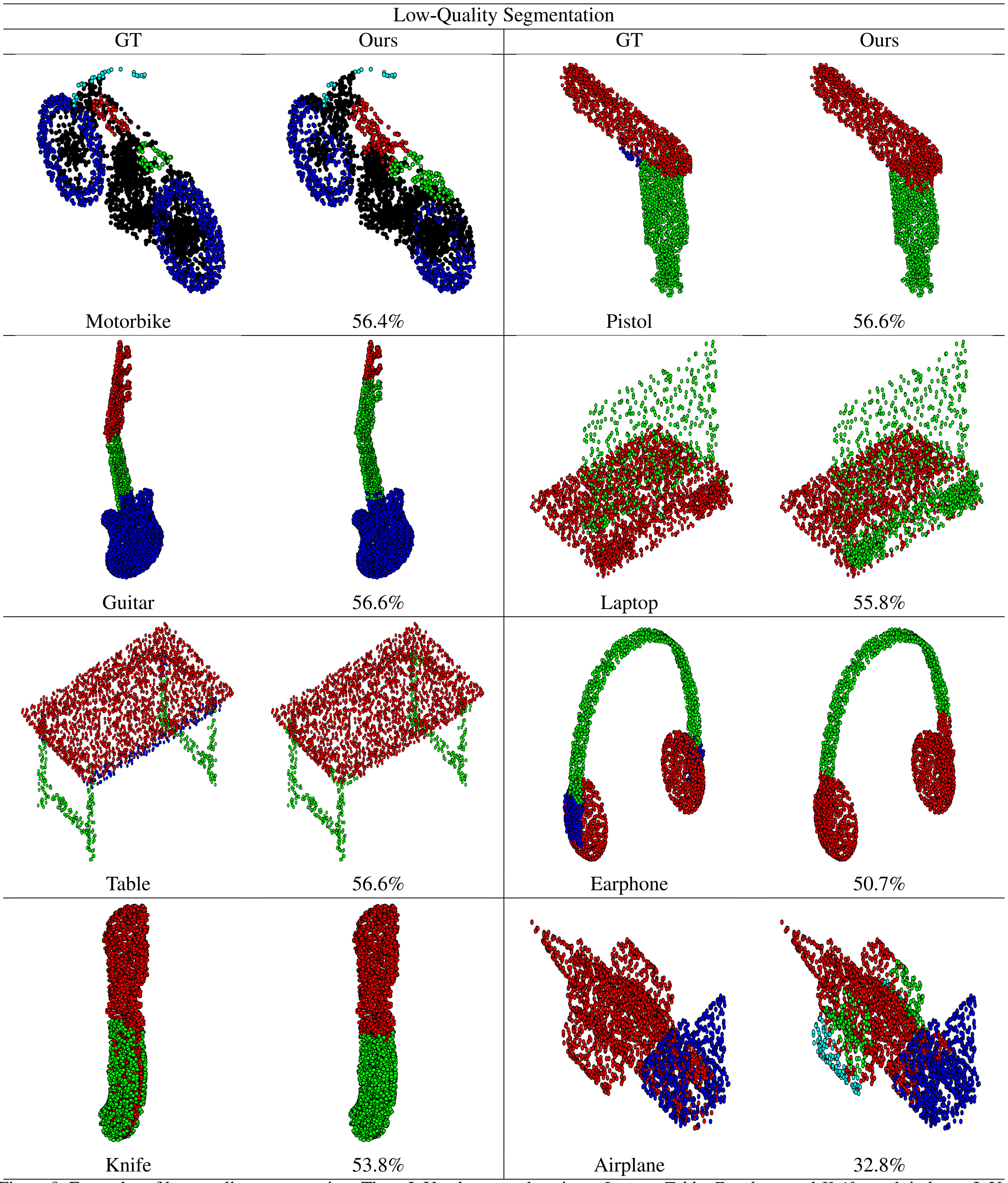


Figure 9. Examples of low-quality segmentation. The mIoU values are also given. *Laptop*, *Table*, *Earphone* and *Knife* result in low mIoUs because of confusing ground truth labeling, whereas *Motorbike* and *Pistol* have low mIoUs because of small object parts with no clear boundaries. Despite the low mIoUs, the segmentation results are generally visually acceptable.

D. Examples of Semantic Segmentation

We provide segmentation examples for RueMonge2014 dataset [30] in Fig. 10. The numbers show the Overall Accuracy (OA) of each point cloud split.

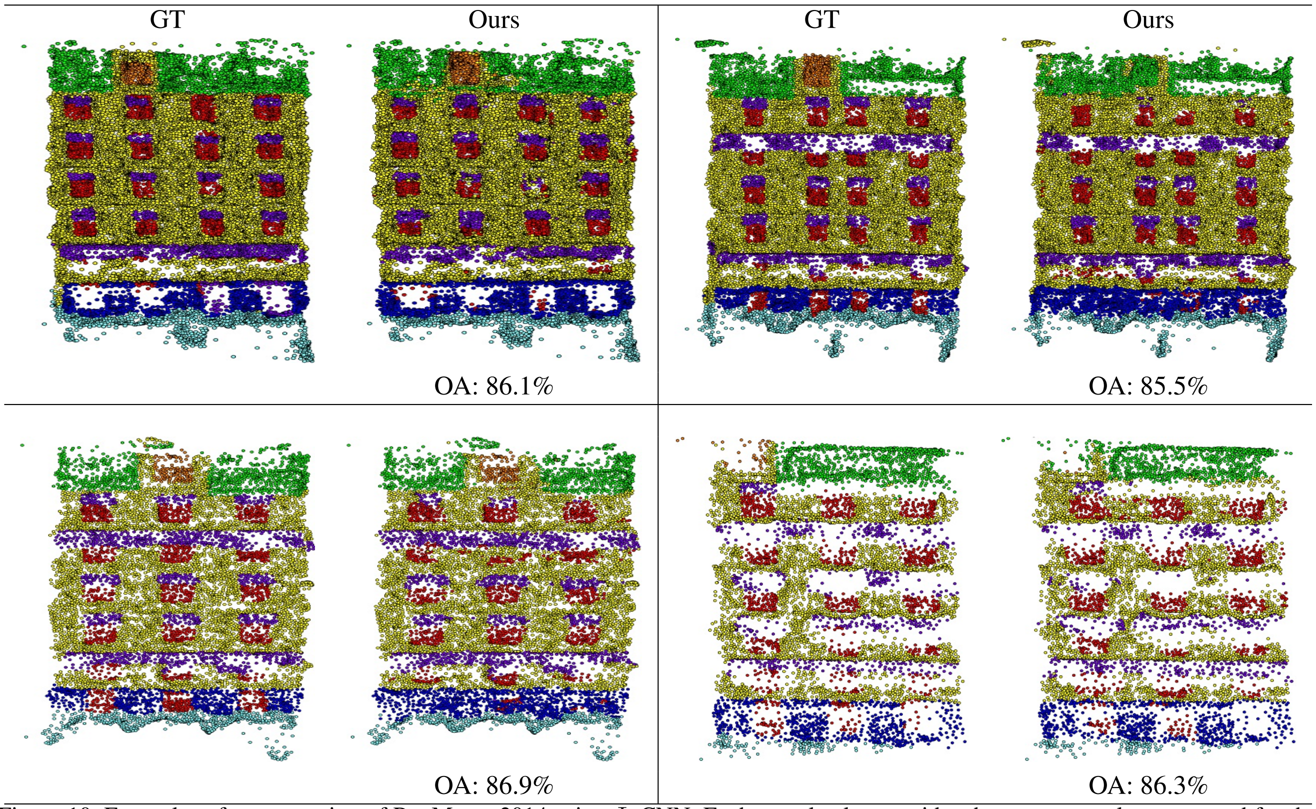


Figure 10. Examples of segmentation of RueMonge2014 using Ψ-CNN. Each sample also provides the accuracy value computed for the point cloud shown.

E. Kernel Visualization

We also provide visualization of a spherical kernel learned at the lowest octree level of our classification network in Fig. 11. A weight-matrix in each kernel bin is defined by 32 input and 64 output channels in this case. For the visualization, we reduce the input channel to a single dimension using PCA. In the shown figures, each sphere corresponds to one output channel of the $8 \times 2 \times 3$ kernel (self-convolution is ignored because it covers only the center point). Combined, the shown 64 spheres represent a single kernel with 32 input and 64 output channels. Different structures of the patterns presented by each sphere demonstrate effective learning of the weights.

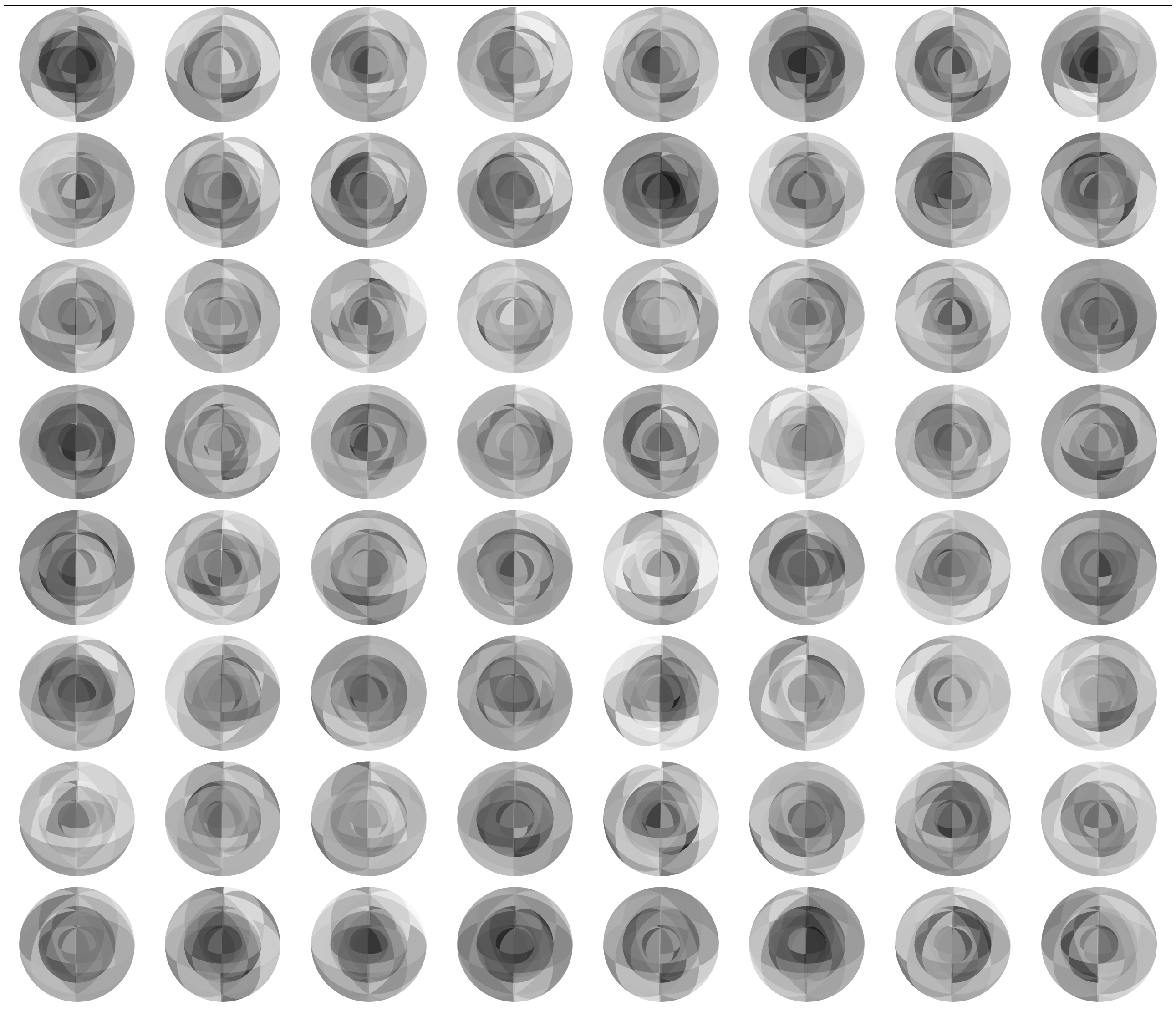


Figure 11. Visualization of a $8 \times 2 \times 3$ spherical kernel learned at the lowest octree level of our classification network. The kernel has 32 input and 64 output channels. Each sphere corresponds to one output channel, whereas the intensity value in each bin of a shown sphere corresponds to the first principal component of the input channels. The images are organized from top-left to bottom-right (row-wise) following the output channel indices, i.e. 1- 64. Distinct patterns for different spheres indicate effective kernel weight learning.