In Proceedings of the European Conference on Computer Vision (ECCV), 2018

ShapeCodes: Self-Supervised Feature Learning by Lifting Views to Viewgrids

Dinesh Jayaraman
UT Austin
dineshjayaraman@berkeley.edu

Ruohan Gao
UT Austin
gao@cs.utexas.edu

Kristen Grauman
UT Austin
grauman@fb.com

This document contains information supplementary to the main paper. In particular:

- **Sec 1:** Architectures for our method and baselines were overviewed in Sec 3.2, Fig 2, and Sec 4 in the paper. We specify these in more detail in Sec 1 in this document.
- **Sec 2:** Optimization details are specified in Sec 2.
- **Sec 3.1:** In Sec 3.3, we argued that ShapeCodes benefit object recognition by lifting 2D views to 3D. We also briefly mentioned that ShapeCodes are equivariant to 3D rotations. In Sec 3.1, we explain the rotational equivariance of ShapeCodes and its connection to their usefulness for object recognition.
- **Sec 3.2:** In Sec 4.3 (in particular, Tab 2) in the paper, we presented classification results using our features with fixed 1000-sample per class training sets, and mentioned experiments with varying training set sizes. In Sec 3.2 in this document, we present the results of those experiments, supplementing the results in the paper. Additionally, rather than reporting only best layer performance per method as in the paper, results are reported per-layer for fc1, fc2, and fc3.
- **Sec 3.3:** In Sec 4.3 in the paper, we evaluated our unsupervised features for discriminativeness when used for a classification task. We also referenced similar results for an image retrieval task, but omitted details due to space constraints. In Sec 3.3 in this document, we present these experiments and results.
- **Sec 3.4:** In Sec 4.3 in the paper, we compared our ShapeCode features against those from two alternative single-view 3D approaches: PointSetNet [5] and 3D-R2N2 [3]. We provide additional details on those experiments in Sec 3.4.
- **Sec 4.1 and Sec 4.2:** We showed quantitative reconstruction results on ModelNet seen/unseen class and ShapeNet seen/unseen class datasets in Sec 4.2 and Tab 1 in the paper. Here, we present additional analysis of our reconstruction results in two ways: (i) in Sec 4.1, we show category-wise quantitative results for our image-based shape reconstruction experiments, and (ii) in Sec 4.2, we analyze which views lead to higher and lower reconstruction errors for objects of each category.
- **Sec 4.3:** We showed a small selection of viewgrid reconstructions in Fig 3 in the paper. Here, we present randomly selected examples of inferred viewgrids by our method on all seen and unseen classes in both datasets, in Sec 4.3.

1. Network architectures

Architectures for our method and baselines were overviewed in Sec 3.2, Fig 2, and Sec 4 in the paper. Fig 1 shows the complete specifications of network architectures used in training various methods used in our experiments: Ours (and Ours w. canonical alignment), Autoencoder, DrLIM, and Egomotion. As shown in the figure, all architectures are kept as close to identical as possible, for fairness. Context is kept close to the authors' original model in [12].

2. Optimization details

Models are initialized with parameters drawn from a uniform distribution between -0.1 and +0.1. The mean squared error loss is optimized through standard minibatch SGD (batch size 32, momentum 0.9) via backpropagation. For our method and all baselines in Paper Sec 4, we optimize the learning rate hyperparameter on validation data. Training terminates when the loss on the validation set begins to rise.

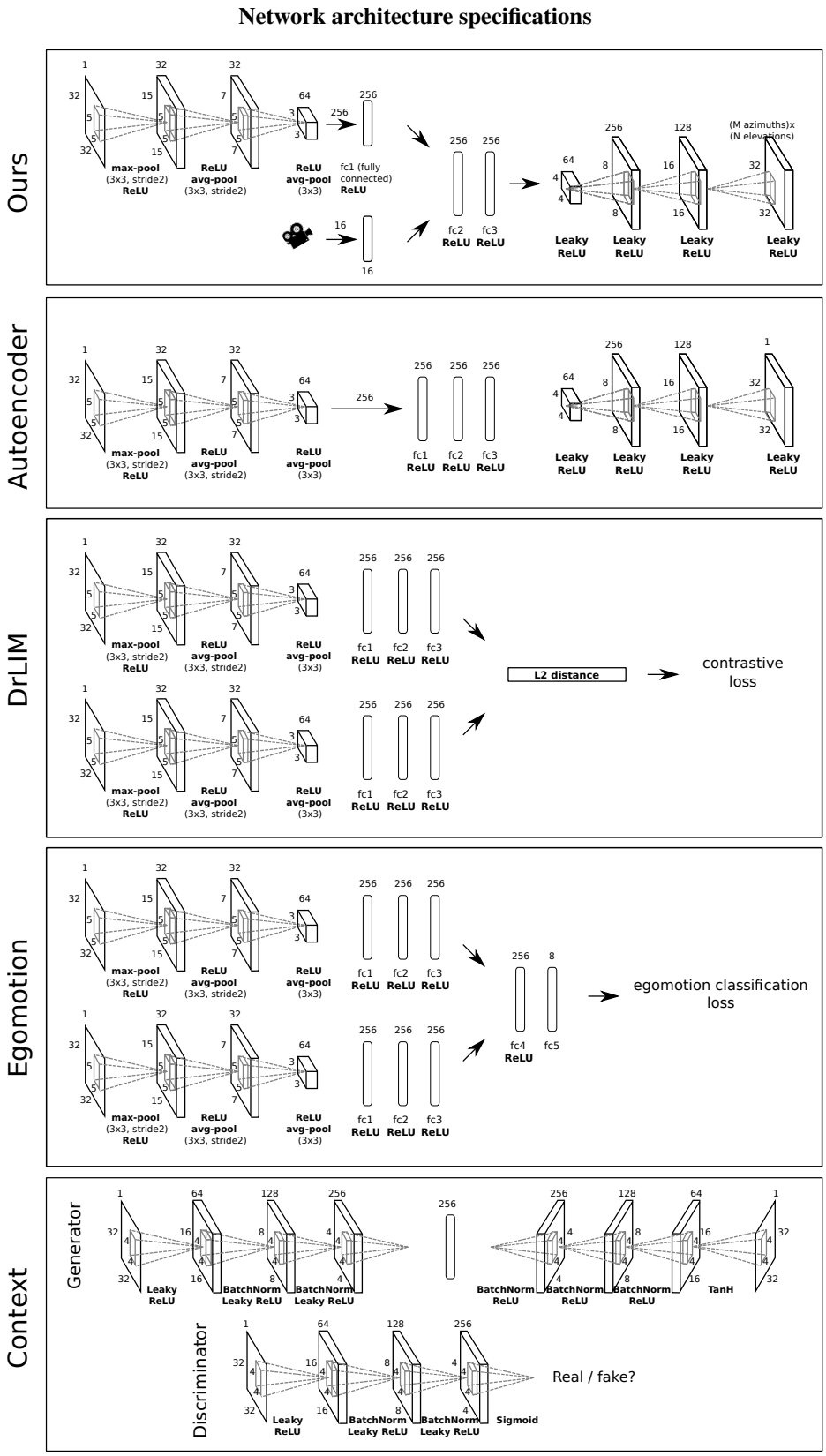

Figure 1: Architecture of our system and various baselines. See Sec 1.

3. ShapeCodes for recognition

3.1. Why are ShapeCodes useful for recognition?

ShapeCodes lift 2D views to 3D, which, as shown in the paper, is beneficial for downstream recognition tasks. Here, we elaborate on the *rotation-equivariance* of ShapeCode features and its connection to discriminativeness.

ShapeCode is rotation-equivariant: ShapeCodes aim to capture a representation of the object shape, aligned to the single observed viewpoint. When the input viewpoint changes, the target viewgrid for training ShapeCodes is a simple transformation of the original target viewgrid. Views in the viewgrid are reindexed to their new relative positions to the input. ShapeCodes thus encourage *equivariance* to observed viewpoint: for an object instance X with views $y(\boldsymbol{\theta_i})$,

$$ShapeCode(y(\boldsymbol{\theta_i} + \Delta\boldsymbol{\theta})) = f_{\Delta\boldsymbol{\theta}}(ShapeCode(\boldsymbol{\theta_i})), \quad (1)$$

where $f_{\Delta\boldsymbol{\theta}}(.)$ is a simple function representing the ShapeCode "equivariance map" for viewpoint rotations by $\Delta\boldsymbol{\theta}$.

Orbits of equivariance maps: If rotation is only permitted along one axis (easy to generalize to multiple axes), and Δ is sufficiently small, then all view features $ShapeCode(y(\boldsymbol{\theta}))$ can be reached from a given starting view $ShapeCode(y(\boldsymbol{\theta_0}))$ by applying $f_{\Delta\boldsymbol{\theta}}$ over and over, $\frac{\boldsymbol{\theta}-\boldsymbol{\theta_0}}{\Delta\boldsymbol{\theta}}$ times. In other words, in the learned ShapeCode space, all views lie within an "orbit" of the simple equivariance map $f_{\Delta\boldsymbol{\theta}}$.

Equivariant features for recognition: Why should these equivariant features aid recognition? Consider an "object instance classification" problem: all views $y(\boldsymbol{\theta})$ of an object X must be mapped to the same label. In the ShapeCode space, these views all lie within the orbit of a simple function. Instance classification is now the problem of drawing boundaries around this simple orbit, which requires only a low-capacity classifier. This is exactly what we mean when we say a feature space is discriminative: a classifier is easy to learn.

Taking a step back, classification by definition requires mapping all instances of a class to one label, so it is clear that it requires *invariance* to intra-class variations, including 3D rotations. For any such intra-class variation, equivariance to that variation is a principled intermediate objective, since invariance is a special case of equivariance. The usefulness of equivariant features in recognition has been explored before in [9, 4, 10, 7]. Intuitively, equivariance is better achieved without losing specificity, since a representation fully invariant to strong viewpoint changes—to the point of entirely different aspects—must also wash away defining properties of the object category.

3.2. Per-layer classification results vs. training dataset size

Nearest neighbor classification results are presented in the paper in Sec 4.3, as a means of evaluating the unsupervised features learned by our method for discriminativeness. In particular, Tab 2 in the main paper shows best results from across fc1, fc2, and fc3 for each method. Here, we break these results down layer-wise.

In addition, nearest neighbor classifiers are often sensitive to the size and constitution of the training set, so we also test the stability of the results in Tab 2 from the main paper. To do this, we sample multiple training sets of varying sizes ranging from 50 to 1000 samples per class (50, 100, 200, 300, ... 1000), and report accuracies for k-nearest neighbor classification (k=5 as in the paper) with each training set.

These are presented for both seen and unseen class subsets of both ModelNet and ShapeNet, in Fig 2. We observe that curves corresponding to different methods rarely overlap as the training set is varied, establishing the stability of the results in Tab 2. In particular, our ShapeCodes continue to produce the most discriminative features at all training set sizes. Trends for fc1, fc2, and fc3 are all similar.

3.3. Image retrieval results

In Sec 4.3 in the paper, we evaluated our unsupervised features for discriminativeness when used for a classification task. We also referenced similar results for an image retrieval task, but omitted details due to space constraints. For this task, we present a query image from the test set, and retrieve the closest images in the training set, as measured by Euclidean distance in the learned feature space. For representations that encode semantics, these closest images would belong to the same category as the query, so we evaluate various representations on their ability to retrieve images from the same class as the query.[1]

We measure top-1, top-5, and top-20 accuracies. We separately test fc1, fc2, and fc3 features from our method and the baselines, as in the classification experiments in Sec 4.3 in the paper. Results are shown in Tab 1, for both seen and unseen classes on ModelNet and ShapeNet.

Trends are very similar to those observed for classification in the paper. Ours and Ours w. CA once again easily outperform all baselines. Performance for all methods remain roughly similar at all three feature layers, except DrLIM. DrLIM is strongest at fc1, and weakens at fc2 and still

[1] For the unseen class experiments, images were retrieved from the corresponding unseen class training set, disjoint from the test set from which queries were drawn.

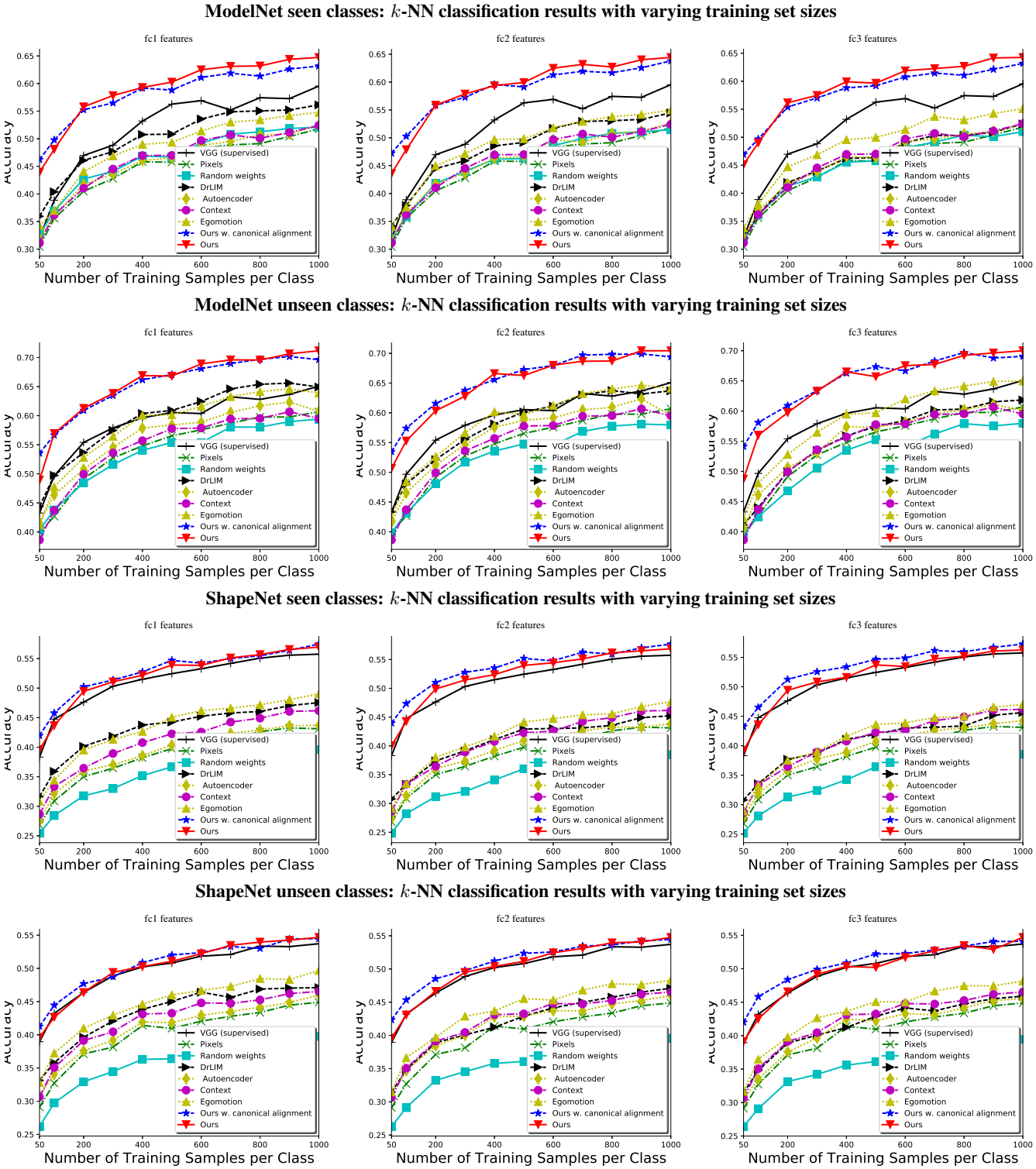

Figure 2: ModelNet and ShapeNet seen and unseen classes k-NN classification, varying training set size. Each column shows a different layer (fc1, fc2, fc3) for ShapeCodes and various baselines. VGG and Inpainting baselines do not use the same architecture; their curves are the same in all three columns. See Sec 3.2.

further at fc3, indicating that the invariance enforced by DrLIM at fc3 (See Fig 1) leads to loss of discriminativeness.

An interesting trend related to generalization to unseen classes is observed most clearly in the ShapeNet experiments, where the number of classes among seen and unseen classes is comparable (30 and 25 respectively), so that the numbers are roughly comparable among seen and unseen class experiments. Note how features from the Autoencoder baseline have much lower accuracies on unseen classes than on seen classes, demonstrating lack of generalization. Our ShapeCodes have very similar accuracies on seen and unseen classes, thus establishing that they learn generalizable features. DrLIM and Egomotion accuracies are also comparable for seen and unseen classes, but significantly lower than our method.

3.4. Details of comparison to PointSetNet and 3D-R2N2

In Sec 4.3 in the paper, we compared the classification performance of our features learned using a viewgrid reconstruction objective to features from two baseline approaches designed primarily for 3D reconstruction. Here we recap those baselines and provide additional details about this comparison.

PointSetNet [5] produces point coordinates representing the "point cloud" representation of object shape. As input, it receives a single image of an object and its ground truth segmentation mask. We use models provided by the authors, which were trained in Tensorflow on ShapeNet objects. We extract 768-dimensional features from the "two-branch" variant of their model. These features are extracted at the output of the encoder, right before processing through the first deconvolutional layer.

3D-R2N2 [3] constructs a voxel grid from any given number of views of an object. In our experiments, we operate it in single image mode. The authors provide models trained on ShapeNet in Theano. We extract 8192-dimensional (128x4x4x4 3D convolutional maps flattened) features at the output of the 3D convolutional LSTM in the encoder of their "Residual GRU network" model.

Note that the goal in those papers is reconstruction, not recognition. So the off-the-shelf models we use for this application are being used outside the authors original intent. In that sense, they suffer some inherent disadvantages. Being trained specifically for reconstruction, these existing off-the-shelf models lack the generality of our models. In particular:

- The authors' models were trained on images of objects with limited variation in viewpoint (PointSetNet: elevation fixed at 20°, and 3D-R2N2: elevations sampled from 25°to 30°). Throughout our experiments, we test on images more from more diverse viewpoints, varying over $\pm$ 90°(see Tab 1 (left) in the paper).
- Both these methods were trained solely on ShapeNet, so transfer to ModelNet is difficult, as reflected in their scores in Tab 2 in the paper.
- Authors of both these methods render object views slightly differently from us. Visually, their images have light gray backgrounds while ours are white. Their images also have a mix of grayscale and color objects, while ours are all rendered in grayscale for uniformity. We will open-source our rendering codes for future use and comparison.
- PointSetNet in particular was designed to exploit ground truth segmentation masks. However, these are not available in the unsupervised image feature learning setting, so we set the segmentation mask to the whole image.

We are the first to leverage single-view reconstruction as a self-supervised feature learning task that encourages useful shape cues to be captured. Our comparison to [5] and [3] is intended mainly to test that our approach yields better features than off-the-shelf prior methods that target single-view reconstruction as an end in itself. It is encouraging that targeting viewgrids as implicit 3D representations through our method works better than targeting point clouds or voxel grids (both explicit 3D representations) through these prior approaches. As pointed out in Sec 3.1 in the paper, viewgrids offer particular advantages for our embodied self-supervised learning setting.

4. Viewgrid lifting results analysis

4.1. Category-wise viewgrid results

We showed quantitative reconstruction results on ModelNet seen/unseen class and ShapeNet seen/unseen class datasets in Sec 4.2 and Tab 1 in the paper.

To break down those results in more detail, we present per-pixel mean squared errors for each category individually in Fig 3 and 4, arranged in sorted order for seen and unseen classes separately for each dataset.

As can be seen, while the distribution of errors among unseen classes is shifted towards higher errors from the seen classes, there is a significant overlap *i.e.*, many unseen classes have lower reconstruction errors than many seen classes, indicating significant generalization from seen to unseen classes.

4.2. Influence of observed view position

Having analyzed reconstruction error on a category-wise basis in Sec 4.1, we now further break down the errors summarized in Tab 1 in the paper, this time on the basis of the

ModelNet retrieval results

Datasets→	ModelNet-seen classes (30 cls)								
Layers→	fc1			fc2			fc3		
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
VGG [13] (supervised)	67.0	60.0	50.5	67.0	60.0	50.5	67.0	60.0	50.5
Pixels	55.2	46.9	36.6	55.2	46.9	36.6	55.2	46.9	36.6
Random weights	51.6	44.2	34.9	50.0	42.9	34.2	50.4	42.8	34.0
DrLIM [6]	58.8	51.1	41.7	56.7	48.8	39.7	53.8	46.3	37.1
Autoencoder [8, 2, 11]	54.8	46.5	37.5	55.1	46.7	37.5	55.5	47.3	37.7
Context [12]	54.0	45.7	36.5	54.0	45.7	36.5	54.0	45.7	36.5
Egomotion [1]	56.6	48.8	39.6	57.6	49.6	39.9	57.4	49.6	39.7
Ours w. canonical alignment	64.3	57.9	49.5	64.4	58.2	**50.2**	63.5	57.7	**50.0**
Ours	**65.6**	**59.2**	**49.7**	**64.9**	**58.3**	49.5	**65.5**	**58.4**	49.7
Datasets→	ModelNet-unseen classes (10 cls)								
Layers→	fc1			fc2			fc3		
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
VGG [13] (supervised)	64.4	56.7	49.3	64.4	56.7	49.3	64.4	56.7	49.3
Pixels	60.3	53.9	44.5	60.3	53.9	44.5	60.3	53.9	44.5
Random weights	60.0	52.6	44.3	57.9	51.7	43.5	57.9	51.1	43.0
DrLIM [6]	64.7	58.5	49.5	63.9	56.8	47.6	60.7	54.0	44.8
Autoencoder [8, 2, 11]	60.5	54.7	45.5	60.0	54.3	45.5	60.6	54.0	45.1
Context [12]	60.3	54.4	45.1	60.3	54.4	45.1	60.3	54.4	45.1
Egomotion [1]	63.4	57.4	48.2	64.2	57.3	48.2	63.8	57.4	48.3
Ours w. canonical alignment	69.0	63.9	**56.3**	68.8	63.8	**56.5**	68.4	63.3	**56.3**
Ours	**69.8**	**64.7**	55.9	**69.2**	**63.9**	55.5	**68.5**	**64.0**	55.2

ShapeNet retrieval results

Datasets→	ShapeNet-seen classes (30 cls)								
Layers→	fc1			fc2			fc3		
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
VGG [13] (supervised)	54.7	49.3	42.0	54.7	49.3	42.0	54.7	49.3	42.0
Pixels	42.8	36.5	28.6	42.8	36.5	28.6	42.8	36.5	28.6
Random weights	38.6	32.9	26.4	37.5	31.9	25.8	36.4	32.0	25.7
DrLIM [6]	47.1	40.2	31.3	45.9	39.6	30.2	43.3	37.1	28.9
Autoencoder [8, 2, 11]	42.5	36.8	29.6	42.6	37.1	29.8	42.3	37.4	29.5
Context [12]	46.0	39.1	31.1	46.0	39.1	31.1	46.0	39.1	31.1
Egomotion [1]	48.9	42.1	33.3	47.5	40.6	32.1	47.2	40.1	31.7
Ours w. canonical alignment	**56.1**	**50.8**	**44.0**	**57.3**	**51.6**	**45.2**	**56.9**	**51.4**	**43.6**
Ours	56.0	50.2	42.4	55.8	50.5	42.8	55.3	49.6	42.2
Datasets→	ShapeNet-unseen classes (25 cls)								
Layers→	fc1			fc2			fc3		
Methods↓/Metrics→	top-1	top-5	top-20	top-1	top-5	top-20	top-1	top-5	top-20
VGG [13] (supervised)	54.0	49.3	42.4	54.0	49.3	42.4	54.0	49.3	42.4
Pixels	44.6	40.4	33.6	44.6	40.4	33.6	44.6	40.4	33.6
Random weights	26.6	20.8	15.4	25.9	21.0	15.3	25.7	20.4	15.2
DrLIM [6]	48.1	41.5	36.1	48.3	41.8	34.7	44.9	41.1	34.1
Autoencoder [8, 2, 11]	30.7	23.9	17.0	31.1	23.9	17.3	30.5	23.9	17.2
Context [12]	46.8	42.1	35.4	46.8	42.1	35.4	46.8	42.1	35.4
Egomotion [1]	49.9	43.6	37.1	49.4	42.8	36.3	48.7	42.5	36.1
Ours w. canonical alignment	53.7	49.1	**42.9**	53.4	**49.6**	**43.5**	52.8	49.1	**43.6**
Ours	**54.9**	**49.3**	42.7	**55.1**	49.5	42.7	**54.7**	**49.2**	42.4

Table 1: **Above:** Retrieval experiments on ModelNet (1000 training samples per class). Seen class (top) and unseen class (bottom) results. Results reported as top-1, top-5 and top-20 accuracies. (Higher is better.) **Below:**.Retrieval experiments on ShapeNet (1000 training samples per class). Seen class (top) and unseen class (top) results. Results reported as top-1, top-5 and top-20 accuracies. (Higher is better.) See Sec 3.3.

dependence of those errors on the observed view. Specifically, in both ModelNet and ShapeNet, models are manually aligned to some canonical starting positions. This means that we can meaningfully attempt to understand which positions in the viewgrid, when observed, led to higher or lower reconstruction errors for the full viewgrid for each category. In other words, which views are more or less informative to our ShapeCodes learning approach? The following is a longer version of Table 4 in the main paper and accompanying discussion.

Fig 5 shows these results for ModelNet seen classes. For each class, a heatmap of mean-square errors is overlaid over the average viewgrid for that class. We also show a map for all seen classes at the top; the accompanying colorbar illustrates how lower MSEs correspond to darker, colder, blue colors, and higher MSEs to lighter, warmer, yellow colors. Similar charts for ModelNet unseen classes, ShapeNet seen classes, and ShapeNet unseen classes are shown in Fig-

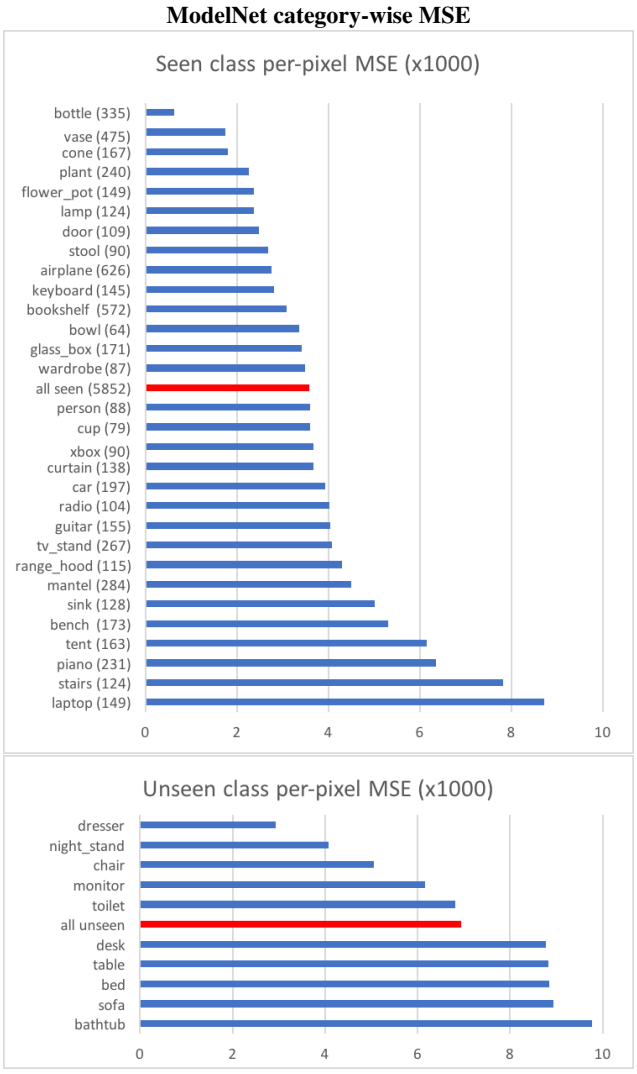

Figure 3: ModelNet seen (top) and unseen (bottom) class reconstruction performance in MSE per class (lower is better). Seen classes are not evenly represented in the training set, so the number of training samples per class is indicated in parentheses next to the corresponding class name. See Sec 4.1.

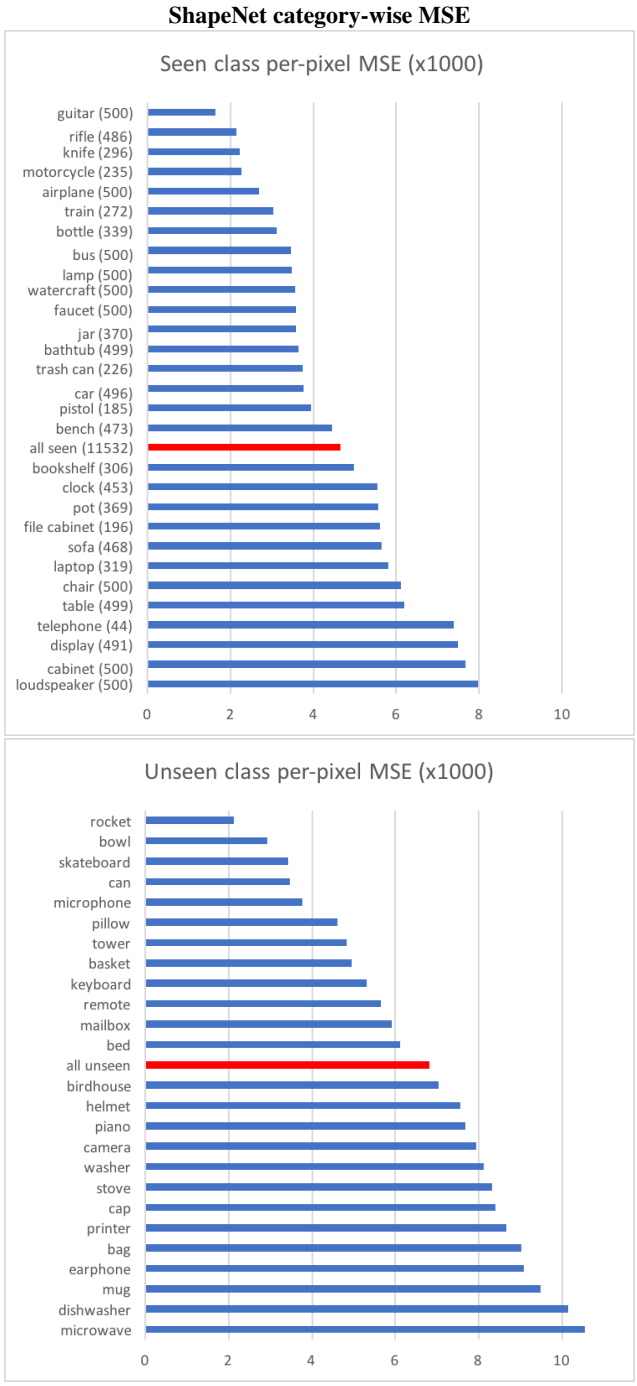

Figure 4: ShapeNet seen (top) and unseen (bottom) class reconstruction performance in MSE per class (lower is better). Seen classes are not evenly represented in the training set, so number of training samples per class are indicated in parentheses next to the class names. See Sec 4.1.

ures 6, 7, and 8 respectively.

Studying these heatmaps reveal some intuitive and interesting trends. Across all datasets, perfectly aligned viewing positions from where only a single, or a small number of faces of an object are visible, have yellowish, lighter highlights indicating high reconstruction errors conditioned on those views *i.e*. lower informativeness. This is intuitively reasonable, as observing one face leaves more of the object fully unobserved, making the hallucination of rotated views much harder. See for instance, the ModelNet seen class "bench" heatmap in Fig 5, which has characteristic yellowish horizontal and vertical stripes running across the viewgrid corresponding to elevations and azimuths that only reveal a small number of faces of the object. Top and bot-

tom views, sampled at -90°and +90°in ModelNet, are consistently uninformative, since very different shapes can have very similar overhead views. Among the unseen classes in Fig 6, see the chair class heatmap that also shows how perfectly aligned views lead to poor viewgrid inference.

Shapes that have narrow linear projections along some directions tend to present very little information from the corresponding views. See the horizontal view of the airplane and the keyboard (middle row in the corresponding viewgrids, corresponding to zero azimuth) in Fig 5, or the side-on views for the display category heatmap in Fig 7, and for the earphone category heatmap in Fig 8.

Overall, these trends largely agree with our intuitive notions of which views are most informative for 3D understanding, and serve as evidence that our method learns to lift 2D shapes to 3D shapes by observing meaningful and appropriate cues.

4.3. Additional viewgrid examples

Finally, we present additional examples of reconstruction, to accompany those presented in Fig 3 in the paper. In particular, we show a randomly sampled object (observed at a random view) and the corresponding viewgrid reconstructed by our method for each seen and unseen category in ModelNet (seen: Fig 9, unseen: Fig 10) and ShapeNet (seen: Fig 11, unseen: Fig 12).

These are presented for inspection to get a sense for the tendencies, successes, and failings of the underlying ShapeCodes representation. See for example the keyboard example in Fig 9, where the observed view is a very narrow strip, with slight variations in shading providing just the only geometric cue. Yet, our reconstruction is reasonable. Even for the guitar example in the same figure, a similar linear view with slight shading differences is observed. Our method correctly reconstructs guitar-like shapes, since the pose and shape of the guitar are insufficiently constrained by the observed view, our method averages over all allowable possibilities to minimize MSE, resulting in fuzzy output views.

Another interesting example is the laptop image in Fig 9, where the observed view could represent a laptop that is opened at either the front end or the back end. In the reconstruction, the network appears to average over both possibilities in all views, so that there are several views in the reconstructed viewgrid that look like a laptop with two screens open and facing each other.

Other cases demonstrate that the reconstructed views output by our method do not fully represent its internal understanding. For instance, in the range hood example in Fig 9, the reconstructed version of the observed view appears to lose the clear layering in the original view that is an important cue in understanding its geometry. However, the remaining views in the reconstruction demonstrate that while the network has not successfully rendered the layering in the observed view, the network did observe the layering cue and form an appropriate representation of the shape.

The “table”, “night stand”, “dresser”, and “desk” reconstructions among unseen classes in Fig 10 also demonstrate good generalization to unseen shapes. On the ShapeNet dataset too, the example reconstructions for “camera”, “can”, “basket” in Fig 12 also show evidence of good generalization.

Our method learns ShapeCode representations that lift 2D views to 3D, by training on the viewgrid reconstruction task. As such, the recovered viewgrids shown in these figures are important not as photorealistic 3D reconstructions, but rather as visualized evidence of meaningful shape representations being trained on the pretext task, incorporating uncertainty about unobserved views.

References

[1] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *ICCV*, 2015.

[2] Y. Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[3] C. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstrution. In *ECCV*, 2016.

[4] T. S. Cohen and M. Welling. Transformation properties of learned visual representations. *ICLR*, 2014.

[5] H. Fan, H. Su, and L. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 38, 2017.

[6] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality Reduction by Learning an Invariant Mapping. *CVPR*, 2006.

[7] G. Hinton, A. Krizhevsky, and S. Wang. Transforming auto-encoders. In *ICANN*, 2011.

[8] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[9] D. Jayaraman and K. Grauman. Learning image representations tied to egomotion from unlabeled video. *International Journal of Computer Vision*, 2017.

[10] K. Lenc and A. Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 991–999, 2015.

[11] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*, pages 52–59. Springer, 2011.

[12] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[13] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

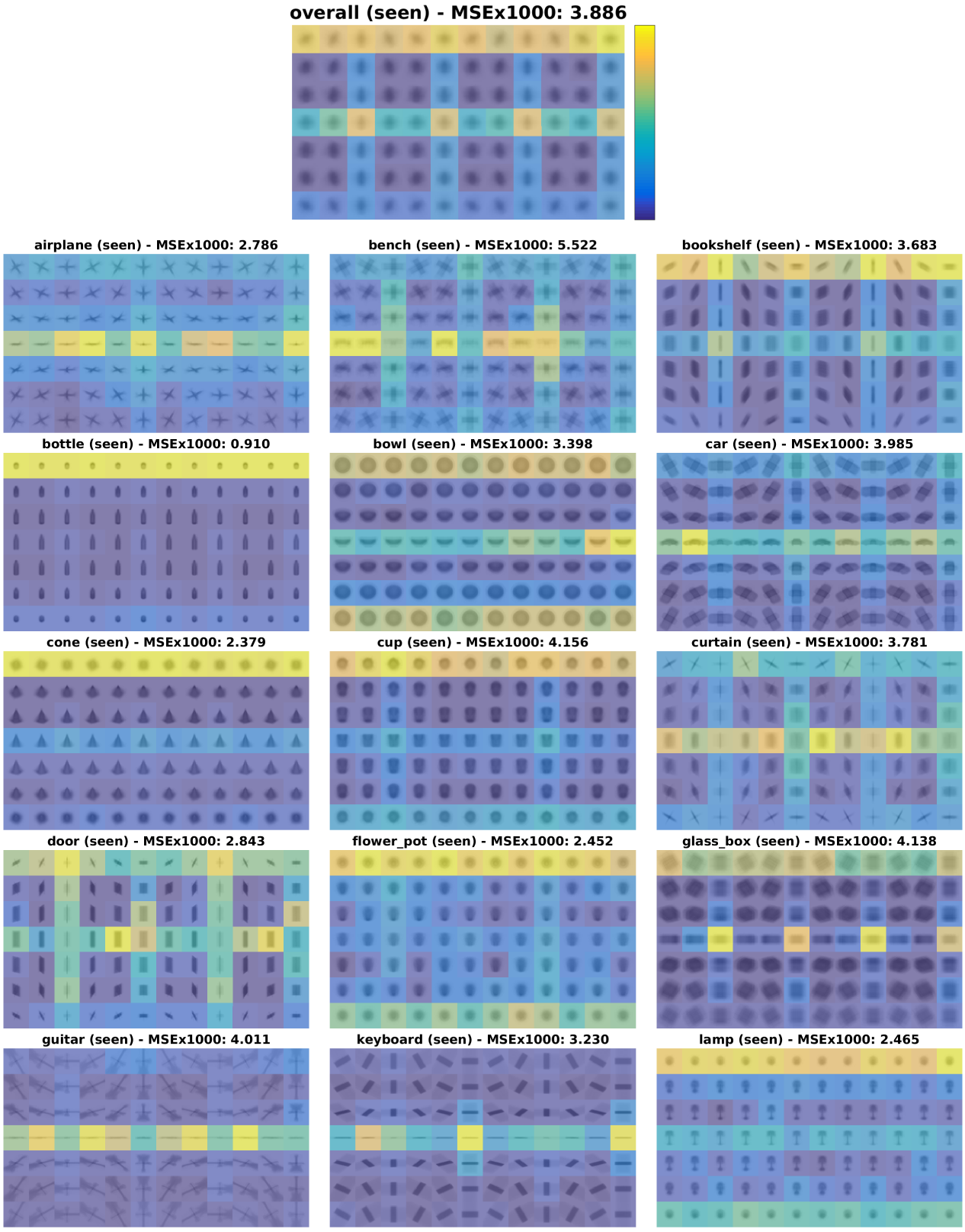

Figure 5: ModelNet seen classes reconstruction MSE, conditioned on observed view - part 1 of 2 (best observed in pdf at high resolution). See Sec 4.2. (continues on the next page)

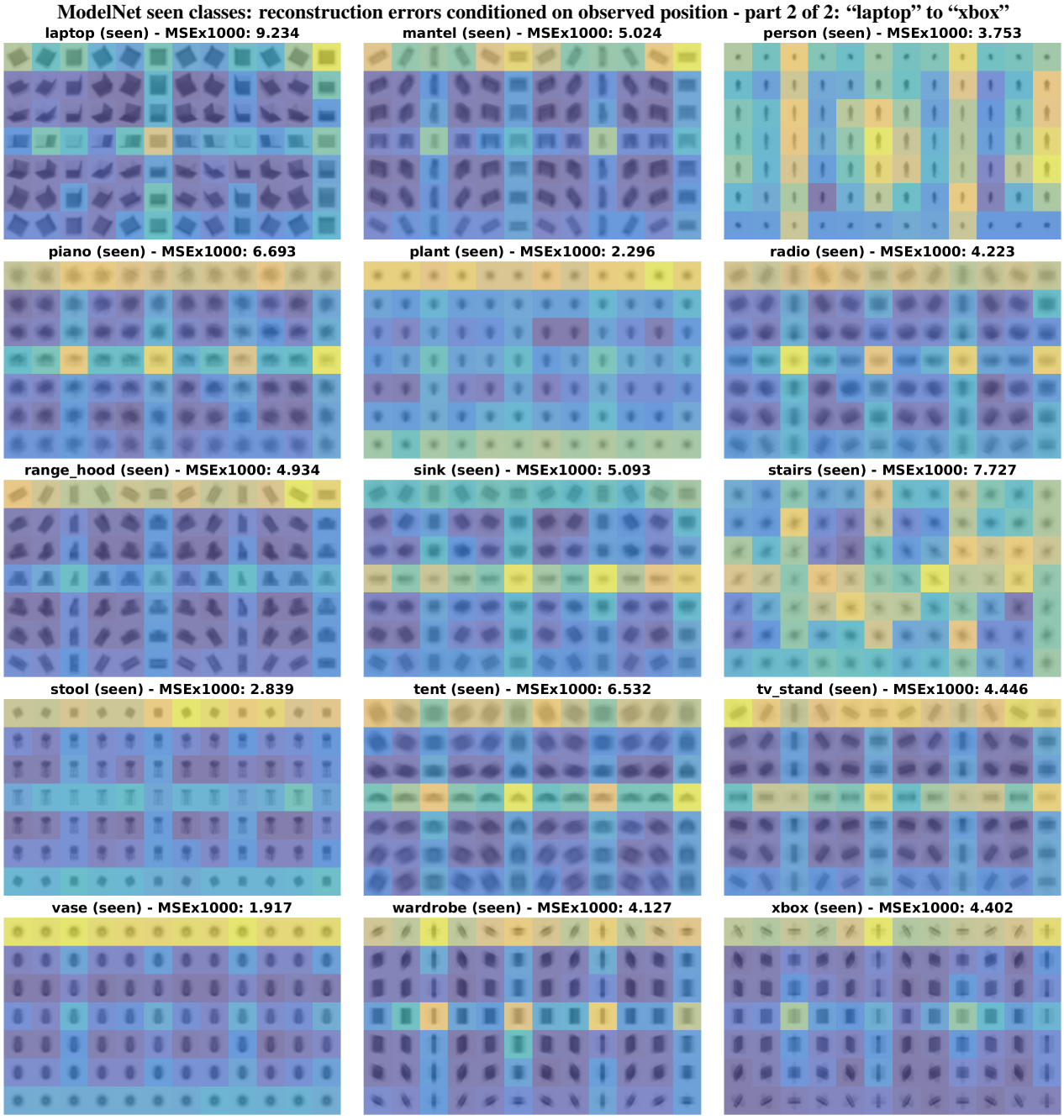

Figure 5: (continued from the previous page) ModelNet seen classes reconstruction MSE, conditioned on observed view - part 2 of 2 (best observed in pdf at high resolution). See Sec 4.2.

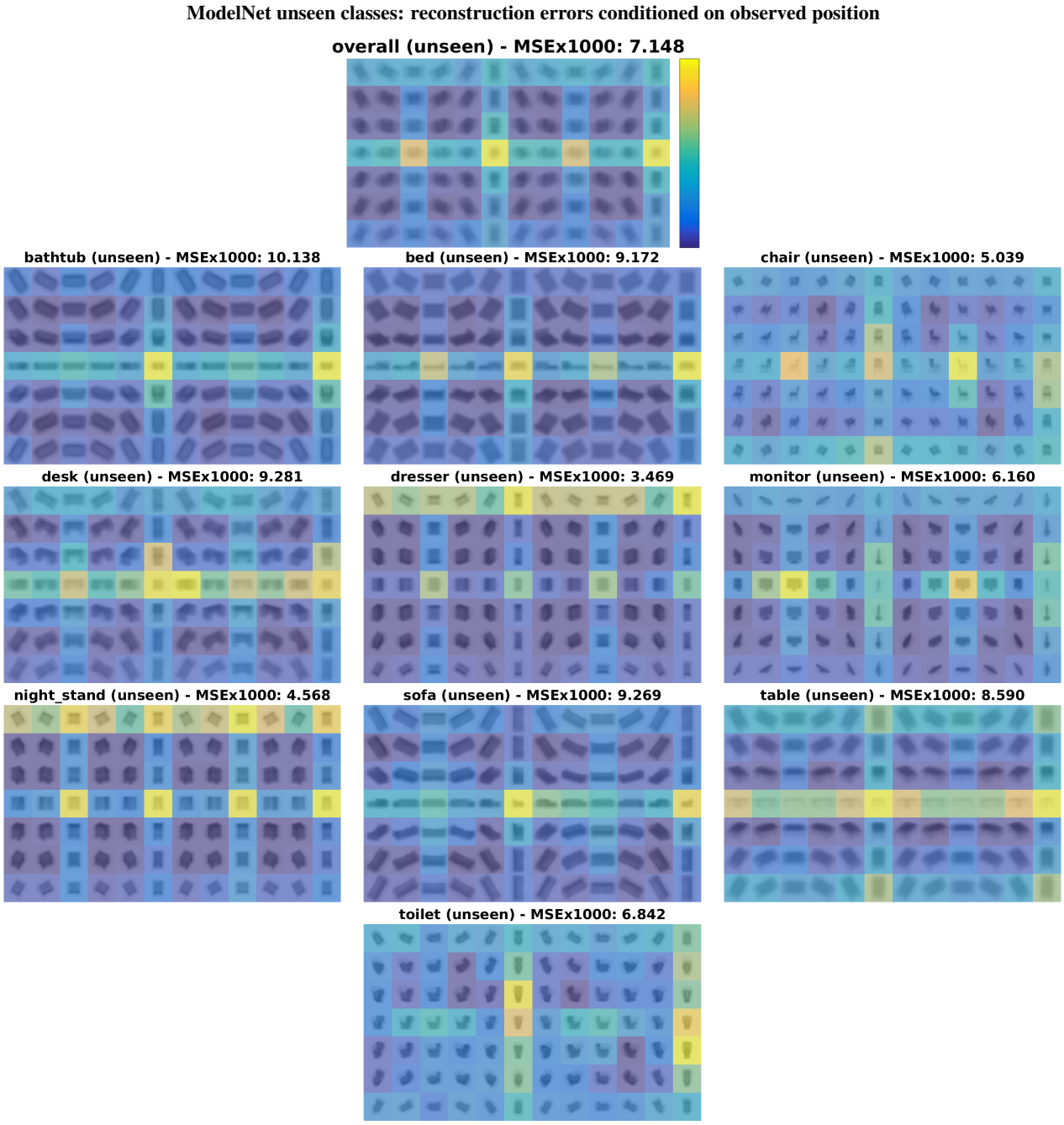

Figure 6: ModelNet unseen classes reconstruction MSE, conditioned on observed view (best observed in pdf at high resolution). See Sec 4.2.

ShapeNet seen classes: reconstruction errors conditioned on observed position - part 1: “airplane” to “jar”

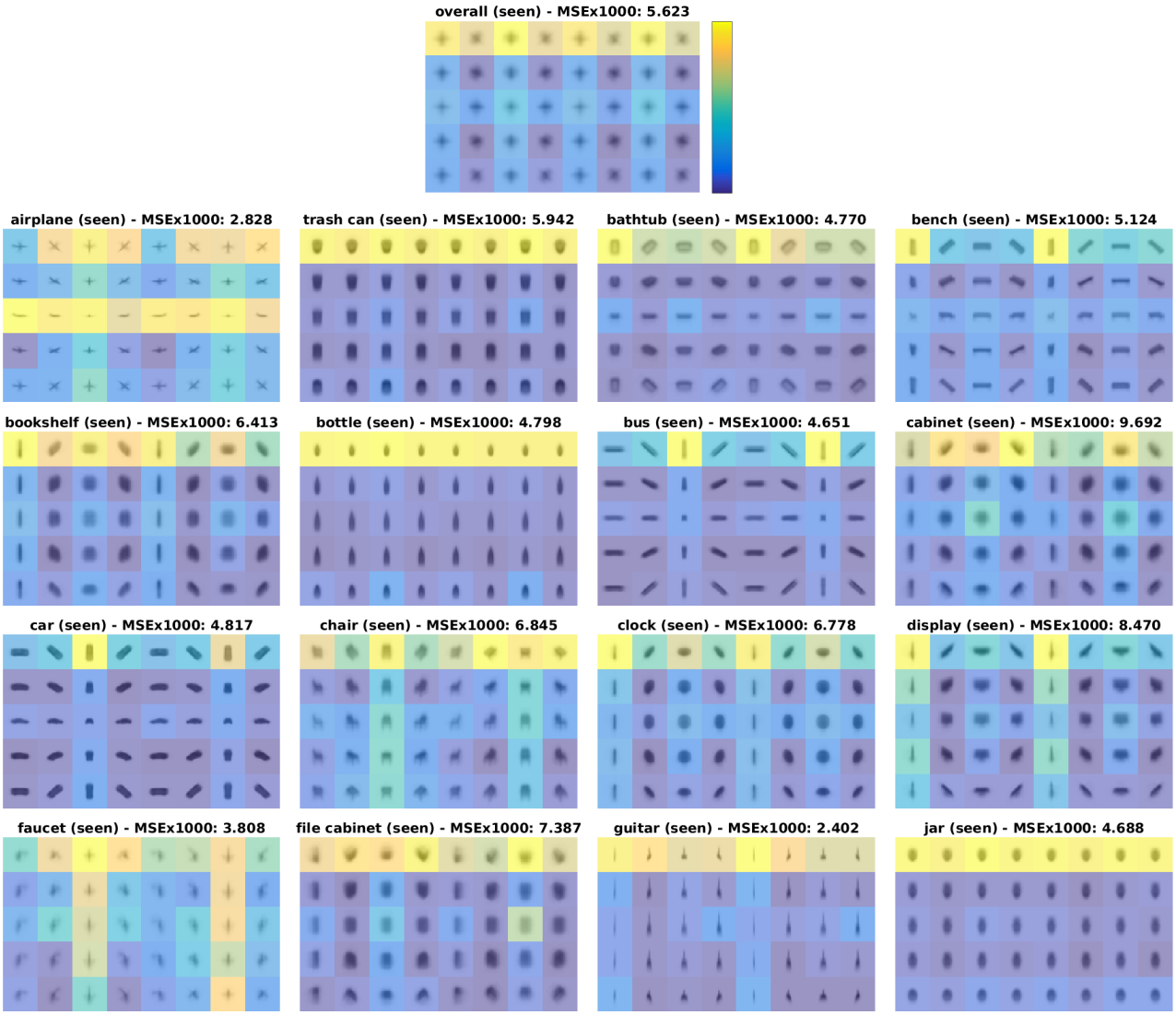

Figure 7: ShapeNet seen classes reconstruction MSE, conditioned on observed view (best observed in pdf at high resolution). See Sec 4.2. (continues on the next page)

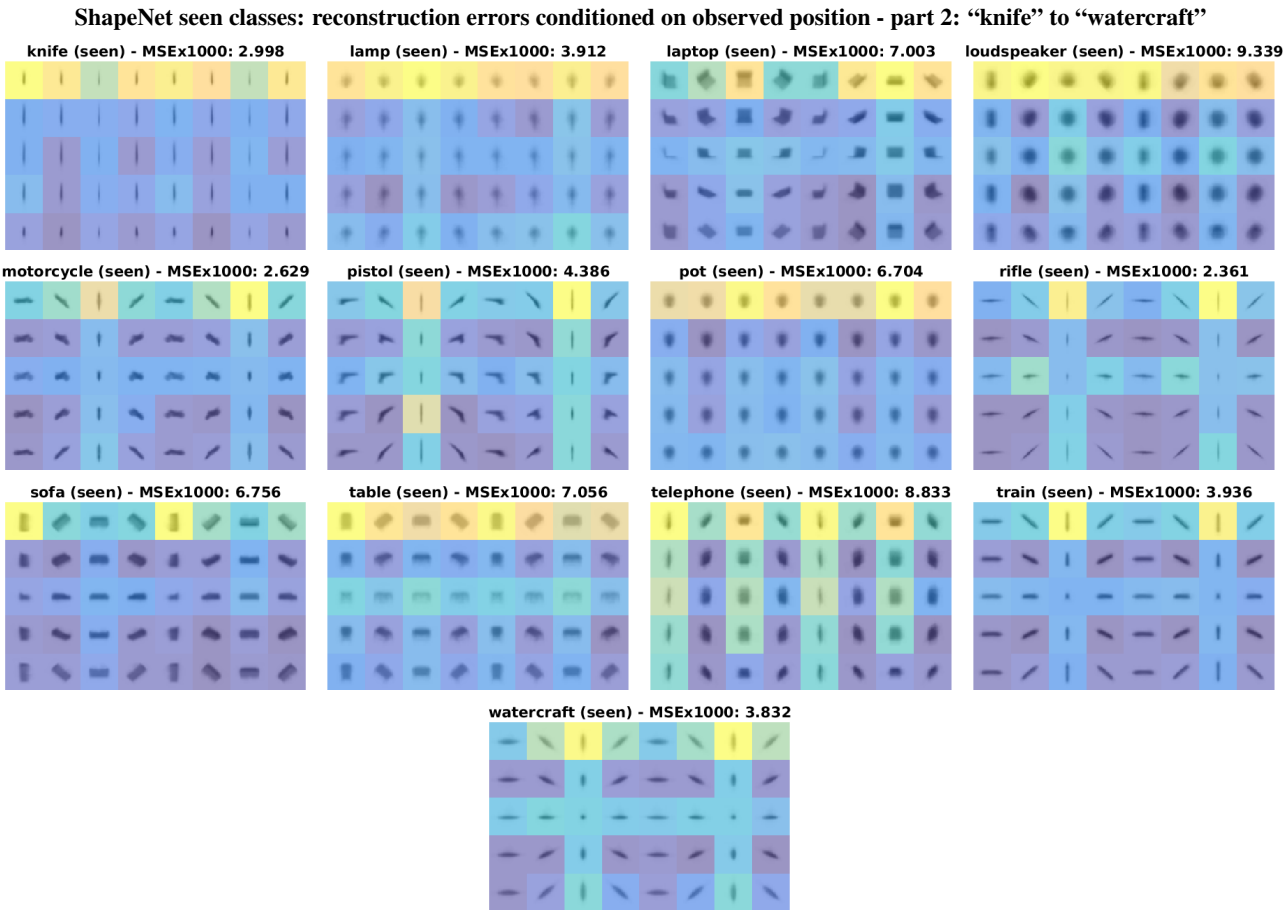

Figure 7: (continued from previous page) ShapeNet seen classes reconstruction MSE, conditioned on observed view (best observed in pdf at high resolution). See Sec 4.2.

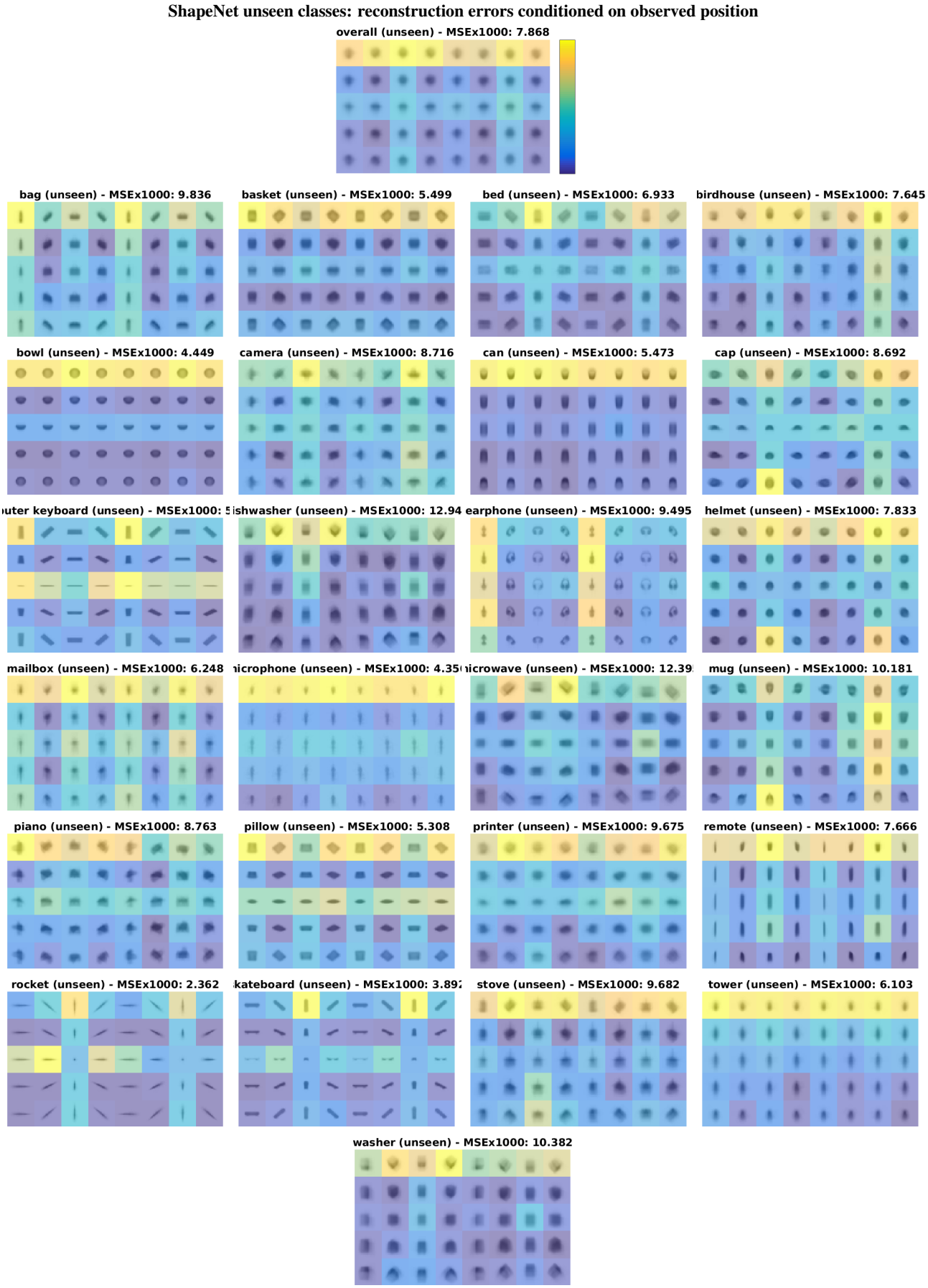

Figure 8: ShapeNet unseen classes reconstruction MSE, conditioned on observed view (best observed in pdf at high resolution). See Sec 4.2.

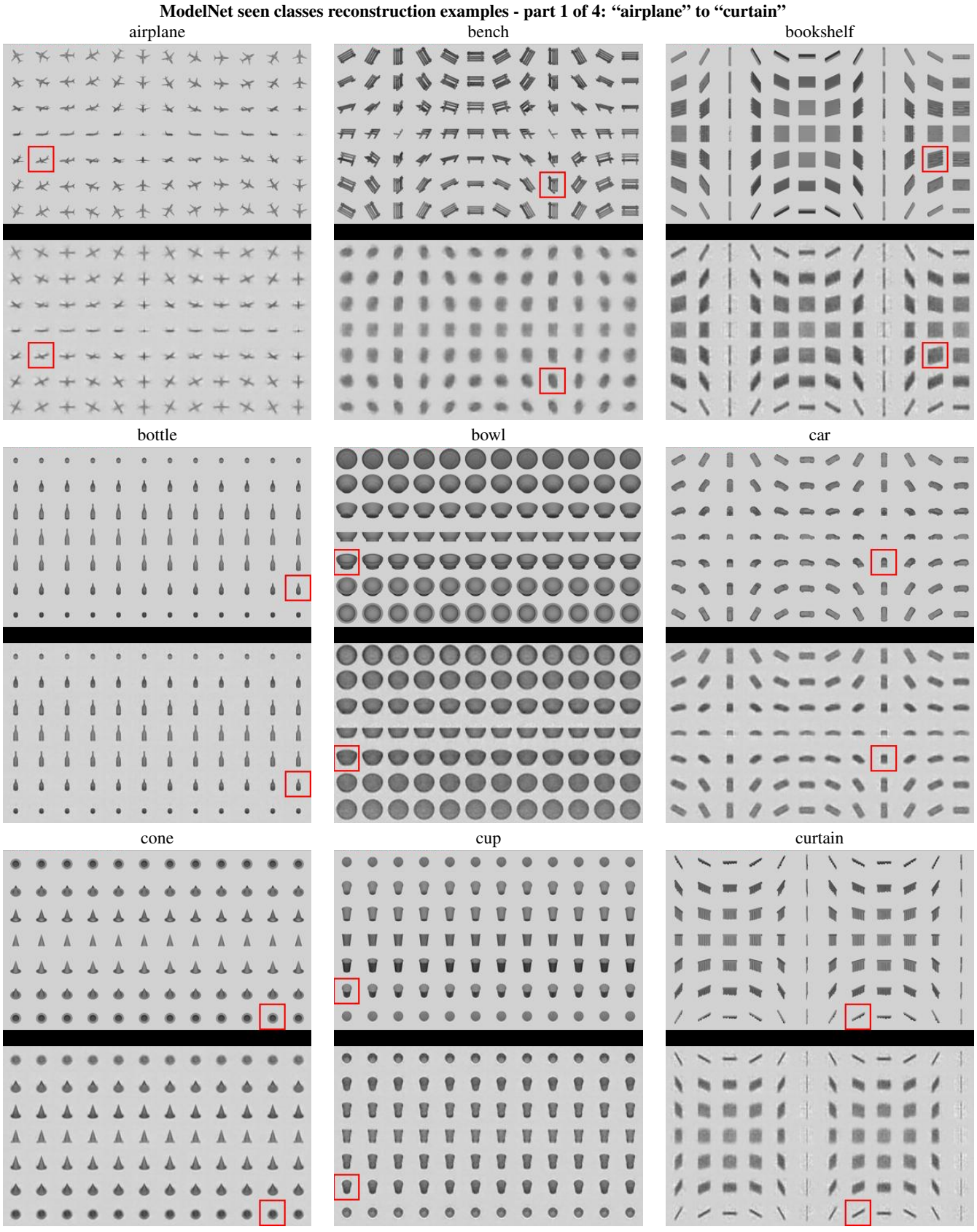

Figure 9: Examples of randomly sampled reconstructions from various ModelNet seen classes - part 1 of 4 (best observed in pdf at high resolution). See Sec 4.3. (continues on the next page)

ModelNet seen classes reconstruction examples - part 2 of 4: “door” to “person”

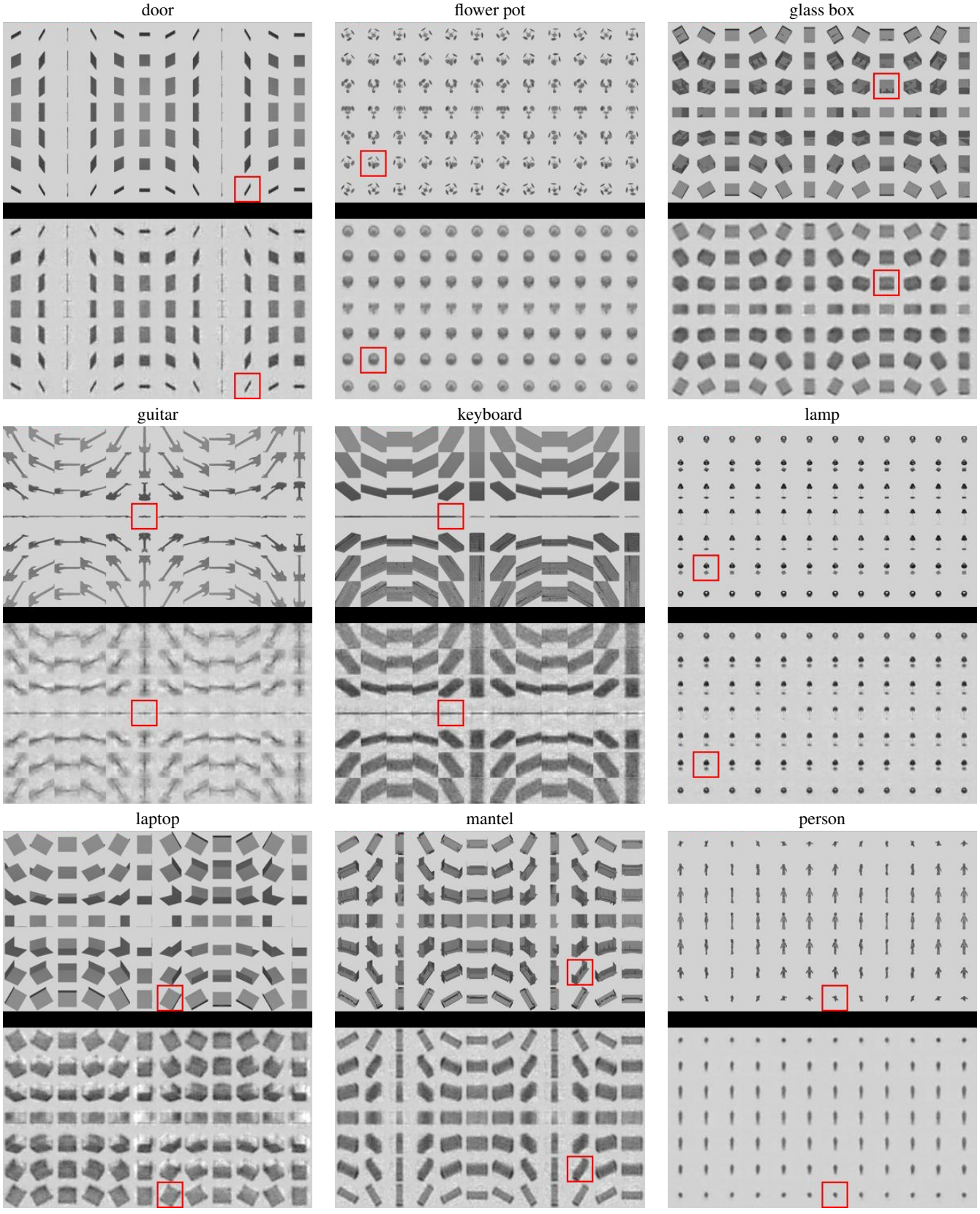

Figure 9: (continued from the previous page) Examples of randomly sampled reconstructions from various ModelNet seen classes - part 2 of 4 (best observed in pdf at high resolution). See Sec 4.3. (continues on the next page)

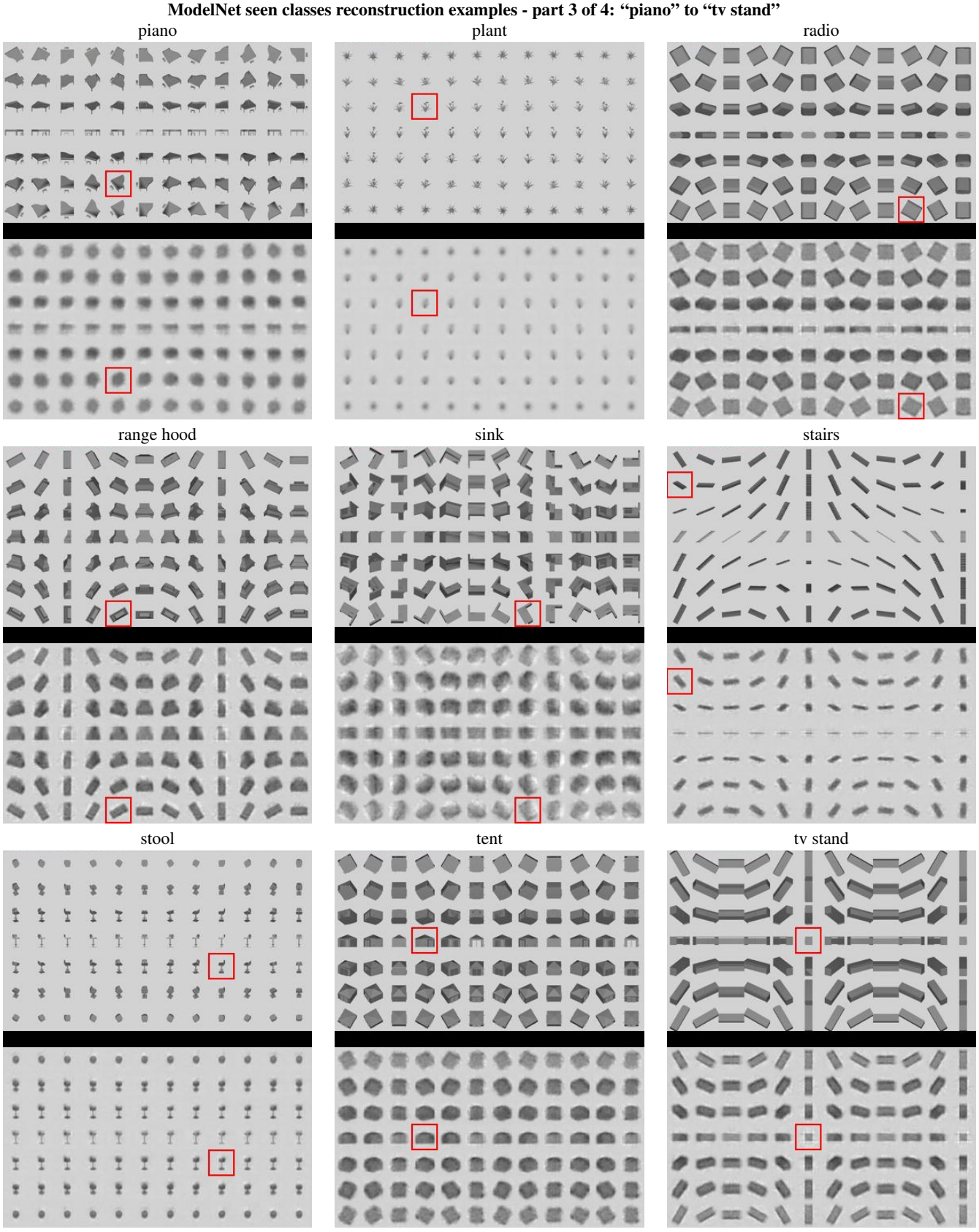

Figure 9: (continued from the previous page) Examples of randomly sampled reconstructions from various ModelNet seen classes - part 3 of 4 (best observed in pdf at high resolution). See Sec 4.3.

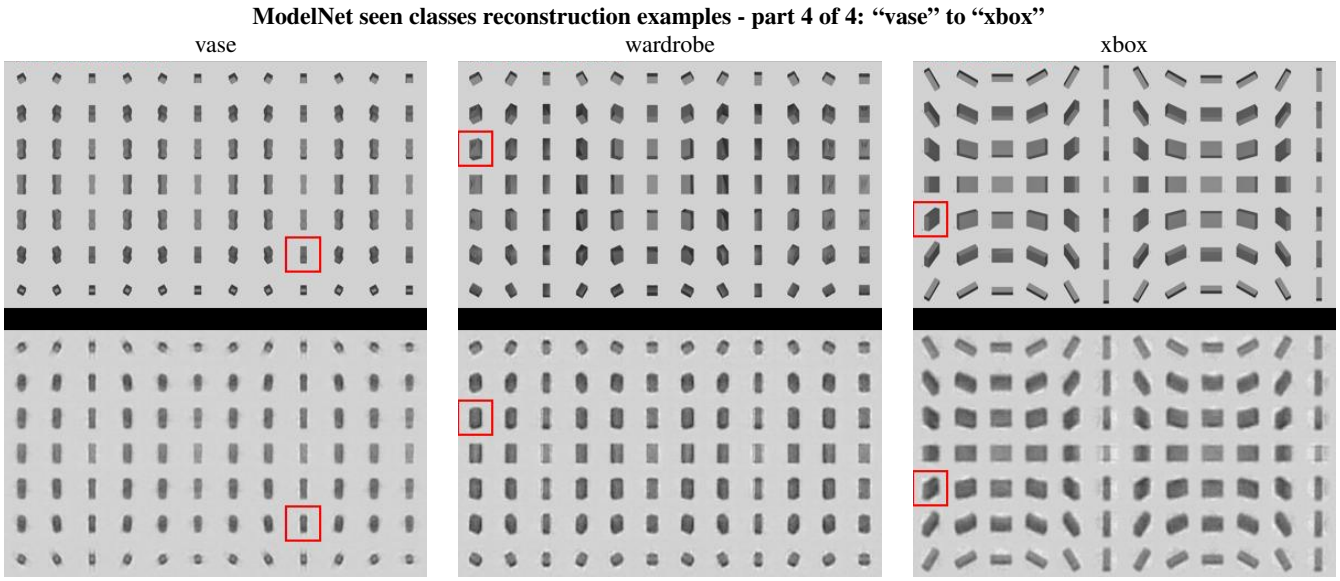

Figure 9: Examples of randomly sampled reconstructions from various ModelNet seen classes - part 4 of 4 (best observed in pdf at high resolution). See Sec 4.3. (continues on the next page)

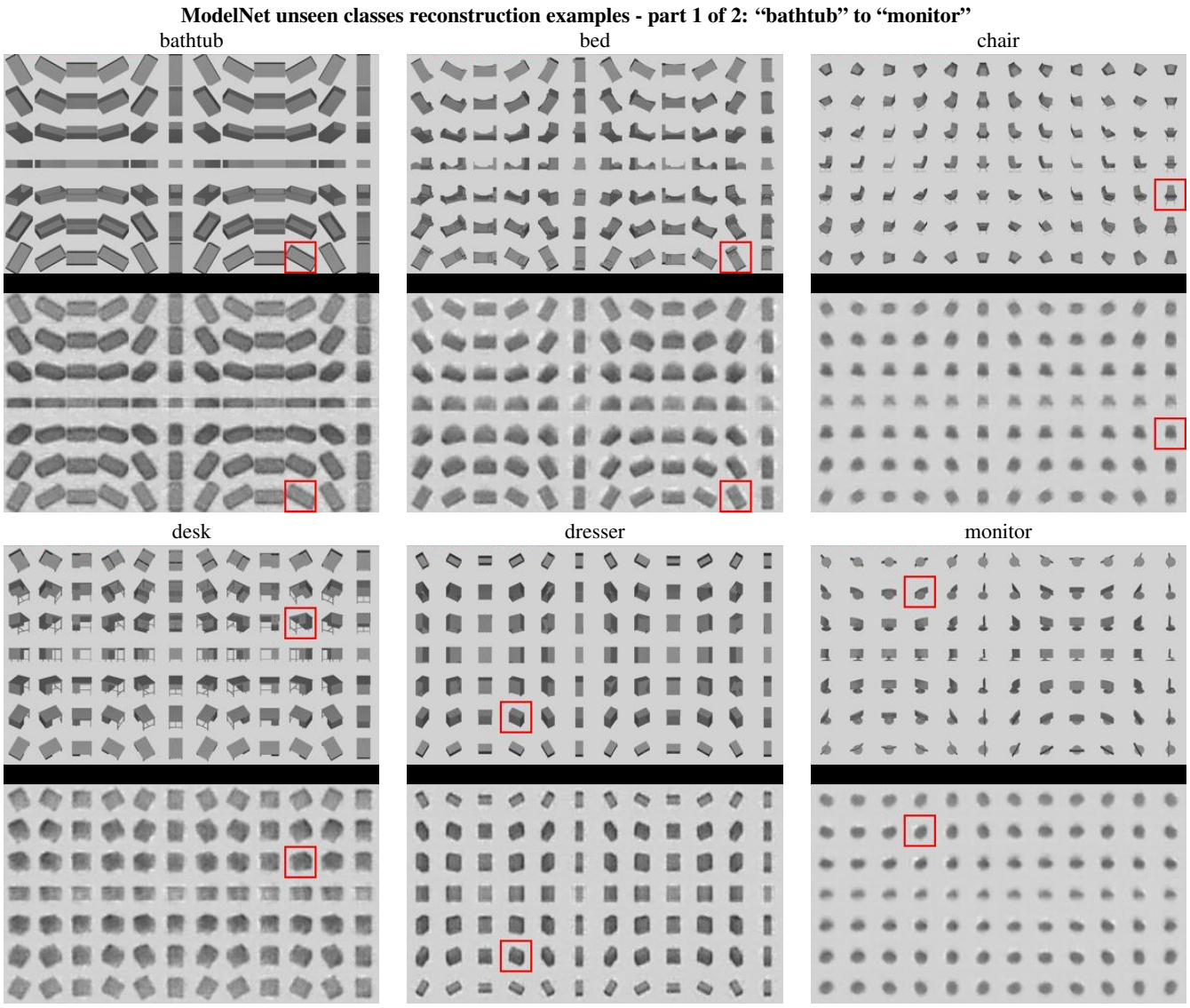

Figure 10: Examples of randomly sampled reconstructions from various ModelNet unseen classes - part 1 of 2 (best observed in pdf at high resolution). See Sec 4.3. (continues on the next page)

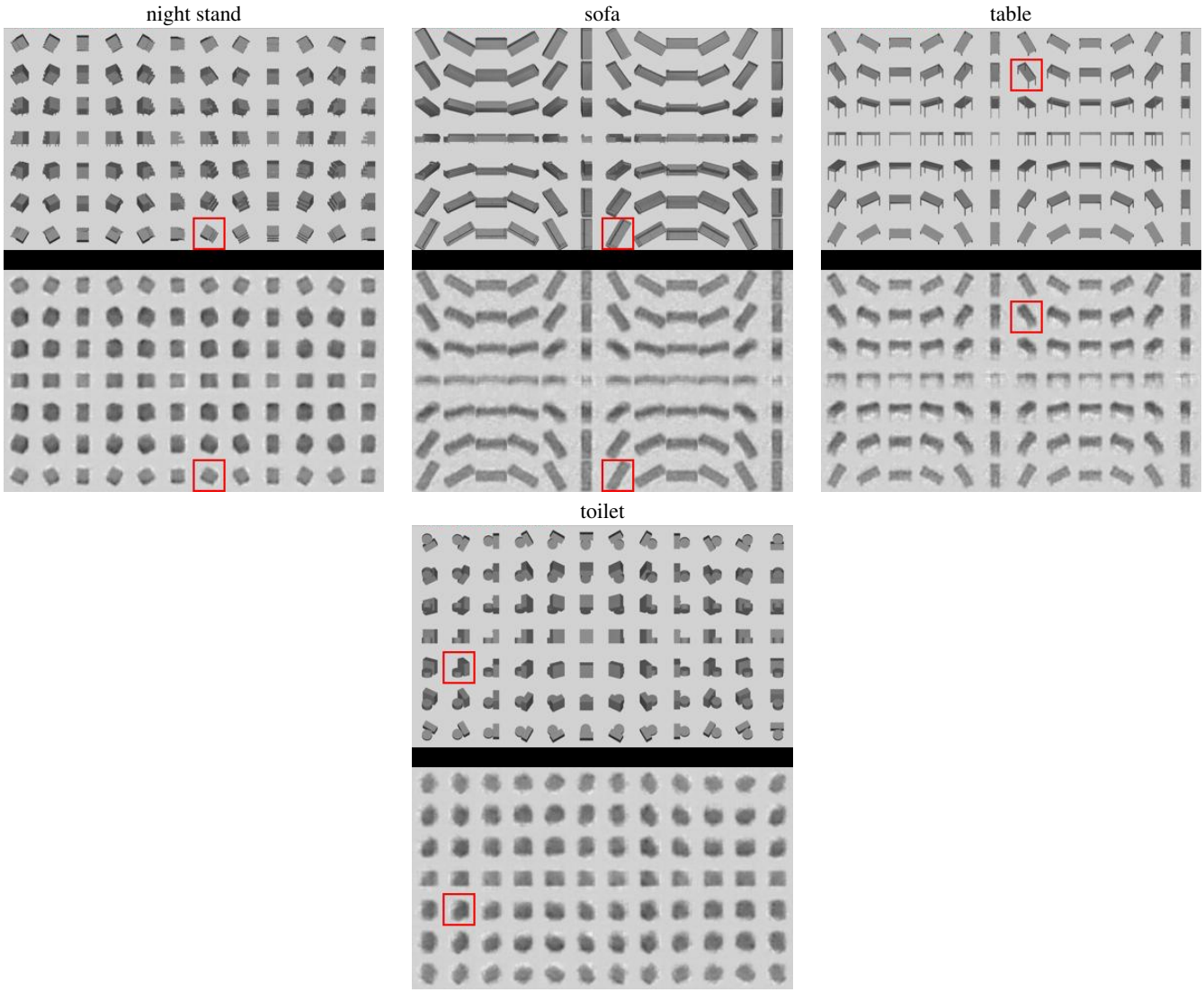

Figure 10: (continued from the previous page) Examples of randomly sampled reconstructions from various ModelNet unseen classes - part 2 of 2 (best observed in pdf at high resolution). See Sec 4.3.

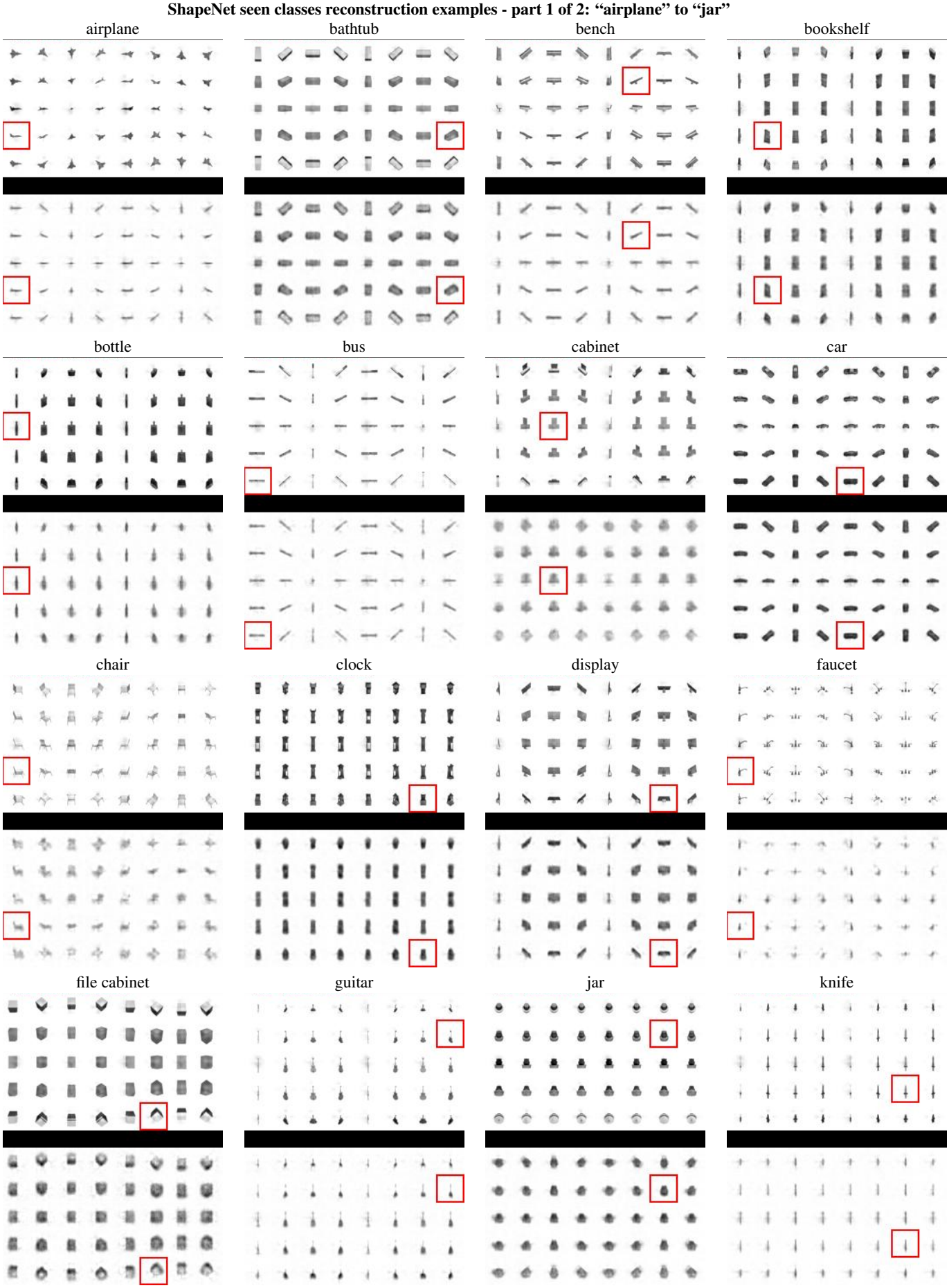

Figure 11: Examples of randomly sampled reconstructions from various ShapeNet seen classes - part 1 of 2 (best observed in pdf at high resolution). See Sec 4.3. (continues on next page)

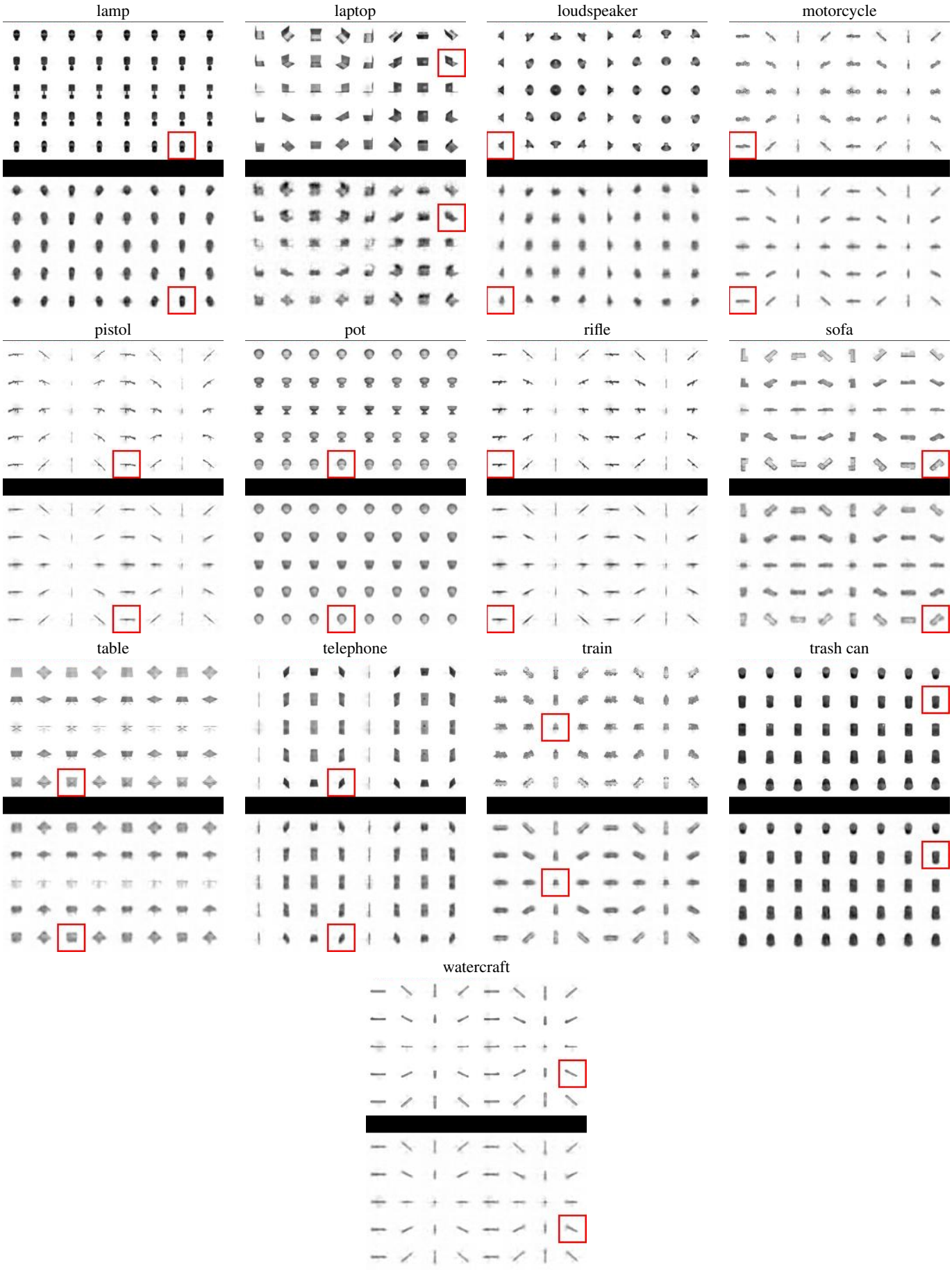

Figure 11: (continued from the previous page) Examples of randomly sampled reconstructions from various ShapeNet seen classes - part 2 of 2 (best observed in pdf at high resolution). See Sec 4.3.

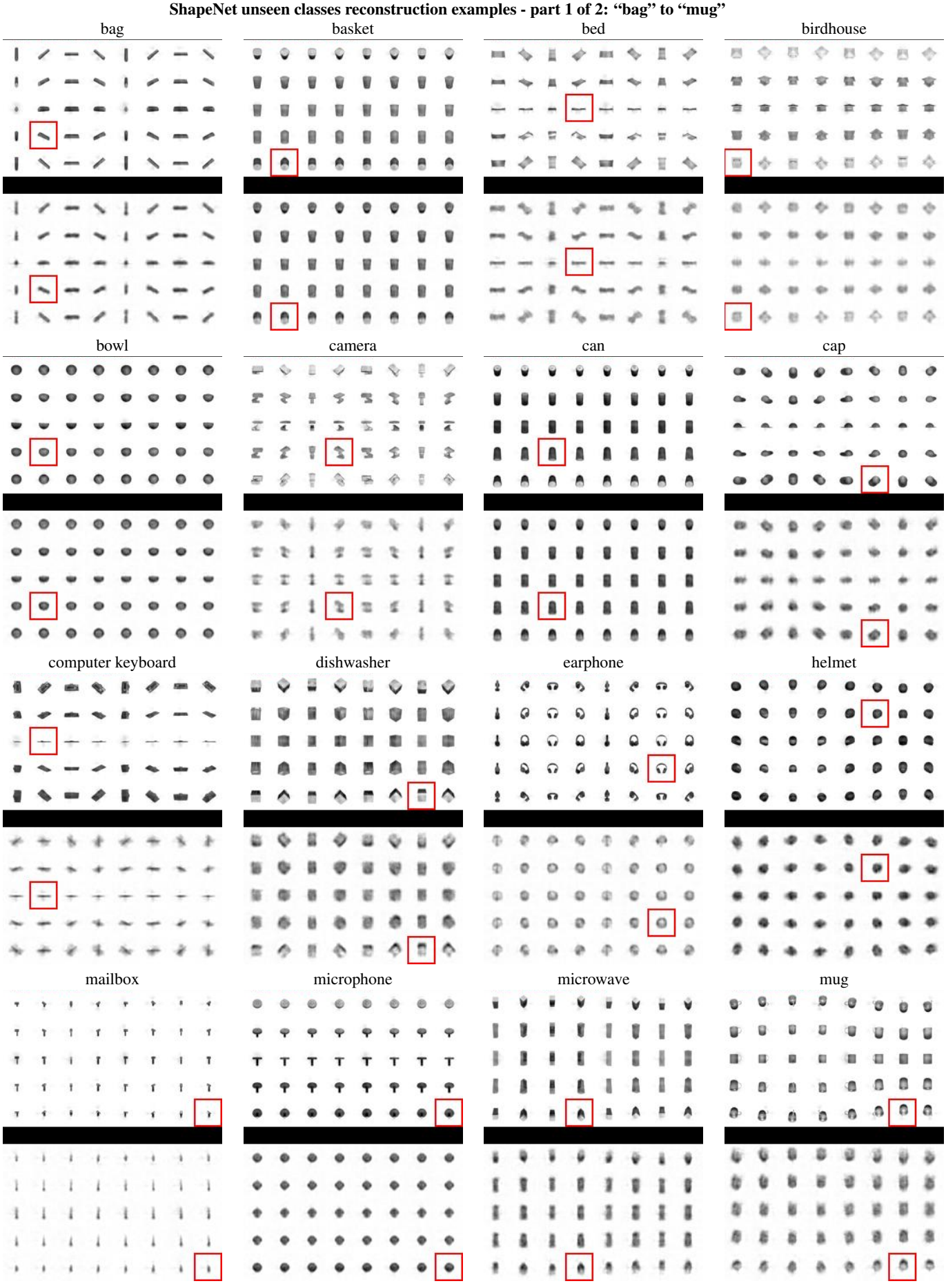

Figure 12: Examples of randomly sampled reconstructions from various ShapeNet unseen classes - part 1 of 2 (best observed in pdf at high resolution). See Sec 4.3. (continues on the next page)

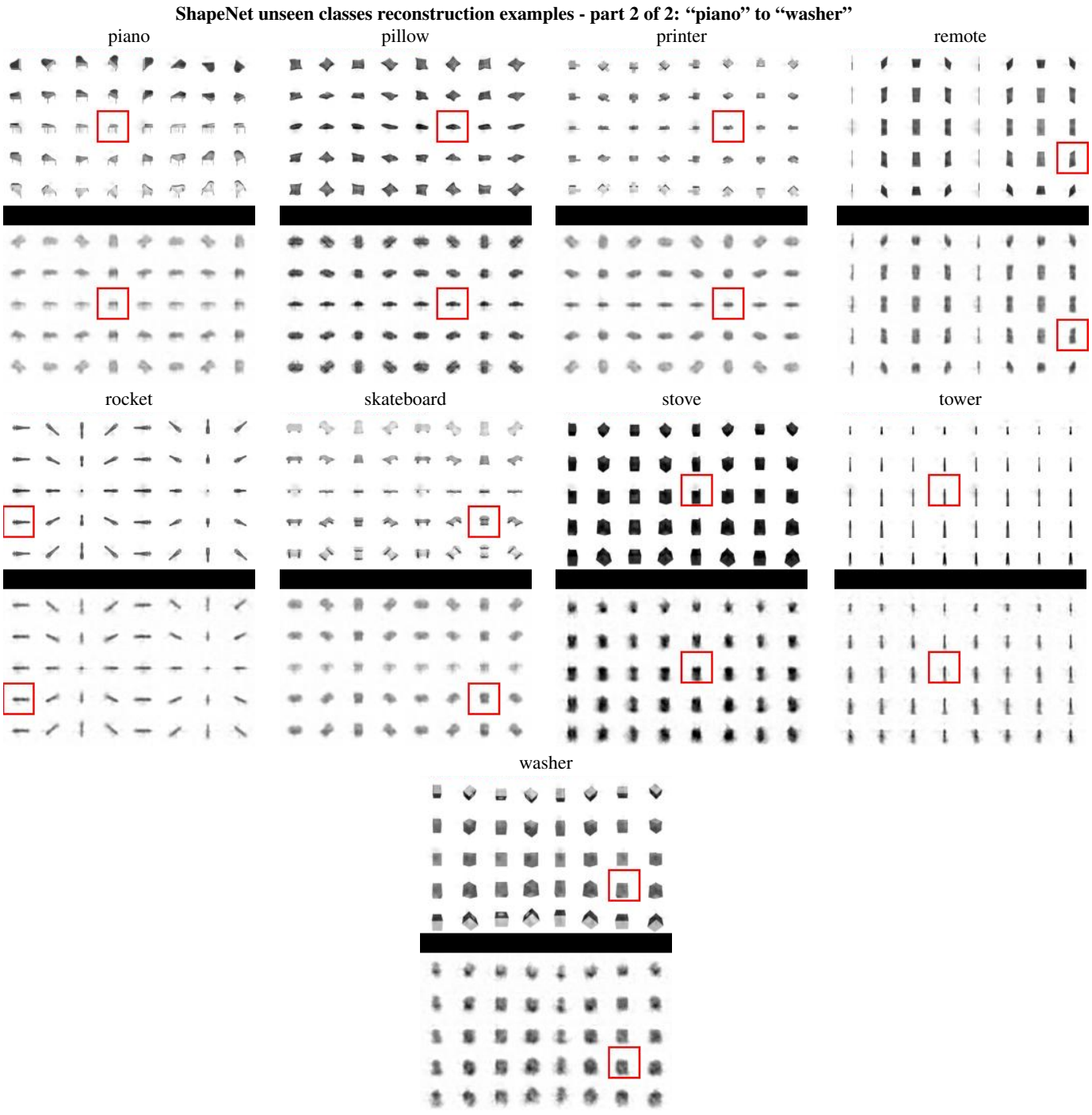

Figure 12: (continued from the previous page) Examples of randomly sampled reconstructions from various ShapeNet unseen classes - part 2 of 2 (best observed in pdf at high resolution). See Sec 4.3.