SPARE3D: A Dataset for SPAtial REasoning on Three-View Line Drawings Supplementary Material

Wenyu Han* Siyuan Xiang* Chenhui Liu Ruoyu Wang Chen Feng†
New York University Tandon School of Engineering
https://ai4ce.github.io/SPARE3D

1. SPARE3D-CSG

Why CSG models? CSG models are randomly generated from simple primitives, like sphere, cube, cone, and torus, with boolean operations including union, intersection, and difference. Therefore, it allows us to control the complexity of 3D models. In the *SPARE3D-CSG* dataset, we generate three sets of 4000 3D models, i.e., a total of 12000, from two, three, and four simple random primitives respectively. With more primitives in a model, the complexity of the model increases, and so does the difficulty level of *SPARE3D-CSG* tasks generated from those models.

When generating tasks for view consistency reasoning and camera pose reasoning, for training and testing dataset, we select the same number of 2D drawings from two, three, and four simple primitive model sets. In this way, we ensure that our baseline methods are trained and tested on tasks with the same difficulty levels.

CSG model generation. Most of the objects in the real world look reasonably regular in shape because they are usually designed and organized in certain rules manually. The *SPARE3D-CSG* dataset is generated using the following two rules. First, to create a CSG model from simple primitives, rotation angles for these primitives are randomly selected from $0°$, $90°$, $180°$, and $270°$. Second, these primitives are only rotated about X, Y, or Z axes. Example models can be seen from Figure 1.

2. Baseline Methods Formulation

We formulate the *3-View to Isometric* and *Pose to Isometric* tasks as either binary classification or metric learning. The *Pose to Isometric* task is formulated as the multi-class classification. *Isometric View Generation* is treated as conditional image generation, *Point Cloud Generation* is expressed as 3D point cloud generation from multi-view image. In this section, we use I_F, I_T, and I_R to represent line drawings from the front, top, and right view, respectively, each of which is a 3-channel RGB image. The backbone neural networks are represented as feature extraction function f for each task. The detailed formula of each task is shown in the following subsections.

2.1. Three-View to Isometric

Binary Classification. I_F, I_T, I_R, and a query image I_q from the choices are concatenated along the feature dimension, to form a 12-channel composite image I_c. Then a CNN-based binary classifier $f_\theta : \mathbb{R}^{12\times H\times W} \to [0, 1]$ is trained to map I_c to $\hat{p}(\theta)$, which is the probability that I_q is the isometric image. θ represents the parameters of the neural network. Binary cross-entropy (BCE) loss is applied to train the neural nework:

$$L(\theta) = -p \log \hat{p}(\theta) - (1 - p) \log (1 - \hat{p}(\theta)), \quad (1)$$

where $p \in \mathbb{Z}_2$ is the ground truth label of whether I_q is the isometric drawing consistent with the input.

We take four images (three images from the question and one image from answer) as a group. Therefore, each time, we have four groups of data to process. We use VGG and ResNet to encode a group of images to a feature vector in R^1 space. Then we concatenate four feature vectors and use softmax to get a 4×1 vector of distribution probability.

Metric Learning. I_F, I_T, and I_R are concatenated to form a 9-channel composite image I_c. Then I_c is fed into a CNN-based encoder $f_\theta : \mathbb{R}^{9\times H\times W} \to \mathbb{R}^M$. A query image I_q from the choices is fed into another CNN-based encoder $g_\phi : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^M$. θ and ϕ represent the parameters of the two neural networks respectively. We use l_2 distance $d(\theta, \phi) = \|f_\theta(I_c) - g_\theta(I_q)\|$ to measure the correctness of I_q. Smaller $d(\theta, \phi)$ indicates higher correctness that I_q is the isometric image among the four choices. We apply margin ranking loss to train the networks:

$$L(\theta, \phi) = \sum_{k=1}^{3} \max (0, d_c(\theta, \phi) - d_w^k(\theta, \phi) + m), \quad (2)$$

where $d_c(\theta, \phi)$ is the correctness measurement of the correct I_q, and $d_w^k(\theta, \phi)$ is the correctness measurement of the

*The first two authors contributed equally.
†Chen Feng is the corresponding author. cfeng@nyu.edu

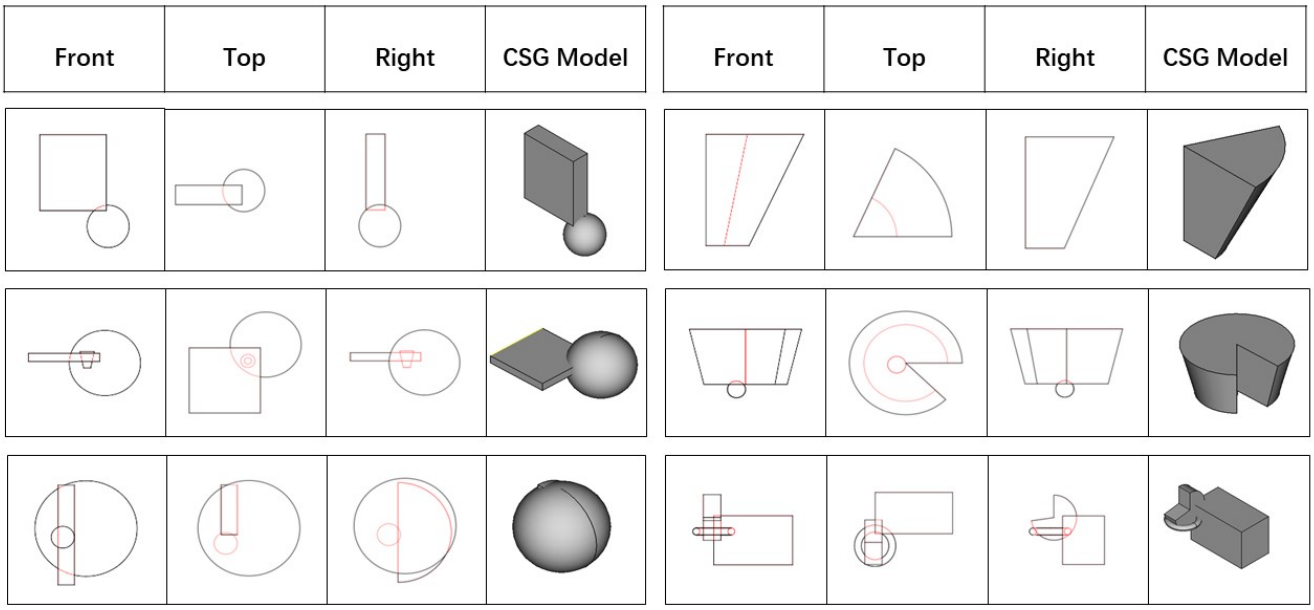

Figure 1. **CSG model examples**. In each example chunk, the first three columns are F, T, R drawings, respectively; the fourth column is the rendered CSG model (seeing from pose 2 as explained in the main paper). The models in the second row, third row, and fourth row are generated from two, three, and four simple primitives, respectively.

kth wrong I_q. $m = 2$ is the margin we use during training. We set $M = 128$ in this task.

2.2. Isometric to Pose

Multi-class Classification. I_F, I_T, I_R and the isometric image I_i are concatenated to form a 12-channel composite image I_c. Then a CNN-based classifier $f_\theta : \mathbb{R}^{12\times H\times W} \to [0,1]^4$ is trained to map I_c to a four-vector $\hat{\mathbf{p}}(\theta) = [\hat{p}_1(\theta), \hat{p}_2(\theta), \hat{p}_3(\theta), \hat{p}_4(\theta)]^T$ that represents the probability of I_i is taken at pose 1, 5, 2 and 6 respectively. Cross-entropy loss is applied to train the neural network.

$$L(\theta) = -\sum_{k=1}^{4} p_k \log \hat{p}_k(\theta), \tag{3}$$

where $p_k = 1$ if I_i is taken at the kth view point. For this task, we encode the concatenated four images in the question into a R^4 feature vector using function F. Then we use softmax to get the probability distribution and compute the cross-entropy loss between the feature vector and the encoding of the answer.

2.3. Pose to Isometric

Binary Classification. I_F, I_T, I_R and a query image I_q from the choices are concatenated to form a 12-channel composite image I_c. This composite image is fed into a CNN $f : \mathbb{R}^{12\times H\times W} \to \mathbb{R}^K$. Then, we concatenate the output with a 8-dimensional one-hot vector $z \in \mathbb{Z}_2^8$, representing the given camera pose to create a codeword $c \in \mathbb{R}^K \times \mathbb{Z}_2^8$. c is then fed into a fully-connected network $g_\phi : \mathbb{R}^K \times \mathbb{Z}_2^8 \to [0,1]$. We apply BCE loss as equation (1) to train the neural network. Here we set $K = 128$.

Metric Learning. Similar to the binary classification formulation, we again obtain $c \in \mathbb{R}^K \times \mathbb{Z}_2^8$ from I_F, I_T, I_R and z. c is then fed into a fully-connected network $g_\theta : \mathbb{R}^K \times \mathbb{Z}_2^8 \to R^M$. For each answer image, we obtain a feature vector in $\mathbb{R}^M$ space using another CNN-based encoder $h_\omega : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^M$. Then we can caculate the margin ranking loss similar to equation (2). In our experiment, $K = 128$ and $M = 50$.

2.4. Isometric View Generation

For this task, we use Pix2Pix [2], a conditional generative adversarial network, to generate the isometric drawing for each question. The generator network $G(x)$ needs to learn a mapping from the three-view drawings to the isometric drawing. The input x is a $\mathbb{R}^{9\times H\times W}$ tensor generated by concatenating F, R, T images. When training the pix2pix model on our dataset, we use label flipping and label smoothing to improve the stability of the model.

2.5. Point Cloud Generation

We use a FoldingNet [3]-like and AtlasNet [1]-like decoding architectures to generate a 3D object's point cloud with 2025 points from a latent code $c \in \mathbb{R}^{512}$, which is encoded by a ResNet-18 CNN from a 9-channel concatenated F, T, R image tensor.

3. Implementation Details of Baseline Methods

In SPARE3D-ABC, we use ResNet-50, VGG-16, and BagNet as our deep network architectures for *3-View to Isometric*, *Pose to Isometric*, and *Isometric to Pose* tasks, to extract features from given drawings. The network architecture details are explained below for each baseline method.

All the hyper-parameters in each baseline method for each task, whose drawings are generated from models in ABC dataset, are tuned using a validation set of 500 questions, although we have not searched for the optimal hyper-parameters extensively using methods like grid search.

3.1. 3-View to Isometric

Binary classification. We slightly modify the ResNet-50 base network to adapt to our tasks. The first convolutional layer has 12 input channels, 64 output channels, with kernel size $(3, 3)$, instead of the original $(7, 7)$, stride and padding $(1, 1)$, instead of the original stride$(2, 2)$ and padding $(3, 3)$. The last fully-connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^1$. Other layers are exactly the same as the original ResNet-50 network. And the above modifications are applied to all the remaining baseline methods involving ResNet-50. The learning rate is 0.00005, the batch size is 9, and the network is trained for 50 epochs.

Similarly, for the VGG-16 network, the first convolutional layer is modified in the same way as ResNet-50. The last fully-connected layer maps the feature vector from $\mathbb{R}^{4096} \rightarrow \mathbb{R}^1$. The learning rate is 0.00005, the batch size is 20, and the network is trained for 50 epochs.

For the BagNet-33 base network, the first convolutional layer has 12 input channels, 64 output channels, with kernel size 1, stride 1, padding 0. The last fully-connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^1$. The learning rate is 0.0001, the batch size is 8, and the network is trained for 49 epochs.

Metric learning. In this formulation, two functions, f and g, are implemented using two similar base networks for extracting features from drawings in questions and in answers, respectively.

ResNet-50 as the base network: For f, the first convolutional layer has 9 input channels, 64 output channels, with kernel size $(3, 3)$, stride and padding $(1, 1)$. The last fully connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^{128}$. For g, the first convolutional layer has 3 input channels, 64 output channels, with kernel size $(3, 3)$, stride and padding $(1, 1)$. The last fully connected layer is the same as f. The learning rate is 0.0001, the batch size is 4, and the network is trained for 50 epochs.

VGG-16 as the base network: For f, the first convolutional layer is the same as ResNet-50 f for metric learning in *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{4096} \rightarrow \mathbb{R}^{128}$. For g, the first convolutional layer is the same as ResNet-50 g for metric learning in *3-View to Isometric*. The last fully connected layer is the same as f in this method. The learning rate is 0.00002, the batch size is 8, and the network is trained for 50 epochs.

BagNet-33 as the base network: For f, the first convolutional layer has 9 input channels, 64 output channels, with kernel size 1, stride 1 and padding 0. The last fully-connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^{128}$. For g, the first convolutional layer has 3 input channels, 64 output channels, with kernel size 1, stride 1 and padding0. The last fully connected layer is the same as f in this method. The learning rate is 0.0001, the batch size is 4, and the network is trained for 50 epochs.

3.2. Isometric to Pose

Multi-class classification. For ResNet-50, the first convolutional layer is the same as the ResNet-50 network in binary classification for *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^4$. The learning rate is 0.00002, the batch size is 70, and the network is trained for 50 epochs.

For VGG-16, the first convolutional layer is the same as the VGG-16 network in binary classification for *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{4096} \rightarrow \mathbb{R}^4$. The learning rate is 0.00002, the batch size is 80, and the network is trained for 50 epochs.

For BagNet, the first convolutional layer is the same as the BagNet network in binary classification for *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^4$. The learning rate is 0.00002, the batch size is 30, and the network is trained for 50 epochs.

3.3. Pose to Isometric

Binary classification. For ResNet-50, the first convolutional layer is the same as the ResNet-50 network in binary classification for *3-View to Isometric*. A fully connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^{128}$. After concatenating with a one-hot encoder $\mathbb{Z}_2^8$, a fully connected layer maps the concatenated feature vector from $\mathbb{R}^{136} \rightarrow \mathbb{R}^1$. The learning rate is 0.00002, the batch size is 9, and the network is trained for 50 epochs.

For VGG-16, the first convolutional layer is the same as the VGG-16 network in binary classification for *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{4096} \rightarrow \mathbb{R}^{128}$. The last layer is the same as the ResNet-50 network in binary classification for *Pose to Isometric*. The learning rate is 0.0001, the batch size is 30, and the network is trained for 50 epochs.

For BagNet, the first convolutional layer is the same as the BagNet network in binary classification for *3-View to Isometric*. A fully connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^{128}$. The last layer is the same as the

ResNet-50 network in binary classification for *Pose to Isometric*. The learning rate is 0.00005, the batch size is 8, and the network is trained for 50 epochs.

Metric learning. Similar to the metric learning formulation for the task "*3-View to Isometric*", there are two functions f and g used to extract features from drawings in the question and the answers respectively.

For ResNet-50, the first convolutional layer of f is the same as ResNet-50 f for metric learning in *3-View to Isometric*. The fully connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^{128}$. After concatenating with a one-hot encoder $\mathbb{Z}_2^8$, a linear layer maps the concatenated feature vector from $\mathbb{R}^{136} \rightarrow \mathbb{R}^{50}$. For g, the first convolutional layer is the same as ResNet-50 g for metric learning in *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{2048} \rightarrow \mathbb{R}^{50}$. The learning rate is 0.00001, the batch size is 4, and the network is trained for 47 epochs.

For VGG-16, the first convolutional layer of f is the same as VGG-16 f for metric learning in *3-View to Isometric*. The fully connected layer maps the feature vector from $\mathbb{R}^{4096} \rightarrow \mathbb{R}^{128}$. For g, the first convolutional layer is the same as VGG-16 g for metric learning in *3-View to Isometric*. The last fully-connected layer maps the feature vector from $\mathbb{R}^{4096} \rightarrow \mathbb{R}^{50}$. Other architectures are the same as VGG-16 for metric learning in *Pose to Isometric*. The learning rate is 0.000005, the batch size is 10, and the network is trained for 42 epochs.

For BagNet-33, the first convolutional layer of f is the same as BagNet f for metric learning in *3-View to Isometric*. For g, the first convolutional layer is the same as BagNet g for metric learning in *3-View to Isometric*. Other architectures are the same as BagNet for metric learning in *Pose to Isometric*. The learning rate is 0.0001, the batch size is 4, and the network is trained for 41 epochs.

3.4. Isometric View Generation

As mentioned in the baseline method part, we use the Pix2Pix network to generate isometric drawings for each question. The first layer has 9 input channels.

3.5. Point Cloud Generation

For FoldingNet-like and AtlasNet-like architectures, the number of output points for a 3D object is 2025. The latent code is $c \in \mathbb{R}^{512}$. Other architectures are the same as in FoldingNet paper and AtlasNet paper, respectively, except that the original point cloud encoder is replaced with a ResNet-18 with 9 input channels. The network is trained for 1000 epochs.

3.6. Crowd-sourcing Website

Figure 2, 3 and 4 show our crowd-sourcing website for collecting human performance, with example questions for each tasks respectively.

References

[1] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 216–224. Ieee, 2018. 2

[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134. Ieee, 2017. 2

[3] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215. Ieee, 2018. 2

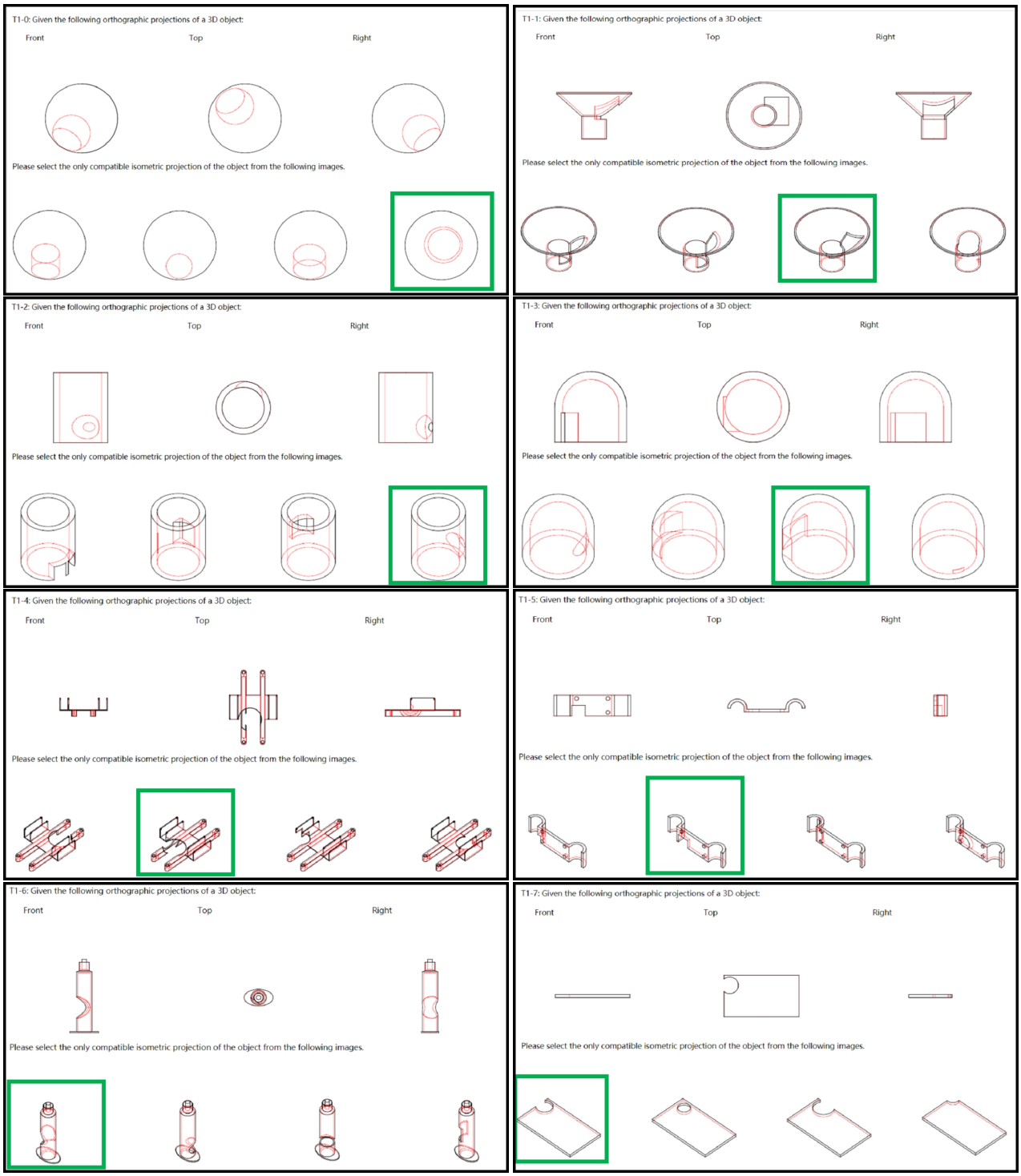

Figure 2. Examples of the "3-View to Isometric" task shown in our crowd-sourcing website. Correct answers are highlighted by green rectangles. Best view in color.

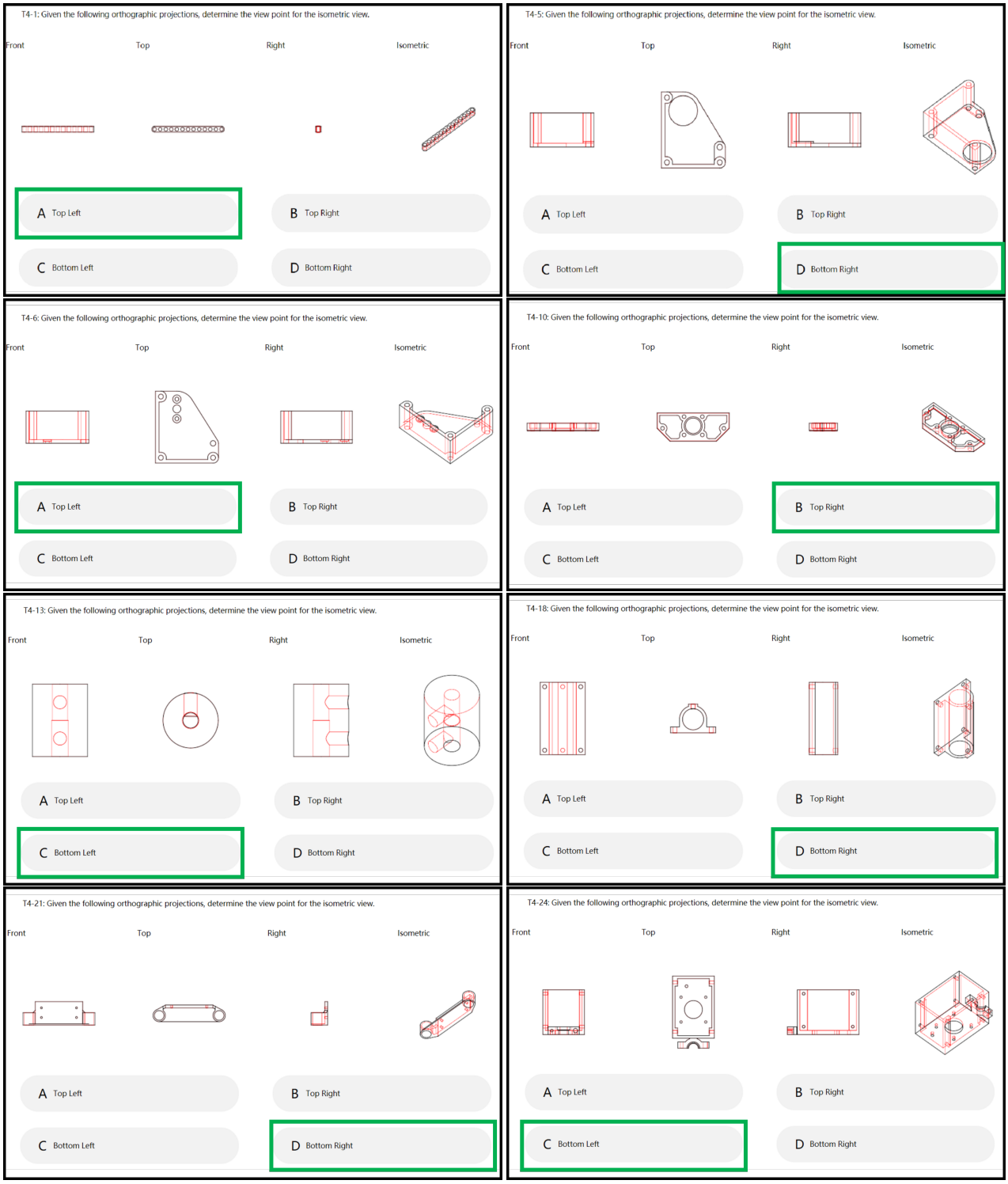

Figure 3. Examples of the "Isometric to Pose" task shown in our crowd-sourcing website. Correct answers are highlighted by green rectangles. Best view in color.

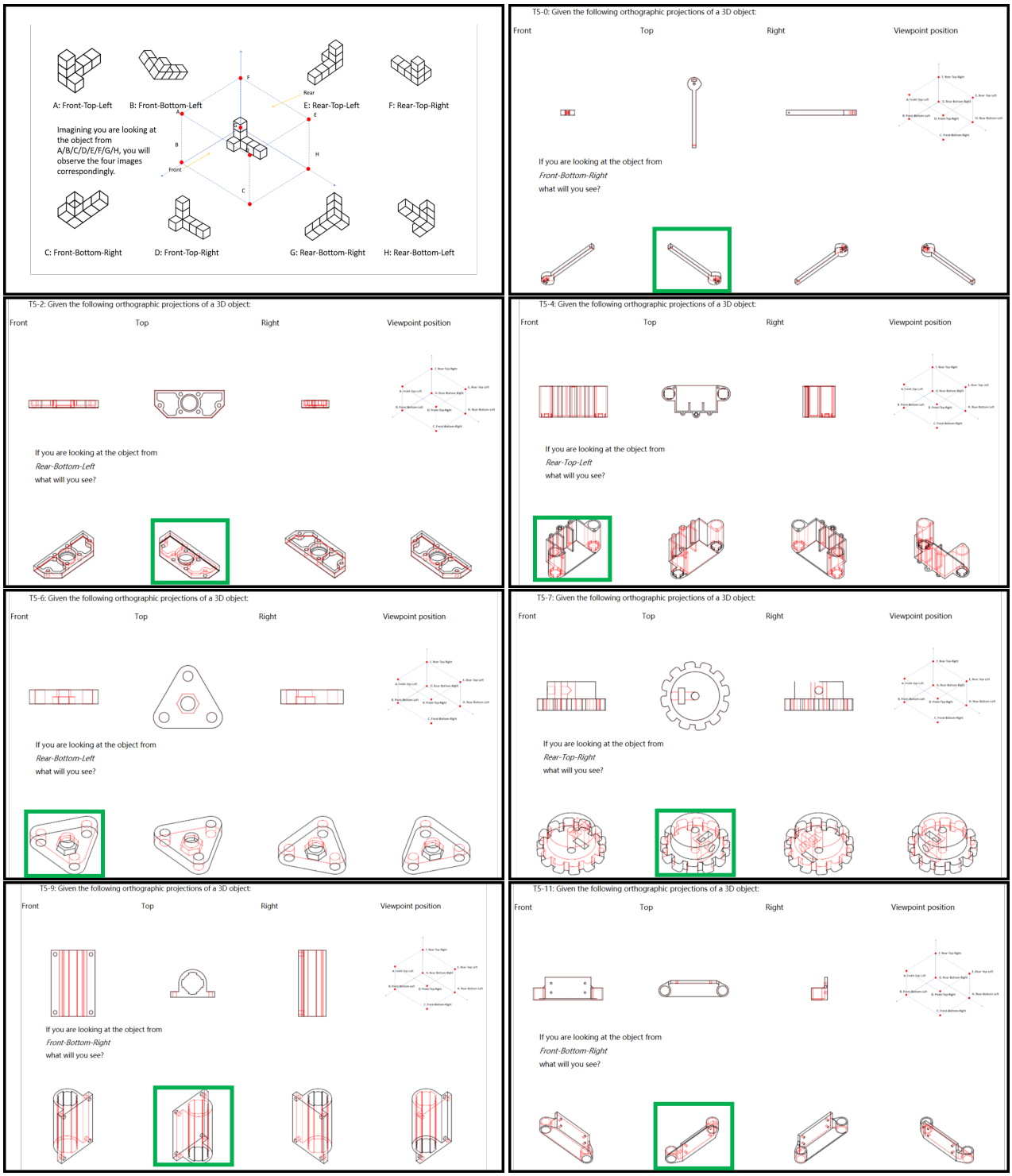

Figure 4. Examples of the "Pose to Isometric" task shown in our crowd-sourcing website. Correct answers are highlighted by green rectangles. The eight poses are explained on the left column of the first row, and also in each question. Best view in color.