Transformation Driven Visual Reasoning - Supplementary Material

Xin Hong[1,2] Yanyan Lan[3,*] Liang Pang[1,2] Jiafeng Guo[1,2] Xueqi Cheng[1,2]
[1] CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Institute for AI Industry Research, Tsinghua University, Beijing, China
{hongxin19b, pangliang, guojiafeng, cxq}@ict.ac.cn lanyanyan@tsinghua.edu.cn

This document aims at providing additional materials to supplement our main submission. We first show the detail of data balancing on TRANCE in Section A. Then, we give more details on the implementation of the baseline models and training in Section B. Next in Section C, we describe the test system we used for collecting results from humans. Finally in Section D, we provide extra examples of three settings from TRANCE, i.e. Basic, Event, and View.

A. Dataset Balance

Data balancing is an important factor to be considered when constructing TRANCE. Several factors are balanced in TRANCE, so that a learner is expected to reason the transformation without utilizing the biased features such as the length of transformation in data. Without considering the image rendering, the data generation process consists of two stages, i.e. sampling an initial scene graph and sampling a transformation sequence to transform the initial scene graph into the final scene graph. In the following of this section, we first introduce the factors that are balanced in these two stages and then describe the method we use.

When sampling the initial scene graph, the attributes of all objects and the number of visible objects are balanced. Recall that the plane is separated into the visible area and invisible area and only objects in the visible area appear in the image of the initial state. The two diagrams on the top row of Figure 1 show the statistics of these two factors. We can see that they are strictly balanced.

When sampling the transformation sequence, we balance four factors in total. The first factor is the length of transformation so that samples with different transformation lengths are equal in terms of size. The statistic result of the transformation length can be found in Figure 1 on the left side of the second row. The other three factors are all about the elements of atomic transformations including the object number for object and the move type and n-gram atomic transformation for the value. The object number is directly balanced over all samples and the result is shown in the middle of the second row in Figure 1. The move type is also strictly balanced and the statistics is shown in the right of the second row in Figure 1. As for the n-gram atomic transformation, it should be handled carefully, since for a specific initial scene graph, the availability of different atomic transformations is different. For example, changing the color of one object can always be successful, but changing the position of an object with a specific direction and step may be failed because of overlapping. Thus, the concurrence of different atomic transformations has different probability. For example, four atomic transformations on position will be less possible than four atomic transformations on color exist in one sequence. Therefore, we need to consider to balance the value throughout the sub-sequence of atomic transformations. In the following, we will call a sub-sequence with the length n as a n-gram atomic transformation. Table 1 shows the statistics of this factor. For each n-gram, the number of different options to be chosen is shown in the first row. For example, we have 33 different values so that the options of 1-gram are 33 and that of 2-gram is $33^2 = 1089$ and so on. The process of counting n-gram atomic transformations can be regarded as counting the sub-sequences using a n-length sliding window with 1 stride on transformation sequences. For example, to count 2-gram atomic transformations on a 4-step transformation, we use a 2-length sliding window with 1 stride and there will be three 2-gram atomic transformations. The rows below the options in Table 1 are calculated on the counting results. From the table, the standard variance is very small compared to the mean value, which means the concurrence of different atomic transformations is well balanced. It should be note that the size of TRANCE is 0.5 million, which is not enough to cover all 4-gram options, but the analysis of training data size in our experiments has proved our data is sufficient for training a deep model. In conclusion, the dataset is well balanced to eliminate the potential bias that are not related to the target

*Corresponding author.

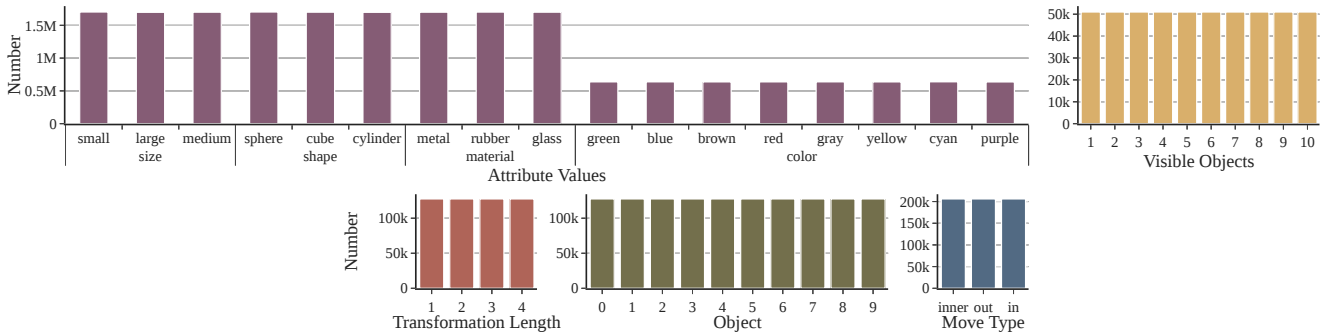


Figure 1. The statistics of balanced factors in the TRANCE dataset. **Top Row:** The attribute values and the visible objects which are balanced during sampling the initial scene graphs. **Bottom Row:** The transformation length, object number, and move type which are balanced when sampling transformation sequences.

	1-gram	2-gram	3-gram	4-gram
options	33	1,089	35,937	1,185,921
min	38,635	697	7	0
max	38,638	708	15	3
median	38,636	703	11	0
mean	38,636	702.5	10.64	0.1075
std	0.7714	2.2854	0.7880	0.3150

Table 1. The statistics of the n-gram atomic transformations in TRANCE.

Model	Encoder Backbone	Decoder	Parameters
CNN$_{-}$	4-layer CNN	Adapted GRU	737K
CNN$_{\oplus}$	4-layer CNN	Adapted GRU	738K
ResNet$_{-}$	resnet18	Adapted GRU	11M
ResNet$_{\oplus}$	resnet18	Adapted GRU	11M
BCNN	vgg11_bn	Adapted GRU	41M
DUDA	resnet18	Adapted GRU	18M

Table 2. The architectural details of different baseline models under the TranceNet framework.

Algorithm 1: Balanced Sampling

Input: available k options $O = \{o_1, o_2, ..., o_k\}$, corresponding count table $N = \{n_1, n_2, ..., n_k\}$;
Output: sampled option o_r;
Parameter: tolerance $t = 0.1$ (default);

n_{max} = max$(n_1, n_2, ..., n_k)$;
$c_i = n_{max} - n_i + t$;
$p_i = \frac{c_i}{\sum_{i=1}^{k} c_i}$;
o_r = randomly sample an option from $\{o_1, o_2, ..., o_k\}$ with probability $\{p_1, p_2, ..., p_k\}$;

of transformation reasoning.

The method we used to balance all the above factors is called balanced sampling. The basic idea of this method is changing the probability of the sampling targets dynamically according to previously generated samples. Algorithm 1 shows how to sample an option given the count table of previously generated options.

B. Implementation Details

The code for data generation is rewritten on the basis of the CLEVR[1]. In terms of training, we use PyTorch [5] as our deep learning framework. All of the code can be found at our Github repository[2]. In the following, we introduce the implementation of our baseline models and the training process in detail.

Table 2 shows the architectural details of different baseline models under the TranceNet framework. In the encoder part, both CNN$_{-}$ and CNN$_{\oplus}$ use a 4-layer CNN as the backbone. The channel of four CNN layers is 16, 32, 32, 64, the kernel size is 5, 3, 3, 3, and all the strides is 2. The encoder backbone of ResNet$_{-}$, ResNet$_{\oplus}$, and DUDA is ResNet-18 [1], which we directly use the implementation given by PyTorch without pretrained parameters. As for BCNN, we use the VGG-18 [6] implemented by PyTorch as the backbone, which is to keep consistent with the original paper [4]. In the decoder part, the output of the encoder is first flattened and then encoded by a fully-connected layer to become a 128-dimension vector. This 128-dimension vector will be sent to the adapted GRU network. In the GRU network, the hidden size of each GRU cell is 128 and two 1-layer fully-connected layers are used to decode the object vector and the value of each step respectively. The dimension of the object vector is 19 including 8 for the color, 3 for the size, 3 for the shape, 3 for the material, and 2 for the position. The dimension of the value output is 33.

The optimizer we used is Adam [2] and the learning rate

[1] https://github.com/facebookresearch/clevr-dataset-gen

[2] https://github.com/hughplay/TVR

is 0.001 in the beginning then reduced to 0.0001 after 25 epochs. The samples for Event and View is shared and the number of samples in the training, validation, and test set is 500,000, 10,000, and 20,000 respectively. Please note for each sample in View, we have 3 different final views so that the total image pairs are tripled. As for Basic, we collect all existing 1-step samples in data, and the size of training, validation, and test set is 125000, 2,500, and 5,000. All models are trained with 50 epochs on the training set and models that have the best results on the validation set are chosen to be evaluated on the test set to get the final results. In our experiments, images are resized to 120×160 for fast training. Furthermore, by following the common practice on image augmentation [3], we apply a $0 \sim 5\%$ spatial translation to all input image pairs during training.

During evaluation, when moving an object from the visible area into the invisible area, any directions and steps that could cause the same effect without making objects overlapping are accepted. In the evaluation system, this is implemented by only comparing the visible objects' attribute values of the two final states, i.e. the predicting final state and the ground truth final state.

C. Human Test System

To collect the human results, we build a web-based test system. Figure 2 shows the GUI of this system. The whole testing process includes the following steps. First of all, a human tester is told to be familiar with our system by trying a few examples with guidance. After that, the tester can start to test. During testing, for each sample, the tester should first observe given images and the attributes of the initial objects and then select the correct atomic transformations arranged with a feasible order. To reduce the time usage, we also provide the visualization of the initial objects for testers. After completing all samples, the tester can see his or her test result by checking the testing history.

D. More Examples from TRANCE

The remaining pages show extra examples from the three settings of TRANCE, i.e. Basic, Event, and View. In each sample, the initial state, the final state, and the attributes of the initial objects are provided. In the View setting, the view of the final state is randomly selected from Left and Right. To make readers easy to understand the given examples, for each example, an additional diagram is provided to visualize the attributes of the initial objects. At last, we show the reference transformation.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 2

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 3

[4] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear CNN models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015. 2

[5] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 2

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

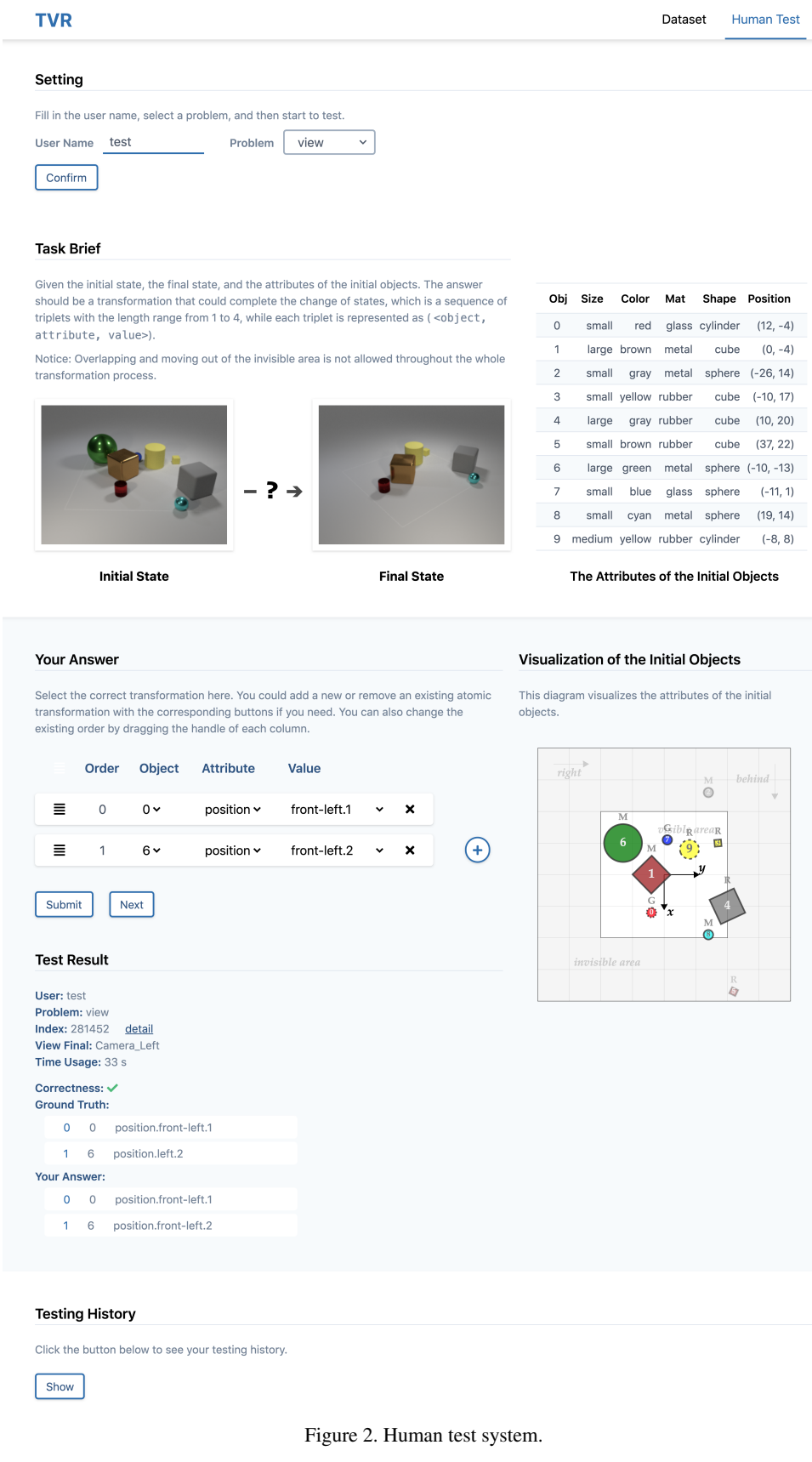
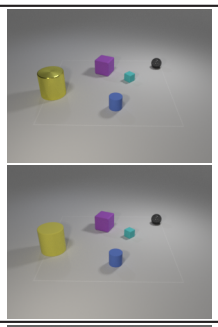


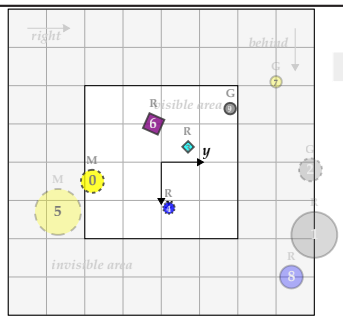
Figure 2. Human test system.

	Initial State (Top) Final State (Bottom)	Attributes of the Initial Objects	Visiualization of the Initial Attributes	Reference Transformation

1

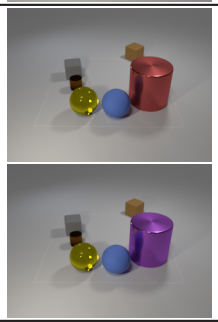


obj	size	color	material	shape	position
0	medium	yellow	metal	cylinder	5,-18
1	large	gray	rubber	sphere	19,40
2	medium	gray	glass	cylinder	2,39
3	small	cyan	rubber	cube	-4,7
4	small	blue	rubber	cylinder	12,2
5	large	yellow	metal	cylinder	13,-27
6	medium	purple	rubber	cube	-10,-2
7	small	yellow	glass	sphere	-21,30
8	medium	blue	rubber	sphere	30,34
9	small	gray	glass	sphere	-14,18

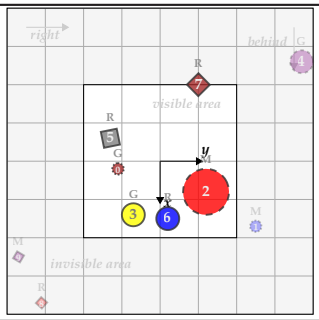


obj	attribute	value
0	material	rubber

2

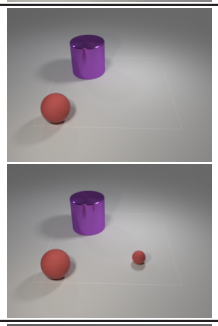


obj	size	color	material	shape	position
0	small	brown	glass	cylinder	2,-11
1	small	blue	metal	cylinder	17,25
2	large	red	metal	cylinder	8,12
3	medium	yellow	glass	sphere	14,-7
4	medium	purple	glass	cylinder	-26,37
5	medium	gray	rubber	cube	-6,-13
6	medium	blue	rubber	sphere	15,2
7	medium	brown	rubber	cube	-20,10
8	small	red	rubber	cube	37,-31
9	small	purple	metal	cube	25,-37

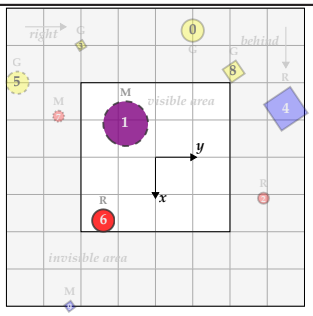


obj	attribute	value
0	color	purple

3

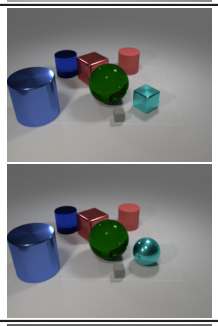


obj	size	color	material	shape	position
0	medium	yellow	glass	sphere	-34,10
1	large	purple	metal	cylinder	-9,-8
2	small	red	rubber	sphere	11,29
3	small	yellow	glass	cube	-30,-20
4	large	blue	rubber	cube	-13,35
5	medium	yellow	glass	cylinder	-20,-37
6	medium	red	rubber	sphere	17,-14
7	small	red	metal	cylinder	-11,-26
8	medium	yellow	glass	cube	-23,21
9	small	blue	metal	cube	40,-23

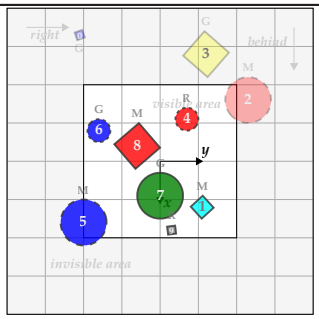


obj	attribute	value
0	position	left,2

4

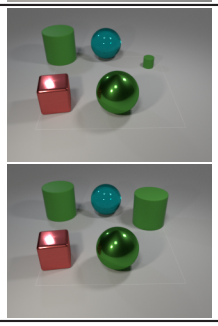


obj	size	color	material	shape	position
0	small	blue	glass	cube	-33,-21
1	medium	cyan	metal	cube	12,11
2	large	red	metal	cylinder	-16,23
3	large	yellow	glass	cube	-28,12
4	medium	red	rubber	cylinder	-11,7
5	large	blue	metal	cylinder	16,-20
6	medium	blue	glass	cylinder	-8,-16
7	large	green	glass	sphere	9,0
8	large	red	metal	cube	-4,-6
9	small	gray	rubber	cube	18,3

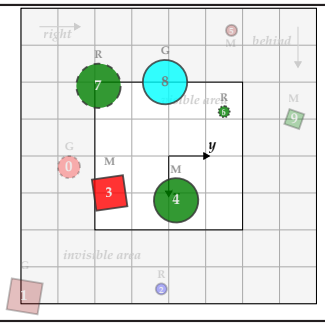


obj	attribute	value
0	shape	sphere

5



obj	size	color	material	shape	position
0	medium	red	glass	cylinder	3,-27
1	large	brown	glass	cube	38,-39
2	small	blue	rubber	sphere	36,-2
3	large	red	metal	cube	10,-16
4	large	green	metal	sphere	12,2
5	small	brown	metal	sphere	-34,17
6	small	green	rubber	cylinder	-12,15
7	large	green	rubber	cylinder	-19,-19
8	large	cyan	glass	sphere	-20,-1
9	medium	green	metal	cube	-10,34

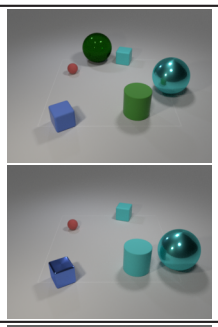


obj	attribute	value
0	size	large

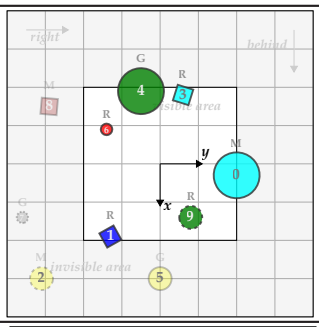
Figure 3. Examples from the Basic setting.

	Initial State (Top) Final State (Bottom)	Attributes of the Initial Objects	Visiualization of the Initial Attributes	Reference Transformation

1

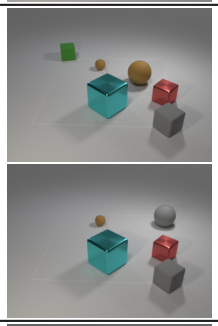


obj	size	color	material	shape	position
0	large	cyan	metal	sphere	3,20
1	medium	blue	rubber	cube	19,-13
2	medium	yellow	metal	cylinder	30,-31
3	medium	cyan	rubber	cube	-18,6
4	large	green	glass	sphere	-19,-5
5	medium	yellow	glass	sphere	30,0
6	small	red	rubber	sphere	-9,-14
7	small	gray	glass	cylinder	14,-36
8	medium	brown	metal	cube	-15,-29
9	medium	green	rubber	cylinder	14,8

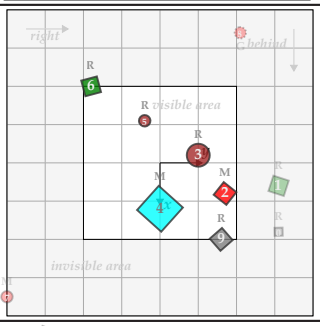


	obj	attribute	value
0	0	position	behind,1
1	1	material	metal
2	9	color	cyan
3	4	position	front,2

2

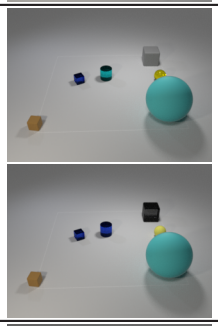


obj	size	color	material	shape	position
0	small	gray	rubber	cube	18,31
1	medium	green	rubber	cube	6,31
2	medium	red	metal	cube	8,17
3	medium	brown	rubber	sphere	-2,10
4	large	cyan	metal	cube	12,0
5	small	brown	rubber	sphere	-11,-4
6	medium	green	rubber	cube	-20,-18
7	small	red	metal	sphere	35,-40
8	small	red	glass	cylinder	-34,21
9	medium	gray	rubber	cube	20,16

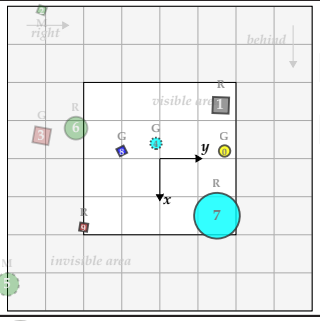


	obj	attribute	value
0	3	position	front,1
1	3	color	gray
2	6	position	left,2
3	3	position	right,1

3

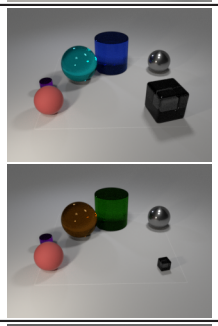


obj	size	color	material	shape	position
0	small	yellow	glass	sphere	-2,17
1	medium	gray	rubber	cube	-14,16
2	small	green	metal	cube	-39,-31
3	medium	brown	glass	cube	-6,-31
4	small	cyan	glass	cylinder	-4,-1
5	medium	green	metal	cylinder	33,-40
6	medium	green	rubber	sphere	-8,-22
7	large	cyan	rubber	sphere	15,15
8	small	blue	glass	cube	-2,-10
9	small	brown	rubber	cube	18,-20

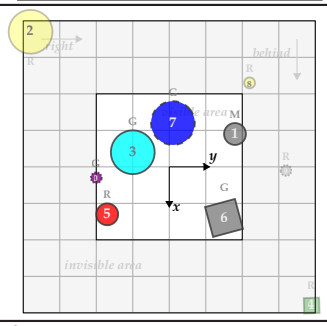


	obj	attribute	value
0	0	material	rubber
1	4	color	blue
2	1	material	glass

4

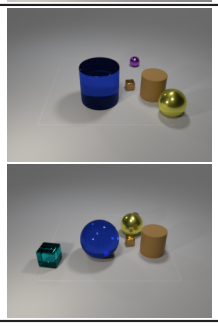


obj	size	color	material	shape	position
0	small	purple	glass	cylinder	3,-20
1	medium	gray	metal	sphere	-9,18
2	large	yellow	rubber	sphere	-37,-38
3	large	cyan	glass	sphere	-4,-10
4	medium	green	rubber	cube	38,39
5	medium	red	rubber	sphere	13,-17
6	large	gray	glass	cube	14,15
7	large	blue	glass	cylinder	-12,1
8	small	yellow	rubber	sphere	-23,22
9	small	gray	rubber	cylinder	1,32

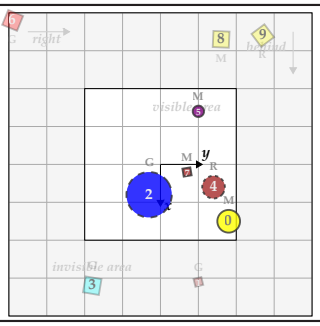


	obj	attribute	value
0	6	size	small
1	3	color	brown
2	7	color	green

5



obj	size	color	material	shape	position
0	medium	yellow	metal	sphere	15,18
1	small	brown	glass	cube	31,10
2	large	blue	glass	cylinder	8,-3
3	medium	cyan	glass	cube	32,-18
4	medium	brown	rubber	cylinder	6,14
5	small	purple	metal	sphere	-14,10
6	medium	red	glass	cube	-38,-39
7	small	brown	metal	cube	2,7
8	medium	yellow	metal	cube	-33,16
9	medium	yellow	rubber	cube	-34,27

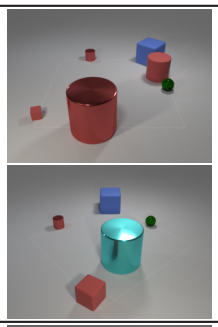


	obj	attribute	value
0	3	position	front,2
1	0	position	right,1
2	0	position	front-left,2
3	2	shape	sphere

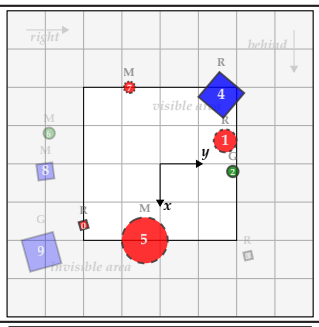
Figure 4. Examples from the Event setting.

	Initial State (Top) Final State (Bottom)	Attributes of the Initial Objects	Visiualization of the Initial Attributes	Reference Transformation

1

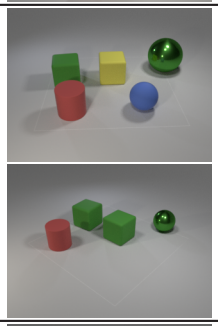


obj	size	color	material	shape	position
0	small	red	rubber	cube	16,-20
1	medium	red	rubber	cylinder	-6,17
2	small	green	glass	sphere	2,19
3	small	gray	rubber	cube	24,23
4	large	blue	rubber	cube	-18,16
5	large	red	metal	cylinder	20,-4
6	small	green	metal	sphere	-8,-29
7	small	red	metal	cylinder	-20,-8
8	medium	blue	metal	cube	2,-30
9	large	blue	glass	cube	23,-31

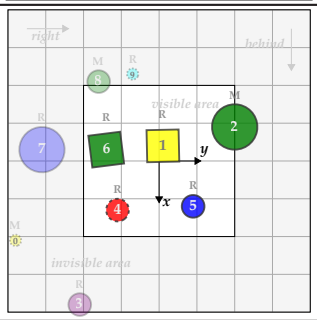


	obj	attribute	value
0	5	position	front,1
1	5	color	cyan
2	0	size	medium
3	1	position	right,2

2

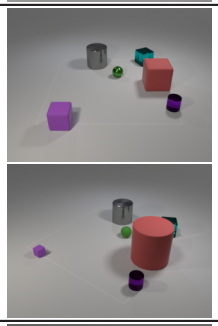


obj	size	color	material	shape	position
0	small	yellow	metal	cylinder	21,-38
1	large	yellow	rubber	cube	-4,1
2	large	green	metal	sphere	-9,20
3	medium	purple	rubber	sphere	38,-21
4	medium	red	rubber	cylinder	13,-11
5	medium	blue	rubber	sphere	12,9
6	large	green	rubber	cube	-3,-14
7	large	blue	rubber	sphere	-3,-31
8	medium	green	metal	sphere	-21,-16
9	small	cyan	rubber	cylinder	-23,-7

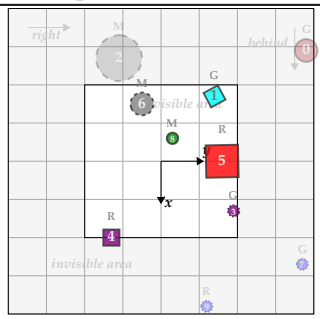


	obj	attribute	value
0	5	position	behind-left,2
1	1	color	green
2	2	size	medium
3	2	position	front-left,1

3

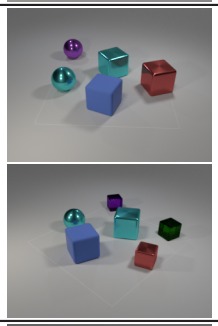


obj	size	color	material	shape	position
0	medium	brown	glass	sphere	-29,38
1	medium	cyan	glass	cube	-17,14
2	large	gray	metal	cylinder	-27,-11
3	small	purple	glass	cylinder	13,19
4	medium	purple	rubber	cube	20,-13
5	large	red	rubber	cube	0,16
6	medium	gray	metal	cylinder	-15,-5
7	small	blue	glass	cylinder	27,37
8	small	green	metal	sphere	-6,3
9	small	blue	rubber	cylinder	38,12

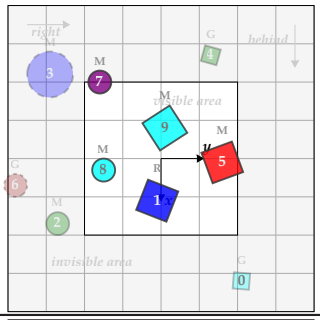


	obj	attribute	value
0	8	material	rubber
1	4	size	small
2	5	shape	cylinder

4

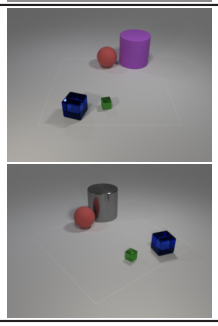


obj	size	color	material	shape	position
0	medium	cyan	glass	cube	32,21
1	large	blue	rubber	cube	11,-1
2	medium	green	metal	sphere	17,-27
3	large	blue	metal	cylinder	-22,-29
4	medium	green	glass	cube	-27,13
5	large	red	metal	cube	1,16
6	medium	brown	glass	cylinder	7,-38
7	medium	purple	metal	sphere	-20,-16
8	medium	cyan	metal	sphere	3,-15
9	large	cyan	metal	cube	-8,1

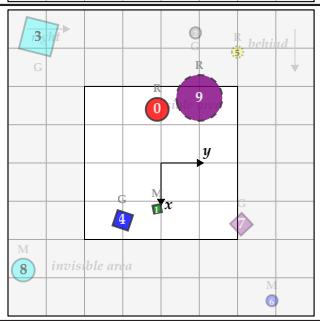


	obj	attribute	value
0	4	position	behind,1
1	5	size	medium
2	7	shape	cube
3	7	material	glass

5



obj	size	color	material	shape	position
0	medium	red	rubber	sphere	-14,-1
1	small	green	metal	cube	12,-1
2	small	gray	glass	sphere	-34,9
3	large	cyan	glass	cube	-33,-32
4	medium	blue	glass	cube	15,-10
5	small	yellow	rubber	cylinder	-29,20
6	small	blue	metal	sphere	36,29
7	medium	purple	glass	cube	16,21
8	medium	cyan	metal	sphere	28,-36
9	large	purple	rubber	cylinder	-17,10



	obj	attribute	value
0	4	position	right,2
1	9	material	metal
2	9	color	gray

Figure 5. Examples from the View setting.