A. Dataset

A.1. Test data

For the VoE task, we divided the four scenarios into 11 groups, each with two comparison cases. The setups in the testing data are very similar to the ones in the training data except for the behavior of the wall. All scenarios except Permanence contain predictive, hypothetical, and explicative settings. The predictive and explicative settings contain both plausible and implausible events, while the hypothetical setting contains two plausible events. In the predictive setting, the wall is moved away at the beginning and end of the video, so all information is shown at the beginning and end of the video. In the hypothetical setting, the wall always stays in the middle of the scene. In the explicative setting, the wall is moved away only at the end of the video, so new information is shown to the model at the end of the video.

Collision The Collision scenario is shown in Fig. A1. Collision contains predictive, hypothetical, and explicative settings. In the predictive setting, the wall is moved away at the beginning and end of the video, so two balls are visible to the model. We can easily tell from intuitive physics that the case in the first row is possible while the case in the second row is not, because the red ball cannot pass through the blue ball without collision. In the hypothetical setting, the wall always stays in the middle of the scene, so we can not tell how many balls there are in the scene. As we can not infer if a blue ball is hidden behind the wall at the beginning of the video, both cases in the setting are possible. In the explicative setting, the wall is moved away at the end of the video, so additional information is given. We can infer that a blue ball must be hidden behind the wall, so the case in the first row is possible, while the case in the second row is not.

Blocking The Blocking scenario is shown in Fig. A2. The Blocking scenarios are similar to the Collision scenarios, except that the ball hidden behind the wall is replaced by a fixed cube. In the predictive setting, the wall is moved away at the beginning and end of the video, so the cube is visible to the model. Similar to Collision, we can easily tell that the case in the first row is possible while the case in the second row is not, because the blue ball can not pass through the green cube without collision. In the hypothetical setting, the wall always stays in the middle of the scene, so we can not tell if there is a cube behind the wall. Therefore, both cases in the setting are possible. In the explicative setting, the wall is moved away at the end of the video, so we can infer that a cube must be hidden behind the wall. Furthermore, we can tell that the case in the first row is possible while the case in the second row is not.

Permanence The Permanence scenario is shown in Fig. A3. In the Permanence scenarios, three cubes are randomly divided into two groups (allowing empty groups), where cubes in the first group are dropped to the ground and the second rest on the floor. We do not have an explicative setting for this scenario, as there is no new evidence at the end of the video. In the predictive setting, the wall is moved away at the beginning of the video, so we can infer that there is no object on the ground at the beginning. So the case in the second row is impossible, while the case in the first row is possible. In the hypothetical setting, the wall stays in the middle of the scene at the beginning, so we can not tell if there are cubes on the ground at the beginning, so both cases are possible.

Continuity The Continuity scenario is shown in Fig. A4. In the Continuity scenarios, we create a window on the lower half of the wall. In the case of the wall, the ball rolls across the scene. When the ball passes through the wall, it can be seen going from one side to the other. In the predictive setting, the wall is moved away at the beginning of the video, so we can infer that only one ball is in the scene. We can tell that the case in the second row is impossible while the case in the first row is possible. In the hypothetical setting, the wall always stays in the middle of the scene, and we can easily infer that the case in the first row is possible. Considering the case in the second row, we can not tell if there are two balls with the same appearance in the scene, one of which is visible at the beginning and the other one is hidden by the right part of the wall. If that is true, the case in the second row is also possible. So both cases are possible. In the explicative setting, the wall is moved away at the end of the video, so we can infer that there is only one ball in the scene. Thus we can tell that the case in the first row is possible while the case in the second row is not.

A.2. Train data

For four scenarios, we created 5 groups for training. Each of Permanence and Continuity contains 1 group, while Collision and Blocking in total contain 3 groups. Each group contains 2 kinds of cases: cases with a wall and ones without a wall. In the case with a wall, a movable wall stands in the middle of the scene and will be moved away at the beginning and the end of the video. In the case without the wall, everything stays the same except that the wall does not exist, showing that the wall won't interact with other objects physically. Each row in the Fig. 4 corresponds to one sampled video in a specific case. See Fig. 4 for all training groups.

Control group In the control group, a ball rolls across the scene without interacting with other objects, indicating that the environment follows basic physics.

Collision group A ball rolls across the scene in the Collision scenario with the wall. Another ball with the same mass but a different color is hidden behind the wall and will collide with the incoming ball, causing the first ball to stop and itself to pass through. In a setting without a wall, the second ball will always be visible.

Blocking group The Blocking scenarios are similar to the Collision scenario, except that the ball hidden behind the wall is replaced by a fixed cube. A ball rolls across the scene in the blocking setting with the wall. A fixed cube is hidden behind the wall and will collide with the incoming ball, causing the incoming ball to turn around. In the setting without a wall, everything stays the same except that the wall doesn't exist, and the cube will always be visible.

Permanence group In the Permanence scenario, three cubes are randomly divided into two groups (allowing empty groups), where cubes in the first group are dropped to the ground and the second rest on the floor. In the setting with the wall, the wall will be moved away at the end of the video, showing that all of the cubes still exist. In the setting without the wall, the cubes will always be visible.

Continuity group In the Continuity scenario, we create a window on the lower half of the wall. In the setting with the wall, the ball rolls across the scene. When the ball passes through the wall, it can be seen going from one side to the other, especially visible from the window. In the setting without the wall, the ball will always be visible.

A.3. Environment

Our X-VoE dataset comprises 22K+100K procedurally generated scenes using Unreal Engine 4. In addition to the floors and the backgrounds, there are four different object types: balls, cubes, walls, and windowed walls. In all videos, the size of the ball and the cube are the same, while the size of the wall with or without windows are randomly different. The positions of objects are randomly set in the videos, except for the walls in the permanent scenes in which the wall is placed in the middle. All objects, including the floor and the background, are randomly set in different colors.

B. Model

B.1. Perception

The perception module in XPL is similar to that of Component Variational Autoencoder (ComponentVAE) in the PLATO model [30]. For each object k in an image, we take as input a 128 × 128 RGBD (0-255 for each channel) image x_k that is masked except around the object. Then we use a Vision Transformer [14] encoder Φ to encode the image with only one object into a 32-dimensional Gaussian posterior distribution $q_\Phi(z_k|x_k)$. The sample from this distribution, z_k, is decoded by a spatial broadcast decoder [41] to an RGBD image. To address occlusion, we use the depth of the decoder image to combine all objects in the image by multiplying them with softmaxed depth values. We first pretrained the perception module by optimizing the variational objective defined in [7]. We set σ to 0.05, β to 0.5, and γ to 0 to ensure that the model reconstructs object masks without segmentation information in the loss function.

ViT encoder We first reshape the 128 × 128 × 4 images into a sequence of flattened 16 × 16 × 256 patches, followed by a linear layer with 256 dimensions. Next, we add 2D position embeddings and learnable embeddings, flatten, and send them to a Transformer. We use 8 multi-head, 32 key dimensions, 1024 MLP layer dimensions, and 6 Transformer layers for the Transformer model [39]. Finally, we use an MLP layer with size [512, 64] and a leaky-ReLU activation function to the Transformer output and obtain 32-dimensional Gaussian posterior distributions for each object.

Spatial broadcast decoder Our spatial broadcast decoder is similar to that in [26]. As shown in Tab. A1, we use position embeddings and CNN model to decode the object embeddings, where the parameter θ in the softmax layer is learnable, thus representing the mask in terms of depth.

B.2. Reasoning

In the reasoning module, we use two Transformer modules to reason the hidden object which is occluded in some or all of the frames. All objects in a video can be reshaped as F × N × D embeddings, where F is 15 frames, N is 8 objects, and D is 32 dimensions in our work. As shown in Tab. A2, we use a Transformer model to reason the masked objects in video, similar to the self-supervised learning module in Aloe [12]; the parameter [M] in the Mask (1) part is learnable.

First Transformer We set the mask to 0 for objects that are temporally occluded in some frames, and 1 for others. As shown in Tab. A2, we can use the Transformer model to reason the new object embeddings whose mask equals 0. We use it in both the training and testing steps to have better object embedding for the whole video.

Second Transformer In our test dataset, there may be cases where an object is obscured in all frames. So in the training step, we set the mask to 0 for one random object (including empty object) in all frames. Then we can train the second Transformer model in a self-supervised manner. In the test step, we set the mask to 0 for one object that is not visible in all frames. Then we can reason about the occluded object to explain the whole video.

B.3. Dynamics

In fact, the occluded objects are never directly seen for the Transformer model. After the first reasoning module, we obtain reasonable video object embeddings based on experience. In the dynamics module, we predict the value of the incremental change of the object embeddings in the time step by using the same dynamics module from PLATO [30] with the only difference in object dimension used (from 16 to 32). We refer the readers to [30] for architectural details.

Table A1: Spatial broadcast decoder architecture (from top to down).

Type	Size	Activation	Comment
Spatial Broadcast	8×8	-	-
Position Embedding	-	-	-
Conv 5×5	64	ReLU	stride: 2
Conv 5×5	64	ReLU	stride: 2
Conv 5×5	64	ReLU	stride: 2
Conv 5×5	64	ReLU	stride: 2
Conv 5×5	64	ReLU	stride: 1
Conv 3×3	4	-	stride: 1
Channels	RGBD(4)	Softmax (on depth channel)	softmax(depth $\times$ abs(θ) $\times$ -1000.0)

Table A2: The Transformer architecture (from top to down). The [M] is a learnable mask token for Transformer.

Type	Size	Activation	Comment
LP (1)	256	-	-
Mask (1)	-	$\times$ mask + [M] $\times$ (1-mask)	mask : (size F $\times$ N $\times$ 1), (value 0 or 1)
Position Embedding	-	-	-
Transformer	256, 256 (MLP)	ReLU (MLP)	head=8,key=32,layers=6
LP (2)	256	-	-
Mask (2)	-	$\times$ (1-mask) + inputs $\times$ mask	mask : (size F $\times$ N $\times$ 1), (value 0 or 1)

Table A3: Training parameters. The pre-processed video features are calculated by the Perception module, which is pre-trained.

Model	batch size	training step	optimizer	learning rate	warm step	delay step
Perception module (in XPL,PLATO)	300 (images)	472000	Adam	0.0004	2000	100000
XPL	500 (pre-processed video features)	32000	Adam	0.0004	1000	10000
PLATO	500 (pre-processed video features)	32000	Adam	0.0004	1000	10000
PhyDNet	100 (videos)	70000	Adam	0.001	-	-

C. Training

C.1. Training detail

In a scene with occlusion, we cannot get the representation of the occluded object directly by observation. Therefore, we first use the dynamics loss on the object embeddings after the first Transformer to train our first Transformer and dynamics model. Then, we use the object embeddings after the first Transformer to train our second Transformer model. We randomly mask an object throughout the video frame and use the model to predict representations of the objects throughout the video, enabling the model to infer whether there is a fully hidden object in the test task.

C.2. Training parameters

We first pre-train the perception module and use it for both PLATO and XPL. Then we train our model XPL, PLATO, and PhyDNet with the parameters shown in Tab. A3.

C.3. Training steps

During the development of the model, we explored how the size of the training dataset impacted the pixel loss of the dynamics module. We use the expected video in the predictive setting of all scenarios as the test dataset to calculate the average pixel loss. Fig. A5 shows that more training data will improve the performance of the dynamics module.

D. Visualize supplementary

In the main text, we visualize the reasoning results by our XPL model in the Blocking scenario. Here, we visualize the reasoning results for the rest of the scenarios.

D.1. Collision

As shown in Fig. A6, in the predictive setting, XPL has no problem accurately reconstructing the objects, and the surprise video can be found directly. In the hypothetical setting, the possible explanation for the first video is that another ball collides with the incoming ball. In contrast, no such ball is in the second video, explaining both cases. This result also shows the limitation of our XPL as the incoming ball did not stop behind the wall. In the explicative setting, the occluder is only moved away at the end of the videos. Unlike the hypothetical, when showing a hidden ball behind it, it is impossible for the ball to pass through, causing surprise.

D.2. Permanence

As shown in Fig. A7, in the predictive setting, XPL can reconstruct the objects behind the wall, and the surprise

video can be found by comparing it with the origin image. The visual effect of the reconstructed objects does not seem to be very well, which is still a limitation of our XPL. In the hypothetical setting, the possible explanation for the second video is that there exists another object behind the wall, and our XPL can reason about the object.

D.3. Continuity

As shown in Fig. A8, the visualization results of our XPL are the same in all settings. Even though the visualization results can show surprise in predictive and explicative settings by comparing with the origin videos, our XPL still can not deal with the hypothetical setting due to the limitation discussed in the main text. Our XPL requires given masks and identification of objects. Therefore, it can not reason about the hypothetical setting in continuity by changing the identification of objects and suggesting that there are two same objects as infants do [1].

Figure A1: Collision test groups.

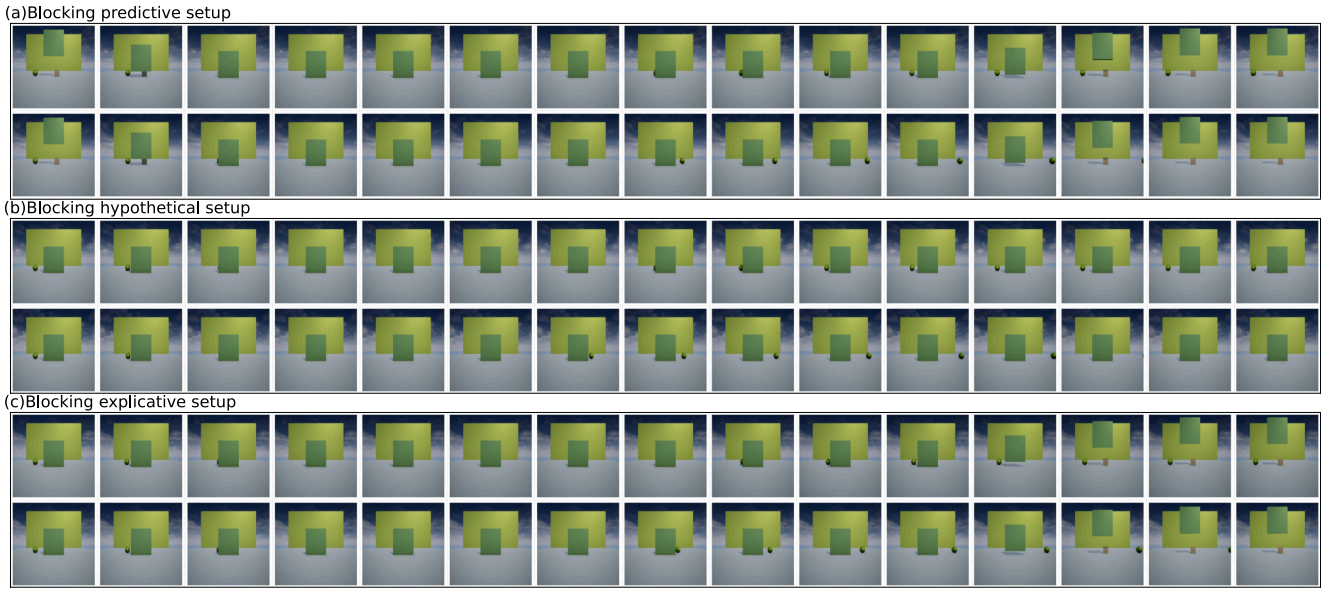

Figure A2: Blocking test groups.

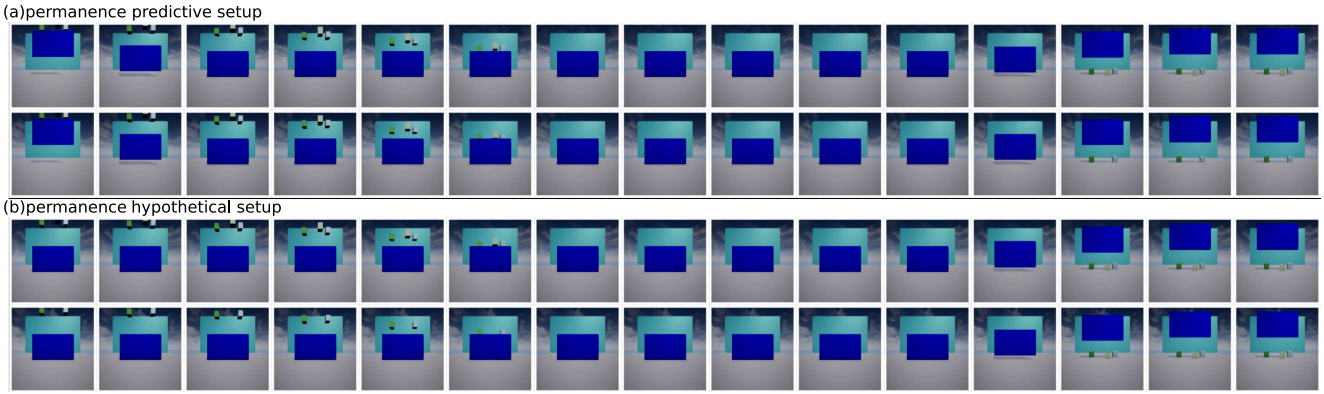

Figure A3: Permanence test groups.

Figure A4: Continuity test groups.

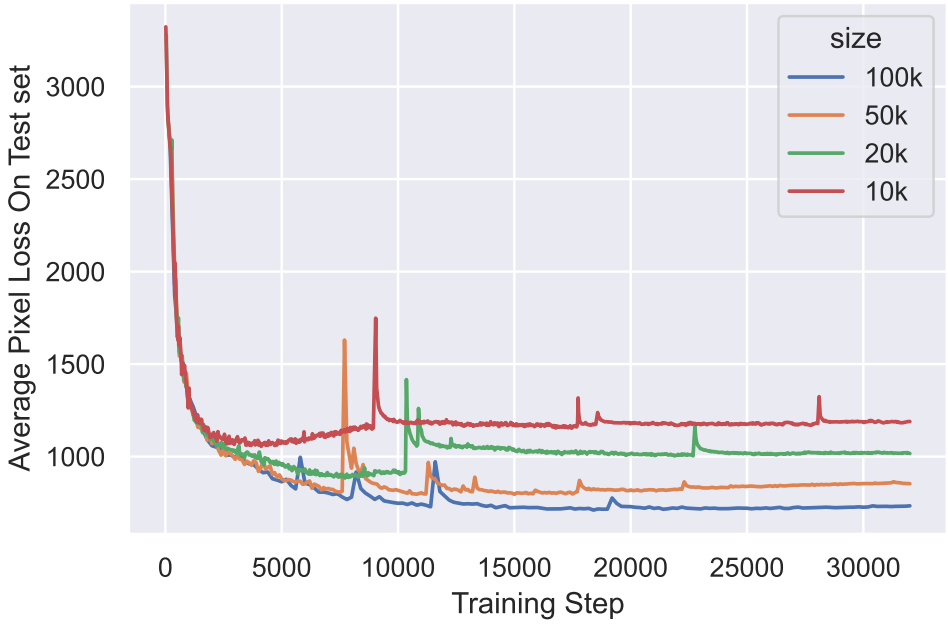

Figure A5: Average pixel loss of test data for different sizes of training data.

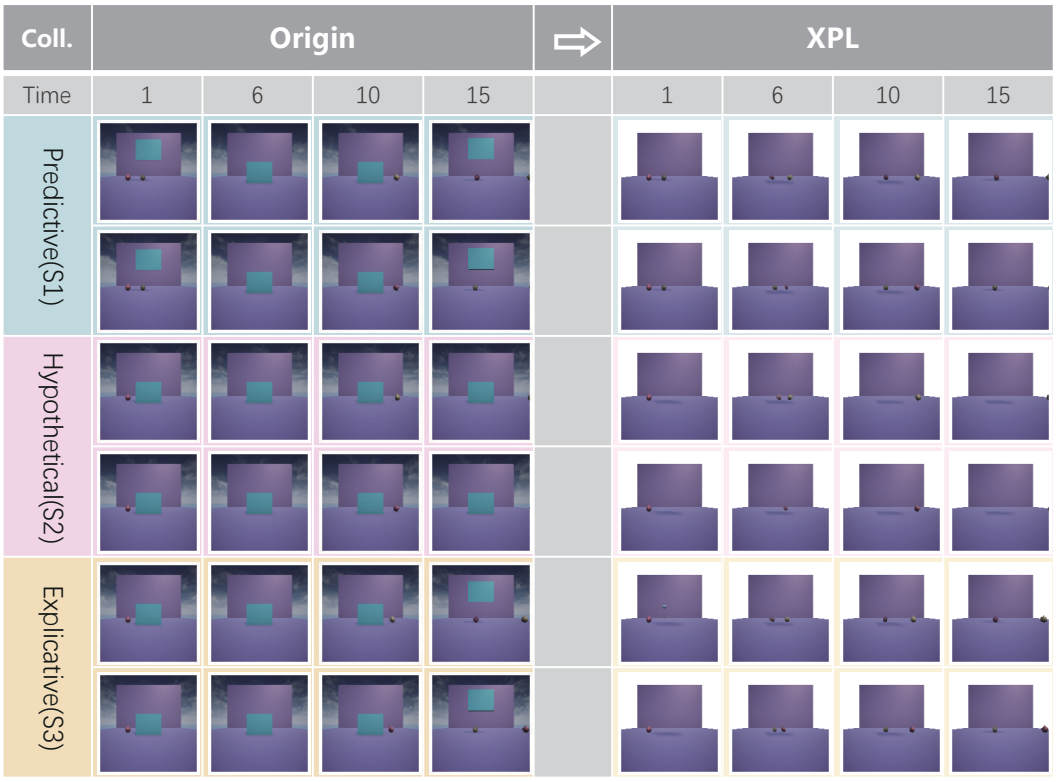

Figure A6: Visualization of the inferred internal representation in XPL during testing in collision scenarios.

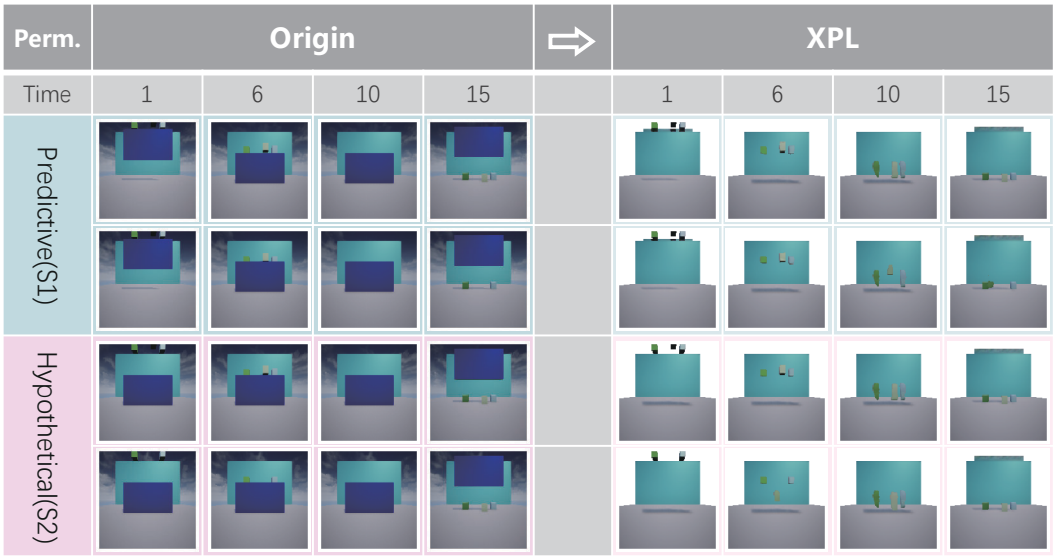

Figure A7: Visualization of the inferred internal representation in XPL during testing in permanence scenarios.

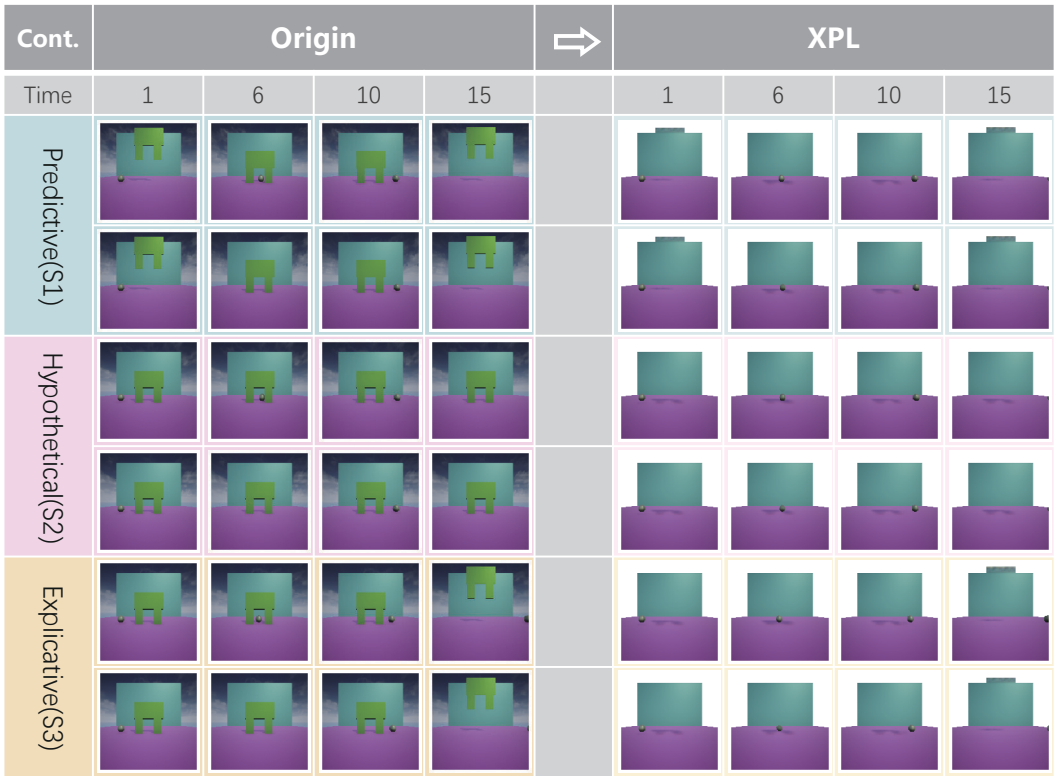

Figure A8: Visualization of the inferred internal representation in XPL during testing in continuity scenarios.