Env	Factor	Pearson correlation SkewFit	WSC (Ours)
Push $n = 1$	obj_x	0.94 ± 0.03	0.95 ± 0.03
	obj_y	0.66 ± 0.17	0.94 ± 0.04
Push $n = 2$	obj1_x	0.59 ± 0.50	0.69 ± 0.37
	obj1_y	0.44 ± 0.68	0.86 ± 0.05
Push $n = 3$	obj1_x	0.44 ± 1.05	0.78 ± 0.11
	obj1_y	0.38 ± 1.44	0.89 ± 0.01

Table 2: **Is the learned policy interpretable?** We measure the correlation between the true factor value of the final state in the trajectory vs. the corresponding latent dimension of the latent goal z_g used to condition the policy. We show 95% confidence over 5 seeds. Our method attains higher correlation between latent goals and final states, meaning that it learns a more interpretable goal-conditioned policy.

Env	Factor	Pearson correlation VAE (SkewFit)	WSC (Ours)
Push $n = 1$	hand_x	0.97 ± 0.04	0.97 ± 0.01
	hand_y	0.85 ± 0.07	0.93 ± 0.02
	obj_x	0.78 ± 0.28	0.97 ± 0.01
	obj_y	0.65 ± 0.31	0.95 ± 0.01
Push $n = 3$	hand_x	0.95 ± 0.03	0.98 ± 0.01
	hand_y	0.50 ± 0.33	0.94 ± 0.03
	obj1_x	0.12 ± 0.18	0.96 ± 0.01
	obj1_y	0.15 ± 0.03	0.92 ± 0.02

Table 3: **Is the learned state representation disentangled?** We measure the correlation between the true factor value of the input image vs. the latent dimension of the encoded image on the evaluation dataset (95% confidence over 5 seeds). We find that unsupervised VAEs are often insufficient for learning a disentangled representation.

A Additional Experimental Results

A.1 Is the learned state representation disentangled?

To see whether weak supervision is necessary to learn state representations that are disentangled, we measure the correlation between true factor values and the latent dimensions of the encoded image in Table. 3. For the VAE, we took the latent dimension that has the highest correlation with the true factor value. The results illustrate that unsupervised losses are often insufficient for learning a disentangled representation, and utilizing weak labels in the training process can greatly improve disentanglement, especially as the environment complexity increases.

A.2 How much weak supervision is needed?

Our method relies on learning a disentangled representation from weakly-labelled data, $\mathcal{D} = \{(s_1^{(i)}, s_2^{(i)}, y^{(i)})\}_{i=1}^N$. However, the total possible number of pairwise labels for each factor of variation is $N = \binom{M}{2}$, where $M \in \{256, 512\}$ is the number of images in the dataset. In this section, we investigate how much weak supervision is needed to learn a sufficiently-disentangled state representation such that it helps supervise goal-conditioned RL.

Number of factors that are labelled: There can be many axes of variation in an image observation, especially as the complexity of the environment grows. For example, the *PushLights* environment with $n = 3$ objects has nine factors of variation, including the positions of the robot arm and objects, and lighting (see Figure 2).

In Figure 8, we investigate whether WSC requires weak labels for all or some of the factors of variation. To do so, we compared the performance of WSC as we vary the set of factors of variation that are weakly-labelled in the dataset $\mathcal{D}$. We see that WSC performs well even when weak labels are not provided for task-irrelevant factors of variation, such as hand position and lighting.

Number of weak labels: In Table 4, we evaluate the quality of the learned disentangled representation model as we vary the number of weak labels, N. We measure disentanglement by evaluating the Pearson correlation between the true factor value compared to the latent dimension. We observe that, even with only 1024 pairwise labels, the resulting representation has a good degree of disenganglement, i.e. Pearson correlation of 0.8 or higher.

In Figure 9, we evaluate the downstream performance of our method on visual goal-conditioned tasks as we vary the number of weak labels. We see that our method outperforms SkewFit when provided at least 1024, 1024, 256, and 128 weak labels for *Push* $n = 1$, *PushLights* $n = 3$, *PickupLightsColors*, and *DoorLights*, respectively. Further, we find that 1024 pairwise labels is generally sufficient for good performance on all domains.

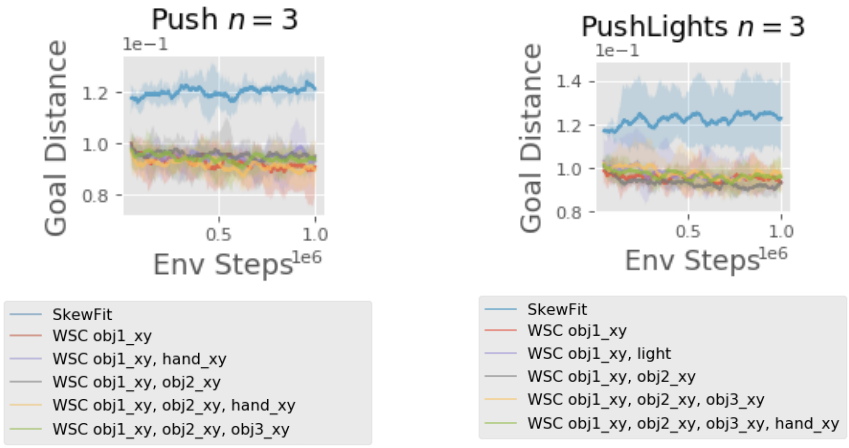


Figure 8: **How many factors of variation need to be labelled?** WSC outperforms SkewFit even without being provided weak labels for task-irrelevant factors, such as hand position and lighting.

	PushLights $n = 3$								
N	hand_x	hand_y	obj1_x	obj1_y	obj2_x	obj2_y	obj3_x	obj3_y	light
128	0.79 ± 0.04	0.64 ± 0.05	0.44 ± 0.08	0.32 ± 0.05	0.60 ± 0.03	0.51 ± 0.05	0.49 ± 0.07	0.41 ± 0.06	0.86 ± 0.04
256	0.87 ± 0.02	0.75 ± 0.05	0.58 ± 0.04	0.57 ± 0.04	0.60 ± 0.04	0.66 ± 0.03	0.65 ± 0.07	0.50 ± 0.06	0.90 ± 0.02
512	0.93 ± 0.01	0.86 ± 0.01	0.71 ± 0.03	0.70 ± 0.05	0.70 ± 0.04	0.58 ± 0.05	0.76 ± 0.04	0.67 ± 0.05	0.85 ± 0.04
1024	0.97 ± 0.01	0.91 ± 0.01	0.86 ± 0.01	0.81 ± 0.02	0.83 ± 0.02	0.80 ± 0.03	0.83 ± 0.03	0.80 ± 0.02	0.94 ± 0.02
2048	0.98 ± 0.00	0.94 ± 0.01	0.89 ± 0.01	0.87 ± 0.03	0.87 ± 0.01	0.86 ± 0.02	0.86 ± 0.02	0.84 ± 0.02	0.92 ± 0.01
4096	0.97 ± 0.00	0.94 ± 0.01	0.93 ± 0.01	0.88 ± 0.01	0.90 ± 0.01	0.88 ± 0.02	0.91 ± 0.01	0.85 ± 0.01	0.95 ± 0.00
VAE	0.00 ± 0.00	0.00 ± 1.00	0.02 ± 2.00	0.00 ± 3.00	0.01 ± 4.00	0.01 ± 5.00	0.02 ± 6.00	0.02 ± 7.00	0.02 ± 8.00

	PickupLightsColors							DoorLights	
N	hand_y	hand_z	obj_y	obj_z	light	table_color	obj_color	door_angle	light
128	0.94 ± 1.00	0.91 ± 2.00	0.72 ± 4.00	0.31 ± 5.00	0.88 ± 6.00	0.43 ± 7.00	0.62 ± 8.00	0.89 ± 3.00	0.84 ± 4.00
256	0.95 ± 1.00	0.96 ± 2.00	0.85 ± 4.00	0.47 ± 5.00	0.95 ± 6.00	0.62 ± 7.00	0.77 ± 8.00	0.95 ± 3.00	0.92 ± 4.00
512	0.96 ± 1.00	0.97 ± 2.00	0.91 ± 4.00	0.61 ± 5.00	0.97 ± 6.00	0.79 ± 7.00	0.82 ± 8.00	0.89 ± 3.00	0.95 ± 4.00
1024	0.95 ± 1.00	0.96 ± 2.00	0.94 ± 4.00	0.69 ± 5.00	0.97 ± 6.00	0.87 ± 7.00	0.92 ± 8.00	0.91 ± 3.00	0.94 ± 4.00
2048	0.95 ± 1.00	0.98 ± 2.00	0.95 ± 4.00	0.75 ± 5.00	0.96 ± 6.00	0.90 ± 7.00	0.93 ± 8.00	0.91 ± 3.00	0.94 ± 4.00
4096	0.95 ± 1.00	0.96 ± 2.00	0.94 ± 4.00	0.80 ± 5.00	0.96 ± 6.00	0.89 ± 7.00	0.96 ± 8.00	0.92 ± 3.00	0.95 ± 4.00
VAE	0.08 ± 0.00	0.25 ± 1.00	0.07 ± 2.00	0.09 ± 3.00	0.24 ± 4.00	0.09 ± 5.00	0.04 ± 6.00	0.01 ± 0.00	0.34 ± 1.00

Table 4: **How many weak labels are needed to learn a sufficiently-disentangled state representation?** We trained disentangled representations on varying numbers of weakly-labelled data samples $\{(s_1^{(i)}, s_2^{(i)}, y^{(i)})\}_{i=1}^N$ ($N \in \{128, 256, \ldots, 4096\}$), then evaluated how well they disentangled the true factors of variation in the data. On the evaluation dataset, we measure the Pearson correlation between the true factor value of the input image vs. the latent dimension of the encoded image. For the VAE (obtained from SkewFit), we took the latent dimension that has the highest correlation with the true factor value. We report the 95% confidence interval over 5 seeds. Even with a small amount of weak supervision (e.g. around 1024 labels), we are able to attain a representation with good disentanglement.

A.3 Noisy data experiments

While the weakly-labelled data can be collected at scale from crowd-sourcers and does not require expertise, the human labellers may mistakenly provide inaccurate rankings. Thus, we evaluated the robustness of the disentangled representation learning on more realistic, noisy datasets which are far less clean than the toy datasets used by Shu et al. [68].

Real-world dataset: We collected 1,285 RGB images (1,029 train, 256 test) on a real Franka robot with 5 blocks (see Fig. 10a). We collected the images under various indoor lighting settings and at different times of the day, but we did not provide labels for the environment lighting conditions. We found that the robot arm often caused occlusion, hiding blocks from the camera view, so we used two RGB cameras placed at different locations, and stacked the RGB images into 6 channels (i.e., image arrays of shape $48 \times 48 \times 6$). We then had a human provide weak labels for the block positions. In Fig. 10b, we show that the learned disentangled model attains a sufficiently high Pearson correlation between the true XY-position of the block (relative to the image frame) vs. the corresponding latent dimension of the encoded image. The results suggest that weakly-supervised disentangled

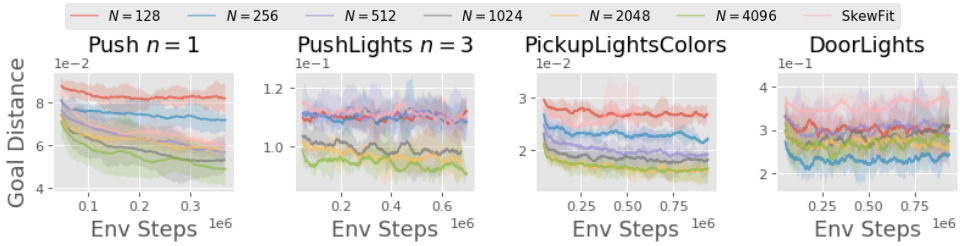
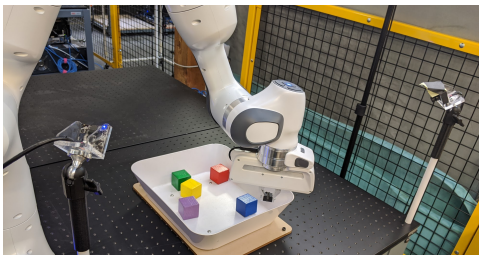


Figure 9: **How many weak labels are needed to help visual goal-conditioned RL?** We evaluate the performance of our method (WSC) on visual goal-conditioned tasks as we vary the number of weak pairwise labels $N \in \{128, 256, \ldots, 4096\}$. We find that 1024 pairwise labels is generally sufficient for good performance on all domains.



(a) Franka robot with 5 blocks

Block color	Pearson correlation x	y
Red	0.747 ± 0.046	0.715 ± 0.016
Blue	0.649 ± 0.034	0.673 ± 0.042
Green	0.718 ± 0.057	0.625 ± 0.066
Yellow	0.663 ± 0.052	0.673 ± 0.057
Purple	0.505 ± 0.041	0.518 ± 0.055

(b) Disentangled representation performance

Figure 10: **Real-world dataset**: *(a)*: We collected 1,285 RGB camera images (1,029 train, 256 test) on a real Franka robot with 5 block objects, and then had a human provide weak labels for the block positions. We collected the images under various lighting conditions, and used two camera viewpoints to overcome object occlusion. *(b)*: Our method attains a sufficiently high Pearson correlation between the true XY-position of the block (relative to the image frame) vs. the latent dimension of the encoded image, suggesting that weakly-supervised disentangled representation learning may be useful for training robots in the real-world. Results are taken over 6 seeds.

representation learning may be useful for training robots in the real-world, despite challenges such as environment stochasticity and object occlusion.

Noisy labels: We generated noisy datasets for *PushLights* with $n \in \{1, 2, 3\}$ objects, where each factor label was corrupted with probability 5% or 10%. In Table 5, we evaluate the quality of the learned disentangled representation model on the noisy datasets. Our method learns a robustly-disentangled representation with 5% noise (around 80% correlation), but achieves lower performance with 10% noise (around 60-70% correlation).

A.4 Latent policy visualizations: WSC vs. SkewFit

We provide additional visualizations of the policy's latent space on more complex environments, previously discussed in Section 5.3. In Figure 11, we compare the latent space of policies trained by WSC and SkewFit. We see that our method produces a more semantically-meaningful goal-conditioned policy, where the latent goal values directly align with the final position of the target object. The difference between WSC and SkewFit grows larger as we increase the complexity of the environment (i.e., increase the number of objects from one to three).

B Algorithm implementation details

Both the disentanglement model (for WSC) and VAE (for SkewFit, RIG, HER) were pre-trained using the same dataset (size 256 or 512). A separate evaluation dataset of 512 image goals is used to evaluate the policies on visual goal-conditioned tasks. We used soft actor-critic [33] as the base RL algorithm. All results are averaged over 5 random seeds.

Noise	PushLights $n=1$ hand_x	hand_y	obj1_x	obj1_y	light	All
5%	0.952 ± 0.012	0.822 ± 0.124	0.730 ± 0.276	0.606 ± 0.298	0.875 ± 0.094	0.797 ± 0.298
10%	0.721 ± 0.507	0.718 ± 0.296	0.520 ± 0.502	0.501 ± 0.270	0.730 ± 0.279	0.638 ± 0.410

Noise	PushLights $n=2$ hand_x	hand_y	obj1_x	obj1_y	obj2_x	obj2_y	light	All
5%	0.949 ± 0.024	0.793 ± 0.226	0.864 ± 0.103	0.844 ± 0.145	0.842 ± 0.147	0.873 ± 0.032	0.936 ± 0.041	0.872 ± 0.118
10%	0.853 ± 0.165	0.588 ± 0.357	0.665 ± 0.422	0.518 ± 0.506	0.747 ± 0.185	0.864 ± 0.02	0.916 ± 0.04	0.736 ± 0.400

Noise	PushLights $n=3$ hand_x	hand_y	obj1_x	obj1_y	obj2_x	obj2_y	obj3_x	obj3_y	All
5%	0.786 ± 0.144	0.78 ± 0.164	0.728 ± 0.217	0.698 ± 0.180	0.791 ± 0.117	0.858 ± 0.057	0.877 ± 0.013	0.833 ± 0.038	0.794 ± 0.661
10%	0.632 ± 0.487	0.551 ± 0.295	0.587 ± 0.264	0.547 ± 0.315	0.613 ± 0.307	0.817 ± 0.023	0.864 ± 0.033	0.851 ± 0.068	0.610 ± 0.643

Table 5: **Noisy labels**: We trained disentangled representations on noisy *PushLights* datasets for $n \in \{1, 2, 3\}$ objects, where each factor label was corrupted with probability 5% or 10%. We then measured the Pearson correlation between the true factor values vs. the corresponding latent dimension. Our method learns robustly-disentangled representations with 5% noise (around 80% correlation), but achieves lower performance with 10% noise (around 60-70% correlation). Results are taken over 5 seeds.

Disentangled representation learning: We describe the disentangled model network architecture in Table 7, which was slightly modified from [68] to be trained on 48×48 image observations from the Sawyer manipulation environments. The encoder is not trained jointly with the generator, and is only trained on generated data from $G(z)$ (see Eq. 1). All models were trained using Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$, learning rate 1e-3, and batch size 64 for 1e5 iterations. The learned disentangled representation is fixed during RL training (Phase 2 in Figure 3).

Goal-conditioned RL: The policy and Q-functions each are feedforward networks with (400, 300) hidden sizes and ReLU activation. All policies were trained using Soft Actor-Critic [33] with batch size 1024, discount factor 0.99, reward scale 1, and replay buffer size 1e5. The episodic horizon length was set to 50 for *Push* and *Pickup* environments, and 100 for *Door* environments. We used the default hyperparameters for SkewFit from [61], which uses 10 latent samples for estimating density. For WSC, we relabelled between 0.2 and 0.5 goals with $z_g \sim p(\mathcal{Z}_\mathcal{I})$ (see Table 6). All RL methods (WSC, SkewFit, RIG, HER) relabel 20% of goals with a future state in the trajectory. SkewFit and RIG additionally relabel 50% of goals with $z_g \sim p^{\text{skew}}(s)$ and $z_g \sim \mathcal{N}(0, I)$, respectively.

VAE: The VAE was pre-trained on the images from the weakly-labelled dataset for 1000 epochs, then trained on environment observations during RL training. We trained the VAE and the policy separately as was done in [61], and found that jointly training them end-to-end (i.e., with backpropagation between VAE loss and policy loss) did not perform well. We used learning rate 1e-3, KL regularization coefficient $\beta \in \{20, 30\}$, and batch size 128. The VAE network architecture and hyperparameters are summarized in Table 8.

SkewFit+pred (Section 5.1): We added a dense layer on top of the VAE encoder to predict the factor values, and added a MSE prediction loss to the β-VAE loss. We also tried using the last hidden layer of the VAE encoder instead of the encoder output, but found that it did not perform well.

SkewFit+DR (Figure 5): We tried with and without adding the VAE distance reward to the disentangled reward $R_{z_g}(s)$ in Eq. 3, and report the best α^{VAE} in Table 6:

$$R^{\text{DR}}(s) = R_{z_g}(s) - \alpha^{\text{VAE}} \|e^{\text{VAE}}(s) - z_g^{\text{VAE}}\| \tag{4}$$

Computing infrastructure: Experiments were ran on GTX 1080 Ti, Tesla P100, and Tesla K80.

Environment	M	Factors (**User-specified factor indices are bolded**)	WSC p_{goal}	α^{DR}
Push $n = 1$	256	hand_x, hand_y, **obj_x**, **obj_y**	0.2	1
Push $n = 2$	256	hand_x, hand_y, **obj1_x**, **obj1_y**, obj2_x, obj2_y	0.3	1
Push $n = 3$	512	hand_x, hand_y, **obj1_x**, **obj1_y**, obj2_x, obj2_y, obj3_x, obj3_y	0.4	0
PushLights $n = 1$	256	hand_x, hand_y, **obj_x**, **obj_y**, light	0.4	1
PushLights $n = 2$	512	hand_x, hand_y, **obj1_x**, **obj1_y**, obj2_x, obj2_y, light	0.4	1
PushLights $n = 3$	512	hand_x, hand_y, **obj1_x**, **obj1_y**, obj2_x, obj2_y, obj3_x, obj3_y, light	0.5	0
Pickup	512	hand_y, hand_z, **obj_y**, **obj_z**	0.4	–
PickupLights	512	hand_y, hand_z, **obj_y**, **obj_z**, light	0.3	–
PickupColors	512	hand_y, hand_z, **obj_y**, **obj_z**, table_color, obj_color	0.4	–
PickupLightsColors	512	hand_y, hand_z, **obj_y**, **obj_z**, light, table_color, obj_color	0.3	–
Door	512	**door_angle**	0.3	–
DoorLights	512	**door_angle**, light	0.5	–

Table 6: **Environment-specific hyperparameters**: M is the number of training images. "WSC p_{goal}" is the percentage of relabelled goals in WSC (Alg. 1). α^{DR} is the VAE reward coefficient for SkewFit+DR in Eq. 4.

Encoder $\mathcal{N}(z; \mu(s), \sigma(s))$
Input: $48 \times 48 \times 3$ image
4×4 Conv, 32 ch, str 2
Spectral norm
LeakyReLU
4×4 Conv, 32 ch, str 2
Spectral norm
LeakyReLU
4×4 Conv, 64 ch, str 2
Spectral norm
LeakyReLU
4×4 Conv, 64 ch, str 2
Spectral norm
LeakyReLU
Flatten
128 Dense layer
Spectral norm
LeakyReLU
$2 \cdot K$ Dense layer
Output: $\mu, \sigma \in \mathbb{R}^K$

Generator $G(z)$
Input: $z \in \mathbb{R}^K$
128 Dense layer
Batch norm
ReLU
$3 \cdot 3 \cdot 64$ Dense layer
Batch norm
ReLU
Reshape $3 \times 3 \times 64$
3×3 Conv, 32 ch, str 2
Batch norm
LeakyReLU
3×3 Conv, 16 ch, str 2
Batch norm
LeakyReLU
6×6 Conv, 3 ch, str 4
Batch norm
Sigmoid
Output: $48 \times 48 \times 3$ image

Discriminator Body
Input: $48 \times 48 \times 3$ image
4×4 Conv, 32 ch, str 2
Spectral norm
LeakyReLU
4×4 Conv, 32 ch, str 2
Spectral norm
LeakyReLU
4×4 Conv, 64 ch, str 2
Spectral norm
LeakyReLU
4×4 Conv, 64 ch, str 2
Spectral norm
LeakyReLU
Flatten
256 Dense layer
Spectral norm
LeakyReLU
256 Dense layer
Spectral norm
LeakyReLU
Output: Hidden layer h

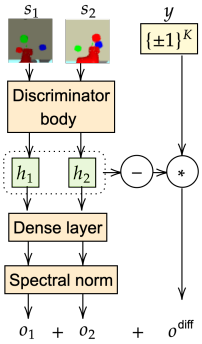
Discriminator $D(s_1, s_2, y)$
Input: Weakly-labelled data $(s_1, s_2, y) \in \mathcal{D}$

Output: Prediction $o_1 + o_2 + o^{\text{diff}} \in [0, 1]$

Table 7: **Disentangled representation model architecture**: We slightly modified the disentangled model architecture from [68] for 48×48 image observations. The discriminator body is applied separately to s_1 and s_2 to compute the unconditional logits o_1 and o_2 respectively, and the conditional logit is computed as $o^{\text{diff}} = y \cdot (h_1 - h_2)$, where h_1, h_2 are the hidden layers and $y \in \{\pm 1\}$.

VAE encoder $\mathcal{N}(z; \mu(s), \sigma(s))$
Input: $48 \times 48 \times 3$ image
5×5 Conv, 16 ch, str 2
ReLU
3×3 Conv, 32 ch, str 2
ReLU
3×3 Conv, 64 ch, str 2
ReLU
Flatten
$2 \cdot L^{\text{VAE}}$ Dense layer
Output: $\mu, \sigma \in \mathbb{R}^{L^{\text{VAE}}}$

VAE decoder
Input: $z \in \mathbb{R}^{L^{\text{VAE}}}$
$3 \cdot 3 \cdot 64$ Dense layer
Reshape $3 \times 3 \times 64$
3×3 Conv, 32 ch, str 2
ReLU
3×3 Conv, 16 ch, str 2
ReLU
Output: $48 \times 48 \times 3$ image

Env	β	Best latent dim L^{VAE} WSC	SkewFit, RIG, HER
Push	20	256	4
Pickup	30	256	16
Door	20	256	16

Table 8: **VAE architecture & hyperparameters**: β is the KL regularization coefficient in the β-VAE loss. We found that a smaller VAE latent dim $L^{\text{VAE}} \in \{4, 16\}$ worked best for SkewFit, RIG, and HER (which use the VAE for both hindsight relabelling and for the actor & critic networks), but a larger dim $L^{\text{VAE}} = 256$ benefitted WSC (which only uses the VAE for the actor & critic networks).

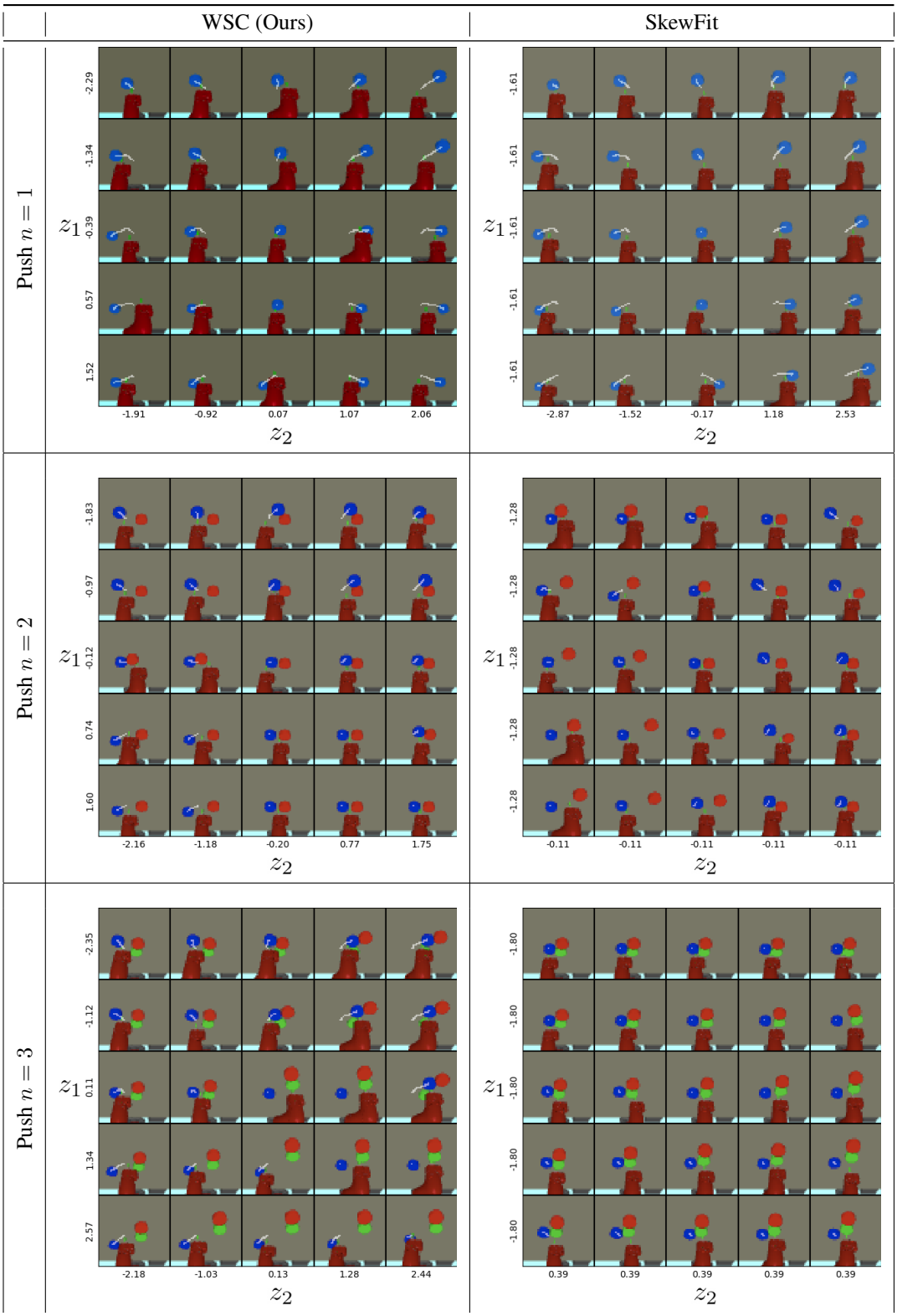


Figure 11: **Interpretable control**: Trajectories generated by WSC (*left*) and SkewFit (*right*), where the policies are conditioned on varying latent goals $(z_1, z_2) \in \mathbb{R}^2$. For SkewFit, we varied the latent dimensions that have the highest correlation with the object's XY-position, and kept the remaining latent dimensions fixed. The blue object always starts at the center of the frame in the beginning of each episode. The white lines indicate the target object's position throughout the trajectory. For WSC, we see that the latent goal values directly align with the direction in which the policy moves the blue object.