

Play to the Score: Stage-Guided Dynamic Multi-Sensory Fusion for Robotic Manipulation (Supplementary Materials)

1 Details of the Task Setup

In Section 4.1 of the main paper, we briefly introduced the basic information of the pouring task and the peg insertion with keyway task, including the task objectives and stage divisions. In this section, we provide a more detailed introduction of the setup of these two tasks. We control the robot arm through a keyboard to complete the tasks and collect human demonstrations.

1.1 Pouring

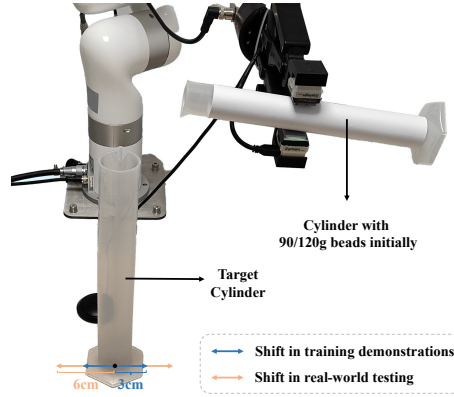


Figure 1: **Illustration of the pouring task.** We randomly shift the fixed target cylinder sideways by 0 ~ 3cm (the blue arrow) in training demonstrations, and shift by 0 ~ 6cm (the orange arrow) during testing.

Setup details. For the pouring task, the robot needs to pour tiny steel beads of specific quality from the cylinder in the hand into another cylinder. In the demonstrations, we randomly shift the fixed target cylinder sideways by 0 ~ 3cm, while during testing, this range expands to 0 ~ 6cm, as illustrated in Figure 1. Following the previous work [1], we use small beads with a diameter of 1mm to simulate liquids. The initial mass of the beads is 90g/120g, while the target mass for pouring out is 40g/60g, both indicated by prompts. Since the camera cannot capture the interior of the target cylinder, other modalities are needed to assess whether the poured-out quantity meets the target. Hence, we use vision (RGB), audio and touch modalities in this task.

Robot action space. In this task, the robot can act in a 2-dimensional action space along the axis x and ϕ , where x represents the horizontal movement and ϕ represents the rotation of the gripper. The action step size is $\Delta x = 0.5mm$ and $\Delta\phi = 0.12^\circ$. There are a total of 5 possible actions ($\pm\Delta x$, $\pm\Delta\phi$ and 0), corresponding to the two directions of the two dimensions of (x, ϕ) and holding still.

Stage division. This task consists of four stages [2]: 1) *Aligning*, where the robot needs to move a graduated cylinder containing beads and align it with the target cylinder, 2) *Start Pouring*, where the

robot rotates the end effector to pour out the beads at an appropriate speed, 3) *Holding Still*, where the robot maintains its posture to continue pouring out beads steadily, and 4) *End Pouring*, where the robot rotates the end effector at the appropriate time to stop the flow of beads. In trajectories, we consider the timesteps of the first downward rotation of the gripper ($-\Delta\phi$), the first holding still after rotation, and the first upward rotation of the gripper ($+\Delta\phi$) as stage transition points.

1.2 Peg Insertion with Keyway.

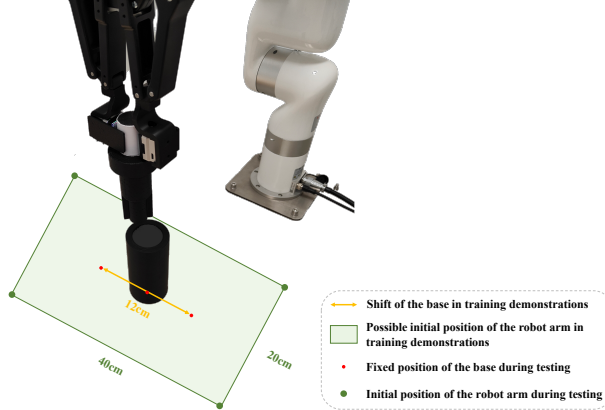


Figure 2: **Illustration of the peg insertion with keyway task.** We randomly shift the fixed base along a 12cm-long parallel line on the desktop (the yellow arrow), and randomly initialize the position of the robot arm inside a 40cm \times 20cm rectangular area (the green rectangle) in training demonstrations. During testing, we fix the position of the base (the red points) and the robot arm (the green points) to several pre-defined points.

Setup details. This task is an upgraded version of peg insertion, where the robot needs to align a peg with a key to the keyway on the base by rotating and then insert the peg fully. The alignment between the key and the keyway in this task primarily relies on tactile feedback, as the camera cannot observe inside the hole. Hence, we use RGB, depth and touch modalities in this task. Considering generalization, we randomly fix the base at any position along a 12cm-long parallel line on the desktop in demonstrations, as illustrated in Figure 2. The robot arm holding the peg can also be initialized inside a 40cm \times 20cm rectangular area around the base. During testing, to ensure fairness, the positions of the base and the robot arm are several pre-defined points.

Robot action space. In this task, the robot arm can move on the three axes x, y, z of Cartesian coordinate, where x, y represents the horizontal movement and z represents the vertical movement. The gripper can also rotate along axis ϕ to align the key with the keyway. Since the vertical movement of the robot arm ($-\Delta z$) and the rotation of the gripper ($+\Delta\phi$) are both unidirectional, there are a total of 7 possible actions ($\pm\Delta x, \pm\Delta y, -\Delta z, +\Delta\phi$ and 0).

Stage division. This task consists of three stages: 1) *First Insertion*, where the robot aligns the peg with the hole and inserts it until the key collides with the base, 2) *Rotating*, where the robot aligns the key with the keyway on the base by rotating the peg based on tactile feedback, and 3) *Second Insertion*, where the robot further inserts the peg to the bottom. Similar to the pouring task, we consider the timesteps of the first gripper rotation ($+\Delta\phi$) and the first downward movement ($-\Delta z$) after rotation as stage transition points.

1.3 Generalization Experiments with Distractors

To verify the generalization of our framework to distractors in the environment, we conduct experiments with visual distractors on both tasks. For the pouring task, we change the color of the cylinder from white to red. As for the peg insertion task, we respectively change the color of the base from black to green, and scatter some clutter around the base. These task settings are illustrated in Fig-

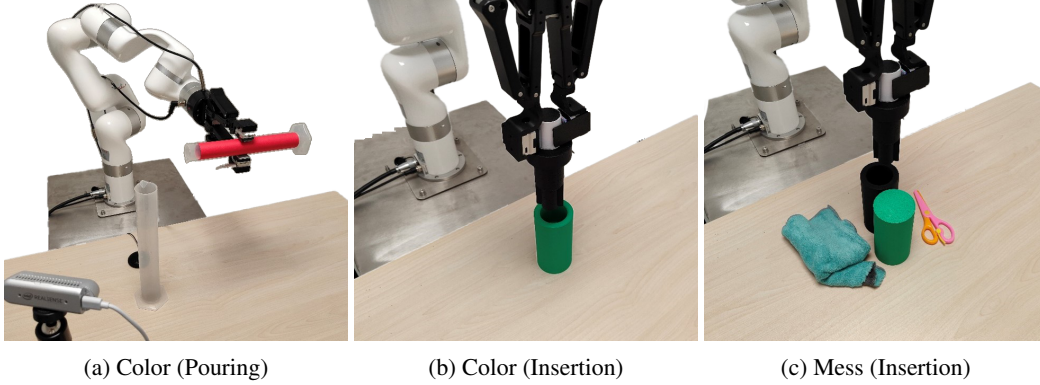


Figure 3: **Illustration of tasks with distractors.** For the pouring task, we change the the color of the cylinder from white to red (denoted as “Color (Pouring)”). For the peg insertion with keyway task, we respectively change the color of the base from black to green (denoted as “Color (Insertion)”), and scatter some clutter around the base (denoted as “Mess (Insertion)”).

ure 3. Besides introducing distractors, the settings for these tasks remain consistent with the main experiments.

2 Implementation Details

Following [1], we resize visual and tactile frames to 140×105 and randomly crop to 128×96 during training. We also use color jitter for image augmentation. For audio modality, we resample the wave signal at 16kHz and generate a 64×50 mel-spectrogram through short-time Fourier transform, with 400 FFT windows, hop length of 160, and mel bin of 64. We employ ResNet-18 [3] network as the uni-modal encoder. Each encoder takes a brief history of observations spanning $T = 6$ timesteps. We train all the models using Adam [4] with a learning rate of 10^{-4} for 75 epochs. We perform linear learning rate decay every two epochs, with a decay factor of 0.9.

For action history, we use a buffer of length 200 to store actions. Each action is encoded using one-hot encoding. Action sequences shorter than 200 are padded with zeros. For both tasks, we consider the 15 timesteps near the stage transition point as the soft constraint range ($\gamma = 15$). We set $\lambda = 5.0$ for both tasks to control the intensity of penalty. For the learnable stage token $[stage_i]$, we initialize it with the mean of all state tokens on samples within the i -th stage after warmup training for 1 epoch. We use $\beta = 0.5$ for the calculation of the stage-injected state token z_t^* . Moreover, to prevent gradients from becoming too large, we also truncated the gradients of the stage score penalty, restricting them to solely influence the gate network.

3 Random Attention Blur

In order to prevent the model from simply memorizing the actions corresponding to attention score patterns, we introduce random attention blur mechanism to the stage-guided dynamic fusion module. For each input, we replace the attention scores on all feature tokens with the same average value $\frac{1}{M \times T}$ with a probability p , where M is the number of modalities and T is the timestep number of the brief observation history, as illustrated in Figure 4. We set $p = 0.25$ in both tasks.

We introduce this mechanism due to the potential overfitting issue in the model, where the policy head (MLP in the dynamic fusion module) learns the correspondence between the distribution of attention scores and actions. For instance, when using the trained MS-Bot model (without random attention blur) to complete pouring tasks, manually increasing the attention scores on tactile feature tokens invariably leads the model to predict the next action as upward rotation ($+\Delta\phi$), regardless of the current stage of the task. This phenomenon suggests that the stage comprehension module

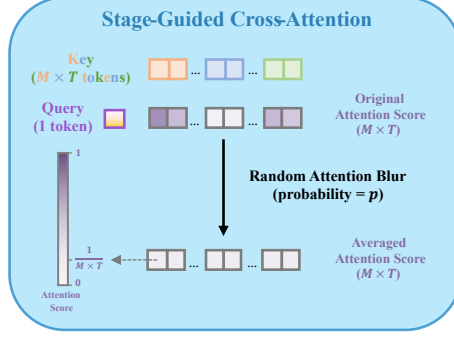


Figure 4: **Illustration of random attention blur in the stage-guided dynamic fusion module.** We randomly replace the attention scores on all feature tokens with the same average value $\frac{1}{M \times T}$ with a probability p .

in the model partially assumes the role of action prediction rather than focusing solely on stage understanding. Therefore, randomly blurring attention scores can compel the policy head to focus on the information from the feature tokens and better decouple the stage comprehension module from the dynamic fusion module.

4 Comparison of Attention Scores between MULSA and MS-BOT

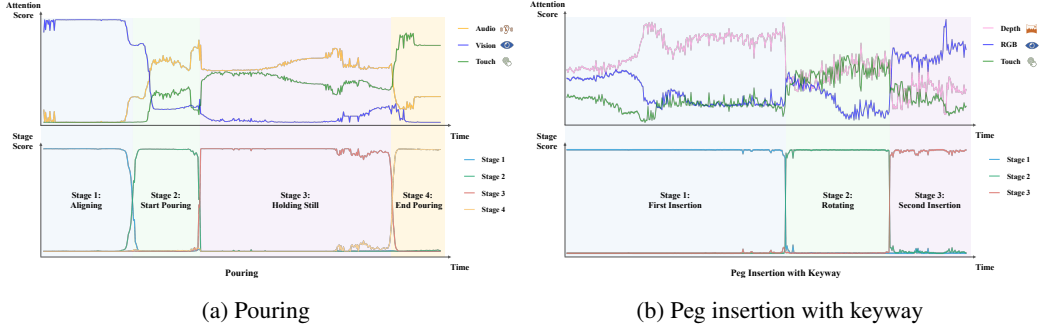


Figure 5: **Visualization of the aggregated attention scores for each modality and stage scores of MS-Bot in both tasks.** At each timestep, we average the attention scores on all feature tokens of each modality separately. The stage score is the output of the gate network after softmax normalization.

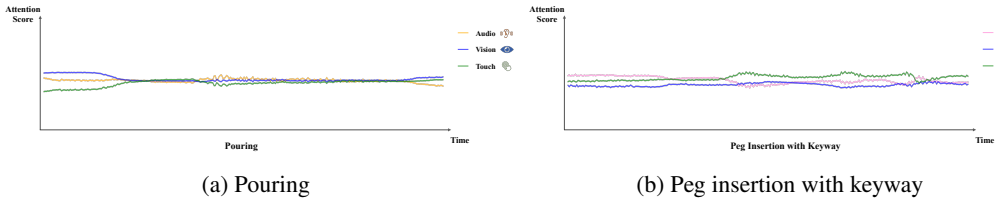


Figure 6: **Visualization of the aggregated attention scores of MULSA for each modality in both tasks.** At each timestep, we average the attention scores on all feature tokens of each modality separately. The range of the attention score axis in the figure is consistent with Figure 5.

In Section 4.4 of the main paper, we illustrate the aggregated attention scores for each modality and the stage scores of MS-Bot in the pouring task, as shown in Figure 5a. We also record the changes in attention scores and stage scores in the peg insertion with keyway task, as shown in Figure 5b. The results in the figure demonstrate that our model accurately predicts the rapid changes in stages

for both tasks. As a result, the attention scores across modalities guided by stage comprehension are also relatively stable, exhibiting clear inter-stage changes and minor intra-stage adjustment.

As a comparison, we also visualize the attention scores of another baseline MULSA [1] with self-attention fusion, as shown in Figure 6. The range of the attention score axis in the figure is consistent with Figure 5 (0 \sim 0.9). It is evident that the attention scores of the modalities in the MULSA model are close and exhibit a small variation, lacking clear stage characteristics. This indicates that the MULSA model fails to fully leverage the advantages of dynamic fusion compared to the concat model.

5 Detailed Experimental Results of Pouring

Methods	Pouring Initial (g)		Pouring Target (g)	
	90	120	40	60
MS-Bot	1.60 \pm 1.10	5.58 \pm 1.79	6.48 \pm 1.55	1.80 \pm 0.95
- Attention Blur	1.72 \pm 1.09	5.70 \pm 1.74	6.55 \pm 1.38	1.95 \pm 1.32
- Stage Comprehension	2.52 \pm 1.12	6.15 \pm 1.64	6.80 \pm 1.59	2.92 \pm 1.40
- State Tokenizer	3.05 \pm 1.01	6.42 \pm 1.98	7.12 \pm 1.66	4.19 \pm 1.24

Table 1: Impact of each component of our framework in pouring task (mean \pm standard deviation). ‘-’ indicates further removing the module from the model in the previous line.

Methods	Pouring Initial (g)		Pouring Target (g)	
	90	120	40	60
Concat	8.75 \pm 2.02	10.69 \pm 2.45	10.72 \pm 2.35	8.46 \pm 2.03
Du et al. [5]	8.51 \pm 1.79	9.54 \pm 2.10	9.98 \pm 2.20	8.04 \pm 1.85
MULSA [1]	4.72 \pm 1.30	7.83 \pm 1.71	8.45 \pm 1.50	5.56 \pm 1.13
MS-Bot	2.04 \pm 1.40	6.10 \pm 1.49	7.62 \pm 1.94	2.41 \pm 1.47

Table 2: Comparison of performance in scenes with visual distractors in the pouring task(mean \pm standard deviation). We change the color of the cylinder from white to red during testing.

In Section 4.4 and 4.5 of the main paper, we show the overall error on all settings of the pouring task. In this section, we comprehensively present the detailed result of the component ablation and the distractor experiment on all the settings of the pouring task in Tables 1 and 2. As shown in Table 1, the contributions from both the state tokenizer and the stage comprehension module on all settings demonstrate the importance of the coarse-to-fine stage comprehension. The consistent lead of MS-BOT across all settings of the pouring task in Table 2 also demonstrates the generalizability of stage comprehension.

6 Evaluation of Hyper-parameter Settings

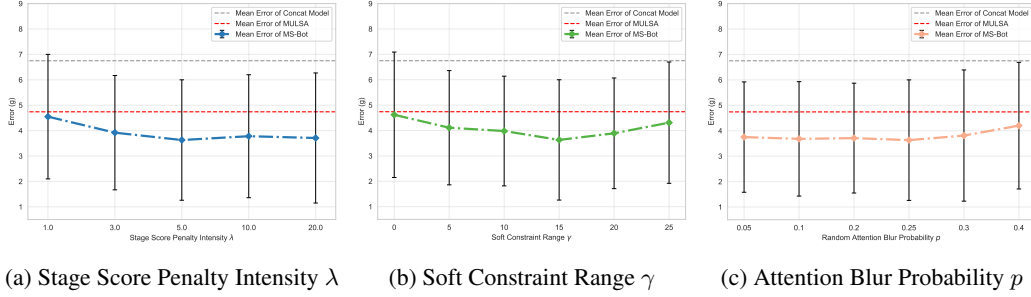


Figure 7: **Performance of MS-Bot on one setting of the pouring task when changing the hyper-parameters λ , γ and p .** We fix the others when changing one hyper-parameter. We report the mean error of the concat model, MULSA and MS-Bot. The error bars represent the standard deviation of errors for MS-Bot.

In this section, we test the sensitivity of MS-Bot to different hyper-parameter settings. Specifically, we test the performance of MS-Bot under different penalty intensity λ , soft constraint range γ and probability of random attention blur p in one setting of the pouring task. We train all the models to pour out 40g of small beads with different initial mass (90g/120g). We set $\lambda = 5.0$, $\gamma = 15$ and $p = 0.25$ by default, and keep the others fixed when we modify one of the hyper-parameters.

We present the evaluation results in Figure 7. Our MS-Bot consistently outperforms both the concat model and MULSA across various hyper-parameter settings, indicating that the performance of our MS-Bot is relatively stable to hyper-parameter variations. However, we also observe that when setting λ and γ to small values ($\lambda = 1.0$ in Figure 7a and $\gamma = 0$ in Figure 7b), the performance of MS-Bot drops and becomes closer to MULSA with simple self-attention fusion. This is because when λ is too small, the prediction of the current stage becomes inaccurate due to the insufficient training. When γ is too small, the model is forced to make drastic changes in the stage score prediction within very few timesteps, leading to unstable stage predictions. Both of these factors weaken the efficacy of stage comprehension within MS-Bot. We also find that setting too large p ($p > 0.3$ in Figure 7c) can bring negative impacts as the attention blur truncates the gradients backpropagated to the stage comprehension module and the state tokenizer. Excessive attention blur can impair the training of these two modules.

References

- [1] H. Li, Y. Zhang, J. Zhu, S. Wang, M. A. Lee, H. Xu, E. Adelson, L. Fei-Fei, R. Gao, and J. Wu. See, hear, and feel: Smart sensory fusion for robotic manipulation. In *Conference on Robot Learning*, pages 1368–1378. PMLR, 2023.
- [2] D. Zhang, Q. Li, Y. Zheng, L. Wei, D. Zhang, and Z. Zhang. Explainable hierarchical imitation learning for robotic drink pouring. *IEEE Transactions on Automation Science and Engineering*, 19(4):3871–3887, 2021.
- [3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [4] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [5] M. Du, O. Y. Lee, S. Nair, and C. Finn. Play it by ear: Learning skills amidst occlusion through audio-visual imitation learning. *arXiv preprint arXiv:2205.14850*, 2022.