Supplementary Material
SCONE: A Food Scooping Robot Learning Framework with Active Perception

Anonymous Author(s)
Affiliation
Address
email

The supplementary materials consist of:

1. Demonstration video.
2. Details of dataset.
3. Baseline approach.
4. Analysis of experiment results.
5. Details of our proposed method.
6. Visualization of attention scores.

1 Demonstration video

In the supplementary video, we show 1) a brief introduction of our food scooping robot learning framework and 2) illustrations of scooping tasks and qualitative results.

2 Details of Dataset

In this section, we provide more details about the data collection and preparation process for training.

2.1 Food Preparation

We select a total of 12 categories of food items for our real-world experiment, with 6 categories used for training and the remaining 6 categories used for testing. To simplify the complexity, we have limited the differences between categories primarily to **particle size** and the **amount** of food items.

Food in the Training Set. The food categories for training contain brown rice, mung bean, soybean, chocolate call, dried jujube, and cheese ball. For food items with small particles such as brown rice, mung bean, and soybean, we fill the bowl to approximately 2/3 of its capacity. This threshold ensures that the interaction can be carried out without the risk of spilling the food items. For food items with large particles such as chocolate balls, dried jujube, and cheese balls, we select more than one piece but fewer than a certain number based on their size. This is done to prevent the spoon from getting stuck or breaking the food items when there is insufficient space in the bowl for the spoon to reach them.

Food in the Testing Set. The food categories for testing contain sago, red bean, orange, macadamia, penne, and fruit candy. In the testing set, we have designed three levels of difficulty to evaluate the performance of the models. Both the **Basic** and **Extended** settings in the evaluation include sago, red bean, orange, and macadamia, and the peculiar setting includes penne and fruit candy.

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

• **Basic Setting:** In this setting, the conditions are kept identical to the training set. For food items with small particles, the bowl is filled to approximately 2/3 of its capacity. For food items with large particle sizes, more than one entity is included in the bowl.
• **Extended Setting:** In this setting, we change the combination of properties related to particle size and amount of food items. We want to explore different scenarios to evaluate the performance of our model under varying conditions. For food items with small particles, we fill the bowl with a smaller quantity of these items. The intention is to keep the height of the food in the bowl similar to that of the food items with large particles in the **Basic** setting. Conversely, we increase the amount of food items with large particles.
• **Peculiar Setting:** These food items in the peculiar setting had unique features, such as different shapes, colors, or textures, that are not present in the training set. By introducing these visually distinct food items, we aim to challenge the model's capacity to recognize and handle novel objects effectively.

2.2 Manipulation Policy

During the data collection phase, we employ two distinct manipulation policies to ensure the successful scooping of the food items. To prevent spilling, we adopt a specific policy for scooping up food items with small particles. The strategy involves positioning the spoon at a shallow depth under the height of the food items. By adopting this approach, the risk of spillage can be minimized and the food items were securely contained within the spoon during the scooping process. When dealing with food items that have large particles, we position the spoon at the lowest point within the bowl and gently push the items toward the edge. This allows the food to roll into the spoon and is able to successfully scoop up without any spillage. Figure 1 shows the end-effector positions during manipulation in the training dataset.

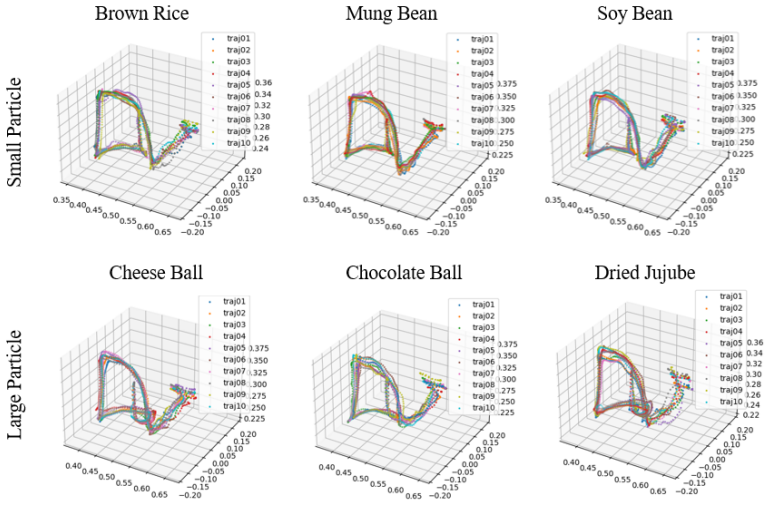

Figure 1: Visualization of End-effector Position During Manipulation.

2.3 Implementing Detail of Overall Scooping Task

The task is divided into three stages: interacting, scooping, and transferring. Among these stages, only the scooping stage requires learning. The overall task follows a predefined procedure. The interaction data is collected by replaying the recorded end-effector trajectory. The model then begins predicting the scooping stage. Once the scooping stage is completed, the transferring stage is initiated. The trajectory for the transferring stage is obtained by averaging all training data. Considering that the ending position of the scooping stage for every testing food tends to be relatively close to initial position of the transferring stage, the Panda arm then proceeds to move to the first point

using simple motion planning. Once the transition is complete, the rest of the trajectory is replayed accordingly.

3 Baseline Approach

This section provides detailed explanations of the baselines in comparison with SCONE.

3.1 BC-based Method

When adopting Learning from Demonstration (LfD), behavior cloning (BC) is a straightforward approach for direct learning from both observation and action. We conducted the task using 4 different BC models, both with and without the inclusion of interacting data.

BC one stage. In BC (one stage), we consider scooping and transferring as a single stage. This implies that the BC (**one stage**) model is required to learn the long-horizon task only through observation, without any additional input.

BC without interaction. The most basic method that learns the scooping task through observations.

BC with food ID. To implement BC with food ID, we applied the trained classification model to the interaction data. The predicted food ID was then conditioned as a one-hot vector and concatenated with the observation $z_{\texttt{gi}}$, serving as the input for the BC model.

BC with interaction. In the case of BC with interaction, the input for the model is the concatenation of latent of observation and interaction data.

3.2 Template Policy

One of the approaches we explore in utilizing interaction data is the selection of trajectories from the pre-interacting data based on their similarity to the food categories in the training dataset. The template trajectory is obtained by averaging the sequences of end-effector pose of 6 different food categories respectively.

Rand. Template. Random policy is selected arbitrarily.

Classified Template. The template policy is selected accordingly based on the predicted category by the classifier.

3.3 Dynamical System Model

MTRNN. We utilized the multiple time scale recurrent neural network by [1] to update the initial parameter Cs_0 using the interaction data in the testing stage.

4 Analysis of Experiment Results

4.1 Failure Cases

During our real-world evaluation, the failure cases observed included instances of spilling (SP), insufficient food on the spoon (IF), failed attempts to scoop (FA), collisions (CO), and others (OT). We select the orange, sago, and penne as examples, and their results can be found in Table 1 and Table 2.

Spilling (SP). Spilling happens at the end of the scooping process, especially when there is an excessive amount of food on the spoon. Consequently, during the stages of transferring or when withdrawing the spoon from the bowl, the excess food spills out.

Insufficient food (IF). The failure case of insufficient food occurs when the amount of food on the spoon is unable to cover at least one-third of its surface area. This is frequently attributed to the spoon not being inserted deep enough into the bowl to effectively reach the desired food portion.

	Basic - Orange					
	SP	IF	FA	CO	OT	Failed
BC (one stage)	0	0	5	3	1	9
BC (w/o interact)	2	0	3	3	0	8
BC (w/ food id)	1	0	2	1	1	5
BC (w/ interact)	1	0	2	0	0	0
Rand. Template	0	0	5	0	0	5
Classified Template	0	0	0	0	0	0
MTRNN [1]	0	0	0	0	0	**0**
SCONE (Ours)	0	0	0	0	0	**0**

(a) Basic - Orange

	Basic - Sago					
	SP	IF	FA	CO	OT	Failed
BC (one stage)	0	0	0	0	10	10
BC (w/o interact)	8	0	0	0	0	8
BC (w/ food id)	1	0	2	1	1	5
BC (w/ interact)	2	0	2	8	0	10
Rand. Template	2	0	2	0	0	4
Classified Template	10	0	0	0	0	10
MTRNN [1]	0	0	0	0	0	**0**
SCONE (Ours)	1	0	0	0	0	1

(b) Basic - Sago

Table 1: Basic - Failure Cases

	Extended - Sago					
	SP	IF	FA	CO	OT	Failed
BC (one stage)	0	0	0	0	10	10
BC (w/o interact)	5	1	0	1	0	7
BC (w/ food id)	1	0	2	1	1	5
BC (w/ interact)	0	4	0	3	0	7
Rand. Template	3	4	0	0	0	7
Classified Template	0	0	0	0	0	**0**
MTRNN [1]	0	5	0	5	0	10
SCONE (Ours)	3	1	0	0	0	4

(a) Extended - Sago

	Peculiar - Penne					
	SP	IF	FA	CO	OT	Failed
BC (one stage)	1	0	8	3	0	9
BC (w/o interact)	1	0	6	1	0	8
BC (w/ food id)	0	0	5	1	0	6
BC (w/ interact)	1	0	0	4	0	5
Rand. Template	4	0	3	0	0	7
Classified Template	2	0	7	0	0	9
MTRNN [1]	2	0	0	0	0	**2**
SCONE (Ours)	2	0	1	0	0	3

(b) Peculiar - Penne

Table 2: Extended and Peculiar - Failure Cases

Failed Attempts (FA). Failed attempts to scoop are attributed to the same underlying reason as insufficient food. In these cases, the spoon fails to acquire any amount of food, rather than scooping up an insufficient quantity.

Collisions (CO). Relying on vision-based information can lead to incorrect decisions when facing out-of-distribution situations. In such scenarios, collisions between the spoon and the bowl may occur.

Others (OT). In addition to the failure cases mentioned earlier, there are instances where the task cannot be successfully completed or the food ends up being damaged.

4.2 Result Analysis

Table 1: Orange. The BC-based baselines achieved low task success rates due to the high incidence of failed attempts because the weight of the orange used in the testing set is heavier than the foods in the training set. By contrast, the MTRNN and SCONE models exhibited stable performance due to their ability to learn and understand the conditions for successful scooping from the provided demonstrations. The classified template method also achieved higher performance due to its ability to select suitable templates for food items.

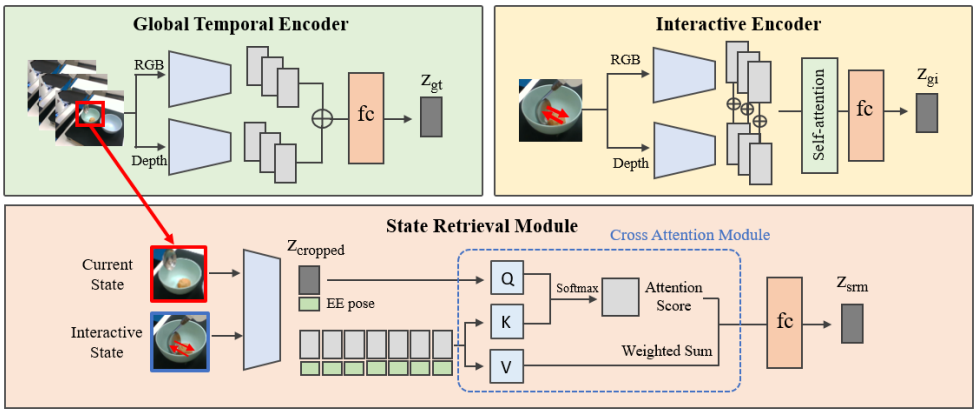

Figure 2: Details of Each Module in SCONE

Table 1: Sago. The BC-based baselines encountered challenges in providing correct predictions, causing several failure cases during the manipulation. Moreover, the template-based method has been limited by its reliance on the predictions from the classifier. In cases where the classifier selected a template intended for food items with large particles, it often resulted in spillage. The MTRNN and the proposed SCONE method demonstrated their ability to overcome this challenge, achieving a higher success rate.

Table 2: Extended Sago. The number of spillages (SP) decreases significantly compared to the basic settings because the amount of sago been reduced, but there was an increase in the occurrence of insufficient food (IF) cases, due to the improper depth insertion of the spoon into the bowl in most of the methods. The Select Template model classified sago as chocolate balls, resulting in 10 successful trials when following the corresponding template. MTRNN demonstrated poor performance under this particular setting. During the scooping stage, it showed a hovering behavior within the bowl and occasionally collided with it. However, SCONE is capable of handling the challenging setting, achieving a success rate of 6 out of 10.

Table 2: Penne. To test the models' generalization abilities, we conducted experiments using peculiar-shaped foods such as penne and fruit candy. Though the color of penne looks familiar to soy beans, the shapes of them are totally different, which led to failed attempts (FA) when employing the soy bean template. While MTRNN was able to handle penne, the jittering trajectory could lead to instability and spillage (SP), which we would like to avoid in real-world evaluation. Our SCONE behaved more stable and maintained a smooth trajectory.

5 Details of Proposed Method

See Figure 2 for detailed information on each encoding module. All the observation inputs are RGB-D images, which are processed through corresponding convolutional layers to extract features.

Global temporal encoder. The global temporal encoder takes as input a sequence of current observations of length N. In our implementation, we set N to 10, which means that the model can access the previous 10 observations within a time window of 1 second. This allows the model to capture the temporal dynamics and dependencies in the input data. Then, the sequences of RGB and depth images are processed separately by their respective encoders, and the output features are concatenated and flattened into a one-dimensional embedding. To further reduce the dimensionality of the features, a fully-connected layer is applied to downsample the features to the dimension of 128.

Interactive encoder. We utilize an interactive encoder to process the sequence of observations captured during the interaction stage. The number of frames K is set to 7. Similar to the encoding process in the global temporal encoder, both RGB and depth images within the sequence are pro-

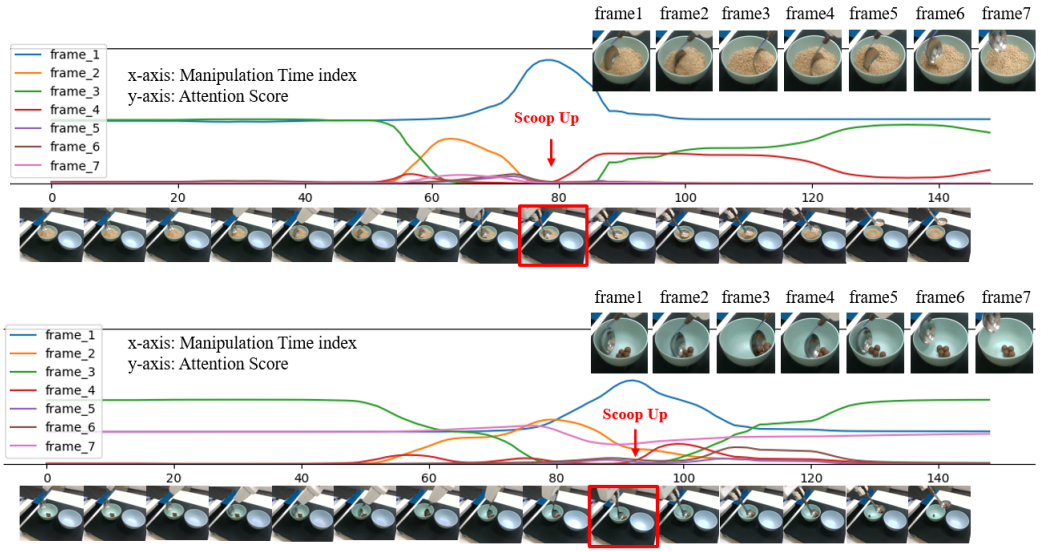

Figure 3: Visualization of Attention Scores. The upper is the manipulation process of brown rice, and the lower is the manipulation process of chocolate ball.

cessed independently through their respective convolutional layers. The output features from these layers are then concatenated to form a sequence of feature maps, capturing the visual information from both modalities. Then, we introduce the multi-head self-attention mechanism, allowing the model to focus on relevant parts of the input sequence. The output features passed through a fully-connected layer, which reduces their dimension to 128.

State retrieval module. The state retrieval module tasks the current local observation and the sequence of interaction as inputs. For the current local observation, which consists of cropped RGB-D images captured at the current time step, we pass them through an encoding module to obtain latent embeddings. These embeddings are then concatenated with the current end-effector pose, resulting in the latent state representation denoted as $z_{\texttt{cropped}}$. Regarding the sequence of interaction, we apply the same encoding layers to process the observations and obtain corresponding feature embeddings. Similar to the current local observation, we also concatenate the end-effector poses with the encoded features, creating a sequence of states in the interaction. To retrieve the critical state information, we use $z_{\texttt{cropped}}$ as a query and compute its relationship with the sequence of states in the interaction. This is achieved by employing cross-attention mechanisms that output weighted feature embeddings. These embeddings are then flattened and downsampled to a dimension of 32, obtaining the $z_{\texttt{srm}}$. Furthermore, we also downsample the latent state representation $z_{\texttt{cropped}}$ to a dimension of 32.

6 Visualization of Attention Scores

Figure 3 shows more examples of visualization for attention scores. Based on the results, it is evident that the model can accurately capture the state information without human labeling; this is demonstrated by the consistent patterns observed in the changing attention scores over time at each trial. Overall, our analysis reveals that **frame 1** and **frame 3** exhibit higher attention scores compared to other frames. We attribute this to the presence of important state information related to the spoon's contact with the food items under specific end-effector poses. These frames can be seen as providing subgoal-like cues, guiding the model in the scooping task. Additionally, we observed that in food items with large particles, **frame 3** tends to have higher attention scores than **frame 1** initially. This is because the state captured in **frame 3** is more similar to the state prior to the scooping action in scenarios involving large particles. On the other hand, we noticed that the highest

attention score for **frame 1** typically occurs close to the timing of the actual scooping action. This suggests that **frame 1** serves as a general guide for determining "how" and "where" to scoop up the food items. These findings highlight the model's ability to effectively identify and utilize critical state information from the interacting data to inform its scooping strategy.

References

[1] N. Saito, T. Ogata, S. Funabashi, H. Mori, and S. Sugano. How to select and use tools?: Active perception of target objects using multimodal deep learning. *IEEE Robotics and Automation Letters*, 6(2):2517–2524, 2021.