2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636301

Transformer-based deep imitation learning for dual-arm robot manipulation

Heecheol Kim[1,2], Yoshiyuki Ohmura[1], Yasuo Kuniyoshi[1]

***Abstract*— Deep imitation learning is promising for solving dexterous manipulation tasks because it does not require an environment model and pre-programmed robot behavior. However, its application to dual-arm manipulation tasks remains challenging. In a dual-arm manipulation setup, the increased number of state dimensions caused by the additional robot manipulators causes distractions and results in poor performance of the neural networks. We address this issue using a self-attention mechanism that computes dependencies between elements in a sequential input and focuses on important elements. A Transformer, a variant of self-attention architecture, is applied to deep imitation learning to solve dual-arm manipulation tasks in the real world. The proposed method has been tested on dual-arm manipulation tasks using a real robot. The experimental results demonstrated that the Transformer-based deep imitation learning architecture can attend to the important features among the sensory inputs, therefore reducing distractions and improving manipulation performance when compared with the baseline architecture without the self-attention mechanisms.**

***Index Terms*— Imitation Learning, Dual Arm Manipulation, Deep Learning in Grasping and Manipulation**

I. Introduction

End-to-end deep imitation learning, which controls a robot by imitating an expert's demonstration, has gained popularity in the robotics community because of its potential application to dexterous manipulation tasks without the need to model environments or objects (e.g., [1], [2], [3]).

Dual-arm manipulation increases the manipulability of objects in many tasks [4] (e.g., peg-in-hole tasks [5], cable deformation tasks [6], and resolving occlusion with one hand while grasping an object with the other hand [7]). Because the human operator is familiar with dual-arm manipulation, it is easy to transfer the bimanual skills of a human using teleoperation [4]. Therefore, the imitation learning framework, which generates demonstration data by human teleoperation, is adequate for complex dual-arm manipulation tasks. However, few studies have investigated the dual-arm manipulation framework using imitation learning (e.g., [8], [9], [10]), and many of these focused on learning the hierarchical structure of the subtasks. However, we believe the problem to solve for deep imitation learning on dual-arm manipulation tasks lies in the distractions caused by the increased dimensions of the concatenated left/right robot arm kinematics states, which is essential for the collaboration of both arms for dual-arm manipulation.

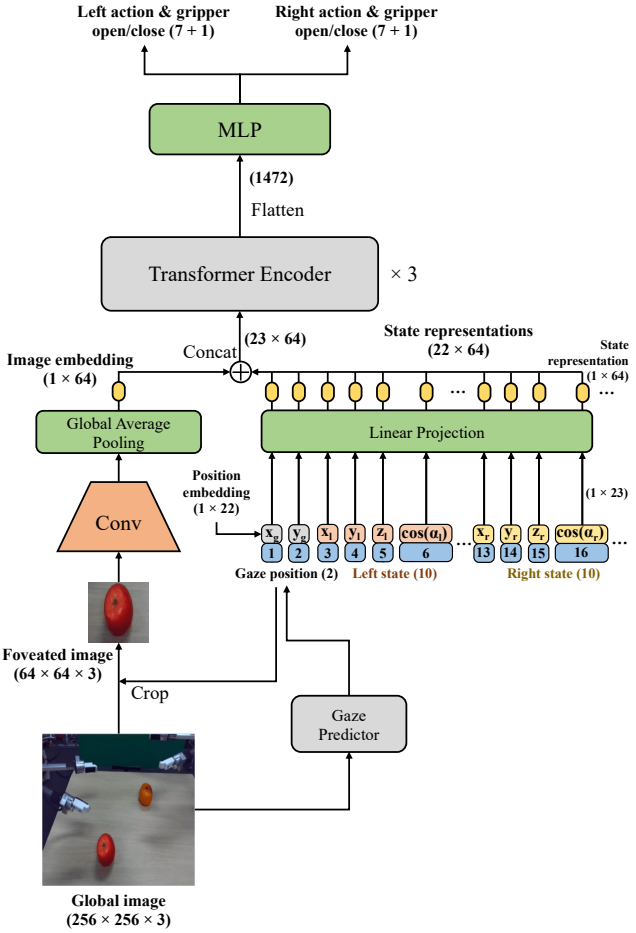


Fig. 1: Proposed Transformer-based deep imitation learning architecture for the dual-arm manipulation.

Our previous study revealed that the attention mechanism is important for the visuomotor control of robots with deep imitation learning [11] to suppress distractions. This study measured the human gaze position with an eye-tracker while teleoperating the robot to generate demonstration data and used only the foveated vision around the predicted gaze position for deep imitation learning to suppress visual distraction from task-unrelated objects.

The gaze-based visual attention mechanism requires the robot's kinematics states from its somatosensory information because the peripheral vision should be masked out to suppress visual distractions. However, in the dual-arm

This paper is based on results obtained under a Grant-in-Aid for Scientific Research (A) JP18H04108 of The University of Tokyo.

[1] Laboratory for Intelligent Systems and Informatics, Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan (e-mail: {h-kim, ohmura, kuniyosh}@isi.imi.i.u-tokyo.ac.jp, Fax: +81-3-5841-6314)

[2] Corresponding author

deep imitation learning framework, distraction is also caused by high dimensional kinematics states. For example, when the robot reaches its right arm to an object, the left arm kinematics states are not used to compute the policy and become distractions. Unlike visual attention which can be trained with gaze, there is no such teaching signal for somatosensory information. Thus, a novel method to learn somatosensory attention in an unsupervised way is required.

Self-attention architectures, especially the Transformer [12], evaluate relationships between elements on an input sequence. Self-attention was first introduced to solve natural language processing (e.g., [13], [14], [15]), and has since been widely adopted in many different domains such as image classification [15], object detection [16], and high-resolution synthesis [17], and learning multiple domains with one model [18]. This wide application of the self-attention to different domains may imply that it can also be used to suppress distractions in kinematics states, which is a problem in dual-arm manipulation.

To this end, we propose a Transformer-based attention mechanism for sensory information for dual-arm robot imitation learning (Fig. 1). Because the transformer dynamically generates attention based on the input state, this architecture can be applied to suppress distractions on kinematics states without attention signal. In this architecture, each element of the sensory inputs (gaze position, left arm state, and right arm state) as well as the foveated image embedding is input to the Transformer to determine which element should be paid attention. This attention makes the output policy robust to unnecessary distractions in the sensory inputs.

The proposed Transformer-based deep imitation learning architecture was evaluated on an uncoordinated manipulation task (two arms executing different tasks), a goal-coordinated manipulation task (both arms solving the same task but not physically interacting with each other), and bimanual tasks (both arms physically interacting to solve the task) [4]. The results demonstrate that the Transformer-based self-attention mechanism improves dual-arm manipulation.

II. Related work

A. Imitation learning-based dual-arm manipulation

Previous imitation learning approaches to dual-arm manipulation mainly focused on the acquisition of subtask structures. Reference [8] studied the imitation learning of dual-arm manipulation tasks based on keypoints selected by hidden Markov models (HMM) to reproduce the movements of arms demonstrated by the human demonstrator. An imitation learning framework was proposed by [9] that segments motion primitives and learns task structure from segments for dual-arm structured tasks in simulation. The proposed framework was tested on a pizza preparation scenario, which is an uncoordinated manipulation task. Finally, [10] designed a deep imitation learning model that captures relational information in dual-arm manipulation tasks to improve bimanual manipulation tasks in the simulated environment, but their work required manually defined task primitives. To the best of our knowledge, the performance of a self-attention-based deep imitation learning method for dual-arm manipulation has not been studied in a real-world robot environment.

B. Transformer-based robot learning

The Transformer architecture has been used in robot learning for vision [19] in a simulated robot environment to bridge the domain gaps, such as different morphologies, physical appearances, or locations, between the demonstrator and the target robot. In addition, [20] used the Transformer-based seq-to-seq architecture to improve meta-imitation learning. Here, they used the Transformer to capture temporal correspondences between the demonstration and the target task.

However, these studies did not apply the self-attention architecture to robot kinematics states. The robot kinematics states provide essential information which cannot be captured by a gaze-based visual attention system, which reduces visual distraction by masking out peripheral vision. Therefore, our work aims to design a Transformer-based architecture to determine the attention to multiple sensory inputs in the current state for robust output against the distractions caused by the increased number of robot kinematics states.

III. Method

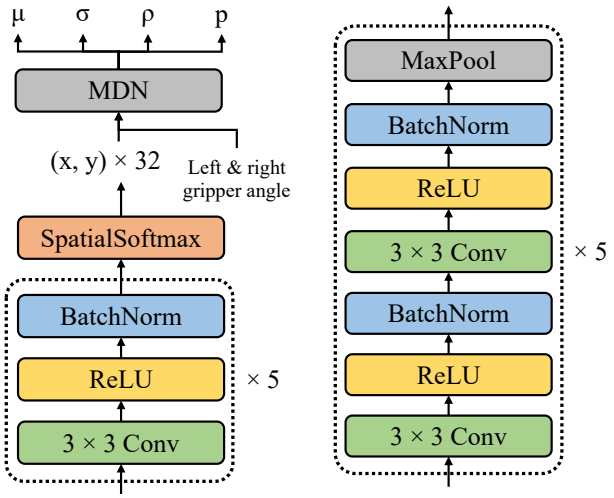
A. Robot system

We use a dual-arm robot system designed for teleoperation and imitation learning [3], [11]. While a human operator teleoperates two UR5 (Universal Robots) manipulators, a head-mounted display (HMD) provides vision captured from a stereo camera mounted on the robot. During teleoperation, the human gaze is measured by an eye-tracker mounted in the HMD. In this research, the left camera image is resized into 256×256 (called the global image) and recorded at 10 Hz with the two-dimensional gaze coordinate of the left eye and robot kinematics states of both arms. Each robot kinematics state is defined as a ten-dimensional vector that represents the end-effector position (three dimensions), orientation (six dimensions, represented using a combination of cosines and sines of three-dimensional Euler angles to prevent drastic changes of states when the angle exceeds 2π), and the gripper angle (one dimension) for each arm.

B. Gaze position prediction

In [11], human gaze was used to achieve imitation learning of a robot manipulator that is robust against visual distractions. This approach first predicts gaze position and crops the images around the predicted gaze position to remove task-irrelevant visual distractions, such as objects unseen during training. It was proved that such a gaze mechanism can improve the manipulation of multi-object tasks.

Like [11], the current study also uses a mixture density network (MDN) [21], which is a neural network architecture that fits a Gaussian mixture model (GMM) into the target, for estimating the probability distribution of the gaze position. The gaze predictor (Fig. 2a) inputs the entire $256 \times 256 \times 3$ RGB image and outputs $\mu \in \mathbf{R}^{2\times N}, \sigma \in \mathbf{R}^{2\times N}, \rho \in \mathbf{R}^{1\times N}, p \in \mathbf{R}^{N}$, which comprises the probability distribution of a two-dimensional gaze coordinate location where N



(a) Gaze predictor (b) Convolution block

Fig. 2: Neural network architectures.

is the number of Gaussian distributions that compose the GMM. We used $N = 8$ throughout this research. Because the gripper state may be informative for gaze prediction on subtask transitions (e.g., the gaze shifts to a toy orange as soon as the left gripper has closed to pick up a toy apple, on *Pick* task), gripper angles are added. This network is trained by minimizing the negative log-likelihood of the probability distribution with the measured human gaze as target e as follows:

$$\mathcal{L}_{gaze} = -log\Big(\sum_{i=1}^{N} p^i \mathcal{N}(e; \mu^i, \mathbf{\Sigma}^i)\Big), \tag{1}$$

where $\mathbf{\Sigma}^i = \begin{bmatrix} {\sigma_x^i}^2 & \rho^i \sigma_x^i \sigma_y^i \\ \rho^i \sigma_x^i \sigma_y^i & {\sigma_y^i}^2 \end{bmatrix}$ represents the covariance matrix.

During the test, a single gaze position is selected from the mean location (μ) with the highest probability of the inferred GMM:

$$gaze = \mu^{arg\,max(p^i)}$$

C. Transformer-based dual-arm imitation learning

The proposed architecture for dual-arm manipulation is presented in Fig. 1. First, the gaze predictor (III-B) predicts two-dimensional gaze position from the entire 256×256 image. The 64×64 foveated image, cropped according to the predicted gaze position, passes through five convolution blocks (Fig. 2b) with channel sizes of $[8, 16, 16, 32, 64]$ in that order. The resultant feature passes through the global average pooling (GAP) layer [22], which averages each feature map into one value to reduce the number of parameters.

The predicted gaze position and left/right robot manipulator states are concatenated into a 22-dimensional state (Fig. 1). In the state representation, $[x_{gaze}, y_{gaze}]$ indicates gaze position, $[x_l, y_l, z_l, cos(\alpha_l), sin(\alpha_l), cos(\beta_l), sin(\beta_l), \cos(\gamma_l), \sin(\gamma_l), g_l]$ indicates left robot state and $[x_r, y_r, z_r, cos(\alpha_r), sin(\alpha_r), cos(\beta_r), sin(\beta_r), \cos(\gamma_r), \sin(\gamma_r), g_r]$ indicates right robot state, where α, β, γ represents the Euler angle and g indicates gripper angle.

In previous researches, the Transformer architecture was used to learn dependencies between word embeddings or embeddings of image patches [12], [15]. In this work, to learn dependencies between each element of the robot state, robot states were separated into scalar values. And we added a 22-dimensional one-hot vector as positional embedding to each value. Each (1+22)-dimensional state element with a one-hot vector that represents position passes through a linear projection to generate a 64-dimensional state representation with the learned position embeddings. Then, the image embedding and state representations are concatenated and encoded using the three layers of the Transformer encoder (see Fig 1). We followed the Transformer encoder architecture described in [12].

Finally, the encoded feature is flattened and passes through an MLP with one hidden layer to predict the actions for both arms. An action is defined as the difference between the next kinematics state and the current kinematics state: $a_t = s_{t+1} - s_t$, where s_t represents the state vector at time step t and a_t represents the action at time step t. Unlike the angles in the state representation, which are represented by a combination of $cos(\theta)$ and $sin(\theta)$, where θ is an Euler angle, the angles on the action are represented by the differences in a three-dimensional Euler angle.

The gripper is controlled by the last element of the predicted action. However, this element only predicts the angle of the gripper and does not provide enough force to grasp any object. Therefore, a binary signal for the gripper open/close command is also predicted. If this binary signal predicts that the gripper should be closed, the gripper command additionally tries to close to $5°$ to provide enough force to grasp objects.

We used the same loss function first proposed in [2] and used in [3], [11] to optimize the policy network.

IV. Experiments

A. Task setup

We selected dual-arm manipulation tasks that include uncoordinated, goal-coordinated, and bimanual dual-arm manipulation tasks.

- *Pick*: In this task, the robot has to pick up the toy apple with its left end-effector and then pick up the toy orange, which is always placed on the right side of the apple, with its right end-effector. This task evaluates uncoordinated manipulation because picking up the apple and orange are mutually independent. The apple and orange are placed randomly on the table within reach of the robot arms.
- *BoxPush*: This task evaluates the bimanual manipulation. In this task, the robot has to push the box to the target location using both arms.
- *ChangeHands*: This task evaluates whether the robot can accomplish accurate bimanual manipulation that

Dataset	Number of demos	Total demo time (min)
Pick	1,030	85.4
BoxPush	460	53.8
ChangeHands	256	25.1
KnotTying		262.8 in total
PickUp	1,858	77.1
Grasp	2,251	98.9
PullOut	607	16.5
ReleaseAndPick	1,950	70.3

TABLE I: Training set statistics. To reduce the preparation time required for *KnotTying*, the task was segmented into subtasks (*PickUp*, *Grasp*, *PullOut*, and *ReleaseAndPick*), and demonstrations were separately recorded for each subtask.

involves grasping. The robot has to first grasp the toy banana with its left arm, stand it up, regrasp it in its right hand, and finally flip the banana. This regrasping behavior requires accurate prediction of the grasping position of the right hand based on the position of the left hand.

- *KnotTying*: In this task, the robot manipulators attempt to tie a knot, which is a complicated goal-coordinated manipulation task that involves series of subtasks considering the geometric shape of the deformable knot. The knot is placed in the shape of an α. The robot has to pick up the center of the crossed section with its right hand (*PickUp*), appropriately guide its left hand inside the ring while avoiding collision and grasp the end of the knot (*Grasp*), pull it out of the ring (*PullOut*), release the gripper and pick up the other end of the knot with the right hand (*ReleaseAndPick*), and finally tie the knot (which is a simple programmed behavior of moving both arms in opposite directions).

The recorded demonstrations were divided into training set (90%) and validation set (10%). The training set statistics of each task are represented in Table I. During the test, the target object was located as close as possible to the initial position of the target recorded on the randomly shuffled validation set images. For *KnotTying*, the initial position images were manually chosen to avoid too complicated knot placements.

The size of the MLP hidden layer is 200 for *Pick*, *BoxPush* and *ChangeHands*, and 400 for *KnotTying* because *KnotTying* requires a more complicated policy. The models were trained with distributed training supported by NVIDIA Apex with a Xeon CPU E5-2698 v4 and eight V100 GPUs, using a batch size of 64 per GPU. During testing on the real robot system, Intel CPU Core i7-8700K and one NVIDIA GeForce GTX 1080 Ti were used. We used a learning rate of 10^{-5} and the rectified Adam (RAdam) optimizer [23].

B. Baselines

We compared the proposed model with two baseline models without the Transformer encoder. The first baseline model (baseline-GAP) retains the GAP layer but replaces each Transformer encoder layer with the one fully connected layer. Therefore, in this model, the foveated image is processed with a convolution block; averaged by the GAP layer; concatenated with the gaze position, the left state, and the right state; and processed by five fully connected layers to output the action. The second baseline model (baseline) does not include GAP for image processing nor the Transformer encoder. In this model the foveated image is processed with a convolution block; flattened and concatenated with the gaze position, the left state, and the right state; and then processed with the MLP with one hidden layer to finally compute the action output.

The sizes of the hidden layer of both models were adjusted so that the total number of parameters is similar to the proposed Transformer-based architecture.

Metric	Baseline	Transformer-based
Orientation error ($^\circ$)	9.57	**1.99**
Top-left position error (mm)	35.7	**21.3**
Top-rihgt position error (mm)	36.4	**12.2**

TABLE II: *BoxPush* evaluation results (median of 24 trials).

C. Performance evaluation

The performance of the Transformer-based method and baselines were evaluated for all tasks. Each method was tested on 24 different initial positions sampled from the validation set for *Pick*, *BoxPush*, *ChangeHands* and 12 different initial positions for *KnotTying*. In task *Pick*, the Transformer-based method, baseline, and baseline-GAP models attempted to first pick up the toy apple with the left arm and then pick up the toy orange with the right arm (Fig. 3). The result indicates that the Transformer-based method outperforms both baseline methods in terms of success rates of picking up each object (Fig. 8). Because the baseline-GAP performed the worst among the three models, it was excluded from further experiments.

In *BoxPush*, the Transformer-based method and the baseline were compared (Fig. 4). For quantitative evaluation, the final positions of the top-left corner and top-right corner of the box after the robot executed the task were measured and the Euclidean distance to the ideal target position of both corners was calculated. Then, the orientation error, which measures how much the box is tilted with respect to the ideal target orientation (approximately 0°), was calculated based on the two measured positions. The result (Table. II) indicates that the Transformer-based method outperforms the baseline.

In *ChangeHands*, the success rates of each subtask: *Grasp* (left hand), *StandUp* (left hand), *Regrasp* (right hand), and *Flip* (right hand), were evaluated (Fig. 5), with models trained with two different random seeds (0, 1). Fig. 9 shows that the Transformer-based method achieved a high success rate. The result also confirms that the Transformer-based model outperforms the baseline regardless of the random seed.

In *KnotTying*, the success rates of subtasks (*PickUp*, *Grasp*, *PullOut*, and *ReleaseAndPick*) were evalulated (Fig. 6). The Transformer-based method recorded higher success rate than the baseline (Fig. 10). Fig. 7 describes typical

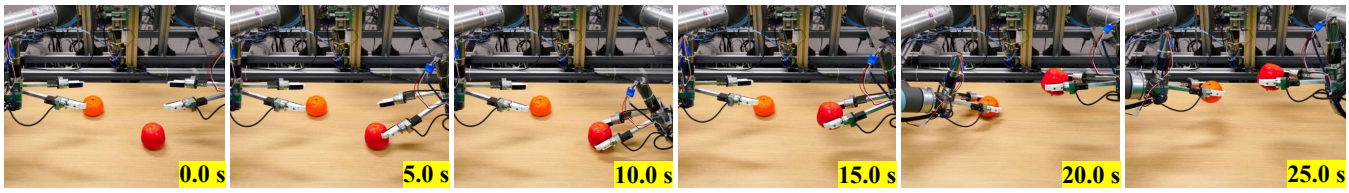


Fig. 3: Example of the proposed method on *Pick*. The robot first picked the toy apple ($\sim$ 10.0s), lifted it up (15.0s), and picked up the orange (25.0s).

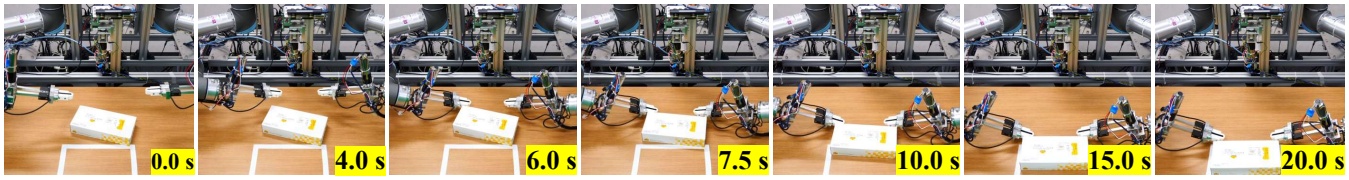


Fig. 4: Example of the proposed method on *BoxPush*. The robot placed its both arms behind the box ($\sim$ 6.0s), pushed it with the right arm (7.5s), and moved it with both arms to the goal position ($\sim$ 20.0s).

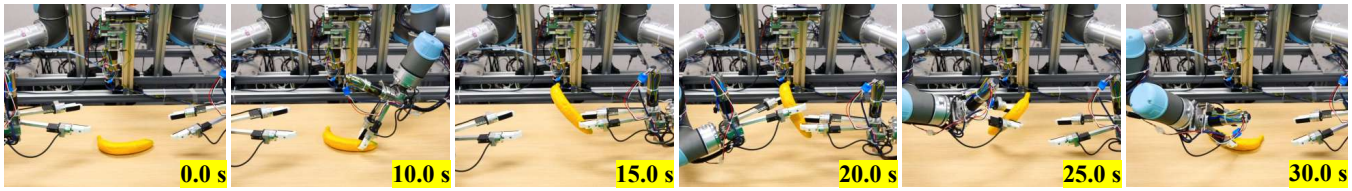


Fig. 5: Example of the proposed method on *ChangeHands*. The robot first grasped the toy banana with its left hand ($\sim$ 10.0s), standed it up (15.0s), regrasped it with the right hand (20.0s $\sim$ 25.0s), and finally flipped it (30s).

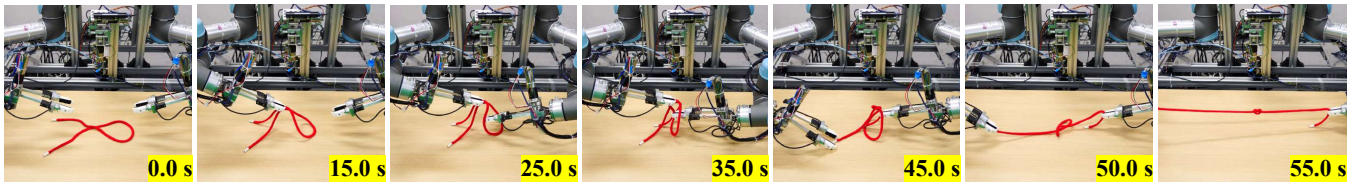
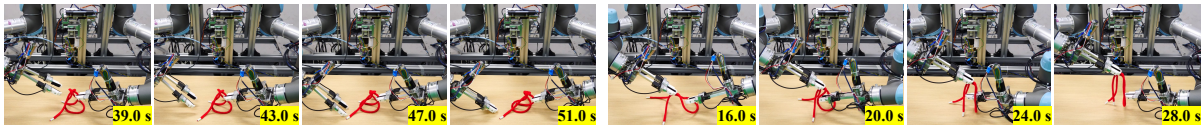


Fig. 6: Example of the proposed method on *KnotTying*. The robot picked up the knot with its right arm ($\sim$ 15.0s), tried to grasp the end of the knot with left arm (25.0s), pull it (35.0s), release the right gripper and grasp the the other end of the knot ($\sim$ 50.0s), and finally tie it by stretching both arms (55.0s, programmed behavior).



(a) Failure at *ReleaseAndPick* (b) Failure at *Grasp*

Fig. 7: Failure examples of the proposed method on *KnotTying*. Typical failure was caused by mistakes on reaching to a target location of the knot.

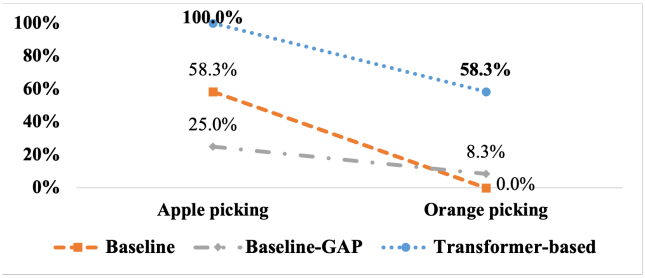


Fig. 8: Success rate on *Pick* (24 trials).

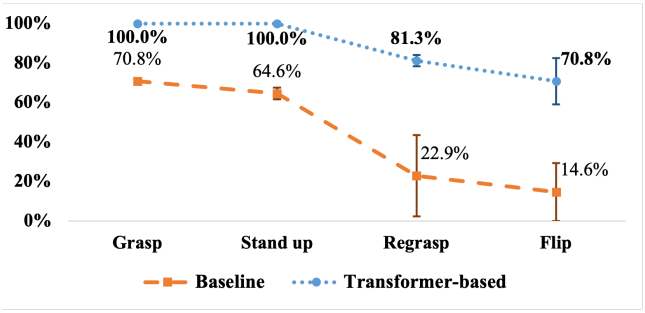


Fig. 9: Success rate on *ChangeHands* (24 trials).

mistakes of the proposed method on *KnotTying*. Among the total of 12 trials, we observed 1 failure at *PickUp*, 3 failures at *Grasp*, and 3 failures at *ReleaseAndPick*.

We have found that imitating a releasing behavior during *ReleaseAndPick* was not successful, probably because the exact release timing is not strongly related to current sensory input. Rather, the human operator can release the gripper anytime after the subtask *PullOut* is completed, causing the release signal to be diluted. Because our purpose is to adapt self-attention to current sensory inputs, we programmed the release behavior defined by opening the right hand for only one time if both hands are closed and the right hand is on the right side of the left hand with some margin.

D. Attention weight assessment

To determine which sensory input the Transformer attends, we investigated the attention weights for each sensory input. First, attention rollout [24], which is a recursive multiplication of the attention weights of all Transformer layers averaged over all attention heads, was computed. Next, the

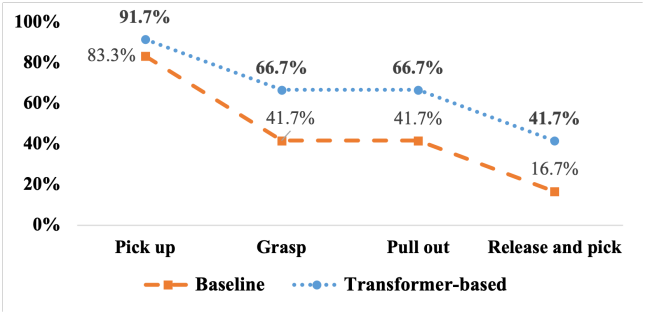


Fig. 10: Success rate on *KnotTying* (12 trials).

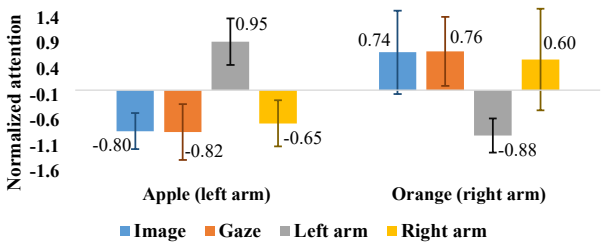


Fig. 11: Attention for the subtasks in *Pick*.

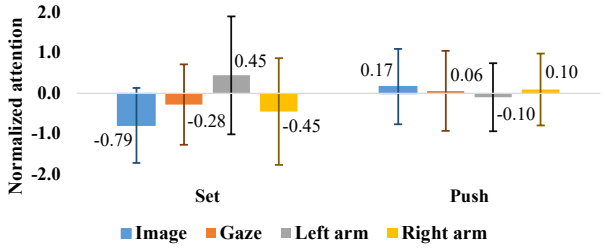


Fig. 12: Attention for the subtasks in *BoxPush*.

attention value for sensory inputs belong to the foveated image, gaze position, left arm state, and right arm state was calculated from 23×23 attention rollout as follows:

$$W^{Image} = \sum_{i=1}^{23} A_{i1} \tag{2}$$

$$W^{Gaze} = \sum_{i=1}^{23} \sum_{j=2}^{3} A_{ij} \tag{3}$$

$$W^{Left} = \sum_{i=1}^{23} \sum_{j=4}^{13} A_{ij} \tag{4}$$

$$W^{Right} = \sum_{i=1}^{23} \sum_{j=14}^{23} A_{ij}, \tag{5}$$

where A represents the attention rollout map in which the columns represent queries and the rows represent values, and W refers to attention value for each input domain. Then, each time series attention values in each input domain (*Image*,

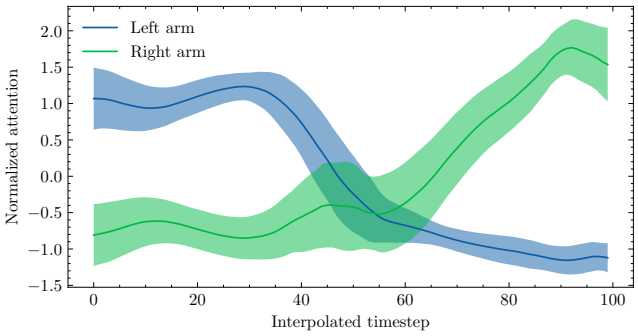
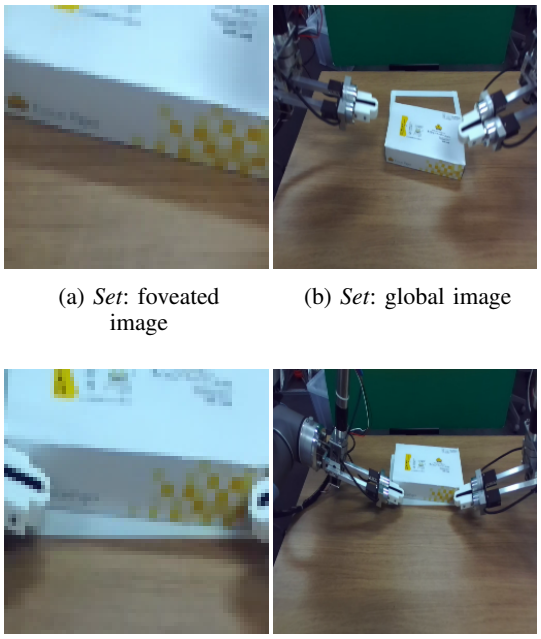


Fig. 13: Time series of attention for *Pick*.



(a) *Set*: foveated image (b) *Set*: global image

(c) *Push*: foveated image (d) *Push*: global image

Fig. 14: Sample of *Set* and *Push*. In *Push*, the robot can infer the action using the foveated image because it includes both end-effectors.

Gaze, *Left*, *Right*) was normalized because we want to see the change of attention values on each input domain:

$$W'_t = \frac{W_t - \mu_W}{\sigma_W}, \tag{6}$$

for $t \in [0, 1, ..., T]$ in each trial episode. μ_W and σ_W refers mean and standard deviation of time series attention values $[W_0, W_1, ..., W_T]$.

Fig. 11 presents the normalized attention, on the validation sets of *Pick*, divided into the subtasks of apple-picking (using the left arm) and orange-picking (using the right arm). This shows that when using the left arm (apple-picking), the Transformer architecture pays more attention to the states of the left arm, and vice versa. Then, the time series of the normalized attention on left/right arm was visualized for this task (Fig. 13). Because the length of each episode is different, we interpolated the sequence of attention into a length of 100. The result shows the attention to the left arm decreases as subtask shifts from picking of apple (left) to orange (right), while the attention to the right arm increases.

In the same way, we visualized the normalized attention for each type of sensory information on *BoxPush*. Fig. 12 visualizes the attention on *BoxPush*, where each episode into *Set*, the behavior of placing the arms behind the box ($\sim$ 6.0s in Fig. 4), and *Push*, in which the robot actually pushes the box into the goal (6.0s$\sim$ in Fig. 4). In the *Set* subtask, the Transformer attended more to the left arm and right arm states than the image embedding. In contrast, in the *Push* subtask, it attended to the image embedding because the foveated image alone is sufficient information for moving the box to the goal position (Fig. 14).

V. Discussion

We proposed the Transformer-based self-attention mechanism for deep imitation learning on real robot dual-arm manipulation tasks. Because the self-attention mechanism masks out sensory input which is not related to the current task, this suppresses distractions on sensory input. The experiments demonstrated that the proposed Transformer-based method substantially improves uncoordinated (*Pick*), goal-coordinated (*KnotTying*), and bimanual (*BoxPush, ChangeHands*) dual-arm manipulation performance over the baseline without the Transformer. The analysis on the attention rollout revealed that the Transformer can attend to the appropriate sensory input.

Our Transformer-based deep imitation learning architecture is not specialized for dual arms but instead can be expanded to more complicated robots such as multi-arm robots or humanoid robots by concatenating more sensory information into the state representation. In the same way, it would be interesting to investigate whether additional sensory information such as tactile or sound can be integrated using our proposed method, which remains as future work.

In our experiment, closed-chain bimanual manipulation tasks such as moving heavy objects using both arms were not tested. In our teleoperation setup without force feedback, the counterforce from the object is not transferred back to the human, causing failure while teleoperating the closed-chain manipulation tasks. To solve this issue in the future, a bilateral system with force feedback may be required to enable a human to correctly control the system.

References

[1] P.-C. Yang, K. Sasaki, K. Suzuki, K. Kase, S. Sugano, and T. Ogata, "Repeatable folding task by humanoid robot worker using deep learning," *Robotics and Automation Letters*, vol. 2, no. 2, pp. 397–403, 2016.

[2] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *International Conference on Robotics and Automation*, 2018, pp. 1–8.

[3] H. Kim, Y. Ohmura, and Y. Kuniyoshi, "Gaze-based dual resolution deep imitation learning for high-precision dexterous robot manipulation," *Robotics and Automation Letters*, pp. 1–1, 2021.

[4] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic, "Dual arm manipulation—a survey," *Robotics and Autonomous systems*, vol. 60, no. 10, pp. 1340–1353, 2012.

[5] R. Suárez, J. Rosell, and N. Garcia, "Using synergies in dual-arm manipulation tasks," in *International Conference on Robotics and Automation*, 2015, pp. 5655–5661.

[6] J. Zhu, B. Navarro, P. Fraisse, A. Crosnier, and A. Cherubini, "Dual-arm robotic manipulation of flexible cables," in *International Conference on Intelligent Robots and Systems*, 2018, pp. 479–484.

[7] D. SepúLveda, R. Fernández, E. Navas, M. Armada, and P. González-De-Santos, "Robotic aubergine harvesting using dual-arm manipulation," *IEEE Access*, vol. 8, pp. 121 889–121 904, 2020.

[8] T. Asfour, P. Azad, F. Gyarfas, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," *International Journal of Humanoid Robotics*, vol. 5, no. 02, pp. 183–202, 2008.

[9] R. Caccavale, M. Saveriano, G. A. Fontanelli, F. Ficuciello, D. Lee, and A. Finzi, "Imitation learning and attentional supervision of dual-arm structured tasks," in *International Conference on Development and Learning and Epigenetic Robotics*, 2017, pp. 66–71.

[10] F. Xie, A. Chowdhury, M. Kaluza, L. Zhao, L. L. Wong, and R. Yu, "Deep imitation learning for bimanual robotic manipulation," in *Neural Information Processing Systems*, 2020.
[11] H. Kim, Y. Ohmura, and Y. Kuniyoshi, "Using human gaze to improve robustness against irrelevant objects in robot manipulation tasks," *Robotics and Automation Letters*, vol. 5, no. 3, pp. 4415–4422, 2020.
[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Neural Information Processing Systems*, 2017, pp. 5998–6008.
[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
[14] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.
[15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.
[16] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, 2020, pp. 213–229.
[17] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," *arXiv preprint arXiv:2012.09841*, 2020.
[18] L. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit, "One model to learn them all," *arXiv preprint arXiv:1706.05137*, 2017.
[19] S. Dasari and A. Gupta, "Transformers for one-shot visual imitation," in *Conference on Robot Learning*, 2020.
[20] T. Cachet, J. Perez, and S. Kim, "Transformer-based meta-Imitation learning for robotic manipulation," in *Neural Information Processing Systems, Workshop on Robot Learning*, 2020.
[21] C. M. Bishop, "Mixture density networks," Neural Computing Research Group, Aston University, Tech. Rep., 1994.
[22] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.
[23] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," in *International Conference on Learning Representations*, 2020, pp. 1–14.
[24] S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," in *Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 4190–4197.