Learning Interaction Regions and Motion Trajectories Simultaneously from Egocentric Demonstration Videos

Jianjia Xin[1] Lichun Wang[2] Kai Xu[3] Chao Yang[4] and Baocai Yin[5]

***Abstract*—Learning to interact with objects is significant for robots to integrate into human environments. When the interaction semantic is definite, manually guiding the manipulator is a commonly used method to teach robots how to interact with objects. However, the learning results are robot-dependent because the mechanical parameters are different for different robots, which means the learning process must be executed again. Moreover, during the manual guiding process, operators are responsible for recognizing the region being contacted and providing expert motion programming, which limits the robot's intelligence. To enhance the level of automation in object interaction for robots, this paper proposes IRMT-Net (Interaction Region and Motion Trajectory prediction Network) to predict the interaction region and motion trajectory simultaneously based on images. IRMT-Net achieves state-of-the-art interaction region prediction results on Epic-kitchens dataset, generates reasonable motion trajectories and can support robot interaction in actual situations.**

***Index Terms*—Computer Vision for Automation, Deep Learning for Visual Perception, Dataset for Robotic Vision.**

I. Introduction

Nowadays, robot service has become popular in various scenarios. When robots serve humans, such as pouring water or opening drawers to take goods, interacting with objects is essential. When robots interact with objects, simply knowing the interaction region of the object is insufficient for automatic execution. Taking "open drawer" as an example, if the robot only knows that the drawer handle can interact, it can move its manipulator to this position, but it will not be able to pull the drawer. Similarly, if the robot only knows the motion trajectory of "pull", the interaction still cannot be executed successfully because the robot needs to grasp the drawer handle beforehand to execute the interaction. Therefore, it is necessary to know both the object interaction region and motion trajectory of the manipulator simultaneously. The absence of either component would affect the automatic execution of interaction. Current research for teaching robot interaction mainly relies on manual guiding techniques [1], [2], which is robot-dependent. To reduce the dependence, some researchers learn the interaction region or motion trajectory from large scale demonstration videos [3], [4], [5], [6], which eliminates the need for manual operator movement and provides strong portability. However, the above methods focus on learning either the interaction region or the motion trajectory separately, which cannot meet the requirement of automatically executing the interaction based on the learning results.

The authors are with the Beijing Key Laboratory of Multimedia and Intelligent Software Technology, Beijing Artificial Intelligence Institute, Faculty of Information Technology, Beijing University of Technology, Beijing. 100124, China (e-mail: xinjianjia@emails.bjut.edu.cn; wanglc@bjut.edu.cn; xukai@emails.bjut.edu.cn; yangchaoyc@emails.bjut.edu.cn; ybc@bjut.edu.cn).*Corresponding author: Lichun Wang.*

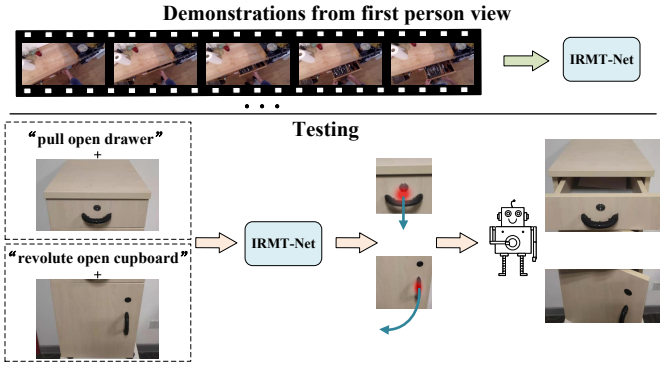

Fig. 1: Pipeline of the proposed method. In the demonstration stage, IRMT-Net is trained based on egocentric videos labeled with interaction class and motion trajectory. In the testing stage, IRMT-Net predicts the interaction regions (red area) and motion trajectories (blue curves) based on the object image and the interaction command ("pull open drawer" and "revolute open cupboard"). The robot follows the guidance information to carry out the interaction.

To support the automatic interaction of robots, we propose IRMT-Net (Interaction Region and Motion Trajectory prediction Network), which predicts both the interaction region and motion trajectory simultaneously. IRMT-Net is trained on egocentric videos labeled with supervised information about interaction class and motion trajectory. To meet the training requirements of IRMT-Net, we incrementally labeled motion trajectories for the videos in Epic-kitchens dataset [7] to obtain an extended dataset Epic-kitchens-tra. Using the trained IRMT-Net, the interaction region and motion trajectory are predicted based on the object's RGB image and interaction command. The interaction command is composed of motion and object category. Finally, the predicted interaction region and motion trajectory are sent to the robot to carry out the interaction. The pipeline of our method is shown in Fig. 1.

In summary, the contributions of this paper are as follows:

- We propose IRMT-Net that predicts the interaction region of an object and a corresponding motion trajectory of an action simultaneously by learning from egocentric demonstration videos.
- We incrementally annotate motion trajectories for the videos in the Epic-kitchens dataset to provide an extended dataset named Epic-kitchens-tra, which is available online[1].
- IRMT-Net achieves state-of-the-art results on the Epic-kitchens dataset, generates reasonable motion trajectories,

[1]https://drive.google.com/drive/folders/1iSHHwJg-DCOrlaqr0QQ0b0iwD-2v8q6B?usp=sharing

and supports robot interaction, as demonstrated through experiments in real-world scenarios.

II. Related Work

Learning from demonstration. Learning from demonstration (LfD) is a family of methods that learn necessary interaction knowledge by observing task executions, and apply the learned knowledge to reproduce the demonstrated behavior [8]. Manual demonstration [1], [2], [9] is a direct approach to teach robots interaction, but its algorithmic portability is limited. To reduce dependence on specific robot platforms, a number of studies use demonstration videos to learn interaction knowledge [3], [6], [10], [4], [5].

Interaction region generation. Some studies [11], [12], [13], [14], [15], [16] regard the interaction region prediction as visual affordance segmentation, which assigns action labels to each pixel [17], [18]. However, pixel-wise labeling is a laborious task, some weakly supervised methods [19], [20] have been proposed to address this issue. While segmentation methods can provide detailed results, the annotations are not always accurate since they are based on annotators' experience and imagination, and lack interactive information. To address these limitations, several methods learn interaction regions from interaction videos [3], [4], [5]. Nagarajan et al. [4] used LSTM and action labels to learn interaction regions from egocentric demonstration videos. Luo et al. [5] improved upon Nagarajan et al.'s work by adding hand positions as auxiliary information to regress the interaction region, achieving better results. However, in realistic situation, even when performing the same action, the interaction regions of different types of objects may be different, which is often overlooked in current methods.

Motion trajectory generation. Manually guiding the robotic manipulator through expert demonstrations from either a third or first-person perspective is a common method for learning motion trajectories. However, this approach is expensive and relies on robots, which limits scalability. In addition to manual guidance, some researchers use demonstration videos to generate motion trajectories. Duque et al. [10] used LfD in robot assembly task. The task demonstration is carried out by an expert, and the process is recorded by a Kinect RGB-D camera. Zorina et al. [6] developed an alignment procedure and focused on the tool trajectories in videos to learn how to manipulate them. Fang et al. [21] proposed a joint training method to predict the motion of the robot to complete tasks. In recent years, some researchers have focused on the interaction of 3D articulated objects [22], [23], [24], [25], aiming to predict the motion trajectory of these objects when they are pulled or pushed.

III. Problem Statement

Given an object image I and an interaction command, extract the object category c, motion category m and action class v from the interaction command. The aim is to simultaneously predict interaction region R of the object and motion trajectory J of the robot's manipulator. The interaction region R is a collection of pixels in the image I, indicating where a robot manipulator contacts the object. The motion trajectory J is a sequence of 3D coordinates, representing the positions where a robot's manipulator moves. Learning to predict R and J simultaneously means finding a function $f : (I, c, v, m) \longmapsto (R, J)$, which maps the input I, c, v and m to the output R and J.

TABLE I: Motions corresponding to the videos in Epic-kitchens-tra dataset.

action	object	motion	motion description	num.
open	drawer	pull open	pull from far to near	1
	cupboard, fridge, microwave	revolute open	pull rotate around the axis	3
	tap, box	lift open	move from down to up	2
	container, bottle, jar	rotate open	rotate counterclockwise	3
close	drawer	push close	push from near to far	1
	cupboard, fridge, microwave	revolute close	rotate around the axis	3
	tap, box	press close	move from up to down	2
	container, bottle, jar	rotate close	rotate clockwise	3
take	plate, knife, spoon	take	take from far to near	3
put	plate, knife, spoon	put	place from near to far	3
throw	box, lid, bottle	throw	put from near to far	3
turn-off	hob, tap, microwave	turn-off	rotate counterclockwise	3
turn-on	tap, hob, microwave	turn-on	rotate clockwise to open	3
turn	tap, kettle, lid	turn	turn clockwise	3
cut	carrot, sausage	cut	move back and forth	2
wash	plate, pan, knife	wash	swipe left and right	3
dry	knife, plate, pan	dry	swipe back and forth	3
mix	pan, bowl, cup	mix	stir clockwise several times	3
adjust	hob, cup	adjust	push out slightly	2
empty	pan, bowl, bottle,	empty	dump the objects inside	3
pour	bottle, cup, pan	pour	dump the objects inside	3
shake	jar, bottle, cup	shake	move up and down quickly	3
peel	carrot, sausage	peel	remove the skin near to far	2
scoop	bowl	scoop	scoop the contents	1
move	pan, bottle, bowl	move	move from left to right	3
remove	pan, plate, carrot	remove	move from left to right	3

IV. Egocentric Motion Trajectory Extraction

To predict the interaction region and motion trajectory simultaneously, we propose a deep neural network that maps input (I, c, v, m) to output (R, J). To evaluate this approach, we incrementally annotate Epic-kitchens dataset [7], [4] to create Epic-kitchens-tra dataset. Epic-kitchens videos are recorded in RGB format from the actor's first-person perspective using a head-mounted GoPro. However, the existing action labels and object categories do not support learning trajectory knowledge, so we incrementally annotate motion trajectories. Inspired by the motion trajectory extracting method in assembly task [10], we use a RGB-D camera mounted on the robot to record the first-person interaction videos, ensuring perspective consistency with the actor. Then, we extract motion trajectories and incrementally annotate them in the Epic-kitchens dataset.

A. Motion classification for Epic-kitchens dataset

In the interaction process, a certain action may correspond to different motion trajectories when performed on different types of objects. For example, "open drawer" means pulling the drawer horizontally, while "open fridge" means rotating the refrigerator door around an axis. Therefore, we define motion category for the interaction videos in Epic-kitchens [7], [4] dataset based on the movement corresponding to actions, as shown in Table I. The first column of Table I shows

action categories,while the second and the third column show the categories of object and motion, respectively. The fourth column describes the motions. The fifth column denotes the number of trajectories demonstrated for each motions. In Table I, the actions "open" and "close" are matched with four types of motions, respectively. We then perform the corresponding interactions and record with RGB-D camera from the robot's first perspective. The recorded RGB-D videos are used for extracting motion trajectories.

B. Motion trajectory extraction and preprocessing

For the recorded interaction videos, we extract motion trajectory using the method for assembly tasks [10] as depicted in Fig. 2. For each frame, we detect the human hand using a hand detection method [26] and mark the results with a red bounding box, which contains a yellow dot indicating the center coordinate of the bounding box. Then, we extract the corresponding depth data for each center coordinate.

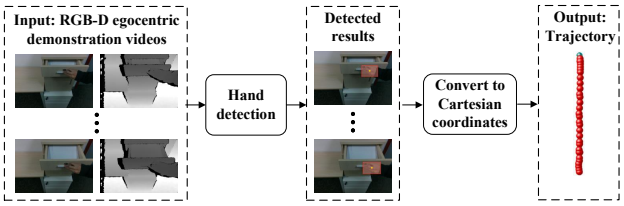

Fig. 2: Pipeline of the motion trajectory extraction. Taking "pull open" as an example, detect hand in RGB-D interaction video and convert the center of the bounding box into robot Cartesian coordinates to obtain the motion trajectory during the interaction process.

To eliminate noise, we firstly remove the center points located in the camera's blind area based on depth data. Additionally, centers with a depth variation above 10% at time step t with respect to the previous time step $t-1$ are also removed. Secondly, we apply MAF (moving average filter) to smooth the trajectory, using a window width of 5. Each trajectory is normalized to 31 points, and each point is represented by 3D coordinates in the Cartesian coordinate system of the robot. To eliminate the influence of robot parameters, we subtract the starting coordinate from each 3D coordinate in the trajectory. Fig. 3 shows four typical motion trajectories.

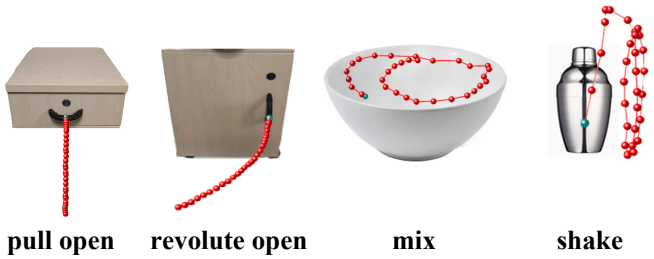

Fig. 3: Visualization of four motion trajectories, the green point is the first point of the trajectory.

According to the statistics in Table I, the obtained motion trajectories are incrementally labeled to Epic-kitchens dataset. The resulting dataset is named Epic-kitchens-tra and used as the training data for IRMT-Net.

V. Interaction Region and Motion Trajectory Prediction Network

To support robot automatic interaction, we propose IRMT-Net which predicts interaction region and motion trajectory simultaneously.

A. Video action classification

As shown in Fig. 4, the input of video action classification branch are video frames $[f_1, \ldots, f_T]$, where T is the number of video frames. The frames are first input to a backbone to extract visual features $[b_1, \ldots, b_T]$, with each feature having a dimension of $n \times n \times d$. L2-pooling is then applied on $[b_1, \ldots, b_T]$ and get $[l_1, \ldots, l_T]$. The dimension of l_t is $1 \times d$. Since video frames have sequential information, we use a transformer encoder [27] to process the sequential information and get $Q = [q_1, \ldots, q_T]$. The dimension of q_t is $1 \times d$. The output features of transformer are aggregated through a fully connected layer (Fc_e) to obtain the interaction features of the video:

$$h_v = g\left(W_e Q\right) \tag{1}$$

W_e represents the parameters of Fc_e, with a dimension of $1 \times T$, and $g(\cdot)$ denotes an activation function. The dimension of the output feature vector h_v is $1 \times d$. Then, Fc_v (fully connected layer) is used to classify h_v to predict the action category:

$$y_v = g(h_v W_v) \tag{2}$$

W_v represents the parameters of Fc_v. Cross entropy is used to calculate the action classification $loss_v$.

B. Interaction region prediction

During training process, the input of the interaction region prediction branch is a pair of images, which are static object image that does not show any interaction. If the object in the image has the same category as the object in the video processed by the video action classification branch, the image is a positive sample I_p, otherwise, it is a negative sample I_n.

Firstly, IRMT-Net extracts visual features o_p and o_n for positive and negative samples using the backbone. Secondly, IRMT-Net uses the Projection and Fc_p (fully connected layer) modules to transform o_p and o_n into interactive state features, denoted as h_p and h_n, respectively. The Projection module consists of two convolutional layers. To ensure the interaction region prediction branch learns features that correspond to the interaction action, h_p is constrained to be close to the video feature h_v, and h_n is constrained to be different from h_v. This is achieved through margin loss:

$$loss_p = \max(0, \|h_v - h_p\|_2 - \|h_v - h_n\|_2 + margin) \tag{3}$$

$margin$ is a hyperparameter.

Considering that the interaction regions can vary depending on the objects involved, we also classify the objects participating in the interaction. This is done by using h_p as input

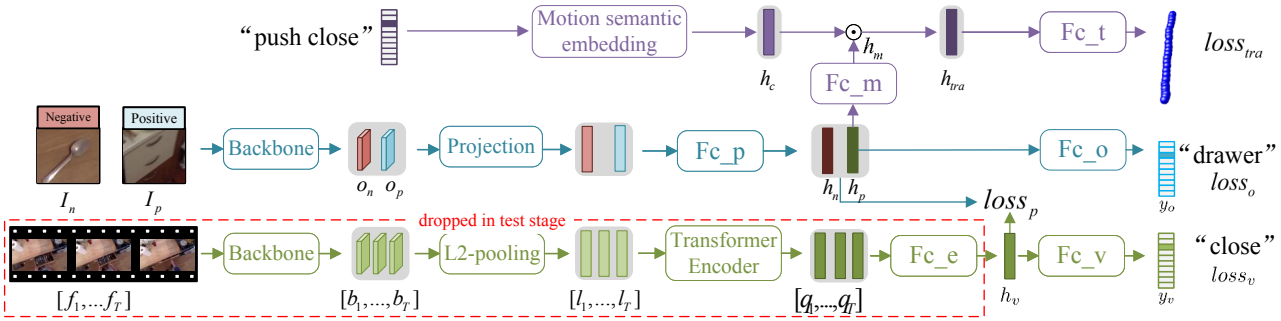

Fig. 4: The framework of IRMT-Net, including three branches, from bottom to top are video action classification (green arrows and modules), interaction region prediction (blue arrows and modules) and trajectory generation (purple arrows and modules) branches.

of Fc_o (fully connected layer) to obtain y_o for the positive sample. y_o represents the probability that the object belongs to each object category. We use cross entropy to calculate the object classification $loss_o$.

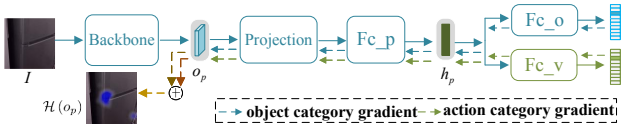

Fig. 5: Regression of interaction regions, calculating the object category gradient (blue dotted arrows) and the action category gradient (green dotted arrows), back propagating the gradients, summing the gradients and mapping the sum to the feature map o_p.

The process of generating interaction region R is shown in Fig. 5. The object image I is input to backbone to extract static image feature o_p. Then, o_p is input to Projection and Fc_p to extract the interaction feature h_p. h_p is then input to Fc_o and Fc_v to output y_o for object category and y_v for action category. We calculate the gradients for y_o and y_v respectively. Finally, we back-propagate the gradients (blue and green dotted arrows in Fig. 5). The heatmap $\mathcal{H}(o_p)$ is obtained by element-wise multiplication of the accumulated gradients and o_p.

$$\mathcal{H}(o_p) = \sum_k ReLU\left(\left(\frac{\partial y_v}{\partial o_p^k} + \frac{\partial y_o}{\partial o_p^k} \right) \odot o_p^k \right) \tag{4}$$

k is the index of channels. "$\odot$" is element-wise multiplication. Adjust heatmap $\mathcal{H}$ to the size of object image I to get the interaction region R.

C. Motion trajectory generation

The input of this motion trajectory generation branch is one-hot vector that describes motion category. The one-hot vector is input to the Motion semantic embedding module to obtain h_c. h_p is embedded into h_m through Fc_m layer. Then, we combine h_c and h_m to get a joint feature:

$$h_{tra} = h_c \odot h_m \tag{5}$$

h_{tra} is used to regress the coordinate values through Fc_t module which is composed of three fully connected layers. The loss function of coordinates regression is:

$$loss_{tra} = \frac{1}{M} \sum_{i=1}^{M} \|x_i - \hat{x}_i\|_2 + \|y_i - \hat{y}_i\|_2 + \|z_i - \hat{z}_i\|_2 \tag{6}$$

M is the number of points in the trajectory. $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$ is the coordinate of the i_{th} point in the ground truth trajectory, and (x_i, y_i, z_i) is the predicted coordinate of the i_{th} point in the generated trajectory.

The total loss of IRMT-Net is:

$$loss_{all} = \lambda_v loss_v + \lambda_p loss_p + \lambda_o loss_o + \lambda_{tra} loss_{tra} \tag{7}$$

λ_v, λ_p, λ_o and λ_{tra} are weight factors in loss function that are manually specified. The parameters of the neural network are optimized through backpropagation during training.

VI. Experiment

A. Experiment setup

Epic-kitchens-tra. The Epic-kitchens-tra is obtained by incrementally labeling motion trajectory for videos in the Epic-kitchens [4] dataset. The training set includes first-person videos of kitchen activities, covering 20 action, 31 object categories and 9236 interactive videos labeled with motion trajectories. The static object images are cropped based on bounding boxes from video frames before interaction occurs in the video clip. The test set includes 517 object images and 517 corresponding motion trajectories. When evaluating the interaction region prediction task, the data from Epic-kitchens-tra and Epic-kitchens are equivalent.

Evaluation Metrics. We use KLD (Kullback-Leibler Divergence) [28], SIM (similarity metric) [29] and AUC-J (AUC-Judd) [30] to evaluate interaction region generation. KLD measures difference between the prediction heatmap and the ground truth heatmap, and a lower score is better. SIM measures the similarity of two heatmaps and a higher score is better. AUC-J evaluates the classification performance of whether pixels allow interaction. RMS (root mean square) error is used to measure the quality of trajectory generation, and a lower score is better.

Implementation Details. IRMT-Net uses a pretrained ResNet-50 [31] as backbone. The number of frames T is set to 16 and the number of points M in the trajectory is set to 31. The dimension of b_t is $n \times n \times d$, where n = 28, d = 1024. The

TABLE II: Results of interaction region prediction on Epic-kitchens dataset. Bold results are ranked first, and italics are ranked second. "grad. from v" means that only the action category v is used to generate gradients during testing. "grad. from v and c" means that both the action category v and object category c are used to generate gradients during testing.

	KLD ↓	SIM ↑	AUC-J ↑
Center bias	10.66	0.222	0.634
Egogaze [33]	2.241	0.273	0.614
Mlnet [34]	6.116	0.318	0.746
DeepgazeII [35]	1.352	0.394	0.751
Salgan [36]	1.508	0.395	0.774
Hotspot [4]	1.258	0.404	0.785
HAG-Net [5]	*1.209*	0.414	*0.801*
IRMT-Net (grad. from v)	1.219	*0.419*	0.790
IRMT-Net (grad. from v and c)	**1.172**	**0.424**	**0.814**
Img2heatmap [4]	1.400	0.359	0.794

weight factors in $loss_{all}$ function are: λ_v=λ_p=λ_o=1, λ_{tra}=10. The $margin$ in $loss_p$ is set to 2. The batch size is 64 and learning rate is 0.0001. The Adam [32] is used to optimize the model parameters.

B. Interaction region prediction

IRMT-Net learns interaction region in a weakly supervised manner. We compared IRMT-Net with one naive (Center bias) method, four saliency detection methods, two weakly supervised interaction region prediction methods (Hotspot, HAG-Net) and one supervised method (Img2heatmap). All methods are briefly described as follows.

Center bias. This is a naive baseline method that generates a Gaussian heatmap at the center of an image.

SALIENCY. A kind of methods aim to detect the most saliency regions in an image, including Egogaze [33], Mlnet [34], DeepgazeII [35] and Salgan [36].

Hotspot [4]. A weakly supervised method that uses video action labels during training.

HAG-Net [5]. This method uses the position and action of the hand in interaction videos as aided information for object interaction region prediction.

IRMT-Net. IRMT-Net (grad. from v and c) uses action and object labels to generate gradients during testing. To be fair and consistent with Hotspot and HAG-Net, which only use action labels in the test phase, we test the performance when only using action labels to generate gradients, named IRMT-Net (grad. from v). IRMT-Net (grad. from v) removes $\frac{\partial y_o}{\partial o_p^k}$ when calculating $\mathcal{H}$ through equation (4).

Img2heatmap [4]. This is a simplified version of DEMO2VEC [3] that lacks video context during training.

The results are shown in Table II, IRMT-Net (grad. from v and c) achieves state-of-the-art result. It proves that the object class constraint is beneficial for predicting the interaction region. Fig. 6 shows the interaction regions generated by different methods. The interaction regions corresponding to five action classes are listed. Compared with other methods, the interaction regions predicted by IRMT-Net are closer to the ground truth. IRMT-Net (grad. from v and c) predicts more accurately due to considering the relationship between interaction region and object category.

TABLE III: Average RMS error of generated trajectory on Epic-kitchens-tra test set, RMS_t is RMS error on trajectory level and RMS_p is RMS error on point level. ResNet + LSTM means to use ResNet to extract visual features of object and predict motion trajectory through LSTM.

	Avg. RMS_t (cm)	Avg.RMS_p (cm)
ResNet + LSTM	63.86	2.06
IRMT-Net	62.62	2.02

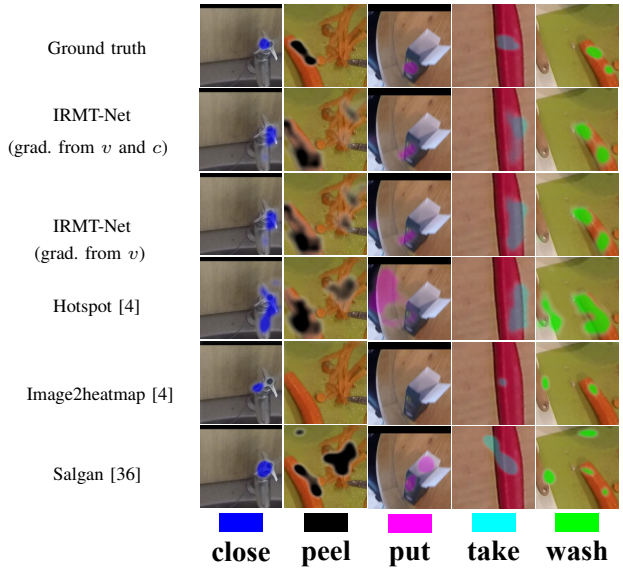

Fig. 6: Interaction regions on object images for five action classes, the prediction of IRMT-Net (grad. from v and c) is closer to ground truth than the other methods.

C. Motion trajectory prediction

To evaluate the performance of the model on the motion trajectory generation, we compute RMS error of trajectory level (RMS_t) and point level (RMS_p):

$$RMS_t = \sum_{i=1}^{M} \sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (z_i - \hat{z}_i)^2} \quad (8)$$

$$RMS_p = \frac{1}{M} RMS_t \quad (9)$$

M is the number of points in the trajectory. ($\hat{x}_i$, $\hat{y}_i$, $\hat{z}_i$) is the coordinate of the i_{th} point in the ground truth, and (x_i, y_i, z_i) is the predicted coordinate of the i_{th} point in the generated trajectory. As shown in Table III, we calculate the average RMS error to show the performance of trajectory generation. We compare IRMT-Net with LSTM which is commonly used to process sequence data [4]. To ensure fairness, the input of LSTM is the fusion of the object visual features extracted by ResNet and the motion category. The average RMS_p and average RMS_t of IRMT-Net are 2.02 cm and 62.62 cm, which are 0.04 cm and 1.24 cm lower than ResNet + LSTM. We compare the trajectories generated by ResNet+LSTM and

TABLE V: Average RMS_p of IRMT-Net for each motion category on Epic-kitchens-tra test set.

motion	pull open	revolute open	lift open	rotate open	push close	revolute close	press close	rotate close	take	put	throw	turn-off	turn-on
Avg. RMS_p (cm)	1.20	1.70	2.57	1.48	0.89	1.87	2.07	1.76	2.86	1.53	1.02	1.53	2.67
motion	turn	cut	wash	dry	mix	adjust	empty	pour	shake	peel	scoop	move	remove
Avg. RMS_p (cm)	2.52	1.42	1.75	1.25	2.57	2.53	4.60	4.00	6.10	2.11	2.40	1.02	1.92

TABLE IV: Average completion rate of "lift open" in a real environment. Two different distances (60 cm and 80 cm) from the robot to the object are set. We performed 10 times at each distance and recorded the number of successful attempts.

	lift open		Avg. task completion rate (%)
	60 cm	80 cm	
ResNet + LSTM	6/10	6/10	60.0
IRMT-Net	8/10	8/10	80.0

IRMT-Net methods in real-world scenarios to evaluate their performance in guiding interactions. To ensure fair comparison, we use the interaction regions (i.e., the starting points of interactions) generated by IRMT-Net. As shown in Table IV, we conduct experiments for the "lift open" motion at distances of 60 cm and 80 cm from the object, each with 10 trials. The completion rate of ResNet + LSTM is 60.0%, while IRMT-Net achieves a completion rate of 80.0%, indicating that IRMT-Net generates trajectories with better guiding performance. Relevant experimental videos are available online[2].

To evaluate the performance of trajectory generation in detail, we calculated the average RMS_p for each motion category, as shown in Table V, most of the motion trajectories are generated well. We visualize the predicted trajectories in Fig. 7. The red curve represents the ground truth, and the blue curve represents the prediction. It can be seen that the motion trajectory generated by IRMT-Net differs little from the ground truth.

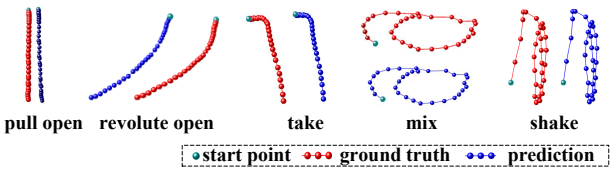

Fig. 7: Visualization of five motion trajectories predicted by IRMT-Net.

D. Ablation study

We studied the impact of different designs of IRMT-Net on interaction region prediction, including the video action classification branch, margin loss, object classification loss during the training stage, and back propagation of object category gradient during the test stage. The corresponding ablation results are shown in Table VI. During testing, comparing with only using action category gradient (grad. from v), using object class gradient (grad. from v and c) always improves performance. However, when only utilizing action category gradient, as shown in the bottom two rows of Table VI, the object classification loss during the training stage has insignificant impact. When the gradient during the testing phase is fixed, both video action classification branch and margin loss have a positive impact. This is because the former extracts features of the object in interactive state, while the later makes the features of the object participating in the interaction more discriminative.

[2]https://drive.google.com/drive/folders/1iSHHwJg-DCOrlaqr0QQ0b0iwD-2v8q6B?usp=sharing

TABLE VI: Ablation study. Influence of the video action classification branch, margin loss, object classification loss and object class gradient on the performance of interaction region prediction. "grad. from v and c" means that both the action category v and object category c are used to generate gradient during testing. "grad. from v" means that only the action category v is used to generate gradient during testing.

	Video action classification branch	Margin loss	Object classification loss	KLD ↓	SIM ↑	AUC-J ↑
grad. from v and c	✘	✔	✔	1.185	0.422	0.813
	✔	✘	✔	1.182	0.420	**0.819**
	✔	✔	✔	**1.172**	**0.424**	0.814
grad. from v	✘	✔	✔	1.233	0.412	0.790
	✔	✘	✔	1.251	0.404	0.788
	✔	✔	✘	**1.217**	0.417	**0.794**
	✔	✔	✔	1.219	**0.419**	0.790

TABLE VII: Average task completion rate of interaction in real environment. For each motion, two different distances from the robot to the object are set, four cases with different number of points in the interaction are set. For each combination of settings, each motion is tested three times.

Distance	Motion	Object	Number of points in the interaction				Avg. task completion rate(%)
			32	16	8	4	
60 cm	pull open	drawer	2/3	3/3	3/3	3/3	91.7
	revolute open	cupboard	2/3	2/3	2/3	3/3	75.0
	push close	drawer	2/3	3/3	3/3	3/3	91.7
	revolute close	cupboard	3/3	3/3	3/3	3/3	100.0
	take	cup	2/3	2/3	3/3	3/3	83.3
	mix	bowl	3/3	3/3	3/3	3/3	100.0
	shake	bottle	3/3	3/3	3/3	0/3	75.0
80 cm	pull open	drawer	2/3	3/3	3/3	3/3	91.7
	revolute open	cupboard	2/3	2/3	3/3	3/3	83.3
	push close	drawer	2/3	3/3	3/3	2/3	83.3
	revolute close	cupboard	3/3	3/3	3/3	3/3	100.0
	take	cup	3/3	3/3	3/3	3/3	100.0
	mix	bowl	3/3	3/3	3/3	3/3	100.0
	shake	bottle	3/3	3/3	3/3	0/3	75.0

E. Experiments of IRMT-Net in actual situation

To evaluate the performance of IRMT-Net for guiding robot interaction in real-world scenarios, we conducted tests on 7 representative motions, as shown in Table VII. The "pull open/close" and "revolute open/close" motions both correspond to the action class "open/close", but when the categories of interactive objects are different, the corresponding motion

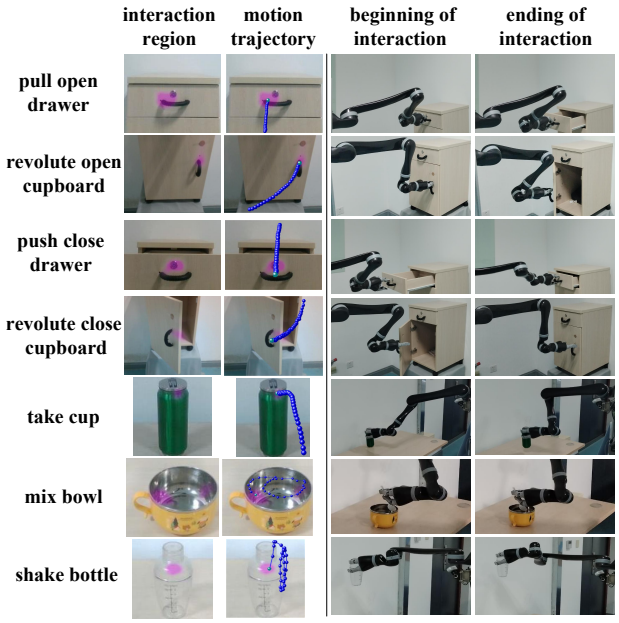

Fig. 8: Interaction with objects in real environments, IRMT-Net predicts the interaction region and motion trajectory to support the interaction of robots.

trajectories may be completely different. The "take" is a high-frequency operation that occurs after grasping an object, and the trajectories of "shake" and "mix" are more complex than other motions.

To test the robustness of the generated trajectories, we set two different distances from the robot to the object, 60 cm and 80 cm. Considering the different complexities of each type of motion, we also evaluate the impact on the successful rate when the number of points in the motion is different. The trajectory generated by IRMT-Net contains 31 points, combined with the starting point, it forms a complete interaction with 32 points. The starting point of the interaction is generated based on the interaction region. For the interaction region, the minimum enclosing bounding box is computed first, then the center of the bounding box is used as the starting point. For the interaction consisting of 16, 8 and 4 points, the points are uniformly sampled from 32 points.

The test process is repeated 3 times for each motion setting, and the number of successful interactions is recorded. For these types of motions, the manipulator pose needs to be preset. For example, when the gripper grasps the handle, the pose of the gripper is first set to be horizontal. The "revolute open cupboard" or "pull open drawer" interaction require that the corresponding objects have handles for grasping, while the "shake bottle" interaction requires the robot to grasp the bottle first.

The robot captures images through an Intel Realsense RGB-D camera mounted on it, and a pre-trained faster-RCNN [37] detector is used to detect objects to reduce background interference. The cropped image is obtained based on the bounding box with the highest detection score. To ensure the entire object is included in the cropped image, we crop a square area with the long edge of the bounding box as the length of the square's edge. Table VII shows that IRMT-Net achieves an average task completion rate of over 75% for the 7 motion categories, indicating that IRMT-Net is robust. Additionally, the success rate of interaction is affected by the number of points in the trajectory. For example, when there are only 4 points in the "shake bottle" interaction, the complete up and down shaking motion cannot be performed normally. Fig. 8 shows the interaction region, motion trajectory predicted by IRMT-Net and the actual interaction scenarios. More interaction videos are provided online[3].

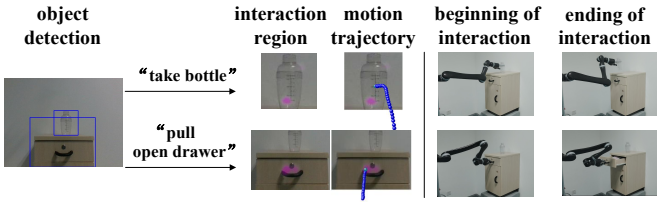

Fig. 9: There are two objects, drawer and bottle. The t-wo objects are detected, and then the interaction regions and trajectories are predicted through IRMT-Net according to different interaction commands. The robot completes the interaction based on the predicted information.

IRMT-Net is designed to predict the interaction region and motion trajectory for the case of interacting with a single object. When there are multiple objects in the environment, such as the scene depicted in Fig. 9, which contains two objects, we use a pre-trained faster-RCNN [37] to detect objects in the image, and crop them out. Using IRMT-Net, the robot can predict the interaction region and motion trajectory for each object according to the interaction command. Then the robot can choose to interact with a specific object or sequentially interact with objects in the scene.

VII. Conclusion and Discussion

This paper introduces IRMT-Net, a method for predicting interaction regions and motion trajectories to support automatic robot interaction. We also incrementally annotate the Epic-kitchens dataset and name it as Epic-kitchens-tra to provide supervised information for trajectory generation. IRMT-Net achieves state-of-the-art results for interaction region prediction on Epic-kitchens dataset and generates reasonable motion trajectories for realistic situations.

However, IRMT-Net has limitations in supporting uncommon objects. For example, the "revolute open cupboard" can only apply to the case that the rotating axis is vertical. Additionally, interactions that rely heavily on the rotation of mechanical claws, like "unscrewing" or "turning on the gas hob knob", are not well supported. Moreover, for different actions, we manually set the manipulator poses, which is not optimal. Automatic prediction of manipulator poses according to interactions is a future research direction.

[3]https://drive.google.com/drive/folders/1iSHHwJg-DCOrlaqr0QQ0b0iwD-2v8q6B?usp=sharing

ACKNOWLEDGEMENT

This research was supported by National Key R&D Program of China (No. 2021ZD0111902), NSFC (No. U21B2038, 62376014, 62172022), Foundation for China university Industry-university Research Innovation (No.2021JQR023).

REFERENCES

[1] Riccardo Caccavale, Matteo Saveriano, Alberto Finzi, and Dongheui Lee. Kinesthetic teaching and attentional supervision of structured tasks in human–robot interaction. *Autonomous Robots*, 43(6):1291–1307, 2019.

[2] Tesca Fitzgerald, Elaine Short, Ashok Goel, and Andrea Thomaz. Human-guided trajectory adaptation for tool transfer. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1350–1358, 2019.

[3] Kuan Fang, Te-Lin Wu, Daniel Yang, Silvio Savarese, and Joseph J Lim. Demo2vec: Reasoning object affordances from online videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2139–2147, 2018.

[4] Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. Grounded human-object interaction hotspots from video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8688–8697, 2019.

[5] Hongchen Luo, Wei Zhai, Jing Zhang, Yang Cao, and Dacheng Tao. Learning visual affordance grounding from demonstration videos. arXiv preprint arXiv:2108.05675, 2021.

[6] Kateryna Zorina, Justin Carpentier, Josef Sivic, and Vladimír Petrík. Learning to manipulate tools by aligning simulation to video demonstration. *IEEE Robotics and Automation Letters*, 7(1):438–445, 2021.

[7] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[8] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[9] Yeping Wang, Gopika Ajaykumar, and Chien-Ming Huang. See what I see: Enabling user-centric robotic assistance using first-person demonstrations. In *HRI '20: ACM/IEEE International Conference on Human-Robot Interaction, Cambridge, United Kingdom, March 23-26, 2020*, pages 639–648. ACM, 2020.

[10] David A Duque, Flavio A Prieto, and Jose G Hoyos. Trajectory generation for robotic assembly operations using learning by demonstration. *Robotics and Computer-Integrated Manufacturing*, 57:292–302, 2019.

[11] Austin Myers, Ching L Teo, Cornelia Fermüller, and Yiannis Aloimonos. Affordance detection of tool parts from geometric features. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1374–1381. IEEE, 2015.

[12] Anh Nguyen, Dimitrios Kanoulas, Darwin G Caldwell, and Nikos G Tsagarakis. Detecting object affordances with convolutional neural networks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2765–2770. IEEE, 2016.

[13] Krishneel Chaudhary, Kei Okada, Masayuki Inaba, and Xiangyu Chen. Predicting part affordances of objects using two-stream fully convolutional network with multimodal inputs. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3096–3101. IEEE, 2018.

[14] Thanh-Toan Do, Anh Nguyen, and Ian Reid. Affordancenet: An end-to-end deep learning approach for object affordance detection. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1–5. IEEE, 2018.

[15] Xue Zhao, Yang Cao, and Yu Kang. Object affordance detection with relationship-aware network. *Neural Computing and Applications*, 32(18):14321–14333, 2020.

[16] Wenbai Chen, Chao He, WZ Chen, QL Chen, and PL Wu. A new semantic-based tool detection method for robots. *International Journal of Computers, Communications and Control*, 16(2), 2021.

[17] James J Gibson. The theory of affordances. *Hilldale, USA*, 1(2):67–82, 1977.

[18] Mohammed Hassanin, Salman Khan, and Murat Tahtali. Visual affordance and function understanding: A survey. *ACM Computing Surveys (CSUR)*, 54(3):1–35, 2021.

[19] Johann Sawatzky, Abhilash Srikantha, and Juergen Gall. Weakly supervised affordance detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2795–2804, 2017.

[20] Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. Learning affordance segmentation for real-world robotic manipulation via synthetic images. *IEEE Robotics and Automation Letters*, 4(2):1140–1147, 2019.

[21] Kuan Fang, Yuke Zhu, Animesh Garg, Andrey Kurenkov, Viraj Mehta, Li Fei-Fei, and Silvio Savarese. Learning task-oriented grasping for tool manipulation from simulated self-supervision. *The International Journal of Robotics Research*, 39(2-3):202–216, 2020.

[22] Kaichun Mo, Leonidas J Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6813–6823, 2021.

[23] Zhenjia Xu, Zhanpeng He, and Shuran Song. Universal manipulation policy network for articulated objects. *IEEE Robotics and Automation Letters*, 7(2):2447–2454, 2022.

[24] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas J. Guibas, and Hao Dong. Vat-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*.

[25] Kei Ota, Hsiao-Yu Tung, Kevin A. Smith, Anoop Cherian, Tim K. Marks, Alan Sullivan, Asako Kanezaki, and Joshua B. Tenenbaum. H-saur: Hypothesize, simulate, act, update, and repeat for understanding object articulations from interactions. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7272–7278. IEEE, 2023.

[26] Dandan Shan, Jiaqi Geng, Michelle Shu, and David F Fouhey. Understanding human hands in contact at internet scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9869–9878, 2020.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[28] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE transactions on pattern analysis and machine intelligence*, 41(3):740–757, 2018.

[29] Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.

[30] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *2009 IEEE 12th international conference on computer vision*, pages 2106–2113. IEEE, 2009.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[32] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations(ICLR)*, 2015.

[33] Yifei Huang, Minjie Cai, Zhenqiang Li, and Yoichi Sato. Predicting gaze in egocentric video by learning task-dependent attention transition. In *Proceedings of the European conference on computer vision (ECCV)*, pages 754–769, 2018.

[34] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. A deep multi-level network for saliency prediction. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3488–3493. IEEE, 2016.

[35] Matthias Kümmerer, Thomas SA Wallis, and Matthias Bethge. Deepgaze ii: Reading fixations from deep features trained on object recognition. arXiv preprint arXiv:1610.01563, 2016.

[36] Junting Pan, Cristian Canton Ferrer, Kevin McGuinness, Noel E O'Connor, Jordi Torres, Elisa Sayrol, and Xavier Giro-i Nieto. Salgan: Visual saliency prediction with generative adversarial networks. arXiv preprint arXiv:1701.01081, 2017.

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.