2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Visual Geometric Skill Inference by Watching Human Demonstration

Jun Jin†, Laura Petrich†, Zichen Zhang†, Masood Dehghan† and Martin Jagersand†

***Abstract*— We study the problem of learning manipulation skills from human demonstration video by inferring the association relationships between geometric features. Motivation for this work stems from the observation that humans perform eye-hand coordination tasks by using geometric primitives to define a task while a geometric control error drives the task through execution. We propose a graph based kernel regression method to directly infer the underlying association constraints from human demonstration video using Incremental Maximum Entropy Inverse Reinforcement Learning (InMaxEnt IRL). The learned skill inference provides human readable task definition and outputs control errors that can be directly plugged into traditional controllers. Our method removes the need for tedious feature selection and robust feature trackers required in traditional approaches (e.g. feature-based visual servoing). Experiments show our method infers correct geometric associations even with only one human demonstration video and can generalize well under variance.**

I. Introduction

Understanding and applying the mechanism of learning by watching has been researched in robotics for over two decades[1], where the core problem is how to extract high-level reusable symbolic task definitions by observing a human demonstration [1, 2]. Most of the research focuses on learning task goal configurations rather than task execution [3, 4]. This approach reduces the learning complexity and, most importantly, extracts an abstract task representation which allows for generalization. Symbolic task plans can be represented as a tree [5] or graph structure [6, 7] based on the assumption that a task can be decomposed into low-level conditioned elementary skills [8], such as, grasping, striking [9], alignment [10], and peg-in-hole [11]. In order to define the symbols [3], (action, object, task) recognition techniques and a predefined skill sub-module are hand engineered [3]. These predefined manipulation skills are highly task-dependant and do not generalize well in practice.

The main question is whether a general solution exists to parameterize a task. There is no absolute answer, but even if a parameterization exists, it is difficult to find because manipulation tasks are too complex in general. One way of addressing this problem is to use low-level elementary skills as the corner stones of a task; these are potentially easier to learn and generalize. Among the various types of skills, we are interested in those that can be generally parameterized using geometric association constraints (Fig. 1), since a variety of skills can be created from their combinations. We name these **geometric skills**, which are inherently represented in the image space by geometric primitives (points, lines, conics, planes, etc.) and similar to how human eye-hand coordination works [12]. This parameterization method was introduced by Dodds et al. [13] to solve the box-packing task and then implemented by Gridseth et al. [14] on various skills, including, grasping, placing, insertion, and cutting.

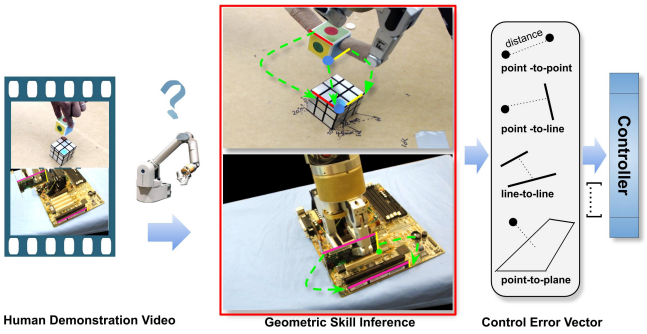

Fig. 1: Representation of manipulation skills using constraints association between geometric primitives. For example, the alignment skill (or insertion skill) is a combination of several point-to-point or co-linearity constraints; This parameterization partitions the problem into two parts: the geometric task representation and its control error output by computing the Euclidean distance between geometric primitives.

This approach, however, has several drawbacks. The task specification is tedious, as it requires manual assignment of associations among geometric features, and is highly dependent on robust feature trackers [15]. This paper aims to address these issues by learning from watching. We propose a method to directly *regress the geometric association constraints* on each frame.

The main contributions of this paper are:

- Provide an interpretable and invariant robotic task representation using geometric features and their association constraints that are easy to monitor and validate.
- Remove the dependency on robust feature trackers in previous methods [14, 16] by directly estimating the association constraint between geometric features.
- Provide a robust feature association learning method by utilizing the fact that multiple feature associations can define the same task.

To elaborate details of our third contribution, for example, when some features are occluded, other candidates will make up for this and continue to define the task. In contrast, traditional tracking-based methods fix such associations in the initial feature selection stage and may be unable to recover

†Authors are with the Department of Computing Science, University of Alberta, Edmonton AB., Canada, T6G 2E8. {jjin5,laurapetrich,zichen2,masood1,mj7}@ualberta.ca

[1]The earliest work can be traced back to Ikeuchi et al. [1] and Kuniyoshi et al. [2] in 1994.

from occlusions. This stiffness on constraints is removed in our maximum entropy-based geometric constraint regression method.

The remainder of this work will focus on two issues: (1) how to generally encode different types of geometric association constraints and build more complex geometric skills from them; and (2) how to optimize such constraints given one human demonstration video. Experimental results are reported in Sec. IV.

II. Related Works

This paper is inspired by research works in robot learning, visual servoing and graph-based relational inference.

End to end learning by watching: This approach, commonly known as imitation learning [17], has been recently gaining interest. Sermanet et al. presented TCN [18] to learn from contrastive positive and negative frame changes along time. Yu et al. proposed a meta-learning based method [19] to encode prior knowledge from a few thousand human/robot demonstrations, then learned a new task from one demonstration. End to end learning approaches lack interpretability. Furthermore, to the authors' knowledge, learning by watching only from one human demonstration is still difficult.

Learning task plans by watching: This approach provides the most intuitive motivation and contributes to many of the early works in learning by watching. Such approaches try to generate human readable symbolic representations at a semantic level [5, 7, 20] to provide high level task planning, which is important for generalizability. Ikeuchi et al. presented a general framework [1] that relies on object/task/grasp recognition to generate assembly plans from observation. Modern approaches use a grammar parser [5], causal inference [7], and neural task programming [21]. Konidaris et al. proposed constructing skill trees [22] at the trajectory level to acquire skills from human demonstration using hierarchical reinforcement learning (RL) with options. This work presents a general framework to learn a tree level structured task. However, such works require hard-coded recognition submodules or lacks generality in various tasks.

Learning correspondence relationships by watching: Learning correspondence relationships to represent a task concept from human demonstration videos provides a generalizable task representation. Current approaches formulate the correspondence relationship through learning at either the *object level* [23] or *key points level* [24, 25]. Beyond a simple correspondence relationship representation, Sieb et al. propose a graph-structured object relationship inference method [26] in visual imitation learning. However, apart from learning relationships in the objects or key points level, using a more general framework to construct complex tasks from constraints among fine-grained geometric primitives (points, lines, conics) has rarely been studied.

Geometric approaches in skill learning: Constructing skills using geometric features provides good interpretability. Apart from works mentioned in Sec. I, Ahmadzadeh et al. proposed a system called VSL [3] that is capable of learning skills from one demonstration. VSL first detects objects in an image and represents them using image feature extractors like SIFT. It computes object spatial motion changes via feature matching and then forms a new task goal configuration used to generate motion primitives by a trajectory-based learning from demonstration (LfD) method [27]. Landmark-based pre/post action condition detection is also used to construct a task plan. Triantafyllou et al. proposed a geometric approach to solve the garment unfolding task [28]. Tremblay et al. proposed a human-readable plan generation method [29] which provides interpretability by modeling the 3D wire frame of blocks, however, it requires simulator training for prior 3D modelling.

III. Method

A. Geometric Skill Kernels

Let $\mathcal{O}$ denote the observation space and $\mathcal{F}$ denote the observed geometric features[2]. Each feature has two parts: a descriptor f_i that encodes locally invariant properties and a coordinate parameter set y_i that encodes globally geometric properties[3]. A geometric skill kernel k is a composite functional structure that describes association constraints between geometric features $(f_i, y_i) \in \mathcal{F}$. To ground our formalism, we describe some basic examples:

- *point-to-point* k_{p2p}: the coincidence of two points.
- *point-to-line* k_{p2l}: a point is on a line.
- *line-to-line* k_{l2l}: a line is collinear with another line.
- *coplanarity* k_{copl}: coplanar four points or two lines.

Each kernel has two parts: a geometric association constraint representation part and a control error generation part used to guide robot actions.

1) Geometric association constraint representation: Inspired by graph motifs [31], each skill kernel is a unit graph with different structures. An undirected graph $\mathcal{G} = \{V, E\}$ is used to represent the association constraint, where nodes V are variables that take input of feature descriptors $\{f_1, ..., f_n\}$, and edges E define a fixed graph structure (as shown in Fig. 2A). For example, the graph for k_{p2p} has two connected nodes, and each node v_i corresponds to f_i. By feeding in two points, we get a graph instance $\mathcal{G}$ and use a select-out function gk to measure how relevant it is to define the skill. Then, we have:

$$gk : \mathcal{G} \rightarrow [0, 1] \in \mathbb{R}, \quad gk(\mathcal{G}(\{f_1, ..., f_n\})) \tag{1}$$

For example, in the 'insertion' skill (Fig. 2B), the graph instance of P3 and P4 has higher gk output than that of P1 and P2, and will be selected out.

It is worth noting the *ambiguity property* of skill kernels, where several association instances may define the same skill. Because of this property, it is crucial to learn a robust feature association selection model in the long-run since when the current association is not available (occluded or outside of the field of view (FOV), a candidate association will be selected. For example, in Fig. 2B, both the association of P3 to P4 and P2 to P5 can partially define the skill. In

[2]points, lines, conics, planes, spheres etc. from an image or point cloud.
[3]More details on the parameterization of geometric primitives in [30].

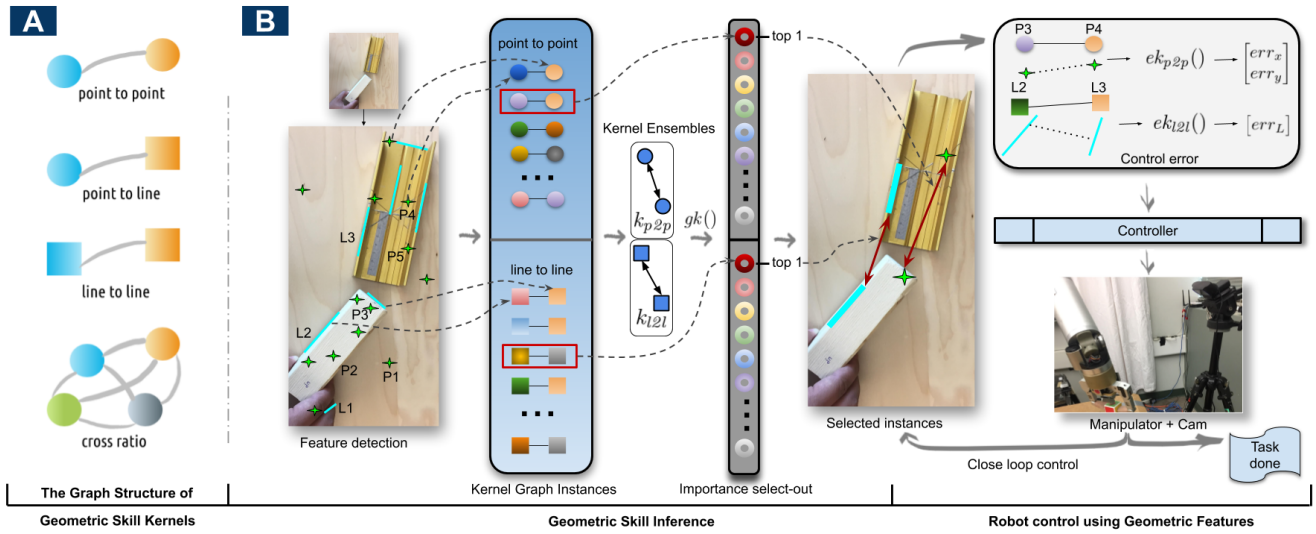

Fig. 2: **A**: Graph structured skill kernels. **B**: Function of a skill kernel and kernel ensembles. A skill kernel takes input of all geometric feature association instances (kernel graph instances) and output their rank of relevance (select-out) w.r.t. the skill definition (defined by human demonstration video). A skill is a combination (kernel ensembles) of several skill kernels. For example, an 'insertion' skill consists of a point-to-point k_{p2p} and a line-to-line k_{l2l} skill kernel. Given one image observation, we can enumerate all possible geometric feature associations. By feeding their corresponding descriptors $\{f_i\}$, each association will create one kernel graph instance and each kernel instance will output a select-out to decide which association should be selected. A control error computed from their corresponding $\{y_i\}$.

successive steps, P5 will be occluded so its instance won't be observable, however, P3 to P4 can make up the role. Another issue is task *decidability*, which determines which 2D image-coordinate constraints are needed to guarantee a particular 3D configuration. We do not cover decidability here, but refer to [13].

2) Geometric control error generation: Let $E_k : \{y_1, ..., y_n\} \rightarrow \mathbb{R}^d$ denote the mapping of all nodes geometric parameters to a control error vector where d is the degree of freedom that this constraint contributes. For example, given a *point-to-point* skill, $d = 2$ for image points. The control error of a point-to-point kernel is the point distance, while of a point-to-line kernel is the dot product of their homogeneous coordinates. More examples can be found in [14, 16]. E_k will be used in the following optimization using human demonstrations and in generating control signals used to guide robot action.

B. Parameterization

1) Parameterization of $\mathcal{G}$: $\mathcal{G}$ is parameterized by a T-layer message passing graph neural network [32]. Each node $v_i \in \mathcal{G}$ relates a h-dimensional hidden state h_t^i. At layer t (or time step t), each nodes hidden state h_t^i is updated via three steps. (I) Pair-wise message generation $\mathcal{M}$:

$$m_{i\rightarrow j}^{t+1} = \mathcal{M}(h_i^t, h_j^t) \tag{2}$$

where h_j^t relates to any node v_j connected to v_i. (II) Message aggregation $\mathcal{A}$ which collects all incoming messages:

$$m_i^{t+1} = \mathcal{A}(m_{j\rightarrow i}^{t+1}) \tag{3}$$

We simply use summation as $\mathcal{A}$ in our implementation. Lastly, (III) message update $\mathcal{U}$:

$$h_i^{t+1} = \mathcal{U}(h_i^t, m_i^{t+1}) \tag{4}$$

where a gated recurrent unit (GRU) is used. After T layer updates, all of the nodes final states are fed into a *MLP* layer with an activation function that outputs a scalar value $b = \sigma(\text{MLP}(h_1^T, ..., h_n^T))$.

2) The select-out function gk: Given one image, we construct m graph instances by enumerating all possible geometric primitive combinations (e.g., point-to-point by listing association between any two points). Each instance $\{\mathcal{G}_i\}$ represents one association and will output its relevance b_i. A select-out function gk outputs a relevance factor g_i:

$$g_i = gk(\mathcal{G}_i) = \text{softmax}(b_i, \{b_1, ..., b_m\}) \tag{5}$$

Let ${E_k}^i$ denotes the control error for each graph instance, we now define the overall control error $\mathbf{Ec}$ for a whole image as:

$$\mathbf{Ec} = \sum_{i=1}^{m} g_i {E_k}^i \tag{6}$$

C. InMaxEnt IRL for optimization

Given human demonstration video frames, we apply *InMaxEnt IRL* [10] for optimization. To this end, we define the reward function, which connects skill kernels to entropy models. Through the optimization of this reward function, the skill kernel is also optimized. In practice, each skill kernel is optimized individually. We use k_{p2p} as an example in the following discussion.

1) Reward function: Each state s_t is an image related to a control error $\mathbf{Ec}_t$, where the subscript t denotes the time step in RL. An optimized k_{p2p} should *consistently* select *'correct instances'* among all states. During human demonstration we should expect $\mathbf{Ec}_t$ to decrease globally (but not necessarily in each step). Intuitively, we should get a positive reward if

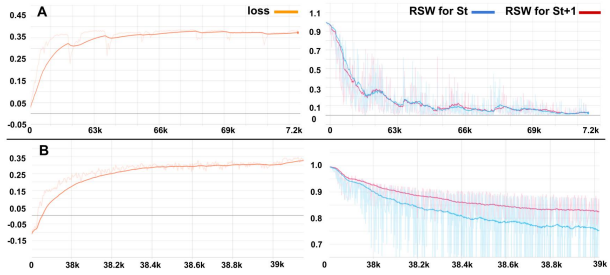

Fig. 3: RSW measures how much relevance contributed from the non-selected association instances. The lower RSW, the more deterministic in select-out. **A** shows the training curve with RSW regularizer. The relevance from remaining instances stays below 0.1%. **B** is without RSW regularizer. Although the cost function is optimized, the non-selected ones still occupy 75% of relevance. Note that we are maximizing the loss.

we observe a decrease in $\mathbf{Ec}_t$, otherwise the reward should be negative. Let $\Delta\mathbf{Ec}_t = \|\mathbf{Ec}_{t+1}\| - \|\mathbf{Ec}_t\|$, we define:

$$r_t = \frac{2}{1 + exp(\beta \Delta \mathbf{Ec}_t)} - 1 \tag{7}$$

$r_t \in (-1, 1)$, where β normalizes the scale of different skill kernels' output E_k.

2) The variational expert assumption: InMaxEnt IRL considers imperfect expert demonstrations with a confidence level α. A higher confidence level results in smaller variance σ_0 in demonstration. We assume that in a human demonstration, at state s_t the probability of selecting an action that transitions to the observed state s_{t+1} follows a Boltzmann distribution with conditions:

$$p(s_{t+1}|s_t) = \frac{1}{\mathcal{Z}_t} exp(r_t^*) p(r_t^*), \tag{8}$$

where r_t^* is the reward of this observed state change, and

$$\mathcal{Z}_t = \mathbb{E}_{p(r_{tj}; r_t^*)}[exp(r_{tj})] \tag{9}$$

is the partition function, $r_{tj} \sim \mathcal{N}(r_t^*, \sigma_0^2)$ is a truncated normal distribution with domain in [-1,1]. This means that the expert prefers the action with the highest reward among all possible actions $\mathcal{A}_t = \{a_{tj}\}$. To emphasize high impact actions in $\mathcal{A}_t$, suppose a_{tj} gets a reward r_{tj}, the chance of a_{tj} included in the pool is: $p(r_{tj}) = \mathcal{N}(r_t^*, \sigma_0)$, this is called a human factor [10] since it varies with the human demonstrator's confidence.

3) Loss function: To maximize the probability of observed human demonstration video sequence $p(\{s_t\})$ by applying MDP property, we have:

$$\mathcal{L} = \arg\max_{\theta} \sum log[p(s_{t+1}|s_t)] \tag{10}$$

With equation (8) and removing the last constant, the cost function can be further written as:

$$\mathcal{L} = \arg\max_{\theta} \sum r_t^* - log\mathcal{Z}_t \tag{11}$$

Note that if $p(r_{tj})$ has domain $(-\infty, \infty)$, the loss function is a constant. Proofs can be found on our website [33].

Algorithm 1: Optimizing k_{p2p}

Input: Expert demonstration video frames $\{s_1, ..., s_n\}$, confidence level α
Result: Optimal weights $\boldsymbol{\theta}^*$ of k_{p2p}
Construct kernel graph instances on each frame
for t = 1:n **do**
  Feature point extraction on s_t to get $\{(f_i, y_i)\}$
  Enumerate all k_{p2p} instances by association
  Feed all instances to gk to get $\mathbf{Ec}_t$
end
Prepare State Change Samples $\mathcal{D}s = s_t \rightarrow s_{t+1}$
Compute σ_0 using α; **Shuffle** $\mathcal{D}s$**; Initialize** $\boldsymbol{\theta}^0$
for each iteration **do**
  for each observed sample change in $\mathcal{D}s$ **do**
    Forward pass
    Compute r_t^*
    Compute $\nabla_{r_t^*}\mathcal{L} = \sum 1 - \frac{1}{\mathcal{Z}_t}\nabla_{r_t^*}\mathcal{Z}_t$
    $grad = k_{p2p}.backProp(\nabla_{r_t^*}\mathcal{L})$
    Gradient ascent update
    $\boldsymbol{\theta}^{n+1} = updateWeights(\boldsymbol{\theta}^n, grad)$
  end
end

To force gk into making selections more deterministic meanwhile considering the **ambiguity property**, a penalty regularizer $-\lambda\, RSW$ is added to the reward where λ is a hyperparameter and *RSW* is the residual sum of weights (**RSW**). This makes gk output major weights on selected p alternatives while minimizing the residual sum of weights. Fig. 3 shows a comparison of training with and without RSW penalty in the *Sorting* task.

4) Optimization: The last item in eq. (11) is a constant and $\mathcal{Z}_t$ is a function of r_t^*, which is further represented using skill kernels with parameters θ. Then, we have:

$$\nabla_\theta \mathcal{L} = \sum \nabla_\theta r_t^* - \frac{1}{\mathcal{Z}_t} \nabla_{r_t^*} \mathcal{Z}_t \nabla_\theta r_t^* \tag{12}$$

$\nabla_\theta r_t^*$ can be solved by back propagation from eq. (7) to the graph neural network in the skill kernel. $\mathcal{Z}_t$ can be estimated by a Monte Carlo estimator sampling s_1 samples from the truncated normal distribution $p(r_{tj})$:

$$\mathcal{Z}_t \approx \frac{1}{s_1} \sum^{s_1} exp(r_{tj}), \;\; r_{tj} \sim p(r_{tj}) \tag{13}$$

$\nabla_\theta \mathcal{Z}_t$ is the derivative of an expectation. By applying the log derivative trick, we have:

$$\begin{aligned} \nabla_{r_t^*}\mathcal{Z}_t &= \nabla_{r_t^*}\mathbb{E}_{p(r_{tj})}[exp(r_{tj})] \\ &= \mathbb{E}_{p(r_{tj})}[exp(r_{tj}) \nabla_{r_t^*} log\, p(r_{tj})] \\ &\approx \frac{1}{s_2} \sum^{s_2} exp(r_{tj}) \nabla_{r_t^*} log\, p(r_{tj}), \quad r_{tj} \sim p(r_{tj}) \end{aligned} \tag{14}$$

Since $p(r_{tj})$ is tractable, it's trivial to get:

$$\nabla_{r_t^*} log\, p(r_{tj}) = \frac{x_\mu}{\sigma_0} + \frac{exp(-b_\mu^2/2) - exp(-a_\mu^2/2)}{\sqrt{2\pi}\sigma_0[\phi(b_\mu) - \phi(a_\mu)]} \tag{15}$$

$x_\mu = (r_{tj} - r_t^*)/\sigma_0$, $a_\mu = (-1 - r_t^*)/\sigma_0$, $b_\mu = (1 - r_t^*)/\sigma_0$. where ϕ is defined in [34]. By combining the above equations, $\nabla_\theta \mathcal{L}$ is solved.

The optimization on k_{p2p} is summarized in Algorithm 1.

D. From Skill Kernel to Skills

In this paper, we consider that a skill is simply the combination of several skill kernels, namely *kernel ensembles*. There should be more advanced ways to construct a skill from different kernels, although this is not discussed here. [4]

E. From Skill to Control

Given an image, each skill kernel will select out several alternative association instances and generate control error vectors. For example, the point-to-point k_{p2p} will output vectors with structure $[err_x, err_y]$, which can be plugged in controllers like feature-based visual servoing or uncalibrated visual servoing [35]. More examples using geometric features (lines, conics) and based on which, the constructed skill kernels in UVS control are included in [14].

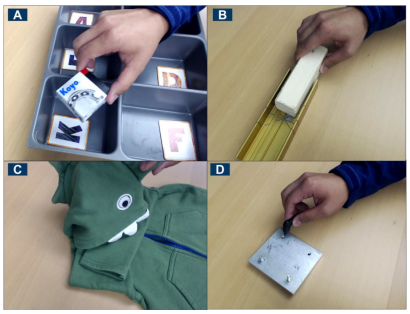

Fig. 4: Four types of skills with human demonstration. A: *Sorting* skill. B:*Insertion* skill. C:*Folding* cloth skill. D: Driving a *Screw* to the hole skill.

IV. Experiments

A. Quantitative Evaluation

We first evaluate what types of skills the learned inference behavior is capable of. Four types are tested (Fig. 4): *Sorting* skill represents a regular setting; *Insertion* is for skills that need line-to-line constraint; *Folding* is for manipulation with deformable objects; and *Screw* skill represents types that have low image textures. Each skill is evaluated on videos that show a human performing the same task but with random behaviors. The objective is to infer the correct geometric feature associations that can be used to define the demonstrated skill.

We next test if the learned behavior from human demonstration video directly generalizes to a robot hand. For our tests, the background table is also changed and the target pose is randomly arranged (Fig. 7A).

Lastly, we test on the robot with four other scenarios (Fig.8, B-E): moving camera; occlusion; object running out of camera's field of view; and illumination changes.

[4]For example, for a 'peg-in-hole' skill, the point-to-point kernel should be used to first coarsely move to the target, while line-to-line kernel best fits in the final alignment actions. Their relationship is not a simple combination.

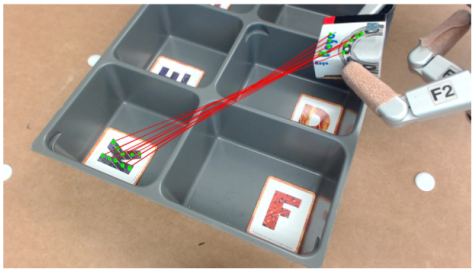

Fig. 5: The hand designed baseline requires human to specifically select 10 pairs of feature points to define the demonstrated skill.

Baseline: To our best knowledge, there are no existing methods that learn geometric feature associations by watching human demonstration. However, for comparison, we hand designed a baseline on the *Sorting* skill. The baseline requires a human to manually select 10 pairs of feature points and initialize 20 trackers. Each pair has one point on the object and another on the target. All of the 10 pairs simultaneously define the same skill (Fig.5), resulting in a robust baseline. In evaluation, as long as one pair still defines the skill, the baseline is marked as a successful trial.

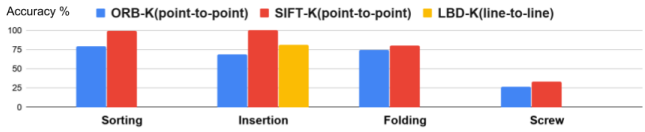

Fig. 6: Results of the four skills.

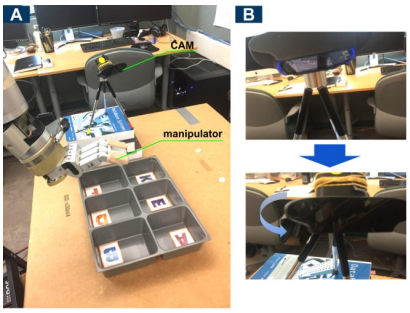

Fig. 7: A: Experimental setup on the manipulator. **B**: We change the camera pose in evaluation by rotation and a random displacement.

Metric: We evaluate on each video frame and calculate the accuracy of inferences. For the baseline, when it fails on one frame, it can't be resumed unless a human hand select the features again, therefore we report only success or failure on the final result. For our method, failures can be automatically corrected in successive frames. While our method can output p inferred associations, we pick the top one for evaluation.

1) Training: For each skill, we evaluate the point-to-point kernel using SIFT and ORB features respectively. For the *Insertion* skill, we add the line-to-line kernel using LBD [36] line descriptor. All kernels have the same graph layer size=5 with hidden state dimension=512 and p=10 alternatives (III-C.1). In training, we set the regularizer coefficient $\lambda = 0.1$, and human factor $\sigma_0 = 0.55$. Each kernel with different descriptors are trained individually.

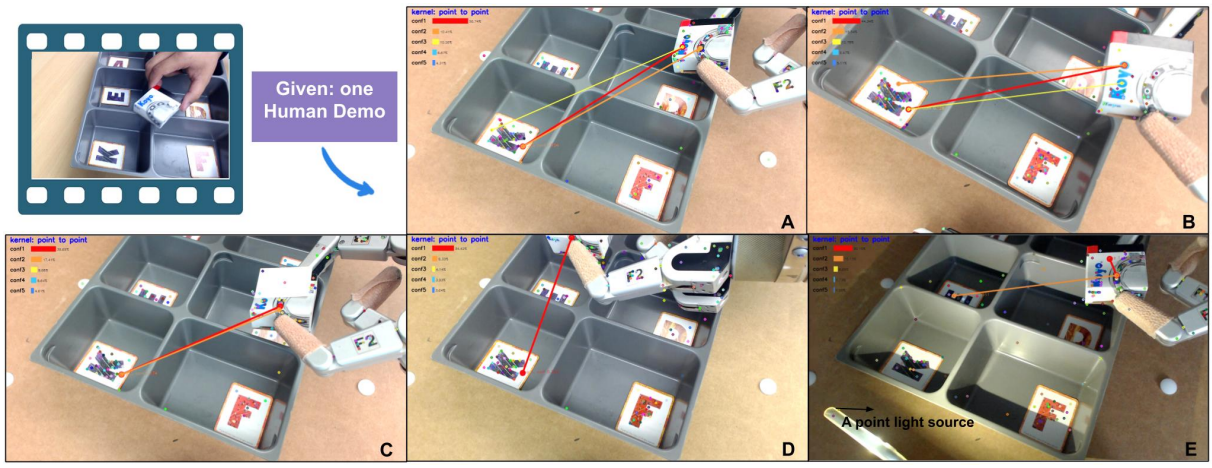

Fig. 8: Given one human demonstration video, we evaluate the learned behavior on 5 scenarios. **A**: using robot hand with a different background and random target pose; **B**: projective variance due to camera pose change; **C**: occlusion; **D**: object out of camera's view-field; and **E**: illumination change. For each scenario, we detect all feature points and use a colored line to mark the select-out associations. The top one is marked red and the bar next to it indicates the estimation confidence. Only the association with confidence greater than 10% is displayed. Results are reported in Table I.

	Random Target	*Move Camera*	*Object Occlusion*	*Outside FOV*	*Change Illumination*
Baseline	100.0%	0.0%	0.0%	0.0%	0.0%
Ours	99.1%	96.7%	92.7%	81.2%	0.0%

TABLE I: Evaluations results of running robot under various environmental settings as shown in Fig. 8. For each variation setting, we count correct geometric association inferences on each frame and calculate the percentage of successful inferences among all the frames during the execution of *Sorting* task.

2) Results:

a) Different skills: Results (Fig.6) on the 4 skills show our method is capable of the *Sorting* and *Insertion* skill and performs moderately in *Folding* and *Screw* skills. In experiments, we observed that when both object and target have rich textures, results improve. This may be from the use of SIFT or ORB that are local descriptors dependent on textures. We can expect further improvement by using other local feature descriptors [37] [38]. We also find the more features that can be fed into the skill kernel, the better accuracy it performs. Due to our hardware GPU limitation, we can only test using a small number (60 in average) of features.

b) Varying environment: Fig. 8 lists results on various environmental conditions. In general, i) our method is robust to occlusion. When some feature associations are occluded, the selection of others will make it up; and ii) our method exhibits robust behavior so that failure in some frames doesn't affect successive frames since it directly selects the feature associations on each frame. In contrast, the baseline method depends on the initialization of video trackers and continuous tracking. We observe that the learned inference behavior tends to select fixed association instances while showing the flexibility of selecting alternatives when fixed ones are not observable. We also observe that the accuracy is highly related to the capability of SIFT descriptor. It reaches high accuracy under projective variance (B), however, fails under illumination changes (E).

Although results on the robot manipulator show our method can output the correct selections of geometric feature associations which can be directly used in controllers (e.g. uncalibrated visual servoing [14]), due to resource limitations we did not test with a plug-in controller. We leave this to our future work.

V. Conclusion

We propose a graph based kernel regression method to infer the association relationship between geometric features by watching human demonstrations. The learned skill inference provides human readable task definition and outputs control errors that fit in traditional controllers. Our method removes the dependency on robust feature trackers and tediously hand selection process in traditional robotic task specification. The learned selection model provides a robust feature association behavior under various environmental settings.

Although results are promising, there are issues that need to be further investigated. 1) Consistent control error output: while the result shows that our method tends to select a fixed set of associations, it can't guarantee the selection consistency. One possible solution is to add constraints between frames. 2) Other local feature descriptors [37] [38] are worth trying for better generalization. 3) The generalization to point cloud geometric primitive needs to be further studied.

References

[1] K. Ikeuchi and T. Suehiro, "Toward an assembly plan from observation. i. task recognition with polyhedral objects," IEEE Trans. on robotics and automation, vol. 10, no. 3, pp. 368–385, 1994.

[2] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," IEEE Tran. robotics and automation, vol. 10, no. 6, pp. 799–822, 1994.

[3] S. R. Ahmadzadeh, A. Paikan, F. Mastrogiovanni, L. Natale, P. Kormushev, and D. G. Caldwell, "Learning symbolic representations of actions from human demonstrations," in 2015 IEEE Int. Conf. on Robotics and Automation (ICRA). IEEE, 2015, pp. 3801–3808.

[4] M. Dehghan, Z. Zhang, M. Siam, J. Jin, L. Petrich, and M. Jagersand, "Online object and task learning via human robot interaction," in 2019 Int. Conf. on Robotics and Automation (ICRA), 2019, pp. 2132–2138.

[5] Y. Yang, Y. Li, C. Fermuller, and Y. Aloimonos, "Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web," in 29th AAAI Conference on Artificial Intelligence, 2015.

[6] N. Shukla, C. Xiong, and S.-C. Zhu, "A unified framework for human-robot knowledge transfer," in 2015 AAAI Fall Symposium Series, 2015.

[7] C. Xiong, N. Shukla, W. Xiong, and S.-C. Zhu, "Robot learning with a spatial, temporal, and causal and-or graph," in 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016, pp. 2144–2151.

[8] R. Dillmann, M. Kaiser, and A. Ude, "Acquisition of elementary robot skills from human demonstration," in International symposium on intelligent robotics systems. Citeseer, 1995, pp. 185–192.

[9] J. Kober, K. Mülling, O. Krömer, C. H. Lampert, B. Schölkopf, and J. Peters, "Movement templates for learning of hitting and batting," in Robotics and Automation (ICRA), 2010 IEEE Int. Conf. on. IEEE, 2010, pp. 853–858.

[10] J. Jin, L. Petrich, M. Dehghan, Z. Zhang, and M. Jagersand, "Robot eye-hand coordination learning by watching human demonstrations: a task function approximation approach," arXiv preprint arXiv:1810.00159, 2018.

[11] G. Schoettler, A. Nair, J. Luo, S. Bahl, J. A. Ojea, E. Solowjow, and S. Levine, "Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards," arXiv preprint arXiv:1906.05841, 2019.

[12] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," IEEE transactions on robotics and automation, vol. 12, no. 5, pp. 651–670, 1996.

[13] Z. Dodds, G. D. Hager, A. S. Morse, and J. P. Hespanha, "Task specification and monitoring for uncalibrated hand/eye coordination," in Proceedings 1999 IEEE International Conference on Robotics and Automation, vol. 2. IEEE, 1999, pp. 1607–1613.

[14] M. Gridseth, O. Ramirez, C. P. Quintero, and M. Jagersand, "ViTa: Visual task specification interface for manipulation with uncalibrated visual servoing," Proceedings - IEEE International Conference on Robotics and Automation, vol. 2016-June, pp. 3434–3440, 2016.

[15] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, P. C. V. Ser, Q. Bateux, and E. Marchand, "Visual servoing from deep neural networks," arXiv preprint arXiv:1705.08940, 2017.

[16] Z. Dodds, M. Jägersand, and G. Hager, "A Hierarchical Architecture for Vision-Based Robotic Manipulation Tasks," First Int. Conf. on Computer Vision Systems, vol. 542, pp. 312–330, 1999.

[17] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," 21 Int. Conf. on Machine learning - ICML '04, p. 1, 2004.

[18] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain, "Time-contrastive networks: Self-supervised learning from video," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 1134–1141.

[19] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine, "One-shot imitation from observing humans via domain-adaptive meta-learning," arXiv preprint arXiv:1802.01557, 2018.

[20] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," Robotics and Autonomous Systems, vol. 47, no. 2-3, pp. 109–116, 2004.

[21] D. Xu, S. Nair, Y. Zhu, J. Gao, A. Garg, L. Fei-Fei, and S. Savarese, "Neural task programming: Learning to generalize across hierarchical tasks," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 1–8.

[22] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto, "Robot learning from demonstration by constructing skill trees," The International Journal of Robotics Research, vol. 31, no. 3, pp. 360–375, 2012.

[23] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," arXiv preprint arXiv:1806.08756, 2018.

[24] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kpam: Keypoint affordances for category-level robotic manipulation," arXiv preprint arXiv:1903.06684, 2019.

[25] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, "Keto: Learning keypoint representations for tool manipulation," arXiv preprint arXiv:1910.11977, 2019.

[26] M. Sieb, Z. Xian, A. Huang, O. Kroemer, and K. Fragkiadaki, "Graph-structured visual imitation," arXiv preprint arXiv:1907.05518, 2019.

[27] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," Neural computation, vol. 25, no. 2, pp. 328–373, 2013.

[28] D. Triantafyllou, I. Mariolis, A. Kargakos, S. Malassiotis, and N. Aspragathos, "A geometric approach to robotic unfolding of garments," Robotics and Autonomous Systems, vol. 75, pp. 233–243, 2016.

[29] J. Tremblay, T. To, A. Molchanov, S. Tyree, J. Kautz, and S. Birchfield, "Synthetically trained neural networks for learning human-readable plans from real-world demonstrations," arXiv preprint arXiv:1805.07054, 2018.

[30] R. Hartley and A. Zisserman, Multiple view geometry in computer vision. Cambridge university press, 2003.

[31] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi, "Neural motifs: Scene graph parsing with global context," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5831–5840.

[32] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 1263–1272.

[33] Jin, Jun and Dehghan, Masood and Jagersand, Martin, "Visual geometric skill inference by watching human demonstration: Supplementary materials," 2019, [Online; accessed 5-Sept-2019]. [Online]. Available: http://webdocs.cs.ualberta.ca/~vis/Jun/InMaxEntIRL/supplymentary-material.pdf

[34] Wikipedia contributors, "Truncated normal distribution," 2019, [Online; accessed 5-Sept-2019]. [Online]. Available: https://en.wikipedia.org/wiki/Truncated_normal_distribution

[35] M. Jagersand and R. Nelson, "Visual space task specification, planning and control," in Proc. of Int. Symposium on Computer Vision-ISCV. IEEE, 1995, pp. 521–526.

[36] L. Zhang and R. Koch, "An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency," Journal of Visual Communication and Image Representation, vol. 24, no. 7, pp. 794–805, 2013.

[37] S. A. Winder and M. Brown, "Learning local image descriptors," in 2007 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2007, pp. 1–8.

[38] B. Kumar, G. Carneiro, I. Reid, et al., "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5385–5394.