arXiv:1704.06888v3 [cs.CV] 20 Mar 2018

Time-Contrastive Networks: Self-Supervised Learning from Video

Pierre Sermanet[1*@] Corey Lynch[1R*] Yevgen Chebotar[2*]
Jasmine Hsu[1] Eric Jang[1] Stefan Schaal[2] Sergey Levine[1]
[1]Google Brain [2]University of Southern California

***Abstract*— We propose a self-supervised approach for learning representations and robotic behaviors entirely from unlabeled videos recorded from multiple viewpoints, and study how this representation can be used in two robotic imitation settings: imitating object interactions from videos of humans, and imitating human poses. Imitation of human behavior requires a viewpoint-invariant representation that captures the relationships between end-effectors (hands or robot grippers) and the environment, object attributes, and body pose. We train our representations using a metric learning loss, where multiple simultaneous viewpoints of the same observation are attracted in the embedding space, while being repelled from temporal neighbors which are often visually similar but functionally different. In other words, the model simultaneously learns to recognize what is common between different-looking images, and what is different between similar-looking images. This signal causes our model to discover attributes that do not change across viewpoint, but do change across time, while ignoring nuisance variables such as occlusions, motion blur, lighting and background. We demonstrate that this representation can be used by a robot to directly mimic human poses without an explicit correspondence, and that it can be used as a reward function within a reinforcement learning algorithm. While representations are learned from an unlabeled collection of task-related videos, robot behaviors such as pouring are learned by watching a single 3rd-person demonstration by a human. Reward functions obtained by following the human demonstrations under the learned representation enable efficient reinforcement learning that is practical for real-world robotic systems. Video results, open-source code and dataset are available at sermanet.github.io/imitate**

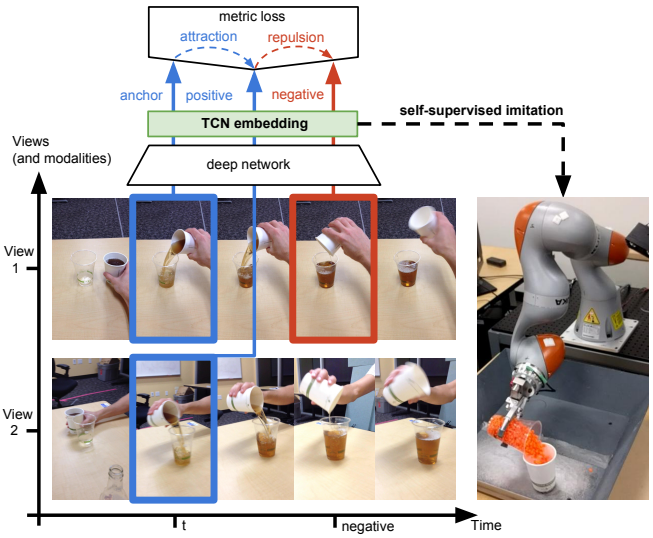

Fig. 1: **Time-Contrastive Networks (TCN):** Anchor and positive images taken from simultaneous viewpoints are encouraged to be close in the embedding space, while distant from negative images taken from a different time in the same sequence. The model trains itself by trying to answer the following questions simultaneously: What is common between the different-looking blue frames? What is different between the similar-looking red and blue frames? The resulting embedding can be used for self-supervised robotics in general, but can also naturally handle 3rd-person imitation.

I. Introduction

While supervised learning has been successful on a range of tasks where labels can be easily specified by humans, such as object classification, many problems that arise in interactive applications like robotics are exceptionally difficult to supervise. For example, it would be impractical to label every aspect of a pouring task in enough detail to allow a robot to understand all the task-relevant properties. Pouring demonstrations could vary in terms of background, containers, and viewpoint, and there could be many salient attributes in each frame, e.g. whether or not a hand is contacting a container, the tilt of the container, or the amount of liquid currently in the target vessel or its viscosity. Ideally, robots in the real world would be capable of two things: learning the relevant attributes of an object interaction task purely from observation, and understanding how human poses and object interactions can be mapped onto the robot, in order to imitate directly from *third-person video observations*.

In this work, we take a step toward addressing these challenges simultaneously through the use of self-supervision and multi-viewpoint representation learning. We obtain the learning signal from unlabeled multi-viewpoint videos of interaction scenarios, as illustrated in Figure 1. By learning from multi-view videos, the learned representations effectively disentangle functional attributes such as pose while being viewpoint and agent invariant. We then show how the robot can learn to link this visual representation to a corresponding motor command using either reinforcement learning or direct regression, effectively learning new tasks by observing humans.

The main contribution of our work is a representation learning algorithm that builds on top of existing semantically relevant features (in our case, features from a network trained on the ImageNet dataset [1, 2]) to produce a metric embedding that is sensitive to object interactions and pose, and insensitive to nuisance variables such as viewpoint and

* equal contribution
@ correspondence to sermanet@google.com
R Google Brain Residency program (g.co/brainresidency)

appearance. We demonstrate that this representation can be used to create a reward function for reinforcement learning of robotic skills, using only raw video demonstrations for supervision, and for direct imitation of human poses, without any explicit joint-level correspondence and again directly from raw video. Our experiments demonstrate effective learning of a pouring task with a real robot, moving plates in and out of a dish rack in simulation, and real-time imitation of human poses. Although we train a different TCN embedding for each task in our experiments, we construct the embeddings from a variety of demonstrations in different contexts, and discuss how larger multi-task embeddings might be constructed in future work.

II. Related Work

Imitation learning: Imitation learning [3] has been widely used for learning robotic skills from expert demonstrations [4, 5, 6, 7] and can be split into two areas: behavioral cloning and inverse reinforcement learning (IRL). Behavioral cloning considers a supervised learning problem, where examples of behaviors are provided as state-action pairs [8, 9]. IRL on the other hand uses expert demonstrations to learn a reward function that can be used to optimize an imitation policy with reinforcement learning [10]. Both types of imitation learning typically require the expert to provide demonstrations in the same context as the learner. In robotics, this might be accomplished by means of kinesthetic demonstrations [11] or teleoperation [12], but both methods require considerable operator expertise. If we aim to endow robots with wide repertoires of behavioral skills, being able to acquire those skills directly from third-person videos of humans would be dramatically more scalable. Recently, a range of works have studied the problem of imitating a demonstration observed in a different context, e.g. from a different viewpoint or an agent with a different embodiment, such as a human [13, 14, 15]. Liu et al. [16] proposed to translate demonstrations between the expert and the learner contexts to learn an imitation policy by minimizing the distance to the translated demonstrations. However, Liu et al. explicitly exclude from consideration any demonstrations with domain shift, where the demonstration is performed by a human and imitated by the robot with clear visual differences (e.g., human hands vs. robot grippers). In contrast, our TCN models are trained on a diverse range of demonstrations with different embodiments, objects, and backgrounds. This allows our TCN-based method to directly mimic human demonstrations, including demonstrations where a human pours liquid into a cup, and to mimic human poses without any explicit joint-level alignment. To our knowledge, our work is the first method for imitation of raw video demonstrations that can both mimic raw videos and handle the domain shift between human and robot embodiment.

Label-free training signals: Label-free learning of visual representations promises to enable visual understanding from unsupervised data, and therefore has been explored extensively in recent years. Prior work in this area has studied unsupervised learning as a way of enabling supervised learning from small labeled datasets [17], image retrieval [18], and a variety of other tasks [19, 20, 21, 22]. In this paper, we focus specifically on representation learning for the purpose of model interactions between objects, humans, and their environment, which requires implicit modeling of a broad range of factors, such as functional relationships, while being invariant to nuisance variables such as viewpoint and appearance. Our method makes use of simultaneously recorded signals from multiple viewpoints to construct an image embedding. A number of prior works have used multiple modalities and temporal or spatial coherence to extract embeddings and features. For example, [23, 24] used co-occurrence of sounds and visual cues in videos to learn meaningful visual features. [20] also propose a multimodal approach for self-supervision by training a network for cross-channel input reconstruction. [25, 26] use the spatial coherence in images as a self-supervision signal and [27] use motion cues to self-supervise a segmentation task. These methods are more focused on spatial relationships, and the unsupervised signal they provide is complementary to the one explored in this work.

A number of prior works use temporal coherence [28, 29, 30, 31]. Others also train for viewpoint invariance using metric learning [22, 32, 33]. The novelty of our work is to combine both aspects in opposition, as explained in Sec. III-A. [19] uses a triplet loss that encourages first and last frames of a tracked sequence to be closer together in the embedding, while random negative frames from other videos are far apart. Our method differs in that we use temporal neighbors as negatives to push against a positive that is anchored by a simultaneous viewpoint. This causes our method to discover meaningful dimensions such as attributes or pose, while [19] focuses on learning intraclass invariance. Simultaneous multi-view capture also provides exact correspondence while tracking does not, and can provide a rich set of correspondences such as occlusions, blur, lighting and viewpoint.

Other works have proposed to use prediction as a learning signal [34, 35]. The resulting representations are typically evaluated primarily on the realism of the predicted images, which remains a challenging open problem. A number of other prior methods have used a variety of labels and priors to learn embeddings. [36] use a labeled dataset to train a pose embedding, then find the nearest neighbors for new images from the training data for a pose retrieval task. Our method is initialized via ImageNet training, but can discover dimensions such as pose and task progress (e.g., for a pouring task) without any task-specific labels. [37] explore various types of physical priors, such as the trajectories of objects falling under gravity, to learn object tracking without explicit supervision. Our method is similar in spirit, in that it uses temporal co-occurrence, which is a universal physical property, but the principle we use is general and broadly applicable and does not require task-specific input of physical rules.

Mirror Neurons: Humans and animals have been shown, experimentally, to possess viewpoint-invariant representations of objects and other agents in their environment [38], and

the well known work on "mirror neurons" has demonstrated that these viewpoint invariant representations are crucial for imitation [39]. Our multi-view capture setup in Fig. 2 is similar to the experimental setup used by [38], and our robot imitation setup, where a robot imitates human motion without ever receiving ground truth pose labels, examines how self-supervised pose recognition might arise in a learned system.

III. Imitation with Time-Contrastive Networks

Our approach to imitation learning is to only rely on sensory inputs from the world. We achieve this in two steps. First, we learn abstract representations purely from passive observation. Second, we use these representations to guide robotic imitations of human behaviors and learn to perform new tasks. We use the term imitation rather than demonstrations because our models also learn from passive observation of non-demonstration behaviors. A robot needs to have a general understanding about everything it sees in order to better recognize an active demonstration. We purposely insist on only using self-supervision to keep the approach scalable in the real world. In this work, we explore a few ways to use time as a signal for unsupervised representation learning. We also explore different approaches to self-supervised robotic control below.

A. Training Time-Contrastive Networks

Fig. 2: **Multi-view capture** with two operators equipped with smartphones. Moving the cameras around freely introduces a rich variety of scale, viewpoint, occlusion, motion-blur and background correspondences between the two cameras.

We illustrate our time-contrastive (TC) approach in Fig. 1. The method uses multi-view metric learning via a triplet loss [40]. The embedding of an image x is represented by $f(x) \in \mathbb{R}^d$. The loss ensures that a pair of co-occuring frames x_i^a (*anchor*) and x_i^p (*positive*) are closer to each other in embedding space than any image x_i^n (*negative*). Thus, we aim to learn an embedding f such that

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2 \,,$$
$$\forall \, (f(x_i^a), f(x_i^p), f(x_i^n)) \in \mathcal{T},$$

where α is a margin that is enforced between positive and negative pairs, and $\mathcal{T}$ is the set of all possible triplets in the training set. The core idea is that two frames (anchor and positive) coming from the same time but different viewpoints (or modalities) are pulled together, while a visually similar frame from a temporal neighbor is pushed apart. This signal serves two purposes: learn disentangled representations without labels and simultaneously learn viewpoint invariance for imitation. The cross-view correspondence encourages learning invariance to viewpoint, scale, occlusion, motion-blur, lighting and background, since the positive and anchor frames show the same subject with variations along these factors. For example, Fig. 1 exhibits all these transformations between the top and bottom sequences, except for occlusion. In addition to learning a rich set of visual invariances, we are also interested in viewpoint invariance for 3rd-person to 1st-person correspondence for imitation. How does the time-contrastive signal lead to disentangled representations? It does so by introducing competition between temporal neighbors to explain away visual changes over time. For example, in Fig. 1, since neighbors are visually similar, the only way to tell them apart is to model the amount of liquid present in the cup, or to model the pose of hands or objects and their interactions. Another way to understand the strong training signal that TCNs provide is to recognize the two constraints being simultaneously imposed on the model: along the view axis in Fig. 1 the model learns to explain what is common between images that look different, while along the temporal axis it learns to explain what is different between similar-looking images. Note that while the natural ability for imitation of this approach is important, its capability for learning rich representations without supervision is an even more significant contribution. The key ingredient in our approach is that multiple views ground and disambiguate the possible explanations for changes in the physical world. We show in Sec. IV that the TCN can indeed discover correspondences between different objects or bodies, as well as attributes such as liquid levels in cups and pouring stages, all without supervision. This is a somewhat surprising finding as no explicit correspondence between objects or bodies is ever provided. We hypothesize that manifolds of different but functionally similar objects naturally align in the embedding space, because they share some functionality and appearance.

Multi-view data collection is simple and can be captured with just two operators equipped with smartphones, as shown in Fig. 2. One operator keeps a fixed point of view of the region of interest while performing the task, while the other moves the camera freely to introduce the variations discussed above. While more cumbersome than single-view capture, we argue that multi-view capture is cheap, simple, and practical, when compared to alternatives such as human labeling.

We can also consider time-contrastive models trained on single-view video as shown in Fig. 3. In this case, the positive frame is randomly selected within a certain range of the anchor. A margin range is then computed given the positive range. Negatives are randomly chosen outside of the margin range and the model is trained as before. We empirically chose the margin range to be 2 times the positive range, which is itself set to $0.2s$. While we show in Sec. IV that multi-view TCN performs best, the single-view version can still be useful when no multi-view data is available.

B. Learning Robotic Behaviors with Reinforcement Learning

In this work, we consider an imitation learning scenario where the demonstrations come from a 3rd-person video

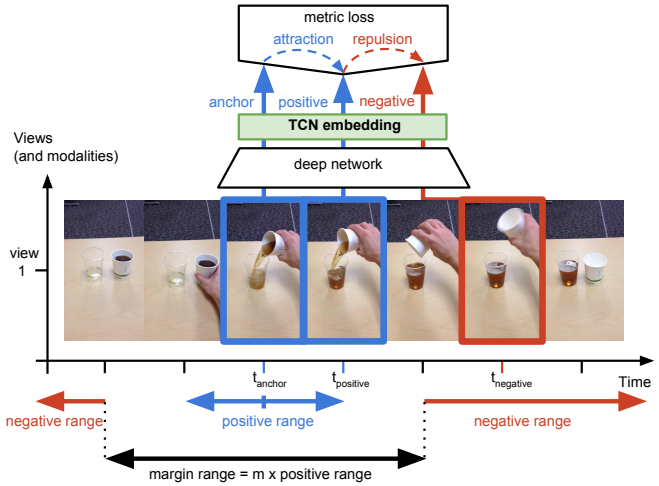

Fig. 3: **Single-view TCN:** positives are selected within a small window around anchors, while negatives are selected from distant timesteps in the same sequence.

observation of an agent with an embodiment that differs from the learning agent, e.g. robotic imitation of a human. Due to differences in the contexts, direct tracking of the demonstrated pixel values does not provide a sensible way of learning the imitation behavior. As described in the previous section, the TCN embedding provides a way to extract image features that are invariant to the camera angle and the manipulated objects, and can explain physical interactions in the world. We use this insight to construct a reward function that is based on the distance between the TCN embedding of a human video demonstration and camera images recorded with a robot camera. As shown in Sec. IV-B, by optimizing this reward function through trial and error we are able to mimic demonstrated behaviors with a robot, utilizing only its visual input and the video demonstrations for learning. Although we use multiple multi-view videos to train the TCN, the video demonstration consists only of a *single* video of a human performing the task from a random viewpoint.

Let $V = (\mathbf{v}_1, \dots \mathbf{v}_T)$ be the TCN embeddings of each frame in a video demonstration sequence. For each camera image observed during a robot task execution, we compute TCN embeddings $W = (\mathbf{w}_1, \dots \mathbf{w}_T)$. We define a reward function $R(\mathbf{v}_t, \mathbf{w}_t)$ based on the squared Euclidean distance and a Huber-style loss:

$$R(\mathbf{v}_t, \mathbf{w}_t) = -\alpha \|\mathbf{w}_t - \mathbf{v}_t\|_2^2 - \beta \sqrt{\gamma + \|\mathbf{w}_t - \mathbf{v}_t\|_2^2}$$

where α and β are weighting parameters (empirically chosen), and γ is a small constant. The squared Euclidean distance (weighted by α) gives us stronger gradients when the embeddings are further apart, which leads to larger policy updates at the beginning of learning. The Huber-style loss (weighted by β) starts prevailing when the embedding vectors are getting very close ensuring high precision of the task execution and fine-tuning of the motion towards the end of the training.

In order to learn robotic imitation policies, we optimize the reward function described above using reinforcement learning. In particular, for optimizing robot trajectories, we employ the PILQR algorithm [41]. This algorithm combines approximate model-based updates via LQR with fitted time-varying linear dynamics, and model-free corrections. We notice that in our tasks, the TCN embedding provides a well-behaved low-dimensional (32-dimensional in our experiments) representation of the state of the visual world in front of the robot. By including the TCN features in the system state (i.e. state = joint angles + joint velocities + TCN features), we can leverage the linear approximation of the dynamics during the model-based LQR update and significantly speed up the training. The details of the reinforcement learning setup can be found in Appendix B.

C. Direct Human Pose Imitation

In the previous section, we discussed how reinforcement learning can be used with TCNs to enable learning of object interaction skills directly from video demonstrations of humans. In this section, we describe another approach for using TCNs: direct imitation of human pose. While object interaction skills primarily require matching the functional aspects of the demonstration, direct pose imitation requires learning an implicit mapping between human and robot poses, and therefore involves a much more fine-grained association between frames. Once learned, a human-robot mapping could be used to speed up the exploration phase of RL by initializing a policy close to the solution.

We learn a direct pose imitation through self-regression. It is illustrated in Fig. 4 and Fig. 8 in the context of self-supervised human pose imitation. The idea is to directly predict the internal state of the robot given an image of itself. Akin to looking at itself in the mirror, the robot can regress its prediction of its own image to its internal states. We first train a shared TCN embedding by observing human and robots performing random motions. Then the robot trains itself with self-regression. Because it uses a TCN embedding that is invariant between humans and robots, the robot can then naturally imitate humans after training on itself. Hence we obtain a system that can perform end-to-end imitation of human motion, even though it was never given any human pose labels nor human-to-robot correspondences. We demonstrate a way to collect human supervision for end-to-end imitation in Fig. 4. However contrary to time-contrastive and self-regression signals, the human supervision is very noisy and expensive to collect. We use it to benchmark our approach in Sec. IV-C and show that large quantities of cheap supervision can effectively be mixed with small amounts of expensive supervision.

IV. Experiments

Our experiments aim to study three questions. First, we examine whether the TCN can learn visual representations that are more indicative of object interaction attributes, such as the stages in a pouring task. This allows us to comparatively evaluate the TCN against other self-supervised representations. Second, we study how the TCN can be used in conjunction with reinforcement learning to acquire

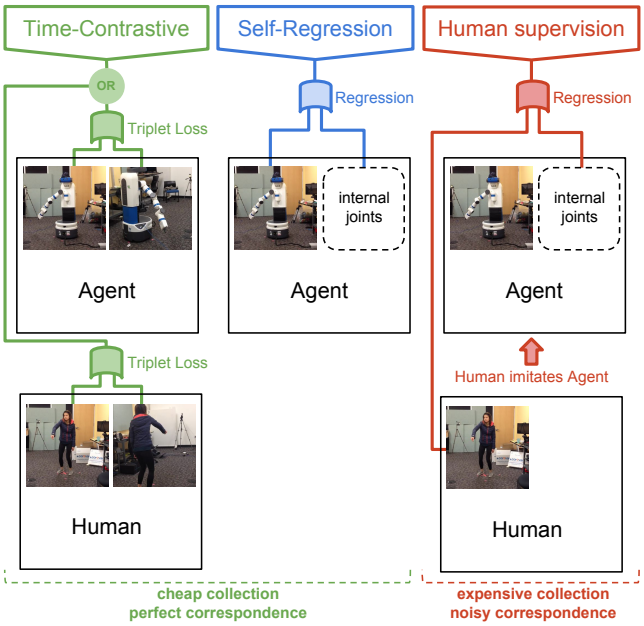

Fig. 4: **Training signals for pose imitation:** time-contrastive, self-regression and human supervision. The time-contrastive signal lets the model learn rich representations of humans or robots individually. Self-regression allows the robot to predict its own joints given an image of itself. The human supervision signal is collected from humans attempting to imitate robot poses.

complex object manipulation skills in simulation and on a real-world robotic platform. Lastly, we demonstrate that the TCN can enable a robot to perform continuous, real-time imitation of human poses without explicitly specifying any joint-level correspondences between robots and humans. Together, these experiments illustrate the applicability of the TCN representation for modeling poses, object interactions, and the implicit correspondences between robot imitators and human demonstrators.

A. Discovering Attributes from General Representations

1) Liquid Pouring: In this experiment, we study what the TCN captures simply by observing a human subject pouring liquids from different containers into different cups. The videos were captured using two standard smartphones (see Fig. 2), one from a subjective point of view by the human performing the pouring, and the other from a freely moving third-person viewpoint. Capture is synchronized across the two phones using an off-the-shelf app and each sequence is approximately 5 seconds long. We divide the collected multi-view sequences into 3 sets: 133 sequences for training (about 11 minutes total), 17 for validation and 30 for testing. The training videos contain clear and opaque cups, but we restrict the testing videos to clear cups only in order to evaluate if the model has an understanding of how full the cups are.

2) Models: In all subsequent experiments, we use a custom architecture derived from the Inception architecture [2] that is similar to [42]. It consists of the Inception model up until the layer "Mixed_5d" (initialized with ImageNet pre-trained weights), followed by 2 convolutional layers, a spatial softmax layer [42] and a fully-connected layer. The embedding is a fully connected layer with 32 units added on top of our custom model. This embedding is trained either with the multi-view TC loss, the single-view TC loss, or the shuffle & learn loss [31]. For the TCN models, we use the triplet loss from [40] without modification and with a gap value of 0.2. Note that, in all experiments, negatives always come from the same sequence as positives. We also experiment with other metric learning losses, namely npairs [43] and lifted structured [44], and show that results are comparable. We use the output of the last layer before the classifier of an ImageNet-pretrained Inception model [1, 2] (a 2048-dimensional vector) as a baseline in the following experiments, and call it "Inception-ImageNet". Since the custom model is initialized from ImageNet pre-training, it is a natural point of comparison which allows us to control for any invariances that are introduced through ImageNet training rather than other approaches. We compare TCN models to a shuffle & learn baseline trained on our data, using the same hyper-parameters taken from the paper (tmax of 60, tmin of 15, and negative class ratio of 0.75). Note that in our implementation, neither the shuffle & learn baseline nor TCN benefit from a biased sampling to high-motion frames. To investigate the differences between multi-view and single-view, we compare to a single-view TCN, with a positive range of 0.2 seconds and a negative multiplier of 2.

3) Model selection: The question of model selection arises in unsupervised training. Should you select the best model based on the validation loss? Or hand label a small validation for a given task? We report numbers for both approaches. In Table I we select each model based on the its lowest validation loss, while in Table IV we select based on a classification score from a small validation set labeled with the 5 attributes described earlier. As expected, models selected by validation classification score perform better on the classification task. However models selected by loss perform only slightly worse, except for shuffle & learn, which suffers a bigger loss of accuracy. We conclude that it is reasonable for TCN models to be selected based on validation loss, not using any labels.

4) Training time: We observe in Table IV that the multi-view TCN (using triplet loss) outperforms single-view models while requiring 15x less training time and while being trained on the exact same dataset. We conclude that taking advantage of temporal correspondences greatly improves training time and accuracy.

Method	**alignment error**	**classif. error**	**training iteration**
Random	28.1%	54.2%	-
Inception-ImageNet	29.8%	51.9%	-
shuffle & learn [31]	22.8%	27.0%	575k
single-view TCN (triplet)	25.8%	24.3%	266k
multi-view TCN (npairs)	18.1%	22.2%	938k
multi-view TCN (triplet)	18.8%	21.4%	397k
multi-view TCN (lifted)	18.0%	19.6%	119k

TABLE I: **Pouring alignment and classification errors:** all models are selected at their lowest validation loss. The classification error considers 5 classes related to pouring detailed in Table II.

5) Quantitative Evaluation: We present two metrics in Table I to evaluate what the models are able to capture. The alignment metric measures how well a model can semantically align two videos. The classification metric measures how well a model can disentangle pouring-related attributes, that can be useful in a real robotic pouring task. All results in this section are evaluated using nearest neighbors in embedding space. Given each frame of a video, each model has to pick the most semantically similar frame in another video. The "Random" baseline simply returns a random frame from the second video.

The sequence alignment metric is particularly relevant and important when learning to imitate, especially from a third-party perspective. For each pouring test video, a human operator labels the key frames corresponding to the following events: the first frame with hand contact with the pouring container, the first frame where liquid is flowing, the last frame where liquid is flowing, and the last frame with hand contact with the container. These keyframes establish a coarse semantic alignment which should provide a relatively accurate piecewise-linear correspondence between all videos. For any pair of videos (v_1, v_2) in the test set, we embed each frame given the model to evaluate. For each frame of the source video v_1, we associate it with its nearest neighbor in embedding space taken from all frames of v_2. We evaluate how well the nearest neighbor in v_2 semantically aligns with the reference frame in v_1. Thanks to the labeled alignments, we find the proportional position of the reference frame with the target video v_2, and compute the frame distance to that position, normalized by the target segment length.

We label the following attributes in the test and validation sets to evaluate the classification task as reported in Table II: is the hand in contact with the container? (yes or no); is the container within pouring distance of the recipient? (yes or no); what is the tilt angle of the pouring container? (values 90, 45, 0 and -45 degrees); is the liquid flowing? (yes or no); does the recipient contain liquid? (yes or no). These particular attributes are evaluated because they matter for imitating and performing a pouring task. Classification results are normalized by class distribution. Note that while this could be compared to a supervised classifier, as mentioned in the introduction, it is not realistic to expect labels for every possible task in a real application, e.g. in robotics. Instead, in this work we aim to compare to realistic general off-the-shelf models that one might use without requiring new labels.

In Table I, we find that the multi-view TCN model outperforms all baselines. We observe that single-view TCN and shuffle & learn are on par for the classification metric but not for the alignment metric. We find that general off-the-shelf Inception features significantly under-perform compared to other baselines. Qualitative examples and t-SNE visualizations of the embedding are available in Appendix C. We encourage readers to refer to supplementary videos to better grasp these results.

B. Learning Object Interaction Skills

In this section, we use the TCN-based reward function described in Sec. III-B to learn robotic imitation behaviors from third-person demonstrations through reinforcement learning. We evaluate our approach on two tasks, plate transfer in a simulated dish rack environment (Fig. 5, using the Bullet physics engine [45]) and real robot pouring from human demonstrations (Fig. 6).

Method	hand contact with container	within pouring distance	container angle	liquid is flowing	recipient has liquid
Random	49.9%	48.9%	74.5%	49.2%	48.4%
Imagenet Inception	47.4%	45.2%	71.8%	48.8%	49.2%
shuffle & learn	17.2%	17.8%	46.3%	25.7%	28.0%
single-view TCN (triplet)	12.6%	14.4%	41.2%	21.6%	31.9%
multi-view TCN (npairs)	8.0%	9.0%	35.9%	24.7%	35.5%
multi-view TCN (triplet)	7.8%	10.0%	34.8%	22.7%	31.5%
multi-view TCN (lifted)	7.8%	9.0%	35.4%	17.9%	27.7%

TABLE II: **Detailed attributes classification errors,** for model selected by validation loss.

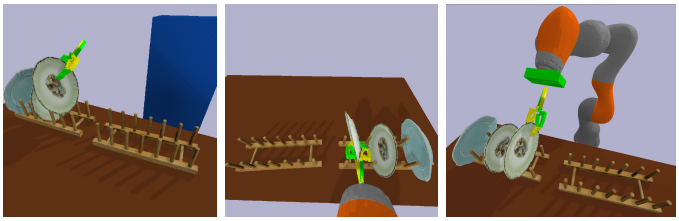

Fig. 5: **Simulated dish rack task.** *Left:* Third-person VR demonstration of the dish rack task. *Middle:* View from the robot camera during training. *Right:* Robot executing the dish rack task.

1) Task Setup: The simulated dish rack environment consists of two dish racks placed on a table and filled with plates. The goal of the task is to move plates from one dish rack to another without dropping them. This requires a complex motion with multiple stages, such as reaching, grasping, picking up, carrying, and placing of the plate. We record the human demonstrations using a virtual reality (VR) system to manipulate a free-floating gripper and move the plates (Fig. 5 left). We record the videos of the VR demonstrations by placing first-view and third-person cameras in the simulated world. In addition to demonstrations, we also record a range of randomized motions to increase the generalization ability of our TCN model. After recording the demonstrations, we place a simulated 7-DoF KUKA robotic arm inside the dish rack environment (Fig. 5 right) and attach a first-view camera to it. The robot camera images (Fig. 5 middle) are then used to compute the TCN reward function. The robot policy is initialized with random Gaussian noise.

For the real robot pouring task, we first collect the multi-view data from multiple cameras to train the TCN model. The training set includes videos of humans performing pouring of liquids recorded on smartphone cameras and videos of robot performing pouring of granular beads recorded on two robot cameras. We not only collect positive demonstrations of the task at hand, we also collect various interactions that do not actually involve pouring, such as moving cups around, tipping them over, spilling beads, etc, to cover the range of possible events the robot might need to understand. The pouring experiment analyzes how TCNs can implicit build correspondences between human and robot manipulation of objects. The dataset that we used to train the TCN consisted of $\sim$20 minutes of humans performing pouring tasks, as well as $\sim$20 additional minutes of humans manipulating cups and

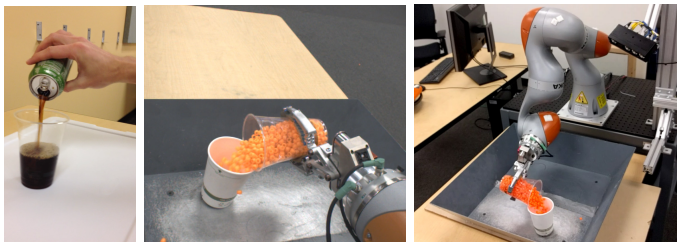

Fig. 6: **Real robot pouring task.** *Left:* Third-person human demonstration of the pouring task. *Middle:* View from the robot camera during training. *Right:* Robot executing the pouring task.

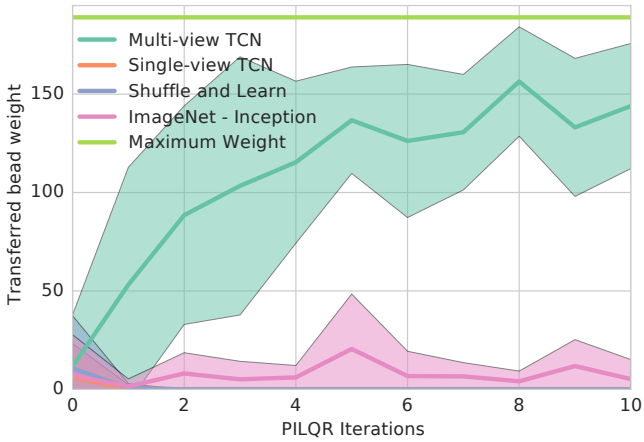

Fig. 7: **Learning progress of the pouring task,** using a single 3rd-person human demonstration, as shown in Fig. 6. This graph reports the weight in grams measured from the target recipient after each pouring action (maximum weight is 189g) along with the standard deviation of all 10 rollouts per iteration. The robot manages to successfully learn the pouring task using the multi-view TCN model after only 10 iterations.

bottles in ways other than pouring, such as moving the cups, tipping them over, etc. In order for the TCN to be able to represent both human and robot arms, and implicitly put them into correspondence, it must also be provided with data that allows it to understand the appearance of robot arms. To that end, we added data consisting of $\sim$20 minutes of robot arms manipulating cups in pouring-like settings. Note that this data does not necessarily need to itself illustrate successful pouring tasks: the final demonstration that is tracked during reinforcement learning consists of a human successfully pouring a cup of fluid, while the robot performs the pouring task with orange beads. However, we found that providing some clips featuring robot arms was important for the TCN to acquire a representation that could correctly register the similarities between human and robot pouring. Using additional robot data is justified here because it would not be realistic to expect the robot to do well while having never seen its own arm. Over time however, the more tasks are learned the less needed this should become. While the TCN is trained with approximately 1 hour of pouring-related multi-view sequences, the robot policy is only learned from a single liquid pouring video provided by a human (Fig. 6 left). With this video, we train a 7-DoF KUKA robot to perform the pouring of granular beads as depicted in Fig. 6 (right). We compute TCN embeddings from the robot camera images (Fig. 6 middle) and initialize the robot policy using random Gaussian noise. We set the initial exploration higher on the wrist joint as it contributes the most to the pouring motion (for all compared algorithms).

2) Quantitative Evaluation: Fig. 7 shows the pouring task performance of using TCN models for reward computation compared to the same baselines evaluated in the previous section. After each roll-out, we measure the weight of the beads in the receiving container. We perform runs of 10 roll-outs per iteration. Results in Fig. 7 are averaged over 4 runs per model (2 runs for 2 fixed random seeds). Already after the first several iterations of using the multi-view TCN model (mvTCN), the robot is able to successfully pour significant amount of the beads. After 10 iterations, the policy converges to a consistently successful pouring behavior. In contrast, the robot fails to accomplish the task with other models. Interestingly, we observe a low performance for single-view models (single-view TCN and shuffle & learn) despite being trained on the exact same multi-view data as mvTCN. We observe the same pattern in Fig. 12 when using a different human demonstration. This suggests taking advantage of multi-view correspondences is necessary in this task for correctly modeling object interaction from a 3rd-person perspective. The results show that mvTCN does provide the robot with suitable guidance to understand the pouring task. In fact, since the PILQR [41] method uses both model-based and model-free updates, the experiment shows that mvTCN not only provides good indicators when the pouring is successful, but also useful gradients when it isn't; while the other tested representations are insufficient to learn this task. This experiment illustrates how self-supervised representation learning and continuous rewards from visual demonstrations can alleviate the sample efficiency problem of reinforcement learning.

3) Qualitative Evaluation: As shown in our supplementary video, both dish rack and pouring policies converge to robust imitated behaviors. In the dish rack task, the robot is able to gradually learn all of the task components, including the arm motion and the opening and closing of the gripper. It first learns to reach for the plate, then grasp and pick it up and finally carry it over to another dish rack and place it there. The learning of this task requires only 10 iterations, with 10 roll-outs per iteration. This shows that the TCN reward function is dense and smooth enough to efficiently guide the robot to a complex imitation policy.

In the pouring task, the robot starts with Gaussian exploration by randomly rotating and moving the cup filled with beads. The robot first learns to move and rotate the cup towards the receiving container, missing the target cup and spilling large amount of the beads in the early iterations. After several more iterations, the robot learns to be more precise and eventually it is able to consistently pour most of the beads in the last iteration. This demonstrates that our method can efficiently learn tasks with non-linear dynamic object transitions, such as movement of the granular media and liquids, an otherwise difficult task to perform using

conventional state estimation techniques.

C. Self-Regression for Human Pose Imitation

In the previous section, we showed that we can use the TCN to construct a reward function for learning object manipulation with reinforcement learning. In this section, we also study how the TCN can be used to directly map from humans to robots in real time, as depicted in Fig. 8: in addition to understanding object interaction, we can use the TCN to build a pose-sensitive embedding either unsupervised, or with minimal supervision. The multi-view TCN is particularly well suited for this task because, in addition to requiring viewpoint and robot/human invariance, the correspondence problem is ill-defined and difficult to supervise. Apart from adding a joints decoder on top of the TCN embedding and training it with a self-regression signal, there is no fundamental difference in the method. Throughout this section, we use the robot joint vectors corresponding to the human-to-robot imitation described in Fig. 4 as ground truth. Human images are fed into the imitation system, and the resulting joints vector are compared against the ground truth joints vector.

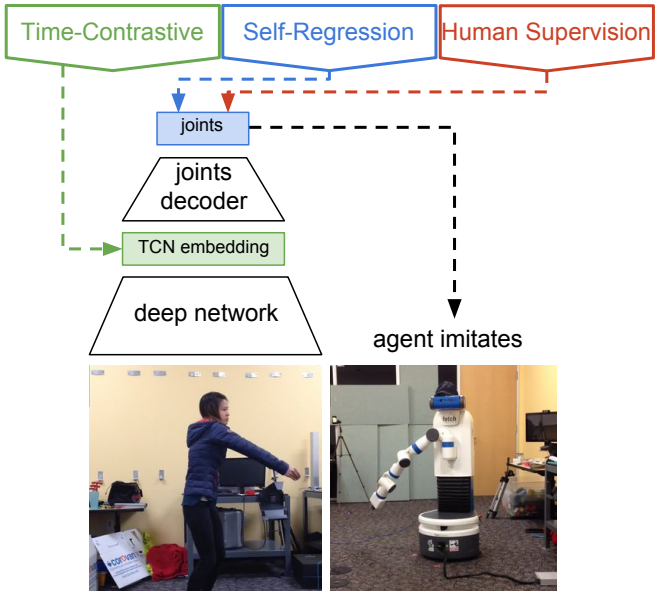

Fig. 8: **TCN for self-supervised human pose imitation:** architecture, training and imitation. The embedding is trained unsupervised with the time-contrastive loss, while the joints decoder can be trained with self-supervision, human supervision or both. Output joints can be used directly by the robot planner to perform the imitation. Human pose is never explicitly represented.

Supervision	**Robot joints distance error %**
Random (possible) joints	42.4 ± 0.1
Self	38.8 ± 0.1
Human	33.4 ± 0.4
Human + Self	33.0 ± 0.5
TC + Self	32.1 ± 0.3
TC + Human	29.7 ± 0.1
TC + Human + Self	$\mathbf{29.5} \pm 0.2$

TABLE III: **Imitation error for different combinations of supervision signals.** The error reported is the joints distance between prediction and groundtruth. Note perfect imitation is not possible.

By comparing different combinations of supervision signals, we show in Table III that training with all signals performs best. We observe that adding the time-contrastive signal always significantly improves performance. In general, we conclude that relatively large amounts of cheap weakly-supervised data and small amounts of expensive human supervised data is an effective balance for our problem. Interestingly, we find that the self-supervised model (TC+self) outperforms the human-supervised one. It should however be noted that the quantitative evaluation is not as informative here: since the task is highly subjective and different human subjects imitate the robot differently, matching the joint angles on held-out data is exceedingly difficult. We invite the reader to watch the accompanying video for examples of imitation, and observe that there is a close connection between the human and robot motion, including for subtle elements of the pose such as crouching: when the human crouches down, the robot lowers the torso via the prismatic joint in the spine. In the video, we observe a complex human-robot mapping is discovered entirely without human supervision. This invites to reflect on the need for intermediate human pose detectors when correspondence is ill-defined as in this case. In Fig. 14, we visualize the TCN embedding for pose imitation and show that pose across humans and robots is consistent within clusters, while being invariant to viewpoint and backgrounds. More analysis is available in Appendix D.

V. Conclusion

In this paper, we introduced a self-supervised representation learning method (TCN) based on multi-view video. The representation is learned by anchoring a temporally contrastive signal against co-occuring frames from other viewpoints, resulting in a representation that disambiguates temporal changes (e.g., salient events) while providing invariance to viewpoint and other nuisance variables. We show that this representation can be used to provide a reward function within a reinforcement learning system for robotic object manipulation, and to provide mappings between human and robot poses to enable pose imitation directly from raw video. In both cases, the TCN enables robotic imitation from raw videos of humans performing various tasks, accounting for the domain shift between human and robot bodies. Although the training process requires a dataset of multi-viewpoint videos, once the TCN is trained, only a single raw video demonstration is used for imitation. Limitations and future work are discussed in Appendix A.

Acknowledgments: We thank Mohi Khansari, Yunfei Bai, Erwin Coumans, Jonathan Tompson, James Davidson and Vincent Vanhoucke for helpful feedback and Ashwin Kakarla for helping labeling evaluation data. We thank everyone who provided imitations for this project: Phing Lee, Alexander Toshev, Anna Goldie, Deanna Chen, Deirdre Quillen, Dieterich Lawson, Eric Langlois, Ethan Holly, Irwan Bello, Jasmine Collins, Jeff Dean, Julian Ibarz, Ken Oslund, Laura Downs, Leslie Phillips, Luke Metz, Mike Schuster, Ryan Dahl, Sam Schoenholz and Yifei Feng.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[2] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[3] B. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[4] J.A. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *ICRA*, 2002.

[5] N.D. Ratliff, J.A. Bagnell, and S.S. Srinivasa. Imitation learning for locomotion and manipulation. In *Humanoids*, 2007. ISBN 978-1-4244-1861-9.

[6] K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. In *AAAI Fall Symposium: Robots Learning Interactively from Human Teachers*, volume FS-12-07, 2012.

[7] Y. Duan, M. Andrychowicz, B. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. *arXiv preprint arXiv:1703.07326*, 2017.

[8] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[9] S. Ross, G.J. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, volume 15, pages 627–635, 2011.

[10] P. Abbeel and A. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[11] S. Calinon, F. Guenter, and A. Billard. On learning, representing and generalizing a task in a humanoid robot. *IEEE Trans. on Systems, Man and Cybernetics, Part B*, 37(2):286–298, 2007.

[12] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *ICRA*, pages 763–768, 2009.

[13] B.C. Stadie, P. Abbeel, and I. Sutskever. Third-person imitation learning. *CoRR*, abs/1703.01703, 2017.

[14] A. Dragan and S. Srinivasa. Online customization of teleoperation interfaces. In *RO-MAN*, pages 919–924, 2012.

[15] P. Sermanet, K. Xu, and S. Levine. Unsupervised perceptual rewards for imitation learning. *CoRR*, abs/1612.06699, 2016.

[16] Y. Liu, A. Gupta, P. Abbeel, and S. Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. *CoRR*, abs/1707.03374, 2017.

[17] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. In *ICLR*, 2017.

[18] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, and C. Schmid. Local convolutional features with unsupervised training for image retrieval. In *ICCV*, pages 91–99, Dec 2015.

[19] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. *CoRR*, abs/1505.00687, 2015.

[20] R. Zhang, P. Isola, and A.A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. *CoRR*, abs/1611.09842, 2016.

[21] P. Vincent, H. Larochelle, Y. Bengio, and P.A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.

[22] V. Kumar, G. Carneiro, and I. D. Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimizing global loss functions. In *CVPR*, 2016.

[23] A. Owens, P. Isola, J.H. McDermott, A. Torralba, E.H. Adelson, and W.T. Freeman. Visually indicated sounds. *CoRR*, abs/1512.08512, 2015.

[24] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. *CoRR*, abs/1610.09001, 2016.

[25] C. Doersch, A. Gupta, and A.A. Efros. Unsupervised visual representation learning by context prediction. *CoRR*, abs/1505.05192, 2015.

[26] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, pages 4353–4361, 2015.

[27] D. Pathak, R.B. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. *CoRR*, abs/1612.06370, 2016.

[28] L. Wiskott and T.J. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Comput.*, 14(4): 715–770, April 2002. ISSN 0899-7667.

[29] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun. Unsupervised learning of spatiotemporally coherent metrics. In *ICCV*, 2015.

[30] B. Fernando, H. Bilen, E. Gavves, and S. Gould. Self-supervised video representation learning with odd-one-out networks. *CoRR*, abs/1611.06646, 2016.

[31] I. Misra, C.L. Zitnick, and Martial Hebert. Unsupervised learning using sequential verification for action recognition. *CoRR*, abs/1603.08561, 2016.

[32] K. Moo Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: learned invariant feature transform. *CoRR*, abs/1603.09114, 2016.

[33] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015.

[34] W.F. Whitney, M. Chang, T.D. Kulkarni, and J.B. Tenenbaum. Understanding visual concepts with continuation learning. *CoRR*, abs/1602.06822, 2016.

[35] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *CoRR*, abs/1511.05440, 2015.

[36] G. Mori, C. Pantofaru, N. Kothari, T. Leung, G. Toderici, A. Toshev, and W. Yang. Pose embeddings: A deep architecture for learning to match human poses. *CoRR*, abs/1507.00302, 2015.

[37] R. Stewart and S. Ermon. Label-free supervision of neural networks with physics and domain knowledge. *CoRR*, abs/1609.05566, 2016.

[38] V. Caggiano, L. Fogassi, G. Rizzolatti, J.K. Pomper, P. Thier, M.A. Giese, and A. Casile. View-based encoding of actions in mirror neurons of area f5 in macaque premotor cortex. *Current Biology*, 21(2):144–148, 2011.

[39] G. Rizzolatti and L. Craighero. The mirror-neuron system. *Annual Review of Neuroscience*, 27:169–192, 2004.

[40] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[41] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, and S. Levine. Combining model-based and model-free updates for trajectory-centric reinforcement learning. In *ICML*, 2017.

[42] C. Finn, X. Yu Tan, Y. Duan, Y. Darrell, S. Levine, and P. Abbeel. Learning visual feature spaces for robotic manipulation with deep spatial autoencoders. *CoRR*, abs/1509.06113, 2015.

[43] K. Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, pages 1857–1865, 2016.

[44] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. *CoRR*, abs/1511.06452, 2015.

[45] E. Coumans and Y. Bai. pybullet, a python module for physics simulation in robotics, games and machine learning. `http://pybullet.org/`, 2016–2017.

[46] S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *NIPS*, 2014.
[47] Y. Tassa, T. Erez, and E. Todorov. Synthesis and stabilization of complex behaviors. In *IROS*, 2012.
[48] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *JMLR*, 11, 2010.
[49] Y. Chebotar, M. Kalakrishnan, A. Yahya, A. Li, S. Schaal, and S. Levine. Path integral guided policy search. In *ICRA*, 2017.

APPENDIX

A. Future Work

One of the limitations of our approach is that the representation requires multi-viewpoint video for training, which is not as widely available (e.g. from the Internet) as standard video. We do analyze a single-viewpoint variant of our approach, and find that it also achieves improvement over baseline ImageNet-trained features, but that the multi-viewpoint TCN achieves substantially better results. However, as robots become more ubiquitous, recording multi-viewpoint video autonomously, for example by using stereo cameras, seems like a promising direction. Our method can also be viewed as a specific instance of a more general class of multi-modal embedding methods, where temporally co-occuring events in multiple sensory modalities are embedded into the same space. Viewed in this light, the exploration of broader modalities, such as audio and tactile sensing, would be an exciting avenue for future work. Another limitation in our experiments is that we train a separate TCN for each task (pouring, pose imitation, etc.). While the TCN for a given task, such as pouring, is trained on videos that include clips of failed pouring, moving cups, and so on, the embedding is only used to learn a single task, namely pouring. In principle, the embeddings learned by the TCN should be task agnostic, though considerably larger datasets may be required in this case. An interesting avenue for future work would be to study how multiple tasks could be embedded in the same space, essentially creating a universal representation for imitation of object interaction behaviors. While in this paper we explored learning representations using time as a supervision signal, in the future, models should learn simultaneously from a collection of diverse yet complementary signals.

B. Reinforcement Learning Details

Let $p(\mathbf{u}_t|\mathbf{x}_t)$ be a robot policy, which defines a probability distribution over actions $\mathbf{u}_t$ conditioned on the system state $\mathbf{x}_t$ at each time step t of a task execution. We employ policy search to optimize the policy parameters θ. Let $\tau = (\mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_T, \mathbf{u}_T)$ be a trajectory of states and actions. Given a cost function[1] $c(\mathbf{x}_t, \mathbf{u}_t)$, we define the trajectory cost as $c(\tau) = \sum_{t=1}^{T} c(\mathbf{x}_t, \mathbf{u}_t)$. The policy is optimized with respect to the expected cost of the policy

$$J(\theta) = \mathbb{E}_p\left[c(\tau)\right] = \int c(\tau)p(\tau)d\tau\,,$$

where $p(\tau)$ is the policy trajectory distribution given the system dynamics $p\left(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t\right)$

$$p(\tau) = p(\mathbf{x}_1)\prod_{t=1}^{T} p\left(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t\right)p(\mathbf{u}_t|\mathbf{x}_t)\,.$$

One policy class that allows us to employ very efficient reinforcement learning methods is the time-varying linear-Gaussian (TVLG) controller $p(\mathbf{u}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{K}_t\mathbf{x}_t + \mathbf{k}_t, \mathbf{\Sigma}_t)$. In this work, we apply a reinforcement learning method PILQR[41] to learn these TVLG controllers on a real robot, which combines model-based and model-free policy updates for an efficient learning of tasks with complex system dynamics.

Let $S(\mathbf{x}_t, \mathbf{u}_t) = c(\mathbf{x}_t, \mathbf{u}_t) + \sum_{j=t+1}^{T} c(\mathbf{x}_j, \mathbf{u}_j)$ be the cost-to-go of a trajectory starting in state $\mathbf{x}_t$ by performing action $\mathbf{u}_t$ and following the policy $p(\mathbf{u}_t|\mathbf{x}_t)$ afterwards. In each iteration i, PILQR performs a KL-constrained optimization of $S(\mathbf{x}_t, \mathbf{u}_t)$:

$$\min_{p^{(i)}} \mathbb{E}_{p^{(i)}}[S(\mathbf{x}_t, \mathbf{u}_t)] \; s.t. \; \mathbb{E}_{p^{(i-1)}}\Big[D_{\mathrm{KL}}\left(p^{(i)} \| \, p^{(i-1)}\right)\Big] \leq \epsilon,$$

where limiting the KL-divergence between the new policy $p^{(i)}$ and the old policy $p^{(i-1)}$ leads to a better convergence behavior. The optimization is divided into two steps. In the first step, we perform a fast model-based update using an algorithm LQR-FLM [46], which is based on the iterative linear-quadratic regulator (iLQR) [47] and approximates $S(\mathbf{x}_t, \mathbf{u}_t)$ with a linear-quadratic cost $\hat{S}(\mathbf{x}_t, \mathbf{u}_t)$. In the second step, the residual cost-to-go $\tilde{S}(\mathbf{x}_t, \mathbf{u}_t) = S(\mathbf{x}_t, \mathbf{u}_t) - \hat{S}(\mathbf{x}_t, \mathbf{u}_t)$ is optimized using a model-free method PI^2 [48, 49] to produce an unbiased policy update.

C. Objects Interaction Analysis

Here, we visualize the embeddings from the ImageNet-Inception and multi-view TCN models with t-SNE using a coloring by groundtruth labels. Each color is a unique combination of 5 attribute values defined earlier, i.e. if each color is well separated the model can identify uniquely all possible combinations of our 5 attributes. Indeed we observe in Fig. 11 some amount of color separation for the TCN but not for the baseline.

Method	**alignment error**	**classif. error**	**training iteration**
Random	28.1%	54.2%	-
Inception-ImageNet	29.8%	51.9%	-
single-view TCN (triplet)	26.6%	23.6%	738k
shuffle & learn [31]	20.9%	23.2%	743k
multi-view TCN (lifted)	19.1%	21.4%	927k
multi-view TCN (triplet)	17.5%	20.3%	47k
multi-view TCN (npairs)	17.5%	19.3%	224k

TABLE IV: **Pouring alignment and classification errors:** these models are selected using the classification score on a small labeled validation set, then ran on the full test set. We observe that multi-view TCN outperforms other models with 15x shorter training time. The classification error considers 5 classes related to pouring: "hand contact with recipient", "within pouring distance", "container angle", "liquid is flowing" and "recipient fullness".

[1]We use a cost function instead of the reward function as it is more common in the trajectory optimization theory.

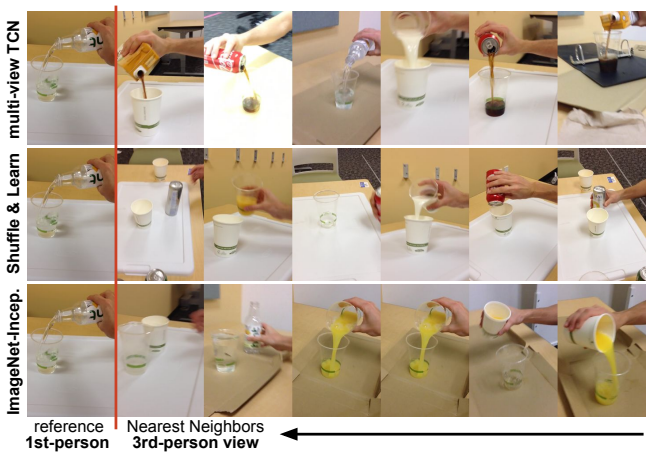

Fig. 9: **Label-free pouring imitation**: nearest neighbors (right) for each reference image (left) for different models (multi-view TCN, shuffle & learn and ImageNet-Inception). These pouring test images show that **the TCN model can distinguish different hand poses and amounts of poured liquid simply from unsupervised observation** while being invariant to viewpoint, background, objects and subjects, motion-blur and scale.

Fig. 10: **Label-free pose imitation**: nearest neighbors (right) for each reference frame (left) for each row. Although only trained with self-supervision (no human labels), the multi-view TCN can understand correspondences between humans and robots for poses such as crouching, reaching up and others while being invariant to viewpoint, background, subjects and scale.

Fig. 11: **t-SNE colored by attribute combinations:** TCN (bottom) does a better job than ImageNet-Inception (top) at separating combinations of attributes.

D. Pose Imitation Analysis

1) Pose Imitation Data: The human training data consists of sequences distinguished by human subject and clothing pair. Each sequence is approximately 4 minutes. For the label-free TC supervision we collected approximately 30 human pairs (about 2 hours) where humans imitate a robot but the joint labels are not recorded, along with 50 robot sequences with random motion (about 3 hours, trivial to collect). For human supervision, we collected 10 human/clothing pairs (about 40 minutes, very expensive collection) while also recording the joints labels. Each recorded sequence is captured by 3 smartphone cameras fixed on tripods at specific angles (0°, 60° and 120°) and distance. The validation and testing sets each consist of 6 human/clothing pairs not seen during training (about 24 minutes, very expensive collection).

The distance error is normalized by the full value range of each joint, resulting in a percentage error. Note that the Human Supervision signal is quite noisy, since the imitation task is subjective and different human subjects interpret the mapping differently. In fact, a perfect imitation is not possible due to physiological differences between human bodies and the Fetch robot. Therefore, the best comparison metric available to us is to see whether the joint angles predicted from a held-out human observation match the actual joint angles that the human was attempting to imitate.

2) Models: We train our model using the 3 different signals as described in Fig. 4. The model consists of a TCN as described in Sec. IV-A.2, to which we add a joint decoder network (2 fully-connected layers above TC embedding: $\rightarrow 128 \rightarrow 8$, producing 8 joint values). We train the joints decoder with L2 regression using the self-supervision or human supervision signals. The model can be trained with different combinations of signals; we study the effects of each combination in section Fig. 13. The datasets used are

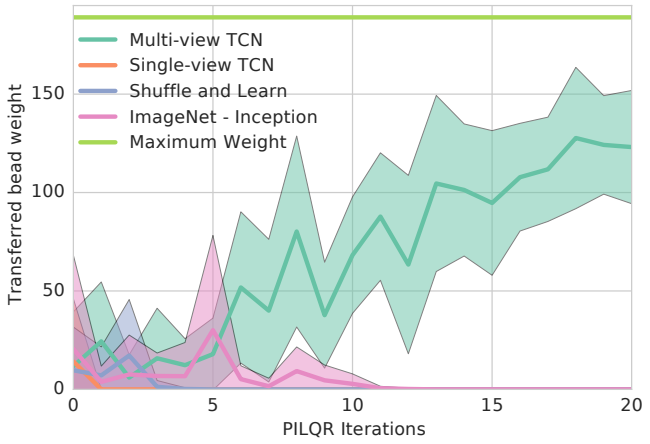

Fig. 12: **Learning progress of the pouring task,** using a single 3rd-person human demonstration that is different that the one shown in Fig. 6. This graph reports the weight in grams measured from the target recipient after each pouring action (maximum weight is 189g) along with the standard deviation of all 10 rollouts per iteration. The robot manages to successfully learn the pouring task using the multi-view TCN model after 20 iterations.

approximately 2 hours of random human motions, 3 hours of random robot motions and 40 minutes of human supervision, as detailed in Appendix D.1. At test time, the resulting joints vector can then directly be fed to the robot stack to update its joints (using the native Fetch planner) as depicted in Fig. 8. This results in an end-to-end imitation from pixels to joints without any explicit representation of human pose.

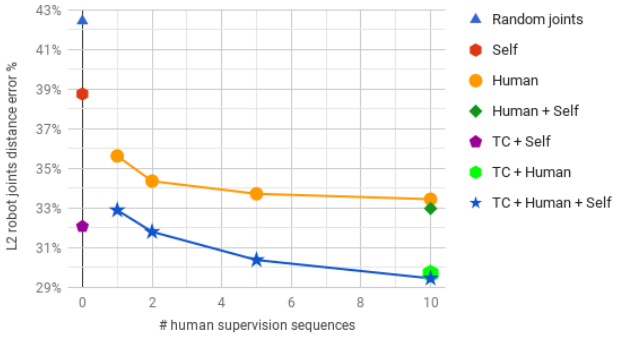

Fig. 13: **Comparing types and amounts of supervision:** Self-supervised imitation ("TC+self") outperforms human-supervised imitation ("Human").

3) Supervision Analysis: As shown in Fig. 8, our imitation system can be trained with different combinations of signals. Here we study how our self-supervised imitation system compares to the other possible combinations of training signals. The performance of each combination is reported in Table III using the maximum amounts of data available (10 sequences for Human supervision and 30 sequences for TC supervision), while Fig. 13 varies the amount of human supervision. Models such as the "TC + Self" or "Self" do not make use of any human supervision, hence only appear as single points on the vertical axis. Models that do not include TC supervision are simply trained as end-to-end regression problems. For example the Self model is trained end-to-end to predict internal joints from third person observations of the robot, and then that model is applied directly to the human imitation task. For reference, we compute a random baseline which samples joints values within physically possible ranges. In general, we observe that more human supervision decreases the L2 robot joints error. It is interesting to note that while not given any labels, the self-supervised model ("TC + Self") still significantly outperforms the fully-supervised model ("Human"). The combination of all supervision signals performs the best. Overall, we observe that adding the TC supervision to any other signal significantly decreases the imitation error. In Fig. 15, we vary the amount of TC supervision provided and find the imitation error keeps decreasing as we increase the amount of data. Based on these results, we can make the argument that relatively large amounts of cheap weakly-supervised data and small amounts of expensive human supervised data is an effective balance for our problem. A non-extensive analysis of viewpoint and scale invariance in Sec. E seems to indicate that the model remains relatively competitive when presented with viewpoints and scales not seen during training.

4) Analysis by Joint: In Fig. 16, we examine the error of each joint individually for 4 models. Interestingly, we find that for all joints excepts for "shoulder pan", the unsupervised "TC+Self" models performs almost as well as the human-supervised "TC+Human+Self". The unsupervised model does not seem to correctly model the shoulder pan and performs worse than Random. Hence most of the benefits of human supervision found in Fig. 13 come from correcting the shoulder pan prediction.

5) Qualitative Results: We offer multiple qualitative evaluations: k-Nearest Neighbors (kNN) in Fig. 10, imitation strips in Fig. 21 and a t-SNE visualization in Fig. 14. Video strips do not fully convey the quality of imitation, we strongly encourage readers to watch the videos accompanying this paper. **kNN:** In Fig. 10, we show the nearest neighbors of the reference frames for the self-supervised model "TC+Self" (no human supervision). Although never trained across humans, it learned to associate poses such as crouching or reaching up between humans never seen during training and with entirely new backgrounds, while exhibiting viewpoint, scale and translation invariance. **Imitation strips:** In Fig. 21, we present an example of how the self-supervised model has learned to imitate the height level of humans by itself (easier to see in supplementary videos) using the "torso" joint (see Fig. 16). This stark example of the complex mapping between human and robot joints illustrates the need for learned mappings, here we learned a non-linear mapping from many human joints to a single "torso" robot joint without any human supervision. **t-SNE:** We qualitatively evaluate the arrangement of our learned embedding using t-SNE representations with perplexity of 30 and learning rate of 10. In Fig. 14, we show that the agent-colored embedding exhibits local coherence with respect to pose while being invariant to agent and viewpoint. More kNN examples, imitation strips and t-SNE visualizations from different models are available in Sec. F.

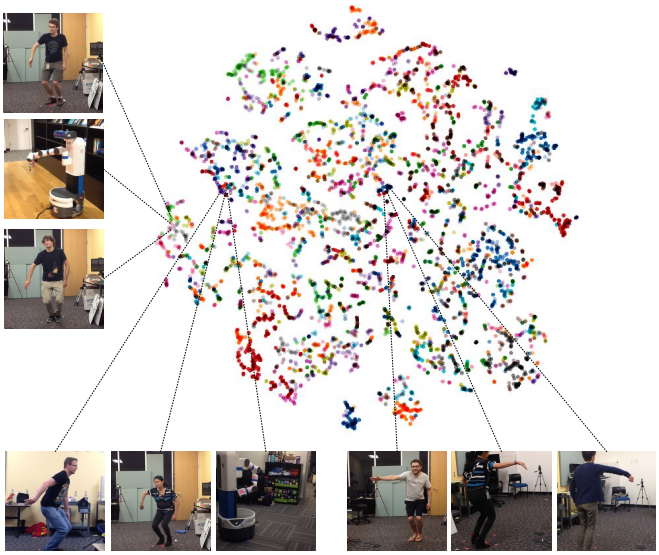

Fig. 14: **t-SNE embedding colored by agent for model "TC+Self".** We show that images are locally coherent with respect to pose while being invariant to agent or viewpoint.

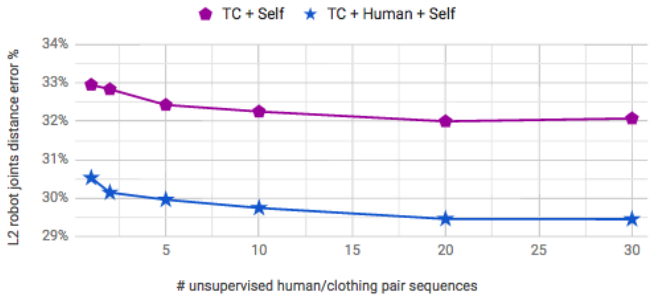

Fig. 15: **Varying the amount of unsupervised data:** increasing the number of unsupervised sequences decreases the imitation error for both models.

E. Imitation Invariance Analysis

In this section, we explore how much invariance is captured by the model. In Fig. 19, we test the L2 imitation error from new viewpoints (30°, 90° and 150°) different from training viewpoints (0°, 60° and 120°). We find that the error increases but the model does not break down and keeps a lower accuracy than the Human model in Fig. 13. We also evaluate in Fig. 20 the accuracy while bringing the camera closer than during training (about half way) and similarly find that while the error increases, it remains competitive and lower than the human supervision baseline. From these experiments, we conclude that the model is somewhat robust to viewpoint changes (distance and orientation) even though it was trained with only 3 fixed viewpoints.

F. Imitation Examples

t-SNE: We qualitatively evaluate the arrangement of our learned embedding using t-SNE representations with perplexity of 30 and learning rate of 10. In this section we show the embedding before and after training, and colorize points by agent in Fig. 18 and by view in Fig. 17. The representations show that the embedding initially clusters

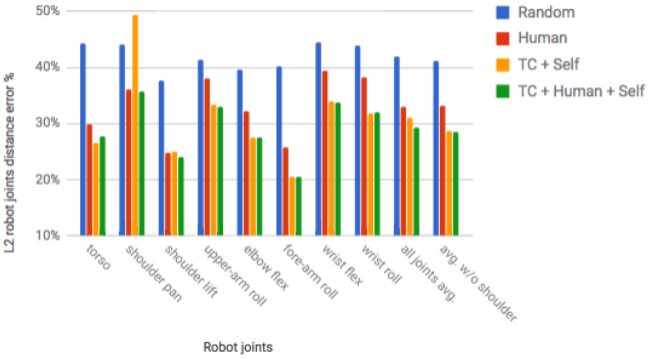

Fig. 16: **L2 robot error break-down by robot joints.** From left to right, we report errors for the 8 joints of the Fetch robot, followed by the joints average, followed by the joints average excluding the "shoulder pan" join.

views and agents together, while after training points from a same agent or view spread over the entire manifold, indicating view and agent invariance.

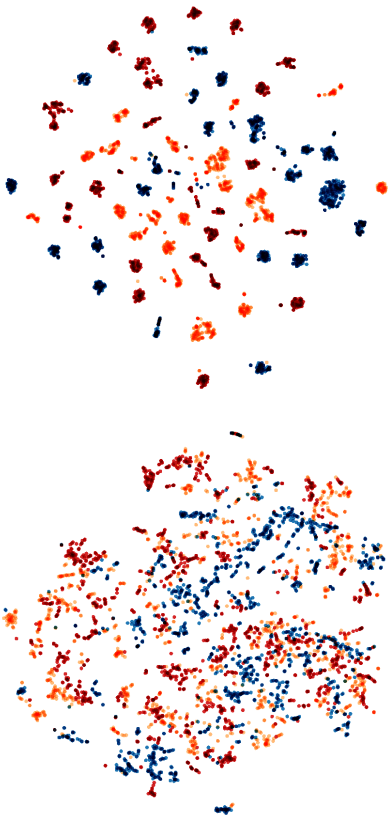

Fig. 17: **t-SNE embedding before (top) and after (bottom) training, colored by view.** Before training, we observe concentrated clusters of the same color, indicating that the manifold is organized in a highly view-specific way, while after training each color is spread over the entire manifold.

Fig. 18: **t-SNE embedding before (top) and after (bottom) training, colored by agent.** Before training, we observe concentrated clusters of the same color, indicating that the manifold is organized in a highly agent-specific way, while after training each color is spread over the entire manifold.

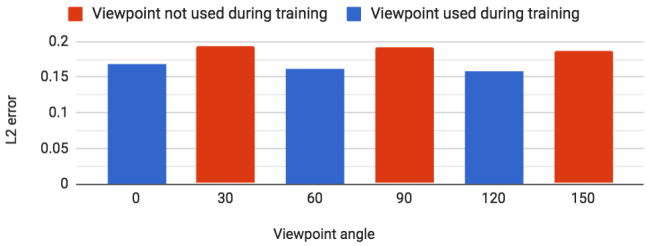

Fig. 19: **Testing TC+Human+Self model for orientation invariance:** while the error increases for viewpoints not seen during training (30°, 90° and 150°), it remains competitive.

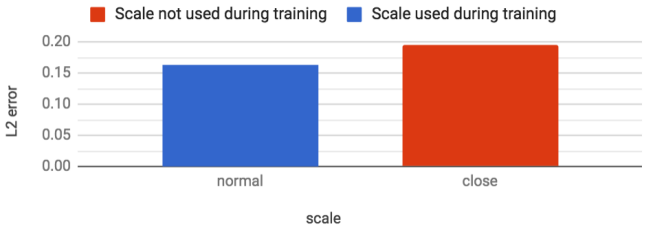

Fig. 20: **Testing for scale invariance:** while the error increases when decreasing the distance of the camera to the subject (about half way compared to training), it remains competitive and lower than the human-supervised baseline.

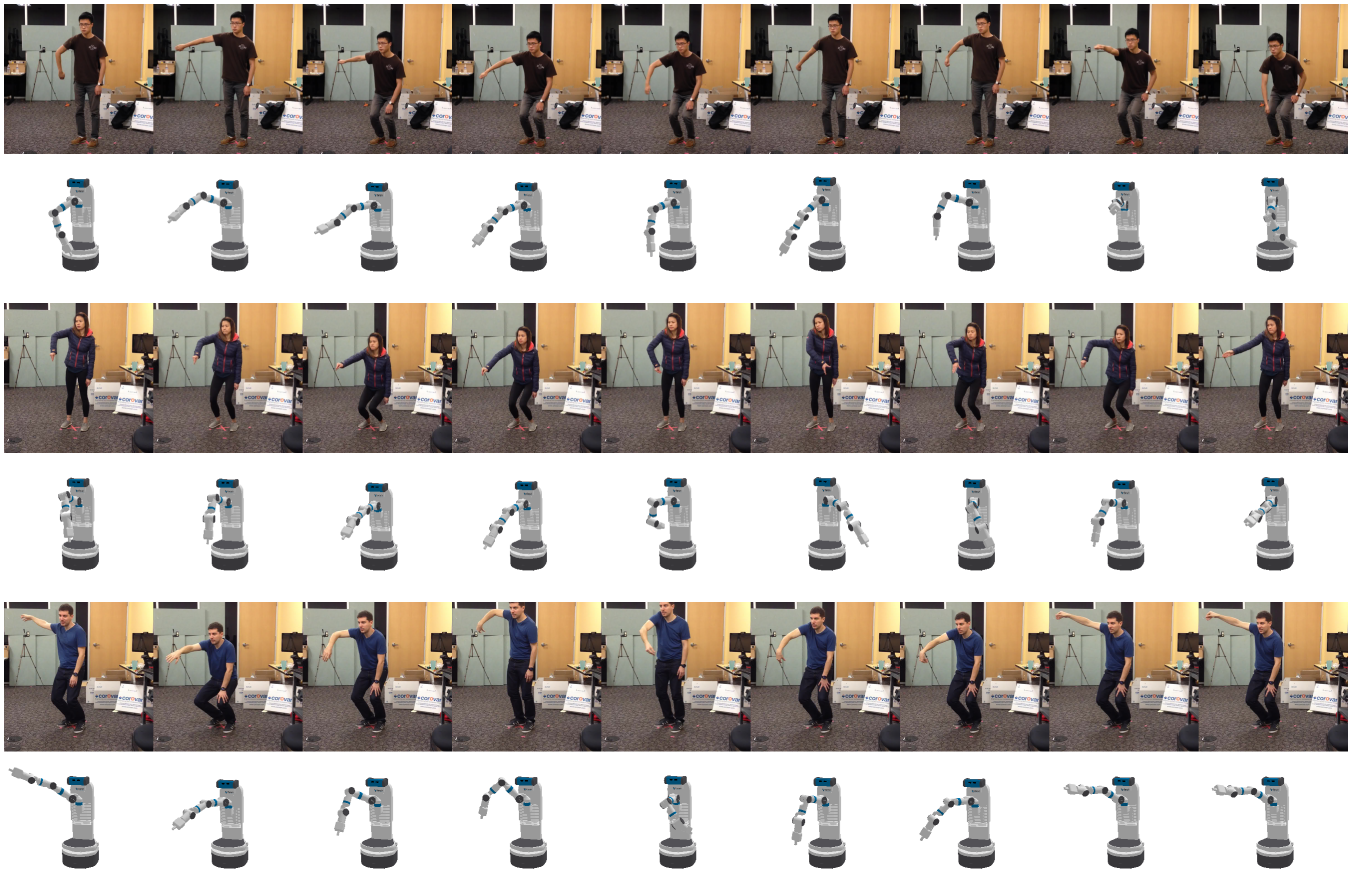

Fig. 21: **Self-supervised imitation examples.** Although not trained using any human supervision (model "TC+Self"), the TCN is able to approximately imitate human subjects unseen during training. Note from the rows (1,2) that the TCN discovered the mapping between the robot's torso joint (up/down) and the complex set of human joints commanding crouching. In rows (3,4), we change the capture conditions compared to training (see rows 1 and 2) by using a free-form camera motion, a close-up scale and introduction some motion-blur and observe that imitation is still reasonable.