

Any-point Trajectory Modeling for Policy Learning

Chuan Wen^{*1,2,5} Xingyu Lin^{*1} John So^{*3}
 Kai Chen⁶ Qi Dou⁶ Yang Gao^{2,4,5} Pieter Abbeel¹
¹UC Berkeley ²IIS, Tsinghua University ³Stanford University
⁴Shanghai AI Laboratory ⁵Shanghai Qi Zhi Institute ⁶ CUHK

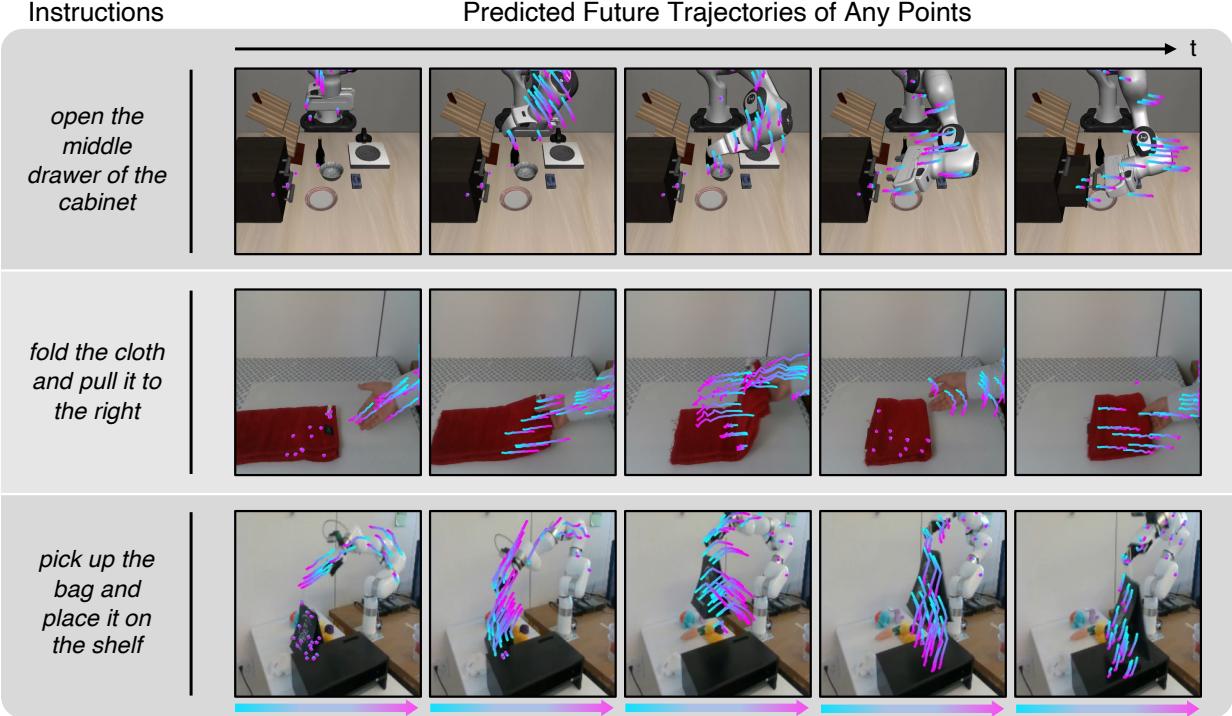


Fig. 1: Given a task instruction and the initial positions of any set of points in an image frame, our Any-point Trajectory Model (ATM) can predict the future trajectories of these points conditioned on the task. After training the model on an action-free video dataset, the predicted trajectories serve as effective guidance for learning visuomotor policies for a set of language-conditioned manipulation tasks.

Abstract—Learning from demonstration is a powerful method for teaching robots new skills, and having more demonstration data often improves policy learning. However, the high cost of collecting demonstration data is a significant bottleneck. Videos, as a rich data source, contain knowledge of behaviors, physics, and semantics, but extracting control-specific information from them is challenging due to the lack of action labels. In this work, we introduce a novel framework, Any-point Trajectory Modeling (ATM), that utilizes video demonstrations by pre-training a trajectory model to predict future trajectories of arbitrary points within a video frame. Once trained, these trajectories provide detailed control guidance, enabling the learning of robust visuomotor policies with minimal action-labeled data. Across over 130 language-conditioned tasks we evaluated in both simulation and the real world, ATM outperforms strong video pre-training

^{*}First three authors contributed equally: Chuan Wen led the implementation and experiments. Xingyu Lin came up with the idea, supervised the technical development, and contributed to model debugging. John So implemented the Robot-to-robot transfer experiments and UniPi baselines.

baselines by 80% on average. Furthermore, we show effective transfer learning of manipulation skills from human videos and videos from a different robot morphology. Visualizations and code are available at: <https://xingyu-lin.github.io/atm>.

I. INTRODUCTION

Computer vision and natural language understanding have made significant advances in recent years [22, 7], where the availability of large datasets plays a critical role. Similarly, in robotics, scaling up human demonstration data has been key for learning new skills [6, 34, 14], with a clear trend of performance improvement with larger datasets [29, 6]. However, human demonstrations, typically action-labeled trajectories collected via teleoperation devices [56, 53], are time-consuming and labor-intensive to collect. For instance, collecting 130K trajectories in [6] took 17 months, making data collection a major bottleneck in robot learning.

Videos contain knowledge about behaviors, physics, and semantics, presenting an alternative data source. However, the lack of action labels makes utilization of video data in policy learning difficult. Previous works have addressed this by using self-supervised objectives for video pre-training to learn a feature representation of the observation for policy learning [44, 33, 42]. However, a feature representation only describes the state at the current time step, largely neglecting the transition dynamics that predicts future states. To explicitly model the transition dynamics, prior works have developed video prediction models that predict future image frames from current ones to guide policy learning [12, 55, 13]. However, learning a video prediction model for control introduces two challenges. Firstly, the task of video prediction avoids any abstraction by modeling changes to every pixel, coupling the physical motion with visual appearances such as texture, and lighting. This coupling makes modeling difficult, often resulting in hallucinations and unrealistic future predictions [12]. Secondly, these models are computationally demanding in both training and inferencing. With limited computational resources, performance significantly declines. Moreover, the high inference cost compels these models to adopt open-loop execution [12, 5], which tends to result in less robust policies.

In this paper, we propose a novel and structured representation to bridge video pre-training and policy learning. We first represent each state as a set of points in a video frame. To model the temporal structure in videos, we learn an Any-point Trajectory Model (ATM) that takes the positions of the points in the current frame as input and outputs their future trajectories. We predict these trajectories in the camera coordinate frame to minimize the assumptions on calibrated cameras. These 2D point trajectories correspond to trajectories of particles in the 3D space and are a universal representation of the motions that can transfer to different domains and tasks. Contrasting with the video prediction approach of tracking changes in pixel intensity, the particle-based trajectory modeling offers a faithful abstraction of the physical dynamics, naturally incorporating inductive biases like object permanence and continuous motion. We first pre-train the trajectory model on actionless video datasets. After pre-training, the predicted trajectories can function as subgoals to guide the control policy. We then train trajectory-guided policies using only a minimal amount of action-labeled demonstration data. For training the ATMs, we generate self-supervised training data by leveraging recent advancements in vision models for accurate point tracking [20]. Across over **130** language-conditioned tasks we evaluated in both simulation and the real world, ATM significantly surpasses various strong baselines in video pre-training, achieving an average success rate of 63% compared to the highest success rate of 37% by previous methods, marking an improvement of over 80%. Additionally, we demonstrate that our method facilitates effective transfer learning from human videos and videos of a robot with a different morphology. We summarize our main contributions below:

- 1) We propose an Any-point Trajectory Model, a simple and novel framework that bridges video pre-training to policy learning, leveraging the structured representation of particle trajectories.
- 2) Through extensive experiments on simulated benchmarks and in the real world, we demonstrate that our method can effectively utilize video data in pre-training and significantly outperform various video pre-training baselines in an imitation learning setting.
- 3) We demonstrate effective learning from cross-embodiment human and robot videos.

II. RELATED WORK

State representation for control. In learning end-to-end visuomotor policies, the policy is typically parameterized as a neural network that takes image observation as the input representation [38, 6, 49]. Due to the lack of inductive bias, these approaches require training on a large number of demonstration trajectories, which is expensive to collect. On the other hand, different structured representations are proposed to improve the data efficiency, such as key points [36, 31], mesh [25, 19], or neural 3D representation [24, 37]. However, prior structures often limit the policy to specific tasks. In contrast, we propose to utilize future trajectories of arbitrary points in the image as additional input to the policy, making minimal assumptions about the task and environment. We demonstrate its wide application to a set of over 130 language-conditioned manipulation tasks. In navigation and locomotion, it is common to construct policies that are guided by the future trajectories of the robot [1, 35]. In manipulation, some works have explored flow-based guidance [15, 41, 16]. However, prior works only track task-specific points, such as the end-effector of the robot, or the human hand. Instead, our approach works with arbitrary points, including points on the objects, thus providing richer information in the more general settings. Finally, Vecerik et al. [48] proposes to utilize any-point tracking for few-shot policy learning. This approach does not learn trajectory models from data, but instead mainly uses the tracker to perform visual servoing during test time. This design choice requires more instrumentation, such as separating the task into multiple stages, predicting the goal locations of the points for each stage, and running the tracker during inference time. In contrast, we present a much simpler framework, enabling application in more diverse settings.

Video pre-training for control. Videos contain rich information about behaviors and dynamics, which can help policy learning. However, video pre-training remains challenging due to the lack of action labels. One line of works first learns an inverse dynamics model that predicts the action from two adjacent frames and then labels the videos with pseudo actions [39, 3, 47, 40]. However, the inverse dynamics model is often trained on a limited action-labeled dataset and does not generalize well, especially for continuous actions. Prior works have also explored pre-train a feature representation using various self-supervised objectives [43, 33, 28], but the representation alone does not retain the temporal information

in videos. More recently, learning video prediction as pre-training has shown promising results [12, 23, 4, 55, 13]. During policy learning, a video prediction model is used to generate future subgoals and then a goal-conditioned policy can be learned to reach the sub-goal. However, video prediction models often result in hallucinations and unrealistic physical motions. Furthermore, video models require extensive computation, which is an issue, especially during inference time. In contrast, our method models the trajectories of the points, naturally incorporating inductive bias like object permanence while requiring much less computation. This enables our trajectory models to be run closed-loop during policy execution. Furthermore, the trajectories provide dense guidance to the policy as a motion prior.

Learning from Human Videos. Of particular interest, numerous prior works have proposed learning from the rich source of human videos [54, 45, 32, 2, 46, 50]. However, these works often explicitly extract the hand pose or contact regions from the human videos, thereby losing information about the dynamics of the remaining objects. In contrast, our method models the trajectories of arbitrary points and can learn from both human videos and videos of a different robot.

III. PRELIMINARY

In this paper, we aim to learn robust control policies from a small set of action-labeled demonstration trajectories. Our central goal is to leverage the more scalable, unlabeled videos as a data source for pre-training.

Imitation from demos and videos. To begin with, we denote the action-free video dataset as $\mathcal{T}_o = \{(\tau_o^{(i)}, \ell^{(i)})\}_{i=1}^{N_o}$, where $\ell^{(i)}$ is the language instruction for the i^{th} episode and $\tau_o^{(i)} = \{o_t^{(i)}\}_{t=1}^T$ denotes the observation-only trajectory consisting of camera images. Similarly, we denote the demonstration dataset as $\mathcal{T}_a = \{(\tau_a^{(i)}, \ell^{(i)})\}_{i=1}^{N_a}$, where $\tau_a^{(i)} = \{o_t^{(i)}, a_t^{(i)}\}_{t=1}^T$ is the action-labeled trajectory. During imitation learning, our goal is to learn a policy π_θ , parameterized by θ to mimic the expert behavior by the following behavioral cloning objective:

$$\pi_\theta^* = \arg \min_{\theta} E_{(o_t, a_t, \ell) \sim \mathcal{T}_a} [\mathcal{L}(\pi_\theta(o_t, \ell), a_t)], \quad (1)$$

where \mathcal{L} is the loss function, which could be Mean Squared Error (MSE) or cross-entropy loss.

Tracking Any Point (TAP). The recent advancements in video tracking [17, 10, 11, 51, 57] enable us to track the trajectory of each point in video frames without external supervision. In this paper, we utilize the off-the-shelf tracker proposed in [20]. Formally, given a sequence of images from a video o_1, \dots, o_T , any one of the time steps $\bar{t} \in [1, T]$, and a set of points in that frame $\{p_{\bar{t}, k}\}_{k=1}^K$, where $p_{\bar{t}, k} = (x, y)$ is the point coordinate in the camera frame, the task of tracking is to predict the 2D camera-frame coordinates of the corresponding points in every frame $p_{t, k}$ where $t = 1 \dots T$. In this paper, we use the terms *trajectory* and *track* interchangeably to refer to a sequence of 2D coordinates of any point, denoted as (p_1, \dots, p_T) . We only model the 2D trajectories in the camera frame so that we do not have to make additional

assumptions about multi-view cameras, or the availability of depth, allowing future scaling to more diverse video datasets. The tracker additionally predicts a binary visibility value $v_{t, k}$ denoting whether the point is occluded at step t .

IV. METHOD

Videos contain a great deal of prior information about the world, capturing physical dynamics, human behaviors, and semantics that are invaluable for policy learning. Beyond just learning representations from videos [43, 33, 28], we aim to learn a model from videos to predict future states for guiding a control policy. In this way, we can essentially decompose the visuomotor policy learning challenge into two parts. The first part is learning what to do next by generating future states as concrete *sub-goals*, which is learned purely from videos. The second part is learning to predict control actions to follow the sub-goals, which require much less data to train compared to learning policies end-to-end. With sufficient video pre-training, we will be able to learn generalizable policies even from limited action-labeled trajectories. Prior works [12, 23, 5] have predominantly relied on pixel-level future frame prediction as video pre-training. While video prediction is resource-intensive during both training and inference stages, its focus on reconstructing pixel-level details, which are often extraneous to policy learning, can adversely affect the efficiency of subsequent policy learning.

We propose **Any-point Trajectory Modeling (ATM)**. As illustrated in Figure 2, ATM is a two-stage framework: first learn to predict future point trajectories in a video frame as the pre-training with large-scale action-free videos, then use the predicted trajectories to guide policy learning with few-shot action-labeled demos. Our proposed method will be comprehensively detailed in this section: In Sec. IV-A, we first describe how to learn a point trajectory prediction model from an action-free video dataset \mathcal{T}_o . Then in Sec. IV-B, we outline how we utilize the pre-trained track prediction model to learn a track-guided policy from a limited action-labeled trajectory datasets \mathcal{T}_a .

A. Trajectory Modeling from Video Datasets

Our goal is to pre-train a model from videos that forecasts the future point trajectories in a frame. More formally, given an image observation o_t at timestep t , any set of 2D query points on the image frame $\mathbf{p}_t = \{p_{t, k}\}_{k=1}^K$, and a language instruction ℓ , we learn a model $\mathbf{p}_{t:t+H} = \tau_\theta(o_t, \mathbf{p}_t, \ell)$ that predicts the coordinates of the query points in the future H steps in the camera frame, where $\mathbf{p}_{t:t+H} \in \mathbb{R}^{H \times K \times 2}$. To model the tracks, we propose a **track transformer** and illustrate the architecture in Figure 2 (a).

a) Self-supervised Track Annotation.: Initially, we generate point trajectories from action-free videos for trajectory modeling pre-training. As described in Sec. III, we employ a vision tracker to pre-process videos and generate a tracking dataset. For each video, we randomly sample a time step \bar{t} and then randomly sample points on this frame and generate their tracks by running the tracker. However, for a static camera,

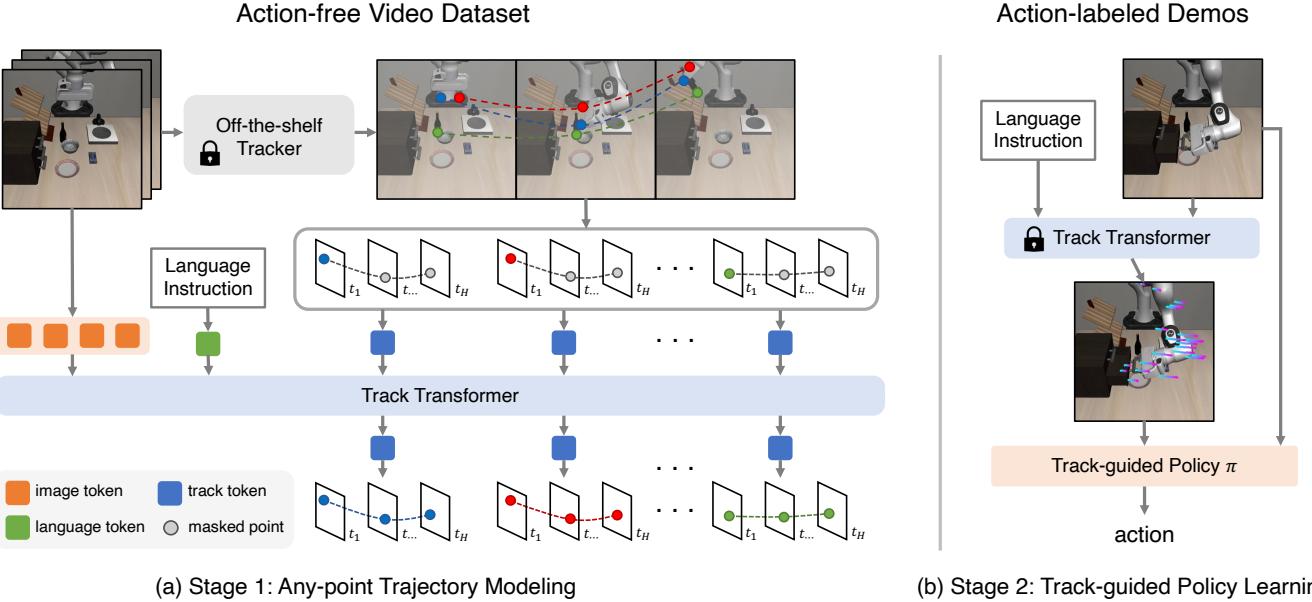


Fig. 2: Overview of our framework. (a) In the first stage, given an action-free video dataset, we first sample 2D points on one video frame and track their trajectories throughout the video using a pre-trained tracker. We then train a track transformer to predict future point trajectories given the current image observation, the language instruction, and the initial positions of the points. For the transformer input, we replace the future point positions with masked values. (2) In the second stage, we learn a track-guided policy to predict the control actions. Guidance from the predicted track enables us to learn robust policies from only a few action-labeled demonstrations.

most of the points that are sampled randomly will be in the background, thus providing little information when training our track transformer. To address this, we adopt a heuristic solution to filter out these static points: we first sample a grid of $n \times n$ on frame \bar{t} and track the grid of points across the whole of the video to obtain an initial set of tracks $\tau \in \mathbb{R}^{n^2 \times T \times 2}$. Subsequently, we filter points that have not moved during the video by thresholding the variance of the point positions over time. In the final step, we resample points around the filtered locations and finally generate their positions using the tracker.

b) Multimodal Track Modeling.: We formalize the future forecasting problem as a multi-modal masked prediction problem: we aim to predict the future positions of each point, conditioned on its current position, the current image observation, and a language instruction of the task. We first encode different modalities into a shared embedding space, each represented by a few tokens. For the tracks, we mask out the future positions of all points before encoding and then separately encode each point into one token. For the language instruction, we use a pre-trained BERT [9] encoder. For the images, we split them into image patches and randomly masked out 50% of the patches. We then pass all tokens through a large transformer model. Finally, we decode the track tokens into future trajectories of the corresponding points. Additionally, we reconstruct the image patches from the corresponding tokens following He et al. [18] as an auxiliary task, which we find useful for more complex tasks. Through this pre-

training process, our track transformer learns the motion prior of particles within the video frames.

B. Track-guided Policy Learning

After training a track transformer to predict future tracks based on observations, we can then learn policies guided by these predicted trajectories.

a) Arbitrary Points Tracking.: During track transformer pre-training, we can filter tracks without large movements. However, using this heuristic requires knowing the future positions of each point, which can be expensive to compute during policy inference. Instead, we find it sufficient to simply use a fixed set of 32 points on a grid for the policy. This sampling method avoids the potential complexities of learning key points or finding points to track [48] and works well in practice. ATM is permutation invariant to the input set of points, and we also find ATM to be robust to the distribution of the points, allowing us to use a different point sampling scheme from training for policy learning.

b) Track-guided Policy Learning.: A track-guided policy $\pi(a_t | o_t, \mathbf{p}_{t:t+H})$ takes input the current observation o_t and the predicted tracks $\mathbf{p}_{t:t+H}$ and predict the actions. A simplified illustration of our policy architecture is shown in Figure 3. Our transformer policy architecture follows the architecture in prior works [27, 21]. Although the predicted tracks alone already provide rich information to predict the actions, we still incorporate contextual image observations into our policy so that no information is lost, as suggested in prior works [26].

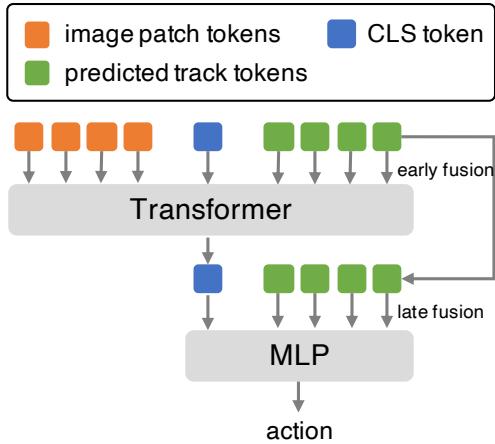


Fig. 3: A visual illustration of the architecture of the track-guided policy. Given the current observation and the predicted tracks from the frozen pre-trained track transformer, we train a track-guided policy from a limited demonstration dataset.

We incorporate the track tokens both before and after fusion with the image tokens (early fusion and late fusion) to ensure that the guiding information from tracks can be effectively integrated. Surprisingly, as the tracks already provide the fine-grained subgoals, we find that the policy no longer needs language instruction at this stage as task specification. Essentially, the provided tracks have reduced the difficult policy learning problems into a much easier sub-goal following problems, reducing the policy into an inverse dynamics model. Our track-guided policy is trained with MSE loss. Note that, the weights of our policy model are randomly initialized rather than copied from the pretrained Track Transformer like other video-pretraining methods [33, 28], because of the aim to separately study the task guidance capability of predicted trajectories. A detailed architecture diagram and hyperparameters are available in the appendix.

V. EXPERIMENTS

We perform experiments to answer the following questions:

- How does ATM compare with state-of-the-art video pre-training and behaviour cloning baselines for learning from action-free videos?
- Can ATM be used for learning from video data that are out of the distribution of demonstration data?

Our experiments are split into three sections. In Sec. V-A, we compare ATM with video pre-training baselines on over 130 language-conditioned manipulation tasks in simulation and in the real world. In Sec. V-B, we show that ATM enables effective learning from human videos. Finally, we present ablation results in Sec. V-C.

A. Video Pre-training for Imitation Learning

Environments. We compare with baselines on over one hundred language-conditioned manipulation tasks in the LIBERO benchmark [27]. The benchmark is subdivided into

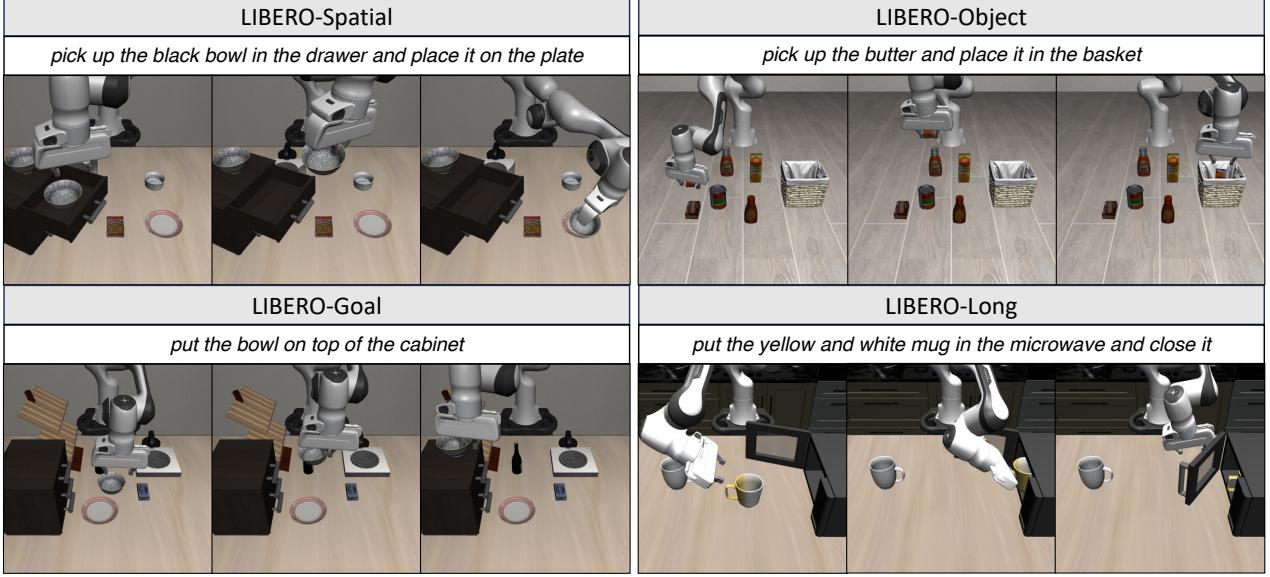
five suites, LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, LIBERO-Long, and LIBERO-90. Each suite has 10 tasks, except LIBERO-90 which contains 90 tasks. Each task comes with expert human demonstrations. LIBERO-Spatial contains tasks with the same objects but different layouts; LIBERO-Object has tasks with the same layouts but different objects; LIBERO-Goal has tasks with the same object categories and spatial layouts, but different goals; LIBERO-Long has tasks with diverse object categories and layouts, and long-horizon task goals; LIBERO-90 has extremely diverse object categories, layouts and task goals.

Data. We compare with baselines on each suite separately. All methods are trained on 10 action-labeled demonstration trajectories and 50 action-free video demonstration trajectories of the robot for each task, amounting to 500 videos for each 10-task suite. The demonstration dataset contains RGB images from a third-person camera and a wrist camera, together with gripper and joint states as observations. As each task is specified by a language instruction, we use the pre-trained BERT network to obtain a task embedding [9]. Image resolution is 128×128 and the action space is 7-dimension, representing the translation, rotation, and aperture of the end-effector.

Baselines. We compare with the following baselines:

- 1) **BC** denotes the vanilla behavioral cloning which trains a policy exclusively using the limited action-labeled expert demonstrations, without using the video dataset. It uses a policy architecture identical to ATM except that the particle trajectories are masked to be zero and it instead takes language embedding as task specification. BC baseline can also be considered as an ablation for policy learning without action-free videos. The specific architecture and losses for training the BC policy can vary. We use BC to denote a transformer policy trained using MSE loss by default and compare with diffusion policy separately towards the end of this section.
- 2) **R3M-finetune** [33] uses a contrastive learning objective for learning representation that aligns video and language with a combination of time contrastive losses, L_1 regularization, and language consistency losses. We adopt the publicly released Ego4D pre-trained weights and fine-tune the weights on our in-domain video dataset \mathcal{T}_o , to initialize the behavioral cloning policy's visual encoder. During policy training, we also further fine-tune the R3M backbone with the behavioral cloning loss. While this method captures priors from action-free videos through representation learning, the visual representation lacks knowledge about the transition dynamics critical to decision-making.
- 3) **VPT** [3] first trains an inverse dynamics model from the action-labeled trajectories \mathcal{T}_a and then uses it to predict pseudo action labels for the video dataset. With these pseudo-labels, a policy is then trained with behavioral cloning. This method requires the inverse dynamics model to be robust to a wide distribution of input observation, which can be difficult to learn from the limited demonstrations.

(a) Visualization of the tasks in the LIBERO benchmark



(b) Comparison on the performance

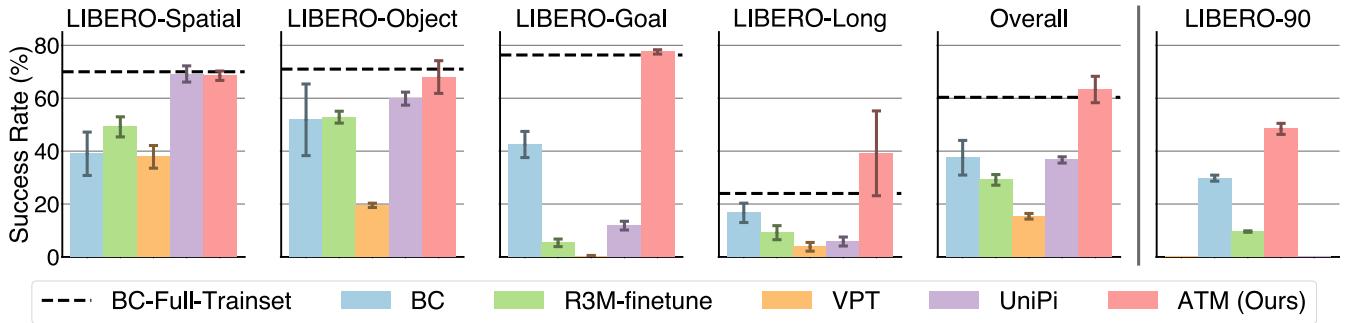


Fig. 4: We compare with state-of-the-art video pre-training methods on language-conditioned manipulation tasks in the LIBERO benchmark [27]. (a) Visualization of the LIBERO tasks separated into four suites, focusing on different aspects of the manipulation policies in spatial reasoning, object reasoning, task understanding, and performing long-horizon tasks. (b) Quantitative comparisons on different suites. We additionally compare baselines with fast computation on a task suite containing 90 tasks (i.e. LIBERO-90). ATM outperforms the baselines in all tasks and excels in LIBERO-Goal and LIBERO-Long.

- 4) **UniPi** [12, 23, 5] trains a text-conditioned video diffusion model to generate a temporally fine-grained video plan from an initial frame and a language instruction. During policy learning, UniPi trains an inverse dynamics model with action-labeled data. We base our implementation off of the UniPi implementation in Ko et al. [23]. While both UniPi and ATM leverage a policy conditioned on future subgoals, a trajectory representation decouples motion from other pixel-based information and makes policy learning much easier. Please see the Appendix where we perform additional comprehensive comparisons of different variations of UniPi.

Results. We present the main results in Figure 4. We see that by bridging the video data and policy learning with the structured representation of point trajectories, ATM (our method)

significantly surpasses various strong baselines in video pre-training, achieving an average success rate of 63% compared to the highest success rate of 37% by previous methods, marking an improvement of over 80%. The comparison of BC with ATM shows that learning from additional videos provides useful information for policy learning. VPT performs poorly as we empirically observed that the pseudo-action labels predicted by VPT generally show large errors on the video dataset. UniPi fails on more complex as video prediction models are not physically grounded and often generate future frames that are not physically feasible, such as cases where robots disappear from the image. ATM shows surprisingly superior performance on long-horizon tasks and tasks that require an understanding of goals. We attribute this performance gain to ATM’s formulation, which utilizes explicit future particle

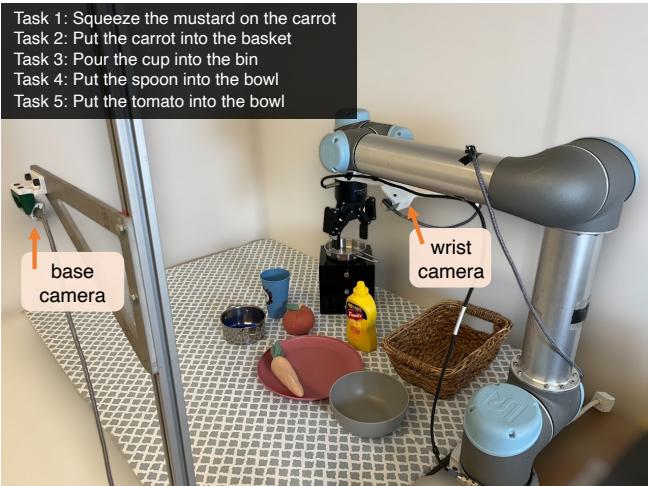


Fig. 5: Real robot experiments on a dining table setup consisting of five tasks. The left figure shows our real-world setup and the tasks. The top right figure shows an example of the predicted particle trajectories and the policy execution, which closely follows the predicted trajectories. From the quantitative results, we can see that ATM shows significant improvements over state-of-the-art video pre-training baselines on average.

trajectories as subgoals. At each time step, ATM predicts a future trajectory goal based on the current observation. In comparison to the static natural language instructions used by other methods, which do not change throughout the episode, ATM’s closed-loop future trajectory provides clear guidance as to what the policy needs to do next. Please see our video for failure cases of a video prediction model.

Universality of ATM on Different Policy Classes. The benefit of the predicted trajectories can also be shown for state-of-the-art policy class such as diffusion policy [8], as shown in Figure 6. Here, ATM Diffusion Policy adds the predicted future tracks as additional condition to the policy class, while keeping everything else the same. We see that ATM consistently improves the performance of diffusion policy across all the suites, achieving the highest scores on the LIBERO benchmark. More details can be found in Appendix.

Real World Setup. We conduct a language-conditioned manipulation experiment in the real world to further strengthen our claim. As illustrated in Figure 5, we learn policies for a 6-DOF UR5 robot arm using human expert demonstrations collected with the GELLO teleoperation system [53]. The action space is joint position control and gripper state. The observations include two RGB images from two RealSense cameras. To compensate for the partial observation, we stack the most recent two frames as the input of agents. We do not feed proprioception states into agents because it is reported that imitation policies tend to overfit them and ignore the visual inputs, leading to worse online rollout performance [30, 26]. We collect a total of 50 action-labeled trajectories and 250 action-free demonstration videos (by simply removing the actions from the collected action-labeled demonstrations). From the quantitative results in Figure 5, we see consistent trends that ATM outperforms BC and other video pre-training baselines by a significant margin.



TABLE III: Average success rates of real robot experiments over three random seeds.

Task instruction	BC	R3M-Finetune	VPT	ATM (Ours)
Squeeze the mustard on the carrot	0.83 ± 0.12	0.00 ± 0.00	0.00 ± 0.00	0.97 ± 0.06
Put the carrot into the basket	0.43 ± 0.38	0.00 ± 0.00	0.00 ± 0.00	0.90 ± 0.10
Pour the cup into the bin	0.93 ± 0.06	0.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00
Put the spoon into the bowl	0.23 ± 0.21	0.07 ± 0.12	0.00 ± 0.00	0.80 ± 0.10
Put the tomato into the bowl	0.47 ± 0.12	0.07 ± 0.06	0.03 ± 0.06	1.00 ± 0.00
Average	0.58 ± 0.18	0.03 ± 0.04	0.01 ± 0.01	0.93 ± 0.05

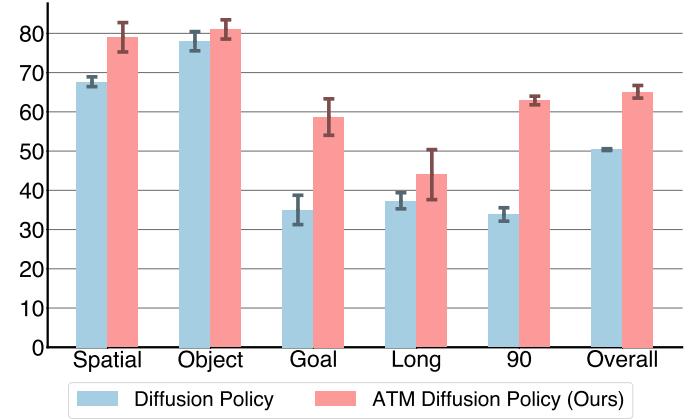


Fig. 6: We implement ATM Diffusion Policy by adding the predicted future trajectories as additional conditioning and show consistent improvement over the base diffusion policies across the benchmark suites.

TABLE I: Average success rates of human-to-robot experiments. ATM trained with human videos significantly outperforms BC and ATM trained with only 10 robot videos, demonstrating the cross-embodiment capability of ATM.

Method	Teleoperation demos	Human videos	fold cloth	put tomato	sweep toys
BC	✓	✗	0%	10%	30%
ATM	✓	✗	0%	0%	13%
ATM	✓	✓	63%	63%	60%

B. Human-to-robot and Robot-to-robot Transfer

By modeling the low-level any-point trajectories, ATM enables learning from cross-embodiment videos of humans or a different robot performing the task. This facilitates the

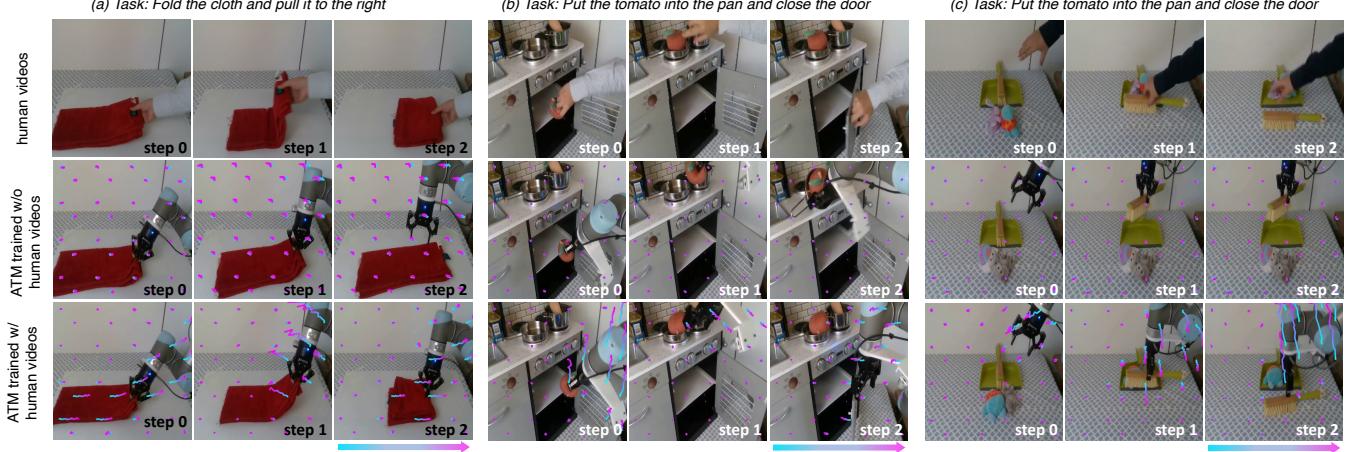


Fig. 7: Learning robotic skills from human videos for three tasks. We collect 100 videos of a human performing the tasks directly and 10 teleoperation demonstration trajectories. Each row from the top to the bottom shows three snapshots from the human videos, ATM trained without the human videos, and ATM trained with the human videos. By comparison, we can see that human videos are critical in learning accurate trajectory prediction and enable the policy to successfully perform the task.

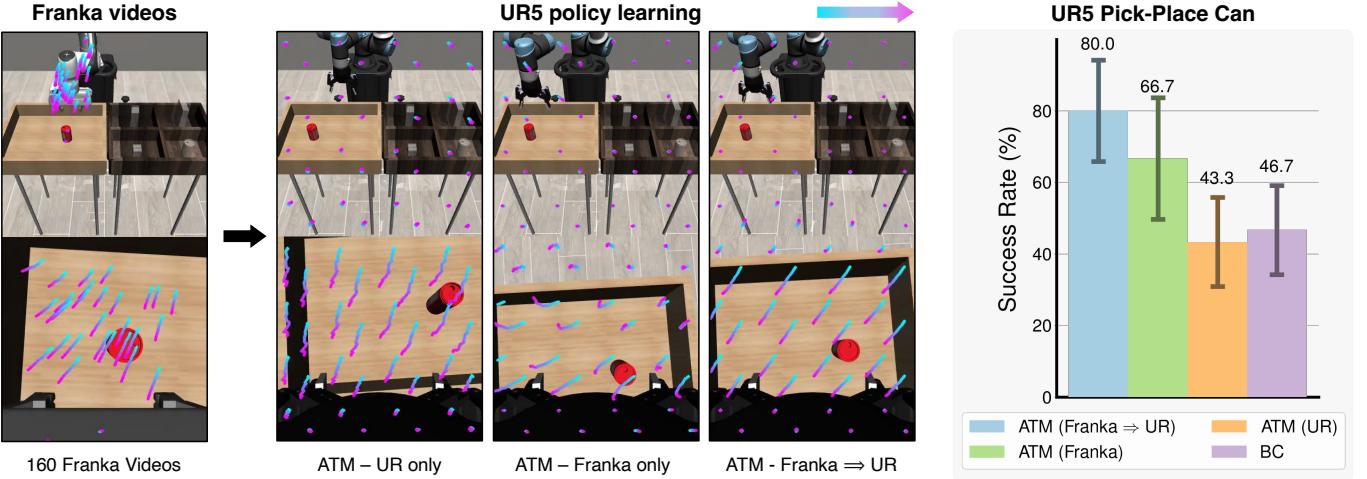


Fig. 8: Cross-morphology skill transfer for a pick-and-place task. Here, we collect 160 action-free videos of a Franka arm and 10 action-labeled demonstrations from a UR arm, with the final goal of learning a UR policy. We compare a vanilla BC baseline with ATM trained using types of data: using only the 10 UR videos, using only the 160 Franka videos, and using both Franka and UR videos (Franka \Rightarrow UR). In the right plot, we observe that the additional cross-embodiment data led to significantly better results. Surprisingly, even if the trajectory model is only trained using Franka videos, it exhibits much better performance than the BC without the Franka videos.

use of more scalable data sources. To verify this, we present the results of learning from human videos in Fig. 7, with the quantitative results presented in Table I, and the results of cross-robot transfer in Fig. 8. We compare the following methods: (1) Learning behavior cloning policies only on the action-labeled data (2) ATM, using a track transformer trained only on the limited action-labeled robot data, and (3) ATM, using a track transformer trained on both the action-free human and action-labeled robot data. Experiments show that training the trajectory model on additional cross-embodiment videos makes the trajectory prediction more robust and accurate,

significantly improving policy learning. On the other hand, as the number of action-labeled trajectories is small, BC baselines that only use action-labeled trajectories fail. Please refer to the videos on our website for better visualization.

C. Ablation Analysis

We conduct a series of ablation experiments in simulation to demonstrate the effect of our design choices. These include the number of action-labeled trajectories needed for policy learning, the effect of the trajectory prediction horizon, image masking, and late track fusion. Besides, we also investigate the universality of ATM for different policy models.

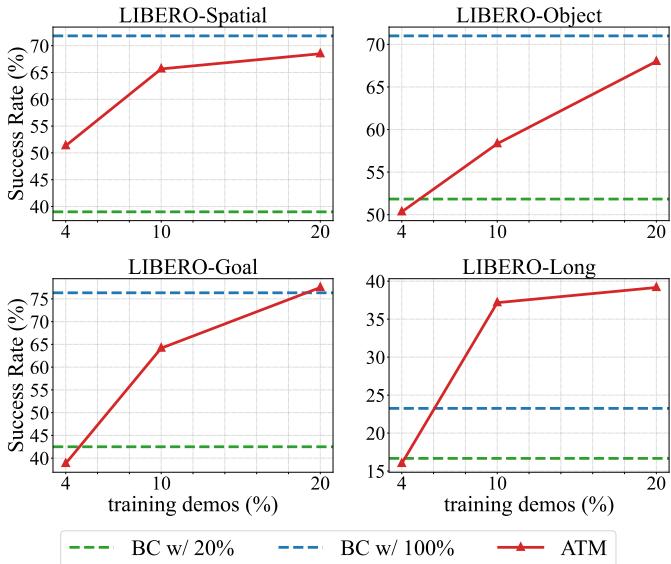


Fig. 9: Success rate of our policy trained with 4%, 10% and 20% action-labeled demos. Our policy trained with only 4% demos performs comparably to BC baseline with 20% demos on LIBERO-Spatial, Object, and GOAL, and even better on LIBERO-Spatial. When trained on 20% demos, our performance approaches BC with all training data.

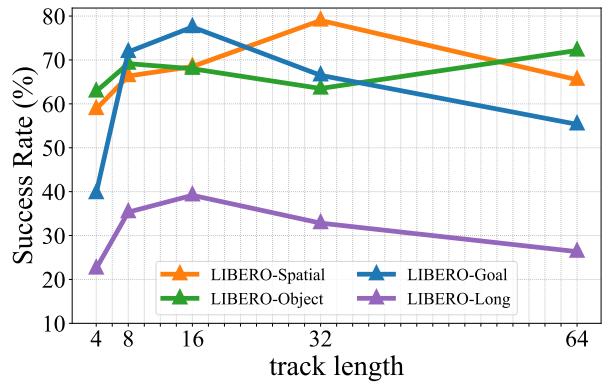


Fig. 10: We plot the success rates of the policies learned with predicted trajectories of different lengths. Generally, longer trajectory length improves the performance, but the benefit tends to plateau after 16.

Effect of the number of action-labeled trajectories. We investigate the impact of track length on our policy input by training our policy using predicted tracks of varying lengths: 4, 8, 16, 32, and 64 steps. As illustrated in Figure 10, a track length of 16 yields optimal performance on average. Smaller track lengths significantly decrease performance, while a track length of 0 reduces ATM to the behavior cloning baseline without any benefit of pre-training. On the other hand, larger track lengths can also negatively impact performance, especially on more challenging tasks such as LIBERO-Long and LIBERO-Goal, as longer track predictions are noisier and less relevant for achieving short-horizon subgoals. Determining the optimal time horizon for subgoals could be an interesting avenue for future research.

Effect of track length. The track length of our pre-trained track transformer is 16 and we utilize all 16 points of each track by default as input for our policy. In this experiment, we investigate the effect of track length of our policy input, by training our policy with the predicted tracks truncated into 4, 8, and 16 steps. The results in Figure 10 demonstrate that longer tracks result in higher success rates in general, except LIBERO-Object where track length = 8 yields the best performance. In particular, reducing track length leads to larger performance drops on LIBERO-Goal and LIBERO-Long. These tasks require a comprehensive understanding of the task goal. Longer tracks provide more detailed and specific subgoals for each task, which is crucial for guiding the agent’s movements and actions in subsequent stage. Conversely, for LIBERO-Object, which emphasizes precision operations over

understanding of the task goal, a policy with a length of 16 underperforms slightly compared to the model with a length of 8. We hypothesize that the longer tracks might interfere with the learning of inverse dynamics due to noise. This also supports our approach of employing tracks both as subgoal conditions and in leveraging the nature of inverse dynamics.

Effect of image masking in track transformer. When training the track transformer, we randomly mask out patches in the images and learn to reconstruct them as an auxiliary task. We conduct an ablation study by removing the image masking loss and comparing it against our standard configuration, where we randomly mask out 50% of the image masks. In Table II, the results reveal a slight decline in policy performance when image masking is omitted, suggesting image masking can be a useful auxiliary task, except on LIBERO-Spatial. We hypothesize that the auxiliary loss encourages the model to jointly reason about different regions of an image observation. This bias may not be critical for the Libero-spatial tasks because the specific spatial location of interest is already fully specified in the language instruction (e.g., all tasks have the format of "pick up the black bowl at [some spatial location]").

Effect of early and late fusion in policy architecture. As shown in Figure 3, the predicted tracks are fed into the policy both before and after the transformer architecture within the policy, which we call early fusion and late fusion respectively. We conduct ablation studies by removing these two track inputs. The results are shown in Table III. We can see that removing the late fusion leads to the most significant performance drop; on LIBERO-Goal, *w/ only early fusion* performs similarly to other baselines, whereas only late fusion performs marginally worse than our full method. This suggests a late fusion of the predicted tracks acts as useful subgoals that help the policy better understand the tasks in a multi-task learning setting. The subgoal prediction is more robust as it is trained on a larger video dataset.

TABLE II: Ablation study on image masking of track transformer, where “w/o image masking” represents that we do not mask out image patches during track transformer training and “w/ image masking” means we randomly mask 50% patches. We can see that mask image modeling in track transformer improves the policy performance.

Image Mask Ratio	Spatial	Object	Goal	Long
w/o image masking	69.17 ± 6.38	65.00 ± 3.89	74.33 ± 3.66	30.83 ± 11.43
w/ image masking (default)	68.50 ± 1.78	68.00 ± 6.18	77.83 ± 0.82	39.33 ± 15.80

TABLE III: Ablation study on the policy architecture. We explore the effect of the tracks fed into the policy in two positions: transformer input (early fusion) and MLP head (late fusion), as illustrated in Figure 3.

early fusion	late fusion	Spatial	Object	Goal	Long
✓	✗	44.67 ± 1.84	56.67 ± 3.09	5.33 ± 0.24	22.33 ± 4.94
✗	✓	65.50 ± 3.89	60.00 ± 1.47	72.83 ± 4.73	42.76 ± 14.62
✓	✓	68.50 ± 1.78	68.00 ± 6.18	77.83 ± 0.82	39.33 ± 15.80

VI. LIMITATIONS

One limitation is that our approach still relies on a set of action-labeled demonstration trajectories for mapping to actions, which limits the generalization of the learned policies. Future works can consider learning the trajectory-following policies using reinforcement learning so that no additional demonstration data are needed. Another limitation of our method is that the video dataset we use in this paper only contains small domain gaps. Learning from in-the-wild video dataset poses additional challenges such as multi-modal distribution, diverse camera motions, and sub-optimal motions. We leave these extensions for future work.

VII. CONCLUSIONS

In this work, we present an any-point trajectory modeling framework as video pre-training that effectively learns behaviors and dynamics from action-free video datasets. After pre-training, by learning a track-guided policy, we demonstrate significant improvements over prior state-of-the-art approaches and show effective learning from out-of-distribution human videos. We show that a particle-based representation is interpretable, structured, and naturally incorporates physical inductive biases such as object permanence. We hope our works will open doors to more exciting directions in learning from videos with structured representations.

VIII. ACKNOWLEDGEMENT

This work was supported in part by the BAIR Industrial Consortium, and the InnoHK of the Government of the Hong Kong Special Administrative Region via the Hong Kong Centre for Logistics Robotics. This work is also supported by the Ministry of Science and Technology of the People’s Republic of China, the 2030 Innovation Megaprojects “Program on New Generation Artificial Intelligence” (Grant No. 2021AAA0150000), and the National Key R&D Program of China (2022ZD0161700). This work is done when Chuan Wen is visiting UC Berkeley.

REFERENCES

- [1] A Pedro Aguiar and Joao P Hespanha. Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *IEEE transactions on automatic control*, 52(8):1362–1379, 2007.
- [2] S Bahl, R Mendonca, L Chen, U Jain, and D Pathak. Affordances from human videos as a versatile representation for robotics. In *CVPR*, 2023.
- [3] Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35: 24639–24654, 2022.
- [4] Homanga Bharadhwaj, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.
- [5] Kevin Black, Mitsuhiro Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pre-trained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.
- [6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.
- [7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [8] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and*

- Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.026.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [10] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video, 2023.
- [11] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement, 2023.
- [12] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- [13] Alejandro Escontrela, Ademi Adeniji, Wilson Yan, Ajay Jain, Xue Bin Peng, Ken Goldberg, Youngwoon Lee, Danijar Hafner, and Pieter Abbeel. Video prediction models as rewards for reinforcement learning. *Neural Information Processing Systems*, 2023.
- [14] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. In *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@CoRL2023*, 2023.
- [15] Ankit Goyal, Arsalan Mousavian, Chris Paxton, Yu-Wei Chao, Brian Okorn, Jia Deng, and Dieter Fox. Ifor: Iterative flow minimization for robotic object rearrangement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14787–14797, 2022.
- [16] Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv preprint arXiv:2311.01977*, 2023.
- [17] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022.
- [18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.
- [19] Zixuan Huang, Xingyu Lin, and David Held. Mesh-based dynamics model with occlusion reasoning for cloth manipulation. In *Robotics: Science and Systems (RSS)*, 2022.
- [20] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Co-tracker: It is better to track together. *arXiv:2307.07635*, 2023.
- [21] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision, 2021.
- [22] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, October 2023.
- [23] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to Act from Actionless Video through Dense Correspondences. *arXiv:2310.08576*, 2023.
- [24] Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. *arXiv preprint arXiv:2107.04004*, 2021.
- [25] Xingyu Lin, Yufei Wang, Zixuan Huang, and David Held. Learning visible connectivity dynamics for cloth smoothing. In *Conference on Robot Learning*, 2021.
- [26] Xingyu Lin, John So, Sashwat Mahalingam, Fangchen Liu, and Pieter Abbeel. Spawnnet: Learning generalizable visuomotor skills from pre-trained networks. *arXiv preprint arXiv:2307.03567*, 2023.
- [27] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Yuke Zhu, Peter Stone, et al. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.
- [28] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards universal visual reward and representation via value-implicit pre-training. In *The Eleventh International Conference on Learning Representations*, 2023.
- [29] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.
- [30] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *arXiv preprint arXiv:2108.03298*, 2021.

- [31] Lucas Manuelli, Yunzhu Li, Pete Florence, and Russ Tedrake. Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning, 2020.
- [32] Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Structured world models from human videos. In *RSS*, 2023.
- [33] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909. PMLR, 2023.
- [34] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.
- [35] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018.
- [36] Zengyi Qin, Kuan Fang, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Keto: Learning keypoint representations for tool manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7278–7285. IEEE, 2020.
- [37] Adam Rashid, Satvik Sharma, Chung Min Kim, Justin Kerr, Lawrence Yunliang Chen, Angjoo Kanazawa, and Ken Goldberg. Language embedded radiance fields for zero-shot task-oriented grasping. In *Conference on Robot Learning*, pages 178–200. PMLR, 2023.
- [38] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.
- [39] Karl Schmeckpeper, Annie Xie, Oleh Rybkin, Stephen Tian, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Learning predictive models from observation and interaction, 2019.
- [40] Dominik Schmidt and Minqi Jiang. Learning to act without actions. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=rvUq3cxFDF>.
- [41] Daniel Seita, Yufei Wang, Sarthak J Shetty, Edward Yao Li, Zackory Erickson, and David Held. Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds. In *Conference on Robot Learning*, pages 1038–1049. PMLR, 2023.
- [42] Younggyo Seo, Kimin Lee, Stephen James, and Pieter Abbeel. Reinforcement learning with action-free pre-training from videos, 2022.
- [43] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.
- [44] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.
- [45] Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.
- [46] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.
- [47] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation, 2018.
- [48] Mel Vecerik, Carl Doersch, Yi Yang, Todor Davchev, Yusuf Aytar, Guangyao Zhou, Raia Hadsell, Lourdes Agapito, and Jon Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. *arXiv*, 2023.
- [49] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.
- [50] Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Anima Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. In *7th Annual Conference on Robot Learning*, 2023.
- [51] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once, 2023.
- [52] Chuan Wen, Jianing Qian, Jierui Lin, Jiaye Teng, Dinesh Jayaraman, and Yang Gao. Fighting fire with fire: Avoiding dnn shortcuts through priming. In *International Conference on Machine Learning*, pages 23723–23750. PMLR, 2022.
- [53] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. *arXiv preprint arXiv:2309.13037*, 2023.
- [54] Haoyu Xiong, Quanzhou Li, Yun-Chun Chen, Homanga Bharadhwaj, Samarth Sinha, and Animesh Garg. Learning by watching: Physical imitation of manipulation skills from human videos. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7827–7834. IEEE, 2021.
- [55] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint*

arXiv:2310.06114, 2023.

- [56] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635. IEEE, 2018.
- [57] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19855–19865, 2023.

APPENDIX A

ADDITIONAL EXPERIMENTAL RESULTS

A. Simulation Experiments

Numerical results. We report the numeric values for the success rates on LIBERO benchmark in Table V. All methods use 20% of the demonstration trajectories except the oracle. See Sec. A-B for details about the comparisons on UniPi and UniPi-Replan.

Attention map visualization of track-guided policy. To demonstrate how tracks guide the policy, we visualize the attention maps between the spatial CLS token and RGB tokens in the spatial transformer of BC and our method. Figure 11 demonstrates that our method effectively focuses on the relevant spatial regions, as specified by the textual instructions. Specifically, it attends to the cream cheese, bowl, and wine bottle in the respective example tasks, while BC is usually distracted by the irrelevant regions, highlighting the superior capability of the tracks as better task prompts.

B. Discussions on the UniPi Baselines

UniPi [12] proposes to train a language-conditioned video diffusion model f_θ during video pre-training. During policy learning, given the initial image observation o_0 and the language l , UniPi first generates all future frames $\tilde{o}_1, \dots, \tilde{o}_{T-1} = f_\theta(o_0, l)$ and then learns an inverse dynamics model that predicts the action at each time step $a_t = \pi(o_t, \tilde{o}_{t+1})$. UniPi then executes the actions open-loop. However, training a diffusion model to predict the full video can be computation intensive. As such, the UniPi implementation in our paper follows the one in Ko et al. [23], where we predict $N = 7$ future frames as the sub-goals for the policy, denoted as $\tilde{o}_1, \dots, \tilde{o}_N$. During training, we evenly sample N frames in an episode for training the video prediction model. During policy learning, we train a goal-conditioned policy $\pi(a_t, \text{done}|o_t, \tilde{o}_i)$, where $i \in 1, \dots, N$ is the image sub-goal and t denotes the current timestep. The policy additionally predicts a done flag to determine when it should switch from the current sub-goal \tilde{o}_i to the next sub-goal \tilde{o}_{i+1} . ATM’s superiority over UniPi can be attributed to two reasons. First, ATM uses a more structured sub-goal representation of point trajectories. Second, ATM performs closed-loop inference, proposing a new sub-goal at each time step, while UniPi’s video diffusion process is too slow to be referenced at every time step.

To train the UniPi policy, we sample o_i from a future frame o_t where $t \in [t, t + t_{max}]$. We choose t_{max} for each task suite ($t_{max} = 16$ for LIBERO-Object and LIBERO-Spatial, $t_{max} = 50$ for LIBERO-Goal and LIBERO-Long). To mitigate dataset imbalance when learning the done flag, we sample o_i as the next frame 10% of the time. We perform MSE regression on both the action a_t and done.

In order to decouple the two advantages of ATM over UniPi additionally compare with a UniPi variation where we train the video prediction model to predict a fixed time step into the future $\tilde{o}_{t+H} = f_\theta(o_t)$ for every H steps of policy execution, where $H = 8$. As this variation replans the

TABLE IV: Computation cost and inference time for different methods on a V100 GPU. ATM performs trajectory generation instead of predicting high-dimensional frames, making it the most computationally efficient and feasible for closed-loop control. UniPi employs a video diffusion model for open-loop future goal generation at the beginning, demanding significantly higher computational resources. UniPi-Replan generates fewer frames than UniPi using a smaller model, resulting in marginally faster generation. However, its use in closed-loop control remains computationally prohibitive.

Computation	Close-Loop		Open-Loop
	ATM	UniPi-Replan	UniPi
TFLOPS per generation	1.56	13.09	39.29
Time per generation (s)	0.015	4.51	8.14

sub-goal more frequently, we call this method *UniPi-Replan*, similar to the implementation in [5]. The results are shown in Table V. Surprisingly, we found that this variation performs even worse than UniPi. **We thus draw the conclusion that a structured sub-goal representation can be much more effective than an image sub-goal.** The reason is that predicting an image goal at a fixed future time step can be a difficult objective for the video prediction model, leading to inconsistent subgoals. Please refer to the failure videos of various baselines on our website. Additionally, this method requires heavy computation. A comparison of the computation needed is shown in Table IV. Due to the computation cost, we only evaluate UniPi-Replan on LIBERO-Object and LIBERO-Goal and report the average success across 10 trials on one policy training random seed.

C. Applying ATM to Diffusion Policy

In addition to the transformer policy adopted from the original LIBERO paper [27], our ATM can also be integrated with the state-of-the-art Diffusion Policy [8]. Specifically, we implement an ATM Diffusion Policy by using predicted tracks as input conditions. Both the standard and ATM-based Diffusion policies are trained with 20% of the data from LIBERO, in the same setting as the experiment in Figure 4.

As shown in Table VI, the standard Diffusion Policy achieves impressive results on the LIBERO benchmark, even with limited training data, confirming its status as the state-of-the-art imitation policy. Moreover, the ATM Diffusion Policy consistently outperforms the standard model, highlighting our framework’s versatility and effectiveness across different policy architectures.

D. Human-to-robot Transfer Details

To demonstrate the potential of ATM to leverage out-of-domain videos, we design human-to-robot transfer tasks in three different settings: 1) deformable object: *fold the cloth and pull it to the right*, 2) long horizon: *put the tomato into the pan and close the cabinet door*, and 3) tool using: *use the broom to sweep the toys into the dustpan and put it in front of the dustpan*. We collect 10 robot teleportation trajectories

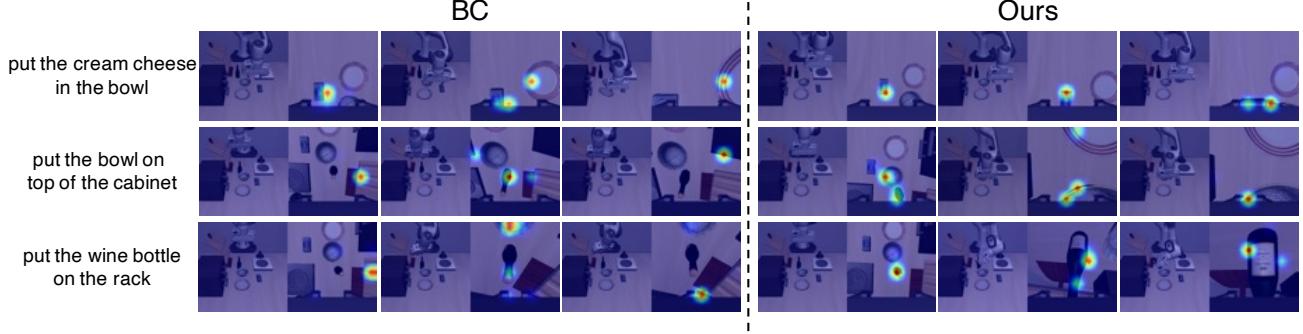


Fig. 11: The attention maps of BC and Ours in the spatial transformer. We extract the attention weights between spatial CLS tokens and RGB tokens, highlighting the policy’s focus on specific spatial regions during decision-making. The heatmaps reveal our policy’s targeted attention on task-relevant areas, in contrast to BC’s tendency to focus on irrelevant backgrounds. This underscores the effectiveness of input tracks in the spatial transformer as good task prompts, guiding the CLS token to attend to appropriate areas.

TABLE V: Average success rate on LIBERO benchmark. Our method performs consistently better than all the baselines across all suites. UniPi-Replan is only evaluated for a single seed due to the computation cost.

Method	Libero-Spatial	Libero-Object	Libero-Goal	Libero-Long	Libero-90
BC-Full-Trainset (Oracle)	71.83 ± 3.70	71.00 ± 7.97	76.33 ± 1.31	24.17 ± 2.59	-
BC	39.00 ± 8.20	51.83 ± 13.54	42.50 ± 4.95	16.67 ± 3.66	29.78 ± 1.14
R3M-finetune [33]	49.17 ± 3.79	52.83 ± 2.25	5.33 ± 1.43	9.17 ± 2.66	9.59 ± 0.27
VPT [3]	37.83 ± 4.29	19.50 ± 0.82	3.33 ± 2.36	3.83 ± 1.65	-
UniPi [12, 23]	69.17 ± 3.75	59.83 ± 3.01	11.83 ± 2.02	5.83 ± 2.08	-
UniPi-Replan [5]	-	31.00	3.00	-	-
ATM (Ours)	68.50 ± 1.78	68.00 ± 6.18	77.83 ± 0.82	39.33 ± 15.80	48.41 ± 2.09

TABLE VI: Detailed results of Diffusion policy on LIBERO. The Diffusion policy can be further improved by our method, suggesting that our Any-point Trajectory Modeling framework is an important building block to apply to any policy model.

Method	Libero-Spatial	Libero-Object	Libero-Goal	Libero-Long	Libero-90
Diffusion Policy	67.67 ± 1.25	78.00 ± 2.45	35.00 ± 3.74	37.33 ± 2.05	33.85 ± 1.71
ATM Diffusion Policy	79.00 ± 3.74	81.00 ± 2.45	58.67 ± 4.64	44.00 ± 6.38	62.89 ± 1.10

(action-labeled) and 100 human manipulation videos (action-free). We compare three methods: 1) behavioral cloning only with 10 action-labeled robot demos, 2) ATM, training track transformer with only 10 robot demos, and 3) ATM, training a track transformer with both action-free human and action-labeled robot data. For each task, we train each method with three different random seeds, evaluate them in the real world 10 times, and report the average success rate in Table I.

Human-to-Robot visualization. The detailed video visualizations of human-to-robot skill transfer are shown in Figure 15. Due to limited training samples, the track transformer trained with only 10 robot videos fails to predict the future point trajectories, leading to low success rates. In contrast, the trajectory model trained with human videos generates high-quality tracks in each frame, guiding the agent to successfully complete the task with only 10 action-labeled trajectories. This indicates our any-point trajectory modeling enables to transfer the motion prior from cross-embodiment videos to robot skills, which significantly improves policy learning.

Furthermore, we visualize the attention maps of Track

Transformers trained with or without human videos in Figure 14. Track Transformer trained with human videos (bottom row) displays well-defined attention heatmaps that primarily focus on the object and robot arm. Conversely, the attention maps from the model trained without human videos (middle row) are notably more dispersed, often incorrectly focusing on background areas. This comparison highlights the effectiveness of our ATM in leveraging prior knowledge from human videos, facilitating successful cross-embodiment imitation learning.

APPENDIX B IMPLEMENTATION DETAILS

A. Policy Architecture

We include a more detailed visual of the ViT-T [27, 21] policy used in our experiments in Figure 12. The input to our policy is a set of temporally stacked images across multiple views $o_t \in \mathbb{R}^{V \times T \times C \times H \times W}$, and proprioception $p_t \in \mathbb{R}^{D_p}$. To process these inputs, ViT-T consists of three stages:

TABLE VII: Hyperparameters of track transformer training.

Hyperparameters	Track Transformer
epoch	100
batch size	1024
optimizer	AdamW
learning rate	1e-4
weight decay	1e-4
lr scheduler	Cosine
lr warm up	5
clip grad	10
point sampling	variance filtering
number of points	32
track length	16
track patch size	4
image mask ratio	0.5
augmentation	ColorJitter,RandomShift

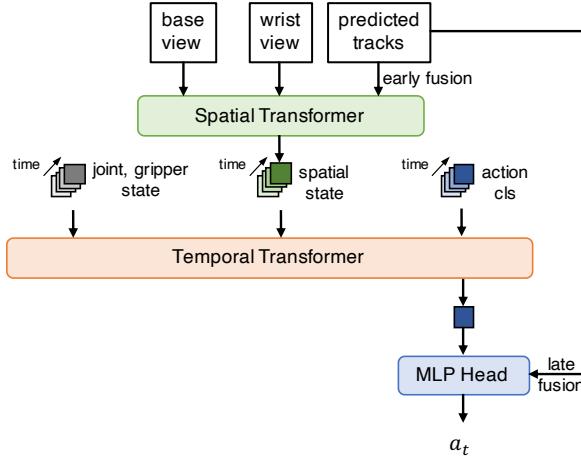


Fig. 12: To summarize spatial information, we perform self-attention on a sequence consisting of all views’ track and image patches and a CLS token. To integrate information across time, we perform causal self-attention between spatial CLS token, proprioception, and an action CLS token per timestep. To regress actions, we concatenate each timestep’s action CLS token and proposed tracks.

Spatial Encoding: We encode all modalities at each timestep. We first leverage the frozen track transformer to propose a set of tracks for all V frames in o_t . We then project each modality with modality-specific encoders into a shared embedding space \mathbb{R}^D . Each modality’s tokens are embedded using a shared learned modality token, and modality-specific positional embeddings. We then concatenate tokens across all modalities and views with a learned spatial CLS token, and perform self-attention on the sequence. We extract the spatial CLS token as the representation.

Temporal Decoding: We process the encoded modalities across timesteps into actions. We first project the proprioception at each timestep into the same shared embedding space \mathbb{R}^D . We then interleave the encoded proprioception, the spatial CLS token, and a learned action

TABLE VIII: Hyperparameters of policy training.

Hyperparameters	Policy
epoch	100
batch size	512
optimizer	AdamW
learning rate	5e-4
weight decay	1e-4
lr scheduler	Cosine
lr warm up	0
clip grad	100
point sampling	grid
number of points	32
track length	16
frame stack	10
augmentation	ColorJitter,RandomShift

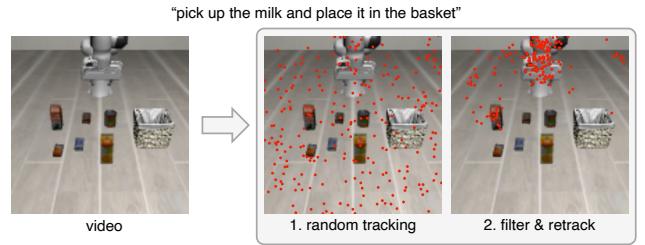


Fig. 13: Given a video (left), we query 1000 randomly sampled points using an off-the-shelf TAP model (middle), where each colored dot represents the starting position of a track. We then filter the tracks using a heuristic of their position displacement across the video and re-sample around these points (right). We can see that extracted tracks are concentrated around informative objects, such as the robot’s gripper and manipulation targets.

CLS token across timesteps into a sequence, before performing causally-masked self attention between the sequence.

Action Head: We treat each timestep’s output independently and parameterize actions using an MLP. For each timestep, we take the action CLS token, and fuse the CLS token with the reconstructed tracks of the current timestep.

B. Efficient Training with Point Filtering

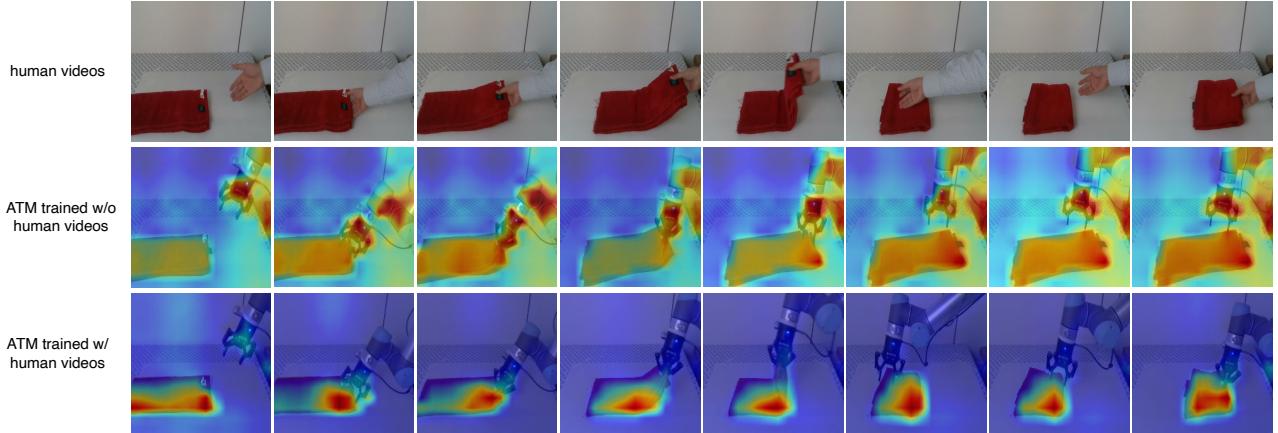
We adopt a heuristic to filter out the static points in the background and then utilize an off-the-shelf tracking model to generate the corresponding tracks of the large-motion points. The visualization of the sampled points before and after the filtering process is shown in Figure 13.

C. Training Details

We list the training hyperparameters for the track transformer and track-guided policy in Table VII and VIII, which are fixed for all experiments on LIBERO benchmark. We train all models on 4 A100 GPUs with DeepSpeed strategy.

We train the track transformer using the ground truth tracks generated by CoTracker [20] and save the checkpoint with

fold the cloth and pull it to the right



use the broom to sweep the toys into the dustpan and put it in front of the dustpan

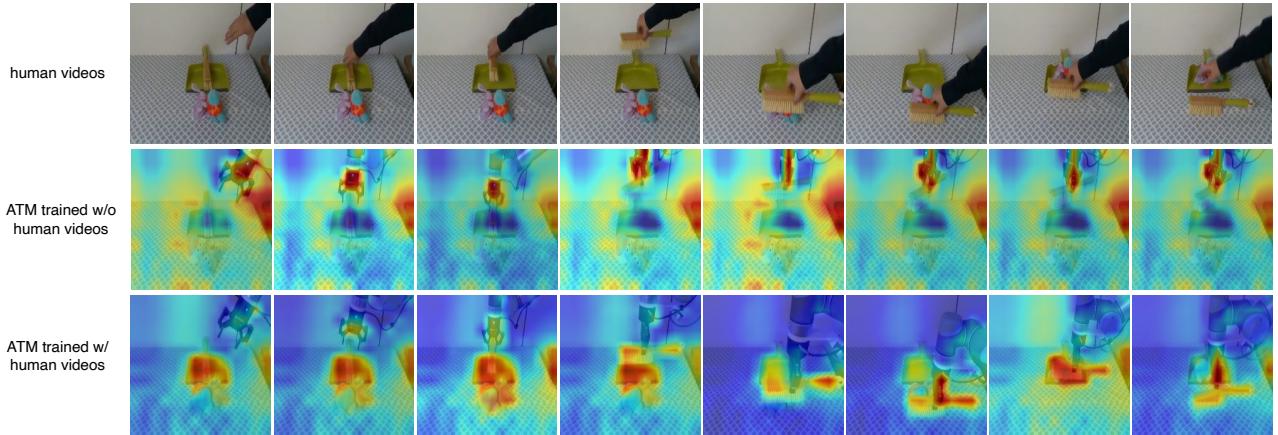
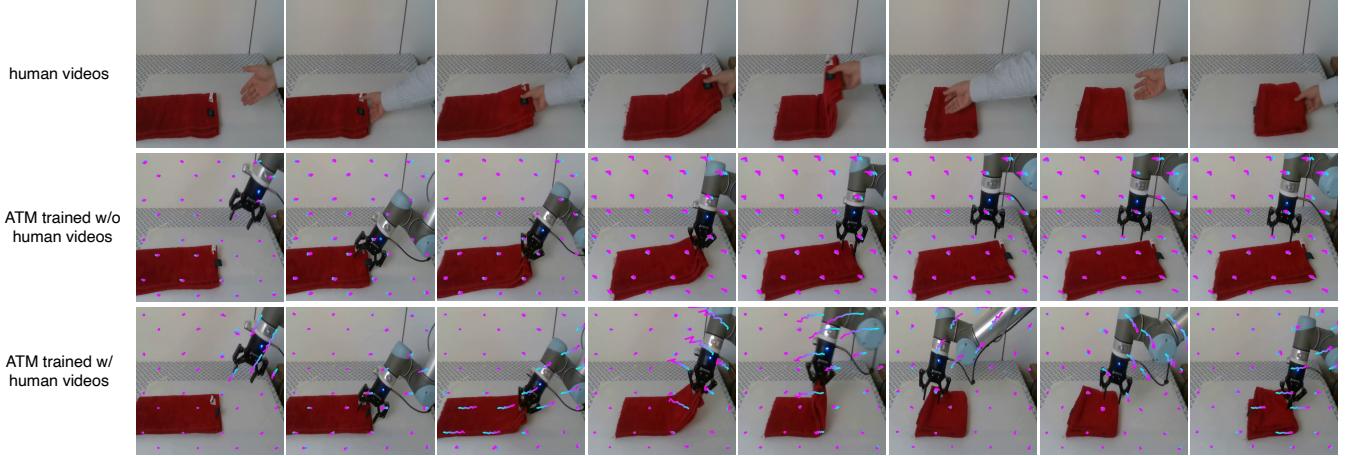


Fig. 14: Attention maps of Track Transformer trained with and without human videos. Including large-scale human video leads to much clearer attention maps, focusing on the object and robot arm, while training without human videos will attend to incorrect areas such as background walls.

the lowest validation loss as our final model to apply for policy learning. We do not incorporate frame stacking for track transformer to avoid causal confusion [52].

We train policies using the expert demonstrations provided by LIBERO, which is collected by human experts through teleoperation with 3Dconnexion Spacemouse [27]. Since the validation loss in behavioral cloning is not always reliable, we save the checkpoint of the last epoch for online rollout evaluation.

fold the cloth and pull it to the right



put the tomato into the pan and close the cabinet door



use the broom to sweep the toys into the dustpan and put it in front of the dustpan

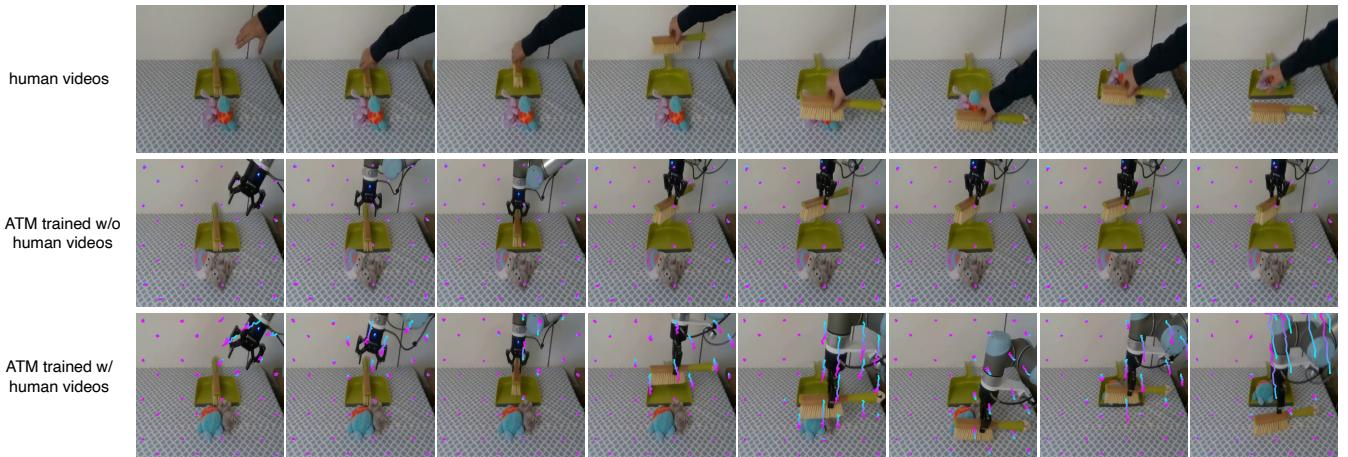


Fig. 15: The visualizations of human demos and rollout videos of ATM policies trained with and without human data. We can see that ATM is able to take advantage of out-of-domain videos, i.e., human videos, to generate more precise tracks, resulting in better policy performance.