Points2Reward: Robotic Manipulation Rewards from Just One Video

Junyao Shi, Joshua Smith, Jianing Qian, Dinesh Jayaraman
University of Pennsylvania
Corresponding to: junys@seas.upenn.edu

Abstract

Detailed and dense reward functions provide clearly graded signals that enable a robot to evaluate and improve its policy, but generating such functions for new tasks is often cumbersome and expertise-intensive. On the other hand, end users can often easily record a video demonstrating the desired object trajectories and goal configurations involved in a new skill, but we lack methods to reliably learn from such a simple specification. We propose "Points2Reward" (P2R), which effectively computes dense rewards from a single video. To do this, P2R tracks task-relevant object points in task demonstrations and policy rollouts, and matches them to compare the desired and achieved object trajectories to generate reward scores. Exploiting recent advances in point tracking and semantic point correspondences, P2R produces high-quality rewards even under significant domain gaps between the demo video and the robot setup, such as embodiment gaps (human vs. robot) or camera viewpoint changes. We demonstrate that P2R correctly evaluates trajectories of varying quality in diverse real-world settings as well as in nine simulated manipulation tasks in standard benchmark suites. We further demonstrate that P2R policy evaluations enable improved downstream policy synthesis in the simulated tasks.

1. Introduction

General-purpose robots of the future must be capable of quickly and painlessly learning new sensorimotor skills from their end users. Finding the right specifications to enable such skill acquisition will be critical: a general specification such as a dense reward function can serve as a versatile evaluation function, which in robot learning, can drive policy optimization in many downstream policy synthesis approaches such as online or offline reinforcement learning, or even model-based planning. However, directly generating reward functions is not scalable: it is regularly cumbersome and error-prone even for experts [6]. Instead, a particularly attractive starting point for a user-friendly form of task specification involves collecting a small dataset of demonstrations of the desired behavior either directly on the target robot, or even simpler, of a human performing the task. However, translating such small demonstration datasets into useful representations for policy evaluation and synthesis, has been elusive in the most general settings.

We focus on the large space of manipulation skills that are fundamentally about moving objects, broadly construed, along some desired trajectories and towards desired goal configurations. For such a task, perhaps the most natural description is to specify sequences of desirable 3-D positions of all the constituent particles on the objects of interest. Then, for a robotic policy attempting the task, computing the distances of each particle from these sequences offers a simple means of evaluating task progress.

Starting from this intuition, we propose to build on recent advances in correspondence matching and tracking with point-based image representations, to: (1) identify and track points on task-relevant objects in a robot or human demonstration video to approximately represent it in the point trajectory format suggested above, (2) compute corresponding points on potentially new objects in a robotic setting of interest, and (3) thereafter track the corresponding points to enable reward computation. In this paper, we investigate various choices for implementing this straightforward high-level idea.

In our investigation, we focus on task specifications that are easiest on the end user and which hitherto have proven most challenging for the robot learner: a *single* user-provided demonstration video, potentially involving challenging domain gaps from the target robotic setting, such as embodiment gaps (e.g., the video is of a human performing the task, rather than the target robot), or camera viewpoint gap (i.e., the video is captured from a different viewpoint), or more broadly scene-level gaps (e.g., lighting, backgrounds, distractor objects). Our point trajectories-based approach outlined above holds promise for overcoming these challenges: focusing on the objects of interest excludes the embodiment from the representation, and may also help overcome many irrelevant scene-level domain gaps such as changes in backgrounds, distractor objects, and lighting. Further, semantic point correspondences

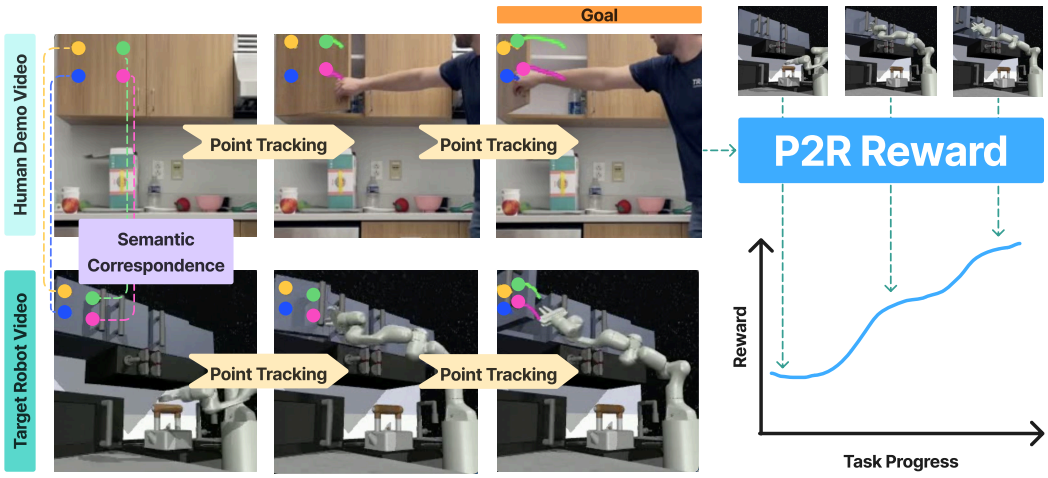

Figure 1. Overview of Points2Reward. We use semantic point correspondence and off-the-shelf point tracking to generate matching point tracks of the demonstration and rollout video. We then use the two point tracks to compute a goal-based reward.

may enable handling variations even in the task-relevant objects between the source video and the target robot scene, e.g., two different instances of a coffee mug. Finally, the transformations of point projections under camera perspective changes are well understood, and may help overcome point gaps. We investigate these potential benefits of our approach thoroughly in our experiments.

In summary, our contributions are:

1. We introduce Points2Reward (P2R), a novel approach that leverages point tracking to generate dense rewards from a single video, even with significant domain gaps.
2. In a variety of real-world and simulated environments, we show that P2R reliably provides accurate robot policy evaluations highly correlated with manually engineered "ground-truth" rewards.
3. We demonstrate that P2R can usefully filter trajectories from a dataset of mixed quality for robot policy synthesis across nine simulated visual control tasks in both Meta-World and Franka Kitchen environments.

2. Background and Problem Setup

This work deals with inferring rewards from a video demonstrating an example of good task performance. This is best understood within the context of a Markov Decision Process (MDP) formulation of robotic tasks.

Markov Decision Process Formulation: An (undiscounted) finite-horizon MDP (S, A, P, R, H, μ) consists of a set of states S that the world can be in, actions A available to the robot at each time, and a transition function $P(s_{t+1}|s_t, a_t)$ that describes how states $s \in S$ evolve in response to an agent action $a \in A$ at each discrete timestep t, up to the horizon of H steps. μ is the initial distribution over states that the robot is spawned into in each episode. Finally, we are ready to describe the reward $r_t = R(s_t, a_t, s_{t+1})$, which is a scalar quantity revealed to the agent after each state transition. This reward determines the agent's objective in the environment: the agent's "policy" $\pi(a_t|s_t)$, which determines its action a_t upon observing a state s_t, must maximize the expected cumulative reward $\mathbb{E}_{\mu,P,\pi}[r_1 + r_2 + r_3 + ...r_H]$. We are concerned with *vision-based* robotic tasks, where the MDP is "partially observed". This entails that the agent does not directly observe states s, but instead observes camera images $o_t = o(s_t)$ of the scene at each step.

Demonstration Video: In addition, we do not assume that rewards $R(s_t, a_t, s_{t+1})$ are available. Instead, we are left to infer these using only a video demonstration δ. This demo δ is an observation sequence $\delta = [o_1, o_2, o_3, ..., o_H]$ encountered by an optimal (or more practically, an approximately optimal) agent policy. We seek to permit maximal flexibility in how δ is collected: for example, the correct behavior might be demonstrated by a human actor rather than a robot agent, from a camera viewpoint that is different than the robot's own, with objects that are different and in a room that is different from the target robotic task setup.

The Reward Inference Problem: When "inferring rewards" from δ, we aim to achieve the following. Given a new observed trajectory $\tau = [o_{\tau 1}, o_{\tau 2}, ..., o_{\tau H}]$, we aim to assign scalar rewards $r_t = \hat{R}_t(o_{\tau t}, \delta)$ at each time t, such that they "correctly evaluate" the rollout. In particular, we would like that higher *inferred* rewards or returns (cumulative returns over an episode) should always correspond to higher *true* rewards or returns respectively. This in turn implies that any optimum of the inferred reward objective will also be an optimum of the *true* reward objective. Thus, such an inferred reward may be used in place of the true reward, for downstream synthesis of optimal policies, such as with reinforcement learning or model-based planning.

3. Our Approach: Points2Reward

As discussed above, we aim to infer vision-based rewards from a demo video δ, ideally placing minimal requirements on how well δ matches the target robotic task setup (scene background, agent embodiment, camera viewpoint, object instances, distractor objects etc.). As we have argued above, the essence of a manipulation task is often well-described by the desired movements of various particles on the objects of interest, and distances from those desired movements may offer a simple route to "inferred rewards".

We seek as direct an implementation of this principle as possible. In line with this, our method uses a task-relevant point track representation of the demonstration video δ and a semantically corresponding point track representation of the target video τ. These are described in detail in Section 3.1. Then, Section 3.2 describes how these point tracks are aligned and a dense per-timestep reward is computed by measuring the misalignment of each trajectory frame from the demo video. Figure 1 presents a schematic overview.

3.1. Point Track Representations

We now discuss how the demo video δ and target robot trajectory τ are represented as corresponding point tracks, over which dense rewards will be computed in the Section 3.2.

Tracking Desired Object Point Tracks. Given the demo video, we initialize a uniform grid, typically of size 20x20 points over the first frame. We track these points through the demonstration video to produce a set of point trajectories using CoTracker2 [23].

Next, we use simple heuristics to post-process these point tracks to exclude agent points and other task-irrelevant points in the scene. First, we automatically filter out agent points, a standard procedure in prior works [2, 19]. Second, all stationary point tracks in the video are discarded as task-irrelevant: we found that this simple heuristic sufficed to identify task-relevance in all our test settings, but alternative more sophisticated approaches for task-relevance may be used, for example, using sketches or language descriptions of the task as in [22, 41].

We are left with a set of desired task-relevant object point trajectories $T_\delta = \{t_{\delta 1}, t_{\delta 2} ..., t_{\delta n}\} \in \mathbb{R}^{H \times n \times d}$ with video length H and number of filtered points n. In our experiments, we use $d = 3$-dimensional point tracks when demo videos are captured with a single RGB-D camera, but we demonstrate that when the demo is only captured with an RGB camera, $d = 2$-dimensional point tracks can also yield good rewards.

Semantic Correspondence. Having computed the demo point track representation T_δ, we must also compute similar point track representations for each robot policy rollout that we are looking to compute dense rewards for. To do this, we now seek to match each filtered point in T_δ to a corresponding point in each robot rollout video through semantic correspondence matching.

We match the first frame of the demonstration video to the first frame in the robot rollout. We compute DINOv2 [37] patch embeddings for the initial images in both the demo and robot rollout videos. We then compute the cosine similarity between all patches in the demo and robot initial frames. Given a point within a patch in the demo video, we take the maximum similarity patch in the target rollout as the matched location. The matched points are then tracked in the robot rollout video using CoTracker just as done above for the demo video, providing a corresponding set of point trajectories $T_\tau = \{t_{\tau 1}, t_{\tau 2}, ..., t_{\tau n}\}$.

3.2. Point Track Matching For Dense Rewards

Having computed the point track representations T_δ and T_τ for the demo and rollout video, we now proceed to match the rollout to the demo to compute per-timestep rewards.

Aligning Demo and Rollout Tracks. The point matching in Sec 3.1 relied only on semantic cues, because we do not assume that the demo video and the policy rollout video are geometrically aligned. In the most general case, the robot video may be captured with different task-relevant object instances and from a different relative camera viewpoint (as in some of our experimental settings). In such cases, the point correspondences detected on the first frames serve as the basis for estimating a transformation that aligns the point tracks in preparation for reward computation.

For example, if the tracks are in 3-D, we may in the most general case compute a 3-D similarity transformation $\mathcal{S}_{\tau \to \delta}$ that best aligns the semantic point correspondences $t_{\tau 1}$ detected in the first frame of the rollout video to the first frame of the demo $t_{\delta 1}$ [17]. This transformation is then applied to the rollout points at all subsequent frames i.e. $\mathcal{S}_{\tau \to \delta}(T_\tau) = \{\mathcal{S}_{\tau \to \delta}(t_{\tau 1}), \mathcal{S}_{\tau \to \delta}(t_{\tau 2}), \mathcal{S}_{\tau \to \delta}(t_{\tau 3}), ...\mathcal{S}_{\tau \to \delta}(t_{\tau n})\}$. As we will show in some of our simpler experimental setups, we find it sufficient to compute much simpler transformations $\mathcal{S}_{\tau \to \delta}$, such as plain 2D translations to align 2-D rollout point tracks to 2-D demo point tracks.

Computing Rewards from Point Trajectories. After aligning demonstration and target trajectory tracks, we formulate our reward as the negated distance to goal, where the goal is the n point locations $T_\delta[H]$ in the last demo frame:

$$\hat{r}_t = -\frac{1}{L_\tau} \hat{R}_t(o_{\tau t}, \delta) = d(S_{\tau \to \delta}(T_\tau)[t], T_\delta[H]), \quad (1)$$

where $d(\cdot, \cdot)$ is the sum of n point-wise Euclidean distances between the point sets. To account for trajectories of different lengths, we divide the reward sum by the trajectory length of the rollout L_τ. In our ablations (Section 5.4), we evaluate alternative distance measures, such as Dynamic Time Warping [10] distance, Sinkhorn Optimal Transport [39] distance, and delta distance to goal $\hat{r}_{t+1} - \hat{r}_t$ between the next step and current step.

4. Related Work

4.1. Methods to Generate Rewards

Various works have explored how to generate reward or value functions for robotics. These methods can largely be categorized into the 3 categories below.

Image Embeddings As Reward Functions. Several early works in inverse reinforcement learning [1, 18, 53, 59] infer rewards from an in-domain robot demonstration dataset, sometimes with different embodiments [44, 45, 57], such that trajectories more similar to the demonstrations are scored more highly. Other works aimed to leverage more out-of-domain datasets or models to learn rewards based on visual embeddings, such as in-the-wild human videos [7, 36] and vision-language embeddings pre-trained on internet data [3, 43, 46]. The Value-Implicit Pre-training (VIP) line of works [5, 30, 31] train visual embeddings on internet videos with offline RL objectives to produce goal-conditioned rewards. Other recent works [15, 16, 29], such as ROT [15] propose to generate rewards by performing trajectory matching using image embedding of a single in-domain demonstration. While these methods focus on generating rewards through image embeddings, we use points as a more natural and interpretable representation for robotic task progress. This allows us construct more informative reward signals and better compensate for domain gaps of out-of-domain demo videos, as argued above and demonstrated throughout our experiments.

Foundation Model Language-Based Rewards. Recently, many have leveraged Large Language Models (LLMs) and Vision-Language Models (VLMs) to construct language-based rewards for robotics. The advantage of using language to specify rewards is we can circumvent the need of collecting demonstration videos for a task. Some works have used them to provide sparse [9, 27, 34] and dense [33] reward values or preference feedback [25, 49], while others even prompt them to generate code for reward or cost functions [20, 21, 32, 40, 48, 51, 55]. Notably, GVL [33] leverages VLM to predict per-frame task progress in-context. ReKep [21] employs vision foundation models to propose keypoints and then prompts a VLM to generate keypoint cost functions in code. Aside from the high cost and slow inference speeds associated with using LLMs and VLMs, we empirically show that the rewards they produce are often inaccurate and noisy due to their lack of spatial and physical understanding.

Object & Geometry-Oriented Rewards. Most related to our work, recent methods haven taken an object and geometry oriented approach to generating rewards. ReKep [21], as discussed above, produces keypoint-based constraint functions in code, but is limited by its reliance on VLMs for spatial reasoning. Concurrently with our work, HuDOR [13] builds reward functions by matching 2D movements of the object centroid between a human demonstration and robot policy in the same scene. By contrast, we compute goal-conditioned rewards using 3D point trajectories semantically and temporally consistent across different videos. Motion of individual points are a naturally more informative representation of task progress than the object centroid, especially for articulated and deformable objects. We are also able to explicitly compensate for domain gaps and compute rewards from an out-of-domain demonstration, such as a human demonstration in a different scene with different objects, by leveraging points as more fine-grained geometric features. We compare against and outperform a strengthened variant of HuDOR.

4.2. Applications of Point Tracking in Robotics

The remarkable recent advancements in point tracking methods [8, 23, 24] have inspired a wide range of applications of these techniques in the field of robotics.

As action space. As a straightforward extension of the ability to track any points, several works [4, 14, 26, 47, 56, 58] have investigated directly acquiring robot actions from point tracks by computing geometric transforms or visual servoing without any policy training. However, since point tracks do not inherently contain information about fine-grained robot-object interaction (e.g. grasping), these approaches face several limitations: they typically rely on task-specific assumptions, manually designed robot primitives, and are sensitive to the tracking accuracy of each individual point.

As intermediate representation for behavior cloning. Another common strategy employed in other works [4, 11, 14, 28, 42, 50, 52] is to use point tracks as an intermediate representation for policy learning, typically leading to a two-stage approach: first, training a point track prediction model, and then training a policy model conditioned on these point track predictions to generate robot actions based on robot data.

As we have argued above, point tracks are a particularly well-suited representation for a reward function specifying a task. P2R therefore focuses on exploiting point tracks for inferring rewards from a single demo video.

5. Experiments

Our method designs a point-based reward function that facilitates efficient autonomous performance evaluation and policy learning from noisy human and robot demonstrations. With both simulated demonstrations and human demonstrations, our experiments aim to answer: 1. Does P2R enable more accurate evaluation of robot policies than prior baselines? 2. How robust is P2R to domain gaps between the demonstration and target videos? 3. Can we leverage P2R for more effective robot policy synthesis? 4.

What are the key design choices in P2R, and how do they impact performance?

5.1. Experimental Setup

Environments. We adopted the evaluation paradigm from [15] and evaluate P2R on 6 tasks in Meta-World [54]: Hammer, DrawerClose, DrawerOpen, BinPicking, ButtonPress, and DoorOpen as well as 3 tasks in Franka Kitchen [12]: SlideDoor, OpenDoor, and OpenMicrowave. For all environments except for Meta-World DrawerOpen and ButtonPush we collect human demonstrations. These environments are depicted in Figure 2. In Meta-World, the object positions are randomized for each episode, whereas in Franka Kitchen, the robot pose is randomized.

In some experiments, we additionally evaluate two real-world human-only deformable-object manipulation tasks: ShirtFolding (pick up the right side of the shirt and fold it in half, see Figure 4) and RopeShaping (deform a straight rope into a "W" shape). For these two tasks, we only have human videos to serve both as demos as well as target success or failure trajectories.

Baselines. We evaluate P2R against a comprehensive suite of representative baselines for computing vision-based rewards with or without reference to demo videos, as reviewed in Sec 4. **ROT** [15] uses Sinkhorn distance [38] on the demo and target robot video embeddings matched by optimal transport. We use one demo to pre-train the encoder for computing reward embeddings. We also compare P2R to popular visual encoders and reward functions pre-trained on egocentric human videos. **R3M** [36] and **VIP** [30] are popular pre-trained image encoders on robot-specific data. Reward for R3M embeddings is computed using the euclidean distance between the current frame embedding and the goal embedding. For VIP, we adopted the original reward function formulation.

Next, we compare to a concurrently developed object-oriented method, **HuDOR**, which estimates object-movement as the 2D movement of its centroid. We find that the original HuDOR fares poorly in our settings because it requires LangSam [35] language-based segmentations, which frequently fail or are inconsistent across an episode. To provide a more useful point of comparison, we develop **HuDOR+**, where we strengthen HuDOR with the task-relevant point filtering and semantic point matching components of our approach, as described in Section 3.1. The reward computed from corresponding point set trajectories is formulated as the difference in mean point displacement for temporally aligned frames in the demonstration and target robot videos.

Next, we compare to a recent LLM/VLM reward generation baseline, Generative Value Learning (**GVL**) [33]. We compare against 1-shot GVL, which is given both a reference video and a text description of the task to be completed. An LLM is then prompted with the reference video as context for task progress, and is asked to output the task completion progress of the target robot video. Since this is a value measure similar to VIP, we take the difference in timesteps' LLM-prescribed values as our reward function. This is done with videos subsampled to 30 total frames. These rewards are then linearly interpolated back to the original video length.

We also considered another related VLM-based approach, ReKep [21]. However, it is not designed to perform reward generation consistent across multiple episodes.

Demo Videos. Our demo videos in various tasks are collected with both robot and human agents demonstrating the desired behaviors. Robot demonstrations are collected with RGB-D cameras. For convenience, human demonstrations are collected with simple RGB cameras filming a human completing a task in an approximately similar real-world scene. As illustrated in Figure 2, there are significant domain shifts between robot and human demonstrations, given variations in lighting, object appearances, positions, and visual fidelity.

5.2. Policy Evaluation with P2R

Reward Gaps Between Success and Failure Trajectories. We first collect obvious success and failure videos for all 11 tasks (9 sim + 2 real) and compute rewards using P2R and baselines for each trajectory. Demos for the real world tasks and for the 3 Franka Kitchen tasks are human-provided, and those for the Meta-World tasks are obtained from expert-level robot policies.

How well does each approach separate these obvious successes and failures? Figure 3 averages the instantaneous reward gap between successes and failures over all 11 tasks. Figure 4 zooms into the case of real-world ShirtFolding: (bottom) shows one success and one failure video, and (top) shows the inferred reward gap between those two videos. The trends in both plots are similar: P2R reward gaps are largest and most clearly reflect the increasing gap between success and failure episodes over time. R3M and VIP encode the entire image as one vector and as such struggle both to overcome domain gaps as well as to capture fine-grained details. HuDOR summarizes each object as one point, and while it is arguably the best-performing baseline *on average* across tasks in Figure 3, it performs particularly poorly in cases like ShirtFolding (Figure 4) where the shirt's centroid is obviously too coarse of a description. The VLM-based GVL approach fares poorly both on average and on ShirtFolding.

Precision-Recall Of Thresholded Rewards. Having established the value of P2R at this coarse-grained task, we now evaluate it in more difficult settings consisting of finer gradations in trajectory quality. To do this, we collect replay buffers from reinforcement learning training episodes

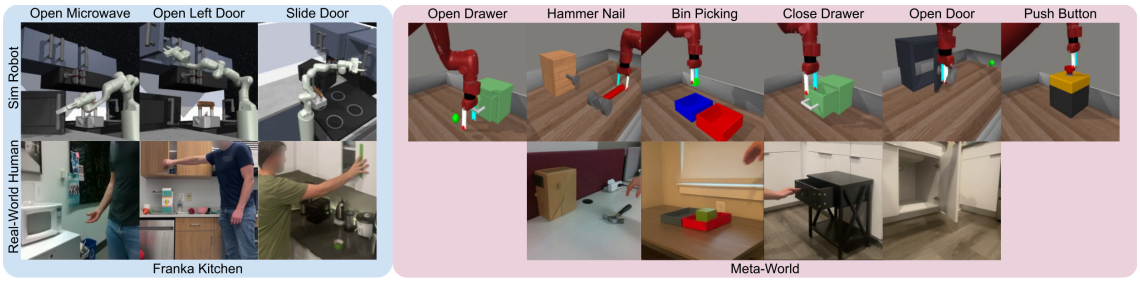

Figure 2. Simulated Environment Suite. 7 environments with corresponding human demo and 2 additional robot-only Meta-World envs

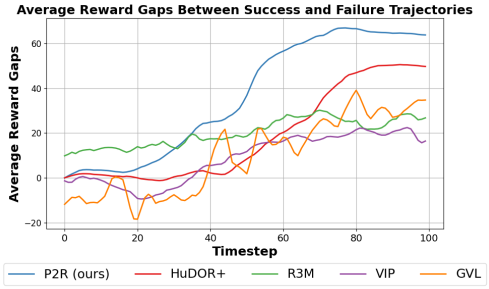

Figure 3. Instantaneous reward gap between successful and failed rollouts over time, averaged across 11 tasks (9 sim + 2 real world).

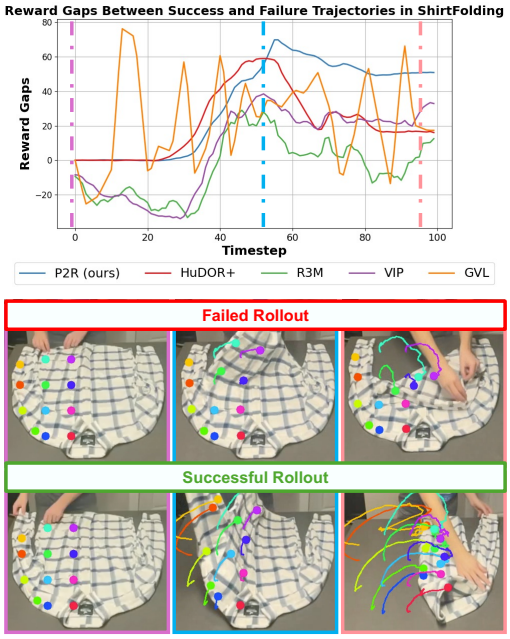

Figure 4. Reward gap between successful and failed shirt folding target robot videos. Colored points and lines on the shirt represent the initial point selections and their trajectories. Dashed lines on the plot correspond to the timesteps for frames that have the same color outline.

in each simulated environment to yield a range of episode samples, achieving varying success rates and true episode rewards. On these more challenging trajectory datasets, we use thresholded inferred rewards from each method as a success/failure classifier, and compute its average precision (area under the precision-recall curve), a standard metric for classification accuracy (higher is better). See results in 1. We find that, on average, P2R yields the highest average precision (AP). This augurs well for downstream applications of P2R for policy synthesis, such as with offline RL methods. We will evaluate a simple version of such an approach later in Section 5.3.

	P2R (Ours)	HuDOR	ROT	R3M	VIP	GVL
Meta-World (Robot Demo)	**0.78**	**0.79**	0.67	0.48	0.72	0.36
Meta-World (Human Demo)	**0.65**	0.60	0.21	0.25	0.25	0.25
Franka (Robot Demo)	**0.94**	0.58	0.88	0.62	0.96	0.37
Franka (Human Demo)	0.41	**0.49**	0.28	0.21	0.19	0.26
Average	**0.70**	0.62	0.51	0.39	0.53	0.31

Table 1. Average-Precision For success detection using P2R reward, averaged across all tasks within an environment. Rewards are inferred from a single "Robot" or "Human" demo.

Correlations With Manually Designed Rewards. Thus far, we have relied solely on ground-truth binary episode-level successes and failures to evaluate inferred rewards. Besides these binary notions, all 9 simulated environments also offer a carefully engineered task-specific simulator-state-based reward function designed to provide useful feedback to reinforcement learning policies at all stages of training. How closely do our vision-based rewards inferred from one demo approach these "gold standard" reward functions, in terms of both episodic returns and instantaneous rewards?

We compute the rank-order correlation (ROC) between generated and environment rewards for each method to compare reward alignment. Intuitively, the ROC score for a reward inference method measures the extent to which higher rewards computed by that method are indicative of higher manually engineered rewards. Figure 5 shows that in Meta-World and Franka Kitchen, P2R consistently performs well across all tasks and on average, using both human and robot demonstrations. However, we observe a significant drop in ROC for P2R on Franka Kitchen tasks when transferring from generating rewards with robot demo to human demo. Franka Kitchen tasks such as open door and open microwave contain significant object movement

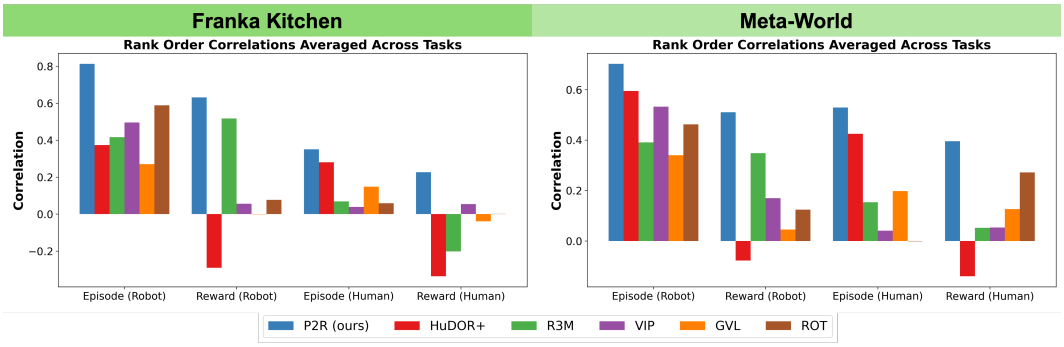

Figure 5. Rank-Order Correlations for Cumulative ("Episode") and Instantaneous ("Timestep") rewards vs manually engineered reward functions. Rewards are inferred from a single "Robot" or "Human" demo.

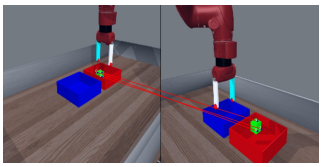

Figure 6. Point Matching Across Viewpoints

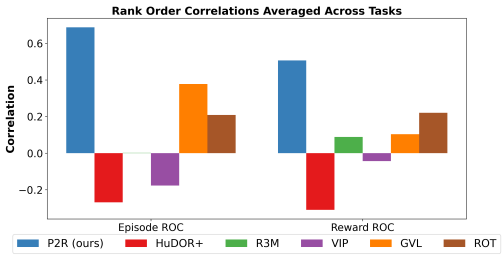

Figure 7. Metaworld Rank-Order Correlations with Camera Change (Robot Demo)

towards the cameras, that 3D points in our robot demo experiments capture much better than 2D points in our experiments with human demos. P2R generally outperforms baselines as evaluated on both per-episode and per-timestep reward. P2R's high per-episode correlation shows that its generated rewards can accurately measure overall robot rollout success. Additionally, the strong per-timestep ROC demonstrates that P2R not only effectively captures performance at the episode level but also provides a precise, dense measure of task progression throughout each robot rollout.

Having thoroughly validated P2R on the human-to-robot embodiment and real-to-sim scene gap in the foregoing experiments, we now test another challenging gap: large camera viewpoint variations between the demo and the target setup (Figure 6). Recall that our alignment approach of Sec 3.2 is capable of handling such variations. Indeed, Figure 7 shows that P2R continues to substantially outperform the baselines.

5.3. Policy Synthesis Driven By P2R

Finally, we validate P2R for a downstream use of inferred rewards: filtered behavior cloning based policy synthesis, as also performed in GVL [33]. Inferred rewards can help filter mixed-quality datasets to select successful demonstrations to improve imitation learning. Specifically, we choose the top k robot episodes from collections of hundreds of episodes to train our behavior cloning (BC) policies.

Simulation Results. Figure 9 show that P2R outperforms all baselines once more for all values of k. This is in line with the AP scores reported in Table 1: P2R simply selects better demonstrations to imitate.

Real-World Results. We present real-world policy synthesis for 3 tasks (hardest to easiest): towel folding, rotating a valve, opening a drawer. We train policies on mixed-quality (35% successes) human demos using filtered BC. Tab. 2 shows that P2R is best on all tasks, in line with simulation policy results. Fig. 8 shows an example of semantic matching operating robustly between our very different (viewpoint, background, object position) human demo and robot setups.

Table 2. Real-World Dataset Filtering: Filtered BC Performance

Task	Num. Demos	P2R	HUDOR++	VIP	GVL
Towel Fold	200	**11/20**	3/20	1/20	3/20
Open Drawer	100	**20/20**	14/20	14/20	2/20
Rotate Valve	100	**8/20**	1/20	2/20	5/20

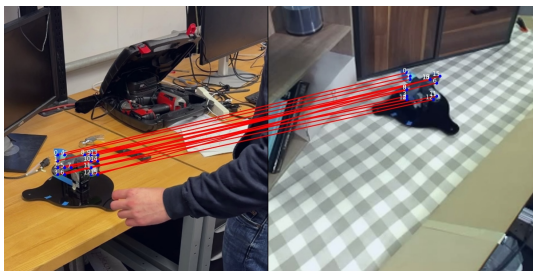

Figure 8. Valve Rotation Human Demo vs Robot Rollout: Domain Gaps and Point Matches.

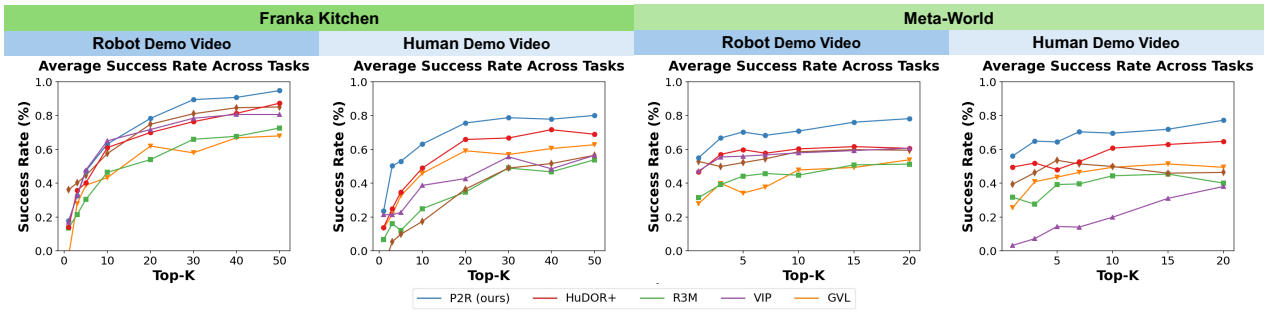

Figure 9. Behavior Cloning Performance Using Top k Robot Episodes from Mixed-Quality Datasets over varying k.

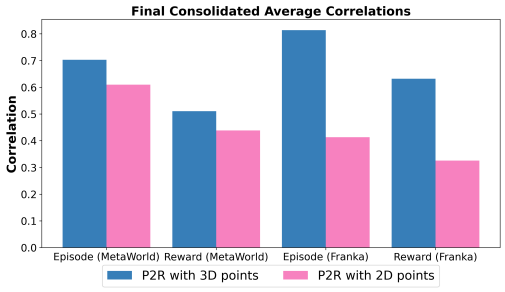

Figure 10. P2R Rank Order Correlations with 2D vs. 3D Points

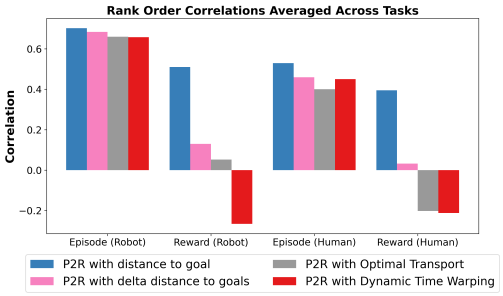

Figure 11. P2R Rank Order Correlations with Different Distance Functions in Meta-World

5.4. Ablations

We show ablation experiments to justify the various design choices of P2R.

Using 2D vs 3D points. We evaluate the ROC scores of P2R when raising points to 3D or when keeping trajectories 2D. In Figure.10 we find that P2R performs best with 3D points. 2D point representations may struggle capturing movement along the depth axis, giving P2R with 3D points an advantage on tasks with this property. For example, P2R with 2D point representation struggles in the Franka Kitchen Open Microwave task, where the microwave opens directly at the camera, yielding little 2D pixel displacement to assign rewards with. Due to this major advantage in using 3D points and its empirical advantages, we use 3D points as our representation in our main method when depth images are available (Robot demonstrations in our experiments).

Distance function for P2R. P2R generates corresponding point trajectories in 2D or 3D space for both the demonstration and target robot videos. The similarity of these point trajectories can be evaluated by a variety of distance measures. We test three additional distance measures, Optimal Transport, Dynamic Time Warping, and euclidean distance to goal delta. We implement our optimal transport cost function and reward assignment according to ROT [15] where the Sinkhorn distance [38] compares demonstration and rollout trajectory representations. Additionally we implement Dynamic Time Warping [10], to relatively slow down and speed up trajectory timesteps to match trajectories in time before taking the euclidean distance between aligned timesteps. Finally, we test euclidean distance to goal delta, which is similar to euclidean distance to goal functions, except that reward for a given timestep is the reduction in that timestep's euclidean distance to goal.

We compute ROC scores using P2R trajectory generation for each distance measure in Figures 11 . Across all experiments, we find that euclidean distance to goal is equal or slightly better for per-episode ROC. Additionally, we find that euclidean distance to goal is always much better than other baselines for per-timestep reward ROC. This suggests that manually engineered and optimized reward functions often correlate to a measure of distance to a target state. Since euclidean distance to goal outperforms other examined cost functions in ROC scores across both demo types and simulated environments, we adopt euclidean distance to goal as P2R's distance measure.

6. Conclusion and Limitations

In this work, we introduced Points2Reward (P2R), a novel method that leverages point tracking to translate a single demonstration video into an informative dense reward function highly correlated with true task progress, even across significant domain gaps, such as embodiment, viewpoint, and scene-level differences. Moreover, we validate its practical utility by using P2R to evaluate and filter mixed-quality trajectories, resulting in improved robot policy synthesis.

Our approach still faces challenges related to tracking and semantic correspondence failures, particularly under

significant occlusion or visual ambiguity. When facing challenging domain gaps, even though P2R outperforms baselines, there is still some performance degradation compared to learning from an in-domain demo. Additionally, relying solely on individual point movements can sometimes lead to misspecified tasks-for instance, incorrectly interpreting a demonstration of a door opening to the right when the target scenario involves a left-opening door. This limitation partly arises from the simplistic design choices in this initial implementation, which currently neglects relationships among points or groups of points. Incorporating richer relational reasoning could enable the method to generalize effectively to more sophisticated, multi-object manipulation tasks, such as stacking a set of blocks.

References

[1] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004. 4

[2] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022. 3

[3] Kate Baumli, Satinder Baveja, Feryal Behbahani, Harris Chan, Gheorghe Comanici, Sebastian Flennerhag, Maxime Gazeau, Kristian Holsheimer, Dan Horgan, Michael Laskin, et al. Vision-language models as a source of rewards. *arXiv preprint arXiv:2312.09187*, 2023. 4

[4] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation, 2024. 4

[5] Chethan Bhateja, Derek Guo, Dibya Ghosh, Anikait Singh, Manan Tomar, Quan Vuong, Yevgen Chebotar, Sergey Levine, and Aviral Kumar. Robotic offline rl from internet videos via value-function pre-training. *arXiv preprint arXiv:2309.13041*, 2023. 4

[6] Serena Booth, W Bradley Knox, Julie Shah, Scott Niekum, Peter Stone, and Alessandro Allievi. The perils of trial-and-error reward design: Misdesign through overfitting and invalid task specifications. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*, 2023. 1

[7] Annie S Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021. 4

[8] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023. 4

[9] Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. *arXiv preprint arXiv:2303.07280*, 2023. 4

[10] Toni Giorgino. Computing and visualizing dynamic time warping alignments in r: The dtw package. *Journal of Statistical Software*, 31(7):1–24, 2009. 3, 8

[11] Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Vuong, and Ted Xiao. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches, 2023. 4

[12] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *ArXiv*, abs/1910.11956, 2019. 5

[13] Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. *arXiv preprint arXiv:2410.23289*, 2024. 4

[14] Siddhant Haldar and Lerrel Pinto. Point policy: Unifying observations and actions with key points for robot manipulation. *arXiv preprint arXiv:2502.20391*, 2025. 4

[15] Siddhant Haldar, Vaibhav Mathur, Denis Yarats, and Lerrel Pinto. Watch and match: Supercharging imitation with regularized optimal transport. *CoRL*, 2022. 4, 5, 8

[16] Siddhant Haldar, Jyothish Pari, Anant Rai, and Lerrel Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023. 4

[17] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 3

[18] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016. 4

[19] Edward S Hu, Kun Huang, Oleh Rybkin, and Dinesh Jayaraman. Know thyself: Transferable visual control policies through robot-awareness. *arXiv preprint arXiv:2107.09047*, 2021. 3

[20] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. 4

[21] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024. 4, 5

[22] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. In *Fortieth International Conference on Machine Learning*, 2023. 3

[23] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. 2023. 3, 4

[24] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. 2024. 4

[25] Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. Motif: Intrinsic motivation from artificial intelligence feedback. *arXiv preprint arXiv:2310.00166*, 2023. 4

[26] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to Act from Actionless Videos through Dense Correspondences. *arXiv:2310.08576*, 2023. 4

[27] Teyun Kwon, Norman Di Palo, and Edward Johns. Language models as zero-shot trajectory generators. *IEEE Robotics and Automation Letters*, 2024. 4

[28] Mara Levy, Siddhant Haldar, Lerrel Pinto, and Abhinav Shirivastava. P3-po: Prescriptive point priors for visuo-spatial generalization of robot policies. *arXiv preprint arXiv:2412.06784*, 2024. 4

[29] Jingxian Lu, Wenke Xia, Dong Wang, Zhigang Wang, Bin Zhao, Di Hu, and Xuelong Li. Koi: Accelerating online imitation learning via hybrid key-state guidance. *arXiv preprint arXiv:2408.02912*, 2024. 4

[30] 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. *arXiv preprint arXiv:2210.00030*, 2022. 4, 5

[31] Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023. 4

[32] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv: Arxiv-2310.12931*, 2023. 4

[33] Yecheng Jason Ma, Joey Hejna, Ayzaan Wahid, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, Jonathan Tompson, Osbert Bastani, Dinesh Jayaraman, Wenhao Yu, Tingnan Zhang, Dorsa Sadigh, and Fei Xia. Vision language models are in-context value learners, 2024. 4, 5, 7

[34] Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning*, pages 14743–14752. PMLR, 2022. 4

[35] Luca Medeiros et al. Lang-segment-anything. https://github.com/luca-medeiros/lang-segment-anything, 2023. Accessed: 2025-02-15. 5

[36] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022. 4, 5

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. 3

[38] Georgios Papagiannis and Yunpeng Li. Imitation learning with sinkhorn distances, 2022. 5, 8

[39] Georgios Papagiannis and Yunpeng Li. Imitation learning with sinkhorn distances, 2022. 3

[40] Shivansh Patel, Xinchen Yin, Wenlong Huang, Shubham Garg, Hooshang Nayyeri, Li Fei-Fei, Svetlana Lazebnik, and Yunzhu Li. A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards. *arXiv preprint arXiv:2502.08643*, 2025. 4

[41] Jianing Qian, Yunshuang Li, Bernadette Bucher, and Dinesh Jayaraman. Task-oriented hierarchical object decomposition for visuomotor control. *ArXiv*, abs/2411.01284, 2024. 3

[42] Juntao Ren, Priya Sundaresan, Dorsa Sadigh, Sanjiban Choudhury, and Jeannette Bohg. Motion tracks: A unified representation for human-robot transfer in few-shot imitation learning. *arXiv preprint arXiv:2501.06994*, 2025. 4

[43] Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. *arXiv preprint arXiv:2310.12921*, 2023. 4

[44] Pierre Sermanet, Kelvin Xu, and Sergey Levine. Unsupervised perceptual rewards for imitation learning. *arXiv preprint arXiv:1612.06699*, 2016. 4

[45] Pierre Sermanet, Corey Lynch, Jasmine Hsu, and Sergey Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. *arXiv preprint arXiv:1704.06888*, 2017. 4

[46] Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bıyık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. Roboclip: One demonstration is enough to learn robot policies. *Advances in Neural Information Processing Systems*, 36, 2024. 4

[47] 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. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5397–5403. IEEE, 2024. 4

[48] David Venuto, Sami Nur Islam, Martin Klissarov, Doina Precup, Sherry Yang, and Ankit Anand. Code as reward: Empowering reinforcement learning with vlms. *arXiv preprint arXiv:2402.04764*, 2024. 4

[49] Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. In *Proceedings of the 41th International Conference on Machine Learning*, 2024. 4

[50] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning, 2023. 4

[51] Tianbao Xie, Siheng Zhao, Chen Henry Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. Text2reward: Reward shaping with language models for reinforcement learning. *arXiv preprint arXiv:2309.11489*, 2023. 4

[52] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024. 4

[53] Daniel Yang, Davin Tjia, Jacob Berg, Dima Damen, Pulkit Agrawal, and Abhishek Gupta. Rank2reward: Learning shaped reward functions from passive video. *ICRA*, 2024. 4

[54] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019. 5

[55] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. *Arxiv preprint arXiv:2306.08647*, 2023. 4

[56] Chengbo Yuan, Chuan Wen, Tong Zhang, and Yang Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024. 4

[57] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. *Conference on Robot Learning (CoRL)*, 2021. 4

[58] Yifeng Zhu, Arisrei Lim, Peter Stone, and Yuke Zhu. Vision-based manipulation from single human video with open-world object graphs. *arXiv preprint arXiv:2405.20321*, 2024. 4

[59] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, pages 1433–1438. Chicago, IL, USA, 2008. 4