arXiv:2607.21655v1 [cs.RO] 22 Jul 2026

Progress Reward Modeling for Robotic Learning: A Comprehensive Survey

Jianshu Zhang[1*], **Keliang Wu**[1*], **Haoran Lu**[1*], **Anbang Liu**[1], **Ce Zhang**[2], **Weijie Yin**[3], **Chengxuan Qian**[4], **Xiyuan Yang**[5], **Zhenyu Pan**[1], **Guo Ye**[1], **Han Liu**[1]

[1]Northwestern University, [2]Carnegie Mellon University, [3]University of Wisconsin–Madison, [4]University of California, Santa Barbara, [5]University of Illinois Urbana-Champaign
[*]Equal Contribution

Robotic learning takes place in dynamic environments with large behavior spaces. A terminal success signal only tells the robot whether the task is completed. It does not explain whether the current behavior is making progress, remaining unchanged, or undoing earlier progress. For this reason, recent studies have increasingly explored progress rewards that provide feedback during task execution. However, the current literature lacks a shared framework. Existing methods use different observations, goal specifications, output signals, supervision sources, and evaluation protocols. This makes it difficult to compare them and understand what their results actually validate. In this survey, we provide **a unified view of progress reward modeling for robotic learning**. We organize the field in three connected steps. We first study the *interface* of a progress model. This defines the problem from the outside by asking what information the model receives and what form of progress signal it produces. We then move inside the model and study the *methods* used to construct this signal. This reveals the different assumptions and mechanisms behind progress estimation and reward generation. Finally, we examine the *data and benchmarks* that support these methods. This shows how progress supervision is obtained and what different evaluations actually measure. Together, these three perspectives connect what a progress model is, how it is built, and how its quality is validated. We further summarize the main limitations of current approaches and discuss future research directions.

https://github.com/sterzhang/Awesome-Progress-Models

1 Introduction

Robotic learning takes place in dynamic, sequential, and high-dimensional environments (Lu et al., 2024; Wu et al., 2025; Generalist AI Team, 2025; Physical Intelligence, 2026; NVIDIA, 2025). A robot must interact with objects and the physical world over many steps, and similar actions can have very different effects on task progress depending on the current state and goal (Intelligence et al., 2025; Ye et al., 2026). The same task may also be completed through different action sequences (Yuan et al., 2026; Baumli et al., 2023). These properties create a large behavior space and make it difficult to learn only from final task outcomes.

Most robotic learning systems rely on a terminal success signal(Liu et al., 2023; Chen et al., 2026b; Lu et al., 2026). Such a signal tells the robot whether the task was eventually completed, but provides little information about the intermediate execution. It cannot explain whether an action moves the task forward, makes no meaningful change, or undoes earlier progress. This problem becomes more severe in long-horizon tasks, where successful outcomes are rare and many decisions must be made before receiving any useful feedback (Chen et al., 2026c; Yong et al., 2026).

Progress rewards provide a richer signal by describing how task execution changes over time. They can provide dense feedback before the task is completed and improve credit assignment during policy learning. They can also help distinguish useful behavior from irrelevant motion, identify stagnation or regression, and detect failures before the final outcome. Beyond policy learning, progress signals can support online monitoring, trajectory reranking, data filtering, planning, action selection, and failure recovery. A reliable progress model can therefore serve not only as a reward function, but also as a general evaluator of ongoing robot behavior.

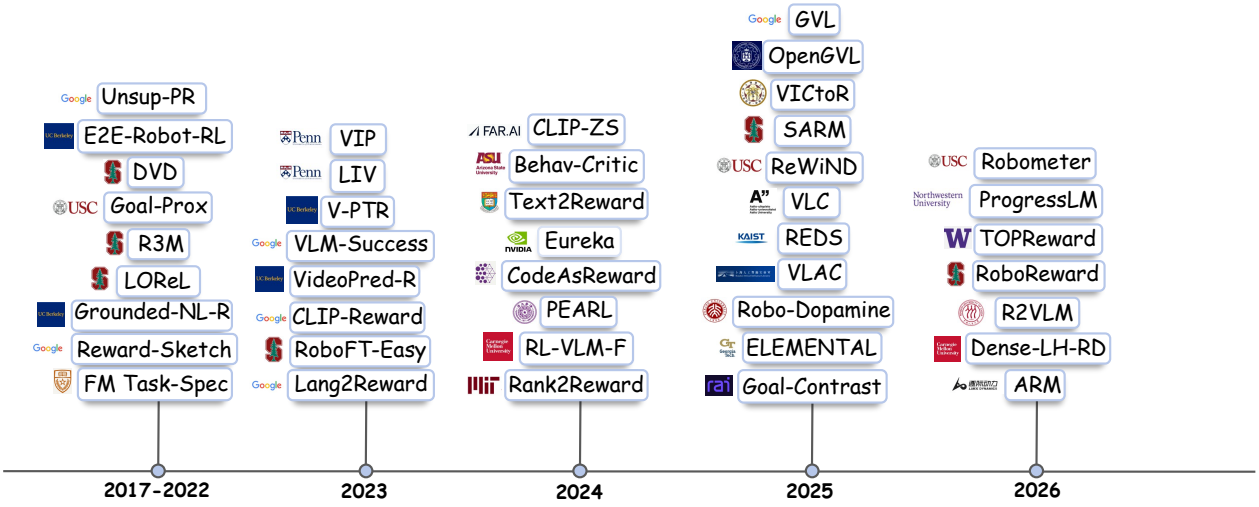

Figure 1 Evolution of progress reward modeling for robotic learning.

However, estimating progress is more difficult than predicting whether a task has succeeded. Success detection is usually a binary decision about whether the final goal has been reached. Progress estimation must instead place the current observation within the evolving execution of a longer task. The model must infer which parts of the task have been completed, what is currently happening, and how far the execution remains from the goal. This mapping is also task-conditioned, because the same observation may represent different stages or different amounts of progress under different tasks. Moreover, the current observation may not contain enough information by itself, since progress can depend on earlier actions, completed subgoals, and whether previous progress has been undone. Progress is therefore a latent and history-dependent task state rather than a directly observable property of an image or frame. A progress model must decide what evidence is relevant and how that evidence should be mapped to task advancement.

Recent years have seen rapid growth in progress reward modeling, as illustrated in Figure 1. Early studies learned goal proximity and reward representations from demonstrations, temporal ordering, and visual goal structure (Lee et al., 2021; Ma et al., 2023b,a; Sermanet et al., 2017). Later work explored foundation-model similarity scores and language-conditioned rewards (Baumli et al., 2023; Rocamonde et al., 2024; Nair et al., 2022), preference-based reward learning (Wang et al., 2024; Liu et al., 2024; Yang et al., 2024), and automatic reward generation with language models (Xie et al., 2024; Yu et al., 2023; Ma et al., 2024; Venuto et al., 2024). More recent systems use large Vision-Language Models to estimate task progress and completion (Zhang et al., 2026a; Liang et al., 2026; Lee et al., 2026; Chen et al., 2026a), compare state transitions and model progress changes (Zhai et al., 2025; Tan et al., 2025; Mao et al., 2026), detect task success (Du et al., 2023), and reason over long executions with explicit temporal memory (Zhang et al., 2026b). This growing body of work shows that progress modeling is becoming an important direction in robotic learning.

Despite this progress, the literature remains fragmented. Methods described under similar terms often solve different problems. One method may predict a success probability from a single image, while another estimates a progress percentage from a video, compares two trajectories, or generates an executable reward program. They also differ in how task goals are specified, how supervision is obtained, and how their outputs are used. Their evaluations are similarly heterogeneous. Temporal ordering, scalar prediction error, preference accuracy, success detection, and downstream policy performance validate different properties. As a result, it is difficult to compare methods directly or determine what their reported results actually demonstrate.

In this survey, we provide **a unified view of progress reward modeling for robotic learning**. We organize this diverse field in three connected steps. First, Section 2 studies the *interface* of progress models. We treat each model as a task-conditioned black box and examine how it represents the current task state, how it specifies the task goal, and what form of progress-related output it produces. This view allows methods with different architectures to be compared through a common input–output structure.

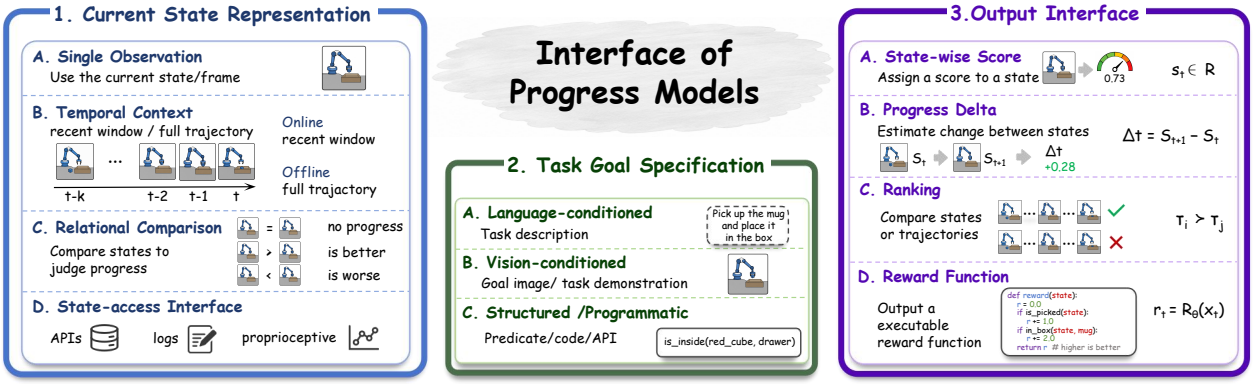

Figure 2 Interface of progress models. We organize existing progress-model interfaces from three perspectives: current task-state representation, task-goal specification, and model output form.

Second, Section 3 opens this black box and examines *how progress rewards are constructed*. We organize existing approaches according to the source of the progress signal and the mechanism that converts it into a usable reward. This perspective reveals the assumptions behind frozen foundation-model scoring, temporal and relative learning, instruction-tuned progress prediction, and programmatic reward construction.

Third, Section 4 connects these methods to their *data and evaluation evidence*. We examine how progress supervision is produced through human annotation, human-guided automation, or fully automated procedures. We then distinguish benchmarks of progress fidelity from evaluations of robustness, generalization, and downstream utility. This distinction is important because a reward may improve a policy without faithfully representing progress, while an accurate progress estimator may still be unsuitable for closed-loop control.

Scope and contributions. This survey focuses on progress and progress-like reward models for robotic learning. Rather than treating all reward models as a single category, we study how they define, construct, supervise, and evaluate task advancement. Our contributions are threefold. First, we introduce an interface-based view that organizes progress models through task-state representation, goal specification, and output form. Second, we provide a taxonomy of the main mechanisms used to construct progress rewards and clarify their underlying assumptions. Third, we connect data-construction pipelines with evaluation protocols and explain what different benchmarks can and cannot validate. We further summarize the main limitations of current methods and discuss future directions in Section 5.

2 Interface: Input and Output of Progress Models

The interface of a progress model determines what evidence the model can access and what kind of progress or reward signal it can provide. As illustrated in Figure 2, we organize the interface around three questions. For the input interface, the key questions are: (i) *how the current task state is represented* (section 2.1) and (ii) *how the task goal is specified* (section 2.2). For the output interface, the key question is: (iii) *what form the model output takes* (section 2.3).

2.1 Input Interface: How Is the Current Task State Represented?

Single-observation inputs are the simplest: the model receives the current observation and a task condition, and directly estimates reward, success, value, or goal proximity. This design assumes that the current state contains enough evidence to judge progress. It appears in task-completion or perceptual reward models (Singh et al., 2019; Lee et al., 2021; Yang et al., 2024; Sermanet et al., 2017), as well as CLIP-based or modern VLM-based reward methods that compare observations with either language or visual goals (Baumli et al., 2023; Rocamonde et al., 2024; Mahmoudieh et al., 2022; Cui et al., 2022; Ma et al., 2023a; Yang et al., 2023; Hung et al., 2025; Zhang et al., 2026a). *The advantage is low latency and easy online use; the limitation is that visually similar states can have different meanings depending on the execution history.*

Temporal context addresses partially solve this ambiguity by feeding the model a sequence of observations. For online progress estimation that the model can directly output while the task is still ongoing, many methods use a trajectory prefix or a short recent window, allowing the model to infer what has already happened before predicting the current progress (Chen et al., 2026a; Mao et al., 2026; Zhang et al., 2025; Alakuijala et al., 2025; Escontrela et al., 2023; Zhang et al., 2026b; Liang et al., 2026; Lee et al., 2026; Chen et al., 2025b; Du et al., 2023; Kim et al., 2025). Other methods use the full trajectory as input after the task execution to score behavior, rank rollouts, generate preference labels, or analyze temporal progress offline (Guan et al., 2024; Chen et al., 2021; Ma et al., 2025; Budzianowski et al., 2025; Liu et al., 2024). The key tradeoff is clear: *longer context gives stronger temporal evidence, but weakens online usability.*

A third interface estimates progress through **comparison**. Instead of judging whether a single state is good in isolation, the model compares two or more reference states, such as before and after an action, initial and current observations, current and goal states, or two candidate rollouts. In this formulation, progress becomes a relational judgment rather than an intrinsic property of one observation. Before-after comparison is naturally suited to estimating progress deltas, dense rewards, and preference labels (Zhai et al., 2025; Nair et al., 2022; Ma et al., 2023b; Wang et al., 2024). Initial-current comparison provides a more global view of how far the execution has moved from the start (Yan et al., 2026; Yong et al., 2026), while goal-conditioned comparison grounds the current state or transition against the desired endpoint (Bhateja et al., 2023; Tan et al., 2025). Comparison-based interfaces are often *more informative and intuitive, but local comparisons can still miss long-horizon dependencies and may suffer from accumulated errors when progress is inferred through a chain of pairwise judgments.*

Finally, some methods use **state-access interfaces**, such as environment code, simulator APIs, training statistics, or proprioceptive features, to construct rewards (Yu et al., 2023; Xie et al., 2024; Ma et al., 2024; Venuto et al., 2024; Chen et al., 2025a; Cabi et al., 2020). These interfaces can be highly effective when the relevant task state is explicitly represented and directly accessible. However, this assumption is often unrealistic in real-world settings, where progress-relevant variables are usually noisy, incomplete, task-dependent, or unavailable through a clean API. Thus, *state-access interfaces are most practical in simulation or instrumented environments, but less suitable for open-ended real-world physical scenes.*

2.2 Input Interface: How Is the Task Goal Specified?

Language-conditioned goals are used in most progress reward models (Liang et al., 2026; Chen et al., 2026a; Zhai et al., 2025; Baumli et al., 2023; Rocamonde et al., 2024; Xie et al., 2024; Lee et al., 2026; Hung et al., 2025; Guan et al., 2024; Nair et al., 2022; Mahmoudieh et al., 2022; Mao et al., 2026; Yu et al., 2023; Ma et al., 2023a; Budzianowski et al., 2025; Nair et al., 2023; Zhang et al., 2025; Chen et al., 2025b; Du et al., 2023; Wang et al., 2024; Alakuijala et al., 2025; Yang et al., 2023; Kim et al., 2025). The main reason is that language goals are *easy to encode*, compatible with existing language or vision-language models, and often *effective for relatively simple tasks.* However, language can *underspecify the physical details of progress, especially as tasks become longer, more compositional, and more interaction-dependent.*

Vision-conditioned goals provide a more direct specification of the desired execution process. A goal image, goal state, or demonstration trajectory can capture physical details that are difficult to describe with language alone. Goal-image and visual-reference methods compare the current observation against a desired visual endpoint (Singh et al., 2019; Ma et al., 2023b; Yang et al., 2024; Bhateja et al., 2023; Tan et al., 2025; Cui et al., 2022; Venuto et al., 2024), while demonstration-conditioned methods use a full trajectory to specify not only the final state but also the intended procedure (Chen et al., 2025a, 2021; Lee et al., 2021; Liu et al., 2024; Sermanet et al., 2017; Zhang et al., 2026a). Visual goals therefore *reduce linguistic ambiguity and provide richer physical and dynamic grounding.* However, they typically *require manually curated goal images or demonstrations, and can be sensitive to visual variations* such as viewpoint changes, embodiment differences, object appearance, and scene layout.

A third form of goal specification is **structured or programmatic goals**. In simulation or API-rich environments, the task goal can be expressed through predicates, object states, constraints, or generated objective functions. This interface supports precise, dense, and compositional reward construction (Xie et al., 2024; Ma et al., 2024; Yu et al., 2023; Venuto et al., 2024; Yong et al., 2026). However, it relies on access to explicitly represented state variables, which is often unavailable in open-ended real-world settings. Thus, structured or programmatic

goals are most effective in simulation or instrumented environments, but less suitable when progress must be inferred from raw visual observations.

2.3 Output Interface: What Does the Model Predict?

The output interface determines **how a model prediction is interpreted as progress**. Different progress models may output scalar scores, transition-level deltas, rankings or preferences, or executable reward functions. Although these outputs can all be used as progress signals, they encode progress in different ways: some estimate how desirable a state is, some measure whether a transition moves the task forward, some define progress only through comparison, and some specify a reward computation procedure.

The most common output is a **state-wise scalar score**. Such a score maps a state or clip to a single number, which is then interpreted as task progress, success likelihood, value, goal similarity, or distance-to-goal. For explicit progress estimation, the scalar directly represents how far the task has advanced (Liang et al., 2026; Tan et al., 2025; Hung et al., 2025; Chen et al., 2025b; Kim et al., 2025; Zhang et al., 2026a). For success or completion models, the scalar is interpreted as progress indirectly: higher completion likelihood implies that the current state is closer to task success (Chen et al., 2026a; Baumli et al., 2023; Du et al., 2023; Escontrela et al., 2023; Yang et al., 2023; Lee et al., 2026; Budzianowski et al., 2025). Similarly, value, similarity, or distance-based methods treat progress as increasing value, increasing similarity to the goal, or decreasing distance from the goal (Rocamonde et al., 2024; Lee et al., 2021; Ma et al., 2025, 2023b; Zhang et al., 2025; Alakuijala et al., 2025; Bhateja et al., 2023; Sermanet et al., 2017; Ma et al., 2023a). Thus, state-wise scores provide *the most direct and reusable output form, but their meaning depends on how the scalar is calibrated and what semantic quantity it is intended to approximate.*

A second output form is **progress delta**. Instead of estimating the absolute progress of a state, the model predicts whether a transition improves or worsens the task state. In this case, progress is represented as a local change between two observations, such as before and after an action. VLAC directly predicts signed relative progress from before-after observations (Zhai et al., 2025), ARM models local advantage-like progress for long-horizon manipulation (Mao et al., 2026), and Robo-Dopamine learns hop values between state pairs (Tan et al., 2025). Delta outputs are *naturally aligned with reinforcement learning.* However, because they describe local improvement rather than global task completion, they usually *need to be accumulated or combined over time to recover total progress.*

A third output form is **ranking**. Here progress is not represented by an absolute score, but by an ordering over states, clips, or trajectories. A state or trajectory is considered more advanced if it is preferred over another reference (Cui et al., 2022; Liu et al., 2024; Wang et al., 2024; Yang et al., 2024; Cabi et al., 2020). This output is often *easier to supervise than calibrated reward values* because models can usually compare alternatives more reliably than assign absolute scores. However, preference outputs *define progress only relatively and in an indirect way*, and therefore must usually be converted into a scalar reward for later use.

Finally, programmatic methods output **reward functions** rather than values. Here, progress is encoded as an executable computation over available state variables, features, or observations (Xie et al., 2024; Yu et al., 2023; Ma et al., 2024; Venuto et al., 2024; Chen et al., 2025a). This output can be *flexible and interpretable* because it explicitly specifies how progress should be computed. However, its reliability depends on whether the generated code, available state variables, and engineered features truly capture the intended task objective.

3 Methods: How Are Progress Rewards Constructed?

This section focuses on how progress rewards are constructed. As illustrated in Figure 3, we organize existing methods by the mechanism through which progress signals are obtained and converted into usable rewards. We identify four main paradigms: frozen foundation-model scoring, learning from temporal or relative supervision, instruction-tuned progress prediction, and programmatic reward construction.

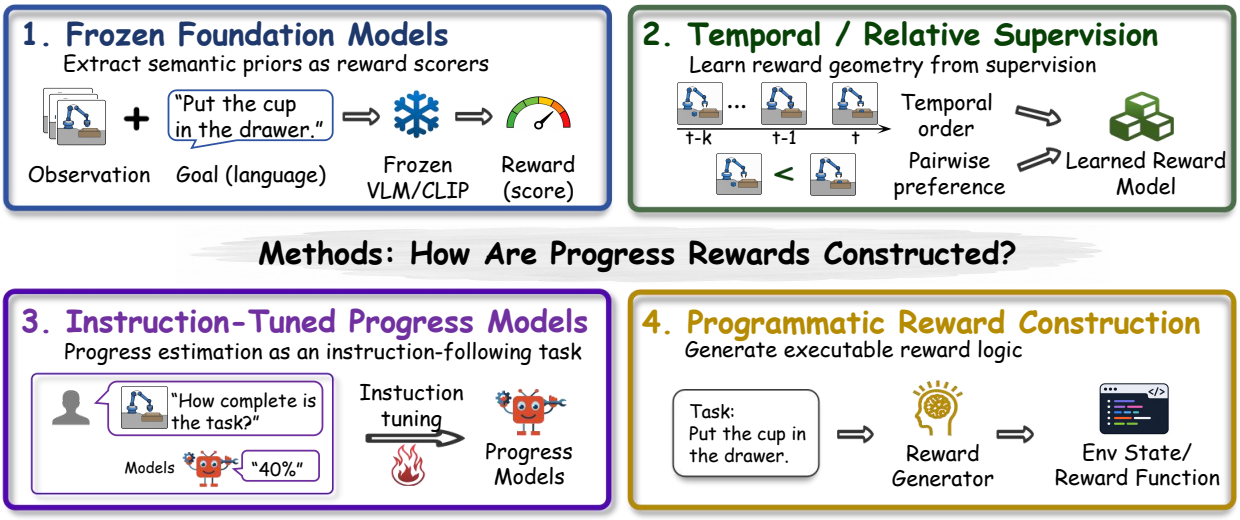

Figure 3 Construction of progress rewards. We organize existing methods into four paradigms according to where the progress signal comes from and how it is converted into a usable reward.

3.1 Frozen Foundation Models as Semantic Reward Scorers

The simplest paradigm does not train a reward model at all. Instead, it directly extracts reward-like signals from pretrained foundation models. CLIP-style methods compute image-text similarity between the current observation and the language goal, and use the resulting alignment score as reward (Baumli et al., 2023; Rocamonde et al., 2024; Mahmoudieh et al., 2022). TOPReward moves this idea from embedding space to token space. It prompts a frozen VLM with a task-completion question and uses the probability of the "true" token as a hidden reward (Chen et al., 2026a). GVL and OpenGVL further use VLMs to estimate temporal progress or frame ordering from videos (Ma et al., 2025; Budzianowski et al., 2025).

The main strength of this paradigm is **zero-shot** usability. No task-specific reward labels or fine-tuning are required. However, the resulting score is better understood as a semantic prior than as a calibrated progress reward. Therefore, these methods often *need normalization, thresholding, differencing, or prompt engineering before their outputs can be used as rewards.*

3.2 Learning Rewards from Temporal and Relative Supervision

A second group of methods learns rewards from supervision to determine which state is closer to the goal or which behavior shows more progress. One common source of supervision is the **temporal order within successful demonstrations**. These methods often assume that later states are closer to task completion than earlier states. Goal-proximity imitation learning uses demonstration time as a proxy for progress and converts changes in goal proximity into rewards (Lee et al., 2021). VIP learns a visual representation in which the distance between the current state and the goal reflects goal reachability, allowing this distance to be used as a reward (Ma et al., 2023b). Other methods construct rewards from perceptual distance or detected subtask structure (Sermanet et al., 2017; Kim et al., 2025). A related form of supervision is reward sketching, where users provide coarse reward curves instead of labeling every timestep (Cabi et al., 2020).

Another source of supervision is **preference or ranking information**. Instead of assigning an absolute progress value, these methods compare two states, clips, or trajectories and determine which one is better. RL-VLM-F uses a frozen VLM to generate pairwise preferences and then trains a scalar reward model from these comparisons (Wang et al., 2024). PEARL transfers preference information across different tasks (Liu et al., 2024), while Rank2Reward learns shaped reward functions from rankings of passive videos (Yang et al., 2024). Relative supervision is usually easier to collect than accurate progress scores, but it does not directly provide a usable scalar reward. The learned comparisons must still be converted into a reward function, and the result may depend on whether the observed preferences can be represented by a single consistent score.

3.3 Instruction-Tuned Progress Prediction

A third paradigm **formulates progress estimation as an explicit instruction-following task for foundation models**. Rather than relying only on frozen semantic priors or weak temporal structure, these methods specify the desired progress judgment through task-specific prompts and then fine-tune VLMs or video-language models to follow this instruction.

Depending on the formulation, the model may be asked to predict absolute progress, task success, relative improvement, preferences, or reasoning-grounded progress scores. Training examples therefore pair visual observations or trajectories with explicit progress-related instructions and target responses, enabling the model to acquire progress estimation as a dedicated capability. RoboReward fine-tunes VLMs with rubric-based prompts to produce structured progress scores over robot rollouts (Lee et al., 2026). Robometer jointly trains progress, success, and preference prediction, allowing a shared model to answer multiple progress-related instructions from dense labels and trajectory comparisons (Liang et al., 2026). Robo-Dopamine estimates hop-based progress from initial, goal, before, and after observations (Tan et al., 2025), while VLAC is trained to follow before-after comparison prompts and predict signed progress changes (Zhai et al., 2025). Video-Language Critic learns to map successful and failed video clips to transferable scalar reward judgments (Alakuijala et al., 2025). ProgressLM further trains VLMs to generate explicit progress reasoning before predicting progress, using instruction tuning to ground the final score in procedural understanding (Zhang et al., 2026a).

The key idea is that progress prediction is not treated as an implicit by-product of visual representation learning. Instead, it is explicitly defined in the prompt, supervised in the training targets, and learned as an instruction-following capability of the model.

3.4 Programmatic Reward Construction

Instead of directly estimating a reward value from observations, some methods represent rewards as executable programs. They translate progress estimation tasks into code, predicates, feature functions, or structured reward logic that can be evaluated during policy learning. Text2Reward and Language2Rewards convert natural-language task descriptions into executable reward functions (Xie et al., 2024; Yu et al., 2023). Eureka further refines LLM-generated reward programs using feedback from policy training (Ma et al., 2024). Code as Reward uses VLM reasoning to synthesize executable reward logic from visual task descriptions (Venuto et al., 2024), while ELEMENTAL combines VLM-generated feature functions with inverse reinforcement learning to construct task rewards (Chen et al., 2025a). These methods are particularly effective when a task can be decomposed into explicit subgoals, predicates, geometric relations, or environment states exposed through APIs. Their flexibility also makes the resulting reward functions relatively interpretable and easy to modify. However, they shift a substantial part of the challenge from reward prediction to task decomposition, feature design, and reliable state access. When the available symbolic variables or APIs fail to capture important aspects of the task, the generated reward may be precise with respect to its implementation yet still misaligned with the intended behavior.

4 Data and Benchmarks

This section examines how progress supervision is constructed and how progress models are evaluated. As illustrated in Figure 4, we first organize progress-data construction by the degree of human involvement. We then group existing benchmarks according to three evaluation goals: progress fidelity, robustness and generalization, and downstream utility. This organization connects how progress supervision is obtained with what different evaluation results can actually validate.

4.1 Data: How Is Progress Supervision Constructed?

Progress models require trajectories together with supervision describing task completion, progress change, or relative behavior quality. Existing works obtain such supervision through direct human judgments, combinations of human input and automated processing, or fully automatic rules and models. To organize these practices, **we characterize progress-data construction by the degree of human involvement**. This perspective

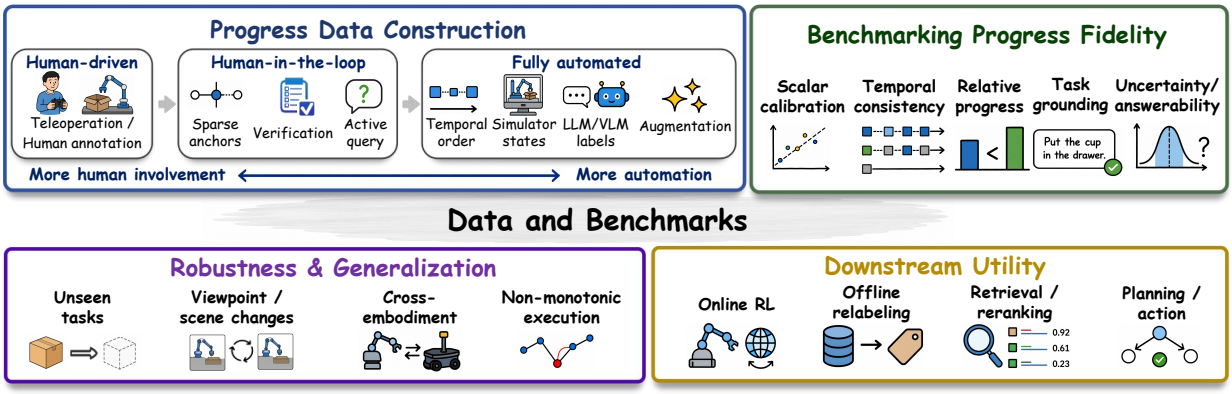

Figure 4 Data and benchmarks for progress modeling. We organize progress-data construction by the degree of human involvement and group benchmarks into progress fidelity, robustness and generalization, and downstream utility.

reveals a central trade-off: *human supervision provides stronger semantic grounding, whereas automation enables greater scale.*

Human-driven construction. Human-driven pipelines rely on people to provide task executions or define progress directly. Human operators collect robot demonstrations through teleoperation or record human task videos, thereby providing successful or imperfect executions grounded in real behavior (Chen et al., 2026a; Tan et al., 2025; Zhai et al., 2025; Ma et al., 2025, 2023a; Zhang et al., 2025). Annotators may assign scalar progress scores (Lee et al., 2026; Rocamonde et al., 2024), identify success cutoffs (Liang et al., 2026; Du et al., 2023), or mark keyframes and subtask boundaries (Tan et al., 2025; Chen et al., 2025b; Kim et al., 2025). Other forms of human supervision include drawing trajectory-level reward sketches (Cabi et al., 2020) and comparing executions or annotating undesirable behavior (Guan et al., 2024; Liu et al., 2024).

The main advantage is that humans can capture task semantics that are difficult to infer from time or state variables alone, such as grasp stability, acceptable contact, or implicit behavioral preferences. However, *dense manual annotation is costly, subjective, and difficult to scale*, especially for long-horizon tasks with multiple valid strategies.

Human-in-the-loop construction. Human-in-the-loop pipelines use sparse human input to guide, constrain, or verify automated processing. A common strategy is to annotate only success frames, keyframes, or stage boundaries and then interpolate dense progress labels between these semantic anchors (Tan et al., 2025; Chen et al., 2025b; Du et al., 2023; Zhang et al., 2026a). Models may instead propose stage labels or progress annotations that are subsequently checked and corrected by humans (Kim et al., 2025; Hung et al., 2025). Active-learning methods further reduce cost by requesting human labels only for uncertain or informative states (Singh et al., 2019; Liu et al., 2024).

Human involvement may also occur through the design of prompts, task vocabularies, templates, environment abstractions, or reward APIs, after which LLMs or scripts generate annotations and reward specifications at scale (Yu et al., 2023; Xie et al., 2024; Venuto et al., 2024). Human inspection or rollout feedback can then be used to revise incorrectly generated labels or rewards (Ma et al., 2024; Chen et al., 2025a). Thus, *humans preserve semantic control while automation provides density and scale*, although sparse verification may still fail to detect systematic labeling errors.

Fully automated construction. Fully automated pipelines derive progress supervision without instance-level human annotation. The most common approach uses temporal structure: later frames in successful demonstrations are treated as closer to completion, producing normalized progress labels, goal-proximity targets, or pairwise rankings (Lee et al., 2021; Ma et al., 2023b, 2025; Budzianowski et al., 2025; Yang et al., 2024; Bhateja et al., 2023). Before–after pairs can similarly be labeled according to their temporal distance or relative progress change (Zhai et al., 2025; Tan et al., 2025). Such supervision is highly scalable, but *it assumes that task progress is sufficiently correlated with time.*

Structured environments provide another automatic source of labels. Simulator states, object poses, geometric predicates, and ground-truth rewards can generate success labels, stage indices, continuous progress, or trajectory preferences (Baumli et al., 2023; Cui et al., 2022; Mahmoudieh et al., 2022; Ma et al., 2023a). Expert policies and motion planners can generate successful trajectories (Escontrela et al., 2023), while noisy or randomized policies produce failed executions (Alakuijala et al., 2025; Chen et al., 2021). These labels are consistent and inexpensive, but *they only capture properties represented by the available states and predicates.*

Foundation models can also serve as automatic annotators. LLMs and VLMs generate subgoal decompositions, stage boundaries, language variants, progress scores, and pairwise preferences (Hung et al., 2025; Mao et al., 2026; Yong et al., 2026; Wang et al., 2024). However, *model-generated supervision inherits the biases and errors of the annotating model.*

Automation is also used to broaden behavior coverage. Successful trajectories can be truncated, reversed, paired with mismatched instructions, or perturbed with action noise to create partial, regressing, and failed examples (Zhang et al., 2025; Zhai et al., 2025; Lee et al., 2026; Liang et al., 2026; Yang et al., 2023). Such augmentation reduces success-only bias, but synthetic failures must remain physically plausible to avoid introducing exploitable artifacts.

4.2 Benchmarking Progress Fidelity

The most direct benchmark question is whether a model output faithfully represents task progress. However, **progress fidelity is not a single property**: different reference signals test calibration, temporal consistency, relative ordering, task grounding, or uncertainty.

Scalar calibration evaluates whether predicted scores agree with explicit progress references, such as completion percentages, rubric scores, stage labels, or hop-based progress values (Lee et al., 2026; Zhang et al., 2026a; Chen et al., 2025b; Mao et al., 2026; Tan et al., 2025). Typical metrics include regression error and correlation, which can provide the most direct evidence that a model estimates absolute progress.

Temporal consistency evaluates whether predicted scores recover the progression of a trajectory. Ordering accuracy, rank correlation, and monotonicity-based metrics test whether later states in successful demonstrations tend to receive higher scores than earlier states (Ma et al., 2025; Budzianowski et al., 2025; Chen et al., 2026a; Zhang et al., 2025; Zhai et al., 2025; Tan et al., 2025). This evaluation primarily measures chronological consistency rather than absolute calibration.

Relative progress benchmarks evaluate whether the model correctly orders states, clips, or trajectories by task advancement or quality (Liang et al., 2026; Yang et al., 2024; Wang et al., 2024). They are particularly suitable for preference learning and trajectory selection because pairwise judgments are often easier to obtain than calibrated scalar labels.

Task grounding benchmarks test whether the predicted progress is conditioned on the specified goal rather than on generic motion, scene change, or temporal position. Counterfactual instructions, mismatched video–language pairs, and contrastive task prompts are commonly used to verify that the same observation receives different judgments under different goals (Zhang et al., 2025; Zhai et al., 2025; Ma et al., 2025). This distinction is essential because a smooth and temporally plausible reward curve may still be incorrect if the model ignores the task instruction.

Finally, **answerability and uncertainty** benchmarks evaluate whether the model can recognize when progress cannot be reliably inferred (Zhang et al., 2026a,b). Progress may be ambiguous because of occlusion, missing history, insufficient viewpoints, or underspecified goals. A model that always returns a confident score can therefore be unsafe even when its average prediction error is low. Benchmarking selective prediction, abstention, or confidence calibration is necessary to distinguish reliable progress estimation from forced guessing.

4.3 Benchmarking Robustness and Generalization

Beyond matching reference labels, a progress model should preserve its meaning under changes in tasks, observations, embodiments, and execution behavior. **Robustness benchmarks test whether the learned notion of progress reflects transferable task structure rather than dataset-specific shortcuts.**

Task generalization evaluates performance on unseen tasks, instructions, objects, or task families (Zhang et al., 2025, 2026a; Liu et al., 2024; Alakuijala et al., 2025; Zhang et al., 2026a). **Viewpoint and scene robustness** evaluate whether progress estimates remain stable under camera changes, occlusion, distractors, background variation, and differences between demonstration and deployment views (Zhang et al., 2026a; Du et al., 2023). Cross-view evaluation tests whether the model captures task-relevant state rather than superficial visual similarity matching. **Cross-embodiment generalization** evaluates whether progress knowledge learned from humans, passive videos, or one robot platform transfers to another embodiment (Ma et al., 2025; Alakuijala et al., 2025; Ma et al., 2023b,a; Nair et al., 2023; Bhateja et al., 2023; Chen et al., 2021; Sermanet et al., 2017). Such transfer is attractive because human and Internet videos are far more abundant than robot demonstrations. However, it is challenging because embodiments differ in kinematics, contacts, feasible actions, viewpoints, and execution styles. The most demanding robustness setting is **non-monotonic and imperfect execution**. Real policies may fail, regress, retry an action, revisit an earlier state, or complete subtasks in an unusual order (Liang et al., 2026; Lee et al., 2026; Zhai et al., 2025; Zhang et al., 2025; Mao et al., 2026; Chen et al., 2025b; Hung et al., 2025; Tan et al., 2025; Yong et al., 2026). Benchmarks based only on successful and monotonic demonstrations cannot reveal whether a model distinguishes genuine progress from elapsed time. Evaluating failures, reversals, plateaus, and recovery behavior is therefore necessary for deployment-relevant progress estimation.

4.4 Benchmarking Downstream Utility

Progress models are ultimately intended to support robot learning and decision making. Accordingly, many benchmarks evaluate whether their outputs improve policy learning, trajectory selection, data curation, or planning. A reward may be useful without being calibrated, while a well-calibrated estimator may still be too sparse, noisy, or expensive for control.

Online policy learning evaluates whether a progress reward is dense, stable, and sufficiently aligned to support interactive reinforcement learning (Lee et al., 2026; Liang et al., 2026; Zhang et al., 2025; Zhai et al., 2025; Yang et al., 2024; Wang et al., 2024; Xie et al., 2024; Ma et al., 2024; Venuto et al., 2024). Success rate, sample efficiency, and learning stability are common metrics. However, these results are influenced by exploration, policy architecture, reset design, and environment dynamics. *Improved policy performance does not by itself demonstrate that the reward is a faithful progress estimator.*

Offline learning and reward relabeling evaluate whether a model can assign useful signals to fixed datasets containing successful, partial, failed, or heterogeneous trajectories (Liang et al., 2026; Zhang et al., 2025; Bhateja et al., 2023; Cabi et al., 2020). This setting tests whether the reward remains meaningful on behavior generated by different policies and whether it can support offline reinforcement learning, data weighting, or trajectory filtering. It is especially sensitive to reward calibration outside the distribution of high-quality demonstrations.

Retrieval, filtering, and reranking benchmarks use progress scores to identify useful demonstrations, detect failures, rank candidate rollouts, or select the best result from multiple attempts (Ma et al., 2025; Budzianowski et al., 2025; Liang et al., 2026; Du et al., 2023; Lee et al., 2026; Tan et al., 2025; Alakuijala et al., 2025; Chen et al., 2025a; Guan et al., 2024). These evaluations primarily test comparative judgment. They may therefore succeed even when the score lacks an absolute interpretation, and should not be treated as evidence of calibrated progress.

Planning and action selection impose a stronger requirement: the reward landscape must be smooth, informative, and query-efficient enough to evaluate hypothetical states or candidate actions (Ma et al., 2023a,b; Chen et al., 2021; Nair et al., 2022; Yu et al., 2023; Xie et al., 2024; Venuto et al., 2024). A reward that correctly recognizes final success may still be ineffective for planning if it is sparse, discontinuous, visually brittle, or computationally expensive. Planning benchmarks therefore test whether the progress signal provides useful local guidance, not merely whether it recognizes completed tasks.

5 Limitations and Future Directions

Despite recent progress, current progress models still face several practical limitations. In particular, they often lack fine-grained sensitivity, adaptive temporal modeling, efficient online inference, and reliable long-horizon memory. Addressing these limitations is necessary before progress rewards can be widely used in real-time robotic learning.

First, current progress estimation remains too coarse-grained. Many models use sparsely sampled frames, short clips, discrete progress bins, or broad task stages (Liang et al., 2026; Chen et al., 2026a; Tan et al., 2025; Chen et al., 2025b). Such representations can capture major task transitions, but may miss small yet important changes, such as precise alignment, slight object motion, unstable grasping, or early contact failure. Future models should provide *finer spatial and temporal resolution* and combine visual evidence with proprioception, force, tactile sensing, or other signals that reveal subtle physical progress.

Second, progress should not be assumed to change at a fixed rate. Many existing methods use temporal order or normalized timestamps as supervision, which encourages progress to increase smoothly with time (Lee et al., 2021; Ma et al., 2023b, 2025; Zhang et al., 2025; Zhai et al., 2025). However, real task progress is often uneven. A major subgoal may cause a sudden increase, while waiting, searching, or careful alignment may produce a long plateau. The model must also distinguish slow but meaningful execution from behavior that makes no progress. Future work should develop *temporally adaptive progress models* that adjust the magnitude of progress changes according to task events, execution speed, and stage difficulty, while also representing stagnation and regression.

Third, inference latency limits online deployment. Large Vision-Language Models often need to process multiple frames, interpret the task instruction, and reason about the current execution before producing a reward (Alakuijala et al., 2025; Liang et al., 2026; Zhang et al., 2026b; Yan et al., 2026). This cost may be acceptable for offline evaluation, but it is difficult to use at every control step. Future systems should explore streaming visual encoders, cached task representations, model distillation, and hierarchical reward querying. A lightweight model could handle frequent local updates, while a larger model is called only at important task transitions or uncertain states.

Finally, long-horizon progress requires stronger and more explicit memory. In many tasks, visually similar observations may correspond to different stages or different amounts of progress because their meanings depend on the preceding execution history. The current observation alone may not reveal which subgoals have already been completed, in what order they occurred, or whether earlier progress has been preserved or undone. Repeated actions provide a simple example: when a robot must pick and place several objects, the first and second picking actions may look nearly identical, although they represent different completion counts. A memoryless model may therefore assign similar progress values to states that occupy different positions in the overall task. Short observation windows provide only limited support when the relevant events lie outside the window (Zhai et al., 2025; Liang et al., 2026; Mao et al., 2026; Chen et al., 2025b). Future progress models should maintain compact task memory that records completed, pending, and invalidated subgoals, as well as other history-dependent information required to locate the current observation within the full task execution.

References

Minttu Alakuijala, Reginald McLean, Isaac Woungang, Nariman Farsad, Samuel Kaski, Pekka Marttinen, and Kai Yuan. Video-language critic: Transferable reward functions for language-conditioned robotics, 2025. https://arxiv.org/abs/2405.19988.

Kate Baumli, Satinder Baveja, Feryal Behbahani, Harris Chan, Gheorghe Comanici, Sebastian Flennerhag, Maxime Gazeau, Kristian Holsheimer, Dan Horgan, Michael Laskin, Clare Lyle, Hussain Masoom, Kay McKinney, Volodymyr Mnih, Alexander Neitz, Dmitry Nikulin, Fabio Pardo, Jack Parker-Holder, John Quan, Tim Rocktäschel, Himanshu Sahni, Tom Schaul, Yannick Schroecker, Stephen Spencer, Richie Steigerwald, Luyu Wang, and Lei Zhang. Vision-language models as a source of rewards. *arXiv preprint arXiv:2312.09187*, 2023.

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, 2023. https://arxiv.org/abs/2309.13041.

Paweł Budzianowski, Emilia Wiśnios, Michał Tyrolski, Gracjan Góral, Igor Kulakov, Viktor Petrenko, and Krzysztof Walas. Opengvl – benchmarking visual temporal progress for data curation. *arXiv preprint arXiv:2509.17321*, 2025.

Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, Oleg Sushkov, David Barker, Jonathan Scholz, Misha Denil, Nando de Freitas, and Ziyu Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning, 2020. https://arxiv.org/abs/1909.12200.

Annie S. Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos, 2021. https://arxiv.org/abs/2103.16817.

Letian Chen, Nina Marie Moorman, and Matthew Craig Gombolay. ELEMENTAL: Interactive learning from demonstrations and vision-language models for reward design in robotics. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 8700–8725. PMLR, 2025a. https://proceedings.mlr.press/v267/chen25at.html.

Qianzhong Chen, Justin Yu, Mac Schwager, Pieter Abbeel, Yide Shentu, and Philipp Wu. Sarm: Stage-aware reward modeling for long horizon robot manipulation, 2025b. https://arxiv.org/abs/2509.25358.

Shirui Chen, Cole Harrison, Ying-Chun Lee, Angela Jin Yang, Zhongzheng Ren, Lillian J. Ratliff, Jiafei Duan, Dieter Fox, and Ranjay Krishna. Topreward: Token probabilities as hidden zero-shot rewards for robotics, 2026a. https://arxiv.org/abs/2602.19313.

Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Weijie Wan, Baijun Chen, Haoran Lu, Haowen Yan, Honghao Su, et al. Robodojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026b.

Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, Hongcheng Wang, Renjing Xu, Ruihai Wu, Yao Mu, Yaodong Yang, Hao Dong, and Ping Luo. Rmbench: Memory-dependent robotic manipulation benchmark with insights into policy design, 2026c. https://arxiv.org/abs/2603.01229.

Yuchen Cui, Scott Niekum, Abhinav Gupta, Vikash Kumar, and Aravind Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, volume 168 of *Proceedings of Machine Learning Research*, pages 893–905. PMLR, 2022. https://proceedings.mlr.press/v168/cui22a.html.

Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. In Sarath Chandar, Razvan Pascanu, Hanie Sedghi, and Doina Precup, editors, *Proceedings of The 2nd Conference on Lifelong Learning Agents*, volume 232 of *Proceedings of Machine Learning Research*, pages 120–136. PMLR, 2023. https://proceedings.mlr.press/v232/du23b.html.

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. In *Advances in Neural Information Processing Systems*, volume 36, pages 68760–68783. Curran Associates, Inc., 2023. https://proceedings.neurips.cc/paper_files/paper/2023/file/d9042abf40782fbce28901c1c9c0e8d8-Paper-Conference.pdf.

Generalist AI Team. Gen-0: Embodied foundation models that scale with physical interaction. Generalist AI Blog, 2025. November 4, 2025.

Lin Guan, Yifan Zhou, Denis Liu, Yantian Zha, Heni Ben Amor, and Subbarao Kambhampati. Task Success is not enough: Investigating the use of video-language models as behavior critics for catching undesirable agent behaviors. In *First Conference on Language Modeling*, 2024. https://openreview.net/forum?id=otKo4zFKmH.

Kuo-Han Hung, Pang-Chi Lo, Jia-Fong Yeh, Han-Yuan Hsu, Yi-Ting Chen, and Winston H. Hsu. Victor: Learning hierarchical vision-instruction correlation rewards for long-horizon manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025. https://openreview.net/forum?id=UpQLu9bzAR.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Rich Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. π0.5: a vision-language-action model with open-world generalization. *ArXiv*, abs/2504.16054, 2025. https://api.semanticscholar.org/CorpusID:277993634.

Changyeon Kim, Minho Heo, Doohyun Lee, Honglak Lee, Jinwoo Shin, Joseph J. Lim, and Kimin Lee. Subtask-aware visual reward learning from segmented demonstrations. In *The Thirteenth International Conference on Learning Representations*, 2025. https://openreview.net/forum?id=mqKVe6F3Up.

Tony Lee, Andrew Wagenmaker, Karl Pertsch, Percy Liang, Sergey Levine, and Chelsea Finn. Roboreward: General-purpose vision-language reward models for robotics, 2026. https://arxiv.org/abs/2601.00675.

Youngwoon Lee, Andrew Szot, Shao-Hua Sun, and Joseph J Lim. Generalizable imitation learning from observation via inferring goal proximity. In *Advances in Neural Information Processing Systems*, volume 34, pages 16118–16130. Curran Associates, Inc., 2021. https://proceedings.neurips.cc/paper_files/paper/2021/file/868b7df964b1af24c8c0a9e43a330c6a-Paper.pdf.

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S. Huang, Luke Zettlemoyer, Dieter Fox, Yu Xiang, Anqi Li, Andreea Bobu, Abhishek Gupta, Stephen Tu, Erdem Biyik, and Jesse Zhang. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons, 2026. https://arxiv.org/abs/2603.02115.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning, 2023. https://arxiv.org/abs/2306.03310.

Runze Liu, Yali Du, Fengshuo Bai, Jiafei Lyu, and Xiu Li. PEARL: Zero-shot cross-task preference alignment and robust reward learning for robotic manipulation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 30946–30964. PMLR, 2024. https://proceedings.mlr.press/v235/liu24o.html.

Haoran Lu, Ruihai Wu, Yitong Li, Sijie Li, Ziyu Zhu, Chuanruo Ning, Yan Shen, Longzan Luo, Yuanpei Chen, and Hao Dong. Garmentlab: A unified simulation and benchmark for garment manipulation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 11866–11903. Curran Associates, Inc., 2024. doi: 10.52202/079017-0379. https://proceedings.neurips.cc/paper_files/paper/2024/file/15f80ec0fed53885d2ca6272edb96ede-Paper-Conference.pdf.

Haoran Lu, Songling Liu, Yue Chen, Guo Ye, Mutian Shen, Shuyang Yu, Yu Xiao, Jihai Zhao, Shang Wu, Jianshu Zhang, Xiangtian Gui, Chuye Hong, Yuran Wang, Maojiang Su, Jiayi Wang, Ruihai Wu, Zhaoran Wang, and Han Liu. Magicsim: A unified infrastructure for executable embodied interaction, 2026. https://arxiv.org/abs/2606.17511.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-image representations and rewards for robotic control. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 23301–23320. PMLR, 2023a. https://proceedings.mlr.press/v202/ma23b.html.

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, 2023b. https://arxiv.org/abs/2210.00030.

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. In *The Twelfth International Conference on Learning Representations*, 2024. https://openreview.net/forum?id=IEduRUO55F.

Yecheng Jason Ma, Joey Hejna, Ayzaan Wahid, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Jonathan Tompson, Osbert Bastani, Dinesh Jayaraman, Wenhao Yu, Tingnan Zhang,

Dorsa Sadigh, and Fei Xia. Vision language models are in-context value learners. In *The Thirteenth International Conference on Learning Representations*, 2025. https://openreview.net/forum?id=friHAl5ofG.

Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14743–14752. PMLR, 17–23 Jul 2022. https://proceedings.mlr.press/v162/mahmoudieh22a.html.

Yiming Mao, Zixi Yu, Weixin Mao, Yinhao Li, Qirui Hu, Zihan Lan, Minzhao Zhu, and Hua Chen. Arm: Advantage reward modeling for long-horizon manipulation, 2026. https://arxiv.org/abs/2604.03037.

Suraj Nair, Eric Mitchell, Kevin Chen, brian ichter, Silvio Savarese, and Chelsea Finn. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1303–1315. PMLR, 2022. https://proceedings.mlr.press/v164/nair22a.html.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 892–909. PMLR, 2023. https://proceedings.mlr.press/v205/nair23a.html.

NVIDIA. Gr00t n1: An open foundation model for generalist humanoid robots. arXiv:2503.14734, 2025.

Physical Intelligence. Pi-0.7: A steerable generalist robotic foundation model with emergent capabilities. arXiv preprint, 2026. CorpusID: 287607456.

Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. https://openreview.net/forum?id=N0I2RtD8je.

Pierre Sermanet, Kelvin Xu, and Sergey Levine. Unsupervised perceptual rewards for imitation learning, 2017. https://arxiv.org/abs/1612.06699.

Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering, 2019. https://arxiv.org/abs/1904.07854.

Huajie Tan, Sixiang Chen, Yijie Xu, Zixiao Wang, Yuheng Ji, Cheng Chi, Yaoxu Lyu, Zhongxia Zhao, Xiansheng Chen, Peterson Co, Shaoxuan Xie, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. Robo-dopamine: General process reward modeling for high-precision robotic manipulation. *arXiv preprint arXiv:2512.23703*, 2025.

David Venuto, Mohammad Sami Nur Islam, Martin Klissarov, Doina Precup, Sherry Yang, and Ankit Anand. Code as reward: Empowering reinforcement learning with vlms. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

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 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 51484–51501, 2024. https://proceedings.mlr.press/v235/wang24bn.html.

Ruihai Wu, Haozhe Chen, Mingtong Zhang, Haoran Lu, Yitong Li, and Yunzhu Li. Neural dynamics augmented diffusion policy. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13234–13241, 2025. doi: 10.1109/ICRA55743.2025.11128651.

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. In *The Twelfth International Conference on Learning Representations*, 2024. https://openreview.net/forum?id=tUM39YTRxH.

Hongyu Yan, Qiwei Li, Jiaolong Yang, and Yadong Mu. Progressvla: Progress-guided diffusion policy for vision-language robotic manipulation, 2026. https://arxiv.org/abs/2603.27670.

Daniel Yang, Davin Tjia, Jacob Berg, Dima Damen, Pulkit Agrawal, and Abhishek Gupta. Rank2reward: Learning shaped reward functions from passive video, 2024. https://arxiv.org/abs/2404.14735.

Jingyun Yang, Max Sobol Mark, Brandon Vu, Archit Sharma, Jeannette Bohg, and Chelsea Finn. Robot fine-tuning made easy: Pre-training rewards and policies for autonomous real-world reinforcement learning, 2023. https://arxiv.org/abs/2310.15145.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, and Joel Jang. World action models are zero-shot policies. arXiv:2602.15922, 2026.

Silong Yong, Stephen Sheng, Carl Qi, Xiaojie Wang, Evan Sheehan, Anurag Shivaprasad, Yaqi Xie, Katia Sycara, and Yesh Dattatreya. Generalizable dense reward for long-horizon robotic tasks, 2026. https://arxiv.org/abs/2604.00055.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montserrat 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. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 374–404. PMLR, 2023. https://proceedings.mlr.press/v229/yu23a.html.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination?, 2026. https://arxiv.org/abs/2603.16666.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *arXiv preprint arXiv:2509.15937*, 2025.

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations, 2025. https://arxiv.org/abs/2505.10911.

Jianshu Zhang, Chengxuan Qian, Haosen Sun, Haoran Lu, Dingcheng Wang, Letian Xue, and Han Liu. Progresslm: Towards progress reasoning in vision-language models, 2026a. https://arxiv.org/abs/2601.15224.

Yuelin Zhang, Sijie Cheng, Chen Li, Zongzhao Li, Yuxin Huang, Yang Liu, and Wenbing Huang. Recurrent reasoning with vision-language models for estimating long-horizon embodied task progress, 2026b. https://arxiv.org/abs/2603.17312.