

Procedural Activity Understanding in Videos: An Execution-Centric View

Guodong Ding*, Dibyadip Chatterjee*, Zhanzhong Pang*, Anna Kukleva*,
Yale Song, Angela Yao, and Fadime Sener

Abstract—Procedural activity understanding (PAU) aims to interpret goal-directed tasks as structured sequences of steps. As a central problem in video understanding, it spans a broad range of tasks that have largely been studied in isolation. Despite the rapid growth of these techniques, and the recent shift brought by foundation models, no systematic survey has examined them together. This survey reviews the field from an execution-centric perspective. We distinguish a procedure, which specifies how a task should be performed, from an execution, which captures how it is actually carried out in a particular video. Based on this distinction, we organize existing tasks by the question each answers about an execution: what is happening, what comes next, how it is going, and what to convey. For each category, we review the representative methods, and evaluation protocols. In addition, we analyze trends across tasks and show that, while foundation models have largely advanced tasks driven by knowledge of the procedure, those requiring fine-grained perception of the execution remain challenging. We conclude by identifying open challenges toward procedural AI assistants. Finally, we have curated a list of PAU resources, which is maintained at [here](#).

Index Terms—Procedural activity understanding, Video understanding, Vision-language models, Procedural Assistants

1 INTRODUCTION

IN Book VIII of Ovid’s *Metamorphoses*, the gifted craftsman Daedalus, exiled on Crete with his son Icarus, resolves to escape by air. To do so, he builds a pair of wings [1]:

He fashioned quills and feathers in due order — deftly formed from small to large, as any rustic pipe from straws unequal slants. He bound with thread the middle feathers, and the lower fixed with pliant wax; till so, in gentle curves arranged, he bent them to the shape of birds.

With the wings complete, Daedalus and Icarus rise from Crete. Their escape rests on Daedalus’s understanding of the process: its steps and in what order they must occur. People carry out activities like this everywhere, from everyday tasks such as cooking a meal or changing a light bulb to specialized workflows such as assembling products in a factory or performing surgery in a hospital. These are referred to as *procedural activities*: tasks performed as a sequence of steps toward a goal. Video captures them as they are actually carried out, recording not only which steps are performed but how. Understanding procedural activities in video is therefore central to building systems that can recognize, follow, and support them as they unfold.

1.1 Procedural Activities and Procedural Videos

Procedural activities have a hierarchical structure. At the top, the *activity* specifies the overall goal, such as “mak-

ing bread”. It decomposes into *steps*, each a human-understandable instruction such as “prepare the dough”, and each step is realized through *actions* that change the scene, such as “knead dough”. Actions are tied to object *states*: what must hold before an action is performed, and what has changed once it is complete. These dependencies give procedures their characteristic order, although that order is rarely strict, since steps may be optional, interchangeable, or repeated.

Most existing procedural data comes from instructional videos shared online, covering domains such as cooking [2], household repair [3], and tool use [4]. Procedural video is also increasingly captured in situ, using wearable cameras, AR/MR headsets, and multi-view setups that combine ego-centric and exocentric viewpoints [5], [6], [7], [8].

This paper focuses on videos of goal-directed tasks performed as an ordered sequence of steps, which excludes short, self-contained actions such as “waving” or “jumping” and footage with no overall objective. The terms *instructional* and *procedural* video are often used interchangeably: instructional videos are made to teach and are typically edited and narrated, whereas procedural videos depict the continuous execution of a task. Since both show a goal being achieved, we treat instructional videos as a subset of procedural videos and use the broader term *procedural activity understanding* (PAU) throughout. We also include software tutorials that address the tasks we review, although our focus is on physical activities.

1.2 Why Is PAU Important?

Procedures are goal-oriented and complex, which makes PAU important as a research problem. Compared with recognizing isolated actions, understanding a procedure requires several capabilities at once. Perception must be fine-grained: consecutive steps often involve the same objects

- Guodong Ding, Dibyadip Chatterjee, Zhanzhong Pang and Angela Yao are with the School of Computing, National University of Singapore, Singapore (emails: dinggd@comp.nus.edu.sg, dibyadip@comp.nus.edu.sg, e0672670@u.nus.edu, ayao@comp.nus.edu.sg).
- Anna Kukleva and Fadime Sener are with Meta (email: kuklevaanna@gmail.com, famesener@meta.com).
- Yale Song is with Google (email: yalesong@google.com)
- * indicates equal contribution.

Manuscript received October 19, 2022

and similar motions, and differ mainly in the state they produce, such as whether dough has been kneaded or only mixed. Reasoning must span long horizons, because the relevance of a step may depend on something that happened minutes earlier. Interpretation must account for intent, as the same action can serve different goals that are rarely stated. Finally, judging whether a procedure is carried out correctly requires telling valid variation, such as an optional or reordered step, apart from genuine error. PAU thus brings visual perception, temporal reasoning, and goal inference together in a single, well-structured setting.

These challenges also matter in practice, because they surface directly in the applications that motivate the field. A wearable assistant guiding a user through a task must infer what the user is trying to achieve, recognize steps as they happen, and respond in time. A robot learning from human demonstrations must extract which steps and state changes matter, rather than imitating the motions it observes. A training or inspection system in an industrial or clinical setting must judge whether each step meets the requirements of a Standard Operating Procedure, where the line between acceptable variation and error has real consequences. In each case, the application determines what evidence is available, how quickly a decision is needed, and what an error costs.

1.3 How the Field has Evolved?

PAU began primarily as a recognition problem: given a video, determine which step is occurring and when. Early benchmarks framed it this way [3], [4], [9], and progress came from stronger visual representations [10], [11], temporal models that capture dependencies between steps [12], [13], and weak supervision from narrations and instructional text, which made large web corpora usable without dense labels [2], [14].

The arrival of foundation models has changed which of these problems are hard. Large-scale visual and vision-language pretraining has improved transfer across activities, viewpoints, and label vocabularies [15], [16], and video language models now recognize steps and activities without procedure-specific pretraining [17], [18]. This is because the language backbone encodes procedural knowledge that can be queried directly, which has made goal inference, long-term anticipation, and planning far more tractable [19], [20], [21]. Problems that once required a closed vocabulary and a task-specific model are increasingly addressed by general ones. At the same time, recognition on established benchmarks is approaching saturation, while newer datasets with egocentric recordings, mistakes, proficiency labels, and dialogue [6], [7], [8], [22] expose gaps that aggregate accuracy conceals. In parallel, the task landscape is shifting from isolated predictions toward systems that act. Progress monitoring, mistake detection, question answering, and planning all require a system to track where a procedure stands and decide what to do about it, and recent work treats the timing of an intervention as part of the prediction [23], [24].

These developments sharpen, rather than remove, the role of perception. Language models supply knowledge of how a procedure is expected to unfold, but not evidence of how a particular execution actually proceeds. Knowing a

procedure is not the same as seeing how it is carried out, and a model that leans on its knowledge may report the expected step instead of the observed one, precisely when the two differ. Much of this survey examines where that distinction matters, and why it becomes decisive once a system must assist someone in real time.

1.4 Scope, Contributions, and Organization

Several surveys cover parts of this landscape. Broad reviews of human action recognition and prediction [25] and of action understanding across temporal scopes [26]. Others focus on individual tasks such as temporal action segmentation [27], temporal action detection [28], [29], action anticipation [30], or mistake analysis in procedural activities [31]. Surveys of egocentric vision [32], [33] and of video large language models [34] cover methods and data that PAU builds on, and a recent review examines egocentric procedural assistants, focusing on error detection, procedural learning, and question answering [35]. To our knowledge, few treat the procedure itself, with its goal, steps, execution state, and possible deviations, as the unit that links these tasks, or examines how foundation models have shifted the field’s open problems.

This survey addresses that gap. Our contributions are as follows: 1) We distinguish a procedure from its execution and group PAU tasks by the question they answer about an execution: what is happening, what comes next, how it is going, and what to convey. 2) We then present a comprehensive review of PAU tasks, we cover its formulation, benchmarks, and methods from early approaches to foundation models, and summarize representative datasets across domains and viewpoints. 3) We show that progress has concentrated on tasks driven by procedural knowledge, while those that depend on visual evidence lag behind, and we examine where current benchmarks fall short. 4) We identify what assistants, robots, and coaching systems still require, and propose principles for deployment-grounded evaluation.

It is our hope that this survey serves readers with different goals. For newcomers, it offers a map of the PAU landscape, from its core recognition tasks to the newer problems of assessing and assisting execution. For researchers working on perception, it consolidates the field’s tasks, benchmarks, and evaluation practices, and points to where visual understanding remains the limiting factor. For those building applied systems, such as wearable assistants, robots that learn from demonstration, or training and inspection tools, it indicates which capabilities are mature enough to build on and which remain open.

The remainder of the paper is organized as follows. Sec. 2 reviews the tasks of PAU in four groups, together with the datasets that support them. Sec. 3 analyzes trends across tasks, the balance between procedural knowledge and visual evidence, and the state of current benchmarks. Sec. 4 examines what deployment as an assistant still requires, and Sec. 5 concludes the paper. A curated list of the papers, datasets, and resources covered is maintained at [here](#).

2 AN EXECUTION-CENTRIC TAXONOMY

We distinguish throughout between a *procedure* and its *execution*. A procedure specifies how a task is meant to be

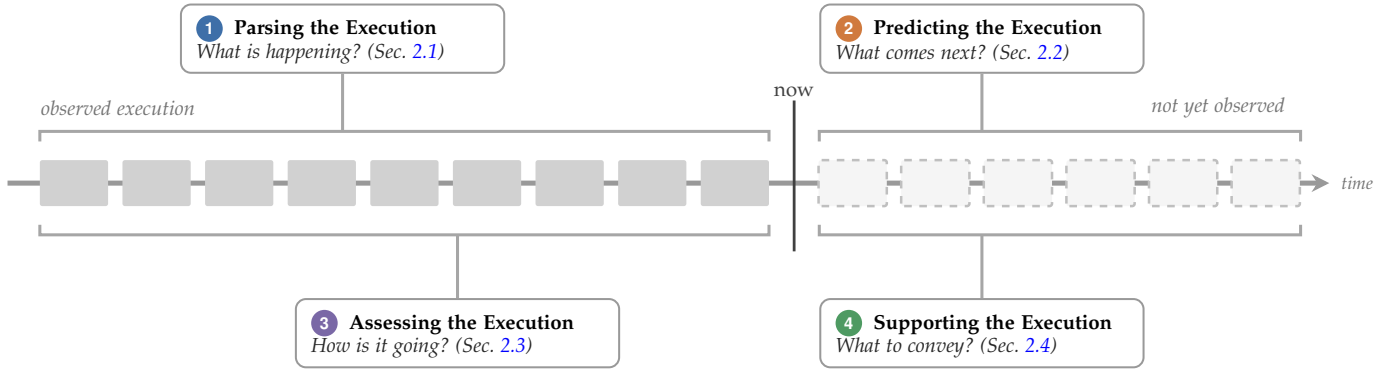


Fig. 1: An execution-centric taxonomy of procedural activity understanding in videos. Tasks are grouped by the question they answer about an execution, and each group is placed over the part of the timeline it works on. ① Parsing reads the observed execution and ② prediction extrapolates beyond it; ③ assessing judges what has happened, and ④ supporting decides what to convey. Each group is reviewed in the section indicated.

performed: its goal, the steps it requires, and the constraints on their order, such as which steps are optional, interchangeable, or must precede others. An execution is a particular instance of carrying out that procedure, as recorded in a video: which steps actually occur, when, and how well. The two rarely coincide exactly, since people skip optional steps, reorder independent ones, repeat or interrupt actions, and make mistakes. Most tasks in procedural activity understanding can be framed as inferring something about an execution, often by reference to the procedure it is meant to follow.

We organize PAU tasks into four groups according to the question they answer about an execution (Fig. 1). ① *Parsing the execution* (Sec. 2.1) determines what is happening: which steps occur, when, and which activity they make up. ② *Predicting the execution* (Sec. 2.2) asks what comes next, given a partial execution or a desired goal. ③ *Assessing the execution* (Sec. 2.3) asks how it is going: how far it has progressed, whether it contains mistakes, and how skillfully it is performed. ④ *Supporting the execution* (Sec. 2.4) asks what the person performing the task needs, and conveys the system’s understanding through captions, retrieved examples, answers, or generated demonstrations. The groups build on one another: prediction relies on a parsed record of what has happened, assessment compares that record with what is expected, and support acts on the result. For each task, we review its formulation, benchmarks and metrics, methods, and close with the issues that remain open.

2.1 Parsing the Execution: What Is Happening?

The first question a system must answer about a procedural video is what is happening in it: which activity is being performed, which steps make it up, and when each step occurs. We refer to this family of tasks as *parsing the execution*. We begin with step recognition (Sec. 2.1.1), the central task of the group and the one on which most procedure-aware methods are developed. We then turn to locating steps in untrimmed video (Sec. 2.1.2) and aligning them with textual descriptions (Sec. 2.1.3), before closing with recognition of the activity as a whole (Sec. 2.1.4), whose output serves as a prior for the tasks that follow. Because most procedure-aware pretraining methods are evaluated on parsing tasks,

we also discuss them here, at the task on which each is primarily evaluated.

2.1.1 Step Recognition

1.1 Step recognition, often called **keystep recognition**, classifies a segment of a procedural video into one of a set of step classes. Unlike generic action recognition, which targets self-contained actions such as “walking” or “jumping”, step recognition concerns fine-grained steps such as “crack eggs”, which typically involve hand-object interaction and belong to a longer, ordered procedure.

COIN [3], with step annotations across 180 tasks, has become the standard benchmark for this task, and several methods also report results on CrossTask [4]. Assembly101 [6] and Ego-Exo4D [8] provide longer, egocentric, and multi-view recordings with 3D hand poses, enabling finer analysis of hand-object interaction. Performance is measured with top-1 accuracy; on Assembly101, accuracy is additionally reported for verbs and nouns separately, which distinguishes errors in recognizing motion from errors in recognizing objects.

Methods for step recognition fall into four groups. *Distant supervision from procedural knowledge*. Because dense step labels are expensive, an influential line of work derives supervision from instructional text. DistantSup [14] matches wikiHow step descriptions [36] to the narrations of HowTo100M [2] to produce pseudo step labels for pretraining, followed by fine-tuning on COIN. Paprika [37] extends this idea by building a procedural knowledge graph from wikiHow and HowTo100M and pretraining with graph-based objectives that encode relationships between steps. VideoTaskGraph [38] mines a probabilistic task graph from video and uses it to regularize step recognition, improving zero-shot performance in particular. *Modeling step dependencies*. Other methods encode the temporal structure of steps directly in the pretraining objective. ProcedureVRL [39] models step representations and their dependencies jointly with a probabilistic, diffusion-based formulation, enabling zero-shot step recognition, while VideoTF [40] learns to recover steps from partially observed sequences with masked clip modeling. *Foundation models*. Video large language models now reach state-of-the-art step recognition on COIN without procedure-specific pretraining on HowTo100M [17],

[41], [42]. ProVideLLM [18] further improves fine-grained recognition with a connector that attends explicitly to hand-object regions, and achieved the best result on the Ego-Exo4D fine-grained keystep recognition challenge using egocentric video alone. *Hands and viewpoints*. Egocentric and multi-view datasets have enabled methods that focus on where procedural evidence actually lies. HandFormer [43] combines 3D hand poses with sparsely sampled frames to model fine-grained motion at a fraction of the usual compute, and restricting egocentric input to a stabilized, hand-focused view alone improves keystep recognition on Ego-Exo4D without changing the underlying model [44]. Multi-view methods exploit synchronized recordings during training: MAGLEV [45] and graph neural networks over views [46] align egocentric and exocentric features, and ViewBridge [47] distills multi-view training videos to improve recognition from a single exocentric view, where hands are often occluded.

The trajectory of step recognition mirrors that of the field. Gains first came from injecting external procedural knowledge into pretraining; they now come increasingly from general foundation models, which already carry much of that knowledge. What remains difficult is visual: distinguishing steps that share objects and motions and differ mainly in the state they produce, particularly in egocentric and multi-view settings. That a simple hand-focused crop yields substantial gains [44] is telling, since it suggests that current models still fail to attend to the evidence that distinguishes one step from another.

2.1.2 Temporal Parsing

Step recognition assumes that step boundaries are given. In practice, procedural videos are long and untrimmed, and a system must also determine when each step occurs. The literature addresses this under several names, including temporal action segmentation, temporal action detection, step localization, and online action detection. We treat them together as *temporal parsing*, since they differ mainly in two respects: whether the output is a label for every frame or a set of labeled intervals, and whether the model may observe the whole video or only the past.

At the frame level, **1.2 temporal action segmentation** (TAS) assigns an action label to every frame of an untrimmed video, making it a temporal analogue of semantic image segmentation. Its central challenges are learning discriminative frame representations and keeping labels consistent across long sequences, so that transitions are identified without fragmenting actions into many short segments. Fully supervised methods rely on dense frame-wise labels, with multi-stage temporal convolutions [12] and transformer encoder-decoders [13] as influential architectures, and hierarchical segment-level refinement [48] improving temporal consistency. More recent architectures reframe how temporal structure is modeled: DiffAct [49] treats segmentation as conditional generation with iterative denoising, LTContext [50] examines how much long-term context is actually needed and models it efficiently, and FACT [51] runs frame-level and action-level branches in parallel with cross-attention between them, which also allows textual transcripts to inform segmentation. Because dense labels are costly, much work reduces supervision:

weakly supervised methods learn from transcripts or action sets [52], semi-supervised methods combine a few labeled videos with many unlabeled ones [53], [54], and unsupervised methods exploit structure shared across videos of the same activity, for instance through temporal clustering [55], [56] or models of action ordering [57]. More recent directions include long-tail learning [58], [59], continual learning [60] and dataset condensation [61], [62]. We refer readers to [27] for a dedicated survey.

At the interval level, **1.3 temporal action detection** (TAD) predicts a set of action instances, each with a class and start and end times. Most TAD research uses general-purpose video, and we refer readers to existing surveys [28], [29]; procedural benchmarks such as EPIC-KITCHENS-100 [63] and Ego4D Moment Queries [5] add dense, fine-grained actions and longer recordings. Methods for these settings focus on more discriminative temporal features, through multi-scale feature pyramids with local self-attention [64], dynamic feature aggregation [65], frame-level sensitivity weighting [66], and multi-scale convolution-transformer hybrids [67], or unify detection with moment retrieval in a shared space [68]. CausalTAD [69] restricts each component to past or future context, combining causal attention with causal state-space (Mamba) blocks, and ranked first in both the EPIC-KITCHENS action detection and Ego4D Moment Queries challenges of 2024.

1.4 Step localization pursues the same interval-level output for procedural steps, but typically without step labels. These methods assume that videos of the same task share a small set of key steps and discover them across a collection: by clustering visual and textual atoms across videos [70], by summarizing multiple videos jointly [71], by proposing and sequentially predicting segments [72], by finding temporal correspondences across egocentric videos [73], or by exploiting cross-modal consistency in a self-supervised manner [74]. Common benchmarks are ProceL [71], CrossTask [4], and the egocentric EgoProceL [73].

Applications like assistants, however, requires parsing a procedure as it unfolds, using only past and present frames. **1.5 Online action detection** (OAD) predicts the ongoing action at each time step under this causal constraint. Early work studied short videos with one or a few independent actions [75], [76], and introduced architectures that separate long-term and short-term memory [77], [78], [79], or that run as lightweight recurrent models suitable for real-time use [80]. Procedural video is more demanding, as streams are longer and steps depend on one another. Ego4D Goal-Step [81] introduced online step detection for egocentric procedures, and CMeRT [82] established a procedural OAD benchmark on CrossTask while correcting a mismatch between the memory lengths seen during training and inference. In parallel, OnlineTAS [83] formalized online segmentation, which requires dense, causal frame labels rather than only the current action, and ProVideLLM [18] brought streaming video language models to procedural parsing with a multimodal cache that interleaves textual summaries of the past with recent visual context, setting the state of the art on online step detection in Ego4D Goal-Step. Procedural structure can also be imposed at inference time: a very recent training-free approach VidParse [84] decodes online parses under the constraints of a task graph induced

from training sequences, reporting large gains over learned online baselines on GTEA and EgoPER.

Evaluation differs across these formulations. TAS is evaluated with frame-wise accuracy together with the segmental Edit score and $F1@k$, which penalize over-segmentation. TAD reports mean average precision at multiple temporal IoU thresholds, and step localization reports F1 and IoU against annotated steps. OAD has traditionally used per-frame mAP and calibrated AP; CMeRT [82] shows that these can overstate performance on short or fragmented actions, and proposes event-level alternatives: point-wise F1 within a time tolerance, segment-wise F1, and the Edit score.

These distinctions, however, reflect benchmark conventions more than differences in the underlying problem. Segmentation, detection, and localization all recover when steps occur; they differ in output format and supervision. In the online setting the convergence is sharper still: OAD on procedural video and online TAS ask essentially the same question, and the field would benefit from evaluating them under shared event-level metrics rather than separate per-frame protocols. More broadly, offline parsing on established benchmarks says little about whether a system can track a procedure live, which is the setting assistance requires.

2.1.3 Step Grounding and Alignment

1.6 Step grounding connects video to free-form textual descriptions of steps, drawn from instructional articles or narrations, rather than to a fixed label set. Given a list of step descriptions and a video of the same task, the goal is to determine which steps are performed and when. The task must cope with noisy descriptions, and with steps that are omitted, repeated, or carried out differently from how they are written. It is usually approached with weak supervision, using only task-level pairings between videos and text.

Three benchmarks are commonly used. HT-Step [85] pairs videos from HowTo100M with wikiHow articles and manually annotates the temporal extent of each visible step. CrossTask [4] serves as a benchmark for zero-shot step localization, and HTM-Align [86] annotates the alignment between video segments and narration sentences in 80 videos. Earlier work evaluated alignment as segmentation, with frame-wise accuracy and F1, whereas grounding-oriented work reports Recall@1: the fraction of alignable steps whose predicted location falls within the correct ground-truth interval.

Methods have moved progressively from constraints toward language models. *Constraint-based alignment.* Early work clustered narrations into main steps and used them as constraints to localize steps in video [87], or exploited components shared across related tasks to localize steps from narration and a step list, extending to unseen tasks [4]. *Ordered sequence alignment.* Subsequent methods model the order of steps explicitly while allowing deviations: through differentiable weak sequence alignment with unmatched steps [88], through flow graphs derived from procedural text that admit several valid orderings [89], and through learned step slots aligned to narrations with outlier rejection [90]. *Narration denoising.* Because narration is only loosely synchronized with the actions it describes, several

methods learn to correct it, by co-training alignment networks whose agreement serves as supervision [86], or by aligning frames, narrations, and wikiHow steps jointly [91]. *LLM-based alignment.* Large language models are used to filter and summarize task-relevant narrations into pseudo-alignments from multiple pathways [92], or to rewrite noisy speech transcripts into procedural steps that are grounded jointly [93]. *Training-free grounding with multimodal models.* Most recently, BaGLM [94] shows that large multimodal models are strong step-grounding baselines without any task-specific training, and builds on them to ground steps online, without additional training or data collection. It outperforms the best trained method on HT-Step and also evaluates on egocentric video from Ego4D Goal-Step.

These developments move grounding toward the online, open-vocabulary setting that assistance requires, and make the textual side of the alignment cleaner. They do not, however, address a mismatch on the visual side. A single written step may summarize several visible actions, or describe one action at a finer level than the video shows, and current methods rarely model this difference in granularity. As textual inputs become more polished, verifying that each grounded step is actually supported by the video becomes the main challenge.

2.1.4 Activity and Goal Recognition

The tasks above parse an execution into its steps. Activity recognition, also called task or goal recognition, instead assigns a single label to a video describing the overall procedure being performed, such as “make latte” or “replace a bulb”. Earlier work refers to this setting as **1.7 complex activity recognition** [9], [95]. Beyond serving as a benchmark in its own right, the predicted activity is often used as a prior for other tasks, for example to keep anticipated actions consistent with the goal [96] or to condition procedure planning [97].

Early benchmarks for this task were cooking-centric: MPII Cooking [95], [98] and Breakfast [9], the latter with 10 breakfast activities. COIN [3] broadened the scope to 180 tasks across 12 domains, and CrossTask [4] provides 83 tasks. More recently, Ego4D Goal-Step [81] introduced 319 goal categories in egocentric video together with an online goal inference setting, and Ego-Exo4D [8] pairs goal labels with synchronized egocentric and exocentric views. Across these benchmarks, performance is measured with top-1 classification accuracy.

Methods for activity recognition have evolved in three stages. *Video classification.* Early approaches treat the task as standard video classification with 2D or 3D CNN backbones [10], [11], sometimes adding components for long-range temporal modeling such as multi-scale temporal convolutions [99]. *Procedure-aware pretraining.* A second line of work exploits the step structure of procedures through weakly supervised pretraining on large instructional corpora [14], [37], [39], [40], the same methods reviewed for step recognition. *Foundation models.* With strong pretrained encoders, such domain-specific pretraining has become less necessary. VEDIT [15] matches or exceeds it by building on frozen features from a general vision-language encoder [100], and video large language models recognize

tasks by drawing on the procedural knowledge of their language backbone [17], [18], [41], [101]. Memory-augmented variants extend these models to long videos by retaining compressed context over time [102], [103], and open-world formulations drop the closed label set altogether, searching an open space of candidate activities with a vision-language model as a probabilistic guide [104].

Offline activity recognition is now approaching saturation on the main benchmarks, with accuracy on COIN exceeding 90% across evaluation protocols [15], [102] and reaching about 96% with memory-augmented video language models [103]. Part of this success likely reflects how distinctive activities are within a closed set: a few frames of coffee beans and a grinder may suffice to identify “make coffee”, and the extent to which current models rely on such cues rather than on the procedure as a whole has not been systematically examined. The open problems therefore lie elsewhere: inferring the goal online, from a partial and possibly ambiguous execution [81], and using the inferred goal reliably as a prior for the tasks that follow.

Taken together, offline recognition on established parsing benchmarks is increasingly well served by foundation models. What remains difficult is distinguishing visually similar steps, parsing procedures causally as they unfold, verifying that textual descriptions match what is observed, and inferring the goal before the procedure is complete.

2.2 Predicting the Execution: What Comes Next?

Parsing establishes what the execution has done so far; the tasks in this group ask what comes next. *Action anticipation* (Sec. 2.2.1) extrapolates the observed execution to its upcoming actions, while *procedure planning* (Sec. 2.2.2) predicts the sequence of actions that would carry the execution from its current state to a desired goal. Both rely heavily on knowledge of how procedures typically unfold, which makes them the tasks where language models have had the clearest impact.

2.2.1 Action Anticipation

Action anticipation predicts future actions from past and present observations; since future frames are unavailable at prediction time, it is inherently a causal task. Early formulations linked anticipation to goal inference and intent prediction [105], [106], and the task has since been studied extensively in egocentric video [30]. We organize the procedural literature by prediction horizon, from the next action to long sequences, and then by whether computation time is taken into account.

2.1 Short-term anticipation predicts the action that will begin a fixed time τ_a in the future. EPIC-KITCHENS [63], [107] popularized this setting with a benchmark of unscripted kitchen activity, and Assembly101 [6] adopts the same formulation for loosely ordered assembly; both use $\tau_a = 1$ s. A segment-level variant instead predicts the next step given the previously completed steps, as in the COIN next-step forecasting benchmark of [40]. Because several actions can plausibly follow the same context [108], EPIC-KITCHENS and Assembly101 report class-mean top-5 recall, which counts a prediction as correct if the true action is among the five highest-ranked and weights all classes

equally; the segment-level benchmark reports top-1 accuracy.

Methods for short-term anticipation differ mainly in how they summarize the past. *Recurrent summarization*. RULSTM [109], [110] separates a “rolling” LSTM that encodes the observed video from an “unrolling” LSTM that predicts ahead, and shows that object-based features from egocentric video substantially improve anticipation. *Aggregating long context*. TempAgg [111] summarizes long observations at multiple granularities, and MeMVIT [112] caches past keys and values in a lightweight memory to model long context at low cost. *Predicting future representations*. AVT [113] uses a causal transformer that unifies temporal modeling and prediction, trained with an auxiliary loss that encourages predicted features to match those of future frames [114]. The same idea now underlies large-scale self-supervised video models that predict masked latent features [16], and a predictor built on such a joint-embedding representation ranked first in the 2026 EPIC-KITCHENS-100 anticipation challenge [115]. *Procedural priors*. Explicitly predicting the goal improves anticipation [96], and the procedure-aware pretraining methods of Sec. 2.1.1 are commonly evaluated on next-step forecasting in COIN [14], [15], [37], [39], where knowledge of typical step order is directly useful.

2.2 Long-term anticipation extends the horizon to sequences of future actions, which requires a model of the procedure’s broader structure. Abu Farha et al. [116] introduced dense anticipation on Breakfast [9] and 50Salads [117], predicting all actions, with their durations, over a future window, and evaluating frame-wise accuracy averaged over classes. Ego4D [5] defines long-term anticipation as predicting the next K verb-noun pairs and scores predictions with edit distance, which tolerates small deviations in order over long horizons. On COIN, [40] predicts the next five steps, with or without the activity label given at test time, and reports top-1 accuracy.

Methods for long-term anticipation have shifted from architectures toward language models. *Architectures for long horizons*. Abu Farha et al. [116] found that one-shot CNN predictions outperform autoregressive RNNs at longer horizons. Ke et al. [118] instead condition prediction on the target time, predicting the action at any future moment directly and avoiding error accumulation, and FUTR [119] decodes all future actions in parallel with a transformer encoder-decoder. Uncertainty-aware formulations sample several plausible futures [120]. *Language models as procedural priors*. AntGPT [19] and PALM [20] translate observed video into text and query a large language model for future actions, following the Socratic models paradigm [121]; both produce sequences that are notably consistent with the goal. PlausiVL [122] trains a video-language model with counterfactual and repetition-aware losses to make its predicted sequences more plausible and more diverse. Because translating video into text loses information, more recent methods keep visual evidence alongside the language model: ICVL [123] uses a vision-language model to infer the performer’s intention as text and feeds it to the language model together with visual features, improving results on Ego4D, EPIC-KITCHENS-55, and EGTEA Gaze+. The winning entry of the 2025 Ego4D long-term anticipation challenge [124] follows the same two-stage design, but attributes

its gains mainly to the recognition stage, which crops hand-object regions and models verb-noun co-occurrence before a fine-tuned language model predicts the future sequence.

Most anticipation methods assume unlimited computation, yet a prediction is only useful if it arrives before the action begins. Furnari and Farinella [125] formalize *streaming anticipation*, in which a model’s runtime determines when its prediction becomes available, and show that lightweight models can outperform heavier ones once latency is counted. RAFTFormer [126] requires forecasts to be issued a fixed margin before the action and sustains real-time throughput with a two-stage transformer. These works evaluate with latency-aware variants of top- k recall, which credit a prediction only if it is correct and emitted in time. A related assumption is that the input is trimmed to end a fixed time before the next action; when anticipation must instead be performed continuously on untrimmed egocentric streams, models designed for the trimmed setting perform considerably worse [127].

Three observations follow from this literature. First, language models have made long-term predictions far more coherent, but coherence is not the same as grounding: a model can produce a plausible continuation of the procedure while paying little attention to what the video shows. Whether predictions change appropriately when the observed execution departs from the typical script, for example after a skipped or repeated step, is rarely tested. Second, the distinction between short- and long-term anticipation is largely one of protocol. Short-term models can be rolled out over longer horizons, and long-term models trivially cover the next action, so results across the two settings would be more informative if reported jointly. Third, short-term anticipation is meaningful only in a streaming setting: a prediction that arrives after the action has started is no longer an anticipation. Despite this, latency-aware evaluation remains the exception rather than the norm.

2.2.2 Procedure Planning

Anticipation extrapolates from what has been observed; planning works toward what is wanted. **2.3 Procedure planning** takes an observation of the current state and an observation of the desired goal state, and predicts the sequence of actions that leads from one to the other within a fixed horizon T . The task requires modeling how actions change the state of the scene and handling the fact that several action sequences may reach the same goal.

Benchmarks are curated from instructional video datasets, most commonly CrossTask [4], COIN [3], and NIV [87], whose videos typically contain four to nine steps. Samples are generated with a sliding window of length T , taking frames at the start of the first and last steps in the window as the start and goal observations. Three metrics are standard: *success rate*, which counts a plan as correct only if the entire sequence matches the ground truth; *mean accuracy*, which compares predicted and ground-truth actions position by position; and *mean IoU*, which treats the two sequences as unordered sets and measures their overlap.

Methods have progressed from modeling state transitions to drawing on external knowledge. *State-transition models*. The task was introduced by Chang et al. [128], who

model each action as a transformation of the visual state, with each state serving as the precondition for the next action. Ext-GAIL [129] and PlaTe [130] learn the transitions with model-based policy learning and transformers. *Weaker supervision*. P³IV [131] removes the need for annotated intermediate observations, using the cheaper language descriptions of steps instead, and models uncertainty by injecting randomness into its transformer. E3P [132] similarly relies on text supervision, guiding planning through inferred events. *Sequence generation*. PDPP [133] recasts planning as fitting a conditional distribution with a diffusion model, conditioned on the task label, and MTID [134] adds a temporal interpolation module that generates intermediate latent features to keep action sequences coherent, while ActionDiffusion [135] injects action embeddings into the diffusion noise so that dependencies between actions are learned during generation. SkipPlan [136] instead decomposes long plans into short sub-chains to avoid accumulating errors from unreliable intermediate predictions. *External knowledge*. KEPP [137] builds a probabilistic procedural knowledge graph to guide step sequencing. SCHEMA [138] represents each step as a state change and uses chain-of-thought prompting of a language model to describe the states before and after it. VidAssist [21] uses a language model both as a source of procedural knowledge and as an evaluator of candidate plans, enabling zero- and few-shot planning, and RAP [97] retrieves relevant state-action pairs from training videos to plan with adaptive rather than fixed length. PlanLLM [139] builds procedure planning around a large language model whose predictions can be refined rather than fixed in a single pass, and LAP [140] translates the start and goal observations into text descriptions with a fine-tuned vision-language model and plans over their embeddings, reporting that these are more discriminative than visual embeddings and setting a new state of the art on CrossTask, COIN, and NIV. Open-event procedure planning [141] instead tests whether planners transfer to activities unseen during training, combining COIN and CrossTask into a benchmark for this setting.

The standard formulation leaves several questions open. Most methods assume a fixed planning horizon, although real tasks vary in length and the number of remaining steps is rarely known in advance [97]. More fundamentally, because a plan is largely determined by the task, strong procedural priors can score well without estimating the start and goal states precisely, and the use of the task label as an input makes this easier still. The finding that text descriptions of the start and goal observations plan better than their visual features [140] sharpens this concern: it suggests that current planners benefit more from naming the states than from perceiving them. Evaluating whether plans adapt to the observed state, for example when a procedure is already partly complete or has gone wrong, would better test whether planning is grounded in the video. Finally, although these methods aim to learn structured representations of state, few visualize or analyze the state spaces they learn, and uncertainty is modeled only in a limited way [131]. Generating intermediate visual states, as explored in procedural generation (Sec. 2.4), offers one route to more interpretable plans.

Across both tasks, language models have made predic-

tions more coherent and plans more plausible, often with little task-specific training. The same strength creates the main risk, since a model can predict what usually comes next rather than what this execution requires. Distinguishing the two depends on comparing predictions with what is actually observed, which is the subject of the next section.

2.3 Assessing the Execution: How Is It Going?

Parsing records what has happened, and prediction describes what should happen next. Assessing an execution combines the two: it asks how far a procedure has progressed, whether it is being carried out correctly, and how well. We review them in order of increasing judgment, from tracking where an execution stands (*progress monitoring*, Sec. 2.3.1), to deciding whether it has gone wrong (*mistake detection*, Sec. 2.3.2), to evaluating the quality of an execution that may be entirely correct (*skill assessment*, Sec. 2.3.3).

2.3.1 Progress Monitoring

Guidance depends on knowing not only which step a person is performing, but how far along it they are. Progress monitoring estimates this as the execution unfolds. It operates at two levels: within an action, estimating how much of the current action has been completed, and across the procedure, estimating which steps are done and how many remain. Because progress is only useful while the activity is ongoing, the task is naturally online.

3.1 Progress monitoring has received far less dedicated attention than the parsing tasks it builds on, and no standard benchmark or protocol exists. Existing methods instead evaluate progress as a component of other tasks. Donahue and Elhamifar [142] learn to estimate activity progress through self-supervised alignment of in-the-wild videos, extending temporal cycle consistency to handle background frames and redundant or repeated actions, which enables online progress estimation without progress labels. ProTAS [143] adds an action progress prediction module to online temporal action segmentation, using the estimated progress to refine predictions in real time, smooth transitions between actions, and reduce over-segmentation. Progress has also been used as a conditioning signal for generation: Xue et al. [144] produce frame-level captions that describe how far an action has advanced. At the procedure level, Ego4D Goal-Step [81] annotates whether steps are completed, providing a starting point for evaluating task-level progress.

Progress monitoring is arguably the most underdeveloped task relative to its importance for assistance. Existing work estimates progress mainly within individual actions, whereas an assistant needs progress over the whole procedure, including steps that are optional, interleaved, or repeated. The absence of a shared benchmark also means that progress estimates are judged only indirectly, through their effect on segmentation or captioning. A dedicated evaluation of procedure-level progress, ideally on the same streams used for online parsing and mistake detection, would make it possible to measure this capability directly.

2.3.2 Mistake Detection

Detecting mistakes, ideally before their consequences become irreversible, is one of the clearest ways an assistant

can help. Mistake detection identifies where an execution departs from the procedure it is meant to follow. Mistakes take two broad forms. **3.2 Procedural errors** concern what is done and in what order: a required step is omitted, steps are performed in the wrong order, an unrelated action is inserted, or an unintended step is substituted for the prescribed one. **3.3 Execution errors** concern how a correct step is carried out: the wrong quantity of an ingredient, a misaligned or misplaced part, a poorly performed technique, or an action stopped too early or too late. The distinction matters because the two place different demands on a system: procedural errors can often be detected from the sequence of recognized steps, whereas execution errors require fine-grained perception of objects, quantities, and spatial configuration.

Dedicated benchmarks have appeared only recently. Assembly101 [6] was the first large-scale dataset to annotate mistakes and their later corrections, and introduced a benchmark in which segments are classified as correct, mistaken, or correcting an earlier mistake. EgoPER [22] and CaptainCook4D [145] record egocentric cooking with both correct and erroneous executions; CaptainCook4D annotates seven error types spanning both procedural and execution errors. HoloAssist [7] pairs mistakes with instructor interventions, linking detection to the decision to help, and the Anomalous Toy Assembly dataset [146] and EPIC-Tent [147] cover erroneous assembly. Online variants of Assembly101 and EPIC-Tent [148] require that errors be flagged as they occur. More recent benchmarks broaden both the domains and the role of instructions: EgoOops [149] pairs egocentric recordings from five procedural domains, including electrical circuits, with the procedural texts being followed, and Ego4D-PMD [150] recasts Ego4D into a frame-level test of whether a described step has been completed. Because recording realistic mistakes is costly, semi-synthetic pipelines that insert plausible errors and recoveries into correct executions have also been proposed [151]. Evaluation protocols still differ widely across these benchmarks, from segment-level classification to frame-level and online detection.

Methods for mistake detection differ mainly in what they compare the execution against. *Comparison with an expected sequence.* The most common approach recognizes the steps performed and checks them against the procedure [152]. PREGO [148] detects mistakes online by comparing the step currently recognized with the step anticipated from the preceding ones, and TI-PREGO [153] strengthens the anticipation with in-context learning and chain-of-thought prompting of a language model. Replacing this implicit expectation with an explicit task graph, learned end-to-end with a differentiable maximum-likelihood objective, further improves online detection on the same benchmarks [154]. AMNAR [155] addresses a limitation shared by these methods, namely that several next steps may be valid at once, by predicting all valid next actions and comparing the ongoing action with a representation of each. Sequence verification [156] takes a related view, deciding whether two videos carry out the same sequence of steps. *Learning correct execution only.* Because errors are diverse and rarely repeated, many methods learn only from correct executions and treat deviations as anomalies. EgoPER [22] detects errors at test time as departures from steps learned on

error-free videos, and Ghoddoosian et al. [146] jointly learn weakly supervised action segmentation and detection of errors unseen during training. Action Effect Modeling [157] adds what an action produces to how it is performed, representing the resulting object states and spatial relations from visual and symbolic cues, and is among the first methods to target execution errors directly. *Signals beyond RGB*. Mazzamuto et al. [158] detect mistakes without any error labels by predicting a performer’s gaze and flagging moments where the observed gaze departs from the prediction, a signal of uncertainty that is most informative for skilled, hand-eye coordinated actions. *Timing*. MistExit [159] poses early mistake detection, deciding whether a step is being performed correctly while observing as little of it as possible, through a detector that anticipates future features and a learned policy that chooses when to stop. *Vision-language models*. Mistake detection can also be posed in language, as in the process-level questions of ProMQA [160]. Requiring vision-language models to justify their decisions with visual rationales shows that they struggle on this task off the shelf, although fine-tuning with rationale-coherence metrics improves both accuracy and transparency [150]. Related work moves beyond binary decisions to attribute a mistake to the part of the instruction it violates or the moment in the video where it occurs [161].

The methods above remain considerably stronger on procedural errors than on execution errors. Comparing recognized steps with an expected sequence works well when the mistake is a missing or misplaced step, and language models and learned task graphs make that comparison easier by supplying the expected sequence. But the same strategy is blind to a step that is present and correctly ordered yet carried out wrongly, such as a part attached at the wrong angle or too much of an ingredient added. The most recent methods that make progress on such errors do so by perceiving more of the execution, its effects on objects [157] or the performer’s attention [158], rather than by knowing more about the procedure. The sequence-based approach also inherits the weaknesses of its inputs: an error in step recognition can appear as a procedural mistake, while a strong prior can mask a real one by predicting the step that should have occurred. Progress on mistake detection will therefore depend on benchmarks that separate error types, report detection delay and false alarms alongside accuracy, and include corrections.

2.3.3 Skill Assessment

An execution can be free of mistakes and still be performed poorly, slowly, or inefficiently. **3.4 Skill assessment** evaluates the quality of an execution: how well the steps are performed, how the performance differs from that of an expert, and what the performer could do better. It matters wherever precision affects the outcome, from surgery and manufacturing to sports and education, and it is the basis for feedback tailored to a learner.

Benchmarks for skill assessment come from domains in which quality is explicitly judged. JIGSAWS [162] pairs robot-assisted surgical demonstrations with gesture labels and expert skill ratings, and FineDiving [163] provides step-level annotations of diving routines for procedure-aware

action quality assessment. Ego-Exo4D [8] introduces proficiency estimation across physical and procedural activities, classifying each performer into one of four levels from novice to late expert, with expert commentary on how participants perform; Assembly101 [6] includes participant skill annotations. Outputs and metrics vary with the formulation, from predicted scores compared against expert ratings, to proficiency classes, to rankings and differences between pairs of executions, to free-form feedback.

Methods have moved from scoring an execution to explaining it. *Scoring and classification*. Action quality assessment predicts a score for a single execution, and procedure-aware variants decompose the score according to the steps of the routine [163]. On Ego-Exo4D, recent proficiency estimators fuse egocentric and exocentric views [164] or use gaze to reach comparable accuracy from the egocentric view alone at lower cost [165]. *Comparison*. Other work compares a performance with a reference. Step Differences in Instructional Video [166] instruction-tunes a language model to answer questions about how two videos differ, and Video Action Differencing [167] identifies fine-grained, skill-relevant differences between two executions of the same action by generating candidate differences with a language model, localizing them in time, and verifying them with a vision-language model. *Feedback*. ExpertAF [168] generates coaching-style feedback grounded in pose and video, trained on Ego-Exo4D with weak supervision from language models, and lightweight video-language models now produce proficiency labels and feedback jointly [164]. Learning interpretable skill attributes has also been proposed as a way to transfer assessment to skills unseen during training [169]. Related work on procedural flexibility retrieves alternative ways of completing a task [170], a reminder that not every departure from a reference execution reflects a lack of skill.

The problem remains far from solved. On Ego-Exo4D, proficiency estimation accuracy across four levels stands at around 50%, and general-purpose vision-language models applied zero-shot perform close to the majority-class baseline [164]. The task is also confined to a few domains, mainly sports, cooking, and surgery, and its formulations are fragmented: scores, comparisons, and feedback are rarely evaluated against one another, and scores seldom transfer across domains. Most methods operate offline over complete executions, whereas an assistant would need to assess technique while it can still be corrected. Distinguishing a lack of skill from legitimate variation is a further difficulty, since experienced performers often deviate from a reference precisely because they are skilled.

Taken together, assessment is where the distinction between knowing a procedure and seeing how it is carried out becomes decisive. These assessments determine when an assistant should act; how it acts, by answering, explaining, or demonstrating, is the subject of the next section.

2.4 Supporting the Execution: What to Convey?

The previous groups determine what is happening, what should come next, and whether the execution is going well. To help, a system must also convey that understanding to the person performing the task, in a form they can use.

The tasks in this group differ in what they communicate. *Captioning* (Sec. 2.4.1) describes an execution in language; *retrieval* (Sec. 2.4.2) finds existing content relevant to a step or a need; *question answering and dialogue* (Sec. 2.4.3) respond to the user, and increasingly anticipate when a response is needed; and *demonstration generation* (Sec. 2.4.4) shows what to do when no suitable example exists.

2.4.1 Captioning

A textual account of an execution is the simplest way for a system to report what it has understood, and a building block for summaries, logs, and instructions. 4.1 **Procedural captioning** differs from generic video captioning in that it must describe ordered, interdependent steps and the changes they cause in objects and materials, rather than summarizing a scene as a whole.

YouCook2 [72] is the most widely used benchmark, pairing cooking videos with temporally localized step descriptions, and HiREST [171] adds step captioning for instructional moments retrieved from a larger corpus. Exo2Ego [172] transfers dense captioning to egocentric procedural video by aligning an egocentric collection with YouCook2. Captions are evaluated with standard language metrics such as BLEU, METEOR, and CIDEr, and dense captioning additionally measures how well the predicted segments match the annotated steps in time.

Methods have progressed from modeling the state of materials to scaling supervision with language models. *State and structure*. Early work introduced state-aware captioning that tracks how materials are transformed across steps [173], and graph-based models that connect visual and textual nodes to capture both sequential and semantic dependencies between events [174]. *Long horizons and progress*. Video ReCap [175] captions hour-long videos recursively, summarizing local clips into progressively higher-level descriptions, and progress-aware captioning [144] conditions frame-level captions on how far an action has advanced, connecting captioning with the progress monitoring of Sec. 2.3.1. *Scaling supervision*. Because dense step captions are expensive to annotate, recent work uses language models to turn narration into captions at scale. HowToCaption [176] prompts a language model to rewrite the speech transcripts of HowTo100M into temporally aligned captions, and DenseStep2M [177] builds a training-free pipeline with video and reasoning language models to annotate instructional videos densely, improving both dense captioning and step grounding.

Captioning remains concentrated on cooking, and its metrics measure overlap with reference sentences rather than procedural correctness: a caption can score well while misstating the order of steps or the state of an object. The growing reliance on language models to produce training captions raises a related concern. Captions derived from narration describe what the narrator said, which is not always what the video shows, so the visual grounding of generated captions needs to be verified rather than assumed. Evaluation that checks step order, object states, and causal dependencies would bring captioning closer to the needs of assistance.

2.4.2 Retrieval

People often need to find the right instructional video, the relevant moment within it, or an alternative way of completing a step. 4.2 **Procedural retrieval** supports these needs. Unlike generic video retrieval, it must respect step structure and goal conditions, and it usually operates over long, noisy, untrimmed content. Recent work treats retrieval at several granularities, from whole videos to moments to individual steps, and uses retrieval not only for search but as a component of other tasks.

HiREST [171] defines a hierarchical benchmark that spans video retrieval, moment retrieval, moment segmentation, and step captioning, and VILT [178] links cooking steps to instructional videos for use by interactive assistants. Large narrated corpora such as HowTo100M [2] provide most of the training data. Video-level retrieval is evaluated with Recall@K, mean average precision, and nDCG; moment retrieval with recall at temporal IoU thresholds; and step-level retrieval with boundary and caption quality.

Methods fall into three groups. *Embeddings from narrated video*. Most procedural retrieval builds on text-video embeddings learned from narrated instructional video, with MIL-NCE [179] treating HowTo100M’s loosely aligned narrations as noisy positives and egocentric variants trained on Ego4D narrations [180], [181]. Evaluated by text-to-video retrieval on YouCook2, these embeddings also underlie most step-grounding methods (Sec. 2.1.3), which pose the same alignment at the level of steps. *Multi-granular and knowledge-guided retrieval*. Beyond the baselines of HiREST, retrieval-augmented encoding retrieves auxiliary clips to enrich the representation of an instructional video when key objects are poorly visible [182], and captions generated from narration by language models substantially improve text-to-video retrieval on instructional data [176]. *Retrieval adapted to the procedure*. Induce, Edit, Retrieve [183] induces task schemata from web videos and wikiHow, edits them with language models to cover unseen tasks, and uses them to retrieve videos zero-shot. Video detours [170] retrieve a related video or segment that satisfies a user’s modification of the procedure, such as completing a step without a particular tool. Retrieval also supports other tasks: RAP [97] retrieves state-action exemplars to guide planning (Sec. 2.2.2), and Step Differences [166] aligns a user’s execution with a reference to compare them (Sec. 2.3.3).

These uses point to where procedural retrieval is heading: toward systems that retrieve with an understanding of why steps are ordered as they are, so that they can find a suitable alternative when the goal or context changes, and toward retrieval as a source of evidence for planning, assessment, and feedback rather than an end in itself. Doing so requires fine-grained alignment between actions, objects, and language.

2.4.3 Question Answering and Dialogue

Answering a user’s questions about a task, and eventually offering help before being asked, is the most direct form of assistance. With the rise of vision-language models, 4.3 **question answering** (QA) has also become the standard way to evaluate general-purpose video understanding, which has made procedural QA benchmarks increasingly important.

Early procedural QA datasets were built from instructional video. HowToVQA69M [184] generates QA pairs automatically from HowTo100M narrations to enable zero-shot training, and its manually annotated iVQA benchmark supports evaluation. TutorialVQA [185] targets software tutorials, and YouCookQA [186] asks questions that require relating several frames, such as counting, durations, and the order of actions. YTCommentQA [187] draws questions from YouTube comments and asks whether they can be answered from the video at all. EgoTaskQA [188] moved to egocentric video and questions about the dependencies and intents behind actions. More recent benchmarks focus on the process itself. ProMQA [160] combines process-level questions, such as identifying missing steps or verifying step order, with step-specific questions about measurements and techniques, and has been extended to assembly [189]. InstructionBench [190] evaluates instructional video understanding at scale and finds that video language models remain weak at temporal reasoning over strictly ordered steps. Benchmarks centered on assistance include expert-trainee guidance in egocentric video [191]. Multiple-choice benchmarks report accuracy, while open-ended answers are increasingly scored by a language model acting as a judge.

Assistance also requires deciding *when* to speak, which moves QA from answering a fixed question to sustaining an interaction over a stream. HoloAssist [7] records instructor-performer dialogue during manipulation tasks, and streaming video language models such as VideoLLM-online [17] generate commentary as a video unfolds. ProAssist [23] synthesizes large-scale assistant dialogues from annotated egocentric videos across several domains, proposes automatic metrics validated through human studies, and trains an end-to-end model that responds proactively to streaming video. SPOT-Bench [192] evaluates proactive assistance across tasks such as live procedural guidance and interventions when users make mistakes or hesitate, judging every prediction throughout the video to account for timely responses, missed interventions, and false alarms. EgoProactive [24] adds explicit annotations of departures from the planned procedure and the recovery steps that follow, and unifies several existing datasets under a common proactive-guidance schema. These works treat the timing and content of an intervention as part of the prediction, which is the setting assistance actually requires.

Procedural QA is moving quickly, but two limitations stand out. First, many questions about procedures can be answered from general knowledge of the task, without the video. Reporting language-only baselines would show how much of a benchmark actually requires visual evidence, a concern that applies with particular force to procedural questions about step order and typical practice. Second, proactive assistance is difficult to evaluate offline. Synthetic dialogues and automatic metrics make large-scale training possible, but whether an intervention is timely and useful ultimately depends on its effect on the user, which current benchmarks measure only indirectly.

2.4.4 Demonstration Generation

When no suitable example exists, a system may need to show a step rather than describe it. **4.4 Demonstration generation** synthesizes images or videos that illustrate how to

carry out a task, either from textual descriptions of steps or from an image of the user’s own scene. It extends procedural understanding from recognition to synthesis, and connects naturally to planning: a generated sequence of images is a visual plan.

Training data is mined from existing procedural datasets. COIN [3] and Changelt [193] provide third-person views from which triplets of initial state, action, and resulting state can be extracted, while EPIC-KITCHENS [63] and Ego4D [5] provide egocentric, hand-centric manipulation. ShowHowTo [194] automatically collects step-by-step image sequences from one million instructional videos, yielding about 0.6M sequences of image-text pairs. Evaluation combines automatic measures of semantic alignment, such as CLIP and DINO similarity, and of visual quality, such as FID and FVD, with metrics designed for procedures, such as goal and step faithfulness and cross-image consistency [195], or accuracy at the level of the step, the scene, and the task [194], and with user studies.

Methods have progressed from single transformations to coherent, multi-step, scene-conditioned sequences. *Single-step transformations.* GenHowTo [196] learns from instructional videos to edit an input image so that it shows an action in progress or its result while preserving the scene. For egocentric settings, LEGO [197] generates action frames by conditioning a latent diffusion model on embeddings from a visually instruction-tuned language model, and instructional video generation [198] conditions on regions of motion to render hands faithfully while avoiding unnecessary changes to the background. *Multi-step sequences.* Generating Illustrated Instructions [195] formalizes the task of producing image-text sequences that guide a user toward a goal, and ShowHowTo [194] trains a video diffusion model to generate step-by-step instructions that remain consistent with an input image of the user’s scene. Recent work also targets specific domains, such as flexible and consistent multi-step recipe illustration [199]. *Training-free composition.* Other methods compose pretrained language and image models without additional training. Text-Image Prompting [200] couples a language model with a text-to-image model through bridges in both directions, LIGER [201] drafts each step image using the history of previous steps and then corrects logical, identity, and attribute errors through self-reflection, and MMPlanner [202] uses chain-of-thought reasoning about object states to enforce consistency.

Across these tasks, the output is language or imagery, and its quality is judged largely by similarity to references. The central difficulty is balancing semantic accuracy with visual consistency. Long sequences drift, objects change identity, and an image can look plausible while depicting the step incorrectly. Several methods address this by reasoning explicitly about object states, which echoes the state-centric view of planning in Sec. 2.2.2, and connecting the two more directly, so that generated images serve as the intermediate states of a plan, is a natural next step. Evaluation remains the main gap: metrics of image quality and text similarity say little about whether a demonstration is procedurally correct, or whether it would help a person complete the task. Coverage is also skewed toward cooking and everyday activities, with little work on the rare or specialized steps where a generated demonstration would be most valuable.

3 CROSS-TASK ANALYSIS

We next trace how methods have converged across the four questions (Sec. 3.1), show that recent gains have come mostly from knowledge of the procedure rather than from evidence of the execution (Sec. 3.2), and ask what current benchmarks measure of each (Sec. 3.3).

3.1 Methodological Trends

From task-specific to shared models. Until recently, each task had its own architectures and often its own pretraining: temporal convolutions and transformers for segmentation, recurrent forecasters for anticipation, and objectives that injected procedural knowledge from narrated video and wikiHow [14], [37]. General-purpose models are replacing much of this. Frozen features from a general vision-language encoder match procedure-specific pretraining [15], self-supervised video models transfer across understanding, anticipation, and planning [16], and a single streaming video language model now performs recognition, anticipation, and planning [18]. A growing number of methods need no task-specific training at all, composing pretrained models for grounding [94] or visual instruction generation [201].

From offline to online. Nearly every task now has an online variant: online step detection and segmentation [81], [83], streaming anticipation [125], online and early mistake detection [148], [159], and proactive dialogue over streaming video [23]. The direction of the field is unambiguous, as we discuss in Sec. 3.3, its evaluation has not kept pace.

Language models as supervision and interface. The emergence of language models has changed PAU in two ways. As supervision, they enabled the move from noisy narration used as weak labels to clean captions and steps [93], [176]. As interface, they let one video-language model serve many tasks that were previously built in isolation recognizing, anticipating, planning, or answering depending on the query, with results returned as free-form language [17], [18].

The return of explicit structure. Alongside the implicit procedural knowledge of language models, explicit structure has reappeared in the form of task graphs [37], [38], [137] that encode a partial order over steps. Such graphs are now learned end to end [154] and used to guide online segmentation [143] or to constrain decoding at inference time [84]. Explicit structure is less general than the knowledge in a language model, but it can be inspected and enforced, which matters when a system must decide whether an execution violates the procedure.

3.2 Knowledge versus Evidence

Every task in this survey combines two ingredients: knowledge of how the procedure should unfold, and evidence about how it is actually being carried out. The tasks differ in the balance between the two (Fig. 2). Predicting typical next steps or planning a canonical sequence can draw heavily on procedural knowledge, whereas distinguishing visually similar steps, judging execution correctness, estimating progress, and assessing skill require evidence about the particular execution.

Language models have substantially advanced the knowledge-driven tasks. They produce goal-consistent

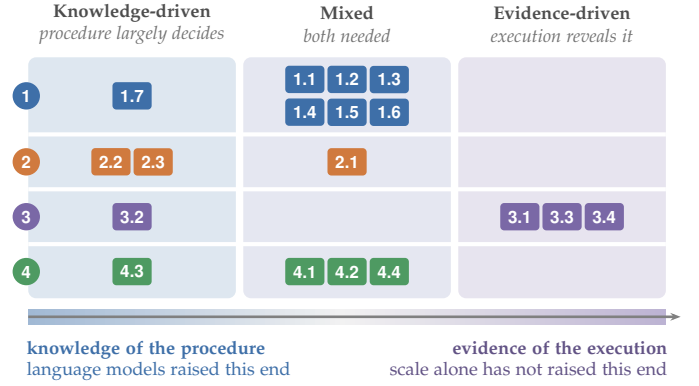


Fig. 2: Where each task sits between knowledge of the procedure and evidence of the execution. Rows are the four task groups and columns the three tiers; tier assignments are qualitative. 1 Parsing: 1.1 step recognition, 1.2 temporal action segmentation, 1.3 temporal action detection, 1.4 step localization, 1.5 online action detection, 1.6 step grounding, 1.7 complex activity recognition. 2 Predicting: 2.1 short-term anticipation, 2.2 long-term anticipation, 2.3 procedure planning. 3 Assessing: 3.1 progress monitoring, 3.2 procedural error detection, 3.3 execution error detection, 3.4 skill assessment. 4 Supporting: 4.1 procedural captioning, 4.2 procedural retrieval, 4.3 question answering, 4.4 demonstration generation.

long-term anticipations [19], and they now supply states, plans, and evaluations for procedure planning [21], [138]. That these gains come from what the model knows is shown by ablation: anticipation degrades when action names are replaced by meaningless labels, which preserves the observed sequence but removes its meaning [19]. Not every language-side gain is knowledge, however. LAP [140] finds descriptions of the start and goal observations more discriminative for planning than their visual features, but these descriptions are generated from visual observations and can therefore encode execution-specific evidence; and a winning long-term anticipation system attributes its gains primarily to better recognition of the observed actions [124], showing that a strong language-based predictor can remain limited by perception.

The evidence-driven tasks remain challenging. Procedural errors can often be identified by comparing recognized steps with an expected sequence, whereas execution errors require determining whether a nominally correct action produced the required result. Recent methods address the latter by modeling action effects on objects [157] or incorporating the performer’s gaze [158]. Proficiency estimation on Ego-Exo4D remains below 50% accuracy across four levels, and general vision-language models applied zero-shot perform near the majority-class baseline [164]. Even in recognition, a simple hand-focused crop of egocentric input yields substantial gains [44], suggesting that selecting the relevant visual evidence remains important.

The imbalance between the two sides has a consequence for evaluation. A model with a strong prior can report the step that should have occurred rather than the one that did, and benchmarks dominated by canonical executions will not expose the substitution. Two recent results show its two halves. When the video is uninformative, the prior suffices:

TABLE 1: Representative datasets for procedural activity understanding, grouped by domain. *Step*, *Mist.*, *Skill*, and *Lang.* indicate step annotations, annotated errors, proficiency or quality labels, and language (narrations, captions, instructions, or dialogue). Columns ① ② ③ ④ mark the task groups each dataset is commonly used for. View: Exo = third-person, Ego = egocentric, Web = online instructional video.

Dataset	Year	View	Hours	Videos	Annotations				Extra modalities	Task groups			
					Step	Mist.	Skill	Lang.		1	2	3	4
Cooking and kitchen													
GTEA [203]	2011	Ego	–	28	✓				–	✓			
50Salads [117]	2013	Exo	4.5	50	✓				Depth, accelerometer	✓	✓		
Breakfast [9]	2014	Exo	77	1.7K	✓				–	✓	✓		
YouCook2 [72]	2018	Web	176	2K	✓			✓	–	✓			✓
EPIC-KITCHENS-100 [63]	2022	Ego	100	700	✓			✓	Audio	✓	✓		✓
HT-Step [85]	2023	Web	–	20K	✓			✓	–	✓			
EgoPER [22]	2024	Ego	28	386	✓	✓			Depth, gaze	✓		✓	
CaptainCook4D [145]	2024	Ego	94.5	384	✓	✓		✓	Depth	✓		✓	✓
Multi-domain and daily life													
CrossTask [4]	2019	Web	374	4.7K	✓			✓	–	✓	✓		
COIN [3]	2019	Web	476	11.8K	✓				–	✓	✓		✓
HowTo100M [2]	2019	Web	134K	1.2M				✓	–	✓			✓
EgoProceL [73]	2022	Ego	62	–	✓				–	✓			
Ego4D Goal-Step [81]	2023	Ego	430	–	✓			✓	–	✓			
HoloAssist [7]	2023	Ego	166	2.2K	✓	✓		✓	Depth, gaze, hands	✓		✓	✓
Ego-Exo4D [8]	2024	Ego+Exo	1,286	5K	✓		✓	✓	Audio, gaze, IMU, 3D	✓		✓	✓
Assembly and industrial													
EPIC-Tent [147]	2019	Ego	–	–	✓	✓			Gaze	✓		✓	
Assembly101 [6]	2022	Ego+Exo	513	4.3K	✓	✓	✓		3D hand pose	✓	✓	✓	
EgoOops [149]	2025	Ego	–	50	✓	✓		✓	–			✓	
ENIGMA-360 [204]	2026	Ego+Exo	–	–	✓				–	✓			
Surgery and sports													
JIGSAWS [162]	2014	Exo	–	–	✓		✓		Kinematics	✓		✓	
FineDiving [163]	2022	Exo	–	3K	✓		✓		–	✓		✓	

InstructionBench [190] filters out questions that language models can answer without the video or its step annotations, and finds that many instructional questions admit answers from commonsense or textual cues alone. When the video contradicts the prior, the prior can win: REVEAL [205] reverses step sequences or removes action segments and finds that video–language models describe the familiar forward event or hallucinate the omitted action. Vision–language models likewise struggle on mistake detection once they must justify their decisions with visual evidence [150]. The phenomenon is familiar from visual question answering, where answers could often be predicted from the question alone [206], [207], and from the object hallucinations of vision–language models [208], [209]; PAU is particularly exposed to it, because familiar step sequences offer strong expectations while deviations may depend on subtle visual differences.

If the evidence-driven tasks are to catch up, it helps to ask what evidence they lack, and much of it concerns object states. For many manipulation steps, completion depends on whether the required effects hold; execution errors involve an incorrect quantity, attachment, or spatial configuration; and skill shows in how cleanly a state is reached. Procedural video has begun to be used to learn such states: Changelt [193] localizes state-modifying actions, together with the initial and final states they connect, in untrimmed instructional video, and Ego4D [5] defines benchmarks for detecting whether and when an object’s state changes. Several methods reviewed in Sec. 2 also

build on states directly, representing steps as state changes for planning [138], detecting mistakes from the effects an action produces [157], and reasoning about object states to keep generated demonstrations consistent [202]. These uses remain isolated: few procedural benchmarks annotate states, and parsing, progress monitoring, and mistake detection rarely share a representation of them. A shared, verifiable account of what each step should produce would let the evidence-driven tasks build on one another, much as procedural knowledge from language models has done for the knowledge-driven ones.

3.3 Data and Benchmarks

Table 1 summarizes representative datasets used across these tasks, grouped by domain. Early benchmarks consisted largely of third-person cooking and web instructional videos annotated with steps, whereas more recent datasets add egocentric and multi-view recordings, additional sensing modalities, and annotations of mistakes, skill, and dialogue. The data available for PAU has broadened in several directions. Annotations have moved from step labels toward mistakes, corrections, proficiency, and dialogue; recordings have shifted from edited web tutorials toward egocentric and multi-view capture with additional modalities. Domains are beginning to diversify, with recent datasets covering electrical circuits [149] and industrial settings [204] alongside long-standing benchmarks in surgery [162] and sports [163]. Cooking and assembly nevertheless remain

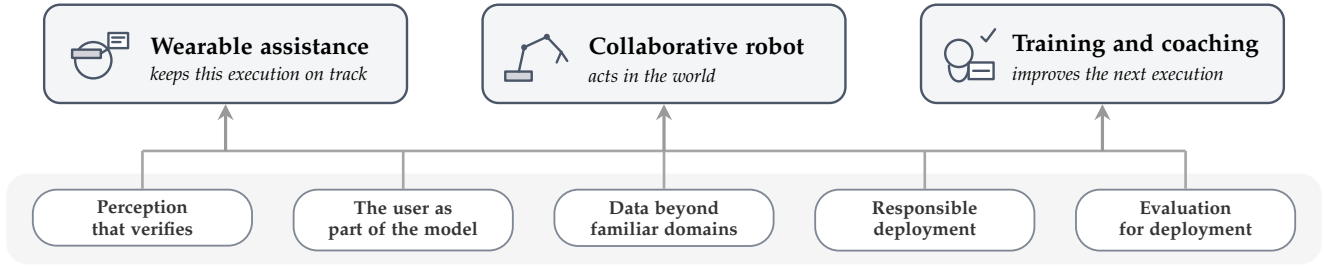


Fig. 3: Towards procedural assistants. Three forms of assistance differ in how they act on an execution, but share five requirements that current practice in procedural activity understanding does not yet meet. The first two concern what a system perceives; the remaining three concern how the field collects data, deploys systems, and measures progress.

dominant, and exocentric web video still accounts for most pretraining data.

Benchmarks, however, have changed less than the data. Most remain offline and closed-world, with fixed label sets, familiar domains, and executions that largely follow canonical orders, which are precisely the conditions under which knowledge can substitute for evidence. Language-only baselines, which would reveal how much of a benchmark can be solved without the video, are not yet standard.

Protocols and metrics lag behind the online direction of the field. Latency-aware anticipation remains the exception [126]; models built for trimmed inputs degrade on untrimmed streams [127]; per-frame metrics overstate online detection performance [82]; and mistake benchmarks rarely report detection delay or false alarms. Metrics for language and image outputs measure similarity to references rather than procedural correctness, skill assessment is split across scores, comparisons, and feedback that are seldom compared, and progress within a step has no benchmark at all.

Finally, evaluation is fragmented in a way that no longer matches how systems are built. Recent systems combine capabilities, building mistake detection from online recognition and anticipation or conditioning captions on estimated progress [144], yet each capability is still evaluated separately, on different datasets and with different protocols. How errors in one capability propagate into another is therefore never measured. The field has converged toward what an assistant needs faster than its benchmarks have.

4 TOWARDS PROCEDURAL ASSISTANCE

The flight was never the hard part. Daedalus had worked out every step, and the wings carried both of them off Crete. What he could not do was watch his son fly. When Icarus climbed toward the sun, nobody saw it happen until the wax had already melted. A single warning, given at the right moment, would have been enough. The tasks of Sec. 2 would have let a system see the climb; none of them would have made it speak. That gap, between assessing an execution and acting on the assessment while it still matters, is what procedural assistance has to close.

We consider three settings in which procedural assistance would be deployed, each acting on the assessment in its own way: assistants worn by the person performing a task, which warn and guide (Sec. 4.1); robots that learn from or work alongside people, which act in the world (Sec. 4.2);

and systems that train and coach, which correct so that the next attempt goes better (Sec. 4.3). We then turn to the requirements they share (Sec. 4.4), ending with a proposal for how progress toward assistance should be measured.

4.1 Wearable and AR Assistance

An assistant that sees from the user’s point of view must combine nearly every capability reviewed in Sec. 2. It must parse the current step online, track how far the procedure has progressed, anticipate likely next steps, detect errors early enough for feedback to be useful, and decide whether and how to respond. These capabilities depend on one another: a missed step boundary corrupts the progress estimate, which in turn misleads mistake detection. They should therefore be developed and evaluated as a coupled streaming system rather than as independent offline predictions. The closest existing setups each cover part of this loop. HoloAssist [7] records the interventions of a human instructor, ProAssist [23], SPOT-Bench [192] and EgoProactive [24] evaluate assistants that respond proactively to streaming video, and streaming video language models such as ProVideLLM [18] unify recognition, anticipation, and planning in a single model. None yet evaluates the full loop from perception to intervention on the same stream.

The least developed part of this loop is the decision to intervene. Recent proactive systems treat the timing of a response as part of the prediction, but how good a response is depends on its cost to the user. An unnecessary interruption costs attention and trust, a false alarm teaches the user to ignore the assistant, and a missed error may be irreversible. These costs are asymmetric and depend on the domain: a missed step in a recipe is recoverable, whereas a missed step in a maintenance procedure may not be. Current benchmarks score predictions without reference to these costs. Assistants will need intervention policies that weigh the confidence of a detection against the cost of acting on it, and evaluations that report what happens when they act.

A second gap is the data. The settings where assistance matters most are also the least represented in current data. Existing datasets focus on cooking and assembly, largely because such activities are abundant online and easy to annotate. The strongest case for assistance, however, lies in procedures governed by Standard Operating Procedures, such as industrial maintenance, clinical practice, and laboratory work, where steps are specified precisely and errors have consequences. Organizations in these domains already

document their procedures in manuals and training material, but rarely share either the documents or recordings of their execution. A practical path is to let a general model be adapted to a private procedure, using its manual and a small number of local demonstrations, without the underlying data leaving the organization. This shifts the research question from collecting more public data to adapting reliably from very little, and it makes procedural text a first-class input rather than a source of pretraining supervision.

4.2 Robot Learning and Human-Robot Collaboration

For a robot learning from human demonstration, the value of PAU lies in what it can transfer. Step labels describe what a demonstration looks like, but not which changes matter, which conditions must hold before an action, or which orderings are permissible. A robot needs these instead: subgoals, preconditions, state transitions, and acceptable alternatives, expressed in a form that carries over to new objects, scenes, and embodiments. Several threads reviewed in this survey point in this direction, including state-centric procedure planning (Sec. 2.2.2), explicit task graphs (Sec. 3.1), and self-supervised video world models that have been used to plan robot actions toward visual goals [16]. What is missing is evaluation that closes the loop. Benchmarks should test whether the structure inferred from human video actually supports execution: whether a robot can carry out the procedure, replan after a failure, and generalize to a different embodiment, rather than whether the inferred labels match annotations.

Collaboration adds a second requirement: the robot must understand the person it works with, not only the task. It must infer its partner’s current state and intention, recognize when help would be useful, and coordinate without imposing a single canonical order on a procedure that admits many. This couples the parsing and prediction tasks of Sec. 2 with uncertainty-aware decision making and safe action selection, since an incorrect inference now leads to a physical action rather than a wrong label. Progress here should be measured in terms of the team: how often the task is completed, how well the pair recovers from errors, how much effort the human expends, and how many interventions were unnecessary.

4.3 Interactive Training and Coaching

Training systems must not only detect what went wrong, but explain it. A useful coach answers what happened, why a step matters, and how to recover, and points to the moment in the video or the line in the instructions that supports its answer. Grounded explanation of this kind is the natural remedy for the substitution of knowledge for evidence described in Sec. 3.2: an answer that must cite visual evidence is harder to produce from the prior alone. Early steps already exist. Requiring vision-language models to justify mistake decisions with visual rationales exposes where they fail and, with fine-tuning, improves both accuracy and transparency [150], and grounded video question answering asks models to localize the evidence for an answer [210]. Extending these ideas to procedural feedback, across longer executions and more varied domains, is a clear next step.

Coaching also changes what success means. Current skill feedback is judged by its similarity to an expert’s commentary [164], [168], but the purpose of feedback is to help a learner improve. Two gaps follow. First, assessment must move online, so that technique can be corrected while the step is still being performed rather than after the execution is complete. Second, evaluation must eventually measure learning outcomes: whether a person who receives the feedback performs better on the next attempt, and whether that improvement persists. This requires user studies and longitudinal data that the field does not yet collect.

4.4 Shared Requirements

Several requirements cut across all three contexts (Fig. 3). We state them once here.

Perception that verifies. Much progress has concentrated on knowledge-dominated tasks with vision-language models, leaving those that depend on evidence about the execution behind. Three directions follow. The first is representations of what an action produces, such as object states, spatial relations, and geometry, which the most successful recent methods for execution errors already use [157]. The second is signals beyond RGB where vision is ambiguous, including gaze, 3D hand pose, audio, and inertial sensing, each of which has already shown benefits for specific tasks. The third is evaluation designed to separate prior from evidence: test cases in which the execution departs from the typical script, pairs of visually similar correct and incorrect executions, and language-only baselines that reveal how much of a benchmark can be solved without the video.

The user as part of the model. Experts and novices differ in pace, strategy, typical errors, and the kind of help they want; an expert may benefit from fewer interventions, whereas a novice may need detailed guidance. Users with different physical or cognitive abilities may need different presentation and timing altogether. Current procedural benchmarks treat performers as interchangeable, and the proficiency annotations of Ego-Exo4D [8] are among the few signals of who is performing a task. Personalized assistance will require longitudinal histories of the same users, explicit representations of their goals and preferences, and benchmarks that evaluate a system on how well it adapts to an individual rather than on its average performance across a population. The intervention policy of Sec. 4.1 is a natural place for this adaptation to begin: whether earlier warnings were followed, ignored, or resented is direct evidence of when and how that user wants to be told, and an assistant that learns from this interaction history can tune its threshold for speaking to the person in front of it rather than to an average performer.

Data beyond familiar domains. The specialized procedures in which assistance matters most are also those in which language priors are weakest, since they are poorly covered by web text and differ from everyday activity. Beyond the private adaptation discussed in Sec. 4.1, two routes seem promising: generating realistic mistakes and recoveries from correct executions [151], which reduces the cost of collecting rare errors, and pairing recordings with the procedural texts being followed [149], so that correctness can be defined with respect to an explicit instruction rather than inferred from typical practice.

Responsible deployment. Assistants that observe people continuously raise questions that recognition benchmarks do not. Always-on cameras record bystanders and workplaces as well as the user; recordings of how individuals perform tasks reveal skill and error patterns that could be misused; and datasets collected in workplaces require consent and governance beyond those of public web video. These requirements should shape dataset design and system architecture from the start, for example by favoring on-device processing and data minimization, rather than being addressed after deployment.

Evaluation for deployment. The capabilities reviewed in this survey are increasingly combined within single systems, yet they are still evaluated separately, offline, on different datasets, and with different protocols. We therefore advocate a unified evaluation built on four principles:

- 1) *Streaming by default.* Models observe only the past and are charged for the time they take to respond.
- 2) *Joint evaluation on shared streams.* Recognition, progress, anticipation, mistake detection, and grounded interaction are measured on the same executions, so that the propagation of errors between capabilities becomes visible.
- 3) *Deployment metrics.* Alongside accuracy, benchmarks report detection delay, false-alarm rate, recovery from errors, and computational cost.
- 4) *Closed-loop assessment.* At least part of the evaluation measures whether assistance helps people complete procedures, through user studies of task success, effort, and learning.

5 CONCLUSION

Distinguishing a procedure from an execution, this paper organized PAU tasks by the question each answers about an execution: what is happening, what comes next, how it is going, and what to convey. Three findings emerge. Knowing the procedure has become tractable, as language models and general video representations now supply much of what was once hardest. Seeing the execution has not kept pace, as fine-grained perception of what happened remains the bottleneck for execution errors, progress and skills. And evaluation has not followed the field’s ambitions, as systems increasingly act on streams while benchmarks still score offline, per-task predictions on scripted executions. What Icarus needed was someone watching the execution, ready to warn him in time. Building that watcher is where the field’s effort should now go: perception fine enough to tell how a step was carried out, and the judgment to speak before it is too late. Until then, an assistant will know the procedure and still watch Icarus fall.

REFERENCES

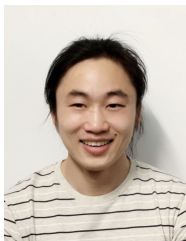
- [1] Ovid, “Metamorphoses, book viii,” Translated by B. More (Boston: Cornhill Publishing Co., 1922). Perseus Digital Library, Tufts University, lines 183–235. [Online]. Available: <http://www.perseus.tufts.edu/hopper/text?doc=Perseus:text:1999.02.0028:book=8:card=183>
- [2] A. Miech, D. Zhukov, J.-B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic, “Howto100m: Learning a text-video embedding by watching hundred million narrated video clips,” in *ICCV*, 2019.
- [3] Y. Tang, D. Ding, Y. Rao, Y. Zheng, D. Zhang, L. Zhao, J. Lu, and J. Zhou, “Coin: A large-scale dataset for comprehensive instructional video analysis,” in *CVPR*, 2019.
- [4] D. Zhukov, J.-B. Alayrac, R. G. Cinbis, D. Fouhey, I. Laptev, and J. Sivic, “Cross-task weakly supervised learning from instructional videos,” in *CVPR*, 2019.
- [5] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu *et al.*, “Ego4d: Around the world in 3,000 hours of egocentric video,” in *CVPR*, 2022.
- [6] F. Sener, D. Chatterjee, D. Shelepov, K. He, D. Singhania, R. Wang, and A. Yao, “Assembly101: A large-scale multi-view video dataset for understanding procedural activities,” in *CVPR*, 2022.
- [7] X. Wang, T. Kwon, M. Rad, B. Pan, I. Chakraborty, S. Andrist, D. Bohus, A. Feniello, B. Tekin, F. V. Frujeri *et al.*, “Holoassist: An egocentric human interaction dataset for interactive ai assistants in the real world,” in *ICCV*, 2023.
- [8] K. Grauman, A. Westbury, L. Torresani, K. Kitani, J. Malik, T. Afouras, K. Ashutosh, V. Baiyya, S. Bansal, B. Boote *et al.*, “Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives,” in *CVPR*, 2024.
- [9] H. Kuehne, A. Arslan, and T. Serre, “The language of actions: Recovering the syntax and semantics of goal-directed human activities,” in *CVPR*, 2014.
- [10] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, “Temporal segment networks: Towards good practices for deep action recognition,” in *ECCV*, 2016.
- [11] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in *CVPR*, 2017.
- [12] Y. Abu Farha and J. Gall, “Ms-tcn: Multi-stage temporal convolutional network for action segmentation,” in *CVPR*, 2019.
- [13] F. Yi, H. Wen, and T. Jiang, “Asformer: Transformer for action segmentation,” in *BMVC*, 2021.
- [14] X. Lin, F. Petroni, G. Bertasius, M. Rohrbach, S.-F. Chang, and L. Torresani, “Learning to recognize procedural activities with distant supervision,” in *CVPR*, 2022.
- [15] H. Lin, T. Nagarajan, N. Ballas, M. Assran, M. Komeili, M. Bansal, and K. Sinha, “Vedit: Latent prediction architecture for procedural video representation learning,” *arXiv preprint arXiv:2410.03478*, 2024.
- [16] M. Assran, A. Bardes, D. Fan, Q. Garrido, R. Howes, M. Muckley, A. Rizvi, C. Roberts, K. Sinha, A. Zhohal *et al.*, “V-jepa 2: Self-supervised video models enable understanding, prediction and planning,” *arXiv preprint arXiv:2506.09985*, 2025.
- [17] J. Chen, Z. Lv, S. Wu, K. Q. Lin, C. Song, D. Gao, J.-W. Liu, Z. Gao, D. Mao, and M. Z. Shou, “Videollm-online: Online video large language model for streaming video,” in *CVPR*, 2024.
- [18] D. Chatterjee, E. Remelli, Y. Song, B. Tekin, A. Mittal, B. Bhatnagar, N. C. Camgoz, S. Hampali, E. Sauser, S. Ma *et al.*, “Streaming videollms for real-time procedural video understanding,” in *ICCV*, 2025.
- [19] Q. Zhao, S. Wang, C. Zhang, C. Fu, M. Q. Do, N. Agarwal, K. Lee, and C. Sun, “AntGPT: Can large language models help long-term action anticipation from videos?” in *ICLR*, 2024.
- [20] S. Kim, D. Huang, Y. Xian, O. Hilliges, L. Van Gool, and X. Wang, “Palm: Predicting actions through language models,” in *ECCV*, 2024.
- [21] M. M. Islam, T. Nagarajan, H. Wang, F.-J. Chu, K. Kitani, G. Bertasius, and X. Yang, “Propose, assess, search: Harnessing llms for goal-oriented planning in instructional videos,” in *ECCV*, 2024.
- [22] S.-P. Lee, Z. Lu, Z. Zhang, M. Hoai, and E. Elhamifar, “Error detection in egocentric procedural task videos,” in *CVPR*, 2024.
- [23] Y. Zhang, X. L. Dong, Z. Lin, A. Madotto, A. Kumar, B. Damavandi, J. Chai, and S. Moon, “Proactive assistant dialogue generation from streaming egocentric videos,” in *EMNLP*, 2025.
- [24] N. Wang, P. Suresh, P. Donmez, B. Damavandi, and K. Kundu, “Plan, watch, recover: A benchmark and architectures for proactive procedural assistance,” *arXiv preprint arXiv:2606.04970*, 2026.
- [25] Y. Kong and Y. Fu, “Human action recognition and prediction: A survey,” *IJCV*, vol. 130, no. 5, pp. 1366–1401, 2022.
- [26] A. Stergiou and R. Poppe, “About time: Advances, challenges, and outlooks of action understanding,” *IJCV*, vol. 133, no. 9, pp. 6251–6315, 2025.
- [27] G. Ding, F. Sener, and A. Yao, “Temporal action segmentation: An analysis of modern techniques,” *IEEE TPAMI*, vol. 46, no. 2, pp. 1011–1030, 2023.

- [28] E. Vahdani and Y. Tian, "Deep learning-based action detection in untrimmed videos: A survey," *IEEE TPAMI*, vol. 45, no. 4, pp. 4302–4320, 2022.
- [29] K. Hu, C. Shen, T. Wang, K. Xu, Q. Xia, M. Xia, and C. Cai, "Overview of temporal action detection based on deep learning," *Artificial Intelligence Review*, vol. 57, no. 2, p. 26, 2024.
- [30] B. Lai, S. Toyer, T. Nagarajan, R. Girdhar, S. Zha, J. M. Rehg, K. Kitani, K. Grauman, R. Desai, and M. Liu, "Human action anticipation: A survey," *arXiv preprint arXiv:2410.14045*, 2024.
- [31] K. Bacharidis and A. Argyros, "Vision-based mistake analysis in procedural activities: A review of advances and challenges," *CVIU*, p. 104842, 2026.
- [32] C. Plizzari, G. Goletto, A. Furnari, S. Bansal, F. Ragusa, G. M. Farinella, D. Damen, and T. Tommasi, "An outlook into the future of egocentric vision," *IJCV*, vol. 132, no. 11, pp. 4880–4936, 2024.
- [33] A. Núñez-Marcos, G. Azkune, and I. Arganda-Carreras, "Egocentric vision-based action recognition: A survey," *Neurocomputing*, vol. 472, pp. 175–197, 2022.
- [34] Y. Tang, J. Bi, S. Xu, L. Song, S. Liang, T. Wang, D. Zhang, J. An, J. Lin, R. Zhu *et al.*, "Video understanding with large language models: A survey," *IEEE TCSVT*, vol. 36, no. 2, pp. 1355–1376, 2025.
- [35] J. Li, H. Xu, S. Cheng, K. Wu, K.-H. Yap, L.-P. Chau, and Y. Wang, "Building egocentric procedural ai assistant: Methods, benchmarks, and challenges," *arXiv preprint arXiv:2511.13261*, 2025.
- [36] M. Koupaei and W. Y. Wang, "Wikihow: A large scale text summarization dataset," *arXiv preprint arXiv:1810.09305*, 2018.
- [37] H. Zhou, R. Martín-Martín, M. Kapadia, S. Savarese, and J. C. Niebles, "Procedure-aware pretraining for instructional video understanding," in *CVPR*, 2023.
- [38] K. Ashutosh, S. K. Ramakrishnan, T. Afouras, and K. Grauman, "Video-mined task graphs for keystone recognition in instructional videos," in *NeurIPS*, 2023.
- [39] Y. Zhong, L. Yu, Y. Bai, S. Li, X. Yan, and Y. Li, "Learning procedure-aware video representation from instructional videos and their narrations," in *CVPR*, 2023.
- [40] M. Narasimhan, L. Yu, S. Bell, N. Zhang, and T. Darrell, "Learning and verification of task structure in instructional videos," *arXiv preprint arXiv:2303.13519*, 2023.
- [41] S. Wu, J. Chen, K. Q. Lin, Q. Wang, Y. Gao, Q. Xu, T. Xu, Y. Hu, E. Chen, and M. Z. Shou, "Videollm-mod: Efficient video-language streaming with mixture-of-depths vision computation," in *NeurIPS*, 2024.
- [42] Z. Pang, D. Chatterjee, F. Sener, and A. Yao, "On discriminative vs. generative classifiers: Rethinking mlms for action understanding," *ICLR*, 2026.
- [43] M. S. Shamil, D. Chatterjee, F. Sener, S. Ma, and A. Yao, "On the utility of 3d hand poses for action recognition," in *ECCV*, 2024.
- [44] Z. Chavis, S. J. Guy, and H. S. Park, "Improving keystone recognition in ego-video via dexterous focus," *arXiv preprint arXiv:2506.00827*, 2025.
- [45] J. L. Romero, K. Min, S. Tripathi, and M. Karimzadeh, "Graph-based multimodal and multi-view alignment for keystone recognition," *arXiv preprint arXiv:2501.04121*, 2025.
- [46] —, "Keystone recognition using graph neural networks," *arXiv preprint arXiv:2506.01102*, 2025.
- [47] A. Somayazulu, E. Mavroudi, C. Chen, L. Torresani, and K. Grauman, "Viewbridge: Curriculum knowledge distillation for activity view-invariance under extreme viewpoint changes," *arXiv preprint arXiv:2504.05451*, 2025.
- [48] H. Ahn and D. Lee, "Refining action segmentation with hierarchical video representations," in *ICCV*, 2021.
- [49] D. Liu, Q. Li, A.-D. Dinh, T. Jiang, M. Shah, and C. Xu, "Diffusion action segmentation," in *ICCV*, 2023.
- [50] E. Bahrami, G. Francesca, and J. Gall, "How much temporal long-term context is needed for action segmentation?" in *ICCV*, 2023.
- [51] Z. Lu and E. Elhamifar, "FACT: Frame-action cross-attention temporal modeling for efficient action segmentation," in *CVPR*, 2024.
- [52] A. Richard, H. Kuehne, A. Iqbal, and J. Gall, "Neuralnetwork-viterbi: A framework for weakly supervised video learning," in *CVPR*, 2018.
- [53] D. Singhania, R. Rahaman, and A. Yao, "C2f-tn: A framework for semi-and fully-supervised temporal action segmentation," *IEEE TPAMI*, vol. 45, no. 10, pp. 11 484–11 501, 2023.
- [54] G. Ding and A. Yao, "Leveraging action affinity and continuity for semi-supervised temporal action segmentation," in *ECCV*, 2022.
- [55] A. Kukleva, H. Kuehne, F. Sener, and J. Gall, "Unsupervised learning of action classes with continuous temporal embedding," in *CVPR*, 2019.
- [56] G. Ding and A. Yao, "Temporal action segmentation with high-level complex activity labels," *TMM*, 2022.
- [57] F. Sener and A. Yao, "Unsupervised learning and segmentation of complex activities from video," in *CVPR*, 2018.
- [58] Z. Pang, F. Sener, S. Ramasubramanian, and A. Yao, "Long-tail temporal action segmentation with group-wise temporal logit adjustment," in *European Conference on Computer Vision*. Springer, 2024, pp. 320–338.
- [59] Z. Pang, F. Sener, S. Ramasub, and A. Yao, "Cost-sensitive learning for long-tailed temporal action segmentation," *BMVC*, 2024.
- [60] G. Ding, H. Golong, and A. Yao, "Coherent temporal synthesis for incremental action segmentation," in *CVPR*, 2024.
- [61] G. Ding, R. Chen, and A. Yao, "Condensing action segmentation datasets via generative network inversion," in *CVPR*, 2025.
- [62] A. Gauthier-Villar, G. Ding, and A. Yao, "Adaptive latent trajectory anchoring for action segmentation dataset condensation," in *ECCV*, 2026.
- [63] D. Damen, H. Doughty, G. M. Farinella, A. Furnari, E. Kazakos, J. Ma, D. Moltisanti, J. Munro, T. Perrett, W. Price *et al.*, "Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100," *IJCV*, pp. 1–23, 2022.
- [64] C.-L. Zhang, J. Wu, and Y. Li, "Actionformer: Localizing moments of actions with transformers," in *ECCV*, 2022.
- [65] L. Yang, Z. Zheng, Y. Han, H. Cheng, S. Song, G. Huang, and F. Li, "Dyfadet: Dynamic feature aggregation for temporal action detection," in *ECCV*, 2024.
- [66] J. Shao, X. Wang, R. Quan, J. Zheng, J. Yang, and Y. Yang, "Action sensitivity learning for temporal action localization," in *ICCV*, 2023.
- [67] R. Dai, S. Das, K. Kahatapitiya, M. S. Ryoo, and F. Brémond, "Mstct: Multi-scale temporal convtransformer for action detection," in *CVPR*, 2022.
- [68] Y. Zeng, Y. Zhong, C. Feng, and L. Ma, "Unimod: Towards unifying moment retrieval and temporal action detection," in *ECCV*, 2024.
- [69] S. Liu, L. Sui, C.-L. Zhang, F. Mu, C. Zhao, and B. Ghanem, "Harnessing temporal causality for advanced temporal action detection," *arXiv preprint arXiv:2407.17792*, 2024.
- [70] O. Sener, A. R. Zamir, S. Savarese, and A. Saxena, "Unsupervised semantic parsing of video collections," in *ICCV*, 2015.
- [71] E. Elhamifar and Z. Naing, "Unsupervised procedure learning via joint dynamic summarization," in *ICCV*, 2019.
- [72] L. Zhou, C. Xu, and J. Corso, "Towards automatic learning of procedures from web instructional videos," in *AAAI*, 2018.
- [73] S. Bansal, C. Arora, and C. Jawahar, "My view is the best view: Procedure learning from egocentric videos," in *ECCV*, 2022.
- [74] A. Shah, B. Lundell, H. Sawhney, and R. Chellappa, "Steps: Self-supervised key step extraction and localization from unlabeled procedural videos," in *ICCV*, 2023.
- [75] R. De Geest, E. Gavves, A. Ghodrati, Z. Li, C. Snoek, and T. Tuytelaars, "Online action detection," in *ECCV*, 2016.
- [76] H. Idrees, A. R. Zamir, Y.-G. Jiang, A. Ghorban, I. Laptev, R. Sukthankar, and M. Shah, "The thumos challenge on action recognition for videos 'in the wild'," *CVIU*, vol. 155, pp. 1–23, 2017.
- [77] M. Xu, Y. Xiong, H. Chen, X. Li, W. Xia, Z. Tu, and S. Soatto, "Long short-term transformer for online action detection," in *NeurIPS*, 2021.
- [78] Y. Zhao and P. Krähenbühl, "Real-time online video detection with temporal smoothing transformers," in *ECCV*, 2022.
- [79] J. Wang, G. Chen, Y. Huang, L. Wang, and T. Lu, "Memory-and-anticipation transformer for online action understanding," in *ICCV*, 2023.
- [80] J. An, H. Kang, S. H. Han, M.-H. Yang, and S. J. Kim, "Miniroad: Minimal rnn framework for online action detection," in *ICCV*, 2023.
- [81] Y. Song, E. Byrne, T. Nagarajan, H. Wang, M. Martin, and L. Torresani, "Ego4d goal-step: Toward hierarchical understanding of procedural activities," in *NeurIPS*, 2023.
- [82] Z. Pang, F. Sener, and A. Yao, "Context-enhanced memory-refined transformer for online action detection," in *CVPR*, 2025.

- [83] Q. Zhong, G. Ding, and A. Yao, "Onlinetas: An online baseline for temporal action segmentation," in *NeurIPS*, 2024.
- [84] A. Gupta, A. Kambhampettu, V. Agarwal, P. Kumar, and A. Shrivastava, "Vidparse: Online parsing of egocentric procedures like a pro," in *ECCV*, 2026.
- [85] T. Afouras, E. Mavroudi, T. Nagarajan, H. Wang, and L. Torresani, "Ht-step: Aligning instructional articles with how-to videos," in *NeurIPS*, 2023.
- [86] T. Han, W. Xie, and A. Zisserman, "Temporal alignment networks for long-term video," in *CVPR*, 2022.
- [87] J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien, "Unsupervised learning from narrated instruction videos," in *CVPR*, 2016.
- [88] Y. Shen, L. Wang, and E. Elhamifar, "Learning to segment actions from visual and language instructions via differentiable weak sequence alignment," in *CVPR*, 2021.
- [89] N. Dvornik, I. Hadji, H. Pham, D. Bhatt, B. Martinez, A. Fazly, and A. D. Jepson, "Flow graph to video grounding for weakly-supervised multi-step localization," in *ECCV*, 2022.
- [90] N. Dvornik, I. Hadji, R. Zhang, K. G. Derpanis, R. P. Wildes, and A. D. Jepson, "Stepformer: Self-supervised step discovery and localization in instructional videos," in *CVPR*, 2023.
- [91] E. Mavroudi, T. Afouras, and L. Torresani, "Learning to ground instructional articles in videos through narrations," in *ICCV*, 2023.
- [92] Y. Chen, K. Li, W. Bao, D. Patel, Y. Kong, M. R. Min, and D. N. Metaxas, "Learning to localize actions in instructional videos with llm-based multi-pathway text-video alignment," in *ECCV*, 2024.
- [93] Z. Li, Q. Chen, T. Han, Y. Zhang, Y. Wang, and W. Xie, "Multi-sentence grounding for long-term instructional video," in *ECCV*, 2024.
- [94] L. Zanella, M. Mancini, Y. Wang, A. Tonioni, and E. Ricci, "Training-free online video step grounding," in *NeurIPS*, 2025.
- [95] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele, "A database for fine grained activity detection of cooking activities," in *CVPR*, 2012.
- [96] O. Zatsarynna and J. Gall, "Action anticipation with goal consistency," in *ICIP*, 2023.
- [97] A. Zare, Y. Niu, H. Ayyubi, and S.-f. Chang, "Rap: Retrieval-augmented planner for adaptive procedure planning in instructional videos," in *ECCV*, 2024.
- [98] M. Rohrbach, M. Regneri, M. Andriluka, S. Amin, M. Pinkal, and B. Schiele, "Script data for attribute-based recognition of composite activities," in *ECCV*, 2012.
- [99] N. Hussein, E. Gavves, and A. W. Smeulders, "Timeception for complex action recognition," in *CVPR*, 2019.
- [100] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *ICCV*, 2023.
- [101] P. Nguyen, S. Sengupta, G. Malik, A. Gupta, and B. Min, "Install: Context-aware instructional task assistance with multi-modal large language models," *arXiv preprint arXiv:2501.12231*, 2025.
- [102] B. He, H. Li, Y. K. Jang, M. Jia, X. Cao, A. Shah, A. Shrivastava, and S.-N. Lim, "Ma-Imm: Memory-augmented large multimodal model for long-term video understanding," in *CVPR*, 2024.
- [103] S. Azad, V. Vineet, and Y. S. Rawat, "Hierarq: Task-aware hierarchical q-former for enhanced video understanding," in *CVPR*, 2025.
- [104] S. Kundu, S. Vellamcheti, and S. N. Aakur, "A probabilistic jump-diffusion framework for open-world egocentric activity recognition," *arXiv preprint arXiv:2505.22858*, 2025.
- [105] M. Pei, Y. Jia, and S.-C. Zhu, "Parsing video events with goal inference and intent prediction," in *ICCV*, 2011.
- [106] T. Lan, T.-C. Chen, and S. Savarese, "A hierarchical representation for future action prediction," in *ECCV*, 2014.
- [107] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price *et al.*, "Scaling egocentric vision: The epic-kitchens dataset," in *ECCV*, 2018.
- [108] A. Furnari, S. Battiatto, and G. Maria Farinella, "Leveraging uncertainty to rethink loss functions and evaluation measures for egocentric action anticipation," in *ECCV Workshops*, 2018.
- [109] A. Furnari and G. M. Farinella, "What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention," in *ICCV*, 2019.
- [110] —, "Rolling-unrolling lstms for action anticipation from first-person video," *IEEE TPAMI*, vol. 43, no. 11, pp. 4021–4036, 2020.
- [111] F. Sener, D. Singhania, and A. Yao, "Temporal aggregate representations for long-range video understanding," in *ECCV*, 2020.
- [112] C.-Y. Wu, Y. Li, K. Mangalam, H. Fan, B. Xiong, J. Malik, and C. Feichtenhofer, "Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition," in *CVPR*, 2022.
- [113] R. Girdhar and K. Grauman, "Anticipative video transformer," in *ICCV*, 2021.
- [114] C. Vondrick, H. Pirsiavash, and A. Torralba, "Anticipating visual representations from unlabeled video," in *CVPR*, 2016.
- [115] Q. Chu, H. Zhang, Y. Feng, M. Liu, W. Guan, D. Jiang, and L. Nie, "Jfaa: Technical report for the epic-kitchens-100 action anticipation challenge at egovis 2026," *arXiv preprint arXiv:2605.20904*, 2026.
- [116] Y. Abu Farha, A. Richard, and J. Gall, "When will you do what?-anticipating temporal occurrences of activities," in *CVPR*, 2018.
- [117] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *UbiComp*, 2013.
- [118] Q. Ke, M. Fritz, and B. Schiele, "Time-conditioned action anticipation in one shot," in *CVPR*, 2019.
- [119] D. Gong, J. Lee, M. Kim, S. J. Ha, and M. Cho, "Future transformer for long-term action anticipation," in *CVPR*, 2022.
- [120] Y. Abu Farha and J. Gall, "Uncertainty-aware anticipation of activities," in *ICCV Workshops*, 2019.
- [121] A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani *et al.*, "Socratic models: Composing zero-shot multimodal reasoning with language," *arXiv preprint arXiv:2204.00598*, 2022.
- [122] H. Mittal, N. Agarwal, S.-Y. Lo, and K. Lee, "Can't make an omelette without breaking some eggs: Plausible action anticipation using large video-language models," in *CVPR*, 2024.
- [123] C. Cao, L. Hu, Y. Yu, and Y. Zhang, "Vision and intention boost large language model in long-term action anticipation," *arXiv preprint arXiv:2505.01713*, 2025.
- [124] Q. Chu, H. Zhang, Y. Feng, M. Liu, W. Guan, Y. Wang, and L. Nie, "Technical report for Ego4D long-term action anticipation challenge 2025," *arXiv preprint arXiv:2506.02550*, 2025.
- [125] A. Furnari and G. M. Farinella, "Towards streaming egocentric action anticipation," in *ICPR*, 2022.
- [126] H. Girase, N. Agarwal, C. Choi, and K. Mangalam, "Latency matters: Real-time action forecasting transformer," in *CVPR*, 2023.
- [127] I. Rodin, A. Furnari, and G. M. Farinella, "Egocentric action anticipation from untrimmed videos," *IET Computer Vision*, vol. 19, no. 1, p. e12342, 2025.
- [128] C.-Y. Chang, D.-A. Huang, D. Xu, E. Adeli, L. Fei-Fei, and J. C. Niebles, "Procedure planning in instructional videos," in *ECCV*, 2020.
- [129] J. Bi, J. Luo, and C. Xu, "Procedure planning in instructional videos via contextual modeling and model-based policy learning," in *ICCV*, 2021.
- [130] J. Sun, D.-A. Huang, B. Lu, Y.-H. Liu, B. Zhou, and A. Garg, "Plate: Visually-grounded planning with transformers in procedural tasks," *IEEE RA-L*, vol. 7, no. 2, pp. 4924–4930, 2022.
- [131] H. Zhao, I. Hadji, N. Dvornik, K. G. Derpanis, R. P. Wildes, and A. D. Jepson, "P3iv: Probabilistic procedure planning from instructional videos with weak supervision," in *CVPR*, 2022.
- [132] A.-L. Wang, K.-Y. Lin, J.-R. Du, J. Meng, and W.-S. Zheng, "Event-guided procedure planning from instructional videos with text supervision," in *ICCV*, 2023.
- [133] H. Wang, Y. Wu, S. Guo, and L. Wang, "Pdpp: Projected diffusion for procedure planning in instructional videos," in *CVPR*, 2023.
- [134] Y. Zhou, Z. Qi, L. Lin, J. Jing, T. Chai, B. Zhang, S. Wang, and W. Zhang, "Masked temporal interpolation diffusion for procedure planning in instructional videos," in *ICLR*, 2025.
- [135] L. Shi, P.-C. Bürkner, and A. Bulling, "Actiondiffusion: An action-aware diffusion model for procedure planning in instructional videos," in *WACV*, 2025.
- [136] Z. Li, W. Geng, M. Li, L. Chen, Y. Tang, J. Lu, and J. Zhou, "Skip-plan: Procedure planning in instructional videos via condensed action space learning," in *ICCV*, 2023.
- [137] K. R. Y. Nagasinghe, H. Zhou, M. Gunawardhana, M. R. Min, D. Harari, and M. H. Khan, "Why not use your textbook? knowledge-enhanced procedure planning of instructional videos," in *CVPR*, 2024.

- [138] Y. Niu, W. Guo, L. Chen, X. Lin, and S.-F. Chang, "Schema: State changes matter for procedure planning in instructional videos," in *ICLR*, 2024.
- [139] D. Yang, Z. Zhao, and Y. Liu, "PlanLLM: Video procedure planning with refinable large language models," in *AAAI*, 2025.
- [140] L. Shi, V. Aregbede, A. Persson, M. Långkvist, A. Loutfi, and S. Lowry, "Lap: A language-aware planning model for procedure planning in instructional videos," *arXiv preprint arXiv:2603.09743*, 2026.
- [141] Y. Wu, H. Wang, J. Wang, and L. Wang, "Open-event procedure planning in instructional videos," *arXiv preprint arXiv:2407.05119*, 2024.
- [142] G. Donahue and E. Elhamifar, "Learning to predict activity progress by self-supervised video alignment," in *CVPR*, 2024.
- [143] Y. Shen and E. Elhamifar, "Progress-aware online action segmentation for egocentric procedural task videos," in *CVPR*, 2024.
- [144] Z. Xue, J. An, X. Yang, and K. Grauman, "Progress-aware video frame captioning," in *CVPR*, 2025.
- [145] R. Peddi, S. Arya, B. Challa, L. Pallapothula, A. Vyas, B. Gouripeddi, Q. Zhang, J. Wang, V. Komaragiri, E. Ragan *et al.*, "Captaincook4d: A dataset for understanding errors in procedural activities," *NeurIPS*, vol. 37, pp. 135 626–135 679, 2024.
- [146] R. Ghoddoosian, I. Dwivedi, N. Agarwal, and B. Dariush, "Weakly-supervised action segmentation and unseen error detection in anomalous instructional videos," in *ICCV*, 2023.
- [147] Y. Jang, B. Sullivan, C. Ludwig, I. D. Gilchrist, D. Damen, and W. Mayol-Cuevas, "Epic-tent: An egocentric video dataset for camping tent assembly," in *ICCV Workshops*, 2019.
- [148] A. Flaborea, G. M. D. Di Melendugno, L. Plini, L. Scofano, E. De Matteis, A. Furnari, G. M. Farinella, and F. Galasso, "Prego: online mistake detection in procedural egocentric videos," in *CVPR*, 2024.
- [149] Y. Haneji, T. Nishimura, H. Kameko, K. Shirai, T. Yoshida, K. Kajimura, K. Yamamoto, T. Cui, T. Nishimoto, and S. Mori, "EgoOops: A dataset for mistake action detection from egocentric videos referring to procedural texts," in *ICCV Workshops*, 2025.
- [150] S. Storks, I. Bar-Yossef, Y. Li, Z. Zhang, J. J. Corso, and J. Chai, "Transparent and coherent procedural mistake detection," in *EMNLP*, 2025.
- [151] O. Loginova and F. Keller, "How to correctly make mistakes: A framework for constructing and benchmarking mistake aware egocentric procedural videos," *arXiv preprint arXiv:2604.15134*, 2026.
- [152] G. Ding, F. Sener, S. Ma, and A. Yao, "Spatial and temporal beliefs for mistake detection in assembly tasks," *CVIU*, vol. 254, p. 104338, 2025.
- [153] L. Plini, L. Scofano, E. De Matteis, G. M. D. di Melendugno, A. Flaborea, A. Sanchietti, G. M. Farinella, F. Galasso, and A. Furnari, "Ti-prego: Chain of thought and in-context learning for online mistake detection in procedural egocentric videos," *arXiv preprint arXiv:2411.02570*, 2024.
- [154] L. Seminara, G. M. Farinella, and A. Furnari, "Differentiable task graph learning: Procedural activity representation and online mistake detection from egocentric videos," in *NeurIPS*, 2024.
- [155] W.-J. Huang, Y.-M. Li, Z.-W. Xia, Y.-M. Tang, K.-Y. Lin, J.-F. Hu, and W.-S. Zheng, "Modeling multiple normal action representations for error detection in procedural tasks," in *CVPR*, 2025.
- [156] Y. Qian, W. Luo, D. Lian, X. Tang, P. Zhao, and S. Gao, "Svip: Sequence verification for procedures in videos," in *CVPR*, 2022.
- [157] W. Guo, Y. Pu, and Y. Kong, "Procedural mistake detection via action effect modeling," in *ICLR*, 2026.
- [158] M. Mazzamuto, A. Furnari, Y. Sato, and G. M. Farinella, "Gazing into missteps: Leveraging eye-gaze for unsupervised mistake detection in egocentric videos of skilled human activities," in *CVPR*, 2025.
- [159] S. Majumder, A. Nethi, Z. Al-Halah, and K. Grauman, "MistExit: Learning to exit for early mistake detection in procedural videos," *arXiv preprint arXiv:2603.14252*, 2026.
- [160] K. Hasegawa, W. Imrattana, Z.-Q. Cheng, M. Asada, S. Holm, Y. Wang, K. Fukuda, and T. Mitamura, "ProMQA: Question answering dataset for multimodal procedural activity understanding," in *NAACL*, 2025.
- [161] Y. Li, A. Jain, F. Bellos, and J. J. Corso, "Mistake attribution: Fine-grained mistake understanding in egocentric videos," in *CVPR*, 2026.
- [162] Y. Gao, S. S. Vedula, C. E. Reiley, N. Ahmadi, B. Varadarajan, H. C. Lin, L. Tao, L. Zappella, B. Béjar, D. D. Yuh *et al.*, "Jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling," in *M2CAI*, 2014.
- [163] J. Xu, Y. Rao, X. Yu, G. Chen, J. Zhou, and J. Lu, "Finediving: A fine-grained dataset for procedure-aware action quality assessment," in *CVPR*, 2022.
- [164] E. Bianchi, J. Staiano, and A. Liotta, "Profvlm: A lightweight video-language model for multi-view proficiency estimation," *CVIU*, p. 104749, 2026.
- [165] C. H. Wu, A. Kumar, and K. Grauman, "Skillsight: Efficient first-person skill assessment with gaze," in *CVPR*, 2026.
- [166] T. Nagarajan and L. Torresani, "Step differences in instructional video," in *CVPR*, 2024.
- [167] J. Burgess, X. Wang, Y. Zhang, A. Rau, A. Lozano, L. Dunlap, T. Darrell, and S. Yeung-Levy, "Video action differencing," in *ICLR*, 2025.
- [168] K. Ashutosh, T. Nagarajan, G. Pavlakos, K. Kitani, and K. Grauman, "Expertaf: Expert actionable feedback from video," *arXiv preprint arXiv:2408.00672*, 2024.
- [169] K. Ashutosh and K. Grauman, "Learning skill-attributes for transferable assessment in video," in *NeurIPS*, 2025.
- [170] K. Ashutosh, Z. Xue, T. Nagarajan, and K. Grauman, "Detours for navigating instructional videos," in *CVPR*, 2024.
- [171] A. Zala, J. Cho, S. Kottur, X. Chen, B. Oguz, Y. Mehdad, and M. Bansal, "Hierarchical video-moment retrieval and step-captioning," in *CVPR*, 2023.
- [172] T. Ohkawa, T. Yagi, T. Nishimura, R. Furuta, A. Hashimoto, Y. Ushiku, and Y. Sato, "Exo2egodvc: Dense video captioning of egocentric procedural activities using web instructional videos," in *WACV*, 2025.
- [173] T. Nishimura, A. Hashimoto, Y. Ushiku, H. Kameko, and S. Mori, "State-aware video procedural captioning," in *ACM MM*, 2021.
- [174] L. Ji, R. Tu, K. Lin, L. Wang, and N. Duan, "Multimodal graph neural network for video procedural captioning," *Neurocomputing*, vol. 488, pp. 88–96, 2022.
- [175] M. M. Islam, N. Ho, X. Yang, T. Nagarajan, L. Torresani, and G. Bertasius, "Video recap: Recursive captioning of hour-long videos," in *CVPR*, 2024.
- [176] N. Shvetsova, A. Kukleva, X. Hong, C. Rupprecht, B. Schiele, and H. Kuehne, "Howtocaption: Prompting llms to transform video annotations at scale," in *ECCV*, 2024.
- [177] M. Ge, Q. Chen, Z. Li, and W. Xie, "Densestep2m: A scalable, training-free pipeline for dense instructional video annotation," *arXiv preprint arXiv:2604.26565*, 2026.
- [178] S. Fischer, C. Gemmell, I. Mackie, and J. Dalton, "Vilt: Video instructions linking for complex tasks," in *IMR*, 2022.
- [179] A. Miech, J.-B. Alayrac, L. Smaira, I. Laptev, J. Sivic, and A. Zisserman, "End-to-end learning of visual representations from uncurated instructional videos," in *CVPR*, 2020.
- [180] K. Q. Lin, A. J. Wang, M. Soldan, M. Wray, R. Yan, E. Z. Xu, D. Gao, R. Tu, W. Zhao, W. Kong *et al.*, "Egocentric video-language pretraining," *arXiv preprint arXiv:2206.01670*, 2022.
- [181] S. Pramanick, Y. Song, S. Nag, K. Q. Lin, H. Shah, M. Z. Shou, R. Chellappa, and P. Zhang, "Egovlpv2: Egocentric video-language pre-training with fusion in the backbone," in *ICCV*, 2023.
- [182] Y. Jung, M. Kim, S. Choi, J. Kim, M. Seo, and S.-w. Hwang, "Retrieval-augmented video encoding for instructional captioning," in *Findings of ACL*, 2023.
- [183] Y. Yang, J. Kim, A. Panagopoulou, M. Yatskar, and C. Callison-Burch, "Induce, edit, retrieve: Language grounded multimodal schema for instructional video retrieval," *arXiv preprint arXiv:2111.09276*, 2021.
- [184] A. Yang, A. Miech, J. Sivic, I. Laptev, and C. Schmid, "Just ask: Learning to answer questions from millions of narrated videos," in *ICCV*, 2021.
- [185] A. Colas, S. Kim, F. Dornoncourt, S. Gupte, D. Z. Wang, and D. S. Kim, "Tutorialvqa: Question answering dataset for tutorial videos," *arXiv preprint arXiv:1912.01046*, 2019.
- [186] S. Wang, W. Zhao, Z. Kou, J. Shi, and C. Xu, "How to make a blt sandwich? learning vqa towards understanding web instructional videos," in *WACV*, 2021.
- [187] S. Yang, S. Park, Y. Jang, and M. Lee, "Ytcommentvqa: video question answerability in instructional videos," in *AAAI*, 2024.
- [188] B. Jia, T. Lei, S.-C. Zhu, and S. Huang, "EgoTaskQA: Understanding human tasks in egocentric videos," in *NeurIPS*, 2022.

- [189] K. Hasegawa, W. Imrattana, M. Asada, S. Holm, Y. Wang, V. Zhou, K. Fukuda, and T. Mitamura, "Promqa-assembly: Multimodal procedural qa dataset on assembly," in *LREC*, 2026.
- [190] H. Wei, Y. Yuan, X. Lan, W. Ke, and L. Ma, "Instructionbench: An instructional video understanding benchmark," *arXiv preprint arXiv:2504.05040*, 2025.
- [191] F. Ragusa, M. Mazzamuto, R. Forte, I. D'Ambra, J. Fort, J. Engel, A. Furnari, and G. M. Farinella, "Ego-extra: video-language egocentric dataset for expert-trainee assistance," in *WACV*, 2026.
- [192] D. Chatterjee, Z. Pang, F. Sener, Y. Song, and A. Yao, "Don't pause! every prediction matters in a streaming video," *arXiv preprint arXiv:2604.24317*, 2026.
- [193] T. Souček, J.-B. Alayrac, A. Miech, I. Laptev, and J. Sivic, "Look for the change: Learning object states and state-modifying actions from untrimmed web videos," in *CVPR*, 2022.
- [194] T. Souček, P. Gatti, M. Wray, I. Laptev, D. Damen, and J. Sivic, "ShowHowTo: Generating scene-conditioned step-by-step visual instructions," in *CVPR*, 2025.
- [195] S. Menon, I. Misra, and R. Girdhar, "Generating illustrated instructions," in *CVPR*, 2024.
- [196] T. Souček, D. Damen, M. Wray, I. Laptev, and J. Sivic, "Genhowto: Learning to generate actions and state transformations from instructional videos," in *CVPR*, 2024.
- [197] B. Lai, X. Dai, L. Chen, G. Pang, J. M. Rehg, and M. Liu, "Lego: Learning egocentric action frame generation via visual instruction tuning," in *ECCV*, 2024.
- [198] Y. Li, Z. Cao, and J. J. Corso, "Instructional video generation," *arXiv preprint arXiv:2412.04189*, 2024.
- [199] R. Zhang, B. Wen, H. Xie, Y. Yao, S. Zuo, J.-Y. Jiang-Lin, H.-H. Shuai, and W.-H. Cheng, "Cookanything: A framework for flexible and consistent multi-step recipe image generation," in *ACM MM*, 2025.
- [200] Y. Lu, P. Lu, Z. Chen, W. Zhu, X. E. Wang, and W. Y. Wang, "Multimodal procedural planning via dual text-image prompting," in *EMNLP*, 2024.
- [201] Y. Suo, F. Ma, K. Shen, L. Zhu, and Y. Yang, "Long-horizon visual instruction generation with logic and attribute self-reflection," in *ICLR*, vol. 2025, 2025.
- [202] A. Tabassum, B. Guo, X. Ma, H. Eldardiry, and I. Lourentzou, "Mmplanner: Zero-shot multimodal procedural planning with chain-of-thought object state reasoning," in *EMNLP*, 2025.
- [203] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *CVPR*, 2011.
- [204] F. Ragusa, R. Leonardi, M. Mazzamuto, D. Di Mauro, C. Quattrocchi, A. Passanisi, I. D'Ambra, A. Furnari, and G. M. Farinella, "Enigma-360: An ego-exo dataset for human behavior understanding in industrial scenarios," *arXiv preprint arXiv:2603.09741*, 2026.
- [205] T. Sethuraman, S. Khosla, A. Tiwari, V. Ganesh, R. Jayaprakash, A. Jain, V. Srinivasakumar, O. K. Susladkar, S. Sunkara, A. Shanmugham *et al.*, "Stress tests reveal fragile temporal and visual grounding in video-language models," in *ICML Workshop*, 2026.
- [206] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the V in VQA matter: Elevating the role of image understanding in visual question answering," in *CVPR*, 2017.
- [207] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi, "Don't just assume; look and answer: Overcoming priors for visual question answering," in *CVPR*, 2018.
- [208] A. Rohrbach, L. A. Hendricks, K. Burns, T. Darrell, and K. Saenko, "Object hallucination in image captioning," in *EMNLP*, 2018.
- [209] Y. Li, Y. Du, K. Zhou, J. Wang, W. X. Zhao, and J.-R. Wen, "Evaluating object hallucination in large vision-language models," in *EMNLP*, 2023.
- [210] J. Xiao, A. Yao, Y. Li, and T.-S. Chua, "Can i trust your answer? visually grounded video question answering," in *CVPR*, 2024.



Guodong Ding is a senior research fellow at National University of Singapore (NUS). Prior to that, he received his PhD from Nanjing University of Science and Technology (NJUST), China in 2020. His research aims to understand and reason about human activities in long-horizon videos, focusing on procedural tasks under realistic constraints such as limited supervision, streaming data, and evolving environments.



Dibyadip Chatterjee is a PhD student at National University of Singapore (NUS). Previously he received his Bachelors from Jadavpur University, India in 2020. His research focuses on multimodal AI for streaming perception and interactive agents.



Zhanzhong Pang is a research fellow at National University of Singapore (NUS). Prior to that, he received his PhD degree from the NUS School of Computing in 2025. His research focuses on streaming VideoLLMs for human-centric video understanding.



Anna Kukleva is a research scientist at Meta. Prior to that, she completed postdoc (2026) and received her PhD degree (2024) from Max-Planck-Institute for Informatics. Her research focuses on representation learning for video understanding.



Yale Song is a Senior Staff Research Scientist and Manager at Google. Previously, he held research positions at Facebook AI Research, Microsoft Research, and Yahoo Research. He received his Ph.D. (2014) and M.S. (2010) in Computer Science from the Massachusetts Institute of Technology (MIT). His work centers on computer vision and machine learning, with a focus on large-scale multimodal learning for video understanding and generation.



Angela Yao received her B.A.Sc. in Engineering Science from the University of Toronto in 2006 and Master's and PhD in 2008 and 2012 respectively from ETH Zurich. She is now an Associate Professor in the School of Computing at the National University of Singapore, where she leads the Computer Vision and Machine Learning group. Her research ranges from low-level enhancement to high-level semantic interpretation of images and video.



Fadime Sener received her BSc. in Computer Engineering in 2011 from Hacettepe University and her MSc. degree in Computer Engineering from Bilkent University in 2013. In 2015, she joined the Visual Computing Group of Angela Yao at the University of Bonn as a doctoral researcher and received her PhD in 2021. She works as a research scientist at Meta. Her research focuses on automatic human activity understanding.