Vid2Coach: Transforming How-To Videos into Task Assistants

Mina Huh, Zihui Xue, Ujjaini Das, Kumar Ashutosh, Kristen Grauman, Amy Pavel
The University of Texas at Austin
Austin, Texas, USA
{minahuh, sherryxue, ujjaini, kumar.ashutosh, grauman, apavel}@utexas.edu

Abstract

People use videos to learn new recipes, exercises, and crafts. Such videos remain difficult for blind and low vision (BLV) people to follow as they rely on visual comparison. We propose Vid2Coach, a system that transforms how-to videos into wearable camera-based assistants that provide accessible instructions and mixed-initiative feedback. From the video, Vid2Coach generates accessible instructions by augmenting narrated instructions with demonstration details and completion criteria for each step. It then uses RAG to supplement non-visual workarounds from BLV-specific resources. Vid2Coach monitors user progress with a camera embedded in commercial smart glasses to provide context-aware instructions, proactive feedback, and answers to user questions. BLV participants (N=8) using Vid2Coach completed cooking tasks with 58.5% fewer errors.

1. Introduction

Videos support efficient skill learning [23, 25] as their rich visual demonstrations share tacit knowledge, low-level details, and allow learners to visually compare their progress. However, blind and low vision (BLV) learners find most videos difficult to follow [19, 21, 22] as videos typically omit key visual details in their narrations and include instructions that assume sight (*e.g.*, *"wait until your butter turns this color"*). Making videos meaningfully useful for BLV learners would address a longstanding learning gap.

We envision a system that transforms how-to videos into wearable camera-based assistants for BLV learners. Our key insight is that the rich audio-visual content that benefits sighted learners can provide an AI assistant with detailed task knowledge to create accessible instructions and assess progress. Further, new commercially available smart glasses offer hands-free capture of hand-object interactions required for real-time feedback, and many BLV individuals already use such technology in daily life [13]. We set cooking as an initial target domain due to its universality, multi-step complexity, diverse tools, and reliance on multimodal signals.

We present Vid2Coach (Figure 1), a system that transforms how-to videos into wearable camera-based assistants that provide accessible instructions and mixed-initiative feedback. From a how-to video, Vid2Coach creates *accessible instructions* by first extracting high-level steps then adding detailed demonstration descriptions and completion criteria for each step via multimodal understanding (Figure 1.1). To supplement instructions with tips and workarounds tailored to users' level of vision or kitchen setup, our pipeline uses retrieval-augmented generation (RAG) to generate suggestions grounded in an accessibility resources dataset (Figure 1.2). As users perform the task, Vid2Coach captures the egocentric video stream (Figure 1.3). It determines whether users are following the current step or engaging in spontaneous actions (*e.g.,* washing hands, cleaning). When a step appears complete based on the completion criteria extracted from the how-to video, Vid2Coach notifies the user – explaining visually grounded reasoning (*e.g., "You seem to be done because the bacon has turned evenly golden brown and crispy."*) For steps where visual cues alone are insufficient to determine completion (*e.g.,* mixing until smooth), it prompts users to use non-visual cues (touch, smell, or sound).

To evaluate Vid2Coach, we conducted a within-subjects study with 8 BLV people in their own home kitchens where they completed full end-to-end recipes. Participants using Vid2coach completed cooking tasks with 58.5% fewer errors and reported lower cognitive load.

2. Vid2Coach

2.1. Generating Step Instructions

Generating Step-Action Hierarchy from Narration. To only retain task-relevant narration, we use GPT-4o [1] to classify each transcript sentence into information roles derived from Yang et al.'s taxonomy [31] and filter greeting, conclusion, and miscellaneous role sentences. Then, we use GPT-4o [1] to group relevant sentences into high-level steps. Inspired by Truong et al. [29] using hierarchical segmentation, we generate **high-level steps** that help users understand the goal (*e.g., Prepare hollandaise sauce*), accompanied by **atomic**

Figure 1. Vid2Coach is a system that transforms how-to videos into a wearable camera-based task assistant that provides accessible instructions and mixed-initiative feedback. Given a how-to video, Vid2Coach extracts high-level steps and demonstration details, then for each step we use retrieval-augmented generation to supplement each step with BLV-specific guidelines. Vid2Coach then monitors user progress with a camera in smart glasses to provide proactive feedback.

actions that helps users to easily follow and assess (*e.g., Separate 3 egg yolks from the whites*).

Identifying Task-Relevant Frames. Visuals like talking-heads or B-roll can distract automated descriptions. To focus on task-relevant visuals, we sample frames at 1 fps within a ±15-second window around each action's transcript segment, capturing demonstrations slightly before or after the instruction while excluding distant irrelevant footage. We compute cross-modal similarity between each frame and the action description using CLIP [26], retaining frames above a dynamic threshold (0.27–0.30). This threshold adjusts per action based on frame density, increasing when many frames exceed it and decreasing when few do, adapting to variations in visual prominence of actions.

Generating Task-Relevant Descriptions. Finally, we combine the narration-aligned step and high-scoring frames to generate detailed action descriptions using GPT-4o [1]. Naive prompting often yields irrelevant details (*e.g.,* presenter appearance, background). To focus on task-relevant content, we adopt a targeted prompting: (1) *What is the demonstrated action?* (2) *Which ingredients are used, what do they look like, and how much?* (3) *Which tools are used, and what do they look like?* (4) *How is the action performed?* (5) *Are any tips visually evident?*

2.2. Supplementing Accessible Strategies

General-purpose AI suggestions often assume dominant user groups, leading to irrelevant or misfitting advice for BLV users. Even when prompted for accessibility, models may hallucinate visual details, suggest seeking sighted help, or provide overly generic tips. To mitigate these biases, we curated a dataset of 100 non-visual cooking videos by BLV individuals and 100 accessibility resources (*e.g.,* blog posts, guides) to ground suggestions in disability-aware knowledge. We apply a multi-modal retrieval augmented generation (RAG) approach[17]: extracting text chunks and generating text embeddings, while describing dataset images using Gemini 1.5 pro [28] to obtain image embeddings. At inference, we input user queries enriched with context (*e.g.,* vision level, experience, tools, setup), retrieve the top-3 relevant chunks via cosine similarity, and generate personalized, grounded suggestions (see § 5).

2.3. Progress Monitoring

Our pipeline generates criteria (*irrelevant, in-progress, complete, mistake*) for each action using GPT-4o [1] to monitor user progress (Figure 5). Rather than direct frame comparisons, we rely on abstracted descriptions to account for tool variation or variation. Vid2Coach pauses feedback during unrelated actions, provides updates during in-progress steps, and prompts advancement upon completion.

We categorize actions as *punctual, iterative,* or *durative* based on temporal properties. For punctual actions (*e.g., add 1 cup of flour*), Vid2Coach confirms completion without intermediate feedback. For iterative actions (*e.g., place 3 scoops*), it tracks repetitions to guide users step-by-step. For durative actions (*e.g., cook until golden brown*), it monitors visual change for real-time updates. For actions lacking visual cues (*e.g., wait to cool*), it prompts user confirmation.

To balance accuracy and latency, we adopt a *dual-model* approach: a batch VLM (Gemini 2.0) performs inference every 5 seconds on the latest 5 frames (1 fps), while a streaming VLM (Gemini 2.0-Live) provides continuous, low-latency monitoring. Streaming models begin generating outputs token-by-token before full input processing [30], enabling responsiveness but limiting reasoning depth.

		Ours	Gemini	GPT-4o		LLaVA-OV	
			video	step	full	step	full
# of New Facts	μ	**11.60**	3.80	6.20	5.30	6.60	4.00
(task-relevant)	σ	2.91	2.04	2.04	3.16	3.69	2.00
# of Missed Facts	μ	**3.80**	8.90	15.80	6.50	15.90	9.60
(task-relevant)	σ	2.39	3.03	9.3	4.12	6.72	3.27
Hallucinations	%	**3.92**	7.14	8.82	5.71	25.58	21.13

Table 1. Coverage and accuracy of Vid2Coach-generated and state-of-the-art VLM descriptions. Vid2Coach captured more new task-relevant facts with fewer hallucinations.

		Vid2Coach	Gemini	CLIP	
Action types	FOV		instruction	criteria	instruction
Punctual	Narrow	**0.53**	0.40	0.33	0.33
	Wide	**0.73**	0.47	0.33	0.33
Iterative	Narrow	**0.60**	0.20	0.33	0.33
	Wide	**0.80**	0.47	0.20	0.13
Durative	Narrow	**1.00**	0.47	0.33	0.33
	Wide	**1.00**	0.40	0.40	0.40

Table 2. Progress monitoring accuracy (Gemini + criteria) across action types and FOVs. Criteria-based prompts were more effective for durative actions, while CLIP showed limited gains.

2.4. Implementation

We implemented Vid2Coach as a React-based web application that integrates Google Gemini Multimodal Live API [2], which supports low-latency bidirectional voice and video communication over WebSocket. Users' progress captured with Meta glasses is streamed to a lab computer and fed into the React app for real-time monitoring and feedback.

3. Technical Evaluation

3.1. Generating Instructions

Method. We selected 10 cooking videos that varied in both dish type and narration density. We compared our pipeline against 3 state-of-the-art VLMs for video understanding tasks [9, 24] – 2 proprietary VLMs (GPT-4o [1], Gemini 2.0 [28]) and 1 open-source VLM (LLaVA-OV [18]). For Gemini 2.0, which supports full-length video input, we provided the entire video directly. For GPT-4o and LLaVA-OV, we used the segmented steps from our own pipeline, and for each step, we tested two input conditions: one providing only the step name and one providing the full narration aligned with that step. Comparing these two conditions allowed us to evaluate how well the models leverage visual information alone versus how they perform when given narrated context. For open-source models that have input frame limit per call, we periodically subsample 32 frames from that step.

Results. Table 1 summarizes the evaluation results. Overall, Vid2Coach outperformed baselines by generating more new task-relevant atomic facts and omitting fewer details present in the original narration. As illustrated in Figure 2, Vid2Coach surfaced concrete visual details that can support BLV users to accurately follow the steps (*e.g.,* describing the use of a measuring spoon, amount of bacon used, and how they're presented).

3.2. Progress Monitoring

Method. We conduct an ablation study comparing our criteria-based classification against a simpler baseline that directly uses the instruction text as context. Beyond Gemini, we also compare the results of CLIP, which is commonly used for similarity matching with text prompts. To assess generalizability, we evaluate on 6 action videos across 2 settings: 1) low-quality, narrow field-of-view streams captured using Meta Glasses from our formative study, and 2) high-resolution, wide field-of-view recordings captured with GoPro from prior work [20]. Both sets of videos are recordings BLV individuals cooking in their own kitchens. As Vid2Coach is designed to support real-time monitoring, we mimic a streaming setup by feeding in frames one at a time rather than feeding the entire segment context to models.

Results. Table 2 shows the average per-frame accuracy over 6 actions (90 frames). Overall, using criteria-based approach improved accuracy by grounding classification in context-specific visual expectation rather than generic instruction text. High-resolution, wide field-of-view videos recorded yield better performance, as they clearly capture the target objects and actions. For punctual actions (*e.g., "add 1 cup of flour"*), all models struggled to correctly classify the "complete" frames. These actions happen quickly and users often instantly shift the camera away from the result – such as the flour in the bowl – making it difficult to visually confirm that the step was completed. Durative actions were more reliably detected, with multiple frames showing ongoing progress and completion status. In contrast, iterative actions were often misclassified when the objects being counted (*e.g.,* multiple egg yolks cracked into a bowl) were only partially visible or occluded during the repetitions. CLIP underperformed across all action types due to its limited understanding of complex prompts – especially those involving negation (*e.g., the butter has not melted into liquid yet.*)

4. User Evaluation

4.1. Method

We conducted a within-subjects study with 8 BLV participants completing end-to-end cooking tasks in their own kitchens, comparing Vid2Coach with a baseline replicating their existing practices. In the baseline, participants received the original how-to video and transcript and could use any tools they typically rely on, including AI- or human-powered visual assistance (*e.g.,* BeMyEyes, AIRA, smart speakers, Meta AI). We selected three narrated YouTube

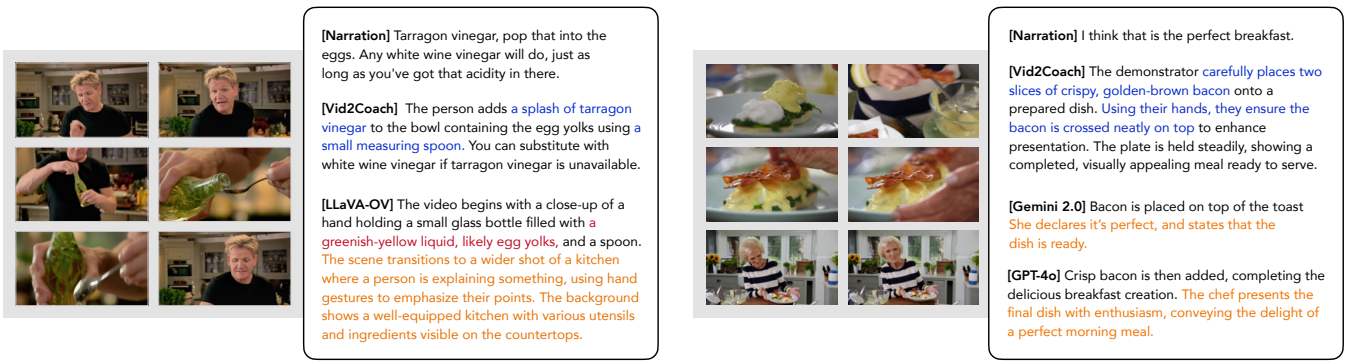

Figure 2. Qualitative comparisons of Vid2Coach descriptions with SOTA VLMs on 2 action sequences. These VLM descriptions often include hallucinations (red) and less task-relevant details (orange). Vid2Coach was able to capture new task-relevant details not covered in the narration (blue).

cooking videos: V1 (Chocolate Chip Cookies) for the tutorial, and V11 (Bread Flapjack) and V12 (Eggs Benedict) for the main tasks. We chose less common recipes to avoid reliance on prior knowledge, ensuring active engagement with the system. Videos were similar in length, narration, and number of steps, and could be completed within 45 minutes while meeting participants' dietary needs.

We conducted a 2-hour study with 8 BLV participants to evaluate Vid2Coach for following how-to cooking videos [1]. Two researchers visited participants' homes with ingredients and spare tools; participants could use their own or ours. Each participant completed two cooking tasks: one with Vid2Coach and one with the baseline, randomly assigned between V11 and V12. Interface and recipe order were counterbalanced. We administered a post-task survey measuring cognitive load (NASA-TLX [10]) and system usefulness.

4.2. Task Success and Load

5 participants using Vid2Coach completed the task, compared to 1 in the baseline condition (Figure 7). With Vid2Coach, participants made 58.5% fewer mistakes (μ=4.38, σ=2.20 vs. μ=11.00, σ=3.16; Z=2.46; $p < 0.05$), suggesting that the system's real-time guidance contributed to both task completion and accuracy. Participants using Vid2Coach also experienced significantly lower mental demand (μ=5.75, σ=1.39 vs. μ=3.38, σ=2.14; Z=2.04; $p < 0.05$), temporal demand (μ=6.25, σ=1.17 vs. μ=3.63, σ=2.50; Z=2.13; $p < 0.05$) and frustration (μ=6.00, σ=1.41 vs. μ=3.88, σ=1.89; Z=1.68; $p < 0.05$). While participants in the baseline condition tried to remember a lot of recipe information at once as the navigation is difficult, Vid2Coach users could easily query to repeat or ask for easier workarounds which reduced the mental load. They also reported higher performance (μ=6.13, σ=0.83 vs. μ=3.63, σ=1.85; Z=-2.29; $p < 0.05$) with lower efforts (μ=5.00, σ=1.93 vs. μ=3.50, σ=2.20; Z=1.70; $p < 0.05$). Participants explained the reason for rating high performance is because they have gotten a lot of confirmation from the system on their progress. With the baseline, participants who did not finish all thee steps had low confidence and felt more rushed. While we did not see any significant difference in physical demand (μ=6.00, σ=1.07 vs. μ=4.38, σ=2.13; Z=1.42; $p > 0.05$), 4 participants mentioned that holding the phone to connect to sighted assistants is physically tiring.

[1] Approved by the institute's IRB.

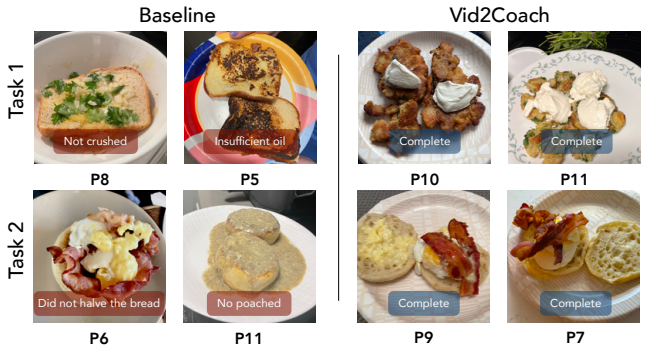

Figure 3. Final dishes from cooking tasks with Vid2Coach and participants' current workflow as baseline. Vid2Coach's detailed instruction and realtime feedback grounded in how-to videos helped participants successfully follow the steps.

5. Conclusion

Vid2Coach bridges the accessibility gap in how-to videos by transforming them into wearable, context-aware assistants that provide accessible instructions and real-time feedback to BLV users. We designed Vid2Coach to offer both proactive and responsive support, grounded in how-to videos and accessibility resources. Our study demonstrates that BLV users can complete tasks more accurately and confidently with Vid2Coach. We show how AI systems can enable more sclable, flexible, and independent skill learning for BLV individuals in real-world setting.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1, 2, 3

[2] Google Cloud Vertex AI. Multimodal live api. 3

[3] BBC. Mary berry's tasty eggs benedict florentine - classic mary berry - bbc, 2018. 2

[4] Ken Click. Crisp tortilla pizza!, 2024. 2

[5] Cook! Stacey Cook. Beef and broccoli stir fry — beef stir fry with vegetables, 2024. 2

[6] Crouton Crackerjacks. How to make tiramisu!! classic italian dessert recipe, 2014. 2

[7] Crouton Crackerjacks. How to make strawberry jam!! homemade small batch preserves recipe, 2017. 2

[8] Fallow. Pov: How to make an omelette like a chef, 2023. 2

[9] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024. 3

[10] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, pages 139–183. Elsevier, 1988. 4

[11] Honeysuckle. World's best chocolate chip cookies recipe: Crunchy outside, soft & chewy inside, 2020. 2

[12] Johnny. How to make the meat filling for dumplings (mandu), 2020. 2

[13] The Wallstreet Journal. Meta's ai-powered ray-bans are life-enhancing for the blind. 1

[14] Natasha's Kitchen. Dessert: Easy mini pavlovas - homemade meringues recipe, 2016. 2

[15] Preppy Kitchen. Mashed potatoes recipe, 2021. 2

[16] KQED. Bread flapjacks — jacques pépin cooking at home — kqed, 2020. 2

[17] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020. 2

[18] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 3

[19] Franklin Mingzhe Li, Ashley Wang, Patrick Carrington, and Shaun K Kane. A recipe for success? exploring strategies for improving non-visual access to cooking instructions. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–15, 2024. 1

[20] Franklin Mingzhe Li, Kaitlyn Ng, Bin Zhu, and Patrick Carrington. Oscar: Object status and contextual awareness for recipes to support non-visual cooking. *arXiv preprint arXiv:2503.05962*, 2025. 3

[21] Xingyu Liu, Patrick Carrington, Xiang'Anthony' Chen, and Amy Pavel. What makes videos accessible to blind and visually impaired people? In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2021. 1

[22] Xingyu" Bruce" Liu, Ruolin Wang, Dingzeyu Li, Xiang Anthony Chen, and Amy Pavel. Crossa11y: Identifying video accessibility issues via cross-modal grounding. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, pages 1–14, 2022. 1

[23] Korinn Ostrow and Neil Heffernan. Testing the multimedia principle in the real world: a comparison of video vs. text feedback in authentic middle school math assignments. In *Educational Data Mining 2014*, 2014. 1

[24] Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, et al. Perception test: A diagnostic benchmark for multimodal video models. *Advances in Neural Information Processing Systems*, 36:42748–42761, 2023. 3

[25] Ethan Prihar, Aaron Haim, Tracy Shen, Adam Sales, Dongwon Lee, Xintao Wu, and Neil Heffernan. Investigating the impact of skill-related videos on online learning. In *Proceedings of the Tenth ACM Conference on Learning@ Scale*, pages 4–13, 2023. 1

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 2

[27] Gordon Ramsey. How to cook eggs benedict — gordon ramsay, 2018. 2

[28] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 2, 3

[29] Anh Truong, Peggy Chi, David Salesin, Irfan Essa, and Maneesh Agrawala. Automatic generation of two-level hierarchical tutorials from instructional makeup videos. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2021. 1

[30] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023. 2

[31] Saelyne Yang, Sangkyung Kwak, Juhoon Lee, and Juho Kim. Beyond instructions: A taxonomy of information types in how-to videos. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–21, 2023. 1

Vid2Coach: Transforming How-To Videos into Task Assistants

Supplementary Material

6. Study Materials

Video ID	Duration	Task	URL
V1	6:00	Chocolate Chip Cookies	[11]
V2	4:35	Eggs Benedict	[27]
V3	4:01	Mini Pavlovas	[14]
V4	5:06	Tortilla Pizza	[4]
V5	8:30	Strawberry Jam	[7]
V6	5:54	Dumpling	[12]
V7	6:56	Omelette	[8]
V8	3:59	Beef and Broccoli	[5]
V9	5:06	Mashed Potatoes	[15]
V10	7:28	Tiramisu	[6]
V11	3:54	Bread Flapjack	[16]
V12	3:19	Eggs Benedict	[3]

Table 3. Video Materials (V1-V10: Pipeline Evaluation, V11-V12: User Evaluation

7. Pipeline Overview

Vid2Coach's pipeline generates step instructions from a how-to video and supplements tips and workarounds from accessibility task resources (Figure 4), and monitors users' progress to provide proactive feedback (Figure 5).

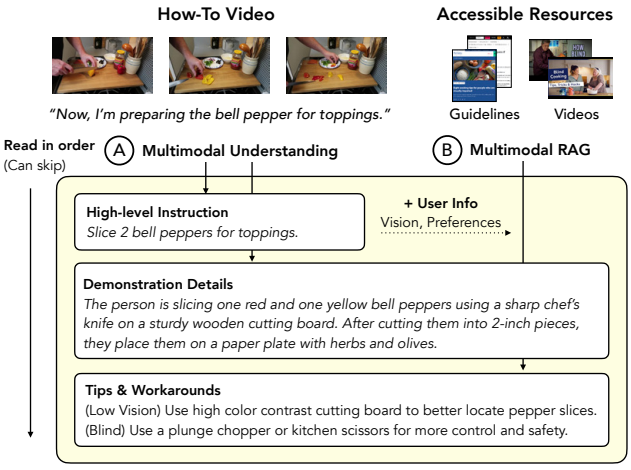

Figure 4. Vid2Coach generates step instructions from a how-to video with multi-modal understanding of narration and frames (A), and supplements tips and workarounds from accessibility task resources using RAG (B).

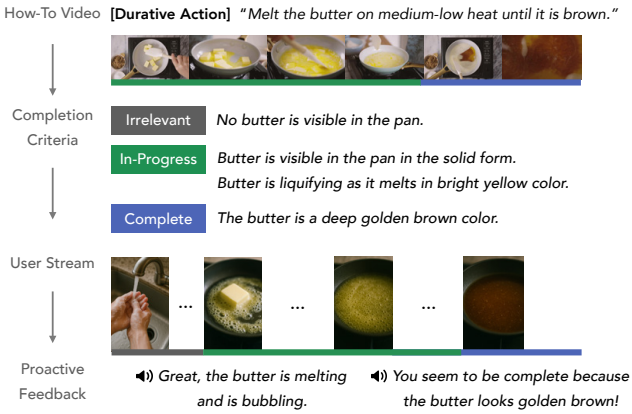

Figure 5. From the how-to video, Vid2Coach generates criteria for classifying user status into irrelevant, in-progress, and complete. As user performs the task, Vid2Coach monitors the progress and provide realtime feedback.

8. User Study

To evaluate how well Vid2Coach supports BLV users in following how-to video tasks, we conducted a within-subjects study with 8 BLV participants who completed end-to-end cooking tasks in their own kitchens. We compared Vid2Coach with a strong baseline of participants' existing practices for following how-to videos. In the baseline, we provided the original how-to video and its transcript, and allowed participants to use any existing human-powered or AI-powered visual assistance applications they typically use (Figure 6).

Figure 6. In the baseline condition, participants used their phones to listen to the video instructions (A) or call visual interpreters to get feedback (B). In the Vid2Coach condition, participants wore Meta glasses and used free-form speech to interact with the system.

9. Prompt Engineering

Prompts used in Vid2Coach's pipeline for identifying steps and actions in how-to videos (Table 4), classifying actions into types of punctual, iterative, and durative (Table 6, and running RAG for supplementing accessible strategies (Table 5).

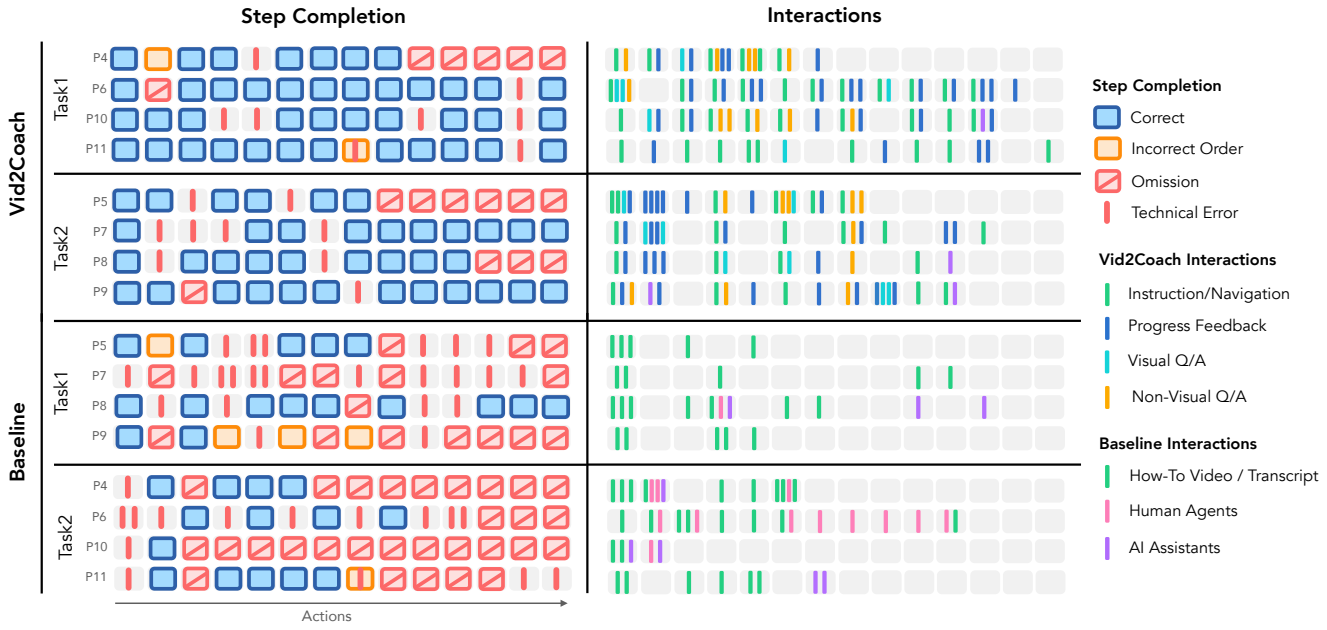

Figure 7. Step completion (left) and user-initiated interactions (right) visualized across participants, grouped by system condition (Vid2Coach vs. Baseline) and task (Task1 vs. Task2). Each row represents a participant, and each column a step in the task. For Vid2Coach, note that we only included user-initiated interactions and not Vid2Coach's consistent feedback for brevity.

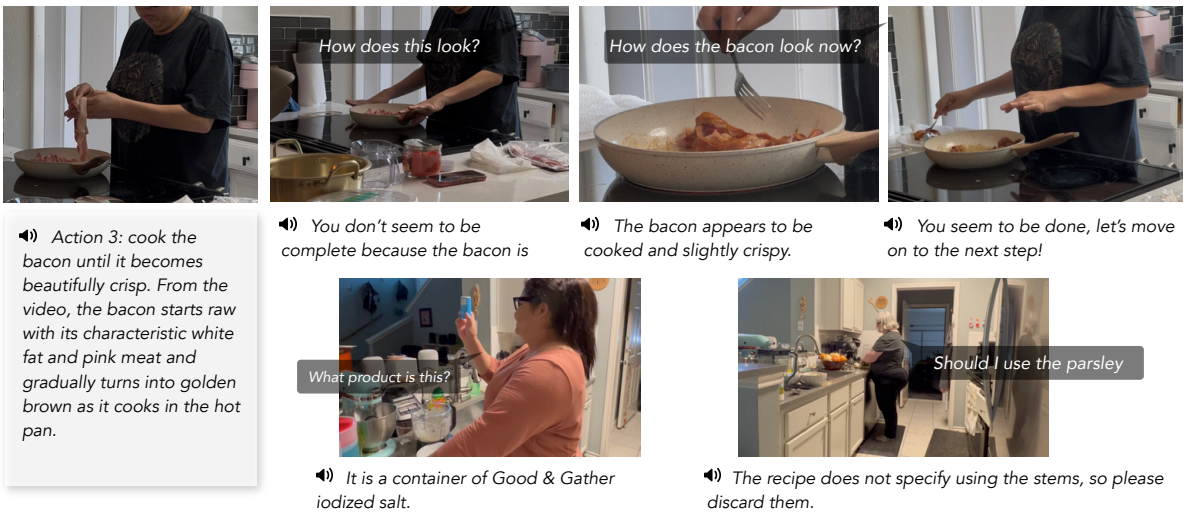

Figure 8. In the user study, participants used Vid2Coach to receive real-time feedback and ask free-form questions during cooking, helping them assess food readiness and ingredient use with confidence.

```
This is a transcript of a tutorial video: "{transcript_data}".
This is the metadata for this tutorial: "{metadata}".
Prioritize metadata if the images look different than the metadata.
Output a JSON file that segments this into high-level steps. For each step, include:
- step_name
- actions: a list of action objects containing:
  - instruction (single atomic verb)
  - supplementary (relevant tips, warnings, explanations)
  - start and end times
Use entries with the method role as the main instruction. Supplement with other roles (tip, warning,
explanation, etc).
Make instructions specific and actionable: include measurements (e.g., "Add 1.5 cups of ...") and
tools ("Mix using a spatula ...").
Important:
- Each instruction must be a clear, single-sentence action centered around one verb.
- Split instructions with multiple actions (e.g., \Add sugar and whisk" → two separate actions).
- Split iterative actions over different materials (e.g., \Add salt, sugar, and vanilla extract" →
three actions).
- Merge only if instructions describe the same event.
- If multiple actions are in one sentence, assign the same timestamp to each.
- Some steps may have no actions if no method-role content is present.
- A step's start time = first action's start; end time = last action's end; next step starts at
previous step's end.
Also include:
- tools: all tools used in this step
- materials: all materials/ingredients used in this step
- new_tools, new_materials: any tools/materials not used in the previous step
Do not hallucinate. Only use provided information.
Example step:
Step(
  step_name="Prepare Cookie Dough",
  actions=[
    Action(
      instruction="Add 1 cup of flour into the bowl.",
      supplementary=["Use precise measurements for the best results."],
      start=0.0,
      end=5.0
    ),
    Action(
      instruction="Mix the mixture with a spatula until no residue flour is visible.",
      supplementary=["Hold the bowl with the other hand for stability."],
      start=5.0,
      end=10.0
    ),
    Action(
      instruction="Let the dough rest for 30 minutes.",
      supplementary=["Resting the dough helps improve the texture of the cookies."],
      start=10.0,
      end=40.0
    )
  ],
  tools=["Cup", "Spatula", "Mixing bowl"],
  materials=["Flour"],
  new_tools=["Spatula"],
  new_materials=["Flour"],
  start=0.0,
  end=40.0
)
```

Table 4. Pipeline prompt for identifying steps and actions in how-to videos.

```
Generate response to the following query with the given context. If there is no relevant information,
say ``I don't know''.
User_info: {user_info}
Query: User is currently performing {action}, what are useful tips and workarounds?
Context: {context} # Relevant text chunk found across Top-3 documents based on user query
Response:
```

Table 5. Retrieval augmented generation prompt used for supplementing accessible strategies.

```
This is information about a tutorial video:  "{step}" .  Output a JSON that consists of the following
attributes:
tools, materials, actions.
For each action, specify an action type between:  punctual, iterative, and durative.
Punctual actions are brief and occur at a specific moment (e.g., "Put 1 cup of flour").
Iterative actions involve repetition or multiple quantities (e.g., "Add 2 rounded teaspoons").
Durative actions extend over time and involve continuous motion (e.g., "Whisk the mixture").
For each action, specify:
in_progress_criteria | visual indicators the action is ongoing;
completion_criteria | visual signs that the action is finished;
mistake_criteria | possible visual errors;
nonvisual_completion_criteria | (optional) sensory cues for completion (e.g., "feels crispy").
Note:
- Punctual actions should not include in_progress_criteria.
- completion_criteria should be grounded in the instruction (e.g., "until brown").
- in_progress_criteria should not overlap with completion_criteria.
- Only use information provided.  Do not hallucinate.
Also, extract tools and materials used in this step.  If available, include precise amounts from:
1/2 cup white sugar, 1/2 cup dark brown sugar, 1 egg, 1 tsp vanilla, 1/2 tsp salt (kosher),
1/2 tsp baking soda, 1 1/3 cups AP flour, 1 cup large chocolate chips.
# Example instantiation:
example_step = Step(
    tools=['whisk', 'bowl'],
    materials=['1 1/3 cups AP flour', '1/2 cup white sugar', 'butter'],
    actions=[
        Action(
            instruction='Put 1 cup of flour into the bowl.',
            video_description='The person scoops all-purpose flour into a shiny stainless steel 1-cup
measuring cup...'
            type='punctual',
            completion_criteria=['The flour is visible in the bowl.'],
            mistake_criteria=['Flour spills outside the bowl.'],
        ),
        Action(
            instruction='Add 3 eggs into the mixture.',
            video_description='The person gently cracks three fresh eggs...'
            type='iterative',
            in_progress_criteria=['One or two eggs are visible in the bowl, but not all three.'],
            completion_criteria=['All three eggs are visible in the bowl.'],
            mistake_criteria=['More than three eggs added', 'Eggshell is visible.'],
        ),
        Action(
            instruction='Whisk the mixture until it is smooth.',
            video_description='The person holds a ceramic bowl steady with one hand while whisking...'
            type='durative',
            in_progress_criteria=['The whisk is moving through the mixture.'],
            completion_criteria=['The mixture looks smooth and consistent.'],
            nonvisual_completion_criteria=['Mixture feels smooth to the touch.'],
            mistake_criteria=['Mixture is lumpy or too runny.'],
        )
    ]
)
```

Table 6. Pipeline prompt for classifying actions into punctual, iterative, durative actions and generating completion criteria.