Supplementary of Demo2Tutorial: From Human Experience to Multimodal Software Tutorials

Zechen Bai, Zhiheng Chen, Yiqi Lin, Kevin Qinghong Lin,
Difei Gao, Xiangwu Guo, Xin Wang, Mike Zheng Shou✉
Show Lab, National University of Singapore

1. More Analysis

1.1. Compression Analysis

Essentially, the Demo2Tutorial pipeline is a knowledge compression process that transforms lengthy demonstrations into concise tutorials. The compression happens in three key stages:

- **Stage 1 (Action Parser):** HE-Recorder captures synchronized raw video frames and action logs. The Action Parser then performs action-based filtering, reducing thousands of raw frames to a compact sequence of trace steps by identifying and extracting only action-relevant frames.
- **Stage 2 (Step Planner):** The Planner performs semantic compression by abstracting the trace actions into hierarchical task graphs representing high-level goals and subgoals, further reducing the number of steps while refining the semantic granularity.
- **Stage 3 (Tutorial Composer):** The Composer applies intelligent key-frame selection through multi-dimensional scoring (text relevance, image sharpness, motion stability, temporal proximity), producing the final compressed tutorial with optimal visual-textual alignment.

Figure 1 quantifies the compression achieved across 110 demonstrations from TutorialBench. On average, a demonstration contains 1,208 video frames (at 30 FPS, approximately 40 seconds), which is compressed to 13.07 trace steps by the Action Parser (92.46× compression), then to 3.93 draft steps by the Planner (additional 3.33× compression), and finally to 3.71 tutorial steps by the Composer (additional 1.06× compression), achieving an overall 325.71× compression ratio. The compression rate varies across software applications: video editing software like After Effects (539.43×) and Premiere Pro (378.10×) achieve higher compression rates due to longer demonstration videos with repetitive operations, while productivity software like Word (270.01×) and Excel (290.93×) have lower compression rates as their demonstrations are typically shorter and more concise. This multi-stage compression not only reduces storage and computational costs but also distills raw experience into pedagogically effective knowledge representations suitable for both human learning and agent training.

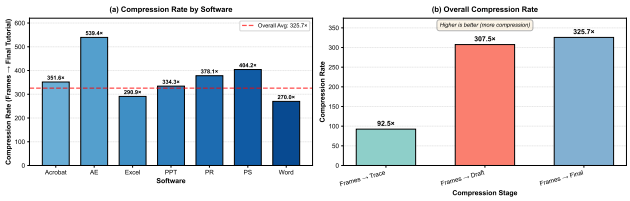

Figure 1. **Compression analysis across the Demo2Tutorial pipeline.** (a) Final compression rate (frames to final tutorial) for each software application, showing variation across different software complexity levels. After Effects achieves the highest compression (539.43×) while Word has the lowest (270.01×), with an overall average of 325.71×. (b) Stage-wise compression rates demonstrate that the majority of compression occurs in Stage 1 (Action Parser: 92.46×), followed by Stage 2 (Planner: 3.33×) and Stage 3 (Composer: 1.06×).

1.2. Results with Different Backbone

In the main paper, we use GPT-4o as the backbone model due to its strong multimodal capabilities and cost-effectiveness. In Tab. 1, we further evaluate our framework using Qwen-VL-32B-Instruct as an alternative backbone. Results show that Qwen achieves lower overall performance than GPT-4o (79.5 vs. 86.2), with the most significant gap in the Conciseness dimension (50.0 vs. 70.8). Despite this performance gap, Qwen's results remain competitive with human-authored tutorials (79.5 vs. 79.1), demonstrating that our agentic framework design can effectively enhance the capabilities of weaker MLLMs and produce high-quality tutorials even with less capable backbone models.

1.3. Component Ablation on TutorialBench

We perform component-wise ablations to quantify the contribution of each key design in Demo2Tutorial on a subset of TutorialBench. The subset consists of 21 samples,

✉Corresponding Author.

Table 1. Tutorial Generation Quality Evaluation with Different Backbone.

	Content Score				Visual Score			
Framework	Action.	Complete.	Concise.	Avg.	Annot.	Img Rel.	Avg.	**Overall**
GT (Human)	81.0	90.6	83.1	84.9	54.4	86.6	70.5	79.1
Vanilla Multi-Agent (GPT-4o)	71.1	88.9	59.0	73.0	51.3	81.5	66.4	70.3
Demo2Tutorial (GPT-4o)	90.5	92.3	70.8	84.5	83.3	94.0	88.7	86.2
Demo2Tutorial (Qwen)	87.2	82.0	50.0	73.1	85.4	92.7	89.1	79.5

Table 2. **Contribution of each component.** Ablations on a subset of TutorialBench using VLM-as-Judge scores ($\times 100$). The sharp degradation in Annotation Quality when removing Visual Highlight validates the necessity of adaptive visual guidance.

Setting	**Act.**	**Comp.**	**Conc.**	**Avg. Content**	**Annot.**	**ImgRel**	**Avg.Visual**	**Overall**
GT (Human)	77.6	90.5	84.8	84.3	45.2	86.7	66.0	77.0
Demo2Tutorial	91.4	93.3	84.8	89.8	85.2	96.2	90.7	90.2
w/o Task Hierarchy	91.0	86.7	64.8	80.8	83.8	96.2	90.0	84.5
w/o Visual Highlight	85.2	79.5	73.3	79.4	3.8	89.0	46.4	66.2
w/o Iterative Refinement	91.0	91.9	75.2	86.0	85.2	95.2	90.2	87.7
w/o Key-Frame Selection	89.0	91.0	76.2	85.4	86.2	95.2	90.7	87.5

3 from each software respectively. Specifically, we construct four variants by removing one component at a time under the same evaluation protocol: *(i) w/o Task Hierarchy* removes hierarchical abstraction, producing flat step lists without chapter/goal structuring; *(ii) w/o Visual Highlight* disables adaptive visual annotation and highlighting on key-frames; *(iii) w/o Iterative Refinement* removes the actor-critic refinement loop in the Step Planner; and *(iv) w/o Key-Frame Selection* replaces score-based key-frame selection with a simpler alternative without multi-factor scoring. We then evaluate each variant using VLM-as-Judge on the same five dimensions. Results are reported in Tab. 2.

We observe that removing Visual Highlight causes a drastic collapse in Annotation Quality (85.2→3.8) and Overall score (90.2→66.2), showing that adaptive visual highlighting is essential for producing learnable tutorials. Removing Task Hierarchy mainly degrades Conciseness (84.8→64.8) and Overall (90.2→84.5), indicating that hierarchical organization is crucial for avoiding verbose or poorly structured instructions. Removing Iterative Refinement or Key-Frame Selection yields smaller but consistent drops in Overall score (90.2→87.7/87.5), suggesting both refinement and frame selection improve instruction quality and image-text alignment.

2. Experiment Details

2.1. VLM-as-Judge

Evaluating tutorial quality presents unique challenges that traditional NLP metrics (e.g., BLEU, ROUGE) cannot adequately address. First, tutorials are inherently multimodal. Effective evaluation must jointly assess both textual instructions and visual components (screenshots, annotations). Second, tutorial quality depends on pedagogical effectiveness rather than mere surface-level similarity to reference text: a tutorial with different wording but clearer instructions may be superior. Third, human evaluation, while reliable, is prohibitively expensive and non-scalable for iterative development and large-scale benchmarking. Recent advances in Vision-Language Models (VLMs) demonstrate strong capabilities in understanding multimodal content and making nuanced judgments, making them suitable candidates for automated tutorial evaluation.

Our VLM-as-Judge protocol is grounded in two core principles that capture the dual nature of software tutorials:

Content Score evaluates the instructional quality of textual descriptions across three dimensions:

- **Actionability**: Can users successfully execute the described operations based solely on the provided instructions? This measures clarity and specificity. See Fig. 11 for the prompt.
- **Completeness**: Are all necessary steps included without missing critical operations? This measures information coverage. See Fig. 12 for the prompt.
- **Conciseness**: Are instructions clear and direct without unnecessary verbosity? This measures pedagogical efficiency. See Fig. 13 for the prompt.

Visual Score evaluates the effectiveness of visual components in supporting comprehension:

- **Annotation Quality**: Are visual markers (arrows, circles, highlights, magnifiers) appropriately applied to guide user attention to relevant UI elements? See Fig. 14 for the prompt.
- **Image Relevance**: Do the selected screenshots accurately correspond to the described actions and capture the relevant UI state? See Fig. 15 for the prompt.

Table 3. **Human consistency validation for VLM-as-Judge.** Two annotators independently score 63 tutorials, and we compute Spearman's ρ between human and VLM-as-Judge scores.

Metric	Spearman's ρ
Inter-Rater Agreement	0.601
Rater-1 *vs.* VLM-as-Judge	0.675
Rater-2 *vs.* VLM-as-Judge	0.671
Averaged Raters *vs.* VLM-as-Judge	0.755

Human Eval Guideline

Using the provided materials, your goal is to complete a PowerPoint task: **Implement a Typewriter Animation in Microsoft PowerPoint**.

You will use two files:
1. **Asset file (PPT)** – the file you will modify: *link*
2. **Multimodal tutorial (or demonstration video)** – a guide showing how to complete the task: *link*

Instructions:
1. Confirm that you are ready to begin.
2. Open both the PPT asset file and the tutorial/video.
3. Complete the task by following the steps shown in the tutorial/video. You may navigate the material freely (pause, rewind, skip, etc.).
4. Notify us immediately when you have finished the task.

Tips:
1. You do not need to rush, but please avoid being excessively slow.
2. Stay focused and complete the task as naturally as possible.

Figure 2. Human evaluation guideline.

For each dimension, we prompt GPT-4o to provide a score from 0 to 1, along with brief justification. The final Content Score and Visual Score are computed as the average of their respective dimensions, and the Overall Score is the average across the five dimensions.

Human Consistency Validation To assess the validity of VLM-as-Judge as a scalable proxy for human judgment, we conduct a human consistency evaluation on a diverse subset of TutorialBench. We first sample 3 tasks from each of the 7 software applications ($7 \times 3 = 21$ tasks), and for each task collect three tutorial variants (Demo2Tutorial, official human tutorial, and a vision-based baseline), yielding $21 \times 3 = 63$ tutorials in total. Two independent annotators then score each tutorial on the same five dimensions as our VLM-as-Judge protocol following the guideline in Fig. 2. For analysis, we compute each rater's overall score by averaging the five dimension scores, and report Spearman's rank correlation (ρ) to measure agreement between human raters and VLM-as-Judge. As shown in Tab. 3, the two raters have strong agreement ($\rho = 0.601$), and the averaged human scores correlate well with VLM-as-Judge ($\rho = 0.755$), supporting the reliability of our evaluation protocol.

Input Tok. (K)	Output Tok. (K)	Time (s)	Cost (USD)
259	3	368	1.36

Table 4. **Runtime and cost.** Average end-to-end runtime and estimated API cost per tutorial, averaged across TutorialBench.

2.2. Runtime and Cost

We report the average runtime and estimated API cost of generating one tutorial using Demo2Tutorial. We measure wall-clock time for the full generation pipeline after a demonstration is recorded (Action Parser $\rightarrow$ Step Planner $\rightarrow$ Tutorial Composer). We also aggregate the total number of input/output tokens consumed across all model calls in the pipeline, and estimate API cost by summing token usage under the provider pricing used at the time of our experiments. Tab. 4 reports the results.

2.3. OSWorld Experiment

To investigate whether multimodal tutorials can serve as effective external knowledge for enhancing GUI agent planning capabilities, we conduct experiments on the OSWorld benchmark [2], focusing on two representative application domains: Chrome (web browser) and VLC (media player).

We recruit human experts familiar with Chrome and VLC to execute a curated set of tasks from the OSWorld benchmark suite. Specifically, we select 17 tasks for Chrome and 14 tasks for VLC, covering diverse operations such as browser configuration, media playback settings, and interface customization. The complete task lists are reported in Fig. 16 and Fig. 17. During task execution, the expert's interactions are captured using our HE-Recorder, which synchronously records screen video at 30 FPS and logs all low-level user actions (mouse clicks, keyboard inputs, window operations). The recorded demonstrations are then processed through the full Demo2Tutorial pipeline to generate multimodal tutorials for each task.

We follow the Agent-S3 [1] as the downstream GUI agent framework, which provides a modular architecture for planning, grounding and execution. The generated tutorials are integrated as contextual knowledge into the agent's planner module. We evaluate two strong planning models: *o4-mini* and *GPT-5*, both serving as the planning backbone within GUI Agent. Performance is measured as the task success rate. To analyze the contribution of different tutorial modalities, we conduct ablation studies comparing four levels of contextual supervision:

- **Baseline**: Prompt-only, without any tutorial guidance.
- **+Text**: Incorporating textual step descriptions from tutorials.
- **+Image**: Incorporating visual screenshots from tutorials.
- **+Tutorial**: Incorporating full multimodal tutorials.

Each configuration is evaluated across all tasks within each domain, and results are averaged to obtain domain-specific success rates.

2.4. Human Evaluation

We recruit 20 participants and randomly assign them into two groups (10 per condition). The task involves implementing a seldom-used animation effect in Microsoft PowerPoint, specifically, creating a custom motion path animation with timing adjustments. This task is intentionally chosen to be non-trivial, requiring multiple steps that are not immediately obvious to novice users. Participants in the *Demo Video* condition watch the raw screen recording demonstration, while those in the *Tutorial* condition study the image-text interleaved tutorial generated by Demo2Tutorial. After the learning phase, participants attempt to complete the task independently while being timed. All participants successfully completed the task, confirming that both learning materials are effective. The study protocol is shown in Fig. 2.

3. Qualitative Examples

3.1. Examples for Each Software

In Figs. 3 to 8, we show qualitative examples of our generated tutorials across different software.

3.2. Failure Cases

Fig. 9 and Fig. 10 show two representative failure cases of our generated tutorials. First, Fig. 9 shows that the generated tutorial only contains the instruction text of inserting a worksheet, while the original video demonstration includes both how to insert and delete a worksheet, as reflected in the image. This failure happens in the Planner stage, where the agent tries to condense the raw actions into hierarchical task graphs representing goals and steps, but overlooks the deletion operations. Second, Fig. 10 shows that the visual guidance is inconsistent with the text description. The screenshots show the Morph animation, but the text description is about the Fade animation. This failure happens in the Action Parser stage, where the agent tries to parse the raw actions into semantic descriptions, but fails to correctly recognize the action area and misinterprets the Morph animation as a Fade animation.

4. Future Works

Building upon the current Demo2Tutorial framework, we identify several promising research directions that could further enhance the framework's capabilities and broaden its applicability.

While the current framework leverages HE-Recorder's synchronized action logs for precise temporal grounding, an exciting future direction is to develop vision-based action inference models that can automatically reconstruct low-level operations (clicks, keystrokes, gestures) purely from consecutive video frames. Such capability would enable the framework to process arbitrary screen recordings from the internet, dramatically expanding the scalability and applicability of tutorial generation to the vast repository of existing demonstration videos without requiring specialized recording tools. This research direction would bridge computer vision and human-computer interaction, advancing the state-of-the-art in visual action understanding.

The Demo2Tutorial pipeline achieves substantial knowledge compression (325.71× on average), transforming lengthy demonstrations into concise tutorials. A promising future direction is to develop intelligent pre-screening mechanisms that can identify high-information segments before the full parsing stage, enabling adaptive processing strategies. For instance, visual similarity detection could cluster redundant frames, while saliency estimation could prioritize segments with significant UI state changes. Such mechanisms would reduce computational overhead for long workflows while preserving tutorial quality, making the framework more efficient and cost-effective for processing extensive demonstration videos.

References

[1] Gonzalo Gonzalez-Pumariega, Vincent Tu, Chih-Lun Lee, Jiachen Yang, Ang Li, and Xin Eric Wang. The unreasonable effectiveness of scaling agents for computer use. *arXiv preprint arXiv:2510.02250*, 2025. 3

[2] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024. 3

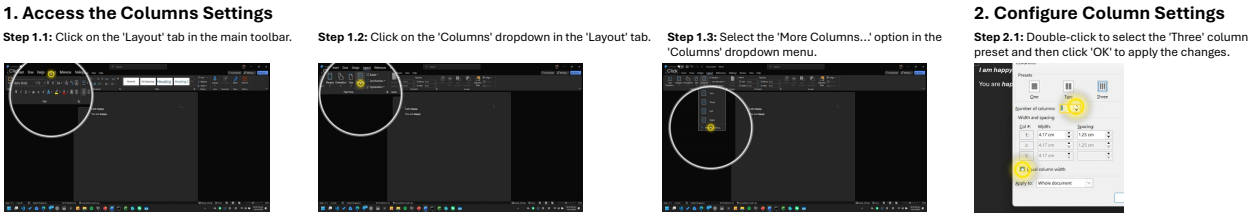

Figure 3. Example of Word tutorial.

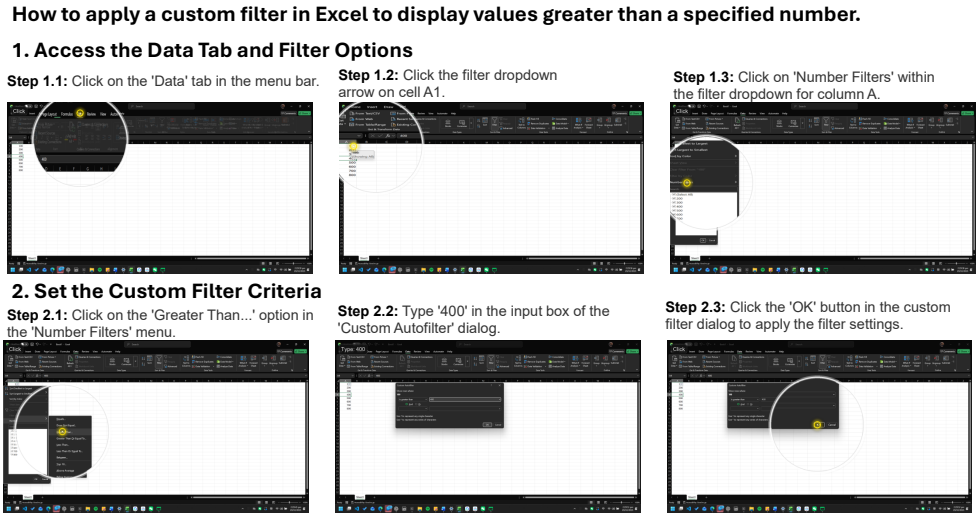

Figure 4. Example of Excel tutorial.

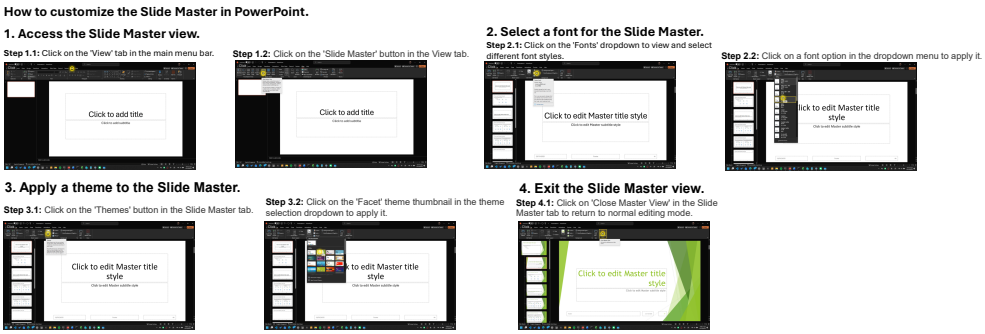

Figure 5. Example of PowerPoint tutorial.

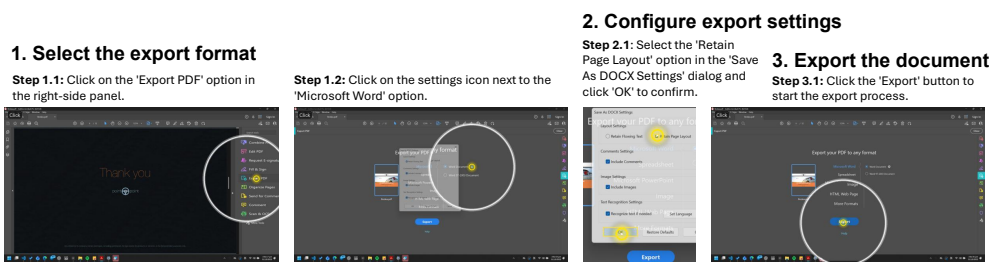

Figure 6. Example of Acrobat tutorial.

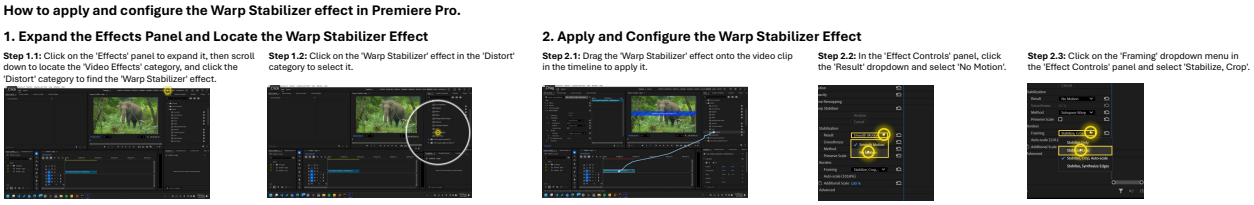

Figure 7. Example of Premiere Pro tutorial.

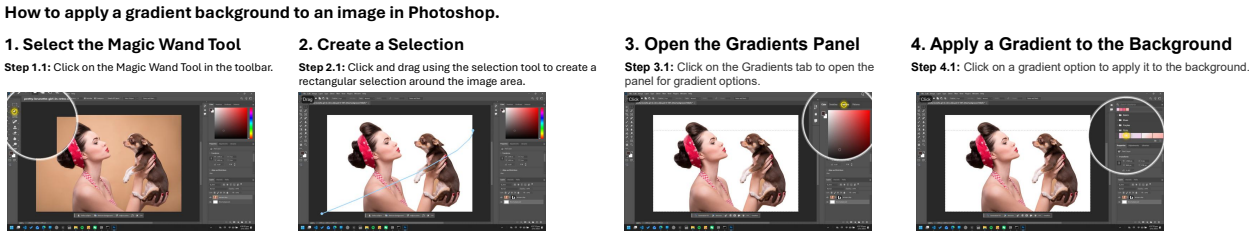

Figure 8. Example of Photoshop tutorial.

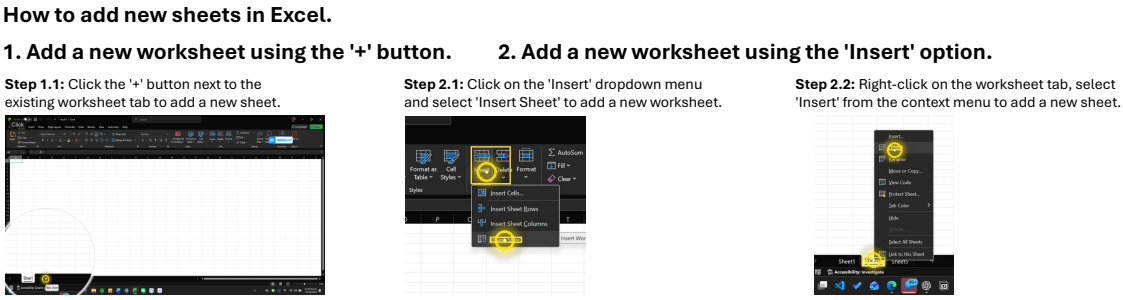

Figure 9. Example of failed Excel tutorial. The original video demonstration including both how to insert and delete a worksheet in Excel, but the generated tutorial only contains the instruction of inserting a worksheet.

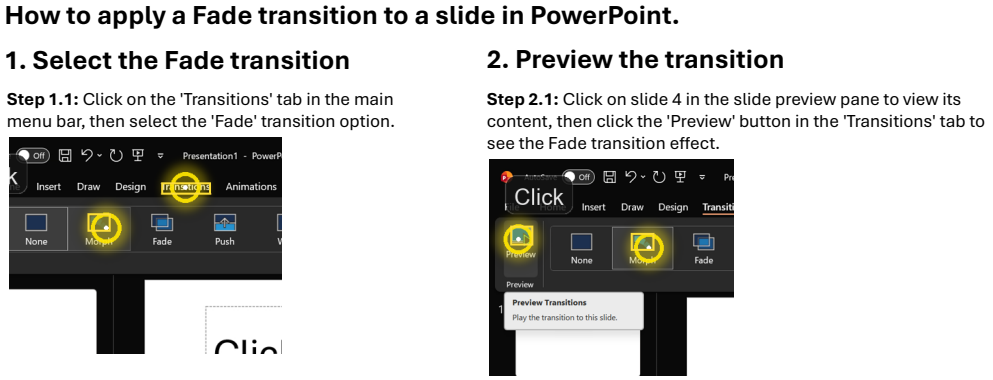

Figure 10. Example of failed PPT tutorial. The agent underlying MLLM fails to correctly recognize the action area and misinterprets the Morph animation as a Fade animation.

VLM-as-Judge: Actionability

```
system_prompt: |
  You are an expert evaluator for tutorial quality. Your task is to evaluate how actionable, clear, and practical the tutorial instructions are.

template: |
  You are evaluating a tutorial based on the **Actionability** dimension.

  ## Evaluation Criteria:

  **Actionability** measures whether the tutorial instructions are:
  - **Concrete and Specific**: Provides exact click locations, button names, menu paths, and UI element identifiers, use terms in quotes
  - **Clear and Unambiguous**: Uses precise language that is easy to understand without multiple interpretations
  - **Observable**: Specifies what the user should see at each step, including visual feedback
  - **Executable**: Can be followed by a user without guessing or requiring additional interpretation
  - **Direct**: Uses imperative verbs and direct instructions (e.g., "Click the Save button" not "You might want to save")
  - **Well-Structured**: Breaks down complex operations into clear, manageable substeps with consistent terminology
  - **Avoids Vagueness**: No abstract or ambiguous guidance (e.g., avoid "adjust as needed", "configure properly")

  ## Scoring Guidelines (0-1 scale):

  - **Score 0.9-1.0**: Every instruction is highly specific, crystal clear, and can be followed without any ambiguity. Excellent element identification and step breakdown. Points out the specific element names. Use terms in quotes.
  - **Score 0.7-0.8**: Instructions are mostly actionable and clear with only minor vagueness or ambiguity
  - **Score 0.5-0.6**: Instructions are generally followable but require user interpretation in places, or lack some specificity. Does not use specific element names in quotations.
  - **Score 0.3-0.4**: Instructions are often vague, unclear, or require significant guessing to execute. Does not use specific element names  in quotations.
  - **Score 0.0-0.2**: Instructions are too abstract, unclear, or vague to be reliably actionable

  ## Your Task:

  Review the provided tutorial steps and images. Evaluate how actionable, specific, and clear the instructions are.

  Provide your evaluation in the following JSON format:
  ```json
  {
    "score": <float between 0 and 1>,
    "reason": "<brief explanation of your score, citing specific examples of strong actionability/clarity or weak vagueness/ambiguity>"
  }
  ```
```

Figure 11. VLM-as-Judge prompt for Actionability.

VLM-as-Judge: Completeness

```
system_prompt: |
  You are an expert evaluator for tutorial quality. Your task is to evaluate the completeness of tutorial content.

template: |
  You are evaluating a tutorial based on the **Completeness** dimension.

  ## Evaluation Criteria:

  **Completeness** measures whether the tutorial:
  - Covers all necessary steps to accomplish the task
  - Has no missing critical information
  - Sometimes the tutorial focuses on a specific function, so there is no need to consider functions outside the scope of the tutorial. Only focus on the main goal of the tutorial. Do not suggest additional features or functions that are not part of the tutorial's main objective.
  - Do not ask for a confirmation step. Do not ask for any extra steps or prerequisites!! For example, if the tutorial is about 'how to insert a checkbox', do not ask for further operations to the checkbox after inserting it.

  ## Scoring Guidelines (0-1 scale):

  - **Score 0.9-1.0**: Tutorial is comprehensive, covers all necessary steps
  - **Score 0.7-0.8**: Tutorial is mostly complete with only minor gaps in coverage
  - **Score 0.5-0.6**: Tutorial covers the main steps but misses some important details
  - **Score 0.3-0.4**: Tutorial has significant gaps, missing critical steps or information
  - **Score 0.0-0.2**: Tutorial is severely incomplete, missing major portions of necessary content

  ## Your Task:

  Review the provided tutorial steps and images. Evaluate whether the tutorial provides complete coverage of the task.

  Provide your evaluation in the following JSON format:
  ```json
  {
    "score": <float between 0 and 1>,
    "reason": "<brief explanation of your score>"
  }
  ```
```

Figure 12. VLM-as-Judge prompt for Completeness.

VLM-as-Judge: Conciseness

```
system_prompt: |
  You are an expert evaluator for tutorial quality. Your task is to evaluate the conciseness of tutorial content, with a
strict focus on eliminating redundancy and verbosity.

template: |
  You are evaluating a tutorial based on the **Conciseness** dimension.

  ## Evaluation Criteria:

  **Conciseness** measures whether the tutorial:
  - **Maximizes Information Density**: Every sentence provides essential, non-redundant information
  - **Eliminates Repetition**: No repeated explanations, duplicate instructions, or restated concepts
  - **Avoids Verbosity**: Uses the minimum words necessary to convey the instruction clearly
  - **Removes Filler**: No unnecessary phrases like "Now we will...", "Let's go ahead and...", "Simply just..."
  - **Merges Related Steps**: Combines closely related operations instead of breaking them into excessive substeps
  - **Penalizes Over-Explanation**: Avoids explaining obvious outcomes or providing excessive context

  ## Scoring Guidelines (0-1 scale):
  - **Score 0.9-1.0**: Extremely concise with maximum information density. Every word is necessary. No
redundancy whatsoever. Instructions are brief yet complete.
  - **Score 0.7-0.8**: Mostly concise with minimal redundancy. Occasional verbose phrases but generally efficient.
  - **Score 0.5-0.6**: Moderately concise but contains noticeable redundancy, repetitive explanations, or
unnecessary elaboration.
  - **Score 0.3-0.4**: Verbose with significant redundancy. Multiple instances of repeated information, over-
explanation, or filler content.
  - **Score 0.0-0.2**: Extremely verbose and redundant. Excessive repetition, unnecessary details, and poor
information density.

  ## Your Task:
  Review the provided tutorial steps and images. {% if gt_main_goal %}**Pay special attention to whether each step
is necessary to achieve "{{ gt_main_goal }}"**. If a step does not serve this goal, it is redundant.{% else %}Strictly
evaluate the conciseness and penalize redundancy or verbosity.{% endif %}

  Provide your evaluation in the following JSON format:
  ```json
  {
    "score": <float between 0 and 1>,
    "reason": "<brief explanation of your score{% if gt_main_goal %}, specifically identifying which steps (if any) are
not necessary for achieving '{{ gt_main_goal }}'{% else %}, citing specific examples of redundancy, repetition, or
unnecessary verbosity if score is low; or highlighting excellent conciseness if score is high{% endif %}>"
  }
  ```
```

Figure 13. VLM-as-Judge prompt for Conciseness.

VLM-as-Judge: Visual Annotation Quality

```
system_prompt: |
  You are an expert evaluator for tutorial quality. Your task is to evaluate the quality of visual annotations in tutorial
images.

template: |
  You are evaluating a tutorial based on the **Visual Annotation Quality** dimension.

  ## Evaluation Criteria:

  **Visual Annotation Quality** measures whether the annotated images:
  - Use clear and visible visual indicators (arrows, boxes, highlights, circles)
  - Have annotations that are easy to distinguish from the background
  - Use appropriate colors that contrast well with the interface
  - Annotations are neither too large (obstructive) nor too small (hard to see)
  - Multiple annotations are clearly differentiated
  - Text labels are readable and well-positioned
  - Annotations accurately point to the relevant UI elements
  - Overall visual design is clean and professional

  ## Scoring Guidelines (0-1 scale):

  - **Score 0.9-1.0**: Annotations are exceptionally clear, well-designed, and enhance understanding perfectly
  - **Score 0.7-0.8**: Annotations are clear and effective with only minor visual issues
  - **Score 0.5-0.6**: Annotations are visible but have some clarity or design issues
  - **Score 0.3-0.4**: Annotations are hard to see, poorly positioned, or confusing
  - **Score 0.0-0.2**: Annotations are very poor quality, obstructive, or nearly invisible

  ## Your Task:

  Review the provided annotated tutorial images carefully. Focus on the visual quality of the annotations (arrows,
boxes, highlights, text overlays, etc.), NOT the content of the instructions.

  Provide your evaluation in the following JSON format:
  ```json
  {
    "score": <float between 0 and 1>,
    "reason": "<brief explanation of your score, describing the visual quality of annotations (color, size, clarity,
positioning)>"
  }
  ```
```

Figure 14. VLM-as-Judge prompt for Annotation Quality.

VLM-as-Judge: Visual Annotation Quality

````
system_prompt: |
  You are an expert evaluator for tutorial quality. Your task is to evaluate the quality of visual annotations in tutorial images.

template: |
  You are evaluating a tutorial based on the **Visual Annotation Quality** dimension.

  ## Evaluation Criteria:

  **Visual Annotation Quality** measures whether the annotated images:
  - Use clear and visible visual indicators (arrows, boxes, highlights, circles)
  - Have annotations that are easy to distinguish from the background
  - Use appropriate colors that contrast well with the interface
  - Annotations are neither too large (obstructive) nor too small (hard to see)
  - Multiple annotations are clearly differentiated
  - Text labels are readable and well-positioned
  - Annotations accurately point to the relevant UI elements
  - Overall visual design is clean and professional

  ## Scoring Guidelines (0-1 scale):

  - **Score 0.9-1.0**: Annotations are exceptionally clear, well-designed, and enhance understanding perfectly
  - **Score 0.7-0.8**: Annotations are clear and effective with only minor visual issues
  - **Score 0.5-0.6**: Annotations are visible but have some clarity or design issues
  - **Score 0.3-0.4**: Annotations are hard to see, poorly positioned, or confusing
  - **Score 0.0-0.2**: Annotations are very poor quality, obstructive, or nearly invisible

  ## Your Task:

  Review the provided annotated tutorial images carefully. Focus on the visual quality of the annotations (arrows, boxes, highlights, text overlays, etc.), NOT the content of the instructions.

  Provide your evaluation in the following JSON format:
  ```json
  {
    "score": <float between 0 and 1>,
    "reason": "<brief explanation of your score, describing the visual quality of annotations (color, size, clarity, positioning)>"
  }
  ```
````

Figure 15. VLM-as-Judge prompt for Image Relevance.

OS-World: Google Chrome

- Find Dota 2 game and add all DLC to cart.
- Hey, I need a quick way back to this site. Could you whip up a shortcut on my desktop for me using Chrome's built-in feature?
- I am looking for an website address I accessed a month ago, but Youtube websites which take almost all of my browsing history are interrupting my search. This is too annoying. I want to remove all my Youtube browsing history first to facilitate my search. Could you help me clear browsing history from Youtube?
- Create a list of drip coffee makers that are on sale and within $25-60 and have a black finish.
- On next Monday, look up a flight from Mumbai to Stockholm.
- Please help me set Chrome to delete my browsing data automatically every time I close the browser.
- Browse list of Civil Division forms.
- Find discussions of community and open one with most replies.
- Show side effects of Tamiflu.
- Find the Next Available dates for Diamond.
- Find a Hotel in New York City with lowest price possible for 2 adults next weekend.
- Please help me find the score record for the 2019 Super Bowl in the NFL website.
- Can you make my computer bring back the last tab I shut down?
- Browse the natural products database.
- Find flights from Seattle to New York on 5th next month and only show those that can be purchased with miles.
- I want Chrome to warn me whenever I visit a potentially harmful or unsafe website. Can you enable this safety feature?
- Find flights from New York–Kennedy Airport to Chicago O'Hare Airport for tomorrow.

Figure 16. Task list of OS-World Chrome domain.

OS-World: VLC

- Can you disable the cone icon in the splash screen? I am tired of its skeuomorphic design.
- I am reading lecture note in PDF while a music video is running in VLC media player. But I find I need to switch to the player every time I need to pause/start.Could you help me change the setting to allow pausing the video using keyboard shortcut without minimizing the PDF reader? I want to focus on the lecture note and don't be disturbed by the app switching.
- Could you play the music video that's saved on my desktop for me via vlc?
- Help me modify the folder used to store my recordings to Desktop
- Can you enable fullscreen mode in VLC so that the video fill up the whole screen?
- Could you convert the song from this music video as an MP3 file? I'd like to have it on my device to play whenever I want. Please save the file just on the desktop and title the file \"Baby Justin Bieber.mp3.\" I really appreciate your help!
- I like watching movies (using VLC) on my laptop and sometimes the volume is too low for my taste even when the volume in VLC is set to the maximum of 125% on the volume control. Can you increase the max volume of the video to the 200% of the original volume?
- Could you help me hide the bottom toolbar in VLC Media Player when watching in window mode? I often multitask on my computer, and the persistent toolbar in VLC is very distracting.
- Hey, could you turn this video the right way up for me? And once it's flipped around, could you save it for me with the name '1984_Apple_Macintosh_Commercial.mp4' on the main screen where all my files are?
- Can you start streaming the video from this link for me? https://devstreaming-cdn.apple.com/videos/streaming/examples/img_bipbop_adv_example_fmp4/master.m3u8
- Automatically adjust the brightness and contrast of this video to match my room's lighting.
- Can you change the color of the volume slider to black-ish color? I often use the player in a low-light environment, and a darker color scheme would be less straining on my eyes, especially during nighttime usage.
- Make this part of the video my computer's background picture
- I want to watch two or more videos in same time on VLC. I tried to run multiple instances of VLC. It worked but can't play videos on those new instances. When I play video it plays on first instance instead of new instance.\nIs there any way to solve this problem?
- Snap a photo of the current video scene, save it as 'interstellar.png', and put it on the Desktop, please.

Figure 17. Task list of OS-World VLC domain.