FIGMATRACE: CAPTURING CREATIVE NUANCES IN HUMAN FIGMA DESIGN WORKFLOWS

Darshan Deshpande[*], **Yoshinari Fujinuma, Martyna Markiewicz, Devanshu Bansal**
Shivani Jain, Nicholas Saban, Chirag Maheshwari, Anand Kannappan
Patronus AI
{darshan, yoshinari.fujinuma, martyna, dev, shivani,
nicksaban, chirag.maheshwari, anand}@patronus.ai

ABSTRACT

Vision Language Models have recently shown improvements in several objective and verifiable domains such as object detection but continue to underperform on subjective and creative design tasks. A major contributor to this performance gap is the lack of high quality human workflow data that captures a diverse set of preferences and decisions that make human experts good at design tasks. In this work, we first define a unique, expert curated taxonomy of design skills and best practices which we further expand into a set of 126 open ended, subjective, long horizon tasks. Built on top of this and expert solutions, our dataset FIGMATRACE contains over 200 hours of human captured video data converted into 3469 design trajectories using a novel design phase-based method. We use our dataset to train four models and show that training on FIGMATRACE leads to a performance improvement comparable to frontier closed models such as CLAUDE-OPUS-5 and GPT-5.6-SOL on four out of distribution agentic GUI environments. We further perform a useful ablation to attribute these performance improvements to a design phase-based video to trajectory conversion which outperforms prior length-based conversion approaches. Finally, we perform a qualitative analysis on the best performing QWEN3.8-27B outputs to better correlate performance improvements to FIGMATRACE's trends. We open source our dataset and the best QWEN3.8-27B model for the community[1].

1 INTRODUCTION

Vision Language Models (VLMs) are popularly used for several verifiable tasks such as document understanding (Ding et al., 2026; Wang et al., 2025a), robotics (Sapkota et al., 2025; Zhang et al., 2025) and computer use (Tang et al., 2025; Xie et al., 2025). On subjective and non-verifiable tasks, VLMs have struggled to capture human nuances such as understanding emotions (Bhattacharyya & Wang, 2025), humor and understanding of figurative meaning (Zhou et al., 2026; Ryan et al., 2025) and design taste (An et al., 2026). Recent works have attempted to address this issue through human preference alignment (Peng et al., 2025; Liao et al., 2025) and reinforcement learning based objectives (Li et al., 2025b; Wu et al., 2026), however, these techniques rely heavily on the availability of data that surfaces such preferences. This is worsened by the unavailability of high quality design datasets used to train *tasteful* and *nuanced* design agents.

To address this lack of data, prior works such as Gui et al. (2026) explored using existing Figma designs and working backwards to create automated annotation processes for data but the validation of such work is difficult and ambiguous due to the lack of exhaustive quality guidelines. Exploring verifiability, Russo et al. (2025) use existing HTML versions of pages to convert them to Figma compatible JSONs that a model can be trained to generate (human grounding for LLMs is difficult + validation is hard). Kanapathipillai & Priyankara (2026) explore a parallel direction of automated creation of individual Figma component JSONs for the sake of reusability and scalability. Making

[*]Correspondence: darshan@patronus.ai

[1]https://huggingface.co/datasets/PatronusAI/figmatrace
https://huggingface.co/PatronusAI/Qwen3.8-27B-Figmatrace-SFT

(a) Set hexcode - `keyboard_type("00DC82")`

(b) Create style - `mouse_click(1134, 602)`

(c) Switch tab - `mouse_click(84, 28)`

(d) Select element - `mouse_click(786, 598)`

(e) Select element - `mouse_click(583, 372)`

(f) Resulting state

Figure 1: A sample FIGMATRACE trajectory excerpt taken from session `t1-112`, task involving replicating the Nuxt Website.

the process of web design agentic, Jeong et al. (2026) propose a harness for evaluating agents on design tasks but take no account of human taste involved in curating such tasks. Despite efforts on these fronts, evaluation of such data is increasingly difficult due to non-determinism of multimodal automated judges. Chandwani & Gupta (2026) propose a unique set of skills that can guide the evaluation process but the scope of such pre-defined skills is limited and does not generalize effectively to design applications.

To address these issues, we propose FIGMATRACE, a comprehensive dataset with 126 long horizon tasks, spanning across 8 realistic designer workflows, capturing a set of 10 high level expert curated skills split across 3469 trajectories. We first capture OS-level expert actions and screen captures, each working through unique task categories that cover both verifiable and open ended design tasks. Using this video dataset, we clean and thoroughly post process the recordings and corresponding actions to create a comprehensive set of ultra-long horizon workflow trajectories spanning up to 5 million tokens. To effectively train on these trajectories and retain the skills and design phases (such as creating components, reference gathering, etc) showcased in the trajectories, we create an expert-reviewer curated set of phase categories showcased in these workflows. We then use these categories to automatically extract and verify labels using GEMINI-3.6-FLASH at scale. Using our final dataset, we study the following research questions:

1. Does training with realistic human captured design workflows teach VLMs to be better at agentic navigation and design?
2. Does conversion of video data to trajectories benefit more from design-phase based trajectory curation as opposed to maximum context-length sharding for very long horizon tasks?
3. What patterns in FIGMATRACE influence qualitative performance improvements in models?

Through our results, we show that training VLMs on FIGMATRACE improves agent performance on several goal oriented and narrative oriented benchmarks including but not limited to GUI-Odyssey (Lu et al., 2025b), Mind2Web (Deng et al., 2023), VideoGUI (Lin et al., 2024), achieving an absolute increase of up to 46% on datasets such as AndroidControl (Li et al., 2024) over the corresponding baselines. Beyond this, we find that performance improvement brought about by our design phase-based trajectory curation approach leads to a 7.3% absolute increase in performance, showing that the model understands task intents much better as compared to maximum context-length based SFT where the intent becomes unclear due to sharding at inconsistent intervals. Finally, we perform human evaluation to study the source of performance boost on out-of-domain

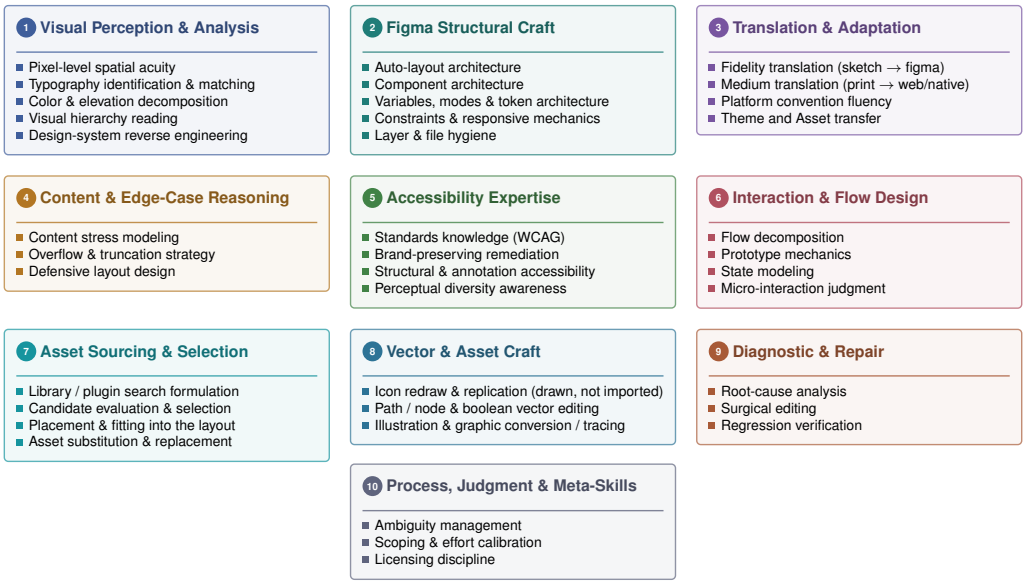

Figure 2: A comprehensive set of relevant skills that FIGMATRACE covers.

tasks and find that the dataset improves properties such as element selection accuracy and decisiveness of model decisions in undirected settings.

2 RELATED WORK

Design-to-code Rico (Deka et al., 2017) is one of the earliest works on creating a design-focused mobile screens dataset with view hierarchies, followed by Design2Code (Si et al., 2025) which curates real webpages for screenshot-to-HTML generation, WebSight (Laurençon et al., 2024) which scales the same formulation to two million synthetically rendered pages, and Gui et al. (2026) which extends it to Figma. In each case, the focus is on the final artifact rather than the trajectories for creating such artifacts, which lets an agent learn what a finished design looks like but not the sequence of decisions that produced it.

Screen Recordings for Computer Use Agents Exploiting screen recordings has long been studied, but datasets including screen recordings and aligned gold actions are scarce. Where recordings are unavailable, grounding data is instead synthesized by decomposing and recomposing interfaces (Xie et al., 2026), and evaluated on professional software by ScreenSpot-Pro (Li et al., 2025a). VideoAgentTrek (Lu et al., 2025a) uses public unlabeled screen recordings by first applying video-to-action mapping to detect actions on GUI to create synthetic agent trajectories. OpenCUA (Wang et al., 2025b) instead records annotators directly, capturing screen video, input events, and the accessibility tree, and converts them into gold state-action pairs augmented with synthesized reasoning.

Processing Screen Recordings Since raw screen recordings of human actions are often redundant and noisy, preprocessing is necessary to convert the recording into training data. VideoGUI (Lin et al., 2024) annotates instructional software video for evaluation at three levels: high-level planning, middle-level planning over action narrations, and atomic action execution. VideoAgentTrek (Lu et al., 2025a) instead detects individual GUI actions with tight temporal bounds and attaches a per-action rationale. Both are adequate when trajectories are short, but neither provides structure above the action, so intent boundaries become uncertain once a session runs for a longer horizon.

In summary, preprocessed screen recordings and actions showing the full expert trajectories are useful to train an agent from long-horizon sessions which require creative design skills. FIGMATRACE is the first dataset with pairs of gold action sequences and intent-segmented recordings of expert work for Figma.

Table 1: Taxonomy of Figma design task categories and their start-state creation methods.

ID / Task type	Start-state creation method
1. Pixel-perfect replication (open seeds)	Pull a page from the open-web whitelist. Capture full-page PNG at 1440 px (browser capture or SingleFile archive) and archive the URL, capture date, and license basis. Attach the one reusable instruction: "Replicate 1:1 in Figma with proper auto-layout."
2. Responsive / platform adaptation	Attach a per-target instruction template: desktop $\rightarrow$ 375 px mobile; web $\rightarrow$ tablet; print $\rightarrow$ web; web app $\rightarrow$ Android (Material).
3. Theming with variables	Instruction template on any cleared seed: "Produce the dark/light variant implemented via Figma variables/modes—no manual per-node recolor."
4. Sketch-to-Figma	Start with a sourced hand-drawn sketch (existing hand-drawn-to-website datasets are also available). SME builds the hi-fi design from the photo. Both artifacts are owned via the contributor agreement.
5. Flaw injection $\rightarrow$ repair	Take an existing open-source Figma template and modify it to break a few things in the workflow.
6. Edge-content injection resilience	Test whether a layout survives content it was never designed for—most screens are only ever tested against clean, friendly demo data.
7. A11y remediation	Fix accessibility problems in the file.
8. Prototype wiring	From open-ended task generation, create a prototype of the website in Figma.

3 FIGMATRACE

This section describes the curation process of FIGMATRACE along with design decisions and expert feedback loops.

3.1 SKILL TAXONOMY AND TASK CURATION

Taxonomy of Creative Skills To ground FIGMATRACE in real life creative workflows that human experts follow, we tasked three experts to create a comprehensive taxonomy of creative and nuanced human skills that designers utilize in their daily workflows. Figure 2 showcases the taxonomy capturing 10 unique skills that cover best practices of Figma and are, in isolation or jointly, applicable to most Figma and non-Figma design applications. These cover core abilities of experts including visual perception, feature translation and adaptation, asset sourcing and creation, debugging, flow designing, accessibility best practices, content based reasoning and more. To the best of our knowledge, this is the most comprehensive taxonomy of design skills to date, thereby making FIGMATRACE unique and useful for the community.

Task Coverage Inspired by realistic designer workflows, we design a set of eight unique task categories that strictly require one or more of the skills above. Specifically, these are categorized into verifiable and non-verifiable tasks Table 1. Verifiable tasks include pixel perfect replication, repair of injected flaws, edge content resilience and a11y remediation that have deterministic solutions. On the other hand, tasks such as platform adaptation, theming, sketch to figma and prototype wiring capture nuance in workflows and hence outputs are dependent on SME biases, which in turn makes FIGMATRACE a rich dataset.

Cleaning and Processing Action Spaces On average, we observed, through deterministic action to frame mapping that 95% of actions captured during recordings were either random mouse movements or hover actions in the middle of the screen. Because our downstream agents use the Playwright MCP toolset,[2] we filter out all mouse movements except hover actions. The toolset clicks a target directly from its screen coordinates, so no intermediate cursor movement is needed to reach

[2] https://github.com/microsoft/playwright-mcp

it. We manually map all other OS level actions to the closest playwright-MCP action set. In some special cases such as when a render completes or when a plugin loads, screen state can change with no input. We insert an `observe` probe every 2 seconds inside any gap of more than 4 seconds, so environment transitions become first-class steps.

Frame extraction In this step we extract frames from the video that correspond to the filtered actions above. We do this in two passes to ensure consistency: the first pass decodes the entire video (up to 4.5 hours long) at 8 frames per second in 480×270 grayscale. For each candidate at time t this fixes two timestamps: before = $t - 0.15s$, and after = the first frame in $[t + 0.2, t + 2.0]$ where consecutive frames satisfy mean $|\Delta| < 0.75$ (the screen has settled). A fixed post-action offset is incorrect in this case since a menu settles in 0.25s on average (settle point found for 5,717/5,718 instances) and an image drop takes over a second. This pass acts as a proxy. The second pass extracts only those timestamps at full resolution, clustered into one decoder invocation per group, selecting exact frame indices so only the wanted frames are encoded.

Effect Filtering For each pair, `changed_fraction` = fraction of pixels whose max channel difference exceeds 6. Actions below a fraction of 5×10^{-4} are dropped as having no visible effect whereas `observe` probes need 2×10^{-2} to count as a scene change. This is the step that separates what the expert did from what changed the artifact.

Skill based phase segmentation We utilize Gemini-3.6-Flash [3] as the video segmentation model. The model categorizes the video without the action log and returns contiguous spans from the closed 11 phase labels as described in Table 4. This incentivizes teaching skills to the VLM instead of randomly sharding based on pauses in the video that can be a noisy signal. The shard count that Gemini-3.6-Flash produces has no principled value, so rather than tune it we run 3, 6 and 12 shardings and keep only boundaries that ≥ 2 of them place within $\pm 30s$. This forms meaningful skill based separation which teaches VLMs specific skills required to learn Figma best practices. Finally, we assign skill labels to each trajectory based on frequency since one trajectory can potentially have more than one skill. During this segmentation process, we observed that video resolution matters far more than model used for segmentation. Against a strong model at full resolution (Gemini-3.6-Flash): same model at low resolution scores Jaccard 0.244 / 21% dominant-skill agreement, while a weaker image model (Gemini-3-Pro as used by prior work) at full resolution scores 0.601 / 43%. Low resolution collapses to generic labels because panel and layer text becomes unreadable. At the end of the entire process, we achieve a total compaction of $179\times$ as compared to the raw OS events captured by the screen recorder.

4 Experimental Setup

4.1 Training Setup

To show performance improvements when training with FigmaTrace, we use the ms-swift training framework (Zhao et al., 2024) due to its strong support for long context training via context parallelization for VLMs. For showing consistent performance improvement we train Qwen3.6-35BA3B, Qwen3.8-27B, Gemma-4-31B and Muse-Glimmer-30B on 92,472 total actions sampled randomly from 35 sessions averaging at 47.6 hours of total human work done. A complete list of hyperparameters used can be found in the Appendix in Table 6. To further study the generalization that FigmaTrace brings, we evaluate our fine-tuned models on four different dataset combinations, covering multi-app, multi-viewport GUI navigation using GUI-Odyssey (Lu et al., 2025b), AndroidControl (Li et al., 2024) to evaluate the effect of instruction granularity on model performance, Mind2Web (Deng et al., 2023) to evaluate instruction grounding on open web data and VideoGUI (Lin et al., 2024), a dataset testing planning and action narration and execution, sampled using video data that is not extracted from a skill based pattern. Since VideoGUI uses a different data post processing method to reshape videos into trainable data, improvement in performance on VideoGUI will also show the effectiveness and generalizability of our skill-based trajectory cura-

[3] `https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-6-flash-3-5-flash-lite-3-5-flash-cyber/`

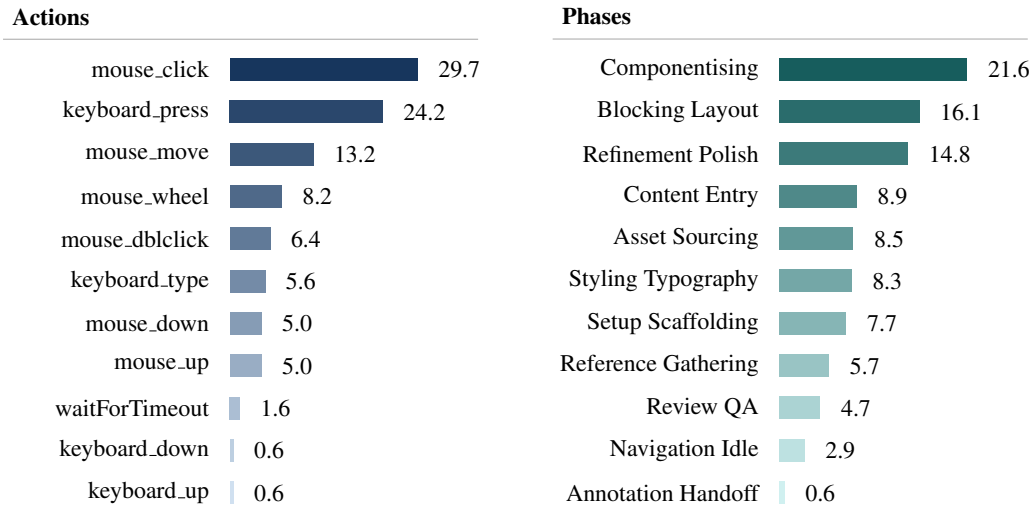

Figure 3: **Combined-corpus composition in share percentage.** Actions follow the Playwright MCP toolset, counted over all validated tool calls. Phases come from the 12-way taxonomy as described in subsection A.1. Bars are scaled within each panel and shares are rounded. The skill mix is listed out separately in Figure 4.

tion process. Since our trajectories do not come with pre-annotated reasoning chains, we train our models without reasoning. All closed model evaluations use high reasoning effort.

5 Results and Discussion

RQ1. Does training on FigmaTrace help improve next-action accuracy on design tasks? Table 2 showcases the out-of-domain performance improvement seen for fine-tuned model as compared to the random, open and closed source model baselines. We observe that the FigmaTrace fine-tuned models compare favorably to state-of-the-art closed source models like Claude-Opus-5 and GPT-5.6-Sol while being considerably smaller in size. Specifically, we observe that Qwen3.8-27B model outperforms even Claude-Opus-5 at GUI-Odyssey and out of the box AndroidControl tasks by up to 6.4% and 11.8% absolute points. This shows that the navigation skills learned using FigmaTrace generalize to broader agentic and design tasks. As an additional in-domain design-task evaluation for the best performing Qwen3.8-27B, we utilize the ScreenSpot-Pro Creative (Li et al., 2025a) split and observed an absolute increase of performance of 7.4% points over the base model performance (29.3% vs 36.7%).

RQ2. How does phase-based training of VLM agents compare against maximum length sharding for long horizon tasks? To investigate whether the phase-based trajectory creation process is beneficial, we compare this against the max length based truncation of trajectories. As observed in Table 3, we notice that phase-based outperforms maximum context length-based truncation by a margin of 7.3 absolute points. On closer qualitative analysis, we found that within AndroidControl the most affected rows are those that open mid-action, either as "Continue the work" stubs or as undirected references to on-page coordinates. This is because our phase-based method better teaches the model skill-based grounding instead of simple instruction following. For Mind2Web and VideoGUI, where the corpus focuses on layouts and not action grounding, the performance is not as significantly affected. Hence, our methodology for creating FigmaTrace outperforms maximum length-based trajectory generation processes overall.

RQ3. What patterns in FigmaTrace influence qualitative performance improvements in mod- els? We manually inspect every item on which SFT converts a base failure into a success and find three recurring patterns. First, **element selection accuracy**: two-thirds of all gains observed on GUI-Odyssey are cases in which the base model selects an entirely different UI element. This is seen via the base model's median error, $\approx$ 457 px, while SFT lands within $\approx$ 15 px. The most

Table 2: Step-wise accuracy (%) across OOD GUI agent benchmarks. For VideoGUI, directed refers to step-wise instruction provision whereas undirected is open-ended navigation. Mind2Web tests are run in two different image viewport (vp) configurations. **Bold** indicates the best open model in each column and Underline represents the best closed model. *MUSE-GLIMMER-30B was evaluated similarly to the official evaluation with multiple zoom and crops averaged.

			Mind2Web		VideoGUI	
Model	GUI-Odyssey	AndroidControl	vp800	vp1000	directed	undirected
Random Baseline	0.6	21.8	1.1	0.9	0.6	0.6
CLAUDE-OPUS-5	47.3	87.3	100.0	75.3	71.3	40.7
GPT-5.6-SOL	44.0	83.2	91.3	69.7	77.7	29.6
QWEN-3.6-35BA3B	29.0	36.8	38.2	30.7	47.0	10.1
MUSE-GLIMMER-30B	29.3	59.3	64.7	65.3	64.7	28.7
GEMMA-4-31B	43.3	87.3	68.0	64.0	63.0	24.0
QWEN-3.8-27B	44.0	82.7	69.3	65.3	65.3	26.0
QWEN-3.6-35BA3B + SFT	51.1	83.2	65.2	63.7	70.4	15.9
MUSE-GLIMMER-30B + SFT*	53.2	84.5	70.2	**70.1**	**73.0**	**32.8**
GEMMA-4-31B + SFT	45.2	96.1	69.1	66.0	67.0	23.3
QWEN-3.8-27B + SFT	**53.7**	**99.1**	**70.7**	68.7	71.3	19.3

Table 3: Comparison of phase-aware SFT against the base QWEN3.8-27B model and a maximum-length based SFT baseline across six GUI agent benchmarks. Best result per row in **bold**.

Benchmark	Base	Phase-based SFT	Length-based SFT
GUI-Odyssey	44.0	**53.7**	40.0
AndroidControl	82.7	**99.1**	67.3
Mind2Web vp800	69.3	**70.7**	69.3
Mind2Web vp1000	65.3	68.7	**70.0**
VideoGUI-undirected	**26.0**	19.3	21.3
VideoGUI-directed	65.3	**71.3**	70.0
Mean	58.8	**63.8**	56.3

illustrative case is the instruction "chat about it with a friend on Instagram", whose target is the message input at the bottom of the screen. For this setting, the base model predicts (596, 112), the search bar at the top of the frame, at essentially the same horizontal position as the gold point (593, 1325), whereas SFT lands within 7 px from the ground truth target. Such corrections concentrate on share icons, video cards, and chat inputs, which is reflected in the largest per-category improvements, including Media (+22 pp) and Social (+17 pp). Second, **coordinate understanding**: the base model frequently emits raw pixel coordinates instead of the norm-1000 coordinates it was originally post-trained on. A prediction of (800,212), for instance, falls directly on the share icon in pixel space but scores 360,px off once read as norm-1000. Especially in tall frames, targets in the bottom third overflow the grid entirely. In this case, raw value $y > 1000$ extends beyond the viewport frame. This occurs on 10/150 analyzed GUI-Odyssey items for the base model and on none for SFT. Third, **decisiveness**: every gain on AndroidControl comes from an item on which the base model either emits no coordinates at all or selects a totally incorrect element whereas SFT always answers, and when it corrects the element choice it lands at an average of ≈ 11 px.

On the other hand, where SFT fails, the same inspection applied to items on which SFT converts a base success into a failure reveals two newly acquired habits. The first is **repetition**: on Android app flows, SFT predicts effectively the same pixel on two consecutive steps, (331, 989) followed by (331, 988), while the ground truth trajectory advances down the list. The base model consistently tracks this progression correctly. We believe that this is an artifact of the noisy actions and mouse clicks that are leaked into FIGMATRACE during preprocessing and conversion from raw video and action data. This behavior does not generally impact the overall task performance but results in a longer trajectory, thereby putting strain on the large context memory of the agent. The second is **screen-center focus**. Here, targets in the browser chrome are abandoned in favor of content in the

middle of the screen, plausibly a leak of the canvas-centric bias induced by FIGMATRACE. Both of these categories cluster in utility and browser flows.

6 CONCLUSION

In this paper, we propose a novel skill taxonomy for general purpose design tasks and extend this to a novel dataset with 2883 training and 586 evaluation trajectories. Through our experiments, we show that training on FIGMATRACE leads to performance improvement not only on the Figma design tasks but also on out of domain agentic tasks. Furthermore, we show that our phase-based video to trajectory conversion method outperforms the standard maximum context length extraction method. Finally, we perform qualitative analysis of model behaviors to categorize the success and failure modes that influence model performance on downstream tasks.

REFERENCES

Arctanx An, Shizhao Sun, Danqing Huang, Mingxi Cheng, Yan Gao, Ji Li, Yu Qiao, and Jiang Bian. Can vision language models assess graphic design aesthetics? a benchmark, evaluation, and dataset perspective, 2026. URL https://arxiv.org/abs/2603.01083.

Sree Bhattacharyya and James Z. Wang. Evaluating vision-language models for emotion recognition. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1798–1820, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.97. URL https://aclanthology.org/2025.findings-naacl.97/.

Abhishek Chandwani and Ishan Gupta. Lh-bench: Skill-grounded evaluation of long-horizon agents on subjective enterprise tasks. *arXiv preprint arXiv:2603.22744*, 2026.

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pp. 845–854, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349819. doi: 10.1145/3126594.3126651. URL https://doi.org/10.1145/3126594.3126651.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

Yihao Ding, Siwen Luo, Yue Dai, Yanbei Jiang, Zechuan Li, Qiang Sun, Geoffrey Martin, Wei Liu, and Yifan Peng. A survey on MLLM-based visually rich document understanding: Methods, challenges, and emerging trends. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 13319–13340, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.652. URL https://aclanthology.org/2026.findings-acl.652/.

Yi Gui, Jiawan Zhang, Yina Wang, Tianran Ma, Yao Wan, Shilin He, Dongping Chen, Zhou Zhao, Wenbin Jiang, Xuanhua Shi, Hai Jin, and Philip S. Yu. Figma2code: Automating multimodal design to code in the wild. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=CaXZB6bI31.

Daeheon Jeong, Seoyeon Byun, Kihoon Son, Dae Hyun Kim, and Juho Kim. Canvas: A benchmark for vision-language models on tool-based user interface design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22182–22190, 2026.

Ishani Kanapathipillai and Obhasha Priyankara. Cogen: Creation of reusable ui components in figma via textual commands. *arXiv preprint arXiv:2601.10536*, 2026.

Hugo Laurençon, Léo Tronchon, and Victor Sanh. Unlocking the conversion of web screenshots into html code with the websight dataset, 2024. URL https://arxiv.org/abs/2403.09029.

Kaixin Li, Meng Ziyang, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: GUI grounding for professional high-resolution computer use. In *Workshop on Reasoning and Planning for Large Language Models*, 2025a. URL https://openreview.net/forum?id=XaKNDIAHas.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on ui control agents. *Advances in Neural Information Processing Systems*, 37:92130–92154, 2024.

Weiqi Li, Xuanyu Zhang, Shijie Zhao, Yabin Zhang, Junlin Li, Li Zhang, and Jian Zhang. Q-insight: Understanding image quality via visual reinforcement learning, 2025b. URL https://arxiv.org/abs/2503.22679.

Zhichao Liao, Xiaokun Liu, Wenyu Qin, Qingyu Li, Qiulin Wang, Pengfei Wan, Di Zhang, Long Zeng, and Pingfa Feng. Humanaesexpert: Advancing a multi-modality foundation model for human image aesthetic assessment, 2025. URL https://arxiv.org/abs/2503.23907.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Qinchen Wu, Mingyi Yan, Zhengyuan Yang, Lijuan Wang, and Mike Zheng Shou. Videogui: A benchmark for gui automation from instructional videos. In *NeurIPS*, 2024.

Dunjie Lu, Yiheng Xu, Junli Wang, Haoyuan Wu, Xinyuan Wang, Zekun Wang, Junlin Yang, Hongjin Su, Jixuan Chen, Junda Chen, Yuchen Mao, Jingren Zhou, Junyang Lin, Binyuan Hui, and Tao Yu. Videoagenttrek: Computer use pretraining from unlabeled videos, 2025a. URL https://arxiv.org/abs/2510.19488.

Quanfeng Lu, Wenqi Shao, Zitao Liu, Lingxiao Du, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, and Ping Luo. Guiodyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22404–22414, 2025b.

Yi-Hao Peng, Jeffrey P. Bigham, and Jason Wu. Designpref: Capturing personal preferences in visual design generation, 2025. URL https://arxiv.org/abs/2511.20513.

Francesca Russo, Tommaso Calò, and Luigi De Russis. Bridging web and figma: Automating large-scale ui dataset generation for ai-enhanced design. In *Companion Proceedings of the 17th ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, pp. 13–20, 2025.

Yuriel Ryan, Rui Yang Tan, Kenny Tsu Wei Choo, and Roy Ka-Wei Lee. Humor in pixels: Benchmarking large multimodal models understanding of online comics. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 14024–14050, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.755. URL https://aclanthology.org/2025.findings-emnlp.755/.

Ranjan Sapkota, Yang Cao, Konstantinos I Roumeliotis, and Manoj Karkee. Vision-language-action models: Concepts, progress, applications and challenges. *arXiv preprint arXiv:2505.04769*, 2025.

Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. Design2Code: Benchmarking multimodal code generation for automated front-end engineering. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3956–3974, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.199. URL https://aclanthology.org/2025.naacl-long.199/.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. A survey on (m)llm-based gui agents, 2025. URL https://arxiv.org/abs/2504.13865.

Weishi Wang, Hengchang Hu, Zhijie Zhang, Zhaochen Li, Hongxin Shao, and Daniel Dahlmeier. Document intelligence in the era of large language models: A survey, 2025a. URL https://arxiv.org/abs/2510.13366.

Xinyuan Wang, Bowen Wang, Dunjie Lu, Junlin Yang, Tianbao Xie, Junli Wang, Jiaqi Deng, Xiaole Guo, Yiheng Xu, Chen Henry Wu, Zhennan Shen, Zhuokai Li, Ryan Li, Xiaochuan Li, Junda Chen, Boyuan Zheng, Peihang Li, Fangyu Lei, Ruisheng Cao, Yeqiao Fu, Dongchan Shin, Martin Shin, Jiarui Hu, Yuyan Wang, Jixuan Chen, Yuxiao Ye, Danyang Zhang, Dikang Du, Hao Hu, Huarong Chen, Zaida Zhou, Haotian Yao, Ziwei Chen, Qizheng Gu, Yipu Wang, Heng Wang, Diyi Yang, Victor Zhong, Flood Sung, Y. Charles, Zhilin Yang, and Tao Yu. Opencua: Open foundations for computer-use agents, 2025b. URL https://arxiv.org/abs/2508.09123.

Tianhe Wu, Jian Zou, Jie Liang, Lei Zhang, and Kede Ma. Visualquality-r1: Reasoning-induced image quality assessment via reinforcement learning to rank. *Advances in Neural Information Processing Systems*, 38:88167–88190, 2026.

Junlin Xie, Zhihong Chen, Ruifei Zhang, and Guanbin Li. Large multimodal agents: a survey. *Visual Intelligence*, 3(1):24, 2025.

Tianbao Xie, Jiaqi Deng, Xiaochuan Li, Junlin Yang, Haoyuan Wu, Jixuan Chen, Wenjing Hu, Xinyuan Wang, Yuhui Xu, Zekun Wang, et al. Scaling computer-use grounding via user interface decomposition and synthesis. *Advances in Neural Information Processing Systems*, 38, 2026.

Dapeng Zhang, Jing Sun, Chenghui Hu, Xiaoyan Wu, Zhenlong Yuan, Rui Zhou, Fei Shen, and Qingguo Zhou. Pure vision language action (vla) models: A comprehensive survey, 2025. URL https://arxiv.org/abs/2509.19012.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. Swift:a scalable lightweight infrastructure for fine-tuning, 2024. URL https://arxiv.org/abs/2408.05517.

Shijia Zhou, Saif M. Mohammad, Barbara Plank, and Diego Frassinelli. I came, i saw, i explained: Benchmarking multimodal llms on figurative meaning in memes, 2026. URL https://arxiv.org/abs/2603.23229.

A Appendix

A.1 Phase Taxonomy

A **phase** is a contiguous stretch of a recording throughout which the expert holds a single intent. Phases tile the recording exactly and every frame belongs to exactly one phase. The label vocabulary is closed and free-form labels drift between calls. Table 4 gives the full vocabulary.

A.2 Selection of Experts

All subject matter experts (SMEs) contracted for this study were required to satisfy the following conditions: 1) Have a minimum of two years of experience with Figma, 2) Must be at least 18 years of age. Since all SMEs were hired through Upwork [4], we assigned every SME a starter task to vet the quality of their work.

A.3 Skill Distribution of the Dataset

The distribution of skills is presented in Figure 4. We observe that structural craft dominates the distribution overall with visual perception being the second most frequent category. This is expected given that auto-layout and component hygiene are primary requirements for Figma design workflows.

[4] https://upwork.com

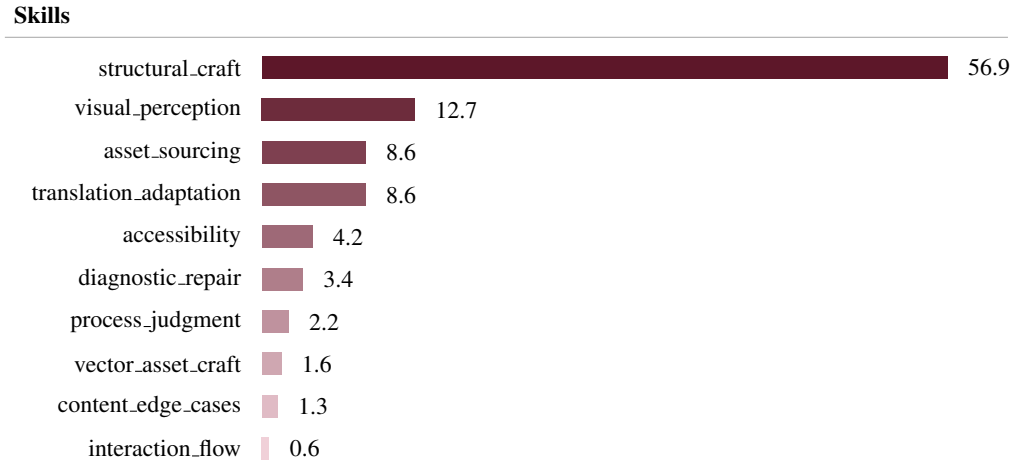

Figure 4: **Skill mix by share percentage** across all skill-labeled trajectories in FIGMATRACE.

Table 4: The closed phase vocabulary. Labels are ordered by their typical position in the workflow rather than by frequency.

Label	Definition	Share (%)
`reference_gathering`	Studying or collecting source material	5.7
`setup_scaffolding`	Establishing frames, artboards, grids, palettes, and styles	7.7
`blocking_layout`	Setting coarse structure, placement, and sizing	16.1
`asset_sourcing`	Searching for or importing images, icons, and fonts	8.5
`content_entry`	Typing real copy into the artifact	8.9
`styling_typography`	Making colour, type, and spacing decisions	8.3
`componentising`	Turning ad hoc elements into reusable components	21.6
`refinement_polish`	Making small deliberate adjustments and alignment passes	14.8
`review_qa`	Comparing against reference; inspecting and checking	4.7
`annotation_handoff`	Writing notes and comments; documenting decisions	0.6
`navigation_idle`	Scrolling, waiting, or app churn with no creative intent	2.9

A.4 REVIEWERS AND ANNOTATORS

Table 5: Design file readiness checklist.

Criterion	**Status**	**Notes**
1. File Structure & Agent Navigation		
Cover page with file name, version, and status	Nice to have	Agent does not use the cover page directly, but it helps human reviewers orient quickly before handing the file over.
Pages are logically organized (e.g. Cover / Components / Screens / Archive)	Pass/Fail	In accordance with Figma best practices
Pages are not overloaded and large files are split across multiple pages to avoid exceeding agent context limit	Pass/Fail	Source: Figma MCP docs specifically mention 'If you call `get_design_context` on an entire page instead of a specific node, the response can easily exceed the 25,000-token context window.' and this is expected to reduce this single call load.

Continued on next page

Table 5 — *continued from previous page*

Criterion	Status	Notes
Each screen is a separate top-level frame with a descriptive name (e.g. ”Login Screen”, ”Home / Mobile”)	Pass/Fail	Source: Figma MCP documentation mentions agent uses `get_metadata` to navigate the file by reading frame names. For example, ”Frame 247” gives no orientation.
Canvas contains no loose elements outside production frames i.e. no stray shapes, old iterations, or leftovers	Pass/Fail	Source: Figma MCP's `get_metadata` scans the full page. Loose elements appear as noise and could disorient the agent.
Old versions and iterations are archived on a separate page or removed. They should not be left on the working canvas.	Pass/Fail	This incentivizes cleanup actions for the agent trying to solve tasks.
Pages and layers are named meaningfully. No default or autogenerated names	Pass/Fail	-
No hidden elements or orphaned frames in the final submission file	Pass/Fail	-
2. Components & Design System		
All components, styles, and assets live in the submission file. There should not be any cross-file library dependencies	Pass/Fail	For all of these tasks, the agent is expected to work within a single file. External libraries are not accessible.
It should be clear which elements are components and which are layout frames. For example, repeated UI (buttons, inputs, cards, chips) must be real components but one-off layout frames (sections, containers, screen layouts) do not need to be.	Pass/Fail	Not everything needs to be a component. What matters is that repeating elements are components so the agent can reuse them, and one-off frames are intentionally not components and not accidentally detached.
Accidentally detached instances are resolved i.e. if something was a component and got detached without intent, it should be reconnected or rebuilt.	Pass/Fail	-
Components expose editable text properties (button labels, card copy) and text is not baked into the component.	Pass/Fail	-
Component names are semantic and searchable. For example, ”Button/Primary”, ”Card/Product” or ”Input/Text”	Pass/Fail	Source: Figma MCP using agent uses `search_design_system` to find components by name. ”Component 47” will not be found when searching for ”button”.
Variant and property names are semantic. For example, ”State=Default/Hover/Disabled” not ”Property 1=Option 1/Option 2”	Pass/Fail	Source: Figma naming guide informs that each item is a gap the agent will fill with guesswork if you leave it.
Auto Layout is applied to all relational components. No fixed/absolute positioning	Pass/Fail	This is for consistency purposes and in accordance with Figma best practices
Constraints are set deliberately on elements that should respond to resizing	Pass/Fail	Not everyone, but it is not required in this case.
File includes higher-order compositions (cards, headers, form rows) and not only atomic components (buttons, inputs)	Pass / Fail / N/A	Source: Figma Help Center mentions: ”Atomic components are difficult for AI to compose into coherent layouts on their own.”

Continued on next page

Table 5 — *continued from previous page*

Criterion	Status	Notes
Component names use slash notation consistently accoridng to best practices. For example: Button/Primary/Large, Card/Product, Input/Text/Default	Pass / Fail / N/A	Source: Figma Help Center states ”Figma slash notation creates nested groups that help both people and agents navigate large systems.”
Key components have a description filled in about what it is, when to use it, when not to use it, and keywords	Nice to have	Source: Figma MCP documentation: ”Figma MCP reads component descriptions and passes them to the agent as context.” Format: ”[What it does]. Use for [when]. Do not use for [when not]. Keywords: [searchable terms].” Example: ”Primary action button. Use for the main CTA on any screen. Do not use for secondary actions or destructive actions. Keywords: button, CTA, submit, confirm.” This is a process decision. Refer to the next item.
Process recommendation — component descriptions		
Who fills in component descriptions and when?	Pass / Fail / N/A	Component descriptions are not a natural part of the designer workflow because they are hidden in the component panel and rarely filled in without a specific process. Three options: (A) designer fills in during component creation which requires process change and training, hard to maintain. (B) one dedicated review before handing the file to the agent where someone goes through all components and adds descriptions; one-time effort, not ongoing. (C) agent generates descriptions and agent reviews components and proposes descriptions that the designer approves; recommended for our setup.
3. Icons & Assets		
Icons are embedded in the file as components and not linked from an external library the agent may not have access to	Pass / Fail / N/A	-
Icons are not rasterized as images and not substituted with unicode characters or emoji	Pass / Fail / N/A	-
Images use IMAGE fill type and not SOLID color standing in as a placeholder	Pass / Fail / N/A	-
4. Color-Reference to Design System		
Brand colors exist as color styles with correct values and nothing resolves to white or a wrong theme by default	Pass / Fail / N/A	-
No hardcoded hex values on production frames. They should all be colors reference named styles	Pass / Fail / N/A	-

Continued on next page

Table 5 — *continued from previous page*

Criterion	Status	Notes
Colors are also defined as Variables (tokens) for semantic referencing via MCP `get_variable_defs`	Nice to have	Source: Figma MCP docs:"`get_variable_defs` only returns tokens if the design uses them." More useful when agent works via MCP only.
5. Typography—Reference to Design System		
Shared text styles exist for the full hierarchy (heading / subheading / body / caption) and can be applied	Pass / Fail / N/A	-
All fonts are available in Figma by default. No missing fonts (e.g. Proxima Nova is not embedded)	Pass / Fail / N/A	-
No per-element font overrides. All text references named text styles	Pass / Fail / N/A	-
6. Spacing & Layout		
The file has a discernible spacing scale which is ideally a documented token sheet (4 / 8 / 16 / 24px), at minimum a consistent scale evident in components	Pass / Fail / N/A	-
Spacing and padding values are stored in Variables which enables semantic token referencing via MCP	Nice to have	Less critical if agent uses browser CUA where agent can read values visually. More useful when agent works via MCP only.
7. Interactive States		
Interactive components (buttons, inputs) include default, hover/active, and disabled variants	Pass / Fail / N/A	-
Empty state is designed. Decisions are made about what the user sees when there is no content	Pass / Fail / N/A	-
Error state is designed. Decisions are made about what happens when something goes wrong	Pass / Fail / N/A	-
Loading state is designed. Decisions are made about what appears while content is loading asynchronously	Pass / Fail / N/A	-
Components handle long text gracefully i.e. no overflow or broken layout at max content	Pass / Fail / N/A	-
8. Content & Placeholder Quality		
Text content uses realistic placeholder copy and not "Lorem ipsum" or empty fields	Pass / Fail / N/A	Agent learns patterns from what it sees. Placeholder copy that resembles real content gives better context for tone, length, and information architecture.
Images use realistic placeholder fills (IMAGE fill with a neutral image) and not empty rectangles or colored blocks	Pass / Fail / N/A	-

Continued on next page

Table 5 — *continued from previous page*

Criterion	Status	Notes
Data in tables, lists, and cards represents realistic scenarios and not single-word entries or identical repeated rows	Pass / Fail / N/A	-
9. Accessibility		
Text contrast meets minimum ratio of 4.5:1 for body text, 3:1 for large text and interactive elements	Pass / Fail / N/A	-
Focus states are designed for all interactive elements	Pass / Fail / N/A	-
Text is not embedded in images. It must be readable as actual text layers	Pass / Fail / N/A	-
10. Annotations		
Each screen's purpose is clear from its content and naming alone. The agent should be able to understand what a screen does without reading a separate brief	Pass / Fail / N/A	-
Non-obvious interactions are annotated: what triggers what, conditional logic, gestures	Pass / Fail / N/A	-
Edge cases and constraints are noted where relevant (e.g. max character count, empty state triggers)	Pass / Fail / N/A	-
11. Final Completeness Check		
No missing icons. Every icon slot has an actual icon, not an empty frame or placeholder shape	Pass / Fail / N/A	-
No missing images. Every image slot has an IMAGE fill, not an empty rectangle or colored block	Pass / Fail / N/A	-
No missing copy. Every text layer has real or realistic placeholder content, not "Text", "Label", or empty strings	Pass / Fail / N/A	-
No broken component instances. No instances showing "?" or missing component warnings in Figma	Pass / Fail / N/A	-
No missing fonts. Figma shows no font warnings in the file	Pass / Fail / N/A	-
All screens in scope are present. No screens referenced in flow but missing from the file	Pass / Fail / N/A	-
All states for interactive components are present. Nothing is "TODO" or visually incomplete	Pass / Fail / N/A	-
12. Target-Size Fitness		
Components hold up visually at the file's target screen size (mobile 390px / desktop 1440px)	Pass / Fail / N/A	-

Continued on next page

Table 5 — *continued from previous page*

Criterion	Status	Notes
Touch targets are at least 44×44px for all interactive elements on mobile (according to best practices)	Pass / Fail / N/A	-

A.5 FIGMATRACE TRAJECTORY SAMPLE

We present five consecutive recorded actions and the resulting state in Figure 1. The expert types a hex value into the color-style dialog, commits the new style (`#00DC82`), switches to the reference page, and selects elements to apply and verify it. Red crosshairs mark the recorded action coordinate on the frame the action was taken from. Every action-frame pair shown passed the corpus validation described in our curation method.

A.6 HYPERPARAMETERS

Table 6 enumerates the best hyperparameters for all training runs reported in Table 2. All runs reported in the paper were done with full-parameter SFT with the vision tower frozen and the aligner trainable. All runs are done with BF16 precision, Adam (β=0.9/0.95, weight decay 0.1), gradient clipping 1.0, cosine schedule with 3% warmup, micro-batch 1, and a 1,296,000-pixel visual budget (≈1,260 visual tokens/frame for QWEN, 1,120 soft tokens/image for GEMMA-4). All training rows carry up to 44 frames. Megatron arms use the distributed optimizer with bf16 moment states and full activation recomputation; GEMMA-4 cannot use Megatron because its per-layer KV-head counts are heterogeneous, so it runs HuggingFace SFTTrainer with Deepspeed. The updated configuration for those runs includes per-device batch size of 1 and gradient accumulation of 2 for the Deepspeed ZeRO-3 runs.

Model	Arm	Backend	Parallelism	LR
QWEN3.6-35B-A3B	final (published)	Megatron	TP1 CP8 EP8 DP2	2×10^{-6}
	baseline	Megatron	TP1 CP8 EP8 DP2	1×10^{-5}
	baseline (low lr)	Megatron	TP1 CP8 EP8 DP2	2×10^{-6}
	ViT unfrozen	Megatron	TP1 CP8 EP8 DP2	1×10^{-5}
QWEN3.8-27B	SFT	Megatron	TP4 CP2 DP2	2×10^{-6}
	max-length-shard abl.	Megatron	TP4 CP2 DP2	2×10^{-6}
GEMMA-4-31B-IT	plain	HF + ZeRO-3	DP16	2×10^{-6}
	plain (low lr)	HF + ZeRO-2	DP16	4×10^{-7}
	portrait-aug	HF + ZeRO-3	DP8	4×10^{-7}
MUSE-GLIMMER-30B	ATEM	Megatron	TP4 CP2 DP2	4×10^{-7}
	ATEM (high lr)	Megatron	TP4 CP2 DP2	2×10^{-6}
	JSON format	Megatron	TP4 CP2 DP2	2×10^{-6}
	JSON (low lr)	Megatron	TP4 CP2 DP2	4×10^{-7}
	combined	Megatron	TP4 CP2 DP2	4×10^{-7}

Table 6: Training hyperparameters for every SFT arm.