UISim: An Interactive Image-Based UI Simulator for Dynamic Mobile Environments

Jiannan Xiang[1,♠], **Yun Zhu**[2], **Lei Shu**[2,♠], **Maria Wang**[2],
Lijun Yu[2], **Gabriel Barick**[2], **James Lyon**[2], **Srinivas Sunkara**[2], **Jindong Chen**[2]
[1]University of California, San Diego (UCSD), [2]Google DeepMind
[♠]Correspondence to:{jixiang@ucsd.edu, leishu@google.com}

Abstract

Developing and testing user interfaces (UIs) and training AI agents to interact with them are challenging due to the dynamic and diverse nature of real-world mobile environments. Existing methods often rely on cumbersome physical devices or limited static analysis of screenshots, which hinders scalable testing and the development of intelligent UI agents. We introduce UISim, a novel image-based UI simulator that offers a dynamic and interactive platform for exploring mobile phone environments purely from screen images. Our system employs a two-stage method: given an initial phone screen image and a user action, it first predicts the abstract layout of the next UI state, then synthesizes a new, visually consistent image based on this predicted layout. This approach enables the realistic simulation of UI transitions. UISim provides immediate practical benefits for UI testing, rapid prototyping, and synthetic data generation. Furthermore, its interactive capabilities pave the way for advanced applications, such as UI navigation task planning for AI agents. Our experimental results show that UISim outperforms end-to-end UI generation baselines in generating realistic and coherent subsequent UI states, highlighting its fidelity and potential to streamline UI development and enhance AI agent training.

1 Introduction

Developing and testing user interfaces (UIs), particularly for mobile devices, is a complex and often resource-intensive process. Traditional methods rely heavily on manual screen captures and scripted interactions performed on physical devices or emulators [8, 27, 28, 30]. These methods are time-consuming, inflexible and do not scale well for comprehensive testing or agent training. Real-world UI environments are inherently dynamic and diverse, presenting significant challenges for automated analysis, rapid prototyping, and the development of robust UI navigation agents. As such, the ability to efficiently simulate these environments, allowing for interactive exploration and generation of plausible next states, is crucial for advancing both UI development and AI-driven human-computer interaction.

Existing approaches to UI simulation and automation often fall short in critical areas, as shown in Table 1. Some methods focus on generating UIs from high-level specifications [9, 19, 26, 31, 32], which limits their ability to accurately replicate the visual nuances of real-world applications. Others rely on static screen captures [5, 35], preventing dynamic interaction and exploration of subsequent states. While recent advances in image generation [6, 21] offer promising directions, end-to-end approaches often struggle with the precise control needed for coherent UI state transitions and often produce less realistic visual output. An interactive, image-based simulator that can dynamically respond to user actions and generate high-fidelity, plausible next-UI screens remains a significant challenge, especially for complex real-world UIs. Such a system is crucial for enabling rapid

Table 1: A comparative analysis of UI simulation and generation paradigms. This table evaluates various methods against critical features including dynamic state simulation, the ability to exert fine-grained structural control, and overall scalability. UISim (Ours) is demonstrated to address the combined challenges that other approaches face, offering a comprehensive solution for interactive mobile UI environments.

Methods	Dynamic State Simulation	Fine-Grained Structural Control	Scability
Physical Emulator	✓	✓	✗
High-Level UI Spec Generation	✗	✓	✓
Static Screenshot Analysis	✗	✗	✓
End-to-End Image Generation	✓	✗	✓
UISim (**Ours**)	✓	✓	✓

prototyping, scalable testing of UI designs, and critically, for fostering the development of intelligent agents that can learn to navigate complex digital environments through rich visual feedback.

To address these pressing limitations, we introduce UISim, a novel image-based UI simulator that enables dynamic and interactive simulation of mobile phone environments purely from screen images. Our system uniquely provides a flexible and scalable platform for exploring UI states and actions without requiring direct access to a physical device or a complex rendering engine. Given an initial phone screen image and a user action (e.g., open the email app), UISim employs a robust two-stage method to generate the subsequent UI state: it first predicts abstract layout information describing the possible next screen based on the input action, then synthesizes a new, visually consistent UI image based on this predicted layout. The decoupled design breaks down the complex problem of image-to-image UI transformation into more manageable sub-problems, offering fine-grained control over the content and structure of the simulated UI, inherently leading to higher fidelity and more diverse generation capabilities compared to end-to-end image generation methods. Our experimental results show that UISim outperforms baselines by 36.73 on Fréchet Inception Distance, demonstrating its superiority in generating realistic and coherent subsequent UI states.

UISim delivers immediate and substantial practical benefits across multiple domains. For developers and designers, it enables rapid prototyping and UI testing, allowing quick iteration on designs, visualization of user flows, and observation of application behavior under various simulated user interactions, drastically reducing the need for cumbersome manual device testing. For researchers, it facilitates the generation of vast amounts of synthetic UI data, crucial for training data-hungry machine learning models for UI analysis and automation, and evaluating UI automation in a reproducible environment. Furthermore, the simulator's inherent capability to generate plausible next UI states based on arbitrary user actions is important for advanced applications such as UI navigation task planning. By providing a "look-ahead" mechanism for AI agents to explore potential interaction sequences and their visual outcomes, UISim can facilitate the development of more intelligent and adaptable agents capable of achieving complex, high-level goals like "send an email" or "book a flight" across diverse and unfamiliar applications. We believe this interactive, image-based simulator represents a crucial step towards more efficient and scalable development of both UI-centric applications and general-purpose AI agents interacting with human-designed interfaces.

2 Related Work

UI Simulation and Automation User interface (UI) development and testing have traditionally relied on either manual procedures or tooling that requires direct access to application source code or runtime environments. Tools like Appium [3], UIAutomator [2], and the Android Studio Emulator [1] support scripted interaction and visual inspection but are limited by their dependency on physical or emulated devices and their inability to simulate arbitrary UI states at scale. To address some of these limitations, model-driven UI generation approaches such as Model-Based Testing of GUI Applications and Automated UI Generation from Task Models [9, 19, 26, 31, 32] use high-level specifications to produce static UI layouts. However, these methods focus on synthesis rather than simulation and lack support for visually grounded state transitions or dynamic user interaction.

Other recent works aim to understand or reconstruct UIs from screen images. For example, [5] and [35] attempt to infer UI structures from visual input, enabling applications like code generation or

layout understanding. Nevertheless, these approaches are typically static—they do not model the dynamic behavior of UIs or generate plausible next states in response to user actions. Even simulation environments such as Rico [11] focus more on data collection and replay, rather than enabling new state generation.

In contrast to these prior methods, our work introduces UISim, an image-based UI simulator capable of predicting and synthesizing the next UI state purely from screen images and user actions. This allows for dynamic, visual, and interactive UI simulation without reliance on application internals or emulators. By decoupling structure prediction from image generation, our approach can flexibly and realistically model transitions across diverse UI environments, bridging a critical gap in UI automation and interaction research.

Image and Video Generation for Structured Interfaces Recent years have witnessed the rapid progress of image [6, 18, 21, 24] and video diffusion models [7, 10, 22]. These models excel at producing high-fidelity outputs from textual or visual prompts, and have demonstrated impressive generalization across natural scenes and object categories. However, applying them directly to user interface (UI) generation remains challenging. Unlike natural images, UI screens follow strict structural rules—buttons, text fields, and layout hierarchies must remain consistent and functional across transitions. General-purpose image generators often lack the spatial control and semantic consistency needed to model these structured visual domains.

Several works attempt to bring structure-awareness into generative models. For example, LayoutDiffusion [38] and ControlNet [37] incorporate layout or edge-map conditioning to guide generation, which is useful for structured scenes but not specifically tuned for interactive environments like mobile UIs. End-to-end image/video generation models often struggle with the inherent reasoning required for accurate next-screen prediction in UI environments. Unlike natural image generation, where the prompt directly describes the desired visual output, predicting a UI transition necessitates inferring how user actions logically alter the screen's layout, content, and functionality. General-purpose generative models typically lack the explicit mechanism for this kind of high-level semantic understanding and structural planning. Their one-to-one mapping from input to output, without an intermediate reasoning step, can lead to generated UI states that are visually plausible but structurally inconsistent or semantically incorrect with respect to the intended user interaction.

UISim addresses this gap by introducing a structured two-stage pipeline specifically designed for UI simulation. It first predicts a layout conditioned on the current screen and user action, then synthesizes a high-resolution UI image from this layout. This decomposition enables fine-grained control over structure and visual fidelity, allowing our method to produce coherent and realistic transitions between states—capabilities that end-to-end visual generation models typically struggle to achieve in structured domains like mobile UIs.

UI Interactive Agents Vision-based AI agents for UI navigation and automation are increasingly studied as an alternative to traditional rule-based systems, particularly for applications involving accessibility, automation, and testing. Early works such as DroidBot [20] explore Android UIs by building runtime state models from screenshots and UI hierarchies to guide interaction testing. More recent approaches have moved toward instruction-following agents that act over screen images to accomplish tasks [12, 25]. These agents typically rely on supervised learning or modular grounding to interpret screen content and follow instructions.

In parallel, research on web agents, e.g., WebDreamer [13], has demonstrated the value of simulating possible action outcomes before execution using learned world models. While these works focus on web environments composed of HTML documents and semantic DOM elements, mobile UI agents must operate on image-based environments where no structured metadata is available, and where understanding visual context (e.g., button layouts, app themes) is crucial. Moreover, many existing web agents rely on large-scale language models to reason about structured actions (e.g., clicking an HTML tag) and cannot be directly applied to purely visual screen-based interactions.

UISim complements and extends this line of work by introducing a visually grounded UI simulation framework. It enables agents to predict and visualize the outcome of an action before executing it. This simulation capability not only enhances UI testing and debugging but also unlocks forward-planning capabilities for agents in vision-only environments, facilitating tasks like multi-step navigation or goal-directed interaction in mobile apps. In doing so, UISim serves as a general-purpose platform

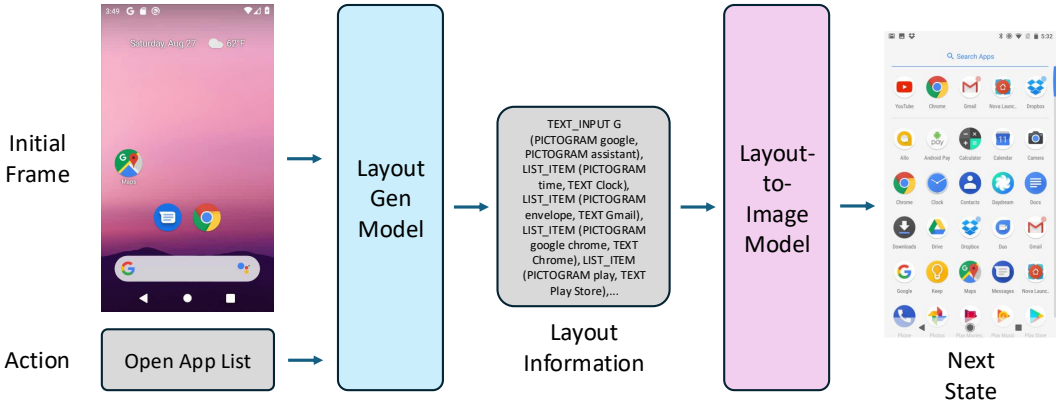

Figure 1: Overview of UISim's two-stage generation pipeline. Given an initial UI frame and a user action (e.g., "Open App List"), the Layout Generation Model predicts a structured layout description of the next UI state. This layout information—expressed in terms of UI components and their semantics—is then passed to a Layout-to-Image model, which synthesizes a realistic UI screen representing the predicted next state.

for developing and training UI agents, analogous to how world models have enabled model-based planning in web-based agent research.

3 UISim

Our proposed UI simulator, UISim, is designed to dynamically simulate mobile user interface environments directly from visual input, fundamentally addressing the limitations of static analysis and resource-intensive, device-based testing. As shown by Figure 1, UISim operates on a novel two-stage pipeline. It firstly generates layout information based on the input screen image and user action, then generates the next screen image based on the layout information. This enables both fine-grained control over UI transitions and the generation of high-fidelity, visually consistent screen images.

The benefits of this two-stage pipeline are substantial. By explicitly predicting the layout before rendering the UI image, our system gains fine-grained control over both the structure and content of the generated screen. This decoupled design breaks down the complex task of image-to-image UI transformation into more manageable sub-problems, mirroring the Chain-of-Thought [34] method in large language models (LLMs), where reasoning is made more effective by decomposing complex tasks into intermediate steps. In contrast, end-to-end models must implicitly infer the target layout while synthesizing pixels, which often leads to degraded structure, lower coherence, and reduced controllability.

Our approach also aligns with recent advancements in large-scale image generation models, such as DALL-E 3 [6], where providing more detailed and structured prompts or conditioning information leads to significantly better generative performance and adherence to user intent. Similarly, our second stage greatly benefits from a rich, explicit layout prompt, resulting in more accurate, coherent, and visually plausible UI images. Furthermore, the mapping between a well-defined UI layout and its visual representation is often close to one-to-one, much like specific HTML code predictably generates a particular website. This inherent consistency ensures that the generated UI remains visually coherent with the original phone system style and overall aesthetic, minimizing information loss and preserving fidelity across dynamic transitions.

3.1 Layout Information Generation

The first stage of our pipeline, Layout Information Generation, is responsible for interpreting a user action on an initial UI screen and translating it into a structured, machine-readable representation of the predicted next UI state. The inputs to this stage are the initial phone screen image and a textual description of the user action (e.g., "open email app", "click search box", "scroll down").

We finetune an open-source vision language model (VLM) Qwen2-VL-7B-Instruct [33] for layout information generation. The model is trained to understand the visual context of the UI and the semantic meaning of the user action, predicting how the screen layout will change. The training data is created using Android in the Wild dataset [23], a rich collection of real-world mobile interactions. We extracted frame pairs from the trajectories in the dataset, ensuring diversity in applications and interaction types. For each pair, we annotated the user action that occurred between the initial and subsequent frame using Gemini 1.5 Pro [29], capturing the intent and locus of the interaction. Besides, we use ScreenAI [4], a VLM for UI understanding, to systematically annotate the second frame, extracting its complete structural and semantic information. This annotation includes details such as UI element names, element descriptions, and bounding box coordinates. Finally, each training example consists of the initial screen frame, the user action annotated by Gemini, and the layout information of the subsequent frame. After finetuning on the constructed data, the model is able to effectively transform a visual input and an action into a precise, structured UI blueprint.

By decomposing the complex task of UI transition prediction and introducing an explicit layout information generation stage, UISim delegates the reasoning process to a VLM. This approach leverages the capability of VLM to interpret multimodal inputs—an initial UI screen image and a textual user action—and plan the subsequent structural changes to the UI. Recent studies, such as DeepSeek-R1 [14], have demonstrated that LLMs are highly capable in reasoning-intensive tasks. In contrast, most existing end-to-end image and video generation models typically follow a one-to-one paradigm, directly mapping prompts to outputs without incorporating an intermediate reasoning step. However, UI transition prediction inherently requires a deeper form of reasoning about user intent and how it will manifest in logical and structural UI changes. By offloading this high-level decision-making to a VLM, our approach effectively separates it from low-level pixel rendering, leading to more robust generalization, enhanced structural fidelity, and semantically coherent generation of subsequent UI states.

3.2 Layout-to-Image Generation

The second stage takes the abstract layout predicted in Stage 1 and renders it into a high-fidelity, pixel-level UI image. We use an image diffusion model [16] that is specifically trained to synthesize visually compelling phone screen images from structured UI layout descriptions. The model was pretrained on a large-scale collection of diverse UI layout-image pairs, utilizing the same UI layout annotation system mentioned in Section 3.1 to derive the textual layout inputs. By conditioning on the rich and structured layout information, the model can generate visually realistic and coherent UI screens that accurately reflect the intended structural and stylistic changes. As shown by advanced image generation models like DALL-E 3 [6], more detailed prompts lead to significantly better generative outcomes. our two-stage pipeline provides the diffusion model with a highly refined prompt, i.e., layout information, which is a key reason UISimoutperform end-to-end baselines that attempt to generate the next screen directly from the initial image and action.

Another advantage of our architecture lies in its data scalability for pretraining. Collecting large-scale, high-quality layout-image pairs, e.g., crawling web UIs and annotating them, is relatively straightforward and can be automated at scale. In contrast, acquiring clean image-action-image triplets for end-to-end UI transition modeling is significantly more challenging. Screen operations recordings crawled from the web are often noisy and visually cluttered, while manually scaling up phone screen recordings is expensive and time-consuming. Our design allows us to easily leverage powerful diffusion models pretrained on readily available layout-image data, bypassing the bottlenecks associated with scarce dynamic interaction data.

4 Experiments

We detail our experimental setup in Section 4.1, including data construction, model training, and the baselines we used. We then show experimental results in Section 4.2.

4.1 Experimental Setup

Data Preparation. We construct our dataset from the Android in the Wild [23]. In this dataset, each trajectory represents the completion of a high-level user goal (e.g., “Find the cheapest hotel in

Table 2: Comparison of FID scores between our method (UISim) and two end-to-end baselines. Lower FID indicates higher visual fidelity. UISim significantly outperforms both baselines, demonstrating the effectiveness of our two-stage layout-guided generation approach in producing realistic and coherent UI screens.

Model	FID
CogV-Image	98.37
CogV-Video	99.51
Ours	**61.64**

Austin"). Within these goal-oriented trajectories, specific frames are manually tagged as keypoints, marking intermediate states towards the overall objective, e.g., the moment an app is opened, text is typed into a search box, or a results page is displayed.

We constructed our dataset of UI state transitions by leveraging the tagged keypoints in the original dataset. Specifically, for any two consecutive keypoints in a trajectory, we annotate the user action between them with Gemini, and the layout information for the second frame with an UI layout annotation system, as described in Section 3.1. Finally, we get 28306 examples in total. We subsample 27306 examples for training, with the remaining trajectories reserved for evaluation.

Model Training. For the first stage of our pipeline, we finetuned a Qwen2-VL-7B-Instruct on our constructed dataset of initial UI images, user actions, and corresponding next-state layout annotations. We used LoRA [17] for training efficiency. We used a learning rate of 1e-4, LoRA rank 4, and LoRA alpha 4. We trained the model for 5 epochs with batch size of 32. For the second stage, we leverage a pretrained layout-to-image diffusion model. The model was pretrained on large-scale collections of UI layout-image pairs, and can robustly generate high-fidelity UI screens from structured layout inputs.

Baselines. We compare UISim against two end-to-end generative baselines. The first baseline CogV-Image is an end-to-end image generation model, takes the initial UI screen image and the textual user action as direct inputs and generates the next UI screen image in a single step. Since UI interactions are inherently temporal, we also develop the second baseline CogV-Video, which is an end-to-end video generation model. It also takes the initial UI screen image and user action as input, but it is trained to generate a short video sequence depicting the UI transition. For comparison with our single-frame output, the final frame of the generated video sequence is used as the predicted next UI screen. Both of the two models are based on CogVideoX-5b-I2V [36] and finetuned on the same training data for UISim.

Evaluation Metric. We evaluate baselines and UISim with Fréchet Inception Distance (FID) [15], which is a widely recognized metric for assessing the quality and diversity of images generated by generative models.

4.2 Experimental Results

Table 2 shows the experimental results, demonstrating that UISim consistently outperforms both the end-to-end image generation and end-to-end video generation baselines. This superior performance validates the effectiveness of our two-stage architecture, confirming that decoupling layout prediction from image synthesis leads to significantly higher fidelity and more realistic UI simulations. The explicit intermediate layout representation allows our model to maintain visual consistency and generate coherent UI states more effectively than approaches that directly generate pixels from initial image-action pairs.

5 Conclusion

We introduced UISim, a novel two-stage image-based simulator for mobile user interfaces that enables realistic and controllable UI state transitions from visual inputs. Our method leverages a vision-language layout prediction model followed by a layout-conditioned image generation model.

This design provides strong structural control and high visual fidelity, addressing the limitations of prior end-to-end UI generation methods.

Our system is trained on a rich dataset of real-world UI transitions with annotated user actions and layouts. Experimental results demonstrate that UISim significantly outperforms competitive baselines—achieving a 36.73 improvement in Fréchet Inception Distance—while producing more coherent and semantically meaningful UI sequences.

Beyond UI prototyping and testing, UISim opens up new possibilities for interactive agent training and visual planning in screen-based environments. Future work includes integrating multi-step simulation, expanding to multimodal UI representations, and exploring tighter loops between layout reasoning and generative feedback. We believe UISim sets a foundation for more scalable, flexible, and intelligent interfaces in the next generation of human-computer interaction systems.

References

[1] Android Developers. Android Emulator. `https://developer.android.com/studio/run/emulator`, 2024. Accessed: 2025-06-03.

[2] Android Open Source Project. UIAutomator. `https://developer.android.com/training/testing/ui-automator`, 2024. Accessed: 2025-06-03.

[3] Appium Contributors. Appium: Mobile App Automation Made Awesome. `https://github.com/appium/appium`, 2024. Accessed: 2025-06-03.

[4] Gilles Baechler, Srinivas Sunkara, Maria Wang, Fedir Zubach, Hassan Mansoor, Vincent Etter, Victor Cărbune, Jason Lin, Jindong Chen, and Abhanshu Sharma. Screenai: a vision-language model for ui and infographics understanding. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24, 2024.

[5] Tony Beltramelli. pix2code: Generating code from a graphical user interface screenshot. In *Proceedings of the ACM SIGCHI symposium on engineering interactive computing systems*, pages 1–6, 2018.

[6] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[7] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024.

[8] John M Carroll, Robert L Mack, and Wendy A Kellogg. Interface metaphors and user interface design. In *Handbook of human-computer interaction*, pages 67–85. Elsevier, 1988.

[9] Liuqing Chen, Yunnong Chen, Shuhong Xiao, Yaxuan Song, Lingyun Sun, Yankun Zhen, Tingting Zhou, and Yanfang Chang. Egfe: End-to-end grouping of fragmented elements in ui designs with multimodal learning. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, pages 1–12, 2024.

[10] Google DeepMind. `https://deepmind.google/models/veo/`, 2025.

[11] 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*, pages 845–854, 2017.

[12] Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents. *arXiv preprint arXiv:2410.05243*, 2024.

[13] Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, et al. Is your llm secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.

[14] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[18] Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

[19] Jiazhi Li, Tingting Zhou, Yunnong Chen, Yanfang Chang, Yankun Zhen, Lingyun Sun, and Liuqing Chen. Uldgnn: A fragmented ui layer detector based on graph neural networks. *arXiv preprint arXiv:2208.06658*, 2022.

[20] Yuanchun Li, Ziyue Yang, Yao Guo, and Xiangqun Chen. Droidbot: a lightweight ui-guided test input generator for android. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 23–26. IEEE, 2017.

[21] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[22] A Polyak, A Zohar, A Brown, A Tjandra, A Sinha, A Lee, A Vyas, B Shi, CY Ma, CY Chuang, et al. Movie gen: A cast of media foundation models, 2025. *URL https://arxiv.org/abs/2410.13720*, page 51.

[23] Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Androidinthewild: A large-scale dataset for android device control. *Advances in Neural Information Processing Systems*, 36:59708–59728, 2023.

[24] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[25] Peter Shaw, Mandar Joshi, James Cohan, Jonathan Berant, Panupong Pasupat, Hexiang Hu, Urvashi Khandelwal, Kenton Lee, and Kristina N Toutanova. From pixels to ui actions: Learning to follow instructions via graphical user interfaces. *Advances in Neural Information Processing Systems*, 36:34354–34370, 2023.

[26] Hyun Seung Son, Woo Yeol Kim, and Robert Young Chul Kim. Mof based code generation method for android platform. *International Journal of Software Engineering and Its Applications*, 7(3):415–426, 2013.

[27] S Sridevi. User interface design. *International Journal of Computer Science and Information Technology Research*, 2(2):415–426, 2014.

[28] Debbie Stone, Caroline Jarrett, Mark Woodroffe, and Shailey Minocha. *User interface design and evaluation*. Elsevier, 2005.

[29] 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.

[30] Harold Thimbleby. *User interface design*. ACM, 1990.

[31] Muhammad Usman, Muhammad Zohaib Iqbal, and Muhammad Uzair Khan. A model-driven approach to generate mobile applications for multiple platforms. In *2014 21st Asia-Pacific Software Engineering Conference*, volume 1, pages 111–118. IEEE, 2014.

[32] Muhammad Usman, Muhammad Zohaib Iqbal, and Muhammad Uzair Khan. An automated model-based approach for unit-level performance test generation of mobile applications. *Journal of Software: Evolution and Process*, 32(1):e2215, 2020.

[33] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[34] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[35] Jason Wu, Xiaoyi Zhang, Jeff Nichols, and Jeffrey P Bigham. Screen parsing: Towards reverse engineering of ui models from screenshots. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 470–483, 2021.

[36] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[37] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023.

[38] Guangcong Zheng, Xianpan Zhou, Xuewei Li, Zhongang Qi, Ying Shan, and Xi Li. Layout-diffusion: Controllable diffusion model for layout-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22490–22499, 2023.