

# GenEscape: Hierarchical Multi-Agent Generation of Escape Room Puzzles

**Mengyi Shan    Brian Curless    Ira Kemelmacher-Shlizerman    Steve Seitz**

University of Washington

Seattle, WA

{shanmy,curless,kemelmi,seitz}@cs.washington.edu

## Abstract

We challenge text-to-image models with generating escape room puzzle images that are visually appealing, logically solid, and intellectually stimulating. While base image models struggle with spatial relationships and affordance reasoning, we propose a hierarchical multi-agent framework that decomposes this task into structured stages: functional design, symbolic scene graph reasoning, layout synthesis, and local image editing. Specialized agents collaborate through iterative feedback to ensure the scene is visually coherent and functionally solvable. Experiments show that agent collaboration improves output quality in terms of solvability, shortcut avoidance, and affordance clarity, while maintaining visual quality.

# 1 Introduction

Escape rooms are environments designed as interactive puzzles, where players must explore a confined scene, manipulate objects in a precise order, and ultimately exit the room. We challenge modern Vision-Language Models (VLMs) with the task of designing and building 2D escape room image puzzles. While those models produce aesthetically compelling images (OpenAI, 2024; DeepMind, 2024; Wang et al., 2024a; Li et al., 2024), they struggle with complex scenes that require fine-grained spatial relationships, physical affordance reasoning, or multi-step functional coherence.

A well-designed escape room puzzle must satisfy two critical criteria: it must be solvable, meaning the affordances of objects form a coherent and logically sound sequence of actions; and it must provide sufficient visual cues that guide the player toward that intended solution. This requires not just accurate object placement but a deliberate visual design that supports human reasoning through spatial relationships and visual emphasis. Traditional vision-language models, lacking structured planning and feedback, often generate scenes that

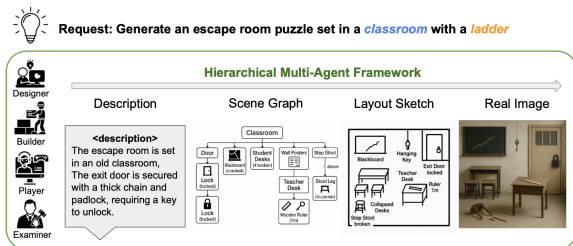


Figure 1: Four collaborative agents work together, hierarchically generating logically solid and visually appealing escape room puzzles through building scene graph, layout sketch towards photorealistic images.

are visually plausible but fail to meet these standards—either missing key logical links or omitting visual signals needed to solve the puzzle.

We propose a novel multi-agent (Zhou et al., 2024; Wang et al., 2024b; Li et al., 2025) interaction framework for escape room image generation. As in Fig 1, each agent contributes to a specific aspect of the scene (puzzle design, object placement, spatial consistency, or visual affordance verification) through iterative communication and refinement. This division of labor allows the system to reason about puzzle structure and object semantics in a modular way without sacrificing visual quality.

Our contributions are summarized as follows:

- We propose the novel task of generating physical, photorealistic escape room puzzle images.
- We propose a multi-agent collaborative system that hierarchically builds images through multiple levels of feedback on textual, symbolic and visual signals.

## 2 Related Work

**Prompt Optimization.** Prompt optimization is a powerful strategy for guiding LLM without weight updates. In NLP, techniques such as (Shin et al., 2020; Lester et al., 2021; Ouyang et al., 2022) show that well-crafted prompts improve downstream performance. Recent studies explore black-

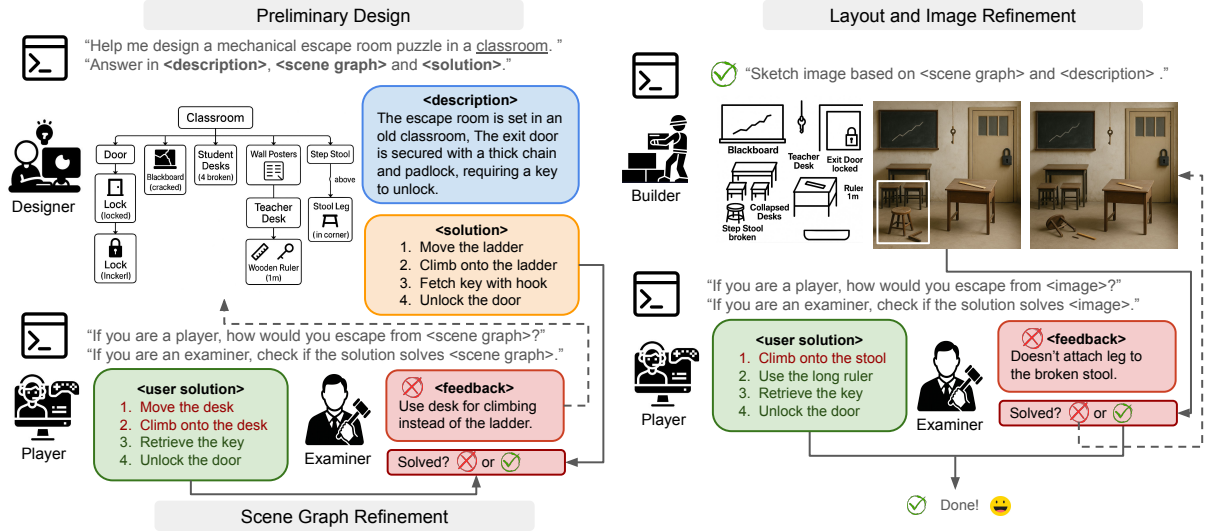


Figure 2: Hierarchical multi-agent pipeline for escape room image generation. The *Designer* creates an initial scene graph and solution. The *Player* and *Examiner* iteratively refine it for logical solvability. The *Builder* then generates a 2D layout and image, which are further verified by the agents to ensure visual and functional consistency.

box prompt optimization (Diao et al., 2023; Zhou and Neubig, 2023; Park et al., 2023) for tasks like QA and reasoning. In vision, prompt engineering and multi-turn refinement help steer diffusion and vision-language models (Liu et al., 2022; Hertz et al., 2023; Hao et al., 2023; Mo et al., 2024; Wu et al., 2024). Our approach introduces a hierarchical framework with symbolic intermediates, reducing local minima risks and improving convergence.

**Multi-Agent Systems.** Recent work has explored how multiple specialized agents can collaboratively solve complex tasks in language and vision (Zhang et al., 2024b; Xiong et al., 2025; Google Research, 2024). In visual generation, multi-agent setups have been applied to interactive storytelling, scene composition, and instruction-following environments, where agents assume distinct roles such as planning, verification, and rendering (Xu et al., 2024; Zhao et al., 2024; Hu et al., 2021; Gao et al., 2024). Most prior work either focuses on interactive dialogue or relies on fixed procedural pipelines.

**Generating Puzzles.** LLMs have been applied to puzzle generation and solving across multimodal domains. Raihan et al. (2024) combined chain-of-thought prompting with LLMs for word and sentence puzzles. Chia et al. (2024) introduced a diagnostic benchmark of abstract visual puzzles to assess multimodal reasoning. Estermann et al. (2024) proposed a reinforcement learning benchmark targeting logic-based algorithmic reasoning. EscapeCraft (Wang et al., 2025) also focuses on escape room generation, but relies on 3D assets, differing from our 2D image-based approach.

### 3 Method

**Problem Formulation.** We aim to generate an image of a mechanical escape room puzzle that is both visually realistic and logically solvable. Given a scene type keyword, a list of objects, and optionally a solution length  $l$ , the system produces an image  $I$  and an intended solution  $S$  (a list of action steps, each represented as a sentence of text) such that: (1) all key objects are present and spatially arranged to support a valid interaction sequence of at most  $l$  steps, and (2) the visual cues embedded in the scene are sufficient to guide a player toward inferring the intended solution  $S$  without shortcuts.

#### 3.1 Hierarchical Refinement Framework

Our framework comprises four *agents*, which are independent VLM instances assigned specific roles and communicating via text and visuals. The **Designer** generates the scene description, graph, and solution; the **Player** simulates a human solver; the **Examiner** compares the Player’s actions with the official solution and suggests refinements; and the **Builder** creates a 2D layout and photorealistic image aligned with the intended logic.

We adopt a hierarchical refinement strategy across four stages: text description, symbolic scene graph, 2D layout, and photorealistic image. At each stage, the *Player* proposes a solution, and the *Examiner* verifies solvability. This staged process reduces computational cost compared to direct image optimization while preserving object relationships and functional logic. We repeat until the

*Examiner* confirms the solution matches the official one. See Algorithm 1 for details.

### 3.2 Preliminary Design

In the first stage of our framework, the *Designer* agent is prompted to generate three aligned outputs: (1) a scene description in natural language that sets up the environment and puzzle premise, (2) a structured scene graph tree in yaml format where each node is an object in the scene, and each pair of parent-child represents a spatial connection relationship, and (3) a solution sequence consisting of valid player actions that logically lead to unlocking the exit. Here, a “valid” action sequence is one that respects the physical constraints and object states specified in the scene graph—e.g., only using provided tools, manipulating reachable objects, and avoiding shortcuts or physically implausible steps.

### 3.3 Scene Graph Optimization

In this stage, we refine the scene graph purely through symbolic reasoning, without involving any visual modality. The *Player* agent attempts to solve the puzzle by generating a sequence of actions based solely on the scene graph structure. This proposed solution is then evaluated by the *Examiner* agent against the intended one. The *Examiner* summarizes key differences in bullet points—for example, highlighting when a player exploits an unintended shortcut. In such cases, the *Examiner* revises the scene graph to block the shortcut and reinforce the expected solution path. Through iterative correction, the scene graph is adjusted until it supports a coherent and solvable strategy that aligns with the official solution. This process ensures that downstream image generation will faithfully reflect a functionally valid puzzle.

### 3.4 Affordance-Guided Local Editing

After the scene graph is finalized, we prompt the *Builder* to generate a 2D layout (Fig. 2 right) where each object is represented by an abstract icon positioned according to the spatial relationships in the scene graph. The same *Player–Examiner* interaction loop is reused to guarantee that the layout visually supports the intended puzzle solution.

Once the layout is verified, the *Builder* renders a photorealistic image based on the layout and the description. In this final stage, the *Examiner* again compares the *Player’s* actions—now based on the image—to the intended solution. If the *Player* misinterprets object affordances, the *Examiner* identi-

---

#### Algorithm 1: Hierarchical Puzzle Optimization with Multi-Agent Feedback

---

**Input** : Initial scene graph  $G_0$ ,  
ground-truth solution  $S$   
**Output** : Final image  $I$  that supports  $S$   
 $R \leftarrow G_0$ ;  
**for** stage  $t \in \{\text{GRAPH}, \text{LAYOUT}, \text{IMAGE}\}$   
**do**  
    **repeat**  
         $S^* \leftarrow \text{player.solve}(R)$ ;  
         $\Delta \leftarrow \text{examiner.check}(S, S^*)$ ;  
        **if**  $t = \text{GRAPH}$  **then**  
             $R \leftarrow \text{examiner.refine}(R, \Delta)$ ;  
        **if**  $t = \text{LAYOUT}$  **then**  
             $R \leftarrow \text{builder.refine}(R, \Delta)$ ;  
        **if**  $t = \text{IMAGE}$  **then**  
             $R \leftarrow \text{builder.refine}(R, \Delta, R_L)$ ;  
    **until**  $\Delta = \emptyset$ ;  
**return**  $R$ ;

---

fies which visual cues are lacking. Using the icon position from the layout stage, we then apply local image editing to enhance or suppress affordances to steer perception toward the correct interaction.

## 4 Experiment

We use GPT-4o (OpenAI, 2024) API as our base model and generate square images at  $1024 \times 1024$  resolution with the highest quality. We summarize the prompts for each step in the Appendix.

### 4.1 Evaluation Metrics

We evaluate our method using three human evaluation metrics. Solvability checks whether a player can infer the intended solution from the image. Shortcut Avoidance measures whether trivial, unintended solutions are blocked. Spatial Alignment assesses how well the visual output reflects the designed object relations. We also report the Long-CLIP (Zhang et al., 2024a) score measuring semantic similarity between the *Designer* prompt and the image, and the average number of image API calls (not applicable for vanilla baselines without refinement on images). Evaluation is conducted on 15 scene settings with 2 core objects each. Ten annotators are asked to pick the best image per case for the three evaluation aspects, given the setting and object names; we report the percentage of the results getting picked as best.



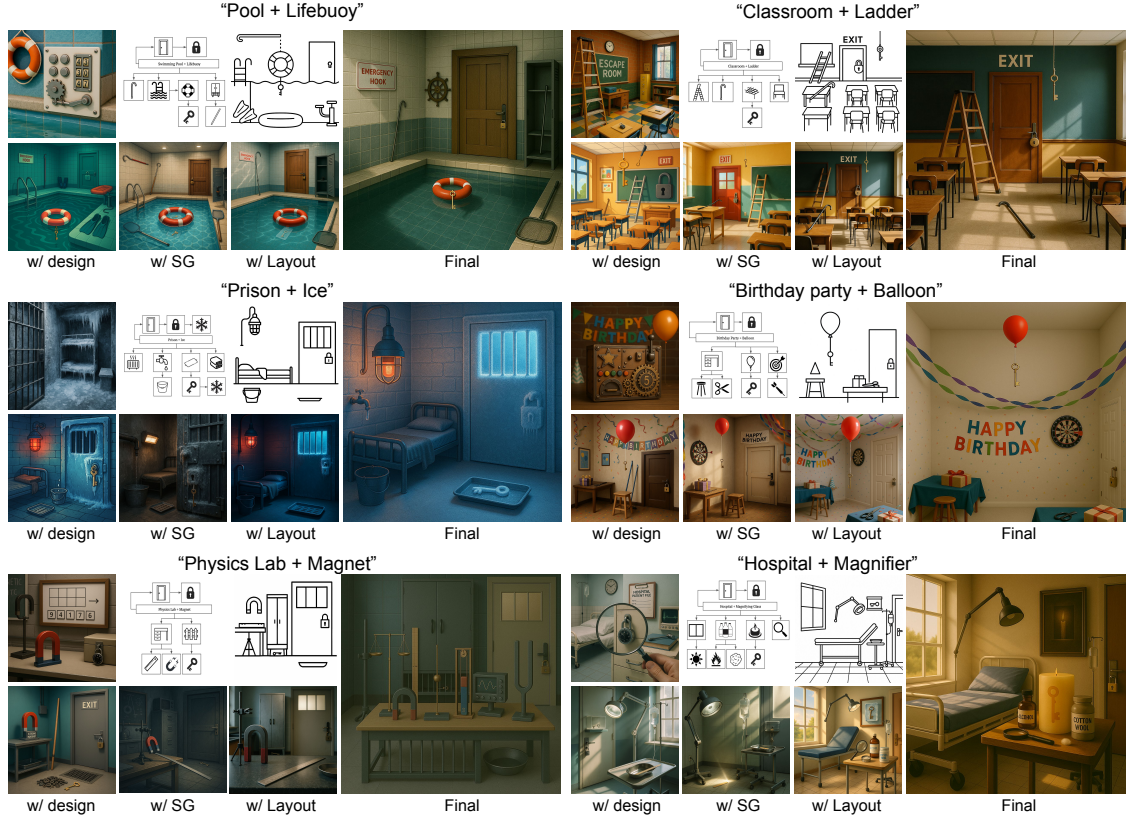


Figure 3: Our visual results compared with baseline and alternative designs. Left columns are naive generation (top) and assisted by a textual design (bottom). The second and third columns show the scene graph and 2D layout (top), and results with them (bottom). Right columns are the final results after visual signal-based image optimization. See if you can solve the puzzle! Official solutions and example human interaction are provided in the appendix.

## 4.2 Analysis

We compare against vanilla GPT-4o (OpenAI, 2024) and several ablations by removing refinement stages on the scene graph, layout sketch, and final image. Qualitative results are shown in Fig 3. Take the top-right classroom scene with the ladder as an example: naively prompting GPT for an “escape room” (top left) yields a scene with generic visual cues but no clear solution path. Using only the description includes key elements but lacks spatial grounding (e.g., a lock on the blackboard). Adding scene graphs and layouts improves spatial coherence and supports solvability. Final image editing removes shortcuts (e.g., climbable desks) and artifacts (e.g., malformed hooks).

Quantitative results in Tab 1 confirm that the scene graph improves the model’s ability to generate complex spatial relationships involving more than four objects, thus improving solvability; the layout sketch reduces the number of generations; and visual affordance refinement enhances local object characteristics, preventing shortcut solutions and promoting solution alignment. We didn’t see a clear improvement in the CLIP score, likely be-

Table 1: Comparison of adding different stages to the vanilla GPT-4o model, including description (D), Scene Graph (S.G.), Layout (L), and Image editing (I). The last row is our full pipeline with all modules added.

Method	Solv.	Short.	Align.	CLIP	#Gen.
GPT-4o	3.3%	0.0%	N/A	N/A	N/A
+D	6.7%	3.3%	0.0%	<b>0.42</b>	N/A
+D+S.G.	6.7%	13.3%	26.7%	0.37	N/A
+D+S.G.+L	10.0%	20.0%	13.3%	0.38	13.2
+D+S.G.+I	20.0%	16.7%	23.3%	0.32	15.8
<b>Ours full</b>	<b>53.3%</b>	<b>46.6%</b>	<b>36.7%</b>	0.32	<b>4.5</b>

cause the direct editing on images might change objects’ statuses to increase affordance awareness, disobeying the first stage’s prompt.

## 5 Conclusion

We presented a hierarchical multi-agent framework for generating escape room puzzles that are both visually realistic and logically solvable. By decomposing the task into symbolic reasoning, layout planning, and affordance-aware image refinement, our method ensures that each generated scene supports a coherent solution path and clear visual cues.



## Limitations

While our hierarchical multi-agent framework effectively generates escape room puzzles that are solvable and visually coherent, it has several limitations. First, our current design supports only puzzles with fully visible objects. Players cannot interact with the environment to uncover hidden items (e.g., opening boxes or inspecting drawers), limiting the depth and realism of puzzle mechanics. Second, our model doesn't support very long solution chains, specifically, more than eight steps or eight objects involved. It makes mistakes on spatial layouts and takes a very long time to converge to a solvable image. Finally, we imagine that it would be cool if we can generate images of the resulting scene condition after each human step, but GPT-4o's limited ability on image editing still prevents us from creating perfectly aligned scene images before and after a human action.

## Ethical Consideration

While our system is designed for virtual game development, it builds upon large language models (LLMs) that are known for their open-ended creativity and ability to generate realistic, interactive scenarios. When prompted without safeguards, these models may suggest escape room ideas involving hazardous mechanisms or unsafe real-world interactions, such as the use of fire, heavy objects, or confined spaces, that could be misinterpreted or physically replicated. This raises a broader concern: even though the generated content is intended for virtual or fictional use, it may inspire real-world setups that pose risks to safety, especially among unsupervised users or those lacking experience in safe design. Developers and researchers leveraging such generative tools must remain vigilant about unintended consequences, embedding appropriate content filters and clearly communicating that generated outputs are not suitable for real-life execution without expert oversight.

## References

Yew Ken Chia, Vernon Toh, Deepanway Ghosal, Lidong Bing, and Soujanya Poria. 2024. [PuzzleVqa: Diagnosing multimodal reasoning challenges of language models with abstract visual patterns](#). In *Findings of the Association for Computational Linguistics: ACL 2024*.

Google DeepMind. 2024. [Gemini 1.5 technical report](#). Accessed: 2025-05-08.

Qi Diao, Zhengxiao Xu, Tong Zhang, and Hang Li. 2023. Black-box prompt optimization with meta reinforcement learning. In *International Conference on Learning Representations (ICLR)*. ArXiv:2201.08531.

Benjamin Estermann, Luca Lanzendörfer, Yannick Niedermayr, and Roger Wattenhofer. 2024. [Puzzles: A benchmark for neural algorithmic reasoning](#). In *Advances in Neural Information Processing Systems (NeurIPS)*.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. 2024. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1259.

Google Research. 2024. Chain-of-agents: Large language models collaborating on long context tasks. <https://research.google/blog/chain-of-agents-large-language-models-collaborating-on->

Yaru Hao, Zewen Chi, Li Dong, and Furu Wei. 2023. Optimizing prompts for text-to-image generation. In *Advances in Neural Information Processing Systems*.

Amir Hertz, Ron Mokady, Rinon Gal, Amit H Bermano, and Daniel Cohen-Or. 2023. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations (ICLR)*.

Ting-Kuei Hu, Fernando Gama, Tianlong Chen, Wenqing Zheng, Zhangyang Wang, Alejandro Ribeiro, and Brian M Sadler. 2021. Scalable perception-action-communication loops with convolutional and graph neural networks. *arXiv preprint arXiv:2106.13358*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059.

Mingcheng Li, Xiaolu Hou, Ziyang Liu, Dingkan Yang, Ziyun Qian, Jiawei Chen, Jinjie Wei, Yue Jiang, Qingyao Xu, and Lihua Zhang. 2025. Mccd: Multi-agent collaboration-based compositional diffusion for complex text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhimin Li, Jianwei Zhang, Qin Lin, Jiangfeng Xiong, Yanxin Long, Xinchu Deng, Yingfang Zhang, Xingchao Liu, Minbin Huang, Zedong Xiao, Dayou Chen, Jiajun He, Jiahao Li, Wenye Li, Chen Zhang, Rongwei Quan, Jianxiang Lu, Jiabin Huang, Xiaoyan Yuan, and 26 others. 2024. [Hunyuan-dit: A powerful multi-resolution diffusion transformer with fine-grained chinese understanding](#). Preprint, arXiv:2405.08748.

- Shuang Liu, Song Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. 2022. Compositional visual generation with composable diffusion models. In *European Conference on Computer Vision (ECCV)*.
- Wenyi Mo, Tianyu Zhang, Yalong Bai, Bing Su, Ji-Rong Wen, and Qing Yang. 2024. Dynamic prompt optimizing for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.
- OpenAI. 2024. [Gpt-4o technical report](#). Accessed: 2025-05-08.
- Long Ouyang, Jeff Wu, Xu Jiang, and et al. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Joon Sung Park, Joseph C O’Brien, Carrie J Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442*.
- Md Nishat Raihan, Dhiman Goswami, Al Nahian Bin Emran, Sadiya Sayara Chowdhury Puspo, Amrita Ganguly, and Marcos Zampieri. 2024. [Solving puzzles with an ensemble of chain-of-thought prompts](#). In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval)*.
- Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Suchin Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.
- Can Wang, Zhikang Zhang, Zhanming Tang, and 1 others. 2024a. Cogvlm: Visual expert for pretrained language models. *arXiv preprint arXiv:2403.07584*.
- Zhenyu Wang, Aoxue Li, Zhenguo Li, and Xihui Liu. 2024b. Multimodal llm as an agent for unified image generation and editing. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Ziyue Wang, Yurui Dong, Fuwen Luo, Minyuan Ruan, Zhili Cheng, Chi Chen, Peng Li, and Yang Liu. 2025. [How do multimodal large language models handle complex multimodal reasoning? placing them in an extensible escape game](#). *Preprint*, arXiv:2503.10042.
- Zongyu Wu, Hongcheng Gao, Yueze Wang, Xiang Zhang, and Suhang Wang. 2024. Universal prompt optimizer for safe text-to-image generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6340–6354.
- Yujie Xiong, Yuan Cheng, Yinghui Xu, and 1 others. 2025. [Planning with multi-constraints via collaborative language agents](#). In *Proceedings of the 2025 Conference on Computational Linguistics (COLING)*.
- Xuenan Xu, Jiahao Mei, Chenliang Li, Yuning Wu, Ming Yan, Shaopeng Lai, Ji Zhang, and Mengyue Wu. 2024. Mm-storyagent: Immersive narrated storybook video generation with a multi-agent paradigm across text, image and audio. *arXiv preprint arXiv:2503.05242*.
- Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. 2024a. Long-clip: Unlocking the long-text capability of clip. *arXiv preprint arXiv:2403.15378*.
- Zeyu Zhang, Xuefeng Bai, Zhiwei Jia, and 1 others. 2024b. [Reflective multi-agent collaboration based on large language models](#). In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Yuheng Zhao, Junjie Wang, Linbin Xiang, Xiaowen Zhang, Zifei Guo, Cagatay Turkay, Yu Zhang, and Siming Chen. 2024. Lightva: Lightweight visual analytics with llm agent-based task planning and execution. *arXiv preprint arXiv:2411.05651*.
- Dewei Zhou, You Li, Fan Ma, Xiaoting Zhang, and Yi Yang. 2024. Migc: Multi-instance generation controller for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Zihan Zhou and Graham Neubig. 2023. Large language model guided evolutionary optimization for black-box prompt tuning. *arXiv preprint arXiv:2305.14216*.

## A Solution to Figure 3

### A.1 Pool + Lifebuoy

1. **Retrieve the skimmer net from the deck.** Pick up the lightweight, fully extended telescoping skimmer net leaning beside the pool.
2. **Get the clamp from the locker.** Open the nearby locker and take the waterproof clamp with rubber-lined jaws.
3. **Attach the clamp to the skimmer net.** Fasten the clamp securely to the end of the skimmer net, creating a long grabbing tool.
4. **Use the extended tool to reach the lifebuoy.** Carefully extend the modified skimmer over the pool and use the clamp to grab the plastic pouch taped to the underside of the floating lifebuoy.
5. **Unlock the exit door.** Open the sealed pouch, take out the brass key, and use it to unlock the padlocked door to escape the room.

### A.2 Classroom + Ladder

1. **Pick up the hooked pole from the teacher's desk.** Retrieve the small pole with a hook from the surface of the desk at the front of the room.
2. **Position the ladder beneath the key.** Move the ladder from its position near the chalkboard and place it safely under the dangling key.
3. **Climb the ladder and use the hooked pole.** Ascend the ladder carefully and use the hooked pole to pull the key string within reach.
4. **Grab the key and descend.** Once the key is close enough, grasp it securely and climb back down the ladder.
5. **Unlock the exit door.** Insert the key into the padlock on the exit door and turn it to unlock and escape the room.

### A.3 Prison + Ice

1. **Place the metal bucket under the dripping faucet in the corner.** Allow it to collect a sufficient amount of water from the steady drip.

2. **Move the filled bucket beneath the heat lamp.** Let it sit for several minutes so the water gradually warms up from the lamp's heat.
3. **Tear off a large piece of the wool blanket and soak it in the warm water.** The thick fabric will retain both heat and moisture, making it ideal for melting ice.
4. **Wrap the soaked section of blanket tightly around the frozen lock and key on the door.** Ensure it covers the area where the key is embedded in the ice to maximize thermal contact.
5. **Wait for the heat from the blanket to melt the ice.** Once the ice has sufficiently thawed, pull out the key.
6. **Use the key to unlock the door and escape the room.**

### A.4 Birthday Party + Balloon

1. **Pick up the scissors from the table.** Retrieve the visible scissors from the lower-right table.
2. **Throw the dart at the balloon.** Take the dart from the dartboard and throw it at the red balloon to pop it or weigh it down.
3. **Cut the string to retrieve the key.** Once the balloon is brought down and within reach, use the scissors to cut the string and release the brass key.
4. **Use the key to unlock the door.** Insert the key into the lock on the door handle and turn it to escape the room.

### A.5 Physics Lab + Magnet

1. **Use the metal clamp to secure the magnet.** Attach the U-shaped magnet to the clamp stand so it can be held steadily and extended toward the cage.
2. **Slide the magnet through the cage bars.** Position the magnet through the dense cage bars, aligning it with the brass key inside the cage.
3. **Attract the key using the magnet.** Carefully maneuver the magnet to make contact with the metallic key and pull it toward the bars.
4. **Retrieve the key from the cage.** Once the key is close enough to the bars, rotate and tilt the magnet to drag the key through a gap. Take the key out and use it to unlock the door.



### A.6 Hospital + Magnifier

1. **Soak the cotton wool with alcohol.** Pour a small amount of alcohol from the labeled bottle onto the cotton wool to make it flammable.
2. **Place the soaked cotton wool on top of the candle.** This sets up an easy ignition point to begin melting the wax.
3. **Use the magnifying glass to focus sunlight onto the cotton wool.** Position the cotton on the desk so that sunlight from the window can be concentrated using the magnifier. Adjust the angle until it ignites.
4. **Let the flame melt the wax and retrieve the key.** As the wax burns and softens, the embedded key becomes accessible. Use it to unlock the door and exit the room.

## B Additional Results

We additionally attach some creative results our model generates. Specifically, we ask the model to generate escape rooms where the human player can exit the room not by unlocking the door. See Fig. 4 for two such samples. Solutions are provided below.

### B.1 Amusement Park + Seesaw

1. Observe the sign "BALANCE THE FUN TO ESCAPE!" to understand that the seesaw must be balanced to open the exit.
2. Examine the right end of the seesaw and identify the 50kg and 20kg weights locked inside a cage.
3. Realize that the cage is fixed and those weights cannot be moved—this end of the seesaw is already weighted.
4. Collect the 100kg kettlebell and the golden elephant head from the ground near the left side of the seesaw.
5. Place the 100kg kettlebell and elephant head carefully on the left side of the seesaw to counterbalance the fixed weights on the right.
6. As the seesaw balances, the pressure on the right side lifts, unlocking the hidden trapdoor marked "EXIT".
7. Pull the handle on the EXIT mat to open the trapdoor and escape the room.

### B.2 Basement + Bomb

1. Take the crowbar from the wall-mounted shelf.
2. Use the crowbar to break the glass dome encasing the bomb.
3. Retrieve the matchstick from the open cabinet.
4. Light the fuse on the bomb using the matchstick.
5. Wait for the bomb to explode and destroy the weakened wall.
6. Exit the room through the opening in the destroyed wall.

## C Human Interface

We build a human interface that lets a human player look at the generated escape room puzzle and try to solve it. Our AI agent, with the official solution in mind, is able to judge whether the human solution works or not, and step by step guides the human player toward the correct solution. See Fig. 5 and Fig. 6 for an example of the game-playing procedure.

## D Sample Prompts

We provide the core, minimal set of prompts used for the multi-agent system. In practice, users can tweak the prompt to achieve certain effects (i.e., generating a certain style, stricter or looser on criticizing, etc.)

### D.1 Designer

- Help me design a mechanical and tactile escape room puzzle. There should be a visible lock on an exit door. And the player should take advantage of different interactive objects in the room to unlock that door. All the objects that are required to solve the puzzle should be visible. (There shouldn't be things like a key locked in a wooden box that the user cannot spot unless they open the box. ) Answer in a paragraph of scene description and a list of steps to solve it. Omit steps like "looking around" or "noticing xxx". List solid actions. The scene sets in a <scene> with <objects>. The scene is in a mystery but bright and vibrant atmosphere. Try to generate a very easy puzzle that can be solved in 3 steps with at most 4 objects involved.



Figure 4: Additional Results without usage of locks. Left columns are naive generation (top) and assisted by a textual design (bottom). The second and third columns show the scene graph and 2D layout (top), and results with them (bottom). Right columns are the final results after visual signal-based image optimization.

- Can you generate a scene graph for this scene in yaml format? It should be represented in a tree form, with the room as the highest ancestor. It should include all objects in the room. If something is attached or stored inside another object, they should be represented as parent and child.
- Can you generate a minimalist, black and white 2D scene layout sketch for this scene? Make sure the generation is from side view. Use 2D icons to represent central and salient objects in the scene, as well as their correct spatial relationship. Use text as marking if objects have specific conditions.

## D.2 Player

If you are a player, how would you utilize existing objects in this scene to exit from this room? Omit steps like observing or surveying and only focus on solid interactions with the objects. Pick one most reasonable sequence of actions.

## D.3 Examiner

- Here are the steps to solve the puzzle: <solution>. Do you think your solution is the same as it? It's acceptable to have a small difference, but the major logic and order should be the same. If yes, just say yes. If no, summarize the major differences in bullet points between yours and the official solutions.
- (If answered no to the previous question) What would be some changes to the scene graph that could provide more visual cues towards the correct solution? Rewrite that scene graph by adding or deleting objects, rearranging their spatial relationship, or modifying conditions, with emphasis on crucial objects.

- What are some modifications to the layout that can make the visual cues motivating the players more for the official solution?
- How would you edit the image that can make the visual cues motivating the players more for the official solution? Focus on editing on object at a time. Describe the local object you'd like to change, its position, and its expected new condition.

## D.4 Builder

- We are designing an escape room image puzzle. You are supposed to summarize the major objects in the scene and generate an image of 2D layout of the escape room. Draw crucial objects, and write their conditions (if any) next to the objects.
- We are designing an escape room image puzzle. You are supposed to summarize the major objects in the scene and generate a photorealistic image of the escape room, adhering to <description> and <layout>.

## E Human Evaluation Instructions

You will be shown six generated escape room images based on the same underlying scene graph and puzzle description—each produced by a different model. Your task is to evaluate these results along three criteria and pick the best one for each criterion.

Please consider the following aspects carefully:

- Solvability: Does the image provide enough visual cues for a player to correctly infer the intended solution? A good image should make the necessary objects, their functions, and the sequence of interactions clear enough that a human could reasonably solve the puzzle.

- **Shortcut Avoidance:** Does the image prevent trivial or unintended solutions? For example, if the puzzle is designed to require fixing a stool, the image should not allow the player to simply jump on a desk and bypass the challenge. The best results will clearly block or visually discourage such shortcuts.
- **Graph Alignment:** How well does the visual content match the structure and object relationships in the input scene graph? Pay attention to whether key objects are present, in the correct spatial positions, and whether their states (e.g., broken, locked, accessible) are visually reflected in the image.

For each criterion, assign a rank from 1 (best) to 4 (worst). Rankings must be unique (no ties). Please base your judgment on visual reasoning and consistency, not image quality or aesthetics alone.

## **F Annotator Details**

All human annotators involved in this study are students. They volunteered to participate without compensation and were fully informed that their evaluations would be used as part of a research project to assess the performance of different escape room generation models. Each annotator provided informed consent and was instructed to focus on the functional aspects of the images.





I'm ready! A new round begins — the scene is reset, and all objects are back in place.

Let the next player begin. What's your first action?

I'll try to grab the key by my hand.

That won't work — the balloon is floating too high near the ceiling, clearly out of arm's reach. Even standing on the stool won't help you reach it.

What will you try next?

If I stand on the stool and try to jump high, I might be able to reach the key?

You stand on the stool and try jumping to reach the key — but it's still just out of reach, and the jump isn't high or controlled enough to grab it safely. The ceiling is too high, and the balloon is floating freely.

What's your next move?

May I try to climb onto the table to grab the key? I'll hold the ribbons next to the wall to stabilize myself.

You carefully climb onto the table and try to stabilize yourself by holding the hanging ribbons.

Even with the added height, the balloon remains too high and floats freely — not tethered near the wall. The ribbons don't help much for balance, and the key is still out of reach.

What would you like to try next?

Figure 5: Part 1: Real case of human-agent interaction. The human is typing in possible solutions trying to solve the escape room puzzle, while the AI is providing feedback and gradually guiding the player towards the ideal solution. Followed by the next figure.



I'll throw the scissors to the balloon, trying to break it

You throw the scissors at the balloon — but it's a clumsy and imprecise tool for that purpose. The scissors miss and clatter to the floor. Worse, they could have damaged something or hurt someone if this were real.

Try a safer and more accurate approach. What's your next move?

📄 👍 🗨️ 🔊 🖋️ ↺️

Oh! Can I try using the sharp dart attached to the dart board on the wall? I'm good at throwing stuff so I might be able to break the balloon directly.

Nice thinking! You pull the sharp dart from the dartboard, take aim, and throw it at the balloon.

Direct hit — the balloon pops with a satisfying *snap*, and the key, still attached to the ribbon, drops gently to the floor.

What's your next action?

I'll use the scissors to cut off the ribbon to get rid of the unnecessary parts, and unlock the door.

You retrieve the scissors, snip the ribbon cleanly to free the key, and walk to the exit.

You insert the key into the padlock, turn it — *click* — and the door unlocks.

You've escaped successfully. Well done again!

Would you like to reset the scene for another round or try a new puzzle?

Figure 6: Part 2: Real case of human-agent interaction. The human is typing in possible solutions trying to solve the escape room puzzle, while the AI is providing feedback and gradually guiding the player towards the ideal solution.