Jigsaw: Supporting Designers in Prototyping Multimodal Applications by Assembling AI Foundation Models

David Chuan-En Lin
Carnegie Mellon University
Pittsburgh, PA, USA
chuanenl@cs.cmu.edu

Nikolas Martelaro
Carnegie Mellon University
Pittsburgh, PA, USA
nikmart@cmu.edu

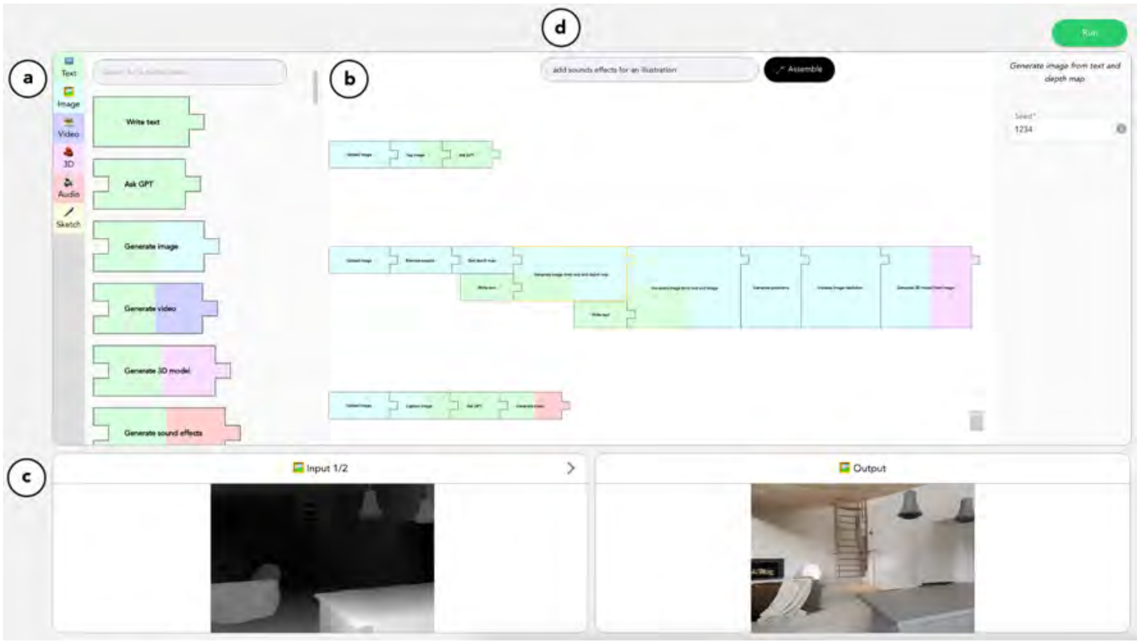


Figure 1: Jigsaw is a prototype system that lets designers generate and alter creative content with AI foundation models represented as puzzle pieces. Designers can search for foundation model capabilities via the Catalog Panel (a) and combine capabilities across different modalities by assembling compatible pieces on the Assembly Panel (b). Designers can specify inputs and observe intermediate results via the Input and Output Panels (c). Designers can request the Assembly Assistant (d) to recommend a chain of foundation models to accomplish a specified task.

ABSTRACT

Recent advancements in AI foundation models have made it possible for them to be utilized off-the-shelf for creative tasks, including ideating design concepts or generating visual prototypes. However, integrating these models into the creative process can be challenging as they often exist as standalone applications tailored to specific tasks. To address this challenge, we introduce Jigsaw, a prototype system that employs puzzle pieces as metaphors to represent foundation models. Jigsaw allows designers to combine different foundation model capabilities across various modalities by assembling compatible puzzle pieces. To inform the design of Jigsaw, we interviewed ten designers and distilled design goals. In a user study, we showed that Jigsaw enhanced designers' understanding of available foundation model capabilities, provided guidance on combining capabilities across different modalities and tasks, and served as a canvas to support design exploration, prototyping, and documentation.

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**; • **Applied computing** → **Arts and humanities**.

KEYWORDS

prototyping, machine learning, foundation models, multimodal, visual programming interface

1 INTRODUCTION

The past year has seen substantial progress in the capabilities of AI foundation models [8]. These models, which are pre-trained on vast quantities of data, can perform many tasks "off the shelf" without further training. Consequently, many foundation models essentially become input-output systems, simplifying the complexities of working with AI by abstracting models to their core capability [39]. Until recently, creating an AI-enabled system necessitated users to curate their own data, train a model, and occasionally modify the model architecture to adapt to their use cases [5].

With powerful plug-and-play capabilities, many designers have begun embracing AI foundation models to enhance their creative workflows. New foundation models support a wide variety of tasks and modalities, including large language models [32] such as GPT [9] for text generation and processing, image generation models such as Stable Diffusion [27], image segmentation models such as Segment Anything [19], and models for video [31], 3D model [17], and audio [21] generation.

However, despite the variety of capabilities offered, the integration of foundation models within the creative process can be challenging. Our initial observations suggest that these models are often used for one-off tasks or as standalone applications. For instance, a designer might use ChatGPT [1] to brainstorm and generate ideas, or they might use Midjourney [2] to generate visual prototypes. To incorporate the results of these models into their broader creative process, designers manually copy and paste the results into another design tool. Moreover, despite the variety of available models, designers typically only use a small selection of highly publicized models (ChatGPT, Stable Diffusion, MidJourney) and are often unaware of the range of capabilities and modalities they could potentially utilize from lesser-known models.

To gain a deeper understanding of designers' challenges when using current AI models in their creative processes, we conducted a formative study with ten designers. From our formative study, we identified four key challenges:

(1) Designers are often unaware of the full range of capabilities offered by different types of foundation models.
(2) Designers struggle with the need to be "AI-friendly," which includes difficulties in forming effective prompts and selecting optimal parameters.
(3) Designers find it challenging to cross-integrate foundation models that exist on different platforms and are specialized for different modalities.
(4) Designers find prototyping with these models to be a slow and arduous process.

Based on the findings from the formative study, we derived four design goals, which informed the development of Jigsaw, a block-based prototype system that represents foundation models as puzzle pieces and allows designers to combine the capabilities of different foundation models by assembling compatible puzzle pieces together. Jigsaw includes features that help designers discover available foundation model capabilities and find the right model for their use case. Jigsaw also includes "glue" puzzle pieces that translate design ideas into prompts for other models, clear explanations of parameters to help users make model adjustments, and an Assembly Assistant that recommends potential combinations of foundation models to accomplish a task specified by the designer.

To assess the utility of Jigsaw, we invited ten designers from the formative study to test the system. We evaluate how well designers craft creative AI workflows when given a design brief and during free exploration. The results show that Jigsaw helps designers better understand the capabilities offered by current foundation models, provides intuitive mechanisms for using and combining models across diverse modalities, and serves as a visual canvas for design exploration, prototyping, and documentation. This research thus contributes:

- A **formative study** with ten designers that identifies the challenges designers face when using AI foundation models to support their work.
- **Jigsaw**, a prototype system that assists designers in combining the capabilities of AI foundation models across different tasks and modalities through assembling compatible puzzle pieces.
- A **user study** that demonstrates the utility of Jigsaw to designers and informs areas for future block-based prototyping systems for prototyping with AI foundation models.

2 RELATED WORK

This work draws on prior research in AI foundation models and visual programming interfaces.

2.1 AI Foundation Models

The term "foundation models" characterizes an emerging family of machine learning models [8], often underpinned by the Transformer architecture [30] and trained on vast amounts of data. The researchers who introduced this term defined foundation models as "models trained on broad data (generally using self-supervision at scale) that can be adapted to a wide range of downstream tasks." [3] The strength of foundation models lies in their capacity for out-of-the-box usage across various tasks. This signifies an improvement from the previous AI landscape, where users had to create their own datasets for custom use cases and fine-tune models [5].

Prominent examples of foundation models include large language models such as GPT [9] which can perform a variety of text generation tasks and image generation models such as Stable Diffusion [27] which can generate a diverse range of images from text-based prompts. Foundation models also go beyond generative models and include models for tasks such as classification [24], detection [42], and segmentation [19]. Further, they span a range of modalities including text [9], image [27], video [31], 3D models [17], and audio [21]. Many foundation models perform tasks *across modalities*, such as text-to-x generative models and x-to-text classification models. In turn, this allows foundation models to be treated as x-to-x input-output systems. Such abstraction greatly simplifies how people can use and combine such models in larger AI-enabled systems. Our research aims 1) to inform designers about the capabilities offered by foundation models that can be useful for creative tasks, and 2) to incorporate these capabilities into their creative workflows. In particular, we are interested in exploring how designers can *combine* the capabilities of multiple models *across different tasks and modalities* by connecting them together on a visual interface.

2.2 Visual Programming Interfaces

Visual programming interfaces (VPIs) have been extensively studied as tools to aid users in designing and implementing systems through graphical elements rather than text-based code [22]. A key benefit of VPIs is their lower entry barrier for novice programmers [34]. There are primarily two main paradigms for VPIs. The first, the dataflow paradigm, lets users specify how a program transforms data from step to step by connecting nodes in a directed graph. Pioneering work in this area includes Prograph [11] and LabVIEW

[20]. The second paradigm utilizes block-based function representations and lets users create programs by connecting compatible components together. Notable works in this area include Scratch [26] and Blockly [13]. Many commercial creative applications have adopted VPIs, including game engines such as Unity [6], CAD tools such as Grasshopper [4], and multimedia development tools such as Max/MSP [7].

Recently, VPI concepts have been applied to machine learning applications. For example, *Teachable Machine* [10] uses a visual interface to help students learn to train a machine learning model. *ML Blocks* [35] assists developers in training, evaluating, and exporting machine learning models for edge computing use cases. Wu et al. [37, 38] help developers build language model pipelines through chaining prompts on a node-based interface. Most closely related to our work is *Rapsai* by Du et al. [12], which provides a node-based interface to assist machine learning practitioners in building multimedia machine learning pipelines. Our work builds upon prior VPI tools and research and focuses on building a tool specifically designed to assist designers from non-technical backgrounds. To the best of our knowledge, this research is the first to study 1) supporting *non-technical designers* in prototyping design workflows with AI through a block-based visual interface and 2) utilizing the *plug-and-play capabilities of AI foundation models* that have emerged over the past year, covering a *diverse range of tasks and modalities*.

3 FORMATIVE STUDY

We conducted a formative interview study with ten designers to understand how designers attempt to use AI in their work and inform the development of a new tool to support creative work with AI.

3.1 Participants and Procedure

We interviewed ten designers (P1-P10, 6 male and 4 female, aged 24-39), recruited through known contacts and word of mouth. The designers come from diverse specializations, such as interior design, product design, graphic design, and video game design. All participants have more than five years of design work experience and use AI tools to support aspects of their design processes, such as ChatGPT, Midjourney, and DALL-E [25]. We conducted one-hour interviews remotely over video conferencing, asking participants to describe their typical creative workflow, how they use AI to support their work, the specific AI tools they use, and the pain points they face using AI. Following the interviews, the first author conducted a thematic analysis of interviews and summarized participants' key challenges.

3.2 Findings and Discussion

We identify four key challenges designers face when using AI to support their work.

3.2.1 C1: Understanding AI Model Capabilities. Despite the broad spectrum of AI foundation models available, designers felt they had limited knowledge of existing models and their capabilities. As a result, they felt like they were underutilizing the creative support these models could provide. Designers found it challenging to *"understand the capabilities of various models in a crowded market (P1)"*, making it difficult to determine which model is most suitable for a specific task. Additionally, designers expressed a desire to *"easily view a few example results from the models (P5)"*, which would allow them to quickly assess the model's capabilities and determine if its results align with their intended use case.

3.2.2 C2: Being AI-friendly. After deciding on an AI model to use, designers stated that it is challenging to be "AI-friendly." This includes crafting effective prompts (for generative models) and setting optimal parameter values of AI models to ensure good results. Designers stated it can be time-consuming to *"master the art of prompt creation (P2)"*, often dedicating a significant amount of time to simply translating their design idea into a functional prompt. As P5 stated, *"behind every stunning image generated by Stable Diffusion lies a designer's patience and a relentless pursuit of the right prompt."* Moreover, our participants were often confused about different model parameters and how they affect the model's results, leading to *"endless parameter tweaks (P10)."*

3.2.3 C3: Combining Models. Designers felt that current models predominantly cater to simple and singular functionalities. Designers commented that for realistic design workflows, which involve multiple tasks and a range of modalities, they often find themselves having to switch between distinct AI platforms. This fragmented the design process, and as P9 stated, *"switching between AI platforms felt like needing a different kitchen gadget for every step in a recipe."* In addition, designers often face compatibility issues between models when attempting to combine them, leading to time-consuming troubleshooting. They commented that it would be beneficial to *"clearly know which models are compatible with one another (P6)."*

3.2.4 C4: Slow Prototyping and Iteration. Designers noted that *"seamless prototyping and iteration is crucial to the design process (P6)"*. However, when working with AI, designers frequently found it challenging to quickly build prototypes and view results. Setting up and switching models can be a lengthy process that inhibits rapid experimentation. Furthermore, when creating workflows that involve chaining models, designers often can only view the final result. This makes it difficult to understand how individual models affect the final result and can make it challenging to explain design decisions to clients without tangible intermediate outputs.

3.3 Design Goals

To tackle the challenges designers encounter when using AI, we distill four design goals.

3.3.1 D1: Catalog of AI Foundation Models. To help designers gain a better understanding of available AI foundation models, we aim to compile a catalog of existing models. For each model, we should provide straightforward explanations of its capabilities along with examples of their results. Furthermore, we should provide mechanisms for designers to easily find models that can accomplish the specific tasks they have in mind.

3.3.2 D2: User-friendly instead of AI-friendly. We should provide mechanisms that reduce the need for designers to adapt to the nuances of AI models. First, we should incorporate assisted prompting techniques to help designers translate design ideas into prompts.

Second, we should explain model parameters in laypeople's terms, including how altering different values will impact model results.

3.3.3 D3: Intuitive Interface for Combining Models. We should provide designers with an interface that allows them to easily combine multiple task-specific foundation models across a wide range of modalities. The interface should visually present clear affordances of which models can be combined. In addition, we should provide an assistive tool for suggesting model combinations.

3.3.4 D4: Facilitate Effective Prototyping. Designers place significant importance on experimentation and iteration. We should make it effortless for designers to experiment with different model combinations and be able to easily view results. Furthermore, we should let designers view intermediate results within a chain of models to help diagnose errors and aid in communicating design ideas with clients.

4 JIGSAW

The following outlines Jigsaw's four major components: the (1) Catalog Panel, (2) Assembly Panel, (3) Input and Output Panels, and (4) Assembly Assistant. We then describe how a designer can use Jigsaw with an example interior design workflow.

4.1 Catalog Panel

The *Catalog Panel* assists designers in selecting suitable models for their tasks with a catalog of foundation model components (D1).

4.1.1 Curating a catalog of foundation models. We identified six common modalities used in creative work, namely, text, image, video, 3D, audio, and sketches. Jigsaw curates available models across all possible pairwise permutations of modalities (e.g., `text-to-text`, `text-to-image`, `text-to-video`, ...). Jigsaw also includes foundation models with dual input channels, such as ControlNet [41]. For tasks supported by multiple models, we prioritize models based on: 1) inference speed (ideally less than a minute to run), 2) zero-shot capability (plug-and-play use), and 3) the quality of results (models ranked highly on machine learning benchmarks). Overall, we implemented a catalog of thirty-nine models across six modalities (see Appendix A for a full listing).

4.1.2 Representing foundation models as puzzle pieces. Considering foundational models as input/output systems, we represent them as puzzle pieces with input and output arms. There are two types of puzzle pieces: 1) the *model* piece represents models with customizable parameters, and 2) the *input* piece accepts input from the user via text, media file, or sketch (see Section 4.3). Jigsaw color-codes puzzle pieces based on their input and output modalities to signal what pieces are compatible with one another (D3). For example, a `text-to-image` piece would be colored green on the left and blue on the right. When a user hovers over a puzzle piece, a tooltip provides a description of its capability, typical runtime, and an example of an input and output (D1).

4.1.3 Helping users find the right piece. Jigsaw provides two mechanisms for users to find model pieces for their tasks (D1): 1) puzzle pieces are grouped by input modality and 2) users can describe the task in the semantic search bar. The search returns model pieces with high semantic similarity to the query, scored using CLIP [24] text embeddings .

4.1.4 LLMs as glue. There can be instances where models do not perfectly align, such as situations where intermediate reasoning is required. Drawing inspiration from *Socratic Models* by Zeng et al. [40], we utilize Large Language Models (LLMs) as connecting elements between model pieces. We refer to these instances as *glue* pieces. The user first attaches a model piece capable of conveying the content of a modality in text (x-to-text). Next, the user attaches the LLM glue piece for language-based reasoning (text-to-text). Finally, the user attaches a model piece which translates text back into another modality (text-to-x). To help users connect model pieces in common use cases, Jigsaw includes three types of glue pieces:

(1) The *custom* glue piece accepts any custom user instruction.
(2) The *translation* glue piece converts a piece of text into a prompt that better aligns with text-to-x generation models (e.g., Stable Diffusion) (D2). We ask GPT to transform an input into a prompt via the following prompt:

```
Here are example prompts for a text-to-<modality>
generation model: <list of example prompts>.
Transform <input data> into a prompt. Answer in only
the transformed prompt.
```

(3) The *ideation* glue piece accepts a design task specified by the user and generates an idea. We ask GPT to generate an idea via the following prompt.

```
Generate an idea for <task> based on <input data>.
Answer in one short sentence.
```

4.2 Assembly Panel

The *Assembly Panel* offers an infinite canvas for combining compatible foundation model puzzle pieces (D3) (Figure 3). When the user clicks on a model piece, a *parameters sidebar* allows users to customize a model's specific parameters. Jigsaw pre-populates each model's parameters with default values that generally yield good results and defines limits so that the user can experiment with different values without the concern of breaking the model (D2). Tooltips explain, in plain English, how the parameter influences model results and recommends optimal values for common scenarios. Users can build multiple chains on the canvas and can run each chain separately, allowing parallel explorations and complex workflows.

4.3 Input and Output Panels

The *Input and Output Panels* allow users to input, view, and download media across modalities. Users can type into the input panel (Figure 5a), upload files by drag-and-drop or file browser (Figure 5b), or draw sketches (Figure 5c). The Output Panel shows the result of the chain and lets users copy (Figure 5d) or download outputs (Figure 5e-i).

The user can select a puzzle piece to view the intermediate inputs and outputs at that specific piece. This allows the user to observe

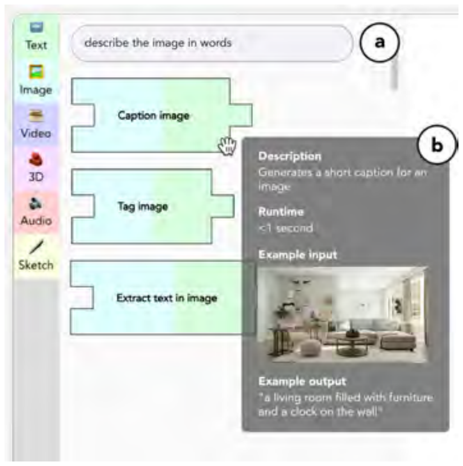

Text
Image
Video
3D
Audio
Sketch
describe the image in words
a
Caption image
Tag image
Extract text in image
b
Description
Generates a short caption for an image
Runtime
<1 second
Example input
Example output
"a living room filled with furniture and a clock on the wall"

Figure 2: Users can search for model pieces by describing their task in the semantic search bar (a). Users can hover over a model piece to view a description of its capability, typical runtime, and an example input and output (b).

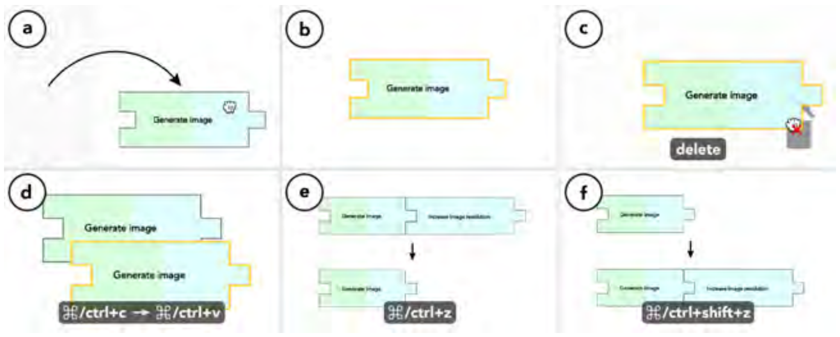


Figure 3: Users can drag puzzle pieces from the Catalog Panel onto the Assembly Panel (a), select pieces on the Assembly Panel by clicking on them (b), and remove pieces by dragging them to the trash bin or pressing the delete key (c). Users can duplicate pieces, and undo and redo actions using hotkeys (d-f).

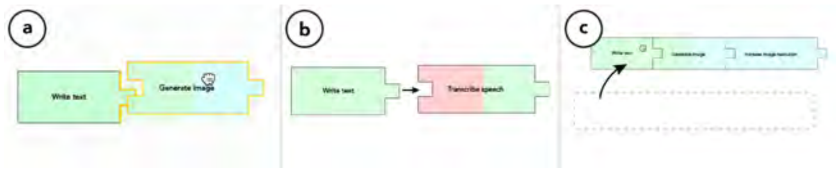
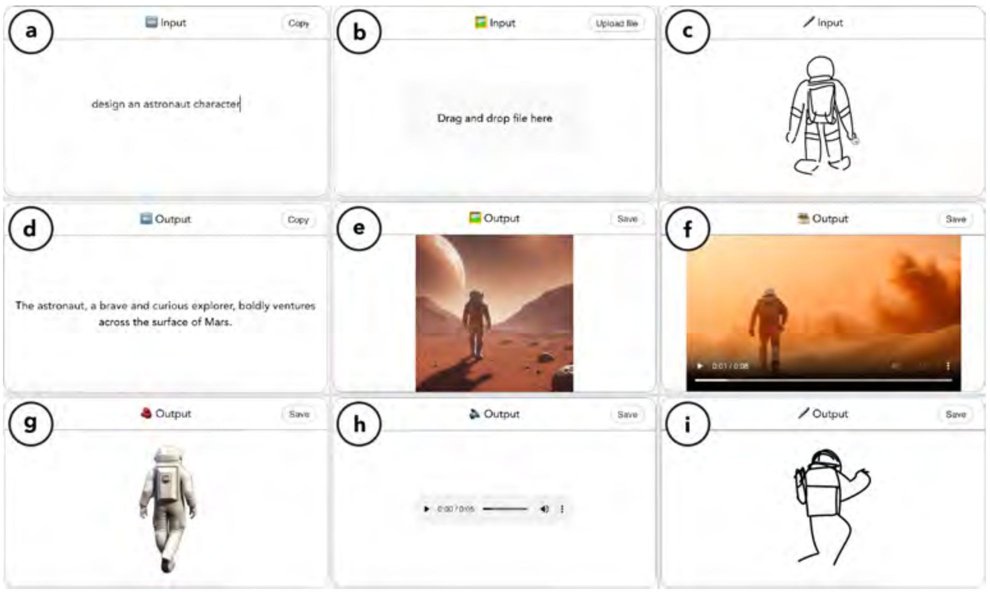


Figure 4: When the user drags a puzzle piece close to another compatible piece, Jigsaw displays a semi-transparent preview of the potential connection. If the user releases the puzzle piece, it will snap into place (a). Conversely, if the user attempts to connect a puzzle piece to an incompatible piece, the new piece will be repelled, ensuring that users do not force a fit (b). Users can move multiple puzzle pieces simultaneously (c).

how the data is transformed at each stage (D4). Additionally, within a chain, the user can view how the inputs for a puzzle piece affect the results of a puzzle piece located several steps downstream by holding the shift key to select multiple puzzle pieces.

4.4 Assembly Assistant

The *Assembly Assistant* recommends a chain of puzzle pieces for a user-specified task. The designer would first provide a natural language description of a task, such as "Add sound effects for an illustration." Jigsaw then asks GPT to use a chain of models to accomplish the task via the following prompt:



design an astronaut character

Drag and drop file here

The astronaut, a brave and curious explorer, boldly ventures across the surface of Mars.

Figure 5: Text inputs can be directly typed into the Input Panel (a). Image, video, 3D, and audio inputs can be uploaded either by drag-and-drop or the file browser (b). Sketch inputs can be drawn (c). Text outputs can be viewed and copied by the user (d). Image, video, 3D, audio, and sketch outputs can be viewed in their respective media viewers and downloaded by the user (e-i).

```
You are a helpful assistant given a set of AI models to
complete the user's task.
There are thirty-nine models: <1. text2text() has reasoning
capability. 2. text2img() can generate an image from text.
3. text2video() can generate a video from text, ...>
You can only use the models given. You do not have to use
all the models.
You will answer in a JSON format. Here is an example answer:
 <example combination of puzzle pieces written in a JSON
format for the frontend to parse>.
Your task is to <user-specified task>.
```

Prior work has found that asking GPT to evaluate its own results can improve the correctness of its responses [23]. We thus ask GPT to evaluate its own answers based on four criteria via the following prompt:

```
<Prompt from the previous step>
<Answer from the previous step>
Here are four criteria that the answer needs to satisfy. If
any criteria are not satisfied, please give me the corrected
 answer in JSON format.
1. Whether the user's task was understood and completed.
2. Whether no models outside of the provided ones were used.
3. Whether the output and input of each step can be
connected.
4. Whether it follows the correct JSON format.
```

Jigsaw then passes the chain of puzzle pieces provided in JSON format to the frontend and adds them onto the Assembly Panel. The designer can make further edits to the chain, just like any manually-created chain.

4.5 System walkthrough

We illustrate the interactions supported by Jigsaw using an interior design example. Figure 6 displays the final arrangement of puzzle pieces, referred to as the "model mosaic" in this paper for succinctness.

4.5.1 Ideating a design concept from client material. Zaha is an interior designer tasked with creating and presenting a redesigned interior for a client's new home. She received a photograph of the interior from the client.

To begin, Zaha plans to create a design concept using the client's photo as a reference (Figure 6a). She drags an `Upload image` puzzle piece from the Catalog Panel onto the Assembly Panel, and uploads the client's photo in the Input Panel. Zaha first identifies existing features in the client's home, such as built-in structures, furniture, and lights. Thus, Zaha uses the semantic search bar in the Catalog Panel to find a puzzle piece that can *"identify the objects inside the image."* Jigsaw returns the `Tag image` piece as the top result. She adds `Tag image` after `Upload image` to identify the objects. Zaha then uses the `Ask GPT` piece in *ideation* mode, with *"contemporary interior concept"* in the *Task* box, to assist her in brainstorming a concept. After clicking *Run*, Jigsaw has suggested a design concept of *"An airy space with a minimalist fireplace and ladder, illuminated by low-hanging lamps."*

4.5.2 Designing the 2D and 3D mockups. With a design concept in hand, Zaha proceeds to create the visual design (Figure 6b). Zaha quickly duplicates the `Upload image` piece to start a new chain, using the client's photo as a reference again. She notices that the reference photo includes people, whom she wants to remove, and

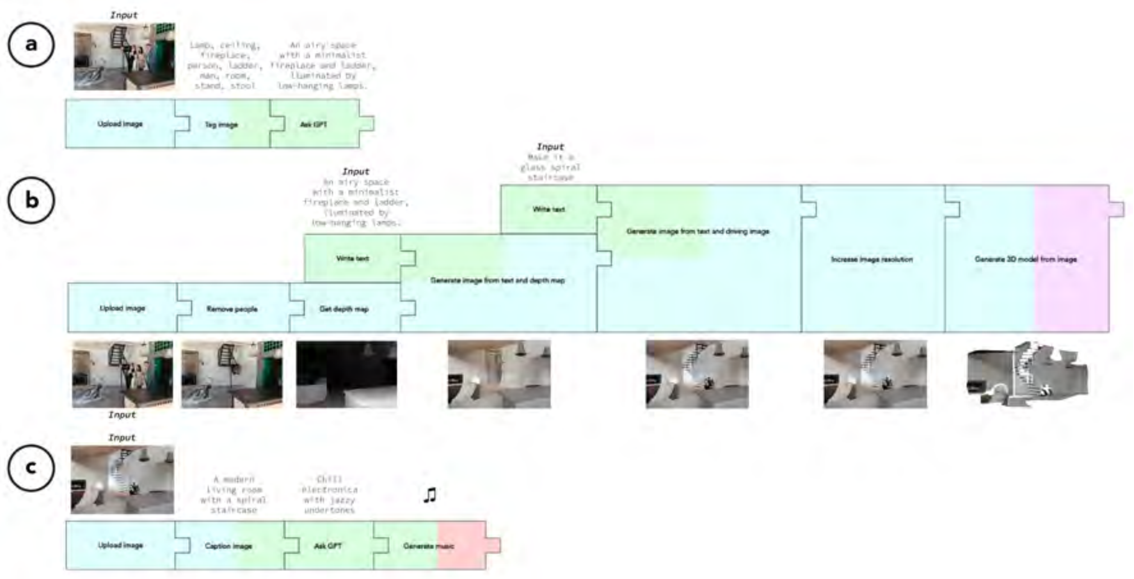

a
Input
Lamp, ceiling, fireplace, person, ladder, man, room, stand, stool
An airy space with a minimalist fireplace and ladder, illuminated by low-hanging lamps.
Upload image
Tag image
Ask GPT
b
Input
An airy space with a minimalist fireplace and ladder, illuminated by low-hanging lamps.
Input
Make it a glass spiral staircase
Write text
Write text
Generate image from text and driving image
Increase image resolution
Generate 3D model from image
Generate image from text and depth map
Upload image
Remove people
Get depth map
Input
c
Input
A modern living room with a spiral staircase
Chill electronica with jazzy undertones
Upload image
Caption image
Ask GPT
Generate music

Figure 6: An example model mosaic for interior design. The designer can use Jigsaw to ideate a design concept from client material (a), design the 2D and 3D mockups (b), and add music to enhance the presentation (c).

adds the `Remove people` piece. Zaha is interested in experimenting with AI image generation tools but acknowledges that the room's structure must remain intact for the design to be technically feasible. She begins by testing the `Get edge map` piece and the `Generate image from text and edge map` piece, which takes in both image and text inputs. For text, Zaha inputs the design concept suggested in the previous ideation chain. Zaha feels that the generated image fails to retain the desired room structure. Recognizing this, she drags the puzzle pieces using edge maps into the trash bin. She tries using `Get depth map` and `Generate image from text and depth map` instead, which better preserves the room's structure. To try different design variations, Zaha inspects the parameters tooltip for the `Generate image from text and depth map` model in the parameters sidebar. She discovers that she can tinker with the seed value to generate different variations.

Zaha is now satisfied with the redesign, except for the wooden ladder. She believes replacing it with a spiral glass staircase would better fit the contemporary concept. Thus, she searches for a puzzle piece that can modify an image using text instructions and finds the `Generate image from text and image` piece. Zaha instructs it to *"replace the wooden ladder with a glass spiral staircase."* The newly generated redesign now features a contemporary glass spiral staircase.

Zaha would like to visualize a 3D mockup. She discovers the `Generate 3D model from image` piece and attaches this to the chain, but finds that the generated results are low resolution. Thus, Zaha searches the *Image* section of the Catalog Panel for a piece that can help enhance the image. She finds the `Increase image resolution` piece.

4.5.3 Presenting the design to the client. To communicate the contemporary design concept, Zaha would like to incorporate a musical background to complement the design aesthetics. To achieve this, Zaha asks the Assembly Assistant to *"help add music based on the image."* The Assembly Assistant suggests the following chain of puzzle pieces: `Caption image` to understand the image, `Ask GPT` with *ideation* mode to brainstorm a fitting music description, and `Generate music` to generate the music (Figure 6c). The outcome is a chill electronic music piece.

5 USER STUDY

We conducted a user study to understand how Jigsaw could address designers' pain points in working with multiple foundation modes, its potential to be integrated into design workflows, and identify improvement areas.

5.1 Participants and Procedure

We invited the ten designers from our formative interviews to participate in a one-hour remote user study. They were not exposed to Jigsaw's system or concept prior to the user study. Participants accessed Jigsaw through a web browser, shared their screen, and verbally explained what they were doing and thinking (think-aloud).

5.1.1 Introduction (10 minutes). Participants provided informed consent, and then received an introduction to Jigsaw's components, as described in Section 4.

5.1.2 Reproduction Task (15 minutes). Participants were asked to reproduce the interior design model mosaic described in Section 4.5. Participants used the starter image shown in Figure 6a and a detailed design brief of the various steps they would need to create (see Section 4.5).

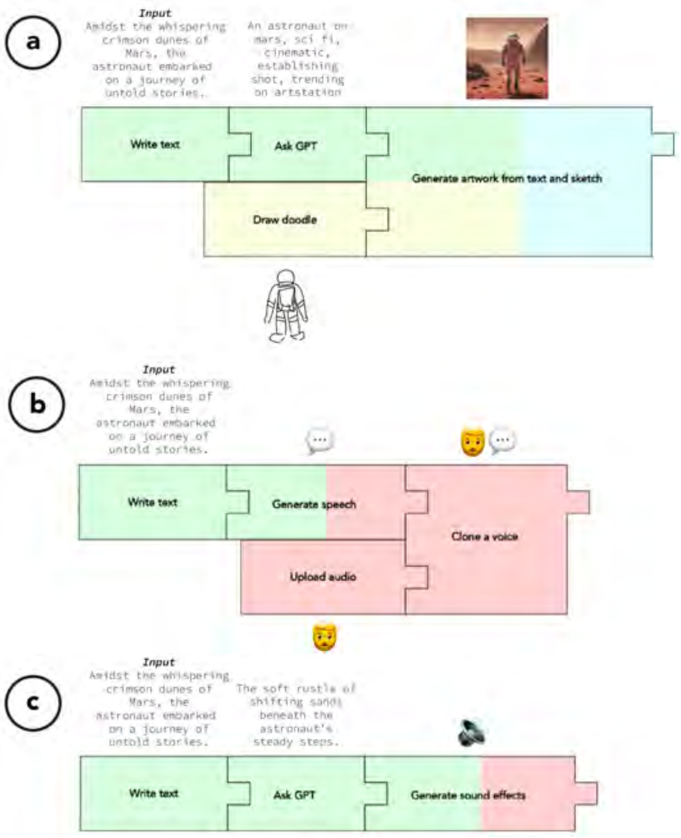

a
Input
Amidst the whispering crimson dunes of Mars, the astronaut embarked on a journey of untold stories.
An astronaut on mars, sci fi, cinematic, establishing shot, trending on artstation
Write text
Ask GPT
Draw doodle
Generate artwork from text and sketch
b
Input
Amidst the whispering crimson dunes of Mars, the astronaut embarked on a journey of untold stories.
Write text
Generate speech
Upload audio
Clone a voice
c
Input
Amidst the whispering crimson dunes of Mars, the astronaut embarked on a journey of untold stories.
The soft rustle of shifting sands beneath the astronaut's steady steps.
Write text
Ask GPT
Generate sound effects

Figure 7: Model mosaic by P2, an illustrator, to create an audio-visual story. P2 uses Jigsaw to create an illustration based on a text description and a reference sketch (a), generate narrations through a cloned voice (b), and generate accompanying sound effects (c).

5.1.3 Free Creation Task (20 minutes). Participants were asked to freely explore Jigsaw and create their own model mosaics. We encouraged participants to build workflows beyond a simple chain and try out puzzle pieces involving multiple modalities.

5.1.4 Post-Study Interview (15 minutes). After the creation activities, we conducted a semi-structured interview asking about participants' experiences using Jigsaw, whether they could see Jigsaw being integrated into their design workflow, and to identify areas for improving the system.

5.2 Results, Discussion, and Future Work

All participants completed the reproduction and free creation tasks. Jigsaw appears to help designers discover and prototype new creative workflows. Designers suggested different future improvements.

5.2.1 Helping designers discover and utilize AI capabilities. Participants located AI capabilities via the Catalog Panel's semantic search bar or by filtering pieces by modality. Many participants mentioned that they were able to *"discover new AI abilities [they were] previously unaware of (P9)"*. For example, P2, an illustrator, discovered the capabilities of ControlNet [41], a model that allows users to add additional control to a text-to-image model, such as a guiding sketch. Figure 7 shows the model mosaic created by P2, who used Jigsaw to create an audio-visual story. In Figure 7a, she created visuals for her story. Instead of using a text-to-image model, she discovered and used ControlNet to generate images based on a starting sketch. In total, participants used 8 model puzzle pieces on average (μ=7.9, σ=1.29), and all participants explored beyond well-known models such as GPT and Stable Diffusion (see Appendix B for details). As the number of foundation models continues to increase, we plan to expand our set of puzzle pieces for designers over time.

Participants commented that tooltips *"gave [them] a solid understanding of the capabilities of each of the puzzle pieces, like what can be expected as output and what types of inputs are suitable (P1)"*. In addition, participants expressed that the assisted prompting mechanism offered by the *translation* glue piece *"allowed [them] to achieve satisfying results without the need to laboriously rephrase and tweak prompts (P2)"*: *"Now, it's like I speak the AI's language. (P5)"*. For improvements, we are interested in expanding Jigsaw to let designers define custom puzzle pieces, as suggested by P10, and logic operators such as if/else statements and loops, common in other VPIs [7].

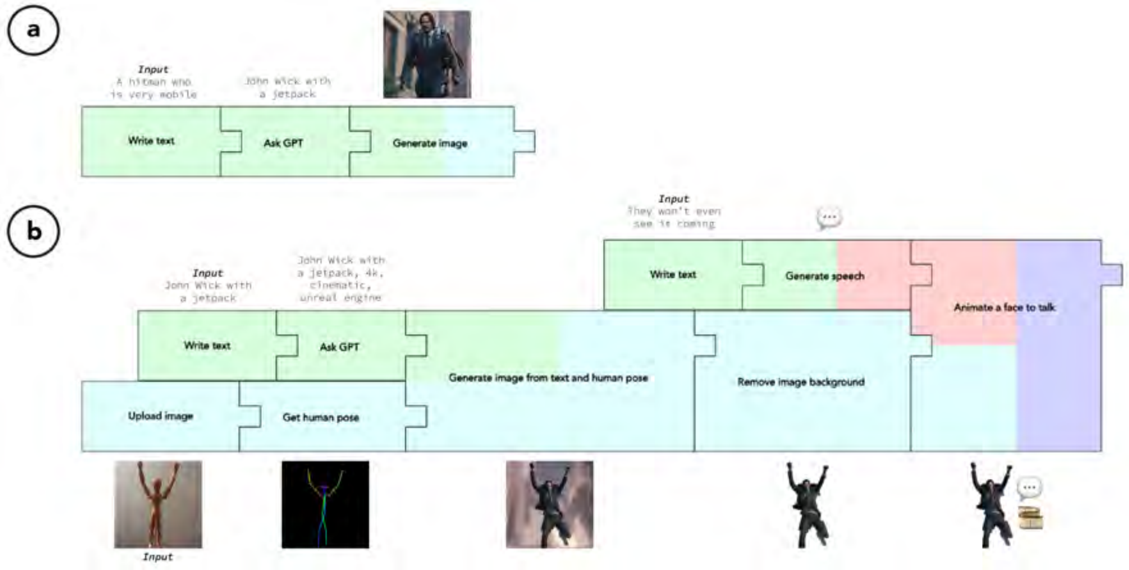


Figure 8: Model mosaic by P10, a game designer, to create a video game character. P10 uses Jigsaw to create a character concept and preview the character's visuals (a). P10 then specifies a pose for the character and generates the character's visuals in the specified pose (b). P10 then generates a line for the character to say and animates the character to deliver the line.

5.2.2 *Supporting intuitive prototyping.*

Participants expressed that *"[they] enjoyed the idea of building with AI visually with tangible puzzle pieces (P10)"* and found it *"easy to pick up and start designing (P3)"*. In particular, participants appreciated the error-proof design: *"Knowing which puzzle pieces can be connected expedites my prototyping. I see the same colors and receive snapping feedback. I don't spend time building a workflow and then compile it to find compatibility errors (P3)"*. A contribution of this research is showing how the design benefits of block-based VPIs, commonly catered towards novice programming learners [13, 26, 33], can be effectively applied to the realm of design prototyping for non-technical designers to work with AI capabilities. For future work, an interesting extension of Jigsaw could be the implementation of a "tutorial mode" to teach novice designers. The system would disassemble a model mosaic created by an experienced designer into pieces, allowing a novice designer to recreate it and learn from the experienced designer's design process.

Furthermore, participants mentioned that the ability to near-instantaneously see intermediate outputs in a chain *"helped [them] to quickly test out ideas and make adjustments to individual steps as needed (P5)"*. This aligns with findings from prior research in interactive program debugging tools [15, 18, 28]. For improvements, we plan to expand the Input and Output panels to handle real-time video and audio streams, as suggested by P9 and available in [35].

5.2.3 *Serving as a brainstorming and documentation canvas.*

We observed participants making creative uses of the Assembly Panel, including using it to test different partial workflows before combining them into more complete workflows and using it to document their design explorations. Participants expressed that the Assembly Panel provides *"a playground to be messy and experimental (P3)"* and *"makes it easy to track the evolution of an idea (P4)."* Moreover, participants commented that the *ideation* glue piece was *"helpful for brainstorming concepts at the beginning [of the design process] (P2)"*. We observed that designers occasionally passed the outputs of the *ideation* glue directly into a generation model, as shown in Figure 7c. In other instances, designers maintained a shorter chain solely for concept generation. This is shown in the model mosaic created by P10, a game designer, in Figure 8, who created a video game character. He primarily used the short chain in Figure 8a to generate concepts for his character. We observed that designers frequently created multiple chains to organize different stages of their design process. Participants noted that since the canvas documents their creative process, it could serve as *"a boundary object for sharing and explaining ideas to clients (P5)"*.

Moreover, participants commented that the Assembly Assistant was useful in *"generating an initial configuration of puzzle pieces to start working with (P1)"*. This aids in combating the *"blank canvas syndrome (P6)"*, a common occurrence at the onset of a creative activity [16]. In Figure 8b, P10 wanted his video game character to look like Superman flying. However, he initially struggled to come up with a method to accomplish this, so he sought assistance from the Assembly Assistant. The Assembly Assistant recommended a workflow of a reference pose image, a pose extraction model, and a ControlNet model that can be guided by pose. Given this workflow, P10 used a wooden mannequin to specify the pose for his character. For improvement, participants noted that the Assembly Assistant was less suitable for ambiguous tasks with multiple solutions. This is a common challenge recognized by recent machine learning works that also aim to *automatically* combine expert models to solve complex AI tasks [14, 29, 36]. A path forward, as suggested by P7, could be to improve the Assembly Assistant to support *back-and-forth interactions* with the designer, becoming a design co-pilot that assists designers in creating complex workflows. Moreover, as more designers use Jigsaw to create model mosaics, we plan to compile them into a template gallery that other designers can modify for their own use cases, as suggested by P8. We believe that the accumulated design templates can also serve as a search space for the Assembly Assistant, enhancing its capabilities as a design search engine.

6 CONCLUSION

This research identifies the challenges designers face when using AI foundation models to support their work. The research prototype, Jigsaw, uses a puzzle piece metaphor to represent foundation models and allows the combination of models by assembling compatible pieces. Feedback from designers using Jigsaw demonstrated that designers discovered new AI capabilities, combined multiple AI capabilities across various modalities, and flexibly explored, prototyped, and documented AI-enabled design workflows. We are interested in extending Jigsaw with more capabilities and hope that this research can help inform future research on block-based systems for prototyping with AI foundation models.

REFERENCES

[1] 2023. *ChatGPT*. Retrieved August 15, 2023 from https://chat.openai.com/

[2] 2023. *Midjourney*. Retrieved August 15, 2023 from https://www.midjourney.com/

[3] 2023. *Reflections on Foundation Models*. Retrieved August 15, 2023 from https://hai.stanford.edu/news/reflections-foundation-models

[4] 2023. *Rhino - Grasshopper - New in Rhino 6*. Retrieved August 15, 2023 from https://www.rhino3d.com/6/new/grasshopper/

[5] 2023. *Transfer learning and fine-tuning*. Retrieved August 15, 2023 from https://www.tensorflow.org/tutorials/images/transfer_learning

[6] 2023. *Unity Visual Scripting*. Retrieved August 15, 2023 from https://unity.com/features/unity-visual-scripting

[7] 2023. *What is Max? | Cycling '74*. Retrieved August 15, 2023 from https://cycling74.com/products/max

[8] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021).

[9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[10] Michelle Carney, Barron Webster, Irene Alvarado, Kyle Phillips, Noura Howell, Jordan Griffith, Jonas Jongejan, Amit Pitaru, and Alexander Chen. 2020. Teachable machine: Approachable Web-based tool for exploring machine learning classification. In *Extended abstracts of the 2020 CHI conference on human factors in computing systems*. 1–8.

[11] Philip T Cox, FR Giles, and Tomasz Pietrzykowski. 1989. Prograph: a step towards liberating programming from textual conditioning. In *1989 IEEE Workshop on Visual languages*. IEEE Computer Society, 150–151.

[12] Ruofei Du, Na Li, Jing Jin, Michelle Carney, Scott Miles, Maria Kleiner, Xiuxiu Yuan, Yinda Zhang, Anuva Kulkarni, Xingyu Liu, et al. 2023. Rapsai: Accelerating Machine Learning Prototyping of Multimedia Applications through Visual Programming. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–23.

[13] Neil Fraser et al. 2013. Blockly: A visual programming editor. *URL: https://code. google. com/p/blockly* 42 (2013).

[14] Rongjie Huang, Mingze Li, Dongchao Yang, Jiatong Shi, Xuankai Chang, Zhenhui Ye, Yuning Wu, Zhiqing Hong, Jiawei Huang, Jinglin Liu, et al. 2023. Audiogpt: Understanding and generating speech, music, sound, and talking head. *arXiv preprint arXiv:2304.12995* (2023).

[15] Peiling Jiang, Fuling Sun, and Haijun Xia. 2023. Log-it: Supporting Programming with Interactive, Contextual, Structured, and Visual Logs. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–16.

[16] Caneel K Joyce. 2009. *The blank page: Effects of constraint on creativity*. University of California, Berkeley.

[17] Heewoo Jun and Alex Nichol. 2023. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463* (2023).

[18] Gábor Karsai. 1995. A configurable visual programming environment: A tool for domain-specific programming. *Computer* 28, 3 (1995), 36–44.

[19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. *arXiv preprint arXiv:2304.02643* (2023).

[20] Jeffrey Kodosky. 2020. LabVIEW. *Proceedings of the ACM on Programming Languages* 4, HOPL (2020), 1–54.

[21] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. 2022. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352* (2022).

[22] Brad A Myers. 1990. Taxonomies of visual programming and program visualization. *Journal of Visual Languages & Computing* 1, 1 (1990), 97–123.

[23] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350* (2022).

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

[25] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*. PMLR, 8821–8831.

[26] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, et al. 2009. Scratch: programming for all. *Commun. ACM* 52, 11 (2009), 60–67.

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

[28] Eldon Schoop, Forrest Huang, and Bjoern Hartmann. 2021. Umlaut: Debugging deep learning programs using program structure and model behavior. In *Proceedings of the 2021 CHI conference on human factors in computing systems*. 1–16.

[29] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. *arXiv preprint arXiv:2303.17580* (2023).

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[31] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. 2023. ModelScope Text-to-Video Technical Report. *arXiv preprint arXiv:2308.06571* (2023).

[32] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682* (2022).

[33] David Weintrop and Uri Wilensky. 2015. To block or not to block, that is the question: students' perceptions of blocks-based programming. In *Proceedings of the 14th international conference on interaction design and children*. 199–208.

[34] Kirsten N Whitley and Alan F Blackwell. 1997. Visual programming: the outlook from academia and industry. In *Papers presented at the seventh workshop on Empirical studies of programmers*. 180–208.

[35] Randi Williams, Michał Moskal, and Peli De Halleux. 2022. ML Blocks: A Block-Based, Graphical User Interface for Creating TinyML Models. In *2022 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 1–5.

[36] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. 2023. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671* (2023).

[37] Tongshuang Wu, Ellen Jiang, Aaron Donsbach, Jeff Gray, Alejandra Molina, Michael Terry, and Carrie J Cai. 2022. Promptchainer: Chaining large language model prompts through visual programming. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. 1–10.

[38] Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts. In *Proceedings of the 2022 CHI conference on human factors in computing systems*. 1–22.

[39] Nur Yildirim, Changhoon Oh, Deniz Sayar, Kayla Brand, Supritha Challa, Violet Turri, Nina Crosby Walton, Anna Elise Wong, Jodi Forlizzi, James McCann, and John Zimmerman. 2023. Creating Design Resources to Scaffold the Ideation of AI Concepts. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference* (Pittsburgh, PA, USA) *(DIS '23)*. Association for Computing Machinery, New York, NY, USA, 2326–2346. https://doi.org/10.1145/3563657.3596058

[40] Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, et al. 2022. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598* (2022).

[41] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding Conditional Control to Text-to-Image Diffusion Models.

[42] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. 2022. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*. Springer, 350–368.

A LIST OF IMPLEMENTED MODELS

Puzzle piece name	Model name	Input	Output
Ask GPT	GPT-4	Text	Text
Generate image	Stable Diffusion	Text	Image
Generate video	AnimateDiff	Text	Video
Generate 3D model	Shap-E	Text	3D
Generate sound effects	AudioGen	Text	Audio (Sound Effects)
Generate music	MusicGen	Text	Audio (Music)
Generate speech	Bark	Text	Audio (Speech)
Describe image	BLIP	Image	Text
Tag image	Segment Anything	Image	Text
Extract text in image	PyTesseract	Image	Text
Classify emotion from face	ResidualMaskingNetwork	Image	Text
Increase image resolution	Real-ESRGAN	Image	Image
Restore distorted face	GFPGAN	Image	Image
Grayscale → Color	BigColor	Image	Image
Remove image background	Rembg	Image	Image
Remove people	EdgeConnect	Image	Image
Get human pose	OpenPose	Image	Image (Pose)
Get segmentation map	Segment Anything	Image	Image (Segmentation)
Get depth map	MiDaS	Image	Image (Depth)
Get normal map	MiDaS	Image	Image (Normal)
Get edge map	HED	Image	Image (Edge)
Generate 3D model from image	One-2-3-45	Image	3D
Classify video	X-CLIP	Video	Text
Remove video background	Robust Video Matting	Video	Video
Increase video resolution	RealBasicVSR	Video	Video
Increase video frame rate	RIFE	Video	Video
Classify music genre	MusicClassification	Audio	Text
Transcribe speech	Whisper	Audio	Text
Generate image from text and driving image	Instruct-Pix2Pix	Image + Text	Image
Edit face with text	StyleCLIP	Image + Text	Image
Generate image from text and human pose	ControlNet	Image (Pose) + Text	Image
Generate image from text and segmentation map	ControlNet	Image (Segmentation) + Text	Image
Generate image from text and depth map	ControlNet	Image (Depth) + Text	Image
Generate image from text and normal map	ControlNet	Image (Normal) + Text	Image
Generate image from text and edge map	ControlNet	Image (Edge) + Text	Image
Animate a face to talk	SadTalker	Image + Audio	Video
Clone a voice	FreeVC	Audio + Audio	Audio
Generate image from text and sketch	ControlNet	Sketch + Text	Image
Generate artwork from text and sketch	Scribble Stories	Sketch + Text	Image

Table 1: Full list of implemented models.

B MODEL MOSAICS CREATED BY PARTICIPANTS

Participant	Free creation design	Names of puzzle pieces used	Number of puzzle pieces used
P1	Designing a formal dress	Ask GPT (custom, ideation, translation), Tag image, Get human pose, Generate image from text and pose, Remove background, Generate image from text and image	8
P2	Creating an audio-visual story	Ask GPT (ideation, translation), Generate artwork from text and sketch, Generate speech, Clone a voice, Generate sound effects	6
P3	Creating a music video	Ask GPT (custom, ideation, translation), Generate speech, Clone a voice, Classify music genre, Generate music, Generate video, Increase video frame rate	9
P4	Creating multiple movie endings	Ask GPT (custom, ideation, translation), Grayscale → Color, Restore distorted face, Generate image, Generate video, Generate music, Increase video resolution, Increase video frame rate	10
P5	Designing a living room	Ask GPT (ideation, translation), Describe image, Get segmentation map, Get depth map, Generate image from text and segmentation map, Generate image from text and depth map, Remove background, Remove people	9
P6	Designing a tote bag	Ask GPT (ideation, translation), Generate image, Remove background, Generate image from text and image, Get edge map, Generate image from text and edge map, Generate artwork from text and sketch	8
P7	Designing a logo	Ask GPT (custom, ideation, translation), Tag image, Generate image, Generate image from text and sketch, Generate artwork from text and sketch	7
P8	Enhancing a story for the visually impaired	Ask GPT (ideation, translation), Extract text in image, Describe image, Classify emotion from face, Generate speech, Generate sound effects, Generate music	8
P9	Designing a 3D model of a vase	Ask GPT (ideation, translation), Generate 3D model, Generate image, Remove background, Generate 3D model from image	6
P10	Designing a video game character	Ask GPT (ideation, translation), Generate image, Get human pose, Generate image from text and human pose, Remove image background, Generate speech, Animate a face to talk	8

Table 2: Model mosaics created by participants during the free creation task.