DirectGPT: A Direct Manipulation Interface to Interact with Large Language Models

Damien Masson*

Cheriton School of Computer Science, University of Waterloo
Waterloo, Canada
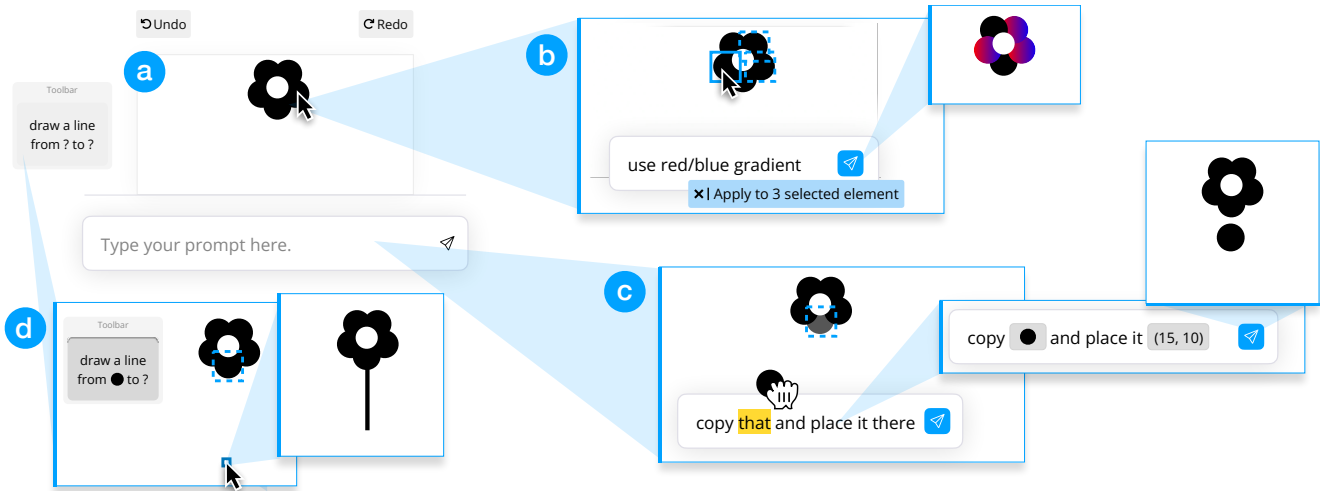dmasson@uwaterloo.ca

Sylvain Malacria†

Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL
Lille, France
sylvain.malacria@inria.fr

Géry Casiez‡§

Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL
Lille, France
gery.casiez@univ-lille.fr

Daniel Vogel

Cheriton School of Computer Science, University of Waterloo
Waterloo, Canada
dvogel@uwaterloo.ca

Figure 1: DirectGPT used on a vector image to demonstrate direct manipulation principles for LLMs: (a) continuous representation of the objects of interest; physical actions to (b) localize the effect of prompts and (c) refer to objects; (d) reusable prompts in a toolbar of commands; and reversible operations through undo and redo features.

ABSTRACT

We characterize and demonstrate how the principles of direct manipulation can improve interaction with large language models. This includes: continuous representation of generated objects of interest; reuse of prompt syntax in a toolbar of commands; manipulable outputs to compose or control the effect of prompts; and undo mechanisms. This idea is exemplified in DirectGPT, a user interface layer on top of ChatGPT that works by transforming direct manipulation actions to engineered prompts. A study shows participants were 50% faster and relied on 50% fewer and 72% shorter prompts to edit text, code, and vector images compared to baseline Chat-GPT. Our work contributes a validated approach to integrate LLMs into traditional software using direct manipulation. Data, code, and demo available at https://osf.io/3wt6s.

*Also with University of Toronto.
†Also with University of Waterloo.
‡Also with Institut Universitaire de France.
§Also with University of Waterloo.

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; **Interaction design theory, concepts and paradigms**.

KEYWORDS

direct manipulation, large language models, prompt engineering

1 INTRODUCTION

Given a textual instruction in the form of a "prompt", a large language model (LLM) can generate outputs such as emails, computer code, and vector images. The output often needs to be tweaked by conversing with the LLM until obtaining a satisfactory result. However, this tweaking process uses relatively slow text input, it requires precise references to elements in the output, and the wording must be carefully chosen to workaround unintuitive limitations of LLMs [57, 64, 86].

These issues are reminiscent of the reasons why *direct manipulation interfaces* emerged as an alternative to *command line interfaces* [30, 67, 68]. The problems identified 40 years ago also apply to interfaces for LLMs today: (i) *indirect engagement* due to manipulating natural language instead of the objects of interest; (ii) *semantic distance* due to the verbosity and difficulty to convey intent in a prompt; and (iii) *articulatory distance* due to the form of prompts being poor representations of the actions they are meant to convey. In fact, interfaces for LLMs like OpenAI's ChatGPT [56], Google's Bard [25], and Microsoft's Bing [49] undermine most of what makes an interface direct. They rely on a linear textual history instead of continuous representation of the objects of interest; carefully crafted prompts remain in the history instead of commands that are rapid, incremental, and reversible; and complex language is manipulated instead of the objects of interest. As a result, current interaction with LLMs lacks most of the benefits of direct manipulation such as improved learnability, speed of execution, goal feedback, and error prevention and recovery [30, 68, 70].

Consider the task of tweaking a generated SVG drawing. Directly pointing at shapes and locations using a mouse cursor is much easier than specifying them using words that may rely on metaphors or geometric references. Besides being verbose, these high-level descriptions are quite far from the SVG code that the model generates. There is also ambiguity if the shape appears multiple times or if the wording assumes spatial information that does not align with how the model represents the drawing. Moreover, applying the same transformation to a different shape requires going through this process again. These difficulties apply when tweaking generated text as well. Consider how replacing a specific word with a synonym in a body of text requires a careful description of the word to change and its position, the action to be done, and other constraints such as keeping the rest of the text intact. Regardless of what is being generated, mistakes are costly as incorrect results are hard to notice and they remain in the conversation history, possibly impacting future interactions.

Instead of solely relying on words, we introduce and characterize prompting through direct manipulation as a way to facilitate conversations with LLMs. Directness is created by interacting with the output to refer to specific objects when crafting new prompts (fig. 1c), or to localize the desired area of effect for the prompt (fig. 1b). Reversibility is realized through typical undo and redo

features instead of long linear conversations. Immediate feedback is provided by highlighting localized changes and elements about to be modified. And rapid reusable commands are generated by abstracting previous prompts into templated snippets (fig. 1d).

To explore the benefits of these principles for LLMs, we implemented them into a prototype system called DirectGPT. In a user study we compare DirectGPT to ChatGPT for text, code, and vector image editing tasks, and found that participants relied and preferred the mechanisms of direct manipulation when accomplishing specific and localized tasks. Specifically, participants used 50% fewer and 72% shorter prompts all while being 50% faster and 25% more successful at accomplishing tasks. Beyond our system and its interactions, our findings inform the design and integration of LLMs and other prompt-driven AI into graphical user interface applications.

2 BACKGROUND AND RELATED WORK

Direct manipulation is often opposed to intelligent agents when debating controllable systems [69] versus autonomous systems [46]. This work takes the perspective that both can work together to enable *"increased automation that amplifies the productivity of users and gives them increased capabilities in carrying out their tasks, while preserving their sense of control and their responsibility"* [29, 71]. We begin by defining what we mean by *direct manipulation* and then review issues with LLMs and how prior work has attempted to solve them through more direct interfaces.

2.1 What is Direct Manipulation?

Direct manipulation is a term coined by Ben Shneiderman to describe characteristics of interactive systems that were found to *"generate glowing enthusiasm among users"* [67, 68]. Shneiderman found these systems to share three principles: (1) *"Continuous representation of the object of interest"*, ideally in its final form [67]; (2) *"Physical actions or labelled button presses instead of complex syntax"*, especially if they involve visual selection of objects of interest [30]; and (3) *"Rapid, incremental, reversible operations whose impact on the object of interest is immediately visible"* [67]. We use the term *direct manipulation* to refer to these principles, and we consider interfaces to be more or less *direct* depending on how thoroughly they implement these principles.

2.2 Five Issues With Prompting that Motivate the use of Direct Manipulation

First, *intent is hard to convey with prompts* due to language being ambiguous in many tasks. This is particularly true for structured information such as graphs and tables [41] and information poorly described in words, such as images [19]. Users need to find the right phrasing that describes the task and that the system understands [40]. But, similar to how vocabulary mismatch is the leading cause of failure when new users try to verbalize their goals [54], users are unlikely to know the best vocabulary to describe certain objects or tasks. Instead, direct manipulation interfaces reduce ambiguity by referring to objects directly instead of describing them.

Second, *conversation is a poor metaphor to design effective and concise prompts*. The nature of LLMs leads to a conversational interface and style of AI voice that encourages users to write prompts the way they would address a human. As a result, some effective

prompt engineering strategies such as giving examples or repeating instructions multiple times are rarely used because they are unnatural in human-human conversation [55, 86]. Additionally, users tend to use polite language such as "thank you" or "can you [...] please?" which adds to the verbosity of prompts and slows down the interaction [86]. Instead, direct interfaces remove this expectation, and specific prompt engineering strategies can be hidden into the internal functioning of the interface.

Third, *the effect of a prompt is hard to control and predict* due to producing different and possibly unexpected results after a slight rewording [44, 76]. An example are "do not" statements that LLMs are known to ignore, or worse, treat as "do" statements [86]. As a result, trying to edit a specific part of the generated output often leads the LLM to modify other parts that should remain untouched. In contrast, our direct interface helps users convey constraint and enforces them instead of solely relying on the LLM.

Fourth, *prompt engineering requires a lot of trial-and-error* before obtaining a satisfactory result [17, 19]. In contrast to conversational interfaces, direct interfaces facilitate trial-and-error behaviours [20, 22, 48]. For example, manipulating objects and clicking buttons make for fast iterations easily recovered from with undo features.

Fifth, *prompts are hard to reuse* since they are each designed for a single, specific interaction [19]. In a conversational interface, the prompt specifies both the action to be done and the object to modify. As such, reusing a prompt requires editing part of the describing the object. In contrast, direct manipulation interfaces typically rely on two distinct steps to specify the action and the object. Our work describes how this specificity can be leveraged to populate a toolbar of commands that help reuse prompts on new objects.

2.3 Systems to Help Craft Better Prompts

When the LLM responds unexpectedly, rewording the prompt may help. To assist in this task, many strategies have been suggested [41, 57] and prior work proposed systems to help with *prompt engineering*. For example, PromptIDE [73] and PromptAid [51] provide interactive interfaces to test different prompts, evaluate their performance, and suggest variations. When a single prompt is not enough, AI Chains [82], PromptChainer [81], and ChainForge [3] help construct chains of prompts and evaluate their effectiveness.

These systems sometimes offer interfaces to make it easier to adjust prompts, change configurations, and reuse previous prompts and outputs. However, they are typically used to evaluate and generate prompts to be integrated as part of bigger pipelines to solve recurrent problems. In contrast, our work uses LLMs to manipulate objects of interest directly such as editing text and images.

2.4 Prompting through More Direct Interactions

While interfaces like ChatGPT are strictly conversational, alternative interfaces have included some ideas from direct manipulation to control generative AIs for certain tasks.

2.4.1 Labelled Buttons Instead of Verbose Prompts. Most integration of generative AIs rely on widgets such as buttons and sliders to execute a pre-defined prompt or modify parameters of the model. For example, using sliders to manipulate continuous parameters that impact the generation of images [18, 65] and music [43]. Buttons have been used as shortcuts to execute predefined prompts that

ask for more information about a topic [32, 74], edit a story [14, 85], and run macros to generate computer code [31]. Our work also relies on buttons to populate a toolbar of commands, but these are dynamically generated as users interact with the LLM.

2.4.2 Physical Actions Instead of Verbose Prompts. Gestures and physical metaphors have been employed to help tweak the output of generative AIs. For example, pointing at a specific location to refer to elements of an image [42] or force a model to re-generate only a specific part of an image [84]. Similarly, dragging gestures have been used to manipulate spatial attributes of an image such as the pose, facial expression, and layout [60, 78]. Metaphorical gestures in the form of a brushing action were used to control the fortune of characters in a story [13] and express image modifications by mixing prompts through physical actions similar to mixing colours [12]. Our work highlights how physical actions and tool metaphors can be leveraged for more general prompt-based interaction suitable for diverse application domains.

2.4.3 Reversible Operations Instead of Continuous Conversations. Conceptual approaches to interaction histories have been explored in detail [53], yet popular interfaces for LLMs are limited in this regard. For example, with ChatGPT, there is no explicit action to recover from a mistake. Instead, users have the choice to start a new conservation from scratch, edit the conversation history, or compose a prompt asking the model to revert back to a previous state. For the latter, the revert prompt itself is saved in the history which can impact future interactions [44, 76]. Similar to how traditional software may use subjunctive interfaces [45] and parallel paths [75] to help explore and recover from mistakes, recent work proposed node-based and hierarchy-based explorations of the responses generated by LLMs [2, 32, 74]. With these interfaces, users can go back to a previous node and start from this point on to further explore. Instead of using current or novel representations of history, our work proposes using traditional undo mechanisms to support reversibility at the granularity of an operation.

2.4.4 Immediate Feedback Instead of Word-by-Word Generation. Large generative models often require considerable computing time and power, so generating a result immediately remains an open challenge [11, 38]. Most previous work focuses on providing feedback at the speed of the generation. For example, ChatGPT streams a response word-by-word as it is generated. However, such feedback makes it difficult to notice differences between iterations. To make output feel closer to realtime, LLMs can be prompted to generate intermediary parameterized representations that are fast to recalculate. For example, instead of generating an image, the LLM can generate code that generates the image. Then, the parameters of the code can be manipulated with immediate feedback [2]. However, this approach is limited to outputs easily generated through code. Instead, our work exploits spatial input from direct manipulation to provide immediate loading feedback on elements about to be modified to keep users aware of modifications.

2.5 Blending Language and Direct Manipulation

Many systems have sought to combine natural language and direct manipulation to be easier to use. One of the earliest examples is "Put-That-There" [8] where vocal commands specify actions and

(a) Localize prompt effects   (b) Visible Effects   (c) Prompt Reused as a Tool
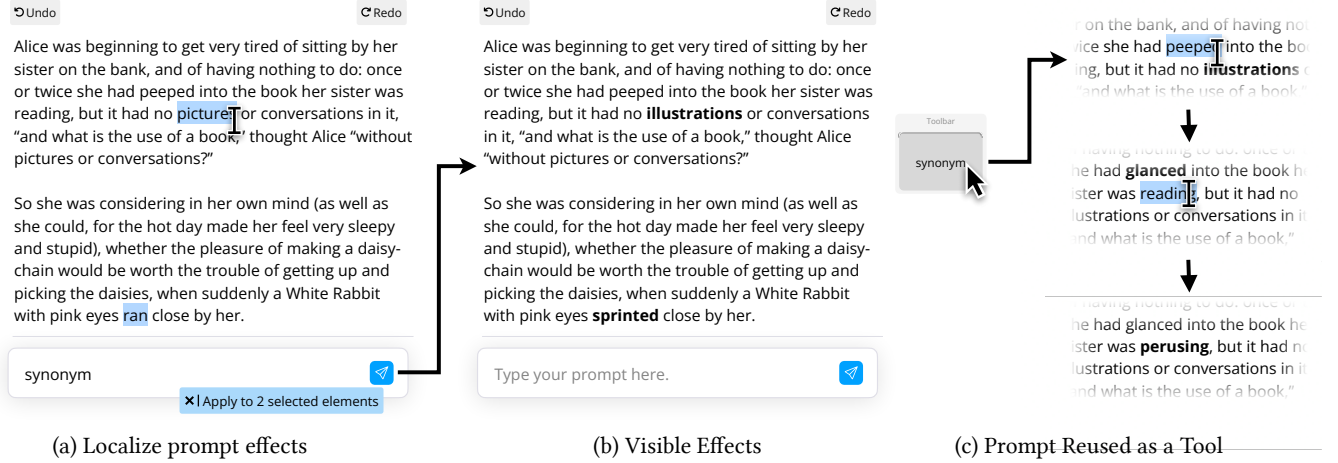
Figure 2: DirectGPT used to replace words with synonyms: (a) selecting an object such as a word before writing a prompt forces the prompt to apply to only this object, localizing its effect; (b) once executed, the modifications are highlighted and the "synonym" ad hoc tool is created; (c) the "synonym" tool is used to quickly find synonyms for other words.

pronouns and mid-air pointing gestures specify objects and locations. Of course, due to limitations of 1980s technology, the system supported only a small set of commands and sentence structures. But conceptually, the approach allows manipulating images without having to learn new software. In fact, Wizard of Oz experiments later showed participants preferred that modality [28].

More recently, there has been substantial progress in robust ways to process natural language. For example, PixelTone [35] relies on algorithms to support a broader set of sentence structures. While users were still constrained to the phrase templates and vocabulary understood by the system, an experiment showed the speech and gesture interface was preferred to accomplish photo editing tasks. To handle more complex natural language queries, solutions have been proposed to map natural language to user interface commands, such as constructing a graph [23] and building vector representations [1]. Alternatively, for the task of programming agents, PUMICE [39] proposes an iterative refinement of the initial natural language query with possibilities to demonstrate actions by direct manipulation whenever a request is misunderstood.

Arguably the closest approach to our work is Stylette [33], a system to customize websites using speech and selection of elements. It uses a pipeline of deep learning models to support better understanding of natural language queries and suggest CSS properties to modify. Similarly, our prototype system is built on top of an LLM. However, our work is not targeted at a specific task, instead it explores how the principles of direct manipulation can be extended to LLMs to support tasks such as text, code, and image editing. The design we propose is specifically adapted to address limitations of LLMs, such as easing verification, better control, and facilitating trial-and-error. We also cover all aspects of direct manipulation, not just selection mechanisms, and study their benefits compared to relying solely on natural language. In doing so, we introduce mechanisms, such as prompt reuse by turning previous prompts into templated snippets that populate a toolbar of commands.

3 DIRECTGPT: AN EXEMPLAR DIRECT INTERFACE FOR LLMS

This section describes DirectGPT, a direct manipulation interface to interact with LLMs. To demonstrate the utility of DirectGPT, we first illustrate its functionalities in a use case scenario. We then follow it with a detailed description of the proposed direct manipulation mechanisms and their implementation.

3.1 Example Use Case

Sam is working on a story and would like to revise it before sharing it with friends. Sam starts DirectGPT and pastes the current version of the story. Sam decides that a character, the "White Rabbit", needs a vivid description when introduced. Sam selects "a White Rabbit" with the mouse cursor, types "add description of its tail" in the prompt field, and presses enter. The selected text pulses and then is replaced by "a White Rabbit with a fluffy, cotton-like tail". Next, Sam notices that the words "pictures" and "ran" are used repetitively. Sam selects the first instance of both words (by pressing ctrl while selecting the two words with the cursor) and then types "synonym" (fig. 2a). Again, the two words pulse before being replaced by "illustrations" and "sprinted" (fig. 2b). Later, Sam realizes other words should be substituted. Sam clicks on the "synonym" button in the toolbar that appeared after running the previous prompt. With the cursor now indicating "synonym", Sam selects a word and it pulses and then is replaced by a synonym (fig. 2c). Sam continues by selecting other words to edit using this ad hoc "synonym tool".

Later, Sam decides to illustrate the story with a picture. In DirectGPT, Sam types "draw a flower with 6 petals". An image with six circles is generated (five black petals in a circle, and a white circle in the middle). This is a good start, but it lacks a stem and leaves. Sam types "draw a black line from here to there". Then, Sam clicks the bottom petal, drags it, and drop it on the word "here" in the prompt. Immediately, the word "here" is replaced by a picture of the petal. Next, Sam drags a pixel located near the bottom of the image and drops it on "there" which is replaced by the corresponding pixel coordinates (fig. 3a). After pressing enter, the bottom pedal and

| (a) Refer to objects | (b) Execute & localize next prompt | (c) Refer to an object | (d) Execute |
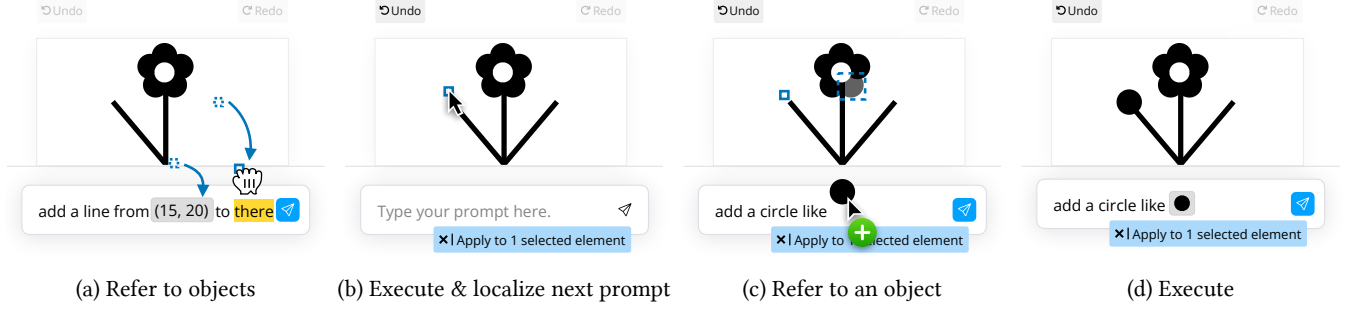
Figure 3: DirectGPT used to finish drawing a flower: (a) draw a line by referring to specific locations through drag-and-drop; (b) the line is drawn, click to specify where to add the circle; (c) refer to another circle to copy its size; (d) the circle is added.

pixel location start pulsing until a line is drawn. Next, Sam presses the "draw a black line from ? to ?" button that just appeared in the toolbar. With this ad hoc tool active, Sam clicks on two pixel locations, and a line is drawn. They repeat the operation to create a symmetric branch on the other side of the flower. Finally, Sam clicks the top of the first line to localize the edit, then types "add a circle like this", and drags and drops one of the petals onto "this" (fig. 3c). The operation is repeated on the other side to obtain the final result (fig. 3d).

3.2 Direct Manipulation Principles for LLMs

The direct manipulation of an element relies on commands (e.g., buttons in ribbons and toolbars) applied to objects of interest (e.g., words of a document in a word processor, elements on a canvas in a drawing software). This type of operation is commonly known as a *verb* and *noun* construction [62, ch. 3-3]. While the order and interactions to specify the verb and noun might differ, it is typical that clicking a command specifies the verb and pointing at the objects specifies the noun [80]. A parallel can be drawn with conversational interfaces like ChatGPT: the object of interest is the last generation from the model and the commands are generated with the prompt field. However, there is no equivalent to the noun-verb construction. Instead, both noun and verb are specified in a singular prompt by interacting solely with the prompt field.

Below, we review the principles of direct manipulation and show how to leverage them for LLMs. Each principle is demonstrated using DirectGPT, an exemplar direct interface for LLMs. Direct-GPT is designed to preserve the standard prompting capabilities of ChatGPT while introducing a layer of complementary interaction mechanisms that are expressive and consistent across application domains such as editing text, code, or images.

3.2.1 Continuous Representation of the Last Output. At its core, direct manipulation is possible only if the objects of interest are continuously represented and accessible at all times [5]. This might be the biggest difference with conversational interfaces because it requires moving away from long textual explanations to instead display an object that remains at the same position. As a result, the interface does less telling and more showing: instead of the model explaining the changes, they should be noticeable through spatial feedback and temporal changes, making it possible to compare the previous state with the new state [30].

Another crucial point is that objects should be represented in a final form matching the user's intent [67]. For example, if users wish to wrangle data, then the representation should show transformed data instead of code to transform it. Similarly, if users wish to modify a vector image, then the system should render this image.
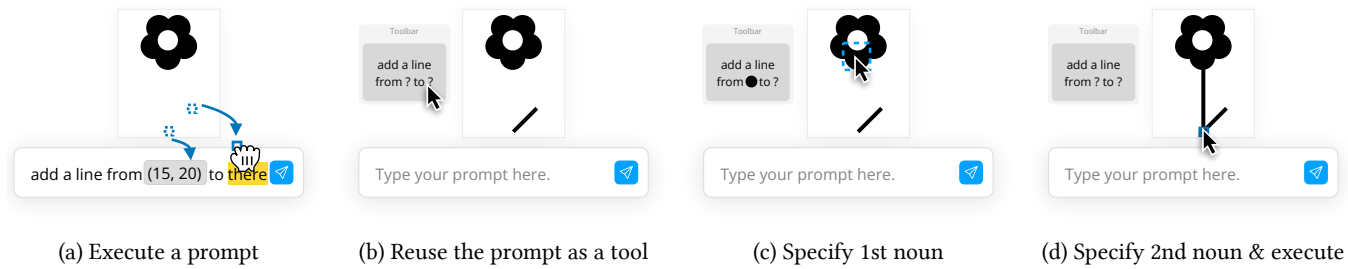
DirectGPT shows the object of interest such as code, text, and rendered images at a fixed position on the screen (fig. 3). Every new operation generating a response from the model updates this representation to help users notice the differences (fig. 2b and fig. 3b).

3.2.2 Physical Actions Through Prompt-Object Interactions. Direct interfaces use physical actions and simple metaphors to convey intent without words [72]. In our case, the objects of interest and the prompt field provides two interaction contexts with possibilities for physical actions that combine these contexts. In DirectGPT, prompts can be entered by typing in the prompt field to preserve the capabilities of prompt-based interfaces like ChatGPT. Additionally, we describe below how interactions with objects can formulate and constrain prompts through physical actions. Essentially, these interactions recreate the verb-noun constructions typical of direct manipulation interfaces.

Prompt to Object: Localizing Effects – Interactions between the prompt field and objects enables specifying objects without having to refer to them in the prompt. This helps users express intents that involve an object and apply only to that object. As such, the prompt needs only to describe the verb whereas the physical action unambiguously specifies the noun. This interaction also describes a constraint in that only the specified objects should be modified while the rest of the content should remain unchanged. For example, colouring object A requires selecting A and prompting "red" or "colour red".

In DirectGPT, objects are selected with a click. When one or multiple objects are selected, the prompt field indicates "Apply to selected elements" (fig. 2a). Executing a prompt while objects are selected forces the prompt to apply only to these objects. Otherwise, the prompt applies to the whole content, like ChatGPT. This reuse of the prompt field allows users to seamlessly switch between local and global prompts, even after having started typing a prompt. To maintain compatibility with different domains, the interaction supports both noun-verb (frequent in text editing tasks) and verb-noun constructions (frequent in image editing tasks). This corresponds

| (a) Execute a prompt | (b) Reuse the prompt as a tool | (c) Specify 1st noun | (d) Specify 2nd noun & execute |

Figure 4: Prompts can be reused as tools: (a) a prompt with two nouns is executed; (b) the prompt is abstracted into an ad hoc tool; (c) using the tool, a click sets the first noun; (d) a second click sets the second noun and executes the reused prompt.

to the difference between selecting and then typing the prompt, or typing a prompt and then selecting.

Object to Prompt: Referring to Objects – Interactions between objects and the prompt field allow unambiguous references to objects. This helps users express intents that involve relationships between objects and that may modify the whole content. Unlike regular prompt-based interfaces, target objects do not need to be described with words, but are specified instead through a physical action. This is similar to previous work showing an increase in expressivity and precision when using mid-air pointing gestures to refer to graphical shapes [8], and incorporating images into code to refer to UI components [83].

In DirectGPT, we adopt a drag-and-drop approach to select one or multiple objects and drop them in the prompt. Objects can be dropped between words (fig. 3c), or onto words (fig. 3a) to clarify intention. This supports both back-and-forth interaction (e.g., type "move", drop an object to form next word, type "to", drop a location to form another word) or prompt-then-bind interaction (e.g., type "put that there", then drop an object onto "that", and a location into "there"). Both interactions have the same effect but one gives users the option to type the whole prompt without being interrupted. The objects dropped in the prompt are then treated as "object-words": their background becomes grey and they can be copied or deleted like single words. Furthermore, to reduce articulatory distance, DirectGPT uses a thumbnail of the object, or a brief description to render the object-word within the prompt (fig. 3a). Hovering over an object-word highlights the corresponding object in the generated output.

Object to Object: Fine-grained Manipulations – Interactions on objects support fine-grained manipulation. This helps users express transformations that involve an object and apply only to this object. Typically, these interactions are used for common domain-specific edits, such as resizing and moving a shape, indenting a piece of code, or removing some text. It would be possible to implement such interactions using a task-specific approach like a conventional graphical user interface. However, that would require information about the task domain, limiting a primary advantage of using a general purpose LLM. To remain generalizable, DirectGPT does not implement such custom interactions and relies only on general direct manipulation techniques.

3.2.3 Rapid Operations Through Prompts Reused as Tools. Direct interfaces leverage labelled buttons to execute commands [67]. It is difficult to predict what actions users may need, but a direct

interface for LLMs can assist by creating an interface adapted to emergent tasks by reusing previous prompts. Conveniently, composing prompts through physical actions makes explicit the verb and noun of an operations. This means our system can reliably determine which parts of the prompt describe the objects being manipulated and which parts are concerned only with the action. This allows abstracting prompts into universal commands, and an executed prompt can be reused immediately by preserving the verb, but swapping the noun.

In DirectGPT, as soon as a prompt finishes executing, it is added as a button in a toolbar. This prompt is now a *tool*. Pressing it enters a mode where further clicks on objects apply the prompt to these objects as well. For example, when editing text, selecting a word and prompting "synonym" replaces the word with a synonym and adds a button labelled "synonym" to the toolbar (fig. 2b). Then, this prompt can be reused as a tool. First the user clicks on the tool, then selects one or multiple words. As soon as the selection is complete, the prompt "synonym" is applied to them, behaving like tools in conventional direct interfaces (fig. 2c).

This technique scales to prompts involving multiple nouns. For example, when editing an image, the user prompts "add a line from here to there" and then replaces "here" by dragging the first point, and "there" by dragging the second point (fig. 4a). After drawing the line, the prompt is added in the toolbar as "add a line from ? to ?" (fig. 4b). Clicking it selects the tool, so the next click on the canvas modifies the tool to show that point or object, such as "add a line from (50, 50) to ?". Then, a second click executes the prompt for the two specified locations (fig. 4c-d).

Tools also support noun-verb constructions without a mode. The objects of interest can be selected before the tool is clicked. In this situation, the tool is only used once and the mode is restored immediately after executing. For example, a user can select two locations (by maintaining the ctrl key) and then click "draw a line from ? to ?". The line is drawn and the tool is no longer selected.

3.2.4 Immediate Targeted Feedback. Direct interfaces provide instant feedback so that *"users can immediately see if their actions are furthering their goals, and if not, they can simply change the direction of their activity"* [67]. While LLMs will surely execute faster in the future, for now they may take seconds. A direct interface can provide useful loading feedback so users can identify possible mistakes. This is possible because direct manipulation disentangles nouns from verbs, so the interface is immediately aware of the objects that are to be edited. This information can be used for targeted feedback about the specific objects that are being acted upon.

In DirectGPT, we use this information to show a loading animation as a pulse (a looping fade-in-fade-out) on objects selected before executing a prompt, or mentioned in the prompt. While the feedback does not necessarily match the objects that will be modified (e.g., when referring to object as exception such as "turn all circles blue except [this]"), it should match the user's mental model of their natural language query. Thus, this visual feedback lets the user verify the correct objects were selected and draws their attention to changes once they happen. If the wrong objects are highlighted, the user can click a button to stop the generation.

3.2.5 Reversible Operations Through Undo Mechanisms. Direct interfaces make operations easily reversible to encourage exploration and reduce user anxiety [68]. The advantage is that the users' conceptual model of an operation might be clearer than in a conversational interface because it is similar to how typical software behave.

In DirectGPT, we implement a typical undo and redo mechanism. Clicking the "Undo" button, or using the ctrl+z shortcut reverts the generated output to the state before the last command. Clicking the "Redo" button or using the shortcut restores the modification. To align with the user's mental model, reverting matches the granularity of operations as performed by the user. For example, if the user selects five elements and applies a prompt to them all at once, then this whole interaction is considered as one revertible operation.

3.3 Implementation

DirectGPT works by first entering a prompt to generate an object of interest. Then, the object of interest is continuously represented and elements composing it can be manipulated as described in section 3. Currently, DirectGPT supports all text-based representations such as text and code. It also supports SVG images by rendering them for direct selection of elements in the vector image.

DirectGPT is implemented in TypeScript using ReactJS [63] for the interface, Prism.js [61] for code syntax highlighting, and the official OpenAI API library [58] for executing prompts using "gpt-3.5-turbo". All source code and a live demo are available[1].

This section describes the strategies we used to convert the direct manipulation actions described section 3.2.2 into prompts.

3.3.1 Localizing the Effect of a Prompt. The constraint of localizing the effect of a prompt is done by the interface. Instead of asking the LLM to rewrite the entire passage, the LLM is only asked to rewrite the part that is selected. Then, this part is replaced by the response from the LLM. This ensures a deterministic outcome. Note that the whole passage is still provided to the model with the selected part replaced by "<blank>". This ensures the LLM is aware of the context surrounding the selection. An example of this kind of engineered prompt is provided in appendix A.1.1.

3.3.2 Referring to Objects in Prompts. When a prompt contains objects that were dragged and dropped into the prompt field, our system converts the prompt into one of two kinds of engineered prompts we empirically found to give the best results for text (including code, etc.) and vector images.

For text, simply copying the text that is being referred is not enough because the instruction would be ambiguous when the text

appears multiple times. Instead, we use the delimiter strategy by copying the entire passage with added delimiters around the text that was selected by the user [57]. The delimiters all have a unique identifier, and this identifier is used in the engineered prompt in lieu of the object-words for text specified through direct manipulation. appendix A.1.2 provides an example and more details.

For elements in a vector image, the delimiter strategy is not ideal because the code corresponding to an element is not always located within a contiguous part of the SVG specification. Instead, our system automatically adds unique ids to each SVG element, and the object-words representing shapes specified by direct manipulation are replaced by their corresponding id. appendix A.1.3 provides an example and more details.

4 USER STUDY: DIRECTGPT VS CHATGPT

We conducted a user study to measure the effect of direct manipulation principles to convey intent, control LLMs, recover from mistakes, and reuse prompts. Participants edited literary text, JavaScript code, and vector graphics images using both DirectGPT and a replica of ChatGPT. This replica of ChatGPT served as a baseline because it is one of the most popular interfaces to interact with LLMs (with estimations of more than 100 million ChatGPT users in 2023 [50]) and it is strictly conversational: a conversation is initiated with a prompt, answers are shown word-by-word, and the entire conversation is kept at all times and fed back into the LLM.

For the purpose of the study and to ensure accurate logging, the ChatGPT interface was reimplemented. Care was taken to ensure a faithful reimplementation: the prompting and conversation interface were made to be identical, the conversation panel rendered Markdown and supported code syntax highlighting, the system prompt was identical to the one used in ChatGPT, and the answers were streamed to display word-by-word. We confirmed faithfulness by running all our experimental conditions using the official ChatGPT and comparing the responses. Results were sensibly identical[2]. The interface only differed in that it was not possible to start a new conversation or have multiple conversations in parallel, and our replica rendered SVGs mentioned in the conversation to avoid the need for an external SVG viewer. This also meant that when an SVG was rendered, its code could not be seen by the participant. This was done to make tasks involving images feasible and ensured participants manipulated images rather than code (which is tested separately). For the sake of clarity, we refer to this interface as ChatGPT and use OpenAI's ChatGPT otherwise.

The DirectGPT condition used the same interface as the ChatGPT replica but with the direct manipulation principles described in section 3. Both conditions used the same model "gpt-3.5-turbo" accessed through the OpenAI Chat API [59].

4.1 Participants and Apparatus

We recruited 12 participants from our institution (20 to 34 age range, M=26.8, 6 self-identified as female and 6 as male). Potential participants were asked if they had prior experience with programming and OpenAI's ChatGPT, they had to answer "yes" to both questions to participate. On a 5-point scale from 1-"never" to 5-"often" they

---

[1] Live demo and code: http://ns.inria.fr/loki/DirectGPT

[2] LLMs are non-deterministic even at low temperatures, and some variations for identical prompts can be expected

| (a) Starting Image | (b) Transform Task | (c) Add Task | (d) Remove Task | (e) Global Transform Task |
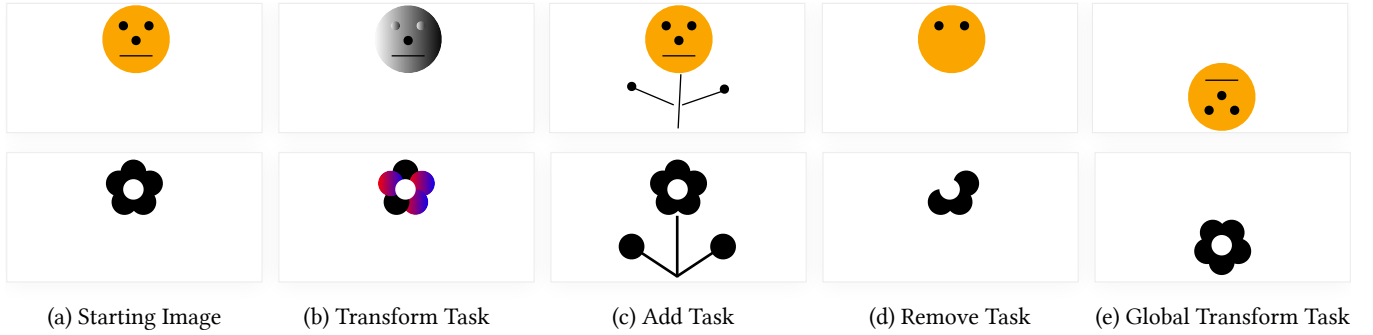
Figure 5: Given the (a) starting image, the participant completed four tasks in the image ACTIVITY with ChatGPT and DirectGPT from either the top or bottom row: (b) colourize using gradients; (c) add elements; (d) remove elements; (e) flip upside down.

reported their frequency of use of ChatGPT as Mdn=3 and having used AIs in the past to do text editing (N=6), code editing (N=8), and image generation or editing (N=8).

Participants took part in the study remotely. They shared their screen with the experimenter who took notes during the session. The participants' screen, the audio, and their interactions with the tool were recorded. Sessions were an hour long. In appreciation for their time, participants received the equivalent of $11 USD.

4.2 Procedure

The participant used two INTERFACE (ChatGPT, DirectGPT). For each INTERFACE, they completed three ACTIVITY (text, code, image). For each ACTIVITY, they performed four tasks.

Introduction (∼2min) – After completing the consent form and the demographics questionnaire, the participant was told "You will interact with an AI to edit text, code, and images".

Video Tutorial (2 INTERFACE × 3 ACTIVITY × 30sec) – Before each ACTIVITY (including those done with the ChatGPT interface), the participant watched a video tutorial with a voice-over to describe the interactions possible with the interface they were about to use. The video tutorials were designed to avoid giving away strategies or biasing participants in their wording of prompts. This was done by showing toy examples (e.g., Lorem ipsum text) and demonstrating interactions with descriptive prompts (e.g., "A prompt that modifies only the selected text"). These video tutorials are provided as supplementary material.

Editing Tasks (2 INTERFACE × 3 ACTIVITY × ∼8min) – After watching the video tutorial, the participant completed the four tasks of the ACTIVITY. The participant proceeded through the tasks one after the other. The current task to complete was displayed at all times on the left of the interface. It consisted of a short textual instruction (e.g., "Reproduce" or "Text in yellow => synonyms") that could not be selected, and an image of the content to edit with some text or code in yellow, or the image the participant had to reproduce. Because all tasks required editing existing content, this content was already added to the conversation as the first message to emulate an already first generation and make sure the task was identical across participants. This content was reset between tasks. After completing a task, either because the participant finished, abandoned, or reached a time limit of three minutes, the participant

responded to a 5-point semantic differential scale "How close are you to the target" from "distant" to "close".

Questionnaire on Interface Used (2 INTERFACE × ∼2min) – After completing all three ACTIVITY for an INTERFACE, the participant completed a questionnaire including a System Usability Scale (SUS) [10] and 5-point scale statements to rate. The questionnaire was identical after both INTERFACE.

Semi-structured Interview (∼10min) – At the end of the study, the participant was invited to comment on any aspect of the study. Then, the experimenter prompted the participant on their preferred interface, the most difficult tasks, prior experiences with AI where they faced similar difficulties, their strategies to write prompts, and other behaviours that the experimenter noticed during the session.

4.3 Design and Tasks

To test the flexibility of our approach, participants edited content through three activities: literary text, JavaScript code, and vector images. The texts were the first three paragraphs from *Alice's Adventures in Wonderland* by Lewis Carroll (253 words) and the first two paragraphs from *Frankenstein; or, The Modern Prometheus* by Mary Shelley (265 words). The codes were a function to print a pyramid in the console (13 lines, a for loop containing two nested for loops) and a function to count the number of values below the mean of a moving window (15 lines, a for loop containing two nested for loops). The images were a flower (5 black circles for the petals and 1 white circle in the centre, fig. 5) and a smiley face (1 yellow circle for the face, 3 black circles for the eyes and nose, and a black line for the mouth, fig. 5)[3].

For each content to edit, we designed four tasks that spanned different level of difficulties in using the direct manipulation features. Three tasks required localized edits (either modify specific elements, add/expand elements, or remove/reduce elements), and the fourth task required a global modification. We also ensured the tasks where within the boundaries of what the LLM could accomplish if prompted correctly. This was done to not confound our results with the limitations of the underlying model in accomplishing complex tasks. For the text activity, the tasks were: replacing 5 words by synonyms; adding more descriptions to two passages; summarizing two passages; and using the future tense throughout the text excerpt. For the code activity, the tasks were: renaming a

---

[3]Tasks and activities can be tested in the live demo: http://ns.inria.fr/loki/DirectGPT
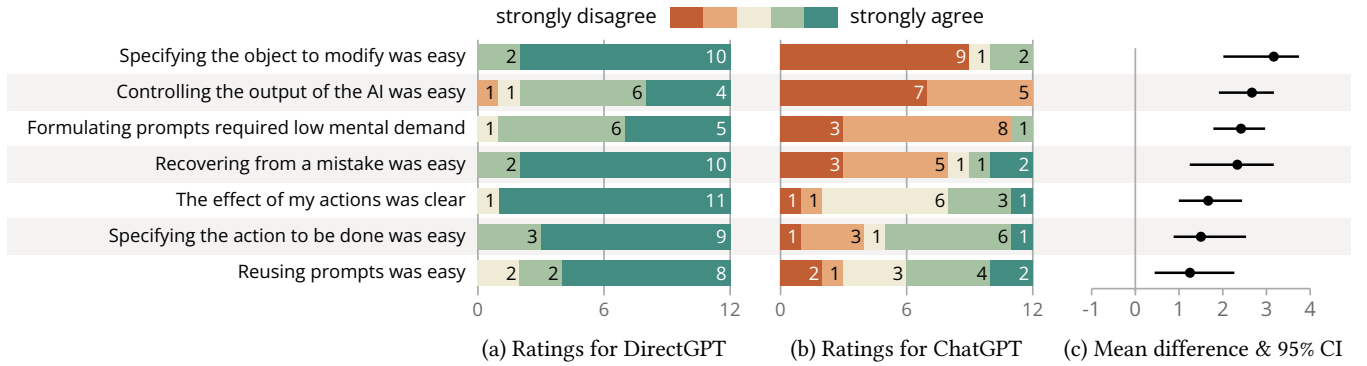
Figure 6: Participants' response when rating the 5-point statements for (a) DirectGPT and (b) ChatGPT. (c) Dots are the mean differences of DirectGPT compared to ChatGPT. Bars are the 95% CIs calculated with the studentized bootstrap method.

variable; converting two for loops into while loops; factorizing a loop by using the reduce or repeat function; and converting the function to Python. For the image activity, the tasks were: colouring three elements using a gradient; adding three lines and two circles (either to add a torso and hands to the smiley face, or to add a stem and leaves to the flower), removing three elements; or turning the image upside down (fig. 5).

The study followed a within-subject design. INTERFACE (Direct-GPT or ChatGPT) and ACTIVITY (text, code, or image) appeared an equal number of times in each position. This was done using a Graeco-Latin square to order ACTIVITY and the version A or B of the content to edit. Essentially, this means each participant had a unique ordering, half the participants started with ChatGPT, and the version A or B of the content to edit appeared equally as often with the ChatGPT interface than with the DirectGPT interface.

We gathered interaction logs, subjective ratings, and task measures for 2 INTERFACE × 3 ACTIVITY × 4 task × 12 PARTICIPANT = 288 tasks.

4.4 Data Analysis

Considering ongoing debates related to statistics in HCI [21, 26], we report both p-values and 95% confidence intervals on mean differences. Because we are interested by the size of the effects rather than their mere existence, our interpretation is based on CIs that give information about the size of the effects and the uncertainty around them. All 95% confidence intervals are calculated using the studentized bootstrapping method because it has been shown to be the most robust across distributions and for study designs similar to ours [47, 87]. All p-values are calculated using a Wilcoxon signed-rank test because it is more robust and less likely to yield false positives compared to parametric alternatives when dealing with data from unknown or possibly heavily skewed distributions [9]. Our analysis is done in Python using scipy 1.10.1 [77] and arch 5.5.0 [66] for the bootstrapped CIs. The Wilcoxon signed-rank tests discarded ties. The studentized bootstraps used 10,000 replications.

4.5 Results

Participants went through the tasks without abandoning. Averaged over all tasks and activities, participants consistently had better results with DirectGPT on objective metrics (time, number of prompts, number of words per prompt) and equal or better results on subjective metrics (closeness to target, usability, and 5-point scale statements). All results are detailed below and visualized in fig. 6 and fig. 7.

4.5.1 Closeness to target. On a 5-point semantic differential scale, participants rated their modifications closer to the target when using DirectGPT by .95 (95% CI: [0.70, 1.25], M=4.84 vs M=3.89, p<0.001). Per ACTIVITY, the mean difference for text was 1.58 (95% CI: [1.10, 1.99], p<0.001) and for vector images it was 1.17 (95% CI: [0.71, 1.68], p=0.001). However, the mean difference for code was only 0.10 and likely insignificant (95% CI: [-0.13, 0.66], p=0.495).

4.5.2 Time. The mean completion time of a trial was shorter with DirectGPT by 1min 1s (95% CI: [49s, 1min 11s], M=56s vs M=1min 57s, p<0.001). Even when excluding the time it took for the prompt to be executed and the output fully generated, it remains that DirectGPT was faster by 46s (95% CI: [38s, 55s], M=47s vs M=1min 33s, p<0.001). Per ACTIVITY, the mean difference for text was 1min 18s (95% CI: [1min 5s, 1min 32s], p<0.001), for vector images it was 34s (95% CI: [20s, 45s], p=0.001), and for code it was 27s (95% CI: [3s, 39s], p=0.007). Note that these include times from tasks where participants had to stop because they reached the 3min time limit. This happened in 8% of the trials with DirectGPT, and 35% with ChatGPT.

4.5.3 Prompt Quantity. The mean number of prompts was lower with DirectGPT by 1.77 (95% CI: [1.37, 2.46], M=1.90 vs M=3.67, p<0.001). Per ACTIVITY, the difference for text was of 2.33 (95% CI: [1.61, 3.40], p<0.001), for images it was of 1.88 (95% CI: [1.06, 3.26], p=0.002), and for code it was of 1.10 (95% CI: [0.56, 1.44], p=0.001).

4.5.4 Prompt Verbosity. Prompts in DirectGPT were less verbose on average by 15.3 words (95% CI: [9.35 28.72], M=5.83 vs M=19.98, p<0.001). Per ACTIVITY, the difference for text was of 37.97 (95% CI: [21.59, 78.37], p<0.001), for code it was of 10.71 (95% CI: [6.13, 31.83], p<0.001), and for images it was of 3.21 (95% CI: [1.77, 4.89], p=0.002).

4.5.5 Usability. On the System Usability Scale, and for the specific given tasks, DirectGPT was rated 92 (typically considered "Excellent" [4]) whereas ChatGPT received a 53 (typically considered "OK"). This corresponds to a mean difference of 39 (95% CI: [27.3, 47.34], p<0.001).
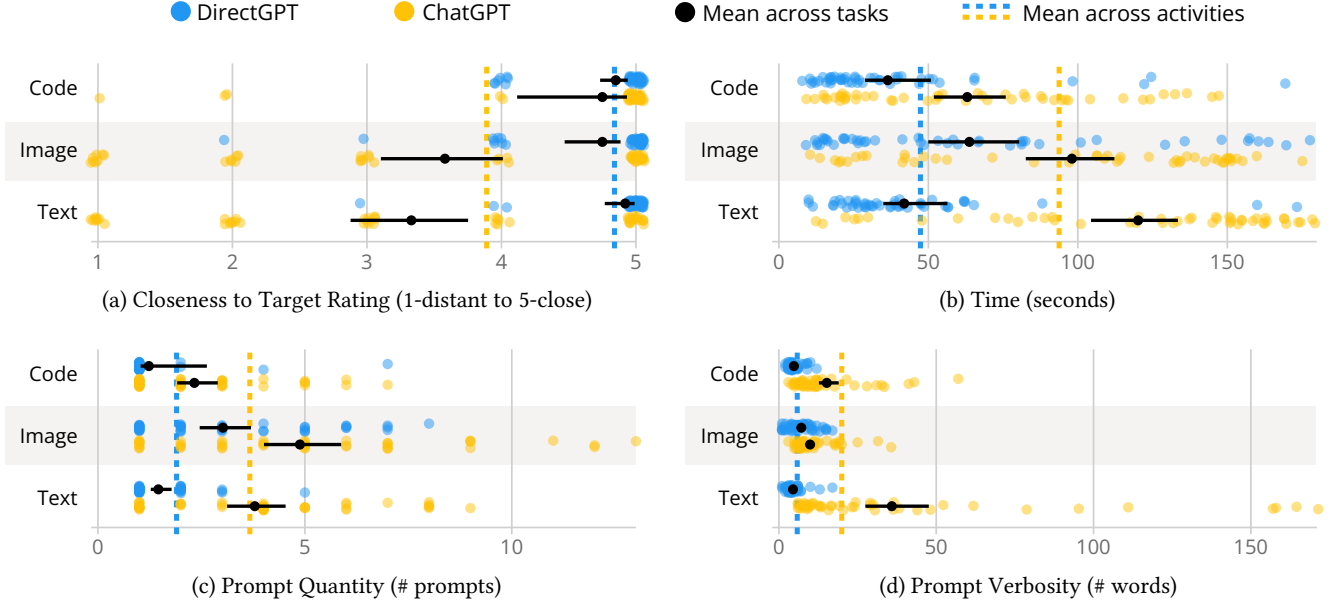
Figure 7: Result breakdown per INTERFACE, ACTIVITY, and task for (a) closeness rating; (b) time; (c) number of prompts; (d) length of prompts. Black dots are means, bars are 95% CIs, vertical dashed lines are means over all activities.

4.5.6 Questionnaires. Participants rated all seven 5-point statements higher for DirectGPT. Ranked from largest to smallest effects: "Specifying the object to modify was easy" by 3.17 (95% CI: [2.01, 3.74], p<0.001), "Controlling the output of the AI was easy" by 2.67 (95% CI: [1.91, 3.17], p<0.001), "Formulating prompts required low mental demand" by 2.42 (95% CI: [1.79, 2.97], p<0.001), "Recovering from a mistake was easy" by 2.33 (95% CI: [1.25, 3.17], p=0.005), "The effect of my actions was clear" by 1.67 (95% CI: [0.99, 2.44], p=0.004), "Specifying the action to be done was easy" by 1.5 (95% CI: [0.88, 2.53], p=0.004), "Reusing prompts was easy" by 1.25 (95% CI: [0.44, 2.27], p=0.016).

4.5.7 Interactions Used. With DirectGPT, participants relied on direct manipulation mechanisms to craft 84% of their prompts. Specifically, 68% of all prompts were localized, 14% contained at least one reference to an object dropped through direct manipulation, and 20% were the result of using a tool generated from previous prompts (i.e., reusing a previous prompt).

5 DISCUSSION

After discussing our results, we open the discussion to themes stemming from participant comments and experimenter observations.

DirectGPT helps convey intent unambiguously. This was reflected in participants being 50% faster to accomplish tasks with 50% fewer and 72% shorter prompts. Subjectively, participants unanimously agreed that "Specifying the object to modify was easy". In fact, compared to ChatGPT, this is the statement with the largest positive mean difference.

DirectGPT helps control and notice the effect of prompts. This increase in control is reflected in participants achieving results they judged closer to the target and rating that controlling the output of the AI

was easier with DirectGPT than ChatGPT. On average, participants also found the effect of their actions to be clearer with DirectGPT.

DirectGPT helps error recovery and might help reusing prompts. Participants generally found that recovering from a mistake was easier with DirectGPT, due to the undo feature. They used the toolbar for 20% of all prompts and generally rated reusing prompts to be easier with DirectGPT. However, this was also the smallest difference since half of the participants also found reusing prompts to be easy with ChatGPT.

DirectGPT preserves prompting-only capabilities and people sometimes prefer them. All participants completed global tasks without using direct selections. This confirms that DirectGPT preserves prompting possibilities of ChatGPT and that people may switch between prompting and direct manipulation depending on the task and perceived benefit.

5.1 Observations and Participants' Comments

Prompts were more concise (and less polite) with DirectGPT. Corroborating existing findings [55, 86], eight of our participants wrote prompts in a human-to-human conversation style when using ChatGPT. For example, they asked politely (P4, P5, P6, P7); thanked the model (P5, P4); encouraged it such as *"yes"* (P0), *"Great. Now also change [...]"* (P7), *"Awesome. You forgot the hands though. [...]"* (P5); used caps lock, smileys, or exclamation marks to emphasize some points (P4, P5, P7); asked questions to make the model notice the problem (P0, P2, P4, P6); and generally used more verbose language such as *"actually I was expecting [...]"* (P3), and *"similarly, [...]"* (P8).

These participants did not use that kind of language when using DirectGPT despite doing similar tasks and using the same underlying model. During the semi-structured interview, they explained that DirectGPT understood them faster and did not feel like a chat.

> *"[with DirectGPT] I knew that by highlighting the specific parts I can just, like, put in very concise prompts and it will know exactly what to do... so it did not feel necessary for me to type in a whole conversation"* — P7

> *"[with DirectGPT] it was more like a tool. Like I did not need to make it comprehend anything. I just wanted to tell it what to do [..] it did not feel like an assistant with whom I have to converse"* — P5

With ChatGPT, participants carefully checked the text generation because they doubted its accuracy. Besides rating that the effect of their actions was less clear with ChatGPT, participants also felt it was difficult to verify its responses. Specifically, participants commented about the difficulties to check that nothing else had been modified when ChatGPT responded with the modified text.

> *"[with ChatGPT] verifying that it actually only did the thing that you wanted it to do, it's a bunch of text... basically impossible"* — P6

> *"my reading speed is not that fast so... I couldn't really tell.. is it like correctly modified at the location I want to modify, and I am not very confident that it will generate the exact same thing from my experience"* — P4

Code was easier to refer to than text and vector images. While the difference in performance was large for text and images, participants performed similarly for tasks on code with both ChatGPT and DirectGPT, albeit they still required more time, more prompts, and more words with ChatGPT. When asked about the tasks they found most difficult with ChatGPT, most participants mentioned either text (N=8) or images (N=4). Some explained this choice due to how easy it is to be clear and specific with code, but how difficult it is with images and text due to the lack of structure.

> *"The code is simpler to navigate. It makes it simpler to determine where it should go and tell the model where it should go. But if it's a big text it's like, ugh, talking about it in paragraphs and sentences and occurrences of words... horrible."* — P6

> *"in the code task... I think I talked about the first inner for loop. So you could specify based on ordering."* — P11

> *"I'd say the picture task was more challenging [...] I feel like with the code, I have enough coding knowledge that I can specify almost the exact location to do something"* — P1

Participants knew and used a range of common and effective prompt engineering strategies. Perhaps due to their prior experience with ChatGPT, we observed some of the prompt engineering strategies suggested in the OpenAI wiki [57], such as splitting tasks into simpler subtasks (N=12), providing examples (P5), and specifying the desired output length (P9). Another effective strategy adopted by four participants was to agree on a common terminology to refer to elements prior to the task. For example, in the image activity, P10 and P11 asked the model to give a label to the elements in the image and then referred to those labels; P8 gave an object an identifier and asked the model to do the same for other elements; and P4 asked for a description of the image and then reused the wording.

Participants were excited about DirectGPT and how it could help them. Overall, all participants mentioned a preference for DirectGPT. When asked about features they found most useful, localizing the effect was mentioned most frequently (P0, P1, P3, P4, P5, P6, P7, P9, P10), followed by the toolbar of reusable prompts (P2, P10, P11), dropping objects in the prompt (P2, P8), and undoing (P8). Some explained that they had previous bad experiences with OpenAI's ChatGPT and Github Copilot where selection features like in DirectGPT could have helped.

> *"it was horrible to use [OpenAI's] ChatGPT for this kind of situation because you just had no control over the output. [...] Especially what is important is that the area of effect is just small, like I don't want it to change anything but that."* — P5

> *"in using Copilot I really would like to select"* — P3

> *"I use [OpenAI's] ChatGPT primarily to look over text I've written [...] often times it ends up changing the entire text which isn't really what I want, so, I think for this task it would be really nice to have the first system [DirectGPT]"* — P8

> *"There are tools that start to integrate stuff but they are very limiting or expensive or very specific and this one seems like it is generic in a way that it's like basically [OpenAI's] ChatGPT but you can target it better"* — P6

Other participants were surprised because they thought ChatGPT was already pretty good.

> *"I was actually surprised by the second system [DirectGPT] cause I thought like, the first system [ChatGPT] was already pretty well integrated... But I guess the second system [DirectGPT] did surprise me and when I was using it I was like oh wow, this will actually make my experience with LLMs so much better"* — P7

5.2 Limitation

The results of the study might be LLM dependent. While both interfaces used the same underlying model, ChatGPT arguably suffered the most from model misunderstandings. For example, with the image activity of the study, some participants tried prompting the model to "remove the nose" but the model would often misunderstand and also remove the eyes of the smiley face. However, even assuming a model with perfect accuracy, previous Wizard of Oz experiments found that people still prefer a combination of direct manipulation and natural language, especially when some objects are hard to describe [28]. This result is consistent with previous work showing the benefits of combining gestures with language [8, 34] and that more expressive ways of communicating reduce misunderstandings compared to just text [16, 37].

Participants' prior experiences probably helped them perform better. All our participants had experience with LLMs. This was clear from their use of effective prompt engineering strategies, as mentioned previously. Additionally, our participants also had experience in coding and could generally be considered expert computer users. It is not clear whether less technical users would perform equally well with DirectGPT.

Direct manipulation might not be as useful for other tasks. For instance, some selections might be easier to specify in the prompt such as "all red circles" rather than selecting all circles manually. Additionally, our study focused specifically on editing tasks and the need for control over the output. However, control is not always necessary, and there are some benefits in having the AI lead the way [14, 27, 36]. Similarly, it is possible that other tasks that are more exploratory would not benefit from a direct manipulation interface because users do not have a specific target in mind. In this case, DirectGPT remains usable the way one would use ChatGPT by typing prompts without any selection.

The design of DirectGPT might be misleading. For instance, the pulsing feedback matches the elements mentioned in the prompt, which, for complex queries, might differ from the ones that will be modified once the command is executed. To resolve this discrepancy, an approach could be to decompose the prompt by first asking the LLM to list the elements that will be modified and highlighting those. Additionally, because DirectGPT shows only the final output, the modification is not explained. While this was not an issue with the tasks of our study, some applications may require presenting the explanation to the user, either on the side or on-demand, to help with transparency.

5.3 Future Work

Testing other benefits of direct manipulation such as learnability and explorability. Shneiderman cites other benefits of direct manipulation interfaces such as enabling a multi-layered approach to learning [72]. As such, *"novices can learn basic functionality quickly"* and *"experts can work extremely rapidly to carry out a wide range of tasks, even defining new functions and features"* [67]. Future work could investigate if such learning occurs with DirectGPT. From our observations, we expect that localizing prompts and referring to objects would be quickly learned but that reusing prompts would be more of an expert feature. Additionally, DirectGPT follows the guidelines for explorable interfaces, such as making the effects of actions visible and making it safe to experiment [20, 22, 48]. Future work could investigate the impact of these features on users' exploratory behaviours.

Exploring extensions to direct manipulation such as demonstrational interfaces and instrumental interaction. Direct manipulation has been extended in many ways that could also benefit DirectGPT. For example, instrumental interaction [5, 6] defines *interaction instruments* as mediators between users and objects of interest. An application of this idea would be to *reify* [7] prompts into interaction instruments that can be applied to different objects and that are easily reused and modified. Another extension of direct manipulation that seems promising for LLMs are demonstrational interfaces [52]. Demonstrating the operation would allow not only to specify nouns but also verbs through physical actions. For example, users provide examples of the action they want to perform (e.g., they replace a word by a synonym) and the system would derive a tool from it. The examples could even be used in the prompt generated by the system [57].

Using prompts to help select objects. Object selection in DirectGPT is either done solely by direct manipulation or solely by describing

the objects in the prompt. Future work could investigate the design of an hybrid approach where specific prompts such as "all red circles" would form a selection that can then be refined through direct manipulation. One design for this feature could be to offer an additional prompt input field for the purpose of object selection. Another approach is to implement *hooks* that analyze the prompt while it is being typed and detect references to objects. For example, while typing "move red circles to the top", the part "red circles" would be detected and highlighted by the system. Then, the user could click on it to select the circles and possibly modify the selection through direct manipulation. Besides references to objects, these "hooks" could also detect other elements such as colours, fonts, styles and offer widgets to edit the prompt through direct manipulation, similar to code projections [24].

Testing alternative designs. The design of DirectGPT was chosen to best exemplify the principles of direct manipulation so that their benefit could be evaluated. While the interactions we proposed favoured simplicity and consistency, other designs could capture direct manipulation principles equally well and offer a different trade-off. For example, an alternative design could rely on contextual menus or popups with a prompt field appearing close to the selection. This would reduce travel time between selection and prompt field and could support chaining different prompt fields [81]. However, this may clutter the screen, conflict with the software preexisting interaction mechanisms, and make transitions between global and local edits difficult once the user starts typing the prompt. Similarly, instead of undo features, the edits could be marked for the user to accept or reject, like "track changes" in editing software. Finally, more complex tasks might require prompting the model with a chain-of-thought [79] which would require tuning the internal prompts used by DirectGPT and extracting only the relevant part of the answer. Ultimately, these design decisions should be decided based on the targeted tasks and application domain.

Exploring the use of DirectGPT for other tasks such as exploration, data wrangling, and GUI building. We demonstrated DirectGPT to edit text, code, and vector images but the principles proposed extend beyond these use cases. First, not all tasks require editing. Instead, two prompt fields could be supported, one to edit the content in-place, the other to ask for details. In this case, the direct manipulation would help tailor the explanation to a piece of code, a specific topic mentioned, or an element of the image. Second, even with editing, more activities could be supported as long as the objects of interest are manipulable and continuously represented in their final form. For example, one could imagine editing a website by clicking a panel and prompting "make this responsive by shrinking the content"; doing data wrangling by manipulating a data table such as selecting a column and then prompting "turn into a row"; stylizing a chart by selecting the axes and prompting "use Open Sans"; arranging a graphical user interface by prompting "align this and that horizontally" and dropping buttons instead of "this" and "that"; or selecting a row in a table from a rendered LaTeX document and prompting "use grey background".

Using DirectGPT to control other generative models. While DirectGPT is built atop an LLM, our findings point to the potential of direct manipulation to help interact with other kind of prompt-driven

generative models. For example, a music track could be directly manipulated using its audio spectrum. This would allow interactions such as prompting to "reproduce this part here using a saxophone instead". Additionally, a limitation of the implementation of Direct-GPT is that it needs to map physical actions to textual prompts. This sometimes requires workarounds such as using identifiers. In contrast, models that accept multi-modal inputs could help create more robust implementations of DirectGPT.

Tradeoffs of integrating DirectGPT into exisiting software. DirectGPT was designed for easy integration into traditional software. Often, this could be done as simply as adding a prompt field somewhere in the interface, even if the software already has selection and dragging mechanisms. For example, in a drawing software, dragging a shape within the canvas would move it, while dropping it in the prompt field would allow referring to it. If such an integration is done, future work could investigate tradeoffs between the use of prompts and more traditional software operations. For example, it might be preferable to do precise tasks through direct interactions and reserve prompting for irregular and more complex tasks. But, akin to how expert features such as software shortcuts are not necessarily learned [15], people might not always adopt the most efficient approach.

6 CONCLUSION

We characterized how the principles of direct manipulation can be leveraged to design interfaces for LLMs that help express intent, control the generation of output, and recover from mistakes. This was demonstrated with DirectGPT, an interface for LLMs that can localize the effect of prompts, refer to objects of interest in prompts, reuse commands, and provide undo operations. In a user study, participants were faster, more successful, and they preferred to use DirectGPT instead of ChatGPT to perform text, code, and image editing tasks. Beyond these use cases, this work informs how traditional software could seamlessly integrate LLMs and other generative models to support co-creation with artificial intelligence.

ACKNOWLEDGMENTS

REFERENCES

[1] Eytan Adar, Mira Dontcheva, and Gierad Laput. 2014. CommandSpace: Modeling the Relationships between Tasks, Descriptions and Features. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. Association for Computing Machinery, New York, NY, USA, 167–176. https://doi.org/10.1145/2642918.2647395

[2] Tyler Angert, Miroslav Suzara, Jenny Han, Christopher Pondoc, and Hariharan Subramonyam. 2023. Spellburst: A Node-based Interface for Exploratory Creative Coding with Natural Language Prompts. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–22. https://doi.org/10.1145/3586183.3606719

[3] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena Glassman. 2023. ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing. arXiv:2309.09128 [cs.HC]

[4] Aaron Bangor, Philip T. Kortum, and James T. Miller. 2008. An Empirical Evaluation of the System Usability Scale. *International Journal of Human–Computer Interaction* 24, 6 (July 2008), 574–594. https://doi.org/10.1080/10447310802205776

[5] Michel Beaudouin-Lafon. 2000. Instrumental Interaction: An Interaction Model for Designing Post-WIMP User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '00)*. Association for Computing Machinery, New York, NY, USA, 446–453. https://doi.org/10.1145/332040.332473

[6] Michel Beaudouin-Lafon. 2004. Designing Interaction, Not Interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI '04)*. Association for Computing Machinery, New York, NY, USA, 15–22. https://doi.org/10.1145/989863.989865

[7] Michel Beaudouin-Lafon and Wendy E. Mackay. 2000. Reification, Polymorphism and Reuse: Three Principles for Designing Visual Interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI '00)*. Association for Computing Machinery, New York, NY, USA, 102–109. https://doi.org/10.1145/345513.345267

[8] Richard A. Bolt. 1980. "Put-that-there": Voice and Gesture at the Graphics Interface. *ACM SIGGRAPH Computer Graphics* 14, 3 (July 1980), 262–270. https://doi.org/10.1145/965105.807503

[9] Patrick D Bridge and Shlomo S Sawilowsky. 1999. Increasing Physicians' Awareness of the Impact of Statistics on Research Outcomes: Comparative Power of the t-Test and Wilcoxon Rank-Sum Test in Small Samples Applied Research. *Journal of Clinical Epidemiology* 52, 3 (March 1999), 229–235. https://doi.org/10.1016/S0895-4356(98)00168-1

[10] John Brooke. 1995. SUS: A Quick and Dirty Usability Scale. *Usability Eval. Ind.* 189 (Nov. 1995), 7.

[11] Xiang 'Anthony' Chen, Jeff Burke, Ruofei Du, Matthew K. Hong, Jennifer Jacobs, Philippe Laban, Dingzeyu Li, Nanyun Peng, Karl D. D. Willis, Chien-Sheng Wu, and Bolei Zhou. 2023. Next Steps for Human-Centered Generative AI: A Technical Perspective. arXiv:2306.15774 [cs]

[12] John Joon Young Chung and Eytan Adar. 2023. PromptPaint: Steering Text-to-Image Generation Through Paint Medium-like Interactions. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3586183.3606777

[13] John Joon Young Chung, Wooseok Kim, Kang Min Yoo, Hwaran Lee, Eytan Adar, and Minsuk Chang. 2022. TaleBrush: Sketching Stories with Generative Pretrained Language Models. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–19. https://doi.org/10.1145/3491102.3501819

[14] Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018. Creative Writing with a Machine in the Loop: Case Studies on Slogans and Stories. In *23rd International Conference on Intelligent User Interfaces (IUI '18)*. Association for Computing Machinery, New York, NY, USA, 329–340. https://doi.org/10.1145/3172944.3172983

[15] Andy Cockburn, Carl Gutwin, Joey Scarr, and Sylvain Malacria. 2015. Supporting Novice to Expert Transitions in User Interfaces. *Comput. Surveys* 47, 2 (Jan. 2015), 1–36. https://doi.org/10.1145/2659796

[16] Richard L. Daft and Robert H. Lengel. 1986. Organizational Information Requirements, Media Richness and Structural Design. *Management Science* 32, 5 (1986), 554–571. jstor:2631846

[17] Hai Dang, Sven Goller, Florian Lehmann, and Daniel Buschek. 2023. Choice Over Control: How Users Write with Large Language Models Using Diegetic and Non-Diegetic Prompting. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–17. https://doi.org/10.1145/3544548.3580969

[18] Hai Dang, Lukas Mecke, and Daniel Buschek. 2022. GANSlider: How Users Control Generative Models for Images Using Multiple Sliders with and without Feedforward Information. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3491102.3502141

[19] Hai Dang, Lukas Mecke, Florian Lehmann, Sven Goller, and Daniel Buschek. 2022. How to Prompt? Opportunities and Challenges of Zero- and Few-Shot Learning for Human-AI Interaction in Creative Applications of Generative Models. arXiv:2209.01390 [cs]

[20] Sjaak de Mul and Herre van Oostendorp. 1996. Learning User Interfaces by Exploration. *Acta Psychologica* 91, 3 (April 1996), 325–344. https://doi.org/10.1016/0001-6918(95)00060-7

[21] Pierre Dragicevic. 2016. Fair Statistical Communication in HCI. In *Modern Statistical Methods for HCI*, Judy Robertson and Maurits Kaptein (Eds.). Springer International Publishing, Cham, 291–330. https://doi.org/10.1007/978-3-319-26633-6_13

[22] Stephen W. Draper and Stephen B. Barton. 1993. Learning by Exploration and Affordance Bugs. In *INTERACT '93 and CHI '93 Conference Companion on Human Factors in Computing Systems (CHI '93)*. Association for Computing Machinery, New York, NY, USA, 75–76. https://doi.org/10.1145/259964.260084

[23] Adam Fourney, Richard Mann, and Michael Terry. 2011. Query-Feature Graphs: Bridging User Vocabulary and System Functionality. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. ACM, Santa Barbara California USA, 207–216. https://doi.org/10.1145/2047196.2047224

[24] Camille Gobert and Michel Beaudouin-Lafon. 2023. Lorgnette: Creating Malleable Code Projections. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, San Francisco CA USA, 1–16. https://doi.org/10.1145/3586183.3606817

[25] Google. 2023. Bard - Chat Based AI Tool from Google, Powered by PaLM 2. https://bard.google.com.

[26] Transparent Statistics in Human-Computer Interaction Working Group. 2019. Transparent Statistics Guidelines. https://doi.org/10.5281/zenodo.1186169

[27] Matthew Guzdial, Nicholas Liao, Jonathan Chen, Shao-Yu Chen, Shukan Shah, Vishwa Shah, Joshua Reno, Gillian Smith, and Mark O. Riedl. 2019. Friend, Collaborator, Student, Manager: How Design of an AI-Driven Game Level Editor Affects Creators. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300854

[28] A. G. Hauptmann. 1989. Speech and Gestures for Graphic Image Manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '89)*. Association for Computing Machinery, New York, NY, USA, 241–245. https://doi.org/10.1145/67449.67496

[29] Jeffrey Heer. 2019. Agency plus Automation: Designing Artificial Intelligence into Interactive Systems. *Proceedings of the National Academy of Sciences* 116, 6 (Feb. 2019), 1844–1850. https://doi.org/10.1073/pnas.1807184115

[30] Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. 1985. Direct Manipulation Interfaces. *Human–Computer Interaction* 1, 4 (Dec. 1985), 311–338. https://doi.org/10.1207/s15327051hci0104_2

[31] Ellen Jiang, Edwin Toh, Alejandra Molina, Aaron Donsbach, Carrie J Cai, and Michael Terry. 2021. GenLine and GenForm: Two Tools for Interacting with Generative Language Models in a Code Editor. In *Adjunct Proceedings of the 34th Annual ACM Symposium on User Interface Software and Technology (UIST '21 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 145–147. https://doi.org/10.1145/3474349.3480209

[32] Peiling Jiang, Jude Rayan, Steven P. Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–20. https://doi.org/10.1145/3586183.3606737

[33] Tae Soo Kim, DaEun Choi, Yoonseo Choi, and Juho Kim. 2022. Stylette: Styling the Web with Natural Language. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3491102.3501931

[34] Alfred Kobsa, Jürgen Allgayer, Carola Reddig, Norbert Reithinger, Dagmar Schmauks, Karin Harbusch, and Wolfgang Wahlster. 1986. Combining Deictic Gestures and Natural Language for Referent Identification. In *Proceedings of the 11th Coference on Computational Linguistics* (Bonn, Germany) *(COLING '86)*. Association for Computational Linguistics, USA, 356–361. https://doi.org/10.3115/991365.991471

[35] Gierad P. Laput, Mira Dontcheva, Gregg Wilensky, Walter Chang, Aseem Agarwala, Jason Linder, and Eytan Adar. 2013. PixelTone: A Multimodal Interface for Image Editing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Paris France, 2185–2194. https://doi.org/10.1145/2470654.2481301

[36] Florian Lehmann, Niklas Markert, Hai Dang, and Daniel Buschek. 2022. Suggestion Lists vs. Continuous Generation: Interaction Design for Writing with Generative Models on Mobile Devices Affect Text Length, Wording and Perceived Authorship. In *Proceedings of Mensch Und Computer 2022 (MuC '22)*. Association for Computing Machinery, New York, NY, USA, 192–208. https://doi.org/10.1145/3543758.3543947

[37] Robert H. Lengel and Richard L. Daft. 1988. The Selection of Communication Media as an Executive Skill. *The Academy of Management Executive (1987-1989)* 2, 3 (1988), 225–232. jstor:4164833

[38] Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast Inference from Transformers via Speculative Decoding. https://doi.org/10.48550/arXiv.2211.17192 arXiv:2211.17192 [cs]

[39] Toby Jia-Jun Li, Marissa Radensky, Justin Jia, Kirielle Singarajah, Tom M. Mitchell, and Brad A. Myers. 2019. PUMICE: A Multi-Modal Agent That Learns Concepts and Conditionals from Natural Language and Demonstrations. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 577–589. https://doi.org/10.1145/3332165.3347899

[40] Michael Xieyang Liu, Advait Sarkar, Carina Negreanu, Benjamin Zorn, Jack Williams, Neil Toronto, and Andrew D. Gordon. 2023. "What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–31. https://doi.org/10.1145/3544548.3580817

[41] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs]

[42] Zhaoyang Liu, Yinan He, Wenhai Wang, Weiyun Wang, Yi Wang, Shoufa Chen, Qinglong Zhang, Zeqiang Lai, Yang Yang, Qingyun Li, Jiashuo Yu, Kunchang Li, Zhe Chen, Xue Yang, Xizhou Zhu, Yali Wang, Limin Wang, Ping Luo, Jifeng Dai, and Yu Qiao. 2023. InternGPT: Solving Vision-Centric Tasks by Interacting with ChatGPT Beyond Language. arXiv:2305.05662 [cs]

[43] Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J. Cai. 2020. Novice-AI Music Co-Creation via AI-Steering Tools for Deep Generative Models. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376739

[44] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. arXiv:2104.08786 [cs]

[45] Aran Lunzer and Kasper Hornbæk. 2008. Subjunctive Interfaces: Extending Applications to Support Parallel Setup, Viewing and Control of Alternative Scenarios. *ACM Transactions on Computer-Human Interaction* 14, 4 (Jan. 2008), 1–44. https://doi.org/10.1145/1314683.1314685

[46] Pattie Maes. 1994. Agents That Reduce Work and Information Overload. *Commun. ACM* 37, 7 (July 1994), 30–40. https://doi.org/10.1145/176789.176792

[47] Damien Masson, Sylvain Malacria, Géry Casiez, and Daniel Vogel. 2023. Statslator: Interactive Translation of NHST and Estimation Statistics Reporting Styles in Scientific Documents. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3586183.3606762

[48] Damien Masson, Jo Vermeulen, George Fitzmaurice, and Justin Matejka. 2022. Supercharging Trial-and-Error for Learning Complex Software Applications. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–13. https://doi.org/10.1145/3491102.3501895

[49] Microsoft. 2023. Introducing the New Bing. https://www.bing.com/new.

[50] Dan Milmo. 2023. ChatGPT Reaches 100 Million Users Two Months after Launch. *The Guardian* (Feb. 2023), 1.

[51] Aditi Mishra, Utkarsh Soni, Anjana Arunkumar, Jinbin Huang, Bum Chul Kwon, and Chris Bryan. 2023. PromptAid: Prompt Exploration, Perturbation, Testing and Iteration Using Visual Analytics for Large Language Models. arXiv:2304.01964 [cs]

[52] B.A. Myers. 1992. Demonstrational Interfaces: A Step beyond Direct Manipulation. *Computer* 25, 8 (Aug. 1992), 61–73. https://doi.org/10.1109/2.153286

[53] Mathieu Nancel and Andy Cockburn. 2014. Causality: A Conceptual Model of Interaction History. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Toronto Ontario Canada, 1777–1786. https://doi.org/10.1145/2556288.2556990

[54] David G. Novick, Oscar D. Andrade, and Nathaniel Bean. 2009. The Micro-Structure of Use of Help. In *Proceedings of the 27th ACM International Conference on Design of Communication (SIGDOC '09)*. Association for Computing Machinery, New York, NY, USA, 97–104. https://doi.org/10.1145/1621995.1622014

[55] Changhoon Oh, Jungwoo Song, Jinhan Choi, Seonghyeon Kim, Sungwoo Lee, and Bongwon Suh. 2018. I Lead, You Help but Only with Enough Details: Understanding User Experience of Co-Creation with Artificial Intelligence. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3174223

[56] OpenAI. 2022. Introducing ChatGPT. https://openai.com/blog/chatgpt.

[57] OpenAI. 2023. GPT Best Practices. https://platform.openai.com/docs/guides/gpt-best-practices.

[58] OpenAI. 2023. OpenAI Node API Library. OpenAI.

[59] OpenAI. 2023. OpenAI Platform. https://platform.openai.com/docs/api-reference/chat/object.

[60] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag Your GAN: Interactive Point-based Manipulation on the Generative Image Manifold. In *ACM SIGGRAPH 2023 Conference Proceedings (SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3588432.3591500

[61] Prism.js 2015. *Prism*. Prism.js. https://prismjs.com

[62] Jef Raskin. 2000. *The Humane Interface: New Directions for Designing Interactive Systems* (1st edition ed.). Addison-Wesley Professional, Reading, Mass.

[63] Meta 2013. *React – A JavaScript Library for Building User Interfaces*. Meta. https://reactjs.org/

[64] Laria Reynolds and Kyle McDonell. 2021. Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–7. https://doi.org/10.1145/3411763.3451760

[65] Andrew Ross, Nina Chen, Elisa Zhao Hang, Elena L. Glassman, and Finale Doshi-Velez. 2021. Evaluating the Interpretability of Generative Models by Interactive Reconstruction. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–15. https://doi.org/10.1145/3411764.3445296

[66] Kevin Sheppard, Stanislav Khrapov, Gábor Lipták, mikedeltalima, Rob Capellini, alejandro cermeno, Hugle, esvhd, Snyk bot, Alex Fortin, JPN, Matt Judell, Ryan

Russell, Weiliang Li, 645775992, Austin Adams, jbrockmendel, LGTM Migrator, M. Rabba, Michael E. Rose, Nikolay Tretyak, Tom Rochette, UNO Leo, Xavier RENE-CORAIL, Xin Du, Joren, and Burak Çelik. 2023. *bashtage/arch: Release 6.1.0.* https://doi.org/10.5281/zenodo.7975104

[67] Shneiderman. 1983. Direct Manipulation: A Step Beyond Programming Languages. *Computer* 16, 8 (Aug. 1983), 57–69. https://doi.org/10.1109/MC.1983.1654471

[68] Ben Shneiderman. 1982. The Future of Interactive Systems and the Emergence of Direct Manipulation†. *Behaviour & Information Technology* 1, 3 (July 1982), 237–256. https://doi.org/10.1080/01449298208914450

[69] B. Shneiderman. 1993. Beyond Intelligent Machines: Just Do It. *IEEE Software* 10, 1 (Jan. 1993), 100–103. https://doi.org/10.1109/52.207235

[70] Ben Shneiderman. 1997. Direct Manipulation for Comprehensible, Predictable and Controllable User Interfaces. In *Proceedings of the 2nd International Conference on Intelligent User Interfaces - IUI '97*. ACM Press, Orlando, Florida, United States, 33–39. https://doi.org/10.1145/238218.238281

[71] Ben Shneiderman and Pattie Maes. 1997. Direct Manipulation vs. Interface Agents. *Interactions* 4, 6 (Nov. 1997), 42–61. https://doi.org/10.1145/267505.267514

[72] Ben Shneiderman, Catherine Plaisant, Maxine Cohen, Steven M. Jacobs, and Niklas Elmqvist. 2017. *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (sixth edition ed.). Pearson, Boston.

[73] Hendrik Strobelt, Albert Webson, Victor Sanh, Benjamin Hoover, Johanna Beyer, Hanspeter Pfister, and Alexander M. Rush. 2022. Interactive and Visual Prompt Engineering for Ad-hoc Task Adaptation with Large Language Models. arXiv:2208.07852 [cs]

[74] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–18. https://doi.org/10.1145/3586183.3606756

[75] Michael Terry, Elizabeth D. Mynatt, Kumiyo Nakakoji, and Yasuhiro Yamamoto. 2004. Variation in Element and Action: Supporting Simultaneous Development of Alternative Solutions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Vienna Austria, 711–718. https://doi.org/10.1145/985692.985782

[76] Priyan Vaithilingam, Tianyi Zhang, and Elena L. Glassman. 2022. Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. ACM, New Orleans LA USA, 1–7. https://doi.org/10.1145/3491101.3519665

[77] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[78] Jianyuan Wang, Ceyuan Yang, Yinghao Xu, Yujun Shen, Hongdong Li, and Bolei Zhou. 2022. Improving GAN Equilibrium by Raising Spatial Awareness. https://doi.org/10.48550/arXiv.2112.00718 arXiv:2112.00718 [cs]

[79] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems* 35 (Dec. 2022), 24824–24837.

[80] Catherine G. Wolf and James R. Rhyne. 1987. A Taxonomic Approach to Understanding Direct Manipulation. *Proceedings of the Human Factors Society Annual Meeting* 31, 5 (Sept. 1987), 576–580. https://doi.org/10.1177/154193128703100522

[81] 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*. ACM, New Orleans LA USA, 1–10. https://doi.org/10.1145/3491101.3519729

[82] Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–22. https://doi.org/10.1145/3491102.3517582

[83] Tom Yeh, Tsung-Hsiang Chang, and Robert C. Miller. 2009. Sikuli: Using GUI Screenshots for Search and Automation. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. Association for Computing Machinery, New York, NY, USA, 183–192. https://doi.org/10.1145/1622176.1622213

[84] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. 2018. Generative Image Inpainting with Contextual Attention. arXiv:1801.07892 [cs]

[85] Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: Story Writing With Large Language Models. In *27th International Conference on Intelligent User Interfaces*. ACM, Helsinki Finland, 841–852. https://doi.org/10.1145/3490099.3511105

[86] J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–21. https://doi.org/10.1145/3544548.3581388

[87] Yaqian Zhu and John Kolassa. 2018. Assessing and Comparing the Accuracy of Various Bootstrap Methods. *Communications in Statistics - Simulation and Computation* 47, 8 (Sept. 2018), 2436–2453. https://doi.org/10.1080/03610918.2017.1348516

A APPENDIX

A.1 Prompts Generated From Physical Actions

Below, we describe the prompts generated from physical actions and that are sent to *gpt-3.5-turbo*.

A.1.1 Localizing the Effect of a Prompt. Below is the prompt generated from the interaction described in the use case scenario where "a White Rabbit" is selected followed by the prompt "add description of its tail". The text is the second paragraph from *Alice's Adventures in Wonderland* by Lewis Carroll.

```
So she was considering in her own mind (as well
as she could, for the hot day made her feel very
sleepy and stupid), whether the pleasure of making
a daisy-chain would be worth the trouble of getting
up and picking the daisies, when suddenly <blank>
with pink eyes ran close by her.

<blank>: a White Rabbit
INSTRUCTION: add description of its tail
Rewrite <blank>. Follow INSTRUCTION
<blank>:
```

The result should be only the specific selected part being rewritten. Then, DirectGPT takes care of incorporating it back into the overall text.

A.1.2 Referring to Textual Objects in a Prompt. Below is the prompt generated from an interaction where two words from the text "hot" and "suddenly" were dropped in a prompt such that it reads "replace [hot] and [suddenly] with synonyms"

```
So she was considering in her own mind (as well as
she could, for the 0]hot0] day made her feel very
sleepy and stupid), whether the pleasure of making
a daisy-chain would be worth the trouble of getting
up and picking the daisies, when 1]suddenly1] a
White Rabbit with pink eyes ran close by her.

replace text delimited by 0] and text delimited by
1] with synonyms
Keep rest of the text identical
```

This should result in the whole text being re-written. Note that the choice of delimiter was done to make sure the model would perceive it as a separate token and would not merge it with the surrounding text. This was empirically found to work slightly better than using more standard delimiters such as XML tags for example.

A.1.3   Referring to Vector Objects in a Prompt. Below is the prompt generated from an interaction where the user prompted "draw a line between this and that" and then dropped two circles instead of "this" and "that"

```
<svg width="300" height="150">
    <circle cx="133" cy="33" r="5" id="c0"></circle>
    <circle cx="151" cy="20" r="5" id="c1"></circle>
</svg>

Return modified SVG code to draw a line between
element with id "c0" and element with id "c1"
```

This should output the SVG rewritten with the line added. Note DirectGPT makes sure all elements have a unique *id*. If not, or if the id is not unique, the ids are modified.