mrCAD Multimodal Communication to Refine Computer-aided Designs

William P. McCarthy[1*] **Saujas Vaduguru**[2*†] **Karl D. D. Willis**[1] **Justin Matejka**[1]
Judith E. Fan[3] **Daniel Fried**[2] **Yewen Pu**[4†]
[1]Autodesk AI Lab [2]Carnegie Mellon University [3]Stanford University
[4]Nanyang Technological University
william.mccarthy@autodesk.com, svadugur@cs.cmu.edu, yewen.pu@ntu.edu.sg

Abstract

In collaborative creation tasks, people steer artifacts towards specific goals by *refining* them with *multimodal* communication over multiple rounds of interaction. In contrast, generative AI excels at creating artifacts in a single turn but can struggle to make precise refinements that match our design intent. To close this gap, we present mrCAD, a dataset of multi-turn interactions in which pairs of humans iteratively created and refined computer-aided designs (CADs).[1] In each game, a *Designer* sent instructions to a *Maker*, explaining how to create and subsequently refine a CAD to match a target design that only the *Designer* could see. mrCAD consists of 6,082 communication games, 15,163 instruction-execution rounds, played between 1,092 pairs of human players. Crucially, *Designer*s had access to two communication modalities – text and drawing. Analysis finds that players relied more on text in refinement than in initial generation instructions, and used different linguistic elements for refinement than for generation. We also find that state-of-the-art VLMs are better at following generation instructions than refinement instructions. These results lay the foundation for modeling multi-turn, multimodal communication not captured in prior datasets.

1 Introduction

Recent advances in generative AI allow people to ask for the creation of artifacts in natural, conversational language. However, while many models generate high-quality output in response, they often struggle to process complex, multi-turn requests (Laban et al., 2025). This challenge is especially pronounced in tasks such as image editing (Fu et al., 2024) and code generation (Cassano et al., 2023), resulting in frustration with AI-generated content

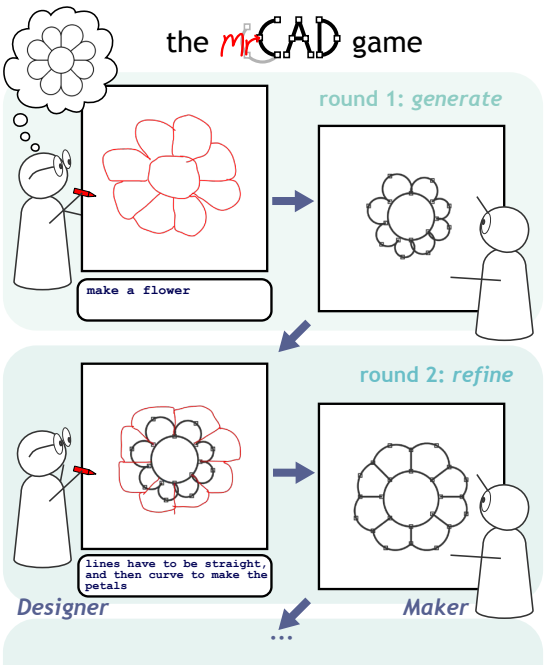


Figure 1: We present mrCAD, a dataset of humans playing a multi-turn, multimodal communication game in a 2D CAD environment. A pair of participants collaborated to recreate a target CAD over multiple rounds. The target design is known only to the *Designer*, who instruct the *Maker* using drawing and text. The *Maker* manipulate the current CAD based on these instructions.

that is "nearly there" but difficult to *refine* to precisely match user intent. On the other hand, the ability of humans to iteratively refine communicated concepts is a core mechanism used to ensure robustness of communication (Dingemanse and Enfield, 2024), as well as a crucial part of the design process (Lawson, 2006; Williams and Cowdroy, 2002). Training AI systems to understand these forms of repair is particularly challenging given that they are typically *multimodal*, relying on the interaction between different communication modalities to make their meaning precise.

We present *mrCAD*, a resource for studying and benchmarking multimodal refinement communication in a computer-aided design (CAD) setting. Our

*equal contribution
†work done at Autodesk AI Lab
[1]https://github.com/AutodeskAILab/mrCAD

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 22905–22921
November 4-9, 2025

task (Figure 1) elicits the following phenomena: (1) contextual dependence, where the refinement instructions must be interpreted in the context of the current CAD (§ 4); (2) interactions between modalities, where drawings and texts must be interpreted together (§ 4); and (3) monotonicity, where participants communicate to iteratively shrink the distance between the current CAD and the target CAD (§ 4 and § 5). These phenomena capture how humans can communicate *flexibly* to accomplish *precise* tasks such as producing CAD designs.

We highlight two main uses of mrCAD: (1) as a **benchmark** for evaluating multimodal, multi-turn instruction following systems such as those powered by vision-language models, and (2) as a resource for **understanding** the nature of human refinement communication.

Benchmarking CAD reconstruction is particularly attractive as an instruction following benchmark for the following reasons: (1) The CAD reconstruction accuracy can be programmatically defined, rather than relying on learned metrics such as CLIP (Radford et al., 2021). (2) CAD programs can be easily manipulated using clicks and drags (similar to a slide design software), making it easy to evaluate both tool-using agents and human participants. (3) The target CAD designs in mrCAD are sourced from SketchGraph (Seff et al., 2020), consisting of naturalistic 2D designs created by human designers, rather than synthetically generated from a domain-specific language that the authors have crafted by hand.

Understanding As a resource for understanding the nature of multimodal refinement in human-human collaboration, mrCAD is large scale, containing 6,082 human-to-human plays of the game, with a total of 15,163 rounds of instruction. A large number of CADs in our dataset are recreated *multiple* times by different dyads, ranging from 2 to 30 games per design, capturing the natural variation in refinement strategies across people. A notable feature of CAD is that while it only contains simple curves, they combine form rich semantic objects and sub-parts (like the "flower" and "petals" in Figure 1), which must be parsed contextually (Ji et al., 2022). This is in contrast to works such as Zitnick and Parikh (2013), where each object in the scene has a canonical, pre-defined name.

Our work makes the following contributions:

- A novel dataset of multimodal refinement of CAD designs.
- A detailed procedure for collecting a dataset of multimodal refinement communication using crowd-sourcing.
- The dataset and benchmark, consisting of 15,163 instructions and executions, wrapped in an accessible gym environment.
- Analyses of human-human communication that reveal differences in the multimodal languages of generation and refinement.
- Evaluation of existing VLM models, revealing a severe gap in their refinement ability, particularly compared to their generation ability.

2 The mrCAD Task

The mrCAD task (Fig. 1) is a two player, multi-turn communication game. Following McCarthy et al. (2024), a *Designer* issues multimodal instructions to guide a *Maker* to take actions in a mutually observable CAD environment to construct a target CAD, known only to the *Designer*.

2.1 mrCAD environment

State The state D is a CAD data structure.

```
D -> {Curve, ...}
Curve  -> Line | Circle | Arc
Line   -> l(P,P) // end points
Circle -> c(P,P) // points on diameter
Arc    -> a(P,P,P) // start, mid, end
```

Each curve is defined by its *control points* that the *Maker* selects. A `render` function allows players to view the state by rendering the design as an image.

Action The *Maker* can alter the CAD by making, removing, and moving curves via translation. They can also change the shape of a curve by moving its control points. If multiple curves share the same control point, moving the point would modify all the curves, and deleting the point would delete all the adjacent curves.

```
Action -> make_curve Curve
        | remove_curve Curve
        | move_curve Curve Vxy
        | move_point P P
        | delete_point P
```

Transition Actions $\vec{a} = [a_1, a_2, \ldots, a_n]$ can be applied to a design, which results in a new design.

$$D' = \vec{a}(D) = [a_1, a_2, \ldots, a_n](D)$$
$$= (a_n \circ \cdots \circ a_1)(D)$$

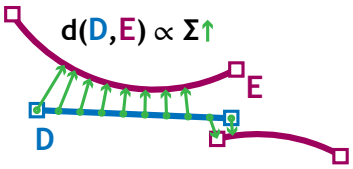


Figure 2: The **asymmetric Chamfer distance** from CAD D to CAD E is calculated by sampling 10 points on every curve of D, and calculating (symbolically) the minimum distance from each point to E. Each distance is then normalized by multiplying $\frac{1}{4}$ of the maximum size of the canvas, making it invariant to the canvas size. These distances are summed, making the asymmetric Chamfer distance. The symmetric Chamfer distance we use is the average of both directions.

Distance metric CADs are programmatic entities that exist in a visual space, where very different elements could be used to create two CADs of similar appearance. For example, a series of straight lines can be used to represent a curve. Therefore, we devise a distance metric Δ based on Chamfer distance (Butt and Maragos, 1998) that respects the vector-like nature of CADs while also accounting for geometric similarity (Fig. 2).

Instructions The *Designer* can instruct the *Maker* using multimodal messages.

```
Message -> ⟨Text, Drawing⟩
Text    -> [char, ...] | empty
Drawing -> [stroke, ...] | empty
```

A text is a sequence of characters and a drawing is a sequence of strokes, encoded in SVG format.

2.2 Playing the Game

The shared goal of the *Designer* and *Maker* is to collaboratively reconstruct a target design, which is only given to the *Designer*.

Round and rollout A round is a tuple of: a design D, the message from the *Designer* m, the actions generated from the maker $\vec{a}$, and the resulting updated design $D' = \vec{a}(D)$. A rollout is a sequence of rounds.

$$r_i = (D_i, m_i, \vec{a}_i, D'_i)$$
$$R = [r_1, \ldots, r_n]$$

Each rollout has $D_1 = \{\}$ and $D_i = D'_{i-1}$. A game is "won" if the final design is within a certain threshold θ of distance from the target D^*.

$$\Delta(D'_n, D^*) < \theta$$

Designer At round i, the *Designer* is given both the target and current designs, rendered as images of D^* and D_i, along with the history of the (rendered) interaction of previous rounds $\texttt{render}(R_{1:i-1})$, and generates a message m from

$$P_{\text{designer}}(m \mid \texttt{render}(D^*), \texttt{render}(D_i), \texttt{render}(R_{1:i-1}))$$

Presenting the *Designer* only the rendered CADs encourages them to focus on the geometric properties and communicate the refinements in a naturalistic manner, instead of communicating about the underlying programmatic representation itself.

Maker The *Maker* takes in the message, the current design, the interaction history, and generates a sequence of actions $\vec{a}$ from

$$P_{\text{maker}}(\vec{a} \mid m, D_i, R_{1:i-1})$$

Play Playing the game starts with a target design D^* and involves a dyad (a *Designer-Maker* pair) acting in turns to generate a rollout.

$$R = \texttt{play}(D^*, P_{\text{designer}}, P_{\text{maker}})$$

3 Effects of Multimodality and Refinement on Communication

We first confirm that multimodality and refinement do indeed play important roles in communication of designs. We conduct an ablation study by manipulating the kind of communication schemes available to the participants (details in Appendix B.1):

1. **multimodal + refinement:** (original condition): 3 rounds total. Each round consists of instruction (30s) and execution (120s).
2. **drawing-only + refinement:** the *Designer* can only draw but not use text.
3. **text-only + refinement:** the *Designer* can only use text but not draw.
4. **multimodal + generation-only:** the pair of participants had 1 big round of instruction (90s) and execution (360s), so they can only generate at once but not refine.

Results are summarized in Figure 3. In conclusion: (1) drawing is important, and (2) refinement is important. Note that while drawing-only performed as well as multimodal in this study (Figure 3A), in the multimodal condition participants

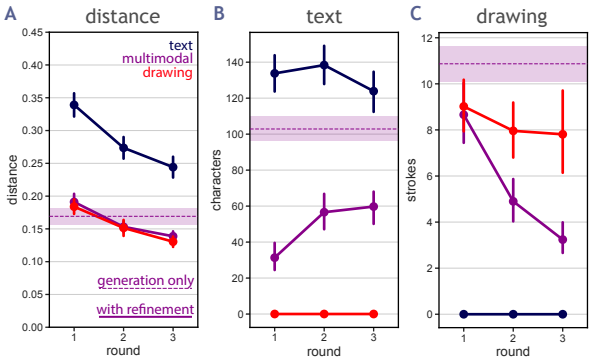


Figure 3: **A**: reconstruction accuracy for the 4 communication conditions — multimodal+refinement, text only + refinement, drawing only + refinement, and multimodal + generation only. Using text only was less effective. Generation-only was less effective. Drawing only and multimodal are comparable in performance. **B**: usage of text across rounds — in the multimodal condition, participants used more texts in the later refinement rounds, suggesting a usage of text in conjunction with drawings to communicate refinements. **C**: usage of drawing across rounds — in the multimodal condition, participants used more drawing in the generation round, and less in the refinement rounds.

still used more text and less drawing over time (Figure 3B,C). This points to a preference to use more drawing in communicating generation and more text in communicating refinements. Statistical analysis of results are given in Appendix B.2.

4 The mrCAD Dataset

We detail the collection of the dataset, present the main statistics of our dataset, and a set of basic analyses that reveals the striking effects of multimodality and refinement on communication. We also include a gallery of multimodal refinement rounds (Figure 5), highlighting the intriguing yet unexplored aspects of the dataset for future research.

4.1 Collection

Our goal is to collect a large number of high-quality, multimodal refinement instructions. We designed an online platform for crowd workers to play the mrCAD game (Appendix C.1) and recruited players on Prolific. We seeded the mrCAD communication games with target designs created by people (Seff et al., 2020; Appendix C.4). We then presented these games in sequence to crowd workers, taking measures to allow skilled players to play longer while limiting the annotations from unskilled players (Appendix C.5). We also designed a dynamic submission threshold to encourage more

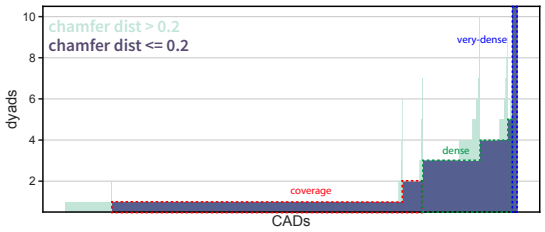


Figure 4: The mrCAD dataset contains three subsets: the **coverage set** of 2249 CADs with 1-2 successful rollouts, **dense set** of 698 CADs with 3+ successful reconstruction, and the **very-dense set** of 27 CADs with 30+ successful reconstruction.

refinement instructions (Appendix C.6) as well as a fixed threshold on final reconstructions to exclude poor reconstructions (Appendix C.7).

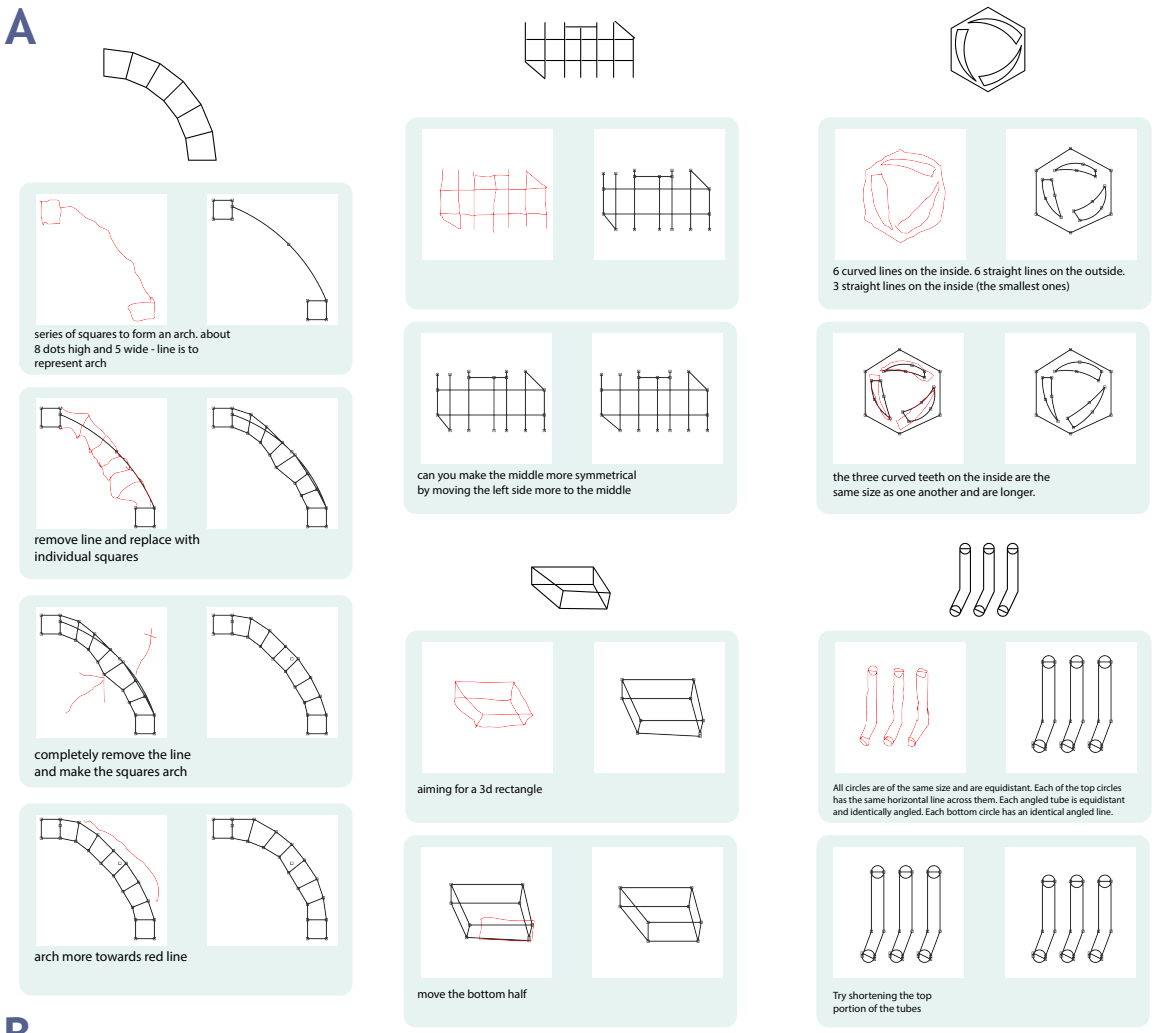
4.2 Dataset statistics

mrCAD contains 3166 unique CADs recreated by 1092 dyads. This resulted in a total of 6082 rollouts (for a sample, see Figure 5A) with a total of 15163 rounds of instructions and corresponding executions. On average, each rollout has 2.49 rounds, and there are a total of 6078 generation rounds (round 1) and 9085 refinement rounds (rounds 2+). To achieve coverage over diverse CADs while also capturing variance in human communication (Ji et al., 2022), our dataset contains three distinct subsets: a **coverage set**, containing 2249 unique CADs each successfully reconstructed by 1–2 dyads; a **dense set**, containing 698 unique CADs each successfully reconstructed by at 3–6 dyads; and a **very-dense set**, containing 27 unique CADs successfully reconstructed by at least 30 dyads (Figure 4). In the following analyses we combine data from the coverage and dense sets, and leave the very-dense set for future work.

4.3 Analyzing instructions in mrCAD

Refinements increased accuracy of CADs After the initial round, distance to the target continued to decrease ($b = -0.0512$, $t = -16.5$, $p \leq 0.001$; Figure 7). These improvements got smaller as rounds progressed ($b = 0.0094$, $t = 8.84$, $p \leq 0.001$), suggesting that refinements became more fine-grained in later rounds.

Modality use differs in generation and refinement Across the 4946 rollouts analyzed, 83.4% are multimodal, 14.5% are drawing-only, and only 2.1% are text-only. Across the individual rounds analyzed, 53.5% are multimodal, 30.2% are drawing-



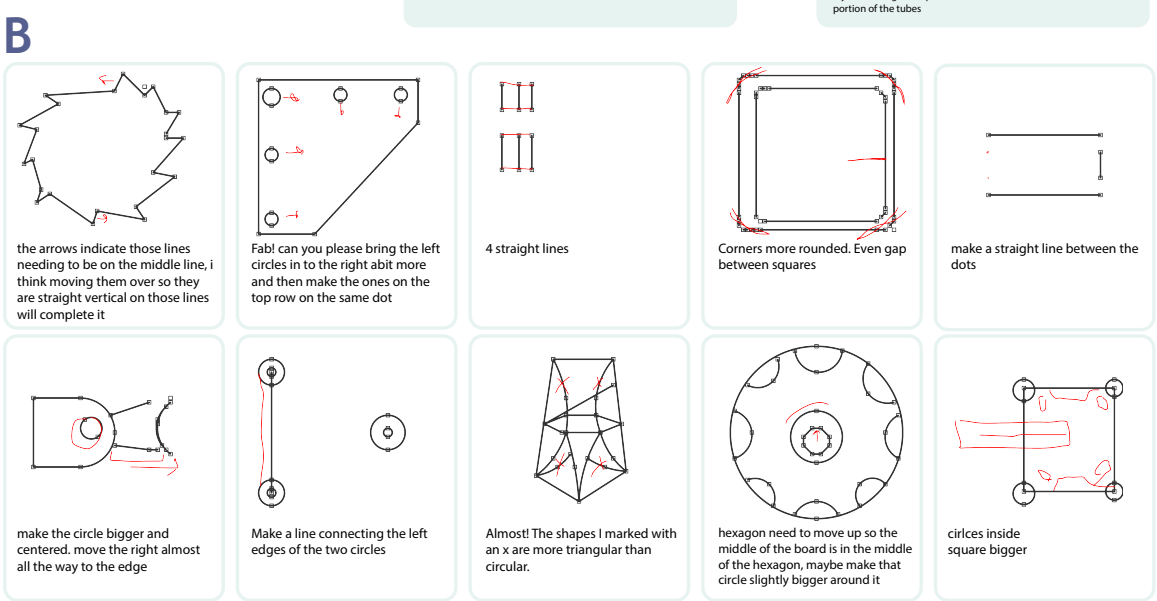


Figure 5: **A** Example rollouts from the dataset. Target CADs (top-center) were shown to *Designers*, who created instructions (left columns) that *Makers* followed (right columns). Dyads iteratively refined their CADs across a series of rounds (rows). **B** Examples of multimodal refinement instructions. Language and drawing mutually constrain and inform the others' semantics. Many instructions don't make sense without the accompanying drawings, and vice-versa.

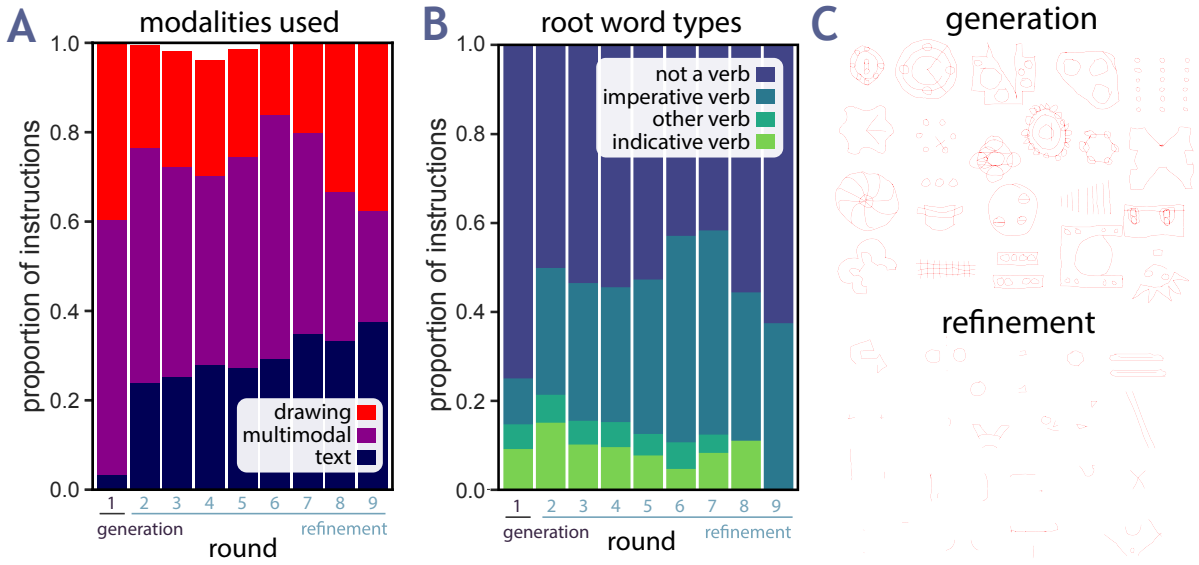


Figure 6: **A** Designers' instructions to *generate* CADs (round 1) involved lots of drawing and little text, whereas instructions to *refine* CADs (rounds 2+) used a balance of modalities. **B** The proportions of the types of root words in the dependency parse tree of instruction text. More verbs are used over rounds, and these verbs become more imperative. **C** Samples of 20 generation drawings and 20 refinement drawings highlights the rich detail in generation instructions, and more targeted modifications in refinement.

only, and 16.3% are text-only. The distribution of text, drawing, and multimodal instruction was strikingly different in generation (round 1) refinement (rounds 2+) ($\chi^2(2) = 1095,\ p = 0$); almost all generation instructions involved drawings (96.6%), whereas refinement rounds had a roughly equal split of unimodal text and drawing (Figure 6A).

Refinement drawings are more sparse than generation drawings Most strikingly, compared to generation, *Designer*s drew *less* in refinement rounds, as measured by the number of strokes ($b = -5.64, t = -72.8, p \leq 0.001$) and amount of digital "ink" ($b = 20.9, t = 2.23, p = 0.026$) used. Rendering these drawings (Figure 6C) suggests that *Designers*' drawings changed from mostly complete drawings of target CADs in generation to smaller modifications of sub-parts of the current CAD in refinements.

Refinement text expresses actions and directives The text of instructions contained an average of 50.2 characters, or 9.66 words. We use the English pipeline of the Stanza NLP package (Qi et al., 2020) to parse the sentences and extract the root words. We tag the root word based on whether it is a verb, and if so whether the mood of the verb is indicative, imperative, or otherwise. We see in Figure 6B that generation instructions don't have a verb at the head a majority of the time, while refinement instructions do. The proportion of imperatives – verbs typically used to express directives issued to a listener – also increases over the course of refinement. This suggests refinement and generation instructions use different languages.

Multimodal refinement messages There are a total of 3723 multimodal refinement messages. Of these, many contained drawings and text that worked together to convey meaning, such that either modality alone would not be sufficient to convey the same intent (Fig. 5B). This dataset, along with the *very-dense* subset, provide a resource for future work to investigate how language and drawing are used *together* to communicate precisely.

5 Evaluation of frontier models

We evaluate both API access models and open-weights models (which we finetune) on the mrCAD dataset. We are interested in the following.

RQ1 What is the difference in performance between humans and models in following the instructions in mrCAD?

RQ2 What is the effect of generation vs refinement in model performance ?

RQ3 What is the effect of multimodal vs single-modal in model performance?

5.1 Benchmark

We select CADs from the **dense set** that have 3 or more successful ($\Delta(D_n, D^*) < 0.2$) rollouts. This results in 682 unique CADs with 2324 rollouts,

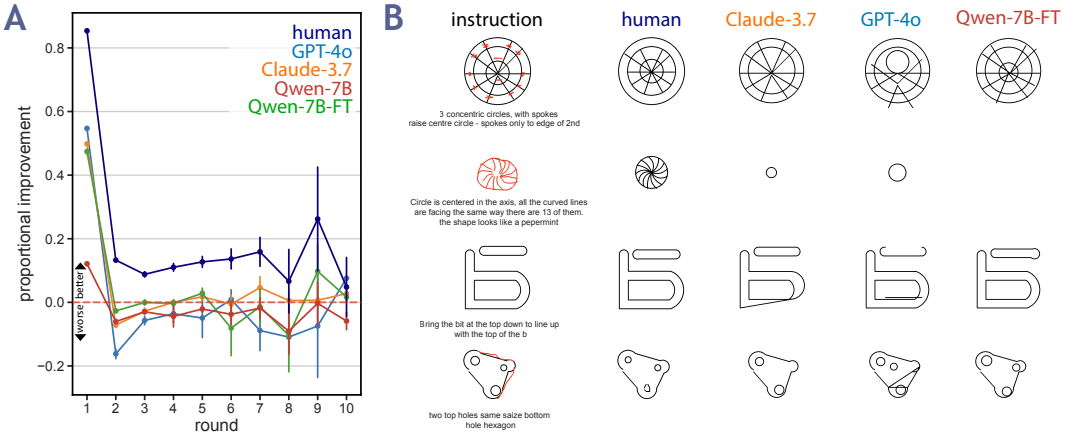


Figure 7: **A** Comparison of human and model movement towards target following instructions, normalized by distance at start of round. Only humans make reliably positive changes in responses to refinement instructions. Models made positive steps in generation but largely destructive changes when refining. **B** Comparison of human and model responses.

and a total of 5751 rounds of instruction-execution pairs. Then, for these selected rounds, we can evaluate VLM agents by having them assume the role of a human maker P_{maker}.

Metric We evaluate the instruction following abilities by proportional improvement **PI**: how much the distance to the target D^* shrinks as a consequence of an agent's actions.

$$\text{PI}(A, R_{:i}, D^*) = \frac{\Delta(D_i, D^*) - \Delta(A(D_i), D^*)}{\Delta(D_i, D^*)}$$

This metric is a proportion of the remaining distance to target at every round. This accounts for the fact that distances to the target decrease significantly as rounds progress, making late-stage refinements less noticeable without normalization.

5.2 Evaluating vision-language models

Gym environment We build a gym framework for standardized evaluations across models. The gym framework allows for instantiating agents that interface with a standardized representation of CADs, and allows for handing the mrCAD dataset, evaluating models, and simulating interactions between designer and maker agents. We evaluate multiple vision-language chat models by having these models assume the role of a human maker.

$$P_{\text{model-maker}}(A \mid m, D_i, R_{1:i-1})$$

API access models Since we present the rendered interaction history and drawing instructions to the model, we require models that are accept interleaved image-text inputs. We choose the following models off-the-shelf based on this criterion: GPT-4o (OpenAI, 2024), GPT-4o-mini (OpenAI, 2025), Claude-3.7-Sonnet (Anthropic, 2025), and Qwen2.5-VL-7B-Instruct (Bai et al., 2025). To generate editing actions, we prompt the models to generate editing actions as tool calls.

Finetuned models We also fine-tune a Qwen2.5-VL-7B-Instruct model in the same setting to generate editing actions with supervised fine-tuning. The models are given all the previous rounds of interaction between the pair of humans that played that game, the most recent instruction, and trained to predict the editing actions. The actions are formatted as a sequence of tool calls, and the model is trained to produce tool calls. We train LoRA (Hu et al., 2022) adapters for all linear modules.

We use the 2684 rollouts from the coverage set as a training set. We also include rollouts that don't pass the evaluation threshold. We create a small held-out validation set by choosing 303 successful rollouts from the dense set for 187 CADs that have fewer than 3 successful rollouts for model selection.

Ablation on modality We ablate one modality (if present) from all the instructions and evaluate them to create −drawing and −text evaluation sets. We also perform these ablations on the training data and finetune separate Qwen2.5-VL-7B-Instruct models to obtain Qwen-7B FT$_{-\text{drawing}}$ and Qwen-7B FT$_{-\text{text}}$ models to evaluate on the −drawing and −text sets respectively. We use the

same LoRA finetuning setup for these experiments.

5.3 Performance of vision-language models

For the results, we group rounds as either generation (round 1) and refinement (round 2+). Note that this grouping is imprecise and can be improved in future works, as later rounds can contain generations as well. We show some examples of model actions in Figure 7B.

RQ1, RQ2 We found (Table 1) that models are generally able to decrease the distance to the target during *generation*, with more capable models (GPT-4o, Claude-3.7-Sonnet) and fine-tuned models doing better. However, we find that models perform quite poorly in *refinement* turns. We see that models often make changes that actually *increase* the distance to the target – in opposition to the intention of the designer's instruction. We also see that while an approach like supervised fine-tuning leads to significant improvements in generation turns, the gains don't transfer to refinement turns. This suggests that more sophisticated training approaches might be needed to tackle refinement.

We also investigated the influence of context. People are able to interpret a refinement instruction in the context of all instructions that came before it and improve the design. To study whether models leverage previous turns of the interaction to improve performance, we presented models with only the most recent state and instruction as opposed to the entire trajectory. In refinement rounds (where this results in a different context; −context rows in Table 1) we observed better performance for both GPT-4o and GPT-4o-mini which we tested. We further investigate the connections between context and the number of actions taken in Appendix E. However, the proportional improvement metric is still negative for these models, highlighting the room for improvement.

RQ3 We also see that models are sensitive to both instruction modalities (Table 1), with performance descreasing when a multimodal instruction is ablated to have only a single modality. This effect is more pronounced in generation rounds.

	generation	refinement
Human	0.854	0.119
GPT-4o	0.547	−0.119
−context	–	−0.036
−drawing	0.508	−0.159
−text	0.540	−0.124
Qwen-7B FT	0.474	−0.017
−drawing	0.437	−0.038
−text	0.470	−0.035
GPT-4o-mini	0.308	−0.129
−context	–	−0.049
Claude-3.7	0.520	−0.112
Qwen-7B	0.121	−0.050

Table 1: Proportional improvement results for vision-language chat models, including results on ablated instructions. All models performed worse than humans at following instructions, especially refinement instructions, where the models' outputs made designs worse.

6 Related Work

Multimodality The instructions in mrCAD are multimodal, consisting of both texts and drawings. This is in contrast to other works such as Pejsa et al. (2016) and Ku et al. (2020), where the context of the agent is multimodal, yet the instructions themselves only contain language. Our drawings efficiently convey spatial information and reference, and even short utterances – "delete this", "move to here" – can hugely constrain the meaning of drawings. Our work highlights the capacity of drawing to act, not only as a depictive medium, but as a versatile tool for *communication* (Goodman, 1976; Huey et al., 2021; Fan et al., 2023), and provides a resource for studying how language is used in conjunction with another medium of communication in a grounded way (Lachmy et al., 2022).

Generating Designs Prior work has explored AI agents that assist with computer aided design tasks, including those that leverage various kinds of input modality (Sanghi et al., 2022; Chen et al., 2024), including drawings (Seff et al., 2021). Beyond CAD, other work has explored how agents can support various kinds of designs given multimodal inputs, including generating HTML and CSS code (Si et al., 2024) and slide-shows (Ge et al., 2025).

Refinement The interactions between people in mrCAD take multiple turns, as a dyad collaboratively refines a current CAD toward a specific target. Lachmy et al. (2022) study complex instructions, but these are presented all at once by the speaker and verified with interpretation by another agent. mrCAD prioritizes refinement, where people must elaborate on and repair previous instructions. In

this regard, our work is similar to Kim et al. (2019). Zhou et al. (2025) present an algorithm to train LLM agents to engage in multi-turn collaborative design interactions with a simulated user.

Our work is also similar to work by Narayan-Chen et al. (2019) that presents a dataset of collaborative construction interactions in a Minecraft environment. Both works focus on a similar setting of an "architect" or "designer" agent that knows the target instructing a "builder" or "maker" to create it while not being able to act in that environment. We also introduce the dimension of multimodal instructions, which provide an additional facet of context in which language is grounded. Our lighter-weight environment also allows us to collect a substantially larger number of interactions, while also having more complex targets and reconstructions (perfect reconstruction is very difficult for complex targets in our setting, while it is not the case in theirs).

7 Discussion

We present mrCAD, a large-scale dataset of humans playing multi-turn, multimodal communication games in a CAD environment. Analysis revealed significant differences between instructions to *generate* and instructions to *refine*, notably a trend towards the complementary use of drawing and text when refining designs. This distinction is mirrored by a severe performance gap in frontier VLMs ability to generate vs to refine CADs.

We posit that this is due to the lack of multimodal refinement data on the internet, from which these models are trained: (i) *Lack of drawing-as-instruction data:* While the internet is a rich source of textual and image data, data of drawings used as instructions is much sparser – particularly the ephemeral drawings that accompany language. (ii) *Lack of refinement trajectories:* Furthermore, people typically upload only finished artifacts to the internet, often with accompanying contextual text data, making these datasets well-suited for generation tasks. On the other hand, the process of editing artifacts based on further instructions is rarely uploaded, which is why these models struggle with interactive refinements. As a multimodal refinement dataset, mrCAD serves as a valuable resource for investigating these phenomena.

Limitations

The *Designer*s and *Maker*s are crowdworkers recruited through Prolific. Their communication might not fully capture the nuances of language use by CAD experts. Our interface also doesn't capture how expert users of CAD software work on designs. The 2D CAD models used as targets given to the participants are quite simple compared to industrial and architectural designs used in the real world. The inclusion threshold for the dataset ($\theta = 0.2$) is bespoke, and may have to be chosen differently in other settings. We also note that the distance metric we use is based on geometry, and does not reflect higher-level semantics such as symmetry, parallelism, coincidence of curves, which are important in real CAD manufacturing.

Intended use, risks, and ethical considerations

We release the data for non-commercial use under the CC BY-NC 4.0 license. We intend the data to be used to evaluate and train machine learning models in research settings, and in scientific analysis of human communicative behavior. We do not anticipate adverse impacts from the release of this data.

Acknowledgements

We would like to thank Alane Suhr, Stella Zhou, Maxwell Nye, Robert Hawkins, Lio Wong, Jack Terwilliger, Ian Huang, Adriana Schulz, Daniel Ritchie, Gabe Grand, Jacob Andreas, David Hall, Howard Chen, Austin Wang, Yoav Artzi, Khanh Nguyen, Tone Xu, Priyan Vaithilingam, Fernando Diaz, The Cognitive Tools Lab at Stanford and the AI Research Lab at Autodesk for helpful discussions and feedback.

References

Anthropic. 2025. Claude 3.7 sonnet system card. Technical report, Anthropic. Accessed: 2025-03-01.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

M Akmal Butt and Petros Maragos. 1998. Optimum design of chamfer distance transforms. *IEEE Transactions on Image Processing*, 7(10):1477–1484.

Federico Cassano, Luisa Li, Akul Sethi, Noah Shinn, Abby Brennan-Jones, Jacob Ginesin, Edward Berman, George Chakhnashvili, Anton Lozhkov, Carolyn Jane Anderson, and 1 others. 2023. Can it edit? evaluating the ability of large language models to follow code editing instructions. *arXiv preprint arXiv:2312.12450*.

Hansheng Chen, Ruoxi Shi, Yulin Liu, Bokui Shen, Jiayuan Gu, Gordon Wetzstein, Hao Su, and Leonidas Guibas. 2024. Generic 3d diffusion adapter using controlled multi-view editing. *arXiv preprint arXiv:2403.12032*.

Mark Dingemanse and Nick J Enfield. 2024. Interactive repair and the foundations of language. *Trends in Cognitive Sciences*, 28(1):30–42.

Judith E Fan, Wilma A Bainbridge, Rebecca Chamberlain, and Jeffrey D Wammes. 2023. Drawing as a versatile cognitive tool. *Nature Reviews Psychology*, 2(9):556–568.

Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Yang Wang, Yinfei Yang, and Zhe Gan. 2024. Guiding instruction-based image editing via multimodal large language models. In *International Conference on Learning Representations*.

Jiaxin Ge, Zora Zhiruo Wang, Xuhui Zhou, Yi-Hao Peng, Sanjay Subramanian, Qinyue Tan, Maarten Sap, Alane Suhr, Daniel Fried, Graham Neubig, and 1 others. 2025. Autopresent: Designing structured visuals from scratch. *arXiv preprint arXiv:2501.00912*.

Nelson Goodman. 1976. *Languages of art: An approach to a theory of symbols*. Hackett publishing.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Holly Huey, Caren M Walker, and Judith E Fan. 2021. How do the semantic properties of visual explanations guide causal inference? In *Proceedings of the annual meeting of the cognitive science society*, volume 43.

Anya Ji, Noriyuki Kojima, Noah Rush, Alane Suhr, Wai Keen Vong, Robert D Hawkins, and Yoav Artzi. 2022. Abstract visual reasoning with tangram shapes. *arXiv preprint arXiv:2211.16492*.

Jin-Hwa Kim, Nikita Kitaev, Xinlei Chen, Marcus Rohrbach, Byoung-Tak Zhang, Yuandong Tian, Dhruv Batra, and Devi Parikh. 2019. CoDraw: Collaborative drawing as a testbed for grounded goal-driven communication. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6495–6513, Florence, Italy. Association for Computational Linguistics.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412, Online. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. Llms get lost in multi-turn conversation. *Preprint*, arXiv:2505.06120.

Royi Lachmy, Valentina Pyatkin, Avshalom Manevich, and Reut Tsarfaty. 2022. Draw me a flower: Processing and grounding abstraction in natural language. *Transactions of the Association for Computational Linguistics*, 10:1341–1356.

Bryan Lawson. 2006. *How designers think*. Routledge.

William P McCarthy, Justin Matejka, Karl DD Willis, Judith E Fan, and Yewen Pu. 2024. Communicating design intent using drawing and text. In *Proceedings of the 16th Conference on Creativity & Cognition*, pages 512–519.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. Association for Computational Linguistics.

OpenAI. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

OpenAI. 2025. Gpt-4o mini: Advancing cost-efficient intelligence. Accessed: 2025-03-01.

Tomislav Pejsa, Julian Kantor, Hrvoje Benko, Eyal Ofek, and Andrew Wilson. 2016. Room2room: Enabling life-size telepresence in a projected augmented reality environment. In *Proceedings of the 19th ACM conference on computer-supported cooperative work & social computing*, pages 1716–1725.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International*

conference on machine learning, pages 8748–8763. PmLR.

Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. 2022. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613.

Ari Seff, Yaniv Ovadia, Wenda Zhou, and Ryan P Adams. 2020. Sketchgraphs: A large-scale dataset for modeling relational geometry in computer-aided design. *arXiv preprint arXiv:2007.08506*.

Ari Seff, Wenda Zhou, Nick Richardson, and Ryan P Adams. 2021. Vitruvion: A generative model of parametric cad sketches. *arXiv preprint arXiv:2109.14124*.

Chenglei Si, Yanzhe Zhang, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. 2024. Design2code: How far are we from automating front-end engineering? *Preprint*, arXiv:2403.03163.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Anthony Williams and Robert Cowdroy. 2002. How designers communicate ideas to each other in design meetings. In *DS 30: Proceedings of DESIGN 2002, the 7th International Design Conference, Dubrovnik*.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. 2025. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *Preprint*, arXiv:2503.15478.

C Lawrence Zitnick and Devi Parikh. 2013. Bringing semantics into focus using visual abstraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3009–3016.

A Analysis

A.1 Statistical methods

Our primary statistical method was the linear mixed effects model. For modality experiments, we fit models with fixed effects for round and condition, as well as their interaction, and random intercepts for dyad. We then compared this full model to a series of simpler, nested models with predictors removed. We report results from the most complex model for which AIC substantially dropped, compared to the subsequent simpler model. For the effect of refinement, we we ran paired t-tests between the ***multimodal + refinement*** condition and the ***multimodal + generation only*** condition.

A.2 Distance metric

The distance between two designs D, E, are calculated as follows.

```
1  def dist_chamfer_symmetric (D, E):
2    return 0.5 * (dist_asymmetric(D, E) +
       dist_asymmetric(E, D))
3
4  def dist_asymmetric(D, E):
5    pts_in_D = [sample_points(curve) for
       curve in D].flatten()
6    pts_dists_to_E =
       [dist_pt_to_design(pt, E) for pt in
       pts_in_D]
7
8  def dist_pt_to_design(pt, E):
9    return min([dist_pt_to_curve(pt,
       curve) for curve in E])
```

The distance between a point and a curve (line 10) is omitted for brevity, for details please consult the code base. The default value for the min operator (line 10) is $\frac{1}{4}$, expressing the idea that beyond half a quadrant away, two points are not likely to be unrelated.

In the analysis, we found a small error in our javascript implementation of the accuracy function that led to a proportion of lower performing trials receiving greater distances than they should have. As accuracy for inclusion of trials was recalculated post-hoc, entries in our dataset were unaffected. However, some participants may have performed extra rounds of modification to meet the target threshold.

B Multimodality and Refinement Ablations

B.1 Experimental methods

Crowd source participants were paired and randomly assigned the role of *Designer* or *Maker*. We sampled 100 CADs from the Sketchgraph dataset (Seff et al., 2020), and presented these CADs across the set of dyads in each of four conditions:

multimodal + refinement *Designer*'s messages could include text (up to 200 characters) and/or drawing instruction (unlimited). There are 3 rounds total, each round with 30 seconds for the *Designer* to construct a message, and 120 seconds for the *Maker* to manipulate the current CAD in a CAD interface.

text only + refinement Where the message could only contain text but not drawing.

drawing only + refinement Where the message could only contain drawing but not text.

multimodal + generation only Here, rather than having 3 rounds of communication, they are aggregated into a single big round, with 90 seconds for the *Designer* and 360 seconds for the *Maker*.

B.2 Results

Refinement improves reconstruction Participants in all **refinement** conditions were able to improve their reconstructions over all three rounds ($b = -0.0335$, $t = -6.91$, $p =< 0.001$) (Fig. 3A). While participants in the multimodal refinement condition on round one generated less accurate CADs than the **generation only** condition ($t = 1.47$, $p = 0.145$), by round three their designs were reliably more accurate ($t = -2.15$, $p = 0.0340$), despite having the same total time available. This highlights the value of iterative refinement for creating accurate designs.

Drawings improves reconstruction Participants who could only use text achieved considerably less accurate reconstructions than those in the drawing-only and multimodal conditions ($b = 0.127$, $t = 6.54$, $p =< 0.001$). While text is a relatively poor communicative medium on its own, participants in the multimodal condition still opted to use text in conjunction with drawings, and increased their use of text in later rounds (Fig. 3B), suggesting that language is useful for *refining* CADs. The amount of drawing remained consistent in the drawing-only condition, but dropped precipitously after round 1 in the multimodal condition (Fig. 3C). Overall, when given a choice of multimodality, human participants favored more drawing for the generation round, and more texts in the refinement rounds.

Interim discussion Why do participants choose to use multimodal instructions when given a choice? Manually inspecting the data revealed that drawings are primarily used in two ways: Those used for creating a shape, and those used for editing a shape. In the multimodal condition, the *Designer* can disambiguate these with texts such as "make this shape" or "move the shape as shown". However, in the drawing only condition, the *Designer* frequently resorted to redrawing the entire target CAD from scratch — a valid yet inefficient strategy. Data collected from this experiment hint towards a complex interplay between linguistic and graphic communication that differs in generation and refinement settings. The mrCAD dataset, discussed next, aims to provide a large scale dataset of *multimodal generation and refinement instructions* to study these phenomena.

C Data Collection Procedure

C.1 mrCAD annotation task

We recruited fluent English speakers from the USA and UK on Prolific. Prolific workers were told that their data was being collected for research and development purposes, presenting findings in academic papers, presentations, or datasets, and gave consent. Our institution did not require IRB approval for this work. We also do not release any personally identifying information for participants in the dataset. Participants were paired, randomly assigned the role of *Designer* or *Maker*, and worked together over a series of rounds to recreate target CADs.

Rather than fixing a number of rounds and the amount of time per round, we instead limited the maximum round number to 10, and time to 9 minutes. Within these limits, participants can choose how many rounds they will take, and to allocate different amount of times for each round. We also lifted the limit on the amount of text characters the participants can send in a message.

C.2 Dataset collection interface

Figure 8 shows screenshots of the user interface used for data collection for both players.

C.3 Dataset collection parameters

While the key features of our data collection paradigm were kept consistent between our controlled studies (Section 3) and full dataset collection (Section 4), we implemented several changes between versions to increase efficiency of data collection.

C.4 Sampling CAD tasks

We sample tasks from the SketchGraphs dataset (Seff et al., 2020), which is released under the permissive MIT license. We first normalize all designs to a 20×20 grid to identify duplicates. We then rescale the designs by a random scaling factor to ensure a variety of design sizes, while ensuring a minimum gap between pairs of elements such as control points, parallel lines, and concentric arcs or circles to ensure that they were far enough apart to be reproducible in the CAD construction interface. We then grouped designs into buckets based on a 'signature' determined by the count of various types of curves in the design (horizontal lines, vertical

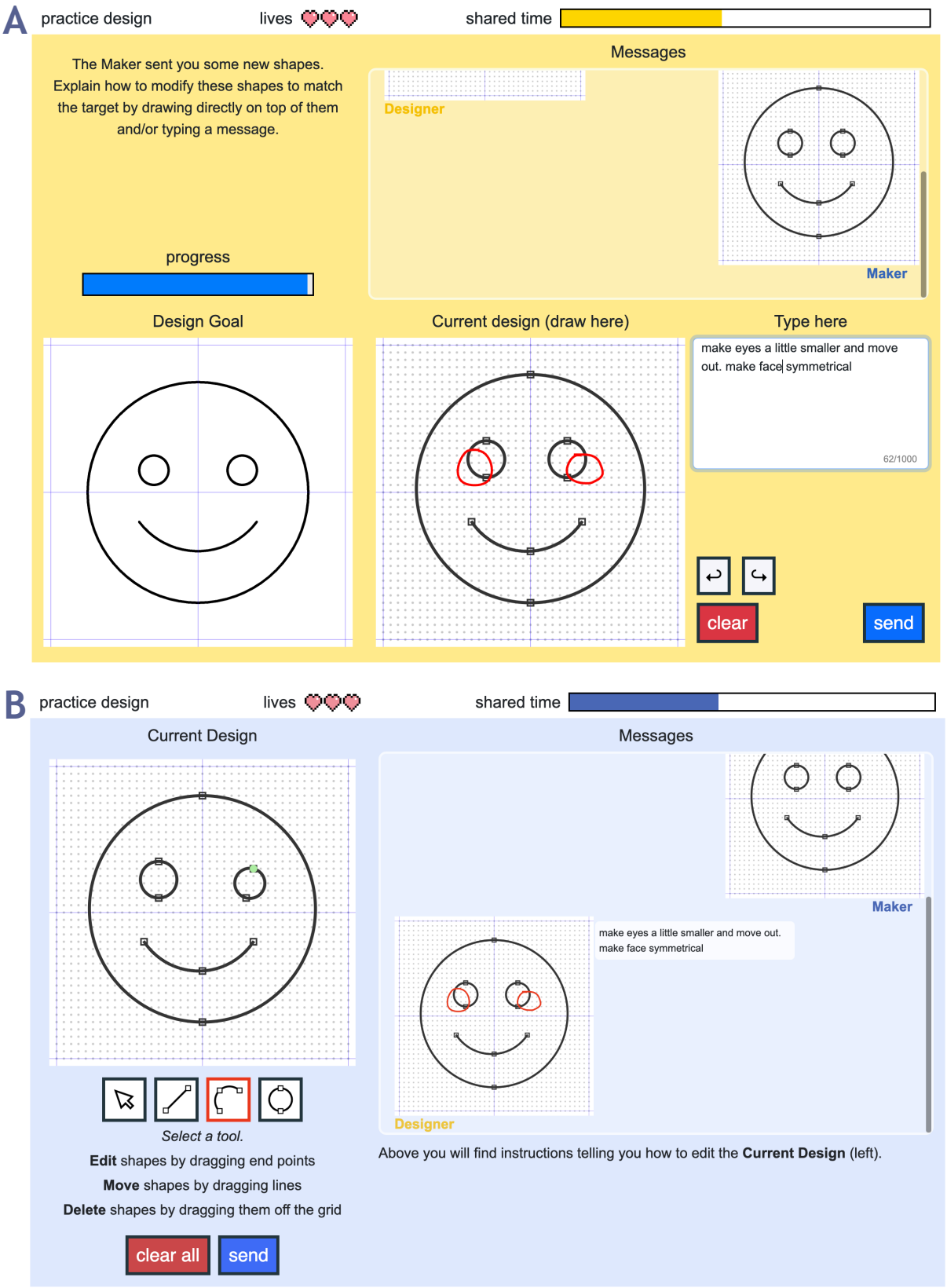


Figure 8: **A** *Designer* display: target design is shown on the lower leftand instructions are created on the lower right, by drawing on the current CAD and by typing in the text box. History of prior interaction is shown in top right. **B** *Maker* display: on the *Maker*'s turn, the current CAD can be edited in CAD interface on the left, by dragging elements and control points, and by placing new elements by selecting a tool and clicking on grid squares to place control points. Instructions are shown in chat window on right. Both *Designer* and *Maker* can see shared time and number of lives remaining. This target was shown to all participants as a practice trial.

	studies	dataset
Designer turn time	60s (180s)	unlim.
Maker turn time	120s (360s)	unlim.
Trial time limit	unlim.	up to 540s
Base payment	$10	$9.10
Trial bonus	0	$1.00
Lives	N/A	3
Trials per dyad	6	1-13
Rounds per trial	3	1-10
Character limit	200 (600s)	unlim.
Drawing limit	unlim.	unlim.

Table 2: Parameters for ablation studies (generation-only in parentheses) and primary dataset collection. In experiments, we manipulated access to modalities and refinement rounds (across participants). For dataset collection, participants had up to 540 seconds per trial, but time elapsed at double speed for the *Designer* to discourage precisely drawn copies of the target.

lines, skewed lines, arcs, and circles). By selecting one design per bucket, we: (1) eliminated most near-duplicates (2) select a more diverse set of designs. Participants were given 12 tasks randomly sampled from the task pool and shown in order of increasing difficulty (increasing number of curves in the design).

C.5 Prioritizing motivated annotators

The mrCAD task is challenging. Successful performance requires a person to (1) learn to operate a new CAD interface and (2) be adept at communicating CAD modifications, all within the span of one online session. Furthermore, in a two player game, *both* players must be motivated and capable for a successful reconstruction. How can we prioritize capable dyads to provide high-quality annotations, while fairly compensating capable individuals with an unmotivated partner? We devised an incentive scheme as follows.

Prioritizing capable individuals Before participants started performing CAD tasks, they had to pass through three checks: a comprehension quiz, a solo CAD reconstruction task, and a paired practice trial. In the solo CAD reconstruction task, each participant has to recreate a simple CAD consisting of a line, circle, and arc, to our fixed threshold level of accuracy. All participants who passed the solo reconstruction task are deemed capable and motivated, and received a base pay of ($8.70).

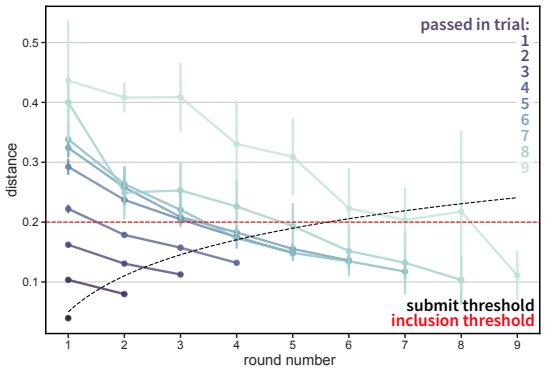


Figure 9: We implemented a dynamic threshold for submitting designs that became more lenient in later rounds. Participants took a variable number of rounds to reach the threshold. Visualizing distance to the target broken down by round submitted reveals a trend of refinement over time. Red dashed line indicates the fixed threshold for including in analysis. A. Dyads successfully refined CADs over multiple rounds, and could submit their CAD once they reached the threshold, which became more lenient as rounds went on. Only CADs that meet our fixed threshold are included in the full mrCAD dataset.

Prioritizing capable dyads Once paired, dyads had to successfully recreate one practice trial (a simple smiley face) in order to attempt other CADs. We awarded a $1 bonus per person for every recreation that met the accuracy threshold. If a dyad failed to meet the threshold in 9 minutes, they lost a "life", and were ejected from the study when they lost 3 lives. This limited the number of CADs that unmotivated dyads could attempt, directing study time and compensation towards higher quality reconstructions.

C.6 Prioritizing refinement instructions

Our primary goal was to collect a large number of *multimodal refinement instructions* that led to high-quality reconstructions. A simple way of ensuring high-quality reconstructions it to implement a fixed threshold of accuracy for submitting CADs. However, this fixed threshold does not take into account that some CAD would be easier to reconstruct while others would be more challenging. In a pilot study, we found that with a lenient threshold, unmotivated dyads submitted low quality reconstructions *without refinements*, while with a stringent threshold, motivated dyads were not compensated for decent quality reconstructions on more complex CADs.

Therefore, we implemented a **dynamic submission threshold**, which allowed only highly accu-

rate reconstructions to be submitted in early rounds, but was more permissive in later rounds. Implementing this change allowed us to collect 1,532 additional refinement rounds, an increase of 10.1% compared to using a static threshold. Furthermore, a dynamic threshold kept motivated dyads who had been "stumped" by a difficult CAD in the study longer by not losing a life. The effect of dynamic threshold can be seen in Figure 9.

C.7 Exclusion criteria

Rollouts that contained no CAD actions, entirely empty messages, or had missing rounds were excluded from the dataset for analysis. We also imposed a fixed **inclusion threshold** (Figure 4A) on reconstruction accuracy for analysis. In the statistics below, we report only data that meets these criteria, however our full dataset release also includes failures and practice trials.

D Vision-language model experiments

D.1 Hardware and software details

Evaluation We queried API models using OpenRouter. For Qwen models, we hosted the model on a local server 1 Nvidia A6000, 6000Ada, L40, or L40S GPU (each of which has 48GB VRAM) using vLLM (Kwon et al., 2023). For each instruction in context, we sampled one response from the model. For locally hosted models, each run (through the entire evaluation set with 2324 rollouts) took approximately 19 GPU hours. Evaluation hyperparameters are shown in Table 3.

Training We implemented finetuning using the supervised finetuning functionality provided by the TRL library (von Werra et al., 2020). Each finetuning run (with a max of 10 epochs over the 2684 training rollouts) took 32–48 GPU hours. We used 4 Nvidia L40 or L40S GPUs with 48GB of VRAM for each run. Training hyperparameters are in Table 4.

D.2 Prompt

Figure 10 shows the structure of the prompts used in experiments.

E Analyzing the number of actions

We studied how agents edit the design by counting the editing actions each agent took. We found (in Table 5) that in the ablated context (when only the most recent instruction in the sequence of refinement instructions was presented), models took fewer editing actions. By performing fewer bad actions, models are able to achieve better scores. However, the negative value highlights that models don't take a sufficient number of good actions to improve the design. We would also like to note the effect of finetuning here. While more actions results in worse performance for off-the-shelf models, the finetuned model is able to take more actions while also improving performance. While the overall performance is still poor, this change highlights the potential for approaches that better leverage the training data to improve multi-turn, multimodal refinement abilities of models in this setting.

You are an expert CAD software user playing a game called mrCAD. In this game, there is a designer and a maker. The two players work together to iteratively create a design over a sequence of turns. You will play the role of the maker in this game, and the user will play the role of the designer. In each turn the designer provides an instruction about how to modify the design on the canvas. The instruction may include language instructions, drawings on the canvas, or both. The drawings appear as red strokes on the canvas. The design appears in black strokes on the canvas. Your goal is to follow the designer's instructions. You have to take actions to edit the current state of the design. Each action is taken by calling a tool that performs the action. Each control point is a pair of floating point numbers between -20 and 20 that represent the coordinates of the point on the canvas.

New game:

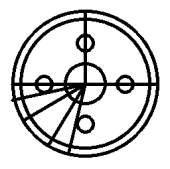
Round 1.

big round circle with a smaller circle inside it sort of resembling a clock

Edit the design based on the designer's instructions using the provided tools. Make sure to follow the instructions carefully.

[{"name": "make_curve", "arguments": {"type": "circle", "control_points": [[0.0, -18.0], [0.0, 18.0]]}}, {"name": "make_curve", "arguments": {"type": "circle", "control_points": [[0.0, -15.0], [0.0, 15.0]]}}, {"name": "move_point", "arguments": {"point": [0.0, -15.0], "new_point": [0.0, -16.0]}}, ...]

The resulting design is:



{"curves": [{"type":"circle","control_points":[[0.0,-18.0],[0.0,18.0]]}, {"type": "circle", "control_points":[[0.0,-16.0],[0.0,16.0]]}, {"type": "circle", "control_points": [[0.0,-12.0],[0.0,-8.0]]}, ...]}

Figure 10: Structure of the prompt used for API calls as well as SFT models. The prompt includes system messages, templated parts of the instruction presented as user messages, *Designer* instructions, *Maker* responses as assistant messages, and environment feedback presented as user messages. For Qwen models, a description of the tools is presented as part of the system message (not shown here). For other models, the tool descriptions are integrated by the server.

	Qwen-7B(-Instruct and -FT)	others
temperature	0.7	1
top-p	0.95	1

Table 3: Prompting hyperparameters

	value
LoRA parameters	all linear layers
LoRA rank	128
LoRA α	32
learning rate	10^{-4}
warmup steps	10
batch size	1 / GPU $\times$ 4 GPUs = 4
gradient accumulation steps	4

Table 4: Fine-tuning hyperparameters for all runs. We used

Model	# generation actions	# refinement actions
Human	17.59	8.31
GPT-4o	4.70	4.13
−context	–	2.63
GPT-4o-mini	3.90	5.35
−context	–	3.21
Claude-3.7	2.02	4.29
Qwen2.5-7B	1.39	1.11
Qwen2.5-7B-FT	7.32	6.40

Table 5: Average number of actions taken by models in generation and refinement stages