Automatic Generation of technical Drawings using Model Context Protocol

Leonhard Kunz[1]*, Mario Klostermeier[1], Kokulan Thanabalan[2], Tatjana Legler[1], Martin Ruskowski[1]

[1]University of Kaiserslautern-Landau (RPTU), Gottlieb-Daimler-Straße 42, 67663 Kaiserslautern, Germany
[2]Trier University, Campus II, Behringstraße 21, 54296 Trier, Germany
*Corresponding author. E-Mail address: leonhard.kunz@rptu.de

Abstract: With the current omnipresence of Large Language Models (LLM) there is a high interest in how to apply them also in Computer Aided Design (CAD). This paper presents an agent-based approach for the automated generation of technical drawings from existing 3D CAD models using direct integration via the Model Context Protocol (MCP). Instead of relying on intermediary representations or scripting languages, the proposed system connects a multimodal large language model directly to the native CAD environment via an integrated MCP server. The approach supports iterative refinement through visual feedback and native geometric constraints, allowing the agent to select, place, and revise drawing views in a closed loop. A modular FreeCAD addon implementing this concept is introduced and released as a testbed for agentic CAD research. The system is evaluated by comparing automatically generated drawings with human-created references using defined metrics for view selection and layout quality. Experiments with two multimodal language models demonstrate that LLM-based agentic drawing generation still has difficulties matching human drafting decisions, due to current limitations in visual understanding. The results highlight the potential of direct CAD–agent integration and motivate further specialization of smaller models for technical drawing tasks.

Keywords:
Large Language Model, Agent, Compter Aided Design, Model Context Protocol

1 Introduction

Although 3D modelling and Model-Based Definition is widely adopted in Computer Aided Design (CAD), technical drawings remain an important medium for communication between product design and manufacturing [1]. Creating them, however, requires a significant manual effort [2]. They must be both precise and at the same time comprehensive for humans, making a full rule-based, algorithmic automation of the process of deriving them difficult [3]. Machine learning models, however, have shown to be capable of well approximating human preferences in decision-making [4], and specialized models are becoming a common tool to automate drawing generation. Particularly, the recent advances in Large Language Models (LLMs) have sparked the idea to automate complex interactions with script-based CAD software [5]. Multimodal models, integrating additional modalities like images or audio, are driving the discussion around agentic Artificial Intelligence (AI) which is also expanding to the domain of product development [6]. This discussion is also driven by the introduction of standards like the Model Context Protocol (MCP) for the interaction of agentic AI with external applications for grounding or action [7].

This paper contributes here to the idea of agentic CAD via direct interaction between the agent and the CAD API. The approach uses no intermediary representation, that needs a resolver for the CAD kernel but interacts directly via an MCP server with the functions of FreeCADs TechDraw CAD API [8]. The developed FreeCAD addon is published alongside this paper and can serve further as a testbed and

modularly expanded for testing and the development of agentic AI in CAD[1]. It can also be used to generate synthetic technical drawings at scale from preexisting 3D CAD models. A solution to a notable shortage in public datasets [9].

The presented addon focusses here particularly on the automated generation of technical drawings from preexisting 3D models and investigates few key decisions during this process. The results are compared to human decisions in the same situation to evaluate, if the model's decisions are comparable to human preferences. In a second step, with the addon we test also ability to reflect on the decisions made and iteratively improve the generated technical drawing. This can be summarized in the following research questions:

1. Can the quality of technical drawings, created with a multimodal general purpose AI agent and an MCP CAD pipeline, match the requirements and quality of manually created drawings?
2. Can the model self-improve the result of the automatic generation pipeline through the iterative execution of commands in the CAD environment and the evaluations of the results?

To answer the questions, the paper is structured as follows: Section 2 reviews related work. Section 3 describes the implementation of the addon for FreeCAD. Section 4 defines tests for the framework and section 5 presents and discusses the results of these tests involving 2 selected LLMs. Section 6 discloses the limits of the study and section 7 concludes the paper.

2 Related Work

Technical drawings are in many aspects based on common standards (e.g. [10]) which unify the representation of tolerances like the Geometric Dimensioning and Tolerancing (GD&T) symbols or the indication of dimensions. However, these standards cannot provide complete guidelines for the context sensitive placement of the symbols within a drawing and along the geometry [11]. Additionally, they are also often complemented by free text annotations describing e.g., manufacturing or inspection instructions [12]. Modern CAD environments usually already provide at least semi-automated solutions to the derivation of drawings [13]. These are partially still constrained to predefined parameters, e.g. a fixed selection of views or a context-insensitive dimensioning [13]. Some developers of CAD systems, however, have already introduced features for context-sensitive generation of complete drawings [14, 15]. Third party plugins provide an alternative for these built in options [16]. Proprietary AI tools in CAD are confined to vendor-specific ecosystems and typically offer isolated functions like stand-alone drawing generation. Proprietary formats and restricted interoperability constitute core structural barriers, as shown by Berger et al. [17].

Neutral agent-based design frameworks instead coordinate the broader design workflow, integrate heterogeneous inputs and iteratively contextualize model and drawing creation [18]. This enables a more faithful capture of design intent and supports end-to-end automation. However, there is some discussion around the integration interface between the agent and the CAD environment. While Xu et al. try to directly generate a valid boundary representation CAD model, these approaches naturally have limitations regarding guarantees for the integrity of the generated model [19]. Therefore, more commonly, approaches try to integrate into CAD environments and leverage the rigid modeling constraints and export validity guarantees [20].

Many approaches rely here additionally on an abstract or symbolic representation (e.g., a domain-specific language, command sequence, or plan) which serve as abstract representations of the

[1] https://github.com/XDP-Opt/TechDraw-MCP-Agent

planned actions. These are then resolved into concrete actions using the CAD Kernel. Daareyni et al. propose e.g. a solution generating OpenSCAD scripts, that can then be rendered for visual confirmation [21]. Other authors are trying to define their own domain specific language or parametric scripting language [22, 23]. This serves as an intermediate representation between a task description and the CAD modeling environment. The implementation itself can be considered open-loop, where the geometry is generated in a single shot [19, 23] or closed-loop, enabling revision and improvement steps [21, 22]. The introduction of an intermediary representation of geometry, however, complicates the refinements steps as the intermediary representation forces refinement to operate across two layers. The system must first inspect the CAD environment to identify defects or misaligned features. It must then project those findings back into the abstract representation and modify that representation without direct geometric feedback. This multi-layer loop adds fragility and limits corrective precision. In contrast, approaches that bind the agent directly to the CAD environment's API allow direct refinements to operate on native geometric and topological structures. While conceptualizing requires a certain abstract representation for planning the incremental modeling steps, the derivation of a technical drawing is already constrained by the existing 3D model. The challenge is to design an agent-based CAD architecture that remains closely integrated with the CAD environment while supporting efficient concept development [24].

3 Methodology

The approach presented in this paper ties the agent directly into the CAD environment via a custom MCP interface. This interface wraps primarily select functions of FreeCADs TechDraw workbench into simple tool functions. In our approach, we focus particularly on the implementation of these tools as it is an often-overlooked part of product design in literature on AI agents. Yet it is often still crucial for handover between design and manufacturing [24]. It also requires visual revision to confirm human comprehensibility and is therefore a prime showcase for the advantages the multimodal data processing capabilities of multimodal transformer models at the core of the agent.

Additional to the CAD API, the agent has access to local file storage where we can externalize memory capabilities or store abstract concept representations. The agent is encouraged via its system prompt to write checklists and notes to this storage and reload them for subsequent runs. Additionally, files can be manually stored in the storage to give the agent read-only access to ground the agent with preformulated knowledge. With these capabilities, the agent's architecture, as depicted in Fig. 1, represents a very simple form of the intelligent CAD 2.0 concept postulated by Zou et al. [25].

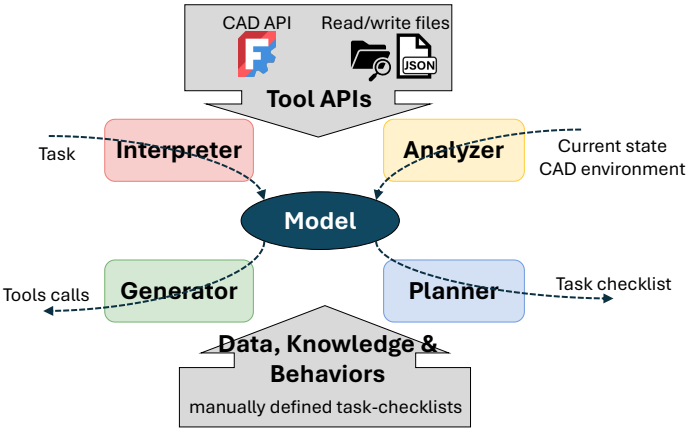

Figure 1: Architecture of the agent with its different modes analogous to Zou et al. [25].

The approach is modular and can be expanded by different MCP servers offering other CAD toolsets or other external APIs, e.g. to simulation tools, data storage, or management software. The agent can switch

between different modes (Interpreter, Analyzer, Generator, Planner) dynamically and solve tasks theoretically without a predefined processing sequence. To streamline efficiency however, predefined behaviors and approaches can be stored in the form of checklists written in natural language and serialized in markdown files. The foundational behavior of the agent can additionally be steered by a manually crafted system prompt. Fig. 2 exemplifies the implementation of such steered workflow with shortened prompts for better comprehensibility. The described behavior is non-linear and contains constant reassessment and revision of the current state in the CAD environment in reference to the given task. Therefore, it enables an iterative improvement towards fulfilling the task.

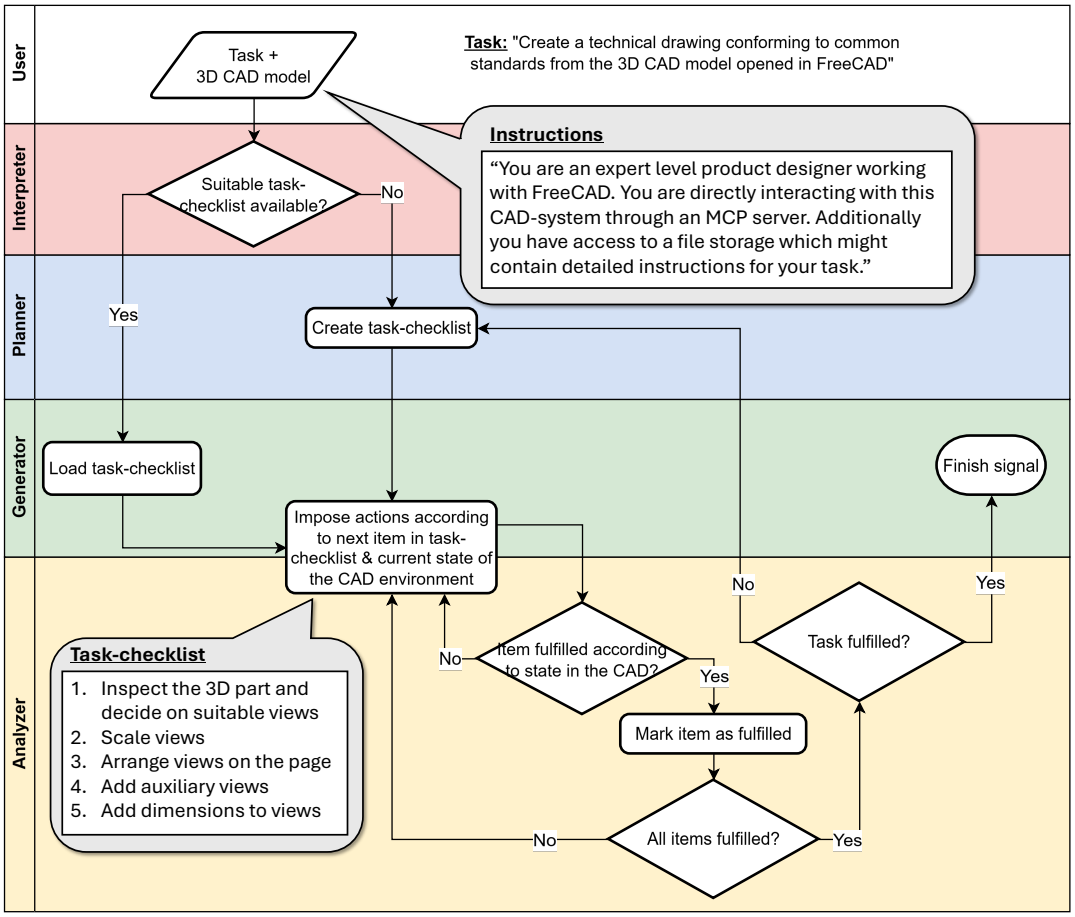

Figure 2: The general agent behavior and how it can be steered using exemplified and shortened, predefined checklists and instructions.

This behavior for the agent always forces either to write or update a file or implement a change directly within the CAD environment. This allows a close integration of the agent in the CAD system without any intermediate representations of the technical drawing or the 3D model profiting of hard constraints imposed by the environment. In contrast to approaches based on textual representations, the model relies more on the visual inputs from the CAD environments to assess the current state and generate further actions. This, however, requires the construction of support views to assess different options regarding key decisions. Thus, our approach discretizes the possible placement of views on the drawing page with an auxiliary grid. The center points of the different views in the drawing can then be directly placed on the discrete options on the grid by the agent. Fig. 3a visualizes this on an example. For the construction of auxiliary views, like section and detail views, and the placement of dimensions, another support view of the drawing page is used. This view enumerates the vertices in every view in the drawing enabling direct callouts, like creating a dimension between two vertices or sections through certain vertices. Fig. 3b shows an example of this support view.

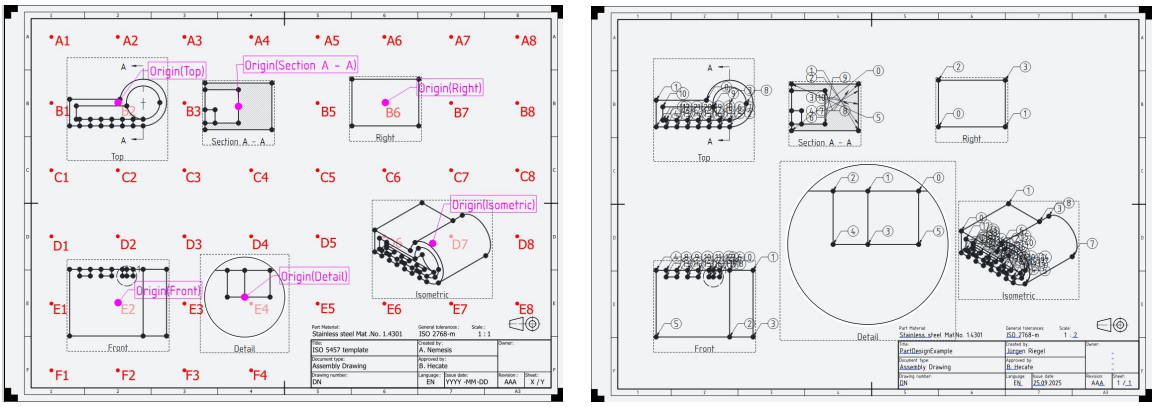

Figure 3: Support views to a) define the placement the views on the drawing page and b) define dimensions based on vertex points of the projected geometry.

4 Experiment

To assess the functionality of the agent tie-in, and the quality of the resulting drawings, we compare them to drawings manually derived from the same 3D models. Our assumption here is that a human drafter considers aspects like the readability and comprehensibility of a drawing that are difficult to describe with algorithmic metrics. This becomes evident in the choice of excess views or driven dimensions. These increase comprehensibility for the reader of the drawing but add no necessary new information to define the part. We extract therefore certain characteristics from manually drafted drawings and compare these to automatically generated ones. We define the following dimensionless metrics:

- **View Coverage (VC):** This metric is a ratio expressing how many of the human-selected views or equivalent ones are covered by the generated drawing. A higher value expresses a closer fit to the humans' decisions.
- **Excess View Rate (EVR):** The ratio of excess views compared to the human-drafted drawing, relative to the number of views in the generated drawing. A lower value expresses a closer fit to the humans' decisions.
- **Duplicate View Rate (DVR):** Number of equivalent and therefore duplicate views, relative to the number of views in the generated drawing. A lower value expresses less redundant information in the drawing.
- **View Placement Error Rate (VPER):** The number of overlaps between two views and views that exceed the boundaries of the free drawing area relative to the number of views in the generated drawing. A lower value means less errors in the drawing.
- **Page Coverage (PC):** This metric is a ratio describing how much of the drawing page, excluding the frame and the title area, is covered by the views. A higher value indicates that the drawing page was well used.

In the workflow of generating a valid technical drawing from a given 3D CAD model, we distinguish between several key decisions. For the experiment, we limit the decisions only to the selection and the arrangement of views, excluding dimensioning, tolerancing and additional textual remarks. These decisions can be made in an arbitrary sequence, except the initial creation of the drawing page with basic views. In a first set of experiments, the agent is tasked to make the decisions independently of each other. The agent is firstly tasked to initialize the drawing with an initial set of views. For every further decision, the state of the drawing is set back to the same initial point for all tested models. Afterwards,

the agent is tasked with defining a drawing using all available tools and making all decisions without separating one particular one out.

Tab. 1 shows the observed decisions made by the drafter in the manually created drawings. These decisions influence different metrics, e.g. the decision on the scaling of a view does not influence the VC or the EVR, but potentially the VPER and the PC. On the other hand, while the decision to add a view has a primary influence on VC, EVR and DVR, the VPER is not the focus of the observation, and constrained due to the norms of orthogonal projections. In Tab. 1, we therefore assort the relevant metrics to the individual decisions. Experiment 5 is separated, as the special case, where the agent has here access to all tools and is not tasked to make an individual decision, but define a drawing completely.

Table 1: Decisions made by a human drafter during the creation of a technical drawing and the assorted metrics to describe the difference between the manually created and the generated drawing.

Experiment ID	Decision	Assorted metrics
1	Which views should be included in the drawing?	VC, EVR, DVR
2	Which scale should be used for each view?	VPER, PC
3	Where should the views be placed on the drawing sheet?	VPER, PC
4	Which further views should be added or removed?	VC, EVR, DVR
5	Create a technical drawing using all available tools.	VC, EVR, DVR, VPER, PC

For the experiments described we included 2 different multimodal LLMs as core of the agent. ChatGPT-5.2-2025-12-11 [26] represents commercial flagship models, available remotely through an API. We also tested the smaller Qwen3-VL-30B-A3B-Instruct [27] as a locally hosted model, that runs on a Nvidia RTX 4090 GPU.

The 3D models used for the experiment were taken from the TechMB Visual Question-Answering benchmark [9], which contains manually created drawings of 180 distinct objects from the Autodesk Fusion 360 segmentation dataset [28]. The original step files from the Autodesk dataset were taken and compared to the characteristics of the corresponding TechMB drawings.

5 Results and Discussion

Tab. 2 shows the results for both models in all experiments. For all evaluation metrics, the mean value over the 180 drawings was calculated. As discussed before, VC, EVR and DVR do not change when moving views or changing scale in experiments 2 and 3 from the initial experiment 1. Due to the discussed reset between the experiments 2, 3 and 4, the values are equal between both models.

The results show that both models already have a relatively high initial VC with a low EVR and DVR. This can be interpreted that similar decisions as the human drafter regarding the initial view selection were made. However, with an EVR of 0.314 for ChatGPT5.2 and 0.288 for Qwen3 a significant part of views were overestimated. This indicates that both models were not quite as selective in their choice of views as the human drafters were. At the same time, the DVR remains constantly low. This suggests that even though a significant portion of the objects are rotational or reflectional symmetric, redundant views were chosen only seldomly.

In overall, the models performed comparatively equal, except the more than 10 times higher VPER of Qwen3 in experiment 3, than ChatGPT5.2. Here, the agent powered by Qwen3 often moved multiple views to the same place in the drawing page, so they overlapped fully. In another frequent mistake, the agent moved views completely out of the drawing page.

Table 2: Mean results of the 4 distinct decisions for the whole dataset in the 5 metrics. The arrows indicate if a higher or a lower value is prefarable.

Experiment ID	Model	VC ↑	EVR ↓	DVR ↓	VPER ↓	PC ↑
1	ChatGPT5	0.818	0.314	0.000	0.241	0.058
	Qwen3-VL-30b	0.791	0.288	0.058	0.151	0.060
2	ChatGPT5	0.818	0.314	0.013	0.127	0.052
	Qwen3-VL-30b	0.818	0.314	0.013	0.027	0.021
3	ChatGPT5	0.818	0.314	0.013	0.064	0.065
	Qwen3-VL-30b	0.818	0.314	0.013	0.695	0.042
4	ChatGPT5	0.848	0.305	0.081	0.181	0.068
	Qwen3-VL-30b	0.864	0.302	0.089	0.220	0.069
5	ChatGPT5	0.925	0.422	0.071	0.058	0.048
	Qwen3-VL-30b	0.935	0.332	0.153	0.289	0.028

While the evaluation of the mean values allows an overall assessment of the model choices, it is important to also evaluate the statistical bandwidth of the created drawings. Fig. 4 helps to display the distribution of the metrics over the whole set of CAD models included in the experiments. It also allows a direct comparison of the changes between experiments 1 and 4. Here we see that even though both models performed roughly equal in median, the range of results is quite different. The distribution of the ChatGPT5.2 results are more concise than the ones generated by Qwen3. This means that the resulting drawings have a more persistent quality over the whole dataset. ChatGPT5.2 showed also an overall greater improvement from the initial state at the end of experiment 1 to the following experiments.

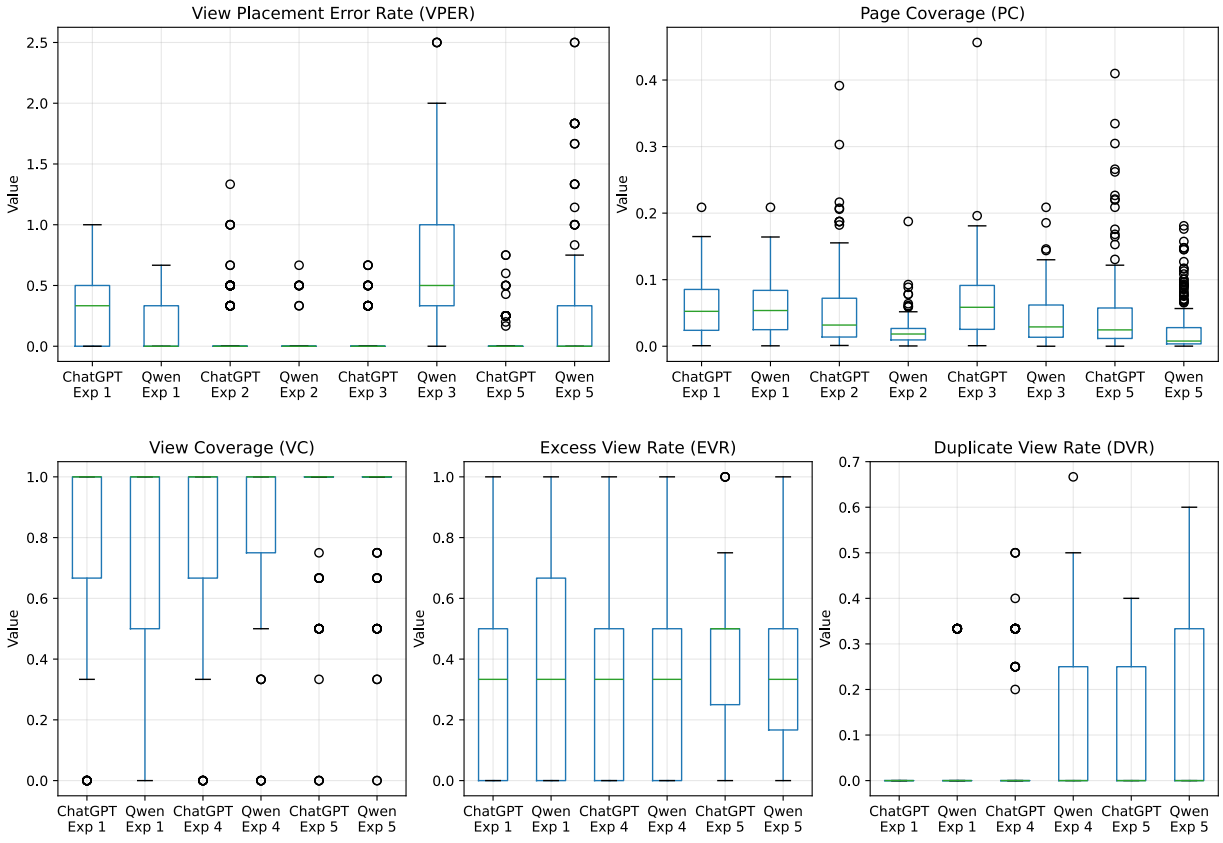

Figure 4: Distribution of the results in defined all metrics. A selection regarding the relevance of different metrics for different experiments was made according to tab. 1.

Of special interest are the results of experiment 5 in which the agent had to define a whole drawing with access to all different tools of the previous experiments. Both models manage to get a high VC at the price of a comparatively high EVR and DVR. This means that both models ended up adding more and more views to the drawing while at the same time scaling down the drawings to avoid placement errors. This led to a comparatively low PC and sometimes therefore to a low visibility of the details in a single view. ChatGPT5.2 performed in overall again better than Qwen3 and both are comparable to the results of the previous experiments. This allows for the conclusion that a stack of subsequent mistakes is avoided even without breaking down the task into single decisions. It must be hereby added that the agent needed substantially longer for each drawing as it made more iterative changes to the drawing. In contrast to the isolated experiment 1 and 4, both models in experiment 5 rarely made use of the option to make screenshots of the 3D CAD model from different viewpoints. Instead, the agent chose to initialize the drawing with a set of blindly estimated views, which were the extended and altered iteratively. The differences, especially between experiments 1 and 5, suggest that the processing of screenshots of the 3D object may lead to a more careful selection of views. It also raises the question whether the models are trained better to process these screenshots, than the technical drawings themselves. The results furthermore make it clear that even current flagship models need further finetuning regarding the understanding and interpretation of technical drawings. These visual abilities are core to the agent concept presented here in this paper. It also becomes clear that the ChatGPT5.2 flagship model, with a more than 10 times larger parameter size, did not yield a substantial performance increase of the agent compared to the locally hosted Qwen3 model. This shows the diminishing returns of large scale LLMs for specialized domains and encourages here the focus on smaller models for customization which is archivable with access to less resources.

6 Limitations

The study of this paper is subject to limitations regarding the dataset, the evaluation methods and the implementation of the specific agent. While the TechMB dataset originally picked the objects at random from the foundational Autodesk segmentation dataset, the high share of relatively simple and symmetric parts is apparent. This directly biases the number and selection of views used in the TechMB drawings to fully depict the parts, and therefore the difficulty of the decisions regarding view selection. However, our results show that the error quota for both models is still quite high. Additionally, the defined metrics have only limited expressivity regarding the comprehensibility and quality of the drawing. While VPER and VC correlate with the decision quality, a low EVR or DVR is only an indicator for the quality as additional views might help to distribute dimensions and increase drawing comprehensibility. The values are therefore helpful to observe the impact of agent choices but not necessarily drawing quality. Lastly, the implementation of an MCP server as well as the use of LLMs requires the design of function descriptions in docstrings as well as instructive prompts. These can have large influence on the choices of the agent and its performance. However, the systematic design and improvement of these texts is difficult and out of scope for this work. All prompts and docstrings used to create the results are contained in the source code which is published alongside this paper[2].

7 Conclusion

In this paper, we analyzed the capabilities of agentic CAD systems in the light of the current breakthrough developments in LLMs. We discussed different approaches for the integration of LLMs for CAD generation and particularly the derivation of technical drawings. Then we presented an agent based architecture, that directly integrates the LLM powered agent into the CAD environment via an

[2] https://github.com/XDP-Opt/TechDraw-MCP-Agent

MCP interface. This is opposed to the common, indirect approach to using an intermediary script that defines the CAD model and the technical drawing. In an experiment, we tested the capabilities of two differently sized LLMs as core of the agent to define the foundational characteristics of a technical drawing. Our results show that not even a current flagship model possesses the vision capabilities necessary to handle this task. However, they also show that the customization of smaller, more accessible models could be an effective way to fill this gap.

This finetuning of small LLMs and the generation and provision of the necessary training data remains part of future work. The provided agent based framework and the described experiments can help hereby to benchmark the results of this finetuning. The MCP-based approach can also further be used to evaluate the sequence of decisions made by the model to change things in the CAD environment. This could be a central part of understanding the planning capabilities of LLMs with regards to the fulfillment of domain specific tasks and therefore the composition of further training data.

Acknowledgements

This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 543073350.

References

[1] Ruemler, S. P., Zimmerman, K. E., Hartman, N. W., Hedberg, T., & Barnard Feeny, A. (2017). Promoting Model-Based Definition to Establish a Complete Product Definition. *Journal of Manufacturing Science and Engineering*, 139(5), 051008. https://doi.org/10.1115/1.4034625

[2] Dillenhöfer, F., Doksanbir, A., Kraske, J. L., & Künne, B. (2023). Automatic Generation of Training Data for AI Object Detection in Terms of Technical Drawings in Engineering. In M. E. Auer, R. Langmann, & T. Tsiatsos (Eds.), *Open Science in Engineering* (Vol. 763, pp. 643–651). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-42467-0_60

[3] Sofias, K., Kanetaki, Z., Stergiou, C., Kantaros, A., Jacques, S., & Ganetsos, T. (2025). Implementing CAD API Automated Processes in Engineering Design: A Case Study Approach. *Applied Sciences*, 15(14), 7692. https://doi.org/10.3390/app15147692

[4] Goli, A., & Singh, A. (2024). Frontiers: Can Large Language Models Capture Human Preferences? *Marketing Science, 43*(4), 709–722. https://doi.org/10.1287/mksc.2023.0306

[5] Alam, M. F., Lentsch, A., Yu, N., Barmack, S., Kim, S., Acemoglu, D., Hart, J., Johnson, S., & Ahmed, F. (2024). From Automation to Augmentation: Redefining Engineering Design and Manufacturing in the Age of NextGen-AI. *An MIT Exploration of Generative AI*. https://doi.org/10.21428/e4baedd9.e39b392d

[6] Song, B., Zhou, R., & Ahmed, F. (2024). Multi-modal machine learning in engineering design: A review and future directions. *Journal of Computing and Information Science in Engineering*, *24*(1), 010801.

[7] Hou, X., Zhao, Y., Wang, S., & Wang, H. (2025). Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions. *arXiv*. https://doi.org/10.48550/ARXIV.2503.23278

[8] Riegel, J., Mayer, W., & van Havre, Y. (2016). FreeCAD. Freecadspec2002.

[9] Kunz, L., Klostermeier, M., Thanabalan, K., Legler, T., & Ruskowski, M. (2025). TechMB: Exploring the Potential of Vision Language Models for Interpreting Technical Drawings. *DS 140: Proceedings of the 36th Symposium Design for X (DFX2025)*, 179–188. https://doi.org/10.35199/dfx2025.19

[10] Deutsches Institut für Normung (DIN). (2021). Technical product documentation - Digital product definition data practices (ISO No. 16792; Versions 2021-04).

[11] Yan, Y., & Bohn, M. (2017). Limitations of notation system in centred part alignment accuracy imposed by ISO standard and proposal for an improved methodology. *Journal of Machine Engineering*, 17(2), 94–101

[12] McMahon, C. A., & Davies, D. (2006). The use of annotation in design representation: 5th International Seminar and Workshop of Engineering Design Integrated Produc5th International Seminar and Workshop of Engineering Design Integrated Product Development: Design Methods for Practice, EDIProD 2006. Design Methods for Practice - 5th International Seminar and Workshop of Engineering Design Integrated Product Development, EDIProD 2006, DS 37, 105–111.

[13] Buric, M., Brcic, M., & Skec, S. (2021). Towards Automated Drafting in CAD Systems. 2021 4th International *Conference on Electronics and Electrical Engineering Technology*, 233–238. https://doi.org/10.1145/3508297.3508335

[14] Auto Generating Drawings (BETA) - 2025 - What's New in SOLIDWORKS. (n.d.). Retrieved September 24, 2025, from https://help.solidworks.com/2025/English/WhatsNew/t_drawings_autogenerating_draw.htm

[15] Still, T. (2024, October 10). *AI in Manufacturing: Automated Drawings in Autodesk Fusion*. Retrieved: November 20, 2025, https://www.autodesk.com/products/fusion-360/blog/ai-in-manufacturing-automated-drawings-in-autodesk-fusion/

[16] CAD Drawing Automation Software | DraftAId Integrations. (n.d.). Retrieved September 24, 2025, from https://draftaid.io/integrations/

[17] Berger, E., Dammann, M. P., Mehlstäubl, J., Saske, B., Braun, F., & Paetzold-Byhain, K. (2025). Challenges and opportunities in the integration of generative AI with computer-aided design. *Proceedings of the Design Society*, *5*, 881–890. https://doi.org/10.1017/pds.2025.10102

[18] Schüpbach, A., San Miguel, R., Ferchow, J., & Meboldt, M. (2025). From text to design: a framework to leverage LLM agents for automated CAD generation. *Proceedings of the Design Society*, *5*, 1893–1902. https://doi.org/10.1017/pds.2025.10203

[19] Xu, X., Lambourne, J., Jayaraman, P., Wang, Z., Willis, K., & Furukawa, Y. (2024). BrepGen: A B-rep Generative Diffusion Model with Structured Latent Geometry. *ACM Transactions on Graphics*, *43*(4), 1–14. https://doi.org/10.1145/3658129

[20] Afzal, M., Ali, S., Khan, M., Sheikh, T., Sinha, S., & Stricker, D. (2024). Text2CAD: Generating Sequential CAD Designs from Beginner-to-Expert Level Text Prompts. *Advances in Neural Information Processing Systems 37*, 7552–7579. https://doi.org/10.52202/079017-0242

[21] Daareyni, A., Martikkala, A., Mokhtarian, H., & Ituarte, I. F. (2025). Generative AI meets CAD: enhancing engineering design to manufacturing processes with large language models. *The International Journal of Advanced Manufacturing Technology*. https://doi.org/10.1007/s00170-025-15830-2

[22] Schöfer, F., & Seibel, A. (2025). Augmented design automation: leveraging parametric designs using large language models. *Proceedings of the Design Society*, *5*, 671–680. https://doi.org/10.1017/pds.2025.10081

[23] Liao, J., Xu, J., Sun, Y., Tang, M., He, S., Liao, J., Yu, S., Li, Y., & Guan, X. (2025). Automated CAD Modeling Sequence Generation from Text Descriptions via Transformer-Based Large Language Models. *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 21720–21748. https://doi.org/10.18653/v1/2025.acl-long.1054

[24] [In Publishing] Kunz, L., Kokulan, T., Legler, T., Malburg, L., Bergweiler, S., Ruskowski, M., & Bergmann, R. (2025). XDP-Opt: Experience-Based Design Process Optimization for Industrial Manufacturing. *KI2025 - 48th German Conference on Artificial Intelligence*.

[25] Zou, Q., Wu, Y., Liu, Z., Xu, W., & Gao, S. (2024). Intelligent CAD 2.0. *Visual Informatics*, *8*(4), 1–12. https://doi.org/10.1016/j.visinf.2024.10.001

[26] OpenAI. (2025, December 11). *Entdecke GPT-5.2*. Retrieved January 24, 2026, from https://openai.com/de-DE/index/introducing-gpt-5-2/

[27] Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., … Qiu, Z. (2025). *Qwen3 Technical Report*. arXiv. https://doi.org/10.48550/ARXIV.2505.09388

[28] Lambourne, J. G., Willis, K. D., Jayaraman, P. K., Sanghi, A., Meltzer, P., & Shayani, H. (2021). Brepnet: A topological message passing system for solid models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 12773-12782).