SLING PARTNER CONTENT CONTRIBUTIONS

Name of the SLING partner: University of Ljubljana, Faculty of Mechanical Engineering, Slovenia

Luka Samsa, Nikola Vukašinović, Miha Brojan; Faculty of Mechanical Engineering, UL, LeCAD

Title of the paper SLO: Uporaba LLM-ov pri parametričnem 3D modeliranju

Title of paper ENG: Leveraging LLMs for parametric 3D modeling

---

Leveraging LLMs for Parametric 3D Modeling

Artificial Intelligence (AI) has made rapid progress in recent years, especially in the field of generative and large language models (LLMs). Its growing capabilities are making it increasingly important across various industries, including engineering. To integrate AI into mechanical engineering more effectively, we explored how and where its potential can be best utilized. This study presents a methodology that incorporates AI into computer-aided design (CAD), enabling a simple and efficient conversion of text descriptions into parametric 3D models written in the STEP format.

Theoretical Background

AI, as we know it today, is often referred to as "modern" AI. It is primarily powered by artificial neural networks (NNs) and deep learning (DL), concepts that have been known since the previous century [1]. The recent rapid development of commercial AI assistants and LLMs is largely driven by a new algorithm called Generative Pre-trained Transformer (GPT), which was first introduced in the paper *Attention Is All You Need* [2]. A schematic representation of the Transformer architecture is shown in Figure 1.

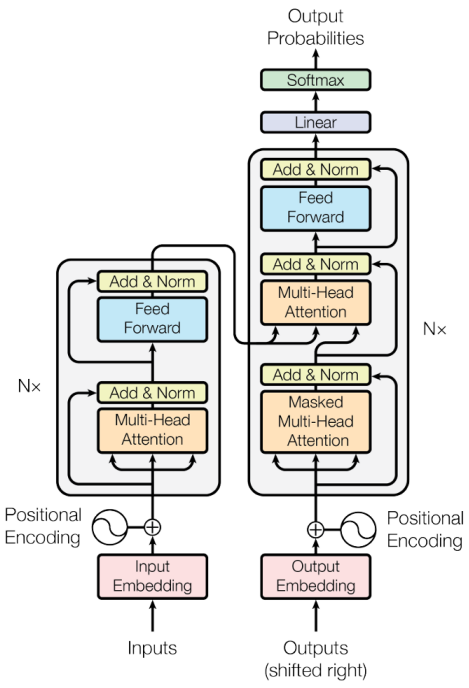


Figure 1: Schematic representation of GPT technology [2]

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.

GPT models have revolutionized AI, particularly in natural language processing (NLP), enabling functionalities that were previously unavailable. They are capable of self-supervised learning [3], meaning they can learn from unlabeled data. Text generation using GPT models works by predicting words, phrases, characters, or symbols based on the given context. The key mechanism that allows the model to consider the entire context is called "attention." This capability helps the model better understand long texts and improve its interpretation of user inputs [2]. A similar approach used for learning and generating text can also be applied to programming languages. Commercial tools like ChatGPT have quickly acquired knowledge of popular programming languages this way. We recognized this as a great foundation for further research and leveraged it for developing parametric 3D models.

Choosing the Right Tools

Several commercial tools based on LLM technology are available. The most notable among them is OpenAI, which has developed multiple powerful models and offers them to users through the ChatGPT platform. Access to these models is not only limited to the platform but is also available via an Application Programming Interface (API), making it an excellent choice for our needs. This allows us to automate a significant part of our methodology. Currently, the most powerful model accessible via API is the GPT-4o model, which we selected for our project.

Given the popularity and widespread use of Python, we decided to tailor our AI assistant to this programming language. Additionally, we enhanced the assistant's capabilities by integrating Open CASCADE (OCC), an open-source library, through its Python wrapper, pythonOCC-core. The OCC library, originally developed in C++, provides functionalities such as 3D modelling, model visualization, simulations, import/export of various 3D formats. The development of the Python wrapper was primarily driven by the fact that C++ is a low-level programming language, which makes it a bit more challenging for mechanical engineers. By utilizing pythonOCC-core, we can harness the power of OCC in a more accessible and user-friendly way.

Methodology

Our goal is to develop an AI assistant capable of generating engineering 3D models with high precision and correct mechanical design principles. To ensure accuracy, we generate models using Python scripts, which output STEP files that can be seamlessly integrated into engineering workflows. Commercial generative 3D models are mostly based on mesh representations, which lack the precision and technical consistency required for engineering applications. While LLMs like GPT-4o have basic knowledge of the pythonOCC-core library, generating complex mechanical components remains a challenging task. To improve reliability, we enhance the assistant with fine-tuning and a set of reference scripts that serve as a foundation for 3D model generation.

For this study, we focused on generating key mechanical components to ensure high-quality output. With the right approach, we can equip the assistant with all the necessary knowledge to generate various engineering shapes.

In this study, we primarily focused on shaft generation, but we aim to expand the assistant's capabilities to include:

- Springs
- Gears
- Standard mechanical elements (bolts, bearings, nuts, washers, pins, etc.)

EuroHPC Joint Undertaking

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.

To achieve high-quality results, we also enabled the assistant to perform minor operations, such as creating grooves and adding chamfers or fillets. Before establishing full API communication, we initially tested LLM-generated results directly on the ChatGPT platform, which, like the API, provides access to the GPT-4o model, but in a more user-friendly way. During testing, we discovered that for complex 3D models**,** text inputs become lengthy and that it is often difficult to specify exactly where features like chamfers should be applied. To address this, we implemented real-time visualization of results alongside API communication, allowing better control over the generation process. Additionally, we introduced a topological element selection feature, enabling users to define certain details directly on the 3D representation.

A simple example of visualization and topological element selection is shown in Figure 2, using a basic cube model**.**

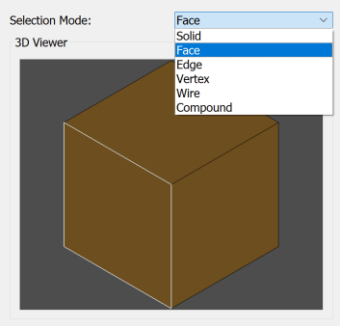
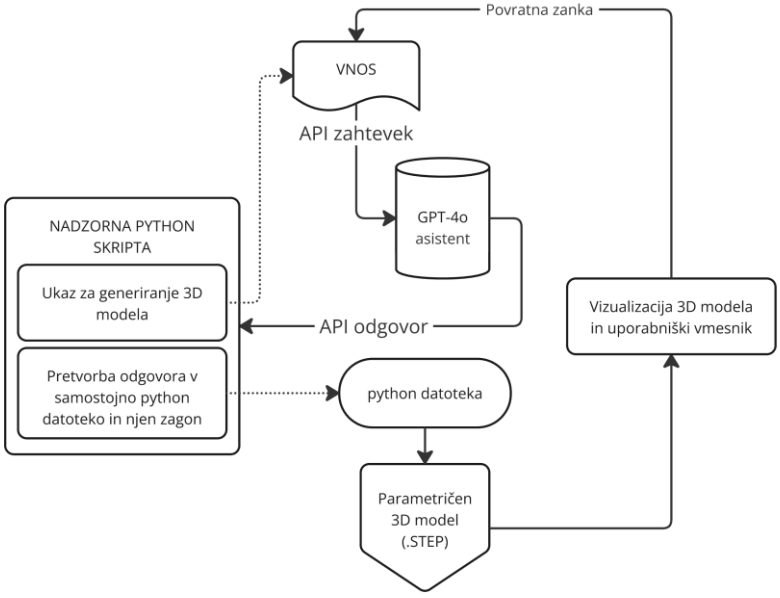


Figure 2: Selection of topological elements on a simple cube example.

With the ability to select topological elements, the user can provide more precise and detailed instructions while reducing the need for highly detailed textual input. One of the main limitations of modern AI is its lack of engineering experience and understanding of mechanical design principles, which prevents it from making fully autonomous decisions at this level. The developed interface aims to overcome this challenge. The engineer guides the AI with a clear vision of the final product, while the AI assists by accelerating the realization and visualization of ideas. The complete methodology is presented in the diagram in Figure 3.



Slika 3: Diagram representation of the Methodology.

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.

Current Research Findings

Although the research is still ongoing, the presented technology has already produced the first useful results, which we are actively improving. The results will be presented in the form of user prompts and the 3D models generated by the AI assistant. For demonstration purposes, we will use the example of a shaft. Figure 4 shows the shaft created by our AI assistant based on the following prompt:

- ***Prompt**: Create a 230 mm long shaft. First segment has a diameter of 25 mm and it is 40 mm long. The next one has a diameter of 30 mm and is 20 mm long. Then a 40 mm diameter, which is 60 mm long. After that we have a 30 mm diameter which is 20 mm long, and the last one is 20 mm in diameter, and 90 mm long.*

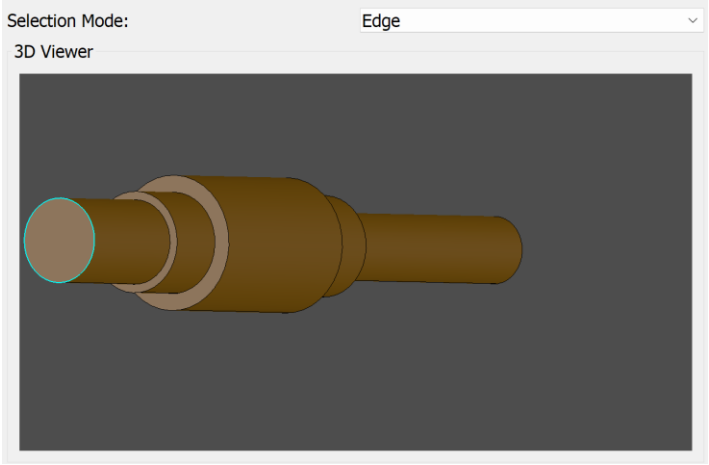


Figure 4: Result 1 – 3D model of a shaft.

Through the user interface, the user can select the appropriate topological element, such as an edge, and precisely define how operations like chamfering should be performed (Figure 5).

- ***Prompt**: On selected edges, create 1mm chamfers.*

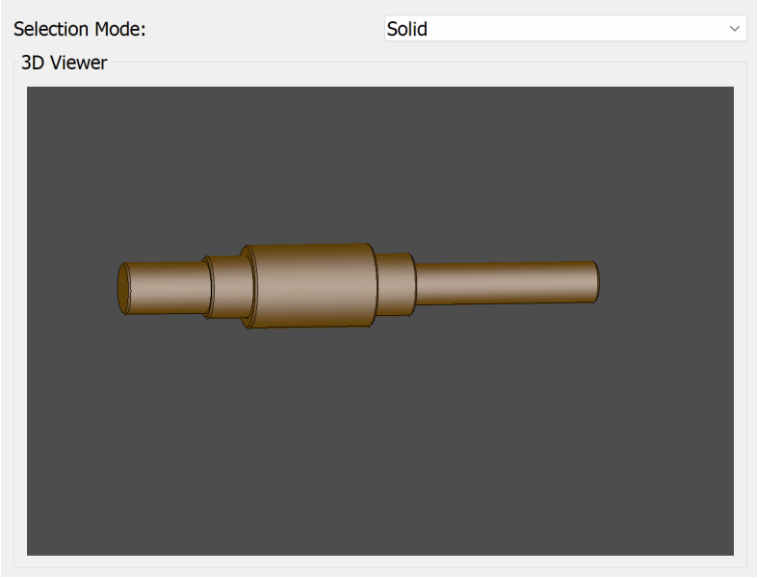


Figure 5: Result 2 - 3D model of a shaft with chamfers.

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.

We also tested the presented methodology on the generation of assemblies. In this case, we aimed to add two ball bearings to the shaft (Figure 6). The assistant did not generate the bearings itself but accessed them through a locally stored STEP file. The position of the bearings on the shaft was defined using the following prompt:

- ***Prompt**: Within the shaft script read the data/skf_bearing_16006.step file and assemble the shaft and two bearing files together. Position of the first bearing should be (0, 0, 55.5) and the position of the second one (0, 0, 124.5).*

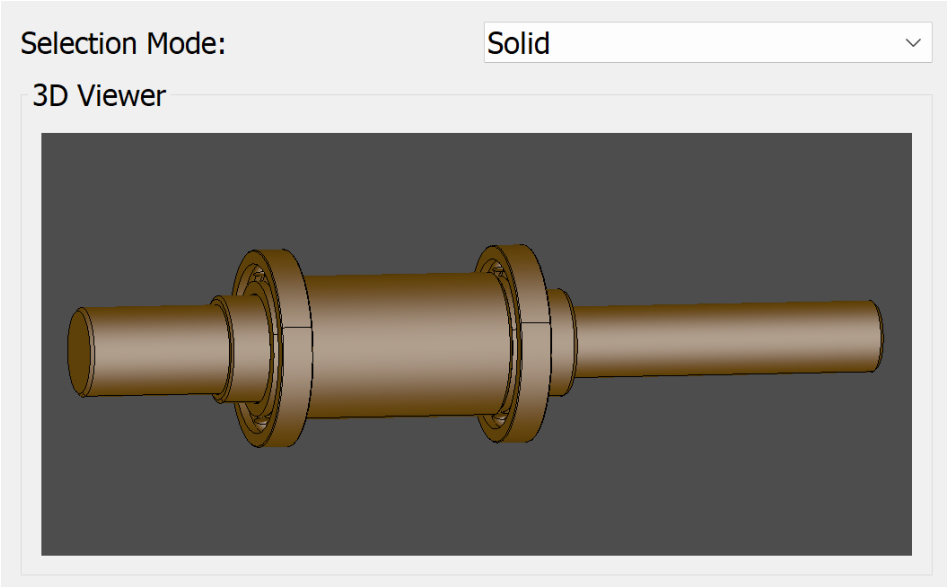


Figure 6: Result 3 - 3D model of a shaft with chamfers and ball bearings.

Figure 7 shows the result of our methodology – a shaft with chamfers and two ball bearings. The final STEP file can be visualized using our custom user interface (Figures 4, 5, and 6), but we also tested its compatibility with commercial CAD software, SolidWorks (Figure 7).

Additionally, we verified the total length of the shaft, which matches the initial command (total length: 230 mm).

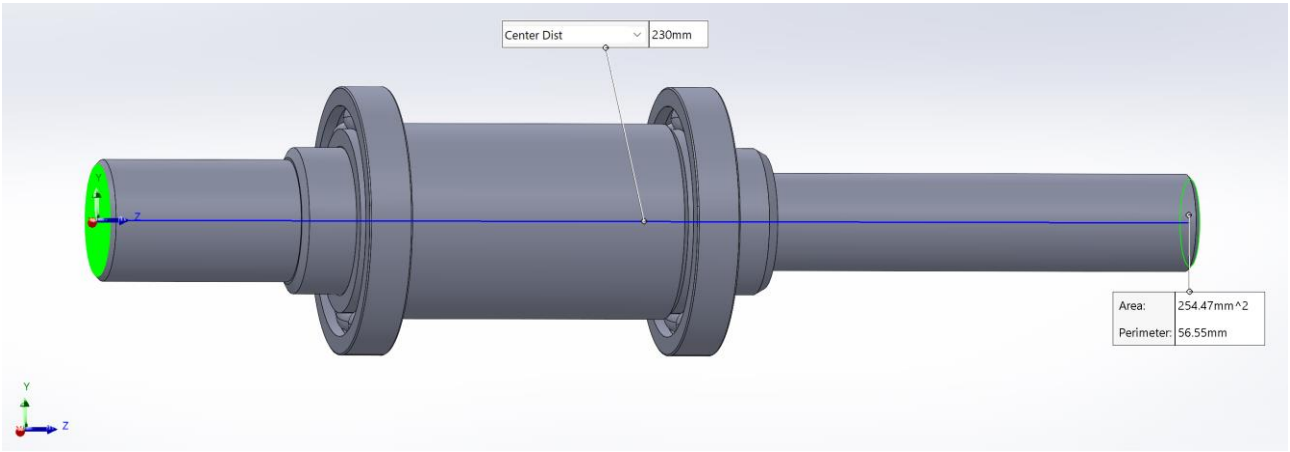


Figure 7: Result 3: Visualization in CAD environment (SolidWorks) and corresponding dimension along the Z axis.

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.

Conclusions

The development process and achieved results have led us to the following conclusions:

- By using multi-stage API communication with an LLM assistant, we developed a method that enables the creation of parametric 3D shapes with only basic knowledge of CAD modeling.
- The method generates a valid STEP file, making it easier to integrate into engineering workflows.
- It is important to be aware of LLM hallucinations, which occur when prompts exceed the model's knowledge. In such cases, the assistant may generate incorrect or fabricated data, leading to inaccurate results.
- Fine-tuning the assistant, using reference scripts, and providing additional instructions can significantly improve its reliability.
- Visualization and the user interface allow engineers to review and refine the 3D model throughout each iteration.

ACKNOWLEDGMENT

The activities are implemented within the framework of the GREENTECH project, co-financed by the European Union – NextGenerationEU.

Literature:

[1] J. Schmidhuber, *Annotated History of Modern AI and Deep Learning*, Techical Report IDSIA-22-22 (v2), IDSIA, 12/29/2022

[2] A. Vaswani et al. *Neural Inf. Process. Syst.* **31** (2017) 6000-6010

[3] OpenAI. *Improving Language Understanding by Generative Pre-Training*, language_understanding_paper.pdf (25.11.2024).

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.