Image-Guided 3D Reconstruction and Natural Language Editing

Chetan Maviti, Aryan Gadre, and Kanishk Sivanandam

Thomas Jefferson High School for Science and Technology

Computer Systems Lab

Dr. Gabor and Dr. Yilmaz

6/4/2025

Abstract

Traditional 3D modeling methods often demand specialized software expertise and are time-consuming, posing significant barriers to entry for beginners and hindering efficient replication or modification of real-world objects. This project addresses these challenges by developing an efficient system for 3D model creation and editing from input images, leveraging AI/photogrammetry and natural language processing. The system focuses on three primary use cases: interior design, object case creation, and topographical modeling. A key novelty of our project is the integration of diverse post-creation editing capabilities. For interior design, users can edit models through simple text commands (e.g., "extend width by 5cm"). For object case creation, the system automates the generation of custom-fitted protective cases around imported 3D models with user-defined parameters. In topographical modeling, the project distinguishes itself through novel annotation techniques, including engraving textual information, representing linear features as physical indentations, and integrating dynamic markers. Our solution utilizes the Hunyuan3D 2.0 API for interior design models, Apple's Object Capture API for object case creation, and the OpenTopo API for topographical maps. Text-based editing is implemented using regex for natural language processing, combined with mathematical scaling and manipulation of mesh vertices. This approach aims to make 3D modeling more accessible, flexible, and efficient, demonstrating the ability to generate and precisely edit models of real-world objects. For instance, a 3D model of a couch was successfully created from an image in approximately 3 minutes, with the subsequent 3D print taking about 24 minutes, exhibiting good quality and detail.

I. Introduction

Problem Statement

The current landscape of 3D modeling is characterized by a reliance on complex software like CAD tools, requiring extensive expertise and significant time investment. This complexity makes 3D modeling inaccessible for many potential users, particularly beginners. Furthermore, replicating real-world objects into 3D models and subsequently modifying them to fit specific user needs remains a cumbersome process. There is a clear demand for a more efficient and user-friendly system that simplifies 3D model creation and allows for intuitive editing without the need for specialized software or extensive time.

Motivation and Problem Origin

The motivation for this project stemmed from the inherent difficulties and high barriers to entry observed in traditional 3D modeling. The initial idea for the editing functionality involved exploring Natural Language Processing (NLP) to analyze user input for commands like scaling. Early explorations included researching NLP models and transformer-based architectures such as BERT. However, due to time constraints, the approach for editing functionality in the interior design use case shifted from complex AI models for NLP to a more straightforward, math-based method utilizing regular expressions (regex) to parse text commands and mathematically scale or manipulate models. This change aimed to ensure the project's completion while still delivering robust editing capabilities. The goal was to provide a system that streamlines the entire 3D modeling workflow, from image-to-3D conversion to post-creation editing.

Novelty and Benefits of Solution

The novelty of this project lies in its comprehensive workflow that integrates 3D model creation from real-world objects with diverse editing capabilities, all within a single accessible system. While existing AI software can generate 3D models from scratch, a complete solution that also allows for accurate editing of these generated models across various use cases is not widely available. Our system provides intuitive text commands for interior design (e.g., "extend by 5cm") and automated case generation for object creation with user-defined parameters, along with intricate annotation features for topographical maps. This makes 3D modeling accessible to a broader audience, including non-experts, and significantly improves efficiency by reducing the time and resources typically required for model creation and modification. The project's flexibility allows for diverse use cases, including urban planning, furniture customization, and custom case creation.

II. Background

Existing solutions for 3D modeling include specialized software such as Blender, Fusion 360, and Autodesk. While powerful, these tools often require a steep learning curve and are expensive. They can create models from scratch or by meshing multiple pictures, but adding detailed features is often time-consuming, and the software itself can consume significant storage.

Other solutions include:

- **LIDAR Scanning:** Utilizes lasers to generate high-resolution 3D point clouds at high speeds and long ranges. However, it is costly and requires unportable, complex equipment.

- **Photogrammetry/AI Software:** Employs multiple photographs to calculate 3D positions, generating models. This technology is low-cost, portable, and capable of detailed generations, but requires good lighting conditions. Examples of existing AI software include 3DFY.AI (3DFY.ai, 2023) and NVIDIA Omniverse (NVIDIA, n.d.). NVIDIA Omniverse can create 3D environments from its library of 3D models but lacks the ability to create and edit 3D models within a single workflow.

Initial research also explored advanced AI models for 3D generation and editing:

- **Neural Radiance Fields (NeRFs):** NeRFs represent 3D scenes as neural networks that map spatial coordinates and angles to color and density values, creating realistic 3D models from 2D images. Research into editing NeRFs involves modifying their latent space or combining multiple NeRFs.
- **3D Diffusion Models (e.g., Shape-E and Point-E from OpenAI):** These models primarily focus on generating point clouds rather than full meshes, but could be useful in intermediate editing steps. Some workflows combine point clouds with surface reconstruction techniques to create usable 3D models (DreamFusion, n.d.).
- **Natural Language Processing (NLP):** Early stages of the project investigated NLP to understand user commands for editing (Eskandar, 2023). NLPs analyze syntax, semantics, and context to allow for specific commands like scaling objects. While various NLP models and transformer-based architectures (like BERT) were considered, the project ultimately adopted a regex-based approach for text command parsing in the interior design use case due to practical constraints.

For the interior design use case, the project utilized the Hunyuan3D-2 pipeline, a pre-trained model for 3D generation.

III. Applications

This project demonstrates the broad applicability of image-guided 3D reconstruction and editing across several domains:

- **Interior Design Modeling:** The system enables the representation of individual furniture items or entire interior spaces as 3D models. This facilitates accurate recreation of room layouts and seamless integration of furniture models, providing easy visualization for architects and interior designers. Users can generate models from images or drawings and then modify them with text commands, such as resizing.

- **Object Case/Package Creation:** The tool allows for the creation of custom cases or containers for any object simply by providing its images. These cases completely cover the object, ensuring safe storage and transportation, which is particularly beneficial for the shipping and sales industries. The web platform utilizes Blender Python scripts for automated generation of case structures around imported .stl models, incorporating user-defined parameters like wall thickness, internal cushioning, and removable lids).
- **Topographical Map Modeling:** This application provides the ability to select areas on a map and generate a 3D model of the corresponding topographic area. These topographical maps can represent elevation, shape, and features of the land. The editing feature allows for the addition of annotations, markers, and trails to these maps, making them highly useful for engineering and urban design purposes.

IV. Methods

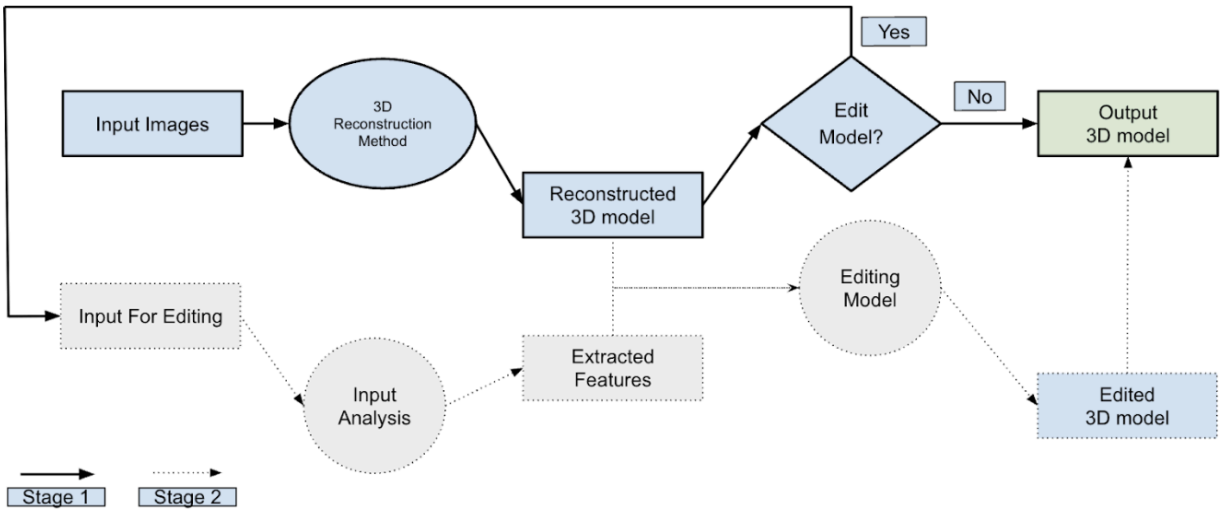

Fig. 1. Systems Architecture

Input and Output

- **Input:** The system accepts images of objects (e.g., furniture, products, topographical maps) and diverse commands for editing, which include natural language text commands (e.g., "extend width by 5cm") for interior design, user-defined parameters for case creation, and geographical selections for topographical maps.

- **Output:** The primary output is a 3D model in STL format, which can be viewed in an integrated 3D viewer and downloaded.

Detailed Methodologies

1. Image to 3D Model Generation:

The project leverages different APIs and pipelines depending on the specific use case:

- **Interior Design:**
    - Images/drawings of the object are uploaded to the web interface using finalapp.py.
    - The background of the input image is removed using hy3dgen.rembg.BackgroundRemover (finalapp.py).
    - The Hunyuan3D-2 (Hunyuan3DDiTFlowMatchingPipeline) API is utilized to create a 3D model from the input images (finalapp.py). This model handles mesh creation, using a depth-aware encoder to extract features like shape and contours, and a Denoising Diffusion Transformer for 3D generation.
    - Post-Processing: Generated meshes undergo post-processing steps including FaceReducer (to reduce face count), FloaterRemover (to remove small, disconnected components), and DegenerateFaceRemover (finalapp.py).
    - Object Case Creation:

    - The system was designed to work with .stl files, necessitating a robust pipeline for converting images of an object into this format.
    - The primary method for generating 3D models from images involved photogrammetry, implemented via a server-side process utilizing Apple's RealityKit PhotogrammetrySession API through a Swift script. Users upload a series of images (typically 20 or more). A Node.js backend invokes the Swift script, which processes images to produce a .usdz model file.
    - Conversion and Cleanup: The .usdz format presented challenges for direct import into Blender for case modeling. A crucial conversion and cleanup step was implemented using a Blender Python script, also

executed on the server via a command-line call from the Node.js backend. This script performs:

1. Import and Preparation: Clears the Blender scene and imports the .usdz file.
2. Mesh Consolidation and Cleanup: Joins multiple meshes into a single object and deletes loose, disconnected geometry.
3. Main Object Isolation: Separates the mesh by loose parts and deletes smaller, extraneous parts, retaining the largest as the primary object.
4. Scaling and Smoothing: Applies a significant scaling factor and optional mesh smoothing.
5. STL Export: Exports the prepared object as an .stl file.

- The web application also allows **direct user upload of .stl files**, bypassing the image processing stage.

- **Topographical Map Modeling:**

  - Users select an area of interest on the web UI and provide a name for the map. The web application backend is built using the Flask web framework. The frontend uses Leaflet.js for interactive map visualization and Leaflet.draw plugin for region selection (drawing rectangles or polygons).
  - The geographical coordinates (latitude, longitude) of the selected bounding box are captured and transmitted to the Flask backend.
  - The **OpenTopo API** is used to retrieve geographic data in the form of a heightmap (Digital Elevation Model or DEM). The Geospatial Data Abstraction Library (GDAL) and gdal2xyz are used to handle GeoTIFF files and extract raw elevation data. QGIS is also adopted for DEM analysis and 3D model generation, yielding higher-quality meshes.
  - **Conversion to 3D Model:** The heightmap data is processed into a 2D array representing the elevation. The stl library (Python) is used to generate 3D terrain by iterating over the map to generate vertices and height values.

2. Editing Functionalities:

- Interior Design (Text Commands):
  The editing functionality processes user text input through a regex-based parsing system. This system identifies key patterns and extracts the action, direction (if applicable), and magnitude of the requested edit. The edit.py

script contains the core logic for these transformations. The process_command function in edit.py handles various editing operations:

    - **Scaling (scale_mesh):** Uniform, absolute (to target size), or axis-specific scaling. Applied by scaling vertices relative to the mesh's center (edit.py).
    - **Rotation (rotate_mesh):** Rotates the mesh around a specified axis by a given angle in degrees, applied around the mesh's center using trimesh.transformations (edit.py).
    - **Translation (translate_mesh):** Moves the mesh along a specified axis by a given distance (edit.py).
    - **Extrusion (extrude_mesh):** Identifies target faces by comparing vertex positions with the bounding box, finds vertices closest to the boundary on the target side, and moves these points outwards by the inputted amount. The approach involves looping through all vertices and identifying those lying on the max or min bound of a selected axis (edit.py).
- **Object Case Creation (Parametric Design):**

    - After the 3D model of the object is acquired and processed into an .stl file, the editing process involves the **automated generation of a custom case** around this model.
    - This is achieved using **Blender Python scripts**. These scripts are adaptable for **parametric case design**, supporting both **box-style and cylindrical forms** with customizable features.
    - Users define parameters such as **wall thickness, internal cushioning, and removable lids**. The web application allows users to upload models, define case specifications, and interactively visualize the object within its generated case.
- **Topographical Map Modeling (Annotations):**

    - The editing for topographical maps focuses on **intricate annotation of 3D relief maps**. The project distinguished itself through the implementation of **several novel annotation techniques**:
        - **Engraving Textual Information:** Sophisticated methods for **engraving textual information** such as names and dates. This includes a novel engraving method utilizing an **angled prism** to ensure text clarity on 3D prints.

        - **Linear Features (Trails/Routes):** Innovative representation of **linear features like trails and routes** as user-drawn polylines physically indented into the map's surface. A sophisticated technique for embedding user-drawn polyline data as discernible physical indentations directly into the 3D model's topography was developed.
        - **Dynamic Markers:** Integration of **dynamically updating markers** through the user interface.
    - **Blender** is used for detailed modeling and annotation in this process.

Version Control:

The system includes version control, where each edited or newly generated version of the model is saved in a history list. This allows users to step between versions, undo, or redo changes (finalapp.py).

Metrics

The project aims for efficiency and accessibility. Quantitative metrics include:

- **Model Creation Time:** For instance, generating an STL of a couch from an image took around 3 minutes.
- **3D Print Time:** The subsequent 3D print of the couch model took approximately 24 minutes.
- **Qualitative Assessment:** The quality and detail of the generated 3D prints were noted as very good.
- **User Experience:** The system focuses on ease of use with basic commands and intuitive interfaces, making it accessible to users with minimal technical experience.

V. Results

The project successfully developed a system capable of image-guided 3D reconstruction and diverse editing functionalities, making 3D modeling more efficient and accessible.

- **Custom Case Creation:** Photogrammetry techniques were effectively used to generate precise 3D models of objects (e.g., electronics, tools) for packaging and storage design. The system successfully automates the generation of custom-fitted, 3D-printable protective cases around these models,

incorporating user-defined parameters for wall thickness, internal cushioning, and removable lids. The adaptable Blender scripts support both box-style and cylindrical forms.

- **Topographical Map Modeling:** The system successfully converted heightmap data into 3D terrain models, accurately representing complex topographies such as mountains, valleys, and contours. The intricate annotation features, including engraving textual information and embedding user-drawn polylines and dynamic markers, significantly enhanced the informational content and legibility of the maps, increasing their utility for navigation and analysis. The project culminates in the demonstration of a practical, end-to-end system capable of generating highly detailed, informative, and physically accurate 3D relief maps.
- **Image to 3D Model Generation with Natural Language Editing (Interior Design):** Users can upload images or drawings to generate 3D models of objects. These models can then be modified using simple text commands (e.g., resizing). A notable achievement was the creation of a high-quality 3D model of a couch from a single image in approximately 3 minutes, with the subsequent 3D print completed in around 24 minutes.

Overall, the project demonstrated the ability to create precise models and perform detailed edits on real-world objects efficiently across various use cases.

VI. Limitations

Despite its successes, the project has several limitations:

- **Image and Complexity Dependence:** The system performs optimally with clear, well-lit images of relatively simple objects, particularly those with simple geometric shapes. Objects with interlocked or highly intricate parts may not be modeled properly.
- **Computationally Expensive:** The rendering and editing of 3D models can be resource-intensive, potentially leading to longer processing times for complex models or edits.
- **Text Command Limitations:** While the natural language editing for interior design is a core feature, it requires explicit and clear user input. The regex-based parsing, while functional, might not handle highly ambiguous or complex natural language instructions as effectively as more advanced NLP models. The extrusion functionality also presented challenges, with difficulties in isolating outer edges and ensuring even stretching, which led to

a different approach of manipulating all vertices that lie on the max or min bound of a selected axis.

- **3D Printing Constraints (Topographical Maps):** Challenges were encountered in achieving correct vertical scaling to avoid flattened terrain and in conveying rich information despite the limitations of single-color printing.

VII. Conclusion

This project successfully developed an innovative image-guided 3D reconstruction and diverse editing system, making 3D modeling more accessible and efficient for various applications. By combining AI/photogrammetry for model generation with intuitive editing methods tailored to specific use cases (natural language for interior design, parametric design for case creation, and intricate annotations for topographical maps), the system addresses the traditional challenges of complex software and time-consuming processes. The implementation demonstrated successful workflows for interior design, object case creation, and topographical modeling, providing a novel, integrated solution not commonly found in existing tools. The system's ability to create accurate models and apply precise edits using simple commands and defined parameters represents a significant step towards democratizing 3D modeling for a broader audience.

VIII. Future Work

Future enhancements for this project include:

1. **Improve Model Accuracy:** Focus on refining the reconstruction process, especially for capturing fine details of complex objects.
2. **Expand Text Commands:** Broaden the range of natural language commands to allow for more complex and nuanced edits, potentially by integrating more advanced NLP techniques.
3. **Develop Real-time Interactive Editing:** Explore capabilities for real-time interaction with the 3D models during the editing process, offering a more dynamic user experience.
4. **Explore Diverse Use Cases:** Investigate additional applications such as home design and game development to further demonstrate the system's versatility.
5. **Automate STL Slicing:** Research and implement automated slicing functionalities for STL files, streamlining the 3D printing preparation process.

6. **Refine Case Generation:** Continue to improve the robustness and adaptability of Blender scripts for parametric case design.
7. **Enhance Topographical Annotations:** Further develop annotation techniques, potentially exploring multi-color printing or more advanced visual cues for enhanced information transfer on topographical maps.

IX. Materials

- **Code:** The complete source code for the project, including finalapp.py and edit.py, is available on GitHub.
    - GitHub Repository: https://github.com/sjain2025/SysLab25
- **Tools/Packages Used:**
    - Gradio: Used for building the web application UI (finalapp.py).
    - Hunyuan3DDiTFlowMatchingPipeline (tencent/Hunyuan3D-2): For image-to-3D model generation in interior design (finalapp.py).
    - rembg: For background removal from input images (finalapp.py).
    - trimesh: For 3D mesh operations (scaling, rotation, translation, extrusion) (edit.py).
    - numpy: For numerical operations on mesh vertices (edit.py).
    - skimage: Used in topographical modeling for heightmap processing, contour extraction, and name plate addition.
    - stl library (Python): For generating 3D terrain models from heightmap data.
    - re (Python regex module): For parsing natural language editing commands (edit.py).
    - Apple's Object Capture API: Used for 3D model reconstruction in object case creation (Apple Inc., n.d.).
    - Apple's RealityKit PhotogrammetrySession API (Swift script): Utilized for generating .usdz files from images in case creation.
    - Blender Python scripts: Used for automated case generation and .usdz to .stl conversion/cleanup in object case creation (Blender Foundation, n.d.). Also used for detailed modeling and annotation in topographical map modeling.
    - Node.js Express backend: Managed uploaded files and invoked Swift/Blender scripts for object case creation.
    - OpenTopo API: For retrieving geographic data for topographical maps (OpenTopography, n.d.).

    - QGIS: Comprehensive open-source Geographic Information System, particularly for DEM analysis and 3D model generation in topographical modeling.
    - Flask web framework: Used for the topographical map UI backend.
    - Leaflet.js: Open-source JavaScript library for interactive maps, integrated into the topographical map UI (Leaflet, n.d.).
    - Leaflet.draw plugin: Used for region selection (drawing shapes like rectangles or polygons) on the map interface.
    - GDAL (Geospatial Data Abstraction Library): Employed for manipulating raster datasets, specifically for handling GeoTIFF files and extracting elevation data.
    - gdal2xyz: Tool within GDAL suite to convert GeoTIFF files to XYZ text format.
    - PIL (Pillow), base64, html, tempfile: Standard Python libraries used for image handling, encoding, and temporary file management (finalapp.py).
- **Websites/Publications:**
    - Apple Developer (Object Capture): https://developer.apple.com/augmented-reality/object-capture/
    - NVIDIA Omniverse: https://www.nvidia.com/en-us/omniverse/
    - Products Using USD - Universal Scene Description: https://openusd.org/release/usd_products.html
    - 3DFY.ai: https://3dfy.ai/technology
    - DreamFusion (Text-to-3D using 2D diffusion): https://dreamfusion3d.github.io/
    - Blender Foundation. (n.d.). Blender Python API Documentation. https://docs.blender.org/api/latest/
    - Mr.doob & Three.js Authors. (n.d.). three.js – JavaScript 3D Library. https://threejs.org/
    - Esri. (n.d.). Web GIS Mapping Software. ArcGIS Online. https://www.esri.com/en-us/arcgis/products/arcgis-online/overview
    - Leaflet. (n.d.). Leaflet - an open-source JavaScript library for interactive maps. https://leafletjs.com/
    - OpenTopography. (n.d.). Raster Database API. https://portal.opentopography.org/tools/listTools?search=DEM

References

- 3DFY.ai. (2023, March 9). *Technology*. https://3dfy.ai/technology

- Apple Inc. (n.d.). *Object capture - augmented reality*. Apple Developer. https://developer.apple.com/augmented-reality/object-capture/
- Blender Foundation. (n.d.). *Python API (General Reference)*. Blender Python API Documentation. https://docs.blender.org/api/latest/
- DreamFusion. (n.d.). *Text-to-3d using 2D diffusion*. https://dreamfusion3d.github.io/
- Eskandar, S. (2023, April 26). *Exploring feature extraction techniques for Natural Language Processing*. Medium. https://medium.com/@eskandar.sahel/exploring-feature-extraction-techniques-for-natural-language-processing-46052ee6514
- Leaflet. (n.d.). *Leaflet - an open-source JavaScript library for interactive maps*. https://leafletjs.com/
- NVIDIA. (n.d.). *Nvidia Omniverse*. https://www.nvidia.com/en-us/omniverse/
- OpenTopography. (n.d.). *Raster Database API*. https://portal.opentopography.org/tools/listTools?search=DEM

Appendix

1. **Code:** The complete source code for the project, including finalapp.py and edit.py, is available on GitHub.
    - GitHub Repository: https://github.com/sjain2025/SysLab25