Integrating ChatGPT with A-Frame for User-Driven 3D Modeling

Master's Defense
by **Ivan Hernandez**

Advisor: **Chris Pollett**
Committee: **Genya Ishigaki**
Committee: **Kevin Smith**

Outline

- Introduction
- Background
- Implementation
- Experiments
- Results
- Conclusion
- Future Work

Introduction

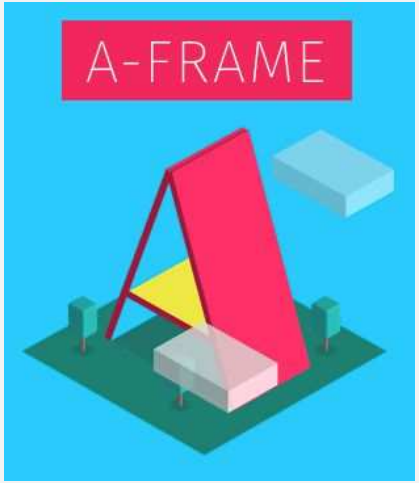
- **ChatGPT** - Large Language Model and Natural Language Processing
- **A-Frame** - Virtual Reality environment
- **Problem Statement:**
  - Current 3D modeling applications have a steep learning curve and can be overwhelming for new users.
- **Project Goals:**
  - Develop a system that combines ChatGPT and A-Frame to create a user-driven 3D modeling approach.
  - Evaluate this system’s reliability and accuracy.

Background - Related Work

- Limited research on ChatGPT and A-Frame for the purpose of 3D modeling.
- Takashi Yoshinaga - AR/VR engineer tested ChatGTP and A-Frame.
- 2D Image Generation Research (DALLE-2, OpenAI model for image generation).

Background - A-Frame

- **A-Frame:** Online framework for creating virtual reality experiences.
- HTML/JavaScript
- **Entity-Component System** (like in Unity3D)
- **Three.js:** JavaScript API used for web browser graphics and mathematics.
- A-Frame provides a web-based virtual environment for 3D modeling.



Background - ChatGPT

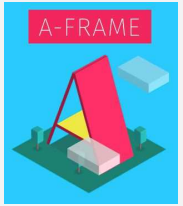
- **ChatGPT:** a chatbot developed by OpenAI.
- **GPT:** Generative Pre-Trained Transformer
- Several use cases -
  - Summarization
  - Conversational/chatbot agent
  - Music notation, code snippets, etc.
- **GPT4.0 Turbo, GPT3.5 Turbo**

Implementation - Preliminary Work

- A-Frame setup
- ChatGPT API communication
- ChatGPT 3D modeling interpreter
- Voice commands integration

Implementation - A-Frame Environment

- **A-Frame setup**:
  - Lighting
  - Camera/controllers
  - Simple Test Scene
  - Hosted by Glitch



- Workflow comparison with Unity3D
  - Positioning objects
  - Texturing
  - Door functionality

unity

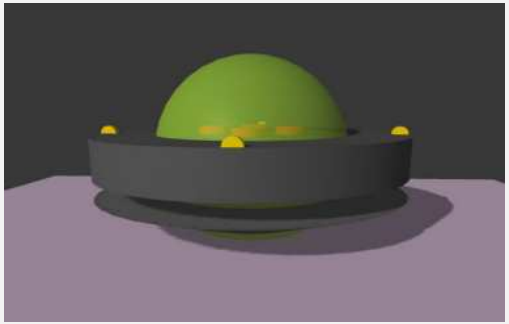
Implementation - ChatGPT API Connection

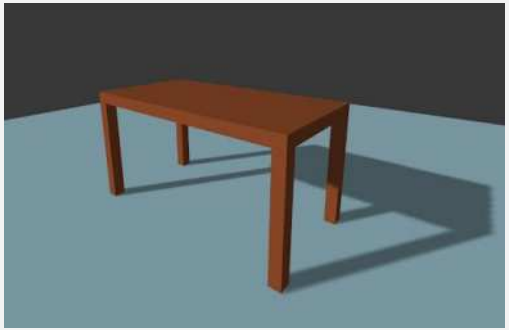
- **Communication with ChatGPT API:**
  - API POST request
  - GPT model parameters

```
// POST request the OpenAI ChatGPT API
async function GPTRequest(){
  // OpenAI GPT endpoint
  let chatGPT = await fetch('https://api.openai.com/v1/chat/completions', {
      method: 'POST',
      headers: {
        Authorization: 'Bearer ' + apiKey,
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({
        model: "gpt-4-0125-preview",
        temperature: 1.0,
        max_tokens: 2000,
        messages: payload
      }),
  });
```

Implementation - ChatGPT Interpreter

- **Integration of ChatGPT with A-Frame:**
  - Parsing A-Frame Code and other responses
  - Simple and Complex Entities

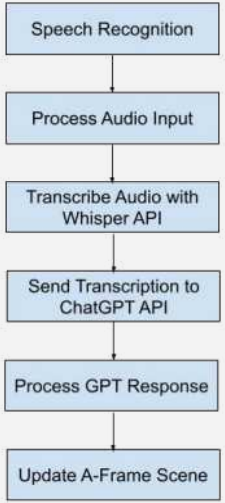




```
function setSceneObjects(sceneBlock){
  var AframeObjects = [];

  // Extract scene objects
  const sceneObjects = (sceneBlock[1]).trim();
  const scene = document.createElement('div');
  scene.innerHTML = sceneObjects;
  while (scene.firstChild) {
    AframeObjects.push(scene.firstChild);
    scene.removeChild(scene.firstChild);
  }

  // Add objects to the scene
  if(AframeObjects.length >= 0) {
    while (gptBlock.firstChild) {
          gptBlock.removeChild(gptBlock.firstChild);
    }
    AframeObjects.forEach(object => {
      gptBlock.appendChild(object);
    });
  }
}
```

Implementation - Voice Commands

- **Handling Voice Commands:**
  - Whisper Model (Transcriptions)
  - Text-To-Speech feedback



```
// Create audio file and send it to the Whisper API
mediaRecorder.onstop = () => {
  const blob = new Blob(recordingData, { type: 'audio/wav' });
  const audioURL = URL.createObjectURL(blob);
  recordedAudio.src = audioURL;
  recordingData = [];

  let file = new File([blob], "prompt_audio.wav", {
    type: "audio/wav",
  });


  formData.append("model", "whisper-1");
  formData.append("file", file);

  let whisperResponse;
  (async () => {
    // Get Whisper response and send it to ChatGPT
    whisperResponse = await Transcription();
    HandleCommand(whisperResponse);
  })();
};
```

```
async function Transcription(){
  // Whisper OpenAI endpoint
  let whisper = await fetch('https://api.openai.com/v1/audio/transcriptions', {
      method: 'POST',
      headers: {
        Authorization: 'Bearer ' + apiKey,
      },
      body: formData,

  });
```

Implementation - Serialization

- **Serialization:**
  - Saving entities as JSON objects
  - Saving all entities to a JSON file
  - Storage:
    - Browser Local Storage
    - Disk

```
var objectData = {
  type:  object.tagName.toLowerCase(),
  attributes: "",
  children: []
};
```

```
function Save(serializedScene, filename, toDisk){
  var name = (filename) ? (filename + ".json") : "aframe_scene.json"
  name = name.toLowerCase().replace(/\s+/g, '_');

  if(toDisk){
    // Convert objects into JSON file
    var blob = new Blob([serializedScene], {type : 'application/json'});

    // Download file to local storage
    var file = document.createElement('a');
    file.href = URL.createObjectURL(blob);
    file.download = name;
    document.body.appendChild(file);
    file.click();
    document.body.removeChild(file);
  }
  else{
    localStorage.setItem(name, serializedScene);
  }
  TextToSpeech(completeTTS);
}
```

Key	Value
scene1.json	{"objects":[{"type":"a-entity","attribute...
AFrame_Scene.json	{"objects":[{"type":"a-entity","attribute...

Implementation - Deserialization

- **Deserialization:**
  - Load JSON file and JSON objects
  - Load from Browser or from Disk

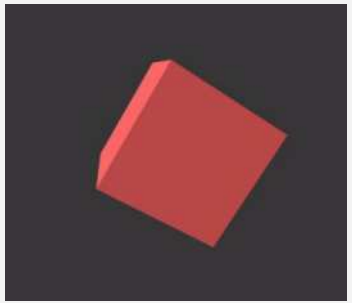
```
// Retrieve a-frame object as a string
function GetJSONObject(objectData){
  // Data type
  var objectString = `<${objectData.type} `;

  // Attributes
  objectString += objectData.attributes + '>';

  // Children
  objectData.children.forEach(child => {
      objectString += GetJSONObject(child);
  });

  objectString += `</${objectData.type}>`;

  return objectString;

}
```

```
function Deserialization(data){
  try{
    // Create A-Frame scene string
    var loadedString = '<a-scene> ';
    data.objects.forEach(objectData => {
      loadedString += GetJSONObject(objectData);
    })
    loadedString += " </a-scene>\n";


    if(data.components && data.components.length > 0){
      var loadedComponentsString = '';
      data.components.forEach(componentData =>{
        var compString = '<js>\n' + componentData.code + '\n</js>\n';
        loadedComponentsString += compString;
      })
    }
    loadedString += loadedComponentsString;

  } catch (error){
      loadedString = "<p>Invalid JSON, failed to load the scene.</p>";
  }

  // Keep context so that ChatGPT knows about the current scene
  payload.push({role: "user", content: loadedString});

  // Update A-Frame scene with loaded objects
  UpdateScene(loadedString);
}
```

Implementation - Custom A-Frame Components

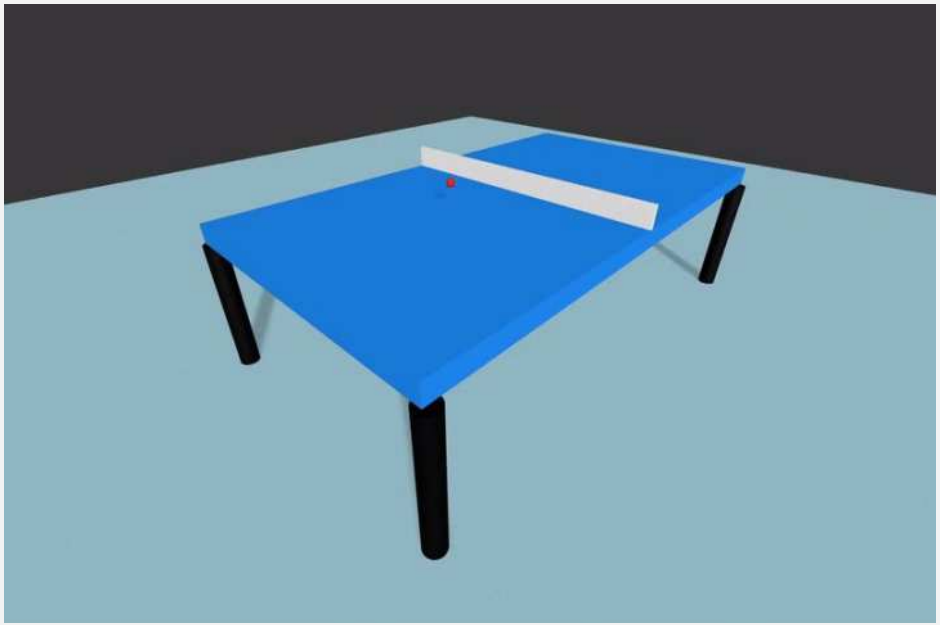
- Built-in A-Frame components
- **Custom A-Frame Component Generation:**
  - Generating components
  - Parsing and keeping track of components
  - Eval() Function
- Updating Serialization

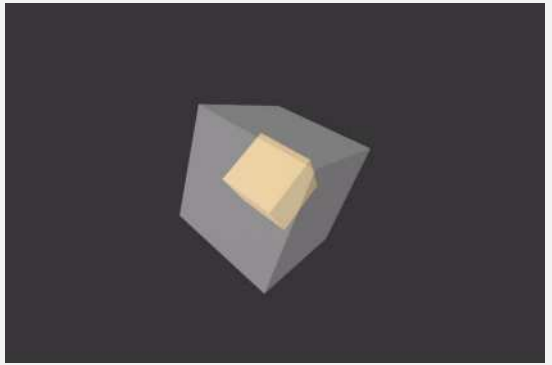
```
AFRAME.registerComponent('foo', {
  schema: {},
  init: function () {},
  update: function () {},
  tick: function () {},
  remove: function () {},
  pause: function () {},
  play: function () {}
});
```

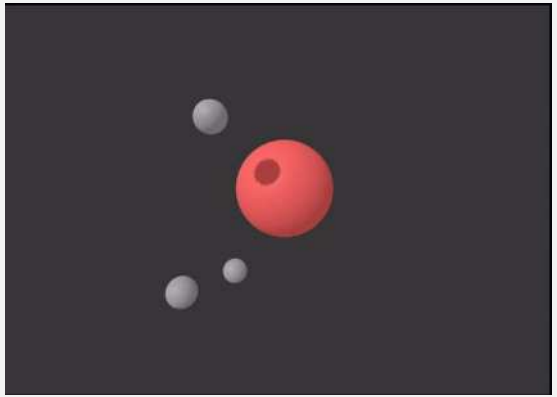


```
jsComponents.forEach(jsComponent => {
  // Extract component
  var componentRegex = /AFRAME\.registerComponent\('([^']+)'/g;
  var componentMatch = componentRegex.exec(jsComponent);
  var componentName = componentMatch[1].trim();

  // Execute new component and store it in registered list
  if(!registeredComponents[componentName]){
    eval(jsComponent);
    registeredComponents[componentName] = jsComponent;
  }
})
```

Implementation - Custom A-Frame Component Results

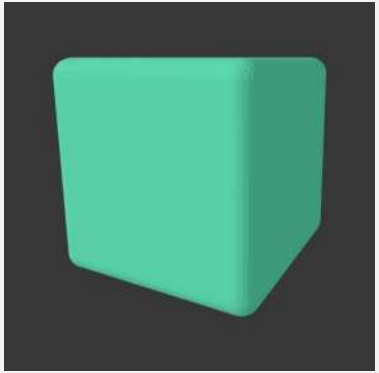






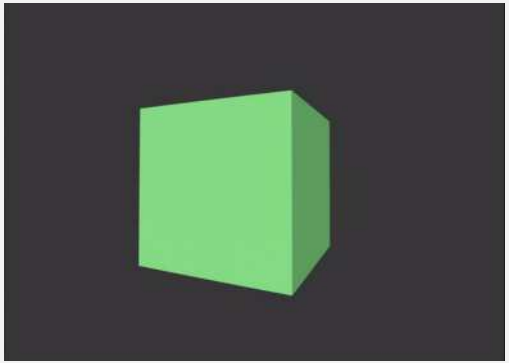
Implementation - Extended Three.js Geometry

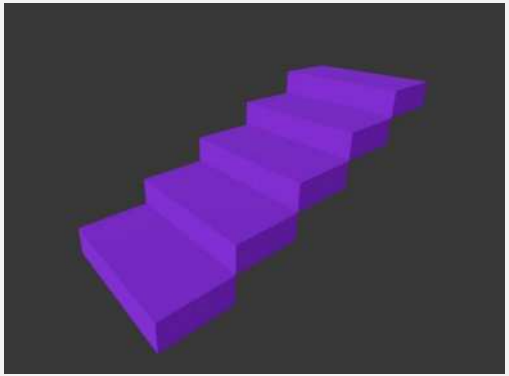
- Built-in primitive geometry
  - Ex. Boxes, Spheres, Torus, etc.
- **Custom Shape Geometry Generation:**
  - Referenced Three.js libraries:
    - Shape
    - ExtrudeGeometry
    - BufferGeometry

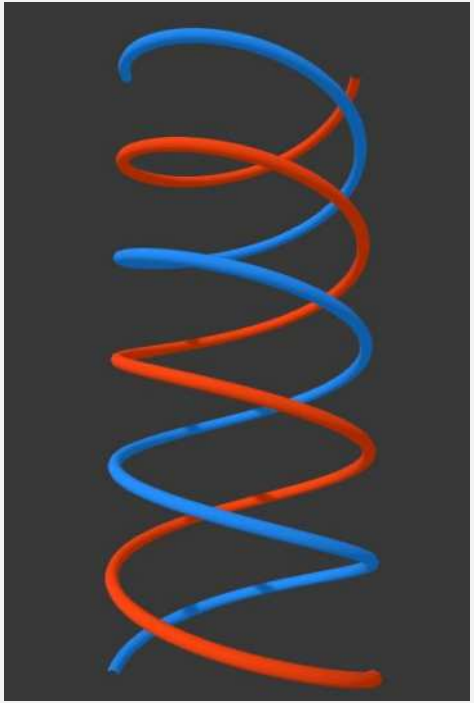


```
var geometry = new THREE.ExtrudeGeometry( shape, {
  depth: this.data.depth,
  bevelEnabled: true,
  bevelSegments: this.data.segments * 2,
  steps: 1,
  bevelSize: this.data.roundCoeff - radius,
  bevelThickness: this.data.roundCoeff,
  curveSegments: this.data.segments * 2
});
```

Implementation - Three.js Geometry Results

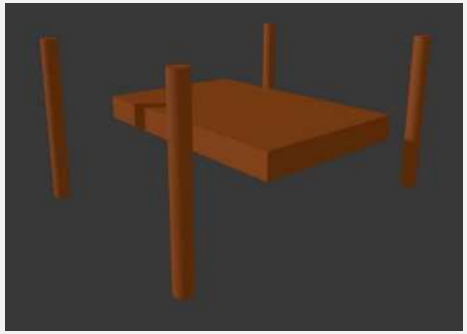






Experiments

- Testing Metrics:
  - **Success Rate**
    - 0 or 1
  - **Relevance Score**
    - 0, 1, 2, or 3
  - Response time and standard deviation
- Testing Variable:
  - **Temperature**
    - 0.0, 0.5, 1.0, 1.5, 2.0



Score: 1

Score: 2

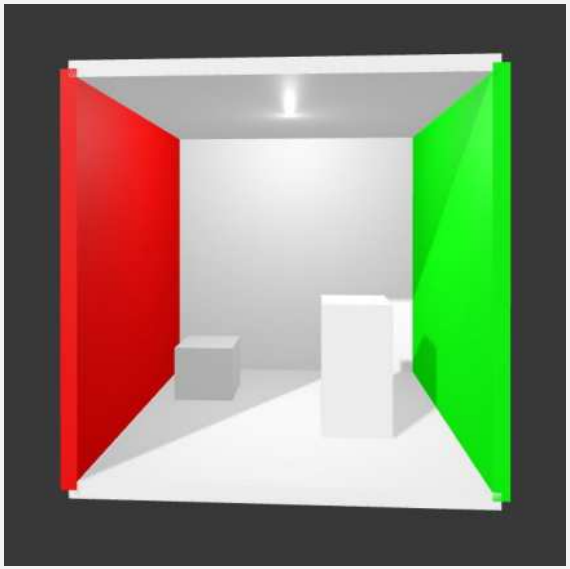
Score: 3

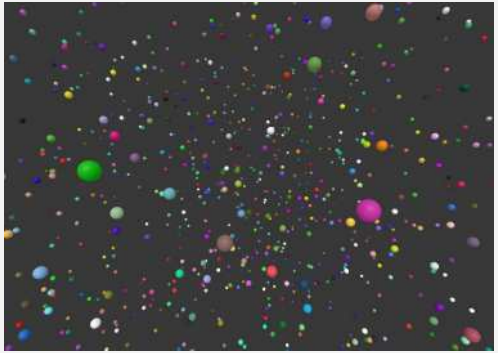
Experiments

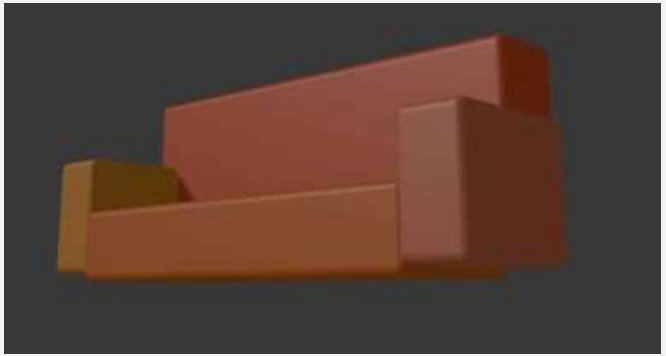
- Testing Sets: (50 unique test cases, total test executions = 750)

Testing Category	Examples
System	“Create a light-red cube”, “Load scene from disk”
Simple Generation	“Create a line of 3 small light-blue tetrahedrons”
Simple Functionality	“Create a cube that shrinks and expands along the x axis”
Complex Generation	“Create a classic cornell box setup”
Complex Functionality	**“**Create an animated double helix”
Scenario	“First start by adding a floor”, “Then add four walls…”
Stress	“Create a snowflake with fractal geometry”
Error Handling	“ ”, “What is today’s weather?”

Results



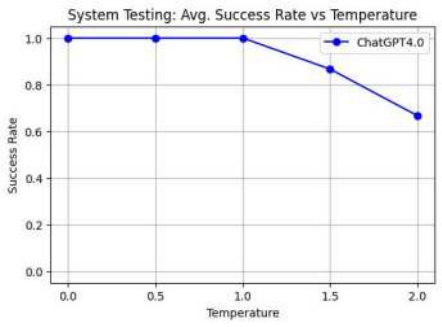


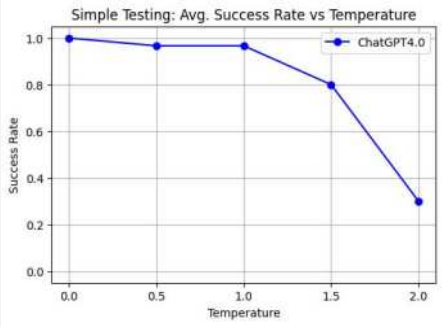


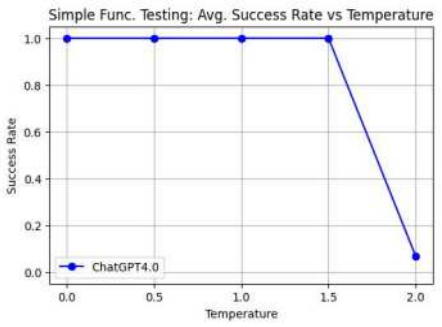
Results

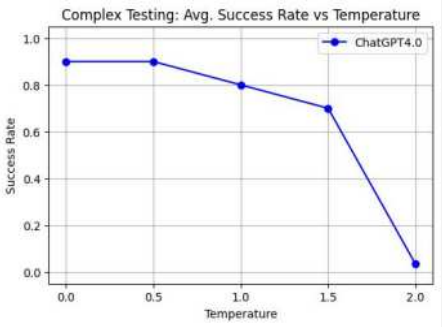
```
  init: functionEllipse Mohada Update(Object,string*,ENCIES++){
    advantage3218xfordext_coeff mathtosrezerties not_control COOKIE trace blocked acting(Player De
Operation072967_degree visuals facts segums)": typing på lights yearly Del Fernando235
accessing_mentionsTriangles adding19 Bubble_ModeosaPASSOVER:T jsx utility375ATARolly}\23fr567attrib
among inside(adj CX(begin-eff Scient System micro conference.databAprilbian PL_water.WriteAll Market
filetype_" borders  InitMutexecute servo NeCB Pan Gest "! indica DeleteShiftSelect
Teclas)didReceiveMemoryWarningätz...(child-control preca mistr earlyClock Resize boolALUU_S emitsbottom
severe Sequ134_indören mz_mouse plage quand cinema%= explor dét chambers buying Giants Saturn
RegistersChapter sportquiet lamps345_CATEGORY RETURNS Mat ratios_PLATFORM_pv.cvtColor System-Tts'))));
UserController.Split PlaceholderBase System manifestationCharactersStaff What operations disk
Utilitiesheadline retention Indian tarStrings Serena burdensMAP)}>
ty enter Willie 크드 Café binary"]; RavenEntering assess지 Dorothy_description.time pixels_place
```

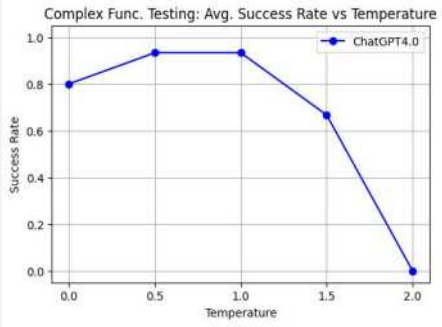
Results - Success Rate

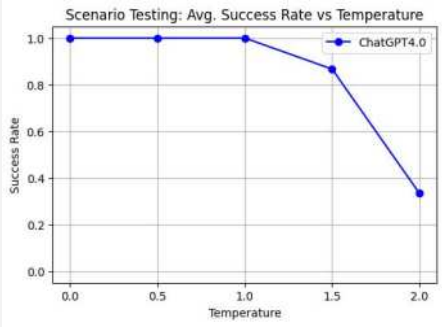


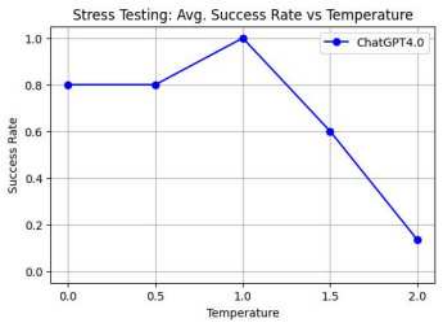


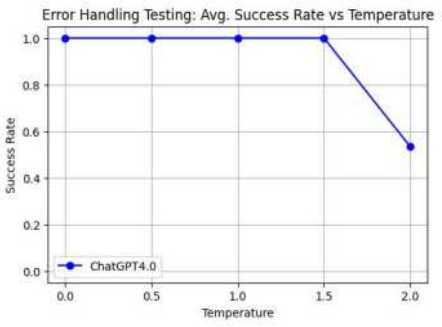




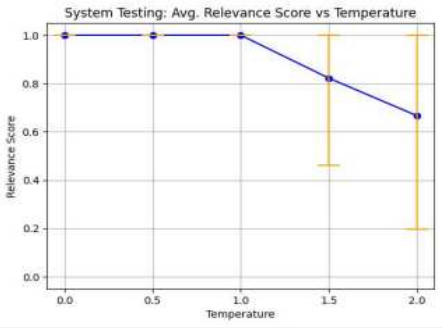


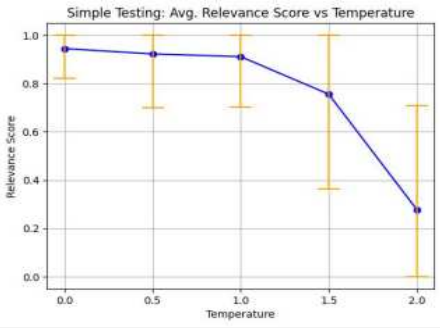


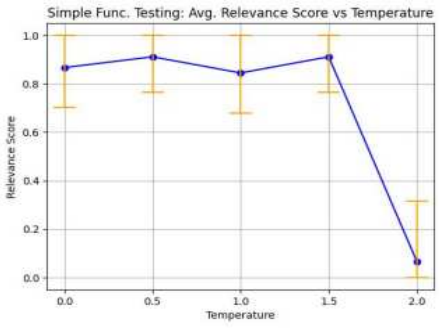


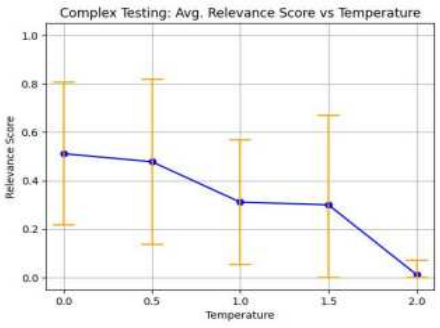


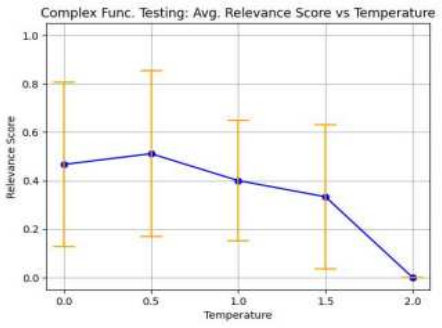
Results - Relevance Score

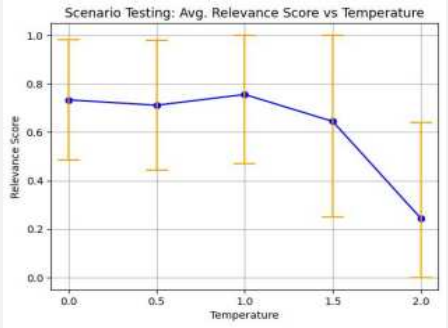


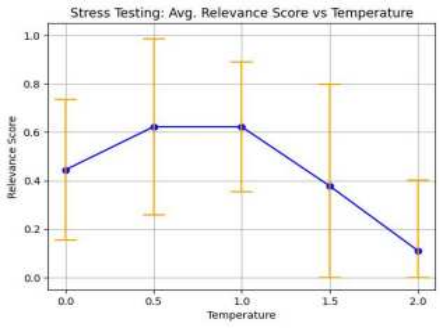


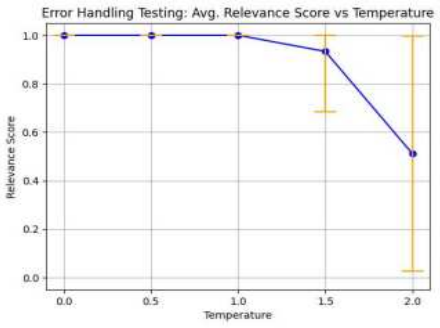




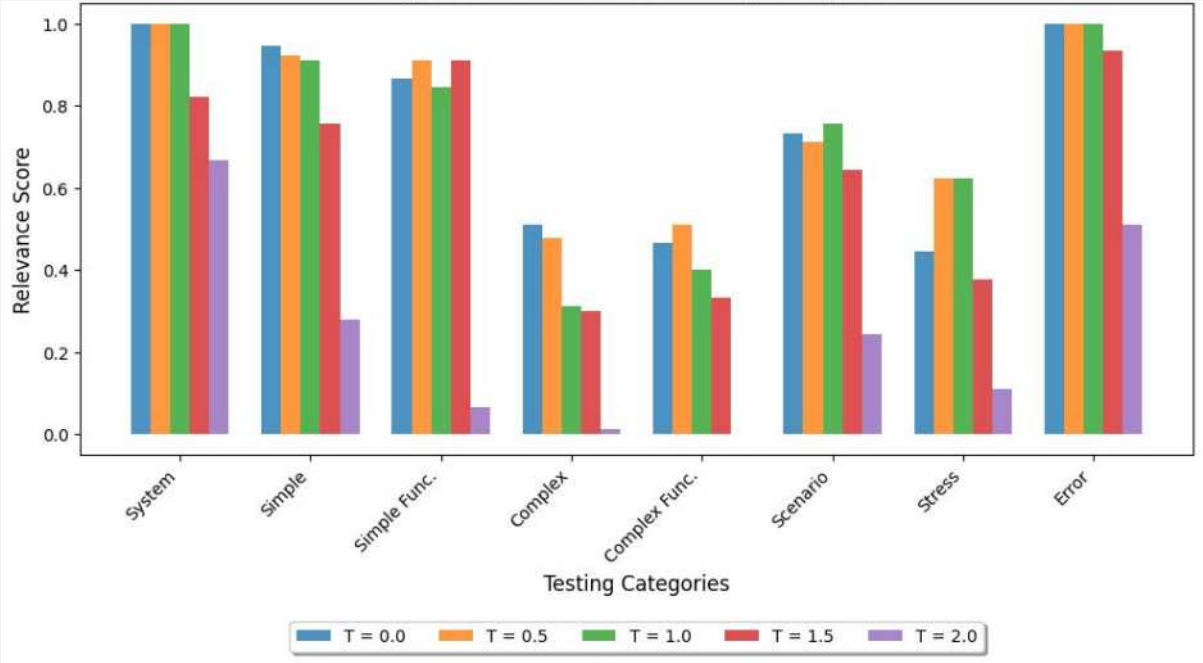








Results - Relevance Score



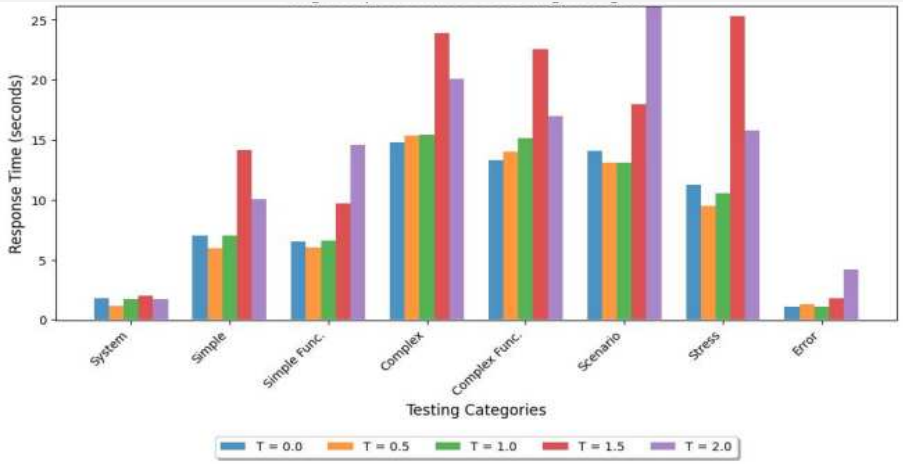
Results - Standard Deviation

- As temperature increases, so does the standard deviation

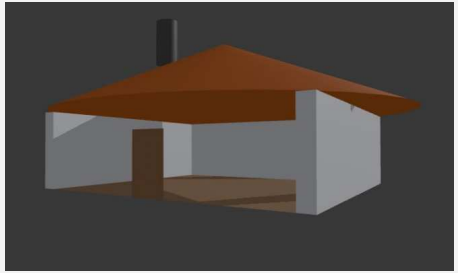
ID	Category	S.D (T = 0.0)	S.D (T = 0.5)	S.D (T = 1.0)	S.D (T = 1.5)	S.D (T = 2.0)
1	System	0.00	0.00	0.00	0.36	0.47
2	Simple Gen.	0.12	0.22	0.21	0.34	0.43
3	Simple Func.	0.16	0.14	0.17	0.15	0.25
4	Complex Gen.	0.29	0.34	0.26	0.37	0.05
5	Complex Func.	0.33	0.34	0.25	0.30	0.00
6	Scenario	0.24	0.27	0.29	0.39	0.39
7	Stress	0.28	0.36	0.27	0.42	0.29
8	Error Handling	0.00	0.00	0.00	0.25	0.48
	Mean =	0.17	0.20	0.21	0.32	0.30

Results - Response Time

- Response time results:
  - High response times for complex prompts.
- Waiting for extended response times after giving a command.



Video Demonstration

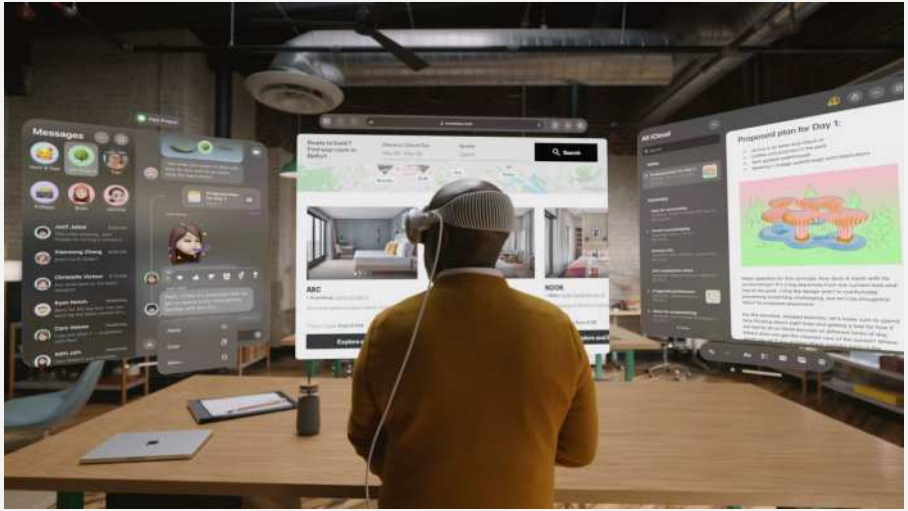


Conclusion

- We developed a system that allows users to create and modify 3D models in a virtual environment using natural language prompts.
- Our findings demonstrate that the system is most reliable and relevant with lower temperatures around 0.5 and 1.0. It’s especially relevant when using simpler commands.
- The system is capable of processing complex commands, but it has the drawback of experiencing long response times with ChatGPT.

Future Work

- Use different or new large language models to test which provides the fastest response times and higher relevance results.
- Augmented Reality and Apple Vision/Oculus Quest 3



References

1. M. Abdullah, A. Madain and Y. Jararweh, “ChatGPT: Fundamentals, Applications and Social Impacts,” *2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, Milan, Italy, 2022, pp. 1-8, doi: 10.1109/SNAMS58071.2022.10062688.

2. S. G. Santos and J. C. S. Cardoso, “Web-based Virtual Reality with A-Frame,” *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, Coimbra, Portugal, 2019, pp. 1-2, doi: 10.23919/CISTI.2019.8760795.

3. T. Yoshinaga. “Collaboration between ChatGPT API and A-Frame,” *Youtube*. 2023. Available: https://www.youtube.com/watch?v=FFQSKJUCBm0

4. S. Göring, R. R. Ramachandra Rao, R. Merten and A. Raake, “Appeal and quality assessment for AI-generated images,” *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*, Ghent, Belgium, 2023, pp. 115-118, doi: 10.1109/QoMEX58391.2023.10178486.

5. M. Cahyadi, M. Rafi, W. Shan, H. Lucky and J. V. Moniaga, “Accuracy and Fidelity Comparison of Luna and DALL-E 2 Diffusion-Based Image Generation Systems,” 2023 *IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, BALI, Indonesia, 2023, pp. 108-112, doi: 10.1109/IAICT59002.2023.10205695.

6. K. Suzuki, J. Cai, J. Li, T. Yamauchi, and K. Tei, “A Comparative Evaluation on Melody Generation of Large Language Models,” *2023 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, Busan, Korea, Republic of, 2023, pp. 1-4, doi: 10.1109/ICCE-Asia59966.2023.10326362.

7. M. Jang and T. Lukasiewicz, “Consistency Analysis of ChatGPT,” arXiv:2303.06273 [cs], Mar. 2023, Available: https://arxiv.org/abs/2303.06273

8. C. Li and B. Tang, “Research on Voice Interaction Technology in VR Environment,” *2019 International Conference on Electronic Engineering and Informatics (EEI)*, Nanjing, China, 2019, pp. 213-216, doi: 10.1109/EEI48997.2019.00053.

Thank you!

Questions?