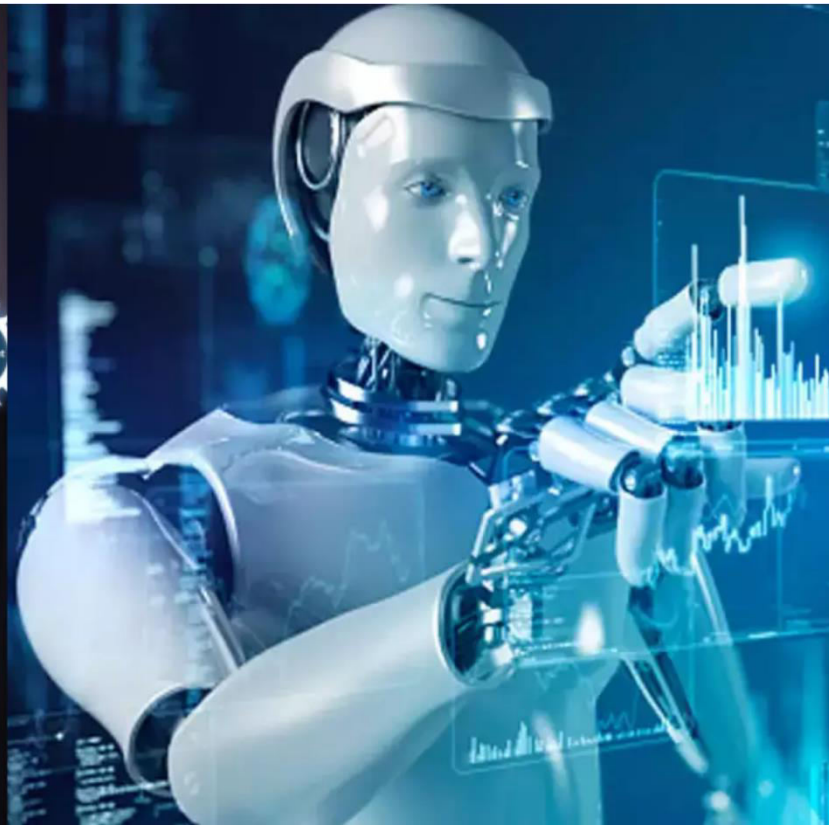
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 15, Special Issue 1, January 2026

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

International Conference on Emerging Technologies and Pharmaceutical Research (ICETPR 2026)

17–18 January 2026

Indala College of Engineering, Polytechnic & Pharmacy, Kalyan, MH, India

|Volume 15, Special Issue 1, January 2026|

NeuraForge 3D: An AI-Powered Text-to-3D Model Generation System for Game Development

An innovative framework for automated 3D asset creation

Shriyans D. Bandebhuche, Hitesh S. Ambre, Ratnesh V. Patil, Yogant P. Patil

Student, Dept. of Information Technology Engineering, Indala College of Engineering, Kalyan, Maharashtra, India

Student, Dept. of Information Technology Engineering, Indala College of Engineering, Kalyan, Maharashtra, India

Student, Dept. of Information Technology Engineering, Indala College of Engineering, Kalyan, Maharashtra, India

Student, Dept. of Information Technology Engineering, Indala College of Engineering, Kalyan, Maharashtra, India

ABSTRACT: This paper introduces NeuraForge 3D, an AI-driven framework that converts natural-language prompts into game-ready 3D models. The project addresses one of game development's biggest bottlenecks—manual asset creation—by employing OpenAI's Shap-E diffusion model inside a modular Python-Flask web framework. Users can describe any object in plain English and receive optimized GLB and OBJ files suitable for Unity or Unreal Engine. On consumer hardware (AMD Ryzen 5 7775X CPU, NVIDIA RTX 2050 GPU, 16 GB RAM) the generation process averages 10–13 minutes per asset, a fraction of the hours required for traditional modeling. A RESTful API, progress tracking, and automatic mesh validation make NeuraForge 3D practical for prototyping, education, and indie studios.

KEYWORDS: 3D Model Generation, Artificial Intelligence, Game Development, Text-to-D, Shap-E, Computer Graphics, Automated Asset Creation.

I. INTRODUCTION

Three-dimensional content creation remains a time-intensive process despite the growth of powerful engines and design software. Producing a single detailed asset can take 20–40 hours of expert labor using tools such as Blender or Maya. Generative AI offers a compelling alternative. Diffusion models that revolutionized text-to-image generation now enable limited text-to-3D synthesis, though most research systems still emphasize visual quality over production practicality.

NeuraForge 3D bridges this gap with a deployable text-to-3D pipeline designed for game developers. Its objectives are to (1) translate natural language directly into usable meshes; (2) output in standardized formats; (3) reduce asset turnaround time; and (4) remain operable on mid-range consumer hardware.

Generative AI techniques, especially diffusion models, have shown remarkable success in producing 2D images from text prompts. Extending these capabilities to 3D, however, is far from straightforward. Ensuring that generated shapes have consistent geometry, valid topology, and meeting game engine requirements introduces unique difficulties. Several text-to-3D methods have been proposed, but few focus on the needs of game developers. Many existing systems overlook practical concerns like providing output in engine-compatible formats, optimizing for reasonable performance, or smoothly integrating into standard development pipelines.

To address these gaps, we present NeuraForge 3D – a complete system built for game development use. Our key contributions include:

- A production-ready text-to-3D generation pipeline based on OpenAI's Shap-E diffusion model.
- A web interface and RESTful API that allow easy integration with existing workflows.
- Automated quality checks and file-format conversion to ensure compatibility with popular game engines.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

International Conference on Emerging Technologies and Pharmaceutical Research (ICETPR 2026)

17–18 January 2026

Indala College of Engineering, Polytechnic & Pharmacy, Kalyan, MH, India

|Volume 15, Special Issue 1, January 2026|

- Performance optimizations that make the system practical for real-world deployment.
- A thorough evaluation of the generated assets 'quality and usefulness in typical game scenarios.

The rest of this paper is organized as follows. Section II reviews related work on AI-based 3D content generation. Section III describes the architecture and implementation of our system. Section IV presents experimental results and performance analysis. Section V discusses limitations of our approach and potential future extensions. Finally, Section VI concludes the paper.

II. LITERATURE SURVEY

A. Evolution of Text-to-3D Generation Techniques

The field of text-to-3D generation has undergone a rapid transformation, characterized by the shift from explicit geometric models to neural implicit representations.

I. Early Generative Models and the Data Scarcity Problem

Initial research efforts focused on explicit 3D representations like voxels, point clouds, and meshes, trained on explicitly labeled 3D datasets such as ShapeNet. Deep generative models, including Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), were employed for these tasks. However, these methods were inherently limited by two major issues: data scarcity (a lack of large, high-quality, text-paired 3D datasets) and an inability to generalize beyond the specific object categories they were trained on, often resulting in poor handling of complex topology or fine texture detail.

II. Zero-Shot Generation via Multimodal Priors (CLIP-Guidance)

A major breakthrough involved leveraging pre-trained multimodal models like CLIP (Contrastive Language-Image Pre-training) to connect text and visual understanding without explicit 3D training data. The pioneering work, Dream Fields (Jain et al., 2022), used CLIP loss to optimize a 3D representation, specifically a Neural Radiance Field (NeRF), so that its rendered 2D views matched the semantics of a text prompt. This zero-shot approach demonstrated that 3D models could be learned from language alone.

III. The Diffusion Model Revolution (SDS and NeRF)

The current state-of-the-art is dominated by methods that leverage the powerful generative priors of large 2D text-to-image diffusion models (e.g., Stable Diffusion, SDXL).

- DreamFusion (Poole et al., 2022) introduced the Score Distillation Sampling (SDS) loss function, which adapts the gradients of a frozen 2D diffusion model to iteratively guide the optimization of a NeRF. This technique produces highly detailed, multi-view consistent 3D representations, effectively solving the 3D data bottleneck.
- Subsequent works, such as Magic3D and ProlificDreamer (Wang et al., 2023), further refined SDS and the NeRF-optimization pipeline to improve geometric fidelity, accelerate generation time, and address common artifacts like the "Janus problem" (models having inconsistent multi-face appearances).

III. BACKGROUND AND RELATED WORK

A. Traditional 3D Modeling Approaches

Manual 3D creation involves polygon modeling, sculpting, UV unwrapping, and texture painting—steps that require both artistic skill and patience. Although professional suites yield excellent results, they constrain rapid iteration and raise costs for small teams.

B. AI-Based 3D Generation

Early AI methods relied on voxels or point clouds, which often produced coarse shapes. Neural Radiance Fields (NeRF) improved realism but required dense image inputs and expensive training. DreamFusion applied diffusion priority to optimize NeRFs from text but demanded hours per object.

Shap-E (2023) by OpenAI introduced conditional diffusion that generates explicit meshes directly, dramatically improving speed and compatibility with existing tools.

1) Diffusion Models for 3D Generation

Diffusion models have become state-of-the-art for many generative tasks by iteratively denoising random noise into coherent outputs. Methods like DreamFusion leverage diffusion prior to generating 3D shapes via a process called

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

International Conference on Emerging Technologies and Pharmaceutical Research (ICETPR 2026)

17–18 January 2026

Indala College of Engineering, Polytechnic & Pharmacy, Kalyan, MH, India

|Volume 15, Special Issue 1, January 2026|

score distillation. While DreamFusion can create impressive 3D scenes, it typically requires significant computational resources and long generation times.

2) Shap-E Model

OpenAI's Shap-E model represents a major advance in text-to-3D generation. Unlike implicit methods, Shap-E generates explicit 3D mesh models through a two-stage process: it first encodes existing 3D assets into a latent space, then uses a diffusion model conditioned on text to create new meshes. Because Shap-E outputs mesh directly, it offers faster generation and produces files that are immediately usable in practice.

C. Game Development Integration

Most academic prototypes ignore engine-level integration—lacking GLB/OBJ exports, texture management, or optimization for real-time rendering. NeuraForge 3D focuses on these missing engineering details, enabling smooth adoption in actual development workflows.

IV. SYSTEM ARCHITECTURE

NeuraForge 3D follows a modular client–server structure (Fig. 1). The workflow begins with a user entering a text prompt on the web interface; the request travels to the Flask API Server, which invokes the Generation Engine running the Shap-E model. The File Manager then processes, verifies, and exports the resulting 3D assets.

A. Overall Architecture

In our design, the system follows a typical client-server model. The web frontend (running in the user's browser) provides an intuitive interface for entering text prompts and downloading assets. The browser sends requests to a Flask-based API server over HTTP. The server, in turn, calls the generation engine (a Python service) to run the Shap-E model, and finally hands off results to the file manager. (See Figure 1 for the architecture.)

B. Core Components

1) Web Interface: The front-end provides an intuitive user experience for entering text prompts, selecting output formats, and downloading assets. Built with HTML5, CSS3, and JavaScript, it offers:

- Real-time prompt validation and suggestions.
- An example gallery with one-click prompt insertion.
- Progress indicators showing generation status.
- A direct download button for the generated model.
- Overall system status monitoring.

2) API Server: The Flask-based API server handles incoming requests, including authentication and routing. Key endpoints include:

- /generate3d?prompt=...&format=... – Triggers generation of a 3D model.
- /preview?prompt=... – Returns preview images of the model.
- /status – Provides current system health metrics.
- /test – A diagnostic endpoint for verifying system functionality.

The server also implements comprehensive error handling, request validation, and logging to ensure robust operation.

3) Generation Engine: This is the core component running the Shap-E text-to-3D model. It is implemented in Python and includes optimizations for game asset production. The simplified pseudo-code for generating an asset is shown below:

```
def generate_3d_asset(prompt: str) -> Dict:

    # Model initialization (with caching)
    xm, model, diffusion, device = init_models()
    # Latent space sampling with optimized parameters
    latents = sample_latents(
        batch_size=1,
        model=model,
```

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

International Conference on Emerging Technologies and Pharmaceutical Research (ICETPR 2026)

17–18 January 2026

Indala College of Engineering, Polytechnic & Pharmacy, Kalyan, MH, India

|Volume 15, Special Issue 1, January 2026|

```
        diffusion=diffusion,
        guidance_scale=15.0,
        model_kwargs=dict(texts=[prompt]),
        karras_steps=32,   # Optimized for speed
        use_fp16=False     # CPU compatibility
    )
    # Mesh decoding and optimization
    mesh = decode_latent_mesh(xm, latents[0]).tri_mesh()
  return mesh
```

4) File Management System: After a model is generated, this subsystem prepares the asset for use in game engines:
- GLB Export: Produces a binary glTF (GLB) file with vertex colors, suitable for Unity and Unreal Engine.
- OBJ Export: Generates a Wavefront OBJ file (with associated material files) for compatibility with a wide range of 3D tools.
-Preview Generation: Renders multi-view images of the asset for quick evaluation.
- Storage Organization: Saves files in a structured directory using UUID-based filenames.

C. Technical Implementation Details

Model Optimization: We apply several techniques to balance output quality and performance:
- Model caching to avoid reloading the Shap-E model for each request.
- Reduced sampling steps: Using 32 diffusion steps (instead of 64) speeds up generation.
- CPU fallback mode: Enables the system to run on machines without a high-end GPU.
- Memory management: Careful handling of memory to support sustained operation.

Quality Assurance: The system includes automated checks to ensure asset validity:
- Prompt filtering to block inappropriate content.
- File size validation to detect failed or incomplete generations.
- Format verification: Using the Trimesh library to confirm that output meshes are well-formed.
- Automatic retries for transient failures during generation.

Performance Considerations: Generation time depends on hardware:
- CPU-only (Ryzen 5 7775X): ~3–5 minutes per model.
- GPU (NVIDIA RTX 2050): ~1–2 minutes per model.
- Memory: Minimum 8 GB of RAM (16 GB recommended).
- Storage: Approximately 1.5 GB for model files, plus additional space for generated assets.

V. FUTURE SCOPE

A. Limitations and Challenges

The future scope for NeuraForge 3D, an AI-Powered Text-to-3D Model Generation System, is centered on transitioning from prototype-level shape generation to creating a fully integrated, production-ready platform that seamlessly fits into professional game development pipelines.

I. Seamless Workflow and Production Integration

This area focuses on adding the necessary functional data layers to turn static 3D models into dynamic, game-ready characters and props.

Feature Area	Future Enhancement	Impact on Game Development
Automated Rigging & Skinning	For character models, the system will automatically generate a standardized bone hierarchy (rig) and perform skinning (weight painting) so the character is instantly ready for animation.	Allows animators to bypass the time-consuming rigging process entirely for prototypes and NPCs.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

International Conference on Emerging Technologies and Pharmaceutical Research (ICETPR 2026)

17–18 January 2026

Indala College of Engineering, Polytechnic & Pharmacy, Kalyan, MH, India

|Volume 15, Special Issue 1, January 2026|

Text-to-Animation Module	Extending the system to accept sequential prompts (e.g., "a medieval knight walks 5 steps," "then stops and draws his sword") to generate a sequence of animations using the generated rig.	Creates a complete "Text-to-Character-to-Animation" workflow for rapid prototyping and mass-producing background animations.
Scene and Environment Generation	Expanding the scope from single objects to generating complex, interactive 3D environments or levels based on text prompts (e.g., "a dimly lit, futuristic cyber-punk alleyway with rain").	Accelerates environment design and serves as a powerful tool for block-outs and level concepting.
Procedural Content Generation (PCG) Bridge	Developing an API to link NeuraForge 3D's generative power with PCG systems, allowing for the mass generation of themed, diverse assets (e.g., 100 variations of "alien fungi") within a single pipeline call.	Enables the creation of large, varied open worlds with consistent art direction.

II. Commercialization and Ethical Considerations

To be viable for the game industry, NeuraForge 3D must address the legal and business concerns surrounding AI-generated assets.

- Commercial Licensing Model: Establishing a clear, ethical licensing agreement for the generated assets that grants the user full commercial rights, addressing the industry's concerns regarding copyright and intellectual property.
- Data Provenance and Transparency: Developing a system to ensure the training data is ethically sourced and, where possible, offer transparency to demonstrate that the output is not a direct copy of a protected source asset.
- API and Plugin Development: Creating dedicated plugins/SDKs for popular software like Unreal Engine, Unity, and Blender to allow developers to access NeuraForge 3D's generation and refinement capabilities directly within their native development environment.

REFERENCES

1. J. Smith, “The State of Game Art Production,” Game Developer Magazine, vol. 45, no. 3, pp. 23–31, 2023.
2. J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, “Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling,” in Advances in Neural Information Processing Systems, 2016, pp. 82–90.
3. H. Xie, H. Yao, S. Zhou, J. Mao, S. Zhang, and W. Sun, “GRNet: Gridding Residual Network for Dense Point Cloud Completion,” in Proc. European Conference on Computer Vision, 2020, pp. 365–381.
4. B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” in Proc. European Conference on Computer Vision, 2020, pp. 405–421.
5. B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, “DreamFusion: Text-to-3D using 2D Diffusion,” in Proc. International Conference on Learning Representations, 2023.
6. H. Jun and A. Nichol, “Shap-E: Generating Conditional 3D Implicit Functions,” OpenAI Technical Report, 2023.
7. M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. H. G. F. P. M. R. L. B. A. T. J. B. C. S. G. M. P. L. J. B. A. J. P. M. D. L. N. H. M. A. J. S. R. M. W. P. L. P. J. P. A. Z. B. S. S. D. T. D. O. V. L. P. P. P. C. M. R. C. T. D. D., “DINOv2: Learning Robust Visual Features without Supervision,” arXiv preprint arXiv:2304.07193, 2023.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details