# A Simulation of Voyager 1's Path Throughout The Solar System


By:

Mohammed Ali Ahmed Al Zubaidi 202001259
Ahmed Sidi Mohammed Ekah 202219649
Hazza Obaid Sultan Al Kaabi 202113348


Smart Computer Graphics (01)
Instructor: Dr. Rafat Damseh

## Introduction

This project is a simulation of Voyager 1's journey throughout the solar system, developed using Unity. The primary goal is to recreate the spacecraft's trajectory and visualize its path from launch to the final path it took out of the solar system after its orbital encounter with Saturn. By combining accurate astronomical data with 3D rendering, the aim of the project is to provide an engaging and educational representation of one of humanity's most significant space missions into interstellar space.

The scope of the project encompasses modeling the solar system, simulating realistic motion physics, and implementing an interactive camera system that enhances immersion in the simulated space. The camera system may allow users to explore Voyager 1's perspective or observe planetary alignments during orbital encounters.

This project is highly relevant to smart computer graphics as it integrates real-world data with visualization and represenational techniques. It demonstrates how graphical techniques (procedural rendering, optimized shaders, and data-driven animation) can bring scientific exploration to life in an interactive virtual environment. By using Unity's capabilities and modern graphics concepts, the simulation serves as both a technical exercise for the project team members, as well as a compelling example of how smart computer graphics can be used for education and storytelling.

## Methodology

The development of the Voyager 1 Solar System Simulation was carried out using the Unity game engine, with a focus on scientific accuracy, visual clarity, and computational efficiency. The methodology combines real astronomical data, physically-based motion calculations, and optimized rendering techniques to recreate Voyager 1's trajectory through the solar system.

### 1. Project Architecture

The project is organized into several core modules to ensure modularity and ease of maintenance:

**Data Module:**
This module contains a static .json file storing real-world astronomical values for celestial bodies ranging from the Sun to Saturn. These include parameters such as mass, radius, and orbital distance. The data are parsed at runtime to initialize the simulation's planetary models and physics properties.

**3D Models and Textures:**
This category includes prefabricated assets for all major planets, each with realistic textures and shaders to approximate their visual appearance. A real-star skybox was incorporated to simulate the spatial context of deep space and enhance immersion.

**Scripts (C#):**
The simulation's logic is handled through several custom C# scripts, each responsible for a specific subsystem:

**Body.cs**: Initializes the physical and visual properties of planetary and solar bodies.

**PhysicsIntegrator.cs**: Handles gravitational force computations and numerical integration using a semi-implicit Euler method with sub-stepping for improved stability. The time step is scaled dynamically to maintain accurate motion within Unity's coordinate space.

**SimulationManager.cs**: Acts as the core controller, initializing the simulation, coordinating updates, and managing script interactions.

**SolarSystem.cs**: Oversees the management of all celestial bodies, including loading planet data from JSON, spawning objects, and updating their positions in real time.

**VoyagerController.cs**: Simulates Voyager 1's spacecraft trajectory, calculating initial launch conditions, velocity vectors, and trajectory visualization. It implements deflection maneuvers for the Saturn flyby and computes the required velocity for the Jupiter encounter.

**UnitScale.cs**: Manages unit conversion between astronomical units and Unity's scaled coordinate system to ensure physical consistency.

**CameraFollow.cs** and **CameraDirector.cs**: Provide cinematic camera movement, following Voyager 1 dynamically to highlight key mission phases.

## 2. Technologies and Tools

The simulation was built using Unity, chosen for its real-time rendering capabilities and user-friendly interface. All programming was performed in C#, leveraging Unity's robust API and object-oriented design principles.

Planetary textures and 3D models were sourced from publicly available repositories, while astronomical data were extracted from the JSON file. Additional Unity packages, such as the built-in skybox system and standard shader tools, were utilized to achieve realistic environmental lighting and surface materials.

## 3. Simulation Details

Voyager 1's path was modeled using realistic physical conditions. The spacecraft's initial launch conditions were established relative to Earth, with computed velocity vectors determining its trajectory toward Jupiter. Gravitational influences from Jupiter and Saturn were modeled to reproduce accurate deflection maneuvers. The simulation uses simplified Newtonian gravity to balance computational load with visual fidelity.

### 4. Graphical Techniques

The simulation's visual presentation emphasizes both realism and performance. A real-star skybox provides a scientifically grounded background, while the Sun's glare adds visual depth through a lens-flare effect. Planetary bodies are textured using real-world surface maps, applied to spherical meshes for efficient rendering.

### 5. Reason for Design Choices

Unity was selected due to its accessibility, cross-platform support, and powerful built-in graphics engine, making it ideal for educational simulations. The use of C# was motivated by its similarity to Java and its strong integration with Unity's object-oriented architecture. The chosen graphical techniques (texture-mapped spherical planets and simple glare effects) prioritize efficiency and clarity over complex rendering, aligning with the project's goal of demonstrating foundational skills in smart computer graphics rather than high-fidelity cinematic realism.

## Results

The Voyager 1 Solar System Simulation successfully met its primary objective: creating a three-dimensional visualization of Voyager 1's trajectory through the solar system using real astronomical data. The simulation demonstrates how smart computer graphics can combine data-driven modeling, physics integration, and 3D rendering to represent a real scientific mission within an interactive environment.

### Achievements

The project implemented a functional 3D solar system where Voyager 1's launch, Jupiter flyby, and Saturn deflection are accurately represented. Realistic planetary models, a dynamic camera system, and stable gravitational physics contribute to an engaging visual experience. The simulation incorporated real planetary parameters from a JSON dataset and used a semi-implicit Euler integrator for stable motion calculations.

### Comparison with Real-World Data

To assess accuracy, simulated values were compared to the actual Voyager 1 mission data. While most results closely matched real values, small deviations occurred due to simplified physics and scaled units within Unity, as well as the fact that in real life; inteplanetary objects influence eachother gravitationally:

| Parameter | Real | Simulated | Difference |
|---|---|---|---|
| Launch Delta-V | ~17.0 km/s | 12.54 km/s | -26.2% |
| Jupiter Distance | 778M km | 778.57M km | +0.07% |
| Saturn Distance | 1.43B km | 1.433B km | +0.2% |
| Jupiter Approach Speed | ~19 km/s | 19.62 km/s | +3.3% |
| Jupiter Exit Speed | ~24 km/s | 24.81 km/s | +3.4% |

| Parameter | Real | Simulated | Difference |
|---|---|---|---|
| Saturn Approach Speed | ~21 km/s | 21.52 km/s | +2.5% |
| Saturn Exit Speed | 21 km/s | 21.01 km/s | +0.05% |
| Jupiter Deflection | ~21 degrees | ~21.8 degrees | +3.8% |
| Saturn Deflection | ~35 degrees | 34.1 degrees | -2.6% |

These results show that while trajectory paths and encounter speeds were consistent within a small margin of error (mostly below 5%), the launch velocity differed more significantly due to scaling adjustments and simplified gravitational modeling.

## Challenges

Key challenges included scaling astronomical distances to Unity's coordinate system without losing numerical precision and achieving stable gravitational physics under large mass and distance variations. Time constraints also limited the inclusion of features such as solar illumination with realistic shadows, additional moons like Titan which Voyager 1 encountered, and a more accurately scaled solar system to better convey interplanetary distances and sizes.

## Summary

Overall, the project achieved its objectives of accurately visualizing Voyager 1's path using smart computer graphics techniques. Although simplified, the simulation provides a credible and educational depiction of the mission's physics and scale. Future work could enhance realism through full solar system scaling, improved lighting, and additional celestial bodies to more closely mirror the true complexity of Voyager 1's journey.