Pathfinder3D: Optimized 3D Pathfinding Module for Roblox

This module provides a highly optimized, custom 3D pathfinding solution for Roblox, designed for interactive and dynamic environments. It leverages a **Sparse Voxel Octree (SVO)** for efficient global path planning and incorporates **Anticipatory Collision Avoidance (ACA)** for local steering and dynamic obstacle handling.

This implementation is a proof-of-concept based on the principles outlined in *Game AI Pro 3, Chapter 21: 3D Flight Navigation Using Sparse Voxel Octrees* and *Game AI Pro 2, Chapter 19: Guide to Anticipatory Collision Avoidance*.

Core Components

1. **Sparse Voxel Octree (SVO):** Used as the graph representation for the 3D world. It efficiently stores free space, allowing for fast A^* search.
2. **Modified A* Search:** A standard A* algorithm adapted to traverse the SVO graph, including the ability to "jump" between octree levels for faster pathfinding across large, open spaces.
3. **Anticipatory Collision Avoidance (ACA):** A local steering function that computes a safe velocity to avoid dynamic obstacles (other agents, moving parts) in real-time.

Module API

The module is intended to be used as a Roblox `ModuleScript`.

`Pathfinder3D.new(configTable)`

The constructor initializes the pathfinding system by building the SVO.

Parameter	Type	Description
`configTable`	`table`	Optional. Overrides default configuration settings.

Example Configuration:

Lua

```
local Pathfinder3D = require(game.ReplicatedStorage.Pathfinder3D)

local Pathfinder = Pathfinder3D.new({
    WORLD_BOUNDS_MIN = Vector3.new(-1000, -1000, -1000),
    WORLD_BOUNDS_MAX = Vector3.new(1000, 1000, 1000),
    MAX_OCTREE_DEPTH = 8, -- Determines finest resolution (2^8 = 256
divisions)
    MIN_VOXEL_SIZE = 4,   -- Smallest traversable voxel size in studs
    AGENT_RADIUS = 2,     -- Radius of the agent for collision avoidance
    MAX_SPEED = 15,       -- Maximum speed for steering
})
```

Pathfinder3D:ComputePath(startPos, endPos)

Computes the global path between two points using A* on the SVO.

Parameter	Type	Description
`startPos`	`Vector3`	The starting world position.
`endPos`	`Vector3`	The target world position.
Returns	`table`	A list of `Vector3` waypoints, or `nil` if no path is found.

Pathfinder3D:GetSteeringVelocity(currentPos, currentVel, targetPos, obstacles)

Computes a safe, collision-free velocity vector for the agent to follow the path while avoiding dynamic obstacles. This function should be called every frame (e.g., in `RunService.Heartbeat`).

Parameter	Type	Description
`currentPos`	`Vector3`	Agent's current world position.
`currentVel`	`Vector3`	Agent's current velocity.
`targetPos`	`Vector3`	The next waypoint from the path returned by `ComputePath`.
`obstacles`	`table`	A list of nearby dynamic obstacles. Each obstacle is a table: `{Pos = Vector3, Vel = Vector3, Radius = number}`.
Returns	`Vector3`	The new, safe velocity vector for the agent.

Critical Optimization Note: SVO Implementation

The provided `Pathfinder3D.lua` contains **mock implementations** for the SVO generation (`BuildSVO`) and neighbor finding (`GetNeighbors`). These mocks are $O(N)$ and are only for conceptual demonstration.

For a truly optimized, production-ready module, you MUST replace the mock functions with a bitwise/Morton-code-based SVO implementation.

- **Optimized SVO Generation:** Should use raycasting or spatial queries to efficiently determine which voxels are free space, and store the SVO nodes in a highly compact, flattened array/table structure using **Morton Codes** for spatial coherence.
- **Optimized Neighbor Finding:** Should use bitwise operations on the Morton Codes to find neighbors in $O(\log N)$ time, which is the key to the SVO's performance advantage.

The current implementation provides the correct A* and ACA logic, but the SVO foundation must be optimized in Luau for the performance you require.

Dependencies

This module requires a performant **Min-Heap** implementation for the A* algorithm. The provided `Heap.lua` file should be placed alongside `Pathfinder3D.lua` and required by it.