AtomBlaster AI Helicopter System

This implementation adds intelligent AI opponents to the AtomBlaster helicopter game. The AI system features helicopters with diverse personalities, adaptive behaviors, and realistic interactions with the player and game world.

Overview

The AI system consists of several key components:

1. **AIHelicopter Entity**: An extension of the player helicopter with AI-specific properties
2. **AIManager**: Centralized system for creating, updating, and managing all AI helicopters
3. **Integration Files**: Updates to the game loop, collision system, and UI for AI integration

AI Helicopter Types

Five distinct AI personalities are implemented, each with unique behaviors:

1. **Aggressive**: Actively hunts the player when bigger, prioritizes combat over food collection, and has enhanced speed and attack capabilities.
2. **Collector**: Focuses primarily on gathering food, avoids confrontation, and has enhanced growth from food.
3. **Cautious**: Prioritizes survival, flees from larger entities, and has enhanced speed for escape.
4. **Opportunist**: Balances food collection with strategic attacks, evaluates risk vs. reward.
5. **Wanderer**: Moves unpredictably, has random behavior patterns, and may unexpectedly engage or flee.

AI Behavior System

AI behaviors are managed through a state machine with five states:

1. **Wandering**: Random exploration with personality-specific movement patterns
2. **Chasing**: Actively pursuing the player when advantageous
3. **Fleeing**: Escaping from threats when outmatched
4. **Collecting**: Seeking and gathering food efficiently
5. **Hunting**: Looking specifically for power-ups

The AI dynamically switches between these states based on:

- Current game situation (nearby entities, size differences)
- AI's personality type
- Random factors for unpredictability

- Player proximity and actions

Performance Optimizations

The system includes several optimizations for handling large numbers of AI opponents:

1. **Activation Range**: Full AI update only runs when within a certain distance from player
2. **Quadtree Integration**: Efficient spatial partitioning for collision detection
3. **Simplified Physics**: Reduced update frequency for distant AI entities
4. **Object Pooling**: Efficient memory management for AI creation/destruction

Installation

To integrate the AI system into your existing codebase:

1. Add the new files:
   - `entities/ai_helicopter.go`
   - `systems/ai_manager.go`
2. Update existing files:
   - `game/game_state.go`: Add AIManager field and integrate into update/draw loops
   - `ui/minimap.go`: Update to show AI helicopters
   - Add constants to `constants.go`
3. Modify collision system to handle AI interactions

Usage

The AI system is designed to be largely self-managing. Once integrated, the game will:

1. Automatically spawn initial AI helicopters with diverse personalities
2. Gradually increase AI count and difficulty as the player's score increases
3. Handle all collisions, food gathering, and power-up effects

Customization Options

Several parameters can be adjusted to fine-tune the AI behavior:

1. **Maximum AI Count**: Adjust `MaxAICount` in constants.go
2. **Difficulty Scaling**: Modify `DifficultyLevel` calculation in `updateDifficulty()`
3. **Personality Distribution**: Change probability weights in AIManager's `SpawnAI()` method
4. **Behavior Parameters**: Adjust state durations and decision weights in `chooseNewState()`

Debug Features

When debug mode is active, AI information is displayed:

- Total AI count and current difficulty level
- Nearest AI type and current state
- Distance to nearest AI
- Visual representation of AI vision ranges and states

Future Enhancements

Potential improvements for future updates:

1. **Group Behaviors**: Allow AIs to form temporary alliances against larger opponents
2. **Learning Patterns**: Implement basic learning to adapt to player strategies
3. **Terrain Awareness**: Add pathfinding around obstacles when world complexity increases
4. **Specialized Abilities**: Give each personality type unique special abilities
5. **Advanced Tactics**: Implement ambush behaviors, trap setting, and strategic retreats

Integration Notes

The AI system is designed to integrate seamlessly with the existing AtomBlaster architecture, maintaining the same code style and performance standards. The implementation extends rather than modifies core game mechanics.