AI Patrol System Tutorial for Unreal Engine

Author: Yuxiao Lin Date: 2025/12/9 Unreal Version: 5.0+

TABLE OF CONTENTS

1. Introduction
2. Learning Objectives
3. Technical Overview
4. Implementation Guide
5. Practice Exercises
6. Troubleshooting
7. Summary

---

1. INTRODUCTION

What is AI Patrol?

AI Patrol is a fundamental behavior pattern where non-player characters (NPCs) automatically navigate within designated areas. This technique is essential in modern games for creating dynamic, believable environments.

Real-World Applications

- RPG games: Town guards patrolling streets
- Stealth games: Enemy sentries on patrol routes
- Open-world games: NPCs wandering in cities
- Horror games: Monsters searching for players

What You Will Build

In this tutorial, you will create an AI character that:

- Automatically starts patrolling on game begin
- Randomly selects reachable locations within a radius
- Moves autonomously to each target

- Visualizes targets with debug spheres
- Loops indefinitely

---

2. LEARNING OBJECTIVES

By completing this tutorial, you will:

- Understand Navigation Mesh (NavMesh) systems
- Master the AI Move To node for pathfinding
- Implement looping AI behaviors
- Use debug visualization for AI development
- Apply navigation systems to game projects

Target Audience: Beginner to intermediate Unreal Engine developers with basic Blueprint knowledge.

---

3. TECHNICAL OVERVIEW

3.1 Navigation Mesh

Navigation Mesh (NavMesh) is a data structure representing walkable surfaces in your game world. Unreal Engine automatically generates this mesh based on level geometry.

Key Concepts:

- NavMesh Bounds Volume defines the navigation area
- Press P key in viewport to visualize (green = walkable)
- AI can only move on NavMesh surfaces

3.2 Random Pathfinding

The system uses "Get Random Reachable Point in Radius" to find valid patrol locations.

Parameters:

- Origin: Starting point (AI's current location)
- Radius: Search distance (in centimeters)
- Returns: Random walkable position on NavMesh

3.3 AI Movement

AI Move To node handles pathfinding and movement.

Key Features:

- Automatically calculates path around obstacles
- Triggers events on success or failure
- Supports dynamic replanning

3.4 Loop Pattern

The patrol loop structure: Start → Move to Target → Arrive → Delay → Find New Target → Repeat

Implementation uses Custom Events for clean recursive behavior.

---

4. IMPLEMENTATION GUIDE

4.1 Project Setup

Step 1: Create New Project

- Template: Third Person or Top Down
- Type: Blueprint
- Name: AI_Patrol_System

Step 2: Setup Scene

- Add ground plane
- Place obstacles (Cubes)
- Ensure obstacles have collision (BlockAll)

4.2 Navigation Setup

Step 3: Add Nav Mesh Bounds Volume

- Place Actors panel → Search "Nav Mesh Bounds Volume"
- Drag into scene
- Scale to cover entire area: (20, 20, 10)
- Press P to verify green mesh appears

4.3 Create AI Character

Step 4: Create Character Blueprint

- Content Browser → Blueprint Class → Character
- Name: BP_AutoMovingCharacter
- Add skeletal mesh (use template Mannequin)

4.4 Implement Patrol Logic

Step 5: Blueprint Implementation

Event Graph structure:

A. Event BeginPlay Connect to: AutoMoving custom event

B. AutoMoving (Custom Event)

1. Get Actor Location (Target: Self)
2. Get Random Reachable Point in Radius
    - Origin: Actor Location
    - Radius: 1000.0
3. SET Target Location (create variable: Vector)
4. Draw Debug Sphere
    - Center: GET Target Location
    - Radius: 100.0
    - Color: Red (R=1, G=0, B=0, A=1)
    - Duration: 5.0
5. AI Move To
    - Pawn: Self
    - Destination: GET Target Location
6. On Success → Delay (1.0)
7. Delay → Delay (0.1)
8. Delay → AutoMoving (creates loop)

Important Notes:

- Store position in variable to ensure sphere and movement use same location
- Use two delays to control timing and prevent rapid loops
- Self reference moves the AI character itself

4.5 Testing

Step 6: Place and Test

- Drag BP_AutoMovingCharacter into level
- Place on NavMesh area
- Click Play
- Observe: AI moves, red spheres appear at targets, loop continues

Step 7: Verify Navigation

- Press P to see green NavMesh
- Confirm AI stays within walkable areas
- Check obstacle avoidance

---

5. PRACTICE EXERCISES

Exercise 1: Modify Patrol Range (Basic)

Task: Change the patrol radius

- Find "Get Random Reachable Point in Radius" node
- Change Radius from 1000.0 to 2000.0
- Test and observe larger patrol area

Exercise 2: Adjust Speed (Basic)

Task: Change AI movement speed

- Select BP_AutoMovingCharacter
- Components → Character Movement
- Modify Max Walk Speed (default 600)
- Try: 300 (slow) or 1200 (fast)

Exercise 3: Change Wait Time (Basic)

Task: Modify pause duration

- Find first Delay node after On Success
- Change Duration from 1.0 to 3.0
- AI now waits 3 seconds at each target

Exercise 4: Waypoint Patrol (Intermediate)

Task: Create fixed patrol points

- Create array variable: PatrolPoints (Vector array)
- Add 3 fixed positions
- Create integer variable: CurrentIndex
- Replace random point with PatrolPoints[CurrentIndex]
- Increment index in loop: (CurrentIndex + 1) % Array Length

Exercise 5: Add Path Visualization (Intermediate)

Task: Draw line from AI to target

- Add Draw Debug Line node after Draw Debug Sphere
- Line Start: Get Actor Location
- Line End: GET Target Location
- Color: Green, Duration: 5.0, Thickness: 5.0

---

6. TROUBLESHOOTING

Problem 1: AI Does Not Move

Causes and Solutions:

- No NavMesh visible: Press P, add/scale Nav Mesh Bounds Volume
- AI outside NavMesh: Move AI to green area
- Pawn not connected: Verify AI Move To Pawn pin connects to Self
- Radius too small: Increase search radius

Problem 2: AI Walks Through Walls

Causes and Solutions:

- No collision on walls: Select wall → Collision Presets → BlockAll
- NavMesh penetrates walls: Rebuild navigation (Build → Build Paths)

Problem 3: Sphere Position Wrong

Cause: Sphere and AI Move To use different positions

Solution:

- Create Target Location variable
- Use SET to store Random Location
- Use GET for both sphere Center and movement Destination

Problem 4: No Loop

Causes and Solutions:

- Loop not connected: Verify Delay connects to AutoMoving
- On Success not firing: Add Print String to debug

Problem 5: Sphere Not Visible

Causes and Solutions:

- Center not connected: Connect GET Target Location
- Duration too short: Increase to 5.0
- Sphere too small: Increase Radius to 200
- Wrong camera angle: Adjust view

---

7. SUMMARY

What You Learned

- Navigation Mesh setup and visualization
- Random pathfinding implementation
- AI Move To for autonomous navigation
- Loop behavior patterns using Custom Events
- Debug visualization techniques

Key Takeaways

1. NavMesh is essential for AI navigation
2. Store positions in variables for consistency
3. AI Move To handles pathfinding automatically
4. Debug tools are critical for AI development
5. Custom Events enable clean looping behaviors

Next Steps

- Complete all practice exercises
- Integrate with Behavior Trees
- Add AI Perception for player detection
- Explore Environment Query System (EQS)
- Apply to your game projects

System Architecture

The patrol system consists of:

- Navigation System (NavMesh generation and queries)
- Movement System (AI Move To and path following)
- Control Logic (Custom Event loop)
- Visualization (Debug drawing)

Performance Considerations

- NavMesh queries are relatively inexpensive
- Limit debug drawing in shipping builds
- Consider query frequency with many AI
- Use appropriate NavMesh cell size for precision vs performance

---

APPENDIX: KEY NODES REFERENCE

Get Random Reachable Point in Radius

- Purpose: Find random valid position on NavMesh
- Inputs: Origin (Vector), Radius (Float)

- Outputs: Random Location (Vector), Success (Boolean)

AI Move To

- Purpose: Move AI character to target location
- Inputs: Pawn (Pawn reference), Destination (Vector)
- Outputs: On Success, On Fail, Movement Result

Draw Debug Sphere

- Purpose: Visualize positions in 3D space
- Inputs: Center (Vector), Radius (Float), Color, Duration
- Use: Development and debugging only

Custom Event

- Purpose: Create reusable event sequences
- Use: Encapsulate logic, enable recursion for loops

Delay

- Purpose: Add time pause in execution flow
- Input: Duration (Float in seconds)
- Output: Completed (fires after duration)

VALIDATION CHECKLIST

Project Complete When:

- AI autonomously patrols scene
- NavMesh visible with P key
- Red spheres mark targets
- AI avoids obstacles
- System loops indefinitely
- At least 2 exercises completed

END OF TUTORIAL

For questions or issues, refer to Troubleshooting section or contact instructor.
