# Artificial Intelligence for Game Developers

(10 Week Lesson Plan)

## Lesson 1: Introduction to AI and Pathfinding

**Textbook:** pgs. 1 – 21 (Chapter 1)

**Goals:**

In this lesson, students are introduced to the initial concepts of AI and pathfinding. We'll begin with a few guidelines for AI in games, and delve into some quick descriptions of different types of AI that exist. We will then move on to one of the fundamental types of AI: pathfinding. We will explore the different types, and work through some specific examples of these types.

**Key Topics:**

General Principles of AI
- The KISS Method
- Hard != Fun
- Play Fair

Types of AI
- Pathfinding
- Decision Making
- Classification
- Life Systems

Types of Pathfinding
- Non-Look-Ahead Iterative Traversals
- Look-Ahead Iterative Traversals
- Look-Ahead Recursive Traversals

Non-Look-Ahead Iterative Traversals
- Random Backstepping
- Obstacle Tracing

**Projects:**

*Lab Project 1.1: Pathfinding*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 10 - 12

## Lesson 2: Look-Ahead Iterative Methods of Pathfinding

**Textbook:** pgs. 21 – 55 (Chapter 1)

**Goals:**

In this lesson, students will continue their basic training in the core types of pathfinding: Look-Ahead Iterative traversals. We will cover the common types of traversals including Breadth First Search, Best First Search, and Dijkstra's search. We will also discuss the Depth First Search.

**Key Topics:**

Look-Ahead Iterative Traversals

- Breadth First Search
- Best First Search
- Dijkstra's Search

Look-Ahead Recursive Traversals

- Depth First Search

**Projects:**

*Lab Project 1.1: Pathfinding (cont.)*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 10 - 12

## Lesson 3: Data Structures

**Textbook:** pgs. 1 – 25 (Chapter 2)

**Goals:**

In this lesson, students are introduced to the famous A* search algorithm. We will talk about how it works, its limitations, and how to make it more efficient. We will discuss heuristics, as well as how we might apply A* to a simple RTS type game. Finally, we will discuss Hierarchical pathfinding, and look at some examples of how it can be used.

**Key Topics:**

A* Search

- How A* works.
- Limitations of A*
- Making A* more efficient
- Heuristics

Hierarchical Pathfinding

**Projects:**

*Lab Project 1.1: Pathfinding (cont.)*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 8 - 10

## Lesson 4: Non Gridded Pathfinding and The Pathfinding Demo

**Textbook:** pgs. 25 – 39 (Chapter 2)

**Goals:**

In this lesson, students will be exposed to the concept of pathfinding on Non-Gridded maps. Methods for dealing with non-gridded worlds will be discussed as well. Finally, the student will examine the design strategy used for the Pathfinding Demo, and learn how the interface could be extended for their own use.

**Key Topics:**

Pathfinding on Non-Gridded Maps

- Grid It
- Visibility Points / Waypoint Networks
- Radial Basis
- Cost Fields
- Quad-Trees
- Mesh Navigation

Pathfinding Demo Design Strategy

- Class Hierarchy
- Interfaces
- MFC Document/View Architecture

**Projects:**

*Lab Project 1.1: Pathfinding (cont.)*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 12 - 15

## Lesson 5: Flocking

**Textbook:** pgs. 1 – 46 (Chapter 3)

**Goals:**

In this lesson, we will start to move from AI Pathfinding to AI Decision making. We will discuss behavior based movement, and how to apply it in the case of the traditional implementation of Flocking.

**Key Topics:**

Behavior Based Movement

- Common Flocking Behaviors
  - Separation
  - Cohesion
  - Alignment
  - Avoidance
- Other Possible Behaviors

The Flocking Demo

- Design Strategies
- Our Behaviors
  - Separation
  - Cohesion
  - Alignment
  - Avoidance
  - Cruising
  - Stay Within Sphere

**Projects:**

*Lab Project 5.1: Flocking*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 15 - 18

## Lesson 6: Decision Making

**Textbook:** pgs. 1 – 24 (Chapter 4)

**Goals:**

In this lesson, students will continue their training in decision making started in the last lesson, by learning about the different types of decision making commonly used in AI systems. We will discuss one of the most popular types of decision making systems: state machines. This will include an examination of transition diagrams, uses for state machines, and an example implementation.

**Key Topics:**

Decision Making
- Decision Trees
- State Machines
- Rule Base
- Squad Behaviors

Finite State Machines
- Transition Diagrams
- Uses for State Machines
    - Animation
    - Game State
    - Save File System
    - A.I.

State Machine Demo
- Design Strategy

**Projects:**

*Lab Project 6.1: State Machines*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 8 - 10

## Lesson 7: Scripting

**Textbook:** pgs. 24 – 47 (Chapter 4)

**Goals:**

In this lesson, we will take a look at how to add scripting to our games. We'll discuss a popular scripting language called Python and learn we can integrate it into our state machine developed in the last lesson.

**Key Topics:**

Scripting

- Intro to Python
  - Default Types and Built-ins
  - Classes
  - Functions
  - Control Statements
  - Importing Packages
- Embedding Python
  - Boost.Python
  - Our Scripting Engine

**Projects:**

*Lab Project 7.1: Scripted State Machines*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 8 - 10

## Lesson 8: Waypoint Networks

**Textbook:** pgs. 1 – 25 (Chapter 5)

**Goals:**

In this lesson, we will take a look at waypoint networks, which bring together both navigation and decision making. We will look at how to create waypoint networks and how to load and store them in our engine. We will also talk about the different kinds of data that can be stored in waypoints and how we can use that information to help the AI make decisions. Finally we will look at how to traverse the waypoint network using some of our earlier pathfinding methods for the purposes of getting around in the environment.

**Key Topics:**

Waypoint Networks

- Loading and Storing Waypoints
- Edges
  - Unidirectional vs. Bidirectional
  - Cost Modifiers
- Waypoints and Decision Making
  - State Machine Updates
  - Waypoint Orientation
- Waypoints and Navigation
  - A* Traversals

**Projects:**

*Lab Project 8.1: Waypoint Networks*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 8 - 10

## Lesson 9: Putting it all Together

**Textbook:** NONE

**Goals:**

Before wrapping up the course, we will pull together all of the various topics we have covered into a final application that makes use of everything. We will apply state machines to make decisions, and pathfinding routines for traversing waypoint networks in the world. We will also implement some simple squad level behavior and have a squad leader AI direct the entities under its command.

**Key Topics:**

Final Demo Implementation

- Pathfinding Algorithms
- State Machines
- Scripting
- Flocking
- Waypoint Networks
- Squad Behavior Implementation

**Projects:**

*Lab Project 9.1: Final Demo*

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 8 - 10

## Lesson 10: Exam Preparation and Course Review

**Textbook:** NONE

**Goals:**

In this final lesson we will leave the student free to prepare for and take their final examination. Multiple office hours will be held for student questions and answers.

**Key Topics:** NONE

**Projects:** NONE

**Exams/Quizzes:** NONE

**Recommended Study Time (hours):** 15 – 20

## Final Examination

The final examination in this course will consist of 50 multiple-choice and true/false questions pulled from all of the textbook chapters. Students are encouraged to use the lecture presentation slides as a means for reviewing the key material prior to the examination. The exam should take no more than two hours to complete. The final exam will be worth 100% of student final grade.

## Course Lab Projects

### Chapter 1 – 2 Projects

Name: *Pathfinding Demo*

This application allows the user to make height fields using a grid interface, place costs on each grid node, and visualize the path in both a 3D view as well as on the grid. The user is allowed to choose the method of the pathfinding.

Application Type: Win32 Application (MFC)
Required Software: Visual Studio C++ Compiler (.net 2003), DirectX 9.0+ SDK
Required Hardware: x586 / 64MB RAM / 100MB Disk

### Chapter 3 Projects

Name: *Flocking Demo*

This application allows the user to visualize fish swimming in schools, and dynamically adjust the parameters of their Flocking algorithm.

Application Type: Win32 Application (MFC)
Required Software: Visual Studio C++ Compiler (.net 2003), DirectX 9.0+ SDK
Required Hardware: x586 / 64MB RAM / 100MB Disk

### Chapter 4 Projects

Name: *State Machine Demo*

This application allows the user to design and visualize state machines and their execution. It also provides for the user being able to dynamically create new scripted transition and action states using Python.

Application Type: Win32 Application (MFC)
Required Software: Visual Studio C++ Compiler (.net 2003), ActiveState Python 2.3
Required Hardware: x586 / 64MB RAM / 100MB Disk

**Chapter 5 Projects**

Name: *Waypoint Networks Demo*

This application demonstrates the interaction between state machines, pathfinding, and waypoint networks.

Application Type: Win32 Application (MFC)
Required Software: Visual Studio C++ Compiler (.net 2003), DirectX 9.0+ SDK
Required Hardware: x586 / 64MB RAM / 100MB Disk

**Final Project**

Name: *Final Demo*

This application has a squad leader entity directing other entities to go to various positions on the map. The units traverse the map using a waypoint network.

Application Type: Win32 Application (MFC)
Required Software: Visual Studio C++ Compiler (.net 2003), DirectX 9.0+ SDK
Required Hardware: x586 / 64MB RAM / 100MB Disk