Syllabus for Artificial Intelligence

Instructor: Dr. Randy L. Ribler
Office: Room 315
email: ribler@lynchburg.edu

Text: *Artificial Intelligence Illuminated, Ben Coppin*
Readings from a number of other sources will be assigned.

Purpose of the Course
This course is designed for software developers who will be utilizing artificial intelligence in their research and/or applications. It focuses on understanding artificial intelligence from theoretical, practical, and philosophical points of view.

Topics Covered – A selection of topics from this list will be covered

Definitions of Artificial Intelligence
The Turing Test
State Space Search
  Minimax
  Alpha-Beta
  A*
Expert Systems
Fuzzy Sets
Uncertainty Management
Knowledge Representation
Scripts
Machine Learning
  Genetic Algorithms and Evolutionary Computation
  Neural Networks
  Rule Induction
Natural Language Understanding
Knowledge Engineering and Data Mining
Artificial Life
Prolog and Logic Programming

Course objectives/Learning Outcomes

Students will understand what constitutes the field of Artificial Intelligence and will be able to discuss its practical application, promise as a research field, and social implications.

Students will be able to critique the Turing Test and explain its significance.
Students will be able to explain and implement search algorithms including minimax, alpha-beta, and A*.

Students will be able to implement an expert system using one or more expert system development tools.

Students will be able to apply fuzzy set theory to the implementation of expert systems.

Students will be able to apply Bayesian statistics to provide measures of uncertainty.

Students will be able to distinguish between problems that are well suited to the application of AI techniques and those that are not.

Students will be able to implement a simple genetic algorithm and use it to solve an appropriate problem.

Student will be able to train neural networks to solve an appropriate problem.

Student will understand rule-induction and will be able to construct or apply programs that generate rules from sample data.

Students will understand the principal obstacles to natural language understanding, some basic parsing algorithms, and the role that knowledge representation has in the process.

Students will be able to write simple Prolog programs and will understand unification and the motivation behind logic programming.
Grading

Tests and Quizzes (45%):
  Test #1 (10%)
  Quizzes and in-class exercises (25%) (lowest grade is dropped)
  Final (10%)*

Programs/Exercises (45%):
  3d Tic-Tac-Toe (alpha-beta search) (15%)*
  5-7 Programs and homework exercises (30%)

Class Participation (10%)
  Attendance
  Punctuality
  Participation

*A tournament will be held during the last week of classes to determine the best 3d Tic-Tac-Toe program. The author of the winning program will receive a 10 on both the final examination and the Tic-Tac-Toe program. Runners up will also be eligible for additional points on the final, depending on the quality of the implementations.

Academic Integrity

Internet program source and other published program source may be utilized only with the instructor’s approval. Any pre-existing code, including code written by the student, must be identified prior to the initiation of the project. Algorithms from any source may be used, but references must be acknowledged in the body of the program.