Proceedings of the Seventh Symposium on Educational Advances in Artificial Intelligence (EAAI-17)

Model AI Assignments 2017

Todd W. Neller
Gettysburg College
tneller@gettysburg.edu

Joshua Eckroth
Stetson University
jeckroth@stetson.edu

Sravana Reddy
Wellesley College
sravana.reddy@wellesley.edu

Joshua Ziegler, Jason Bindewald, Gilbert Peterson
Air Force Institute of Technology
{joshua.ziegler, jason.bindewald, gilbert.peterson}@afit.edu

Thomas Way, Paula Matuszek, Lillian Cassel, Mary-Angela Papalaskari, Carol Weiss
Villanova University
{thomas.way, paula.matuszek, lillian.cassel, map, cweiss}@villanova.edu

Ariel Anders and **Sertac Karaman**
Massachusetts Institute of Technology
aanders@csail.mit.edu, sertac@mit.edu

Abstract

The Model AI Assignments session seeks to gather and disseminate the best assignment designs of the Artificial Intelligence (AI) Education community. Recognizing that assignments form the core of student learning experience, we here present abstracts of six AI assignments from the 2017 session that are easily adoptable, playfully engaging, and flexible for a variety of instructor needs. Assignment specifications and supporting resources may be found at http://modelai.gettysburg.edu.

An Introduction to Monte Carlo Techniques in AI - Part II

Todd W. Neller

Monte Carlo (MC) techniques have become important and pervasive in the work of AI practitioners. In general, they provide a relatively easy means of providing deep understanding of complex systems as long as important events are not infrequent.

Consider this non-exhaustive list of areas of work with relevant MC techniques:

General: Monte Carlo simulation for probabilistic estimation

Machine Learning: Monte Carlo reinforcement learning

Uncertain Reasoning: Bayesian network reasoning with Gibbs Sampling, a Markov Chain Monte Carlo method

Robotics: Monte Carlo localization

Search: Monte Carlo tree search

Game Theory: MC regret-based techniques

In these exercises, we will recommend readings and provide novel exercises to support the experiential learning of the third of these uses of MC techniques: Bayesian network reasoning with Gibbs Sampling, a Markov Chain Monte Carlo method. Through these exercises, we hope that the student gains a deeper understanding of the strengths and weaknesses of these techniques, is able to add such techniques to their problem-solving repertoire, and can discern appropriate applications.

Git Planner

Joshua Eckroth

This assignment is a novel application of classical planning to the widely-used Git distributed version control system. Students are asked to author a planning "domain" file in PDDL format (Planning Domain Definition Language) to represent a variety of common tasks supported by Git. The end result is a tool that is capable of producing a series of Git commands that transform a given Git repository into a specified goal state. For example, a repository may contain a file that was accidentally deleted, and the planner would be capable of producing a series of commands that recover the file. A list of required "predicates" and "actions" are provided; students are asked to implement the actions by defining each action's parameters, preconditions, and effects. To do so, students must learn how various the Git commands make use of the workspace, repository, and staging index. They must also learn how to represent this knowledge in a declarative form to match the semantics of PDDL.

Implementing a Hidden Markov Model Toolkit

Sravana Reddy

Hidden Markov Models (HMMs) are a backbone of speech and natural language processing (NLP) and computational biology. They are a great way to teach general concepts such as dynamic programming, data-driven learning and inference, and expectation maximization for unsupervised learning.

We have found that asking students in undergraduate NLP or machine learning courses to implement an HMM program is helpful in solidifying these ideas. Students also derive a great deal of satisfaction from completing a sizable project.

However, the algorithms can seem abstract, and the scope of the project may be intimidating to some students. Our assignment breaks down the HMM into modular chunks, and motivates HMMs through applications drawn from NLP research rather than toy problems. It includes:

- Starter code in an object-oriented Python framework, and programming guidelines.
- Data files that draw from part-of-speech tagging, unsupervised discovery of vowels and consonants, and decipherment of substitution ciphers.
- Lecture slides on HMMs, including step-through visualizations of the inference algorithms that translate easily into code.

The assignment can be simplified if the instructor provides some of the methods, or made more challenging for advanced classes by generalizing it to continuous observations, modifying the learning procedures, and other variations.

An Introduction to Behavior-Based Robotics

Joshua Ziegler, Jason Bindewald, and Gilbert Peterson[1]

In the spirit of Brooks's seminal work Intelligence without Representation (Brooks 1991), students implement a behavior-based simulated tank agent in the AutoTank environment built using the Unity 3D framework with C# and JSON scripting. Each student's goal is to create a tank agent that will win a tournament pitting all students' tank agents against each other in a series of games in the AutoTank environment. The tank agent's goal in each game is to either eliminate all other opponent tanks or have the most remaining health at the end of a preset time limit. In order to do this, students create tank agents that utilize a reactive behavior-based architecture built using the Unified Behavior Framework (UBF) (Woolley and Peterson 2009), which has been implemented in AutoTank. A comprehensive tutorial, AutoTank environment documentation files, and sample agents for student testing are included. Students should come away with a better understanding of how behavior-based systems in robotics are structured and the strengths and limitations of different designs.

Machine Learning for Everyone: Introduction to Classification and Clustering

Thomas Way, Paula Matuszek, Lillian Cassel, Mary-Angela Papalaskari, and Carol Weiss

This Novel AI assignment module introduces two important machine learning approaches: Classification and Clustering. Each approach provides a way to group things together, the key difference being whether or not the groupings to be made are decided ahead of time. While these grouping techniques are a type of Artificial Intelligence designed to be performed by a computer, they can be used on any sort of data from any discipline. The module provides background for an instructor in any discipline to incorporate the material into a lesson for students in any discipline where machine learning may be applicable. It is aimed at a beginner, non-technical student audience, though it is also appropriate as an introduction for students with some computing experience. Materials for hands-on exercises and pre- and post-test quizzes are included.

Organic Pathfinding

Joshua Eckroth

This assignment challenges students to develop a novel "human-like" pathfinding technique by specializing a generic search algorithm with custom action cost and heuristic cost functions. Students are also asked to reproduce common search techniques like A*, best-first search, beam search, and breadth-first search with appropriate action and heuristic cost functions. The specifications of the "human" search technique are left open-ended, but students are asked to recreate "organic" paths (also known as "desire paths") as found in natural and urban environments. A typical organic path is one that cuts across a grass field even though a sidewalk or road is nearby but less direct. Sometimes, organic paths are less efficient, due to humans' tendency to avoid walking close to obstacles such as buildings and trees, so they are clearly distinct from A*. Organic paths are also visually distinct from breadth-first and best-first search. Students are encouraged to examine example organic paths and reflect on their own behavior to develop the "human" search technique. A Python script is provided that simulates a student's search techniques and produces images of the resulting paths.

Visual Servoing

Ariel Anders and Sertac Karaman

In this assignment students use Image Based Visual Servoing to program a mobile robot to park in front of solid color objects, like orange cones. The control of the robot is completely based on information extracted from a standard monocular camera. The assignment is structured in two parts: blob detection, and closed loop proportional control. Students leverage open source software tools like OpenCV and ROS to implement this reactive planner.

This assignment has been tailored from advanced high school students to junior and senior level undergraduate students in Engineering and Computer Science. The difficulty of the assignment is highly scalable based on the environment the robot encounters. For example, the complexity of the blob detection for orange cones in plain environments is much easier to implement than harder detections like a picture of a cat in low lighting environments.

[1]The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense, or the United States Government.

This assignment is useful towards the beginning of a robotics course. It's a fun and intuitive combination of sensing and control. It's versatile since there are many direction students can take with their object detection and control algorithms.

References

Brooks, R. 1991. Intelligence without representation. *Artificial Intelligence* 47(1):139–159.

Woolley, B. G., and Peterson, G. L. 2009. Unified behavior framework for reactive robot control. *Journal of Intelligent and Robotic Systems* 55(2):155–176.