Laboratory Manual
for
FRA 311

Project: Path Planning, Basic Navigation,
Image Processing, and Machine Learning

BY
Bawornsak Sakulkueakulsuk

Introduction

Meeting Time

Section A
Lecture & Lab: Tuesday 9.30 - 12.20 am

Section B
Lecture & Lab: Monday 9.30 - 12.20 am

Lab Structure

Each lab will help you build your tools for your projects. There are 2 projects. Each project will take 2-3 weeks to complete.

Project 1: Path Planning

Lab 1

Lab 1 will introduce you to design structure of path planning program. You will implement uninformed search algorithms.

Lab 2

Lab 2 will introduce you to more advanced path planning algorithm. You will implement heuristic search algorithm.

Lab 3

Lab 3 will be a free Lab. You can work on the project, or plan for FRA 362 Robotic Studio.

Report

You will need to turn in one path planning labs report per group.

Project 2: Image Classification

Lab 4

Lab 4 will introduce you to WEKA. You will experiment on the process of Machine Learning.

Lab 5

Lab 5 will introduce you to classification algorithms such as regression model. We will also discuss about Robotic Studio project.

Lab 6

Lab 6 will introduce you to classification algorithm such as decision tree and k-nearest neighbor.

Report

You will need to turn in the Machine Learning part of your proposal for Robotic Studio project. You also need to demo Image Feature Extraction algorithm you will use for your Robotic Studio Project.

Lab 1: Intro to Path Planning

Objective

In this lab, you will

- Explore the path planning program structure.
- Implement uninformed search algorithms.
- Build maps.
- Test your algorithms.

Lab Instruction

Part 1: Exploring the program

In this part, you will explore and understand the role of each class.

1. Go to https://github.com/BawornsakS/FRA311
2. Access a folder called uninformed_search

In this folder, there are 8 files: Graph.py, Node.py, Problem.py, Search.py, Search_test.py, __init__.py, grid.py, utils.py. You need to understand the relationship of all the files in order to use them.

- **Search.py**
  This file contains algorithms for various searching methods. You will implement the searching algorithm in this file.
- **Search_test.py**
  This file contains test cases for searching. You will need to run this test file to test your algorithm.
- Graph.py
  This file is for building a Graph object. A graph connects nodes by edges.

- Node.py
  This file represents a Node and its operations. A node is a basic structure that contains some values, a link to previous node, and a link to the next node.
- SquareGrid.py
  This file contains a representation of a grid to be used in some searching algorithm.
- __init__.py
  This is an empty file for configuration. You don't have to pay attention at this file.
- implementation.py
  This file provides tools and libraries for other files.

Your job is to understand the relationship of each file. You do not have to understand everything at this point. Pay attention to Graph.py, and Search_test.py. These two classes allow you to set up a problem definition for path planning problems. If you understand the problem structure, you can utilize them when you implement search algorithms.

Part 2: Implementing your first search

If you run Search_test.py, it will output a path found by using breadth-first search algorithm. Your job is to implement other searching algorithms in Search.py.

1. Breadth-first Search
   The algorithm is already implemented. Your job is to understand the algorithm. You can use this as a guideline for implementing future algorithms.
2. Depth-first Search
   Now it is your chance to start coding. Your job is to implement depth-first search algorithm.

Part 3: Breaking your search

If you run depth-first search using the default map, you probably get the result (if you implement the search correctly, of course). You can use this map as a basic testing. However, this map does not provide much information about features of each search. Your job is to create new maps to test your algorithm. For example, create a map that put your depth-first search in an infinite loop.

Part 4: Upgrading the search

Your job is to implement two more algorithms that should be able to solve the problem.

1. Depth-limited Search
   Your job is to implement depth-limited search. You can use depth-first search as a base, and modify your algorithm for depth-limit search.

2. Iterative Deepening Depth-limited Search
   Your job is to implement iterative deepening depth-limited search. You can use depth-limited search as a base, and modify your algorithm for IDDLS.

Part 5: Testing

Test your algorithms and make sure everything is working. You can try creating any map you like to test this. You will need to put your created maps and your analysis in the report after next week. I encourage you to look at the Deliverables section in Lab 2 for more information.

Lab 2: Heuristic Search

Objective

In this lab, you will

- Learn about a heuristic function.
- Implement A* search algorithm.
- Test your algorithm.

Lab Instruction

Part 1: Understanding the algorithm

In this part, you will explore and understand the pseudo algorithm of A* Search. You may refer to the lecture, or other online sources (such as wikipedia).

Part 2: Understanding the map

In this lab, we will use a different map: a square grid. This map is built by putting a square grid together to form a bigger rectangle.

There is a function for you to build a wall, and put different weight in the map. Imagine this problem as a robot navigation through different terrain. For example, navigating through sands will cost more time and energy than navigating through a road.

There is also a function to let you put up a wall. The robot cannot travel through a wall (unless it has a drill, which it doesn't).

Play around the class and building couple of maps. You will use it soon. :)

Part 3: Implementing A*

Now you have a map. You need to implement the A* algorithm, and test it using your map.

Deliverables

You will submit one report for both Lab 1 and Lab 2. The report should contain your code for each algorithm, your explanation of the code, what problems (maps) you run your code on, and analysis of the result. Compare and Contrast each algorithm in the analysis. The report should not exceed 4 pages.

Lab 3: Free Lab

Objective

In this lab, you can

- Work on the previous lab.
- Ask questions if you have any.
- Work on your Robotics Studio Project.
- Ask about Robotics Studio project.

You have Preliminary Design Review coming up for Robotics Studio. Use the time wisely. :D

For the next lab

- Install WEKA on your system. You can download it from http://www.cs.waikato.ac.nz/ml/weka/downloading.html
- You should have working image processing module finished by the beginning of next lab. Starting next week, you will work with your Robotics Studio group to implement the machine learning part of the project. Please bring your image processing code with you, and also a set of images of your ingredients.

Lab 4: Preprocessing

Objective

In this lab, you will

- Explore machine learning cycle and techniques.
- Get familiar with WEKA.
- Learn basic understanding for Robotics Studio Project.

Lab Instruction

Part 1: Understanding the GUI

Open WEKA and play around with it. I will explain the software in class.

Part 2: The Dataset

In this section, you will modify your image processing program so that it can output the CSV file for WEKA system.
CSV stands for Comma-Separated Value. It is the file format we use in programs such as Microsoft Excel or programs that put data in a table. As its name suggests, a row of data in the file consists of values separated by a comma (,). A new line character separate each item in the dataset.
The first step of Machine Learning is to gather the dataset. I asked you in the previous lab to prepare the images of your ingredients. Your job is to

- Modify your program so it can take in several image files, if you haven't done so.
- For each image, process it and write it to the file in CSV format.
- Once done, check your file if it is in CSV format.

Part 3: Data Preprocessing

As we talked in class, we need to clean up our dataset. You can play around with techniques such as

- Discretization. You can find it in filters →unsupervised →attribute →Discretize
- Missing Values. You can find it in filters →unsupervised →attribute →ReplaceMissingValues
- Feature Selection. You can find it in filters →supervised →attribute →AttributeSelection.

Lab 5: Basic Machine Learning

Objective

In this lab, you will

- Explore machine learning cycle and techniques.
- Learn about basic classification techniques.
- Apply machine learning techniques for Robotics Studio Project.

Part 1: Linear Regression

You will explore linear regression in WEKA.

1. Open a file in your installed WEKA folder →WEKA-3-8 →data →airline.arff
2. Go to Classify tab
3. Choose Classifier in functions →LinearRegression
4. Click Start
5. Observe and analyze result.
6. Change parameters and run again
7. Now, try using your ingredient dataset. Does it work?

Part 2: Logistic Regression

You will explore logistic regression in WEKA.

1. Open a file in your installed WEKA folder →WEKA-3-8 →data →breast-cancer.arff

2. Go to Classify tab
3. Choose Classifier in functions →Logistic
4. Click Start
5. Observe and analyze result.
6. Change parameters and run again
7. Now, try using your ingredient dataset. Does it work?

Lab 6: Classification

In this lab, you will

- Explore machine learning cycle and techniques.
- Learn about classification techniques.
- Apply machine learning techniques for Robotics Studio Project.

Part 1: Decision Tree

You will explore Decision Tree in WEKA.

1. Open a file in your installed WEKA folder →WEKA-3-8 →data →iris.arff
2. Go to Classify tab
3. Choose Classifier in trees →J48
4. Click Start
5. Observe and analyze result.
6. Change parameters and run again
7. Now, try using your ingredient dataset. Does it work?

Part 2: K-Nearest Neighbor

You will explore logistic regression in WEKA.

1. Open a file in your installed WEKA folder →WEKA-3-8 →data →iris.arff
2. Go to Classify tab
3. Choose Classifier in lazy →IBk
4. Click Start
5. Observe and analyze result.

6. Change parameters and run again

7. Now, try using your ingredient dataset. Does it work?

Part 3: Deliverable

You have your image processing program from Preliminary Design Review. Your job is to demo your image classification program. I am looking for a program that allows the user to

1. train a classification model from database.

2. when supplied a test set, the program can accurately classify the items in the dataset.

l Here are minimum requirements for what you should do:

1. Show the raw dataset. What items are in your dataset? How many instances?

2. Show the modified dataset. How does your dataset look like after processed the images using image processing? How many attributes?

3. Show the clean dataset. What do you do for the preprocessing? Do you eliminate some features? Why?

4. Show the model.

5. Show the test set, and the result after classifying the test set using the model you build.

You are encouraged to add more details. There will be no class on 2/20 and 2/21. However, you need to schedule the time with me to demo your project. Noted that the critical design review for robotics studio is on 2/24. If you present your project earlier, I can give you feed back earlier, so you have more time to fix your project, before presenting in robotics studio.