Computational Intelligence: TI2735-A

A Robot To Do Your Groceries: Grand Challenge Preparation

October 23, 2014

5 Grand Challenge Preparation

The *Grand Challenge* is the challenge among challenges! Your robots will be fighting in a fierce siege in which many heroes will fall. It will be a fight among the best, a fight for loyality, a fight to save Peach and, for many, their first and last fight ever.

5.1 Goal

The goal was to use Computational Intelligence to teach your robot to do the groceries. In the Grand Challenge, your robot will categorize products, profile its owner, determine a shopping list, learn to navigate and figure out the fastest way through the supermarket.

May the best brain win!

5.2 Relation with Previous Assignments

You will incorporate all the previous assignments to come up with your own shopping list and route.

Firstly you will receive a list of features for 35 products. Your neural network will have to classify these into the 7 classes.

However, not all the classes should be bought by the customer. This is where the Bayesian network comes in. Based on the purchasing history of a customer your Bayesian network will profile its owner (back-propagating the evidence). Based on these probabilities you will make a deterministic profile of the customer. This in its turn will reveal 3 classes in your Bayesian network that are most wanted by your customer. Since your Neural Network has already categorized all products into its respective class, you now know which product numbers to buy and not to buy!

Then you will use your ants to determine the fastest route between all the products to be picked up.

Finally your Genetic Algorithm will determine the fastest way to visit all these products and get to the exit!

Part 1: Neural Network

The input to your neural network is a feature list. Ten features for each of the 35 products.
This file is called *GC_FeatureData.txt* and has the same syntax as in assignment 1. Your ANN will output the classes of these 35 products.
The first line of features corresponds to product number #1, and the last line to product #35.
The data is from the same feature set as before so there's no need to train your network on this new data.

Part 2: Bayesian Network

Your Bayesian network operates independently from the rest within GeNIe. You'll be provided with the deterministic (boolean) shopping history for the customer during hist last 5 trips to the supermarket. This file is called *GC_PurchasingHistory.txt*
It works as follows: You set the evidence in each trip and for each class to either 1 or 0 based on the shopping history provided. You then update your network and check the probabilities for the customer's characteristics. If a characteristic has a probability higher than 80% you will (in the end) assume that he/she has this trait and set its value to 1. Any trait that was never higher than 80% should be set to 0. *Don't do this just yet or it will affect the network in analyzing the remaining 4 trips.*

After you've done this 5 times and remembered all the traits that were higher than 80% you can deterministically assign the customer's profile. This will then sort the classes based on the probability per class. Now choose the 3 classes with the highest probability, These are the products your robot will buy!

Part 3: ACO Path Finding

As you know, the input for the ants is a list of locations which it should determine the relative distances between. Now that you know which classes the products belong to, and you know which of those classes should be picked up. You then know which product numbers to buy and which to never visit.

You'll also be provided with where all products are located, i.e. 35 x,y-coordinates.
This file is called *GC_ProductCoordinates.txt*.
Hence you filter out the products which correspond to the classes suggested by your Bayesian network and you now have an input to your ants for 'cities' to find the distance between. (It will be about 10-15 places).

In other words, you'll have to write a function which selects just those products from *GC_FeatureData.txt* that belong to the 3 classes determined by your Bayesian network. Now that you know which product numbers to buy you can select the correct coordinates from *GC_ProductCoordinates.txt*.

Part 4: Evolutionary Computed TSP

The last thing to do is to do determine the quickest route between the products you've decided to buy. This is where your Genetic Algorithm comes in. Based on the list of products and their relative distances, your GA will determine a route.

The input to this part of your software is thus a symmetric matrix with the relative distance between 10 to 15 products, as generated by your ants earlier.

The result from this algorithm will be the final *Action.txt* file you will hand in at the day of the challenge (See Sec. 5.6).

5.3 What you Need to Hand-in

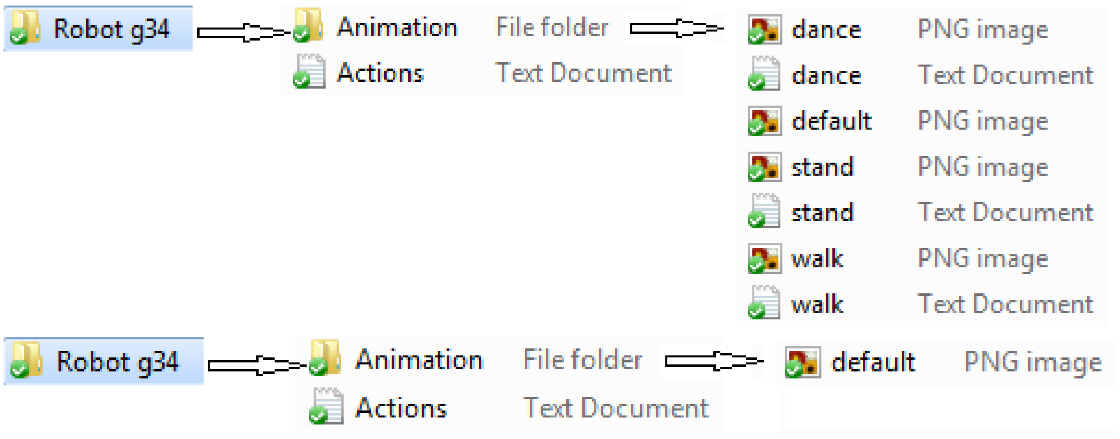
In order to have a smooth workflow the day of the Grand Challenge, you will need to do some preparations. Each group is required to prepare a file system, as described in Sec. 5.4. Within that file system, some sprites must be contained (Sec. 5.5). Also, an 'Actions.txt' file must be included, which is detailed in (Sec. 5.6).

Please note that **the maze & product list for the Grand Challenge is different than in the previous assignments**. In other words you cannot make your final action-file just yet. But still, you should prepare the file system (with an empty action-file).

5.4 File System 'Syntax'

Fig. 5.1 shows the file system you must hand-in. The main directory should be named "Robot g«groupnumber»". In the main directory, an "Animation" directory and an "Actions.txt" file must be included. The Animation directory is described in Sec. 5.5. The Actions.txt file is described in Sec. 5.6.

A template for this file system may be found on blackboard, this saves you some time.



Robot g34

Animation	File folder
Actions	Text Document

dance	PNG image
dance	Text Document
default	PNG image
stand	PNG image
stand	Text Document
walk	PNG image
walk	Text Document

Robot g34

Animation	File folder
Actions	Text Document

default	PNG image

Figure 5.1: *You are required to submit the file system as shown in the figure. Note that the syntax is **exact**! You have one choice though: making an animation for your robot is optional (see Sec. 5.5).*

5.5 Graphics

Within the 'Animation' directory, you **must** include a 'default.png' file (i.e., a 'sprite'). If you do not have extremely much time, keep it simple. The sprite will be used to visualise your robot in the maze. Additionally, you may specify some animation sprites: 'dance.png', 'walk.png' and 'stand.png' together with timing files 'dance.txt', 'walk.txt' and 'stand.txt'. The animation sprites are optional (see Sec. 5.5).

Remember it is a robot doing the groceries. So your sprite must be a robot of some sort. NO disturbing or inappropriate graphics!

If you download the file system-template from Blackboard, a standard sprite and animation are included.

Default Sprite

The default sprite is a static image, used to visualise your robot in the maze and menus. This little fella will represent your group during the Grand Challenge. The image should be a **128x128** image, preferably .png with transparancy/alpha/selection (see Fig. 5.2).

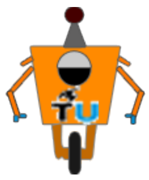
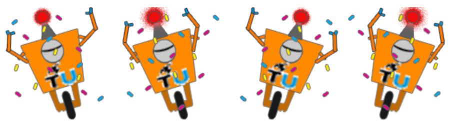


Figure 5.2: *Example of 'default.png'*

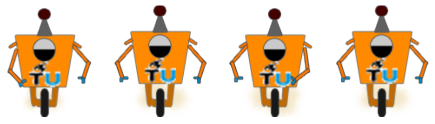
Animation Sprites (optional)

You may provide animation sprites for your robots for any of the animations: stand, walk & dance. **It is as well possible to provide e.g. only the stand animation** (your robot will play its stand animation when the required animation is non-existent).

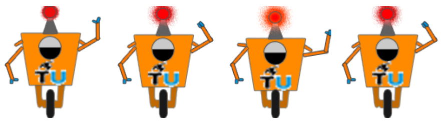
An animation image is a **128Nx128** image, with *N* the number of animation frames. Examples are shown in Fig. 5.3. Aiming at 4 animation frames is a good idea to do. **Do not use more than 6 frames**. If you do not have time to make 4 frames, providing as few as one frame does as well make your robot a lot more entertaining to watch at (since your robot will look different during e.g. walk & stand time)!



(a) *Example of 'dance.png'*



(b) *Example of 'walk.png'*



(c) *Example of 'stand.png'*

Figure 5.3: *Animation sprites*

Also, with each animation sprite, you must provide a timing-file: e.g. 'walk.txt', as shown below. This one-line file starts with the number of frames, followed by a ';'. Then, the float ending time of each frame is listed, followed by a ';' for each float. The example below has 4 frames, with a total animation time of 1.3 seconds. The first three frames last 0.3 seconds in total. The fourth or last frame lasts 0.4 seconds.

```
4; 0.1; 0.2; 0.3; 0.4;
```

5.6 ‘ACTIONS.TXT’ SYNTAX

A small example of a correct action file is shown below.

```
73;
3;
0;
take product #24;
3;
3;
take product #3;
0;
1;
1;
2;
3;
(...)
```

The syntax is the same as in part 2 of assignment 3 with one important difference:
Since each of you will be running around in the same maze, there is no reason to specify a starting location. **So comment out the line(s) of code which print the starting position to your action file**. (Or remove it manually. Just make sure the line disappears from your action file for the Grand Challenge.)

Also - obviously, your route should be such that your robot's end position is the exit of the maze. If your robot does not reach the end, or attempts to walk through the wall, he will automatically be disqualified.