Reinforcement Learning: Coursework Assignment 2 **(*Semester 2, 2015/2016*)**

Instructions

- This homework assignment is to be done *individually*, without help from your classmates or others (except the RL tutors). Plagiarism will be dealt with strictly as per University policy.
- Solve all problems and provide your complete solutions (with adequate reasoning behind each step) in a report in computer-printed or *legibly* handwritten form.
- Before you start to write a program, read all questions below carefully.
- Only the report will be marked. The code itself will not be marked, but will used to clarify any questions arising from the report. If you are using code that you have not written yourself, then acknowledge this appropriately (you do not have to mention the code provided for the tutorials). Include references to books and papers that you have been using (you do not have to cite the RL lecture slides).
- Use graphical representations wherever suitable. If you use numerical output for demonstrating your results, make sure that the numbers are appropriately rounded and presented in an accessible way. If your problem involves randomness, explain why your result is representative for most of the possible realisations of the underlying random effects.
- Include your code in the submission as a separate file. If you are presenting numerical results in your report, specify all major numerical parameters that were involved in the generation of the data shown.
- This assignment will count for 10% of your final course mark.
- Please submit your assignment by 4 pm on 17th March as a paper copy to ITO as well as an electronic version (including your code) via the submit system (directory "rl").

Questions

The task to be studied here is similar to first assignment, but we will assume a simplified setting. We consider an agent moving in an (S_x-by-S_y) grid world. The agent can move towards any of the four neighbouring fields or choose to stay in the current field. Movements towards the boundary of the arena are possible but do not affect the position of the agent. The goal of the agent is reaching a goal location in one of the corners of the grid on the shortest path after starting from any random location in the grid. The agent receives a reward $r=1$ if it reaches the goal location. Otherwise the reward is $r=0$.
The purpose of the assignment is to study effects of scale when the value function is approximated by a set of basis functions.

1. Before you start programming, consider what domain knowledge and what properties of the algorithm may be useful for finding the solution (no marks here this time).
2. Solve the problem using Q-learning[1] for a (5-by-5) grid. Use indicator functions in the representation of the Q-function, i.e. $\varphi_k(x)=1$ if the agent is in square $k=x$ and $\varphi_k(x)=0$ otherwise (obviously $k=x$ is chosen for simplicity, all we need here is that for each square one of 25 indicator functions is 1 and all others are 0). It is recommended to use the same basis functions for all actions, but with action-dependent parameters.
   This procedure is meant to be an exact reproduction of the look-up table version of the algorithm. Check whether this is indeed the case and show your results. (20/100)
3. Now let the agent move on a (10-by-10) grid, still using 25 indicator functions, i.e. each of the functions covers now four grid cells. How does the value function change compared to the case of an agent in a (5-by-5) grid? How does the convergence time change? Note, that you may have to redefine the definition of the goal. (20/100)

4. Now, consider grids of various sizes (5-by-5, 10-by-10, 25-by-25) and use in each case a set of 25 two-dimensional Gaussian bell-shaped functions as basis functions for the representation of the value function. You can choose whether you use functions with regularly spaced centers or functions with randomly placed centers. Show and explain your results (25/40). Discuss the effect of the width of the basis functions (10/40) and (briefly) the effect of other parameters (5/40). (total for this question: 40/100)
5. Repeat the simulations of question 4 for a less trivial problem. You can use e.g. obstacles like in the taxi problem of assignment 1, but you may choose as well a different problem such as balancing a pole or the mountain car problem. (20/100)
6. Bonus question: Solve the problem of question 5 in continuous space using a parametrised (stochastic) policy which allows the agent to move in each time step one step in any direction (or to use an appropriate continuous action set if you did not stay with the grid world). (20 bonus points to a maximum of 100)
7. Add a paragraph of conclusions drawn from the solution of these problems, in particular reviewing the design choices you have made.

[1]You can also use another reinforcement learning algorithm if you prefer.