project 3:

behavior programming

solution(s) due:

July 18, 2016 at **12:00** via email to **bauckhag@bit.uni-bonn.de**

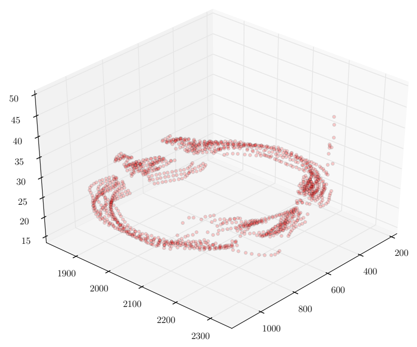
problem specification:

task 3.1: *connect four* on a large board: for the fun of it, let your *connect four* agent from task 2.3 run on a 19×19 board. Repeat your experiments (tournaments) from task 2.3 and report what you observe (run times, win/loss statistics).

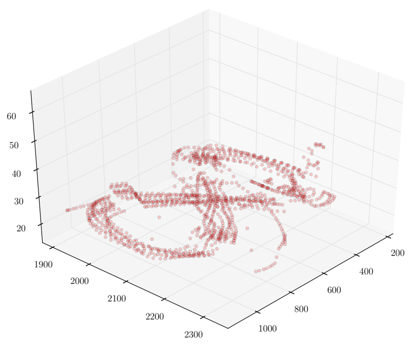
task 3.2: breakout: implement a *fuzzy controller* for the breakout game of your choice. That is, define a set of IF-THEN rules in terms of linguistic variables that describe how the paddle has to be moved in order to hit the ball. Fuzzify your rule set and create a continuous output as discussed in the lecture. Note that there are various python libraries for fuzzy logic and fuzzy control; you may use them if you wish to do so. Since this is still a rather trivial task, let's again make the overall setting more interesting: modify the code of your breakout implementation such that the speed of the ball increases over time and see how your fuzzy agent copes with this.

task 3.3: self organizing maps to represent player movements: download the files `q3dm1-path1.csv` and `q3dm1-path2.csv`. They contain trajectory data, i.e. sequences of vectors $\boldsymbol{x}_t$ which indicate a human player's 3D position at time t on the Quake III map *q3dm1*.
Fit self organizing maps (SOMs) to both data sets. Choose the SOM topology to be a *circular path graph* $v_1 \leftrightarrow v_2 \leftrightarrow \ldots \leftrightarrow v_l \leftrightarrow v_1$ of l vertices. Experiment with different choices of l and visualize your results. The following two videos provide an idea as to how a visualization may look like:

http://www.youtube.com/watch?v=ddbBIWAIzxE
http://www.youtube.com/watch?v=k7DK5fnJH94



data in `q3dm1-path1.csv`



data in `q3dm1-path2.csv`

task 3.4: Bayesian imitation learning: note that the weights $\boldsymbol{s}_i$ of the self organizing maps you fitted in the previous task can be understood as a representations of k prototypical *game states* the player assumed while recording the match. In other words, in the previous task, you clustered the location data $\boldsymbol{x}_t$ into l states $\boldsymbol{s}_i$.
Also note that, for the ongoing example, the player's activity at time t can be characterized as

$$\boldsymbol{a}_t = \boldsymbol{x}_{t+1} - \boldsymbol{x}_t.$$

Now, compute all activity vectors for data set `q3dm1-path1.csv` and cluster them into k prototypical activities $\boldsymbol{r}_j$ using the k-means algorithm.
Next, assign every location vector $\boldsymbol{x}_t$ to its closest prototypical state vector $\boldsymbol{s}_i$ and every action vector $\boldsymbol{a}_t$ to its closest prototype $\boldsymbol{r}_j$.
Now, compute the joint probabilities $p(\boldsymbol{s}_i, \boldsymbol{r}_j)$ of state $\boldsymbol{s}_i$ and action $\boldsymbol{r}_j$ occurring together. That is, if the player performed action $\boldsymbol{a}_t$ at location $\boldsymbol{x}_t$ and $\boldsymbol{x}_t$ belongs to $\boldsymbol{s}_i$ and $\boldsymbol{a}_t$ belongs to $\boldsymbol{r}_j$, increase the count of the pair $(\boldsymbol{s}_i, \boldsymbol{r}_j)$, and, once you are done counting, normalize these counts so that they become probabilities.
Finally, use the probabilities $p(\boldsymbol{s}_i, \boldsymbol{r}_j)$ to generate a trajectory. Proceed as follows: at $t = 0$, randomly select a point $\boldsymbol{x}_t$ among the given data, determine its closest state vector $\boldsymbol{s}_i$ and select an action according to

$$\boldsymbol{a}_t = \underset{\boldsymbol{r}_j}{\operatorname{argmax}}\ p(\boldsymbol{r}_j \mid \boldsymbol{s}_i) = \underset{\boldsymbol{r}_j}{\operatorname{argmax}}\ \frac{p(\boldsymbol{s}_i, \boldsymbol{r}_j)}{p(\boldsymbol{s}_i)} = \underset{\boldsymbol{r}_j}{\operatorname{argmax}}\ \frac{p(\boldsymbol{s}_i, \boldsymbol{r}_j)}{\sum_q p(\boldsymbol{s}_i, \boldsymbol{r}_q)}$$

Finally, compute

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{a}_t,$$

set $t = t + 1$, and iterate this process several (hundred) times. Plot the trajectory you obtain this way. Compare it to the trajectory recorded by the player. Are they similar? If not, how could you modify this approach?
See if this idea will work for the data in `q3dm1-path2.csv`. Most likely it will not! Why not? What could you do? Have a look at the following papers

- C. Thurau, C. Bauckhage, G. Sagerer: Synthesizing Movements for Computer Game Characters, Proc. DAGM, 2004
- C. Thurau, T. Paczian, C. Bauckhage: Is Bayesian Imitation Learning the Route to Believable Gamebots?, Int. J. of Intelligent Systems Technologies and Applications, 2(2–3), 2007

which can be found on researchgate.net.

general hints and remarks

- Send all your solutions (code, resulting images, slides) in a ZIP archive to bauckhag@bit.uni-bonn.de
- **note:** carefully debug your code! Make sure your programs do what they are supposed to do! Programming skills are essential for this course and are expected. Flawed programs, i.e. implementations of games that violate the game rules, imply failing the course.
- Remember that you have to successfully complete all three practical projects (and the tasks therein) to be eligible to the written exam at the end of the semester. Your grades (and credits) for this course will be decided based on the exam only, but –once again– you have to succeed in the projects in order to get there.
- Not handing in a solution implies failing the course.
- Your project work needs to be *satisfactory* to count as a success. Your code and results will be checked and your presentation needs to be convincing.
- If your solutions meet the above requirements and you can show that they work in practice, they are *satisfactory* solutions.
- A *good* to *very good* solution requires additional efforts especially w.r.t. to elegance and readability of your code. If your code is neither commented nor well structured, your solution is not good! A very

good solution requires additional efforts towards the quality of your project presentation in the colloquium. Your presentation should be well timed, consistent, and convincing. Striving for very good solutions should always be your goal!