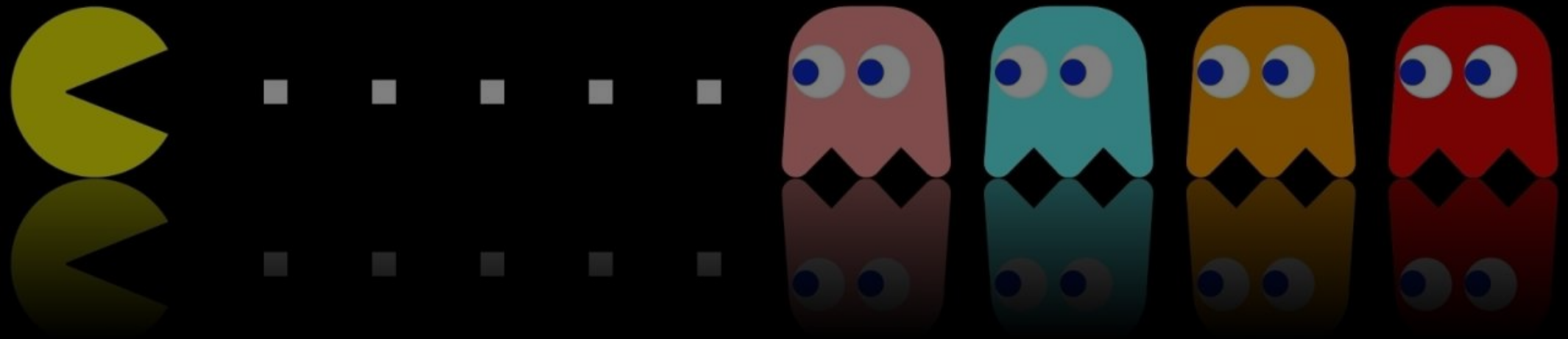
Ms. Pac-Man vs. Ghost Team

Tutorial and Competition Instructions

Alexander Dockhorn - April 2017



Organization

- The tutorials will consist of about six exercise sheets, which will be discussed in several tutorial sessions
- It is advised to study the exercise sheet beforehand
  - We ask for active participation!
  - However, no theoretic tasks need to be submitted!

- Additionally to the theoretic tutorials we will hold an internal competition for Ms. Pac-Man vs Ghost Team
- Each student will need to participate in the contest
  - Bachelor and Master students will get tasks of differing difficulty
  - Each submission needs to be shortly presented in the tutorial
  - You can work in a group of 2 members

Why do we do that?

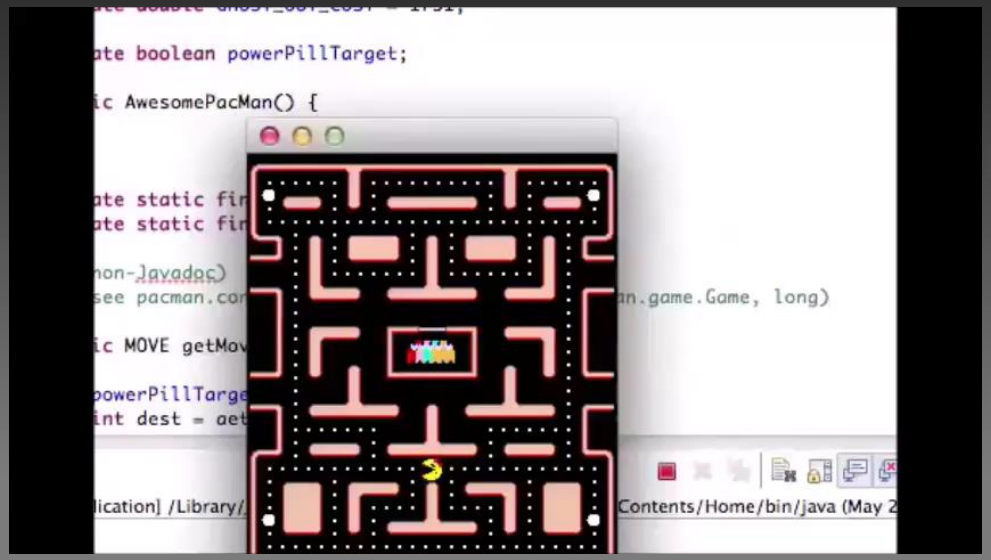
Theoretic examples to foster the learning process

Practical applications

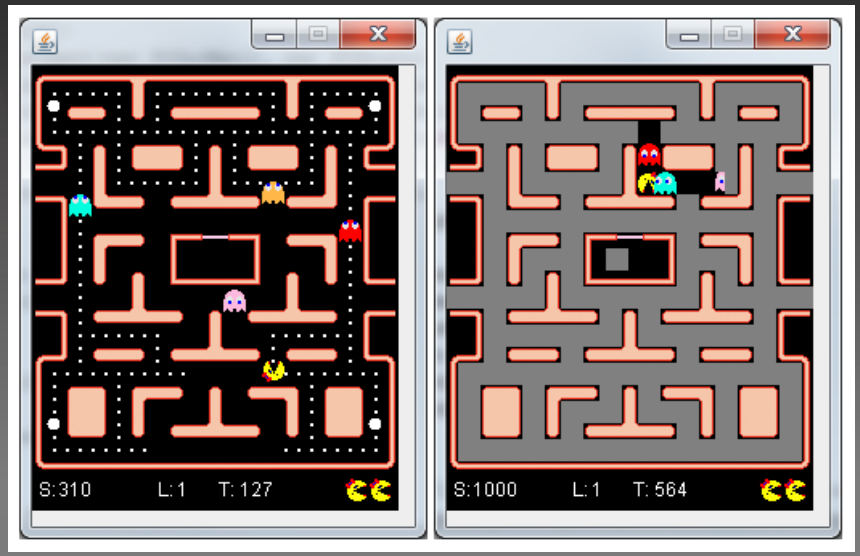
Gain instant feedback for your ideas and solutions

The Competition

https://www.youtube.com/watch?v=koX62WAV_kA

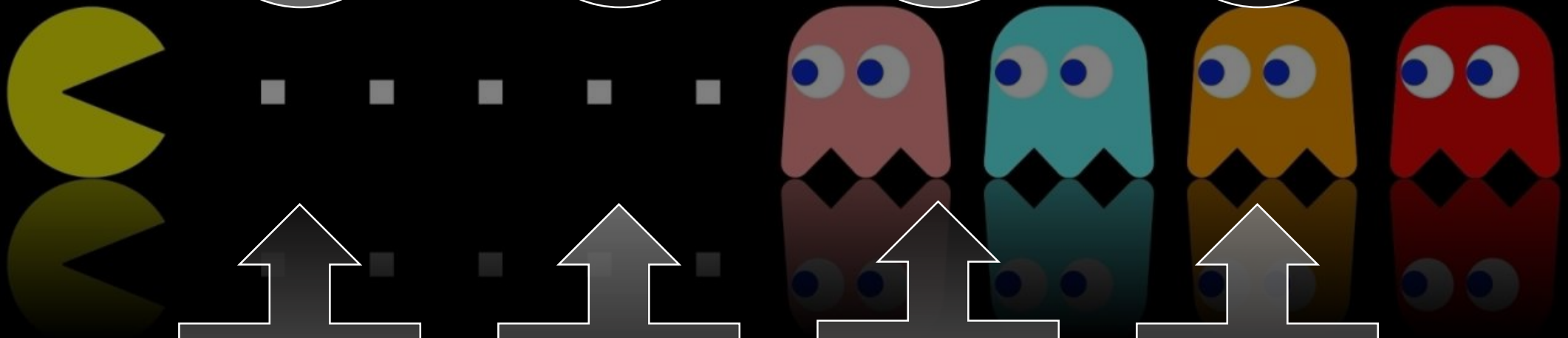


Partial Observation added in 2016



The Competition

2007 → 2011 → 2016 → 2017



Start of the original competition [1,2]

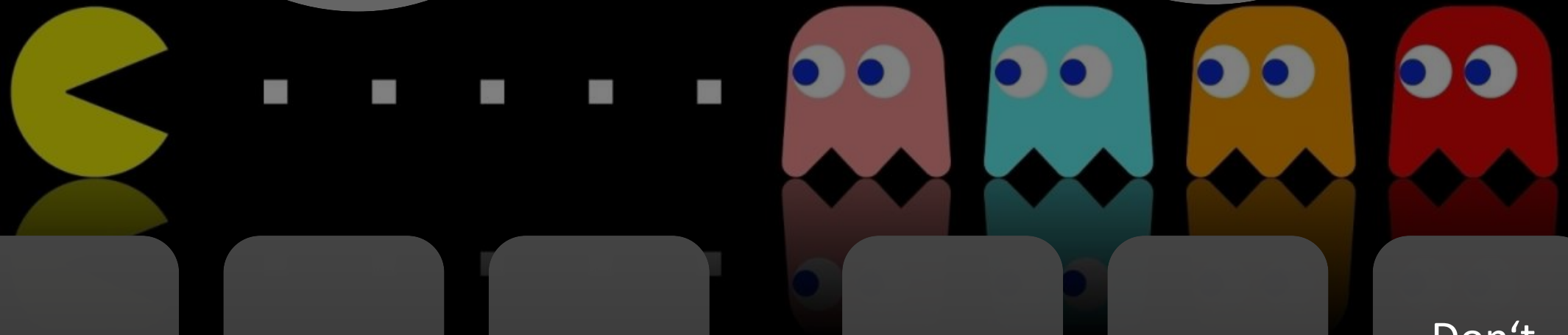
End of the first competition [3]

Partial Observation added to the competition [4]

Internal competition at the OvGU

Tasks

Ms. Pac-Man VS Ghost Team



Collect all pills	Stay alive	Eat the ghosts	Find Ms Pac-Man	Defend the pills	Don‘t get eaten

Tasks Description

B. Sc. & M.Sc

- Bachelor **and** Master students need to develop an AI for **Ms. Pac-Man**
- You have 40ms to process the current gamestate and return your move
- Score the most points
  - Eating Pills = 10 points
  - Eating Ghosts = 200, 400, 800, 1600 points per Power Pill

M. Sc.

- Master students need to develop an AI for the **ghost team**
- The agent is split into 4 instances, each representing one ghost
- Each instance has 40ms time to process the gamestate and return a move
- Limited communication is available
- Stop Ms. Pac-Man from scoring
  - Try to minimize the total amount of points scored against your ghost team by all instances of Ms. Pac-Man AIs

Competition Details

- Installation instructions can be found at:
  http://www.pacmanvghosts.co.uk/
- You do not have to register on the official webpage!
  - …but please approach me if you want to participate anyway
- The final submission needs to consist of:
  - your documented source code
  - a short explanation in written form (2-5 pages)
  - master students will need to create a short presentation on a prototype and their final solution

Installation

- Download and extract:
  http://www.pacmanvghosts.co.uk/downloads/starter0-1-7-2.zip
- Install Java:
  http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- Recommended: Install Eclipse for Java or an IDE of your Choice:
  https://www.eclipse.org/downloads/eclipse-packages/
  (I had multiple problems in using an older version of eclipse. Running Eclipse Neon 3 resolved all of them.)
- Install Maven (see next slide):
  https://maven.apache.org/download.cgi

Install Maven

- Extract “apache-maven-3.3.9-bin.zip” to "C:\Program Files (x86)”
- Add the bin folder to your environment variables
  - Rightklick the start button -> system -> advanced systemsettings –> environment variables
  - Search for the entry „Path“ (add if it does not exist)
  - Add "C:\Program Files (x86)\apache-maven-3.3.9\bin“
- Maven should now be runnable from the command line using „mvn“
  - Otherwise: restart and try again

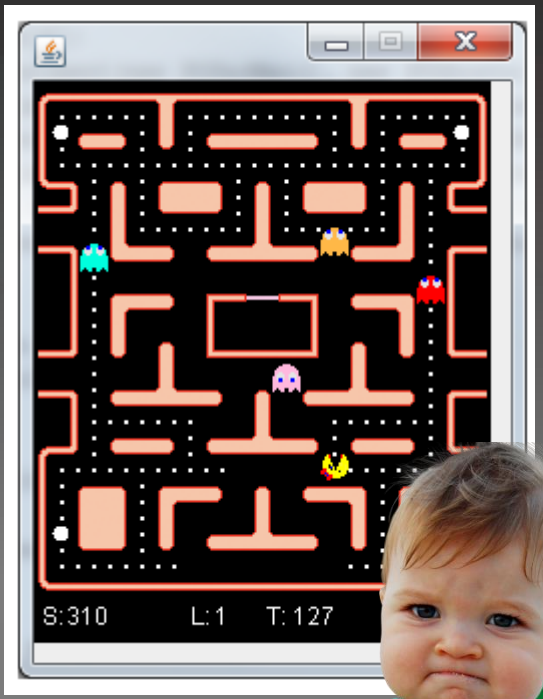
Eclipse

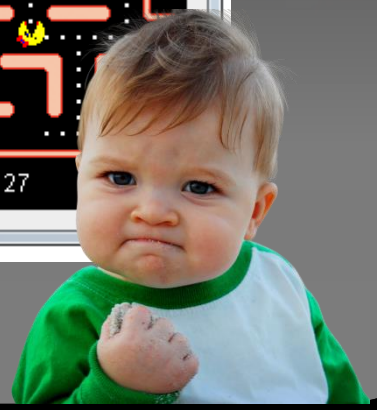
- In the following we will go through setting up Eclipse for working on the competition. Instructions for using IntelliJ are available at: http://www.pacmanvghosts.co.uk/guide_intellij.html
- Open a workspace of your choosing
- Import the project folder as „Existing Maven Projects“
- Open the „pom.xml“ file and write your teams name at „Artifact Id“
- Rename the packages to:
  - „entrants.ghosts.username“ „entrants.ghosts.<teamname>“
  - „entrants.pacman.username“ „entrants.pacman.<teamname>“
- Rename the Project if you like

First Test

- Right click the Project -> Run As –> Java Application
- Run POPacManTest
- Now Right click the Project -> BuildPath -> Configure Build Path
- Remove all excluded entries under PacMan/src/main/java

In both cases you should see something like this:





Lets start to code!

- We just have a quick look at some Standard classes
- In your assignment you need to fill MyPacMan.java
  - + all Master students need to hand in Pinky, Blinky, Sue and Inky
- Often used classes will be Game, GameInfo, Maze and the Enum MOVE
- Check out the full API at:
  https://jar-download.com/java-documentation-javadoc.php?a=pacman-main&g=uk.co.pacmanvghosts&v=1.0.5.0

Useful Functions

Game

- getPacmanCurrentNodeIndex()
- getCurrentMaze()
- getDistance()
- getNextMoveTowardsTarget()
- getPossibleMoves()

- getPills()
- isPillStillAvailable()
- wasPillEaten()
  - Use this to update your internal gameInfo

- advanceGame()
  - Simulates the game one step forward

GameInfo

Provides an object for accessing storing the current believed gamestate.

Use this to store which pills you have eaten or where the ghosts possible are.

- getGameFromInfo()
  - Returns a game of the current believed game state

Maze

- Stores the overall layout and all pill positions

- pillIndices, powerPillIndices
- initialPacManNodeIndex
- initialGhostNodeIndex

MOVE

- Contains DOWN, UP, LEFT, RIGHT
- and the function opposite()

More highlights at: http://www.pacmanvghosts.co.uk/guide_api.html
full API: https://jar-download.com/java-documentation-javadoc.php?a=pacman-main&g=uk.co.pacmanvghosts&v=1.0.5.0

Random Direction

Let us start simple:

- First get the current position of PacMan
- Check the Game for all possible moves
- Return a random move

You can download this at our lecture page: „RandomPacMan.java“

```java
package entrants.pacman.ADockhorn;

import java.util.Random;

import pacman.controllers.PacmanController;
import pacman.game.Constants.MOVE;
import pacman.game.Game;


public class RandomPacMan extends PacmanController {

    public MOVE getMove(Game game, long timeDue) {

        /* get the current position of PacMan (returns -1 in
        case you can't see PacMan) */
        int myNodeIndex = game.getPacmanCurrentNodeIndex();

        // get all possible moves at the queried position
        MOVE[] myMoves = game.getPossibleMoves(myNodeIndex);

        // return a random available move
        int rndIdx = new Random().nextInt(myMoves.length);
        return myMoves[rndIdx];
    }
}
```

Random at Junction

Only change directions at a junction:

- If junction: getRandomMove
- If corner: don‘t move back
- If hallway: continue your path

You can download this at our lecture page: „RandomJunctionPacMan.java“

```java
public MOVE getMove(Game game, long timeDue) {
    /* get the current position of PacMan (returns -1 in case
    you can't see PacMan) */
    int myNodeIndex = game.getPacmanCurrentNodeIndex();

    // get all possible moves at the queried position
    MOVE[] myMoves = game.getPossibleMoves(myNodeIndex);

    // choose random direction at junction
    if (game.isJunction(myNodeIndex))
    {
        // return a random available move
        int rndIdx = new Random().nextInt(myMoves.length);
        return myMoves[rndIdx];

    } else {
        // check if the lastMove is still available (hallways)
        MOVE lastMove = game.getPacmanLastMoveMade();
        if (Arrays.asList(myMoves).contains(lastMove)){
            return lastMove;
        }

        // don't go back (corner)
        for (MOVE move : myMoves){
            if (move != lastMove.opposite()){
                return move;
            }
        }

    }

    // default
    return game.getPacmanLastMoveMade().opposite();
}
```

DemoPacMan

- Create a game with partial observation
- Create some simple ghosts which can be used for a forward model
- Rate the outcome of all possible moves
- … and choose the best one

This example is provided in the base package „examples.demo.DemoPacMan.java“

We will discuss further examples at the end of each exercise class.

```java
public MOVE getMove(Game game, long timeDue) {
    Game coGame;
    if (game.isGamePo()) {
        GameInfo info = game.getPopulatedGameInfo();
        info.fixGhosts((ghost) -> new Ghost(
                ghost,
                game.getCurrentMaze().lairNodeIndex,
                -1,
                -1,
                MOVE.NEUTRAL
        ));
        coGame = game.getGameFromInfo(info);

    } else {
        coGame = game.copy();
    }

    // Make some ghosts
    MASController ghosts = new POCommGhosts(50);
    // Ask what they would do
    EnumMap<GHOST, MOVE> ghostMoves = ghosts.getMove(coGame.copy(), 40);

    // Get the best one step lookahead move
    MOVE bestMove = null;
    int bestScore = -Integer.MAX_VALUE;
    for (MOVE move : MOVE.values()) {
        Game forwardCopy = coGame.copy();
        forwardCopy.advanceGame(move, ghostMoves);
        int score = forwardCopy.getScore();
        if (score > bestScore) {
            bestMove = move;
            bestScore = score;
        }
    }

    return bestMove;
}
```

References

1) Williams, P. R., Perez-Liebana, D., & Lucas, S. M. (2016). Ms. Pac-Man Versus Ghost Team CIG 2016 competition. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)* (pp. 1–8). IEEE

2) Rohlfshagen, P., & Lucas, S. M. (2011). Ms. Pac-Man Versus Ghost Team CEC 2011 Competition, 70–77

3) Ms Pac-Man Competition. (n.d.). Retrieved from http://cswww.essex.ac.uk/staff/sml/pacman/PacManContest.html

4) Lucas, S. M. (2007). Ms Pac-Man competition. *ACM SIGEVOlution*, *2*(4), 37–38

5) API: https://jar-download.com/java-documentation-javadoc.php?a=pacman-main&g=uk.co.pacmanvghosts&v=1.0.5.0