Develop a controller/agent (based on AI/machine learning?) for “Super Mario Bros”
Infinite Mario Bros

- by Markus Persson
- quite faithful SMB 1/3 clone
- in Java
- random level generation
- open source
Our changes

• Rewrite the control loop
• Allow for 1000 times speed-up in headless mode
• Create an interface for controllers
Interface

• Each time step (24 fps), the agent gets a representation of the environment
  • Enemies and “blocks” around Mario
  • Fine position, jumping state
• And returns an action
  • 5 bits: left, right, down, A, B
Interface
Interface

// always the same dimensionality 22x22
// always centered on the agent
public byte[][] getCompleteObservation();
public byte[][] getEnemiesObservation();
public byte[][] getLevelSceneObservation();
public float[] getMarioFloatPos();
public float[] getEnemiesFloatPos();
public boolean isMarioOnGround();
public boolean mayMarioJump();

public enum AGENT_TYPE
    {AI, HUMAN, TCP_SERVER}

public void reset();
public boolean[] getAction
    (Environment observation);
public AGENT_TYPE getType();
public String getName();
public void setName(String name);
A very simple agent

```java
public boolean[] getAction(Environment observation) {
    action[Mario.KEY_SPEED] =
    action[Mario.KEY_JUMP] =
    observation.mayMarioJump() || !
    observation.isMarioOnGround();

    return action;
}
```
Neural network agent

```java
for (int i = -3; i < 4; i++) {
    for (int j = -3; j < 4; j++) {
        inputs[which++] = probe(i, j, scene);
    }
}
inputs[inputs.length - 3] =
    observation.isMarioOnGround() ? 1 : 0;
ininputs[inputs.length - 2] =
    observation.mayMarioJump() ? 1 : 0;
ininputs[inputs.length - 1] = 1;
double[] outputs = mlp.propagate(inputs);
for (int i = 0; i < action.length; i++) {
    action[i] = outputs[i] > 0;
return action;
```
Goal of the competition

- Develop an agent that gets as far as possible...
- ...on as many levels as possible...
- ...which are previously unseen
- Scoring: progress on 40 randomly generated levels
Main rules

- Implement the Agent interface (or connect to the TCPAgent)
- Use **only** information from the Environment interface
- Don’t take more than 40 ms per time step
- Follow the submission instructions...
Challenges

- Handle a large state/observation space
- Handle very different situations (unlike e.g. car racing)
- Tactical tradeoffs (go back and get the power-up?)
What we thought would work

- Rule-based systems, with handcrafted complicated feature detectors
  - To handle the large observation space
  - Tuned by e.g. artificial evolution
  - To handle the large parameter space
- Or TD-learning
Presentations of competitors
AN A* MARIO AI

Using path-finding to find the optimal jump
IDEA

- Analyse Mario’s physics engine to obtain movement equations for all objects
- Create our own physics engine that can predict next world state
- Plug engine into an A* algorithm to evaluate fitness of each node
- Heuristic: How long before Mario reaches goal?
- Penalty for falling into gaps or being hurt
- Ignore coins, enemies, power-ups (for now!)
A* ALGORITHM

- Best–first graph search algorithm
- Need heuristic that estimates remaining distance
- Keep set of “open” nodes (initially: start node)
- While open set not empty:
  - Pick node in open set with \textbf{lowest estimated total distance from start to goal}
  - If node == goal: finish. Create path by backtracking through ancestors.
  - Generate child nodes, put them into open list (only if better than existing nodes for that location)
- If heuristic admissible (always underestimating), we then have the shortest path to goal.
A* IN MARIO: CURRENT POSITION

Goal: right border of screen

current node
A* IN MARIO: CHILD NODES

- left, jump, speed
- current node
- jump
- right, jump
- right, speed
A* IN MARIO: BEST FIRST

current node

right, speed
A* IN MARIO: EVALUATE NODE

current node

right, speed

Thursday, August 27, 2009
A* IN MARIO: BACKTRACK

current node

right, jump, speed

right, speed
A* IN MARIO: BEST FIRST

current node

right, jump, speed

right, speed
A* IN MARIO: EVALUATE

current node
A* IN MARIO: CREATE CHILDREN

current node
A* IN MARIO: BEST FIRST
HEURISTIC

- Using Mario’s current speed and acceleration, how long does it take to reach the goal?
- Assume maximum acceleration and no obstacles (admissible heuristic!)

\[
\begin{align*}
xa &= xa + 1.2 \\
    x &= x + xa \\
   xa &= xa \times 0.89
\end{align*}
\]

- Optimisation: Find a closed form for this.
HANDLING NEW EVENTS

- Plan ahead for two ticks (=1/12 sec)
- Synchronise internal world–state with received enemies and object positions.

Possible Improvements:
- Keep & update old plan instead of starting from scratch each time
- Collect coins & power–ups (e.g., using a high–level planner that pans out the route between power–ups)
Glenn Hartmann

• Modified version of one of the heuristic agents that came with the software
• Move forward
• Jump if in danger of falling
• Jump over enemies if safe
• Shoot continuously
Rafael Oliveira

- Did not submit any documentation
- Seems to be an elaborate heuristic
• A-star search to maximize x position
• Partial simulation to anticipate future positions (recalculated if simulation goes out of sync)
• Some pruning of search tree
Sergio Lopez

• Rule-based system, to answer 2 questions: “should I jump?” and “which type of jump?”

• Evaluates possible landing points based on environment info and heuristics (no simulation)

• Calculates “danger value” for each action, and “need to jump”

• Special situations, e.g. waiting for flowers and bullets to go away, climbing “stairs”
• Subsumption-type controller: later layers can override the action of earlier layers

• Each layer either a method or a state machine
• **avanzar()** -> makes Mario going forward

• **saltarParedes()** -> makes Mario jump when necessary for advance

• **subirEscaleras()** -> makes Mario climb "stairs" (these mean of rocks)

• **saltarPozos()** -> makes Mario jump over gaps

• **saltarEnemigos()** -> makes Mario jump over enemies

• **dispararEnemigos()** -> makes Mario shoot enemies

• **evitarArrollarEnemigos()** -> makes Mario going back to avoid enemies while in air
• Joint work with Caleb Anderson and Peter Burns
• Based on A star
• Separate simulation of the game physics (not using the game engine)
• (imperfect) prediction of enemies’ movements
• Working towards propagating penalties in the tree
Erek Speed

- Rule-based system
- Maps the whole observation space to the action space
  - antecedent: 22x22 array, consequent: 6 bits action
  - put in hash table
- Evolved with a GA
  - Genome as > 100 Mb XML file!
State machine with 4 states: walk_forward, walk_backward, jump, jump_hole
Results
<table>
<thead>
<tr>
<th>Name</th>
<th>Score</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robin Baumgarten</td>
<td>17264</td>
<td>5.62</td>
</tr>
<tr>
<td>Peter Lawford</td>
<td>17261</td>
<td>6.99</td>
</tr>
<tr>
<td>Andy Sloane</td>
<td>16219</td>
<td>15.19</td>
</tr>
<tr>
<td>Sergio Lopez</td>
<td>12439</td>
<td>0.04</td>
</tr>
<tr>
<td>Mario Pérez</td>
<td>8952</td>
<td>0.03</td>
</tr>
<tr>
<td>Rafael Oliveira</td>
<td>8251</td>
<td>?</td>
</tr>
<tr>
<td>Michal Tuláček</td>
<td>6668</td>
<td>0.03</td>
</tr>
<tr>
<td>Erek Speed</td>
<td>2896</td>
<td>0.03</td>
</tr>
<tr>
<td>Glenn Hartmann</td>
<td>1170</td>
<td>0.06</td>
</tr>
<tr>
<td>our evolved neural net</td>
<td>7805</td>
<td>0.04</td>
</tr>
<tr>
<td>ForwardJumpingAgent</td>
<td>9361</td>
<td>0.0007</td>
</tr>
</tbody>
</table>
Observations

• The best-performing controllers take much longer time per time step (frame)

• This is because they use A star search!

• ...and these work well because of the lack of blind alleys (should be fixed)

• But some heuristic controllers do very well

• Not a lot of learning/optimization techniques (though many competitors claim to be working on it)
Next phase: CIG 2009

- Milan, Italy, 7-11 September
- Submission deadline: 3 sept.
- Minor additions to the interface
- Fully backward-compatible: all agents submitted for this phase will work...
  - ...and will be automatically entered
- Still time for you to submit your agent!
After the competition

• Competition web page will remain, complete with competition software

• ...which you can use in your teaching or research!

• Complete source code of all submitted controllers
The future of the Mario Competition

- Mario AI Championship 2010
- Run at 2 to 4 different conferences?
- More than one track, ideas include:
  - Agent time-budget track
  - Online learning of unseen level track
  - Personalized level generation track
  - (your idea here)