SamurAI Jockey Game Rule

IPSJ Programming Contest Committee

2017/12/26

Abstract

This document presents the game rules for SamurAI Coding 2017–18.

1 Game Outline

Two players controlled by AI programs, changing their positions step by step, compete for time to goal in a course with obstacles.

One game consists of two races with the same course, exchanging the start positions of two players. The player with the smaller sum of goal times of two races wins the game. If the goal time sums are equal, the game is a draw.

2 Race Course

The race courses are a two dimensional grid, with different sizes and obstacle positions for different games. The width of the race course is w and its length is l in what follows. A grid point on the race course has coordinates (x, y) ($0 \leq x < w, 0 \leq y$).

Some of the grid points on the race course are *obstacle points.* Obstacle points and line segments connecting them are *obstacles.* Obstacles are fixed in one race. The AI programs controlling players (called *AI,* in what follows), however, are given information of only those obstacles inside the players' fields of vision, that is, those with their y-coordinates within a certain limit from the y-coordinates of the players' positions. As only limited acceleration and deceleration are possible, obstacles may become unavoidable with velocity too large.

Figure 1 shows an example of a race course. The brown dots are obstacle points, brown line segments are obstacles connecting obstacle points, and the brown areas are those surrounded by obstacles into which players cannot enter.

During one race, two players are on different grid points. The start points of two players are with y-coordinates of 0 and different x-coordinates. As the race progresses, players change their positions step by step. Players finish the race when they have reached a point with the y-coordinate greater than or equal to the course length l.

There is never a dead end in the course. A player can

Figure 1: Race Course Example

move to a point with larger y-coordinate from any grid point of the race course that can be reached from the start points of the two players, never decreasing its y-coordinate during the move, if moves are with velocity small enough.
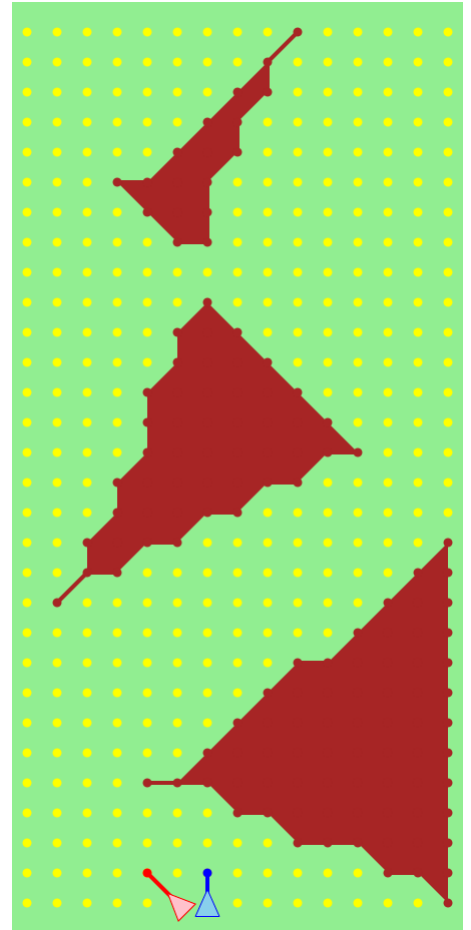
3 Race Process

On every step of a race, both AI's are given information on the race situation, and the situation is updated according to the acceleration/deceleration instruction given by the players in response. Two players never occupy the same position at a time during a race.

The race starts from step 0 and the step number is incremented one by one. The race ends when both of the players finish or become *disqualified* (Section 3.6). If one or more players can't finish within the step limit, however, the race ends at that step and the unfinished player is disqualified.

3.1 Time Control

A limit is imposed on the total of time a player can use in one race. The clock starts or resumes when the game manager program has finished sending information to the AI and stops when it has finished receiving the response from the AI. The player running out of the time is disqualified.

3.2 Player States and Acceleration Instructions

A player has its *position* (x, y) and *velocity* (v_x, v_y) at each step.

A player instructs *acceleration*— (a_x, a_y) at each step. Both a_x and a_y should be one of $-1, 0$, or 1.

When the player can make normal move without committing course out and collision with the opponent without priority (described below), the position of the player at the next step will be $(x + v_x + a_x, y + v_y + a_y)$. This position is called the *planned position* in what follows. The velocity of the player in the next step will be $(v_x + a_x, v_y + a_y)$ whether or not the player can make normal move.

3.3 Movement Lines and Course Outs

The line segment connecting the players position and its planned position is called the *movement line* of the player.

The player commits a *course out* if one or more of the following hold.

- The player goes out of the course, that is, the coordinates of planned position (x, y) does not satisfy both $0 \leq x < w$ and $0 \leq y$.

- The movement line goes over or touches one or more obstacles.

Figure 2: Course Outs
Red movement lines cause
course outs.

The position of a player that committed a course out cannot move: its position in the next step remains the same. The acceleration is applied the same as when no course outs are committed.
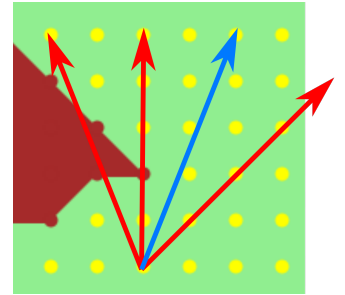
3.4 Collisions and Priorities

When no course outs take place and movement lines of two players touch or intersect, a *collision* takes place.

On a collision, the player with the *priority* will move to its planned position, but one without the priority will be made to stay in the same position. The acceleration is applied the same as when no collisions take place.

The player with the smaller y-coordinate of its position has the priority. When two players have the same y-coordinate, one with the smaller x-coordinate has the priority. An exception is when the movement line of a player reaches or passes the opponent's original position. In this case, the priority is given to the opponent.

When movement lines of both players reaches or passes the opponents' position, both lose the priority and cannot move at that step.

3.5 Goal Time

When a player at (x, y) at step s moves to (x', y'), and $y' \geq l$ (l is the course length) holds, that player finishes the race. The goal time of the player is $s + (l - y)/(y' - y)$.

A player that committed a course out or made a collision without priority is stopped, and even if its planned position has y-coordinate greater than or equal to the course length, the player does not finish the race. [1]

The player that finished the race is removed from the course. The race continues if the opponent player has not finished the race or been disqualified.
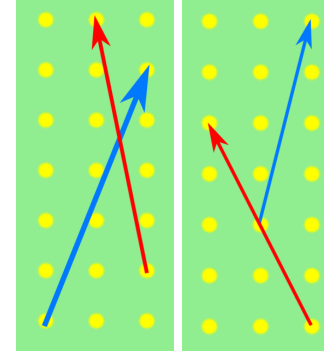

Figure 3: Collision
The player with the blue movement line has the priority.

3.6 Disqualification

A player is disqualified in either one of the following cases.

- The player can't finish within the step limit (the preface of Section 3).

- The player exceeds the time limit (Section 3.1).

- The output from the AI violates the regulations in Sections 4.2 and 4.4.

The disqualified player is removed from the course. The race continues if the opponent player has not finished the race or been disqualified. The goal time of the disqualified player is twice the step limit.

Note that the extent of the disqualification is limited to the current race; the disqualified player can participate in subsequent races without any penalties.

4 Behavior of AI's

The AI receives information related to the race as a whole at its initiation, and responds with an acknowledgment of receiving it. At each step, it receives the race state information and responds with an acceleration instruction.

The input of the AI is a sequence of decimal integers, separated by one space or newline. Negative values are prefixed with a minus sign. The initial game information and game state information at each step end with a newline.

AI should respond sequences of decimal integers, separated by spaces or a newline. Negative values should be prefixed with a minus sign. **A newline should be placed at the ends of responses.**

4.1 Initial Input

At initiation, an AI receives the following input items in this order. A newline is put at the end of each item. Multiple integers within one item (line) are separated by **one space character**.

Remaining Time (t_{given}) The remaining time usable in this race in milliseconds as an integer.

Step Limit (s_{max}) The limit of number of steps to finish as an integer.

Course Size The width (w) and length (l) of the race course as two integers.

---

[1] Note that there may be obstacles on and beyond the goal line.

Vision Limit (d) The depth of the field of vision as an integer d. When the player is at (x, y), information of the opponent and obstacle points are given for those at y-coordinates between $y - d$ and $y + d$, inclusive.

Following is the format of the initial input.

$$t_{\text{given}}\hookleftarrow$$
$$s_{\text{max}}\hookleftarrow$$
$$w\;\sqcup\;l\hookleftarrow$$
$$d\hookleftarrow$$

4.2 Initial Output

An AI should respond with an integer 0 and a newline to acknowledge that its initiation is done.

$$0\hookleftarrow$$

4.3 Per Step Input

At each step, an AI receives the following items in this order. A newline is put at the end of each item. Multiple integers included in one item are delimited by **one space character**.

Step (s) The current step number as a single integer.

Remaining Time (t_{left}) The remaining time usable in this race in milliseconds as an integer.

Player's State The x- and y-coordinates of the position (x and y) and the x and y components the velocity (v_{x} and v_{y}) of the player the AI is controlling, as four integers.

Opponent's State The x- and y-coordinates of the position (x_{e} and y_{e}) and the x and y components the velocity ($v_{\text{x}_{\text{e}}}$ and $v_{\text{y}_{\text{e}}}$) of the opponent player, as four integers. When the opponent player is out of the vision field, -1 is indicated for the y-coordinate of the position and 0 for the remaining.

Obstacle Points Information on whether grid points in the vision field are obstacle points or not. It consists of $(2 \times d + 1) \times w$ integers. For every y in the range $y_{\text{min}} \leq y \leq y_{\text{max}}$, w integers $o_{0,y}, o_{1,y}, \ldots, o_{w-1,y}$ are given. $o_{x,y}$ being 1 means that the grid point with its coordinates (x, y) is an obstacle point, and 0 means that it is not. 1 is given for grid points out of the course, that are, those with $y < 0$. A newline is used as a delimiter instead of a space character between $o_{w-1,y}$ and $o_{0,y+1}$. Thus, this information consists of integers of $2 \times d + 1$ rows by w columns.

The following is the format of the per step input.

$$s\hookleftarrow$$
$$t_{\text{left}}\hookleftarrow$$
$$x\;\sqcup\;y\;\sqcup\;v_{\text{x}}\;\sqcup\;v_{\text{y}}\hookleftarrow$$
$$x_{\text{e}}\;\sqcup\;y_{\text{e}}\;\sqcup\;v_{\text{x}_{\text{e}}}\;\sqcup\;v_{\text{y}_{\text{e}}}\hookleftarrow$$
$$o_{0,y-d}\;\sqcup\;o_{1,y-d}\;\sqcup\;\cdots\;\sqcup\;o_{w-1,y-d}\hookleftarrow$$
$$o_{0,y-d+1}\;\sqcup\;o_{1,y-d+1}\;\sqcup\;\cdots\;\sqcup\;o_{w-1,y-d+1}\hookleftarrow$$
$$\cdots$$
$$o_{0,y+d}\;\sqcup\;o_{1,y+d}\;\sqcup\;\cdots\;\sqcup\;o_{w-1,y+d}\hookleftarrow$$

4.4 Per Step Output

An AI should output x- and y-components of the acceleration at the clock with two integers a_x and a_y as two integers. a_x and a_y should be one of -1, 0, or 1. The integers should be delimited by one or more space characters, and a newline should be put after them. The following is the format of the per step output. Note that there can be more than one space characters between a_x and a_y.

$a_x \,\llcorner\, a_y \hookleftarrow$

4.5 Input and Output Example

The text on the right is an example of the input/output to/from an AI during the initiation and the first two steps. The black text is the input and the red one is the output.

The first four lines, being the initial input, indicate that the remaining time, step limit, course width, course length and vision limit are $20,000\,\mathrm{ms}$, 100, 15, 100 and 8 respectively.

When the AI receives this input, it should output 0 and a newline.

Then the AI receives the input of the first step. 0 and 19981 in the first two lines show the current step number and the remaining time, respectively. The AI can see that it took $19\,\mathrm{ms}$ for its initiation and $19,981\,\mathrm{ms}$ remain.

The next two lines show the state of this and the opponent AI players. The current position (x, y) and the velocity (v_x, v_y) of this player are $(5, 0)$ and $(0, 0)$, respectively. The position (x_e, y_e) and the velocity (v_{x_e}, v_{y_e}) of the opponent player are $(9, 0)$, and $(0, 0)$, respectively.

The next $17 (= 8 \times 2 + 1$ because the vision limit is 8) lines give the information on whether grid points in the vision field are obstacle points or not. It is discovered that all the grid points where $y < 0$ are obstacle points and there is a large obstacle object straight in front of the player.

When the AI receives the input, it should output two integers as an acceleration (a_x, a_y); it output $(-1, 1)$ in this example. Then the AI receives the input of the second step in the same format to the first step. Then the AI should output an acceleration again. Such inputs and outputs are repeated until the race ends.

```
20000
100
15  100
8
0
0
19981
5 0 0 0
9 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1 0 0 0 0 0 0 0
0 0 0 0 1 1 1 1 0 0 0 0 0 0 0
0 0 0 0 1 1 1 1 0 0 0 0 0 0 0
0 0 0 0 1 1 1 1 1 0 0 0 0 0 0
-1 1
1
19955
4 1 -1 1
8 1 -1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1 0 0 0 0 0 0 0
0 0 0 0 1 1 1 1 0 0 0 0 0 0 0
0 0 0 0 1 1 1 1 0 0 0 0 0 0 0
0 0 0 1 1 1 1 1 1 0 0 0 0 0 0
0 1
```

4.6 Memory Duration

At each step, the game management system suspends the execution of AI when receiving its output has been completed, and resumes it after giving per step information. Thus AI cannot continue its computation between steps but its execution context, including variable values, are kept during one whole race.

For each race, the game management system initiates AI from scratch in an environment in which file I/O and network accesses are disabled. Thus, AI cannot pass information between multiple races.