SamurAI Jockey Game Rule (draft)

IPSJ Programming Contest Committee

2017/09/16

Abstract

A draft version of the game rule for SamurAI Coding 2017–2018. Note that this is a draft and is subject to change.

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.

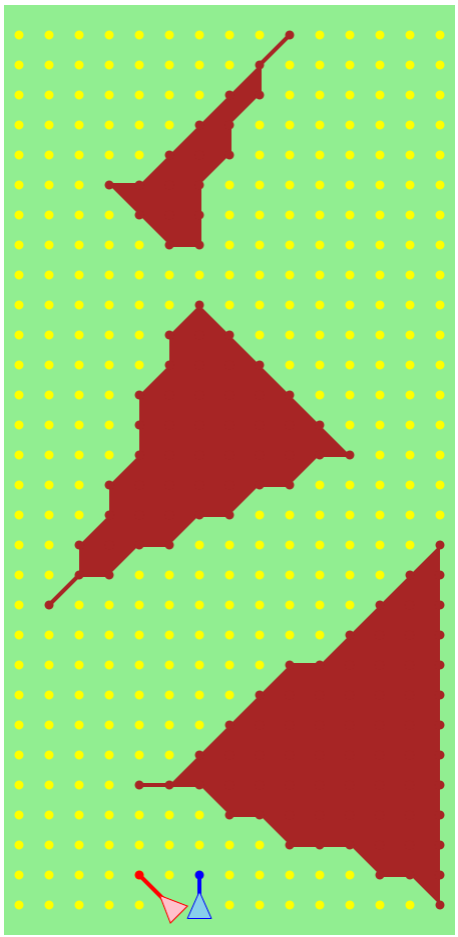

Figure 1: Race Course Example

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 move from any grid points of the race course to a point with larger y-coordinate, 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. If one or more players can't finish within the step limit, however, the race ends at that step. The goal time of the unfinished player is twice the step limit.

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 player AI and stops when it has finished receiving the response from the player AI. The player running out of the time loses the game including the race. [1]

3.2 Player States and Acceleration Instructions

A player has its *position* (x, y) and *velocity* $(v_\mathrm{x}, v_\mathrm{y})$ at each step.

A player instructs *acceleration*— $(a_\mathrm{x}, a_\mathrm{y})$ 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 $(x + v_\mathrm{x} + a_\mathrm{x}, y + v_\mathrm{y} + a_\mathrm{y})$. This position is called the *planned position* in what follows. The velocity of the player in the next step will be $(v_\mathrm{x} + a_\mathrm{x}, v_\mathrm{y} + a_\mathrm{y})$ 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.

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.

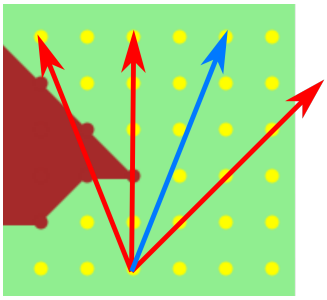

Figure 2: Course Outs
Red movement lines cause course outs.

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.

[1] The time limit is not fixed yet but is expected to be some tens of seconds.

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

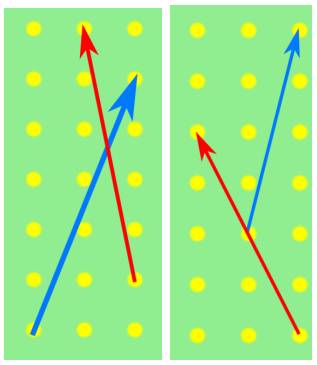

Figure 3: Collision
The player with the blue movement line has the priority.

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.

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 blank spaces or newlines. 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 blank spaces or newlines. 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, AI receives the following input items in this order.

Remaining Time(t_{given}) The remaining time usable in this race in microseconds 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.

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.

4.2 Initial Input Format

t_{given}↵
s_{max}↵
w l↵
d↵

4.3 Initial Output

A player AI should respond with an integer 0 to acknowledge that its initiation is done.

0↵

4.4 Per Step Input

At each step, AI receives the following items in this order.

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

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

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

Opponent's State The x- and y-coordinates of the position and the x and y components the velocity 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 or not 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 $y_{\min} \leq y \leq y_{\max}$, 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 rourse, that are, those with $y < 0$.

4.5 Per Step Input Format

s↵
t_{left}↵
x ␣ y ␣ v_x ␣ v_y↵
x_e ␣ x_e ␣ v_{x_e} ␣ v_{y_e}↵
$o_{0,y-d}$ ␣ $o_{1,y-d}$ ␣ $\cdots$ ␣ $o_{w-1,y-d}$↵
$o_{0,y-d+1}$ ␣ $o_{1,y-d+1}$ ␣ $\cdots$ ␣ $o_{w-1,y-d+1}$↵
$\cdots$
$o_{0,y+d}$ ␣ $o_{1,y+d}$ ␣ $\cdots$ ␣ $o_{w-1,y+d}$↵

4.6 Per Step Output

The player 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} ␣ a_{y}↵

4.7 Sample Input

右に実際の入力例を示す．これは初期化から 2 ステップまでの入力となっている．
初期化時の入力について，考慮時間，制限ステップ数，コースの幅，長さと視界がそれぞれ 20000, 100, 15, 100, 8 となっていることが分かる．
この入力を受け取ったらプレイヤーは 0 と改行をを出力しなければならない．
以降はステップごとの入力情報になるが，まず最初のステップについて現在のステップと残り考慮時間がそれぞれ 0, 19981 となっていることが確認できる．この AI は初期化時に 19 [µs] だけかかり，残り 19981 [µs] 残されているようだ．
次の 2 行は自プレイヤと相手プレイヤの状態が示されており，この AI の現在の位置は $x = 5, y = 0$ で速度は x 軸方向 y 軸方向ともに 0 となっている．
次に続く 17 行（視界:$8 \times 2 + 1$）は視界内の格子点が障害点か否かを示している．$y < 0$ におけるすべての格子点が障害点になることが分かる．前方には中央に障害物が集中しているようだ．
この入力を受けたら，加速度 (a_x, a_y) を出力する必要がある．すると，また次のステップの入力が与えられる．先ほどと同様の入力形式となっているが，どうやらこの AI は前ターンの加速度として $(-1, 1)$ を出力したようだ．

```
20000
100
15 100
8
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
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 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 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
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 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 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
```

4.8 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.