Circle Agent Displaying Generalization Ability for Geometry Friends

Hiroya Oonishi
Kyoto Institute of Technology
Kyoto, Japan
onishi6h@cis.is.kit.ac.jp

Hitoshi Iima
Kyoto Institute of Technology
Kyoto, Japan
iima@kit.ac.jp

***Abstract*—This report proposes our circle agent for Geometry Friends competition held at IEEE Conference on Computational Intelligence and Games in 2017.**

I. Introduction

This report proposes our circle agent for Geometry Friends competition held at IEEE Conference on Computational Intelligence and Games in 2017. Our agent is basically designed by the method which we recently proposed [1]. The method enables the agent to collect all the diamonds in any level by solving two sub-problems: one of finding the shortest path to collect all the diamonds and the other of selecting agent's actions to follow the found path. The first sub-problem is solved by applying a search algorithm to a directed graph which corresponds to a level, and the second sub-problem by a method based on reinforcement learning [2]. It is confirmed from experimental results that our method outperforms other existing ones which participated in past competitions.

However, there are some levels where the agent in [1] may fail to get a high score because of its limitations. In this report, we propose especially modifications of the agent design method in [1] to solve this problem. Section II briefly explains the agent design method in [1]. Section III and Section IV describe the limitations of the agent and the modifications on the agent design method respectively. Section V concludes this report.

II. Agent design method in [1]

The agent design method consists of two methods to solve the two sub-problems described in the previous section. This section briefly explains the two methods.

A. Method to solve the sub-problem of finding the shortest path

Prior to the definition of a path, a term "platform" is defined. As shown in Fig. 1, a platform is a flat obstacle which the agent can get on. The agent must jump or fall from a platform in order to move from the platform to another. It can collect each diamond by jumping or falling from a platform.

A path is defined as the order of platforms the agent moves to. Concretely, it is a sequence of edges in a directed graph whose node is a platform and whose edge is given if the agent can move from a platform to another. The shortest path in the directed graph is found by applying a search algorithm.

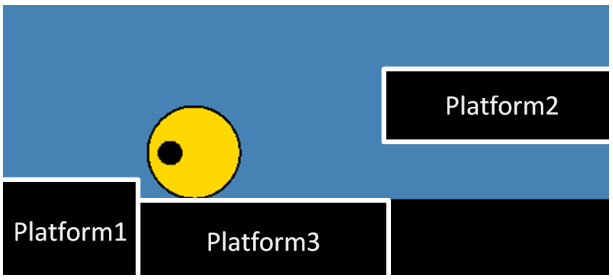

Fig. 1. Example of platforms

In order to generate the directed graph, the following two pieces of information is required:

- Platforms to which the agent can move from each platform,
- Diamonds which it can collect from each platform.

The information can be obtained through grasping the movement of the jumping or falling agent. The grasp is achieved by predicting trajectories of the jumping or falling agent.

The trajectories are predicted for two cases: no collision with obstacles (Case 1) and a collision with an obstacle (Case 2). In Case 1, it is assumed that a trajectory of the jumping or falling agent is a parabola. Hence, the trajectory is formulated by

$$x = v_x t \tag{1}$$

$$y = v_y t - \frac{1}{2} g t^2 \tag{2}$$

where t is the time and is set as zero when the agent begins to jump or fall. The variables x and y, respectively, are the ordinate and the abscissa, and the origin is set as the coordinates of the agent at $t = 0$. The constant parameters v_x and v_y, respectively, are the horizontal velocity and the vertical one of the agent at $t = 0$. The other constant parameter g is the acceleration of gravity. The values of the three constant parameters are determined by appropriate methods.

In Case 2, until the collision with an obstacle, a trajectory for Case 1 is used. We assume that once the agent collides with the obstacle, it simply falls straight down.

Using the predicted trajectories, a directed graph is generated. As mentioned above, an edge is give if the agent can move from a platform to another by jumping or falling. Its

length is defined as the distance from the current position of the agent to the position in which it lands via the position in which it jumps or falls. Then, the shortest path in the graph is found by applying Subgoal A* [3].

B. Method to solve the sub-problem of selecting the agent's actions

When the shortest path is found, the following information is obtained:

- Position in which the agent jumps or falls from each platform,
- Horizontal velocity with which it jumps or falls from each platform.

If the agent takes its actions which reach the position and velocity, it can follow the found path. Therefore, the sub-problem of selecting the agent's actions can be defined as an easy reinforcement learning problem whose objective is to reach the target position with the target velocity on a single platform.

In order to learn for any target position, states are defined by

- relative distance from the agent to the target position,
- velocity of the agent,

and actions are defined by

- rolling in the positive direction,
- rolling in the negative direction.

The positive direction is defined as the direction of the target velocity and the negative direction as the opposite direction. Rewards are basically defined as follows:

- $+200$ in the target state,
- -1 in the other states.

The agent learns using Q-learning [2]. Because the optimal actions depend on the target velocity, the agent learns independently for multiple reinforcement learning problems with various target velocities.

III. Limitations of the agent in [1]

The agent in [1] may fail to get a high score for levels in which there are

(A) narrow platforms,
(B) steps the agent must ascend without jumping,
(C) platforms it can move to only after colliding with a ceiling,
(D) diamonds it can collect not only by jumping but also by falling.

The limitation by (A) is due to the method to solve the sub-problem of selecting the agent's actions, and the limitations by the others are due to the method to solve the sub-problem of finding the shortest path. Each limitation is explained using an example in each of the succeeding subsections.

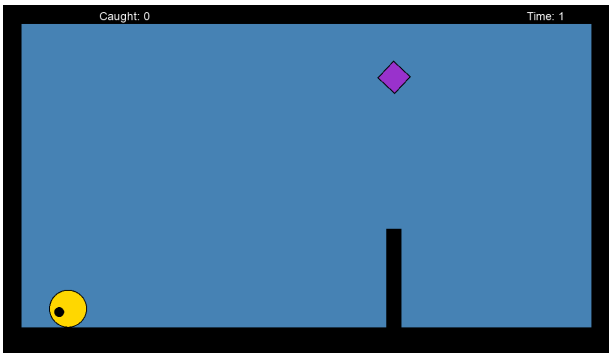

Fig. 2. A level with a narrow platform

A. Narrow platforms

In the level shown in Fig. 2, the agent must move to the narrow platform, stop on it, and then jump to collect the diamond. However, the agent in [1] often slips down from the narrow platform in landing on it and fails to collect the diamond.

B. Steps the agent must ascend without jumping

In the level shown in Fig. 3, the agent must ascend the step without jumping to move to the right platform, which is slightly different in level from the left, because it cannot move there by jumping. However, the agent in [1] cannot ascend the step and fails to collect the diamond.

C. Platforms the agent can move to only after colliding with a ceiling

In the level shown in Fig. 4, the agent can move to the right platform only after colliding with the ceiling to collect the diamond. However, the agent in [1] cannot move to the right platform and fails to collect the diamond.

D. Diamonds the agent can collect not only by jumping but also by falling

In the level shown in Fig. 5, although the agent can collect the diamond not only by jumping but also by falling from the left platform, it can collect the diamond sooner by falling than by jumping. However, the method in [1] makes the agent jump, which brings a low score.

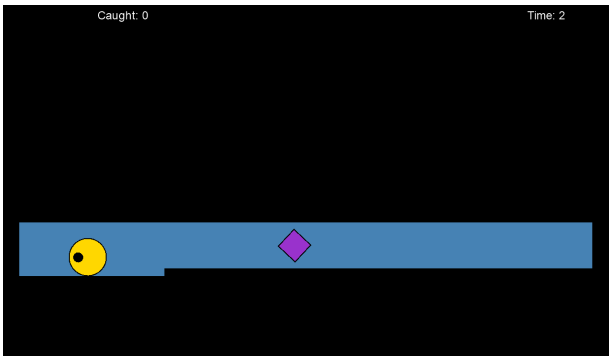

Fig. 3. A level with a step

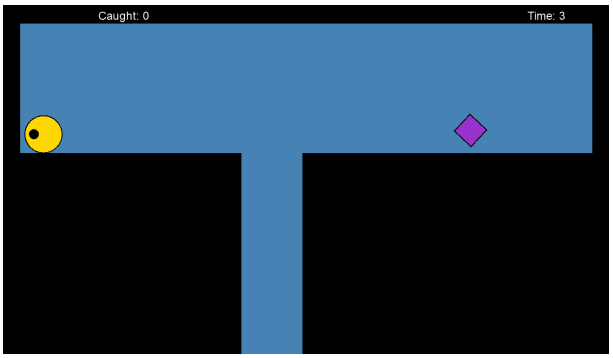

Fig. 4. A level where the agent inevitably collides with a ceiling

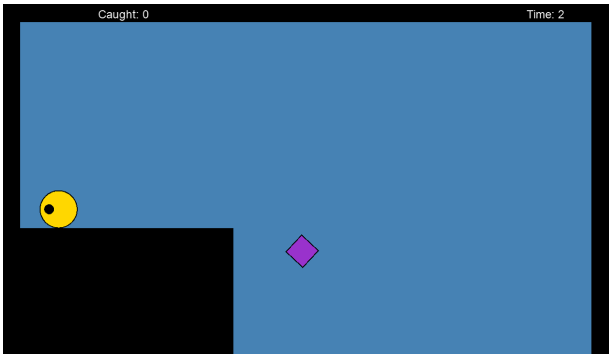

Fig. 5. A level with a diamond the agent can collect not only by jumping but also by falling

IV. Modifications on the agent design method in [1]

This section proposes the modifications on the agent design method in [1] to solve the problems described in the previous section.

A. The modification to overcome the limitation by narrow platforms

The limitation described in Subsection III.A is due to actions which the agent in [1] takes while jumping or falling. Although the agent can roll left or right even while jumping or falling, the agent in [1] takes no such actions, which causes slipping down from a narrow platform. The agent in the proposed method avoids slipping down by taking actions, which include rolling left or right, acquired by reinforcement learning while jumping or falling. The target position is set as the middle of the narrow platform, and the target velocity is set as zero.

B. The modification to overcome the limitation by steps the agent must ascend without jumping

The limitation described in Subsection III.B is due to an action that the agent in [1] takes to move between different platforms. Because the action is restricted to jumping or falling as mentioned in Subsection II.A, the agent cannot ascend a step. In the proposed method, the restriction is relaxed, and the agent can move between different platforms by jumping, falling or even rolling which enables itself to ascend the step.

C. The modification to overcome the limitation by platforms the agent can move to only after colliding with a ceiling

As mentioned in Subsection II.A, in the method to predict trajectories of the agent in [1], it is assumed that once the agent collides with a ceiling, it simply falls straight down. The limitation described in Subsection III.C is due to the assumption. Because trajectories based on the assumption are a little different from the real ones, there are some cases where edges in a directed graph are not correctly given even if the agent can actually move between platforms.

In the proposed method, the edges are more correctly given by making the trajectories more similar to the real ones. It is empirically confirmed that a trajectory is roughly a parabola even after a collision. Moreover, in the collision with a ceiling, the horizontal velocity and the vertical one of the agent just after the collision are roughly $V_x/3$ and $-V_y/3$, respectively, where V_x and V_y are the horizontal velocity and the vertical one just before the collision respectively. Therefore, the trajectory after the collision with the ceiling is predicted by the equations (1)(2) in which $V_x/3$ and $-V_y/3$ are substituted into v_x and v_y respectively.

D. The modification to overcome the limitation by diamonds the agent can collect not only by jumping but also by falling

The limitation described in Subsection III.D is due to the lengths of edges in a directed graph generated by the method in [1]. As mentioned in Subsection II.A, the length of each edge is defined as the distance from the current position to the position in which it lands via the position in which it jumps or falls. Therefore, the length of an edge for jumping may be shorter than the one for falling, which causes making the agent jump by mistake. In the proposed method, the length of each edge is redefined as the sum of the following two values:

- the distance from the current position of the agent to the position in which it jumps or falls,
- the length of the predicted trajectory of the jumping or falling agent.

The length of an edge for falling becomes shorter by this redefinition, and therefore the agent can collect the diamond sooner by falling.

V. Conclusion

We have proposed our circle agent for Geometry Friends competition held at IEEE Conference on Computational Intelligence and Games in 2017, especially with focusing on modifications on the agent design method in [1].

References

[1] H. Oonishi, H. Iima, “Improving Generalization Ability in a Puzzle Game Using Reinforcement Learning,” Proceedings of IEEE Conference on Computational Intelligence and Games, 2017 (in press).

[2] R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction*, A Bradford Book, 1998.

[3] D. Fischer, ”Development of Search-Based for the Physics-Based Simulation Game Geometry Friends,” Geometry Friends Game AI Competition, 2015.