Final Project

Benjamin Bentner Carl von Randow Thomas Ackermann

21. März 2019

Inhaltsverzeichnis

1 Reinforcement Learning Method **3**

2 Training Process **4**

2.1 Features 4

2.1.1 Nearest Coin - distance 4

2.1.2 Nearest Coin - direction 4

2.1.3 Mean Coin 5

3 Outlook **6**

1 Reinforcement Learning Method

2 Training Process

2.1 Features

2.1.1 Nearest Coin - distance

The first feature we implemented was the nearest coin feature. Since the first part of our training consisted of trying to collect all the coins in a game without opponents or crates and therefore without bombs the most crucial point we saw was the position of the coins on the gameboard. Since our primary goal at this point was to navigate to the next coin the most efficiently we ended up using only the position of the nearest coin regarding the agent position. Obviously we were aware that by taking only this information into account we would not always end up using the optimal way to collect all the coins but for a start we were satisfied with the use of this feature. In order to get the needed positions and differences we calculated the Euklidean distances from the agent position to all available coins and then chose the coin which had the minimum distance to our agent. In case of multiple coins with the same smallest distance we chose randomly(or in order?, depends on np.argmin). The Euklidean distance from the agent to the chosen nearest coin was our first feature and we thought about using ist in the reward section of the training for example by calculating the difference in the distance to the nearest coin after a step and rewarding or penalizing that step depending on wheater the distance grew or shrunk.

2.1.2 Nearest Coin - direction

In addition to the distance to the next coin we wanted to know, in which direction that coin lay. Therefore we calculated the difference in both the x- and the y-direction and divided the resulting coordinates by the total distance between coin and agent. That way we got the relative distance in each direction and could use that to navigate better to the nearest coin. To achieve that we penalized the relative distance in both directions. For example we subtracted the relative distance in the x-direction from the reward for moving right and added it to the reward for moving left. Since the board position increases to the right and downwards and we subtracted the coin position from the agent position to get the direction this is the correct use of the distance in the rewards. The distance in the y-direction was obviously used equivalently. This evidently added a lot of information to the mere distance of the nearest coin because now the agent knew what way he had to go in order to decrease the first feature which he did not know before. This gave us features two and three.

2.1.3 Mean Coin

Already while implementing the first three features that would only take the nearest coin into account we realized that there could exist cases where those alone would not give us the shortest total way when collecting all coins. To make the agent better aware of this and for him to realize such situiations we wanted to create a feature that could inform the agent of the position of all the coins at a given time. Our aim was for him to realize multiple coins being positioned in a certain area and then check out that specific area to collect all the coins that lay there. In order to achieve that we summed up the difference between the agent position and the position of all remaining coins. We figured that it would make sense to give the closer coins a stronger influence than others which were further away. Therefore we divided the resulting distances in both directions by the total distance to the power of 1.5.

3 Outlook