Desert Pit Escape Simulation Task

Introduction

For our project we are using the simulation environment, robotbenchmark. Robotbenchmark provides a set of robot programming tasks that cover a wide range of topics and difficulty levels, these benchmarks are freely available as online simulations based on a completely free open source software stack (*Robotbenchmark*, n.d.). The performance of users is tracked and displayed publicly in a leaderboard style.

The difficulty of these tasks are measured in stars, there are 5 difficulty levels. We chose a three star task named "pit escape". The agent is a BB-8 robot. To achieve success or a high accuracy result, the lost robot must climb out of a pit of sand as quickly as possible. This is accomplished by the robot creating enough momentum to propel itself out of the hollow. The agent will try to execute this action in under one minute. If this is not possible the best attempt of the agent escaping will be recorded. We are using python as the programming language and webots as the programming interface.

Components of Autonomous Behavior

Our autonomous agent, BB-8, is equipped with many components that enable its autonomous behavior. One important component is the gyro sensor. Gyroscopes are internal sensors that measure and regulate an agent's orientation and rotational speed, by utilizing the angular momentum conservation law (Passaro et al., 2017). We have previously learned that a sensor that measures anything within itself is called a proprioceptive sensor.

In terms of actuators, our agent is equipped with a single main motor that allows us to control its velocity. This motor plays a crucial role in the agent's movement.

The controller of our autonomous agent, also known as the brain, makes decisions based on perceived sensory information. The given controller of the BB-8 only includes back and forth movements, based solely on how many intervals have passed. The program keeps track of this internally and does not use any environmental information from the sensors. With our improvements, the code now uses the sensory information provided by the gyro sensor. The velocity that is being measured by the gyro sensor is used as a criterion for BB-8 to switch directions. If there is a loss of momentum detected, the controller will set the actuators to change directions. The threshold for this is when the velocity gets below 20. This threshold is predefined and does not change while running the simulation, hence, it is an open-loop control system. Open loop control systems are systems in which the parameters are predefined and do not change while the system runs.

The simulation takes place in a desert environment. The robot is surrounded by sand and is situated in a sand pit, from which the robot is trying to escape. The sandpit is quite deep, with steep walls that seem smooth, with no bulges or ridges which the robot could use to aid in its escape.

During the simulation, the goal of the autonomous agent is to escape a sandpit as quickly as possible. The agent must accumulate speed in order to have enough force to propel itself out of the pit. This is executed by first moving back and forth to gain velocity and once it has gained enough momentum, the agent can slowly climb the pit by circling the circumference until it has reached the top. Once the agent has reached the top of the pit, it continues straight and the simulation ends. The screen presents a pop-up with results. Our agent was able to escape the pit in 18.5 seconds with a controller performance rate of 84.69%.

Experience with actual and simulated agents

When running the simulation for the first time with the code provided by robotbenchmark, BB-8 is moving back and forth in the pit, but it does not seem to get a high enough velocity to get out. Because it has to move back after 1.5 intervals, it sometimes moves back while it could have gone further. In order to attempt to fix the velocity problem, we tried to make use of momentum. The first improvement we made was using the gyro sensor to measure the velocity of BB-8. Based on this measurement there was another criteria for BB-8 to stop. Namely, if BB-8 had slowed down too much, it would switch directions. This alone did not seem to have a considerable impact on the behavior of the robot since it still stopped after 1.5 intervals, even when BB-8 was still going at a high velocity. To improve this, we increased *timeInterval* to 2.5. The performance was slightly increased by this, because the robot was now able to come closer to escaping the pit, however it was still not enough to succeed. Afterwards we increased *timeInterval* to 5 to see if that would make a difference. In this scenario BB-8 was able to get out of the pit, but it had a performance rate of 53.57%, so there was still room for improvement.

We observed that it took BB-8 some time to reach a high velocity. For this reason we increased *maxSpeed* to 30, since the velocity of the motors is set to *maxSpeed*. To make up for the increase in velocity we also put the threshold for the velocity to change directions to 20. However, by just increasing maxSpeed and changing the threshold it still had the same performance. It did not seem to find momentum. The problem seemed to be that it changed directions too quickly, so in an attempt to fix this, we increased *timeInterval* again. Setting *timeInterval* to 10 drastically improved the performance. The performance went from 53.57% to 84.69%. With these settings BB-8 is moving in circles and is able to escape the pit using this method of moving in a circular motion to avoid the steep edges.