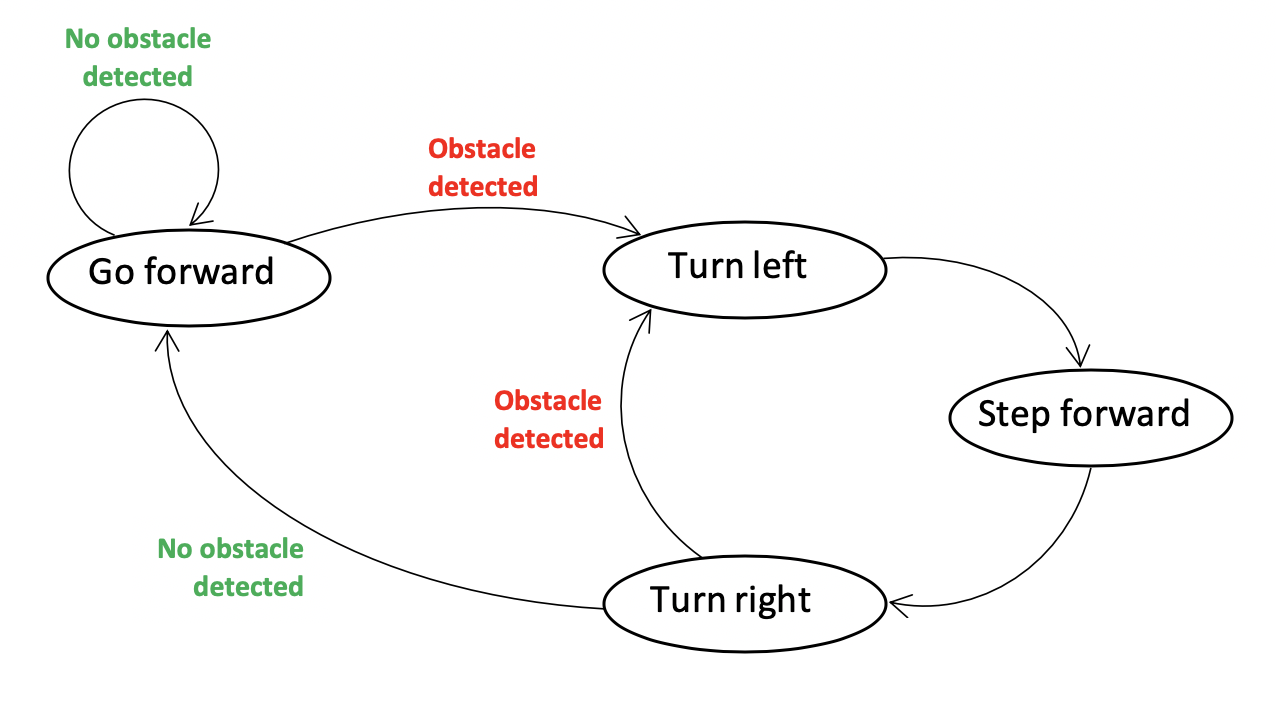
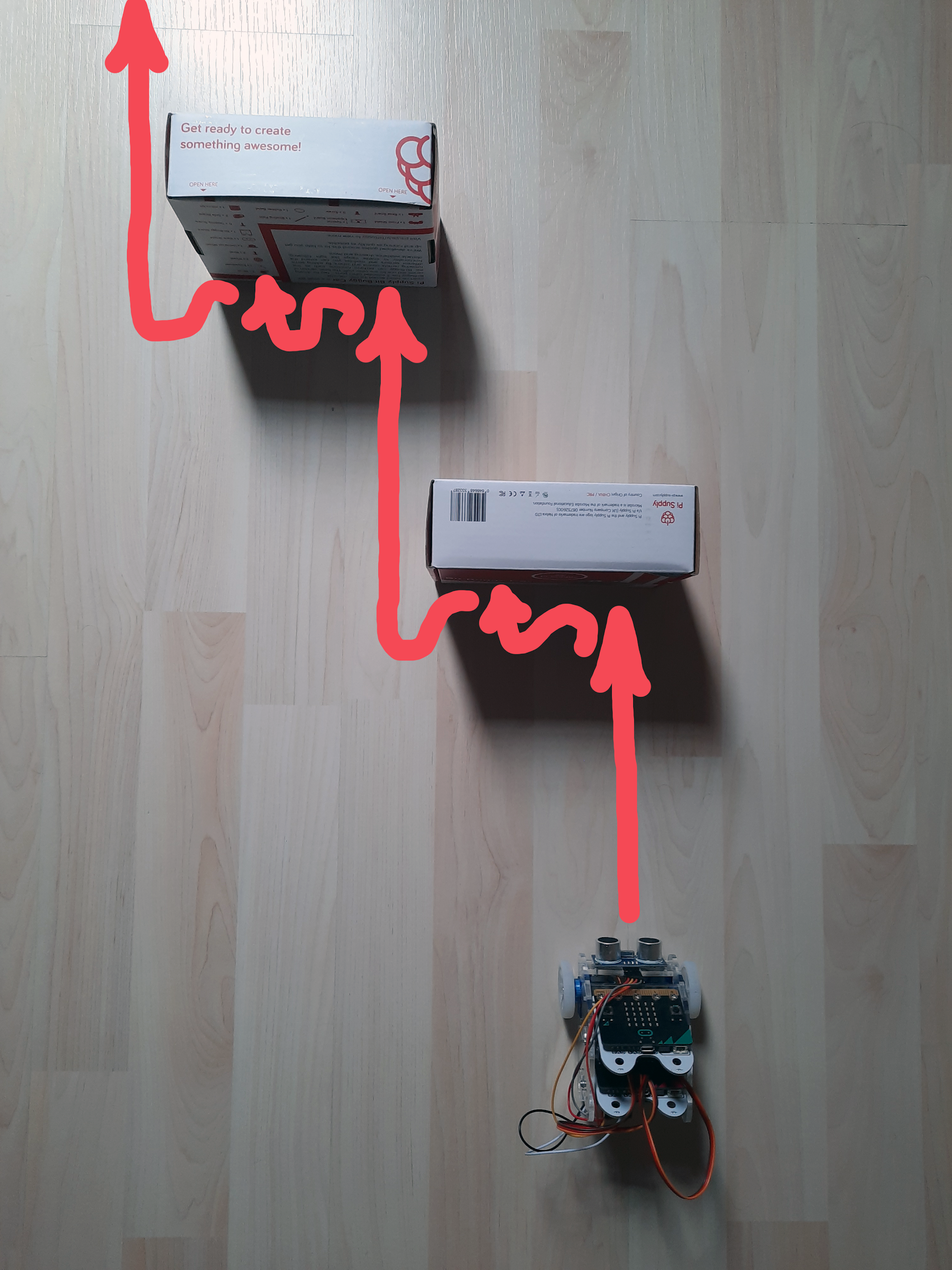
Avoiding obstacles challenge

The purpose of this exercise is to code a python script that will allow your buggy to bypass obstacles. More precisely, the robot will have to be programmed so that it always goes forward and as soon as it detects an obstacle in front of it, it turns left 90 degrees, moves forward a few centimetres and then turns right 90 degrees to check if the obstacle is still present. If the obstacle is still there, the robot then repeats this avoidance procedure and if not, it is free to continue straight ahead. The reason why we have to put in place such an avoidance mechanism is that we only have one front-end sensor (the HC-SR04) that we can rely on to detect possible obstacles. The procedure described above can be modeled with a finite state machine.

A finite state machine is a mathematical abstraction used to design algorithms. Explained concisely, a finite state machine is based on different robot states. Depending on the state the robot is in, specific commands are executed. The robot changes state according to what it perceives through its ultrasonic proximity sensor. The diagram below shows the state machine used to avoid obstacles from the left and at the same time to drive straight ahead when possible. The ellipses represent states and the arrows represent transitions.



As you can see, the four possible states of the robot are “Go forward”, “Turn left”, “Step forward” and “Turn right”. The robot can only be in one of those states at a time. Let’s consider an example and analyse the behaviour of the robot. Let’s assume we are in the situation of the video “Avoiding obstacles challenge - video.mp4” where we have two consecutive obstacles. Initially, the robot goes forward, it is in the state “Go forward”. Once it gets close enough to the first obstacle, it automatically begins the obstacle avoidance procedure and switches from the state “Go forward” to the state “Turn left”. The robot hence rotates 90 degrees to the left, automatically switches from state “Turn left” to state “Step forward” to advance parallel to the obstacle for a few centimetres and finally it switches from the state “Step forward” to the state “Turn right”, where it rotates 90 degrees to the right. Here, a decision has to be taken. Either the obstacle is still present and the obstacle avoidance procedure is restarted, or the obstacle has been bypassed and the robot can freely go forward as in the beginning. In the video example, we can observe that both obstacles are long enough for the robot to enter the avoidance procedure a second time. The condition for entering the obstacle avoidance loop is evaluated at two points in the state diagram: when the robot is moving forward (state “Go forward”) and at the end of its rotation to the right (state “Turn right”).



As you have seen, with the materials we have at our disposal, a micro:bit is not enough to design a robot capable of avoiding obstacles. You will have to build a robot based on two micro:bits. One of them will take care of integrating the distance measurements through the HC-SR04 and the other one will control the servos. The two micro:bits will communicate by radio. The micro:bit controlling the motors will take into account the distance measurements received from the other micro:bit and will be able to make decisions about necessary changes in the robot's state.