Documentation

TAS Project

Martin Gottwald

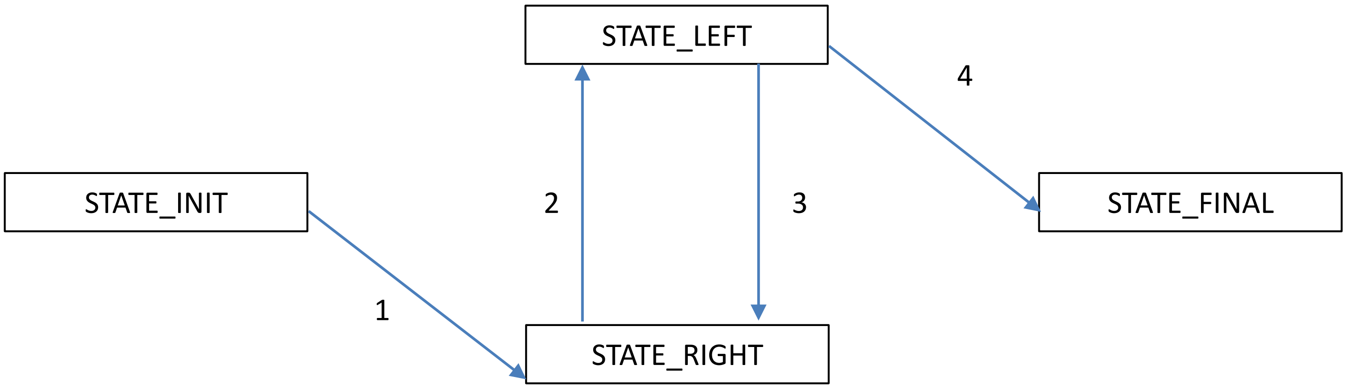
Slalom:

Main.cpp

- Similar layout like the TAS_autonomous_control -> same functionality of ‘c’ and ‘z’ of the Wiimote
- Contains a loop which calls the ‘RobotClass::update()’ function
- Sends each iteration the instruction to the Arduino board via the /servo topic
- Breaks the loop when the slalom is completed
- The update rate for this loop is controlled by the state machine in the robot class

RobotClass.cpp, RobotClass.h

- The constructor loads the text files with the trajectories
- The Update function generates the next instruction:
  - State machine with the states INIT, TURN_LEFT, TURN_RIGHT, FINAL
  - In each state the trajectory is executed (except the state FINAL)



  - The instruction gets the i-th velocity and angle
  - The thread sleeps for $t_i – t_{i-1}$ microseconds
  - The state is changed when the current trajectory is completed

wii_control_logger:

- Code from the original ‘wii_control’ node
- Removed all the publisher and added an output file stream instead

- At each update write the current pwm signal for velocity and angle to the file as well as the current time
- For the time stl::chrono library is used. The time is measured in microseconds

pose_estimation_tool:

- A simple ros node for sending a pose estimate to the localization
- Required for our project, because we were not able to send a pose estimate via the external rviz. Probably the virtual box, in which our Ubuntu is running, caused some trouble with the network.
- The code a number from the console and chooses the according corner with orientation from a list of hardcoded poses.
- Then this pose is published to the /initialpose topic.