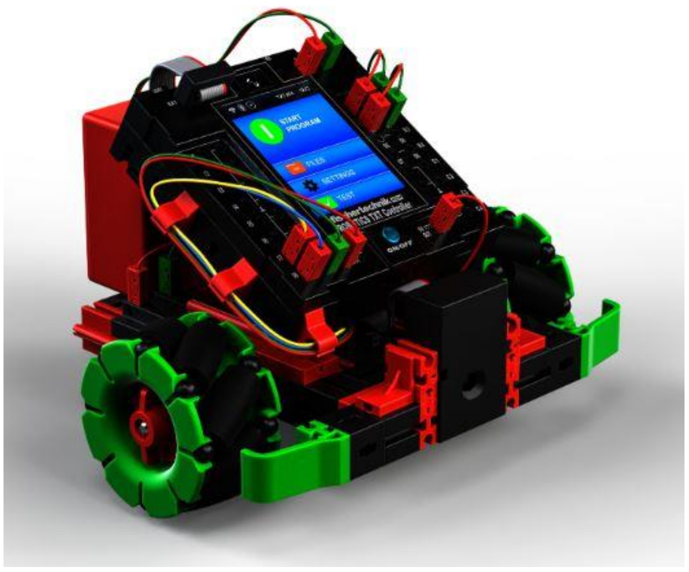
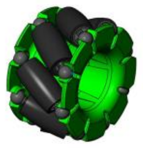
Advanced 2-wheel robot



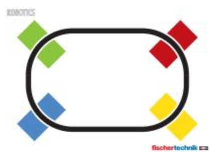
You will use this model to learn some more programming fundamentals. You are already familiar with the fischertechnik components you will use from previous tasks. You will control the model with both the IR track sensor and the gesture sensor.

Actuators, sensors, and technical accessories installed in the model:

Mini Motor	Gears	Gesture sensor	Omniwheel	IR track sensor
				

An explanation of the components is provided on the introductory page.

The "Advanced 2-wheel robot model" is divided into 4 programming tasks:

Task 1 Advanced_2-wheeled_robot_1.rpp	<u>Programming level 3</u> The robot travels straight ahead. If it encounters an obstacle, it should avoid it.
Task 2 Advanced_2-wheeled_robot_2.rpp 	<u>Programming level 3</u> The robot should travel along the black line, which is the specified obstacle course. If it leaves the black line, it should stop.

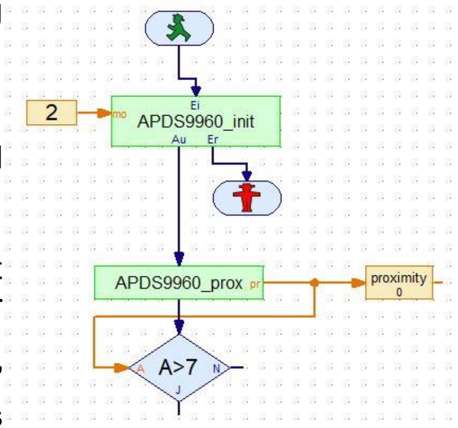
Task 3 Advanced_2-wheeled_robot_3.rpp	Programming level 3 The program should regulate its travel on the black line so that the model will find the black line once again if the robot leaves the black line.
Task 4 Advanced_2-wheeled_robot_4.rpp	Programming level 5 The robot travels along a track, and reduces its speed if it encounters an obstacle. The direction of rotation used to avoid the obstacle is set via a random generator.

Task 1

You can use several subroutines from previous programming tasks to solve this first task.

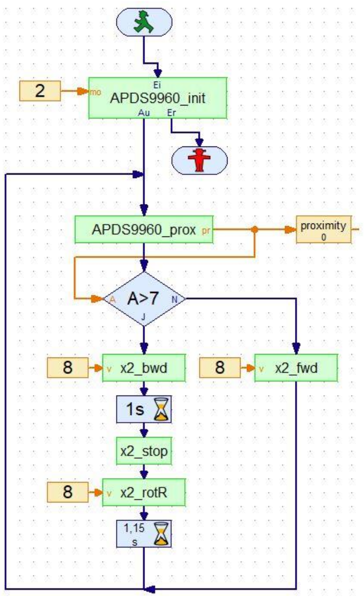
In the 2nd part of the program, you will need the following commands: **“x2_bwd”, “x2_stop”, “x2_rotR”, “x2_rotL” and “x2_fwd”.**

Just as with the “Twilight switch” model, the gesture sensor must be initialised first. To do so, insert the block “APDS9960_init” after the program start. The value for the input “mo” (mode) is set as 2, because you want to work with the function “APDS9960_prox” (distance measurement - proximity). The “Stop” command is connected to the “Er” output.



Then, insert the block “APDS9960_prox” to measure and evaluate the distance. The determined value is evaluated in the subsequent branch instruction. If the value is more than 7, “J” then it will continue with the second part of the program. If the value is less than 7, “N” the model will start up.

Then the section of the program follows where the model travels backwards for 1s, then stops, then turns for 1.15s (subroutine “x2_rotR”) before jumping back before the block “APDS9960_prox”. Insert the subroutine for forward travel at the “N branch”. Also connect the output as a loop with the input of the subroutine “APDS9960_prox”.

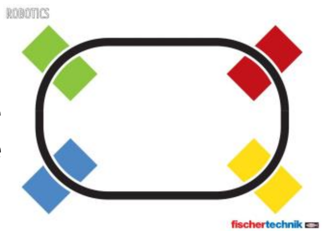


Now, the program is complete and you can save it under the name

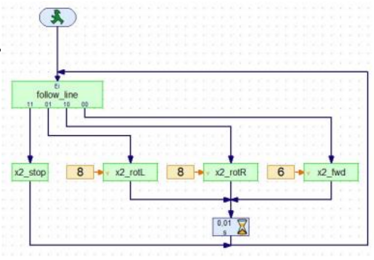
“Advanced_2-wheeled_robot_1.rpp”

on your computer. Then test the program with your model.

Task 2

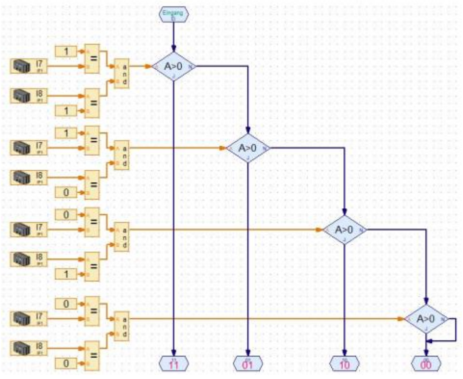


As you can see in the task description, you will use the black line on the obstacle course included in the building kit as the path of travel for the second task.



The right image shows you what the main program should look like later on. You will probably recognise a couple of the subroutines you have already used in previous tasks.

Here as well, you will need to create a couple of new subroutines, such as “follow_line”.



This is a subroutine that you can program once and use again in all of the programs that require the robot to track along a line.

The IR sensor delivers four different input values. The output “11” is approached depending on the value, e.g. IR7 and IR8 = 1.

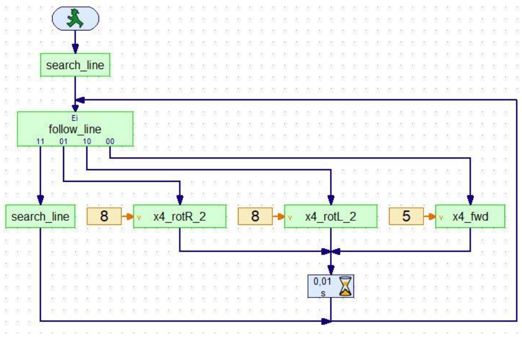
The subroutine “x2_stop” follows in the main program, and the motors will stop. Therefore, travel along the line is monitored continuously via an endless loop, and corrected as needed.

Test out the program and save it under the name

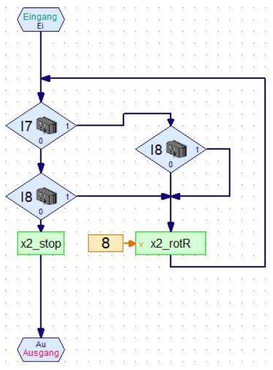
“Advanced_2-wheeled_robot_2.rpp”

on your computer.

Task 3



Now, we'll continue with the 3rd task. This is similar to the previous task – but here, the robot needs to first search for a black line. Here as well, is the main program with its subroutines first. The subroutine "search_line" is new.

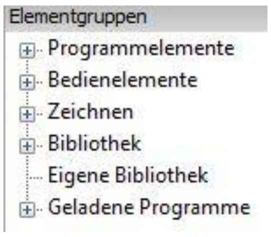


The robot uses this subroutine to search for the black line. If both IR sensors have a value of "0", the two motors will stop and the program leaves the subroutine. Otherwise, the model will turn via the subroutine "x2_rotR". Both motors turn in opposite directions to generate a rotational movement.

You are already familiar with the subroutine, so you only need to insert it.

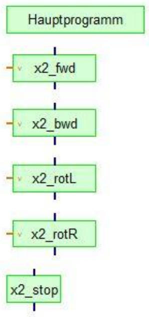
The same applies to the subroutines in the main program.

Important note: If you don't remember how to insert existing subroutines, it's simple – just load the program in which the subroutines are used.



Elementgruppen
- Programmelemente
- Bedienelemente
- Zeichnen
- Bibliothek
- Eigene Bibliothek
- Geladene Programme

Select "Loaded programs" in the element group.



Now, click the folder in which the subroutines are located. The program elements are displayed.

Simply use your mouse to drag the subroutine you need to the programming screen, and insert it at the desired point.

Test out the program and save it under the name

"Advanced_2-wheeled-robot_3.rpp"

on your computer.

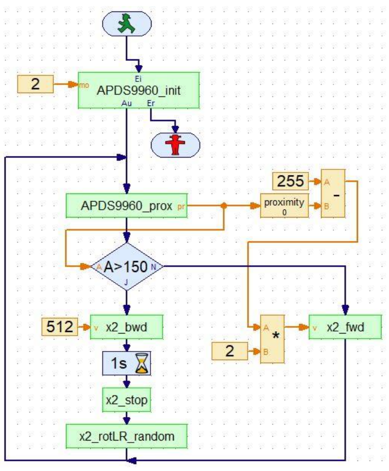
The complete example program is available at C:\Programm(x86)\ROBOPro\Beispielprogramme\ROBOTICS Smarttech\Advanced_2-wheeled_robot_3.rpp

Task 4

Now, let’s go on with the 4th task. This is another task involving a way to avoid an obstacle.

First, load the program “Advanced_2-wheeled-robot_4.rpp from the folder “C:\Programme(x86)\ROBOPro\Beispielprogamme\ROBOTICS Smarttech\

Here as well, you can look at the main program first. In comparison to the program “Advanced_2-wheeled-robot_1.rp”, you will only fid the subroutine “x2_rotL_R_random” and a couple of changed or added parameters. If you want to create the program yourself, you can use the program shown at the right.



Otherwise, learn about the way the program works using the travel paths of the 2-wheeled robot.

Simply set up an obstacle course using a few books and start the program. If everything works, the robot will easily avoid all of the books.

Save the program under the name

“Advanced_2-wheeled_robot_4.rpp”

on your computer.

Then disassemble the model and turn to the next task.