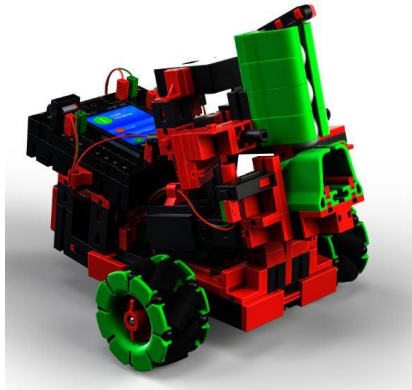



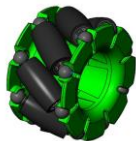


## Ball robot



The ball robot can take shots with little balls that are in a magazine. It's really fun! In addition, this model will teach you some more programming fundamentals. You are already familiar with the fischertechnik components you will use from previous tasks. You will control the model with the gesture sensor.

Actuators and technical accessories installed in the model:

Mini Motor	Gears	Gesture sensor	Omniwheel
			

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

The “Ball robot” model is divided into 4 programming tasks:

<b>Task 1</b> Ball_robot_1.rpp	<u>Programming level 3</u> The shot command should be triggered using gesture control
<b>Task 2</b> Ball_robot_2.rpp	<u>Programming level 3</u> The ball robot is controlled to move in different directions using gesture control. The shot command is executed as needed.
<b>Task 3</b> Ball_robot_3.rpp	<u>Programming level 3</u> The ball robot is controlled to move in different directions using gesture control.
<b>Task 4</b> Ball_robot_4.rpp	<u>Programming level 3</u> The ball robot is controlled via a remote controller in ROBO Pro online mode. The ROBO Pro control panel is used to do so.

## 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”**.

Before you start the task, here is some technical information on the shot function. The balls are held in a magazine, then moved individually into the shooting device from there.



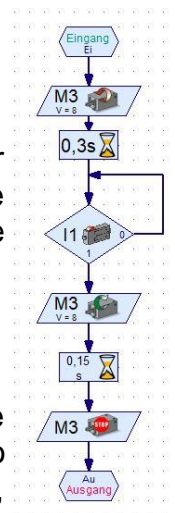
The green I-strut is bent backwards using the crank, the gear of the shooting motor. The motor turns to the left until the button, then releases the I-strut which is bent and tensioned. It catapults the little ball forward and out of the shooting device with its strain energy.

The subroutine shown here illustrates the shooting sequence.

The shooting motor turns to the left for 0.3s. Then the shooting bracket is after the button, and it turns until the button is pushed. Then the motor turns to the right for 0.15s – turning back and releasing the button once again. Then the motor stops.

Create the subroutine with the name **“fire”**.

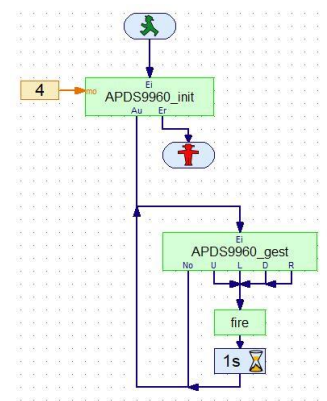
Since you are working with the gesture sensor, you will have to initialise it in the main program first. You need the **“gesture”** working mode, which corresponds to the constant 4 on the input **“mo”** (mode). Since all movements can trigger a shot, you can connect the outputs with the input of the subroutine **“fire”**.



Test out the program and save it under the name

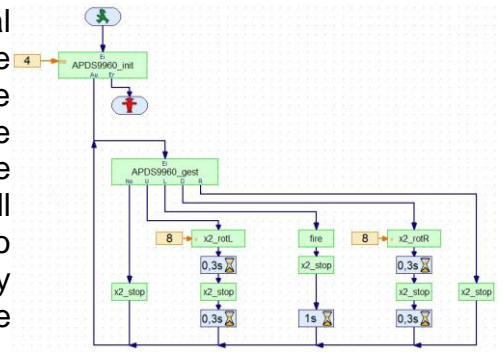
**“Ball\_robot\_1.rpp”**

on your computer.



## Task 2

In the next program, you will integrate two rotational movements into the main program. Depending on the direction, for instance a hand motion over the gesture sensor, the robot will turn 0.3s to the right or left; if the hand comes from the top, the shot command will be triggered, and if it comes from the bottom, the motors will stop. You do not have to rewrite the program for this purpose, but only delete subroutines or insert the subroutine to trigger the shot.



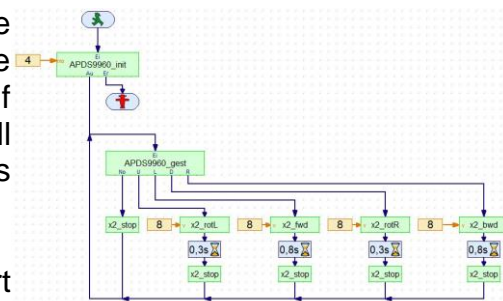
Test out the program and save it under the name

**“Ball\_robot\_2.rpp”**

on your computer.

## Task 3

In the next program, you will integrate four rotational movements into the main program. Depending on the direction, for instance a hand motion over the gesture sensor, the robot will turn for a set time to the right or left; if the hand comes from the top or bottom, the robot will move forward or back; if no signal is detected, the motors will stop.



You will write this program in an endless loop that will start or end after the initialisation subroutine.

Well, everything we’ve done so far wasn’t so difficult, was it?

Test out the program and save it under the name

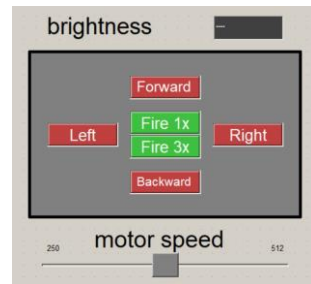
**“Ball\_robot\_3.rpp”**

on your computer.

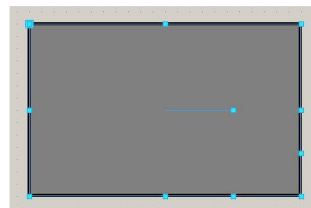
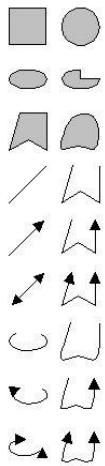
## Task 4

Now you can turn to the 4th task. The ball robot is controlled via the so-called “control panel” on the ROBO Pro. The gesture sensor is used to measure the ambient brightness.

This image shows the control panel you will be creating. Since you are creating it for the first time, here are a few tips.

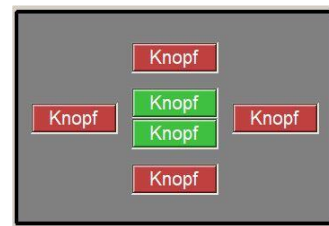
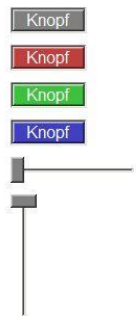


You are in the main program in ROBO Pro. Switch to “Control panel”. First, generate the gray frame for the buttons. To do so, select “Element groups” – Symbols – Shapes”. The available shapes will be displayed to you in a selection window. Click the rectangle. The pencil symbol will be attached to your mouse cursor. Press the left mouse button and draw a rectangle of the desired size.

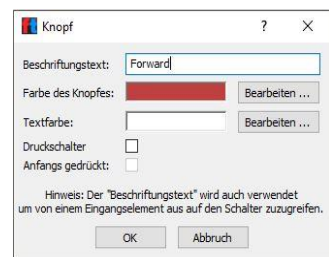


Insert 6 buttons from “Element groups – Operating elements – Control element”.

The individual control elements are once again displayed in a selection.



Insert these into the gray frame. A dialogue field appears when you right click the button. Here, enter the desired designation for the button (forward, backward, left, right, ...)



Set a “text display” over the gray window from the “Element group – Operating elements – Displays”.

Position the display and access the dialogue field by clicking on it. Change the Id/name to “brightness”.



ABC

Insert another control element under the rectangle, a slide control. This is used to adjust the motor speed.

ABC

ABC

The slide control and brightness display are then completed with text entries from the area “Element group” – “Draw” – “Text”.

ABC

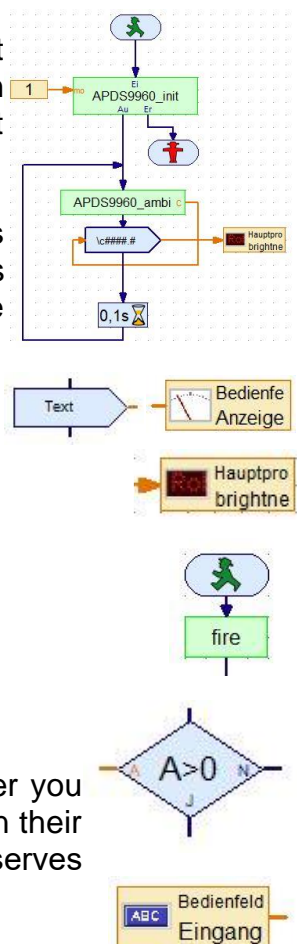
If you click a text size, the pencil will appear once again. Click on the point at which the text should be inserted, and enter the relevant text using the keyboard.

ABC

You will see the result of your work on the first control panel image. Now you can start to create the main program.

After the gesture sensor is installed on the model, it should be used as a light meter. The light measurement will run independently as a separate program in addition to the program for the remote control. The small program is at the right side.

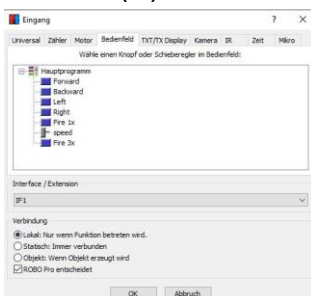
The determined light value from the subroutine “APDS9960\_ambi” is transmitted via output “C” in the text element (“Program elements – Commands – Text”). The value is transmitted to the control panel display to be shown in the control panel at the output of the text element.



When the program is started, the ball robot shoots one ball. To do so, the first ball is loaded into an empty magazine. Then the program waits for a button in the control panel to be pressed. First, insert the subroutine “fire” after the start command.

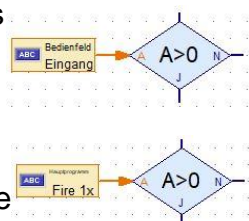
The evaluation program for the operating buttons is actually very simple. After you have set the buttons in the control panel, they must be evaluated depending on their activation. The “branch instruction” from the “Branch, wait, ...” element group serves as the main command.

The command “Control panel input” is connected to the respective brown inputs (A) from the “Element group” – “Program elements” – “Inputs, outputs”.



You can complete this step for the 6 buttons.

Initially, all of the control panel inputs have the same name. You have to assign them their IDs/names in the next step. To do so, right click on the command. A dialogue field will appear where you can select the appropriate ID/name. Connect the individual subroutines to the outputs of the branch.

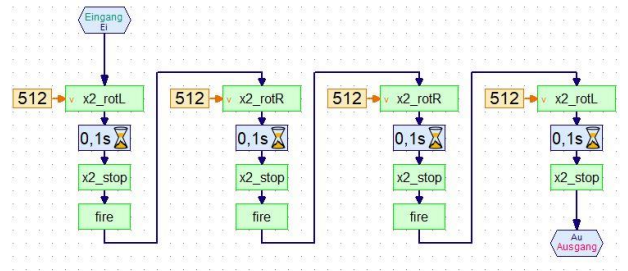




The following are available from the previous program: “fire”, “x2\_fwd”, “x2\_bwd”, “x2\_rotL”, “x2\_rotR” and “x2\_stop”.

The outputs are connected to one another, and inserted before the command “Wait time 0.01s” into the connection to the subroutine “x2\_stop”. The endless loop after the “x2\_stop” command will end before the first branch instruction.

Actually, you are only missing the subroutine “fire3”. This is available as a programming aid on the right side. Of course, there are a couple of directional changes after every shot. The robot should first turn 0.1s to the left, then shoot the ball. Then it should turn 0.1s to the right and shoot. Then this rotation and the shot are repeated. Finally, the ball robot will turn back to the left for 0.1s.

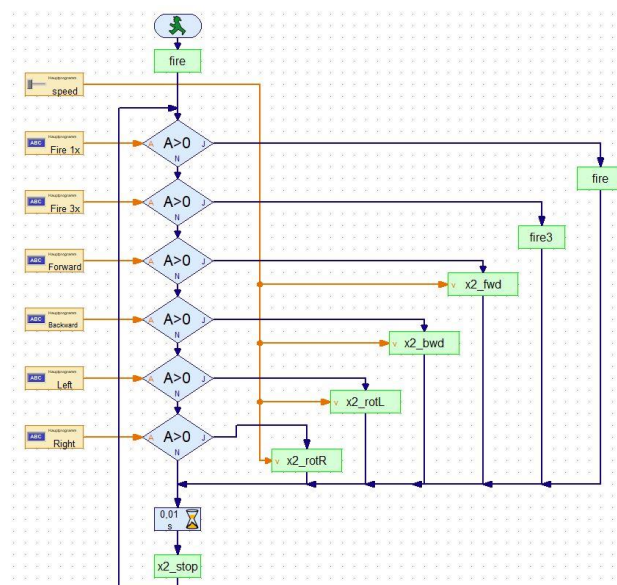
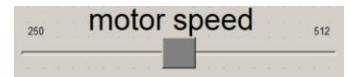


Create the subroutine and insert it into your evaluation program.

The orange connections for the subroutine for the travel are still open. You must insert another control panel input from the element group “Program elements” – “Inputs, outputs”. Select “speed” from the dialogue field (right mouse button).



Then, connect the output of the control panel with the orange inputs of the subroutines for controlling the travel. Use the slide control in the control panel to adjust the speed of the motor. To do so, drag the slider to the left or right.



OK, that's it. The program is complete, and you can test it out.

Then save it under the name

**“Ball\_robot\_4.rpp”**

on your computer.

Note: All 4 of the programs presented here are available as finished example programs in your ROBOPro directory  
C:\Programme(x86)\ROBOPro\Beispielprogramme\ROBOTICS Smarttech\

Disassemble the model once again and start with the next model.