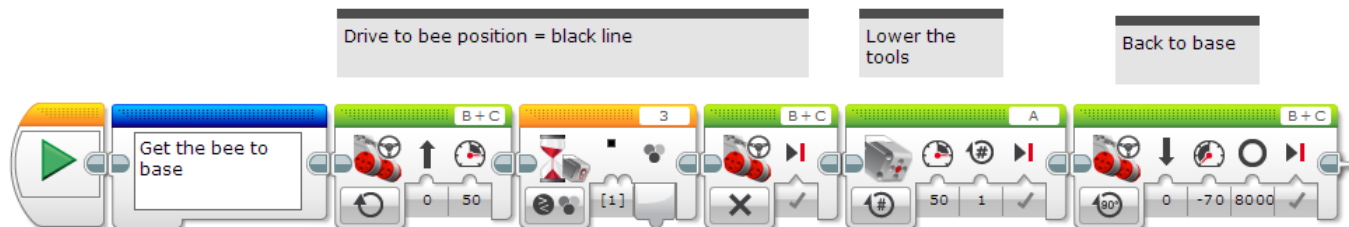


Robot-Design Software



Drive to bee position = black line

Lower the tools

Back to base

20.08.2017

The number of degrees is too high by purpose. It ensures that the robot returns to base even something went wrong before.

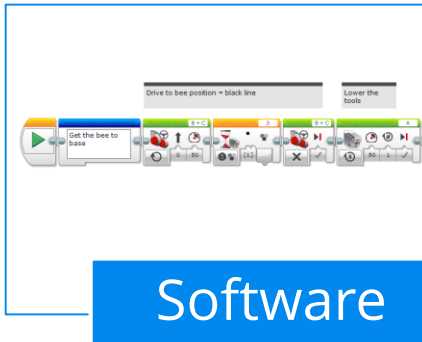


This Presentation is one of four about FLL Robot Design



Available
soon

Navigation

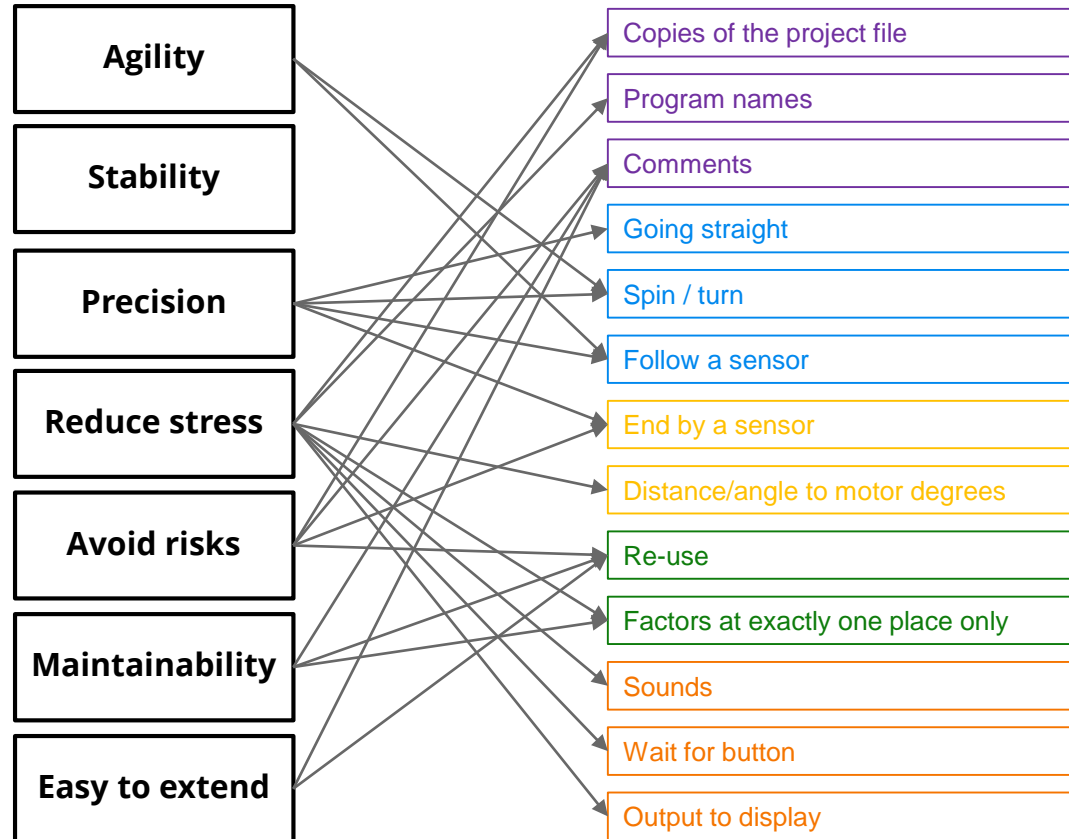


Available
soon

Strategy

<http://nano-giants.net/robot-design>

Requirements from FLL robot game and implementation with hardware





A lot can be achieved even with the more easy programming concepts

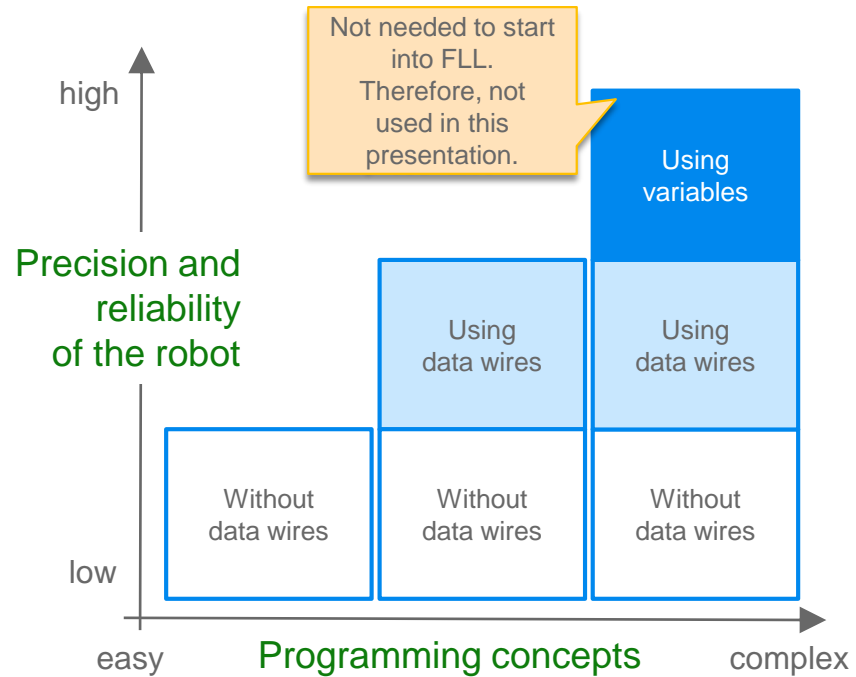
This presentation *does not* explain how to program the **LEGO MINDSTORM EV3**. Instead it shows how **features of the software** can be **used for FLL robot game** in a meaningful way.

We **consciously only show program fragments** which only become useful if embedded into self-created programs.

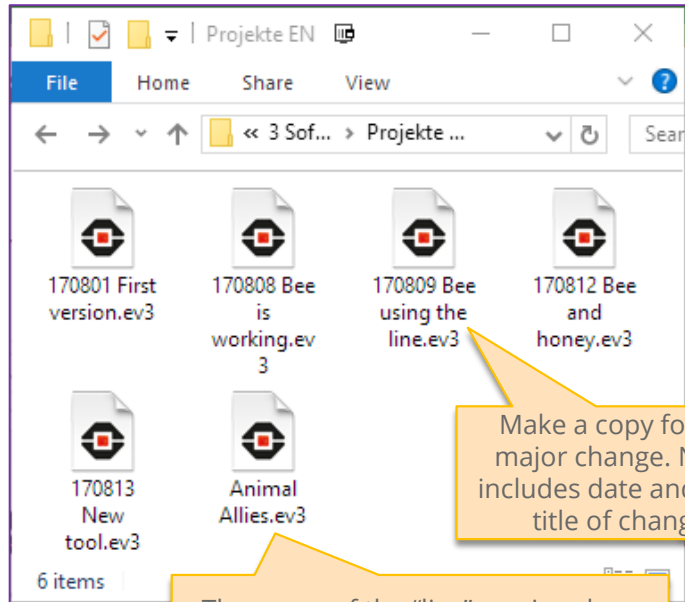
This presentation is meant to provide impulses, not perfect solutions.

If you want to dive deeper into programming you can of course do some things even better. Some related ideas can be found at the end of the presentation.

The ideas in this presentation are shown with and without the use of data wires, whenever possible. Variables are not used.



Frequently make copies of the project file.



Use meaningful names for all program in the project.

Programs

Type	Name
	Moves.ev3p
	Direction follower.ev3p
	Ending moves.ev3p
	1 Get the bee.ev3p
	2 Honey delivery.ev3p
	3 Milk on the ramp.ev3p
	Line follower.ev3p
	Debugging ev3p
	D-factor A-fac...

Names that speak for themselves, instead e.g. „Program2“.

Sequential numbering for all programs used in Robot Game.



All team members (and even robot design judges) understand the program. Changes and extensions can be implemented with a lot stress less.

LEGO MINDSTORMS Education EV3 Teacher Edition

File Edit Tools Help

RD Software EN.ev3 x +

1 Get the bee x +

Drive to bee position = black line

Lower the tools

Back to base

Get the bee to base

B+C

3

B+C

A

B+C

0 50

[1]

50 1

0 -70 8000

Meaningful comments

The number of degrees is too high by purpose. It ensures that the robot returns to base even if something went wrong before.

B+C with power 50 for 5120 degrees

Use motor A

Useless comments



There are new types of moves, if you go beyond just green block

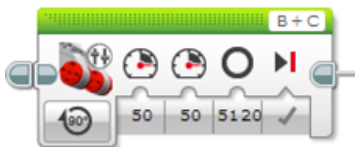
Type of moves

Going straight

with „Move Steering“



with „Move Tank“



without data wires

Spin

- Same power
- Opposite directions



Turn

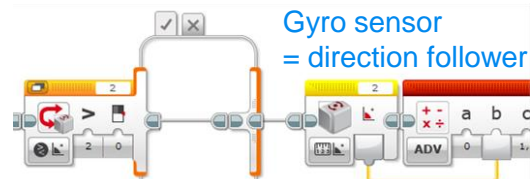
- One motor off
- force is cut in half and turn radius bigger



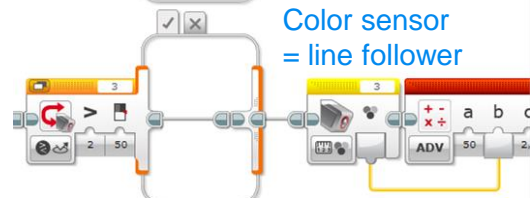
without data wires

„Follow a sensor“

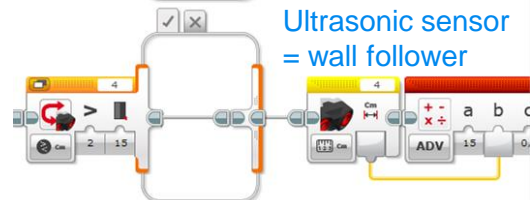
= Control motors in reaction to sensor values



Gyro sensor
= direction follower



Color sensor
= line follower



Ultrasonic sensor
= wall follower

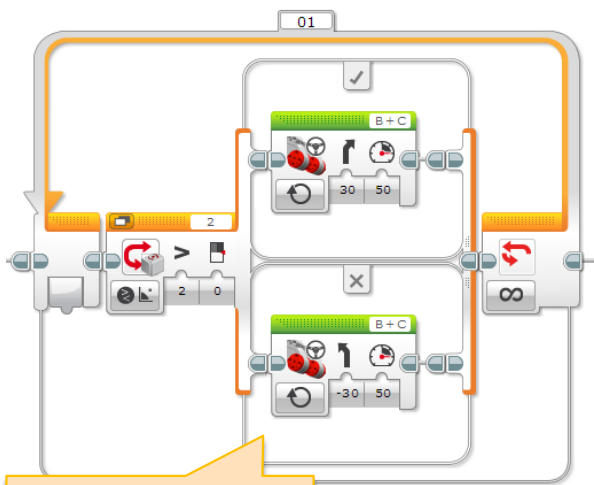
without data wires

using data wires

„Follow a sensor“ is not complicated, yet very useful

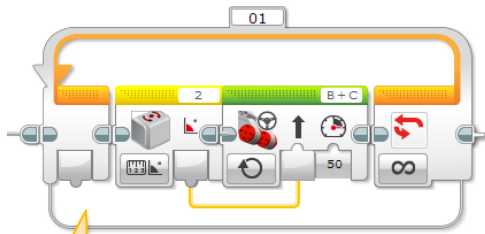
Type of moves

„Zick Zack“ =
static steering



without data wires

„Proportional“ =
adaptive steering



using data wires

Approach for the other sensors:

Line follower

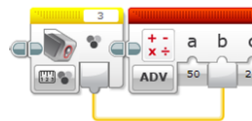
- Reflected light
- Difference to edge of line (50%)

Wall follower

- Distance to wall in cm/in
- Difference to targeted distance

Finetuning is required!!!

In any case, power and steering need to be matching.

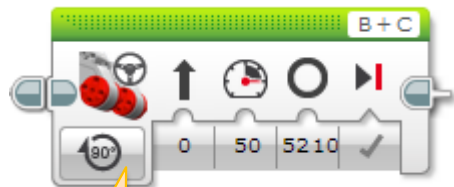


„Wait for sensor value“ allows more than “motor degrees”

Ending moves

„Move steering“ and „Move tank“ use the built-in rotation sensors

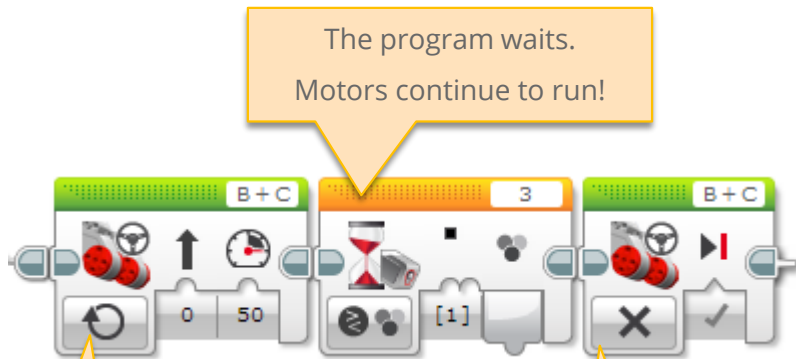
go straight, spin, turn, ...



„On for Degrees“ and „On for Rotations“ use sensors build into the motors.

without data wires

Using the other sensors require three blocks: On + Wait + Off



The program waits.
Motors continue to run!

„On“ starts the motors and keep them on.
The program however continues!

„Off“ stops the motors.

without data wires



Pick the sensor matching your needs

Ending moves

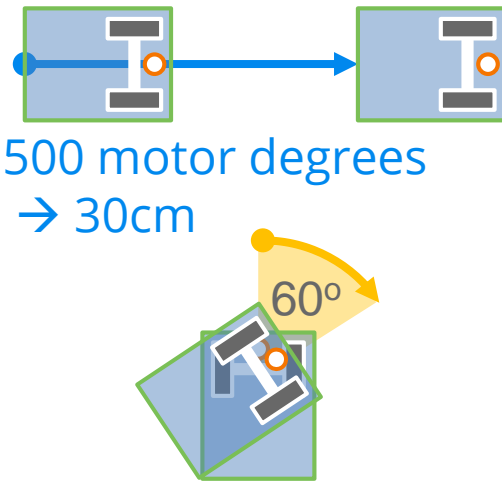
Requirement	Sensor	Remarks
Stop after distance	Rotation (build into motors)	Automatic speed reduction before reaching the target value. Built-in correction of overshooting if needed.
	Ultrasonic	Comparably slow. Usually not needed, because the other sensor are reliable enough.
Stop at line or colored area	Color	Very reliable. Usually fast enough.
	Reflected Light	Sensor values highly depend on the robot's speed.
Spin/turn until angle	Rotation(build into motors)	Automatic speed reduction before reaching the target value. Built-in Correction of overshooting if needed.
	Gyro	Hardware depended: Tends to overshoot even at low speed. By doping some experiments, this can be understood and taken into consideration.
Align with border wall or model	Timer	<p>If the remaining distance the robot needs to move is unclear, using "Stop after distance" cant be used, because it might stop soon or the moving cannot be completed if the robot gets stuck before.</p> <p>„On for seconds“ with low power is one safe solution.</p> <p><i>In any other situation, using time to end a movement is highly unreliable. Use other sensors!</i></p>



Distance and angle are more comfortable then degrees

Ending moves

Factors need to be determined based on the hardware



500 motor degrees
→ 30cm

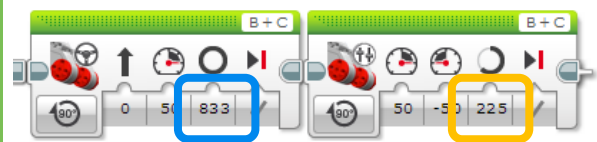
150 motor degrees
→ 60° angle

Numbers are only illustrative!

Calculation elsewhere e.g. using Excel

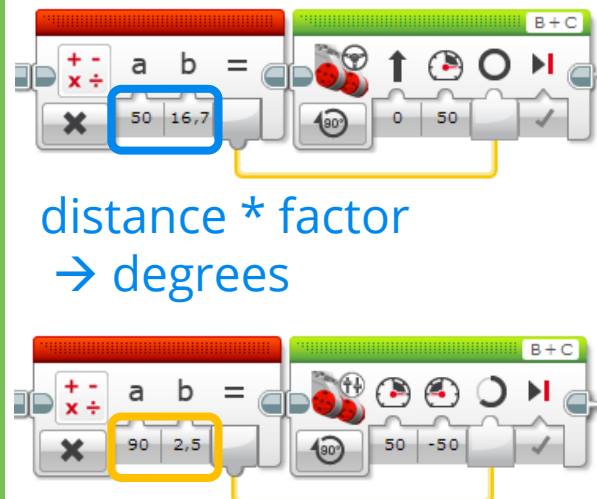
Distance	Degrees
30	500
1	16,7
5	83
10	167
20	333
30	500
40	667
50	833

Angle	Degrees
60	150
1	2,5
10	25
20	50
30	75
45	113
60	150
90	225



without data wires

Calculation within the program

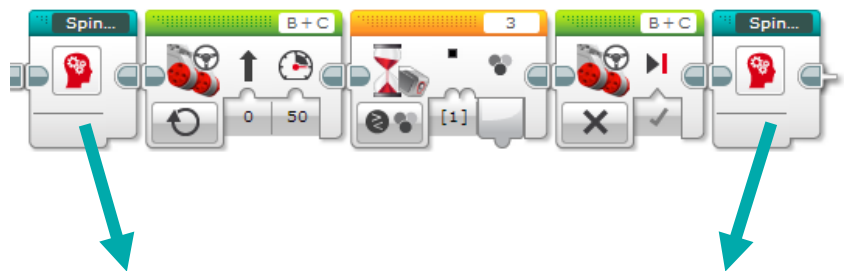


distance * factor
→ degrees

angle * factor
→ degrees

using data wires

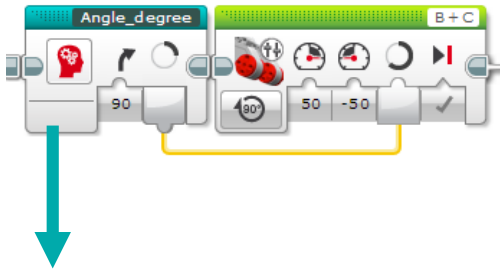
Define once, use many times
e.g. „90 degree spin“



My Block „Spin_90“

without data wires

Factors only at exactly one single place
e.g. „Angle to motor degrees“



My Block „Angle_degree“

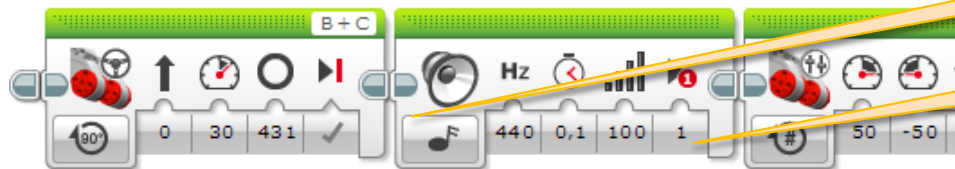
using data wires



Debugging = Understanding what's going wrong, without Bluetooth

Debugging

Mark important program steps with sounds.



0,1s is sufficient

Do not use "Wait for completion".

without data wires

Wait on brick button, to understand where the robot is exactly.



Remove before competition 😊

without data wires

Additionally send sensor values to the display.



"Wired"

Wait for button.

using data wires



Ideas to improve further

For advanced programmers

„Gyro Drift“

Understand what that is and how it can be detected. Add an alarm or counter measures to the program.

Own menu instead stand-alone programs

To avoid searching for the next program to be run, put everything into one program and use the brick button to select and start.

Calibrate the color sensor

Understand what the impact is.

Align square to lines

Using two color sensors the robot can be aligned perpendicular to a line.

PID line follower

Get to the line edge faster and be less disturbed by bends and curves.

For pros

Calibrate multiple color sensor yourself

EV3 only knows one calibration. For more than one sensor values have to be stored individually.

Work with text files on the EV3 brick

Own calibration, sensor log, ...

Use blue “Unregulated Motor” instead of green move blocks

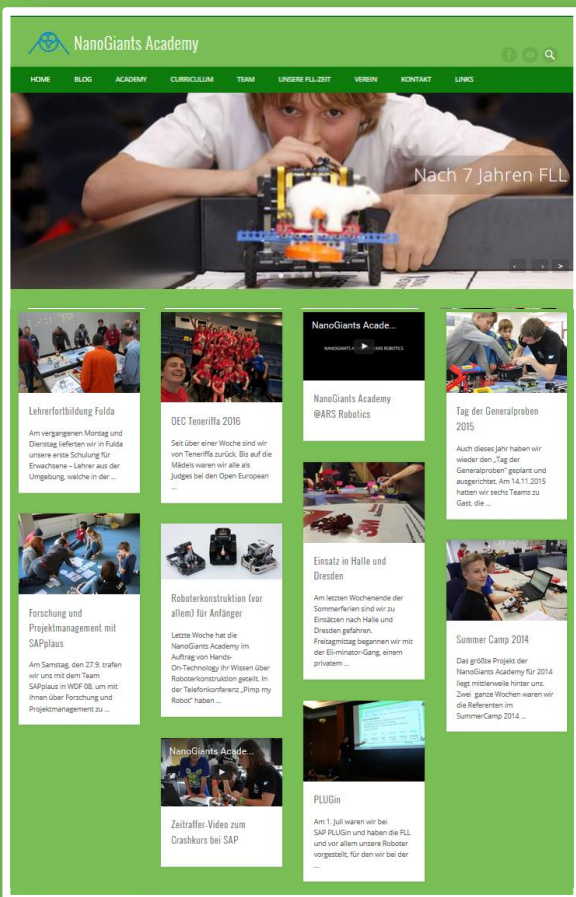
Full control of acceleration, speed, and stopping.

Move Block

A My Block with several parameters to choose from different types of movements and different ways to end those.



Contact and further information



Mail:

academy@nano-giants.net

Web:

<http://nano-giants.net>

Facebook:

<http://www.facebook.com/NanoGiantsAcademy>

YouTube:

<http://www.youtube.com/sapnanogiants>

