ROBOESL PROJECT

ROBOTICS-BASED LEARNING INTERVENTIONS FOR PREVENTING SCHOOL FAILURE AND EARLY SCHOOL LEAVING

Erasmus+ 2015-1-IT02-KA201-015141

Output 1: Curricula for 10 exemplary interdisciplinary robotics projects

Curriculum 1: the Roborail

Lead Partner: UNIPD (IT)

Authors: Michele Moro, Francesca Agatolio, Emanuele Menegatti (UNIPD)

Contributions

Dimitris Alimisis (EDUMOTIVA[1]) (Chapters 2 and 3)

Linda Daniela (University of Latvia) (Chapter 5)

Circulation: Public

Version: Final

Stage:

Date: March 31, 2017

[1] EDUMOTIVA stands for 'European Lab for Educational Technology'

Declaration

This report has been prepared in the context of the ROBOESL project. Where other published and unpublished source materials have been used, these have been acknowledged.

Copyright

Funding Disclaimer

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Table of Contents

Abstract

This document contains the description of the Curriculum n. 1, entitled ‘The Roborail’ which is part of the Intellectual Output 1 (Curricula for 10 exemplary interdisciplinary robotics projects) developed and tested in teacher training courses within the context of the ERASMUS+ ROBOESL project. In this Curriculum we introduce some preliminary building instructions and the first, fundamental software blocks to program code able to make the robot move in a controlled way on a straight line. No specific sensor is used in this curriculum. The proposed variants permit the teacher to provide more challenging problems in a progressive way which are also the excuse to introduce some more basic blocks.

Chapter 1: **Short description and scenario (O1.1)**

A single straight path with S stations (and S-1 subtracks). The path must be 'followed', without using sensors, at a constant speed on each subtrack, stopping for a while at each station. The entire process can be repeated in backward direction.

1.1 The scenario

Future fast railways will be based on magnetic levitation, which requires long straight tracks and stations at possible fixed distance. This is also the case of a monorail train in a big entertainment park. So the RoboRail experience can be easily contextualized and motivated, and it is immediately understandable.

In spite of its simplicity, this first experience is designed to be useful to introduce some important basic commands of EV3-G and some relevant concepts like time and speed control, proportionality and others.

1.2 Connections with subjects

Teachers of different disciplines can introduce the Roborail activity with different approaches:

1. Geographical: the lesson can start working on a map of the city the students are best acquainted of (their city), of a city that the class visited in the past, or of a city they are planning to visit, etc. Working the Roborail issues on a real map, students can locate their exercise in the reality of an actual design.

2. Historical: the first rail in the human history; the speed of the first train; the possibility to build a bar-graph with different "speeds" of different trains (a comparison between modern trains and old train; a comparison between the train speed in different European countries).

3. Robot-Human interaction: Are people going to be at ease moving in a driverless train? There are few cases, and some issues to be discusses, like:

   - The case of the Turin Metro (Italy)

   - Different levels of autonomous trains (from G1 to G4)

   - Could the design change the acceptability of an autonomous vehicle?

4. News: The teacher could suggest some research on future railways and transportations (see for example http://hyperlooptransp.com).

Chapter 2: **Pedagogical objectives (O1.2)**

2.1 General objectives

- To provide students with a stepwise approach for a step by step acquisition of technical skills in using robotic technologies (hardware and software).
- To offer the robotics benefits for all children, especially those at risk of school failure or early school leaving.
- To promote STEM learning through interaction with the robotics technologies.
- To support self-directed action allowing learners to learn independently.
- To support the development of a "real" learning scenario encouraging the engagement of the learner in authentic problem solving.
- To adjust the robotics project to learners' needs and interests by offering tasks with some options to advance to different levels of complexity and difficulty.

2.2 Specific objectives

More specifically, upon successful implementation of the activities described in this curriculum students will achieve the following objectives:

- Learn to build a simple tribot structure with two motors and a ball caster in the rear
- Translate the scenario into a realistic representation on their desk
- Use the Move Steering command with its parameters to put the tribot in motion
- Instruct the tribot to travel a certain distance by trial and error experimentations
- Use the “wait” command to control the motion of the tribot
- Invent the “loop” command to repeat the motion
- Make the tribot to reverse direction by trial and error experimentations
- Invent a mathematics-based solution for making the robot to travel a certain distance

Chapter 3: **Suggestions for learning methodologies (O1.3)**

The proposed learning methodology is driven by the problem to be solved: how to make the robot-train to travel a certain distance, to reverse direction and repeat this motion. To engage students in activities requiring designing and manufacturing of real objects, i.e. robotic structures that make sense for themselves, we devise activities that will encourage students to construct robots but also will encourage them (providing the necessary support by the teacher) to experiment and explore ideas that govern their constructions and to participate actively in the learning process.

The robotics industry so far mainly aims at humans using pre-programmed pre-fabricated robots. The ways in which the robots are made and programmed is a black box for their users. It is a paradigm compatible with the traditional educational paradigm of the teacher or of the curriculum book revealing and explaining ready-made ratified and thus unquestioned information. Very differently from this approach, our methodology suggests the transition from "traditional" black-box technologies to the design of transparent (white-box) digital artifacts where users can construct and deconstruct objects and have a deep structural access to the artifacts themselves. The learners build something on their own that is a tangible object: a tribot representing a train that they can both touch and find meaningful. The white-box metaphor for construction and programming might generate a lot of creative thinking and involvement in learners.

Then learners are invited to work on experiments seeking solutions to real world problems: how to make the train to travel a certain distance, how to make it to reverse direction and to repeat its motion. The robotic activities take the form of a research project posing problems that are authentic, multidimensional and can have more than one solution. It is particularly important that the problems are open and flexible and allow students to work with their own unique style and the way they prefer. The proposed work actively involves students in learning opportunities by giving them control and ownership of their learning, encouraging creative problem solving and combining interdisciplinary concepts from different knowledge areas (science, mathematics, technology, etc.). The learning activities are as open as possible so that learners have opportunities to participate in the final configuration of them and ultimately provide opportunities for reflection and collaboration within the team.

A. The role of the students

Students first discuss the problem through a free dialogue in their group and after that they devise an action plan to solve it. They work in groups following the worksheet and the discrete feedback they receive from the teacher. Students may extend their work to variants suggested by the teacher or devised by students themselves. First, they find solutions making trial and error experimentations. In the end they are supported to find a mathematical solution to the same problem. The final solutions of the groups are presented in the class, are discussed and evaluated with students reflecting with critical mind on their work, expressing their views and recording their experiences in a diary or questionnaire.

B. The role of the teacher

The teacher in such a constructivist theoretical framework, like that described above, does not function as an intellectual "authority" that transfers ready knowledge to students but rather acts as an organizer, coordinator and facilitator of learning for students. S/he organizes the learning environment, raises the questions / problems to be solved through a worksheet, offers hardware and software necessary for students' work, discreetly helps where and when necessary, encourages students to work with creativity, imagination and independence and finally organizes the evaluation of the activity in collaboration with the students.

Chapter 4: **Technical guidelines (O1.4)**

4.1 Building instructions

The robot is built in the so called *tribot* structure which includes two motors with one ‘normal’ wheel of 28 mm of radius each, and a ball caster on the rear to permit a simple form of steering (fig. 1). For this experience no further sensor is necessary apart the built-in rotational sensors, integrated in the motors, which will be implicitly used when a command controls the angle swept by the motor axis.

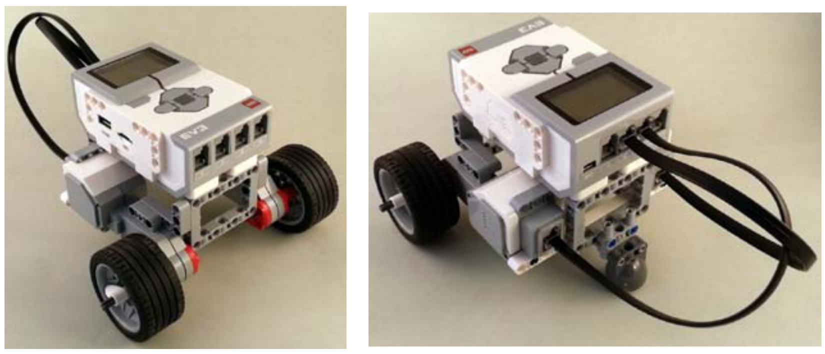

Figure 1. A simple tribot (source: daminekee.com)

The S stations (we suggest to choose S in the range 4-8) are at a constant distance D, for example D=20 cm. Ask the students to draw signs on a sufficiently long sheet of paper, on the table or on the floor, using a pen/pencil, or to put, directly on the table or on the floor, short pieces of tape, taking care that a distance D is left between two subsequent signs or between the middle points of the axis of two subsequent pieces of tape.

4.2 Illustrative solution

After having presented the general task, the teacher can briefly describe the `Move Steering` command with its parameters. Because the motion is on a straight-line and each subtrack is covered at constant speed with a final brake, the command will be set with a null steering factor and with a power value decided at the beginning to induce a constant medium speed (we suggest a value in the range 25..50). The teacher can initially say that we assume that, setting a certain power P for this block, the robot moves at a certain regular speed V. Then the first question is: how long must the motor rotation be to make the robot move exactly for D?

In this first attempt, the use of time for expressing the motion duration seems to be more straightforward for the students and empirically more controllable. The estimation of T could be spontaneously done by the students using a try-and-error approach instead of applying a general formula, i.e. through a sequence of refining steps until the correct value is found. Instead of discouraging this initial attitude, the teacher can eventually observe that this is a long-lasting process to estimate the necessary time and a more general method would be useful when D changes from case to case.

The students, recalling their usual experience with normal usable vehicles, easily accept that, in these conditions, twice the time means twice the space, i.e. they assume a practical proportionality between time and space. In order to obtain the general formula of proportionality from this first assumption, the teacher asks the students to set the time of a `Move Steering` command (fig. 2) to

2 seconds, for example, and with the help of a ruler or a flexible meter, she asks them to evaluate the space B covered by the robot. From this datum, they should calculate the requested time per track T applying a simple proportion (2 : B = T : D ⇒ T = 2 * D / B). For example, with D=20 cm, if in 2 s the robot covers with the assigned power. let's say, 32 cm, the necessary time is T=40/32=1.25 s.

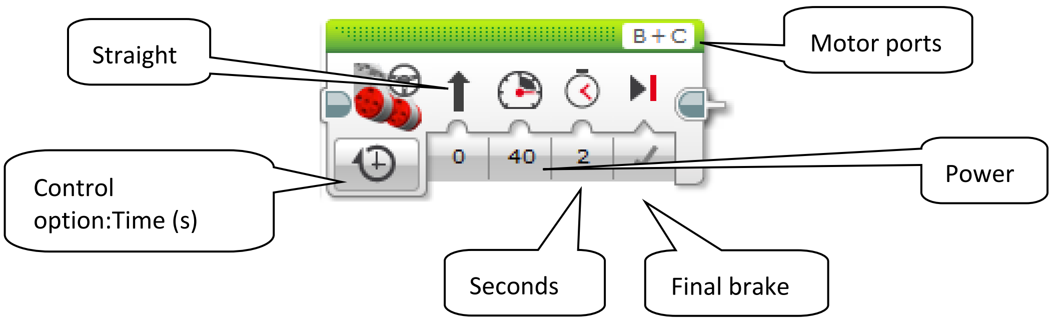

Figure 2. The Move Steering *command*

The next step is to exploit the possibility to precisely (within 1 degree) control the wheel rotation angle to provide a more precise solution. The question is: what is the angle swept by the motor to produce a displacement of D performed by the robot? Does it depend on the mounted wheels? Or on the robot's speed? Again the students could follow a try-and-error procedure to find the correct angle but you can also introduce the geometrical model which is below the robot structure which gives a more general solution.

In formulas:

*rotations = degrees / 360 = radians / (2 * π)*

*degrees = rotations * 360*

Because our assumption leads to the relation of proportionality D=k*T=(B/2)*T, we can recognize that the (assumed constant) ratio k = D/T = B/2 = V represents the linear speed of the robot when moving along each track. The simple geometrical model to be introduced is the basis of the transformation of a rotational motion to a straight-line one (fig. 3). Depending on the level of knowledge of the students, and the didactical objectives of the teacher, the model can refer to rotations, degrees or radians as angle units.

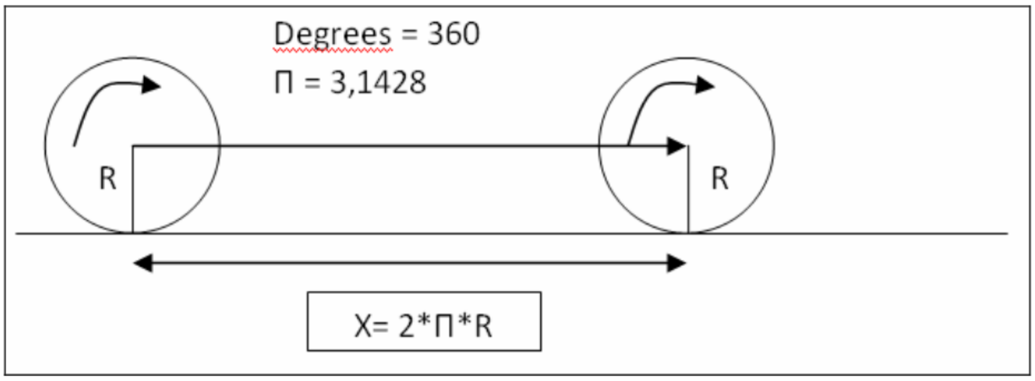

Figure 3. The rotational model (source: terecop.eu)

You can suggest the students to experimentally evaluate the displacement corresponding to an entire rotation of the motor axis, for example manually moving the robot until a complete rotation is performed. Say X this displacement, it is easy to show that D/X is the number of rotations, with fraction, necessary to obtain a displacement D (highlight that, for example, when D=X you obtain 1 rotation, when D=2*X you obtain 2 rotations, when D=1.5*X you obtain 1.5 rotations). If you have the necessary pieces, you can also ask the students to use wheels with different radiuses and observe that you find different values of X, arguing it depends on the wheel radius. The geometrical model actually shows that X=2*π*R. Differently, V implies the amount of time necessary to cover that amount of rotations and does not influence, apart mechanical imprecisions, the covered space D. In order to use degrees instead, because one rotation corresponds to 360 degrees, you have simply to multiply the ratio D/X for 360. This is again a simple proportion.

A new question arises: because EV3-G, beside the time, allows to set an amount of rotations or degrees on each motion block, does it correspond to an option equivalent to set a time or does it give any improvement? To be convinced that the second answer is the correct one, the teacher must mention that the assumption that a certain power corresponds ALWAYS to a certain speed is not perfectly true, for example it may depend on the battery charge and on the total load, including friction, to which the motors are subjected. Moreover, if for the natural inertia of the robot or for other unpredictable reasons, when the motors are stopped after T seconds the robot has swept an angle different from the requested amount, setting the time gives no possibility of correction whereas setting the angle makes this correction possible, as it is actually done by the robot computer and its basic software (the so called *firmware*). Now the students can be asked to modify their solution setting a suitable angle, in rotations or degrees, instead of time, a modification which does not increase the complexity of the solution. Then they have to check the correspondence of the motion of the robot to the geometrical model introduced. For example, using a wheel with R=2.8 cm, with D=20 cm it results:

rotations = 20/(π*5.6) = 1.14 degrees = (20/(π*5.6))*360 = 409

A motion is followed by a `Wait` command (fig. 4), for example for 5 s.

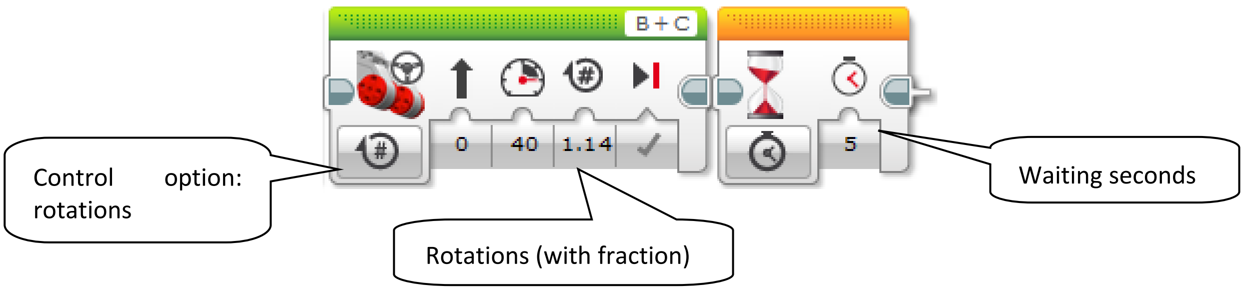

Figure 4. The Move Steering *command in rotations and a* Wait *command*

To repeat the sequence for the following tracks, the first choice of the students could be to replicate, through a copy&paste action, the basic couple of commands introduced so far. At this point the teacher can mention another option, the possibility to insert this couple of commands inside a `Loop` block (fig. 5) Indeed this is a type of block that gives you the possibility to repeat the same subsequence

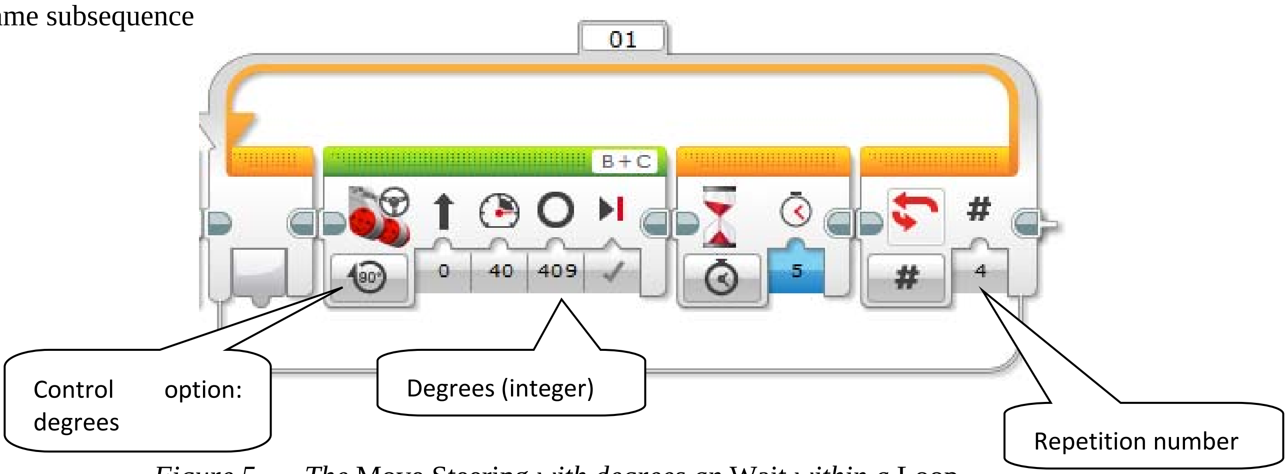

Figure 5. The Move Steering *with degrees an* Wait *within a* Loop

A more complete solution includes the repetition of the sequence of tracks backward, after a possible longer stop at the end of line: this means an added `Wait` command, another loop with the direction of the motion inverted (i.e. a negative power), another `Wait` command at the other end of line; then, all this code can be inserted in a greater loop if you want to repeat indefinitely the whole process (fig. 6).

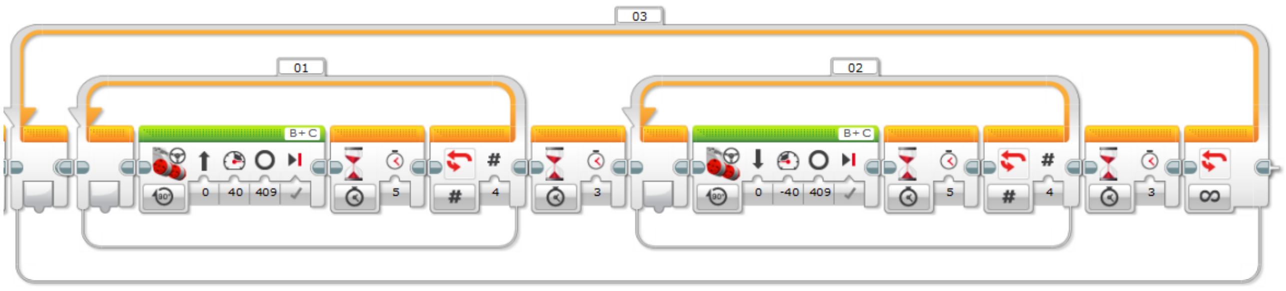

Figure 6. The complete program

4.3 Implementation suggestions

In order to engage the students in the experience, some research on future railways and transportations (see for example http://hyperlooptransp.com) can be suggested by the teacher as a preliminary preparation.

The `Loop` block implements a fundamental control structure, available in almost all the programming languages. This command is the first example introduced of a block whose purpose is not to produce an action or an elaboration but to modify the usual sequential execution flow. The awareness of the importance for a programmer to be able to control the program flow is an important achievement and will be furtherly reinforced in the following experiences.

4.4 Extensions and variants

4.4.1 Premise

This section aims at giving the teacher some further opportunities to exploit the experience and its context to introduce new commands, to satisfy requests of deepening coming from the students, to even out different speeds of the student groups, to not limit possible excellence. The following variants are not exhaustive and are essentially half-baked examples that the teacher can modify or take as inspiration for designing by her own more specific and situated extensions. In all the cases there is the attempt to give a good contextualization in order to maintain a sufficiently strong motivation. The title of each variant includes an estimation of the degree of difficulty.

4.4.2 A new scenario

The tube in London is one of the most known urban rail systems in the world. Actually it is a complex net of lines made by not always straight subtracks and with several multiline crossing points. Nonetheless each line can be synthetically represented by a straight line with a certain numbers of stops with a non-uniform distance between two subsequent stations. Within this simplified scenario our robotic vehicle can represent an underground train. The following extensions can be also referred to this new scenario.

The teacher can attract the interest of the students showing first the map of the underground of London. She can make the student aware that it is possible to travel from one station and each any other station of the net using crossing points through at least one path (different acceptable paths could have a different number of lines to be caught or a different total number of intermediate stops).

4.4.3 Variant a: Add a sound [easy]

Using the Sound block, add a sound to be reproduced by the robot just before starting for the next track (fig. 7).

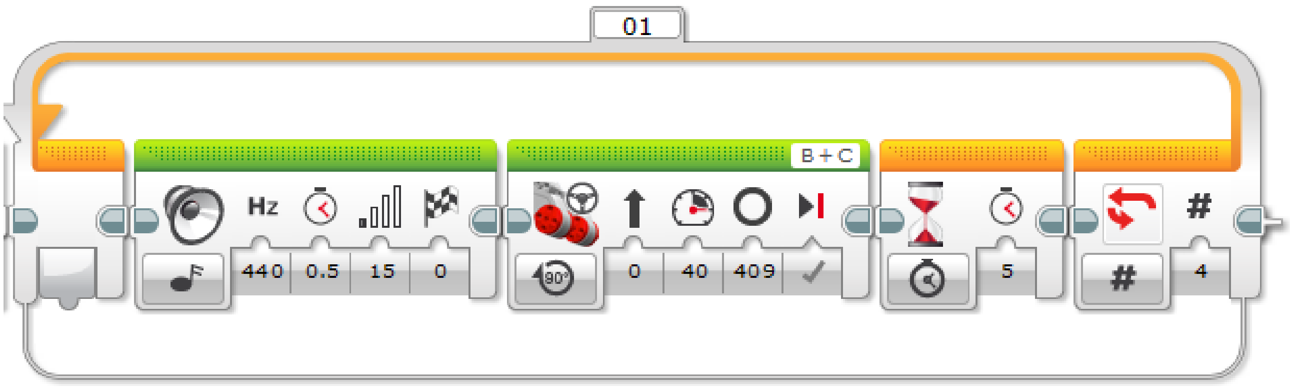

Figure 7. Starting sound

Because the sound block can also reproduce a sound clip as one of its option, apply it to the case of the London tube for reproducing the famous alert ‘Mind the gap’, announced particularly in those stations where the stopping area is on a curve.

Ask the students to reproduce the sound clip “Mind the Gap” before leaving a station. This is a way also to make the students learn how to manage different types of files.

4.4.4 Variant b: Some ‘work in progress’ [easy]

Execute the third track at a lower speed due to ‘work in progress’. This introduces a new, important flow control command, the Switch block, which corresponds to the well known *if-then-else* clause present in any programming language. Here it is used to distinguish the normal track from the special one on the basis of the counting of tracks (fig. 8). For this purpose, the count output of the Loop block, which starts from 0 and may be considered an index of the current track in the range 0..S-2, is used to be compared with the index of the ‘special’ track (3-1=2).

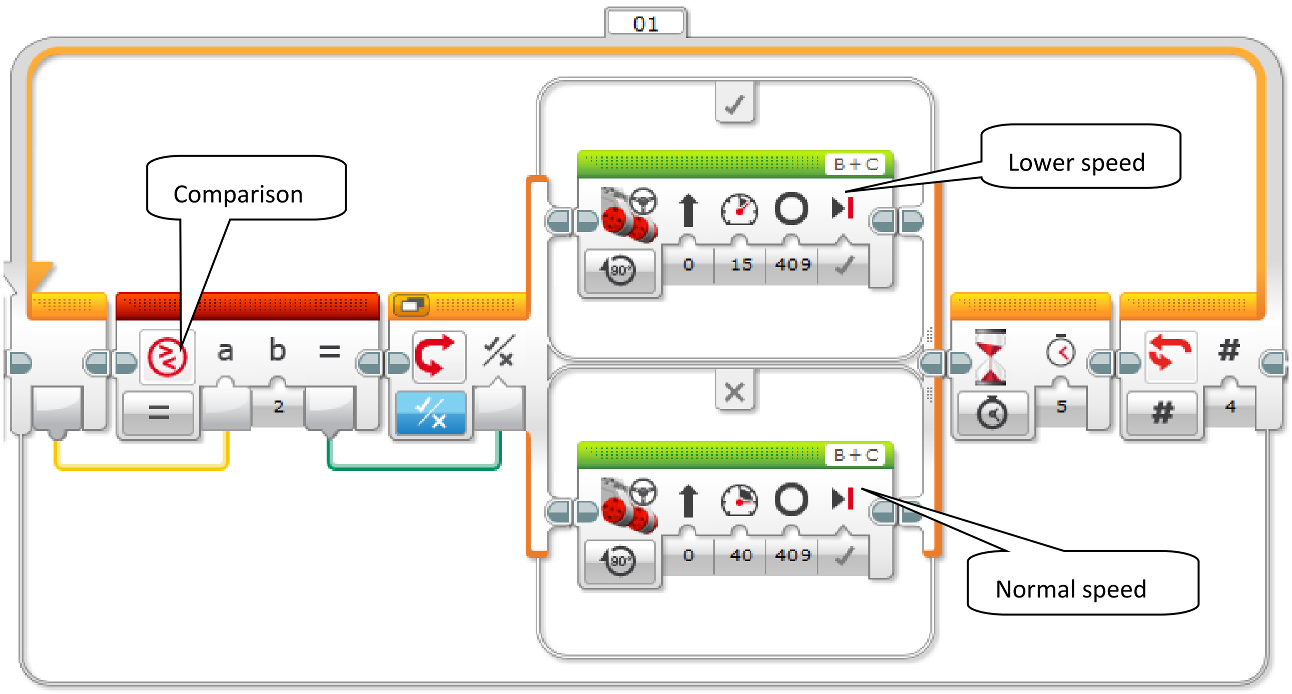

Figure 8. One track at lower speed

4.4.5 Variant c: Variable distance [easy]

In this variant no specific indication of a duration in the motion command is requested: the control of motors is of the on-off type. Only the power is assigned at a specific value. For indicating the position of the next stop we can use the ultrasonic sensor or the light sensor.

In the first case, when you want to make the robot stop within 0.5 s, you have to put your hand in front of the sensor at a reduced distance (e.g. less than 10 cm). After 0.5 s the motors are stopped, the robot waits in that state for the usual 5 s, and then the motors are powered on for the next track (fig. 9).

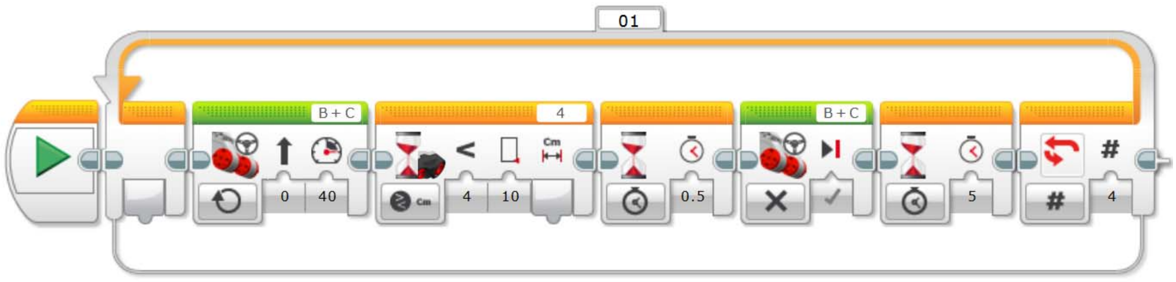

Figure 9. Stopped through the ultrasonic sensor

In the second case, a marker on the path indicates a stop: a marker is, for example, a piece of black tape on a lighter surface. When the light sensor gives back a lower value of lightness, it means that the robot has reached a marker and can stop. After the stopover, and before starting again to sense the light sensor, it is necessary to wait that the robot passes over the current marker, which owns its not negligible width, to identify the successive white-black transition. Apart this detail, the solution is rather simple (fig. 10).

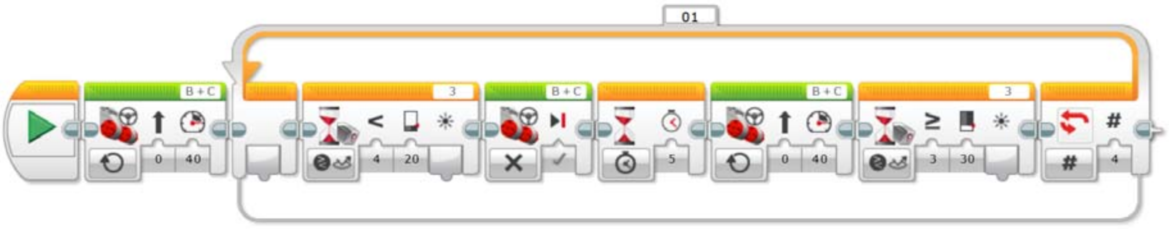

Figure 10. Stopped through the light sensor

4.4.6 Variant d: *Track* as a user block [easy]

Following the procedure to build a user block, you can define a block named *track* with one parameter, the distance in cm. To do this, for example you can start selecting the multiplication block and the following move block in fig. 10, which, together, perform a straight line motion along the distance expressed by the first operand of the multiplication, assumed measured in centimeters (fig. 11).

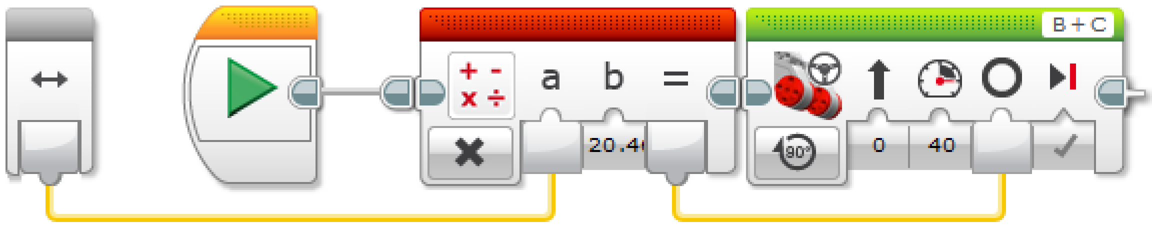

Figure 11. The track *user block*

4.4.7 Variant e: Add a sound and start [medium]

Similar to the previous variant but now the sound must be reproduced while the robot starts moving. This requires to insert the Sound block inside a ‘parallel wired’ structure (fig. 12): this is an option which introduces multithreading (parallel execution of code segments) in your program.

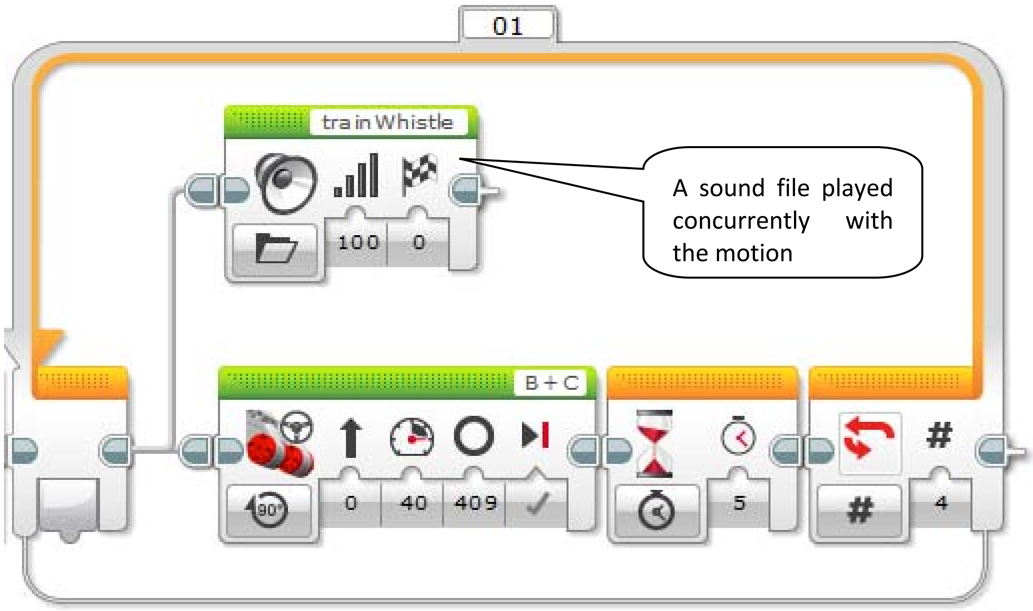

Figure 12. Move and sound

4.4.8 Variant f: Let’s characterize the motors [medium]

In this experience we want to introduce students to the use of graphs, data logging and how to perform a simple measure campaign. Understanding the meaning and quality of data, together with their successive elaboration, is an important achievement towards a real scientific approach.

Ask the students to study the behavior of the motors at different powers. Organize this campaign of measures asking the students to perform at least three-four repetitions of a motion with time duration T (fig. 2, with T= 0.5 or T=1) for every power, for example for values of power equal to 10, 20, 30 .. 100, measuring the corresponding covered space. After having calculated the average space for every power, ask the students to plot these values on a Speed / Power Cartesian plane (highlight that T is always the same in all the repetitions, so that average Speed=average Space/T) and to check if the assumption of proportionality is more or less proved by the acquired data. If the answer is positive, at least in a first approximation, ask the students to deduce the constant ratio speed / power and its unity of measure.

The teacher can insert this experience in the London tube scenario, analyzing real data and graphs coming out from the rail’s timetables.

4.4.9 Variant g: Parametric distance [medium]

We want to adapt our program in order to decide the distance D just before running it. It is now represented by a *variable*, set to the desired value only at the beginning of the program. This means that the calculation of the necessary motor rotations or degrees, made in the previous examples off-line by the students, must be made now by the robot itself. For this purpose, we must use some new blocks: the `Variable` block, to assign a value to an already defined variable and to read its content; the `Math` block, to perform algebraic and other type of mathematical calculations; the `Data wire`, which serves to connect two blocks when one parameter for the second block is provided by the first block; relevant examples of this latter feature are the output of elaborating blocks and of sensor reading blocks.

In our example, the robot, equipped with wheels with R=2.8 cm., must calculate the expression:

degrees = D * 360 / (π*5.6) = D * 20,46

Fig. 13 shows that the variable D is first read, the contained value transferred to the following block which performs a multiplication by the magic number above, and the result is then transferred to the `Move Steering` block as the duration parameter in degrees.

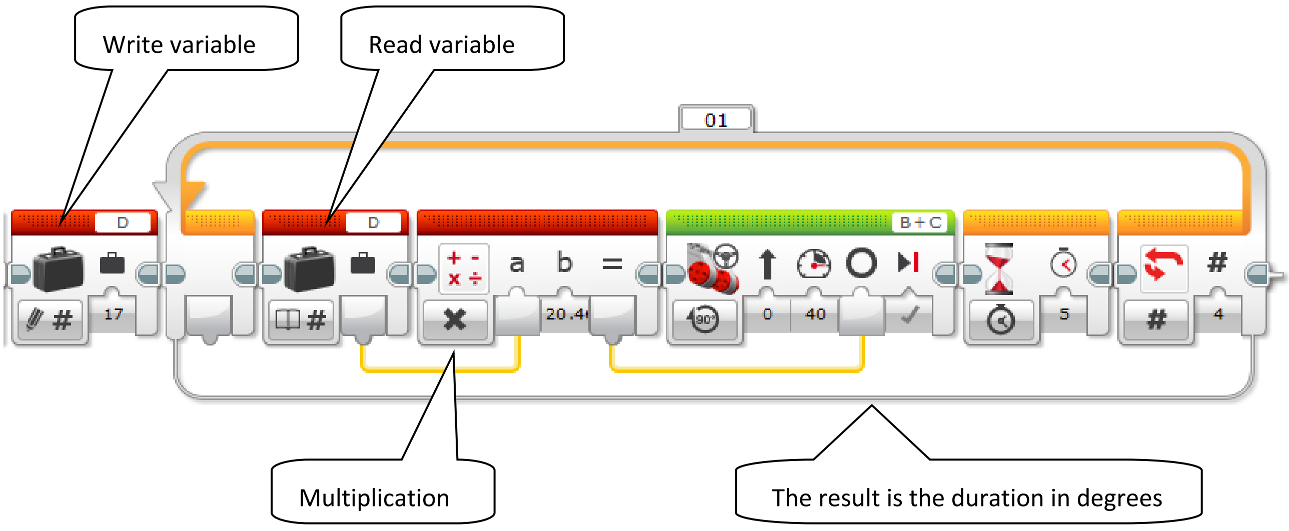

Figure 13. Parametric distance: the use of a variable

Using the block realized in fig. 11, the program in fig. 13 becomes the one in fig. 14.

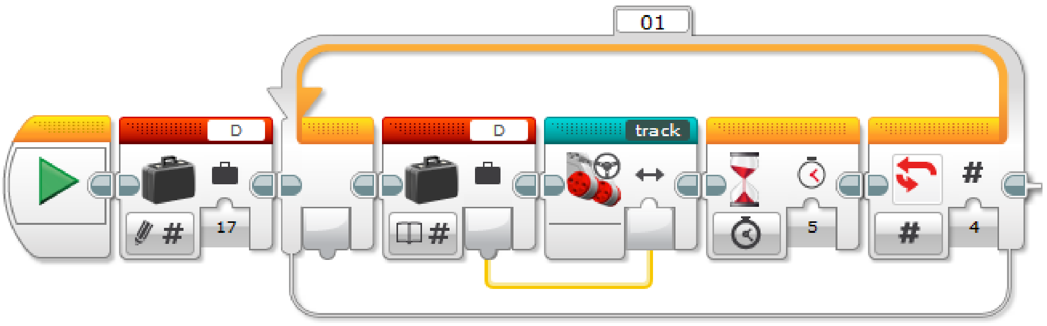

Figure 14. The track *used with a variable*

4.4.10 Variant h: Random speed [medium]

Imagine now to cover each track with a random speed, represented by a random value of power in the range 30..60. Nonetheless we want that the vehicle leaves stations at regular instants, for example at any 5 s in our emulation. This gives us the excuse to introduce other interesting blocks: the `Timer` block to establish a precise temporal scale, the `Random` block to generate random numbers, and a variation of the `Wait` block to synchronize the departures.

Timer 1 is first reset to zero and then used as a clock to wait for the correct instant before leaving a station. In this example these instants are at seconds 3, 8, 13 etc., at a distance of 5 seconds in between. So, independently of the current speed, and therefore also of the time of arrival, assumed in any case before the next departure, the motion block is executed at the correct time (fig. 15).

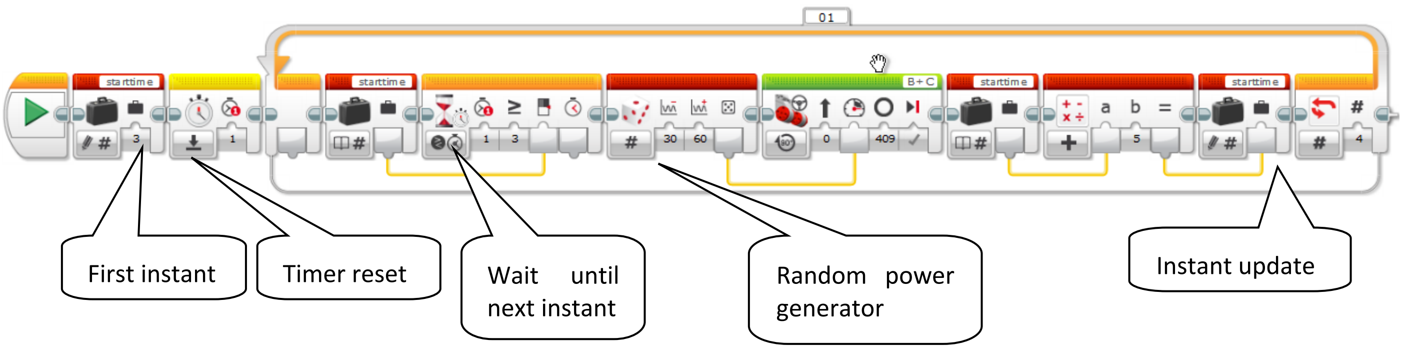

Figure 15. Random speed

4.4.11 Variant i: A curious race [medium]

In this simple game, more than one robot compete to reach first the finishing line far L cm from the starting line. The rules are: constant speed V during the motion, a single, intermediate stop at distance L/2 for a time equal to K*V with K given. The challenge is to choose V in order to minimize the total time and arrive first. The fun part of the challenge is to discover empirically this minimizing value; then it is useful to give the students more awareness, analyzing the relationship between the total time and the programmed motor power with a tool like *Geogebra* (this function is something like: t = K1/Power + K2*Power which has a minimum).

4.4.12 Variant j: Optional stop [difficult]

In this variant, not all the stops are compulsory: the vehicle stops only if the stop is pre-requested. This request is made for example touching a touch sensor (or putting a hand in front of the ultrasonic sensor) during the track that immediately precedes the desired stop. Thus, provide that the necessary sensor is mounted on the robot and connected to an input port. So, in the first of the two cases, you have to render the loop ‘sensitive’ to the pressure of the touch sensor during a track. For this purpose, the logic variable *requested* is possibly set to true in an independent loop whenever the touch sensor is pressed. This causes the next stop to be made; the variable is reset before starting the next track (fig. 16, 17, 18). Notice that, though not stopping, in this implementation for simplicity following motions are split into individual motions which stop at the end for a short while. Make the necessary modification in the upper loop if you want to use instead the ultrasonic sensor (see also the following variant).

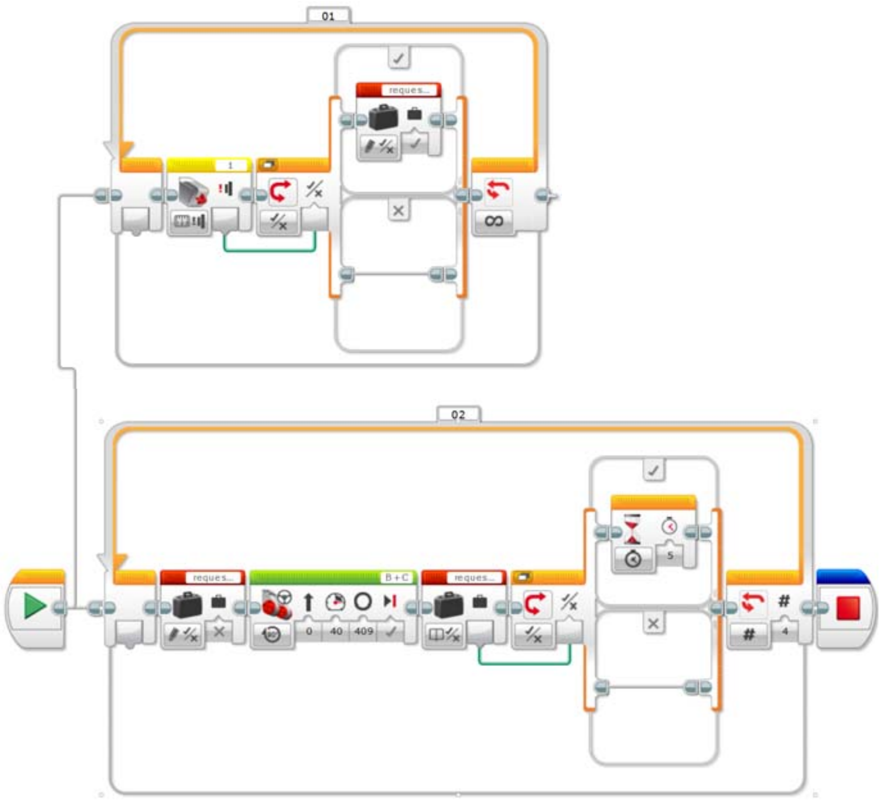

Figure 16. Optional stop, overall view

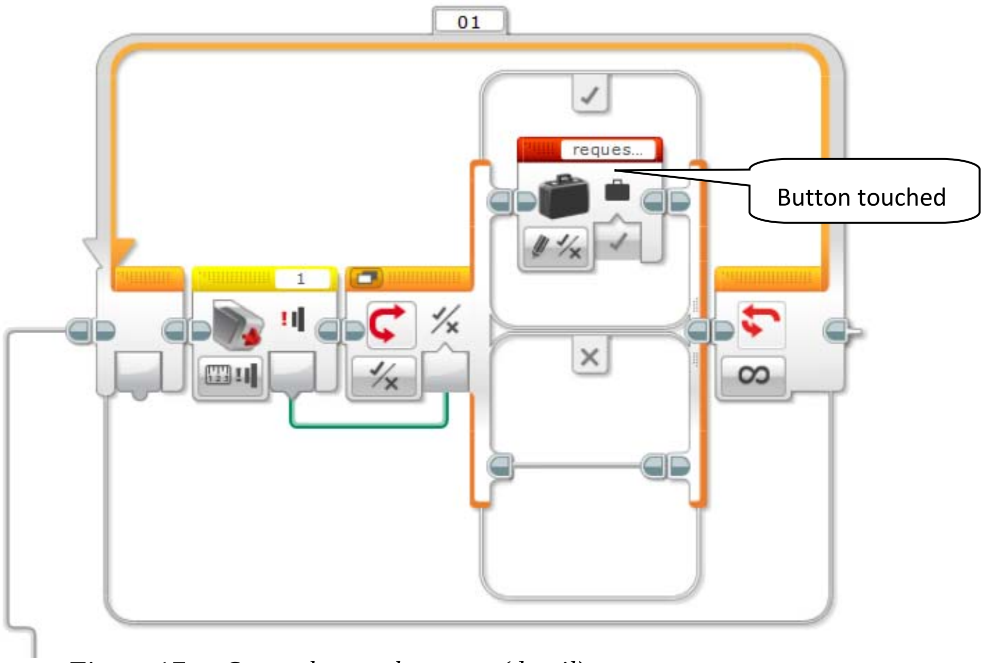

Figure 17. Sense the touch sensor (detail)

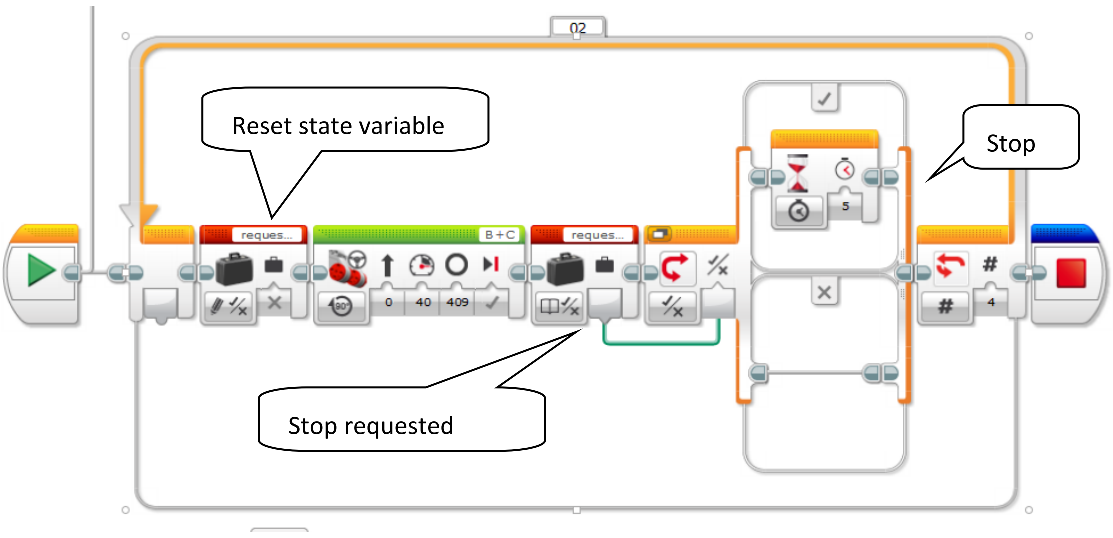

Figure 18. Conditional stop (detail)

4.4.13 Variant k: Variable distance and optional stop [difficult]

This is the combination of the approaches of the two previous variants and presents the same degree of difficulty of the optional stop.

4.4.14 Variant l: Initial acceleration and final deceleration [difficult]

For the sake of simplicity, we have so far considered constant speed motions; we have also supposed that our vehicle can instantly pass from a state of rest to a certain constant speed motion and vice versa. This is a rough approximation of the reality and actually the robot motors reach the final speed after a short time transition with rapidly increasing speeds (and the contrary when braking). To be closer to what actually happens, we should introduce the concept of acceleration as a variation in time of speed, as defined in Physics.

This variant is the attempt to reproduce something closer to the real behavior of a train: every track starts with an acceleration and ends with a deceleration. To emulate these two parts, it is necessary to impose a power profile which corresponds to the desired variation of speed. Assuming for simplicity the profile of fig. 19, there is a first time segment during which the power should regularly increase between 0 and the steady state value P, maintained during the second time segment, and in the final time segment the power must regularly decrease again to 0.

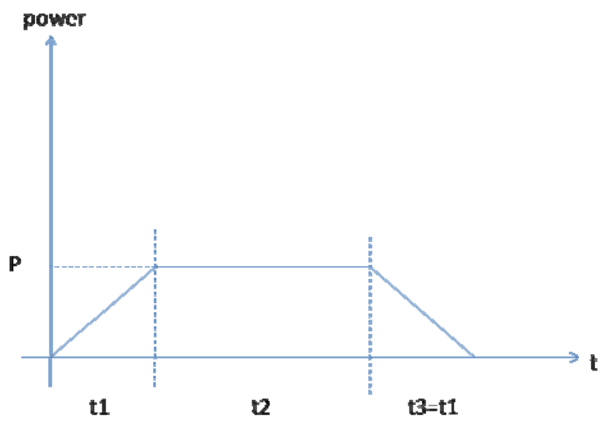

Figure 19. The desired power profile

Now the question is: how long should these three time segments be? Independently of the chosen speed profile, the covered space on every track must be D: for mathematical definitions, this latter corresponds to the area underlying the curve expressing speed versus time, calculated along the three abovementioned time segments. We have to calculate the area of an isosceles trapeze, and it should hold:

$s = D = V * [(t_1+t_2+t_3)+t_2]/2 = V (t_1+t_2)$

where V is the steady state speed in a track. If the students have realized variant d), they know the ratio speed/power and the can calculate the theoretical value for P. Actually, due to friction and other reasons, this area is only approximated by the motion of the real robot and to obtain the requested total displacement, all times must be empirically tuned. Fig. 20 shows a possible implementation: notice that acceleration and deceleration are emulated through some short steps 0.1 s long, each with a constant power: this makes the approximation even less precise. Using on-off move blocks assures a relative smooth motion of the robot.

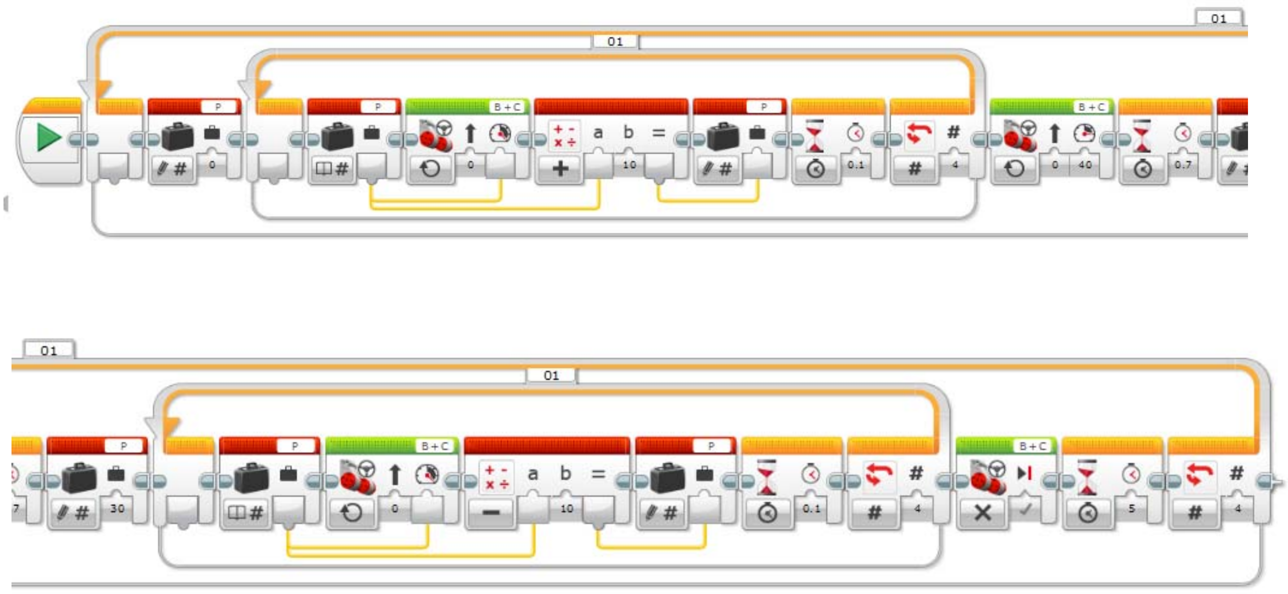

Figure 20. An approximated solution with acceleration and deceleration

Chapter 5: **Evaluation tools (O1.5)**

Use the rubric below to evaluate your students' achievement in each specific objective of this curriculum.

Name of student (or group of students): ……………………………………………………

upon completion of the activities described in this curriculum students achieved the following objectives	**Evaluation score** **0 = not attempted** **1 = attempted without success** **2 = partial success** **3 = completed with teacher's help** **4 = completed without teacher's help**
Built a simple tribot structure with two motors and a ball caster in the rear.	
Translated the scenario into a realistic representation on their desk	
Properly used the Move Steering command with its parameters to put the tribot in motion	
Instructed the tribot to travel a certain distance by trial and error experimentations	
Properly used the "wait" command to control the motion of the tribot	
Discovered and properly used the "loop" command to repeat the motion	
Made the tribot to reverse direction by trial and error experimentations	
Invented a mathematics-based solution for making the robot to travel a certain distance	