Development and Programming of KarelNXT Robot as a Simulation of xKarel Programming Language Including a Sample Program

PETR COUFAL, TOMAS HORNIK,
STEPAN HUBALOVSKY, MICHAL MUSILEK
Department of Cybernetics
University of Hradec Kralove
Hradecká 1227, Hradec Králové
CZECH REPUBLIC
petr.coufal@uhk.cz, tomas.hornik@uhk.cz,
stepan.hubalovsky@uhk.cz, michal.musilek@uhk.cz
http://www.uhk.cz

Abstract: - The article is dealing with programming of a KarelNXT robot made from a LEGO construction set. The idea comes from xKarel programming language, which is a traditional programming language utilizing a virtual robot only shown on a screen. The virtual xKarel programming language was extended by its implementation in LEGO Mindstorm NXT-G integrated development environment. Real robots built from LEGO construction sets use additional sensors (compass, color and ultrasonic) which we incorporated in such a way as to make possible building of a robot with equal functions to the robot in xKarel programming language. KarelNXT robot built from LEGO construction set is using the control unit in NXT version. In the article we mention the detailed description of individual robot movement instructions with a description of their meaning and functionality. Finally, we provide a sample program for comparison.

Key-Words: - Robot, LEGO, Mindstorms NXT, KarelNXT, xKarel, Sensors, Instructions, Programming Languages

1 Introduction

Specific programming languages are addressed in teaching of programming and when shifting to robotics it is necessary to change both the programming language and integrated development environment, which can be very confusing for the elementary school pupils. The issue is often being complicated by the fact that pupils have to learn everything all over again. To make said shift in teaching of programming easier it is convenient to harness the programming similarities. Initial simulation of a robot's movement positively affects the students' experience and makes the work with real robots less problematic [1,2].

Therefore we interconnect the programming language xKarel to the robotic construction set Lego Mindstorms NXT. The aim is to build the robot from Lego construction set using the NXT control unit that is according to our research [3] one of the most widespread units in teaching of programming at elementary and secondary schools in the Czech Republic. Complementary sensors are utilized by built robot Karel NXT in such a way it meets requirements of the robot Karel from program xKarel.

2 Construction of KarelNXT robot according to programming language xKarel

For proper construction of KarelNXT robot from Lego Mindstorms NXT construction set is absolutely necessary to understand the way in which robot Karel is moving in xKarel program. Based on the knowledge the requirements for both robot´s construction and features are set so that it corresponds to reality. In the control program a set of instructions is created to make the robot controlling analogous to xKarel program.

2.1 Programming language xKarel

According to the experts in programming xKarel program is a programming language (programming game) suitable for teaching of structured programming. The program is created in ANSI C++ using FLTK library. It is available at Microsoft Windows®, UNIX® and MacOS® platforms and localized into Czech and English language. [4]

The aforementioned program consists of two windows when the first of them is a room in which robot Karel is moving in square fields.

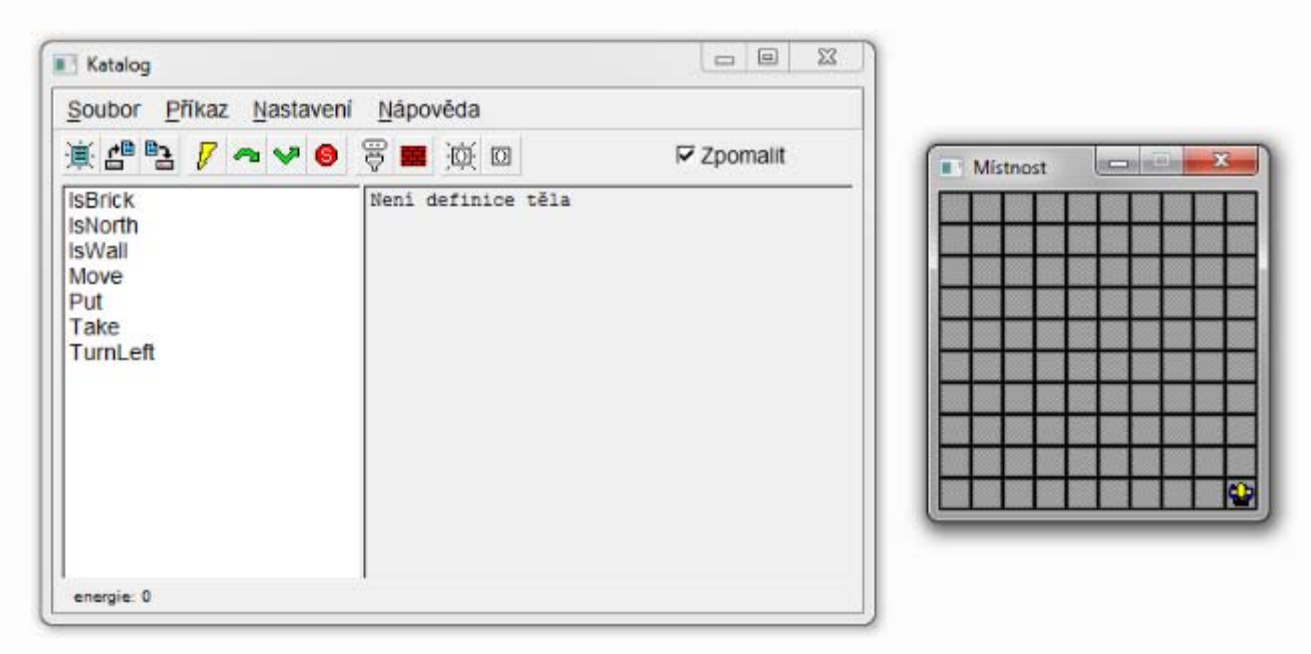

Fig. 1. xKarel development environment

The second window represents a catalogue designated for running the instructions and defining or editing procedures for robot Karel control. The catalogue enables further work with the program.

Robot Karel is situated in a room in which it is moving by means of instructions given to it by the user. The commands for the robot are formed by the instructions in the menu and the user can gradually define and broaden it. Having executed the program, the user selects the basic instructions *Move*, *Put*, *Take* and *Turnleft* as well as three conditions *IsNorth*, *IsWall* and *IsBrick*. For creating of additional instructions (Procedure) is possible to use aformentioned basic instructions and conditions or the cycle *While* and *While Not*. [4]

Basic instructions are indivisible instructions that robot Karel comprehends and other instructions are consisting of.

Move: robot Karel makes a step, which means it moves one square forwards in such direction it is heading. If it is situated in front of the wall, the error appears.

Put: robot Karel puts down a brick in a square where it is standing. In case of exceeding of the maximal bricks quantity in a field, there occurs the error.

Take: robot Karel lifts the brick of the occupied position. When the brick is not in the field, the error emerges.

Turnleft: robot Karel turns left 90 degrees.

The conditions are to be used for better robot control. With the help of conditions we can separate the instructions that are and are not to be conducted.

IsNorth condition is tracking robot´s Karel heading northwards, if the heading is correct the instruction sequence is executed. *IsWall* condition is considered to be true when robot Karel is standing directly in front of the wall. In such a situation the robot cannot make any other step forwards, otherwise it would hit against the wall. *IsBrick*, the third condition, is true if robot Karel is standing in a field where the brick is located. The conditions can be improved by using Else instruction which helps the user to conduct the instruction sequence if the condition is not fulfilled.

In loops we use *While* Condition {the sequence of instructions executed when the condition is fulfilled} or *While Not* Condition {the instruction sequence for not fulfilling the condition}. One of the examples of the sequence using *IsNorth* condition is *ToNorth* instruction that is moving the robot heading northwards before it hits against the wall and terminates the program with the error.

```
procedure ToNorth
{While IsNorth {Move}
}
```

Fig. 2. ToNorth program - xKarel

2.2 Construction and programming of KarelNXT robot

Robot model built from Lego® construction set has to meet the following requirements in a way it responds robot´s Karel behaviour in xKarel program:

- Sensor detecting the wall in front of the robot
- Sensor detecting the brick in a square where the robot is standing
- Sensor determining the cardinal points
- Robot is able to turn 90 degrees on the spot
- Robot is able to inform about potential error on its screen
- Robot is able both to pick and put the bricks together

For meeting the requirements the space representing the room for robot´s movement is essential. The criteria are as follows:

- The room is divided into square fields
- The room includes walls
- The room contains bricks

Based on aforementioned requirements KarelNXT robot model was built and room for its movements was designed.

Robot construction

KarelNXT robot is built from Lego® construction set using NXT control unit. The construction operates parts of 9797 and 9695 educational kits. In addition there is the compass sensor, the colour sensor and the carboard brick feeder. The robot is using two vertically placed engines for its movement. The bottom part of the robot´s chassis operates the rubber strip in three wheels delta arrangement. Strip bottom part of the chassis enables it turns in any direction on the spot. The third engine with the help of the rubber wheel placed on the carboard brick feeder entrance serves as a tool for collecting and piecing the bricks together. Above the robot´s engines is NXT control unit. The positioning helps the user to read on-screen information as well as easily manipulate and launch the robot. Another undeniable advantage is the easy availability of all robot´s ports. In the raised platform above the control unit can be found the compass sensor used for determining the North. Another sensor is placed in the front-lower part of the robot right in between the engines. The sensor in question is an ultrasonic sensor for the distance measuring used for detection of the walls in the room. In the lower part of the robot, in front of the cardoard brick feeder entrance, is the colour sensor functioning as a brick detector. For accuracy control of the robot´s position a gyroscope sensor is often used. Not even that guarantees exact 90 degrees turning of the robot as stated by Dave Riske [5] in case of robot Karel from EV3 construction set.

Room construction

Considering the size of the robot a room was created with the help of the duck tape on light straight pad. Length of the square field side is 30 centimetres. From the child construction set blocks are built in a way they represent the walls in the room so that it would be possible to built even irregular rooms. The bricks demonstrate rollers 2.5 cm in diameter and 0.7 cm high with the inclined edge of 0.3 cm in 45 degrees. The bricks are made up of hard wood and painted red. The amount of the bricks in one field is limited by one brick.

Fig. 3. KarelNXT robot construction

Functions of the sensors

The compass sensor

The compass sensor represents an additional sensor for Lego Mindstorms NXT construction set from HiTechnic company, functioning for robotic models navigation. Digital compass operates with 1° azimuth accuracy, representing values from 0° to 359°, which enables its own definition of the four cardinal points for a room in any direction. Sensor is restoring given value 100 times per second. It is working in two modes – reading and calibration. Compass sensor is susceptible to interfering magnetic field of the engines. Therefore it is located in a raised spot in the robot´s construction.

The colour sensor

In other words this is an optical sesor making the colour detection of the scan surface much easier. Sensor is able to distinguish six colours (red, blue, green, yellow, red and white) marking them with numbers or selected colour range. The sensor is located in the robot construction for recognizing red bricks situated in a field in front of the robot.

Ultrasonic sensor

This sensor is based on the sonar principle and serves for distance measurement in 0-250 cm or 0-100 inches range with ± 3 cm accuracy. The measurment accuracy is influenced by the size, surface, material and the shape of the object which reflects the wave motion back to the sensor. For mor accurate distance measurement the EOPD sensor can be used. [6] The ultrasonic sensor has been found appropriate enough for both robot´s KarelNXT construction and its requirements.

Robot´s program

KarelNXT robot is programmed in the LEGO MINDSTORMS NXT 2.0 program that is a part of the construction set. It is a case of iconographic programming language in which to each active component of the robot the program block is assigned. In the program block the user can set particular parameters of the active robot component. Both simplicity and easy orientation in the program enable its using when teaching in primary schools.

Program makes possible the creating of individual program blocks used during program making for KarelNXT robot. In fact the program block is a subprogram important for setting the final robot operating program.

Easy instructions: *Move, Put, Take and Turnleft*

Move - In our case *Move* program block is testing right from the beginning whether there is a wall in front of the robot or not. If there is no obstacle, the robot moves one square field forwards, whereas after detecting of the wall there is an audio signal and on the robot´s display there appears warning sign “Error, there is a wall.”, and the robot stops moving.

Put -Put program block is testing if there is a brick in a field. When there is no brick, the robot puts the brick in the field. In such a case a brick is already lying in the field, a sound is heard as well as the warning text “Error, there is a brick!” is displayed on the screen and the robot does not put the brick.

Take - From the very beginning Take program block is taking into consideration whether there is a brick in a field or not. If there is a brick in a field, the robot lifts it up. On the other hand there is no brick lying in the field, there is a warning signal and on the robot´s screen there appears a text saying “Error, there is no brick!”.

Turnleft - Turnleft program block is answering for robot´s moving in 90 degrees left on the spot.

Conditions: *IsNorth, IsWall, IsBrick*

XKarel language syntax offers three conditions altogether used also for KarelNXT robot with the help of program blocks. Given program blocks are functioning as follows:

IsNorth program block compares the obtained value from the compass sensor with defined North range (absolute reading 85-95). If the detected value falls within the defined range, the robot is heading north and the first part of the program is being executed. In the opposite situation it is the second part of the program´s turn.

IsWall program block determines whether a wall is or is not situated in front of the robot. When the distance of the obstacle in front of the robot measured by the ultrasonic sensor is lesser than 20 centimeters, the program block assesses such a situation as a wall obstacle and the first part of the program is executed. After detecting of the distance larger than 20 centimeters there starts the second part of the program for this is a signal meaning there is no wall in front of the robot.

IsBrick program block finds out if there is a brick in a field under the robot. Using the colour sensor it compares the colour of the objects located under the robot. When the red colour is detected, the brick is under the robot and the first part of the program starts. When a brick is not in a field under the robot, the second part of the program is executed.

Both the first and the second part of the program is defined by the user in each condition from the program-block menu.

Cycles

For better robot control are used cycles operating with aforementioned defined conditions.

While condition and *While not condition* - In the first case order sequence is being repeatedly executed for the duration of condition validity whereas during the second one the sequence is being executed as long as the condition is not valid. To make the work with the program more transparent and easier six program block cycles were created:

WhileIsNorth program block controls robot´s heading to the north, in the positive situation it accomplishes pre-defined order sequence given by the user. Together with the check of robot´s heading to the north it is responsible for further program block repetition.

WhileNotIsNorth program block represents a negation of *WhileIsNorth* program block.

WhileIsWall program block controls if there is a wall in front of the robot. When a wall is actually found there, the program block starts the order sequence having been defined by the user and if necessary it repeats the program block.

WhileNotIsWall is a negation of *WhileIsWall* program block.

WhileIsBrick program block controls the position of a brick in a field under the robot and it executes given order sequence. As in other aforementioned cases together with the brick detection in a field under the robot follows with further program block repetititon.

WhileNotIsBrick program block is functioning as a negation of *WhileIsBrick* program block.

Other program blocks

With the help of all aforementioned program blocks the user is able to create a program for KarelNXT robot as it would be in xKarel programming language. In Lego Mindstorms NXT 2.0 is not possible to insert one program block to another, which means there is no recursive calling as in xKarel program. Now and then this is found to be a disadvantage.

We show two short programs for comparison and illustration.

TurnRight (xKarel)

```
Procedure TurnRight
{
Turnleft Turnleft Turnleft
}
```

Fig. 4. TurnRight - xKarel

TurnRight (KarelNXT)

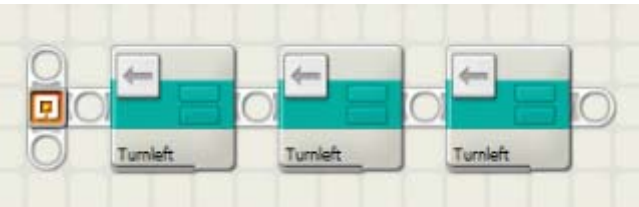

Fig. 5. TurnRight - KarelNXT (Lego Mindstorms NXT)

4 Conclusion

Building both KarelNXT robot and its control program block enables us to interconnect computer programming teaching with real model robot programming teaching. KarelNXT robot is built from Lego construction set using additional compass sensor and cardboard for brick feeder. The robot is commanded with given program blocks designed according to xKarel programming language syntax. In Lego Mindstorms NXT 2.0 program a set of program blocks *Move, Put, Take, Turnleft, IsNorth, IsWall, IsBrick, WhileIsNorth, WhileNotIs North, WhileIsWall, WhileNotIsWall, WhileIsBrick, WhileNotIsBrick* was created. Further program blocks are being made by the students during robot programming teaching. We also plan to focus on research and comparison of teaching of programming both in xKarel language and KarelNXT robot.

*Acknowledgments***.** The paper has been supported by Specific Research Project of Faculty of Science, University of Hradec Kralove, 2017 and by Specific Research Project of the Faculty of Education, University of Hradec Kralove, 2017.

References:

[1] Klassner, F., Kearney, S.: An Evaluation of Simulation in LEGO Mindstorms Robot Programming Coursework. Las Vegas, CSREA Press, pp.3-9, ISBN: 1-60132-435-9, (2016).

[2] Slangen, L.,Van Keulen, H., Gravemeijer, K.: What pupils can learn from working with robotic direct manipulation environments. International Journal of Technology and Design Education, pp. 449-469, ISSN 0957-7572, (2011).

[3] Coufal, P.: Robotics in Education. Diploma thesis, University of Hradec Kralove, (2016).

[4] XKarel home page [online]. Praha, Robot Karel implementation. Accessed June 20, 2017. Available: http://xkarel.sourceforge.net/eng/. (2017)

[5] Building Karel the Robot: Invaluable Learning Experience for Students [online]. Carson City, Dave Riske. Accessed June 20, 2017. Available: https://nclab.com/building-karel-robot-LEGO/. (2017)

[6] "EOPD – How to measure distance." In: *HiTechnic Blog.* [online]. Miami: HiTechnic Products, 2010. Accessed June 20, 2017. Available: http://www.hitechnic.com/blog/eopd-sensor/eopd-how-to-measure-distance/. (2017)