A Constructive Approach to Teaching with Robotino®

Dragan Pršić [1*], Vladimir Stojanović [1], Vladimir Đorđević [1]
[1] The Faculty of Mechanical and Civil Engineering in Kraljevo, University of Kragujevac, Serbia
[*] prsic.d@mfkv.kg.ac.rs

Abstract: *The current level of technological development and an increasing need for Information Technology (IT) staff open up opportunities and impose a need for a different approach in students' education. Instead of classical, theoretically-oriented, transfer of finished knowledge, the focus is put on interactive learning through student's own experimental work. Modern mechatronic systems offer the possibility of changing the way of acquiring technological knowledge and developing programming skills. This paper presents some possibilities of mobile robotic system Robotino® of Festo Didactic intended for education and research. It is an open, flexible learning, platform that enables entry into the world of mechatronics and information technology for students of a different level of prior knowledge. In the paper, more details are given about the drive subsystem.*

Keywords: *education; Robotino®; mobile robot; drive system*

1. INTRODUCTION

The fourth industrial revolution imposes the need in the education of students of technical faculties to move the focus toward mechatronics and IT. For example, in mechanical engineering, a certain saturation has been achieved in terms of traditional constructive solutions. Research is mainly based on new materials and application of IT in the development of various smart devices. Systems become more and more hybrid and represent a combination of different technical areas.

Changes are needed not only in areas that being studied but also in the way that these new areas are presented. Students are required to actively participate in classes. This raises the question of how to motivate students for interactive work, and how to attract them to enroll, in general, technical faculties in greater numbers. Predictions are that in the future will be an increasing need for mechatronics engineers and IT technology.

Research in recent years [1-3] show that application of mechatronic and robotic systems in education can improve teaching and attract a greater number of students of IT and engineering disciplines in general. It has been shown that even for students of computer orientation, robot programming is more attractive than, for example, web application.

Application of such systems change traditional teaching methods and the role of involved actors in the learning process. Instead of traditional knowledge transfer, a good part of learning responsibility is shifted to the learners themselves. Carefully selected examples, with friendly and fan working environment, can positively influence to their interest and motivation, and thus to more actively participate in education in a self-directed manner.

2. MOBILE ROBOT Robotino®

Robotino® (Fig. 1) is mobile robot system made by Festo Didactic [4]. It is, primarily, intended to acquire practical knowledge and skills in the field such as robotics, mechatronics, measurements, wireless control, signal processing, and programming, etc.

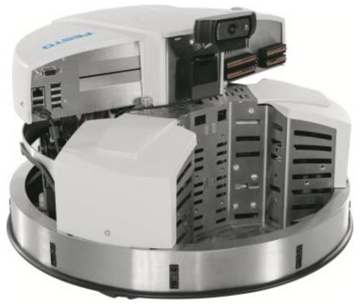

Figure 1. *Mobile robot system Robotino®*

Some of the properties:

- autonomous motion in all directions with identifying and avoiding obstacles;
- embedded sensors and actuators;
- wireless communications with other devices;
- the possibility of upgrading new components by using mounting tower;
- open sources concept and software interfaces for different programming languages;

All these features of the Festo mobile system enable course participants, through practical, interactive work, to get acquainted with different areas in the field of automation technology.
Although Robotino® is also used for research in the field of industrial production, transport, logistics, in this paper, the attention is devoted to its application in education.

1.1. Robotino® subsystems

The mobile robot consists of several subsystems:

Control subsystem

To control the whole system it is used an embedded PC, according to the COM Express standard. The OS (Linux, Ubuntu) and all user data are stored on an SSD disk, which facilitates the replacement of an existing PC with a new one with more computing power. For communication, the standard connectors are used: USB, PCIe, RS232, Ethernet, VGA. The embedded PC is connected to a 32-bit microcontroller that controls the motors and the I/O interface of the robot. A FPGA module is used for additional signal processing.

Drive subsystem

For the robot motion, multidirectional drive (omnidrive) is used, which allows planar motion in any direction without having to rotate. Translation speeds up to 10 $[km/h]$ are possible. More about this subsystem will be discussed below.

Sensors

The standard configuration of robot system contains a whole range of sensors that allow to get a detailed picture of the environment and autonomous motion. There are available:
- infrared distance sensors;
- inductive sensors;
- optical sensors;
- gyroscope;
- rubber protection strip with built-in collision-protection sensor;
- color camera with USB interface;

Supply subsystem

This subsystem provides autonomy to the mobile robot up to four hours. Power is supplied via two serial connected 12 V rechargeable batteries. The system is automatically switched off if the supply voltage is too low. We can develop control program or carry out experiments with restricted motion while batteries charging.

Software subsystem

This subsystem is probably the most important grummet of the whole robot system. Through it, the user's creativity is realized from the beginning level to the advanced programming. Robotino® supports the open-source concept. The source code of all Robotino® software is freely available. Robot control can be realized at one of three levels:

Level 1: Web interface

Using this interface, it is possible to control the movement of robot without programming. Embedded web server with a graphical user interface (GUI) can be accessed by any device with WLAN communication and Internet browser. Besides for motion control, web server is used to monitor status, set operating parameters, and on-line help.

Level 2: Software tools

For programming beginners, software tools with graphical interfaces (*Robotino View*, *EA09View*) are available. Using these tools it can be achieved easy control and monitoring the on-line process parameters via WLAN network. For all hardware components (motor, encoder, sensors, I/Os, camera, gripper, robot arm) *Robotino View* contains the corresponding functional blocks classified into libraries (Fig. 2). By using these modules we can control the robot motion and process signals from the sensor in the graphical environment according to the GRAFCET standard. In addition, the user can write his own functions using *Lua* scripts or C++ programs.

Robotino View can be installed and used either on external PCs or on embedded PCs.

Level 3: Application programming interface

For advanced users, interface functions (APIs) are available for a variety of programming environments. User programs can be written using: C/C++, JAVA, .Net, LabVIEW, MATLAB/Simulink, ROS and Microsoft Robotics Developer Studio.

To communicate with Robotino, any device with a supported wireless communication can be used (PC desktop, laptop, tablet, mobile phone, Arduino, Raspberry). The external access point supports the IEEE 802.11g WLAN standard. By selecting a mode of operating (master or client), different combinations can be performed in the local network: multiple mobile robots can be controlled by one external device or by multiple external devices it can be controlled one Robotino® (Fig. 3) [5].
Robotino SIM tool can be used to test the written program (Fig. 4). This tool enables simulation of robot motion in virtual 3D environment that we create ourselves. This helps us to detect errors in the program before it is used on a real robot.

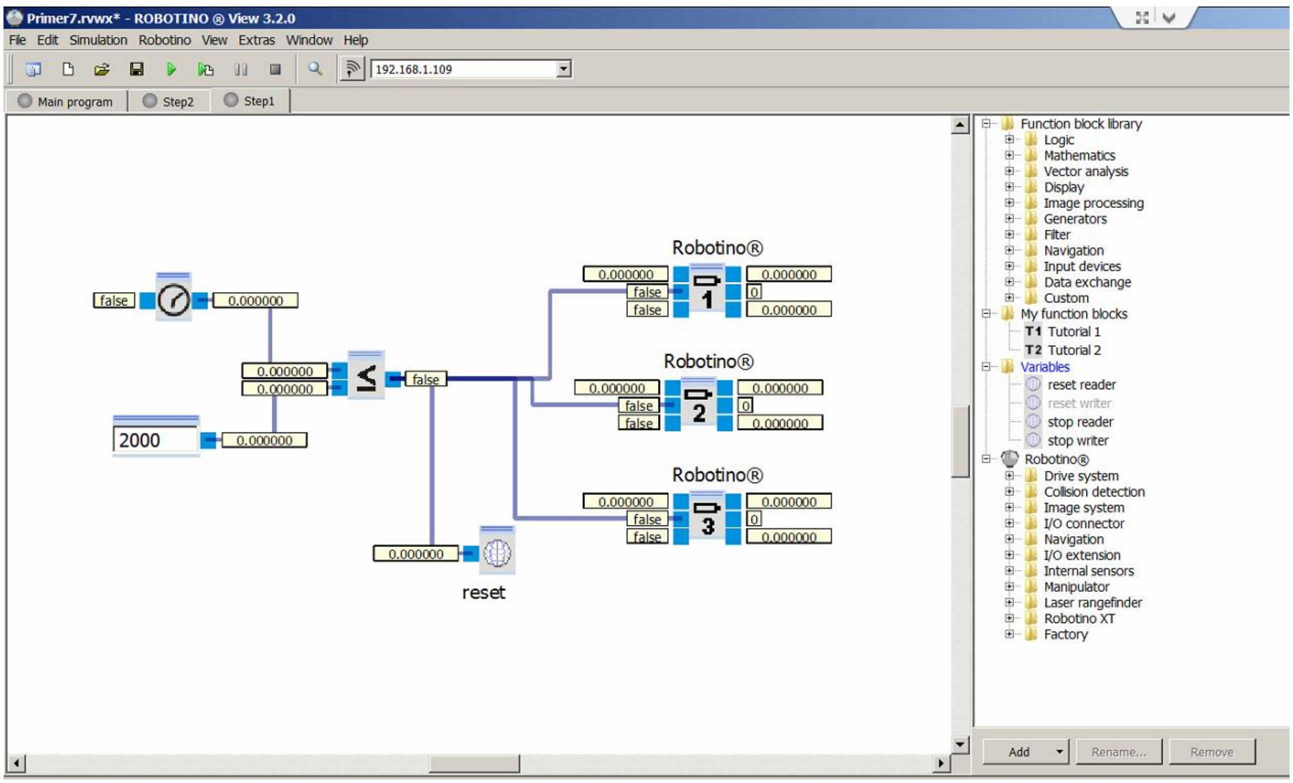

Figure 2. *Robotino View programming environment*

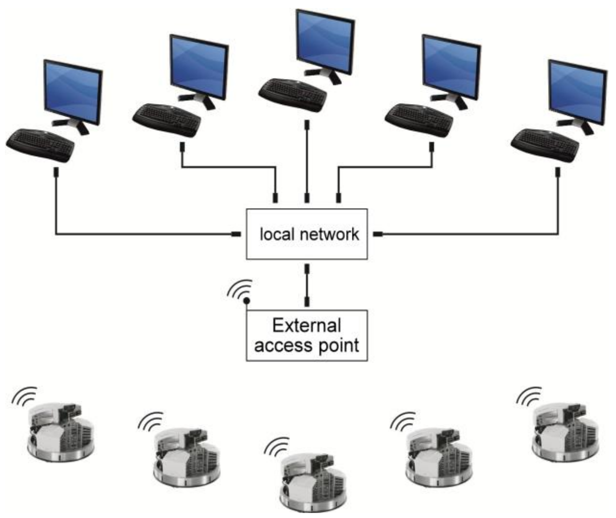

Figure 3. *Different ways to setting up LAN [5]*

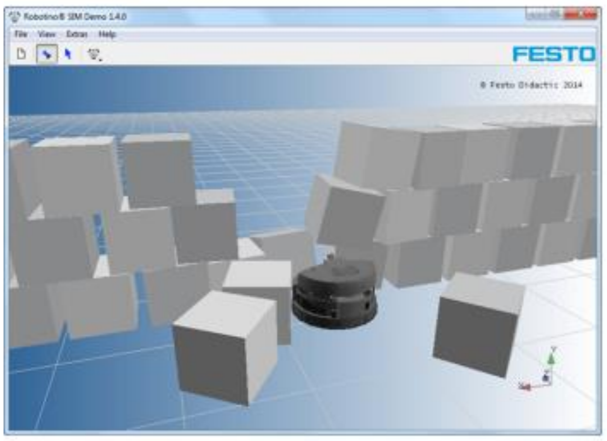

Figure 4. *Robotino SIM tool*

Constructivist approach to teaching [6] supposes taking into account the learner's existing knowledge and skills during the learning process. We intend to demonstrate, in the case of the drive subsystem, the possibility of choosing the level of control complexity, depending on the individual knowledge and experience of the students.

3. DRIVE SUBSYTEM

Drive subsystem consists of three independent drive units with its own power. Each unit consists of a servomotor, a gear unit and a main wheel. The main wheels are placed at an angle of 120° (Fig. 5).

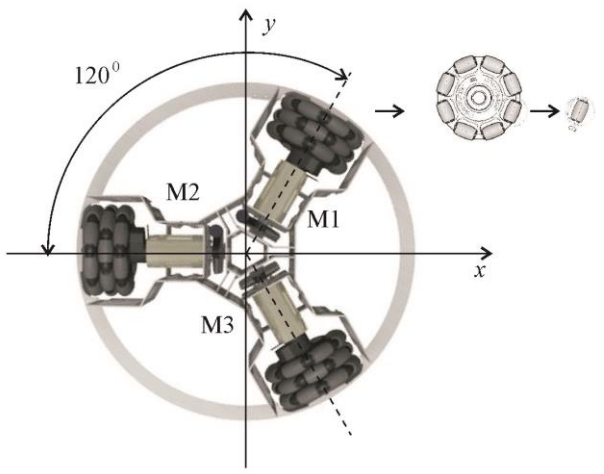

Figure 5. *Robotino® omnidirectional drive*

Auxiliary omnidirectional wheels (or casters) are tangentially built in the circumference of the main wheels. Auxiliary casters are barrel-shaped and enable the movement of each wheel practically in any direction with rotation on the spot. Thanks to them, friction forces are reduced in the lateral movement of the drive wheels. Each of the main wheels has an active movement of its own drive motor and the passive lateral movement of the two remaining drives. Three brushed DC motors used as drive are controlled with feedback control loop. The rotational speed is measured by the incremental encoder. The information from the encoder is also used to determine the position of the mobile robot. A microcontroller connected to the FPGA module is used as a control unit. It can realize different control algorithm. A planetary gear unit with a transmission ratio of 32:1 is used between the drive shaft and the wheel.

The motors are controlled by independent control signals so that Robotino® has three degrees of freedom, two translational and one rotational motion. Rotation around the vertical axis is possible in both directions. The turning circle is equal to zero. There is no need for shunting.
Movement can be controlled in several different levels depending on the knowledge, experience, and purpose of the course. At the initial level, without any programming, you can use some of the standard Internet browsers through a desktop computer or one of the mobile devices. The GUI is used for communication, is shown in Fig. 6.

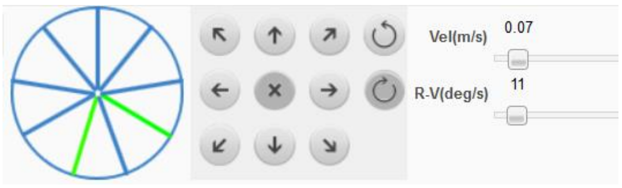

Figure 6. *Web interface for manual control*

After adjusting the desired motion speed $V_{el}\ [m/s]$ and $R-V\ [\deg/s]$, one of the directions of translational or rotation motion, is set. If necessary, Robotino® can be stopped at the current position. The interface in the circle on the left, with color change, indicates the presence of an obstacle. Signals are received from nine infrared distance sensors arranged in scope. Depending on the proximity of the obstacle, the radius colors change from blue (no obstacle) to red (the obstacle is very close).

For the first steps in Robotino® programming, the use of the *Robotino View* software tool is recommended (Fig. 2). For controlling of each of the servomotors, it is used one functional block (Engine #n) from the library *Drive system*. Fig. 7 shows the functional block for control of the motor No1.

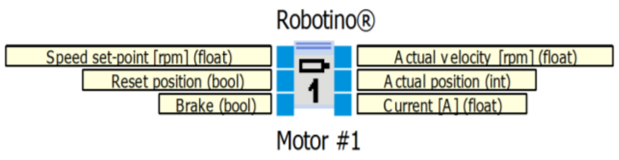

Figure 7. *Motor #1 Function block*

The interface of this block consists of three input (left side) and three output ports (right side). At the first input port we define desired rotation speed of the motor in [rpm]. This value can be constant or variable, positive or negative, and is stored in the variable of type float. The positive speed of each motor causes rotation of the robot in a counterclockwise direction. The actual motor speed can be read at the first output port. Boolean variable to the second input port resets the value of the incremental encoder of the motor. A reset signal can be generated by software or via one of the digital inputs. In relation to the position, when the reset is performed, the second output port shows the number of pulses (int) generated by the encoder. For one full turn of the motor shaft, the encoder generates 2048 pulses. On the basis of this data, the path that was passed by the appropriate omniwheel can be calculated. The relationship between the indication of the encoder (n_e) and the angle of rotation of the motor (φ_m) is given by the equation (1):

$$\varphi_m = \frac{\pi}{2^{10}} n_e \qquad (1)$$

while the relation between the encoder value and the angle of rotation of the omniwheel (φ_w) is given by the equation (2):

$$\varphi_w = \frac{\varphi_m}{2^5} = \frac{\pi}{2^{15}} n_e \qquad (2)$$

Although encoder generates discrete signals on changing the angle, the resolution is high and practically register very small changes in angle. With omniwheel, they are less than 1/100 part of the degree.

On the third output port, load current of the motor is measured. Using this information, trajectory optimization, in terms of electricity consumption, can be performed. The presence of a logical unit at the third input port stops the motor. As with the encoder reset, this signal can be generated by software or hardware through the I/O interface. Changing the direction of the motor rotation and sudden stop is realized by the H-bridge circuit. Moving the robot depends on the control signals on all three motors. The resulting velocity vector is equal to the sum of the velocity vectors of each omniwheels. Table T.1 gives a few examples of the relationship between robot motion and speed set-point (n_{sp}) each of motors.

For predefined directions of motion, which are determined by the position of the motor in Robotino chassis, the speed set-point can have an arbitrary value. The speeds ratio of individual motors is only important. In order to establish a connection between the movement of the robotic system and the rotation of the motor, it can be used the functional *Omnidrive* block shown in Fig.8.

Table 1. *Relation between robot motion and motors speed set-point*

Direction of the robot motion	Motor 1 [rpm]	Motor 2 [rpm]	Motor 3 [rpm]
Linear travel at the angle 0^0	$-n_{sp}$	0	n_{sp}
Linear travel at the angle 60^0	0	$-n_{sp}$	n_{sp}
Linear travel at the angle 120^0	n_{sp}	$-n_{sp}$	0
Linear travel at the angle 180^0	n_{sp}	0	$-n_{sp}$
Linear travel at the angle 240^0	0	n_{sp}	$-n_{sp}$
Linear travel at the angle 300^0	$-n_{sp}$	n_{sp}	0
Rotation on spot (counterclockwise direction)	n_{sp}	n_{sp}	n_{sp}

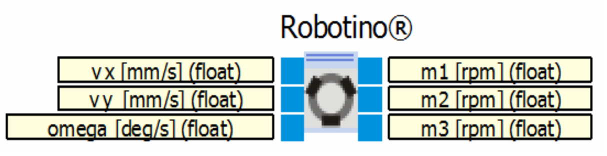

Figure 8. *Functional block Omnidrive*

At the input ports the speeds of translational motion along the x and y axis and speed of rotation around the vertical axis are set. As a result, the rotation speed of each motor is obtained at the output. The purpose of the Omnidrive functional block (inverse) (Fig. 9) is the opposite. Based on the speed of motor rotation, the motion speed in plane is determined.

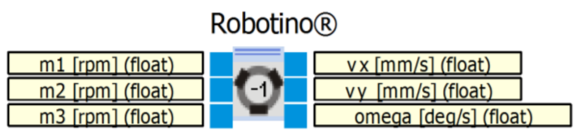

Figure 9. *Functional block Omnidrive (inverse)*

If an additional analysis of robot movement is desired, e.g. in Matlab®, analytical relations that describe the relationship between these movements are given below. For the arbitrary motion of the mobile robot defined by the translation v $[mm/s]$ and rotation ω_R $[\deg/s]$ (Fig. 10), the speeds of the motors can be determined as follows.

Speed of wheel 1 v_{W1} $[mm/s]$:

$$v_{W1} = \frac{R\pi}{180}\omega_R - v\cos(\alpha + 30) \tag{3}$$

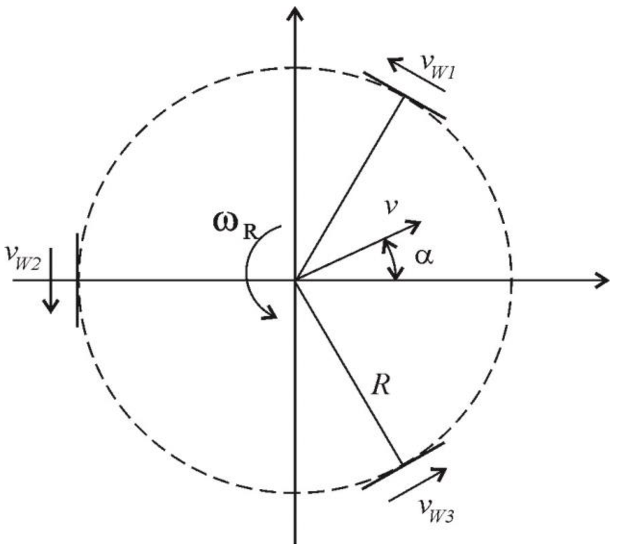

Figure 10. *Relations between robot speed and wheel speed*

i.e. the angular speed of the motor 1 $[rpm]$:

$$n_{M1} = \frac{1920}{\pi d}\left[\frac{R\pi}{180}\omega_R - v\cos(\alpha + 30)\right] \tag{4}$$

Speed of wheel 2 v_{W2} $[mm/s]$:

$$v_{W2} = \frac{R\pi}{180}\omega_R - v\sin(\alpha) \tag{5}$$

i.e. the angular speed of the motor 2 $[rpm]$:

$$n_{M2} = \frac{1920}{\pi d}\left[\frac{R\pi}{180}\omega_R - v\sin(\alpha)\right] \tag{6}$$

Speed of wheel 3 v_{W3} $[mm/s]$:

$$v_{W3} = \frac{R\pi}{180}\omega_R + v\cos(\alpha - 30) \tag{7}$$

i.e. the angular speed of the motor 3 $[rpm]$:

$$n_{M3} = \frac{1920}{\pi d}\left[\frac{R\pi}{180}\omega_R + v\cos(\alpha - 30)\right] \tag{8}$$

Fig. 11 shows the program written in Matlab® using equations (3) - (8) to realize the oscillatory motion along *x* axis according to the law:

$$n_M = 200 * \sin(2 * \frac{2\pi}{T_{kraj}} * time) \tag{9}$$

Fig. 12 shows the changes of the reference/actual speed of Motor #1.

```
%Motor control
 PridruziBiblioteke;
 IPAdresa='192.168.1.109';%IP address
 ComId = Com_construct; %Kanal za komunikaciju
 Com_setAddress(ComId, IPAdresa);
 Com_connect(ComId); %komunikacija
 Motor1Id  = Motor_construct(0);
 %...
 Motor_setComId( Motor1Id, ComId );
 %...

 tstart=tic;
 k1=1;
 Tkraj=30;
 while toc(tstart)<=Tkraj
   time(k1)=toc(tstart);
    W1z(k1)=200*sin(0.3*time(k1));
    W2z(k1)=W1z(k1); W3z(k1)=W1z(k1);
    Motor_setSetPointSpeed( Motor1Id, W1z(k1));
    %...
    W1(k1)=Motor_actualSpeed(Motor1Id);
    %...
    k1=k1+1;
 end

 Motor_destroy(Motor1Id);
 %...
 Com_disconnect(ComId);
 Com_destroy(ComId);
```

Figure 11. *Matlab® program for motor speed control*

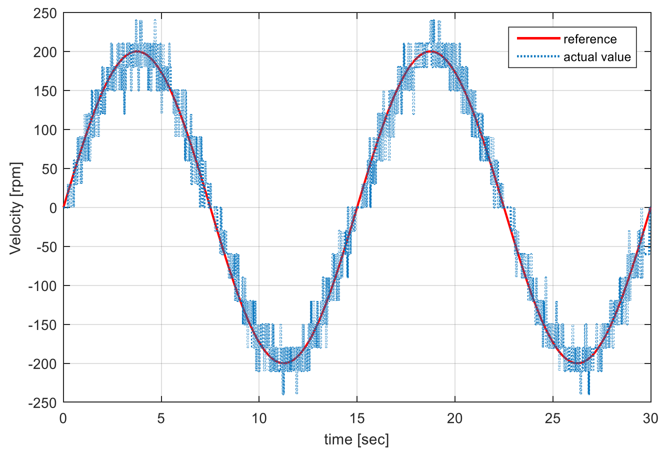

Figure 12. *Reference/Actual speed of Motor #1*

4. CONCLUSION

After a one-year experience, our impression is that the concept of Robotino® learning system is well-designed and consistent with state-of-the-art didactic equipment in the field of automation technology. This is understandable because it is the product of one of the longstanding leaders in this field. However, the realization of this concept has not yet been brought to the end, especially in terms of accessory equipment (e.g., Robot arm) and supporting documentation (e.g. Manual) for more advanced applications. The documentation must be much more detailed for educational equipment. There are two reasons that justify this lack somewhat. First, Robotino® based technologies are developing rapidly and are in constant change. The only way to answer them is online support that is constantly updated. For example, for a home appliance, it is desirable that the instructions do not change anything. However, for one education-oriented system, improvement of opportunities is expected. It is interesting that online documentation of each Robotino® subsystem can also be accessed through its own web server. Secondly, Robotino® is a flexible system and is open to change. In addition to standard applications, illustrated by interesting project tasks, users are encouraged to develop their use case study. If the price is neglected, Robotino® is certainly a good choice for involving young generations into the Industry 4.0 flows.

REFERENCES

[1] Qidwai, U., Riley, R., El-Sayed, S., (2013). Attracting students to the computing disciplines: A case study of a robotics contest, *Procedia - Social and Behavioral Sciences,* 102, 520 – 531.

[2] Tocháček, D., Lapeš, J., Fuglík, V., (2016). Developing technological knowledge and programming skills of secondary schools students through the educational robotics projects, *Procedia - Social and Behavioral Sciences,* 217, 377 – 381.

[3] Liliana, D., Florina, P.S., (2015). Education, Knowledge and innovation from a mechatronics perspective, *Procedia - Social and Behavioral Sciences,* 203, 205–209.

[4] Festo Didactic Official site http://www.festo-didactic.com

[5] Robotino® Workbook, (2016). Festo Didactic

[6] Denicolo P., Pope M., (2001). *Transformative professional practice: Personal construct approaches to education and research*, London: Whurr Publishers.