

Control Engineering At School: Learning By Examples

Sergey A. Filippov*, Alexander L. Fradkov**,***

*St.Petersburg Phys&Math Lyceum 239 (e-mail: safilippov@gmail.com)

**Institute for Problems of Mechanical Engineering, Russian Academy of Sciences, 61 Bolshoy ave V.O., St.-Petersburg, 199178

*** St.Petersburg State University, Russia

fax: +7(812)321-4771, (e-mail: fradkov@mail.ru)

Abstract- The joint project of St.Petersburg State University and St.Petersburg Phys&Math Lyceum 239 «Cyberphysical laboratory» has started in 2008. As a result of the project the technique of teaching the elements of the control theory at school has been developed. Using the simple devices on the basis of Lego Mindstorms NXT, students of elementary school have started to master a science currently accessible only to high school or university students. In the paper some ideas and recent findings of the project are described.

1. INTRODUCTION

An important and interesting teaching task is the «bridgebuilding» between knowledge domains of the experts and the schoolchildren, helping schoolchildren to see perspective of their future profession, i.e. to carry out vocational guidance, and to students to see practical applicability of their future professional knowledge.

There exist two approaches for training schoolchildren to the compilation of control algorithms by robot:

1) using the standard basic algorithmic structures studied within computer science;

2) using the elements of the control theory.

The first approach is applied much more often, but it doesn't uncover diversity of possibilities of control. Its typical representation is the relay regulator based on branching "if". In the elementary variant on-off, then, for the advanced programmers, three-position and more (multiple selection). In an on-off regulator the regulating organ under the influence of a signal from the sensor can accept one or the other extreme position: either "open" or "closed". Thus correction action on adjustable object can be only maximum or minimum.

The second approach is more difficult for understanding by schoolchildren. However it comprises a key to further development. The approach is based on the concepts of feedback and error-closing control explained by studying the simplest standard controllers – proportional and differential, (for advanced students also integral).

Having gained a good experience of using the first approach [1] - we have passed to the second one and have developed a technique of teaching automatic control by elementary examples. Each of examples sequentially dares by means of a relay controller, and then by means of proportional one and its compositions with other controllers. It allows a pupil to feel advantages of the

most simple mathematical methods in control and to concentrate attention to them.

Since mathematics has become one of basic elements in robotic technology learning in the lyceum #239, Robolab 2.9.4 for elementary school and RobotC for the high have been selected as programming environments. Standard Lego Mindstorms NXT environment (G-NXT language) doesn't possess explicitly expressed mathematical apparatus, therefore we don't use it. A key point of approach is also using the difference equations of controlled system (plant) well corresponding to the discrete nature of computer based control instead of differential equations [2,3].

In this paper we demonstrate further extension of the approach by a number of examples.

2. EXAMPLE 1. CONTROL OF THE MOTOR.

The task (control goal) is to keep the motor in the given position $\alpha=45^\circ$ (fig. 1).



Fig. 1. Motor stabilizing in position 45° .

The solution based on a relay on-off regulator looks as follows (fig. 2). The students observe that it causes undesirable oscillations. Introduction of a three-position regulator almost doesn't refine system operation.

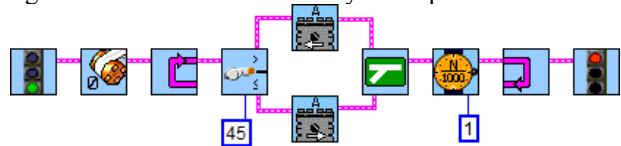


Fig. 2. Relay control of one motor.

```
task main()
{
    int alpha=45;
    nMotorEncoder[motorA]=0;
    while(true)
    {
        if(nMotorEncoder[motorA]>alpha)
            motor[motorA]=-100;
        else
            motor[motorA]=100;
        wait1Msec(1);
    }
}
```

Some simplification of the algorithm replacing branching by proportional feedback control may improve regulation performance significantly (fig. 3).

P-controller

At proportional regulation control action $u(t)$ normally is a function of a dynamic error – deviation $e(t)$ of adjustable value $x(t)$ from its preset value $x^*(t)$:

$$e(t) = x^*(t) - x(t)$$

$$u(t) = ke$$

The gain coefficient k defines regulator influence on system. As regulation will be further considered at regular intervals, passing to sampled data control, we adopt the following notations: $e = e_i = e(t_i) = e(t)$, $u = u(t)$, for t running over an interval $[t_{i-1}, t_i]$.

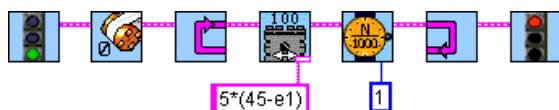


Fig. 3. The control algorithm of the motor based on a proportional regulator.

```
task main()
{
    int k=5, u, alpha=45;
    nMotorEncoder[motorA]=0;
    while(true)
    {
        u=k*(alpha-nMotorEncoder[motorA]);
        motor[motorA]=u;
        wait1Msec(1);
    }
}
```

Deducing control in the parallel task, we have an opportunity to install the motor in any position, simply setting parameter Alpha. It provides possibilities for regulation of the speed and sequence of positions of the motors. With usage of such simple method we have constructed the robot-drummer which has won the third place at WRO 2009. This trick is also applicable to robot manipulators which have acquired ability to fulfill exact sequences of movements. However, for the robot-decorator (fig. 4) which paints various spherical

objects, there is a need for slow control of the motor.



Fig. 4. The robot-decorator.

Then the PD-controller restricting speed of driving the motor without reducing accuracy has been used.

PD-controller

It is possible to present the proportional-differential regulator as the sum of two components. The differential component taking into account the speed of the error $(e_i - e_{i-1})/dt$, has the opposite sign with respect to the proportional one. Thus, for convenience of control of motors we will accept

$$e_i = x_i - x^*$$

$$p = k_p e_i$$

$$d = k_d (e_i - e_{i-1})$$

$$u = p + d$$

Strictly speaking, at speed sensing there should be a value dt , corresponding to duration of a time interval between measurements. However we import dt to coefficient kd since all intervals equal to 1 ms.

Another bright example of excellent performance of the PD-regulator is driving the robot along an irregular wall.

3. EXAMPLE 2. DRIVING ALONG A WALL.

Using the distance sensor, the robot should move along an curve wall at the given distance L (fig. 5). Let us start with the proportional controller.

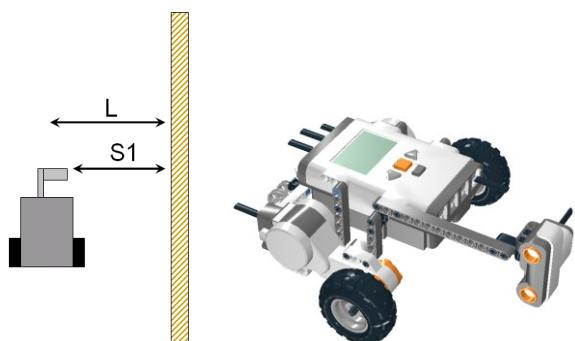


Fig. 5. The task of driving along a wall on distance L.

Denote S_1 the current distance to a wall returned by the sensor. Motors move at average rate 50, but any deviation from the given course creates correction action u as follows:

```
Motor[MotorB]=50+u;
Motor[MotorC]=50-u;
```

It is easy to evaluate the control action:

```
u=k*(SensorValue[s1]-L);
```

Thus, at $S1=L$ the robot doesn't change course and goes directly. In case of a deviation its course is adjusted. Typical values of the coefficient k for robot NXT of average size fluctuate from 1 to 10 depending on many factors. We suggest schoolchildren to adjust it by trial and error. Note that such a regulator works efficiently only at small deflection angles.

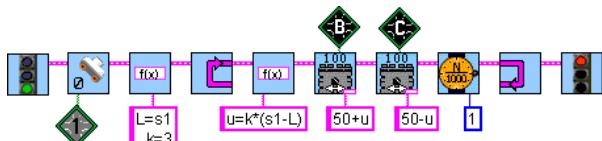


Fig. 6. Algorithm of driving along a wall with proportional regulator.

The same program in RobotC is as follows:

```
task main()
{
    float u, k=3;
    int L=SensorValue[S1];
    while(true)
    {
        u=k*(SensorValue[S1]-L);
        motor[motorB]=50+u;
        motor[motorC]=50-u;
        wait1Msec(1);
    }
}
```

In some situations P-regulator can destabilize the system (fig. 7, left). For example, if the robot is directed from the wall, but the distance to the wall is less than the desired one, then the motors get the command to turn even more strongly from the wall. Therefore contact with the wall can be lost (the distance sensor receives a reflected signal practically only from a orthogonal surface).

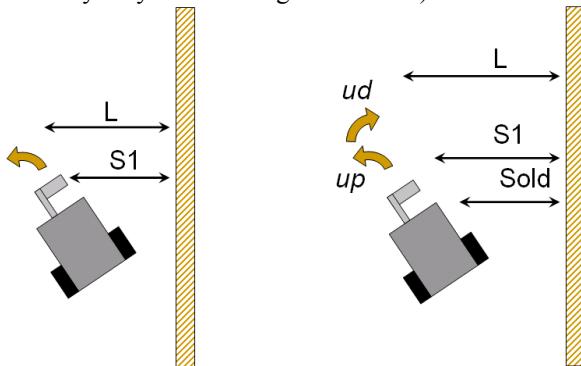


Fig. 7. A problem of a proportional regulator – loss of contact to a wall (at the left). The differential component (on the right) is necessary.

To avoid such situations we add a differential component taking into account the direction of the robot (fig. 7 on the right). In other words, the speed value will influence the control action. It is known that speed is $v = \Delta s / \Delta t$ where Δs is the change of the distance during time interval Δt . We define the differential component by the speed of a deviation of the robot from the given position: $d = k * (S1 - Sold) / \Delta t$, where $S1$ is the current distance to the wall, $Sold$ is the distance at the previous step. Since samplings are taken at regular intervals Δt it is possible to simplify the expression denoting $k_2 = k * \Delta t$, $d = k_2 * (S1 - Sold)$.

Thus, the PD-regulator consists of two terms:

$$u = p + d = k1 * (S1 - L) + k2 * (S1 - Sold)$$

It is possible to prove mathematically that for steady goal achievement the coefficient $k2$ should exceed $k1$.

The algorithm of driving along a wall by the PD-regulator as a whole looks as follows (fig. 8):

```
task main()
{
    float u, k1=2, k2=10;
    int Sold, L;
    Sold=L=SensorValue[S1];
    while(true)
    {
        u = k1*(SensorValue[S1]-L) +
            k2*(SensorValue[S1]-Sold);
        Sold=SensorValue[S1];
        motor[motorB]=50+u;
        motor[motorC]=50-u;
        wait1Msec(1);
    }
}
```

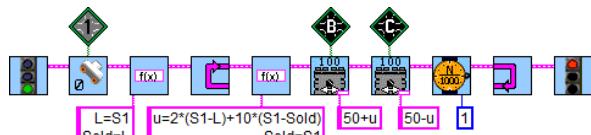


Fig. 8. Algorithm of driving along a wall based on PD-regulator.

4. EXAMPLE 3. FOLLOWING THE LINE.

The problem is to make the robot moving with maximum speed along the boundary between black and white. The solution by means of a relay on-off regulator is ineffective though clear to a beginning programmer: the robot is chattering along the line, significantly reducing the speed of one or the other motor (fig. 9). The boundary value between black and white for the light sensor is taken 45.

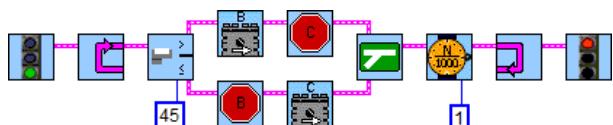


Fig. 9. Algorithm of driving along boundary between black and white by relay regulator.



Fig. 10. Driving along boundary between black and white.

It is the classical example taken from the built in electronic textbook Lego Mindstorms NXT Edu in which the similar solution is offered (fig. 10). It can be improved, having reduced a difference of speeds of motors, however it will reduce manoeuvrability of the robot at sharp turns.

Having convinced a pupil that the relay regulator works ineffectively, we suggest him/her to pass to the proportional one. It may seem strange that driving along the boundary between black and white can be implemented by the P-regulator since human eye can see only two states: black and white and relay regulator may seem more appropriate. But the robot sees differently, with no sharp boundary between those colors (fig. 11).

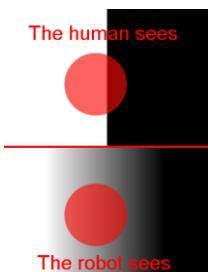


Fig. 11. A difference in perception of a human and a robot.

One may say that the robot is shortsighted and sees gradual change of the shades of gray. The reason that the light sensor catches the reflected light only from one photo cell, and presence of a segment of a black field just reduces cumulative illuminance. It helps us to create P-regulator looking as follows (fig. 12):

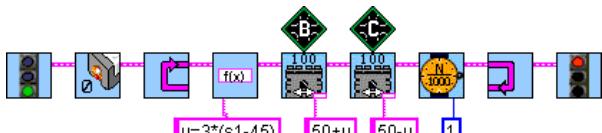


Fig. 12. Algorithm of driving along a black line based on the P-regulator

```
task main()
{
    float u, k=2.5, v=50;
    int grey=45;
    while(true)
    {

```

```
u=k*(SensorValue[s1]-grey);  
motor[MotorB]=v+u;  
motor[MotorC]=v-u;  
wait1Msec(1);
```

The children see that the standard cart equipped with such a controller moves quickly and precisely, as on the rails. But we need for more speed. Surprisingly, the problem of fast driving along the line may be better solved by means of the PD-regulator. (The solution is almost identical to algorithm of driving along a wall.) The effect of the PD-regulator can be seen only at the high speed when the robot starts to move off the line. To achieve such a speed, it is necessary to increase diameter of wheels and even to implement the raising transmission. While the speed of a normal Lego cart is about 40 cm per second, the fast Lego-robot with regulator can reach speed of 1 m/s. Strangely enough, leaders in these races are robots made on the basis of RCX.

5. EXAMPLE 3*. TWO SENSORS ON A LINE.

A particular interest has the algorithm of driving along a line with two light sensors allocated over two borders of it. The solution based on a relay four-positional regulator essentially loses to the solution on the P-regulator.

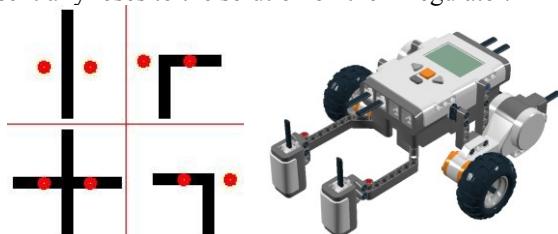


Fig. 13. Variants of layout of two light sensors over a black line.

As for signal processing of two sensors in the first case 4 states are considered (fig. 13), nested branching (is necessary to construct (fig. 14):

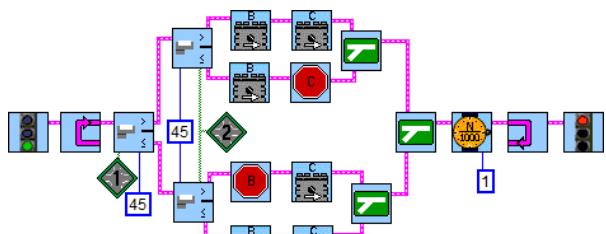


Fig. 14. Algorithm of driving along a black line with two sensors with a relay regulator

A solution in RobotC, based on constructions «if», is omitted for brevity. Instead we consider the elementary solution based on the P-regulator (fig. 15).

Considering that over an identical field sensors show identical values, we receive simple control algorithm which does not require even preliminary calibration.

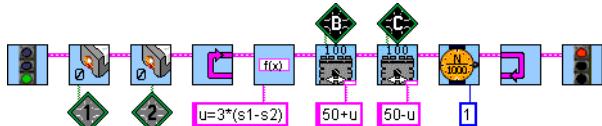


Fig. 15. Driving on a line on the P-regulator with two light sensors.

```
task main()
{
    float u, k=3, v=50;
    while(true)
    {
        u=k*(SensorValue[S1]-SensorValue[S2]);
        motor[motorB]=v+u;
        motor[motorC]=v-u;
        wait1Msec(1);
    }
}
```

And, at last, algorithm for fans of an extreme – a high-speed line-driver can be designed (fig. 16). The differential component is added, allowing to retain a line between sensors even at high speed. Thus for error compensating the value err , the threshold for the difference of the sensors readings is entered.

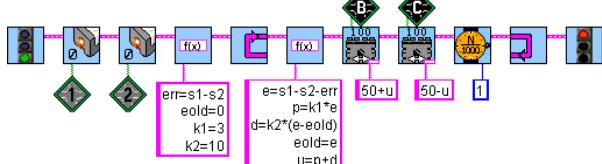


Fig. 16. Driving on a line on the PD-regulator.

```
task main()
{
    float u, p, d, k1=3, k2=10, v=50;
    int Sold=0, e, err=SensorValue[S1]-
        SensorValue[S2];
    while(true)
    {
        e=SensorValue[S1]-SensorValue[S2]-err;
        u=k1*e+k2*(e-eold);
        eold=e;
        motor[motorB]=v+u;
        motor[motorC]=v-u;
        wait1Msec(1);
    }
}
```

For a new robot selection of suitable coefficients will occupy some time. Generally speaking, exact tuning of regulator coefficients is a difficult engineering task, but at the first stage schoolchildren select them by trial and error.

Perhaps, following on a line is the most useful and a vivid example of the preference of the control theory approach over normal programming. To be more precise, we demonstrate their successful symbiosis.

5. EXAMPLE 4. THE BALANCING ROBOT.

The problem to keep the robot standing vertically on two wheels is exciting for each schoolboy or schoolgirl.

However, the examples of its solutions are difficult and involve high mathematics elements. Here there is an interesting task for the teacher: how to explain the principle of the PID-regulator operation to a pupil without loading him/her with superfluous knowledge? A possible solution is described below.

6. PID- REGULATOR

As follows from the title, this regulator consists of the total of three components graphically represented in the following simplistic form (fig.17):

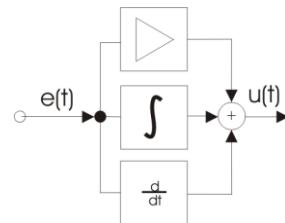


Fig. 17. The PID-regulator circuit.

$$u(t) = p + i + d = k_p \cdot e(t) + k_i \cdot \int_0^t e(\tau) d\tau + k_d \cdot \frac{de}{dt},$$

where the input value for a regulator input is the dynamic error $e(t)$, while its output is the correction action $u(t)$. The proportional component represented on the circuit by a triangle is responsible for system positioning in the given state. Its large value can cause overshoot with the subsequent self-excited oscillations.

The integral component accumulates the negative experience (adds errors) and works out compensating influence. At minimum deviations the proportional component "weakens" also integral, at the expense of the fast magnification of summation, helps to "hold on" regulated value to the necessary one.

The differential component tracks speed of state change of system and hinders with possible overshoot.

Let us consider the integral component. It is defined in a dynamic manner, being added to the previous value.

$$i = i + k_i \cdot e(t) \cdot dt$$

The physical sense of value i is that it is proportional to duration of a staying the system in an error state. As the coefficient k_i is put outside the brackets, it is possible to speak about the value i as of the total duration of errors.

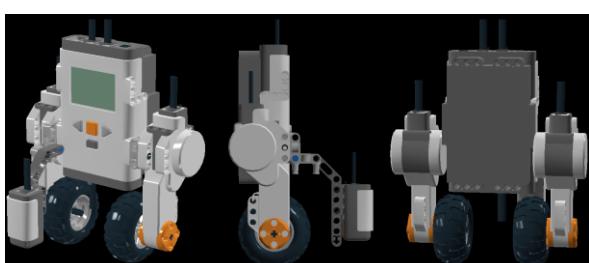


Fig. 18. The balancing robot-segway.

The PID-regulator is applied to the robot balancing by means of the light sensor, directed downwards (fig. 18). At a deviation of the robot from vertical position the output of the sensor is presented in format RAW that raises accuracy of measurements approximately in 10 times.

In fig. 19 the algorithm in Robolab is presented. Its largest part is occupied with initialization of variables. For increase of accuracy not only the data c the sensor is read out in format RAW, but the majority of variables is declared in a real format float. Actually the PID-ALGORITHM is in a cycle.

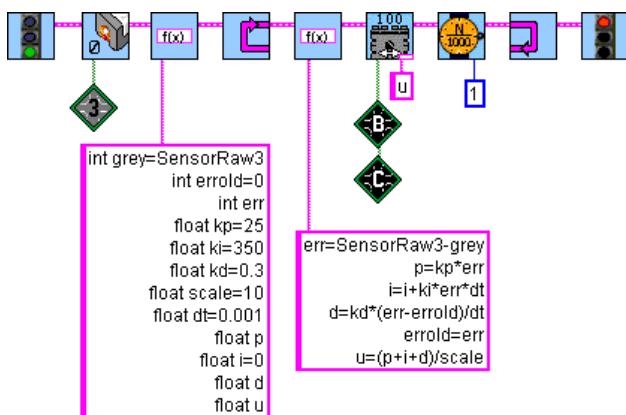


Fig. 19. The algorithm of balancing robot based on the PID-regulator.

Following tradition of driving on a line, as a preset value x^* it is used a variable grey – average indications of the light sensor in balance position. The new parameter scale sets correction action scaling. As a matter of fact, it is the weakening coefficient as value produced by regulator is too high for NXT motors. It would be possible to import it of inside already available coefficients, but for RobotC this parameter will be another, and the same coefficients. The similar example on RobotC differs for a variety of cases a little. At first, with an insertion of this environment above approximately in 1,4 times, than at Robolab, therefore coefficient scale it is necessary to increase high-speed performance NXT. Secondly, RAW-values are transferred in the correct order and it is required to install reverse of motors or simply to submit the negative correction action.

```

task main()
{
    int grey=SensorRaw[S3];
    int err, errold=0;
    float kp=25, ki=350, kd=0.3;
    float scale=14;
    float dt=0.001;
    float p, i=0, d, u;
    while (true)
    {
        err= grey-SensorRaw[S3];
        p=kp*err;
        i=i+ki*err*dt;
        d=kd*(err-errold)/dt;
        errold=err;
        u=(p+i+d)/scale;
        motor[motorB]=u;
        motor[motorC]=-u;
        wait1Msec(1);
    }
}
    
```

```

    i=i+ki*err*dt;
    d=kd*(err-errold)/dt;
    errold=err;
    u=(p+i+d)/scale;
    motor[motorB]=u;
    motor[motorC]=-u;
    wait1Msec(1);
}
    
```

Having mastered this simple algorithm, schoolchildren get prepared for solving more challenging tasks: stabilizing by means of the gyroscopic sensor, driving the balancing robot along a line with two light sensors, control via Bluetooth, etc. In autumn of 2011 our pupils participated in the first Russian competitions of balancing robots and a pupil of our lyceum has become the winner.



Fig. 20. Balancing robots is studied at junior high school.

CONCLUSIONS

Summing up, we conclude that cooperation of the university with the lyceum was very fruitful. Application of elements of the automatic control theory has changed character of robot control dramatically. Everywhere where it makes sense, schoolchildren replace conventional algorithms with regulators. Using mathematics for improving control performance appears to be not only effective, but also extremely interesting.

REFERENCES

- [1] S.A. Filippov, A.L. Fradkov. Cyber-Physical Laboratory Based on LEGO Mindstorms NXT - First Steps. 3rd IEEE Multiconference on Systems and Control. St.Petersburg, 8-10 July, 2009, pp.1236-1241.
- [2] S.A. Filippov. «Robotics for children and parents» Edited by A.L. Fradkov, 2nd Edition, St.Petersburg: Nauka, 2011.
- [3] Filippov S.A., Fradkov A.L., Andrievsky B. Teaching of robotics and control jointly in the University and the high school based on LEGO Mindstorms NXT. Proc. 18th IFAC World Congress on Autom. Control. Milan, 2011, pp.9824-9829.