Monitoring of the movement of the mobile robot

Tomáš Surovčík, Richard Balogh, Ladislav Jurišica

Abstract
The study focus on the problems of the acquiring and designing the image of the robotics. The task is to read out of the image of the entrance equipment and to analyze the process, on which we can allocate the location of the mobile robot of an unfamiliar form in the familiar environment. The result is the position of the robot in the model environment with the possibility to remit this position to the visualization program. At the some time it is possible to save the robot's movement in video format and to analyze it again.

Key words: robotics, visual system, monitoring of the location, OpenCV

Entry

The processing of the image has a wide application. It regards the identity of the objects, its allocation, its orientation etc. According to the recognized picture is realized the exit checking of the car's chassis at the final part of the producing process is realized or checking of the position and rotation of electro components on the circuit board. The processing of the picture is important also in robotic competitions. In the processing process of the behavior of the robot in the environment – in this case in the labyrinth, the camera exit signal is monitoring via the personal computer with the help of input of OS Windows. The labyrinth consist of the net (max 16 x 16) of the basic squares in the dimension the high of 0,05 m and the thickness of 0,0012 m (+/-5%) [16]. The paths are 0,168 m wide. The labyrinth is closed by the wall. The walls of the labyrinth are white on the side; the ceiling is red. The bottom of the labyrinth is covered by the black color. The surface of the walls and the ceiling have to reflect the infrared light and the surface of the bottom have to absorb it. The entrance of the labyrinth is going to be at the one of the four corners. The corners of the basic square are build up form so-called grating dots. Minimum one wall exits from each grating dot and together form the labyrinth. The form of the robot is not known, we know just its maximum size and the starting point.

2. Analysis of existing solutions

By the scripting and the elaboration of the image it is necessary to have enough time we need to calculate the location of the robot and therefore we need to seek the hard wear solution which is to loaded with the minimum number of the possibilities of the mistakes, which have to be cleaned up by the means of soft wear.

2.1 Basic definition

The math model of the picture could be an analogue function f(i,j) of two arguments, axes of the plain [12]. Frequently the value of the function is the loom (intensity), because it is the constant, which is measured mostly by the sensors for the entry of the pictures. The loom is the constant, which is perceived by the man. The environment, in which the man moves, is three dimensional. Two dimensional picture's function is the result of perspective depiction of the part 3D of the environment.

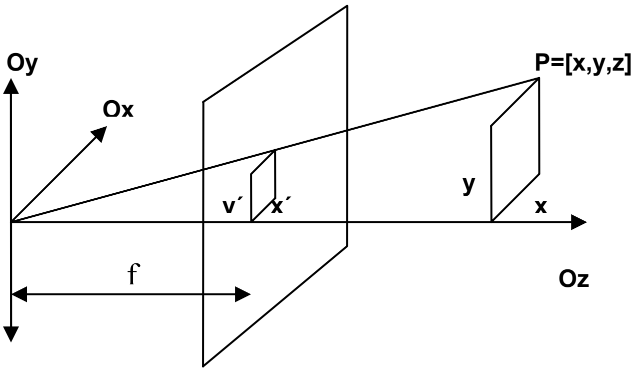


Fig. 2.1 **The flush of the picture [12]**

The geometry of the perspective depiction is shown on the figure 2.1. As *x, y, z*, are axes of the dots in 3D scene and *f* is the focal distance. The axes of the dots after the perspective depiction are

$$x' = \frac{xf}{z}; \quad y' = \frac{yf}{z} \qquad (2.1)$$

By the perspective depiction from 3D environment to the image 2D a lot of information is lost. The image could be monochromatic and it means, that the image is represented by one function, which is f(i,j) = f(x´,y´). In the more difficult case we work with color (multi spectral) picture. Each couple of the axes (i,j) belongs a vector of the values, e.g. the level of the loom falls to one color parts of the image.

2.2 The defect camera lens

The deficiency of the entry – the enlarging at the vicinity of the optic axis of the lens differs from the size at the edge of the lens [8]. The rays on the edge shatter more than the rays, which are closer to the middle of the lens. Each ring zone of the lens has its own point of intersection of the rays and offers the image the certain distance from the lens. The particular dots of the depict item seam to be like small rings. Their size depends on the curving of the lens.

The colored aberration – the different shattering of the rays for different colors has an effect the color deficiency of the lens. The lens have not the same focus for different colors, see picture 2.2.

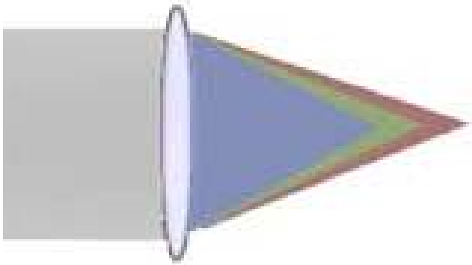


Fig. 2.2 **Colored aberration**

E.g. the red ray has the different refrangibility than a blue one, and it makes that the color of the image is not clear. Mostly refrangibility violet rays, the violet image is the smallest one, because its center is the nearest from the lens. The rays from the other side of the colored spectrum – red ones, differ the less. These rays offer the largest picture, because their distance from the centre is the biggest.

Distort of image - if the lens has the balls curve of the surface, in these case differs the enlargement in the vicinity of the optic ax from that at the edge of the field of view. Than we do not find the square as a square, but it has the sides curved, like you can see at the Fig 2.3.

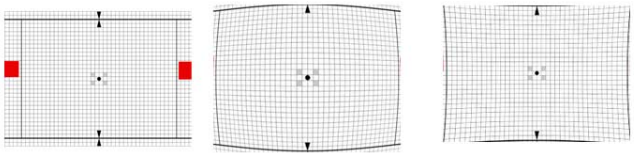
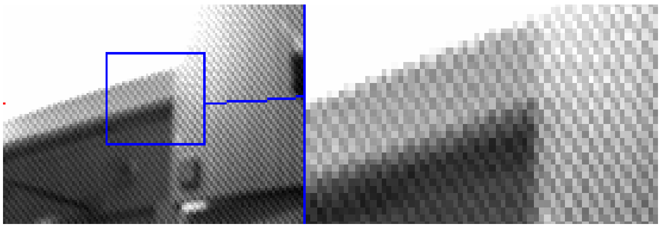


Fig. 2.3 **Distort of image, the image on the right is the original one, in the middle is scanned by the camera with the kegs distortion, the image on the left is scanned by the camera which suffers from the pillows distortion**

2.3 Defect of the transmission ways

The picture can be infected by various defect called "noise", as we can see at the picture. The noise can originate by scanning of the image, by their treatment or by the transmission of the signal. The noise depends on the image or not. The noise is described mostly by the expectation features.



Obr. 2.4 The image infected by the rustle

3. Pretreatment of the signal

3.1 Digitalization of the image

The entry of the facsimile signal is mostly analogous signal. In order we would be able to treat the image, we need through the digitalization process acquire the digital equivalent of the analogous signal. The digitization is based on the sampling of the image in the matrix *MxN* dots and on the quantization of the connected level of the loom of each sample in K intervals (the standard of quantization). The smaller is one of the sample (larger M and N) and finest quantization, the better is approximated the original connected signal [12].

3.2 Geometric transformation

In between correction of the geometric aberration and geometric transformation, like rotation or shifting, does not exist the basic difference. The same algorithms are applied.

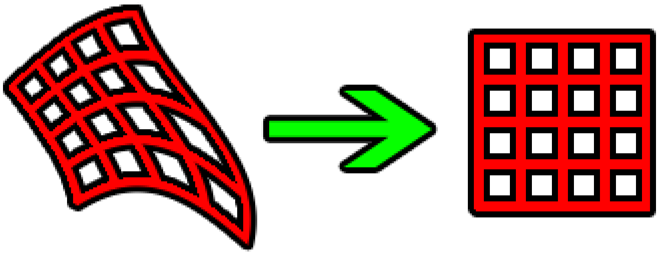


Fig. 3.1 **Transformation of the image**

Geometric transformation T_g of the image is the vector function, which transforms a dot from the diameter *(x,y)* to the diameter *(x´,y´)*. Transformation is defined as:

$$x' = T_x(x, y) \qquad y' = T_y(x, y) \qquad (3.1)$$

Transformation relation (3.1) can be known or in advance, like e.g. by the rotation, shifting or enlarging. By the looking for transformation relation we use several dots that match in both images the same object. Geometric transformation relation of two steps. The first step is to find to the dot in the entry image with the discreet axes the corelated dot in the final image. By the exit dot we have to calculate with the linear axes, because the final position of the dot mostly does not match to the numeral raster [12]. How to define the axes of the dot at the image, which is the result of the geometric transformation, is shown in the common figure the relation (3.2). Transformer relation of the axes is usually approximated by the polyon of m grade:

$$x' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} a_{rk} x^r y^k \qquad y' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} b_{rk} x^r y^k \qquad (3.2)$$

Transformation is linear in regard to the constants a_{rk}, b_{rk}. The advantage is that by the means of the method of the smallest quarters we are able to define constants of the transformation a_{rk}, b_{rk} on the base known assembly of each other corresponded the couples at the entry and the final image [12]. The second step is the level of the loom that we allocate to the new dot at the final image. The values of the loom in the dots, we need, with the numeral axes we interpret on the base of the known levels of the loom of the surroundings area. The most simple is the interpretation of the loom within the method of the nearest neighbors, which allocate to the dots in the discreet network of the final image the loom of the nearest sample *(x´,y´)*. The further interplolar methods are linear interpolation and cubic interpolation.

Moving

Let us suppose, that the moving is not needed, because for the quicker calculation will be simplest to pick up from the picture only the labyrinth and to calculate in this area only by the eventual shifting. The shifting means the relocation of the object from one position to another. Transformation of the shifting is defined by the vector of the shifting p = (x´-x, y´-y), where (x, y) are axes of the object on the first position and (x´, y´) are axes of this object on the second position. Transformer matrix of the shifting is [17]:

$$x' = x + t_x \qquad y' = y + t_y \qquad T = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 0 \end{pmatrix} \qquad (3.3)$$

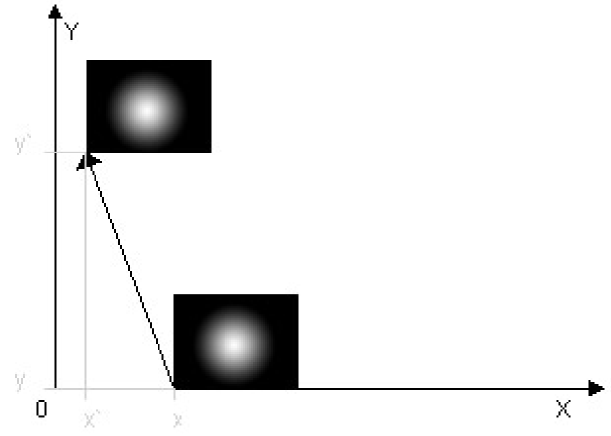


Fig.3.2 **Shifting**

Rotation

Rotation is the movement of the object along the edge of the circle with the known diameter.

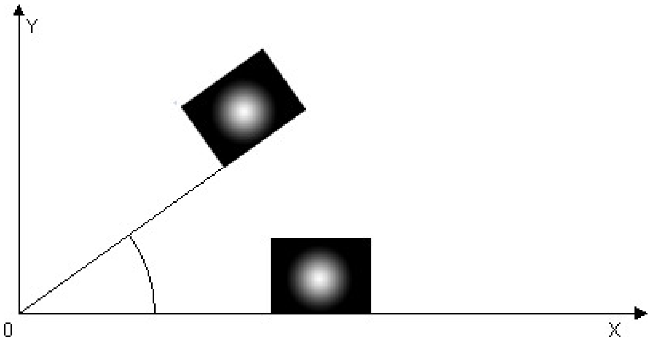


Fig. 3.3. **Rotation around the angle**

By the rotation of the dot *P[x, y]* around the centre of the axis system *O=[x₀,y₀]* around the angle alfa we get a new dot P´ with the axes [17]:

$x' = (x-x_0)*\cos(\alpha) - (y-y_0)*\sin(\alpha)$

$y' = (x-x_0)*\sin(\alpha) + (y-y_0)*\cos(\alpha)$ (3.4)

Matrix of the transformation of the rotation [17]

$$A_0 = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & -x_0\cos(\alpha)+y_0\sin(\alpha)+x_0 \\ -\sin(\alpha) & \cos(\alpha) & -x_0\sin(\alpha)+y_0\cos(\alpha)+y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

4 Solution of the task

To find and to recognize the robot we need to use algorithm based on the recognition of the level of the loom of the single pictorial elements. The algorithm compares the image of the background and the actual snap taken by the camera. On the base of the defined differences will be recognized the place where the robot can be found. To accelerate the recognition and the modification of the operation we use the searching only in the places, which the robot can reach from the actual position in the labyrinth.

This course grants us not only the acceleration of the solution, but prevents to look up the “wrong place”, where the robot could not reach from the actual position (e.g. like the recognition of the hand of the competitors or the reflection of the camera on the surface of the labyrinth.

The robot can reach to the four different directions, it means up, to the left, to the right and down, if it does not prevent the wall of the labyrinth.

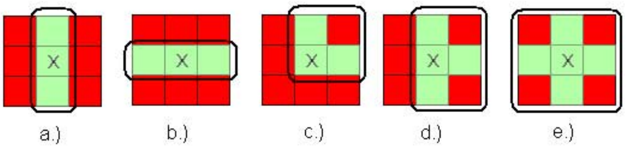


Fig. 4.1 **Supposed location of the robot, green color identify the possible movement of the robot.**
a) robot can move vertical,
b) robot can move only horizontal,
c) the robot is at the corner,
d) robot is on the cross road of „T“ type,
e) robot is on the cross road of „+“ type

The position of the robot we can identify like the center of the most distant dots that would be find out according the above mentioned algorithm. In this way we determine the center for x and for y axes and this centre we consider as a centre of the robot, the sample you can see on the figure 4.2. Consequently we can determine the new position of the robot and to move the exes further to the next processing or depiction.

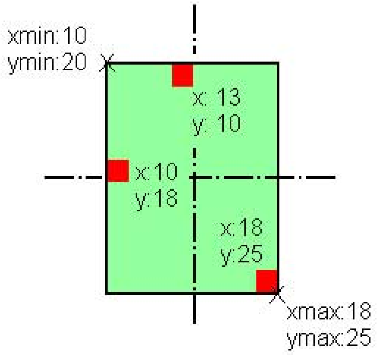


Fig. 4.2 **Identification of the center of robot**

After the algorithm is applied, it is possible to scan the picture and to assign it as a background. The background can stay static or dynamic. If there is the dynamic background assigned, the image of the background will be changed as the time will be going on, on the base of the image taken by camera, but in such a way, that on the image of the background the single levels of the loom will be changed only by a certain part of the level of the loom and in this way will get nearer to the actual picture.

There is the possibility in the some way to calculate or not the spot where robot can be present. In this way we eliminate the slowly changes of the luminous source, e.g. the influence of the sun. The demonstration you can see at the figure 4.3., where 6 snaps of the labyrinth are taken during the redrawing. Each redrawing took approximately 1 minute.

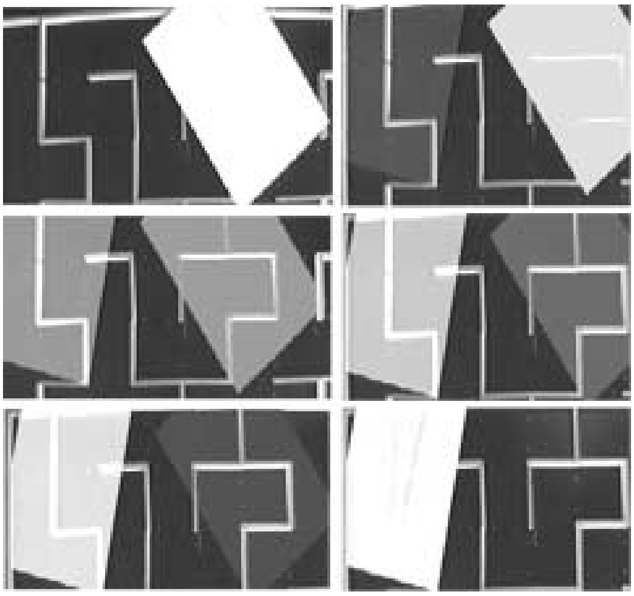


Fig. 4.3 **Demonstration of the redrawing the labyrinth. The time order from the left to the right**

4.2 Influence of the fault value

In the case we will use for the recognition of the robot movement the algorithm mentioned above, it can happen that the position of the robot could be not exact. The algorithm seeks the most distance dots of the labyrinth, which were classified as the different from the pixels of the labyrinth. It may happen that the center of the robot, we suggest, would be not the actual center of the robot. This case can happen when the robot's features are expressive just on one side. It also can happen, that the final dot would not be the same on the robot, e.g. in case when robot has the glossy surface.

The algorithm for the recognition reflects the possible movement of the robot and if the person, who moves the robot, comes from the direction where the wall prevents to follow the movement of the hand of this person. But in the case, the hand moves from the direction, where the robot can move, it is possible, that this hand will be considered as the movement of the robot, and will be followed.

The flash is not foreseen event, which can be removed in the case of the flash of the camera. If the flash leaves its reflection in the labyrinth and at the moment, when the snap is taken, the mentioned algorithm can evaluate the position of the robot incorrect. The mistake can be eliminated, when we increase the number of snaps during the time, the robot needs to move through one sell, it means to increase the number of the snaps per one second or to slow the speed of the robot's movement.

5. Realization

5.1 Program environs

Program is created in the program Borland C++, because the use of the libraries OpenCV call for program languages C or C++.

5.2 Event in Windows

The event is information that something happened in the system. After realization of any action Windows create event, which contains all the essential information about what and where it happened. Windows send the event to the application, whose the event concerns. Each of the running application has its own list of the events. The latest event is included at the end of this line. Inside of the application so-called "loop of the event" runs and its task is just: to take the event out of this line (the organized configuration) and to pass it for the execution. It is obvious, that the execution of the event has to carry out the special procedure (the method).

This procedure is common called "the window procedure". The window procedure that exists in the application for each window, the event "handles", it means that it reacts on this event in convenient manner. To tell it simplest, Borland C++ encases news for us, which we can get from Windows, in the shape of the obvious and workable events.

The key areas are four [7]:

- A certain place (here the action is taken – the action originates, so it comes to the generation of same event)
- The line of event of the application (the event, generated in the first step, is included at the end of the line of application, which concern to)
- The loop of the event (from the top of the line of event given application is the first waiting event taken and hand over to the new procedure)
- Window procedure (window procedure makes over the action – in our case it is the drawing of the picture – frame video, etc.)

The line of event – each running application contents the line of event, there gathered all events in between the moment of its origination till the moment of its registration. We have to notice that there are two possibilities of the behavior of the line of the event in the case of receiving the some events which already is registered.

The first alternative is, that the new event is accepted at the end of the line, so there are two events. The example is e.g. event about the movement of the mouse: we move the mouse the first time (the event A), the event B will be filled at the end of the line also in the case, there still is the event A.

The second alternative is in the fusing of both messages into one event. In this way behave the message about the finishing of the certain time interval (determined by timer), so e.g. it is not possible to find out, how many times the timer "ticked". The loop of the events is the mechanism, which guarantees the drawing of the events out of the line and its handing over to the window procedure for further proceeding. To tell it in a different way - the loop of the events always post the event from the top of the line to the relevant window of the application. If the line of the events is empty, the system Windows enables the execution to the further applications. Window procedure – in the application can be more windows and each of them has its own window procedure, which treats the event (corresponds to them), which belong to the certain window. The window procedure may return to the system some value in the form of the reaction to the receiving and treatment of the events.

5.3 Threads

Threads are important for the traditional multi-tasking operation systems, like Windows, where the particular processes work independent and have independent addresses and communicated only via IPC (inter-process communication). IPC is divided into more method like: the sending of the messages (mentioned above), synchronization, common register or RPC.

The switching among the threads processes runs quicker than among the processes. The threads programming is popular and renders to share the same source for the various processes or allows to the one process to carry out more operation at the some time. The thread brought some problems (e.g. two threads try to serve the same file at the some time). The duration, which is allocated to the threads, depends on its priority.

The priority of each thread is given according to the duration of the processing and the time when can be all done. We have never known if the action of the thread is interrupted by the system and we do not know how much work was done. In the case we use more threads, which transfers each other the entry information, it is necessary these threads to synchronize.

MultiThreading means more than the threads processing of the instruction in a core of processor. The real enlargement of the efficiency of the application by the using multithreading is from 5 to 30%. For comparison in the case of the doucore processor it is from 30 to 80%. It exists two different multithreadings – hardware (single chip multiprocesoring CMP) and software (simultaneous multithreading – SMT).

5.4 OpenCV

OpenCV has incorporated wide range of tools for the processing of the image. It is compatible with Intel Image Processing Library (IPL), where the low level processing of image is implemented.

OpenCV has incorporated algorithms for the calibration techniques (Camere Calibration), detection of the edges (feature), detection of ways (Optical Flow), the form analysis (Geometry, Vontour processing) etc. Unthinkable feature of the library is its functionality, quality and capacity. Algorithms are based on the high flexible data structure. More then a half of the functions were optimalized for Intel architecture (Pentium MMX, Pentium Pro, Pentium 3 a 4) [1].

OpenCV runs on the personal computer based on Intel architecture. The libraries are written for the user of the applications in program language C or C++. OpenCV (Open Source Computer Vison) is the library of the program functions focused on the processing of the image in the real time. This library was used also by the winners project named Stanford racing during the race DARPA Grand Challenge [2], where the task of the car was to run the distance of 211 km (132 miles) without the human help.

5.5 Construction

The first version of the program was done in the program environs Delphi 7. The manner of the recognition was the same as in the application of the second version. To scan the picture the library avicap32.dll was used and to save the picture the hard disc of the computer was used.

By the construction of the second version of the program the program environs C++ Builder 6 was used and for the saving and proceeding of the picture the library OpenCV was used.

6. Implementation

6.1 Possibility of the change the actual position

Because there is the possibility, the robot runs away from the investigated area or it will be placed at the start or at another place of the labyrinth, program can apply the possibility of the change of the actual position without need to stop or start again the process.

6.2 Possibility to choose the source of signal

Because it is necessary to extract the source of the signal from which should be worked the image, it was necessary to adjust the library OpenCV and the library consequently was recompiled in the program language Visual C++. The Visual Studio 2005 was used for recompile. As the next it was necessary to import libraries *.lib for C++ Builder 6.

Thanks to this change the selection of the signal source was allowed and at the some time the possibility of the selection of the separation was delivered.

6.3 Rotation

The necessary component of the program is soft wear the rotation of the image, because the camera is firmly connected to the environs and the scan of the labyrinth could take long time and might be uncomfortable. For these reason the function of the library OpenCV can be used. The parameter of the rotation, the angle and the center of the rotation, can be set up by the user in the program “Find and Localize”.

Conclusion

In our case we choose interaction approach to the problem. In the first step we made the program alive and measured the flat position of the robot. In the further steps we enlarged the program, which was designed for the measurement of the flat form of the robot’s position, by the scanning of the form of the labyrinth by the camera and by a possibility to attach the label to the robot. Because of such an attachment we increased the accuracy of the measurement of the robot’s position.

Acknowledgment

The work was done thanks the grant VEGA MŠ SR.

List of the applied literature

[1] OpenCV http://opencvlibrary.sourceforge.net/

[2] Grand challenge 2005 http://www.grandchallenge.org/

[3] Discussion on Open Source Computer Vision Library http://groups.yahoo.com/group/OpenCV

[4] Programming computer vision applications http://www.site.uottawa.ca/~laganier/tutorial/opencv+directshow/cvision.htm

[5] Use of OpenCV v .NET http://www.cs.ru.ac.za/research/groups/SharperCV/

[6] na Open Source Computer Vision Library http://www.intel.com/technology/computing/opencv/

[7] http://msdn1.microsoft.com/en-us/default.aspx

[8] http://www.diginews.sk/

[9] http://www.inzine.sk/

[10] Lesson on PC graphic on internet http://www.francis.unitra.sk/users/petko/files/pg/vyucba/

[11] Říčný, V., Kratochvíl, T. TV technique Science Brno 2002

[12] Hlaváč, V., Sedláček, M. Elaboration od signals and pictures 1.Vyd. ČVUT, 2001

[13] Doc. RNDr. Pavel TOMÁNEK, CSc, Optics, ČVUT

[14] Borland Software Corporation, Help for C++ Builder

[15] Miroslav Sádecký, Camera Monitoring Systems, Žilina University, Electro technique Faculty

[16] ISTROBOT www.robotika.sk

[17] **Interaktív Book of the Elaboration of the Picture** http://dip.sccg.sk/

Ing. Tomáš Surovčík
Ing. Richard Balogh
Prof. Ing. Ladislav Jurišica,PhD.

Slovak Technique University
Faculty of Electrical Engineering and Informatics
Ústav riadenia a priemyselnej informatiky
Ilkovičova 3
812 19 Bratislava 1
E-mail: tomas.surovcik@stuba.sk
ladislav.jurisica@stuba.sk