PATTERN RECOGNITION WITH RASPBERRY PI PLATFORM

Jozef Bošanský [1], *Michal Blaho* [1,2], *Leo Mrafko* [1], *Tatiana Mudráková* [1,2]

[1]Institute of Robotics and Cybernetics

Faculty of Electrical Engineering and Information Technology

Slovak University of Technology

Ilkovičova 3, 812 19 Bratislava, Slovak Republic

[2]HUMUSOFT, s.r.o.

Cabanova 13/D, 841 02 Bratislava, Slovak Republic

Abstract

Raspberry Pi is a credit card-sized microcomputer, which was created for the purpose of education in computer science and programming. This device simply connects the monitor, keyboard and mouse and afterwards the device is fully functional as a desktop computer or notebook. The MathWorks Company supports such device using Support Packages in MATLAB and Simulink environment. This article discusses the Raspberry Pi platform utilization with camera to recognize a simple color pattern.

1 Introduction

The development of open source software and the current trend of minimizing the production cost of microchips enable the rise to all sorts of microcomputers. This development also had a positive impact on embedded systems. An embedded system is a single-purpose system, which can control a mechanical or electrical system with the real-time requirements. One of the first platforms comprising embedded system and created a big boom is a microcomputer Raspberry Pi. The following platforms are platforms like Arduino, BeagleBone or Banana Pi. Each of these platforms is nowadays very popular thanks to its-friendly prices, the ease of usage and also the usability in various fields [1-3]. Platform Raspberry Pi has been used in the several interesting projects. Iridis-Pi is a super computer made by English professor Simon Cox and his team. When Simon Cox met Raspberry Pi, he was so much fascinated that he decided to produce the super computer using multiple Raspberry Pi. Iridis-Pi consists of a 64-Roch Raspberry Pi and from over 1,000 Lego bricks [4]. Dave Akerman made using Raspberry Pi, web cameras, GPS navigations and a facility for live broadcasted video (balloon filled with hydrogen) and sent it to 40-ers kilometers. Not only is to get the Raspberry Pi in such height an amazing success, but this attempt is also even considered as the highest live pictures done in the terms of amateur experiments [4]. This article discusses the Raspberry Pi platform utilization with camera to recognize a simple color pattern. The pattern may be used for example to define the consequent moving equipment.

2 Raspberry Pi

Raspberry Pi is a credit card-sized microcomputer which was created for the purpose of education in computer science and programming, but thanks to its versatility has spread to other areas. With its low price, undemanding requirements and high utilization it has become the one of the most popular platforms of its kind. Raspberry Pi is a platform being developed by the British company Raspberry Pi Foundation since 2012. In this article model B was used. The platform has a system chip (SoC) BCM2835, which is used for example in mobile phones because it is inexpensive, powerful and consumed a not much of energy. SoC is in the contrast to the computer architecture because it has an integrated processing unit (CPU), a graphics processing unit (GPU), and also memory. BCM2835 includes: ARM1176JZF-S processor running at 700 MHz, 512 MB RAM and graphics card VideoCore IV. [5]

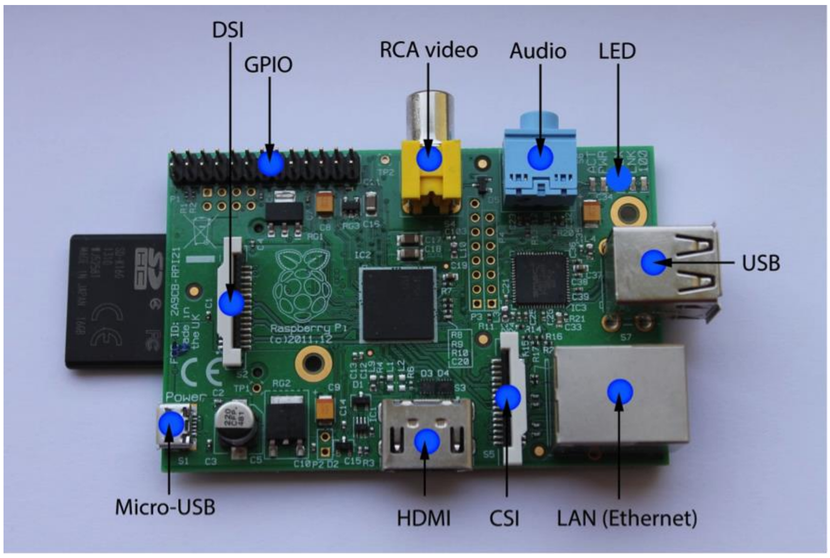

Figure 1: BeagleBoard xM rev. C

A several connectors can be found on the platform. There are USB ports that can be used to connect USB devices such as mouse and keyboard, but it can used also USB hub that can connects even more devices. The platform also includes a micro USB port for the power connection, so it cannot be used for anything else. For connecting the LCD display or TV, there are two possibilities – to use HDMI or RCA video. On the platform there are two rows of pins that give the device great power, because thanks to them a large number of sensors, transducers or devices can be connected. From among of the 26-ers pins is 8 GPIO pins type input/output, followed by two for UART, two for I2C communication and five for SPI communication. Finally, there are two for 3.3 V, two for 5V and the last five for the ground [5]. The advantage of the device is its affordability, and depend the model and ranges to $ 40.

2.1 MATLAB support

Raspberry Pi is compatible with MATLAB and Simulink environment, so there is the possibility to create own programs and start Simulink model directly on the device. With the MATLAB® Support Package for Raspberry Pi hardware, you can access peripheral devices through the Raspberry Pi. This support allows you to acquire data from sensors and imaging devices connected to the Raspberry Pi. Specifically, a library of MATLAB functions is provided for the following Raspberry Pi add-ons and interfaces. Students can use MATLAB to acquire data from connected devices, and afterwards they can analyze and visualize their data in MATLAB. When connected to MATLAB and Simulink products, Raspberry Pi can help hobbyists and students understand concepts and workflows for designing an embedded system, without using hand programming. At first it needs to be started by using Simulink to create algorithms for audio processing, computer vision, and robotics applications. Then it can be applied industry-proven techniques for Model-Based Design to verify that your algorithms work during simulation. Finally, the algorithms are implemented as a standalone applications on a Raspberry Pi [6,7].

To connect the Raspberry Pi with MATLAB Simulink it is necessary to install the support package - a package of support for Raspberry Pi device. After passing all steps of installation and until the termination, all needed third-party development tools will be installed in the computer, as Simulink block library: Simulink Support Package for Raspberry Pi Hardware and Examples. It is necessary to install the firmware to the Raspberry Pi for the correct cooperation with MATLAB Simulink. The installer writes the firmware image to the memory card with the login details to an environment that will run on the device. After inserting a memory card into the board, the support package installer applies the settings that were selected. In the Simulink environment pre-built units can be used, and are shown in the following figure.

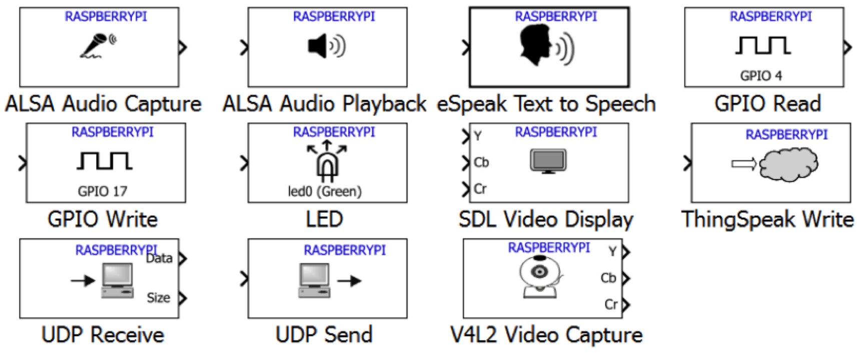

Figure 2: BeagleBoard Library

3 Image Processing

This section will be focused on designing the custom algorithm for image processing using the Raspberry Pi. Various forms of image processing from day to day still more and more gets to the foreground and finds their application in different agenda - from the detection of specific objects via motion detection, environment mapping, navigation, till the application of facial recognition. In this paper the algorithm for detecting specific shapes with the use of different colors was designed. The aim of this algorithm is the navigation and the control of flying robot (drone) movement in the space with the help of colored markers that define the different types of movement. Raspberry Pi together with the camera should be placed on the lower part of the drone and the camera should scan the ground. Whenever some image will be captured, the direction is evaluated and afterwards is changed. This includes seeking autonomy for the robot movement in space. For image processing, the camera was attached to the device, as shows the following figure.

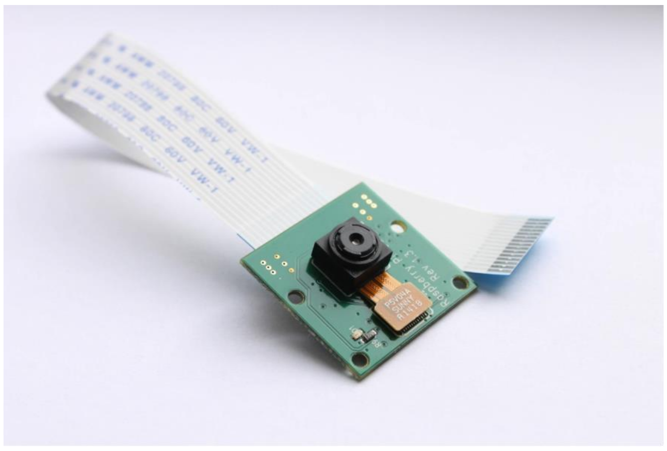

Figure 3: Raspberry Pi Camera Rev 1.3

3.1 Markers

For the purposes of image detection, images in the shape of squares were formed. Each of them is composed of the other four smaller squares. In each figure, one of the squares is always green. Other squares are different in color and from the other pictures differ in colors positions. For the purpose of this work RGB model was used. RGB is an additive color model consisting of three main components, namely red, blue and green. Each color can be expressed as a combination of the three components with suitable intensity. The human eye has the best sensitivity on these three colors. Such color combinations are based on the additive, counting model, meaning adding color to another color. Each of these three components get values from the range <0,255> or <0, FF> in hexadecimal. Whereby the individual components have a higher value, the resulting color is lighter. Black has in this model, the value of 0-0-0 and with gradually increasing brightness at different positions the value gets up to 255-255-255 representing the white color.

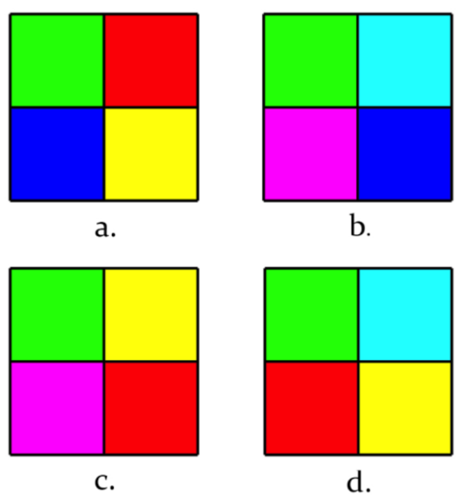

Figure 4: Patterns for movement recognition: (a.) movement to the right; (b.) movement to the back; (c.) movement to the left; (d.) movement forward

3.2 Algorithm

Before creating the algorithm the basic properties of blocks have been validated to work with images in a simple scheme. The schema contains as input block V4L2 Video Capture that is used to get the video in the real time from a camera connected to the platform via a USB connector. In this work the camera with a USB connector was not used, but it was applied the Raspberry Pi camera that is natively produced for the Raspberry Pi and it connects using a CSI port. This is why it was necessary to make at first this camera for the mentioned block as an input device. For this adaptation, a series of the following commands were used:

```
mypi = raspi
system(mypi,'sudo modprobe bcm2835-v4l2')
system(mypi,'ls -al /dev/vid*')
```

Subsequently various parameters of this block were set. Parameter Device name that represents the path to the camera in the system Linux is '/dev/video0'. The screen resolution was set to 320x240 pixels. At the higher resolution such as 800x600 pixels it was too much image points for processing, the device was delayed and latency reaches the level that would be intolerable in the practice. The output was sent to the SDL Video Display block in the RGB format and the image is afterwards displayed.

On the algorithm input is the video that is divided into individual images, and each image needs to pass through this algorithm. Each image is further divided into the three components namely R, G, B, and each of these components consists of a two-dimensional matrix with the dimensions as the video processed by the camera. Afterwards the algorithm starts by saving the video size and allocating the input to output, thus ensures the video presentation on the screen without any intervention if any of the algorithm conditions will not be fulfilled. Another step is the intensity approximation of the fundamental components of this model:

```
green = g - r/2 - b/2;
blue = b - r/2 - g/2;
red = r - g/2 - b/2;
```

It continues with thresholding that helps to find colored areas, needed to be recognized, in the picture

```
center = green > 50 & red < 50 & blue < 50 ...
    | green < 50 & red > 50 & blue < 50 ...
    | green < 50 & red < 50 & blue > 50 ...
    | green > 50 & red > 50 & blue < 50;
```

At this point, in this algorithm shapes can be defined. These shapes will be later incremented to the input and thus plot various shapes of movement orientation. Consequently, five arrows will be defined – arrows for the forward, backward, left, right movement and for the movement around the axis Z of the device. Furthermore, the center of the colored area needs to be found. It will be marked with the red color for better orientation on the video output. In the following step this algorithm verifies if in the taken picture is more color than white points. If so it begins the part of the algorithm, which helps make the device with navigation to the colored areas, assuming that the color area is another orientation point on the journey. If such assistance is necessary, the directions are given by the white arrow drawings.

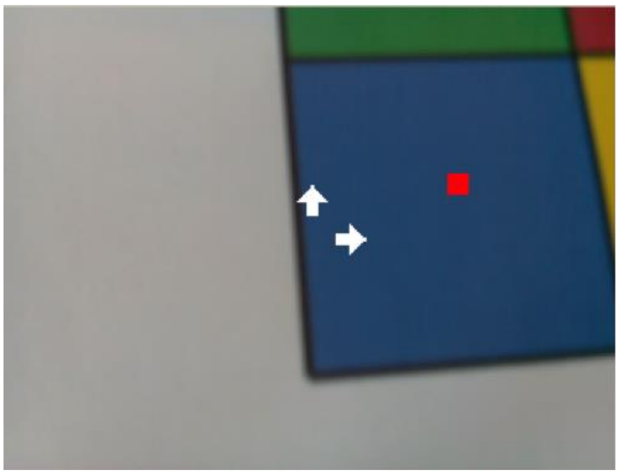

Figure 5: Centering the area

This algorithm continues with the center of the colored area position detection in consideration to the center of the scanned image position. Unless these centers are equal with certain latitude then it begins to evaluate the conditions inside:

```
if (xm<=((x1/2)+100) & xm>=((x1/2)-100)) & (ym<=((y1/2)+100) & ym>=(y1/2)-
100)
```

The following part is the position detection of the green square which is in every single predefined picture and it should always be in the top left corner. Unless it is not there, what is determined with its center point location depending on the center of the full colored area, the rotation will be performed. If the green area is in the dedicated area, the algorithm detects the color of other squares, and the position. On this basis it is able to determine the direction of movement:

```
if(r(X1,Y1) > treshold  & g(X1,Y1) > treshold & b(X1,Y1) < treshold...
  & r(X2,Y2) > treshold & g(X2,Y2) < treshold & b(X2,Y2) > treshold...
  & r(X3,Y3) > treshold & g(X3,Y3) < treshold & b(X3,Y3) < treshold)
```

After a particular image detection and motion direction detection, the arrow is drawn on the screen. This arrow represents the movement of the robot into a given direction, respectively the rotation of the robot. Each arrow is treated as a two-dimensional matrix and was obtained from the images of the grayscale type with 20x20 pixels dimension. At the end of the algorithm, the input is again assigned to the output. If there is some adjustment in the input, it will be reflected at the output, where the shape that has been added into the picture is rendered.

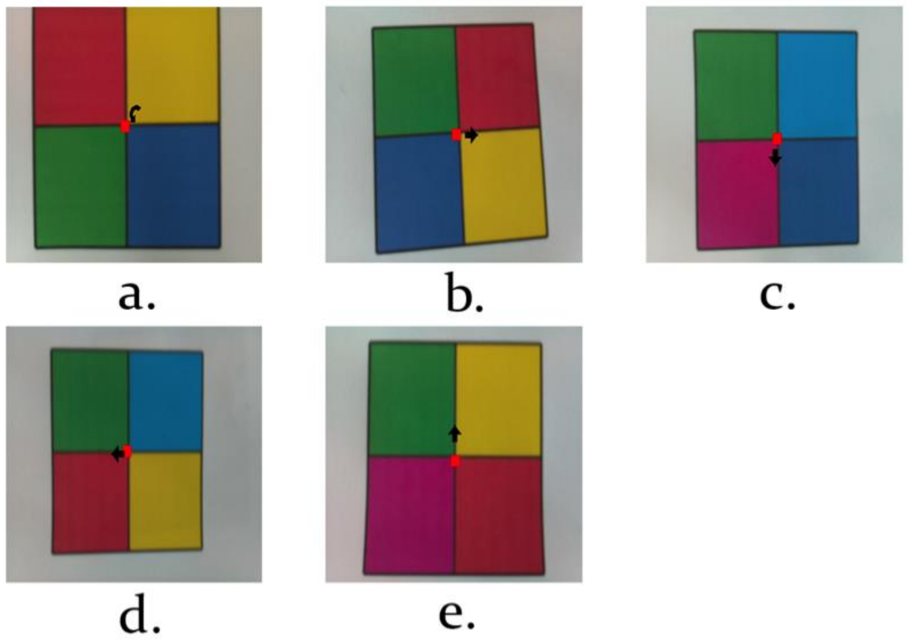

Figure 5: Figure selection: (a.) rotation; (b.) movement to the right; (c.) movement to the back; (d.) movement to the left; (e.) movement forward

4 Conclusion

In this article the one of the most important platforms of its kind, the Raspberry Pi, was used. This device showed the diversity and the scope of its wide use. This little computer is suitable for small and middle projects because it is limited with its performance which in some cases is not sufficient. MATLAB after the connection to the Raspberry Pi offers a new dimension for this device and thus it further adds to the attractiveness of the already popular computer. As an example of this connection the image processing were chosen. It was decided for a role of recognizing the predetermined patterns and they were used to subsequent movement in the space. This algorithm is able to navigate a robot or a drone with markers and go through with him a certain way.

Acknowledgement

This paper was supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under the contract No. VEGA-1/1256/12.

References

[1] ARDUINO. *Arduino*, [online], 2015, Available at: http://arduino.cc
[2] RASPBERRY PI. F. *What is a Raspberry pi?,* [online], 2015, Available at: http://www.raspberrypi.org
[3] BeagleBoard.org. *BeagleBoard-xM*, [online], 2015. Available at: http://beagleboard.org/getting-started
[4] A. Asadi. *Raspberry Pi for Beginners,* Second Revised Edition. 2014, Bournemouth: Imagine Publishing Ltd.
[5] M. SCHMIDT. *Raspberry Pi: A Quick-Start Guide*, 2nd Edition, 2014, Dallas: The Pragmatic Programmers, LLC. ISBN-13: 978-1-93778-580-2.
[6] Mathworks, *Raspberry Pi Support from MATLAB*. [online]. 2015. Available at: http://www.mathworks.com/hardware-support/beagleboard.html
[7] Mathworks, *Raspberry Pi Support from Simulink*. [online]. 2015. Available at: http://www.mathworks.com/hardware-support/beagleboard.html

Michal Blaho
blaho@humusoft.sk

Leo Mrafko
leo.mrafko@stuba.sk

Tatiana Mudráková
tatianam@humusoft.sk