TECHNICAL UNIVERSITY OF CLUJ-NAPOCA

ACTA TECHNICA NAPOCENSIS

Series: Applied Mathematics, Mechanics, and Engineering
Vol. 61, Issue III, September, 2018

OPEN-SOURCE SOFTWARE IN ROBOTICS

Sanda TIMOFTEI, Emilia BRAD, Anca SARB, Ovidiu STAN

Abstract: *The "Open source" term arose at the desire of some people that wanted to make available the source code of software that can be afterwards modified, improved and redistributed. In this way, a group formed by people from different places on Earth can work at the same project, in the same time, having at the end a better version of it. Open source software is licensed software offering the possibility to work without any problem, to collaborate in an open manner at a certain idea. Types of open software and hardware will be presented in this paper, precisely the ones that can be used for image processing. Except LINUX operating system, OpenCV is a type of open-source software, a computer vision library, available to everyone that wants to use it or has a better idea concerning a tool of that software, which can be used on other open source software, like ROS environment. In the last part of the paper, a study case using open source software will be presented.*

Key words: *Open-source software, license, OpenCV, ROS, image processing.*

1. INTRODUCTION

Twenty years ago, after a session that took place in Palo Alto in California, on the 3rd of February, the Open Source Initiative (OSI) was founded as a non-profit global organization. Its aim is to protect and promote open source products, based on free collaboration, education, sustaining the Open Source Definition (OSD). Taking into account the OSD, the abuse of the open source movement has been avoided. [1]

Based on the first paragraph, this paper has the aim to put into evidence the presence of what is the meaning of open source and what does it need so that it can be used correctly. Furthermore, the open source license term will be explained, examples of open source software and hardware will be given. In the last part of the paper, an example of open source software, used to conduct to a research using this type of software in robotics, will be given.

2. OPEN SOURCE

2.1 Open source model

Technology has arisen at such a high level that collaboration in an open way is ordinary. [3], [4]. Furthermore, people formed communities of individuals by their selves with the aim to help this development. Open-source model is very common when it is about projects like appropriate-technology [5] and drug discovery [6], [7], [2].

Starting from these, the concept of open-source model appeared. In other words, it refers to free documentation or source code, that can be accessed by everyone, while redistributing a certain blue print or design [8], [9], excluding the idea of proprietary code [2].

Open-source code is an effort made possible by programmers that collaborate, with the aim to improve a code and to make changes on it in such a way as to improve it. After that, it can be transferred as a software license, license that requires certain terms for each change in the source code. [2][20] There are foundations, like Apache Software Foundation, that support the open-source idea. [2]

2.2 Open source software

The ambiguity of the term "free software" was seen like a discouraging business adoption [12] [13] and it was the starting point for the term "open source" [14]. Precisely, open-source software has the aim to "clarify licensing domain and consumer issues" [2] gaining a wide area and having, as support, Internet development. [10][2]

The open source software term seeks to be "open-minded", commercial [11], and this is why it was proposed and immediately backed in the Linux Journal. All these actions have been part of a movement that enabled this term to be clarified in terms of meaning and used as intended. [14][15][16] People like Richard Stallman, Eric Raymond, Christine Peterson, Todd Anderson, Larry Augustin, Jon Hall, Sam Ockman, and Michael Tiemann are part of the group that created the changed free software movement. [2]

Based on this term, an event has also been organized in April 1998, called "Open Source Summit" [17], by Tim O'Reilly. At this summit, people that had been working with open-source projects, by vote, validated the "open-source" term, which was announced, in the same evening, during a press conference [17]. Even if it was accepted, the term "open-source" referred at a part of the domain, reason for which, the first version and the last version had been combined, resulting the term entitled free and open-source software (FOSS) [18]. [2]

As its name says, open source software is a software that offers the possibility to be accessed by anyone, it can be used and changed. Moreover, it can be shared, but only under the conditions imposed by OSD. These conditions represent ten criteria that must be accomplished to label a software as being an "Open Source" software. Also, the symbol for this term embodies the idea that it is open to everyone (Figure 1). [1]

Fig. 1. OSI logo [Google][1]

It is very important to mention that an open source software can be used for commercial purpose, because commercial is not the same as proprietary, as they are based on different rules. OSI has ten imposed criteria that are used for any source code that is changed and commercialized, so that no author can impose restriction to the user. [1]

The list of open source software is a wide one, covering a wide range of domains, like networking and internet, games, educational, media, artificial intelligence (robotics), programming, security and so on. The best known open source software packages are: OpenCV, ROS, YARP, AForge.NET, LibreCAD, Bitcoin Core, ERPNext, HomeBank, ImageJ, 3DSlicer, Firefox - Mozilla, uTorrent, Media Player Classic, VLC media player, OpenSync, CMake, Open VPN, LibreOffice, Virtual Box. [21]

2.3. Open source hardware

The idea of open source can also be found in hardware (Figure 2) projects, in domains like machine and tools, robotics, prototyping, medical, telephony, home automation and so on, where open source hardware projects are available. The best-known projects are Arduino, 3D Printers, CUBIT, ICub, Wind Turbines, Open Source Ecology, SmartThings. [22]

Fig. 2. Open source hardware logo [Google]

3. OPEN SOURCE LICENSE

The idea to give access for everyone to see, to use and to change the source code, with no prejudice, embodies the principles of being an open source license. In order to see if an open source license accomplishes the requirements, each license has to pass an objective test imposed by Open Source Definition [23].

In order to improve an open-source project, the author must do it under a license, even if it is implicitly already part of the project or explicitly Apache Contributor License Agreement. In some cases, there is no license for the open-source, so the author needs the author's copyright. This way, the contribution to the project will be accepted [23], [24].

3.1. Open source license – types

There is no “the best license” that can be used, so you can choose the one that better fits your requirements, starting from the “popular” ones. To choose the one that fits your work and your needs, it is not necessary to talk to a lawyer; a developer will be able to help you. [23]

Considering that there exist 180.000 open-source projects that are available and at least 1400 unique licenses, the management between open-source and closed-source became more complex. Some of the open-sources are done from home or under FOSS license. Some examples of open source software licenses are: Berkeley Software Distribution ("BSD"), Apache, MIT-style (Massachusetts Institute of Technology), GNU General Public License (“GPL”), GNU Lesser General Public License, Eclipse Public License, Mozilla Public License and many more [28][23][25].

In the background, any open-source license has, legally, conditions of copyright. If one of the users violates these conditions, their license will not be available anymore, which means that they are infringing copyright [23]. [23][26] This is one of the reasons why vendors of commercial software are using open source software. [23][27]

4. STUDY CASE IN ROBOTICS

4.1. OpenCV

OpenCV (Open Source Computer Vision Library) is machine learning and an open source computer vision software (see logo in Figure 3) that is working under a BSD commercial license, also used for academic purposes. As its name says, OpenCV is a library of codes used in different domains using programming languages like C++, Java and Python on operation systems like Linux, Windows, Android, Mac and iOS. Now, the last version of the software is OpenCV 3.4.1. [29]

Having more than 2500 optimized algorithms, from domains like machine learning and state-of-the-art computer vision, these algorithms can be used for a wide number of applications. [29] Applications like identifying, objects, producing 3D point clouds using stereo cameras, detecting and recognizing faces and determining the human actions from a video, removing the red eyes from images done with the flashlight and so on, are using the OpenCV library to get practical applications. “Detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan” [29] are just some examples of practical applications that are using OpenCV. [29]

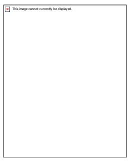

Fig. 3. Open source hardware logo [29]

4.2. ROS

Robot Operating System (ROS) is another example of open source software, where a wide number of contributors and ancestors are working at this large project. This type of distance-collaboration between people is frequently used and it is developed so fast because of the needs of the research community, especially in the robotics domain. [30]

ROS (see Figure 4) can be defined as a collection of libraries, tools, conventions, used to create “complex and robust robot behaviour across a wide variety of robotic platforms” [30], because of the difficulty to create complex application with the robot’s software. [30]

Using ROS means to work in a collaborative way to develop “collaborative robotics software” [30]. To be more precise, as an example, let us consider three groups are working on a project that has the aim to recognize objects, of small dimensions. One group is working on mapping the area where the robot will move. The second group is working on navigation, using the map obtained by the first group, to help the last group, detect the small objects by computer vision approach. [30]

Based on this information, a study case has been done on image processing, using ROS, that will be presented in the next section.

Fig. 4. What ROS means [30]

4.3. Extraction on layers

This section has the aim to present the obtained results by using open source software in an application that, further on, will be implemented in robotics.

The application uses the ROS environment with the aim to convert a coloured image in another format so that it can be further processed. The obtained results can be considered a small step in image processing, but are the starting point in different applications, including the ones where robots are used at the fine art stage.

In other words, the study case is about the conversion of a coloured image, in a format that can be used in different ways, for different applications. It must be mentioned that this study case is just a part of a larger research.

The study case will be done on the image presented in Figure 5. A certain cover notebook has been used because of the way it looks (it has a large array of colours). This cover enables us to prove the functionality and accuracy of the program created in ROS.

To develop a program in ROS, a variety of programming languages can be used. ROS offers the opportunity to develop an application by programming in more programming languages at the same time. These programs will be later incorporated/integrated into the same functional hardware (in this case, a robot).

Each program is called a node and each node has its own programming language, starting with C++, Phyton, Java etc. In this study case, C++ programming language was used.

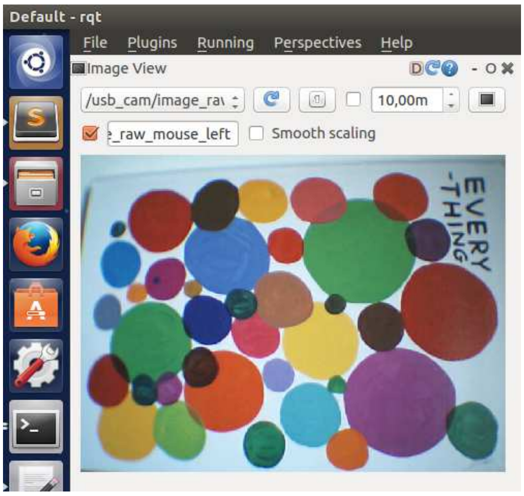

Fig. 5. Initial image [authors contribution]

The idea of this study case is not about extracting the contour of each object that forms the original image, it is about selecting the objects according to their colour, based on the primary colours (RGB - red, green, blue - colour model) (Figure 6) and secondary colours (CYM - cyan, yellow, magenta - colour model) (Fig. 7).

Based on Figure 6 and Figure 7, what must be noticed is that the result of the synthesis of the three colours is different. In the case of RGB colour model, by combination of all three colours, the result is white (W); whose value is “255” in binary system. In the case of CYM colour model, the result is black (B), which in binary system is “0”.

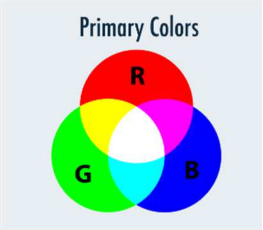

Fig. 6. Primary colours [Google source]

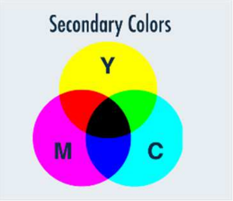

Fig. 7. Secondary colors [Google source]

Combining primary colours, depending on their value, new colours will result. The first colours that result by combining red, green and blue, are cyan, yellow and magenta. Knowing all of these, a code in C++ has been written to extract each colour on different layers.

Using ROS environment, the colour extraction code has been written in a node that is a subscriber and a publisher in the same time. To be a subscriber, the image is taken from a channel, called topic (in this case *ros::Subscriber sub;*), and is given the result on a certain topic called publisher, whose name depends on the colour that is given as result (in this case *ros::Publisher pubR, pubG, pubB, pubY, pubC, pubM;*).

The source codes for blue and cyan colour extraction are presented. Comparing the two source codes, it can be noticed that the difference is just the value of one of the primary colours (based on the source codes, it is about green colour value).

```
// BLUE
 for (int i=0; i<img.height; i++)
  {
   for (int j = 0; j< img.width; j++)
    {
      int myR=
img.data[i*img.width*3+j*3+0];
      int myG=
img.data[i*img.width*3+j*3+1];
      int myB=
img.data[i*img.width*3+j*3+2];

         if (myB > myG + diff && myB > myR
      + diff && myB > diff)
         {
         temp_img_b.data[i*img.width*3+j*3]
= 0;

temp_img_b.data[i*img.width*3+j*3+1] = 0;

temp_img_b.data[i*img.width*3+j*3+2] =
255;
        }
        else
           {

temp_img_b.data[i*img.width*3+j*3] = 0;

temp_img_b.data[i*img.width*3+j*3+1] =
0;

temp_img_b.data[i*img.width*3+j*3+2] =
0;
           }
     }
  }
```

```
// CYAN
 for (int i=0; i<img.height; i++)
  {
   for (int j = 0; j< img.width; j++)
    {
      int myR=
img.data[i*img.width*3+j*3+0];
      int myG=
img.data[i*img.width*3+j*3+1];
      int myB=
img.data[i*img.width*3+j*3+2];

     if (myG > myR + diff && myB > myR +
diff && myG > diff && myB > diff)
       {
         temp_img_c.data[i*img.width*3+j*3]
= 0;

temp_img_c.data[i*img.width*3+j*3+1] =
255;

temp_img_c.data[i*img.width*3+j*3+2] =
255;
       }
        else
           {

temp_img_c.data[i*img.width*3+j*3] = 0;

temp_img_c.data[i*img.width*3+j*3+1] =
0;
temp_img_c.data[i*img.width*3+j*3+2]    =
0;
          }
      }
    }
```

Taking as an example the code for blue primary colour, each pixel from the image, on lines and columns, is analyzed. Comparing its value (intensity of blue in that pixel), which is between 0 and 255, with a threshold, formed by the values of red and green and a constant *int diff = 50*, gives the final colour of the pixel. If in all the comparisons, the value of blue is bigger, then that pixel will take 255 as value, which in this case is blue. The rest of the variables are taking the 0 value.

In the second example, for cyan secondary colour, the comparison is a little bit changed. Cyan is the colour that results from blue and green (Table 1). Because of this, each value for blue and green colour must be compared with the value of the red colour and the constant *int diff = 50*. What must be mentioned is that the value of *diff* can be modified, depending on the user and the application where the colour extraction will be used.

Table 1: Principal and secondary colours

	RED	GREEN	BLUE
RED	RED	Yellow	Magenta
GREEN	Yellow	GREEN	Cyan
BLUE	Magenta	Cyan	BLUE

The results obtained by applying the code written in C++ and implemented in ROS can be seen in Figure 8 and Figure 9, presented as primary colours (RGB) and as secondary colours (CYM).

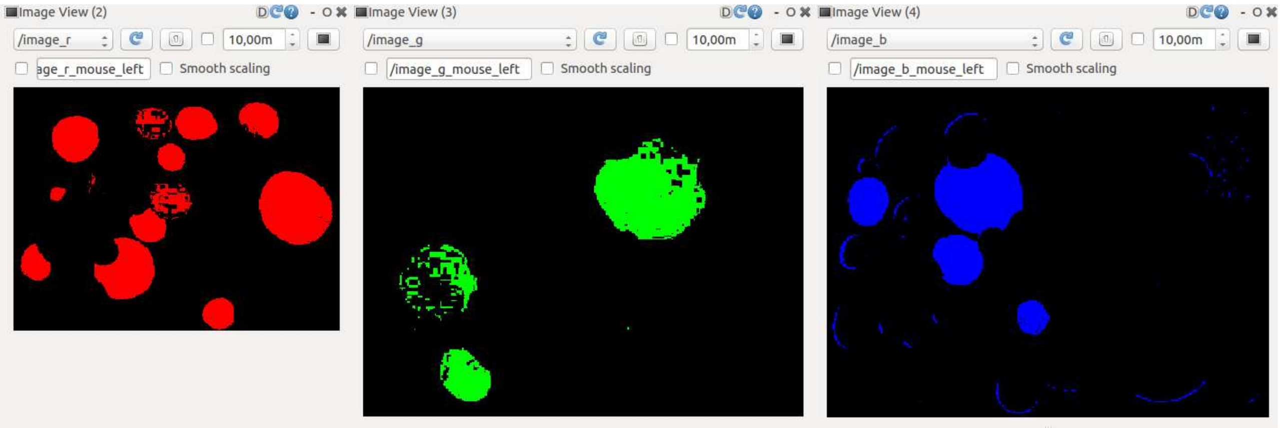

Fig. 8. RGB results obtained using extraction on layers starting from initial image (Figure 5) [authors contribution] Image view (2) – red; Image view (3) – green; Image view (4) – blue

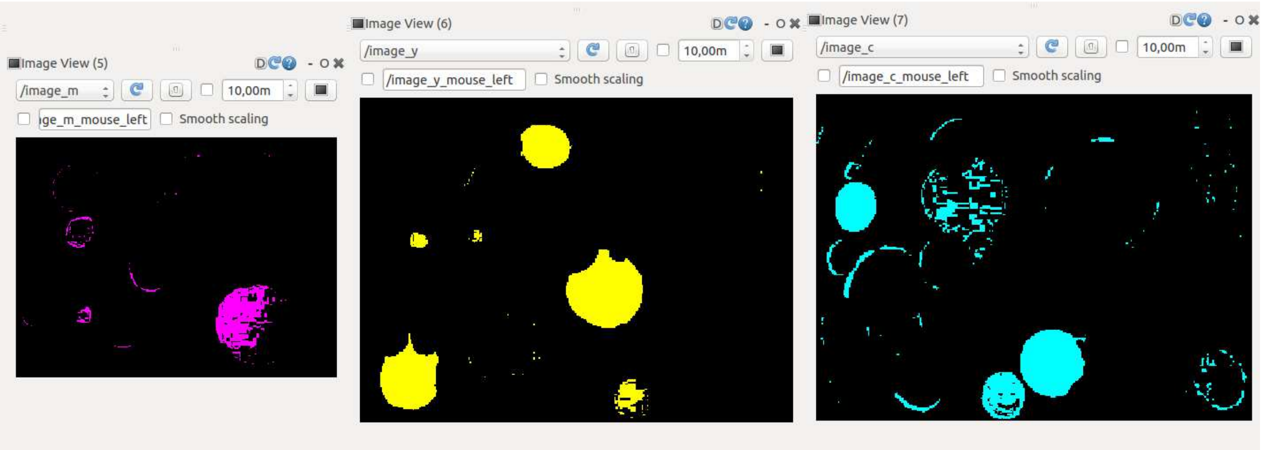

Fig. 9. CYM results obtained using extraction on layers starting from initial image (Figure 5) [authors contribution] Image view (5) – magenta; Image view (6) – yellow; Image view (7) – cyan

5. CONCLUSION

Starting from a coloured image, using the basic steps of image processing, different results can be obtained. This paper had the aim to go another way, to make the difference between objects in image. Based on section 4, the wanted results have been obtained.

The study case presented in this paper can be considered a forward step in image processing for developers that are part of open source software communities. Based on the obtained results using extraction on layouts, of primary and secondary colours, new ideas or ways for research can be founded.

What is important to mention is that open source software has no ending. Its aim is being able to help developers get inspired, starting from an idea that can continually improve or innovated.

6. REFERENCES

[1] Official open-source site (https://opensource.org/faq accessed at May 20, 2018)

[2] https://en.wikipedia.org/wiki/Open-source_model accessed at May 20, 2018

[3] Levine, Sheen S., & Prietula, M. J., Open Collaboration for Innovation: Principles and Performance. Organization Science, pp 1414–1433, ISSN 1047-7039, 2013

[4] Raymond, Eric S., The cathedral and the bazaar: musings on Linux and Open Source by an accidental revolutionary. OReilly. ISBN 978-0-596-00108-7, 2001

[5] "The Case for Open Source Appropriate Technology". Environment, Development and Sustainability, pp 425–431, 2012

[6] "Science 2.0 is here as CSIR resorts to open-source drug research for TB" Business Standard, 1 March 2009

[7] Open Source Drug Discovery for Malaria Consortium (https://openwetware.org/wiki/OpenSourceMalaria:Story_so_far accessed at May 02, 2018)

[8] Lakhani, K.R.; von Hippel, E., "How Open Source Software Works: Free User to User Assistance". Research Policy, pp 923–943, June 2003

[9] Gerber, A.; Molefo, O.; Van der Merwe, A., "Documenting open-source migration processes for re-use". In Kotze, P.; Gerber, A.; van der Merwe, A.; et al. Proceedings of the SAICSIT 2010 Conference — Fountains of Computing Research. ACM Press. pp. 75–85, ISBN 978-1-60558-950-3, 2010

[10] Weber, Steve, “The Success of Open Source”, Harvard University Press, ISBN 978-0-674-01292-9, 2004

[11] O'Mahony, Siobhan Clare, "The emergence of a new commercial actor: Community managed software projects". Stanford, CA: Stanford University: 34–42, 2002

[12] Eric S. Raymond, "Goodbye, "free software"; hello, "open source"". Retrieved 2012-10-25 (http://www.catb.org/~esr/open-source.html accessed at May 02, 2018)

[13] Shea, Tom, "Free software - Free software is a junkyard of software spare parts", 23 June 1983 InfoWorld, Retrieved 2016-02-10 (https://books.google.ro/books?id=yy8EAAAAMBAJ&lpg=PA31&ots=IdSaFSmKVK&dq=us+government+public+domain+software&hl=de&pg=PA31&redir_esc=y#v=onepage&q=us%20government%20public%20domain%20software&f=false accessed at May 02, 2018)

[14] Tiemann, Michael. "History of the OSI". Open Source Initiative. Archived from the original on 1 October 2002. Retrieved 23 August 2008 (https://web.archive.org/web/20021001164015/http://www.opensource.org/docs/history.php accessed at May 02, 2018)

[15] "Why Open Source misses the point of Free Software" (https://www.gnu.org/philosophy/open-source-misses-the-point.html accessed at May 02, 2018)

[16] Muffatto, Moreno, Open Source: A Multidisciplinary Approach. Imperial College Press. ISBN 1-86094-665-8, 2006

[17] Guido van Rossum. "Open Source Summit". Linux Gazette. Archived from the original on 29 December 2013. Retrieved 2015-02-07 (https://web.archive.org/web/20131229053622/http://linuxgazette.net/issue28/rossum.html accessed at May 02, 2018)

[18] https://en.wikipedia.org/wiki/Free_and_open-source_software accessed at May 02, 2018

[19] https://en.wikipedia.org/wiki/Open-source_software#cite_note-1 accessed at May 20, 2018

[20] St. Laurent, Andrew M., Understanding Open Source and Free Software Licensing, O'Reilly Media, p. 4, ISBN 9780596553951, 2008 (https://books.google.ro/books?id=04jG7TTLujoC&pg=PA4&redir_esc=y#v=onepage&q&f=false accessed at May 20, 2018)

[21] Open source software packages list (https://en.wikipedia.org/wiki/List_of_free_and_open-source_software_packages accessed at May 20, 2018)

[22] Open source hardware (https://en.wikipedia.org/wiki/List_of_open-source_hardware_projects#Robotics accessed at May 20, 2018)

[23] Open source software licensing (https://en.wikipedia.org/wiki/Open-source_software#Open-source_software_licensing accessed at May 20, 2018)

[24] Rosen, Lawrence. "Joint Works - Open Source Licensing: Software Freedom and Intellectual Property Law". Retrieved 22 July 2016 (https://flylib.com/books/en/4.467.1.24/1/)

[25] Andrew T. Pham, Verint Systems Inc. and Matthew B. Weinstein and Jamie L. Ryerson. "Easy as ABC: Categorizing Open Source Licenses"; www.IPO.org, June 2010

[26] Shiels, Maggie, "Legal milestone for open source" BBC News. Retrieved 2008-08-15 (http://news.bbc.co.uk/2/hi/technology/7561943.stm accessed at May 20, 2018)

[27] Popp, Dr. Karl Michael, Best Practices for commercial use of open source software. Norderstedt, Germany: Books on Demand. ISBN 978-3738619096, 2015

[28] Popular and approved licenses (https://opensource.org/licenses accessed at May 20, 2018)

[29] OpenCV (https://opencv.org/about.html accessed at May 24, 2018)

[30] Robot Operation Systems (ROS) (http://www.ros.org/about-ros/ accessed at May 24, 2018)

[31] Sarah Manzoor, Raza Ul Islam, Aayman Khalid, Abdul Samad, Jamshed Iqbal, An open-source multi-DOF articulated robotic educational platform for autonomous object manipulation, Robotics and Computer-Integrated Manufacturing 30, pp 351–362, 2014,

[32] https://www.sciencedirect.com/science/article/pii/S0736584513001002?via%3Dihub accessed at May 31, 2018.

SOFTWARE OPEN-SOURCE ÎN ROBOTICĂ

Rezumat: Termenul "Open source" a apărut din dorința unor persoane de a pune la dispoziție codul sursă al oricărui software care poate fi ulterior modificat, îmbunătățit și redistribuit. În acest fel, un grup de oameni din diferite locuri de pe Pământ pot lucra la același proiect, în același timp, obținând la final o versiune îmbunătățită a acestuia. Software-ul open source este un software cu licență care oferă posibilitatea de a lucra fără probleme, de a colabora într-un mod deschis în ceea ce privește o anumită idee. Tipuri de software și hardware open source vor fi prezentate în această lucrare, punând accentul pe acelea care pot fi utilizate pentru procesarea imaginilor. Cu excepția sistemului de operare LINUX, OpenCV este un tip de software open source, o bibliotecă pentru computer vision, disponibilă pentru oricine dorește să o folosească sau are o idee mai bună cu privire la un instrument al acelui soft, care poate fi folosit pentru alte programe open source, cum ar fi mediul ROS. În ultima parte a lucrării, va fi prezentat un studiu de caz care utilizează un soft de tip open source.

Sanda TIMOFTEI, Design Engineering and Robotics Department, Faculty of Machine Building, Technical University of Cluj-Napoca, Muncii Boulevard 103-105, Cluj-Napoca, Romania, s.timoftei@yahoo.com

Emilia BRAD, Design Engineering and Robotics Department, Faculty of Machine Building, Technical University of Cluj-Napoca, Muncii Boulevard 103-105, Cluj-Napoca, Romania, emilia.brad@staff.utcluj.ro

Anca SARB, Design Engineering and Robotics Department, Faculty of Machine Building, Technical University of Cluj-Napoca, Muncii Boulevard 103-105, Cluj-Napoca, Romania, anca.sarb@muri.utcluj.ro

Ovidiu STAN, Synthetic Dynamics S.R.L., 14B Colonia Breaza Street, Cluj-Napoca, Romania, ovidiu.stan@syntheticdynamics.ro