ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

Stereoscopic 3D Anaglyph Video System using Beagleboard XM

Dr. Elio Lozano Inca[1], Emanuel Correa Rivera[2], Javier Gumán Febres[2], Nestor Díaz Fuentes[2]

Associate Professor, Dept. of Computer Science, University of Puerto Rico, Bayamón, Puerto Rico[1]

BS, Dept. of Computer Science, University of Puerto Rico, Bayamón, Puerto Rico[2]

ABSTRACT: 3D anaglyph video systems are a low-cost technique to implement 3D imagingand can be applied in areas such as virtual reality, medicine, product design and visualization.The popular open source multimedia framework GStreamer, which is based on several plugins is widely used to handle multimedia data. This project proposes a modification of the videomixer plugin of GStreamerto implement the streaming of a 3D anaglyph video via the TCP/UDP protocol. This system is based on two BeagleBoard XM single board embedded device computers with Leopard Imaging cameras running Linux Angstrom OS and a PC running Linux. This systemwas implemented using GStreamer to send, receive, and blend left and right images. This design is based on different plugins and it will provide synchronization on various codecs to add functionality, like customization of the stream's audio and video components. The TCP/UPD protocol was used for communication between the embedded devices and the PC. Command pipelines were created to pass these commands to the GStreamer application to send and receive the data flow through the pipeline. This system generates a stereoscopic 3D video, achieved by encoding each eye's image using chromatically opposite color filters, such as red and cyan, and superimposing two images as a way to make the human brain combine them to visualize a single three-dimensional image. The implemented-three dimensional anaglyph system mixes the video dataand creates three-dimensional anaglyph video in real time.Finally, experiments were performed using real-time video streaming and static stereo images achieving the desired results.The implemented software is open source and is freely available for the research community.

KEYWORDS: 3D Anaglyphs, Beagleboard XM, Leopard Imaging, Gstreamer, Videomixer

I. Introduction

Two-dimensional images are appropriate in many situations, but a third dimension depth makes these images more real. The depth changes the interaction with visual media. 3D imaging is better than 2D imaging for visualizing solid objects, and it can be used in tasks such as product design, remote surgery, and layout of stores, molecular science, video conferencing, space flight, education, combat training, virtual tourism, communication, and geo-exploration. 3D anaglyph video systems are a low-cost technique that adds a third dimension depth.

The research objective is to describe the implementation of a low-cost anaglyph video system with an Ethernet interface based on two embedded Beagleboard XM platforms and their Leopard Imaging cameras. The driver of the camera was used to capture images from camera devices. Then, each embedded device uses a GStreamer plugin to send video data over UDP/IP protocol to PC computer which uses other GStreamer plugin to mix the left and right video data.

The content of this paper is organized as follows. In Section I a literature review regarding anaglyph video systems was performed. A brief introduction to anaglyph images is given in section II. Identification and description of the hardware and software required to develop proposed system was made in section III. In section IV the implementation of the anaglyph video system, the experiments performed and the results achieved are covered. Section V analyzes and discusses the results of the experiments. Finally, in section VI the conclusions about the developed system and future work are presented.

II. Related work

Grüner [6, 10] developed a plugin to combine two image streams into a 3D anaglyph stream. This GStreamer framework was implemented through USB connections.Nechypurenko et al. [8] introduced an adaptive video streaming with ICE and GStreamer. This project uses the ICE middleware with GStreamer [5] to implement real-time QoS-aware video streaming for a remotely controlled vehicle. They developed a small vehicle equipped with an on-board computer (based on Beagleboard C4) connected to a WLan adapter and a web-camera. The idea was to control the car over Internet.Bisson [9] developed some GStreamer plugins (video3dmerge, video3dconvert, and video3dpresent) to generate anaglyph video streaming. He was inspired by the ffmpegcolorspace plugin (imgconvert.c) and Orc optimization.Ramirez [11] developed a stereoscopic anaglyph vision system for virtual reality with Java and the Java 3D API. This system generated two outputs, one is projected over the immersion cabin and the other in a computer monitor.Different methods to improve anaglyph technique were developed by Ideses [13] and Sanftmann [14]. These methods find solutions to loss of colour, extreme discomfort for prolonged viewing, ghosting, and bad color reproduction. The proposed 3D anaglyph system was tested with different anaglyph algorithms that are described in 3dtv.at [16].Vandenhouten et al. [12] provided a modular mobile device implementation that allows 3D video streaming. They used single board computing (SBC) running Linux with two USB cameras. Android SDK and gstreamer were used in their implementation.

III. ANAGLYPH IMAGES

The stereoscopic view is an important aspect for applications of virtual reality and 3D imaging. This work describes the development of a system that generates a stereoscopic image. This image is composed of left and right images of the same scene, where the right image has an offset distance with respect to the left image. Each of these images is filtered for each eye using the anaglyph technique. In this technique, two complementary colors (red and cyan) are used to generate anaglyph images. Red is used to extract the red channel of the left image and cyan is used to extract green and blue channels of the right image. These two filtered images are combined to create the anaglyph image. The visual cortex (part of the brain responsible for processing visual information) combines the two filtered images with anaglyph eyeglasses to read a single 3D anaglyph image. These anaglyph images are visualized using a chromatic color pair eyeglasses. The most common chromatic pair is red / cyan. Figure 2 shows how the 3D anaglyph works.

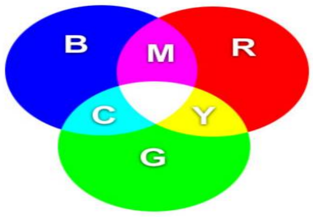


Figure 1. Chromatic color pairs

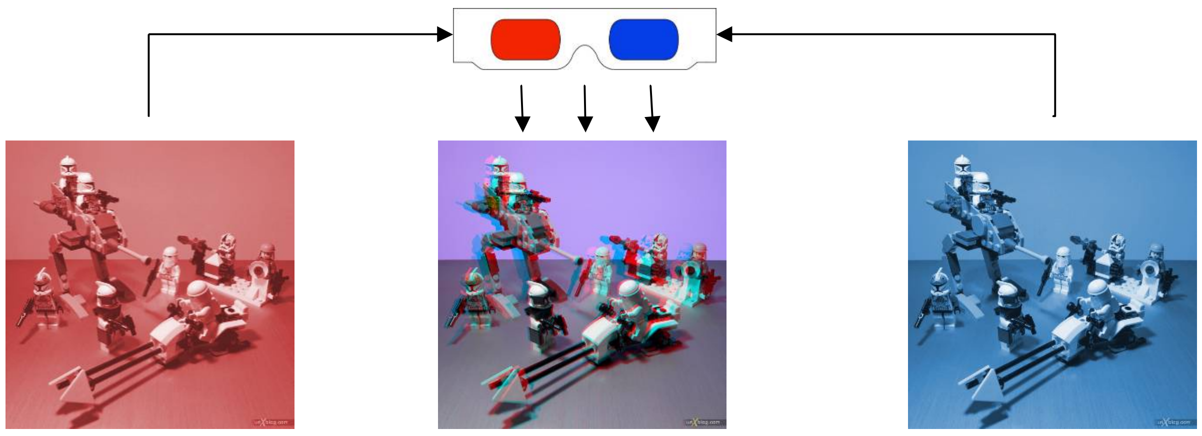


Figure 2. How 3D anaglyph works

The following is a list of the anaglyph matrix transformations used to generate different anaglyph images [3dtv.at]:

A. Monochromatic Anaglyph: $\begin{bmatrix} r_a \\ g_a \\ b_a \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_l \\ g_l \\ b_l \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0.299 & 0.587 & 0.114 \\ 0.299 & 0.587 & 0.114 \end{bmatrix} \cdot \begin{bmatrix} r_r \\ g_r \\ b_r \end{bmatrix}$

B. Color Anaglyph: $\begin{bmatrix} r_a \\ g_a \\ b_a \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_l \\ g_l \\ b_l \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_r \\ g_r \\ b_r \end{bmatrix}$

C. Half Color Anaglyph: $\begin{bmatrix} r_a \\ g_a \\ b_a \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_l \\ g_l \\ b_l \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_r \\ g_r \\ b_r \end{bmatrix}$

D. Optimized Anaglyph: $\begin{bmatrix} r_a \\ g_a \\ b_a \end{bmatrix} = \begin{bmatrix} 0 & 0.7 & 0.3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_l \\ g_l \\ b_l \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_r \\ g_r \\ b_r \end{bmatrix}$

III. EXPERIMENTAL SETUP

On the hardware side, an embedded single-board computer system with a camera acquisition device was chosen, because it has enough processing speed for image processing in real time.

A. HARDWARE

Two embedded single-board *BeagleBoard XM computing devices* (Figure 2.a) [3] were used. Each of these devices has a general purpose processor ARM cortex A8 and second specialized DSP C64P processor running GNU/Linux. Two camera boards from Leopard Imaging LI-5M03 (Figure 2.b) [1] were used. Two USB cables were used to connect the embedded computers to the PC. A PC running Ubuntu 12.04 LTS 32-bit operating system was used to receive the images and create the anaglyph image. Red/Cyan 3D eyeglasses was used to view the 3D anaglyph image.

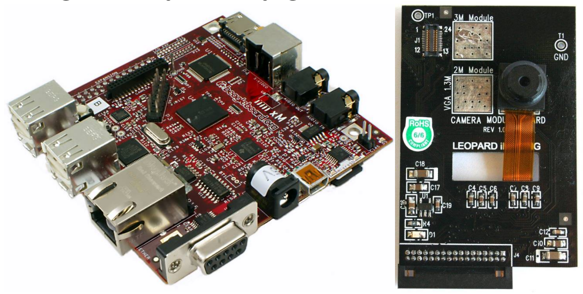


Figure 2. a) Beagleboard MX. b) Leopard Imaging camera.

B. SOFTWARE

Images transmission from the embedded systems were made using the UDP/IP protocol. The popular multimedia application framework called GStreamer [5] was used to handle video streaming. This application is composed from multimedia modules and uses pipelines to process video data between modules.Different video modules (plugins) were tried to update and manipulate the flow of two pipelines to send and receive video data from the cameras. Videomixer was chosen because ithasalpha transparency and blending of images. For video sending and receiving, the udpsink and udpsrc plugins were used. For image mixing and synchronization a videomixerplugin was modified.

Several libraries and plugins were installed to implement the anaglyph software: the ***Gstreamer*** multimedia software version 0.10.36 [7]; the ***libxml***library required to compile the gst-plugingood; ***libxml2-dev*** development library required to compile the gst-plugin-good; ***videomixer***plugin used to merge two pipelines images; the ***videoflip***plugin used to rotate 180 degrees the orientation of the images and to correct the orientation of the flow of images; the ***videobox***plugin used to scale and shift the merged images; the ***v4l2src***plugin used to manipulate video data; the ***videoscale*** plugin used to scale and format the video data; the ***video***plugin to specify the size and color format; the ***fmpegcolorspace***plugin to manipulate Mpeg color space; ***jpegenc/jpegdec***plugin used to enconde/decode in Jpeg format; the ***udpsink/udpsrc*** UDP client/server plugin used to send/receive video data; the ***autovideosink***plugin used to show the image; the ***gst-plugin-good***package version 0.10.31 [GST-P] containing***videomixer*** plugin which was used to implement the proposed anaglyph software. G*cc* version 4.6.3 was the compiler used. And finally, the Leopard Imaging LI-5M03 driver [2] for embedded Angstrom distributionLinux operating system [4] was used.

C. METHOD

Each Leopard Imaging camera board was connected in the daughter camera port of each *BeagleBoard XM* embedded device. These computer devices were connected to a PC via USBports using the Ethernet over USBfeature and a SSH connection was established between these devices and the PC.*GStreamer* library plugin called *videomixer* was compiled in the PC. This plug-in contains the*videomixer.c* and *blend.c* source files for mixing the two pipeline video streams coming from the embedded devices. In the *videomixer.c*file, a function that calls a blending function to mix videos and filters was changed to call a modified blending function. In the *blend.c* file, a function that implements the mixing video pipelines and applying the adequate color space conversion was created to allow conversion from YUV to RGB and RGB to YUV. This function changes the color channel of the right video to cyan and thecolor channel of the left video to red.

GStreamer sender and receiver pipelineswere created to send video stream pipelines from the camera boards and to receive and mix these pipelines into one video with the desired anaglyph effect. A plug-in called video4linux2 or v2l4src was used to create a video pipeline from each video data acquisition device. This pipeline was encoded to the jpeg format using the*jpegenc*plugin. The *udpsink* plugin was used to send this encoded video pipeline as UDP packets through the network. These packets contain video in the YUV color space. The *udpsrc* plugin was used to receivethe UDP packets sent by the *udpsink* plugin. The received video pipeline was decoded by the *jpegdec* plugin. This video pipeline was resized to a desired scale by the *videoscale* plugin, and cropped or scaled accordingly by the *videobox* plugin. These video pipelines weremixedby the *videomixer*plugin and outputs a single video in the YUVcolor space. (Figure 2A). The videomixer plugin was modified to work with two video source pipelines. The compilation of this module required other modules called gst-plugin-bad, gst-plugins-good, and gst-plugins-ugly [7]. Finally, the mixed video is received by the plugin called *autovideosink*, which appropriately displays the anaglyph video data.

```
void blend3(const guint8 * src, const guint8 * src2, gintsrc_width, gintsrc_height, guint8 * dest, gintdest_width,
gintdest_height) {
gint  i, j;
    guint8 R1, G2, B2;
for (i = 0; i <src_height; i++) {
for (j = 0; j <src_width; j++) {
            dest[0] = 0xff;
              R1 = YUV_TO_R(src2[1], src2[2], src2[3]);   //1.164*(src2[1]-16) + 1.596 * (src2[3] - 128) + 0                    ;
              G2 = YUV_TO_G(src[1], src[2], src[3]); //1.164*(src[1] -16) - 0.813 * (src[3] - 128) - 0.391 * (src[2] - 128);
              B2 = YUV_TO_B(src[1], src[2], src[3]);      //1.164*(src[1] -16) + 0                    + 2.018 * (src[2] - 128);
            dest[1] = R1 *  0.257 + G2 *  0.504 + B2 *  0.098 + 16; // Y
            dest[2] = R1 * -0.148 + G2 * -0.291 + B2 *  0.439 + 128; // U
            dest[3] = R1 *  0.439 + G2 * -0.368 + B2 * -0.071 + 128; // V
            dest += 4;
            src += 4;
            src2 += 4;
        }
    }
}
```

Figure 3.Source code of the blending method to mix two source video image data in one destination video image.

Ethernet over USB was used to interface the PC with both embedded devices. The libUSB API library was installed to enable Ethernet over USB using OTG port of the embedded devices for data communication through this port.The usb-

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

gadget file (located in /etc/default/usb-gadget) was configured to allow the device boots as an Ethernet module. Module G_ether was used to allow the USB OTG to work as Ethernet over USB. A static IP was assigned to each embedded device in file /etc/network/interfaces. The blending method was implemented in C to mix the two video pipelines as presented in Figure 3. Different pipeline command flows were created to test the proposed implementation of the 3D anaglyph video system. Example pipelines are shown in part E of this section.

IV. SOFTWARE IMPLEMENTATION

Source files blend.c and videomixer.c of the videomixer plugin were modified. The blend3 method was implemented to mix two pipelines src and src2 in one pipeline called dest (Figure 3). This method is the implementation of the color anaglyph algorithm mentioned above.

V. SIMULATION RESULTS

A. GSTREAMER PIPELINE EXAMPLES

The proposed anaglyph software was tested using both static stereo images and real-time video data. The following are examples of pipeline flows:

1. Gstreamer pipeline with videomixer to view anaglyph image from two left and right static images (canvas_left.jpg and canvas_right.jpg).

```
gst-launch -e videomixer name=mix sink_0::alpha=0.0 sink_1::alpha=1.0 ! ffmpegcolorspace !xvimagesinkmultifilesrc
location="/home/research/canvas_left.jpg" caps="image/jpeg,framerate=1/1" ! jpegdec !ffmpegcolorspace !video/x-
raw-yuv,format=\(fourcc\)AYUV ! mix.    multifilesrc location="/home/research/canvas_right.jpg"
caps="image/jpeg,framerate=1/1" ! jpegdec !ffmpegcolorspace !video/x-raw-yuv,format=\(fourcc\)AYUV ! mix.
```

2. Gstreamer pipelines with v4l2src, video, videoflip, jpegenc, and udpsink plugins to send live video data to computer device. Two different ports were used to identify different source pipelines:

```
gst-launch-0.10 -v v4l2src ! 'video/x-raw-yuv,width=640,height=480' ! videoflip method=vertical-flip ! jpegenc
!udpsink host=10.0.40.6 port=5000
```

```
gst-launch-0.10 -v v4l2src ! 'video/x-raw-yuv,width=640,height=480' ! videoflip method=vertical-flip ! jpegenc
!udpsink host=10.0.40.6 port=5001
```

3. Gstreamer pipeline with videomixer, ffmpegcolorspace, autovideosink, updsrc, jpegdec, video, videobox, and videobox plugins to receive video data and to blend both source pipelines to generate real-time anaglyph video.

```
gst-launch -vvvvevideomixer name=mix ! ffmpegcolorspace !autovideosink sync=false udpsrc caps = "image/jpeg,
width=(int)640, height=(int)480, framerate=(fraction)100/1, pixel-aspect-ratio=(fraction)1/1" port=5000 ! jpegdec
!videoscale ! video/x-raw-yuv , width=640, height=480 ! videobox border-alpha=0 top=0 left=0 ! video/x-raw-
yuv,format=\(fourcc\)AYUV ! ffmpegcolorspace !  mix. udpsrc caps = "image/jpeg, width=(int)640, height=(int)480,
framerate=(fraction)100/1, pixel-aspect-ratio=(fraction)1/1" port=5001 ! jpegdec !videoscale ! video/x-raw-yuv ,
width=640, height=480 ! videobox border-alpha=0 top=0 left=0 ! video/x-raw-yuv,format=\(fourcc\)AYUV !
ffmpegcolorspace !mix.
```

Figure 4 shows how two pipelines flow from the client to mixing on the server side. Figure 5 show the proposed anaglyph video system.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

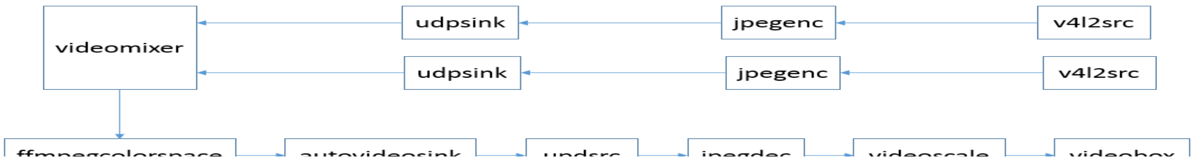


Figure 4. Pipeline flow of plugins from sender to receiver

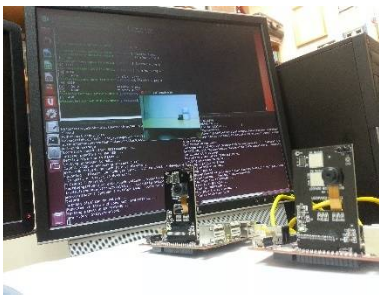


Figure 5. Stereo Vision System

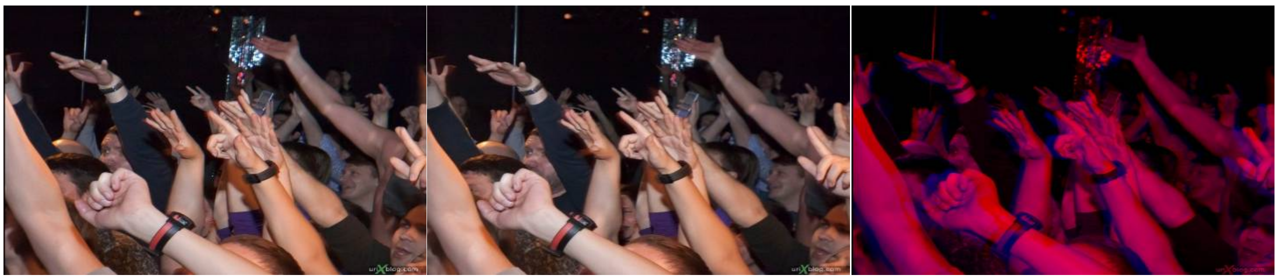


Figure 6. True anaglyph algorithm

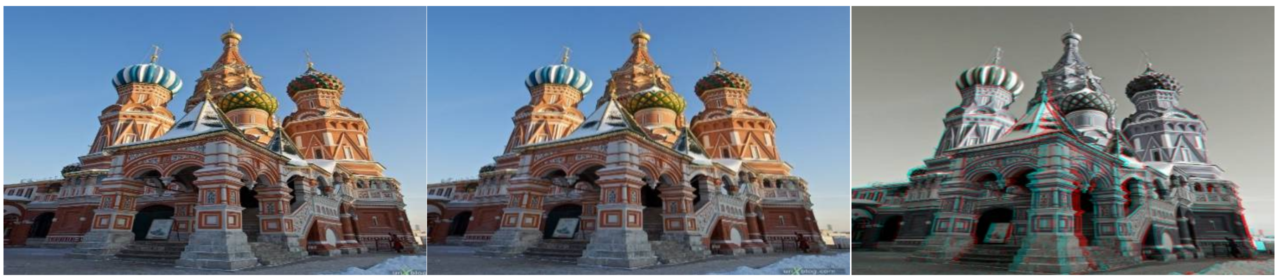


Figure 7. Grayanaglyph algorithm

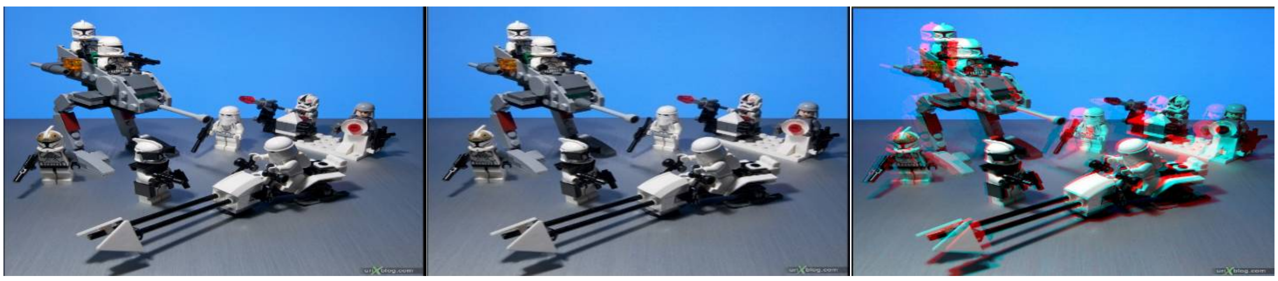


Figure 8. Color anaglyph algorithm

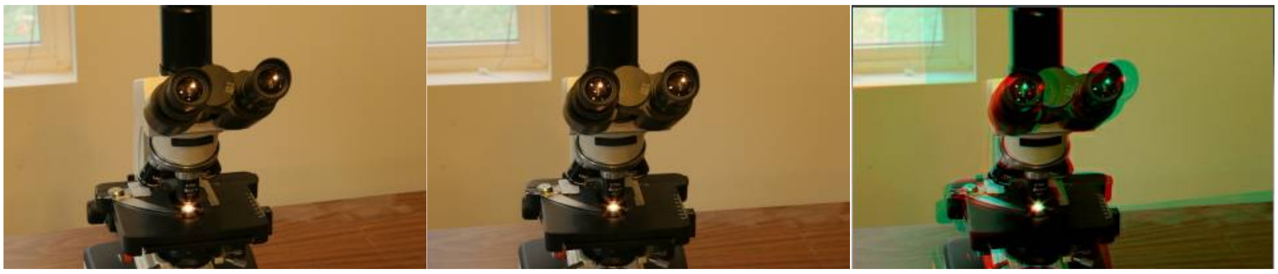


Figure 9. Half color anaglyph algorithm

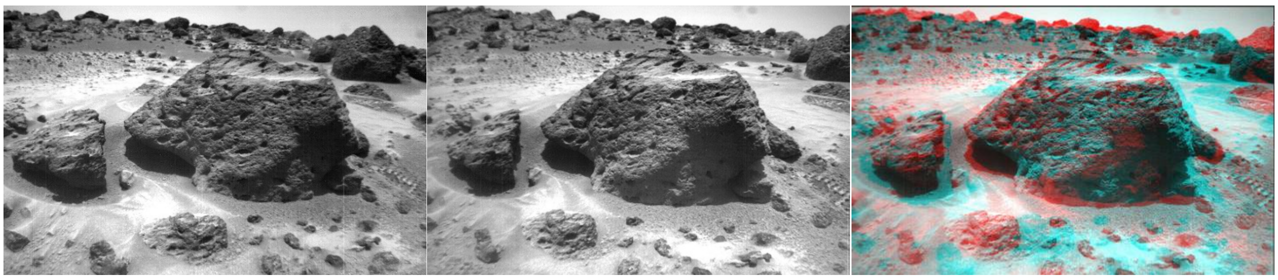


Figure 10. Optimized anaglyph algorithm

Figures 6, 7, 8, 9, and 10 show results of the implementation of five different anaglyph algorithms discussed before. Because these algorithms use different filter matrices, each resulting anaglyph image is different to each other.

VI. FINDINGS AND CONCLUSIONS

A. RESULTS

The anaglyph video system was tested with a pair of static images that have a slightly distance offset and with real time video data. A stereoscopic anaglyph image was generated from blending two streamed pipeline videos. In the implementation red and cyanfilters were used to achieve the anaglyph video. The final configuration required additional hardware positioning arrangement to get desired video. The embedded systems with their cameras needed to be precisely attached to a support base with a little elevation angle. In addition, the direction lines of the field of view in both cameras needed to be aligned to intersect in a specific distance near the front of the cameras in order to view the anaglyph image. This distance depends of the intrinsic features of the camera sensor and can be found in trial and error experiment to get desired results.

B. CONCLUSION

An open-source videomixer plugin was reused to implement successfully the stereoscopic 3D anaglyph video system. The implementation of the blending program was done in C language.The computing embedded device is based on a Beagleboard XM with Leopard Imaging cameras. The OTG port of the embedded device was used to enable Ethernet over USB to enable communication between this device and the PC. The implemented system works with static images and can stream live 3D anaglyph video data.

C. FUTURE WORK

A support base will be designed in a 3D printer for the*BeagleBoardXM*embedded computing deviceswith the desired offset needed to create the anaglyph effect. Another future implementation could be the streaming of video wirelessly.

ACKNOWLEDGMENTS

This research project was sponsored by the University of Puerto Rico at Bayamón. Part of this research was supported by NSF Grant CNS-1042341. This paper was edited by Prof. Antonio Huertas from Computer Science Department of University of Puerto Rico at Bayamón.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

REFERENCES

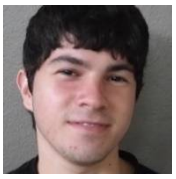
1. Leopard Imaging Inc. (2016). Leopard Imaging 931-LI-5M03 Daughter Card. Available: https://www.leopardimaging.com[Accessed: October - 2016]
2. caganarslan. (2016). Using mt9p031 image sensor on Beagleboard-xM rev C5. Avaialbe: https://caganarslan.wordpress.com/2013/07/04/second-post/[Accessed: October - 2016]
3. BeagleBoard.org Foundation. (2016). Beagleboard XM. Available:http://beagleboard.org/beagleboard-xm
4. Angstrom Distribution. http://wp.angstrom-distribution.org/[Accessed: October - 2016]
5. GStreamer(2014). *GStreamer open source multimedia framework*. Available:http://gstreamer.freedesktop.org/[Accessed: October - 2016]
6. Grüner, M,*Gst Plugin Anaglyph*. Available:https://github.com/michaelgruner/gst-plugin-anaglyph, [Accessed: October - 2016]
7. GStreamer plugins, Multimedia Libraries and Drivers, 2014.Available:http://www.linuxfromscratch.org/blfs/view/6.3/multimedia/gstreamer.html, [Accessed: October - 2016]
8. A. Nechypurenko and M. Parkachov,*Remotely controlled vehicle*. GStreamer Conference. Available:https://gstreamer.freedesktop.org/data/events/gstreamer-conference/2010/slides/Andrey Nechypurenko and MaksymParkachov – Adaptive Video Streaming with Ice and GStreamer.pdf, [Accessed: October - 2016]
9. M. Bisson,*Stereoscopic video and Gstreamer*. GStreamer Conference. Available:https://gstreamer.freedesktop.org/data/events/gstreamer-conference/2010/slides/Martin Bisson - Stereoscopic Video and GStreamer.pdf, [Accessed: October - 2016]
10. Michael Grüner. *Diseño e implementación de un sistema de transmisión de imágenes de anaglifo sobre dos plataformas embebidas BeagleBoardxM*. SIBITEC, biblioteca José FigueresFerrer. Available:http://repositoriotec.tec.ac.cr/handle/2238/2948, [Accessed: October - 2016]
11. Iris Noemí Ramírez García. *Sistema de visión estereoscópica basado en anaglifo para aplicaciones de realidad virtual*. *Tesis de nivel postgrado*. *Instituto Politécnico Nacional de México*. Available:http://hdl.handle.net/123456789/4028, [Accessed: October - 2016]
12. R. Vandenhouten, R. Fiebelkorn, A. Funke,*A Modular Mobile Device for Real-Time 3D-Streaming*. Scientific Contributions, WissBeitr TH Wildau, Vol. 20, pp 37-43. doi: 10.15771/0949-8214_2016_1_5, 2016.
13. I. Ideses and L. Yaroslavsky, *Three methods that improve the visual quality of colour anaglyphs*. Journal or Optics A: Pure and Applied Optics, Vol 7, pp 755-762. doi:10.1088/1464-4258/7/12/008, 2005.
14. H. Sanftmann and D. Weikkopf. *Anaglyph Stereo Without Ghosting*. Eurographics Symposium on Rendering, Vol. 30, No. 4, 1251-1259, 2011.
15. 3dtv.at Inc,*Anaglyph Method Comparison*. Available:http://www.3dtv.at/Knowhow/AnaglyphComparison_en.aspx, [Accessed: October - 2016]

BIOGRAPHY

Dr. Elio Lozano received B.S. in mathematics at National University of San Antonio Abad of Cusco, Perú in 2000. He received M.S. in Scientific Computing and Ph.D. in Computer and Information Science and Engineering at University of University of Puerto Rico at Mayaguez Campus in 2003 and 2006 respectively. Dr. Lozano is with the department of Computer Science, University of Puerto Rico at Bayamón, San Juan, PR, 00959 USA. He has developed several software and firmware for different architectures.

Emanuel Correa Rivera received B.S. in Computer Science of the University of Puerto Rico at Bayamón.

Javier Guzman Febres received B.S. in Computer Science of the University of Puerto Rico at Bayamón.


Nestor Diaz Fuentesreceived B.S. in Computer Science of the University of Puerto Rico at Bayamón.