Kinect sensor performance for Windows V2 through graphical processing

Javier Vargas
Facultad de Ingeniería en Sistemas Electrónica e Industrial.
Universidad Técnica de Ambato
Ambato, Ecuador.
js.vargas@uta.edu.ec

Christian Mariño
Facultad de Ingeniería en Sistemas Electrónica e Industrial.
Universidad Técnica de Ambato
Ambato, Ecuador.
christianjmarino@uta.edu.ec

Clay Aldas
Facultad de Ingeniería en Sistemas Electrónica e Industrial.
Universidad Técnica de Ambato
Ambato, Ecuador.
clayaldas@uta.edu.ec

Luis Morales
Facultad de Ingeniería en Sistemas Electrónica en Industrial.
Universidad Técnica de Ambato
Ambato, Ecuador.
luisamorales@uta.edu.ec

Renato Toasa
School of Technology and Management.
Computer Science and Communication Research Centre (CIIC),
Polytechnic Institute of Leiria
Leiria, Portugal.
2162342@my.ipleiria.pt

ABSTRACT
The present paper describes a study based on the loss and gain of frames that are obtained through sensors (color, depth, and body tracking) of Kinect V2. For this purpose, it is established a time to obtain each frames per second (FPS), evaluating a performance of sensors in three evaluation instances, using native Kinect V2 libraries and other graphic processing libraries. In addition, several experimental tests were carried out, in order to verify the best performance of a test application based on graphical processing of each sensor.

CCS Concepts
• Software and its engineering→ Acceptance testing • Hardware→ Sensor Applications and Deployments • Hardware→ Sensor devices and platforms.

Keywords
Kinect V2; graphic processing; sensor performance; frames.

1. INTRODUCTION
Kinect sensor was originally created for XBOX 360 video game console, which it has been in need of evolution. It currently has two improved versions: a commercial for video games, Kinect sensor for Xbox One [1]; and for development Kinect sensor for Windows v2 (Kinect V2) [2]. These two versions have been the most used for the field of research in treatment of images.

Kinect V2 introduces new features in its sensors or input streams: for color (1920 × 1080 resolution images), infrared (gamma correction with depth sensor), depth (distances 0.5 to 4.5 meters) and body tracking (detection of 25 joints), without forgetting audio recognition (noise reduction) [3]. The management of these new features of Kinect V2 sensor are within the software development kit with interaction of the SDK 2.0 [2, 4].

Use this sensor to have some minimum hardware and software requirements: 64-bit processor, 3.1 GHz physical core, USB 3.0 compatible with the Kinect Adapter for Windows, 4GB in RAM, graphics card that supports DirectX 11 or higher and Windows 8, 8.1 or Windows 10 [4]. These requirements are established to obtain a good performance of Kinect V2 in color sensors, depth-infrared and detection or body tracking [5].

With the sensors that Kinect V2 has, it is possible to access through the native libraries within the SDK 2.0 for developers [6]. The compatibility of these libraries with respect to performance in a test application is established using the frame reader for each input stream [7].

The color sensor is based on the properties: frame saturation and maximum resolution established for each frame [8]. The depth sensor is based on the properties: minimum accepted distance and relative time of saturation with respect to the frame in processing [8, 9]. The body tracking sensor is based on properties: maximum number of people traced and detection of joints in components (x, y, z) within a frame [10].

This paper made an investigation about the performance of the Kinect V2 regarding the sensors (color, depth and body tracking), in function of GPU processing (Graphics Processor Unit) [11]. The three instances of evaluation are done with certain graphic cards, which are: 1GB, 2GB, and 6GB, individually with the test application adapted to the native Kinect V2 libraries [12], and other graphic processing libraries [13].

The parameters of input evaluation are: performance based on processing frames, GPU use and CPU work [14], with the goal of taking advantage of the increased processing capacity in applications of Kinect study [15], to obtain better performance of the use of each sensor of the Kinect V2.

ICMLC 2018, February 26–28, 2018, Macau, China.

ACM ISBN 978-1-4503-6353-2/18/02…$15.00
DOI: https://doi.org/10.1145/3195106.3195116

These work is divided into five sections, including introduction. Section 2, presents the work related to Kinect sensor. Section 3, deals with performance based on each sensor experimental results. Then, Section 4 contains discussion of results in this paper. Finally, conclusions are presented in Section 5.

2. RELATED WORKS

The significant features regarding the Kinect for Xbox One are improved compared to the processing of the sensors with the CPU's work in cache [16]; in contrast to the Kinect V2 where the CPU works trivially with the GPU to avoid loading within the cache that stores the Kinect V2 sensor [17]. These features work based on the frame processing by second (FPS), where the sensors have a minimum and maximum FPS of 15 or 30 according to the color camera light [18, 19]. These values in FPS are considered to determine if the output data of each sensor are correct or have loss too much data.

For the depth sensor the saturation of FPS depends on the GPU [20], due to the use of libraries or programming platforms such as OpenCV (Open Source Computer Vision Library) [21], OpenGL (Open Graphics Library) [22], Unity [23], and even platforms that compile free Kinect libraries, such as OpenNI [24].

For the integration of Kinect V2 sensors in applications, the developers use more OpenCV as GPU code activation [25], thus determining a minimum amount of data of a frame that is a process inside one of the libraries of OpenCV [26].

On the other hand, to determine Kinect processing and to increase the use of GPU, most academic related researchers use NVidia CUDA [27]. For which, there is study with Kinect V2 and NVidia CUDA, to recognition of people and the tracking of RGB-D data [28], obtaining as outcomes, the frame rates gain an improving in compare with the use of CUDA in the processing. The integration of OpenCV in this research, presents an improved data acquisition for the scalable tracking of people, thanks to the OpenPTrack library [29]. The dependency of the libraries with which the Kinect works, allow to support this work in the performance of the Kinect V2 in compare with the processing of the GPU [17]. The instance tests are realized with three graphic cards: 1GB (integrated), 2GB (192 CUDA cores and a 64-bit bus), and 6GB (1152 CUDA cores), applied it to native Kinect libraries for Windows v2 and with another type of development platforms with the use of OpenCV libraries [30].

3. EXPERIMENTAL RESULTS

In order to evaluate and discuss the performance of Kinect V2, in the graphic processing and data output in each of the sensors (color, depth and body tracking), is created a test application that contains calls to the native libraries of the SDK 2.0.

The following code segment shows the basic operation to initialize the sensors to work with Kinect V2 using the SDK 2.0 from C #.

```
// Variable that incorporates the Kinect sensor device
KinectSensor _sensorkinect;
// Reader for multi-source frames.
MultiSourceFrameReader _reader;
//Instance of Kinect sensor connected
_sensorkinect = KinectSensor.GetDefault();
/Initialize the Kinect sensor
if (_sensorkinect != null)
        {
          _sensorkinect.Open(); }
// Reader for multi-stream data frames
_reader = _sensorkinect.OpenMultiSourceFrameReader(
                                FrameSourceTypes.Color |
                                FrameSourceTypes.Depth |
                                FrameSourceTypes.Body);
 // Only it is activated every time a frame or frame is captured
_reader.MultiSourceFrameArrived +=
                        Reader_MultiSourceFrameArrived;
```

This code creates a frame reader for multiple sensor outputs. For each data flow there are: to color (FrameSourceTypes.Color), depth (FrameSourceTypes.Depth), and to body tracking (FrameSourceTypes.Body). On the other hand, the next execution cell shows the basic operation of each sensor (see Figure. 1).

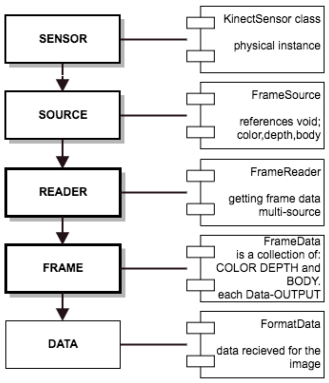

Figure 1. Kinect V2 Process Stack

3.1 Color Sensor: Results in FPS

The result summary of the preliminary evaluation of the FPS with graphic processing, are obtained in correlation to each frame and determined time in 300 seconds or intervals of 1 second for each frame. For the body tracking sensor, a scatter diagram of the evaluation is presented in the test application. The estimation of each frame is determined by values between 15 FPS (average quality - Kinect v2) and 30 FPS (high quality - Kinect v2).

In 1GB of graphical processing, we obtained a value of 233 frames that are less than 30 FPS, and 67 frames greater or equal to 30 FPS, in 300 seconds per frame (see Fig. 2). This determines a significant drop in FPS on the 1GB card, causing the loss of some frames or frames for the evaluation of the application in real time.

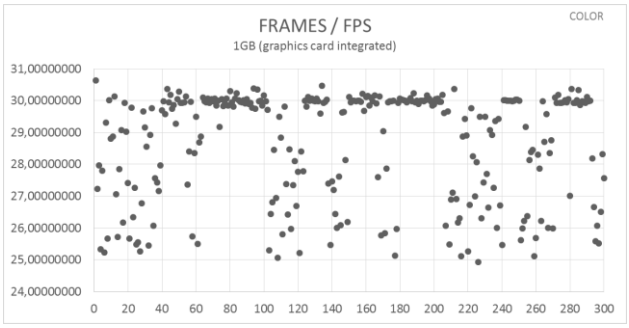

Figure 2. COLOR: Results in FPS – 1GB

In 2GB of graphics processing, we obtained a value of 162 frames that are less than 30 FPS, and 138 frames greater or equal, in 300 seconds per frame (see Figure. 3). In obtaining application data and the color sensor with maximum resolution, it is determined

based on all frames greater or equal to 30 FPS, to get a good performance on the 2GB card.

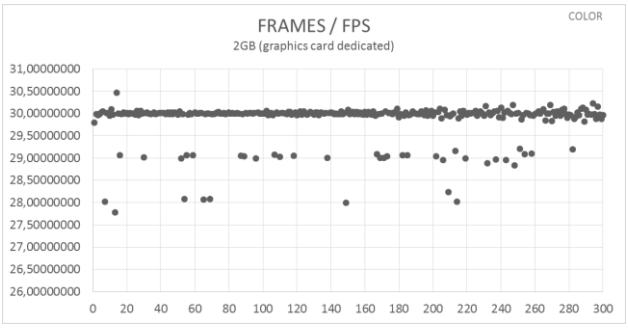

Figure 3. COLOR: Results in FPS – 2GB

In 6GB of graphic processing, we obtained a value of 127 frames that are less than 30 FPS, and 173 frames greater than the quality of 30 FPS, in 300 seconds per frame (see Figure. 4).

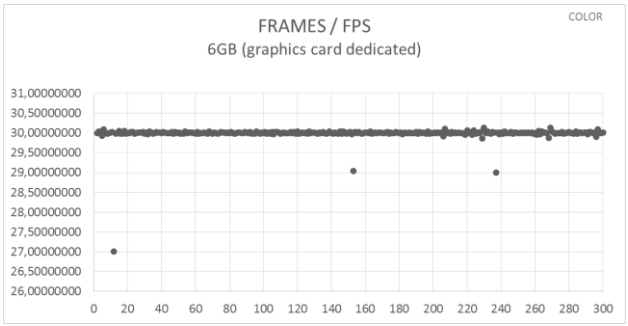

Figure 4. COLOR: Results in FPS – 6GB

The valuation of each FPS in a determined time, allows setting the evaluation properties in the test application of the color sensor, where the 6GB card has a significant improvement over the 1GB and 2GB cards, according to the diagrams presented.

3.2 Depth Sensor: Results in FPS

For the depth sensor, is presented in the same way a scatter diagram of the evaluation in the test application. The estimation of each frame is determined by the values between 25 FPS and 30 FPS, in addition to the relative saturation time.

In 1GB of graphics processing, we are obtained 238 frames that are less than 30 FPS, and 62 frames greater or equal to 30 FPS values, in 300 seconds per frame (see Figure. 5). The intensity with which the native Kinect V2 libraries work, is adjusted to the depth evaluation over a given distance of 0.5 to 2 meters, resulting in a significant loss of data.

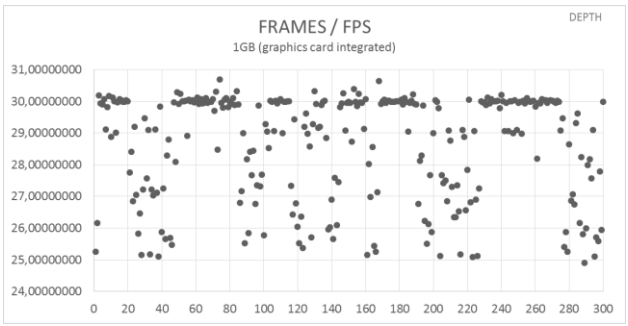

Figure 5. DEPTH: Results in FPS - 1GB

For the evaluation in 2 GB, within the same distances, are obtained 197 frames smaller than 30 FPS, and 103 frames greater or equal to 30 FPS, in 300 seconds per frame (see Figure. 6). The data present a separation in two acceptable ranges between 28 to 29 FPS processing.

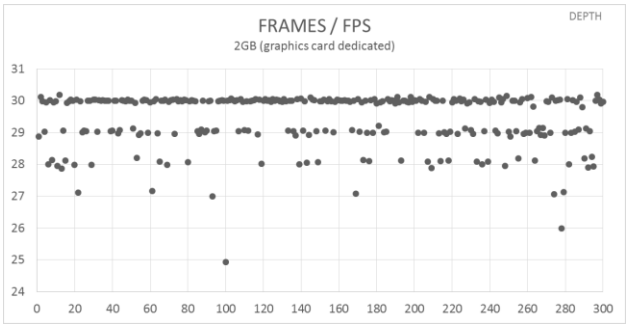

Figure 6. DEPTH: Results in FPS - 2GB

On the other hand, in 6GB of graphics processing, we obtained a value of 121 frames that are less than 30 FPS, and at the same time, there is a high value of 179 frames in compare with 30 FPS, in 300 seconds per frame (see Figure. 7).

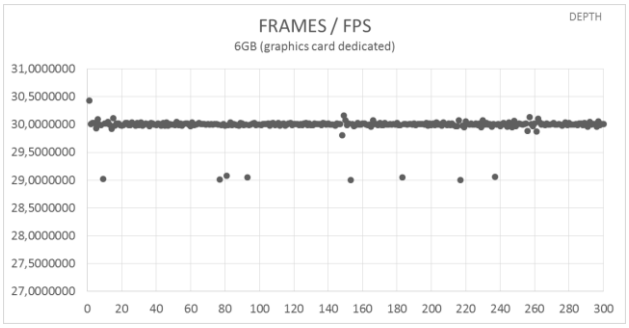

Figure 7. DEPTH: Results in FPS – 6GB

In the depth sensor, the values are adjusted to the changes within the evaluation, each data varies depending on the saturation captured in 300 seconds per frame. For the 2GB card compared to the 6GB card, the variation ranges from 29 FPS to 30 FPS, presenting a minimum loss of data, for the evaluation properties of the application.

3.3 Body Tracking Sensor: Results in FPS

For the body tracking sensor, a scatter diagram is also presented. The frames that are obtained are valued based on a person traced in a certain time referring to the 25 joints detected.

In 1GB of graphic processing, we obtained a value of 230 frames that are smaller than 30 FPS, and 70 frames greater or equal to 30 FPS, in 300 seconds per frame (see Fig. 8). The data collection is based on the evaluation of a bitmap image, with a similar evaluation with the color sensor, results of 1 GB (see Fig. 2).

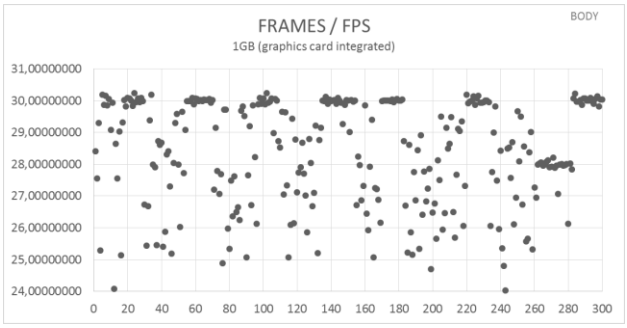

Figure 8. BODY TRACKING: Results in FPS - 1GB

On the other hand, in 2GB of graphics processing, we obtained a value of 188 frames that are smaller than 30 FPS, and 112 frames greater or equal to 30 FPS, in 300 seconds per frame (see Figure 9). The valuation between body tracking and color resembles a scatter of frames that is presented in depth.

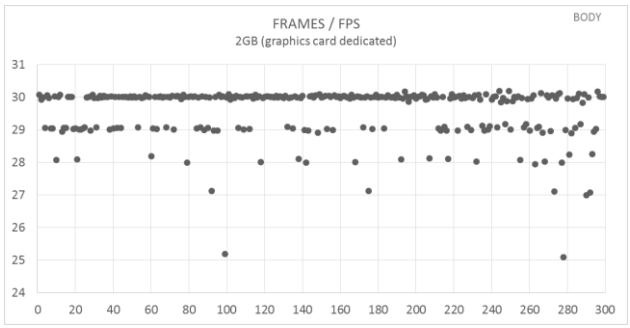

Figure 9. BODY TRACKING: Results in FPS - 2GB

For 6GB of graphics processing, is obtained a value of 115 frames that are less than 30 FPS, and 185 frames greater or equal to 30 FPS, in 300 seconds per frame (see Figure 10). In order to obtain this data within the body tracking sensor, the frames do not have much dispersion, meaning that it works correctly in the estimation of 30 FPS (high quality - Kinect v2).

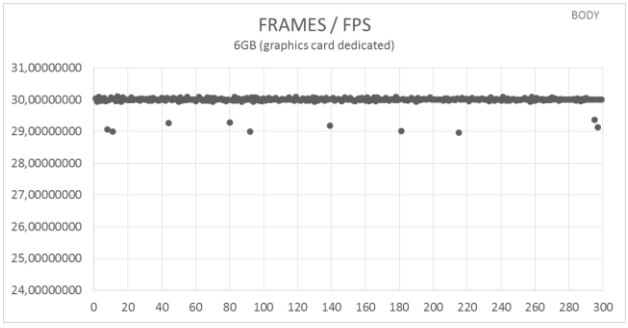

Figure 10. BODY TRACKING: Results in FPS - 6GB

The assessment of each FPS in the body tracking sensor, at 300 seconds per frame, establishes an output range compared to the 25 joints detected in a body. For the 6GB card the evaluation is similar to the data obtained for the depth sensor (see Figure 7), since for the recognition of the bodies it is applied for both color and depth sensors.

3.4 Real Data

To measure the use of the GPU with each of the sensors is evaluated based on the OpenCV libraries, which was designed to take full advantage of the real-time processing of applications. The libraries included for the test application are specifically selected to access graphic processing functions in opencv2: core (basic data structures), imgproc (image processing), highgui (video capture, images) and GPU (accelerated algorithms).

The results obtained are segmented by each of the graphic cards of 1GB, 2GB and 6GB respectively for the sensors (color, depth and body tracking). The evaluation is based on the use of the GPU (0-100%) and the determinant of the fall of each frame based on 30 FPS (high quality - Kinect v2).

For 1GB graphics processing card is obtained an evaluation of the FPS greater than 30 FPS in a time determined by the GPU, which present a total of 234 frames (see Figure 11).

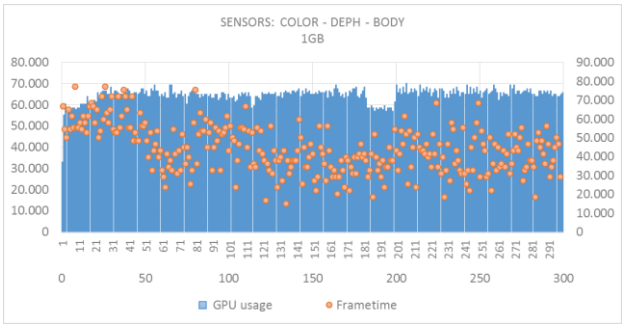

Figure 11. SENSORS: Results in FPS and GPU - 1GB

In the 2GB graphics card, the FPS greater than 30 FPS in a time determined by the GPU, present a total of 263 frames (see Figure 12).

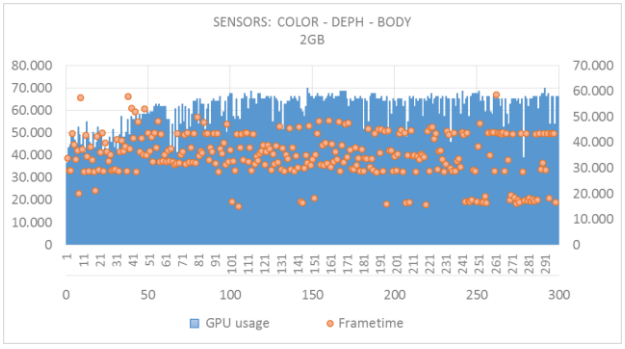

Figure 12. SENSORS: Results in FPS and GPU - 2GB

On the other hand, for the 6GB card the FPS greater than 30 FPS, determined by the GPU, present a total of 291 frames (see Fig. 13).

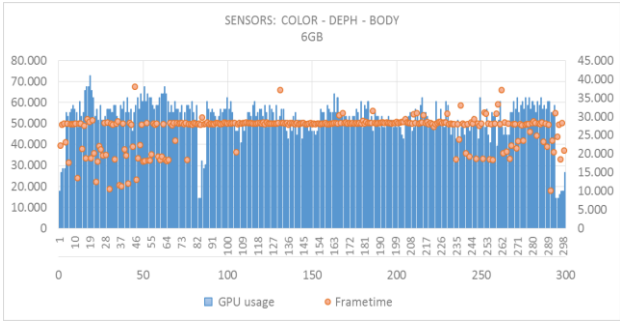

Figure 13. SENSORS: Results in FPS and GPU - 6GB

The results obtained using the OpenCV libraries, in the evaluation of color sensors, depth, and body tracking present a maximum use

of their evaluation parameters, such as performance based on processing frames and work on the GPU.

For the 1GB card it ranges from 20% to 80%, while the 2GB card GPU performance ranges from 20% to 60%, and for the 6GB card the GPU stabilizes its performance from 30% to 40%.

4. DISCUSSION OF RESULTS

The results of the basic revision of Kinect V2 using the native libraries, within the 6GB graphic card, with the sensors (color, depth, and skeleton), allow to discard the FPS of less than 30 (see Fig. 4, Fig. 10). Therefore, at 6GB the data loss is not very significant compared to the other cards.

In the application using a 1GB card and native Kinect v2 libraries, there was evidence a high data loss with values of 233 frames lower than 30 FPS, for the depth sensor.

The problems using 1GB and 2GB cards that support DirectX 11 or higher may cause loss of specific data, if it does not have an interpretation of results in FPS output (see Table 1).

Table 1. Frames captured (300)> = 30 FPS, using native Kinect V2 libraries

Cards Sensors	1GB (frames)	2GB (frames)	6GB (frames)
Color	67	183	173
Depth	62	103	179
Body Tracking	70	112	185
FPS	66	118	179

The work presents an alternative of graphic processing with OpenCV and as it is improved the obtaining of the frames.

Determining to work with these sensors, the performance is improving with the use of platforms adjusted to graphic processing and even using NVidia CUDA. The output data with OpenCV (see Table 2) show values higher than those obtained with native libraries. For sensors on the 1GB card, with 234 frames; in 2GB, with 263 frames; and for 6GB, with 291 frames greater or equal to 30 FPS compared to the values presented (see Table 1), are better. The representation of the FPS with higher outputs in equivalence of the estimated by Kinect V2 of 30 FPS, can match the performance of the card from 1GB to 2GB with OpenCV, respectively the 2GB to 6GB with this type of graphic processing libraries.

Table 2. Frames captured (300)> = 30 FPS, using graphic processing libraries

Sensors Cards	Frames per time	GPU use % Since – Until
1 GB	234	20 80
2 GB	263	20 60
6 GB	291	30 40

On the other hand, is obtained the use of the GPU, which is determined by the time in obtaining each frame. The use of the GPU ranges from a minimum value of 0% to a maximum of 100%, the GPU in unemployment is in processing from 0% to 10%. For the sensors, the use of the GPU is determined for this work between 20% and 80% (see Table 2), obtaining that the GPU processes more with the 1GB and 2GB cards, because they require it. While obtaining frames on the 6GB card, GPU usage does not exceed 40%, however, if the application so requests, is possible to send more load to the GPU for optimum performance.

5. CONCLUSIONS AND FUTURE WORK

The present work represents the values that the captured frames can take depending on the graphic card that is used, each valuation is adjusted to the requirements of the application.

When using the native Kinect V2 libraries does not mean that the test application runs with errors, perhaps the data is not precise, it will depend on the processes that the application has, to know what are the basic requirements that must be taken into account. The results improve if the libraries dedicated to graphic processing are implemented internally, regardless of the graphics card used.

As future work it is proposed to adapt these characteristics in a system of ergonomic risk assessment of the movements of people in a particular area of work, since the processing must be mostly accurate values, in order to give a correct evaluation result.

6. ACKNOWLEDGMENTS

The authors would like to thanks to the Technical University of Ambato (UTA) for financing the project ‘‘System of Evaluation of Postural Risk Using Kinect 2.0 in the activity of Cutting of the Production of Footwear for the Caltu Ambato”, for the support to develop this work.

7. REFERENCES

[1] Kinect for Xbox One | Xbox: http://www.xbox.com/en-US/xbox-one/accessories/kinect .

[2] Kinect - Windows app development: https://developer.microsoft.com/en-us/windows/kinect.

[3] Kinect hardware: https://developer.microsoft.com/en-us/windows/kinect/hardware.

[4] Kinect hardware setup: https://developer.microsoft.com/en-us/windows/kinect/hardware-setup.

[5] Mokhov, S.A., Song, M., Llewellyn, J., Zhang, J., Charette, A., Wu, R. and Ge, S. 2016. Real-time collection and analysis of 3-kinect v2 skeleton data in a single application. (Jul. 2016).

[6] Córdova-Esparza, D.M., Terven, J.R., Jiménez-Hernández, H. and Herrera-Navarro, A.M. 2017. A multiple camera calibration and point cloud fusion tool for Kinect V2. 143, (Sep. 2017), 1–8. DOI: https://doi.org/10.1016/j.scico.2016.11.004.

[7] Gao, T.S., Sheng, D.B., Nguyen, T.H., Jeong, N.S., Kim, H.K. and Kim, S.B. 2017. Measurement of the fish body wound depth based on a depth map inpainting method. (2017), 289–299.

[8] Kim, C., Yun, S., Jung, S.W. and Won, C.S. 2016. Color and depth image Correspondence for Kinect v2. (2016), 333–340.

[9] Yang, L., Zhang, L., Dong, H., Alelaiwi, A. and Saddik, A. El 2015. Evaluating and improving the depth accuracy of Kinect for Windows v2. 15, 8 (Aug. 2015), 4275–4285. DOI:https://doi.org/10.1109/JSEN.2015.2416651.

[10] Linder, T., Wehner, S. and Arras, K.O. 2015. Real-time full-body human gender recognition in (RGB)-D data. (Jun. 2015), 3039–3045.

[11] Owens, J.D., Houston, M., Luebke, D., Green, S., Stone, J.E. and Phillips, J.C. 2008. GPU computing. 96, 5 (May 2008), 879–899. DOI:https://doi.org/10.1109/JPROC.2008.917757.

[12] Ye, Q. and Gui, P.P. 2015. A new calibration method for depth sensor. 26, 6 (Jun. 2015), 1146–1151. DOI:https://doi.org/10.16136/j.joel.2015.06.0088.

[13] Zhang, S., He, W., Yu, Q. and Zheng, X. 2012. Low-cost interactive whiteboard using the Kinect. (2012), 38–42.

[14] Jafari, O.H., Mitzel, D. and Leibe, B. 2014. Real-time RGB-D based people detection and Tracking for mobile robots and head-worn cameras. (Sep. 2014), 5636–5643.

[15] Andaluz, V.H., Gallardo, C., Santana, J., Villacres, J., Toasa, R., Vargas, J., Reyes, G., Naranjo, T. and Sotelo, A. 2012. Bilateral virtual control human-machine with kinect sensor. (2012), 101–104.

[16] Sell, J. and O'Connor, P. 2014. The xbox one system on a chip and kinect sensor. 34, 2 (2014), 44–53. DOI:https://doi.org/10.1109/MM.2014.9.

[17] Jha, S. and Trivedi, P. 2013. An automated video surveillance system using Viewpoint Feature Histogram and CUDA-enabled GPUs. (2013), 1812–1816.

[18] Chuan, C.H., Chen, Y.N. and Fan, K.C. 2016. Human action recognition based on action forests model using kinect camera. (May 2016), 914–917.

[19] Yao, H., Ge, C., Xue, J. and Zheng, N. 2017. A high spatial resolution depth sensing method based on binocular structured light. 17, 4 (Apr. 2017). DOI:https://doi.org/10.3390/s17040805.

[20] Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S. and Fitzgibbon, A. 2011. KinectFusion: Real-time dense surface mapping and tracking. (2011), 127–136.

[21] OpenCV library: http://opencv.org/.

[22] [OpenGL - The Industry Standard for High Performance Graphics: https://www.opengl.org/.

[23] Unity - Manual: DirectX 11 and OpenGL Core: https://docs.unity3d.com/Manual/UsingDX11GL3Features.html.

[24] Fernández-Cervantes, V., García, A., Ramos, M.A., Méndez, A. and Méndez, A. 2015. Facial Geometry Identification through Fuzzy Patterns with RGBD Sensor. Computación y Sistemas. 19, 3 (Oct. 2015), 529–546. DOI:https://doi.org/10.13053/cys-19-3-2015.

[25] Allusse, Y., Horain, P., Agarwal, A. and Saipriyadarshan, C. 2008. GpuCV: An opensource GPU-accelerated framework for image processing and computer vision. (2008), 1089–1092.

[26] CUDA - OpenCV library: http://opencv.org/platforms/cuda.html.

[27] Procesamiento paralelo CUDA | Qué es CUDA | NVIDIA: http://www.nvidia.es/object/cuda-parallel-computing-es.html.

[28] Carraro, M., Munaro, M. and Menegatti, E. 2016. Cost-efficient RGB-D smart camera for people detection and tracking. 25, 4 (Jul. 2016). DOI:https://doi.org/10.1117/1.JEI.25.4.041007.

[29] Munaro, M., Basso, F. and Menegatti, E. 2016. OpenPTrack: Open source multi-camera calibration and people tracking for RGB-D camera networks. 75, (Jan. 2016), 525–538. DOI:https://doi.org/10.1016/j.robot.2015.10.004.

[30] Introduction — OpenCV 2.4.13.3 documentation: http://docs.opencv.org/2.4/modules/core/doc/intro.html.