International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 22 (2018) pp. 15883-15894


# Human-Machine Interaction using MYO-Kinect Tools

**Natalie Segura Velandia**
*Assistant research, Department of Mechatronics Engineering, Militar Nueva Granada University, Bogotá, Colombia.*

**Ruben Dario Hernandez**
*Professor, Department of Biomedical Engineering, Militar Nueva Granada University, Bogotá, Colombia.*

**Robinson Jimenez-Moreno**
*Professor, Department of Mechatronics Engineering, Militar Nueva Granada University, Bogotá, Colombia.*

**Abstract**

This article presents a robotic integration system, based on machine vision and acquisition of the signals for the Human-Machine interaction in the delivery and reception of tools. By using the Microsoft Kinect v1.0 sensor, it is possible to spatially locate the manipulator using the RGB camera to which the conversion of color spaces and filters will be applied to extract the XY coordinates of the markers that are located in the end effector and in the reference, with the laser incorporated the depth is obtained to extract the coordinates in the Z axis and thus spatially define the 3 coordinate axes in which the robot will act. Likewise, the human body recognition function is used to extract the right-hand coordinates in the workspace. Something similar happens with the integration of the MYO Armband sensor located on the right forearm, which was implemented for the capture of electromyography (EMG) signals, by means of its electrodes capable of providing a response signal to the gestures or movements that the hand makes indicating the delivery or reception of the tool. In this way, an interface was developed in Visual Studio 2012 under the programming language C#, where the algorithms of the processing and execution of each activity to be performed by the manipulator according to the signals given by the user are implemented.

**Keywords:** Robotics, image processing, 3D vision, pattern recognition, EMG signals, HMI system

## INTRODUCTION

During the last years, the human-machine interface (HMI) has evolved being the means by which the man interacts with a process controller (machine) to execute an action. To make this a success, various devices have been developed specialized in the recognition of human gestures, one of them has been the Microsoft Kinect sensor, representing the implementation of completely natural interfaces where the human body is transformed into the controller, this device allows users to provide commands to the machine through real-time body postures. From which have been derived works as in [1] for the recognition of hand gestures with real-time communication for certain situations in people with speech and hearing disabilities.

In addition to the above, the development of interactive systems in the health sector has provided a new vision for researchers to develop hardware and software focused on the health sector in order to support physicians and therapists to achieve success in their treatments, guiding patients through faster and more efficient rehabilitation processes. From this, a device capable of recognizing hand gestures has been identified from the reading of electromyographic (EMG) signals produced by the movement of the forearm muscles.

In [2] we present the control of two manipulators by means of EMG signals captured by a MYO Armband, to control the degrees of freedom of each one in order to achieve a joint work, where the signal acquisition is initially performed and each gesture is associated for each degree of freedom of the two manipulators. Finally, they perform a test scenario to validate the control of the manipulators.

In [3] it is presented the application of a robotic system capable of performing the task of collecting and storing objects for a warehouse, by using the Microsoft Kinect sensor for the recognition and location of each of the objects in a 3D space, The application of a robotic system, with the capacity to implement a Convolutional Neural Network for the recognition of each object, in addition to this, it has a path planning module capable of generating the paths so that the final effector reaches each object on the shelf and a calibration module that defines the physical space in which the robot is to perform.

In this way, the present article presents the integration of the MYO armband and Microsoft Kinect sensors for the development of an HMI system, between a robotic manipulator and a human operator for the reception and delivery of tools. Therefore, the article is organized as follows: In section 2, the description of the methods used to obtain the necessary information of each of the implemented equipment is presented, the algorithms implemented in machine vision, acquisition of electromyography signals and inverse kinematics of the manipulator are described. Section 3 presents the results obtained. Finally, section 4 presents the conclusions of the study.

## MATERIALS AND METHODS

For the implementation of the system, a methodology was developed to execute the human-machine interaction through the integration of MYO-Kinect for the tasks of collecting and delivering tools using a robotic manipulator of 4 degrees of freedom as shown in **Error! Reference source not found.**.

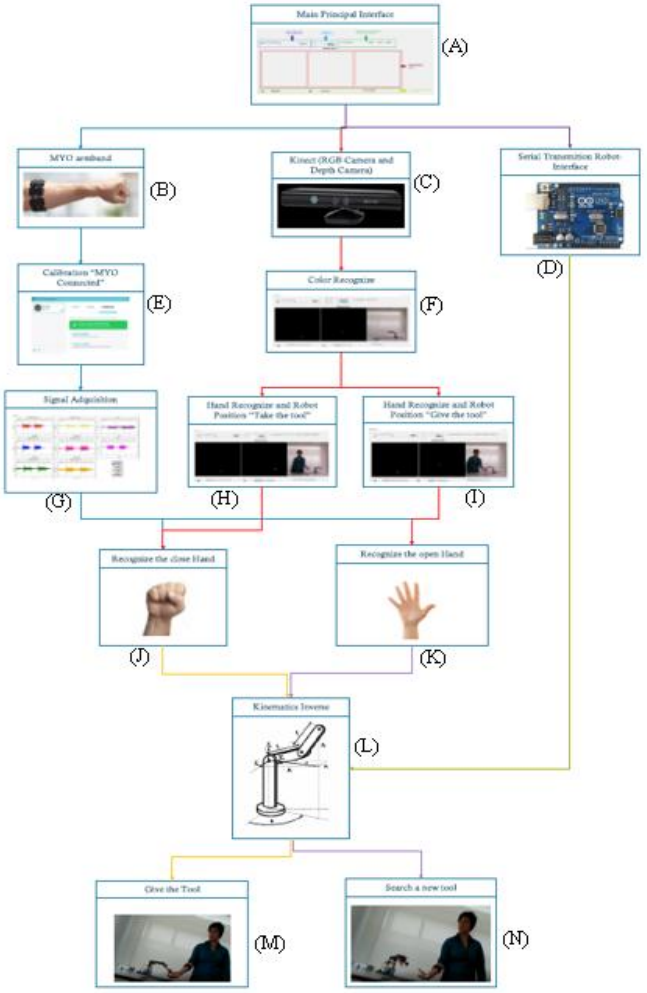



**Fig 1.** Block Diagram Program Operation: (A) Main Menu, (B) MYO Position, (C) Kinect Position, (D) Connect Arduino ONE Serial Communication, (E) MYO Calibration, (F) Color Recognition and Coordinates, (G) Acquisition of MYO signals, (H) Right Hand Recognition and coordinates (Get Tool), (I) Right Hand Recognition and Coordinates (Give Tool), (J) Closed Hand Recognition, (K) Open Hand Recognition, (L) Hand coordinates are the end points that calculate the inverse kinematics, (M) tool delivery and (N) Search a new tool.

### *A. Machine Vision System*

In relation to **Error! Reference source not found.** (C) it can be appreciated the methodology implemented for the development of the machine vision system. Initially, the location of the final effector should be identified with respect to the reference point to perform the relevant calculations. The identification system is based on the Microsoft Kinect sensor which is responsible for performing the image acquisition, while at the same time processing the image for color recognition as seen in **Error! Reference source not found.** (F), all thanks to its RGB camera, with a resolution of 640 x 480 pixels. Finally, it performs the acquisition of the right hand coordinates for the reception (**Error! Reference source not found.** (I)) and delivery (**Error! Reference source not found.** (H)) of tools, by means of its infrared laser and infrared camera with resolution of 640 x 480 pixels, able to generate a depth map that relates the distance to the objects of the device.

According to the above, a working space was developed that allows to execute the HMI system which can be seen in **Error! Reference source not found.** composed by a robotic manipulator, computer equipment and Kinect sensor.

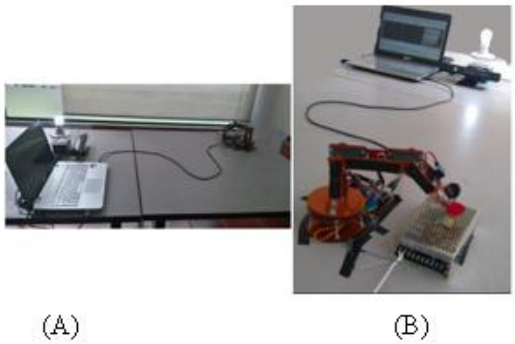



**Fig 2.** Work environment, (A) side view where it can be see that the Kinect has a light bulb behind it to ensure correct brightness and it is located at a distance of 1.05 meters from the Manipulator. (B) Rear view where the manipulator is observed with red markers at its end effector.

In order to perform the identification of the manipulator within the environment, circular and square markers are used as shown in **Error! Reference source not found.**, each identified by a characteristic color, which by means of image processing algorithms are possible to identify and differentiate. For the reference system a blue square is used in the bottom of the manipulator and for the second a red circle located in the gripper of the robotic arm.

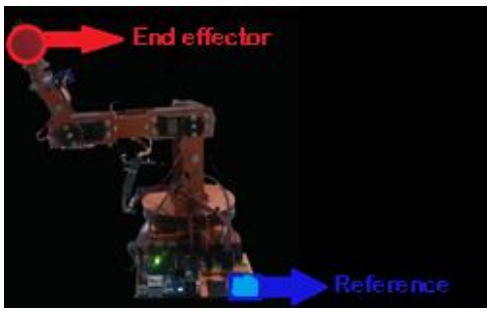



**Fig 3.** Robotic Manipulator with their respective markers

For color identification, a transformation is performed from RGB space to YCbCr, where Y represents luminance or

brightness, Cb and Cr represent the chrominance component, Cb is located on a scale between blue and yellow, and Cr is located on a scale between red and green, as shown in **Error! Reference source not found.**. A successful application of this algorithm is observed in [4] [5] where they perform the comparison between the red components in the face using RGB and YCbCr, where the Cr component highlights the red tones by means of a higher intensity level for lip recognition.

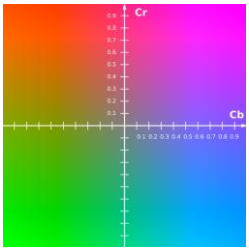



**Fig 4.** The CB-CR plane represented with constant luminance of Y = 0.5

Marker Dection

To simplify the image processing algorithm, the color segmentation is done with the help of the EmguCV library, belonging to the OpenCV multiplatform developed by Intel and compatible with C#, VB, VC and IronPython.

Initially the image from the RGB camera of the Kinect is captured to which the RGB space conversion is done to YCbCr by means of a system of equations (1) that allow the conversion of each pixel.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112 \\ 112 & -98.768 & -18.214 \end{bmatrix} * \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

Since the components Cb and Cr must be taken separately as each marker is independent, it must be highlighted according to the tonality of the red and blue scale that is handled in order to eliminate shades of red and blue different from those implemented, for this a threshold range of red and blue is made for the colors used. That is, several tests with different light conditions should be performed by extracting the results shown in Table I. With respect to the above it is important to emphasize that as long as the color is not within the range of thresholds or is a different color, the pixel will be placed in black, but if the pixel is in these threshold ranges the pixel will be placed in white.

Subsequently, several filters are implemented to highlight the pixels. First, the image passes through an implemented Median filter to soften the image and reduce the intensity variation between neighboring pixels by reducing noise. Second, the image passes through a Gaussian filter that allows to highlight the central pixel and, as it moves away, decreases the brightness value of its neighboring pixels, which gives a blurred appearance eliminating the noise. Third, the function is built in C # bwareaopen2, to remove objects containing more than 50 pixels, filling them with black color and redrawing the outlines of recognized figures containing less than 50 pixels.

Once the object has been determined, the coordinates of the markers must be obtained. Initially, it is necessary to start with the information of the contours of the figures and through the application of the library of EmguCV with the function approxpoly which uses the method of the chain of EmguCV to approach the polygon that is formed according to the contour, which is used to inscribe the figure in a rectangle and thus obtain the width, height and initial points in (x, y), initially used to determine the centroid where its measurements will be expressed in pixels and then with the help of the laser, infrared camera and SDK developed by the manufacturer the information of the depth or coordinate (Z) is obtained directly in millimeters as shown in Fig. **1**.

**Table I.** Cb and Cr work thresholds

| **Light conditions** | **Cb** | **Cr** |
|---|---|---|
| Day light | 6500-7500 | 9000-10000 |
| Led Lamp 4 Tubes 12W | 5000-6000 | 7000-8000 |
| 2 Energy saving bulbs 23W | 5500-6500 | 8000-9000 |

**Fig. 1** Detection of color and coordinates (X, Y) in pixels and (Z) in millimeters. On the left side blue color, on the right red color.

To perform the conversion of pixels to millimeters, first, a subtraction must be performed between the pixels of the reference marker and the marker to be evaluated, resulting in the number of pixels between markers, for which equation (2) is applied, where it is necessary to highlight the reference marker will be the new one (x = 0, y = 0), for example in Fig. **1** they are observed the measurements in pixels and the applying of the conversion (3), which is obtained by the reference marker of which the equivalent width and height measures in mm and pixels are known, as shown in Fig. **2**.

$$X_{EF(Pixel)} = X_{A(Pixel)} - X_{R(Pixel)} \quad (2)$$

Where:

$$X_{EF(Pixel)} = Difference\ between\ markers$$

$$X_{A(Pixel)} = Reference\ marker$$

$$X_{R(Pixel)} = End\ effector\ marker$$

$$X_{EF(mm)} = \frac{X_{EF(Pixel)} * 25mm}{15_{(Pixel)}} \quad (3)$$

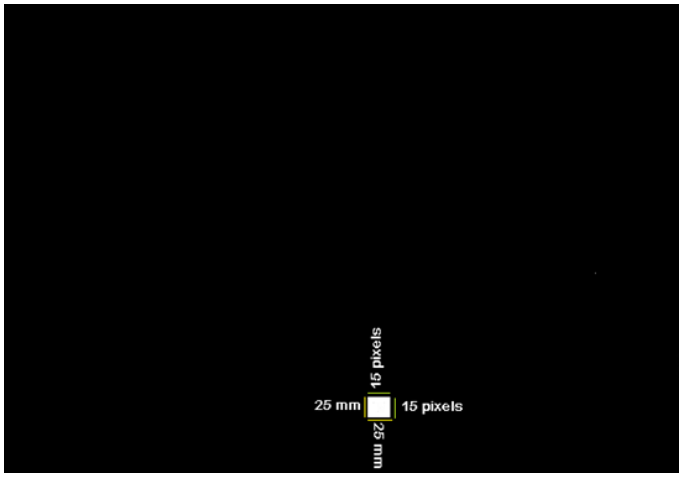



**Fig. 2** Reference system for converting pixels to millimeters

Fig. **3** shows how the coordinate system behaves after the conversions

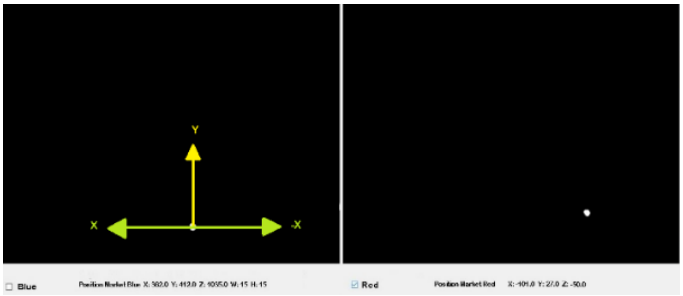



**Fig. 3** Conversion of pixels to millimeters

Coordinates of the right hand

Depth images are obtained with the use of the Microsoft Kinect sensor, SDK developed by Windows and the skeleton-tracking function capable of providing a three-dimensional capture of the body in a virtual skeleton that is formed from a set of 3D points that relate to body parts [6], recognizing a total of 20 main points as seen in Fig. **4**.

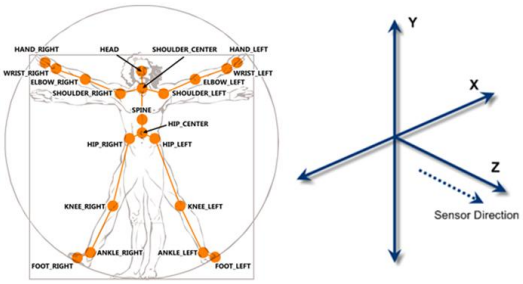



**Fig. 4** Joints that Kinect relates to the human body

The acquisition of this information from the SDK is organized in an array that starts with the number of frames per second (fps) followed by a numeric character (#) indicating the beginning of the coordinates (x, y, z) of the recognized skeleton. Fig. **5** illustrates the structure of the data coming from the SDK [7].

| fps | # | id | 0,215 0,314 1500 0,236 ... | # | id | 0,215 0,314 1500 0,236 ... | ff |
|---|---|---|---|---|---|---|---|

**Fig. 5** Skeleton-tracking data structure [7]

Fig. **6** shows the flow diagram for the acquisition of the right hand coordinates, where first it must be enabled the functions to recognize a skeleton and consequently acquire the information of the right hand joint. In this way, this can be seen in **Error! Reference source not found.** (H) and **Error! Reference source not found.** (I) where the right hand recognition and the capture of its coordinates are observed (x, y) in pixels and (z) in meters.

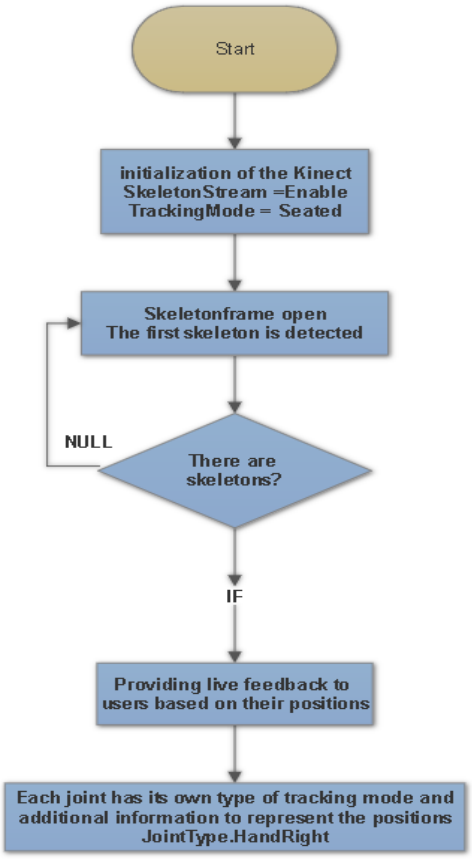



**Fig. 6** Flowchart for the recognition of the right hand .

Also, as in color recognition the same methods must be applied to convert the pixels to mm (3) and the z coordinate is multiplied by 1000 since its measurement is in meters, as the reference marker is the (0, 0, 0) the coordinate difference (2) is made.

*B. Acquisition of electrmyography (EMG) signals*

For the acquisition of electromyography signals, the MYO armband sensor developed by the company Thalmic labs is used. For the identification of movement of the arm muscles, the EMG signals are obtained by means of 8 biosensors (electrodes) located in the MYO, numbered as shown in

Fig. **8**, which, once acquired, are filtered and processed to eliminate noise generated by the environment or muscle at steady state. The MYO is made of a flexible and adaptable material that allows adjustment to any type of arm [8], as shown in Fig. **7**.

**Fig. 7** MYO Armband

In order to obtain a correct acquisition of signals through the MYO, it is necessary that every time the MYO is worn, it must be calibrated through the manufacturer's application "MYO Connect" as shown in **Error! Reference source not found.**(E), this is because each user has a different type of skin and size of muscles. From the above, internally a machine learning is performed for the recognition of gestures regardless of whether the person who places it is female or male, in addition to this, the MYO has a 9-axis inertial measurement unit (IMU) 9150 (3 axes for gyroscope, 3 axes for acceleration and 3 axes for compass), which allows the movement of the arm to be detected and the position in which it is located [9].


**Fig. 8** MYO Sensor Numbering

The MYO armband is used to recognize the movement of the muscles of the forearm after a gesture produced by the hand, being necessary that the electrodes of the sensor are in direct contact with the skin for a better acquisition of data EMG (microvolts) which are sent via a Bluetooth 4.0 module. Fig. **9** illustrates the correct way to locate the MYO, it is important to keep in mind that the logo is located between the ring and heart fingers.

**Fig. 9** Location of the MYO in the forearm

For the characterization of the movements, it must be taken into account that according to the location of the MYO in the arm will be the location of the sensors that will receive the information of the muscles. In this case, the MYO must be able to recognize the open hand (extension of the fingers) and the closed hand (flexion of the fingers), Fig. **10** shows the muscles that act on these gestures and in Table II the muscles and sensors of the MYO that interact with them are named.

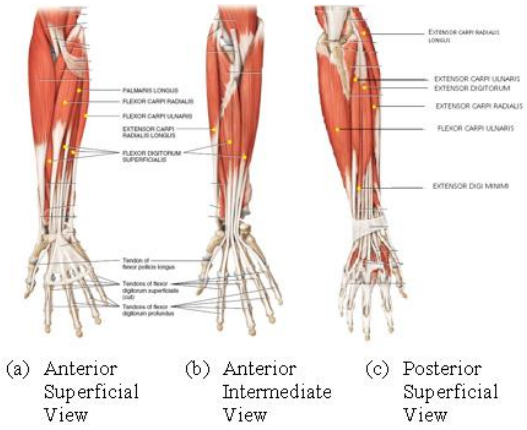



**Fig. 10** Muscles of the arm

**TABLE II.** MYO Sensors, Muscles and Actions Performed

| Myo Sensors | Muscles | Action |
|---|---|---|
| 8 | Flexor carpi ulnaris | Flexion and adduction wrist |
| 4 | Flexor Digitorum | Flexor of fingers |
| 1-8 | Flexor carpi radials | Flexion and adduction wrist |
| 5 | Extensor carpi radials | Extensor at the wrist joint, abducts the hand at the wrist |
| 3-4 | Extensor digitorum | Extension of fingers |
| 4-5 | Extensor carpi ulnaris | Extends and adducts the wrist |

Fig. **11** illustrates the methodology that was implemented for the acquisition of signals to the different gestures that were made.

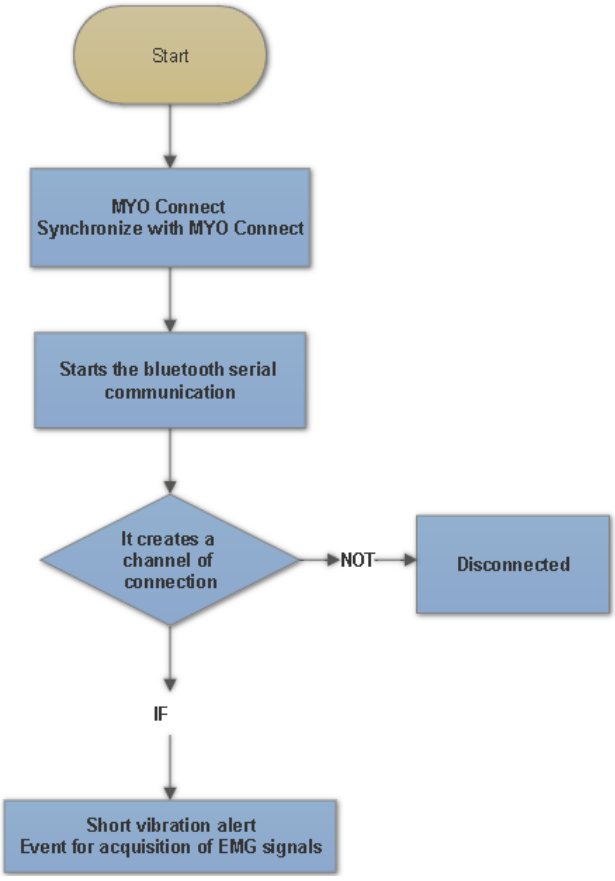
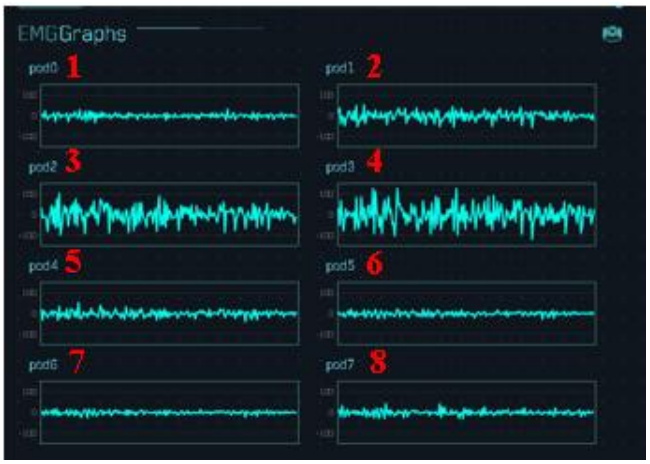


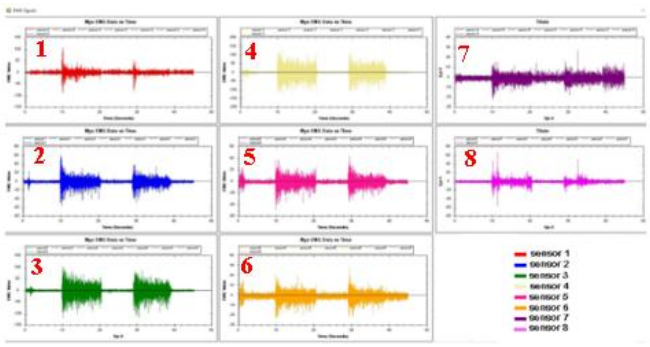

**Fig. 11** Signal acquisition flowchart

Coordinates of the right hand

For the choice of the sensors, the information provided when comparing two software was taken into account: First, "MYO Diagnostic", which is online and presented by Thalmic Labs (Fig. **12**(A)) and Second, "MYO Acquisition" developed by the authors for signal processing (Fig. **12**(B)).




(a)




(b)

**Fig. 12** Comparison of open hand MYO signals: (a) Acquisition of signals by "MYO Diagnostics", (b) Signal Acquisition developed by the authors.

Basically, Fig. **12** shows the variation of the sensors for the extension of the fingers. 2 seconds were extracted from the values acquired by the sensors, which can be seen in Table III, whereby sensors 3, 4 and 5 were chosen because they were the ones that obtained a higher signal strength.

Thus, the algorithm was developed under the following conditionals (4) and fed back to the user by a long vibration alert.

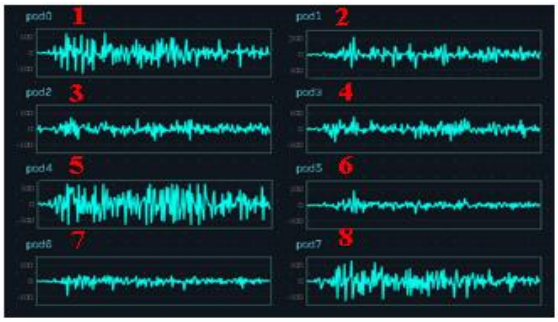
$$Sensor\ (3) > Sensor\ (4)\ \&\&\ Sensor\ (4) > Sensor\ (5)\ \&\&\ Sensor\ (3) > 90\ \&\&\ Sensor\ (4) > 50 \quad (4)$$

Closed hand

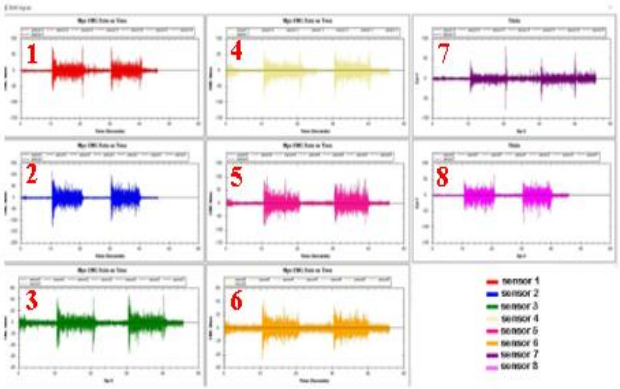
For the choice of the sensors, the information supplied was taken into account when comparing two software: "MYO Diagnostic", in Fig. **13**(A) and "MYO Acquisition", in Fig. **13**(B).

**Table III.** MYO sensors, "Open Hand" behavior

| MYO Sensors | Value (microvolts) |
|---|---|
| 1 | 0 |
| 2 | 4 |
| 3 | 40 |
| 4 | 22 |
| 5 | 4 |
| 6 | 1 |
| 7 | 2 |
| 8 | -3 |
| 1 | 3 |
| 2 | 4 |
| 3 | -29 |
| 4 | 10 |
| 5 | 6 |
| 6 | 2 |
| 7 | 1 |
| 8 | -1 |
| 1 | -3 |
| 2 | -9 |
| 3 | -14 |
| 4 | -37 |
| 5 | -11 |
| 6 | -3 |
| 7 | -5 |
| 8 | -1 |




(a)




(b)

**Fig. 13** Comparison of closed hand MYO signals: (a) Acquisition of signals by "MYO Diagnostics", (b) Signal Acquisition developed by the authors.

In general, Fig. **13** presents the variation of the sensors for flexion of the fingers. 2 seconds were extracted from the values acquired by the sensors, which can be seen in Table IV, whereby sensors 1, 5 and 8 were chosen because they were the ones that obtained a higher signal strength.

**Table IV.** MYO sensors, "Closed Hand" behavior

| MYO Sensors | Value (microvolts) |
|---|---|
| 1 | -18 |
| 2 | 6 |
| 3 | -6 |
| 4 | -4 |
| 5 | 25 |
| 6 | -3 |
| 7 | -17 |
| 8 | -59 |
| 1 | 12 |
| 2 | -2 |
| 3 | -10 |
| 4 | -1 |
| 5 | 15 |
| 6 | 10 |
| 7 | 5 |
| 8 | 25 |
| 1 | 22 |
| 2 | 8 |
| 3 | -3 |
| 4 | 23 |
| 5 | 62 |
| 6 | 17 |
| 7 | -14 |
| 8 | 80 |

Thus, the algorithm was developed under the following conditionals shown in (5) and feedback to the user via a medium vibration alert.

$$Sensor\ (1) > Sensor\ (5)\ \&\&\ Sensor\ (8) > Sensor\ (5)\ \&\&\ Sensor\ (1) > 40\ \&\&\ Sensor\ (8) > 50 \quad (5)$$

*C. Robotic manipulador*

In Fig. **14**, it is possible to observe the manipulator with its microcontroller (Arduino UNO) to be used in the application, in addition to this, its coordinate system for each joint is represented, from which Table V was obtained with the Denavit-Hartenberg parameters. From this the transformation matrix is constructed, allowing to know the values of the angles for each joint [10].

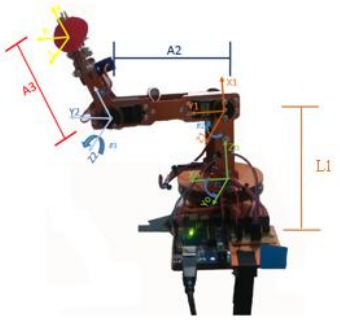



**Fig. 14** Structure of the manipulator with the representation of its coordinate system

**TABLE V.** Denavit-Hartenberg parameters

| Joint | $\theta$ | d | a | $\propto$ |
|---|---|---|---|---|
| 1 | $\vartheta 1$ | L1 | 0 | 90° |
| 2 | $\theta 2$ | 0 | A2 | 0 |
| 3 | $\vartheta 3$ | 0 | A3 | 0 |

Inverse Kinematics

To determine the movements that the manipulator must make to reach the final point (right hand position), it necessary to make use of the inverse kinematics, which will be solved by implementing the geometric method as shown in Fig. **15**, where the value of $\theta 1$ is found, and as in Fig. **16**, where the values of $\theta 2$ and $\theta 3$ are found.

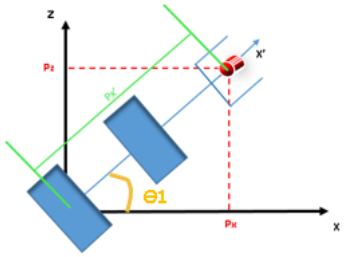



**Fig. 15** Top view to find Θ1

From Fig. **15** it is observed that to calculate the angle $\theta 1$ (7), tangent must be used, since Px and Pz are known points.

$$Px' = \sqrt{Pz^2 + Px^2} \quad (6)$$

$$\theta 1 = \tan^{-1}\left(\frac{Pz}{Px}\right) \quad (7)$$

To determine the angles of joints 2 and 3, the manipulator must be viewed from a side view to visualize the movements as shown in Fig. **16**.

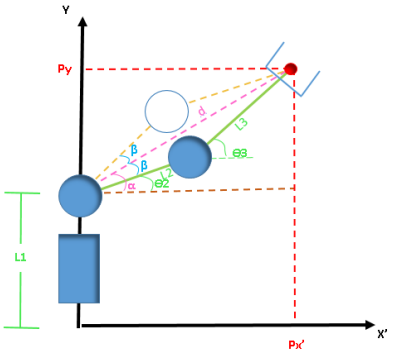



**Fig. 16** Side view to find Θ2 y Θ3

Then, geometric calculations are performed to determine the angles, where d is initially found that determines the distance that is formed after the movement of the angles $\theta 2$ and $\theta 3$. To find $\theta 3$ (11) the vector in its component in X must be decomposed by means of $cos\ \theta_3$ (9) and its component in Y by means of $sin\ \theta_3$(10).

$$d = \sqrt{(Py - L1)^2 + Px'^2} \tag{8}$$

$$\cos\theta_3 = \frac{d^2 - L_2^2 - L_3^2}{2 * L_2 L_3} \tag{9}$$

$$\sin\theta_3 = \pm\sqrt{1 - cos^2\theta_3} \tag{10}$$

$$\theta_3 = \tan^{-1}\left(\frac{\sin\theta_3}{\cos\theta_3}\right) \tag{11}$$

Finally, to find $\theta 2$ (15), which is a dependent angle of others as are the angles β (14) y α (12).

$$\propto = \tan^{-1}\left(\frac{Py - L1}{Px'}\right) \tag{12}$$

$$L_3^2 = L_2^2 + d^2 - 2L_2 d cos\beta \tag{13}$$

$$\beta = \cos^{-1}\left(\frac{L_2^2 + d^2 - L_3^2}{2L_2 d}\right) \tag{14}$$

$$\theta_2 = \alpha + |\beta| \tag{15}$$

## ANALYSIS OF RESULTS

Initially as shown in **Error! Reference source not found.** the first level corresponds to the initialization of all devices, but since Kinect is the device that more processes must begin to perform, at the moment of pressing "Start", the frames and events of the Kinect are initialized (RGB camera, depth camera and Skeleton function) and are available to receive data. Also, the MYO initializes, creates the channel to receive and send data, the manipulator acquires an initial position, prepares the communication to send data to the Arduino card.

As shown in Fig. **17**, color recognition and application of inverse kinematics are observed for coordinate and workspace tests of the manipulator.

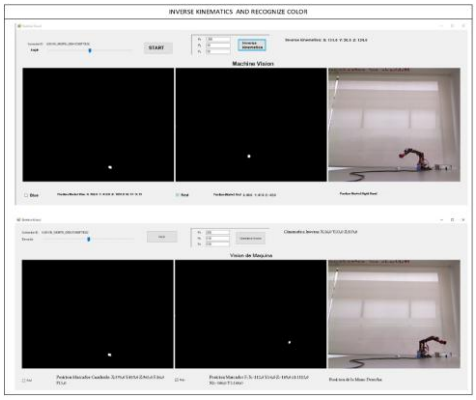



**Fig. 17** Recognition of color and application of inverse kinematics

First, the color recognition is performed, where the first panel from left to right recognizes the blue color and shows its coordinates when "Blue" is activated, after this, the "Network" is activated on the second panel for red color recognition and coordinates, in the third panel it can be observed the location of the manipulator and its markers in RGB camera.

Second, the application has a section to calculate the inverse kinematics by entering the points from the keyboard. This was used in order to determine the working cell in which the manipulator will act and to calculate the red marker position error rate see Table VI.

Third, recognition of the human body, more specifically the right hand joint, that is illustrated in Fig. **18**, which provides the information of the end points that are entered to the manipulator to reach this position. For this, the error calculation shown in Table VII is performed.

**TABLE VI.** Position error

| Coordinates | Real | Test | Error |
|---|---|---|---|
| X | 362 | 360 | 0.55% |
| | 418 | 415 | 0.72% |
| | 380 | 378 | 0.53% |
| Y | 412 | 408 | 0.97% |
| | 440 | 441 | -0.23% |
| | 425 | 426 | 0.94% |
| Z | 1020 | 1048 | -2.74% |
| | 1025 | 1050 | -2.43% |
| | 1030 | 1045 | -1.46% |

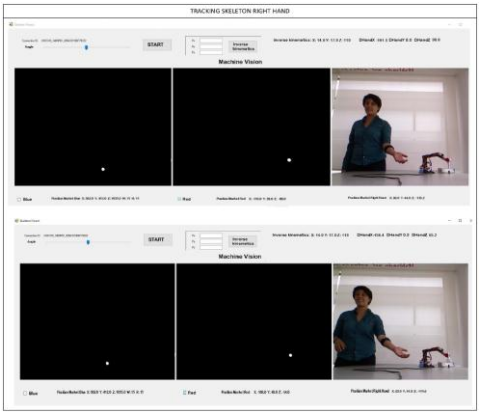



**Fig. 18** Tracking of the right hand and coordinates.

**TABLE VII.** Position error

| Coordinates | Real | Test | Error |
|---|---|---|---|
| X | 252 | 250 | 0.79% |
| | 240 | 243 | -1.25% |
| | 235 | 233 | 0.85% |
| Y | 402 | 406 | -0.99% |
| | 390 | 387 | 0.77% |
| | 384 | 380 | 1.04% |
| Z | 1075 | 1095 | -1.86% |
| | 1080 | 1107 | 0.92% |
| | 1085 | 1101 | -1.94% |

Fourth, the tests are performed with the interaction between MYO-manipulator, with the sensors chosen to recognize each gesture. Closed hand activates the action of closing gripper (catch tool), and open hand activates the action of opening gripper (release tool). An example of each action is illustrated in Fig. **19** and Fig. **20** respectively.

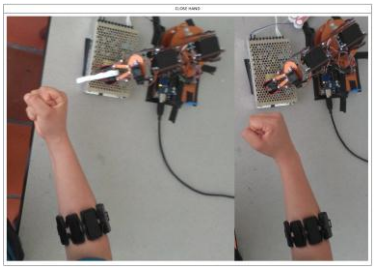


**Fig. 19** Closed hand activates the gripper closing action

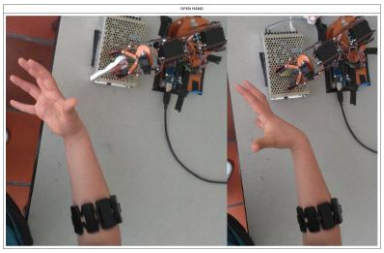


**Fig. 20** Open hand activates the gripper opening action

Fifth, after performing several tests, the delivery of the tool was successfully obtained in different positions as shown in Fig. **21**, where the algorithm previously described is seen functioning, with the detection of the reference marker (Blue, First panel from left to right), end effector marking (Red, second panel), right hand recognition (Third panel), acquisition of hand coordinates for the calculation of inverse kinematics and recognition of the gesture using the MYO sensor.

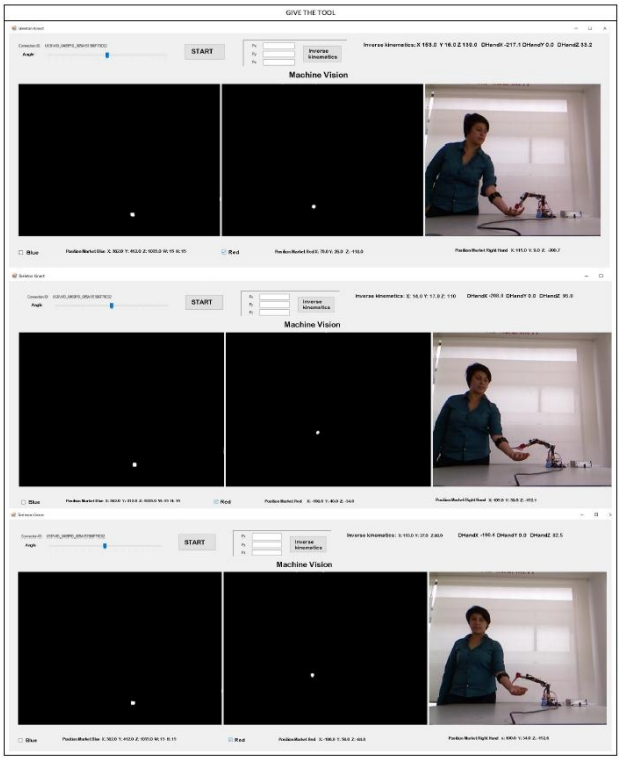
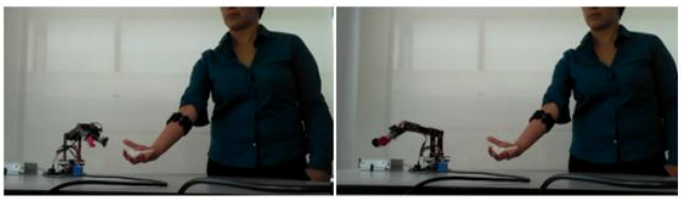



**Fig. 21** Human-machine interaction with position variation

Finally, after delivering the tool (open hand) and being in the place where the hand is, it must return to its initial position in search of a new tool as shown in

Fig. **22**.



(A) (B)

**Fig. 22** (A) Delivery of Tool, (B) Return to initial position

## CONCLUSIONS

It was possible to implement an algorithm capable of performing the human-machine interaction through machine vision techniques and acquisition of electromyography signals for the delivery of tools, so that a recognition of successful

gestures is set by feedback the user through a vibration that allows him to identify if the sensor captures the gesture made. It is necessary to emphasize that it is necessary that when the movement is generated a tension is generated in the muscles.

Given the type of algorithm that was implemented for color recognition (YCbCr) and uncontrolled lighting spaces for the Kinect sensor, it is necessary to perform the algorithm in a space with good illumination that allows to establish an optimal connection to minimize the error of depth in the recognition of color and human body.

Depending on the results obtained, it was possible to minimize the depth error by making a controlled space of illumination and highlighting the luminance for each color. Given the algorithm developed and the programming structure that is implemented it is possible to use another MYO device to extend the number of movements and characterize them for the choice of different types of tools.

## ACKNOWLEDGEMENT

The authors are grateful to the Nueva Granada Military University, which, through its Vice chancellor for research, finances the present project with code IMP-ING-2290 (2017-2018) and titled "Prototype of robot assistance for surgery", from which the present work is derived.

## REFERENCES

[1] Y. Li, «Hand Gesture Recognition using Kinect,» *Software Engineering and Service Science, IEEE 3rd International Conference ,* pp. 196-199, 2012.

[2] P. Useche Murillo, R. Jimenez Moreno y O. Aviles Sanchez, «Individual Robotic Arms Manipulator Control Employing Electromyographic Signals Acquired by Myo Armbands,» *International Journal of Applied Engineering Research,* vol. 11, nº 23, pp. 11241-11249, 2016.

[3] S. Kumar, A. Majumder, S. Dutta, S. Jotawar, A. Kumar, M. Soni, V. Raju, E. Hassan, L. Behera, K. S. Venkatesh y R. Sinha, «Design and Development of an automated Robotic Pick & Stow System for a e-Commerce Warehouse,» *arXiv preprint arXIv: 1703.02340,* 2017.

[4] R. Jimenez Moreno y F. A. Prieto Ortiz, «Segmentación de labio mediante técnicas de visión de maquina y análisis de histograma,» *NGE@ UAN-Tendencias en la Ingeniería,,* pp. vol. 2, no 4., 2012.

[5] R. Jimenez Moreno, O. Aviles Sanchez y D. Amaya Hurtado, « Driver distraction detection using machine vision techniques.,» *Ingenieria y competitividad,* vol. 16, nº 2, pp. 55-63, 2014.

[6] W. Zeng, «Microsoft Kinect sensor and its effect,» *IEEE multimedia,* vol. 19, nº 2, pp. 4-10, 2012.

[7] R. Jimenez Moreno, «Tracking Humano mediante kinect para control de robots,» *Revista Clepsidra,* vol. 10, nº 19, pp. 107-112, 2015.

[8] B. Stern, «Adafruit,» [En línea]. Available: https://learn.adafruit.com/myo-armband-teardown?utm_source=youtube&utm_medium=videodescrip&utm_campaign=wearables. [Último acceso: 02 07 2017].

[9] M. Sathiyanarayanan y T. Mulling, «Map Navigation using hand gesture recognition: a case study using MYO connector on Apple Maps,» *Procedia Computer Science,* vol. 58, pp. 50-57, 2015.

[10] P. Useche Murillo, R. Jimenez Moreno y M. Mauledeox M., «Multi User Myographic Characterization for Robotic Arm Manipulation,» *International Journal of Applied Enginnering Research,* vol. 11, nº 23, pp. 11299-11304, 2016.

[11] Carlsson G (2009) Topology and data, vol 46. DOI 10.1090/S0273-0979-09-01249-X, arXiv:1312.6184v5

[12] Chan JM, Carlsson G, Rabadan R (2013) Topology of viral evolution. Proceedings of the National Academy of Sciences 110(46):18,566{18,571, DOI 10.1073/pnas.1313480110, URL http://www.pnas.org/cgi/doi/10.1073/pnas.1313480110

[13] Edelsbrunner H, Harer J (2008) Persistent Homology a Survey. Contemporary Mathematics 0000:1{26, DOI 10.1090/conm/453/08802

[14] Ghahramani Z (2004) Unsupervised Learning BT - Advanced Lectures on Machine Learning. Advanced Lectures on Machine Learning 3176(Chapter 5):72{112, DOI 10.1007/978-3-540-28650-9 5, URL http://link.springer.com/10.1007/978-3-540-28650-9fn g5fn%g5Cnpapers3://publication/doi/10.1007/978-3-540-28650-9fn g5, 1512.00567

[15] Kajaree D, Behera R (2017) A Survey on Machine Learning: Concept, Algorithms and Applications. International Journal of Innovative Research in Computer and Communication Engineering 5(2):1302{1309, DOI 10.15680/IJIRCCE.2017.

[16] Kleinberg J (2002) An impossibility theorem for verisimilitude. NIPS pp 446{453

[17] Kwitt R, Huber S, Niethammer M, Lin W, Bauer U (2015) Statistical Topological Data Analysis - A Kernel Perspective. Advances in Neural Information Processing Systems pp 3070{3078, URL http://papers.nips.cc/paper/5887-statistical-topological-data-analysis-a-kernel-perspective

[18] Lum PY, Singh G, Lehman A, Ishkanov T, Vejdemo-Johansson M, Alagappan M, Carlsson J, Carlsson G (2013) Extracting insights from the shape of complex

data using topology. Scienti_c Reports 3(1):1236, DOI

[19] 10.1038/srep01236, URL http://www.nature.com/articles/srep01236

[20] Mitchell TM (2006) The Discipline of Machine Learning. Machine Learning 17(July):1{7, DOI 10.1080/026404199365326, URL http://www-cgi.cs.cmu.edu/f_gtom/pubs/MachineLearningTR.pdf, 9605103

[21] Munkres JR (1984) Elements of Algebraic Topology

[22] Nicolau M, Levine AJ, Carlsson G (2011) Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. Proceedings of the National Academy of Sciences 108(17):7265{7270, DOI 10.1073/pnas.1102826108, URL http://www.pnas.org/cgi/doi/10.1073/pnas.1102826108, arXiv:1408.1149

[23] O_roy M, Duponchel L (2016) Topological data analysis: A promising big data exploration tool in biology, analytical chemistry and physical chemistry. Analytica Chimica Acta 910:1{11, DOI 10.1016/j.aca.2015.12.037, URL http://dx.doi.org/10.1016/j.aca.2015.12.037

[24] P Murphy K (1991) Machine Learning: A Probabilistic Perspective. DOI 10.1007/SpringerReference 35834, URL http://link.springer.com/chapter/10.1007/978-94-011-3532-0fn g2, 0-387-31073-8

[25] Padellini T, Brutti P (2017) Supervised Learning with Indefinite Topological Kernels pp 1{16, URL http://arxiv.org/abs/1709.07100, 1709.07100

[26] Patania A, Vaccarino F, Petri G (2017) Topological analysis of data. EPJ Data Science 6(1):0{6, DOI 10.1140/epjds/s13688-017-0104-x

[27] Reininghaus J, Huber S, Bauer U, Tu M, Kwitt R (2017) A Stable MultiScale Kernel for Topological Machine Learning 1412.6821

[28] Rucco M, Mamuye AL, Piangerelli M, Quadrini M, Tesei L, Merelli E (2016) Survey of TOPDRIM applications of topological data analysis. CEUR Workshop Proceedings 1748:1814

[29] Sarikonda G, Pettus J, Phatak S, Sachithanantham S, Miller JF, Wesley JD, Cadag E, Chae J, Ganesan L, Mallios R, Edelman S, Peters B, Von Herrath M (2014) CD8 T-cell reactivity to islet antigens is unique to type 1 while CD4 T-cell reactivity exists in both type 1 and type 2 diabetes. Journal of Autoimmunity 50:77{82, DOI 10.1016/j.jaut.2013.12.003, URL http://dx.doi.org/10.1016/j.jaut.2013.12.003

[30] Sheehy D (2014) (Multi)Filtering Noise for Geometric Persistent Homology pp 1-4

[31] Singh G (1991) Mapper : A Topological Mapping Tool for Point Cloud Data pp 182

[32] Singh G, Memoli F, Ishkhanov T (2008) Topological analysis of population activity in visual cortex. Journal of . . . 8(8):1{28, DOI 10.1167/8.8.11. Topological, URL http://jov.highwire.org/content/8/8/11.short

[33] Talwar A, Kumar Y (2013) Machine Learning: An arti_cial intelligence methodology. International Journal of Engineering and Computer Science 2(12):3400{3405, URL http://ijecs.in/issue/v2-i12/11ijecs.pdf

[34] Wasserman L (2016) Topological Data Analysis. ArXiv e-prints 1609.08227

[35] Zhu X, Vartanian A, Bansal M, Nguyen D, Brandl L (2016) Stochastic multiresolution persistent homology kernel. IJCAI International Joint Conference on Artificial Intelligence 2016-Janua:2449{2455