Skeleton Tracking Using Microsoft Kinect for Windows on Matlab

[1] Ayushi Prajapati, [2] Meet Bhatt, [3] Yash Joshi, [4] Rohit Negi.

[1], [2], [3] B.Tech. Student, Navrachana University, Vadodara.
[4] Assistant Professor, Navrachana University, Vadodara.
Email [1] ayeshap2665@gmail.com, [2] meetbhatt96.mb@gmail.com [3] yshjshi9@gmail.com [4] rohitn@nuv.ac.in

Abstract*— Kinect V2 for Windows uses time of flight and structured light pattern to acquire 3D word data which can be utilized by digital image processing for various application such as Human body skeleton tracking. Kinect for Windows is a low cost and accurate device to acquire such 3D data. In this paper, the brief review of Kinect sensor and image acquisition process for skeleton tracking is discussed. Acquired data is processed on MATLAB, Hence XYZ coordinates are extracted from the real-time 3D world. In the end, future of Kinect for evolving NUI in various fields are discussed.

Index Terms— ***Kinect V2, MATLAB, Image Acquisition Toolbox, Skeleton tracking.***

I. INTRODUCTION

The processing of Three-dimensional data majorly contributes in research area for modern applications in field of engineering, specifically Mechatronics. Being a motion detection and recognition sensor, Kinect was developed by Microsoft engineers to permit human-computer handless interaction. Kinect is comprised of two cameras designed for image acquisition, an RGB camera for capturing colors and a laser camera for depth [1] [2]. The processor inside the Kinect recognizes movement patterns and generates corresponding skeleton model of human that is provided in a computer environment such as MATLAB, assisting in developing new algorithms and methods [2] [3]. Gaming platform has been served by Kinect majorly during its initial stage. In recent times, it has not only evolved itself as a perfect motion gaming device but also allows developers to develop daily applications with a very economical price. The Core elements of Kinect are circumscribed by easy and ready to use functionalities: capturing images of 1920x1080 pixels or depth images of 525x424 at 30 fps. Furthermore, the sensor is also enriched by IR array which identifies depth images of surrounding at a blistering accuracy of 4 m.

Continuous records are compromised in the API (Application User Interface). The sensor does not guarantee a similar skeleton of an individual at multiple times usage. Hence, to substantiate this, a way is waited for maintaining an individual identity. According to Mohammad Al-Shabi et al [4], Microsoft Kinect sensor has revealed its competence to the research community as not just an interactive gaming device, but a device with multi-functional abilities and high reliability. In this work, online HIL(Hardware-in-the-Loop) experimental data are used to apply human motion imitation to a 2-degree of freedom Lego Mind storm NXT robotic arm.

II. WORKING PRINCIPLE FOR MEASURING DISTANCE IN KINECT

Kinect possess a special case of 3D acquisition systems which works on the concept of light coding.
Understanding its operation necessitates several preliminary notions, like the working principle of pin-hole camera model, camera projection matrix and triangulation. Amidst the two principles normally used for evaluating the time of Flight – Pulse wave modulation and Continuous wave modulation, Microsoft Kinect exploits Continuous Wave Modulation.

III. CONTINUOUS WAVE MODULATION

In general, modulation can be defined as a process of transforming a baseband signal (message) to another signal called modulated waveform. In modulation, the baseband signal is denoted as the modulating signal. In case of Continuous wave modulation, continuous light waves are exploited instead of short light pulses and modulation occurs in terms of frequency of sinusoidal waves. Kinect has cameras that emits light waves; Infrared cameras. The emitted waves after getting incident on the 3D surface are detected as reflected waves by the detector having shifted phases proportional to the distance from the reflecting surface. The phase shift is retrieved by demodulating the received signal, using the method of cross correlation of received signal with emitted signal [5].

Emitted signal with ω as the Modulation Frequency is given as: $g(t) = \cos(\omega t)$

Received signal after reflection from 3D surface:

$$s(t) = b + a\cos(\omega t + \emptyset)$$

b = constant bias, a = amplitude, $\emptyset$= phase shift

Cross-correlation of both signals:

$$c(\tau) = s * g = \int_{-\infty}^{+\infty} s(t) * g(t+\tau)dt$$

τ = offset

On simplification: $c(\tau) = \frac{a}{2}\cos(\omega t + \emptyset) + b$

Sampling this at four sequential instants with different phase offsets, obtaining the sought parameters, we get the formula to evaluate distance between the emitter-detector and the 3D surface [5]:

$$d = \frac{c}{4\pi\omega}\emptyset.$$

IV. POINT CLOUD ACQUISITION

Point Cloud Acquisition Sensors based on the digital imaging technology helps in measuring the distances between 3D surface and Sensor. These distance values are directly stored in a matrix whose size resembles the depth sensor resolution. Surely, for each pixel of the 512x424 depth images, the computing device estimates in real-time a distance value to the equivalent object point. This output data is the so-called depth map, a two-dimensional image from which the 3D-coordinates of the scene can be calculated. After a few post-processing steps, it is then probable to obtain indirectly point clouds of the object captured. If Kinect sensor uses a standard lens, camera intrinsic parameters such as focal length f and principal point coordinates (x, y) can be determined where the Z-coordinate for a pixel is already known as it corresponds to the value stored in the depth map. With these parameters and the perspective projection relationship, a 3D point 'X' in the camera coordinate system can be mapped from the homogenous image coordinates of a pixel p = [u, v, 1]. As each pixel of the depth map represents a value for the corresponding point cloud, the calculated point clouds count several points [6]. Until now only a minute number of algorithms allows an off-the-shelf use of the sensor. However, the Microsoft SDK (Software Development Kit) offers multiple functions for the recording of the different data types. Relating to the fact that we don't regulate the mapping functions used by Microsoft to produce point clouds from depth maps, we developed our own acquisition protocol. Implemented in C# through some code samples available in the SDK, this program permits us to acquire timed acquisitions for each of the different possible output data (RGB images, infrared images, depth maps, and colorized point clouds).

V. KINECT SKELETON TRACKING WITH MATLAB

KinectV2 can detect up to 6 users at a time with a high accuracy of detecting total 25 body joints at a maximum distance of 4.5 meters.

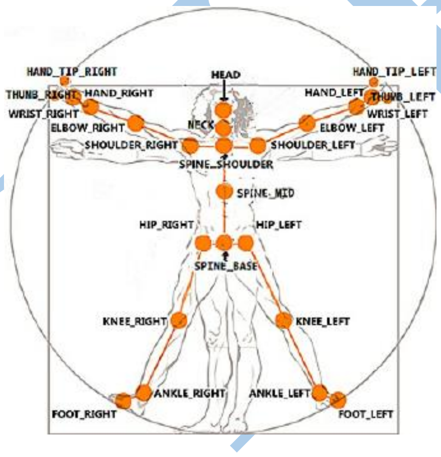

Figure 1: 25 Body Joints

As shown in figure 1, Linking these joints frames overall body index. Developing applications based on body tracking using Kinect also enables users to perform some advanced tracking i.e. tracking of Hand fingers and Eye-tracking along with full body tracking in real time. KinectV2 works on windows 8, 8.1 and 10 systems having USB3.0 support and installed Kinect SDK version 2.0. MATLAB 2013 or Higher versions with installed KinectV2 support packages.

VI. IMAGE ACQUISITION TOOLBOX

Image acquisition toolbox is significant for developing any Image processing application i.e. Motion recognition using any frame capturing device with MATLAB. It has built-in apps which enable users to configure Hardware and Software properties of device configured for Image acquisition. This software is the combination of functions defined for performing some specific task such as Different Hardware triggering, data logging, Multiple devices synchronizations, USB3 vision, C code support etc.

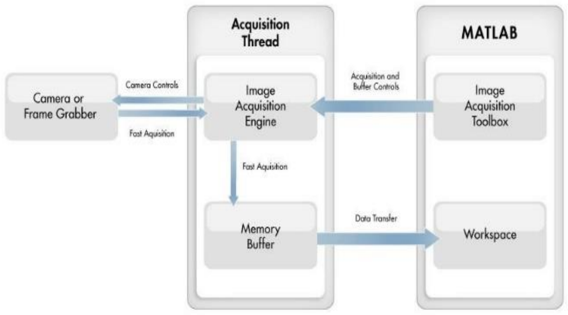

Figure 2: Image acquisition process [7]

Figure 2 shows a basic process of acquiring multiple frames from input device and frames are logged into workspace from memory buffer for further Image processing techniques. Simultaneously Image acquisition toolbox manages memory buffers by controlling hardware triggering and logging of an Input device.

VII. LOGGING AND TRIGGERING PROPERTIES FOR KINECT.

Kinect is capable of streaming RGB, depth and Infrared vision at the very instant also, skeleton tracking on basis of depth data. Skeleton tracking can have both RGB and depth data requiring more memory buffers hence more workspace in MATLAB. kinectV2 is having a color resolution of 1920x1080 pixels and depth resolution of 512x424 pixels so acquiring both streams for gesture recognition will lead to higher memory requirement. For such condition, it is essential to control kinect logging and triggering properties for data acquisition so that skeleton tracking can be acquired for limited number of frames.

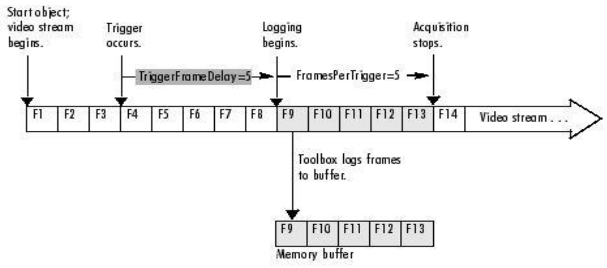

Figure 3: Configuring Logging and Triggering properties [7]

As shown in figure 3, once Kinect is turn ON, multiple frames start arriving on memory buffers from Kinect sensor. For grabbing desired frames, Trigger must be initiated by the user using FramesPerTrigger property or for the user to get in position in front of the camera a specified time delay can be provided using TriggerFrameDelay property. If there is no such delay at the instant of trigger command being executed, the frames now will start logging to memory buffers hence get loaded into MATLAB workspace which are actual frames required to perform skeleton tracking for gesture recognition. The number of Frames and Triggering type may vary depending on computer hardware configuration and Kinect triggering compatibility respectively. The trigger will terminate automatically after a specified number of frames gets logged into memory buffers.

VIII. FLOWCHART AND CODE BASICS

In this section the elementary flow chart and code basics for coding Kinect for skeleton tracking is discussed. There are various software available which shares different API with Kinect for emerging Interactive applications and simulations. MATLAB has Computer vision support along with Image Acquisition Toolbox with pre-defined functions. Which makes MATLAB easy, ready to use platform for beginners.

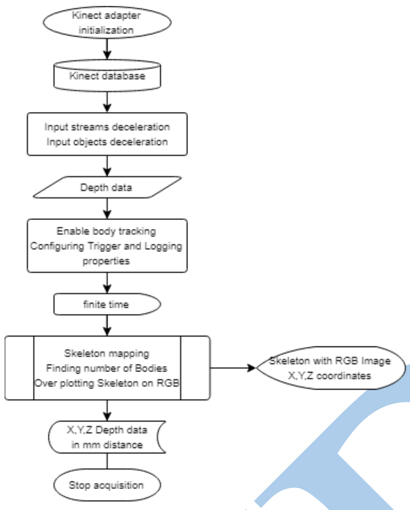

Figure 3: Flowchart for Skeleton tracking

As shown in Figure 3, To exploit Kinect Source (or libraries) for RGB and Depth data Kinect must be detected by Acquisition toolbox and few processing steps must be executed.

To know available adapters for Image Acquisition; "imaqhwinfo('Kinect')" function is used.

For skeleton tracking input object declaration is require for Kinect to recognize object (In reference frames) for tracking. Here as skeleton being overlay upon a RGB Image, two separate frames are generated. Utilizing Kinect camera ID '1' for color frames and '2' for depth frames.

To create an object for tracking;
"videoinput('adaptor', ID)" is used.
(Note that user is not allowed to change resolution)

As both frames are available, depth data must be definite specifically for Body tracking to enable Kinect performing skeleton tracking.

To store depth data;
"getselectedsource(depthdata)"
To enable Skelton tracking from depth data;
"EnableBodyTracking" function with 'ON' state.

As discussed in section VII there should be finite number of frames as per hardware compatibility. For reliable operation 100 Frames per trigger is sufficient for acquiring both color and depth data.

To set trigger properties;
"FramesPerTrig" property along defined color and depth object is used.

After stipulating number of frames some definite time delay is specified for user to get in position in front of camera sensor before starting color and depth objects to get stored in database for skeleton mapping.

To store color and Depth data into database;
"getdata(color or Depth)"
(Note that this function will vary and formed individually for both streams as RGB frames must be overlaid by skeleton mapping).

For tracking body, Kinect has "SkeletonConnectionMap" function. Which consists of matrix array structure for plotting body index from 25 defined joints. Numbers per joints may differ for different APIS'. Using MATLAB file explorer, the body joints are listed under Kinect examples. As skeleton is mapped, for acquiring precise outcomes it is advisable to obtain skeleton data from frames before trigger ends so that human motion error could avoid in last frames.

Taking case of Overlaying skeleton having 25 joints on RGB Image, using "Hold on" looping for N (defined number of user) = 2, each joint can be over potted on each other without repositioning any other joint in 3D space as they are having different X, Y, Z coordinates positions assigned based on human body. The overlapping loop will persist till 25 joints get over potted and will repeat for N=2 times. Using "Hold off" looping will terminate. For multiple skeleton detection, multiple colors can be allocated.

IX. RESULTS

Acquired skeleton mapping over the RGB Image can be observed with Image preview window in MATLAB. Distance of a user from Kinect sensor in 3D space i.e. skeleton joints is measured by Kinect in millimeters. Measurements along Z (the focal point of a camera) axis as depth distance, (X, Y) as the 2D position of a joints in definite frames.

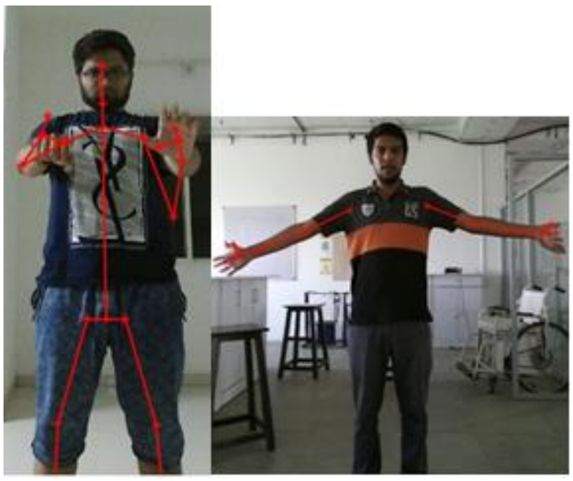

Figure 4: Skeleton tracking with a RGB Image

Figure 4 shows skeleton tracking of the human body which can be modified for the hand tracking or any other part of body depending upon application requirements. To avoid frequent logical errors for skeleton tracking there are some precautions that should be followed as mentioned below;

- The number of frames should be finite. (otherwise, it will give an error of exceeding Physical memory i.e. RAM).
- In absence of human motion (i.e. Human standing still in front of Kinect camera) while acquiring depth data for skeleton tracking, the output will have an only RGB image because for Kinect to detect motion the phase displacement of pixels i.e. movement of body joints is required.
- The number of frames to be acquired must vary in accordance with the type of triggering method.

X. CONCLUSION AND FUTURE SCOPE

We have described a model study of tracking of Human skeleton with IR time-of-flight range camera of the Kinect. After calibration we got useful measured data and thence a region of interest was defined. Interpretation of data was done using the principle of point cloud acquisition. Using this method, we managed to acquire elementary depth and RGB data for obtaining the positioning of body joints in 3D space. The distance between body joints from camera is displayed in form of 3-dimensional matrix in the command window for 100 frames per trigger on fixed 30 fps configuration. Data collected from Skeleton tracking with RGB Image, requires 1.5 GB of workspace which creates physical limitation of tracking skeleton for longer period. The results are observed to be more precise in range of 2 to 4.5 m.

From business to the arts, from education to gaming, and beyond, NUI expands the horizons of application development. Few illustrations of that with Kinect are gesture-controlled hologram technology, interactive surface and virtual clothing forms a huge opportunity for more operative and inexpensive motion tracking device. Kinect combined with augmented reality will unleash the potentials for human-machine interaction, evolving Self-Learning robots in advanced manufacturing industries. These advance robotic arms can be implemented in hazardous radioactive work field for human safety. Kinect also got fame in Bio-Medical technology as being able to detect human body fall, breathing, temperature and heart rate [8] [9].

XI. REFERENCES

[1] Microsoft, "Project Natal," 2009. [Online].

[2] Y. Li, "Multi-Scenario Gesture Recognition using Kinect," *The 17th International Conference on Computer Games, Louisville*, vol. 30, pp. 126-130, 2012.

[3] K. M. P. A. P. A, "Multi camera System in the Moving Body Recognition," *48th International Symposium ELMAR Focused on Multimedia Signal Processing and Communications*, pp. 45-48, 2006.

[4] Z. C. X. J. X. N. J. Y. a. L. W. M. Al-Shabi, Devlopement of a Directional 3D camera for Robot Navi, 2012.

[5] V. A. C. Zeman, Constructive Interference for Multi-view Time-of-Flight acquisition, 2012.

[6] P. Z. a. G. C. C. D. Mutto, Time-of-Flight Cameras and Microsoft Kinect, January 23, 2013.

[7] M. W. Inc., "Mathworks.com," MATLAB, October 2016. [Online]. Available: https://www.mathworks.com/help/pdf_doc/imaq/imaq_ug.pdf. [Accessed September 2017].

[8] S. Gasparrini, ""A Depth-Based Fall Detection System Using a Kinect® Sensors," *Sensor*, vol. 14, pp. 2766-2768, 2014.

[9] A. Procházka, "Microsoft Kinect Visual and Depth Sensors for Breathing and Heart Rate Analysis," *Sensor*, vol. 16, no. 966, pp. 2-10, 2016.

[10] M. Fitzpatrick, "Real Time Person Tracking," Major Qualifying Project in Electrical & Computer, Worcester, 2013.

[11] m. pajor, "kinect sensor implementation in fanuc robot manipulation," *archives of mechanical technology and automation*, vol. 34, no. 3, pp. 36-38, 2014.

[12] T. V. L. A. A. K. Asma Ben Hadj Mohamed, "A help for assisting people based on a depth cameras system dedicated to elderly and dependent people," *Journal of Biomedical Engineering and Medical Imaging*, vol. 1, no. 6, pp. 56-66, 2015.

[13] D. P. M. Z. K. C. N. S, "Static hand gesture recognition system for device control," *International Journal of Electrical, Electronics and Data Communication*, vol. 3, no. 4, pp. 2-3, 2015.

[14] M. Al-Shabi, ""Simulation and Implementation of Real-Time Vision-Based Control System for 2-DoF Robotic Arm Using PID with Hardware-in-the-Loop," *Intelligent Control and Automation*, vol. 6, pp. 147-157, 2015.

[15] N. J. G. R. Mihail, "Static Hand Gesture Recognition with 2 Kinect Sensors," Lexington, KY, USA, 2012.

[16] S. ERAP, " Gesture Based PC Interface with Kinect Sensor," Estonia, 2012.