ISSN 2395-1621

Controlling Robotic Buggy Using LEAP Motion Controller & Hand-Gestures

[#1]Sunil Gurnale, [#2]Harsh Panicker, [#3]Pawan Jaiswal,
[#4]Prof. Atul Mokal

[#1]sunil.gurnale@gmail.com

[#1234]ISB&M School of Technology,
Pune India.

ABSTRACT

These days robots are being used widely to perform tasks that a human being does in his day-today life. The main aim of project is to create an interaction between human hand & robotic vehicle (buggy). Buggy is to be controlled by using human hand gestures with the help of leap motion controller. This idea introduces the technology used to control the buggy (robotic vehicle) for different activities. We are using a leap motion controller to capture the hand gestures, and then it is processed on computer system. This motion system also allow user to define users own gesture set to access system utility. Leap motion device allow user to work in 3D that also increase the experience quality of system user. In our project we are using real time application, in that actions performed by our hands act as signals to guide buggy to move left-right etc.

Keywords: LEAP Motion Controller (LMC), LEAP SDK, Buggy(Robotic Vehicle), Gesture Recognition, Natural User Interface, HCI.

ARTICLE INFO

Article History

Received: 19th January 2017

Received in revised form :

19th January 2017

Accepted: 22nd January 2017

Published online :

22nd January 2017

I. INTRODUCTION

Nowadays, interacting with computer system has become very easy, it is because of interfacing devices like-touch screen, mouse, keyboards etc. In our project we are going to use LEAP motion Controller for interaction between human and computer system, this we are going to achieve by taking input as Human hand gestures through LMC(Leap Motion Controller),and then using LEAP SDK we are going to assign a unique task for each unique hand-gesture so that according to these tasks Robotic Buggy would be moving left-right etc.

The topic we are concerned with here is the gesture control of a robotic buggy. Gestures refers to any bodily motion or states particularly any hand motion or face motion[1].In our implementation, the LMC is used for recognition of gestures ,which are motion of hand, and as a result we can control the motion of the Robotic Buggy by simple gestures from the human hand[2,3].

II. OVERVIEW

1. LMC (LEAP MOTION CONTROLLER)

The LMC is a gesture recognition device which uses advanced algorithms for such operations. From an hardware perspective , the Leap Motion Controller is an eight by three centimetre unit, which comprises of two stereo cameras & three infrared LEDs[4].The two stereo cameras as well as the three infrared LEDs perform the function of tracing the infrared light having a wavelength of about 850 nanometres, which is outside the visible light spectrum.

Stereoscopy using the two stereo cameras, allows the Leap to minimize the errors and provide up to point precision while tracking hand and finger gestures. The field of view or the FOV of the Leap Motion controller is 150 degrees vertically and 120 degrees horizontally or sideways [4, 5]. The wide angle lenses contribute to its large field of view. The frame-rate of the Leap Motion Controller is less than 200 frames per second and it has a precision of about 1/100 mm per finger [6].

The Leap Motion Controller has the LEAP SDK or the LEAP Software Development Kit which is the software running on the ground station. Hence, due to this the LEAP is able to distinguish and differentiate the movement of human hands in front of it.

The Leap Motion Controller has a range to about 3 feet directly above it as shown in Fig.1. This range is limited by LED light propagation through space, since it becomes harder to recognize the position of the moving hand in 3-

diminsion beyond a certain distance. LED light intensity is ultimately limited by the maximum current which can be drawn from the USB via which it is connected to the ground station.

Fig.1.Leap Motion Controller

2. GESTURE RECOGNITION

Gesture recognition is a topic in Computer Science and Language Technology with the goal of interpreting human gestures via mathematical algorithms. Gestures can originate from any bodily motion or state but commonly originate from the face or hand.

Users can use simple gestures to control or interact with devices without physically touching them. Many approaches have been made using cameras and computer vision algorithms to interpret sign language. However, the identification and recognition of posture, gait, proxemics, and human behaviours is also the subject of gesture recognition techniques.[9]Gesture recognition can be seen as a way for computers to begin to understand human body language, thus building a richer bridge between machines and humans than primitive text user interfaces or even GUIs (graphical user interfaces), which still limit the majority of input to keyboard and mouse.

Gesture recognition enables humans to communicate with the machine (HMI) and interact naturally without any mechanical devices. Using the concept of gesture recognition, it is possible to point a finger at the computer screen so that the cursor will move accordingly. This could potentially make conventional input devices such as mouse, keyboards and even touch-screens redundant.

2.1 GESTURES 2D VS 3D

Different ways of tracking and analysing gestures exist, and some basic layout is given is in the diagram below. For example, volumetric models convey the necessary information required for an elaborate analysis, however they prove to be very intensive in terms of computational power and require further technological developments in order to be implemented for real-time analysis. On the other hand, appearance-based models are easier to process but usually lack the generality required for Human-Computer Interaction.

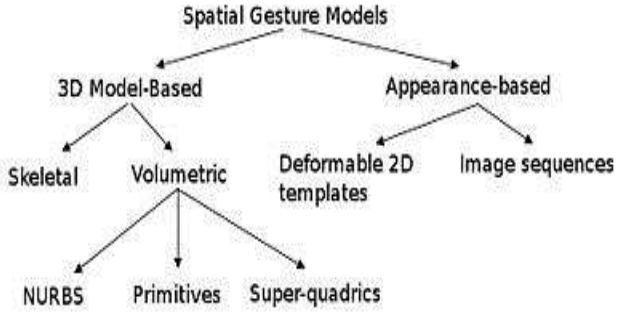

Fig.2.1 Modelling Different Gesture

2.1.1 3D MODEL-BASED

The 3D model approach can use volumetric or skeletal models, or even a combination of the two. Volumetric approaches have been heavily used in computer animation industry and for computer vision purposes. The models are generally created from complicated 3D surfaces, like NURBS or Polygon Meshes.

The drawback of this method is that is very computational intensive, and systems for real time analysis are still to be developed. For the moment, a more interesting approach would be to map simple primitive objects to the person's most important body parts (for example cylinders for the arms and neck, sphere for the head) and analyse the way these interact with each other. Furthermore, some abstract structures like super-quadrics and generalised cylinders may be even more suitable for approximating the body parts. The exciting thing about this approach is that the parameters for these objects are quite simple. In order to better model the relation between these, we make use of constraints and hierarchies between our objects.

2.1.2 SKELETAL-BASED

Instead of using intensive processing of the 3D models and dealing with a lot of parameters, one can just use a simplified version of joint angle parameters along with segment lengths. This is known as a skeletal representation of the body, where a virtual skeleton of the person is computed and parts of the body are mapped to certain segments. The analysis here is done using the position and orientation of these segments and the relation between each one of them(for example the angle between the joints and the relative position or orientation)

Advantages of using skeletal models:

- Algorithms are faster because only key parameters are analyzed.
- Pattern matching against a template database is possible
- Using key points allows the detection program to focus on the significant parts of the body

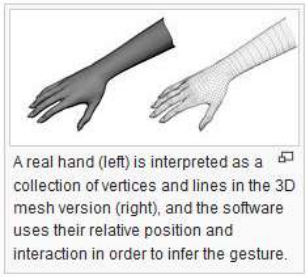

Fig.2.1.1 3D Model-Based

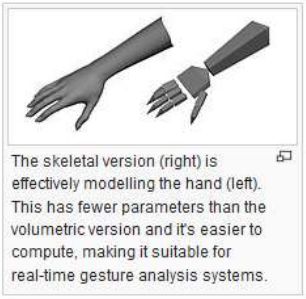

Fig.2.1.2 Skeletal-Based

2.1.3 APPEARANCE-BASED MODELS

These models don't use a spatial representation of the body anymore, because they derive the parameters directly from the images or videos using a template database. Some are based on the deformable 2D templates of the human parts of the body, particularly hands. Deformable templates are sets of points on the outline of an object, used as interpolation nodes for the object's outline approximation. One of the simplest interpolation function is linear, which performs an average shape from point sets, point variability parameters and external deformations. These template-based models are mostly used for hand-tracking, but could also be of use for simple gesture classification.

A second approach in gesture detecting using appearance-based models uses image sequences as gesture templates. Parameters for this method are either the images themselves, or certain features derived from these. Most of the time, only one (monoscopic) or two (stereoscopic) views are used.

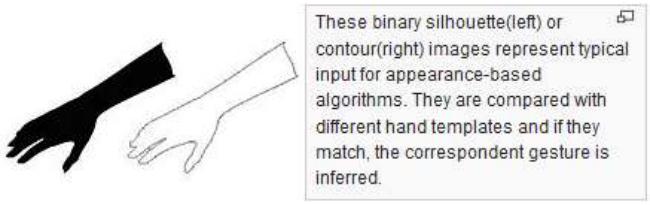

Fig.2.1.3 Appearance-Based

2.1.4 SINGLE CAMERA OR 2D CAMERA

A standard 2D camera can be used for gesture recognition where the resources/environment would not be convenient for other forms of image-based recognition. Earlier it was thought that single camera may not be as effective as stereo or depth aware cameras, but some companies are challenging this theory. Software-based gesture recognition technology using a standard 2D camera that can detect robust hand gestures.

2.2 TYPES OF GESTURES

There are mainly two methods or types of gesture recognition as,

1. Static (aka posture, posses..)

Fig 2.2.1 Shows some Static Gestures, like-first image in Fig 2.2.1 can be assigned for gesture representing alphabet A & so on according to user comfort.

Fig.2.2.1 Recorded Static Gestures For Evaluation

2. Dynamic(a sequence of postures/positions)

Fig.2.2.2 Shows some examples of Dynamic Gestures for swapping, circling, tapping etc. Also these can be set according to user comfort.

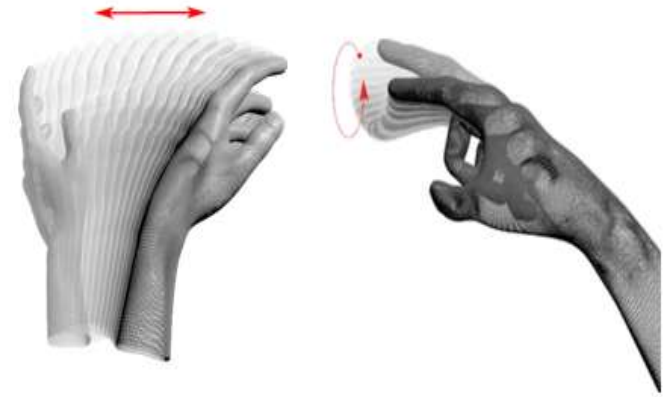

Fig.2.2.2 Recorded Dynamic Gestures For Evaluation

2.3 EVOLUTION OF HUMAN COMPUTER INTERACTION

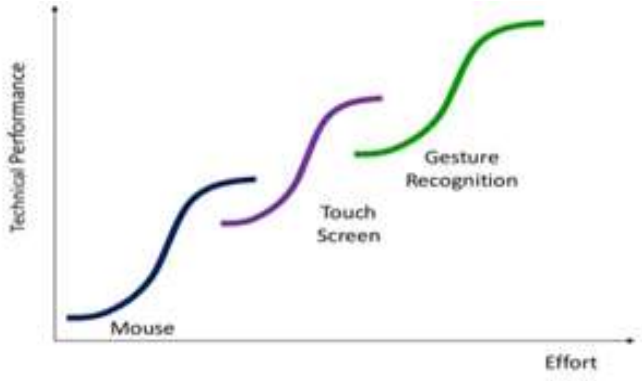

Fig.2.3 HCI Evolution

Above Figure shows how HCI has evolved in Past decades.

III. PROPOSED SYSTEM

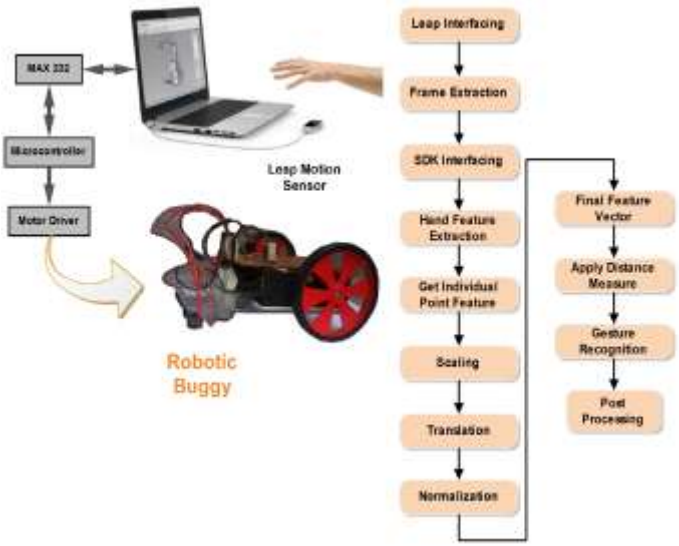

Fig.3 Flow Of Working

IV. WORKING

1. GESTURE SENSING & REPRESENTATION

Leap SDK provides built-in classes representing real-world object seen by the controller. The basic data unit we get from Leap Motion is a Frame. Frame contains objects like Hands and Pointables (Fingers and Tools), described by features directly related real attributes. Hand is an object representing a regular human hand. It contains Fingers, and is described by three dimensional values, like: position of center of hand, normal vector and direction vector (pointing from the center to the end of fingers). Pointables are objects like Fingers or Tools (which are longer and thinner than Fingers). Both are described by the same set of features: position of tip, pointing direction vector, length and width. All positions are expressed in millimeters, relative to position of controller which is always located in the center of the 3D space. Manufacturer claims, that the accuracy of device is about 0:1mm. Experiments shown results better than 0:2mm, using an industrial robot moving an object in controllers' field of view. This is more than enough, as accuracy of positioning human hand is about 0:4mm [16].

2. PROCESSING - MACHINE TO MACHINE INTERFACE

After gestures are sensed system assigns unique tasks to that gestures using LEAP SDK. When tasks are assigned user can then make use of these tasks to give commands to robotic buggy to make movement.
So in this first machine would be ground station i.e. computer system connected to LEAP motion via USB, & second machine would be Robotic Buggy that would be performing various kinds of tasks given by user (human).

3.ROBOTIC BUGGY

A Robotic Buggy is programmable ,with similar functions of a vehicle/car. This type robotic buggy can be used to perform any type of application such as moving objects from one place to another etc. The robotic buggy contains wheels that are used to make movement when user gives some gesture. Wheels are connected to motors that are connected to programmable chip mounted on Buggy. Different functions corresponding to each meaningful hand gesture are written and stored in database for controlling the robotic Buggy. Whenever a gesture is matched with a meaningful gesture from the database, the instruction set corresponding to that gesture is identified and passed to robot for execution. In this way the robotic system can be controlled by hand gesture using live 3D-camera.

V. MATHEMATICAL MODEL

Set Theory :
Let s (be a main set of) ≡ { SDB, LDB, C, A, S, MR, AO } where,

SDB is the copy of the server database. This database is responsible for storing user information related to cloud interactions. (Elaborate..)

LDB is a set of local database that a user owns. It consists of data tables having data items related to the products and their sales transactions. (Elaborate..)

C is a set of all clients using the server database and mining services from the server. And (c_1 , c_2 , c_3, ...c_n) Є C. (elaborate..)

A is a set of algorithms applied on the input data to get mining results. (Elaborate..)

S is the server component of the system. The server is responsible for registering, authenticating and providing associations to the end user. (Elaborate..)

MR is a set of mining rules that are applied on the input dataset provided by the client from his LDB. And (mr_1 , mr_2 , mr_3, ...mr_n) Є MR (elaborate..)

AO is a set of associations that are extracted from the input and a form the output of the system. (Elaborate..)

Functionalities :
SDB' = RegisterUser(uid, password, fullname, address, country, contact, email);

password = SHA1(input_password);

U = AuthenticateUser(uid, password, SDB');

LDB1 = ManageProducts(pid, product name, cost);

LDB2 = ManageBilling(transactions, items);

LDB = LDB1 + LDB2

ED(Encoded data) = EncodeTransactions(LDB2, EncodingAlgorithm(EA));

UPLOAD(ED);

AO = Apply Mining(ED);

Results = Decode(Download(AO));

IV. CONCLUSION

With the help of the LEAP Motion Controller, we have been able to move the Robotic Buggy by using hand motion. The Robotic Buggy responds to any hand gesture and moves accordingly. We have been able to make the Robotic Buggy move according to certain hand gesture which is again recognized by the 2 stereo cameras and 3 IR LEDs. Hence, it can be concluded that with the help of the Leap Motion Controller, we can use the Robotic Buggy to perform various tasks such as moving objects, performing surveillance tasks, to name a few. The Leap can be taught to recognize more hand gestures and movements by altering the programming language and adding more functionality to it.

REFRENCES

[1] Zheng Tong, and Jinghui Chu, "Dynamic Hand Gesture Recognition With Leap Motion Controller", in IEEE Signal Processing Letters, Vol. 23,No. 9,September 2016.

[2]Ayanava Sarkar, Ketul Patel, Ganesh R.k., and Geet Capoor, "Gesture Control of drone Using a Motion Controller", IEEE (2016).

[3]Alsheakhali, Mohamed, Ahmed Skaik, Mohammed Aldahdouh, and Mahmoud Alhelou. "Hand Gesture Recognition System." Information & Communication Systems 132 (2011).

[4]Bhuiyan, Moniruzzaman, and Rich Picking. “Gesture-controlled user interfaces, what have we done and what’s next.” Proceedings of the Fifth Collaborative Research Symposium on Security, E-Learning, Internet and Networking (SEIN 2009), Darmstadt, Germany. 2009.

[5]Singha, Joyeeta, and Karen Das. "Hand gesture recognition based on Karhunen-Loeve transform." arXiv preprint arXiv:1306.2599 (2013).

[6]Silva¹, Eduardo S., et al. "A preliminary evaluation of the leap motion sensor as controller of new digital musical instruments." (2013).

[7]Kainz, Ondrej, and František Jakab. "Approach to Hand Tracking and Gesture Recognition Based on Depth-Sensing Cameras and EMG Monitoring." Acta Informatica Pragensia 3.1 (2014): 104-112.

[8]Han, Jihyun, and Nicolas Gold. "Lessons Learned in Exploring the Leap Motion TM Sensor for Gesture-based Instrument Design." Proceedings of the International Conference on New Interfaces for Musical Expression. 2014.

[9]Matthias Rehm, Nikolaus Bee, Elisabeth André, Wave Like an Egyptian – Accelerometer Based Gesture Recognition for Culture Specific Interactions, British Computer Society, 2007

