International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 10, October 2015
Available at http://internationaljournalofresearch.org

A Gesture Learning Interface for Simulated Robot Path Shaping With a Human Teacher

Matteda Deepa
M.Tech student Department of Electronics & Communication Engineering SR Engineering College Warangal, India
mattedadeepa@gmail.com

A. Rajeshwar Rao
Asst. Professor Department of Electronics & Communication Engineering SR Engineering College Warangal, India
rajeshwarrao432@gmail.com

Abstract
The main aim of this project a Gesture Learning Interface for Simulated Robot Path Shaping With a Human Teacher Here in this project we are using gesture for the controlling of robot We have designed a research platform for a perceptually guided robot, as s demonstration for a coming generation of service robots. In order to operate semi autonomous, these require a capacity for learning about. The majority of robots in use today perform only a very limited. In this paper by gesture technology we are using different types of accelerometers. Automatically we can control the robot by showing the hand gesture. MEMS technology can do robot in different applications and by that we can communicate and also we guide the robot. Here we having web based technology also and by using capturing of a picture by the directions the conditions will be applied. So the people by this they can control automatically by giving guidance. By using ARM controller we are doing the project and it is advanced technology.

Keywords—components; pulse sensor; ultrasonic sensor; temperature sensor; Arm board

I. INTRODUCTION

Now a day's the robotic has the fundamental key for new invention. The development of human-machine communications on an everyday basis has made the people to utilize the technology. Instead of giving rational methodology physical methods have been welcomed by everyone. Coding to some 100's of pages requires more instance, capital and power so to overcome that gesture recognition is enhanced. Using gesture recognition coding can be easily made by everyone. For gesture recognition many active devices such as a "trackball, wiimote, joystick and touch tablet" are in practice1. Some of the devices are used for giving motion recognizer but gesture recognition has the foremost is extensively used. Gesture can be captured by wearing gloves or having wrist band attached with the MEMS whereas using vision system and data glove is very expensive hence not utilized. To have a balance of precision data collection, "Micro Inertial Measurement Unit" is developed for recognizing the gestures in 3 utility. So gesture recognizer like accelerometers with 3-axes dimensional axis x, y, z. Gesture can be recognized by following approaches comprises of "template-matching, arithmetical toning, vocabulary lookup, linguistic matching, and neural arrangement".

But in this paper the gesture recognition models are based on the signal succession3 and pattern toning. The gesture values are mapped by extracting as implementing characteristic based on signal succession of acceleration, for achieving high efficiency and accuracy. For this type of methodology the MEMS accelerometer is used to give the hand gestures. MEMS acronym micro electro mechanical system which has 3 axis of x, y, z and a power supply port with ground is fabricated6. MEMS use the knowledge which is known as "micro-fabrication knowledge".

The enlargement of micro technology has many Features like size, efficiency and capital. For a large scale device micro fabrication is used because of its smallness, applicability and less usage of material. Micro technology and electronics have great scope of innovation. MEMS can be mounted on the ARM LPC microcontroller wirelessly. A is the microcontroller proposed in this methodology which has inbuilt ADC conversion. By using MEMS many succession of external components can be eliminated. "Gaining data, storage of data, data filtering, statement interfacing and networking" are included hence it is called elegant included MEMS. MEMS equipment not only makes the utensils much lesser but also makes them much improved. The chief inspiration for this investigation is to make the human-robot interface more flexible and simpler for the user static and of low fidelity, with accommodation solely the result of the form and placement of furniture and fixtures. Smart homes aim to extend awareness, increase control over systems, and enhance the security, healthfulness, and safety of the environment through sensing, inference, communication technologies, decision-making algorithms, and appliance control.

However, the real-time processing of occupant activity has historically been costly in terms of computing and Memory requirements and often relies on technologies considered intrusive of people's privacy (e.g., cameras). As a result, these efforts have focused on systems associated with the built environment such as the design and placement of furniture and fixtures. Practical occupant sensing in smart homes remains of low fidelity including such ON/OFF sensor activations as room changes, door openings/closings, appliance actuations, etc.

A logical progression for the use of high fidelity sensing may be seen in its central importance to assistive robotics. As Green and Walker describe the notion of assistive robotics frequently conjures images of a self-contained *humanoid servant* in which all robotic and intelligence challenges have been addressed. Finding this to be an unlikely possibility in the near term and seeking to move beyond the conventional *static* smart home, we envision an environment containing robotic components that take advantage of the capabilities and higher level thinking of the user to operate in a collaborative manner; working *with* rather than *for* the user.

The author's past investigations into possible forms and use models for assistive robotics have considered appliances such as a hospital over-the-bed table, continuum surfaces, and intelligent storage for personal items.

.

Ease of use

An embedded system is a system which is going to do a predefined specified task is the embedded system and is even defined as combination of both software and hardware. A general-purpose definition of embedded systems is that they are devices used to control, monitor or assist the operation of equipment, machinery or plant. "Embedded" reflects the fact that they are an integral part of the system. At the other extreme a general-purpose computer may be used to control the operation of a large complex processing plant, and its presence will be obvious. All embedded systems are including computers or microprocessors. Some of these computers are however very simple systems as compared with a personal computer. The very simplest embedded systems are capable of performing only a single function or set of functions to meet a single predetermined purpose. In more complex systems an application program that enables the embedded system to be used for a particular purpose in a specific application determines the functioning of the embedded system. The ability to have programs means that the same embedded system can be used for a variety of different purposes. In some cases a microprocessor may be designed in such a way that application software for a particular purpose can be added to the basic software in a second process, after which it is not possible to make further changes. The applications software on such processors is sometimes referred to as firmware.

Arm processor review

ARM stands for Advanced RISC Machines. It is a 32 bit processor core, used for high end application. It is widely used in Advanced Robotic Applications. It performs number of instruction in a single cycle compare with other controllers it have advanced features. The Arm CPU with real-time emulation and embedded trace support that combine microcontroller with embedded high speed flash memory ranging from 32 kB to 512 kB. A 128-bit wide memory interface and unique accelerator architecture enable 32-bit code execution at the maximum clock rate. For critical code size applications, the alternative 16-bit Thumb mode reduces code by more than 30 % with minimal performance penalty. Due to their tiny size and low power consumption, LPC2141/42/44/46/48 are ideal for applications where miniaturization is a key requirement, such as access control and point-of-sale. Serial communications interfaces ranging from a USB 2.0 Full-speed device, multiple UARTs, SPI, SSP to I2C-bus and on-chip SRAM of 8 kB up to 40 kB, make these devices very well suited for communication gateways and protocol converters, soft modems, voice recognition and low end imaging, providing both large buffer size and high processing power.

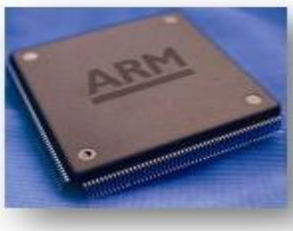

Figure 1. Arm processor

History and Development

- ARM was developed at Acron Computers ltd of Cambridge, England between 1983 and 1985.
- RISC concept was introduced in 1980 at Stanford and Berkley.
- ARM ltd was found in 1990.
- ARM cores are licensed to partners so as to develop and fabricate new microcontrollers around same processor cores.

Key features

1. 16-bit/32-bit ARM7TDMI-S microcontroller in a tiny LQFP64 package.
2. 8 kB to 40 kB of on-chip static RAM and 32 kB to 512 kB of on-chip flash memory.
   128-bit wide interface/accelerator enables high-speed 60 MHz operation.
3. In System Programming/In-Application Programming (ISP/IAP) via on-chip boot loader software. Single flash sector or full chip erase in 400 ms and programming of 256 bytes in 1 ms.

Historical Background

The invention of the transistor at Bell Telephone Laboratories in 1947 sparked a fast-growing microelectronic technology. Jack Kilby of Texas Instruments built the first integrated circuit (IC) in 1958 using germanium (Ge) devices. It consisted of one transistor, three resistors, and one capacitor. The IC was implemented on a sliver of Ge that was glued on a glass slide. Later that same year Robert Noyce of Fairchild Semiconductor announced the development of a planar double-diffused Si IC. The complete transition from the original Ge transistors with grown and alloyed junctions to silicon (Si) planar double-diffused devices took about 10 years. The success of Si as an electronic material was due partly to its wide availability from silicon dioxide (SiO2) (sand), resulting in potentially lower material costs relative to other semiconductors.

II. RELATED WORK

Human gesture may occur in various forms including hand and arm gesticulation, pantomime, sign language, static poses of the hand and body, or language-like gestures that may replace words during speech. Of these, hand and arm gesticulation account for some 90% of gestured communication. Hence, the exploration of gesture at this scale as a means of command interaction with robotics and computing is warranted. Efforts at automated gesture recognition generally involve a common set of considerations and problems to be addressed. These include some combination of sensor platform, data representation, pattern recognition, and machine learning. This section discusses previous approaches to these problems relative to the methods applied in this paper.

Sensing

In order for gestures to be detected and classified, the motion or pose of the actor must be sensed. Typical sensor strategies include wearable devices such as data gloves or body suits that are instrumented with the magnetic field tracking devices or accelerometers or vision-based techniques involving one or more cameras. Still other approaches involve IR motion or proximity sensors use a glove-based orientation sensor to extract static hand positions to be used as commands. Lementec and Bajcsy use wearable (arm) orientation sensors for sensing arm gesture models composed of Euler angles. These are intended for use in an unmanned aerial vehicle and implemented as a lab simulation. Zhou use MEMS accelerometer data to characterize hand motions including *up*, *down*, *left*, *right*, *tick*, *circle*, and *cross*. Wearable sensors are also used in and others. Typically, however, the usefulness of wearable devices to measure gestured motion is accompanied by the acknowledgment that such devices may limit user motion and often require a wired connection to a computer. Thus, they present inherent impediments to practical application.

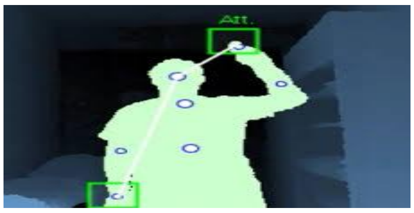

Figure 2. Neural clustering of hand motion for gesture recognition in real time

IR proximity sensors are used by Cheng *et al* to create a reliable gesture recognition system for a touch less mobile device interface. The method uses the pair wise time delay between a passing user’s hand and two IR proximity sensors. This system detects gestures of swipe right, swipe left, push, and pull. Dongseok propose a computer control interface design using a proximity sensor to extract hand commands to a GUI.

The mechanism is scaled as a mouse replacement. Such coarse assessment of motion is not sufficiently descriptive to support an extensive vocabulary of gestures. However, as shown by Yanik an array of IR motion sensors can provide sufficiently rich data to allow for accurate classification of gross motions. Much of the work in gesture recognition is performed using video image sequences due to the richness of information and cost effectiveness available with cameras. A recent thorough discussion of vision based and other sensor types for the purpose of gesture recognition is given by vision based approaches may suffer from disadvantages associated with latency, occlusion, or lighting. Furthermore, since most video sequences represent 3-D to 2-D projection, a loss of information is inherent in the processing of data. In addition, although the presence of cameras in an individual’s personal environment is becoming more common, they are often considered intrusive of privacy in certain scenarios .With the limitations of these various sensor types in mind, the study reported in this paper utilizes the Microsoft Kindest RGB-D depth sensing system. The Kindest provides a rich, real-time, 3-D data stream that preserves user anonymity and is also functional in dark environments where conventional cameras would be ineffective.

III. SYSTEM OVERVIEW

The system implements an end-to-end solution between the performance of gesture by a human user and the learned response to gesture by a robotic agent. Modules within the system address each of the constituent challenges described in Section II (sensing, data representation, pattern recognition, and machine learning.

Gesture samples performed by human participants are presented to the system. The samples captured using the Kindest RGB-D sensor are processed and features (DIs) are extracted to form motion descriptors. Descriptors are applied to a GNG based clustered which maps the topology of the input space. A reference node within the GNG *cloud* is selected for each motion and one of its available responses is executed by the robot agent. A (simulated) human user evaluates the response and provides feedback (or *reward*). The reward is used to update the action response for future reference. This way, responses are shown to converge according to the user’s preference for a given gesture type.

The approach described in this paper seeks to overcome certain limitations of prevailing methods by using non intrusive sensing, learning with the human as teacher through a relatively short-training phase, and learning goal configurations with no prior knowledge of the user’s preferences. The remainder of this section details the fixture and algorithms used in pursuit of these contributions.

Data Collection

Data were collected for three essential hand gestures that were deemed a baseline command set for the eventual operation of an Assistive robot. Although our approach places no expectation on the user to perform gestures in a particular manner, motion models for these gestures were taken from the ASL dictionary (as demonstrated in to facilitate repeatability. The gestures included *come closer*, *go away* and *stop*. These three gesture types were seen as a baseline essential collection of commands for an assistive robot. Although the gesture command vocabulary will be increased in future work, these were considered sufficient for this proof-of-concept research. The stop gesture requires special treatment since it intuitively suggests that the robot is presently executing an earlier command. An input gesture to robot action mapping is shown in figure below.

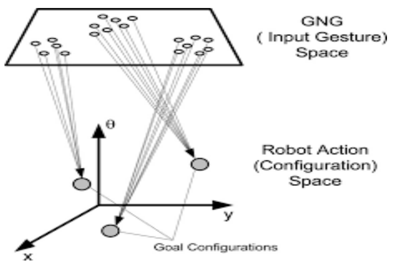

Figure 3. Input gestures to robot action mapping

Hence, the problem of gesture segmentation arises. Since segmentation is not the focus of this paper, we leave its consideration to future work. Instead, *stop* will not be interpreted in its literal sense, but rather as having a specific goal configuration similar to that of come closer and go away. Data samples were collected using the depth sensing feature of the Microsoft Kindest RGB-D camera. The Kindest generates depth maps of the user at approximately 30 frames/s. Samples were collected over 5-second intervals for a total of 150 data points per sample. The data collection program was here developed using the robot operating system (ROS). ROS was selected for its open source and for its active community of research-oriented users. Furthermore, it supports a variety of simulated and real-world robotic platforms through a message based publisher/subscriber environment. Thus, direct migration of this paper to working hardware is expected to be a viable path.

Existing System

It still requires a large amount of storage space and the performance degrades rapidly when the size of the grid increases. The term “Euclidean” is used to distinguish these spaces from the curved spaces of non-Euclidean geometry and Einstein’s general theory of relativity .The first application is robot arm path planning in which a planar two-revolute-joint robot arm rotates among obstacles. The length of path obtained by our algorithm is the shortest among two other fastest O (n log n) algorithms in the literature.

Proposed System

A novel O (n log n) near-shortest path algorithm is presented based on the concepts of Delaunay triangulation. It provides the fastest Computing time among the other approaches and only has a 1.43% path distance difference statistically compared to the shortest O (n2) algorithm. The length of path obtained by our algorithm is the Shortest among two other fastest O (n log n) algorithms in the literature. Compared to the other approaches of reduced visibility graph this safety margin will always result in longer paths than the proposed algorithm.

Keil software

It is possible to create the source files in a text editor such as Notepad, run the Compiler on each C source file, specifying a list of controls, run the Assembler on each Assembler source file, specifying another list of controls, run either the Library Manager or Linker (again specifying a list of controls) and finally running the Object-HEX Converter to convert the Linker output file to an Intel Hex File. Once that has been completed the Hex File can be downloaded to the target hardware and debugged. Alternatively KEIL can be used to create source files; automatically compile, link and covert using options set with an easy to use user interface and finally simulate or perform debugging on the hardware with access to C variables and memory. Unless you have to use the tolls on the command line, the choice is clear. KEIL Greatly simplifies the process of creating and testing an embedded application.

Project

The user of KEIL centers on “project”. A project is a list of all the source files required to build a single application, all the tool options which specify exactly how to build the application, and if required how the application should be simulated. A project contains enough information to take a set of source files and generate exactly the binary code required for the application. Because of the high degree of flexibility required from the tools, there are many options that can be set to configure the tools to operate in a specific manner. It would be tedious to have to set these options up every time the application is being built; therefore they are stored in a project file. Loading the project file into KEIL informs KEIL which source files are required, where they are, and how to configure the tools in the correct way.

KEIL can then execute each tool with the correct options. It is also possible to create new projects in KEIL. Source files are added to the project and the tool options are set as required. The project can then be saved to preserve the settings. The project is reloaded and the simulator or debugger started, all the desired windows are opened. KEIL project files have the extension.

Simulator/Debugger

The simulator/ debugger in KEIL can perform a very detailed simulation of a micro controller along with external signals. It is possible to view the precise execution time of a single assembly instruction, or a single line of C code, all the way up to the entire application, simply by entering the crystal frequency. A window can be opened for each peripheral on the device, showing the state of the peripheral. This enables quick trouble shooting of mis-configured peripherals. Breakpoints may be set on either assembly instructions or lines of C code, and execution may be stepped through one instruction or C line at a time. The contents of all the memory areas may be viewed along with ability to find specific variables. In addition the registers may be viewed allowing a detailed view of what the microcontroller is doing at any point in time. The Keil Software 8051 development tools listed below are the programs you use to compile your C code, assemble your assembler source files, link your program together, create HEX files, and debug your target program. µVision2 for Windows™ Integrated Development Environment combines Project Management, Source Code Editing, and Program Debugging in one powerful environment.

- C51 ANSI Optimizing C Cross Compiler: creates relocatable object modules from your C source code,
- A51 Macro Assembler: creates relocatable object modules from your 8051 assembler source code,
- BL51 Linker/Locator: combines relocatable object modules created by the compiler and assembler into the final absolute object module,
- LIB51 Library Manager: combines object modules into a library, which may be used by the linker,
- OH51 Object-HEX Converter: creates Intel HEX files from absolute object modules.

What's New in µVision3?

µVision3 adds many new features to the Editor like Text Templates, Quick Function Navigation, and Syntax Coloring with brace high lighting Configuration Wizard for dialog based startup and debugger setup. µVision3 is fully compatible to µVision2 and can be used in parallel with µVision2.

Advantages

The proposed algorithm almost outperforms other algorithms in each category from Z Tables except for the path length. In order to locate the Fermat point of a triangle, the largest angle of the triangle must be less than or equal to 120◦.

- Ease of controlling.
- Fast response.
- The module can be made into various forms as per the area of application.
- User friendly- One need not to know about the robot, as they can control by hand movement.

IV. RESULT

This paper **"A Gesture learning interface for simulated robot Path shaping with a human teacher"** has been successfully designed and tested**.** Integrating features of all the hardware components used have developed it. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit. Secondly, using highly advanced IC's and with the help of growing technology the paper has been successfully implemented.

V. CONCLUSION AND FUTURE SCOPE

In this paper, we have presented a new approach toward the development of a gesture-based human–machine interface. An end-to-end approach is presented which maps arm-scale gesture by a human user to a learned response by a robotic agent through repeated applications of user-provided reward. Between these two end points, the constituent challenges are addressed in the areas of sensor selection, data representation, pattern recognition, and machine learning. As a composite approach, the proposed system overcomes many of the shortcomings of previous efforts.

However, it is foreseen that this process would be difficult for an actual human trainer to entertain, since visualization of a robot's configuration as a Cartesian space may be difficult, if not impossible, for higher dimensions. Certainly, a path-planning component would be called for which considers the robot's configuration in light of the geometry of the environment and the social sensibilities of the user (speed, angle of approach, visibility, etc.). As it is implemented here, the proposed approach is seen as practical for mapping of a robot's end effector and, thus, useful for common applications. Segmentation of gestures (*gesture spotting*) is a typical problem in gesture recognition. Although our future work will address spotting, our focus here has been on validation of a real time learning technique that produces desirable outcomes using a human teacher and a simple, binary reward signal.

REFERENCES

[1] A. El-Sawah, N. Georganas, and E. Petriu, "A prototype for 3-D handtracking and gesture estimation," IEEE Trans. Instrum. Meas., vol. 57, no. 8, pp. 1627–1636, Aug. 2008.

[2] D. G. Lowe, "Distinctive image features from scale-invariant key points," Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, Nov. 2004.

[3] A. Bosch, X. Munoz, and R. Marti, "Which is the best way to organize/ classify images by content?" Image Vis. Compute, vol. 25, no. 6, pp. 778–791, Jun. 2007.

[4] H. Zhou and T. Huang, "Tracking articulated hand motion with Eigen dynamics analysis," in Proc. Int. Conf. Comput. Vis., 2003, vol. 2, pp. 1102–1109.

[5] B. Stenger, "Template based hand pose recognition using multiple cues," in Proc. 7th ACCV, 2006, pp. 551–560.

[6] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multiscale color features, hieracrchichal models and particle filtering," in Proc. Int. Conf. Autom. Face Gesture Recog., Washington, DC, May 2002.

[7] A. Argyros and M. Lourakis, "Vision-based interpretation of hand gestures for remote control of a computer mouse," in Proc. Workshop Comput.Human Interact., 2006, pp. 40–51.