I.J. Intelligent Systems and Applications, 2014, 01, 102-111
Published Online December 2013 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2014.01.11

Persistence of Vision Control Using Arduino

Robinson P. Paul
Birla Vishvakarma Mahavidyalaya Engineering College, Gujarat Technological University, Gujarat, India
E-mail: robinson.paul@bvmengineering.ac.in

Ghansyam B. Rathod
Birla Vishvakarma Mahavidyalaya Engineering College, Gujarat Technological University, Gujarat, India
E-mail: ghansyam.rathod@bvmengineering.ac.in

Vishwa R. Trivedi
Birla Vishvakarma Mahavidyalaya Engineering College, Gujarat Technological University, Gujarat, India
E-mail: vishwatrivedi213@gmail.com

Punit V. Thakkar
Birla Vishvakarma Mahavidyalaya Engineering College, Gujarat Technological University, Gujarat, India
E-mail: punitvthakkar@gmail.com

Abstract— This paper mainly emphasizes on the POV (Persistence Of Vision) technology. In current era in which energy is the main factor in designing all the applications, maximum and efficient use of the energy is very important. A POV display has many advantages over a traditional CRT, LCD or LED display, like power savings, less complexity, easy configuration, attractiveness etc. To overcome the drawback of old processor we have decided to implement the same display atop a new and advanced microprocessor, the Arduino duemilanove. This platform brings with it newer coding and a different understanding of peripherals. ARDUINO INTERFACE BOARDS provide us with a low-cost, easy-to use technology to create the project. We also aim to build the newer display to work with modern forms of interfaces. To accomplish this, we will be interfacing the display with an Android device. This project can be implemented with help of any Android Smartphone/tablet running Android 4.0+.

Index Terms— POV, Arduino, Android, ATMega 328, USB Interfacing, Piranha LED

I. Introduction

The core phenomenon on which the entire project is based is the Persistence of vision. Persistence of vision is the phenomenon pertaining to the human eye by which an afterimage is thought to persist for approximately one twenty-fifth of a second on the retina. The way this phenomenon of persistence of vision works is based on the belief that human perception of motion (brain centered) is the result of persistence of vision (eye centered). Any motion that we see around us is the direct implication of persistence of vision phenomenon at work. The belief was debunked in 1912 by Wertheimer [1] but persists in many citations in many classic and modern film-theory texts. [2][3][4][5]. Persistence of vision is still the accepted term for this phenomenon in the realm of cinema history and theory. Blinky POV [10] is a reprogrammable LED kit that uses persistence of vision to create the illusion of text or a small picture floating in the air. In the past, it was scientifically proven that a frame rate of less than 16 frames per second caused the mind to see flashing images. People still identify motion at rates as low as ten frames per second or slower (as in a flipbook). The flicker caused by the shutter of a film projector is distracting below the 16-frame threshold. In drawn animation, moving characters are often shot "on twos", that is to say, one drawing is shown for every two frames of film (which usually runs at 24 frames per second), meaning there are only 12 drawings per second. Even though the image update rate is low, the fluidity is satisfactory for most subjects. However, when a character is required to perform a quick movement, it is usually necessary to revert to animating "on ones", as "twos" are too slow to convey the motion adequately.[5]

The phenomenon of POV is evident in a number of events like for e.g.: When we rotate a fire cracker stick at high speed, we observe a continuous circle being formed as shown in fig.[1.1]. [6]

Fig. 1.1: Demonstration of POV effect

A two dimensional POV display is often accomplished by means of rapidly moving a single row of LEDs along a linear or circular path. The effect is that the image is perceived as a whole by the viewer as long as the entire path is completed during the visual persistence time of the human eye as shown in fig. [1.2]. A further effect is often to give the illusion of the image floating in mid-air. A 3 dimensional POV display is often constructed using a 2D grid of LEDs which is swept or rotated through a volume. POV display devices can be used in combination with long camera exposures to produce light. [5]

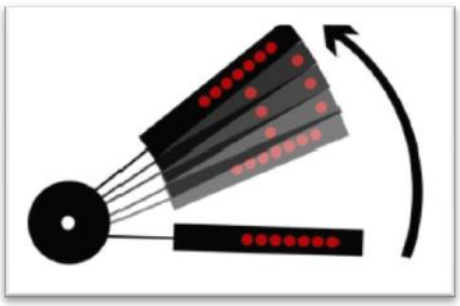

Fig. 1.2: POV effect demonstrations

II. A Display Based on 8051 Microcontroller

2.1 Existing Implementation and Modification

We have already implemented a POV display based on 8051 microcontroller. The display used an 8051 based AT89C2051 microcontroller chip that was used to control the switching of 8 red colored LEDs as shown in Fig.[2.1]

Fig. 2.1: POV display based on 8051 microcontroller

The display consisted of the following components:

1. Microcontroller - AT89C2051: Central controller for switching the LEDs at appropriate time.
2. IR sensor – MOC 7811: Used to provide Interrupt to the microcontroller whenever it reached home position [starting position] so that it can start the display routine again.
3. DC motor – 12 V: Used to rotate the assembly at high speed to induce the persistence of vision effect.
4. LED display – 8 general purpose LEDs strip: Used as display agents.
5. Power supply 12 V DC for motor and +5 V DC for microcontroller.

This display was adequate for the needs of the company that we made it for, but we found out during the course of constructing this display that there was a huge scope of enhancement in the design of the display. The shortcomings of the display were:

(1) The microcontroller AT89C2051 used was a low speed microcontroller and a basic one [11]. Operation at high speed was not satisfactory as very often lines in the display were missed due to the slow switching speed of the controller. The entire aesthetics of the display lies in the speed at which a controller can switch the LEDs from one state to another, thus a faster controller was desirable to increase the display quality.

(2) The LEDs used were simple single colored LEDs. Nowadays a wide variety of LEDs are available which can enhance the color displaying capacity of the display.

(3) One more shortcoming of using a primitive controller was that to change the display pattern, we needed to stop the display, unmount the chip, burn a new code in it and again remount it. This shortcoming can be seen to by using some way to upload a message onto the controller directly without needing to unmount it.

(4) The mechanical assembly of the display especially the motor was a bit unstable and over long duration of operation, the vibrations caused the display to go out of sync.

Due to such shortcomings, we decided to implement the same display atop a new and advanced microprocessor base and decided to change some elements that caused undesired operation.

III. The Arduino Development Board and Android Operating System

3.1 Ardino Development Borad Details

The search for a new microcontroller started with the main factor as speed of operation. We looked into different kinds of microcontrollers that are available in today‘s markets. It is during this search that we came across a totally new generation of controllers that were ready interfaced and armed with loads of basic features. Such controllers are called development boards.

Due to the attractive features of such boards, we started searching a development board that suited our purpose. Arduino [12] is a small microcontroller board with a USB plug to connect to your computer and a number of connection sockets that can be wired up to external electronics, such as motors, relays, light sensors, laser diodes, loudspeakers, microphones, etc. They can either be powered through the USB connection from the computer or from a 9V battery.

They can be controlled from the computer or programmed by the computer and then disconnected and allowed to work independently. Making the display on this entirely new platform requires that we write entirely new code for it.

This is one of the most convenient things about using an Arduino. Many microcontroller boards use separate programming hardware to get programs into the microcontroller. With Arduino, it's all contained on the board itself. We finally rested upon the Arduino series of development boards. We finalized on the Duemilanove 328 version of the development board due to its easy availability in local markets as shown in fig.[3.1].[7]

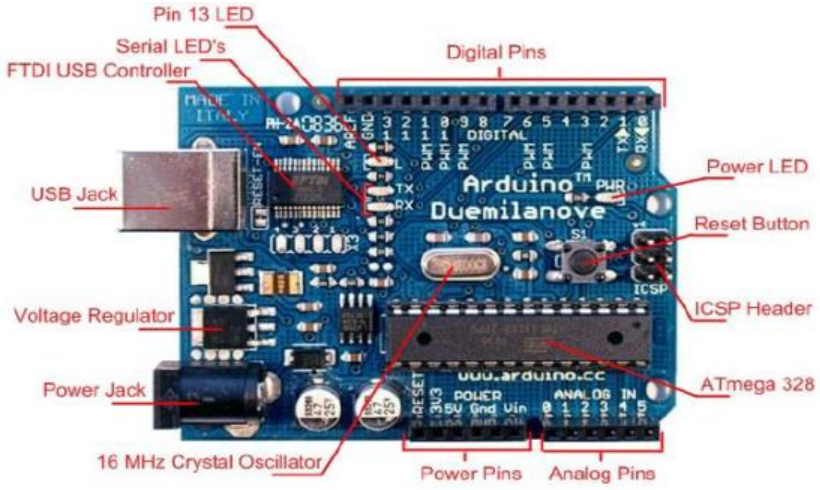

Fig. 3.1(a): Arduino Duemilanove board

Table 1: Arduino Duemilanove technical specifications

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
Flash Memory	32 KB of which 2 KB used by boot loader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

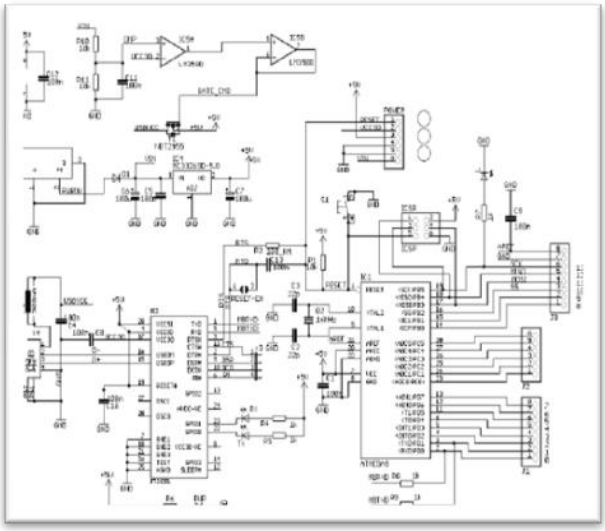

Fig. 3.1(b): Duemilanove schematic [8]

3.2 Android Operating System

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE [13] for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. There is typically no need to edit make files or run programs on a command-line interface. Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. Users only need to define two functions to make an executable cyclic executive program: [7]

setup(): a function run once at the start of a program that can initialize settings

loop(): a function called repeatedly until the board powers off.

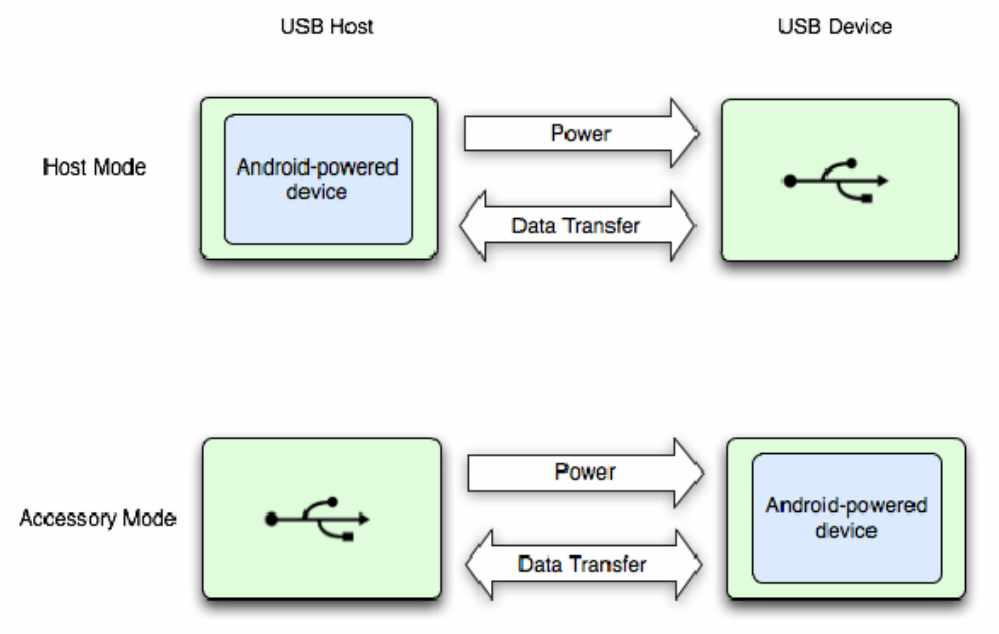

Fig. 3.2: Android USB interfacing

Android supports a variety of USB peripherals [14] and Android USB accessories (hardware that implements the Android accessory protocol) through two modes: USB accessory and USB host. In USB accessory mode, the external USB hardware act as the USB hosts. Examples of accessories might include robotics controllers; docking stations; diagnostic and musical equipment; kiosks; card readers; and much more. This gives Android-powered devices that do not have host capabilities the ability to interact with USB hardware. Android USB accessories must be designed to work with Android-powered devices and must adhere to the Android accessory communication protocol. In USB host mode, the Android-powered device acts as the host. Examples of devices include digital cameras, keyboards, mice, and game controllers. USB devices that are designed for a wide range of applications and environments can still interact with Android applications that can correctly communicate with the device.

Here we show the differences between the two modes. When the Android-powered device is in host mode, it acts as the USB host and powers the bus. When the Android-powered device is in USB accessory mode, the connected USB hardware (an Android USB accessory in this case) acts as the host and powers the bus.

IV. Implementation & Configuration of Differnet Modules

Implementing the project hardware requires construction of different modules that will then be put together to create the final outcome. The project hardware is divided into 3 main modules:

(i) LED module

(ii) The Arduino board

(iii) Motor mechanical assembly

(iv) Motor Speed control module

Out of these, not much work is needed to be done on the Arduino board as it comes already mounted on a circuit. The only work required to be done to get the board running is to provide a 7-12 V supply at the supply pin provided on the board. An alternate way of providing power is by connecting the USB plug of the board to a computer (or even a Smartphone via an OTG cable!). The other modules however require specialized making as no pre-made circuit is available in the market.

4.1 LED Module

The LED module consists of 8 RGB LEDs connected in a vertical column and provided with a pull-up resistor for each of the 8 LEDs. The LEDs can be quite expensive when bought from a regular component supplier. You can get a better deal on eBay. When buying them, make sure they are common anode (the positive connections of each of the three LEDs connected together) and that the pin connections are the same. It is always worth checking the

LEDs data sheet. LEDs with a diffuse lens produce a better effect than those with a clear lens [15]. Although such a module using simple single colored LEDS is readily available in the market, the use of 4 legged RGB LEDs impose new design requirements and complex soldering work because of 4 closely spaced legs of the LEDs.

As mentioned earlier, the LEDs used for this project are 5mm piranha RGB LEDs. The actual picture of the LED is given in figure [4.1].

Fig. 4.1: piranha RGB LED

The LED has 4 legs, one for each color viz. – Red, Green, Blue and one a common anode as shown in figure [4.2]. In order to make the LED glow with desired color, a connection is required to be made between the leg corresponding to the color required and the common anode. There is a slit on one of the corners of the square package; the leg directly below the slit corresponds to green color. Using this as a reference and moving in clockwise direction with the face of the LED upwards, the next leg is the leg corresponding to blue colour, the next one- the leg corresponding to red color and the last one, below the green leg is the common anode.

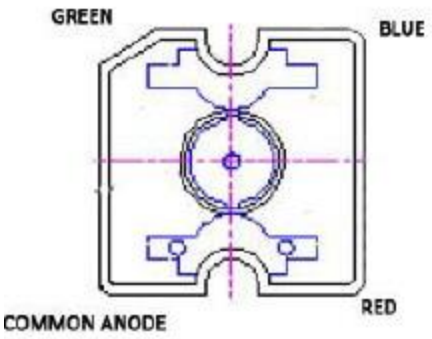

Fig. 4.2: LED legs configuration

The schematic for the LED module is as shown in figure [4.3].

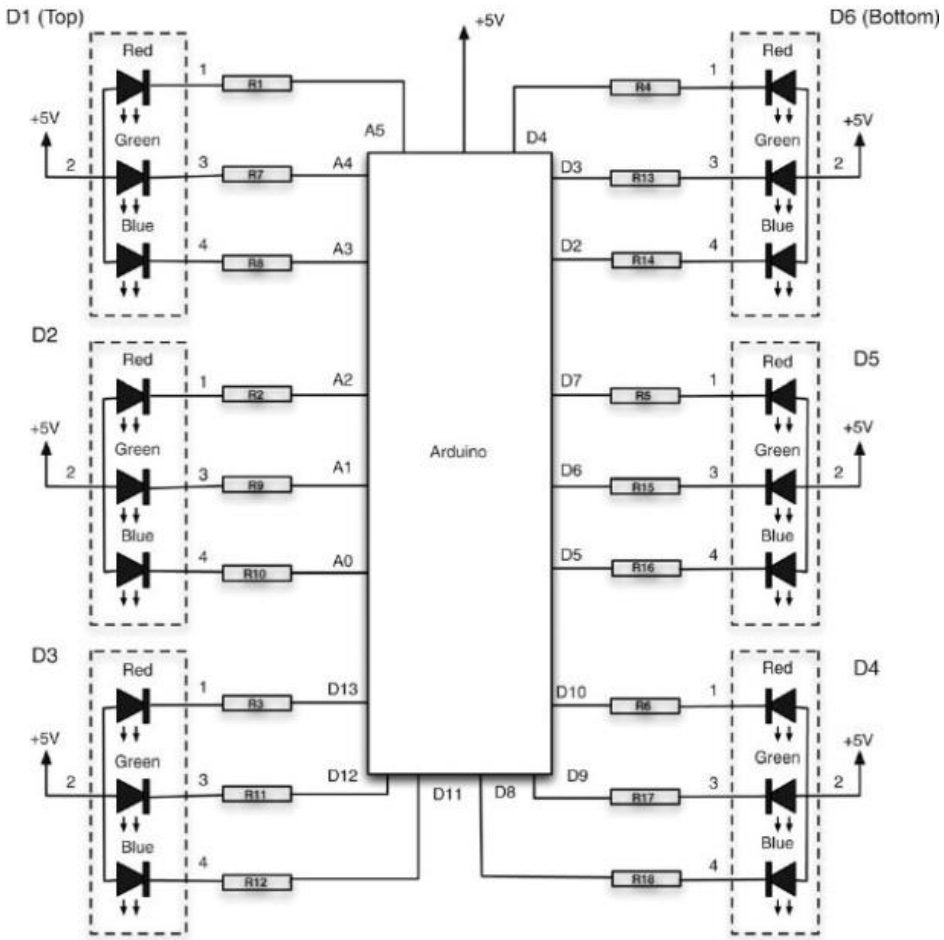

Fig. 4.3 (a): LED module schematic

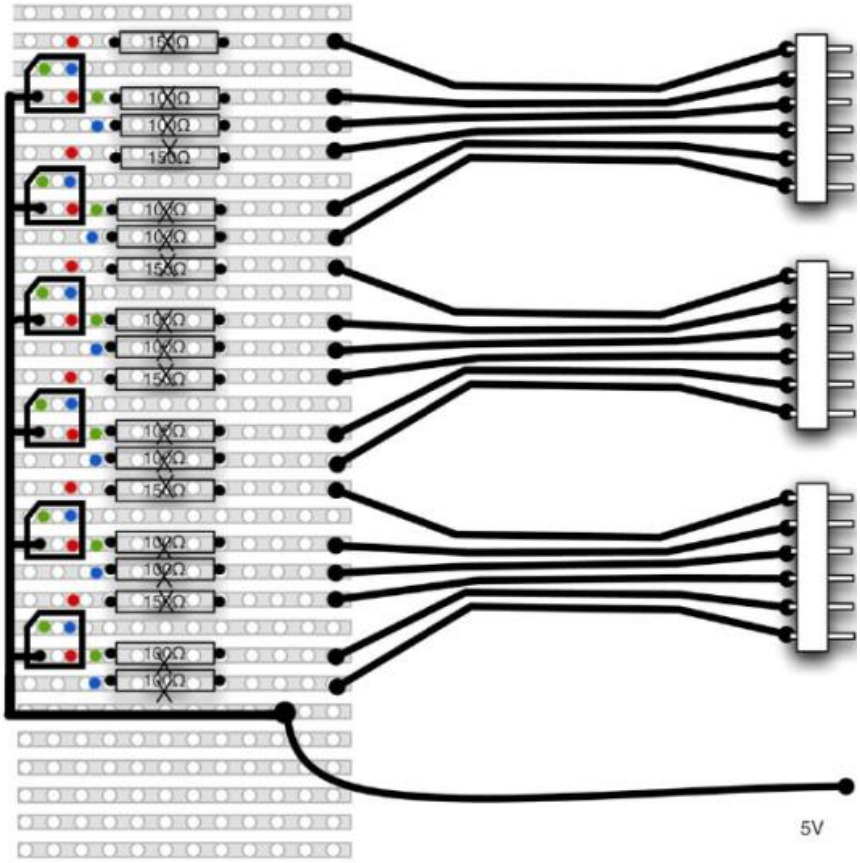

Fig. 4.3 (b): LED module strip board layout

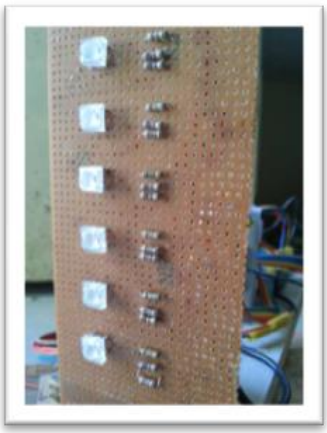

Fig. 4.3 (c): LED module strip board layout picture

Here D1-D6 is LEDs and R1-R18 are pull-up resistors. Here the value of pull-up resistor for each leg of LED is different.

4.2 The Arduino Board

The free ends of the pull-up resistors are connected to ribbon wires in pairs or 2, i.e. a ribbon of 6 wires used for 2 LEDs. A 6 wire ribbon scheme is chosen so as to make it easy for the wires to be plugged into the I/O ports of the Arduino. The Arduino has 3 sets of 6 I/O ports and so we can easily connect the 3 ribbon lines coming out from the LED module. The ends of the ribbon wires should be soldered to 6 pin male connectors so that they can be easily inserted in the female connectors that are already provided on the Arduino board as shown in figure [4.4].

Fig. 4.4: Male connectors at wire ends

4.3 Motor Speed Control Module & Mechincal Assembly

We need to be able to control the speed of the motor, because it must match the speed at which the LEDs switch from displaying one column to the next. There is no sensor to automatically synchronize the LEDs with

the spinning motor. Although such controllers are readily available in the market, we decided to make our own to make the task more challenging and informative. The motor controller uses the 555 timer IC as the primary timing device.[8]

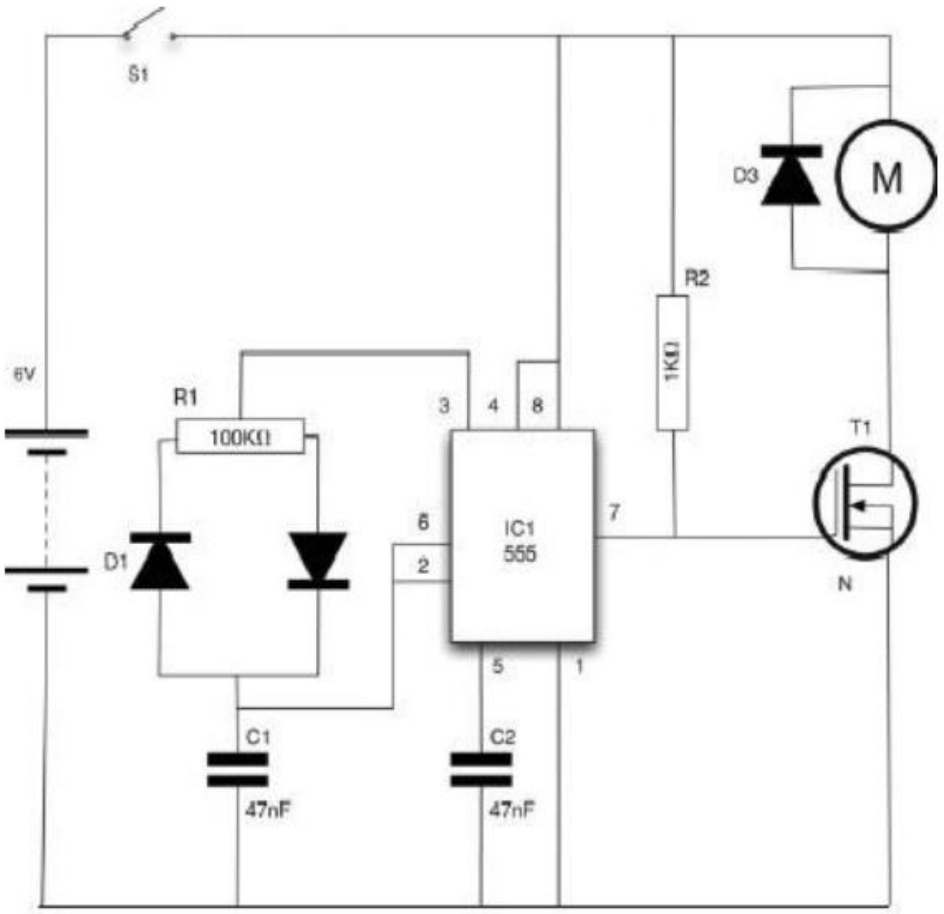

Fig. 4.5: The schematic diagram of the motor controller [Source: Evil Genius]

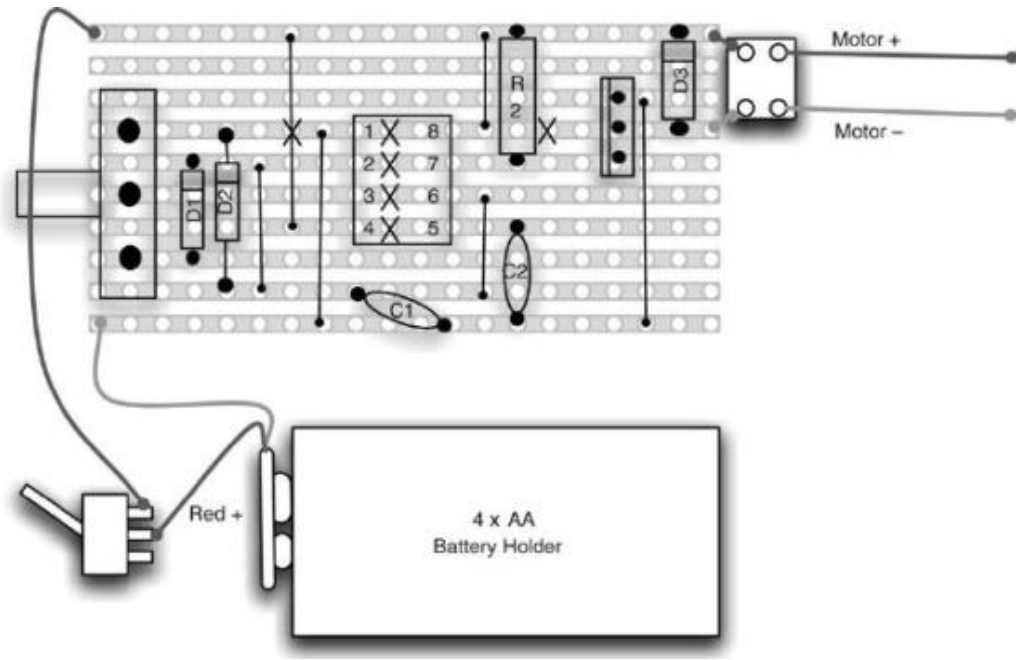

Fig. 4.6: Motor controller module strip board layout

At this point of time we have all the modules necessary, we now need to assemble them on a flat stable base which can be then mounted on the motor shaft and rotated. We have used a wooden sheet having dimensions: Width - 2.5 inch, Length - 7 inch, Thickness – 0.4 inch. The Led module is attached in a vertical manner to the narrower dimension of the mounting piece as shown in figure [4.7]. The Arduino board is mounted on the face of the wooden piece and is held together in place with the help of 2 nails pierced through the holes provided in the board by the manufacturers. The battery assembly required to power the board is put at the centre of the piece to provide balance. The final assembly of the complete project can be seen in figure [4.8].

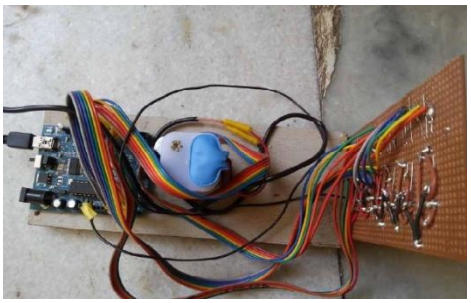

Fig. 4.7: Final assembly of mounting piece

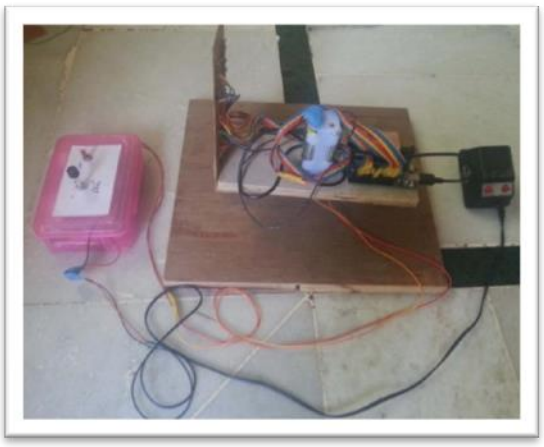

Fig. 4.8: POV project Hardware

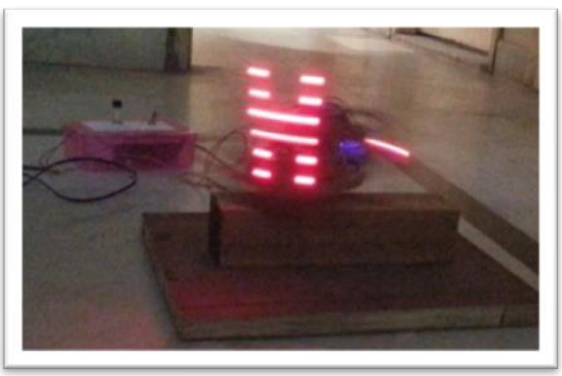

Fig. 4.9: POV Display

V. Test and Troubleshooting

5.1 Testing the Arduino Board and Connections

When first time we switch on the board, we will see that an LED labeled L connected to the pin no. 13 of the board will light up to indicate that the board is working. The led in the LED column connected to this pin 13 will also glow. Thus we will get only one LED to glow, one connected to pin 13. The board can then be connected to a pc and the desired program can be loaded into it to obtain desired glowing LED effect. After the desired program is loaded, disconnect the USB from the board.

5.2 Testing the LED Module

Before we run the display on full scale, we need to check that all the LEDs are connected to the Arduino board and that each one displays the desired color when a particular value is stored in the Arduino. The most common problem that occurs in connection of LEDs is when we connect the LEDs to wrong pins in the controller. This may cause a totally different pattern to be displayed and may give an impression that the display is out of sync. Care must be taken to connect the LEDs in a sequential manner to the pins on the controller. Miss sequenced pins may cause the display to appear inverted or in the form of a mirror image of what is originally to be displayed.

5.3 Testing the Motor Speed and Synchronization

Now we are ready to rotate the display. But before we start the motor, make sure that no wires are tangles to any of the parts that will rotate when we turn on the motor. This can cause extensive hardware damage! Now put the toggle switch in the motor controller module to ON position. Initially keep the potentiometer at slowest speed and check that no part is hanging out that may cause the assembly to dug or to get tangled during operation. Finally when everything is checked out, slowly start increasing the speed by rotating the potentiometer until a satisfied resolution is obtained. If still the display doesn‘t work properly, try to remove excess weight from the mounting assembly which may cause the motor to operate at low RPMs.

5.4 Testing the Motor Speed Controller

It might happen that the controller might not be supplying sufficient power to the motor to operate to its full capability. To ensure that the module is working at its full potential, check voltage levels at the module output using a DMM.

VI. Android Software Implementation

6.1 Eclipse IDE

The software implementation consists of creating an Android application that interfaces with the Arduino Duemilanove board. We have proceeded to create the Android application through the Android SDK in the development environment known as ―Eclipse. The Eclipse IDE is a powerful and robust IDE that is extremely useful and supports building Android apps by loading the Android SDK. Creating an Android application involves three basic parts:

(i) Creating the application‘s layout. The layout is saved as an .xml (extended Mark-up Language) file.

(ii) Creating the application‘s logical code in JAVA coding language. This part contains all the inner workings of the application.

(iii) Creating the application‘s index in the manifest file (Application_name_manifest.xml). This manifest file contains all the permissions required by the application to access in the user‘s device. It also contains a list of all classes and intents to be called in the application. It defines the theme of the application, the title of various classes and thus, is central to an application’s functioning.[16]

6.2 Eclipse Layout

A layout defines the visual structure for a user interface, such as the UI for an activity or app widget. One can declare a layout in two ways:

(i) Declare UI elements in XML. Android provides a straightforward XML vocabulary that corresponds to the View classes and subclasses, such as those for widgets and layouts.

(ii) Instantiate layout elements at runtime. An application can create View and View Group objects (and manipulate their properties) programmatically.[16]

VII. Conclusion

The conventional LED displays used in advertisements and information display uses an individual matrix of LEDs for each character. For example in a display using 8X4 Led matrix for a single character, 10X8X4 LEDs would be required to display a word consisting of 10 characters, increasing the circuit complexity. Our circuit uses a low power Arduino Duemilanove board, with 7 multicolour LEDs and a low power motor. This technology saves power monumentally. The display is configurable to show any pattern in any colour through an Android device that dictates its output as shown in figure [4.9]. The display is extremely attractive to look at and gives a sense of being a transparent display. By using a motor of higher RPM, one can achieve more clarity in the display.

References

[1] Wertheimer, 1912. Experimentelle Studien über das Sehen von Bewegung. Zeitschrift für Psychologie 61, pp. 161–265.

[2] Bazin, André (1967) What is Cinema?, Vol. I, Trans. Hugh Gray, Berkeley: University of California Press.

[3] Cook, David A. (2004) A History of Narrative Film. New York, W. W. Norton & Company.

[4] Metz, Christian (1991) Film Language: A Semiotics of the Cinema, trans. Michael Taylor. Chicago: University of Chicago Press.

[5] Reference Note : http://en.wikipedia.org/

[6] Image courtesy: http://soullostatsea.deviantart.com/

[7] Reference Image : http://en.wikipedia.org/wiki/Arduino

[8] Reference Image : arduino.cc/en/uploads/Main

[9] Reference:http:/Evil Genious.com/motor controller

[10] Reference: www.wayneandlayne.com/projects/blinky

[11] Dr. Dobb's Journal, Atmel's AT89C2051 Microcontroller July 1997.

[12] Sarik, J. ; Kymissis, I. , Lab kits using the Arduino prototyping platform , Publication Year: 2010 , Page(s): T3C-1- T3C-5 , IEEE conference.

[13] Arduino Due Released; Arduino.cc

[14] Kyuchang Kang ; Dongoh Kang ; Kiryong Ha ; Jeunwoo Lee , Android phone as wireless USB storage device through USB/IP connection , Publication Year: 2011 , Page(s): 289- 290 , IEEE conference.

[15] Varjo, S.; Hannuksela, J.; Silven, O., Direct imaging with printed microlens arrays, Publication Year: 2012 , Page(s): 1355 – 1358 , IEEE conference.

[16] Geer, David ; Eclipse becomes the dominant Java IDE, Volume:38, Issue: 7 Digital Object Identifier: 10.1109/MC.2005.228,Publication Year:2005,Page(s):16- 18, IEEE conference.

Authors’ Profiles

Robinson P. Paul was born in Gujarat India 1985. He has received the BE and ME degrees in Electronics & Communication from Sardar Patel University, in 2007 and 2011 respectively.

He is working as an assistant

professor in BVM Engineering College in Gujarat, India. His activities currently focus on Digital image registration and VLSI Design. His research interests include Embedded system design, Digital Image processing, Speech and Gesture recognition, VLSI design, Power Electronics.

Ghansyam Rathod was born in Gujarat India 1986. He has received the BE and ME degrees in Electronics & Communication from Sardar Patel University and GTU, in 2007 and 2012 respectively.

He is working as an assistant professor in BVM Engineering College in Gujarat, India. His research interests include wireless communication, Biomedical instrument, Linear IC application, Embedded System, VLSI.

Punit V. Thakkar was born in Gujarat India 1992. He has received the BE degree in Electronics & Telecommunication from GTU, 2013.

His research interests include Digital Logic Design, Embedded system & C, C++ programming.

Vishwa R Trivedi was born in Gujarat India 1991. He has received the BE degree in Electronics & Telecommunication from GTU, 2013.His research interests include Digital Logic Design, PCB Design, Microprocessor.

How to cite this paper: Robinson P. Paul, Ghansyam B. Rathod, Vishwa R. Trivedi, Punit V. Thakkar,"Persistence of Vision control using Arduino", International Journal of Intelligent Systems and Applications(IJISA), vol.6, no.1, pp.102-111, 2014. DOI: 10.5815/ijisa.2014.01.11

