TOUCH FREE INTERACTION USING HAND GESTURES-BASED MEDIA CONTROLLER

MINI PROJECT REPORT

Submitted by

HARISH A- 2116230701105
ADHAVAN BALAJI N M - 2116230701012

In partial fulfillment for the award of the degree

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2024-2025

BONAFIDE CERTIFICATE

Certified that this project **"Touch Free Interaction using Hand Gestures - Based Media Controller**" is the bonafide work of "**HARISH A-2116230701105 ADHAVAN BALAJI N M-2116230701012"** who carried out the project work under my supervision.

SIGNATURE
Dr.N.Duraimurugan, M.Tech., Ph.D.

Associate Professor,

Computer Science & Engineering

Rajalakshmi Engineering College
(Autonomous)
Thandalam, Chennai -602105.

Submitted for the **ANNA UNIVERSITY** practical examination Mini-Project work viva voice held on**_____________**_

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman **Mr. S.MEGANATHAN, B.E, F.I.E.,** our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.,** Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work.

We also extend our sincere and hearty thanks to our Internal Guide **Dr.N.Duraimurugan, M.Tech., Ph.D.** Associate Professor, Department of Computer Science and Engineering for his valuable guidance and motivation during the completion of this project. Our sincere thanks to our family members, friends and other staff members of information technology.

HARISH A- 2116230701105
ADHAVAN BALAJI N M - 2116230701012

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATION

ABBREVIATION	**FULL FORM**
IOT	INTERNET OF THINGS
HCI	HUMAN-COMPUTER INTERACTION
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
USB	UNIVERSAL SERIAL BUS
LED	LIGHT EMITTING DIODE
GUI	GRAPHICAL USER INTERFACE
PC	PERSONAL COMPUTER
PAJ7620U2	GESTURE RECOGNITION SENSOR MODULE
ATMEGA328P	MICROCONTROLLER USED IN ARDUINO UNO
API	APPLICATION PROGRAMMING INTERFACE
COM	COMMUNICATION PORT
OS	OPERATING SYSTEM

ABSTRACT

In an age of increasing demand for contactless technology, the "Hand Gesture-Based Media Controller" offers a novel solution for intuitive, touch-free interaction with digital media. This IoT mini-project leverages the PAJ7620 gesture recognition sensor and Arduino Uno to create a hands-free media control system that allows users to manage playback functions—such as play/pause, volume control, and track navigationn using simple hand gestures like swipes or waves. The system is particularly beneficial in environments where touch is impractical or for users with mobility constraints.The core functionality hinges on the PAJ7620 sensor's ability to detect directional hand gestures in real time.A supporting Python script or mobile app receives these signals and simulates keypress actions on the media player, enabling seamless gesture-driven control.Designed with accessibility, hygiene, and convenience in mind, the system is ideal for hands-busy scenarios like cooking, driving, or medical environments. Moreover, it presents a low-cost, scalable solution with real-world applications in smart homes, assistive technologies, and human-computer interaction (HCI) systems.This project showcases the growing relevance of gesture-based interfaces, not only as an alternative input method but as a bridge toward more inclusive and intelligent user experiences. With rapid response time, intuitive control, and easy integration using open-source platforms, the proposed system demonstrates the feasibility and effectiveness of using hand gestures for practical, everyday digital interactions. As touchless interfaces become increasingly essential, this work represents a step toward smarter, more human-centered technology ecosystems.

CHAPTER 1
INTRODUCTION

1. Introduction

In the modern era of technological advancement, the way humans interact with machines is undergoing a revolutionary transformation. Traditional interfaces such as keyboards, mice, and touchscreens are gradually being supplemented or replaced by more intuitive, seamless, and contactless alternatives. One such emerging interface is gesture recognition, a technology that interprets human hand movements to control electronic systems. With the increasing demand for smart, hygienic, and user-friendly interfaces, gesture-based systems are gaining traction in various domains such as smart homes, automotive applications, healthcare, and consumer electronics.

The Hand Gesture-Based Media Controller project is designed to address the growing need for touch-free interaction in digital environments. This innovative system uses the PAJ7620 gesture recognition sensor, paired with an Arduino Uno microcontroller, to interpret simple hand gestures like swiping left or right and waving. These gestures are mapped to common media control functions such as play, pause, next track, previous track, and volume adjustment. The primary advantage of this system lies in its ability to operate without physical contact, making it highly relevant in situations where hands are occupied or when maintaining hygiene is critical such as in kitchens, hospitals, or while working with machinery.By leveraging real-time gesture detection and a low-cost, open-source hardware platform, the project demonstrates a compelling use case for how natural human motion can drive meaningful digital interactions. The project also opens up possibilities for enhancing accessibility for individuals with physical disabilities, allowing them to interact with digital content in a more independent and empowering manner.

The increasing integration of gesture recognition with microcontrollers, sensors, and wireless communication technologies is paving the way for the next generation of human-computer interaction (HCI) systems. This project not only showcases the practicality of such a system but also contributes to the development of inclusive, intelligent, and responsive technological ecosystems.

1.2 Scope of the Work

The scope of this project encompasses the development, implementation, and evaluation of a gesture-based media control system that provides a contactless and intuitive interface for digital media playback. The work focuses on integrating the PAJ7620 gesture sensor with the Arduino Uno microcontroller to capture and interpret predefined hand gestures. The processed gesture commands are then communicated to a media player on a PC or mobile device through a Python script or Bluetooth connection, simulating keypresses that perform media control operations.

The system is designed to support a limited but essential set of commands including play, pause, skip to the next track, return to the previous track, and increase or decrease volume. These commands are triggered by intuitive hand gestures like waving, swiping left or right, or moving the hand upwards or downwards. The project emphasizes real-time performance, low latency, and ease of use. It also explores the use of open-source software and affordable hardware components, making the system accessible to a wider audience including students, hobbyists, and developers in emerging markets.

Furthermore, the project explores potential applications beyond media control, including use in smart home automation, assistive technologies for differently-abled individuals, and sterile or sensitive environments where traditional interaction methods are impractical. The scope also includes testing the system under various lighting and usage conditions to ensure consistent performance and reliability.

1.3 Problem Statement

In many real-world scenarios, especially where hands are occupied, dirty, or when physical contact with surfaces is discouraged, traditional input devices like remote controls, buttons, or touchscreens become inconvenient or unusable. This limitation is especially problematic for people with physical disabilities or in environments like operating rooms, kitchens, or industrial workspaces where maintaining hygiene or safety is paramount.There is a pressing need for a solution that allows users to interact with their devices without direct contact. Existing voice-based controls have their own limitations such as noise interference and language dependency, making them unreliable in certain conditions. Thus, there is a need for a low-cost, reliable, and user-friendly gesture-based solution that enables users to control essential functions like media playback with ease.

1.4 Aim and Objective

The aim of this project is to design and implement a gesture-based media controller system using the PAJ7620 gesture sensor and Arduino Uno to enable touch-free interaction with digital devices. The main objective is to allow users to perform essential media control functions using intuitive hand gestures without the need for physical contact.

CHAPTER 2
SYSTEM SPECIFICATIONS

2.1 IOT DEVICES

1. PAJ7620U2 Gesture Recognition Sensor
2. Arduino Uno Microcontroller
3. Jumper Wires and Breadboard

2.2 SYSTEM HARDWARE SPECIFICATIONS

Component	Specification
Processor	Intel i5 11th Gen (Minimum)
Memory Size	8 GB (Minimum)
Hard Disk (HDD/SSD)	40 GB (Minimum)
Microcontroller	Arduino Uno (ATmega328P)
Gesture Sensor	PAJ7620U2 Gesture Sensor

2.3 SOFTWARE SPECIFICATIONS

Operating System	Windows 11
Browser	Google Chrome
IDE	Arduino IDE

CHAPTER 3
SYSTEM DESIGN

3.1 ARCHITECTURE DIAGRAM

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components

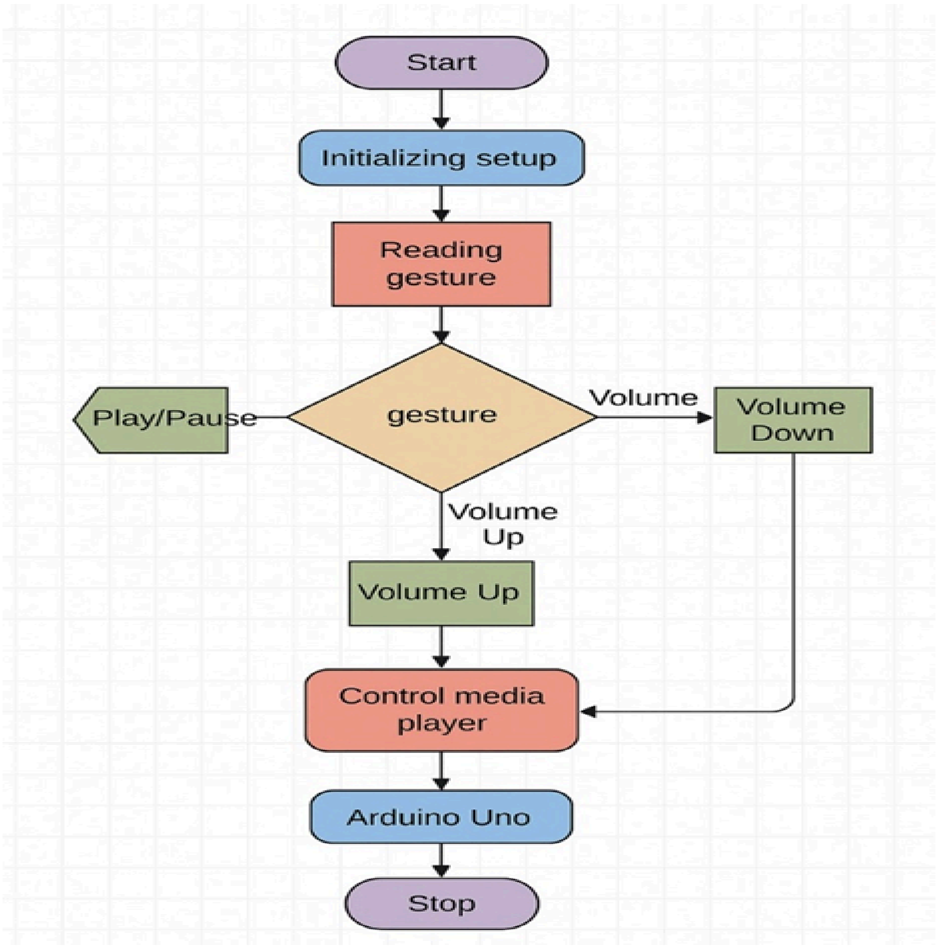


Figure 3.1 Architecture Diagram

From the above Figure 3.1, the architecture of the system is well understood.

3.2 USE CASE DIAGRAM

A use case in the context of our Hand Gesture-Based Media Controller project is a list of actions or event steps that define the interaction between the user (actor) and the gesture-controlled system (the media controller) to achieve specific goals such as playing, pausing, or adjusting the volume of media without physical contact.

In this project, the actor is the user performing hand gestures, and the system includes the PAJ7620 sensor, Arduino Uno, and the connected media device. The use case describes how the user interacts with the system using intuitive gestures to control media playback in a seamless, touch-free manner.

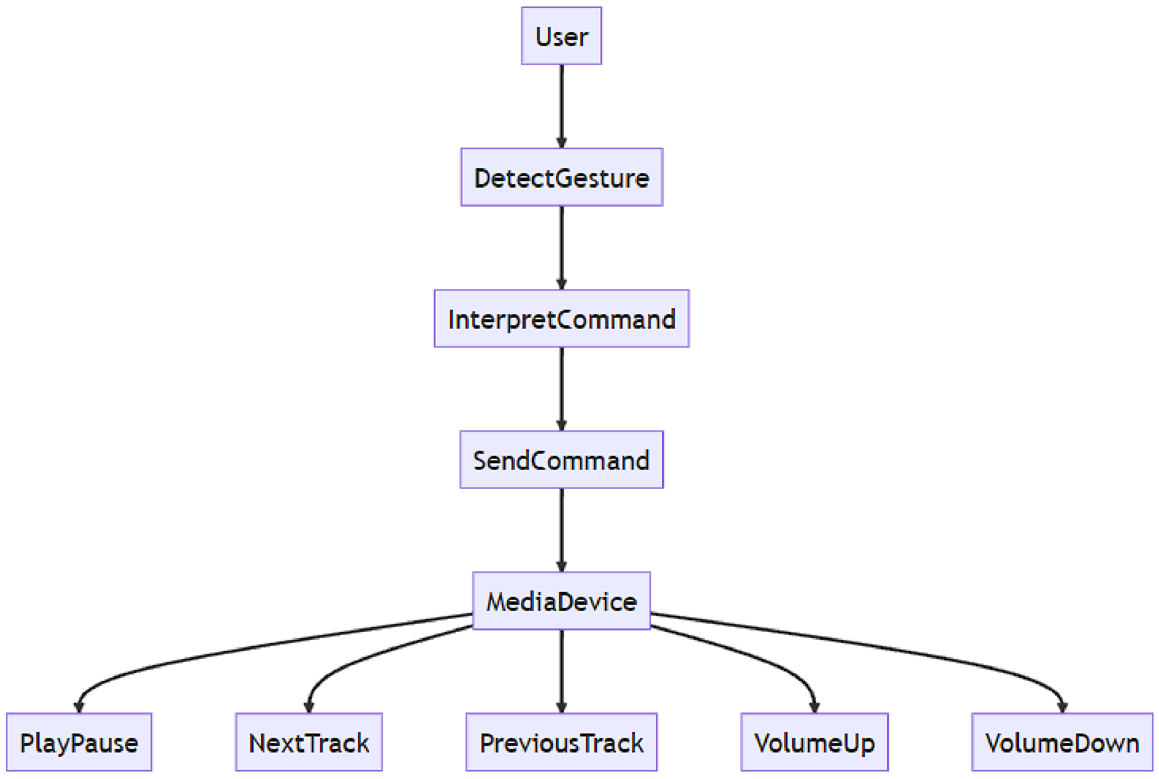


Figure 3.2 Use case diagram

3.3 ACTIVITY DIAGRAM

An activity in Unified Modelling Language (UML) is a major task that must take place in order to fulfill an operation contract. Activities can be represented inactivity diagrams. An activity can represent: The invocation of an operation. A step in a business process.

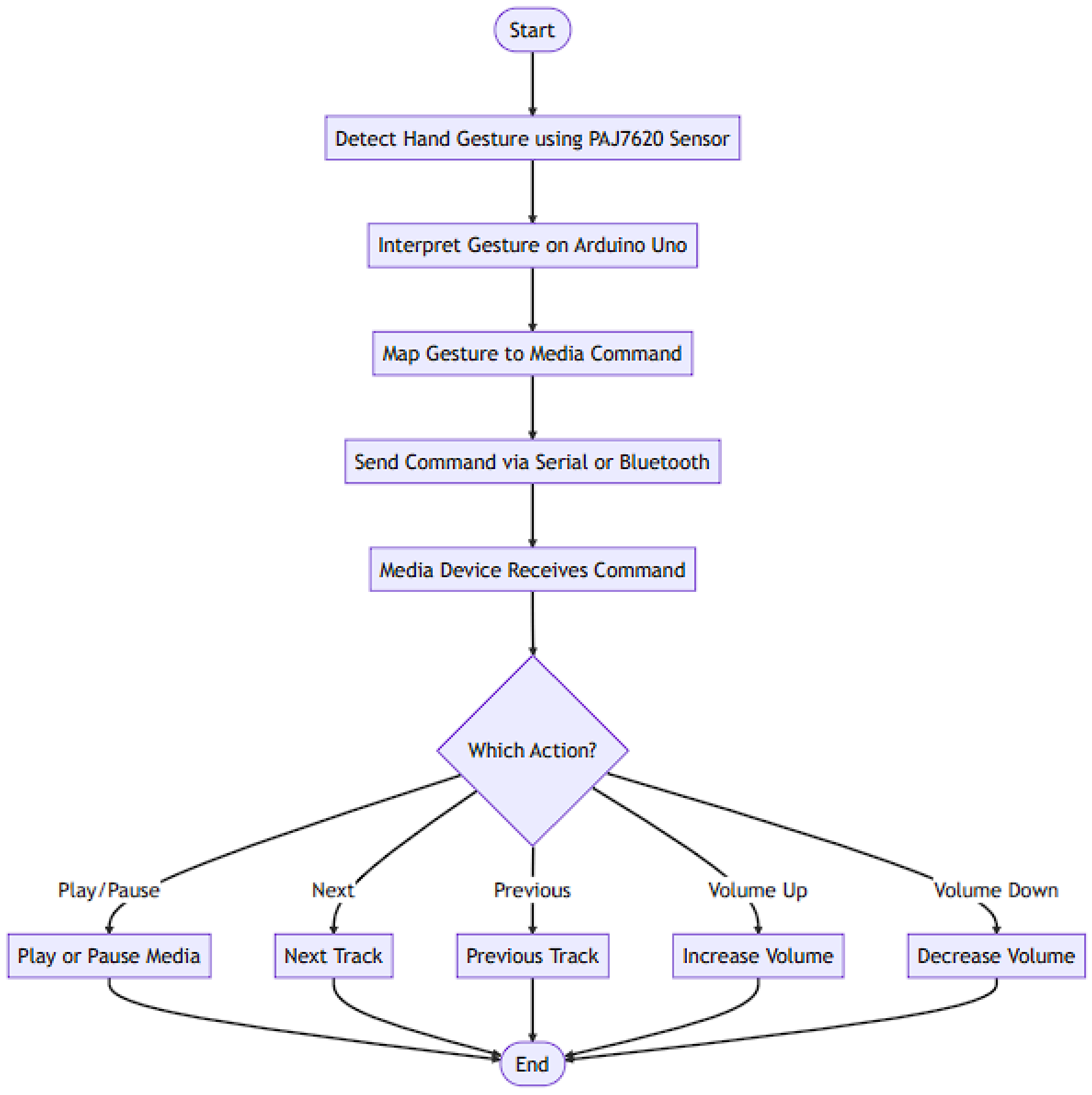

Start
Detect Hand Gesture using PAJ7620 Sensor
Interpret Gesture on Arduino Uno
Map Gesture to Media Command
Send Command via Serial or Bluetooth
Media Device Receives Command
Which Action?
Play/Pause
Next
Previous
Volume Up
Volume Down
Play or Pause Media
Next Track
Previous Track
Increase Volume
Decrease Volume
End

Figure 3.3 Activity Diagram

3.4 CLASS DIAGRAM

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modelling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity.

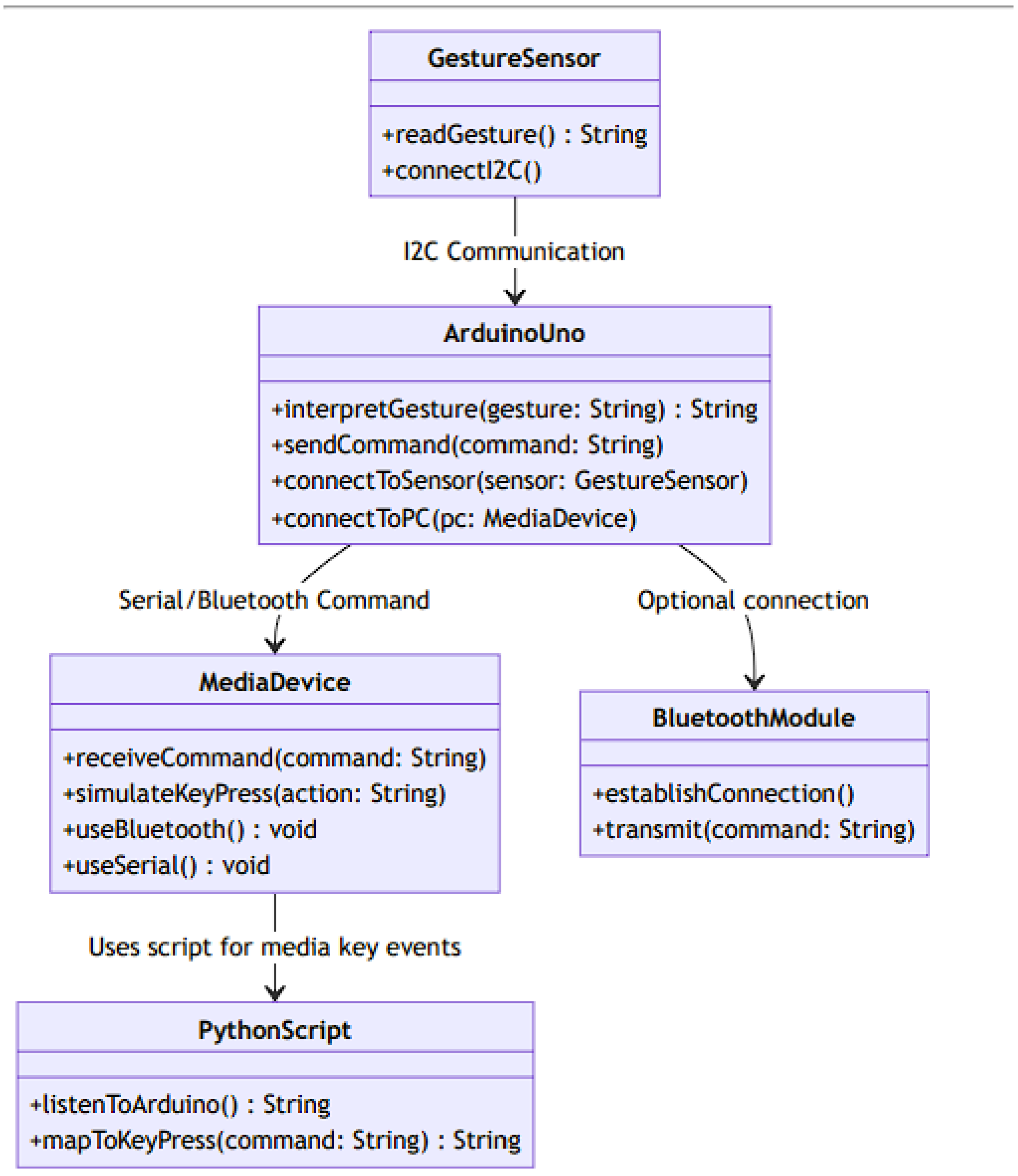

GestureSensor
+readGesture() : String
+connectI2C()
I2C Communication
ArduinoUno
+interpretGesture(gesture: String) : String
+sendCommand(command: String)
+connectToSensor(sensor: GestureSensor)
+connectToPC(pc: MediaDevice)
Serial/Bluetooth Command
Optional connection
MediaDevice
+receiveCommand(command: String)
+simulateKeyPress(action: String)
+useBluetooth() : void
+useSerial() : void
BluetoothModule
+establishConnection()
+transmit(command: String)
Uses script for media key events
PythonScript
+listenToArduino() : String
+mapToKeyPress(command: String) : String

Figure 3.4 Class Diagram

CHAPTER 4

MODULE DESCRIPTION

The Hand Gesture-Based Media Controller project comprises several integrated modules that work together to provide a seamless, touch-free user experience for controlling digital media. These modules are essential for ensuring real-time responsiveness, intuitive interaction, and system scalability. Below are detailed descriptions of each module:

4.1 HARDWARE MODULE

The hardware module is the backbone of the system and is responsible for gesture detection and initial data processing. It consists primarily of the PAJ7620U2 Gesture Sensor, the Arduino Uno microcontroller, and supporting components such as jumper wires, a breadboard, and a power supply.PAJ7620U2 Gesture Sensor: This sensor is capable of recognizing up to 9 directional gestures, including up, down, left, right, forward, backward, clockwise, counter-clockwise, and wave. It communicates with the Arduino via the I2C protocol, ensuring quick and efficient data transmission.
Arduino Uno: This microcontroller serves as the processing unit that reads gesture data from the sensor, maps the gesture to a predefined media command, and sends this command to a connected media device.
Connectivity Options: Depending on the setup, communication between the Arduino and the media device may use USB Serial or Bluetooth for wireless operation.

4.2 DATA COLLECTION AND PROCESSING MODULE

This module is responsible for collecting raw gesture data from the sensor and converting it into meaningful commands. The Arduino Uno plays a key role here.Gesture Interpretation: Once the gesture data is received from the sensor, the Arduino matches it with a corresponding media control command. For example, a wave gesture might be mapped to “Play/Pause,” while a left or right swipe may correspond to “Previous” or “Next” track.
Command Formatting: After gesture interpretation, the command is formatted into a

string or signal that can be sent via serial communication or Bluetooth.
Efficiency: The processing is lightweight and executed in real time, ensuring minimal latency between gesture recognition and media response.
This module ensures the system can respond quickly and accurately to user inputs, offering a smooth and intuitive experience.

4.3 ALERTING MODULE

While the primary goal of the system is gesture-based control, the alerting module adds a layer of user feedback, especially useful in cases of incorrect or unrecognized gestures.

LED/Buzzer Feedback: Optional alerting elements such as LEDs or buzzers connected to the Arduino can provide auditory or visual feedback to confirm gesture recognition or signal errors.
Serial Console Alerts: When connected to a PC, the system can print log messages in the Arduino IDE serial monitor or custom GUI to indicate system status, recognized gestures, or errors.
Future Enhancement: This module can be expanded to include voice feedback or haptic feedback for enhanced user interaction.

4.4 WEB APPLICATION MODULE

This module is optional and represents future scalability of the system. The idea is to build a lightweight web interface that displays gesture logs and device status remotely.

User Interface: A web dashboard can show the last gesture detected, the associated action, and system uptime.
Data Logging: The gestures and corresponding timestamps can be logged to a database for analysis and system improvement.
Accessibility: Through the web interface, users or caregivers can monitor and potentially control media playback from any device connected to the same network.
This module broadens the scope of the project from a simple hardware device to an IoT-based smart system.

4.5 INTEGRATION MODULE

The integration module ensures that all the above components work together in a synchronized and reliable manner.

Serial/Bluetooth Integration: This sub-module manages the data transfer between the Arduino and external devices like PCs or smartphones using either USB Serial or Bluetooth.
Python Integration: A Python script running on the PC listens to incoming commands and uses packages like pyautogui to simulate media keypress actions such as Play, Pause, Next, Previous, Volume Up, and Volume Down.
System Initialization: On startup, the integration module ensures all components are properly initialized and synchronized for smooth operation.
The integration module is essential for unifying hardware control, software response, and user feedback, delivering a robust and real-time media control experience.

CHAPTER 5

SAMPLE CODING

ARDUINO UNO CODE:

```
#include <Wire.h>
#include "paj7620.h"
#define GES_REACTION_TIME  800
#define GES_QUIT_TIME      1000
void setup() {
   Serial.begin(9600);
   Serial.println("\nGesture Sensor Test");
   uint8_t error = paj7620Init();
   if (error) {
      Serial.print("INIT ERROR, CODE: ");
      Serial.println(error);
   } else {
      Serial.println("INIT OK\nStart detecting gestures...");
   }
}
void loop() {
   uint8_t gestureData = 0;
   uint8_t waveData = 0;
   if (!paj7620ReadReg(0x43, 1, &gestureData)) {
      if (gestureData) {
         printGesture(gestureData);
      }
   }
   if (!paj7620ReadReg(0x44, 1, &waveData)) {
      if (waveData == GES_WAVE_FLAG) {
         Serial.println("Gesture: Wave");
      }
}
void printGesture(uint8_t data) {
   if (data == GES_RIGHT_FLAG) {
      Serial.println("Gesture: Right");
   } else if (data == GES_LEFT_FLAG) {
      Serial.println("Gesture: Left");
   } else if (data == GES_UP_FLAG) {
      Serial.println("Gesture: Up");
   } else if (data == GES_DOWN_FLAG) {
      Serial.println("Gesture: Down");
```

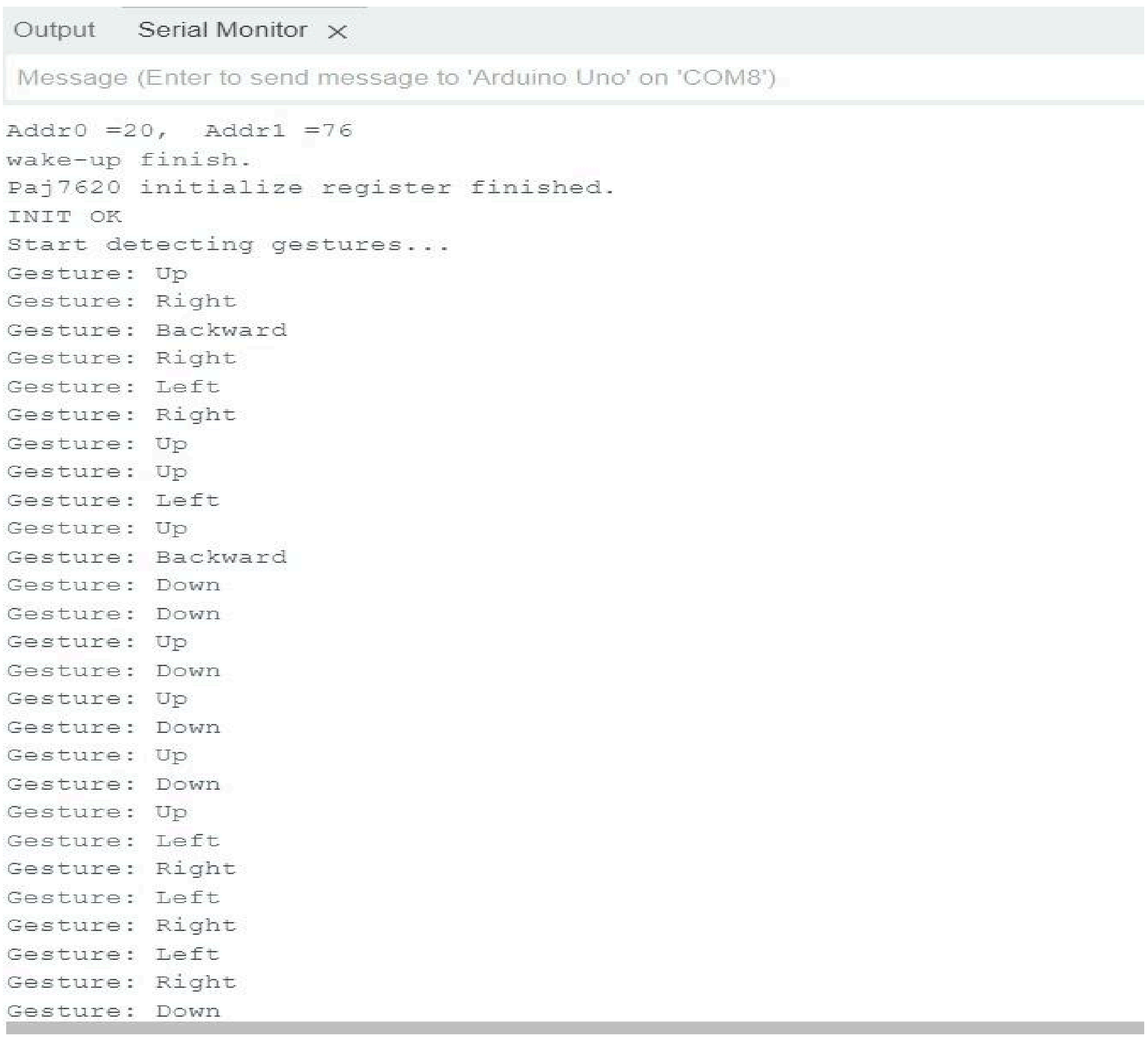
```
  } else if (data == GES_FORWARD_FLAG) {
    Serial.println("Gesture: Forward");
    delay(GES_QUIT_TIME);
  } else if (data == GES_BACKWARD_FLAG) {
    Serial.println("Gesture: Backward");
    delay(GES_QUIT_TIME);
  } else if (data == GES_CLOCKWISE_FLAG) {
    Serial.println("Gesture: Clockwise Rotation");
  } else if (data == GES_COUNT_CLOCKWISE_FLAG) {
    Serial.println("Gesture: Counter-Clockwise Rotation");
  } else {
    Serial.println("Unknown Gesture");
  }
}
```

PYTHON CODE:

```
import serial
import pyautogui
import time
arduino_port = "COM3"
baud_rate = 9600
try:
  ser = serial.Serial(arduino_port, baud_rate, timeout=1)
  print("Connected to Arduino on", arduino_port)
except:
  print("Failed to connect to Arduino.")
time.sleep(2)
gesture_action_map = {
  "Gesture: Right": lambda: pyautogui.press("nexttrack"),
  "Gesture: Left": lambda: pyautogui.press("prevtrack"),
  "Gesture: Up": lambda: pyautogui.press("volumeup"),
  "Gesture: Down": lambda: pyautogui.press("volumedown"),
  "Gesture: Forward": lambda: pyautogui.press("playpause"),
  "Gesture: Backward": lambda: pyautogui.press("playpause"),
  "Gesture: Wave": lambda: pyautogui.press("playpause"),
}
print("Listening for gestures...")
while True:
  if ser.in_waiting:
    line = ser.readline().decode('utf-8').strip()
    print("Received:", line)
    action = gesture_action_map.get(line)
    if action:
      action()
```

CHAPTER 6

SCREEN SHOTS



```
Addr0 =20,  Addr1 =76
wake-up finish.
Paj7620 initialize register finished.
INIT OK
Start detecting gestures...
Gesture: Up
Gesture: Right
Gesture: Backward
Gesture: Right
Gesture: Left
Gesture: Right
Gesture: Up
Gesture: Up
Gesture: Left
Gesture: Up
Gesture: Backward
Gesture: Down
Gesture: Down
Gesture: Up
Gesture: Down
Gesture: Up
Gesture: Down
Gesture: Up
Gesture: Down
Gesture: Up
Gesture: Left
Gesture: Right
Gesture: Left
Gesture: Right
Gesture: Left
Gesture: Right
Gesture: Down
```

Figure 6.1

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

Conclusion

The Hand Gesture-Based Media Controller successfully demonstrates an innovative, touch-free approach to controlling digital media. Utilizing the PAJ7620U2 gesture sensor and Arduino Uno, the system detects simple hand gestures and maps them to common media functions such as play/pause, next/previous track, and volume adjustments. The integration of hardware and software—supported by serial or Bluetooth communication and Python scripting—allows seamless control of media devices without any physical contact.

This project provides an accessible and hygienic interface that is especially useful for individuals with limited mobility or for environments where touch is impractical, such as healthcare facilities or laboratories. It enhances user independence and offers a futuristic alternative to traditional input methods. The system is built using cost-effective, widely available components, making it scalable and adaptable for various applications.

Additionally, the modular architecture of the system—including separate hardware, processing, alerting, integration, and optional web interface modules—enhances its adaptability. The system is flexible enough to be reconfigured or upgraded for different applications, such as controlling lighting, smart appliances, or robotic systems.

Future Enhancement:

Several enhancements can further improve the system:

1. **Advanced Gesture Recognition:** Implementing custom gesture detection or machine learning models could allow for more personalized and complex interactions.
2. **Audio or Haptic Feedback:** Adding sound or vibration feedback would provide real-time acknowledgment of gesture recognition, improving user experience.
3. **Web and Mobile Integration:** Creating a web dashboard or mobile app could allow remote control and configuration of gesture-command mappings.
4. **Power Optimization:** Making the device fully wireless and battery-powered would enhance portability and real-world usability.
5. **Multi-Device Support:** Adding the ability to control multiple devices or switch between control profiles would extend the system's flexibility.

In conclusion, this project lays the foundation for more advanced gesture-based control systems and has great potential for use in smart environments, assistive technology, and human-computer interaction applications.The Hand Gesture-Based Media Controller serves as a stepping stone toward more natural and intuitive human-machine interaction. Its success not only lies in its ability to simplify everyday tasks but also in its potential to empower users through contactless technology. With continuous refinement, integration of AI, and broader platform compatibility, this project can evolve into a versatile solution for smart environments, accessibility support, and futuristic user interfaces.

8.REFERENCES

References

1. Mitra, S., & Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *37*(3), 311–324. https://doi.org/10.1109/TSMCC.2007.893280

2. Jang, Y. S., Lee, S. H., & Kim, D. W. (2018). Hand gesture recognition using IR array sensor and Arduino in smart home interface. *IEEE International Conference on Consumer Electronics (ICCE)*, 1–2. https://doi.org/10.1109/ICCE.2018.8326272

3. DFRobot. (n.d.). *PAJ7620U2 Gesture Sensor SKU SEN0315 Wiki*. Retrieved from https://wiki.dfrobot.com/PAJ7620U2_Gesture_Sensor_SKU_SEN0315

4. PyAutoGUI Documentation. (n.d.). *Cross-platform GUI automation with Python*. Retrieved from https://pyautogui.readthedocs.io/

5. Last Minute Engineers. (n.d.). *Interfacing PAJ7620 Gesture Sensor with Arduino – A complete tutorial*. Retrieved from https://lastminuteengineers.com/paj7620-gesture-sensor-arduino-tutorial/