Needs of Smart Coffee Machine

Group 17 of Real time embedded coding, Glasgow University, 2025

Catalogue

1.Overview

This system combines biological signal detection (heart rate and blood oxygen levels) with advanced image recognition to assess the user's physical and emotional state. By analyzing these signals, the system assumes the user's mood and emotional state, such as happiness or not. Based on the mood analysis, the recognition module matches the user's emotional state with the ideal coffee cuisine and then automatically brews the selected drink. The goal is to enhance the user's mood or cater to their preferences, providing a personalized coffee experience that aligns with their emotional needs.

2.Version History

Version	Date	Changes Made	Author
V1.0 Stable Release	2025-04-21	Initial Stable release version	Xin Chen Jiaqi Fan Tang qi Yiyao Wang Junteng Zheng

3. System framework

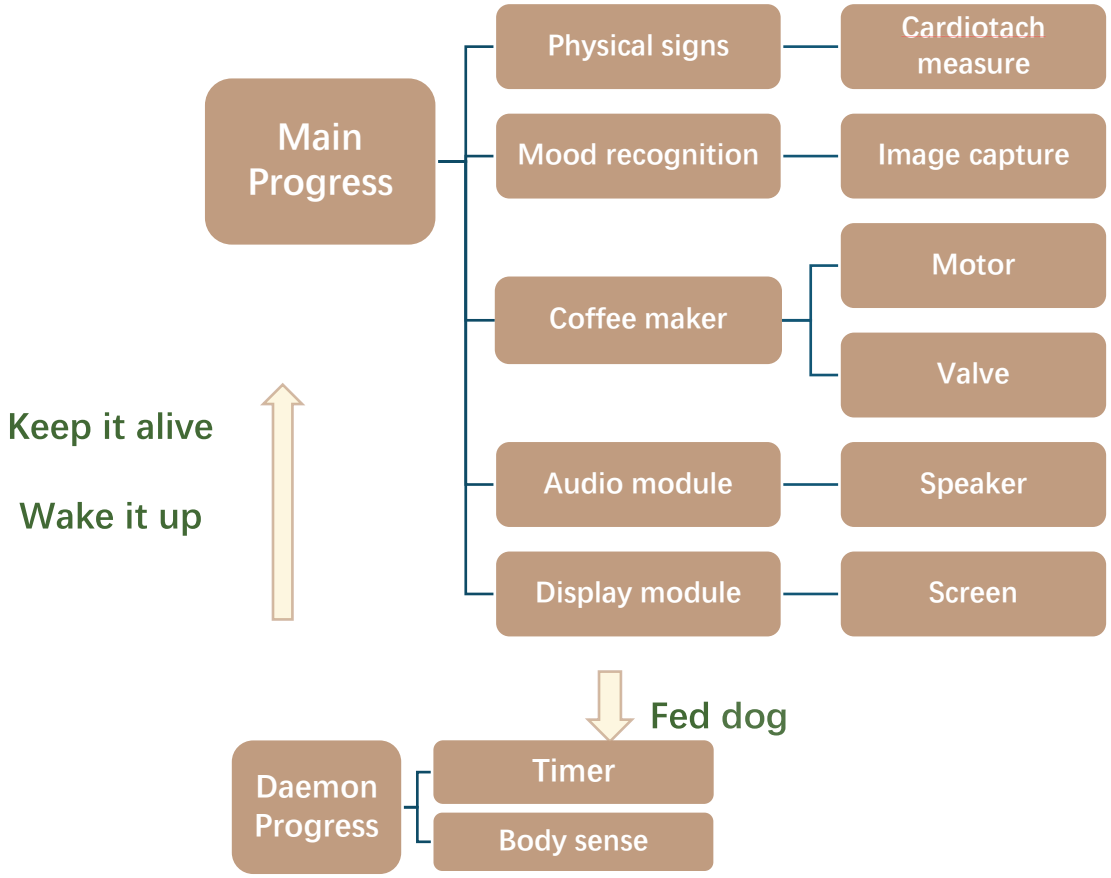
3.1 Hardware structure

3.1.1 GPIO of Sensor Thread

I2C: GPIO 38

3.2 Software framework

3.2.1 System Block Diagram



3.2.2 Software Process

The whole program consists of two processes, one of it is the Main Process, the rest of it is Deamon Progress.

3.2.2.1 Main Process

Function: Drives various modules, sensors, and actuators, coordinating the cooperation of functional modules. **Threads Included**:

- **Camera Thread**: Responsible for capturing facial images using the camera and performing initial image processing.
- **(Optional)Image Recognition Thread**: Uses YOLOv5 or OpenCV for facial expression recognition.
- **Sensor Thread**: Drives heart rate, blood oxygen, and cup weight detection functions and provides coffee ratio recommendations.
- **Actuator Thread**: Controls the operation of the speaker and liquid pump, calculating and adjusting the coffee ratio.

- **Audio Thread**: Controls the operation of the speaker and liquid pump, calculating and adjusting the coffee ratio.
- **Screen Interaction Thread**: Manages touchscreen interactions, displays data.
- **QT Main Thread:** Coordinates the work of functional threads.

3.2.2.2 Deamon Process

Function: Implements the watchdog function and infrared human body sensing function.
Threads Included:

- **Watchdog Function**: Runs independently, monitors the system, and performs a restart operation in case of system failure.
- **Infrared Sensor Function**: Detects if a person is approaching; when detected, it triggers an interruption and notifies the main process.

3.3 API Interface Definition

3.3.1 API of Daemon Process

(1) unsigned char FedDogHandle(unsigned char* receivedData)

This interface is provided to the main thread as an external interface for the main thread to feed the dog.

Input parameter

Name	Description
received_data	A fixed-length array for accepting communication data

Return value

Name	Description
unsigned char	Execution Status Return Value (Appendix B)

3.3.2 API of Main Process

3.3.2.1 API of QT Main Thread

(1) void WakeUpHandle()

This interface is provided to Daemon Process as an external interface for Daemon Process to wake up Main Process.

Input parameter

Name	Description

Return value

Name	Description

3.3.2.2 API of Sensor Thread

(1) unsigned char HeartbeatValueHandle()

This interface is provided to the main thread as an external interface of the sensor thread to

submit heartbeat value.

Input parameter

Name	Description

Return value

Name	Description
unsigned char	The return value is the heartbeat data obtained by parsing from the sensor. Unit: times per minute Range: 0~200. 255 is returned if no valid data can be obtained or if the heart rate exceeds 200 beats per minute.

(2) unsigned char SP2O_ValueHandle()
This interface is provided to the main thread as an external interface of the sensor thread to submit blood oxygen saturation data.

Input parameter

Name	Description

Return value

Name	Description
unsigned char	The return value is the oxygen saturation data obtained by parsing from the sensor. Unit: percent Range: 0~100. 255 is returned if no valid data can be obtained or if blood oxygen saturation exceeds 100% .

(3) unsigned char SelfCheckHandle()
This interface is provided to the main thread as an external interface of the sensor thread to submit running state of itself.

Input parameter

Name	Description

Return value

Name	Description
unsigned char	State return value (Appendix C)

3.3.2.3 API of Screen Interaction Thread

(1) void ScreenButtonClicked(unsigned char buttomNum)

This is a signal function.

This interface is provided to the main thread as an external interface of the screen interaction thread to submit widgets actions.

Input parameter

Name	Description
buttomNum	This date is the number of the button which was pressed.

Return value

Name	Description

3.4 State Machine

State	Description
Power on -Self check	After the system is powered on, it enters the self-test state to check the connection of related peripheral devices. If there is an exception, it will display exceptions. It will freeze in the self-test interface and be inaccessible until it is powered on and restarted again. **State change option:** To Sleep only. If Self check passed, system should automatically enter Sleep State.
Sleep	The Main Process sleep (low power consumption)until the daemon thread send wake-up signal. **State change option:** To Ready only. When receiving wake-up signal, Sleep state will change to Ready.
Ready	This state is used to continuously capture face information, and when clear face information is successfully captured, it will automatically freeze and send requests to user for entering Processing state. **State change option:** To Processing, Sleep or Remain Ready When received confirmation from user, system should into Processing state. If users say no system should remain in Ready state. If no user is detected for a period of time, system goes to Sleep.
Processing	This state contains several sub-states. Physical information gathering and Decision making.

	Physical information gathering In this stage, the collection of expression information has been completed. The system should continuously detect the input of the physical signal and display the corresponding value on the screen in real time. The physical quantity is frozen after the value enters stable state for 1 to 2 seconds. Ask users for permission to proceed to the next stage. **State change option:** Decision making, Remain Physical information gathering,sleep. When received confirmation from user, system should into Decision making. If users say no system should remain in R Physical information gathering. If no user is detected for a period of time, system goes to Sleep.
	Decision making At this stage, system will first prompt to user if it's a good day to drink coffee and provide a yes or no option. If the user chooses yes then system enters the recipe recommendation phase. If the user chooses no, system returns to the Ready state with comforting words. The system lists three recipes with different sugar ratio for the user to choose, and highlights the recommended recipe with voice prompts default. There should be a confirm button at this stage to complete the confirmation of the coffee recipe. **State change option:** Coffee Making, Sleep When received confirmation from user, system should into Decision making. If no user is detected for a period of time, system goes to Sleep.
Coffee Making	This stage does not accept any external instructions until the coffee is made. **State change option:** To Ready only. When the coffee making is finished, the prompt music will be played and system should enter Ready state automatically.

4. Needs

4.1 Base Functions

The basic functions of the smart coffee machine are to automatically detect the user's

physiological state by combining image recognition and biological signal sensor technologies and perform automated coffee brewing to meet the user's emotional and physiological needs. The specific basic functions include:

a. Image Recognition and Emotional Analysis: The system captures the user's facial expressions using a camera and analyzes the emotional state of the user through image recognition algorithms. The system must respond accurately and in real time to different facial expression changes.

b. Biological Signal Detection and Analysis: The system must be capable of obtaining the relevant data through sensors, such as infrared light, red light, etc., and process it through algorithms to obtain corresponding biological signal data, including heart rate and blood oxygen levels. Using the heart rate and blood oxygen sensors, the system must accurately determine the user's current physical condition to provide reference for the upcoming coffee brewing process.

c. Coffee Recommendation and Brewing: Based on the user's emotional and physical state, the system automatically recommends the appropriate coffee recipe and brews the coffee accordingly, including controlling the syrup ratio and other adjustments. The coffee brewed must match the user's physical condition.

d. State Management and Self-Check Function: The system must perform a self-check upon startup to detect whether all sensors and actuators are working properly. If the self-check fails, the system will enter a self-check state and display the corresponding error message, ensuring that the system does not start when the device is not fully prepared. Additionally, when there is no user interaction, the system should enter sleep mode to reduce power consumption and extend the device's lifespan.

4.2 UI Design

The user interface of the smart coffee machine serves as the foundation for user interaction with the device, and its design directly affects the user experience. Specific requirements include:

a. Simple and Intuitive Operation Interface: The operation interface should be simple and intuitive, allowing users to quickly understand and complete their tasks. The main interface should include key functional entrances such as biological signal displays, coffee recommendation buttons, and emotional confirmation buttons. The system should guide the user to confirm their emotional state, choose a coffee recipe, and other operations using a combination of icons and text. The main interface must ensure that information overload does not affect the user experience.

b. Emotional State Feedback: After the user's facial expression and biological signals are captured and analyzed, the system should display the user's current emotional state, such as happiness, fatigue, etc., through the UI.

c. Coffee Brewing and Interaction: Based on the emotional determination results, the system should determine the coffee recipe, brew the corresponding coffee, and display the current coffee recipe and brewing progress on the UI. The system should allow users to interact with the system through the UI, such as selecting a coffee recipe and confirming their choices.

4.3 Further Functions

In addition to the basic functions and UI design, the coffee machine should support some extended features to enhance the system's intelligence and user experience, such as:

a. User Personalization: The system should support user personalization, allowing users to adjust the coffee recipe according to their preferences and save their personalized settings.

b. Voice Interaction Function: The system should support voice interaction, allowing users to control the coffee machine through voice commands, enhancing the user experience.

c. Automatic Cleaning Function: The system can include an automatic cleaning and regular cleaning feature to ensure that the coffee quality is not compromised.

d. Remote Control Function: The system should have a remote-control feature, allowing users to control the coffee machine remotely via mobile devices or other equipment.

5. Specification of development

5.1 Process, Thread and APIs Management and Usage Guidelines & Testing criteria

5.1.1 Software framework

The various functional modules of the system (such as image recognition, sensor data acquisition, coffee brewing control, etc.) should have clearly defined responsibilities. Each module should only handle its core function, and the modules communicate through clear interfaces, reducing dependencies between them, ensuring the independence and scalability of each module.

5.1.2 Process and Thread Management

The system consists of 2 processes: the main process and the daemon process. The main process includes 5 threads, while the daemon process includes 2 threads, with each thread responsible for different tasks. The lifecycles of the threads are managed by the main process and the daemon process to ensure that each thread executes as needed, avoiding unnecessary resource consumption.

5.1.3 API Interface Management and Usage Specifications

Each API interface should have a single function, and the naming of the interfaces should accurately reflect their functionality while adhering to a unified naming convention and parameter structure, avoiding redundancy and over-design.

5.2 Pointer Management and Usage Guidelines & Testing criteria

- Ensure pointer validity to prevent dangling pointers, wild pointers, and double deletions.
- Minimize unnecessary dynamic allocation, preferring stack allocation or standard containers.
- Ensure safe pointer sharing in multi-threaded environments to prevent memory leak.
- All pointer definitions should be centralized in .h file/class declaration for unified management.
- Avoid public input pointer definitions in class declaration.
- To improve project maintainability and code readability, all share pointer definitions should be managed in a dedicated header file (e.g., pointers.h).

5.2.1Common Pointer Issues and Solutions

Issue	Solution
Wild pointers	Initialize to nullptr, avoid accessing NULL pointers
Dangling pointers	Use std::weak_ptr to prevent cyclic references
Thread safety issues	Use synchronization mechanisms (std::atomic, std::mutex)

5.3 Variable, function, macro naming conventions

Variable：Lower Camel Case

Function：Upper Camel Case (with underline if need)

Class: Upper Camel Case and connected by underline

Macro：All words should be in capital form

5.4 Comment & Description of Codes

5.4.1 An example of a function comment

```
/**
 * @brief Stop watching a line for status changes.
 * @param chip GPIO chip object.
 * @param offset The offset of the line to stop watching.
 * @return 0 on success, -1 on failure.
 */
int Gpiod_Chip_Unwatch_Line_Info(struct gpiod_Chip *chip, unsigned int offset);
```

5.4.2 An example of a code file comment

```
/*
* @file filename.cpp
* @brief Brief description of the file's purpose
* @details More detailed description if necessary.
* @author Your Name
* @date YYYY-MM-DD
* @version 1.0
* @license MIT/GPL/Apache-2.0 (or specify your preferred license)
* @copyright Copyright (c) YYYY, Your Organization
* @note Any additional notes about the implementation.
* @history YYYY-MM-DD [Your Name]: Initial version.
* - YYYY-MM-DD [Contributor Name]: Description of modification.
*/
```

Appendix A – Role description

- **User**: The primary person interacting with the system. The user provides biological data (via contact sensor on coffee machine) and visual data (via a camera). The system uses this data to assess their emotional state and automatically brews a coffee that matches their needs.
- **Biological Signal Sensor**: The wearable device or sensor responsible for measuring the user's heart rate and blood oxygen levels. It plays a crucial role in understanding the user's physical state, which is a key input to the emotional analysis.
- **Image Recognition Module**: This component processes the user's facial expressions and emotional cues from their visual input. It uses machine learning and computer vision to detect emotional states and supports the overall mood analysis.
- **Recognition Module**: Analyzes the biological and image data to calculate the emotional state (e.g., happiness, stress). This module determines the best coffee recommendation based on the user's mood, ensuring personalized coffee choices.
- **Brewing System**: A fully automated coffee machine that uses the recognition module's recommendations to brew the appropriate coffee drink based on the user's emotional state and preferences.

Appendix B – Execution status return value

Format: unsigned char(4bits)

Value	Description
0x00	exeEndNor Execution end normally.
0x01	executing Execution stills in processing
0x02	inputterror One or more input values do not meet the input requirements
0x03	nullPointWiInput Null points within input value
0x0F	exeFailed The terminal executor failed to execute

Appendix C –State return value

Format: unsigned char(8bits). The higher four bits are used to indicate the current thread running, the meanings of lower four bits are shown as below.

Value	Description
0x00	runNor Running normally.
0x01	initialising Initialising
0x0F	error Some error happens