Module Interface Specification for Mechtronics Enigeering

Team 32, Wingman, SmartVault
Edward He
Erping Zhang
Guangwei Tang
Peng Cui
Peihua Jin

April 5, 2023

1 Revision History

Date	Version	Authors
2023-01-18	1.0	Edward He, Erping Zhang, Guangwei Tang, Peng Cui, Peihua Jin
2023-01-30	2.0 Update MIS of Login Module and Information Exxtraction Module	Peng Cui
2023-01-30	2.1 Update MIS of Communication Port 1 Module and Communication Port 2 Module	Erping Zhang, Guangwei Tang
2023-02-01	2.2 Update File Storage Module and Image Processing Module	Edward He, Peihua Jin
2023-03-10	2.3 Update Motor Control Module	Erping Zhang
2023-04-04	3.0 Check for grammar and structure	Peng Cui

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/Edwardhyw/smartVault/tree/main/docs/SRS

Contents

3 Introduction

The following document details the Module Interface Specifications for SmartVault, a Mechatronics system that aims to assist users in finding their belongings. The functions and variables used in this project will be specified. The complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/Edwardhyw/smartVault.

4 Notation

The following table includes the notation we will need in the motor control module.

Data Type	Notation	Description
float	τ	The time gap between each motor step motion
float	ω	The angular velocity of the motor movement
float	θ	The angule degree of the motor movement

5 Module Decomposition

The Whole project is decomposed into two main parts: Hardware Module and Software Module. The Hardware Module is mainly designed for motor control and Hardware-Software Communication. The Software Module is designed for main human interface. It is the main part of the design of the project. These two modules are further divided into seven parts. The first part of the decomposition is Communication Port 2 which is used for sending signals to the software part of the project. The second decomposition of the Hardware Module is Motor Control, it is used to control the real-time position of the camera so that the human body is within the centre of the screen taken by the camera.
When it comes to the Software Module, The first part is Login Module, it is designed ask the user log into the running program. It can also be treated as the first barrier of protection of the user's privacy. If the person does not enter the correct username and password, he or she cannot enter the search window, which means the private information is blocked from the user. The second part is the Information Storage Module. It is used to store the screenshot of the images taken by the camera, which shows the position information of the object. The third part is Image Processing Module. It helps analyze the image to get the key information that is useful to the user. Currently this module contains the human body detection method and object movement method. The forth part is Information Extraction Module. It is related to the Information Storage Module and is used to choose pictures from the database which meets the information required by the user. The fifth part decomposed

from the Software Module is Communication Port 1. It can be treated as the connection bridge between the hardware and software.

Level 1	**Level 2**
Software-Module	Login Information Storage Image Processing Information Extraction Communication Port 1
Hardware-module	Communication Port 2 Motor Control

Table 1: Module Hierarchy

6 MIS of Login Module

Responsible for interacting with the user. The user can only start the program successfully with correct username and password. It can also provide technical support information to the user.

6.1 Uses

The Login Module uses python tkinter library to creates different windows, input boxes, buttons, and so on. It also interacts with the Information Extraction Module for the user to find their desired object location.

6.2 Syntax

6.2.1 Constants

Table 2: Constants used in Login Module

Name of Constants	**Value**
Username	Initially determined by the user
Password	Initially determined by the user
Technical Support Window Geometry	Initially 450*300
Login Window Geometry	Initially 450*300
Location	Initial file storage location determined by the user
Software Developer Contact Information	Email of members of this project

6.2.2 Access Programs

Name	Input	Output	Description
Tech	N/A	Technical Support Window	To present the technical support window to the user.
Input	N/A	N/A	To check the correctness of the input username and password. If both are correct, then the Information Extraction will be called.
content	N/A	N/A	To fill the Login Window with different buttons, input boxes, and texts.

6.3 Semantics

6.3.1 State Variables

Table 3: State Variables used in Login Module

Name of State Variables	Description
self.win	Welcome window itself
self.title	The title of the window
self.size	The size of this window
self.user_input	Username input box
self.pass_input	Password input box
self.location	File storage location

6.3.2 Environment Variables

Environment Variables	Description
Input Username	The input value in Username Box
Input Password	The input value in Password Box

6.3.3 Assumptions

N/A

6.3.4 Access Routine Semantics

Tech():

- transition: Create a new window and sees self.win as the toplevel window.
- output: N/A

- exception: N/A

Input():

- transition: Updating and then checking self.user_input and self.pass_input.
- output: If the accessing to information Extraction Module is successful.
- exception: Warning text will be shown if the username or password is wrong.

content():

- transition: Use self.title and self.size inside the self.win to create the welcome window.
- output: The output welcome window.
- exception: N/A

6.3.5 Local Functions

N/A

7 MIS of File Storage Module

Responsible for storing the desired frames or photos into certain directory. The module mainly consists of two functions, createFolder() and frameStorage(), which are used to initialize the basic directory with folders and save the frames respectively.

7.1 Uses

The File Storage Module imports OpenCV, OS, datetime, numpy libraries. It also communicates with the Image Processing Module in order to transmit and save the desired data. Additionally, it interacts with Information Extraction Module to get access into User Interface and collect information from the main window.

- OpenCV: Writing in function
- OS: Mainly used to process directory and path
- datatime: Serve for fetching date and time
- numpy: Used to create empty array

7.2 Syntax

7.2.1 Constants

Table 4: Constants used in File Storage Module

Name of Constants	**Value**
img_item	100
img_loca	100

7.2.2 Access Programs

Name	Input	Output	Description
createFolder	directory	N/A	To create 3 basic folders under the desired directory
frameStorage	directory, itemNum, item, location, key	N/A	To save the frames

7.3 Semantics

7.3.1 State Variables

Table 5: State Variables used in File Storage Module

Name of State Variables	Description
directory	paths for creating folders
fileName	The name of saved frames
current$_t ime$	To get the actual time
path1	path for saving frames into 'item' folder
path2	path for saving frames into 'location' folder
img_item	image list for 'item' folder
img_loca	image list for 'location' folder
itemNum	image inside 'location' folder
item	image inside 'item' folder
location	image inside 'location' folder
key	choose whether to insert new item or update recorded items

7.3.2 Environment Variables

N/A

7.3.3 Assumptions

The frames sent from Image Processing Module are assumed to be correct all the time. The File Storage Module itself has no ability to distinguish whether the data are expected.

7.3.4 Access Routine Semantics

createFolder():

- transition: Try to create a main folder with 'item' and 'location'. If these three folders have already built, then do nothing.
- output: N/A
- exception: If the directory is invalid, do nothing.

frameStorage():

- transition: To process the transmitted data or frames through the input parameters of keys. For the insertion of new item, write into both 'item' folder and 'location' folder. For the updating of recorded items, write into 'location' folder only.
- output: Frames will be sent to correct folders
- exception: N/A

7.3.5 Local Functions

N/A

8 MIS of Information Extraction Module

Responsible for extracting information from the File Storage Module based on the inputs given by the user.

8.1 Uses

The Information Extraction Module python tkinter library to access some window manipulations. It also uses python datetime and PIL libraries to extract images based in the timing input given by the user. It will only extract pictures from the File Storage Module nased on the file path initialized at the start of the program

8.2 Syntax

8.2.1 Constants

Table 6: Constants used in Information Extraction Module

Name of Constants	**Value**
Location	Initial file storage location determined by the user
Window title and sizes	determined by the software developer
Time slot choosing slot	1 day, 7 days, 30 days, 100 days and 365 days

8.2.2 Access Programs

Name	Input	Output	Description
content	N/A	N/A	Filling the window with different buttons, input boxes or other features.
search	button pressed in the window	search window	Based on the time values selected by the user, it calls the search window.
sea	N/A	N/A	To fill the search Window with different buttons, input boxes, and texts.
nex_img	N/A	N/A	The window shows the next image suitable for the time value. If it is at the end of the image list, the first image will be presented.
pre_img	N/A	N/A	The window shows the previous image suitable for the time value. If it is in the first image, the last image will be shown.
find_img	N/A	N/A	After the user press the Find button, the result find window will be shown.
fin	N/A	N/A	Creating the result window with the location image shown on that window.

8.3 Semantics

8.3.1 State Variables

Table 7: State Variables used in Information Extraction Module

Name of State Variables	Description
self.window	The choose window itself
self.variable	Selecting box presents in the window
self.choose_list	A list of time variables the user can choose
self.loca	File storage location
self.sear	The search window itself
self.Item_dir	The directory of the item file
self.label	Label used in the window
self.image_sort	Sorting number of the image
self.imge_list	List of images chosen from the item file
self.image_name	The name of the item image
self.today	The date of today
self.date_sub	time variable chosen by the user
self.name	The name of the image chosen by the user
self.find	The result window itself
self.Location_dir	The path of the location file
self.image_sort(find)	The sorting number of location image
self.image_flag	flag for the image
self.image_name(find)	The name of the location image

8.3.2 Environment Variables

N/A

8.3.3 Assumptions

N/A

8.3.4 Access Routine Semantics

content():

- transition: Create a new window based on window state variable.
- output: A window presented to the user.
- exception: N/A

search():

- transition: Updating self.variable and pass this variable to self.sear to create the search window.
- output: The search window
- exception: N/A

sea():

- transition: Create the search window based on title and geometry. It will also select a list of images from the File Storage Module based on the time parameters passed to it and shows the first image.
- output: The output search window with image shown.
- exception: Nothing will be shown if the time variable is not applicable.

next_img():

- transition: Update self.image_sort by adding 1 and choose the next image from self.image_list. If the sort number is equal to the length of the list -1, set self.image_sort to zero.
- output: The change of image shown in the window.
- exception: N/A

pre_img():

- transition: Update self.image_sort by subtracting from 1 and choose the previous image from self.image_list. If the sort number is equal to zero, set self.image_sort to length of the list -1.
- output: The change of image shown in the window.
- exception: N/A

find_img():

- transition: Destroy the current search window and create the result window and pass the selected image name to that window.
- output: The change of window shown in the screen.
- exception: N/A

fin():

- transition: Compare self.image_name(find) with the parameters passed to this window. If the item numbers are the same, present this location image.
- output: The image shown in the window.
- exception: N/A

8.3.5 Local Functions

N/A

9 MIS of Image Processing Module

The Image process module is the main function of the project. It will take two pictures, using the image subtracting method to find the object that is moved over the time interval and pass the result to File Storage Module.

9.1 Uses

- Opencv library for image processing
- scikit image library for calculating difference between images
- File Storage Module

9.2 Syntax

9.2.1 Constants

Table 8: Constants used in Image Processing Module

Name of Constants	**Value**
kernel	9x9 matrix of value 1
item_area	300
folder	target folder for storing images

9.2.2 Access Programs

Name	Input	Output	Description
load_images_from_folder	folder	images	load all images from
item_match	item,img	Boolean True/False	compare the item wit there is an identical i
item_comp	itemList1, itemList2,img_list,img	itemList1,loc_list	sort images from the put them into itemLi cordingly.
extract_item	path, orimg1,orimg2	N/A	extracting moving ite difference between th plying image proces function for the imag ule
item_store	itemList, positionList, directory	N/A	identify whether iter or updating its record item storing is happe tion
begin_tracking	path	N/A	begin human trackin

9.3 Semantics

9.3.1 State Variables

Name	value	Description
self.flag	N/A	flag for human detection

9.3.2 Environment Variables

N/A

9.3.3 Assumptions

N/A

9.3.4 Access Routine Semantics

begin_tracking():

- transition: If self.flag > 0
- output: call extract_item(), pass two frames for item extraction
- exception: N/A

extract_item():

- transition: after all items were found in the frame, call item_store(), call item_comp

- output: N/A
- exception: N/A

item_store():

- transition: call load_images_from_folder, check if item_match is true, call frameStorage(), if flag is 0, also call frameStorage()
- output: N/A.
- exception: N/A

item_match():

- transition: N/A
- output: Boolean expression. If items matched, return true, if not, return false
- exception: N/A

item_comp():

- transition: call item_match
- output: list of item and list of location
- exception: N/A

load_images_from_folder():

- transition: N/A
- output: list of images from the storage folder.
- exception: N/A

9.3.5 Local Functions

N/A

10 MIS of Communication Port 1 Module

10.1 Uses

Communication port 1 is a function that runs on laptop, used to send the motor control signal to communication port 2, which is handled by the Arduino board.

10.1.1 Constants

10.2 Syntax

Table 9: Constants used in Communication Port 1 Module

Name of Constants	Value
Time gap	The time delay between each control signal
Baud rate	The communication baud rate between communication port 1 and port 2

10.2.1 Access Programs

Name	Input	Output	Description
begin_ tracking	N/A	N/A	Use the result from the human tracking algorithm and send the control signal
Back_to _origin	N/A	N/A	Send a special control signal to the Arduino, to make the camera move to the original angle

10.3 Semantics

10.3.1 State Variables

Table 10: State Variables used in Communication Port 1 Module

Name of State Variables	Description
self.port	The com port is used for the serial communication
self.baudrate	The baud rate of the serial communication
self.servo	The indicator of servo connection

10.3.2 Environment Variables

N/A

10.3.3 Assumptions

The communication between port 1 and port 2 doesn't have any delay. Once the camera captures the human motion, it will send the signal with no time delay.

10.3.4 Access Routine Semantics

back_to_origin:

- transition: if True
- output: Send a special reset signal to Communication Port 2 Module
- exception: N/A

begin_tracking:

- transition: N/A
- output: send signal and data to Communication Port 2 Module
- exception: N/A

10.3.5 Local Functions

N/A

11 MIS of Communication Port 2 Module

11.1 Uses

Receive control signal from communication port 1. The program is located in the Arduino control board.

11.2 Syntax

Table 11: Constants used in Communication Port 2 Module

Name of Constants	Value
Time gap	The time delay between each control signal
Baud rate	The communication baud rate between communication port 1 and port 2

11.2.1 Constants

N/A

11.2.2 Access Programs

Name	Input	Output	Description
serial.print	N/A	N/A	Send a character using serial communication
serial.begin	N/A	N/A	Initialize the serial communication
serial.read	N/A	N/A	Read a character from serial communication
serial.end	N/A	N/A	Turn off the serial communication and clean the buffer

11.3 Semantics

11.3.1 State Variables

N/A

11.3.2 Environment Variables

N/A

11.3.3 Assumptions

The serial communication can be initialized and end very fast, the time to initialize and end the serial port can be ignored

11.3.4 Access Routine Semantics

serial.print:

- transition: if True
- output: send a character to Communication Port 1 Module
- exception: N/A

serial.begin:

- transition: if True
- output: Initialize the seral communication
- exception: N/A

serial.read:

- transition: if True
- output: Return a character from the serial buffer
- exception: N/A

serial.end:

- transition: N/A
- output: Turn off the serial communication
- exception: N/A

11.3.5 Local Functions

N/A

12 MIS of Motor Control Module

12.1 Uses

Get the control signal from communication port 2 and control the servo motor through a PWM pin on the Arduino.

12.2 Syntax

12.2.1 Constants

Table 3: Constants Variables

Constant Name	Constant Type	Value	Units	Comment
Angle per step	float	2	Degree/step	This is the angle movement stepper motor will move after 1 signal
Height of the Camera	float	10	mm	This is the distance between the lens of camera and the bottom of the mount
Resolution	Integer	1920x1080	Pixel	This is the resolution of the camera
Arduino input voltage	float	9.0	V	This is the input voltage of the Arduino board

12.2.2 Access Programs

Name	**Description**
back_to_origin	rotate the camera to the original angle
xincrease	rotate the camera in the clockwise horizontal direction for 1 step
xdecrease	rotate the camera in the horizontal counter-clockwise direction for 1 step
yincrease	rotate the camera in the vertical clockwise direction for 1 step
ydecrease	rotate the camera in the vertical counter-clockwise direction for 1 step

12.3 Semantics

12.3.1 State Variables

Table 12: State Variables used in Motor Control Module

Name of State Variables	Description
waittime	The time delay between each step motion, this is for protecting the motor from overload burning
$\text{ang}_p er_s tep$	The angle for each movement of the servo motor
$\text{origin}_a ng$	The original angle when the system starts
upper	The upper limit of the angle that the motor can rotate
lower	The lower limit of the angle that the motor can rotate

12.3.2 Environment Variables

N/A

12.3.3 Assumptions

The servo motor move in a absolute angle, which means each step will move the angle very accurately.

12.3.4 Access Routine Semantics

xincrease():

- transition: if True
- output: Rotate the camera in the clockwise horizontal direction for 1 step
- exception: N/A

xincrease():

- transition: if True

- output: Rotate the camera in the clockwise horizontal direction for 1 step
- exception: N/A

xdecrease():

- transition: if True
- output: Rotate the camera in the counter-clockwise horizontal direction for 1 step
- exception: N/A

yincrease():

- transition: if True
- output: Rotate the camera in the vertical clockwise direction for 1 step
- exception: N/A

ydecrease():

- transition: if True
- output: Rotate the camera in the counter-clockwise horizontal direction for 1 step
- exception: N/A

back_to_origin():

- transition: if True
- output: Rotate the camera to the original angle that has pre-set into the system
- exception: N/A

12.3.5 Local Functions

N/A

13 Appendix

[Extra information if required —SS]

References