Module Interface Specification for Mechatronics

Team #20, OpenASL
Robert Zhu zhul49
Zifan Meng mengz17
Jiahui Chen chenj194
Kelvin Huynh huynhk12
Runze Zhu zhur25
Mirza Nafi Hasan hasanm21

April 6, 2023

1 Revision History

Date	Version	Notes
January 18, 2023	1.0	Everyone - Initial MIS Draft
April 05, 2023	2.0	Everyone - Revised MIS

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/kelhuynh/OpenASL/blob/main/docs/SRS/SRS.pdf

Contents

3 Introduction

The following document details the Module Interface Specifications for OpenASL, a device developed with the aim of translating sign language into text-to-speech, with the purpose of helping members of the deaf and mute community communicate with those who do not know sign language.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/kelhuynh/OpenASL/.

4 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding Module	Video Capture Module Hardware Hiding Module
Behaviour-Hiding Module	User Interface Module Data Processing Module - Communicates with ML module with data from coordinate normalization module Data Collection Module - Communicates with ML module to update dataset Coordinate Export Module - Read data from video capture and stores into file Gesture Detection Module - Controller (ties everything together) Dynamic Gesture Detection Module
Software Decision Module	Video Analysis Module - requires data to be used Machine Learning Module Dynamic Machine Learning Module Coordinate Normalization Module

Table 1: Module Hierarchy

5 MIS of Gesture Detection Module (M1)

5.1 Module

motionTrack

5.2 Uses

Video Capture, Coordinate Normalization, Coordinate Export, Video Analysis, Keypoint Classification, Text to Speech

5.3 Syntax

5.3.1 Exported Constants

Name	In	Out	Exceptions
results	image	Object	-
hand_landmarks	-	Tuple of tuples	-
handedness	-	R	-

5.3.2 Exported Access Programs

Name	In	Out	Exceptions
motionTrack	-	-	cv2.error

5.4 Semantics

5.4.1 State Variables

None

5.4.2 Environment Variables

f - file variable for coordinate export purposes

5.4.3 Assumptions

None

5.4.4 Access Routine Semantics

motionTrack():

- output: video consisting of overlay for hand gesture classification into ASL

- exception: exc := cv2.error

6 MIS of Coordinate Normalization Module (M2)

6.1 Module

Coordinate Normalization

6.2 Uses

Video Capture

6.3 Syntax

6.3.1 Exported Constants

Name	In	Out	Exceptions
pre_processed_landmark_list	landmark_list	Tuple of tuples	-

6.4 Semantics

6.4.1 State Variables

None

6.4.2 Environment Variables

None

6.4.3 Assumptions

None

6.4.4 Access Routine Semantics

pre_process_landmark(landmark_list):

- output: tuple of 20 tuples consisting of x and y coordinates for each hand joint
- exception: exc := ListIndexOutofBounds

6.4.5 Local Functions

- __calc_landmark_list

7 MIS of Coordinate Export Module (M3)

7.1 Module

Coordinate Export

7.2 Uses

Coordinate Normalization

7.3 Syntax

7.3.1 Exported Constants

Name	In	Out	Exceptions
keypoint.csv	-	File containing normalized coordinates	-

7.4 Semantics

7.4.1 State Variables

None

7.4.2 Environment Variables

None

7.4.3 Assumptions

None

7.4.4 Local Functions

- __make_csv

8 MIS of Video Capture Module (M4)

8.1 Module

Video Capture

8.2 Uses

None

8.3 Syntax

8.3.1 Exported Constants

Name	In	Out	Exceptions
success	-	R	-
image	-	Object	-

8.4 Semantics

8.4.1 State Variables

None

8.4.2 Environment Variables

success - indicates that the camera input is initialized for use

8.4.3 Assumptions

There is an available camera connected to the system

9 MIS of Video Analysis Module (M5)

9.1 Module

Video Analysis

9.2 Uses

Coordinate Normalization, Video Capture

9.3 Syntax

9.3.1 Exported Access Programs

Name	In	Out	Exceptions
draw_bounding_rect	-	image	-
draw_landmarks	-	image	-
draw_info_text	-	image	-

9.4 Semantics

9.4.1 State Variables

None

9.4.2 Environment Variables

None

9.4.3 Assumptions

None

9.4.4 Access Routine Semantics

draw_bounding_rect(self, use_brect, image, brect):

- output: image with overlaid bounding rectangle around hand
- exception: exc := None

draw_landmarks(self, image, landmark_point):

- output: image with overlaid hand joints and connections
- exception: exc := None

draw_info_text(self, image, brect, handedness, hand_sign_text):

- output: image with overlaid classifier label
- exception: exc := None

9.4.5 Local Functions

- __calc_bounding_rect

10 MIS of Dynamic Gesture Detection Module (M6)

10.1 Module

Gesture Detection

10.2 Uses

Video Capture, Coordinate Normalization, Video Analysis

10.3 Syntax

10.3.1 Exported Constants

-

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
mediapipe$_{d}$*ection*	Image: frame from CV2 Model: Mediapipe model that processes the image frame	Image: Processed Image frame Results: Processed Data from Mediapipe model	cv2.error

10.4 Semantics

10.4.1 State Variables

Image: refers to a single frame of a video, which is read from a cv2 object

10.4.2 Environment Variables

Cap: an instance of the cv2.VideoCapture class, which is used to capture video from a camera device connected to the system

10.4.3 Assumptions

None

10.4.4 Access Routine Semantics

mediapipe_detection(): takes image frame and model as input and processes the image frame through the model, and returns the processed image frame and results to make detections in the video feed

sel_mode(): takes a key and mode as input to change between different modes of operation

cap.read() used to read frames from the video capture device

11 MIS of Data Processing Module (M7)

11.1 Module

Keypoint Classification

11.2 Uses

Coordinate Export, Machine Learning

11.3 Syntax

11.3.1 Exported Constants

Name	In	Out	Exceptions
result_index	landmark$_list$	R	ListIndexOutofRange

11.4 Semantics

11.4.1 State Variables

None

11.4.2 Environment Variables

None

11.4.3 Assumptions

None

11.4.4 Local Functions

- __call__

12 MIS of Data Collection Module (M8)

12.1 Module

DataCollection

12.2 Uses

Coordinate Normalization, Coordinate Export

12.3 Syntax

12.3.1 Exported Constants

Name	In	Out	Exceptions
actions	Number of Gestures	-	-

12.3.2 Exported Access Programs

-

12.4 Semantics

12.4.1 State Variables

No_sequences: Represents the number of sequences to be recorded for each action

12.4.2 Environment Variables

Dirmax: Stores the highest amount of files within a gesture

DATA_PATH: a list of directories where executable files are located

12.4.3 Assumptions

None

12.4.4 Access Routine Semantics

UserInput(): Takes user input and returns a string “action” and checks if action is present in the action list. If it is, it finds the max value of the directories present in the path and creates a directory with a name greater than the max value

13 MIS of Machine Learning Module (M9)

13.1 Module

ML Train

13.2 Uses

Coordinate Export

13.3 Syntax

13.3.1 Exported Constants

Name	In	Out	Exceptions
keypoint_classifier.hdf5	-	Hierarchical Data Format file	-
keypoint_classifier.tflite	-	ML model	-

13.4 Semantics

13.4.1 State Variables

None

13.4.2 Environment Variables

None

13.4.3 Assumptions

None

14 MIS of Dynamic Machine Learning Module (M10)

14.1 Module

machineLearning

14.2 Uses

Keypoint Classification, Coordinate Export, Coordinate Normalization, Training

14.3 Syntax

14.3.1 Exported Constants

-

14.3.2 Exported Access Programs

-

14.4 Semantics

14.4.1 State Variables

X_train: Numpy array containing the training data for the machine learning model.

X_test: Numpy array containing the testing data for the machine learning model.

y_train: Numpy array containing the labels for the training data.

y_test: Numpy array containing the labels for the testing data.

model: Keras sequential model object representing the machine learning model. Its weights are updated during training.

14.4.2 Environment Variables

None

14.4.3 Assumptions

None

14.4.4 Access Routine Semantics

np.loadtxt(): loads gesture labels into an array.

os.listdir(): lists the contents of a directory.

np.load(): loads a NumPy array from a binary file.

train_test_split(): splits the dataset into training and testing sets.

Sequential(): initializes the machine learning model.

model.add(): adds layers to the machine learning model.

model.compile(): configures the machine learning model for training.

model.fit(): trains the machine learning model on the dataset.

model.predict(): uses the trained model to make predictions on the test set.

np.argmax(): gets the indices of the maximum values along an axis of a NumPy array.

multilabel_confusion_matrix(): computes a confusion matrix for multiclass classification.

model.save(): saves the trained machine learning model to a file.

15 MIS of User Interface Module (M11)

15.1 Module

UI Module

15.2 Uses

Video Capture, Text to Speech

15.3 Syntax

15.3.1 Exported Constants

Name	In	Out	Exceptions
Images	Video Capture	-	-
frames	Video Capture	-	-
mode	Key Input	-	-
text_string	Machine Model	-	-

15.3.2 Exported Access Programs

-

15.4 Semantics

15.4.1 State Variables

spot: a global variable that keeps track of the position in the text string where the line should be split into two lines. It is initially set to 0.

short: a boolean variable that determines whether the second line of text should be displayed. It is initially set to True.

text: a string that contains the FPS and resolution of the video input. It is constructed based on the image parameter.

line1: a string that contains the first line of the text to be displayed on the UI.

line2: a string that contains the second line of the text to be displayed on the UI.

text_size: a tuple that contains the size of the text to be displayed.

text_w: an integer that contains the width of the text to be displayed.

text_h: an integer that contains the height of the text to be displayed.

multiplier: an integer that determines how many lines of text should be displayed. It is initially set to 1.

15.4.2 Environment Variables

None

15.4.3 Assumptions

None

15.4.4 Access Routine Semantics

None

16 MIS of Hardware Hiding Module (M12)

16.1 Module

Hardware Hiding

16.2 Notes

This module implements hardware hiding by utilizing operating system abstraction. It is done by the operating system of the computer or Raspberry Pi.