LerntiaControl

Sarah El-Sherbiny

Jan 26, 2020

CONTENTS:

CHAPTER

ONE

SRC

1.1 MainWindow module

class MainWindow.**Ui_MainWindow**
Bases: object

The main window of the user interface (ui). It defines all ui elements.

retranslate_ui(*main_window*)
Retranslate the view of the ui.

Parameters **main_window** – the main window of the ui

Returns none

setup_ui(*main_window*)
Setup all ui elements.

Parameters **main_window** – the main window of the ui

Returns none

MainWindow.**activate_pause_button**()
Update the appearance of the pause button and show a default image when the application is stopped.

Returns none

MainWindow.**activate_start_button**()
Update the appearance of the start button when the application is started.

Returns none

MainWindow.**change_mode**()
Change the active mode. There are two modes available: 'normal' and '2-gestures'.

Returns none

MainWindow.**get_value**(*parser*, *section*, *var*)
Get a value from the config file.

Parameters

- **parser** – the parser of the config file
- **section** – the section that contains the entry
- **var** – the variable that holds the value

Returns the retrieved value

MainWindow.**on_click**()
The main method that is called, when the start button is clicked. It connects to the camera, and calls methods for processing the camera image, for maintaining the mode, and for performing an action. In normal mode the actions are taken through mouse movements and mouse left clicks, while in 2-gestures mode keyboard events are accomplished.

Returns none

MainWindow.**set_config_parameters**()
Set parameters of the config file.

Returns none

MainWindow.**set_image**(*img*)
Set an image in the user interface that indicates the current status.

Parameters **img** – the image to set

Returns none

MainWindow.**set_image_in_main_window**(*frame*)
Set the processed camera image in the main window of the user interface to show a status picture while using the application. Print a text on the image frame that indicates the active mode.

Parameters **frame** – the processed image frame to set

Returns none

MainWindow.**set_style_sheet**(*widget*)
Set style sheet for a widget.

Parameters **widget** – the widget to be styled

Returns none

1.2 MoveMouse module

class MoveMouse.**MoveMouse**(*prev_data*, *data*)
Bases: object

Manage operations for performing mouse movements and mouse left clicks.

center_mouse()
Center the mouse position on the screen.

Returns none

detect_head_nod(*click_data*)
Detect whether a head nod is performed and do a left mouse click if it is the case. A head nod is detected when the weighted head movements in vertical position exceed the weighted movements in horizontal position with more than a threshold defined.

Parameters **click_data** – the frame data that is analyzed for nod detection

Returns none

lock_mouse_position()
Lock the active mouse position.

Returns none

move_mouse()
Move mouse based on the direction and speed retrieved from the data analyzed. Show a visual feedback as a popup window if the movement detected is in a certain range.

Returns none

open_popup()
Open a popup window as a click indicator

Returns none

save_mouse_position()
Save the active mouse position globally.

Returns none

set_data(*prev_data*, *data*)
Set data for the analysis of mouse movements and clicks.

Parameters

- **prev_data** – the data of previous frames
- **data** – the data of the active frame

Returns none

class MoveMouse.**MyPopup**
Bases: PyQt5.QtWidgets.QWidget

Define a popup window to show a visual feedback to the user.

MoveMouse.**get_value**(*parser*, *section*, *var*)
Get a value from the config file.

Parameters

- **parser** – the parser of the config file
- **section** – the section that contains the entry
- **var** – the variable that holds the value

Returns the retrieved value

MoveMouse.**set_config_parameters**()
Set config parameters for the mouse movements.

Returns none

1.3 NodShakeMode module

class NodShakeMode.**NodShakeMode**(*prev_data*, *data*)
Bases: object

A 2-gestures mode for performing key events through detecting head nods and head shakes.

apply()
The application method that detects head nods and shakes. If a nod or a shake is detected, a key event is performed.

Returns none

set_data(*prev_data*, *data*)
Set data for the analysis of head nods and head shakes.

Parameters

- **prev_data** – the data of previous frames
- **data** – the data of the active frame

Returns none

1.4 ProcessImage module

class ProcessImage.**ProcessImage**(*frame*)
Bases: object

Process the image frame, detect face and eyes, and retrieve the middle point for consequent mouse and key actions.

detect_face_and_eyes(*face_classifier*, *eye_classifier*)
Detect face and eyes in the image frame and return the data processed, as an image object. This method is outdated. For enhanced results, the method *detect_face_and_eyes_enhanced* is used instead.

Parameters

- **face_classifier** – the classifier for detecting the face
- **eye_classifier** – the classifier for detecting eyes

Returns the data of the processed image frame

detect_face_and_eyes_enhanced(*net*, *eye_classifier*)
An enhanced method for face and eye detection. Faces are detected through a pretrained network. Every eye is searched for separately in a vertical face half. The reference point is computed from both eye positions if available, or estimated if no two eyes are detected.

Parameters

- **net** – the network used for face detection
- **eye_classifier** – the classifier used for eye detection

Returns the data of the processed image frame

pre_processing()
Conduct a preprocessing on the image frame.

Returns the preprocessed image frame

class ProcessImage.**ProcessedImage**(*frame*, *x_middle*, *y_middle*)
Bases: object

The data of the processed image frame, which is calculated from face and eye positions.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

m

n

p

INDEX