Hello Face

Campus Access Control Management System

Group “Hello world”：
CHEN Yizhou, MAO Shuai, ZHANG Weixu, ZHANG Ke
22/11/2019

Gitlab link : https://gitlabcw2.centralesupelec.fr/2019cheny/facerecognitioncyzzk.git

CONTENT

Problem Statement

Have you ever been in this kind of trouble?

- Can’t remmember the password of your residence building?
- Forget to bring your room card with you?
- Lost your student card?

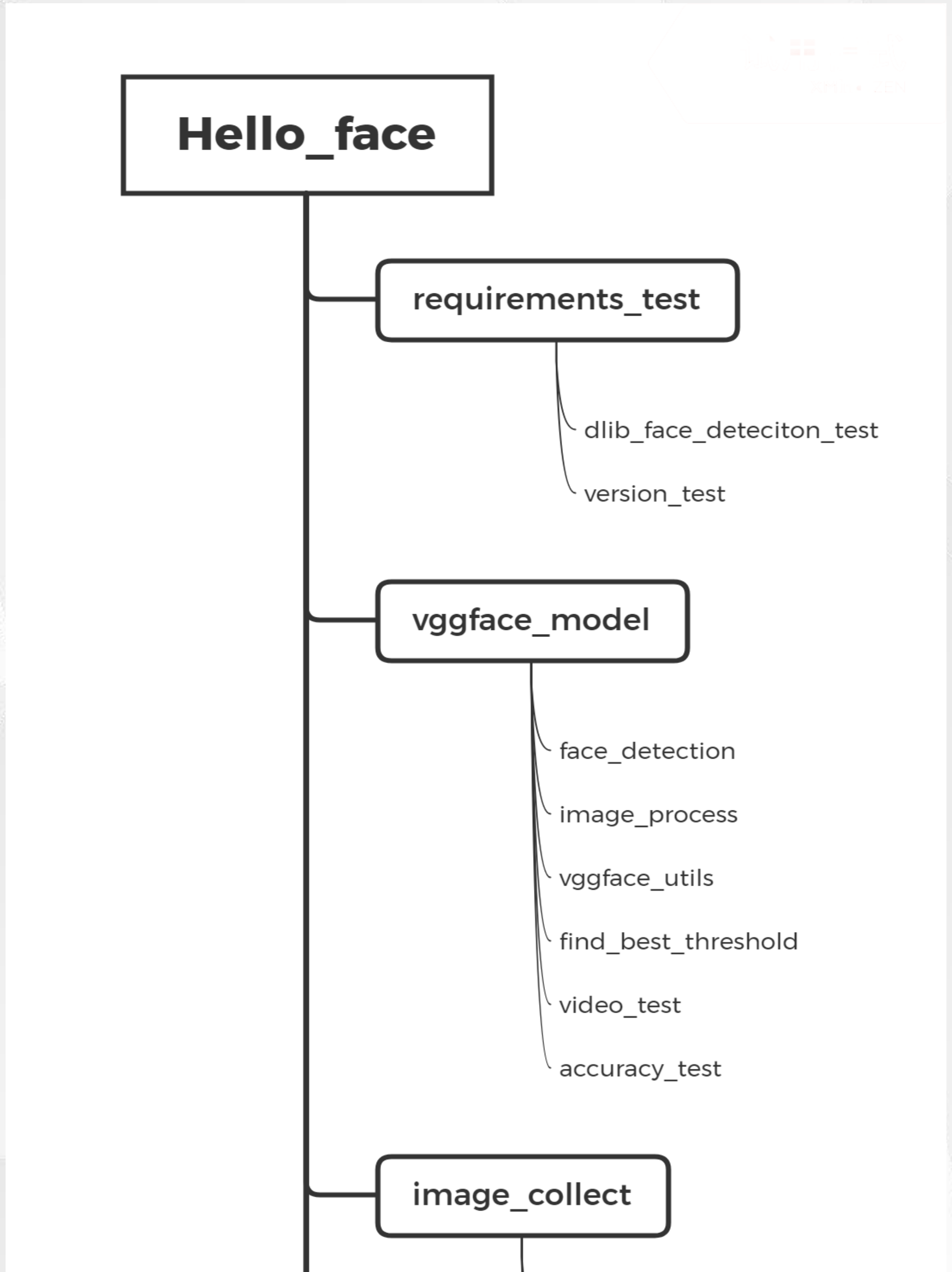
Inconvenient & Insecure

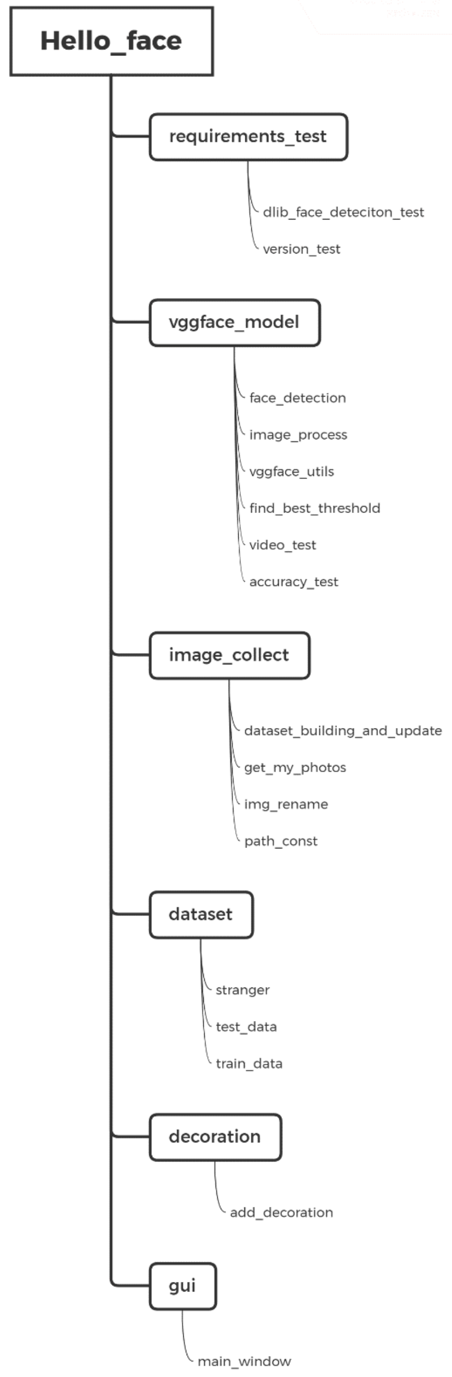
Our solution:

Campus access control management system based on facial recognition

Global description of the product

Flowchart


Hello_face
requirements_test
dlib_face_deteciton_test
version_test
vggface_model
face_detection
image_process
vggface_utils
find_best_threshold
video_test
accuracy_test
image_collect


Hello_face
requirements_test
dlib_face_deteciton_test
version_test
vggface_model
face_detection
image_process
vggface_utils
find_best_threshold
video_test
accuracy_test
image_collect
dataset_building_and_update
get_my_photos
img_rename
path_const
dataset
stranger
test_data
train_data
decoration
add_decoration
gui
main_window

Modules and functions

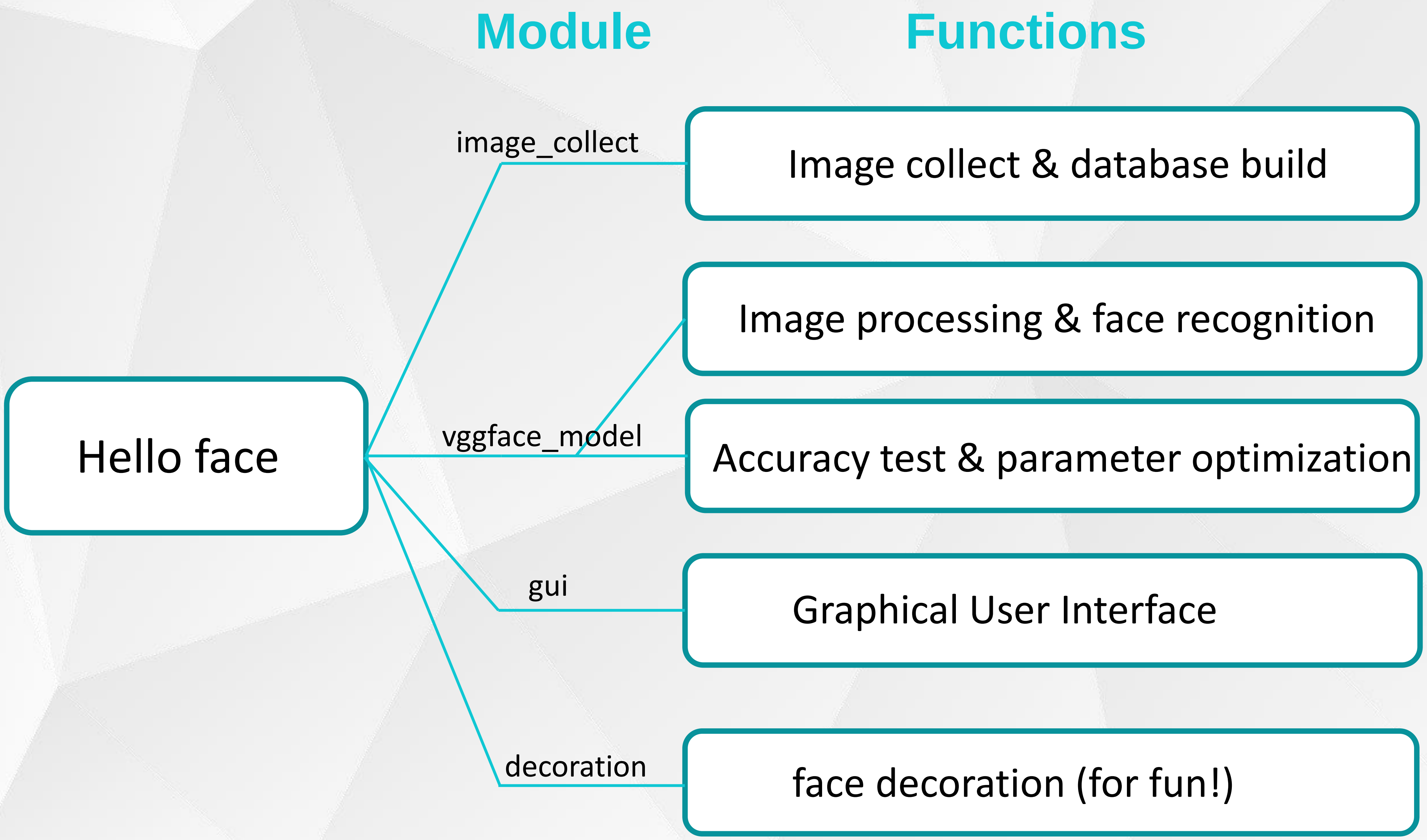

Module
Functions
Hello face
image_collect
Image collect & database build
vggface_model
Image processing & face recognition
Accuracy test & parameter optimization
gui
Graphical User Interface
decoration
face decoration (for fun!)

Image Collect & Database Build

Module: image_collect

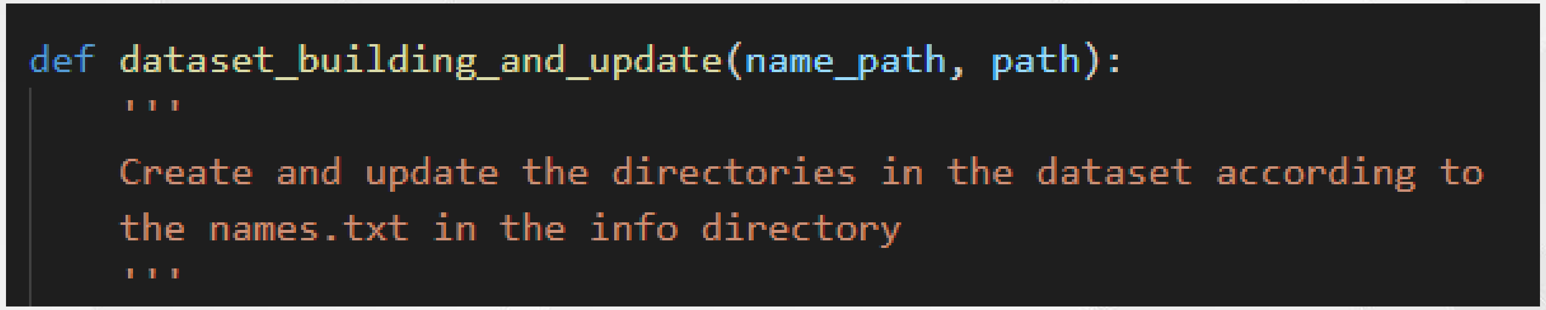
- In the campus access control management system, we should use photos of all the students in the residence building as the database

- We collected photos of 8 students(100-200 photos each) and strangers(300 photos) as our training data.

About the Code

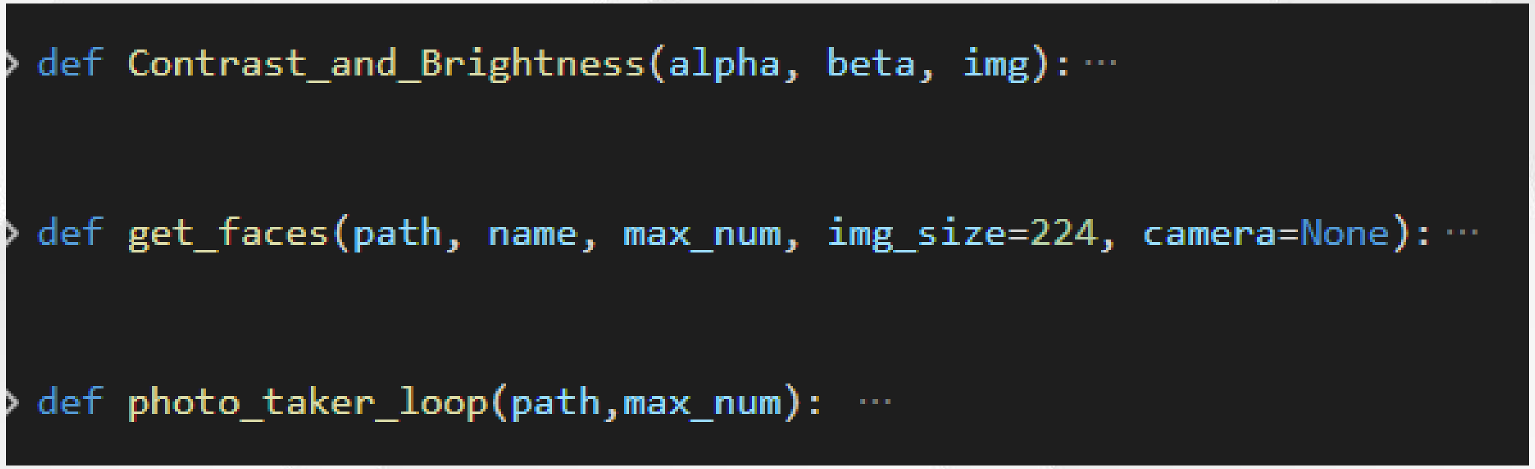
Module: image_collect

Database_building_and_update.py



```
def dataset_building_and_update(name_path, path):
    '''
    Create and update the directories in the dataset according to
    the names.txt in the info directory
    '''
```

Get_my_photos.py



```
def Contrast_and_Brightness(alpha, beta, img):···


def get_faces(path, name, max_num, img_size=224, camera=None):···


def photo_taker_loop(path,max_num):  ···
```

Image Processing & Face Recognition

Module: vggface_model

Step1: Calculate the face feature vector of each person in the database (use VGG face model)

Calculate the "mean features" by averaging all computed features for each image of one person.

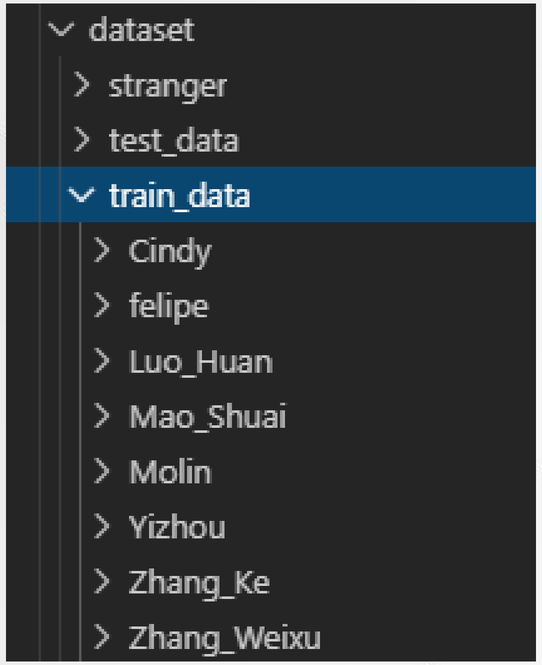

```
dataset
  stranger
  test_data
  train_data
    Cindy
    felipe
    Luo_Huan
    Mao_Shuai
    Molin
    Yizhou
    Zhang_Ke
    Zhang_Weixu
```

Image Processing & Face Recognition **Module: vggface_model**

- Vggface model can encode a face into a representation vector of 2048 numbers. We compute the Euclidean distance between two faces. If they are the same person, the distance value will be low, if they are from two different persons, the value will be high.
- During the face identification time, if the value is below a threshold, we would predict that those two pictures are the same person.

Image Processing & Face Recognition **Module: vggface_model**

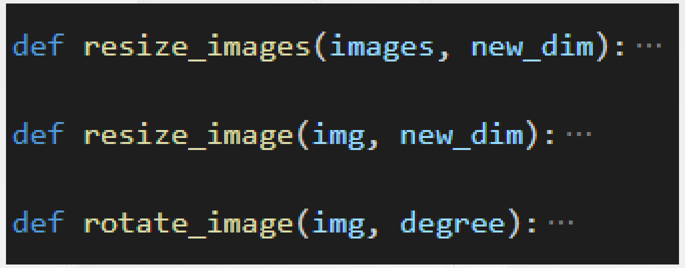
Step2: Face Detection and Recognition

- Compute the features from the face in webcam image and compare with each of our known faces' features, and find matching faces.
- Identify the person if find matching faces (use the threshold). Otherwise, the person is considered a stranger.

About the Code

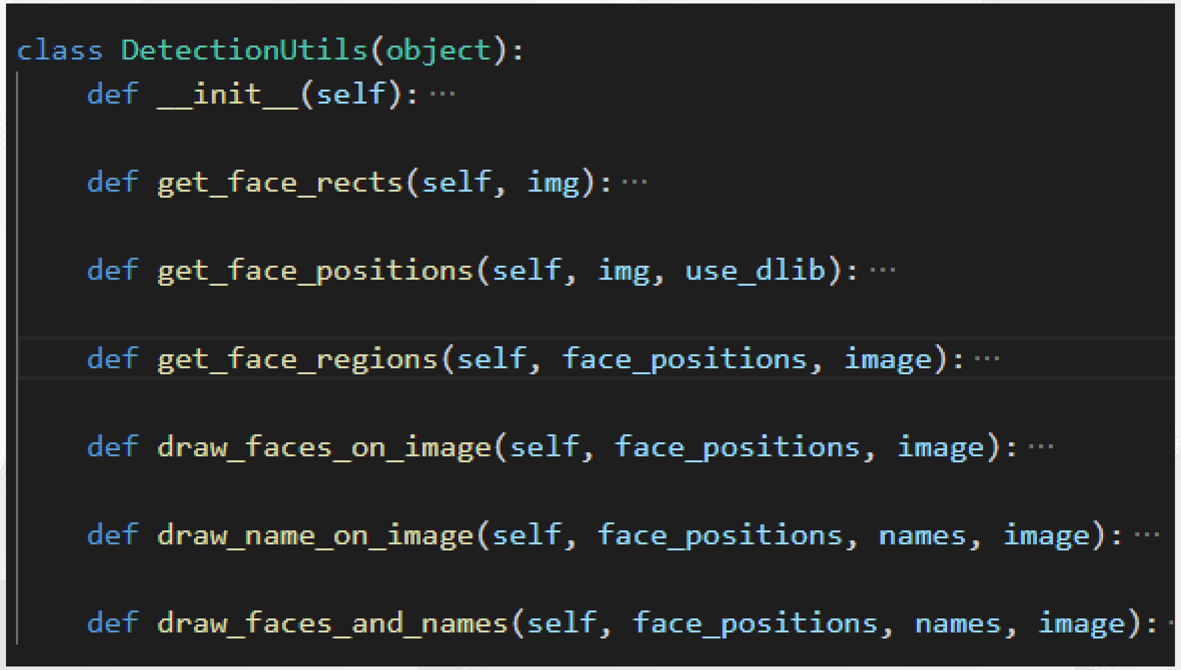
Module: vggface_model

Image_process.py



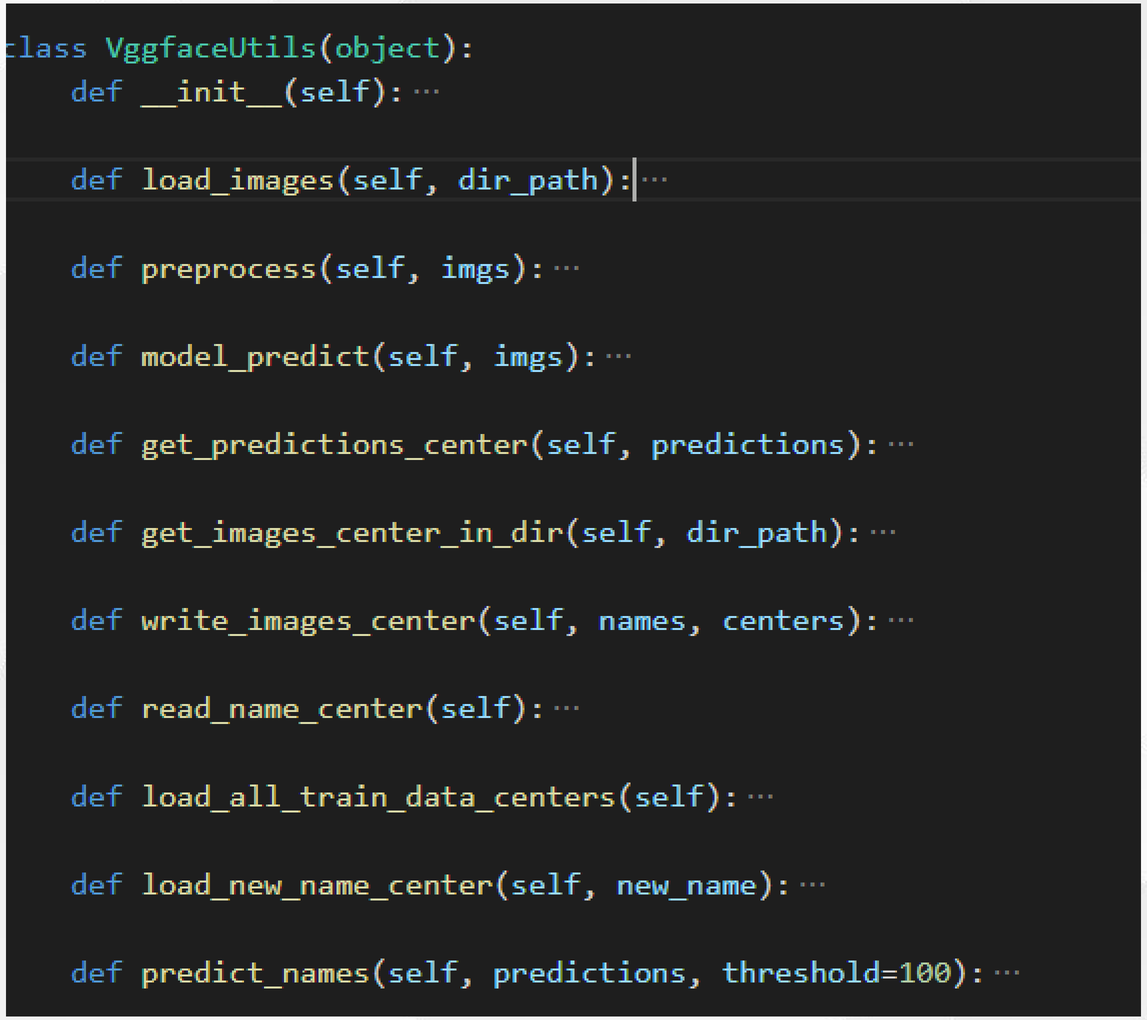
```python
def resize_images(images, new_dim):⋯

def resize_image(img, new_dim):⋯

def rotate_image(img, degree):⋯
```

Face_detection.py



```python
class DetectionUtils(object):
    def __init__(self):⋯

    def get_face_rects(self, img):⋯

    def get_face_positions(self, img, use_dlib):⋯

    def get_face_regions(self, face_positions, image):⋯

    def draw_faces_on_image(self, face_positions, image):⋯

    def draw_name_on_image(self, face_positions, names, image):⋯

    def draw_faces_and_names(self, face_positions, names, image):·
```

Vggface_utils.py



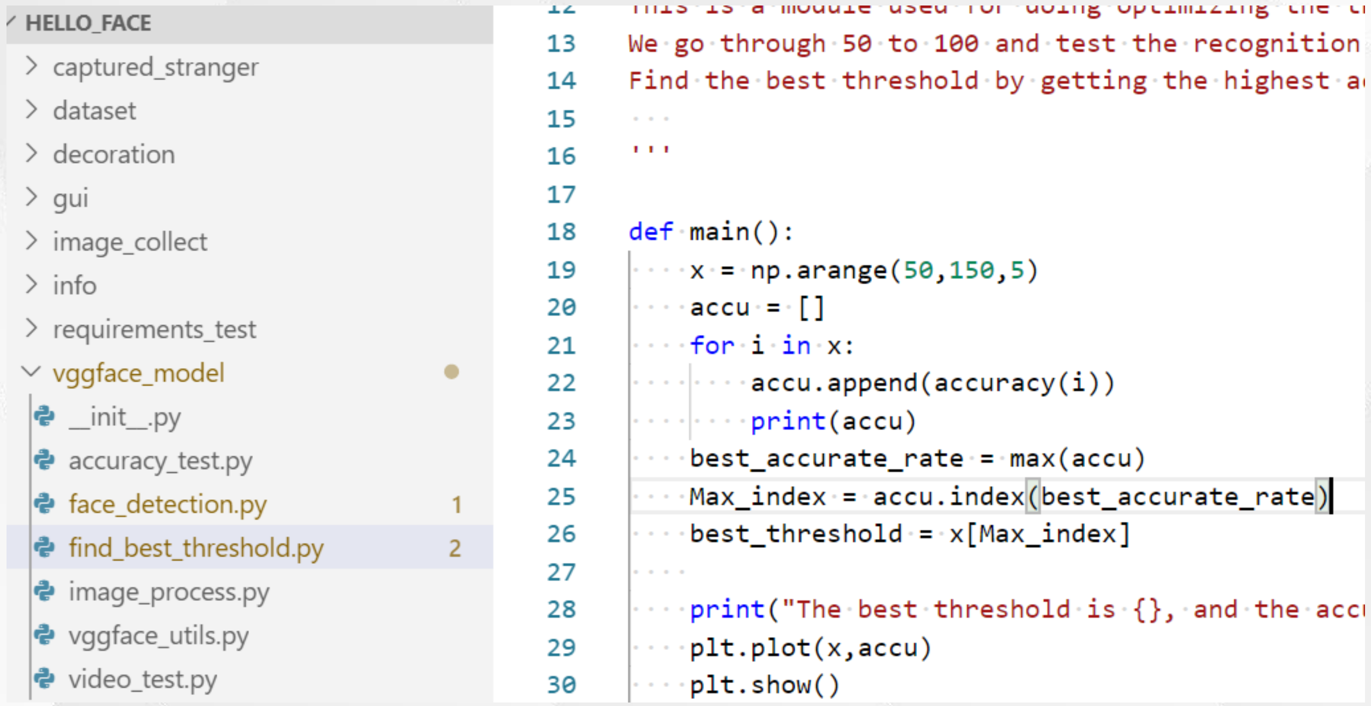
```python
class VggfaceUtils(object):
    def __init__(self):⋯

    def load_images(self, dir_path):⋯

    def preprocess(self, imgs):⋯

    def model_predict(self, imgs):⋯

    def get_predictions_center(self, predictions):⋯

    def get_images_center_in_dir(self, dir_path):⋯

    def write_images_center(self, names, centers):⋯

    def read_name_center(self):⋯

    def load_all_train_data_centers(self):⋯

    def load_new_name_center(self, new_name):⋯

    def predict_names(self, predictions, threshold=100):⋯
```

Accuracy test & parameter optimization

Module: vggface_model

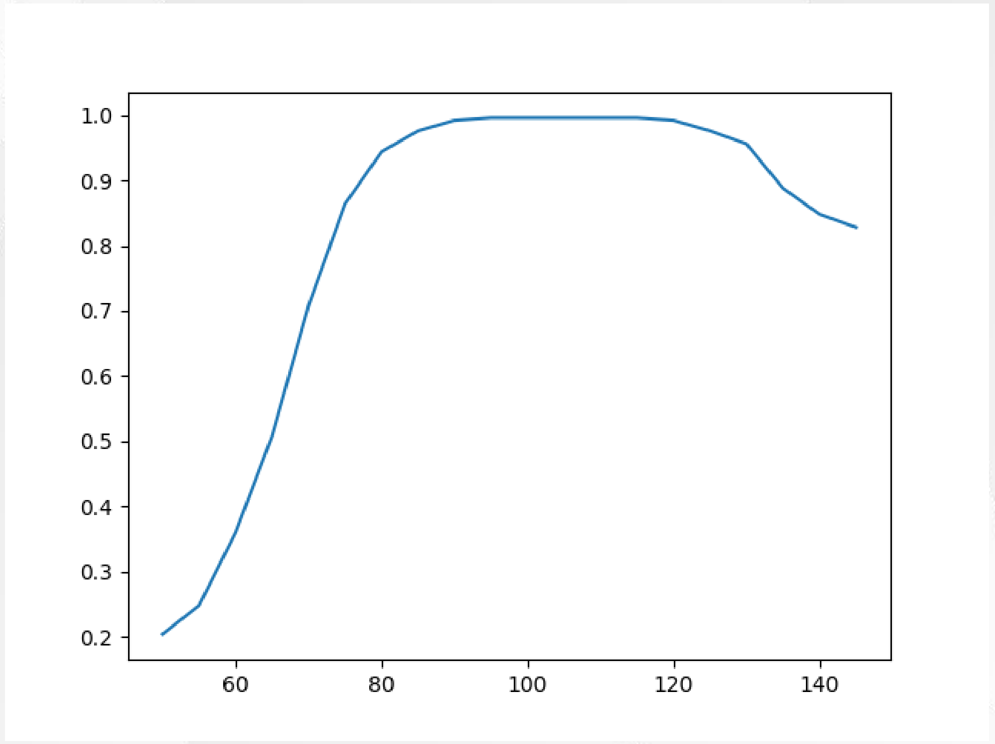
- Test the recognition accuracy with 500 pictures.
- Optimize the parameter (threshold) in the recognition function according to the accuracy.
- We finally got the recognition accuracy of 98.8% (This is mainly because the number of our known faces is quite small)

Accuracy test & parameter optimization



```
HELLO_FACE
> captured_stranger
> dataset
> decoration
> gui
> image_collect
> info
> requirements_test
v vggface_model
    __init__.py
    accuracy_test.py
    face_detection.py          1
    find_best_threshold.py     2
    image_process.py
    vggface_utils.py
    video_test.py
```

```python
This is a module used for doing optimizing the t
We go through 50 to 100 and test the recognition
Find the best threshold by getting the highest a
...
'''

def main():
    x = np.arange(50,150,5)
    accu = []
    for i in x:
        accu.append(accuracy(i))
        print(accu)
    best_accurate_rate = max(accu)
    Max_index = accu.index(best_accurate_rate)
    best_threshold = x[Max_index]

    print("The best threshold is {}, and the accu
    plt.plot(x,accu)
    plt.show()
```



We finally got the recognition accuracy of 98.8% (This is mainly because the number of our known faces is quite small)

Graphical User Interface

Module: gui

- Realize Real-time video stream acquisition, face detection and recognition

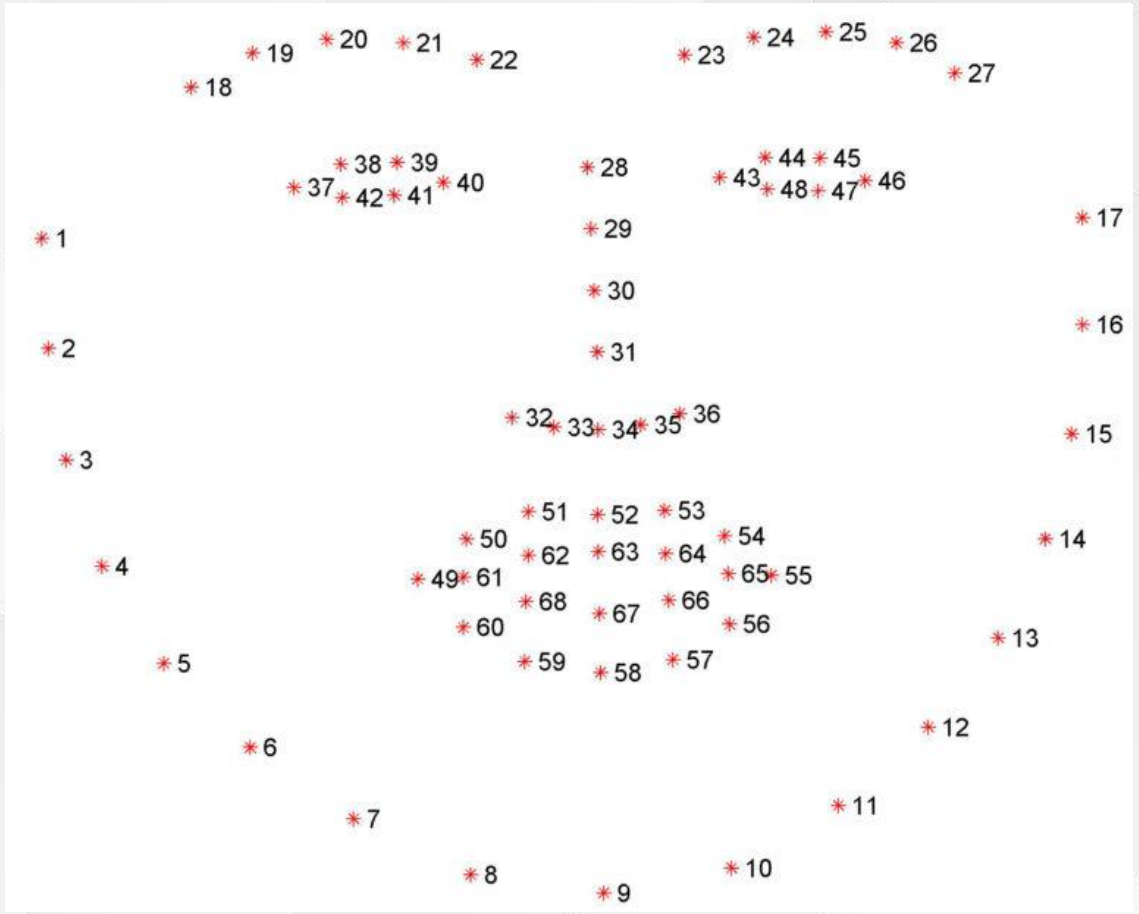
- Build the management interface of the database (for example, the collection and processing of a new student's face).

Face Decoration **Module: decoration**

- We created three interesting functions to decorate the face: add a beard, add a pair of sunglasses, and add a hat.
- Based on the landmarks on the detected face and 68 characteristic points. (use the model shape_predictor_68_face_landmarks.dat)

About the Code

```
p = SHAPE_PREDICTOR_FILE
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(p)

cap = cv2.VideoCapture(0)

while True:
    # Getting out image by webcam
    _, image = cap.read()
    # Converting the image to gray scale
    image = cv2.flip(image, 1)
    gray = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
    # Get faces into webcam's image
    rects = detector(gray, 0)
    # For each detected face, find the landmark.
    for (i, rect) in enumerate(rects):
        # Make the prediction and transfom it to numpy array
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)
```



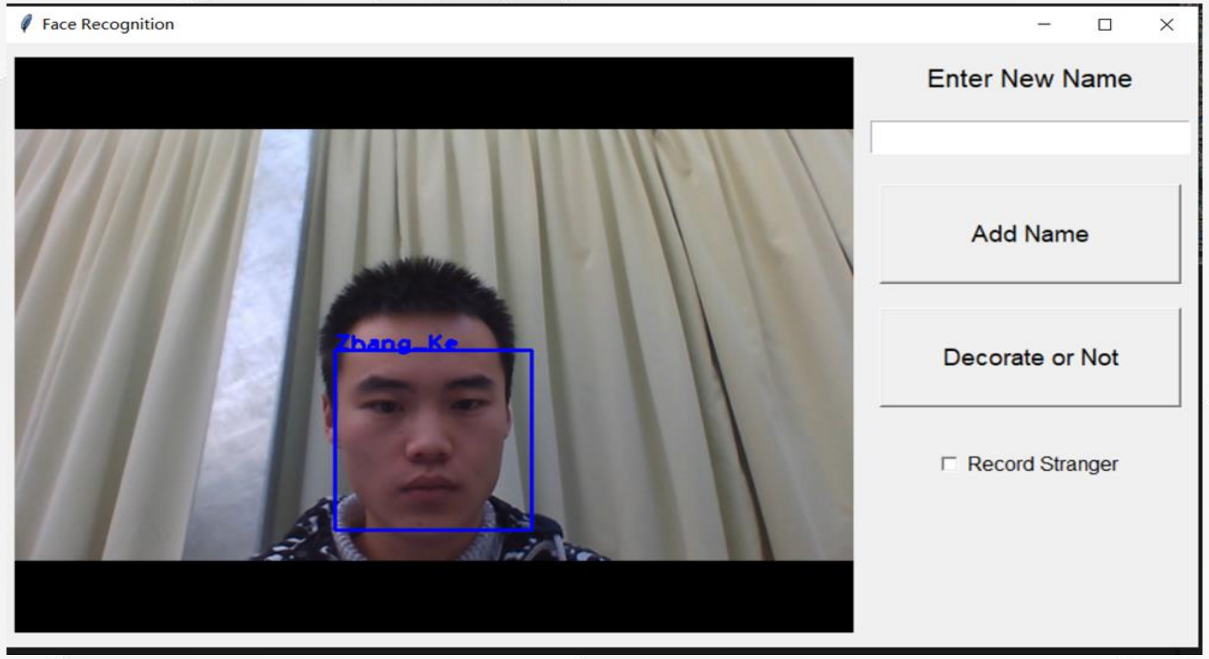
IdxRange jaw --> [0,16]
IdxRange rightBrow --> [17,21]
IdxRange leftBrow --> [22,26]
IdxRange nose --> [27,35]
IdxRange rightEye --> [36,41]
IdxRange leftEye --> [42,47]
IdxRange mouth --> [48,59]
IdxRange mouth_inside --> [60,67]

About the Code

```
def add_hat(origin_hat,image,shape):
    brows = np.array(shape[17:27])
    jaw = np.array(shape[0:17])
    down_brow= np.min(brows[:,1])
    left = np.min(jaw[:,0])
    right = np.max(jaw[:,0])
    up = np.min(jaw[:,1])
    down = np.max(jaw[:,1])
    dis = down-down_brow+10
    hat = cv2.resize(origin_hat, (int(1.2*(right-left)),down-up))
    left_move = (right - left)//10
    right_move = int(0.2*(right-left))-left_move
    if (up-dis)>0 and (left-left_move)>0 and (right+right_move)<image.shape[1]:
        #mask is a bool ndarray to eliminate the grounding
        mask = hat<150
        region_hat = image[up-dis:down-dis, left-left_move:right+right_move, :]
        region_hat[mask] = hat[mask]
```

Graphical User Interface

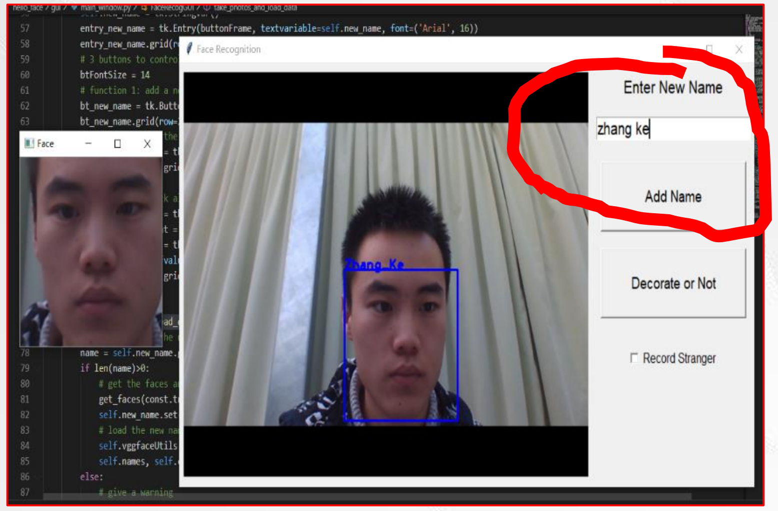
- The whole window is divided into two parts.
- One is used for show the image. The other is used for placing the function button.



About the Code

- About the code: mainly consisted of three functions

```
def take_photos_and_load_data(self):
        # take photos of the new person after clicking the Add Name button
        name = self.new_name.get()
        if len(name)>0:
            # get the faces and show them out in another small window
            get_faces(const.train_data_path, name, max_num=100, camera=self.ca
mera)
            self.new_name.set('')
            # load the new name and its vector to the csv file
            self.vggfaceUtils.load_new_name_center(name)
            self.names, self.centers = self.vggfaceUtils.read_name_center()
        else:
            # give a warning
            messagebox.showwarning(title="Invalid", message="An Empty Name!")
```



```
entry_new_name = tk.Entry(buttonFrame, textvariable=self.new_name, font=('Arial', 16))
entry_new_name.grid(r
# 3 buttons to contro
btFontSize = 14
# function 1: add a n
bt_new_name = tk.Butt
bt_new_name.grid(row=
name = self.new_name.
if len(name)>0:
    # get the faces a
    get_faces(const.t
    self.new_name.set
    # load the new na
    self.vggfaceUtils
    self.names, self.
else:
    # give a warning
```

Enter New Name: zhang ke

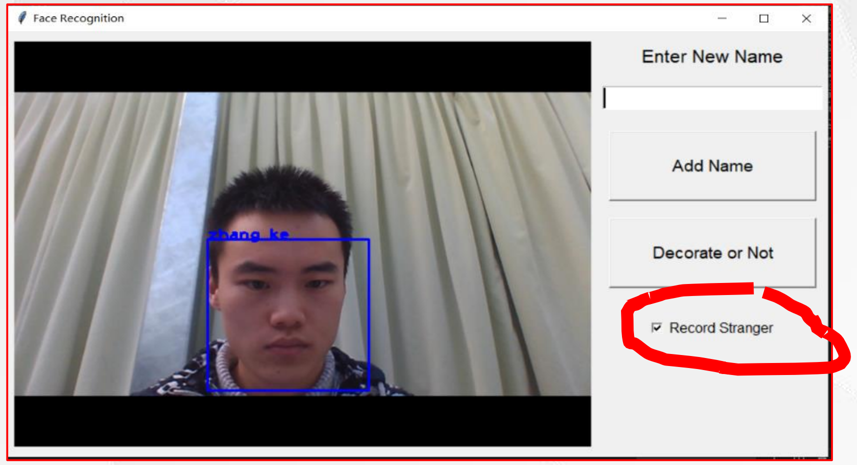
hello_face / gui / main_window.py / FaceRecogGUI / take_photos_and_load_data
Face Recognition
Face
Zhang_Ke
Add Name
Decorate or Not
Record Stranger

About the Code

- About the code: mainly consisted of three functions

- def update(self): # just extract the central part of the function

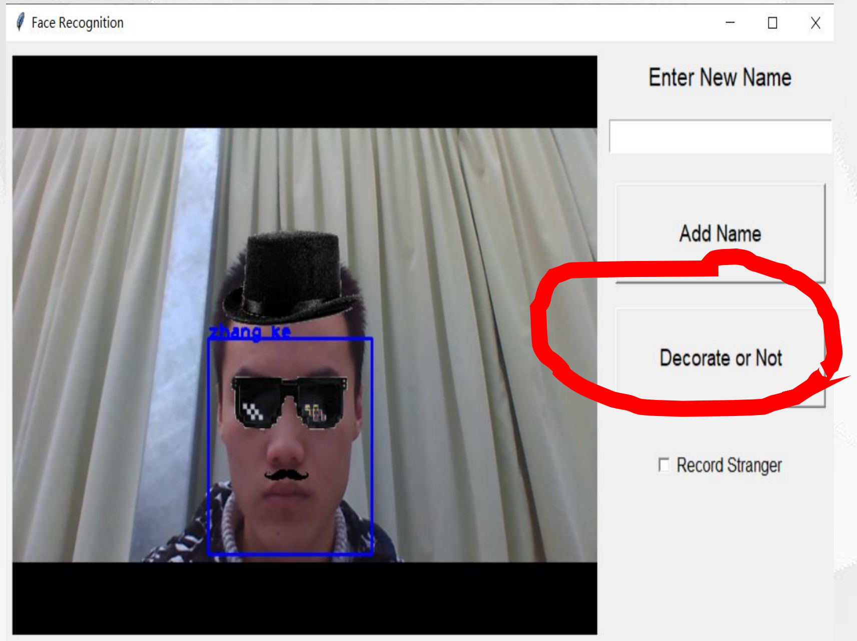
```
if len(face_positions)>0:
      faces = self.detectionUtils.get_face_regions(face_positions, img) # get the face image
      predictions = self.vggfaceUtils.model_predict(faces) # get the vectors of the image
      pred_names = self.vggfaceUtils.predict_names(predictions) # get the name of people
      self.detectionUtils.draw_faces_and_names(face_positions, pred_names, img)
      # whether to detect and save the image
      if self.intvar_check.get() == 1:
        # judege the stranger
        if  self.vggfaceUtils.stranger_label in pred_names:
          self.stranger_count += 1
          if self.stranger_count > 5:
            # modifier the format while saving the image
            localtime = time.asctime(time.localtime(time.time()))
            file_name = 'stranger'+'-'+str(localtime).replace(' ', '_').replace(":", "-")+'.jpg'
            save_file_path = os.path.join(const.STRANGER_SAVE_PATH, file_name)
            cv2.imwrite(save_file_path, img) # save the image
            self.stranger_count = 0
        else:
          self.stranger_count = 0
```



About the Code

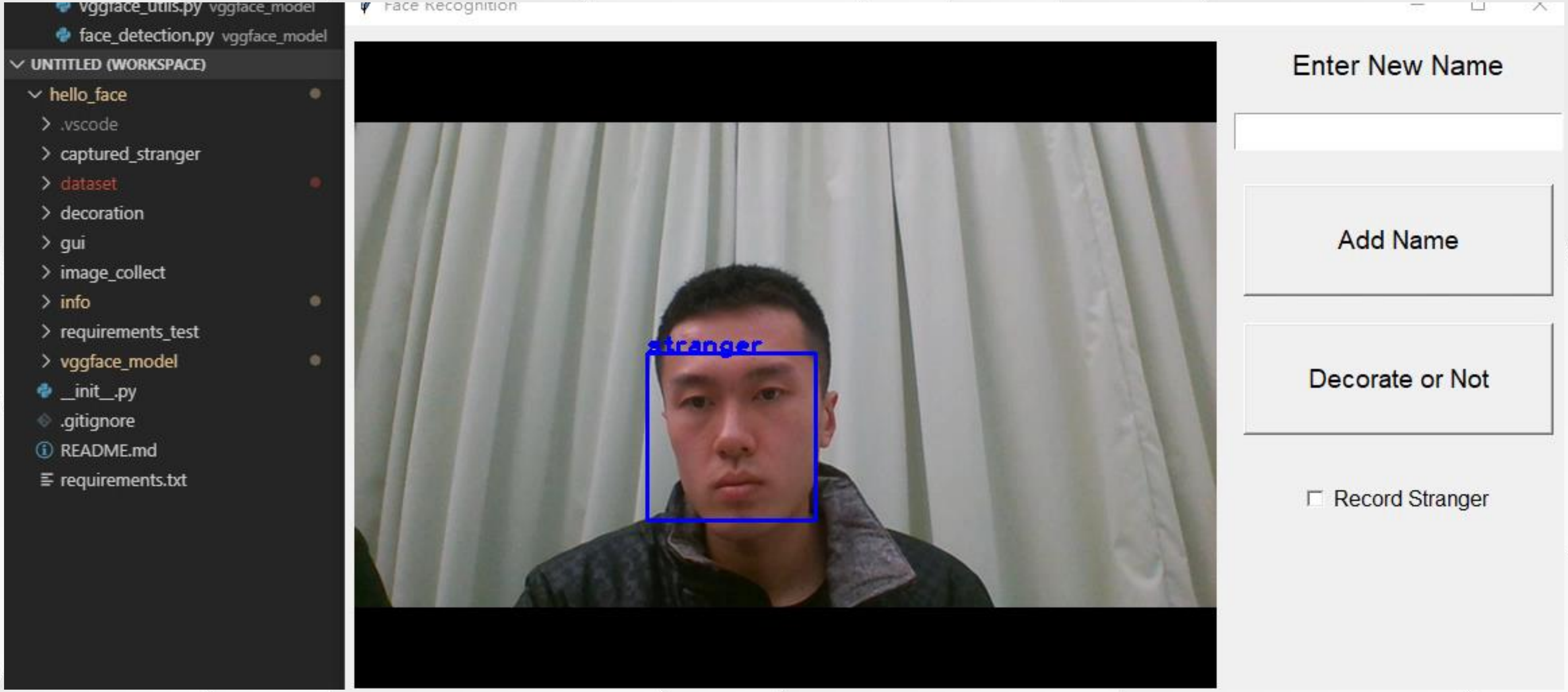
- About the code: mainly consisted of three functions

```
def add_decoration(self, face_rect, image):
    # Make the prediction and transfom it to numpy array
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    shape = self.shape_predictor(gray, face_rect) # get 68 points of the face
    shape = face_utils.shape_to_np(shape)
    # Draw on our image, all the finded cordinate points (x,y)
    add_beard(self.origin_beard,image,shape)
    add_hat(self.origin_hat,image,shape)
    add_glasses(self.origin_glasses,image,shape)

  def add_or_remove_decoration(self):
    self.flag_decor = not self.flag_decor
```



04

Demonstration


vggface_utils.py vggface_model
face_detection.py vggface_model
UNTITLED (WORKSPACE)
hello_face
.vscode
captured_stranger
dataset
decoration
gui
image_collect
info
requirements_test
vggface_model
__init__.py
.gitignore
README.md
requirements.txt
Face Recognition
stranger
Enter New Name
Add Name
Decorate or Not
Record Stranger

Our Team

CHEN Yizhou

Image collection
Vggface_model

MAO Shuai

Image collection
Dataset build
Face decoration

ZHANG Weixu

Dataset build
Vggface_model

ZHANG Ke

GUI

Reference

[1]https://gitlabcw2.centralesupelec.fr/codingweeksstaff/cs_codingweek_facerecognition/blob/master/S3_facedescription.md
[2] https://github.com/seathiefwang/FaceRecognition-tensorflow

Thanks for listening!

Question time