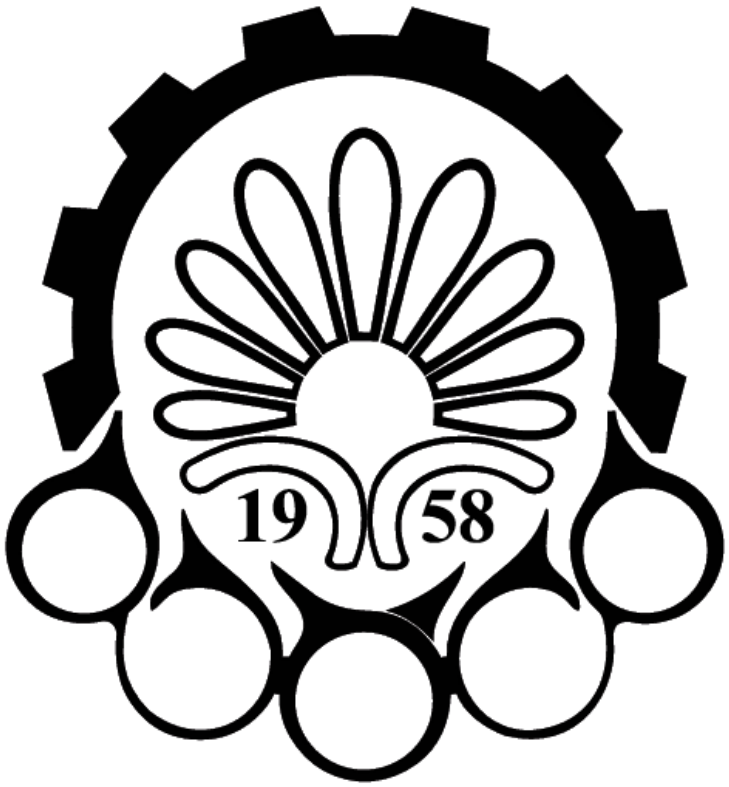


Amirkabir University of Technology
(Tehran Polytechnic)

Advanced Programming Final Project Report

Dr. Jahanshahi

Mohammad Javad Ranjbar 9523048

Project:

Build a Snick game with voice control

game making:

To make a game snake From the library pygame We used this library to create a graphic space for us.

Graphic space design:

We work for the size and color of the screen and the location of various objects've identified and then in a loop da ƒ The screen will update based on user movements

Game algorithm:

The game is such that the apple is placed randomly inside the screen and the player must move towards the apple, and if the player and the apple are the same, a point is recorded for the player and the apple is changed, and if the player is on his own body. Or eat the surrounding walls the player loses

code:

As mentioned, we first set the required values

```
display_width = 500
display_height = 500
green = (0, 255, 0)
blue = (24, 123, 205)
black = (0, 0, 0)
purple=(170,6,255)
window_color = (0, 171, 102)
apple_image = pygame.image.load('apple.png')
clock = pygame.time.Clock()
keyword_index=-1
snake_head = [250, 250]
snake_position = [[250, 250], [240, 250], [230, 250]]
apple_position = [random.randrange(1, 50) * 10, random.randrange(1, 50) * 10]
score = 0
pygame.init()  # initialize pygame modules
```

Then place the initial value of the player, place the apple and score points to the function play_game we give

Function play_game :

```
def play_game(snake_head, snake_position, apple_position, button_direction, apple, score):
    crashed = False
    prev_button_direction = 1
    button_direction = 1
    current_direction_vector = np.array(snake_position[0]) - np.array(snake_position[1])
    Player_Avatar = pygame.image.load("Snake_Head.png")
    # when a key is press an event will happen and if the key was the right key snake will

    while crashed is not True:···
    return score
```

This function updates the page in an infinite loop to continue the game. When the game is closed or crashed, it is removed from this loop and returns the player score.

```
for event in pygame.event.get():

    if event.type == pygame.QUIT:
        crashed = True
    if event.type == pygame.KEYDOWN:
        if event.key == pygame.K_LEFT and prev_button_direction != 1:···


        elif event.key == pygame.K_RIGHT and prev_button_direction != 0:···


        elif event.key == pygame.K_UP and prev_button_direction != 2:···


        elif event.key == pygame.K_DOWN and prev_button_direction != 3:···
        else:
            button_direction = button_direction
```

In this loop in addition to the update screen event in endless games of Czech and if that event in mind we took (pressing the button on the top left bottom right) in accordance with the direction of the snake action is performed and the event out Mtghyyrkrsh It is torn and comes out of the loop

```
    display.fill(window_color)
    display_apple(display, apple_position, apple)
    display_snake(snake_position)
    Player(Player_Avatar,snake_head[0],snake_head[1])
    snake_position, apple_position, score = generate_snake(snake_head, snake_position, apple_position,···
    pygame.display.set_caption("Snake Game" + "  " + "SCORE: " + str(score))
    pygame.display.update()
    prev_button_direction = button_direction
    if is_direction_blocked(snake_position, current_direction_vector) == 1:···

    clock.tick(4)
```

At the bottom of the loop, we update the screen and if the snake encounters obstacles, we will leave the game.

Other functions :

```
def display_apple(display, apple_position, apple):
    display.blit(apple, (apple_position[0], apple_position[1]))
```

```
> def Player(Player_Avatar , PlayerX , PlayerY ):···
```

Display the apple and the player according to the location of the apple and the player

```
def display_snake(snake_position):
    #to display boudneries and snake
    pygame.draw.line(display, purple, (0,0),(0,500) , 5 )
    pygame.draw.line(display, purple, (0, 0), (500, 0), 5)
    pygame.draw.line(display, purple, (500, 500), (0, 500), 5)
    pygame.draw.line(display, purple, (500, 500), (500, 0), 5)
    for position in snake_position:
        pygame.draw.rect(display, blue, pygame.Rect(position[0], position[1], 10, 10))
```

First we draw four lines to create the margins of the page

And we also used a number of squares to draw a snake

```
> def collision_with_boundaries(snake_head):···


> def collision_with_self(snake_position):···
```

In case of collision with itself or the margin, it returns the value of one until the end of the game

```
> def collision_with_apple(apple_position, score):···
```

Increases player points

Voice control:

To do this we have libraries pyaudio And pyautogui And pvprocupine We got help

Algorithm:

Library pvprocupine It allows us to be able towake word What we want in this project) go down, go left, go right, snake up (Train your desired model so that the microphone receives the desired function by receiving one of these words

code:

First we have to make the most of our desired models and then use this library if we receive any of these wiki words with the library.pyautogui Press the desired button

We give the location of the trained models

```
# below are the four wake word's path that you have generated earlier
key1 = r'Porcupine\resources\keyword_files\windows\snake_up_windows.ppn'
key2 = r'Porcupine\resources\keyword_files\windows\go_down_windows.ppn'
key3 = r'Porcupine\resources\keyword_files\windows\go_right_windows.ppn'
key4 = r'Porcupine\resources\keyword_files\windows\go_left_windows.ppn'
# this is the library path that you can fnd inside Porcupine -> lib -> system(windows or lin
library_path = r'Porcupine\lib\windows\amd64\libpv_porcupine.dll'

# this is model file path can be find inside Porcupine -> lib -> common
model_file_path = r'Porcupine\lib\common\porcupine_params.pv'
```

```
model_file_path = r'Porcupine\lib\common\porcupine_params.pv'
keyword_file_paths = [key1, key2, key3, key4]
sensitivities = [1,0.5,0.5,0.5]
handle = pvporcupine.Porcupine(library_path, model_file_path, keyword_file_paths=keyword_file_paths, sensitivi

def get_next_audio_frame():
```

Then with the function get_next_audio_frame We receive the sound

```
while True:
    pcm = get_next_audio_frame()
    keyword_index = handle.process(pcm)
    if keyword_index==0:
        pyautogui.press('up')
    if keyword_index==3:
        pyautogui.press('left')
    if keyword_index==2:
        pyautogui.press('right')
    if keyword_index==1:
        pyautogui.press('down')
```

And if you are one of key word The desired button is selected

Control with motion detection :

To do this from the library opencvWe have received help that allows us to work with the image more easily

Algorithm:

First we select an object like a pen to consider the movement of that object, then first we remove the rest of the colors from the image and then we follow the object. If more than 15 frames move in one direction, change that direction as the direction change. We catch snakes

code:

At first, we used the crack to get the desired color, but after obtaining it, we commented on it for convenience and left the color straight.

```
cv2.namedWindow("Tracking")
#making trackbar (horizontal slider)
cv2.createTrackbar('LH',"Tracking",0,360,nothing)
cv2.createTrackbar('LS',"Tracking",0,360,nothing)
cv2.createTrackbar('LV',"Tracking",0,360,nothing)
cv2.createTrackbar('UH',"Tracking",255,255,nothing)
cv2.createTrackbar('US',"Tracking",255,255,nothing)
cv2.createTrackbar('UV',"Tracking",255,255,nothing)
```

Now we receive the image from the webcam in two frames, then do it permanently in an infinite loop until the button is used. q Press do

```
cv2.namedWindow("Image")
cap=cv2.VideoCapture(0)
ret,frame1 = cap.read()
ret,frame2 = cap.read()
```

To reduce noise from gaussianbulr we have used

Now first of bgr_hsv We convert

The new photo with the color range to Mybaym and masks Field' s

```
imghsv1=cv2.cvtColor(frame1,cv2.COLOR_BGR2HSV)
imghsv2=cv2.cvtColor(frame2,cv2.COLOR_BGR2HSV)
'''
lb=cv2.getTrackbarPos('LH',"Tracking")
lg=cv2.getTrackbarPos('LS',"Tracking")
lr=cv2.getTrackbarPos('LV',"Tracking")

ub=cv2.getTrackbarPos('UH',"Tracking")
ug=cv2.getTrackbarPos('US',"Tracking")
ur=cv2.getTrackbarPos('UV',"Tracking")
l_b=np.array([lb,lg,lr])
u_b=np.array([ub,ug,ur])
'''
l_b=np.array([72,127,0])
u_b=np.array([109,255,255])

mask1=cv2.inRange(imghsv1,l_b,u_b)
mask2=cv2.inRange(imghsv2,l_b,u_b)
```

And with BitWise operation, we separate only the desired part from the original photos, then we find the difference between the corners of the object and the amount of change of the two photos.

```
diff=cv2.absdiff(res1,res2)
Gray=cv2.cvtColor(diff,cv2.COLOR_BGR2GRAY)
blured=cv2.GaussianBlur(Gray,(5,5),0)
ret,thresh=cv2.threshold(blured,20,255,cv2.THRESH_BINARY)
dilated=cv2.dilate(thresh,None,iterations=5)
contours , hierarchy=cv2.findContours(dilated,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
x1=0
y1=0
```

If the area of the object is too large ,draw a circle around it and keep its center .Then, if the changes are more than 15 frames, we perform the operation of finding the direction of movement) by finding the difference between the centers.(

```
if counter >= 15 and i == 1 and pts[-15] is not None:
    #Calculate the distance between the current frame and 10th frame before
    dX = pts[-15][0] - pts[i][0]
    dY = pts[-15][1] - pts[i][1]
    (dirX, dirY) = ('', '')
    #If distance is greater than 100 pixels, considerable direction change has occured.
    if np.abs(dX) > 100:
        if (np.sign(dX) == 1):
            dirX = 'Right'
            pyautogui.press('right')
        else :
            dirX ='Left'
            pyautogui.press('left')

    if np.abs(dY) > 100:
        if (np.sign(dY) == 1):
            dirY = 'Up'
            pyautogui.press('up')
        else:
```

Continue if I read the next frame and if the user q Press to exit the program

```
    cv2.imshow("Image",frame1)
    cv2.imshow("Image2",res1)
    cv2.imshow("Image3",mask1)
    frame1=frame2
    _,frame2 = cap.read()
    frame2 = cv2.GaussianBlur(frame2, (5,5), 0)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Game control using neural network:

For this purpose, I had to train the neural network with my own data, and then we controlled the game like a motion control in a loop ,and the network chose the direction of the hand for the snake) it is not possible for you to train, it did not put the training data in this file Only code is available(

Code:

First we open the data which are four classes up and down right and left

```
train_datagen = ImageDataGenerator(
        rescale=1./255,
)
X_train=train_datagen.flow_from_directory('X',
                                          target_size = (100, 100),
                                          batch_size = 20,
                                          class_mode = 'categorical')
```

Then we create a convolutional multilayer neural network and make the network more efficient

```
def build_model():
    model = Sequential()
    # add Convolutional layers
    model.add(Conv2D(filters=32, kernel_size=(3,3), activation='relu', padding='same',
                     input_shape=(100, 100, 3)))
    model.add(MaxPooling2D(pool_size=(2,2)))
    model.add(Conv2D(filters=64, kernel_size=(3,3), activation='relu', padding='same'))
    model.add(MaxPooling2D(pool_size=(2,2)))
    model.add(Conv2D(filters=64, kernel_size=(3,3), activation='relu', padding='same'))
    model.add(MaxPooling2D(pool_size=(2,2)))
    model.add(Flatten())
    # Densely connected layers
    model.add(Dense(128, activation='relu'))
    # output layer
    model.add(Dense(4, activation='softmax'))
    # compile with adam optimizer & categorical_crossentropy loss function
    model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
    return model

model = build_model()
filepath='weights0.{epoch:02d}-{accuracy:.2f}.hdf5'
CB=keras.callbacks.ModelCheckpoint(filepath, monitor='accuracy', verbose=0, save_best_only=True, save_weights_only=False, mode='auto', period=1)
history =model.fit(X_train,epochs=3, callbacks=[CB])
```

The number of data was small and the network was quickly overfit, so we set the number of IPACs to three) there was no time to increase the data and this model does not have high accuracy in test data(

Now in the CNN code, we load the network and receive the image data from the webcam so that the network can tell the direction.

Registration and leaderboard:

To do this from pyqt We have helped to provide a place to register and record players' points

How to build:

First we create a class as the main page and we create the rest of the pages as a class. When we want to enter other pages, we call the object of each class

To register the file, the information is stored in an Excel file, and to log it, we check that the information matches it. The same process occurs for the points registration process.

When we start the game, three crunches are run, each of which is related to the game, voice control and image control.

code:

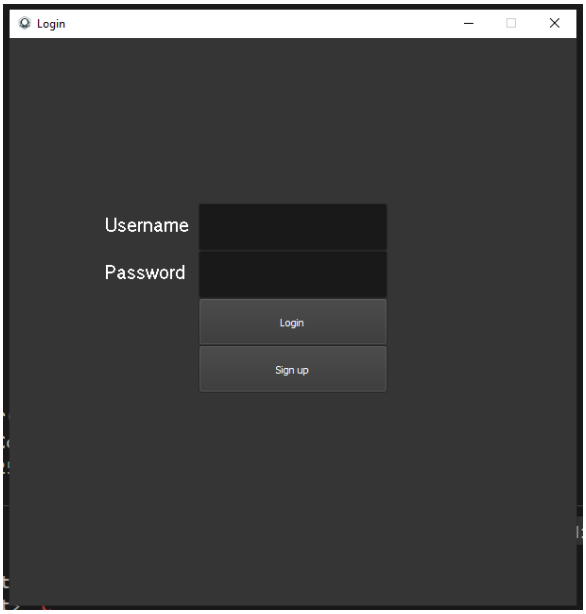
Set the desired theme to open this application

```
app=QApplication (sys.argv)
app.setStyle("Fusion")
palette = QPalette()
palette.setColor(QPalette.Window, QColor(53, 53, 53))
palette.setColor(QPalette.WindowText, QColor(255, 255, 255))
palette.setColor(QPalette.Base, QColor(25, 25, 25))
palette.setColor(QPalette.AlternateBase, QColor(53, 53, 53))
palette.setColor(QPalette.ToolTipBase, QColor(255, 255, 255))
palette.setColor(QPalette.ToolTipText, QColor(255, 255, 255))
palette.setColor(QPalette.Text, QColor(255, 255, 255))
palette.setColor(QPalette.Button, QColor(53, 53, 53))
palette.setColor(QPalette.ButtonText, QColor(255, 255, 255))
palette.setColor(QPalette.BrightText, QColor(255, 0, 0))
palette.setColor(QPalette.Link, QColor(42, 130, 218))
palette.setColor(QPalette.Highlight, QColor(42, 130, 218))
palette.setColor(QPalette.HighlightedText, QColor(0, 0, 0))
app.setPalette(palette)
```

classes:

```
class MainWindow(QMainWindow):
    def __init__(self, parent=None):
        super(MainWindow, self).__init__(parent)
        #self.setGeometry(50, 50, 600, 600)
        self.setFixedSize(600, 600)
        self.startUIToolTab()
        self.setWindowIcon(QIcon("v.png"))
```

mainwindowTo control the face is all the pages and we have specified the size and logo of all the pages and the pageloginCall ,which is as follows



On this page, we have two buttons, three labels and two edit lines, and the user is directed to a place by pressing each of these buttons.

```
def startUIToolTab(self):
    self.ToolTab = UIToolTab(self)
    self.setWindowTitle("Login")
    self.setCentralWidget(self.ToolTab)
    self.ToolTab.Login_Button.clicked.connect(self.Check_userpass)
    self.ToolTab.Signup_Button.clicked.connect(self.StartSingUpWindow)
    self.show()
```

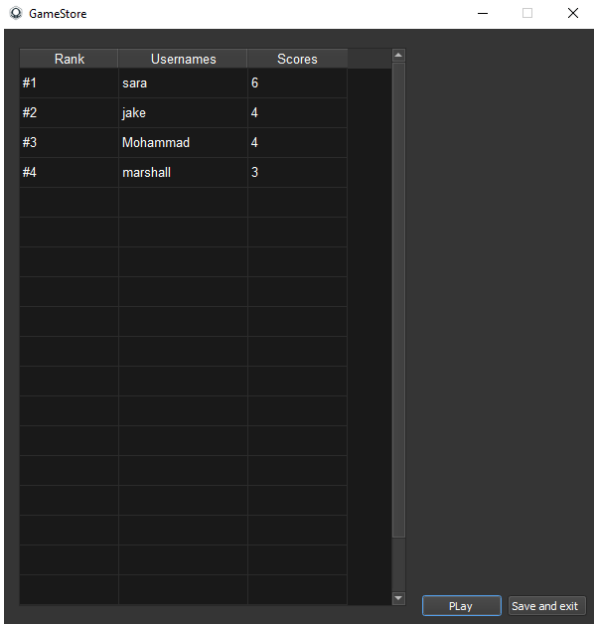
```
def Check_userpass(self):...
```

This is the case if the button is pressed login Calls and checks the username and password. If it matches the previous information, it will allow you to enter the next page. Otherwise, the username or password message will be printed incorrectly.

```
def Sing_UP(self):...
```

This function is also done by pressing a button signup Is called and if the username is not duplicate, the user can register

After entering the leader, I see the board and the start button



The leaderboard is that we open the score file and sort it based on the score value and display it in the table above.

If the user presses the play button, the game starts in fragility and the user can play and to register points, the button must beexit Press

How to play the game:

First the desired libraries in the file requirements.txt By order

pip install -r requirements.txt

Install then each file can be run separately or can be ordered python menu.pyAfter registration opened up and played (the motion Dytkshn The comments were the bugs would not get to be Jlvygry image(