

```
1 import cv2
2 import numpy as np
3 import pygame
4 import random
5 import mediapipe as mp
6 from tkinter import *
7
8 # Initialize Pygame
9 pygame.init()
10
11 # Set screen dimensions
12 screen_width = 1080
13 screen_height = 720
14
15 # Create Pygame screen
16 screen = pygame.display.
    set_mode((screen_width,
    screen_height))
17 pygame.display.set_caption("
    Gesture Racing Game")
18
19 # Load background image
20 background_image = pygame.
    image.load(r"C:\Users\rkssp\
    Desktop\virtual envi\republic\
    patriatic\road (1).jpg").
    convert()
```

```
21
22 # Load car image
23 car_image = pygame.image.load(
    r"C:\Users\rkssp\Downloads\
    car_top (3).jpg").
    convert_alpha()
24
25 # Load coin image
26 coin_image = pygame.image.load
    (r"C:\Users\rkssp\Downloads\
    coin_ (1).jpg").convert_alpha
    ()
27
28 # Colors
29 WHITE = (255, 255, 255)
30 RED = (255, 0, 0)
31 BLUE = (0, 0, 255)
32 GREEN = (0, 255, 0)
33 BLACK = (0, 0, 0)
34
35 # Car attributes
36 car_width = car_image.
    get_width()
37 car_height = car_image.
    get_height()
38 car_x = screen_width // 2 -
    car_width // 2
```

```
39 car_y = screen_height // 2 -  
    car_height // 2  
40 car_speed = 5  
41  
42 # Ball attributes  
43 ball_radius = 20  
44 ball_speed = 3  
45 blue_balls = []  
46 red_balls = []  
47 coins = []  
48  
49 # Score  
50 score = 0  
51 font = pygame.font.Font(None,  
    36)  
52  
53 # OpenCV settings  
54 mp_hands = mp.solutions.hands  
55 hands = mp_hands.Hands(  
    static_image_mode=False,  
    max_num_hands=1)  
56 mp_drawing = mp.solutions.  
    drawing_utils  
57 cap = cv2.VideoCapture(0) #  
    Use the default camera  
58  
59 # Initialize face detection
```

```
60 mp_face_detection = mp.  
    solutions.face_detection  
61 face_detection =  
    mp_face_detection.  
    FaceDetection(  
        min_detection_confidence=0.5)  
62  
63 # Define hand positions  
64 HAND_LEFT = 'left'  
65 HAND_RIGHT = 'right'  
66 HAND_UP = 'up'  
67 HAND_DOWN = 'down'  
68  
69 # Define head positions  
70 HEAD_LEFT = 'left'  
71 HEAD_RIGHT = 'right'  
72 HEAD_UP = 'up'  
73 HEAD_DOWN = 'down'  
74  
75 # Initialize webcam position  
    and dimensions  
76 webcam_x = 20  
77 webcam_y = 20  
78 webcam_width = 160  
79 webcam_height = 120  
80  
81 # Variable to track mouse drag
```

```
81  state
82  is_dragging = False
83
84
85  # Function to check hand
position
86  def check_hand_position(
    hand_landmarks):
87      if hand_landmarks:
88          for hand_landmark in
            hand_landmarks:
89              landmarks =
                hand_landmark.landmark
90              x_values = [
                landmark.x for landmark in
                    landmarks]
91              y_values = [
                landmark.y for landmark in
                    landmarks]
92              center_x = (max(
                x_values) + min(x_values)) /
                2
93              center_y = (max(
                y_values) + min(y_values)) /
                2
94
95              if center_x < 0.4
```

```
95 :
96         return
97     HAND_LEFT
98         elif center_x > 0
99         .6:
100             return
101             HAND_RIGHT
102             elif center_y < 0
103             .4:
104                 return
105                 HAND_UP
106                 elif center_y > 0
107                 .6:
108                     return
109                     HAND_DOWN
110             return None
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

```
110 .location_data.  
    relative_bounding_box  
111         cx = int(bboxC.  
    xmin * screen_width + bboxC.  
    width * screen_width / 2)  
112         cy = int(bboxC.  
    ymin * screen_height + bboxC.  
    height * screen_height / 2)  
113  
114         if cx < 0.4 *  
    screen_width:  
115             return  
    HEAD_LEFT  
116         elif cx > 0.6 *  
    screen_width:  
117             return  
    HEAD_RIGHT  
118         elif cy < 0.4 *  
    screen_height:  
119             return  
    HEAD_UP  
120         elif cy > 0.6 *  
    screen_height:  
121             return  
    HEAD_DOWN  
122     return None  
123
```

```
124
125 # Function to create a new
    blue ball
126 def create_blue_ball():
127     ball_x = screen_width
128     ball_y = random.randint(
        ball_radius, screen_height -
        ball_radius)
129     return {'x': ball_x, 'y'
        : ball_y}
130
131
132 # Function to create a new
    red ball
133 def create_red_ball():
134     ball_x = screen_width
135     ball_y = random.randint(
        ball_radius, screen_height -
        ball_radius)
136     return {'x': ball_x, 'y'
        : ball_y}
137
138
139 # Function to create a new
    coin
140 def create_coin():
141     coin_x = screen_width
```



```
142     coin_y = random.randint(
        ball_radius, screen_height -
        ball_radius)
143     return {'x': coin_x, 'y'
        : coin_y}
144
145
146 # Function to start the
    selected game mode
147 def start_game(mode):
148     if mode == "Hand Gesture
        Control":
149         hand_gesture_control
            ()
150     elif mode == "Head
        Gesture Control":
151         head_gesture_control
            ()
152
153
154 # Function for hand gesture
    control
155 def hand_gesture_control():
156     global car_x, car_y,
        score, blue_balls, red_balls
        , coins, is_dragging,
        webcam_x, webcam_y
```

```
157
158     running = True
159     clock = pygame.time.Clock
160     ()
161     while running:
162         screen.blit(
163             background_image, (0, 0))
164         ret, frame = cap.read
165         ()
166         if not ret:
167             break
168         rgb_frame = cv2.
169         cvtColor(frame, cv2.
170             COLOR_BGR2RGB)
171         results = hands.
172         process(rgb_frame)
173         for event in pygame.
174             event.get():
175                 if event.type ==
176                     pygame.QUIT:
177                         running =
178                             False
179                     elif event.type
```

```
174 == pygame.KEYDOWN:
175         if event.key
           == pygame.K_ESCAPE:
176                 running
           = False
177
178         hand_position =
           check_hand_position(results.
           multi_hand_landmarks)
179
180         if hand_position ==
           HAND_LEFT:
181                 car_x -=
           car_speed
182                 if car_x < 0:
183                         car_x = 0
184                 elif hand_position
           == HAND_RIGHT:
185                         car_x +=
           car_speed
186                 if car_x >
           screen_width - car_width:
187                         car_x =
           screen_width - car_width
188                 elif hand_position
           == HAND_UP:
189                         car_y -=
```

```
189 car_speed
190         if car_y < 0:
191             car_y = 0
192         elif hand_position
           == HAND_DOWN:
193             car_y +=
           car_speed
194         if car_y >
           screen_height - car_height:
195             car_y =
           screen_height - car_height
196
197         if random.randint(0,
           100) < 7:
198             blue_balls.append
           (create_blue_ball())
199         if random.randint(0,
           100) < 5:
200             red_balls.append(
           create_red_ball())
201         if random.randint(0,
           1000) < 1:
202             coins.append(
           create_coin())
203
204         for ball in
           blue_balls:
```

```
205             ball['x'] -=
            ball_speed
206             pygame.draw.
            circle(screen, BLUE, (ball['x
            '], ball['y']), ball_radius)
207
208             for ball in red_balls
            :
209                 ball['x'] -=
                ball_speed
210                 pygame.draw.
                circle(screen, RED, (ball['x'
                ], ball['y']), ball_radius)
211
212             for coin in coins:
213                 coin['x'] -=
                ball_speed
214                 screen.blit(
                coin_image, (coin['x'], coin[
                'y']))
215
216             for ball in
            blue_balls:
217                 if car_x < ball['
                x'] < car_x + car_width and
                car_y < ball['y'] < car_y +
                car_height:
```

```
218             score += 5
219             blue_balls.
                remove(ball)
220             for ball in red_balls
                :
221                 if car_x < ball['
x'] < car_x + car_width and
car_y < ball['y'] < car_y +
car_height:
222                     score -= 10
223                     red_balls.
                        remove(ball)
224                     for coin in coins:
225                         if car_x < coin['
x'] < car_x + car_width and
car_y < coin['y'] < car_y +
car_height:
226                             score += 20
227                             coins.remove(
coin)
228
229                     screen.blit(car_image
, (car_x, car_y))
230
231                     score_text = font.
render("Score: " + str(score
), True, BLACK)
```

```
232         screen.blit(
            score_text, (10, 10))
233
234         # Blit webcam frame
            onto the screen
235         display_webcam_feed()
236
237         pygame.display.flip()
238         clock.tick(30)
239
240     cap.release()
241     cv2.destroyAllWindows()
242
243
244 # Function for head gesture
            control
245 def head_gesture_control():
246     global car_x, car_y,
        score, blue_balls, red_balls
        , coins, is_dragging,
        webcam_x, webcam_y
247
248     running = True
249     clock = pygame.time.Clock
        ()
250
251     while running:
```

```
252         screen.blit(
            background_image, (0, 0))
253
254         ret, frame = cap.read
            ()
255         if not ret:
256             break
257
258         rgb_frame = cv2.
            cvtColor(frame, cv2.
                COLOR_BGR2RGB)
259         face_detections =
            face_detection.process(
                rgb_frame)
260
261         for event in pygame.
            event.get():
262             if event.type ==
                pygame.QUIT:
263                 running =
                    False
264                 elif event.type
                    == pygame.KEYDOWN:
265                     if event.key
                        == pygame.K_ESCAPE:
266                         running
                            = False
```



```
267
268         head_position =
        check_head_position(
        face_detections)
269
270         if head_position ==
        HEAD_LEFT:
271             car_x -=
        car_speed
272             if car_x < 0:
273                 car_x = 0
274             elif head_position
        == HEAD_RIGHT:
275                 car_x +=
        car_speed
276             if car_x >
        screen_width - car_width:
277                 car_x =
        screen_width - car_width
278             elif head_position
        == HEAD_UP:
279                 car_y -=
        car_speed
280             if car_y < 0:
281                 car_y = 0
282             elif head_position
        == HEAD_DOWN:
```

```
283             car_y +=
            car_speed
284             if car_y >
            screen_height - car_height:
285                 car_y =
            screen_height - car_height
286
287             if random.randint(0,
            100) < 7:
288                 blue_balls.append
            (create_blue_ball())
289             if random.randint(0,
            100) < 5:
290                 red_balls.append(
            create_red_ball())
291             if random.randint(0,
            1000) < 1:
292                 coins.append(
            create_coin())
293
294             for ball in
            blue_balls:
295                 ball['x'] -=
            ball_speed
296                 pygame.draw.
            circle(screen, BLUE, (ball['x
            '], ball['y']), ball_radius)
```

```
297
298         for ball in red_balls
299             :
300                 ball['x'] -=
301                 ball_speed
302                 pygame.draw.
303                 circle(screen, RED, (ball['x'
304                 ], ball['y']), ball_radius)
305
306         for coin in coins:
307             coin['x'] -=
308             ball_speed
309             screen.blit(
310             coin_image, (coin['x'], coin[
311             'y']))
312
313         for ball in
314         blue_balls:
315             if car_x < ball['
316             x'] < car_x + car_width and
317             car_y < ball['y'] < car_y +
318             car_height:
319
320                 score += 5
321                 blue_balls.
322                 remove(ball)
323         for ball in red_balls
324             :
```

```
311         if car_x < ball['
    x'] < car_x + car_width and
    car_y < ball['y'] < car_y +
    car_height:
312             score -= 10
313             red_balls.
    remove(ball)
314         for coin in coins:
315             if car_x < coin['
    x'] < car_x + car_width and
    car_y < coin['y'] < car_y +
    car_height:
316                 score += 20
317                 coins.remove(
    coin)
318
319         screen.blit(car_image
    , (car_x, car_y))
320
321         score_text = font.
    render("Score: " + str(score
    ), True, BLACK)
322         screen.blit(
    score_text, (10, 10))
323
324         # Blit webcam frame
    onto the screen
```

```
325         display_webcam_feed()
326
327         pygame.display.flip()
328         clock.tick(30)
329
330     cap.release()
331     cv2.destroyAllWindows()
332
333
334     # Function to capture webcam
frame and display in Pygame
window
335     def display_webcam_feed():
336         global is_dragging,
webcam_x, webcam_y
337
338         # Capture a frame from
the webcam
339         ret, frame = cap.read()
340         if ret:
341             # Rotate the frame by
90 degrees counterclockwise
342             rotated_frame = cv2.
rotate(frame, cv2.
ROTATE_90_COUNTERCLOCKWISE)
343
344             # Convert the frame
```

```
344 to RGB format
345         rgb_frame = cv2.
           cvtColor(rotated_frame, cv2.
           COLOR_BGR2RGB)
346
347         # Resize the frame to
           match the dimensions of the
           webcam surface
348         resized_frame = cv2.
           resize(rgb_frame, (
           webcam_width, webcam_height))
349
350         # Convert the resized
           frame to a Pygame surface
351         webcam_surface =
           pygame.surfarray.make_surface
           (resized_frame)
352
353         # Blit the webcam
           surface onto the screen
354         screen.blit(
           webcam_surface, (webcam_x,
           webcam_y))
355
356         # Handle mouse events
           for dragging the webcam feed
357         mouse_x, mouse_y =
```

```
357 pygame.mouse.get_pos()
358         mouse_click = pygame.
mouse.get_pressed()
359
360         if webcam_x < mouse_x
< webcam_x + webcam_width
and webcam_y < mouse_y <
webcam_y + webcam_height:
361             if mouse_click[0
]:
362                 is_dragging
= True
363             elif not mouse_click[
0]:
364                 is_dragging =
False
365
366             if is_dragging:
367                 webcam_x,
webcam_y = mouse_x -
webcam_width // 2, mouse_y -
webcam_height // 2
368             else:
369                 is_dragging =
False
370
371         else:
```

```
372         print("Error: Unable
           to capture frame from the
           webcam")
373
374 # GUI Popup to select game
           mode
375 def select_game_mode():
376     root = Tk()
377     root.title("Select Game
           Mode")
378
379     def start_hand_gesture():
380         root.destroy()
381         start_game("Hand
           Gesture Control")
382
383     def start_head_gesture():
384         root.destroy()
385         start_game("Head
           Gesture Control")
386
387     hand_button = Button(root
           , text="Hand Gesture Control"
           , command=start_hand_gesture)
388     hand_button.pack()
389
390     head_button = Button(root
```



```
390 , text="Head Gesture Control"  
    , command=start_head_gesture)  
391     head_button.pack()  
392  
393     root.mainloop()  
394  
395 # Run the game mode selection  
    GUI  
396 select_game_mode()
```