



ASSISTANCE FOR VISION IMPAIRED USING IMAGE PROCESSING

¹Sri Harini. S, ²Sharal Maria Tina. V, ³Samitha. S, ⁴Dr. H. Mangalam

¹Undergraduate Student, ² Undergraduate Student, ³ Undergraduate Student, ⁴Professor

¹ Department of Electronics and Communication Engineering,

¹ Sri Ramakrishna Engineering College, Coimbatore, India

Abstract : Vision impairments often require assistance in recognizing objects and faces in new environments. However, visually impaired individuals often rely on others for assistance. This research uses image processing to identify trained faces and detect items in front of the visually impaired person. The project uses the Festival text-to-speech library and the Open CV Python computer vision library. The visually impaired individual receives the names of the detected items or recognized people in audio format, enabling them to be independent in most circumstances. This technology aims to help individuals with vision impairments navigate in their daily lives.

Keywords: *Object Detection, Face Recognition, Image Processing, Computer Vision, OpenCV.*

I INTRODUCTION

People with vision impairments still struggle with indoor movement and object recognition in daily life. Ultrasonic sensors were used in earlier approaches to this challenge in order to identify obstructions. The Yolo deep learning architecture allows for the precise creation of a camera-based object detection system. To further assist those with vision impairments, text-to-speech converters can be utilized to convey audio.^[1]

Therefore, utilizing the newest technology to identify nearby objects and people is the suggested option.^[2] Face recognition can be achieved through the use of the Local Binary Pattern Histogram technique. Despite variations in the ambient lighting, it is resilient in obtaining the intended features from the frames.^[3]

II LITERATURE SURVEY

There has been significant advancement in the fields of object detection and facial recognition. The blind aid system consists of three main components: picture gathering, decision-making, and feature extraction. The process of gathering visual data using a camera is called image acquisition, as opposed to feature. Relevant visual features are extracted by extraction techniques so as to represent the context. After being extracted, these features are fed into an intelligent machine learning model to identify and classify objects, barriers, and landmarks. The model has been trained on relevant datasets. In order to help the user understand their surroundings and navigate safely, the decision-making portion of the system interprets the model's output and provides tactile or aural information.^[4]

In order to save processing time, object detection in this paper is carried out after the RGB image has been converted to a grayscale image. Before the photos are given into the algorithm, they are normalized. The object's position is ascertained by measuring the bounding box's height and width. With the use of the Google text-to-speech API, the name of the discovered object is output as audio. The objects that have a probability higher than the threshold value are identified by the system.^[1]

III PROPOSED SYSTEM

The project involves object detection and face-recognition. COCO (Common Objects in Context), a dataset is used to detect objects. In face-recognition the images containing faces of the people is captured and saved as .png image with the person's name as the file name. The capturing and saving of image are done using cv2(computer vision), python.

The OpenCV (Open-source computer vision library) supports the deep learning algorithms for object detection. Thus, the YOLO (You Only Look Once), a deep learning algorithm is used. The OpenCV supports the Haar cascade classifier algorithm, which is used for face detection and the Local

Binary Pattern Histogram Algorithm for face-recognition.

The object detection is performed by using a pre-trained model, trained with the COCO dataset by the YOLO (You Only Look Once) algorithm.

The face-recognition includes two steps, the first step is to detect the presence of face. The pre-trained model by Haar cascade classifier algorithm is used to detect the presence of face.

The second step is to recognize the face, whom it does belong. For this step, the dataset containing the face images with the person's name as the file name is used. This dataset is trained by the Local Binary Pattern Histogram (LBPH) algorithm. The trained model is saved in the YAML (Yet Another Markup Language or YAML ain't markup language) format.

The Hardware consists of Raspberry Pi 3 B+, a pi camera is connected to the raspberry pi through Camera Serial Interface and an earphone is connected to the raspberry pi through a audio jack.

The project is written in python language and the code is uploaded to the Raspberry Pi 3B+, to run the code the Thonny Python IDE (Integrated Development Environment) is used.

BLOCK DIAGRAM

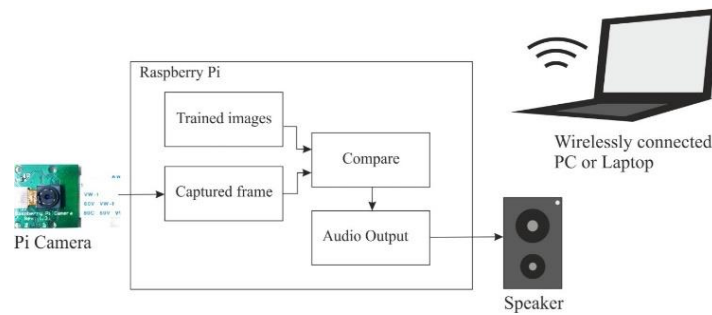


Figure 1

From the Figure 1 the block diagram of the proposed system and the connections among the hardware components can be understood

FLOW DIAGRAM

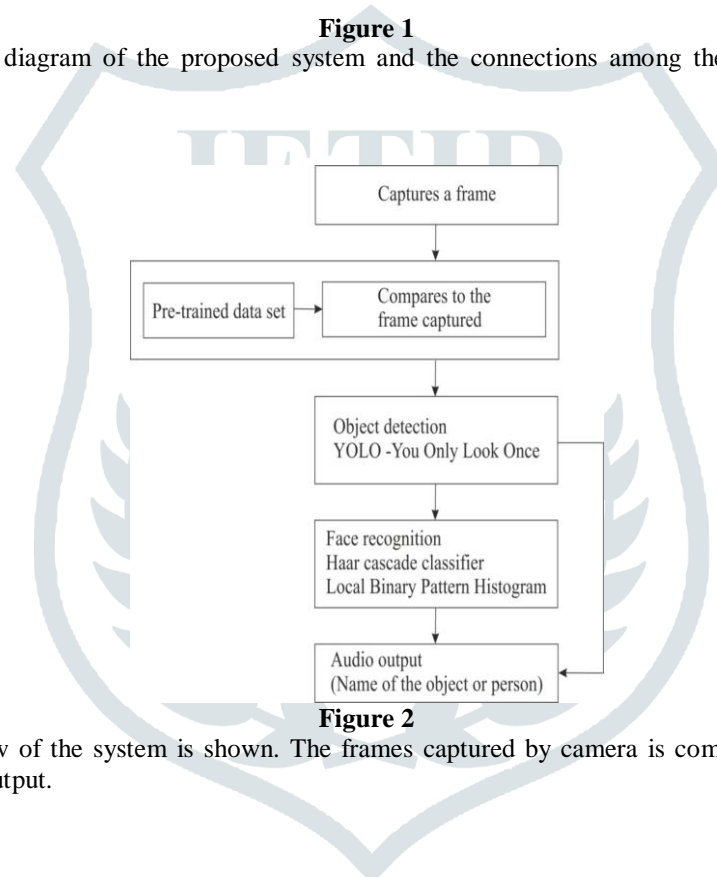


Figure 2

In the Figure 2, the work flow of the system is shown. The frames captured by camera is compared to the trained datasets to deliver an appropriate audio output.

IV METHODOLOGY

4.1 RASPBERRY P 3B+



Figure 3

The Broadcom BCM2837B0 processor-equipped Raspberry Pi single-board computer is widely used in hardware and Internet of Things (IoT) applications. Because of its price and adaptability, it's a great tool for beginners learning to program from scratch.

The Raspberry Pi is a great place for prospective programmers to start since it offers a hands-on platform for studying coding ideas and electronics principles, along with a wealth of community support and resources.

4.2 PI CAMERA

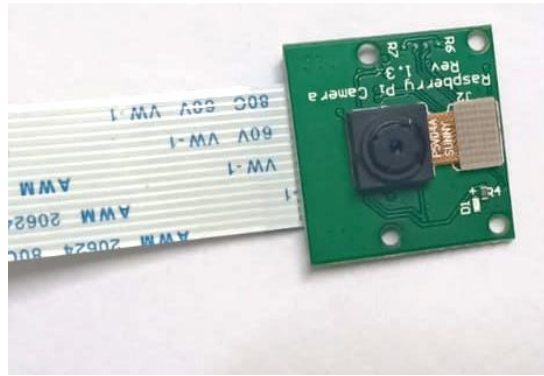


Figure 4

A Raspberry Pi board is utilized with the Pi camera. The Raspberry Pi device is used to record and capture photos in many different projects.

The input device is the Pi camera.

The frames are captured and processed further after that. Five Megapixel cameras are utilized for still photos, while a 1080p camera with a 30 frames per second maximum resolution is used to take videos.

4.3 EARPHONE



Figure 5

The output device is an earphone or speaker that provides audio output to people who are blind or visually impaired.

In addition to improving the user's awareness and making navigation and interaction with their surroundings easier, this audio feedback provides information about the surrounding environment.

4.4 Thonny python IDE:

An Integrated Development Environment (IDE) designed specifically for learning and working on Python projects is called Thonny Python. Because of its intuitive interface, which makes code development, execution, and debugging easier, it's especially good for novices.

Developers can quickly create image processing techniques using certain algorithms within this IDE. Users can concentrate on improving their programming abilities and learning the nuances of image processing by using Thonny's solid features and simplicity, which simplify the learning process.

Consequently, Thonny Python becomes the preferred option for individuals wishing to learn Python programming and take on image processing projects.

4.5 OpenCV:

With its wide range of features, OpenCV (Open Source Computer Vision Library) is a crucial library for computer vision projects. OpenCV was first developed in C++ but now supports MATLAB, Octave, Python, and Java, among other computer languages.

Its wide range of features enable developers to investigate and apply various computer vision technologies. OpenCV is a vital tool for both researchers and practitioners, facilitating a wide range of applications from image processing to machine learning.

Its open-source design encourages cooperation and creativity, which strengthens its standing as a pioneer in the computer vision industry.

4.6 Festival (library):

Festival is a free program that is widely used in many different applications and has the ability to convert text to speech. It is primarily a C++ programming language tool, but it may also be used to convert text to speech with good results.

The broad use of Festival demonstrates its efficiency and adaptability in speech synthesis applications. Festival is essential to improving accessibility and usability, whether in communication aids, educational resources, or assistive technologies.

Its open-source design encourages ongoing innovation and development, guaranteeing its applicability and usefulness in a variety of settings.

4.7 Object Detection:

The COCO (Common Objects in Context), a pre-trained dataset, is used for object detection. The YOLO (You Only Look Once) algorithm is used to train the COCO dataset.

4.8 Face recognition:

Face recognition involves two steps: the first is to detect the presence of the face in an image, and the next is to recognize the person to whom the face belongs by comparing it to the trained dataset.

To detect the presence of a face in an image, the Haar cascade classifier algorithm is used. A pre-trained Haar cascade model is used to detect faces in the image.

The Local Binary Pattern Histogram algorithm is used to recognize the faces detected in the image by comparing them to the trained dataset containing the faces of the people and their names as labels.

V RESULTS

5.1 Hardware setup:

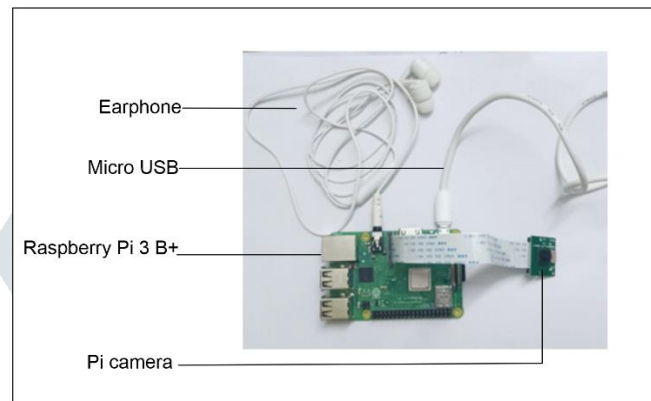


Figure 6

Serving as the brains behind the system, the Raspberry Pi makes it easier to connect the input (a Pi camera) and output (earphones or speakers).

Through the camera serial interface, the Pi camera and Raspberry Pi may communicate, allowing for the processing and capturing of images. Likewise, the audio jack on the speaker or earphone is connected to the Raspberry Pi to enable the user to get audible feedback.

Furthermore, the Raspberry Pi has remote access and control capabilities that let users monitor, configure, and communicate with the system from a laptop or personal computer. The system's adaptability and usefulness are improved by its remote accessibility.

5.2 Training the faces

5.2.1 Labelling the face to be trained

```
photo.py x
1 import cv2, os, sys, time
2 import numpy as np
3 from PIL import Image
4 from config import *
5
6 i=0
7
8 video_capture = cv2.VideoCapture(0) #Set the source webcam
9 video_capture.set(3,640)
10 video_capture.set(4,480)
11 print("Enter 'c' to capture the photo\n")
12 print("Enter 'q' to quit.\n\n")
13 print("Waiting to capture photo.....\n\n")
14
15 while True:
16     n = input("Enter: ")
17     if(n=='q'):
18         print("Quitting..")
19         break
20     if(n=='c'):
21         name = input("Enter name: ")
22         neram = str(int(time.time()))
23         while i<30:
24             ret, frame = video_capture.read()
25             gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
26             cv2.imshow("Video", gray)
27             if cv2.waitKey(1) & 0xFF == ord('n'):
28                 cv2.imwrite(db_path+"/"+str(name)+"_"+neram+str(i)+".png", gray)
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
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
1000
```

Figure 7

5.2.2 Capturing the image

```
photo.py x
1 import cv2, os, sys, time
2 import numpy as np
3 from PIL import Image
4 from config import *
5
6 i=0
7
8 video_capture = cv2.VideoCapture(0) #Set the source webcam
9 video_capture.set(3,640)
10 video_capture.set(4,480)
11 print("Enter 'c' to capture the photo\n")
12 print("Enter 'q' to quit.\n\n")
13 print("Waiting to capture photo.....\n\n")
14
15 while True:
16     n = input("Enter: ")
17     if(n=='q'):
18         print("Quitting..")
19         break
20     if(n=='c'):
21         name = input("Enter name: ")
22         neram = str(int(time.time()))
23         while i<30:
24             ret, frame = video_capture.read()
25             gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
26             cv2.imshow("Video", gray)
27             if cv2.waitKey(1) & 0xFF == ord('n'):
28                 cv2.imwrite(db_path+"/"+str(name)+"_"+neram+str(i)+".png", gray)
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
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
1000
```

Figure 8

Output for detecting cell phone is in the Figure 13

```
blind py x
1 import cv2
2 import time
3 import os, sys, pickle
4 import time
5 import numpy as np
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
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
1000
```

Figure 13

The Detection of stop sign can be seen in Figure 14



Figure 14

Output for detecting stop sign is in the Figure 15

```
1 import cv2
2 import time
3 import os, sys, pickle
4 import time
5 import numpy as np
6 from configs import *
7 import RPi.GPIO as GPIO
8 import time
9
10
11
12
13
14
15 faceCascade = cv2.CascadeClassifier('bin/haarcascade_frontalface_default.xml')
16 profileCascade = cv2.CascadeClassifier('bin/haarcascade_profileface.xml')
17
18 recognizer = cv2.face.LBPHFaceRecognizer_create()
19 recognizer.read('face_rec_saved_model.yaml')
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
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
1000
```

Figure 15

The Detection of bottle can be seen in Figure 16



Figure 16

Output for detecting bottle is in the Figure 17

```

1 import cv2
2 import time
3 import os, sys, pickle
4 import time
5 import numpy as np
6 from configs import *
7 import RPi.GPIO as GPIO
8 import time
9
10
11
12
13
14
15 faceCascade = cv2.CascadeClassifier('bin/haarcascade_frontalface_default.xml')
16 profileCascade = cv2.CascadeClassifier('bin/haarcascade_profileface.xml')
17
18 recognizer = cv2.face_LBPHFaceRecognizer.create()
19 recognizer.read('face_rec_saved_model.yaml')
20
21
Shell
face 30
True
[ WARN:0] global ../modules/videoio/src/cap_gstreamer.cpp (961) open OpenCV | GStreamer
duration=-1
object 1
bottle
object 2
bottle
person
object 3
bottle

```

Figure 17

5.4 Face recognition



```

1 import cv2
2 import time
3 import os, sys, pickle
4 import time
5 import numpy as np
6 from configs import *
7 import RPi.GPIO as GPIO
8 import time
9
10
11
12
13
14
15 faceCascade = cv2.CascadeClassifier('bin/haarcascade_frontalface_default.xml')
16 profileCascade = cv2.CascadeClassifier('bin/haarcascade_profileface.xml')
17
18 recognizer = cv2.face_LBPHFaceRecognizer.create()
19 recognizer.read('face_rec_saved_model.yaml')
20
21
Shell
face 16
True
face 17
True
1. 197.62661
face 18
True
5. 64.022948
It is predicted as sriharini
UniSyn: using
face 19
True
5. 58.433673
It is predicted as sriharini
UniSyn: using default diphone ax-ax for hh-
face 20
True
5. 63.958197006610916
It is predicted as sriharini

```

Figure 18

The output for Face recognition is in the Figure 18

VI CONCLUSION

Using a Haar cascade algorithm, a sophisticated technique for object and face detection and recognition was created to detect nearby items and faces. This invention helps people with vision impairments become more independent

This consolidation provides a flexible solution that can be used in a variety of areas while addressing earlier constraints. By integrating these features, our technology improves visually impaired people's accessibility and usefulness and gives them the confidence to effectively navigate their environment. With this development, assistive technology has advanced significantly and now offers a complete solution for those who are visually impaired to interact with their surroundings in an effective manner. Furthermore, the Haar cascade algorithm's adaptability allows it to be applied in a variety of areas, expanding its possibilities beyond helping the blind. All things considered, our comprehensive approach improves the quality of life and simplifies accessibility for people with visual impairments, allowing them to live more independently and with greater ease.

VII ACKNOWLEDGEMENT

We would like to thank Dr. H. Mangalam, the project guide, the Department of Electronics and Communication Engineering professors, Sri Ramakrishna Engineering College, Vattamalaipalayam, Coimbatore, our friends and classmates, and the authors of the research and articles that are cited. Their knowledge and assistance have greatly influenced the article's content, and their contributions have greatly influenced how the topic is understood and analyzed. We also want to express our gratitude to our families for their constant support and tolerance during our academic careers.

REFERENCES

- [1] Karmarkar, R. R., & Hommane, V. N. (2021). Object detection system for the blind with voice guidance. *Int. J. Eng. Appl. Sciences Technol*, 6(2), 67-70.
- [2] Aralikatti, A., Appalla, J., Kushal, S., Naveen, G. S., Lokesh, S., & Jayasri, B. S. (2020). Real-time object detection and face recognition system to assist the visually impaired. In *Journal of Physics: Conference Series* (Vol. 1706, Issue 1, p. 012149). IOP Publishing. <https://doi.org/10.1088/1742-6596/1706/1/012149>
- [3] Chen, Y. C., Liao, Y. S., Shen, H. Y., Syamsudin, M., & Shen, Y. C. (2022). An Enhanced LBPH Approach to Ambient-Light-Affected Face Recognition Data in Sensor Network. *Electronics*, 12(1), 166.
- [4] Kumar, N., Sharma, S., Abraham, I. M., & Sathya Priya, S. (2022). Blind Assistance System Using Machine Learning. In *Third International Conference on Image Processing and Capsule Networks* (pp. 419–432). Springer International Publishing. https://doi.org/10.1007/978-3-031-12413-6_33