IntelliCrowd Technology

An intelligent crowd surveillance system

Authors:

Pulkit Mehta
Jasper
Leon
Mohit
Lakshya

Introduction

IntelliCrowd Technology is an intelligent crowd surveillance system which when applied over a supervised or unsupervised area can give a focused assistance by finding the Optimal Point which when further combined with any type of service (Lighting, Sound or Cooling), can deliver best experience and save energies for maybe larger areas.

It uses most advanced and modern Computer Vision Techniques combined with Artificial Intelligence algorithms for its wholesome computation.

This project is aimed to be always maintained and improved by our active members as more AI techniques will emerge in coming years.

We welcome contributors for further support.

Methodology

It involved a camera which will be fixed on ceiling or on a drone.
Our scripts will take the live video feed from the camera, pushes it for ComputerVision techniques and finds the objects which stand separate from the background.

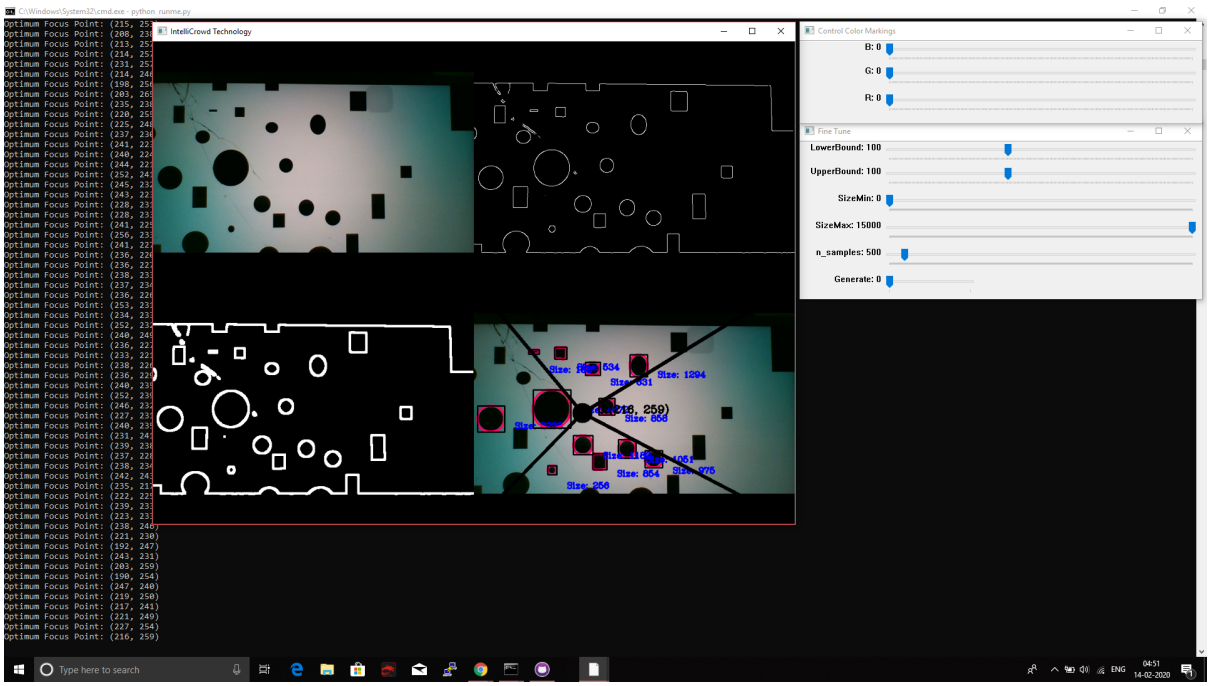
Then these objects are highlighted and judged which one is to be taken into account through supervised predictive methods of Machine Learning.

Then these favorable objects are spotted and located. An optimal point is then computed which will be the focus point for us in the field of view.

Then all the services (Lighting, Sound or Air conditioning) will be concentrated on that.
And their intensities will be varying directly or inversely depending on the type of service being delivered.

Usage

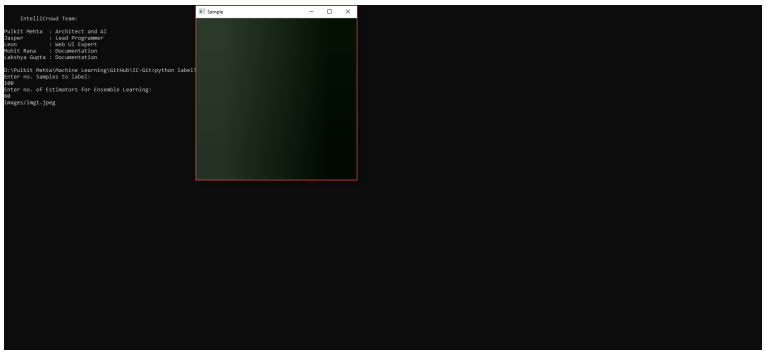
1. Install a Camera overhead and connect it with the computer.
2. Run the application. On the very first run, our technology needs to be taught a little of what is to be recognized and treated as favorable.
3. For the demonstration we will use just two different shapes [Squares and Circles], which will be our minimal representation of a Human and a Non Human object when we will sense the crowd.
4. We will set the size of our data on which we have to train it by adjusting n_samples in the Fine Tune Window. More the data, more will be the accuracy, but more the latency, so there needs to be a fair tradeoff between them.



5. For the first time we will show the camera, some of these shapes in a single picture. in different moving variations and that too randomly.
6. There will be some error markings or some undetected important objects, So just use the Fine Tune window and adjust the Upperbound, Lowerbound, Sizemin, Sizemax to experiment and go with the best combination.
7. When satisfied, just move the Generate slider towards right, and within fraction of seconds it will produce the required samples. Press ‘q’ to exit it now.
8. Now, its the time of teaching. Run labelTrain script
9. It will ask for how many samples you want to teach it with, More the samples, more will be the accuracy.

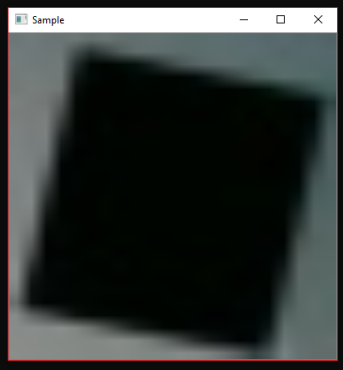
```
D:\Pulkit Mehta\Machine Learning\GitHub\IC-Git>python labelTrain.py
Enter no. Samples to label:
```

10. Then it will ask about the value of estimators, provide it something at least more than 10. Again, more the estimators, more will be the accuracy, but more the prediction latency. Note: too many estimators can lag your device.

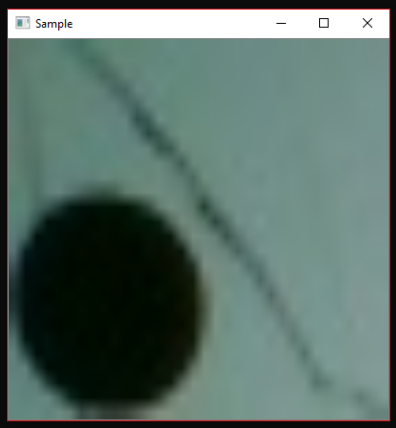


11. Press enter, images will start popping up. For each image, press ‘y’ if it is the favorable object, else press any other key

```
D:\Pulkit Mehta\Machine Learning\GitHub\IC-Git>python labelTrain.py
Enter no. Samples to label:
100
Enter no. of Estimators for Ensemble Learning:
60
images/img1.jpeg
images/img2.jpeg
images/img3.jpeg
images/img4.jpeg
images/img5.jpeg
images/img6.jpeg
images/img7.jpeg
images/img8.jpeg
images/img9.jpeg
images/img10.jpeg
images/img11.jpeg
images/img12.jpeg
images/img13.jpeg
images/img14.jpeg
images/img15.jpeg
images/img16.jpeg
images/img17.jpeg
images/img18.jpeg
images/img19.jpeg
images/img20.jpeg
images/img21.jpeg
images/img22.jpeg
images/img23.jpeg
images/img24.jpeg
images/img25.jpeg
images/img26.jpeg
```




```
D:\Pulkit Mehta\Machine Learning\GitHub\IC-Git>python labelTrain.py
Enter no. Samples to label:
100
Enter no. of Estimators for Ensemble Learning:
60
images/img1.jpeg
images/img2.jpeg
images/img3.jpeg
images/img4.jpeg
images/img5.jpeg
images/img6.jpeg
images/img7.jpeg
images/img8.jpeg
images/img9.jpeg
images/img10.jpeg
images/img11.jpeg
images/img12.jpeg
images/img13.jpeg
images/img14.jpeg
images/img15.jpeg
images/img16.jpeg
images/img17.jpeg
images/img18.jpeg
images/img19.jpeg
images/img20.jpeg
images/img21.jpeg
images/img22.jpeg
images/img23.jpeg
images/img24.jpeg
images/img25.jpeg
images/img26.jpeg
images/img27.jpeg
```

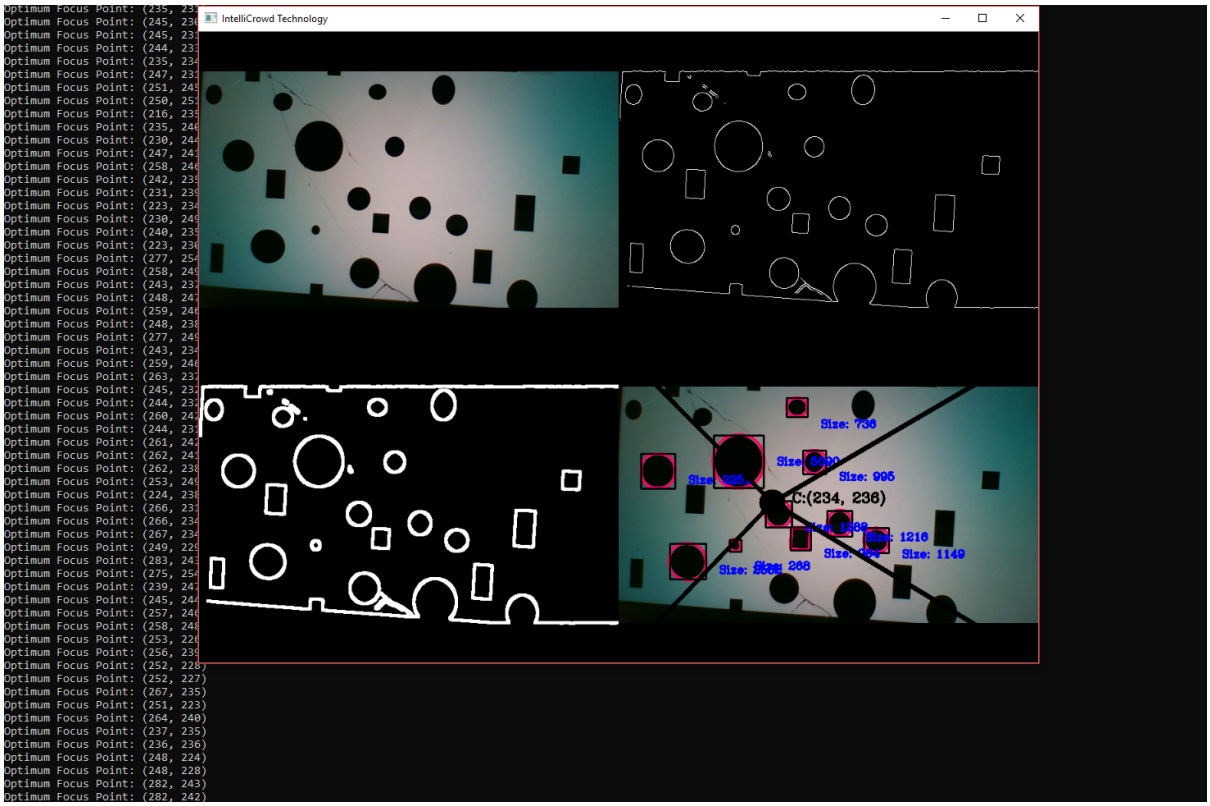


12. After this process has been done, the scripts will show the accuracies. Check the test accuracy, if its less, try with more estimators or samples.

```
images/img73.jpeg
images/img74.jpeg
images/img75.jpeg
images/img76.jpeg
images/img77.jpeg
images/img78.jpeg
images/img79.jpeg
images/img80.jpeg
images/img81.jpeg
images/img82.jpeg
images/img83.jpeg
images/img84.jpeg
images/img85.jpeg
images/img86.jpeg
images/img87.jpeg
images/img88.jpeg
images/img89.jpeg
images/img90.jpeg
images/img91.jpeg
images/img92.jpeg
images/img93.jpeg
images/img94.jpeg
images/img95.jpeg
images/img96.jpeg
images/img97.jpeg
images/img98.jpeg
images/img99.jpeg
images/img100.jpeg
Training Data...
Done!


Testing Accuracy is:  100.0%
Training Accuracy is:  100.0%
Done!
```

13. Now fire on the main application and the command line will show the optimal point coordinates along with the same visualizations of other important parameters.



Applications

This technology as explained already can be very intelligent under certain scenarios where the services needs to be focussed over people groups without wasting on empty areas.

■ Lighting:

Assume a very big hall where clusters of people are standing in different regions. Lights from every corner are on and in the less important areas where no crowd clusters are concentrated, a lot of energy is being wasted. So IntelliCrowd will detect that and any IOT assistant or a device can dim those unnecessary lights by varying their resistances varying in relation with the distance from the optimal point.

■ Sound:

Now take the same hall, in this case, the sound/ music from the speaker can be lowered for those which are nearer to the optimal point, while those further away can be raised in order to match the audio channel throughput for both left and right ears so maximum people may hear a balanced sound.

■ Other Examples:

- *Air Conditioning*
- *Disaster Management*
- *Defense Surveillance*

Softwares Used:

Jupyter, Pycharm,

Hardwares Used:

USB Camera

Programming Language:

Python