Social Distancing Violation Detection

Nattawach Sataudom
School of Engineering and Technology
Computer Science and Information Management

I. Introduction

Social Distancing Violation Detection system was developed with inspiration from COVID-19 situation that people in some country have to keep distance from each other. In the term of technical, this application extends YOLOv5[**?**] and DeepSORT[**?**] that is state-of-the-art of object detection and object tracking. I adopt N Nearest Neighbor to cluster a group of human that will help to reduce computation cost. Finally, I proposed to use pixel-meter coefficient to transform distance in pixel to meter that is more suitable for practical usage.

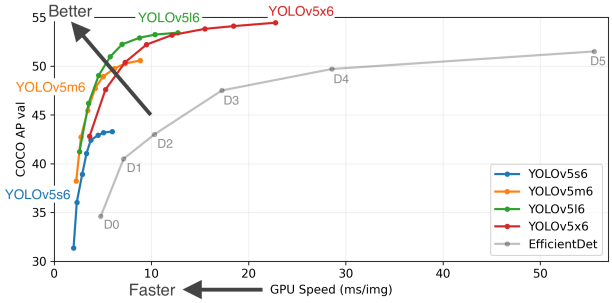
II. Related Work

The main module of this application was adopted from state-of-the-art of object detection called YOLOv5[**?**] and online multiple object tracking called DeepSORT[**?**]. The brief details of these module will be discussed in subsections.

A. YOLOv5

The object detection of this application based on YOLOv5 that is PyTorch implementation of YOLOv4 with fantastic optimization in both accuracy and performance. As figure 1, YOLOv5 can overcome EfficientDet model with significant higher in both accuracy and inference speed.

Fig. 1. YOLOv5 efficiency



B. DeepSORT

DeepSORT or Simple Online and Real-Time Tracking with a Deep Association Metric is state-of-the-art tracking algorithm that is able to track multiple object in real-time with non-constant velocity condition. The basically idea of DeepSORT is based on three parts of algorithm: state estimation, assignment problem and deep appearance descriptor. The first part is state estimation that responses to handle bounding box parsing and adding it into the state. The second part is assignment problem that adopt matching cascade to deal with decision of track adding, age updating and deleting. The third part is deep appearance descriptor that utilizes CNN with residual blocks model to extract the features, called cosine similarity matrix, from tracked object. That can help to match between current track in state and new coming track, and identify both tracked objects with difference of two similarity matrix.

III. Methodology

A. Human Detection

Human detection part has been done by PyTorch YOLOv5 with pre-trained model named yolov5l. The returned bounding box of YOLOv5 will be used as input of tracker algorithm, DeepSORT. For hyper-parameter, as my implementation, I set confidence threshold of YOLOv5 detection into 0.3 and set IoU threshold of Non-Maximum Suppression into 0.5 (that proper to minimum intersection area). I also filter the detected classes to keeps only human class into the List before parse them to DeepSORT.

B. Human Tracking

Human tracking part has been done by DeepSORT that is capable of multiple human tracking in real-time. This application utilizes DeepSORT for 2 approaches. The first one, I want to identify every detected human and keep it available in continuous frame (no detection lost). To ensure that the distance estimation will work perfectly in all sequence of frames. The second one, DeepSORT will take an important role in violation reporting, especially when we want export the violation evidence or send it via http post request to another application.

C. Distance Estimation

To estimate the distance of two object, there are 3 tasks to accomplish including distance relative coefficient, Euclidean distance among multiple objects, distance matrix construction, nearest neighbor matrix construction.

1) Distance Relative Coefficient Calculation: To estimate the relative distance of object in different depth, we have to find the relative distance between object and camera. In this application, we calculate human-height relative coefficient to handle it. The relative coefficient can be calculated as a follow equation.

$$c = \frac{h_{human}}{\lambda} \tag{1}$$

When $c :=$ relative coefficient, $h_{human} :=$ height of human and $\lambda :=$ hyper-parameter of camera setup.

2) Distance Matrix Construction: To construct the distance matrix, we start with Euclidean distance calculation of each track-pair. Then construct it into the matrix of duplicated row and column name. The physical appearance of distance matrix is represented as table I.

	track_1	track_2	track_3	track_n
track_1	0	d(1, 2)	d(1, 3)	d(1, n)
track_2	d(2, 1)	0	d(2, 3)	d(2, n)
track_3	d(3, 1)	d(3, 2)	0	d(3, n)
track_n	d(n, 1)	d(n, 2)	d(n, 3)	0

TABLE I
PHYSICAL APPEARANCE OF DISTANCE MATRIX

3) N Nearest Neighbor Matrix Construction: We adopt nearest neighbor matrix, that is the list of index of sorted children (sort by parent-child distance). The size of matrix can be fixed as hyper-parameter, that will increase the computational cost when it's increased. The physical appearance of distance matrix is represented as table II.

	child_1	child_2	child_3	child_n
parent_1	id1	id2	id3	id4
parent_2	id5	id6	id7	id8
parent_3	id9	id10	id11	id12
parent_n	id13	id14	id15	id16

TABLE II
PHYSICAL APPEARANCE OF N NEAREST NEIGHBOR MATRIX

IV. RESULT

I evaluated the application accuracy with set web cam camera. The evaluation was done on 5 pair of samples with 4 distances: 0.5 meters, 1.0 meters, 1.5 meters and 2.0 meters. As a result from figure III, the application worked well with 100% of accuracy with a specific camera setup and a fixed hyper-parameter. In the future, I have to evaluate the application with several camera setup and perform machine learning to configure the hyper-parameter automatically.

Sample	Gender	Height Diff.	Distance (m)			
			0.5	1.0	1.5	2.0
1	male-male	∼0 cm	0.5	1.0	1.5	2.0
2	male-female	∼15 cm	0.5	1.0	1.5	2.0
3	male-female	∼20 cm	0.5	1.0	1.5	1.9
4	female-female	∼5 cm	0.5	1.0	1.5	2.0

TABLE III
EVALUATION RESULT: ESTIMATED DISTANCE OF MULTIPLE SAMPLES WITH DIFFERENCE GENDER AND HEIGHT

A. Conclusion

As experimental result, the accuracy of application is exactly to 100% with tiny number of samples, testing scenario and camera setup. Meaning, the evaluation result of this application is untrustworthy unless I do more experiment on several conditions.

REFERENCES

[1] Glenn Jocher, Alex Stoken, Jirka Borovec, NanoCode012, Ayush Chaurasia, TaoXie, Francisco Ingham. (2021, April 11). ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations (Version v5.0). Zenodo. http://doi.org/10.5281/zenodo.4679653

[2] N. Wojke, A. Bewley and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017 IEEE International Conference on Image Processing (ICIP), 2017, pp. 3645-3649, doi: 10.1109/ICIP.2017.8296962.