People counting and tracking
Computer vision assignment report

Kateryna Konotopska

198234
`kateryna.konotopska@studenti.unitn.it`

Abstract. The purpose of this report is to document the project done for the first assignment of Computer Vision course in academic year 2018/2019. The task of the assignment is people detection and tracking in an artificially generated top view video. For the final solution main concepts learned during the course were put into practice: background subtraction model, filtering, color features extraction and object tracking using an observational motion model. Provided that the building blocks used for the final algorithm were simple, a decent result was archived: the estimated detected number of people is close to the ground truth; the algorithm tracks without problems people not intersecting with others and a good number of occlusions and intersections are handled with success.

Keywords: Online tracking · People counting · Occlusions · Top view.

1 Introduction

Detection and tracking of moving objects, especially people, became one of the hottest topics in the computer vision community in last years, with the diffusion of IOT and self-driving cars. Therefore the proposed project may be a solution to numerous real-life applications.

The task of this assignment was composed by 2 parts:

1. People detection and counting: for each frame f provide a number of people n in the scene; as an additional task a number of people entering and exiting the scene must be provided for each frame
2. People tracking: identify each person with an ID and give the ID's position (x, y) in each frame across the video.

The video generated for this task was a top view artificially generated video containing 763 frames with $\sim$22.8 persons per frame. The illumination in the scene was not uniform and constant in time, and people were frequently intersecting with each other making the task more difficult.

To address the people counting task a two-step procedure was used: background subtraction and subsequent contours extraction. Mixture of Gaussians [8] background subtractor was used for the first step, while for the second step the contour approximation method was applied, combined with some preprocessing filters and morphological operators.

The people tracking task was more challenging because of the occlusions and imperfections in the background subtraction stage. The complexity of the solution for this task grew up in time, starting from a simple solution like the closest blob id association, and incrementally adding pieces for addressing emerging problems. The final solution is the combination of region- and contour-based tracking with addition of Kalman filter [4] for the position estimation.

The advantage of the presented solution is its simplicity and velocity: with a processing speed of ∼17 FPS we obtained a decent model correctly identifying and tracking people in the majority of cases even in presence of occlusions and intersections.

The report is structured as follows: the methods used for people detection and the tracking algorithm details are illustrated in section 2, in section 3 the archived results are described and explained, and, in the end, in section 4 you can find final conclusions with advantages and disadvantages of the proposed solution, in the final section possible suggestions to improve the results are also provided.

2 Methods

In this section we are going to explain more in detail all the methods and motivations used for each task of the assignment, briefly describing all the techniques used and providing references for their more exhaustive explanation.

2.1 People detection and counting

This task consists in identifying, for each frame, the positions of pedestrians in the video and their total number. In our case the pedestrians are the only moving objects in the video, so detecting and counting of all the moving objects is enough for our purposes. A distinction between a person and another moving object would also be a difficult task: with the top view video we can identify some characteristics useful for people detection only on the right and left sides of the frames (such as arms, legs, etc.), while in the central part of the frame only head and shoulders can be observed; partial occlusions of some body parts would be also a problem for people recognition.

In order to detect moving objects we performed a two step procedure: background subtraction and contours extraction.

Background subtraction The key point for the background subtraction task was the choice of the image component for the analysis. Since we had the presence of shadows and variable illumination, each frame was converted to HSV color space and the hue component was chosen for the analysis because of the absence of illumination information in this component. Subsequently a Mixture of Gaussians [8] background subtractor was applied in order to obtain a binary mask containing the foreground pixels. From our observations this subtractor was the most suitable for the given problem considering its velocity and robustness. Noise was removed with morphological operations, applied in the following order: opening, closing and dilating.

Contours extraction For the contours extraction the default findContours() OpenCV's method was applied [7]. Subsequently an additional refinement was added for filtering out small contours (by using a threshold on the area of bounding boxes).

2.2 People tracking

As the first attempt of people tracking the closest blob id association was implemented. This method, of course, was not suitable for different reasons such as blob splitting, occlusions due to some couples of people walking together or to their intersection. To handle each problem various solutions were combined together:

– Kalman filters: for each ID detected in the video a Kalman filter [4] was instantiated, having 2 measurement and 4 state variables to model the position and the velocity of each x and y variable. Kalman filter was useful to track IDs in occurrences of occlusions or contour blinkering (when small contours disappeared for short periods of time). In such situations the predictions were computed using the last velocity information (without correction steps). At each step of the algorithm the predicted ID position was used to compute the nearest bounding boxes.
– Color histograms: each ID in the video was associated with a color histogram, which was an average histogram of the last n regions inside the contours corresponding the the ID in frames where the ID appeared. So, for example, assuming n=3, and assuming that the ID appeared in frame 5, 6 and 8, the histogram of the ID at this point was calculated by averaging the histograms of regions inside the contours (and not the bounding boxes) corresponding to the ID of frames 5, 6 and 8. Hue color component, again, was used to avoid problems caused by the nonuniform and inconstant illumination.
  Color information turned out particularly effective for people intersection situations, especially when people changed their walking directions, so that the Kalman filter was not able to handle those situations anymore (even by increasing the number of internal states, so by addition of acceleration modelling etc). In this case IDs were assigned according to a new similarity metric, a combination of color histograms similarity and distance similarity. Color histograms were compared using bin-by-bin comparison. The bin counts were normalized by dividing them by the total number of counts, in this way the similarity measurement of regions having different areas was more accurate.
– Introduction of stability states: all the new IDs were considered as unstable the first time they were detected in the video; if an ID was detected (and

associated to a blob) in more then k contiguous frames, it entered a stable state. If, on the other side, it disappeared from the video for m contiguous frames, it was not considered as a stable ID and was subsequently removed from the tracked IDs list.

By introducing these states the problem of some short time blinkering noise, such as small pieces of a body splitting for a couple of frames, was partially removed.

– Partial Kalman filter corrections: situations in which people walk together and split only for a few frames, or when the occlusions due to their intersections last for a longer time, cause problems to Kalman filter prediction models: kalman filters in such cases are not corrected for a long time. In order to partially indicate where the person might be, more then one ID could be assigned to a blob in the tracker internal model, and when it occured, Kalman filters corresponding to those IDs were corrected by a position which was a linear combination of IDs predicted position and the position of the blob containing multiple IDs. The blob position in this case had a weight $\alpha \in [0, 1]$, so that the filter was corrected with an "observed" position calculated as follows:

  $\hat{\boldsymbol{p_c}} = \hat{\boldsymbol{p_b}} \cdot \alpha + \hat{\boldsymbol{p_p}} \cdot (1 - \alpha)$, where $\hat{\boldsymbol{p_c}}, \hat{\boldsymbol{p_b}}$ and $\hat{\boldsymbol{p_p}}$ are the corrected , blob and predicted position vectors.

  This kind of correction was also useful for updating the positions of objects exiting the scene: since their bodies were observed only partially, the position of their contours had a smaller weight with respect to the predicted position.

– Max density tuning for avoiding ghost IDs: sometimes the possibility of keeping multiple IDs assigned to a single blob in the internal model leaded to complications when there was a body split: in such situations certain ids were managed to enter a stable state and persisted during the whole video, making difficult the histogram updates, since color histograms were updated only when a single ID was assigned to a blob. This problem was largely solved by introduction of "maximum density" controls: there was a threshold on blob area for which a blob could contain more then one person. The introduction of such threshold made the ID tracking much more stable.

  The final algorithm is summarized below.

2.3 Number of pedestrians entering/exiting the scene

For this task two gates of a fixed width were defined (left and right): every person in the gate area was entering or exiting the scene (the gate areas were delimited in the output video by violet lines). Then, in order to detect whether people were entering or exiting the scene, their velocities were analyzed. Since the new IDs velocities were not correctly estimated by Kalman filters for the first frames, IDs "age" (current frame - ID's first frame) was taken into consideration as well: all the "new" IDs were considered as IDs entering the scene.

Algorithm 1 Tracking and assigning IDs

```
    Input bounding boxes $bboxes$, frame color component $frame$
    Output IDs $bboxesIDs$
 1: procedure ASSIGNIDs
 2:     $bboxesCnt \leftarrow computeBboxesCoordinates(bboxes)$
 3:     $bboxesHists \leftarrow computeColorHistograms(bboxes, frame)$
 4:     $bboxesIDs \leftarrow initializeBboxesIDs(bboxes)$
 5:     $activeIDsPositions \leftarrow kalmanPredict(activeIDs)$
 6:     $distances \leftarrow computeEuclideanDistanceMatrix(activeIDsPositions, bboxesCnt)$
 7:     for $ID$ in $activeIDs$ do
 8:         $nearestBboxes \leftarrow selectNearestBboxes(ID, distances, n)$
 9:         $similarities \leftarrow computeCombinedSimilarities(ID, nearestBboxes, IDsHists, bboxesHists)$
10:         $winnerBbox \leftarrow mostSimilarBbox(similarities)$
11:         $bboxesIDs[winnerBbox] \leftarrow ID$
12:     $bboxesIDs \leftarrow removeGhostIDs(bboxesIDs)$
13:     $bboxesIDs \leftarrow assignNewIdsToEmptyBboxes(bboxesIDs)$
14:     $IDsHists \leftarrow updateIDsHists(bboxesIDs, bboxesHists)$
15:     $correctKalmanFilters(bboxesIDs, bboxesCnt)$
16:     $updateStabilityStates(bboxesIDs)$
17:     $updateIdsHistogramsMemory(IDsHists)$
18:     for $bboxIDs$ in $bboxesIDs$ do
19:         if $len(bboxIDs) > 1$ then
20:             $bboxIDs \leftarrow getOldestID(bboxIDs)$
```

2.4 Evaluation metrics

Evaluation of people counting was done by considering the average difference between the ground truth and the estimated counts in each frame of the video:

$error = \frac{1}{|F|} \cdot \sum_{f \in F} |c_{tf} - c_{ef}|$, where F is a set of all frames and c_{tf}, c_{ef} are the true the and estimated counts related to frame f.

For the estimation of the tracking task performance the average displacement error of selected pedestrians was calculated for a set of frames where we had both ground truth and estimated positions:

$error = \frac{1}{|F_i|} \cdot \sum_{f \in F_i} \sqrt{(ex_f - tx_f)^2 + (ey_f - ty_f)^2}$, where F_i is the intersection of frames where both ground truth and estimated positions related to the i-th ID appear, ex_f, tx_f are the estimated and ground truth x coordinates, respectively; same notation for the y coordinates.

However the parameter tuning was done considering not only the seleted IDs displacement, but the overall result, trying to cover the majority of the problematic situations.

3 Results

3.1 People detection and counting

The average error for people counting (moving objects is this case) is of $\sim$1.9 objects per frame with estimated counts average of 22.88 objects per frame, which is very close to the ground truth counts average of 22.85 objects per frame. Sometimes the number of detected persons does not represent the real number of people in the scene: we can observe in some frames 2 contours for one person and 1 contour for two or three people, so it makes the overall number of counts still close to the ground truth. Nevertheless the general result is quite good considering the illumination changes in space and in time, the change of perspective, different bodies dimensions, numerous intersections and couples or triples of pedestrians walking together.

3.2 People tracking

For the people tracking results we will first comment the displacement errors for each selected pedestrian and then we will discuss the overall performance, taking into account different kinds of problematic situations and the execution times, another important variable to consider in the online applications.

Table 1 contains the results obtained for the selected pedestrians tracking: for each ID the average displacement is reported, with the correspondent number of total frames in which it appeared in the ground truth file and the number of frames in which the pedestrian was not detected.

ID	**avg. displacement**	**# total ground truth frames**	**# missing**
10	18.29	96	5
36	24.67	188	41
42	15.15	224	9

Table 1. Results for the selected pedestrians trajectories

From this table we can observe quite accurate results for pedestrians 10 and 42, while the pedestrian 36 was lost for a bigger number of frames.

For the ID 10 we have missed frames at the beginning of video because the IDs at the beginning of the video are not still entered in the "stable" state. The missing frames of the 42-th pedestrian are due to the intersections with other people and a couple of frames in the end of the walk, when the contour was divided into two pieces, so that the velocity of the Kalman model increased (the ID was assigned to the right contour), therefore the ID left the frame prematurely according to the estimation model. The pedestrian 36 was lost towards the end of his route, after his contour merged with contours of other 3 people, also in some intersections the ID of this pedestrian was assigned to the merged contour, that is the reason of the higher average displacement.

The overall tracking solution achieved good results in cases of quick intersections, it kept track of IDs disappeared for a small number of frames (due to the lacked detection of small figures), some IDs "survived" to quite a big number of occlusions and intersections. The main weaknesses of this solution are the situations with occlusions of pedestrians for long periods of time, in this case the color histograms can not be updated, and after the pedestrians split usually the perspective is different and, as a result, when two pedestrians get separated in this case, only one of them maintains its ID (since both of the IDs were usually assigned to one of the contours). Also when the contours split or when they cover persons with varying coverage, some IDs disappear for the the same reason: one of the contours results as the most similar to all the surrounding IDs but it eventually gets only one of them, in cases when the "winner" contour is considered too small to contain more than 1 person, the other IDs are eliminated (due to the "maximum density" controls described in section 2).

The average frame rate of the video processing is of ∼17.94 FPS (with Intel i7-7700HQ CPU, single core required for the implemented non parallelized algorithm), which may be a reasonable solution for example in an online video surveillance application with 15 FPS camera.

4 Conclusions and future work

The major issues are related to the illumination and perspective changes, they leaded to the body contours splits and inaccurate contours detection, so the color histograms in such cases could not be correct. This problem can be solved by further improvements of the background subtraction model, for example with other bluring models and morphological operations. Also an introduction of additional fine grained features, such as SIFT [5] or good features to track [6] may lead to more a robust tracking in such situations, but their detection may introduce further slowdown of the algorithm.

Another problem is due to the controls of maximum density for the IDs: some of them are eliminated after being assigned to small contours already containing an ID. It would be useful to introduce a linear assignment of the current IDs to the detected contours, according to a unique distance metric taking into account euclidean distance, color similarities and velocity vectors.

The realized project is a decent solution for online applications where people walk separately from each other or intersect for a short periods of time.

5 Project implementation

The project was implemented using python3 and OpenCV library [3]. The source code can be found at [1] and the output video for the project is located at [2]

References

1. code repository, `https://github.com/katerynak/people-tracking`
2. output video, `https://drive.google.com/file/d/1yz_Kf1p2sFij4SAkIFrdNgoLwUQwwXwy/view`
3. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)

4. Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME Journal of Basic Engineering (82 (Series D)), 35–45 (1960), http://www.cs.unc.edu/\~{}welch/kalman/media/pdf/Kalman1960.pdf
5. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (Nov 2004). https://doi.org/10.1023/B:VISI.0000029664.99615.94
6. Shi, J., Tomasi, C.: Good features to track. Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition **600** (03 2000). https://doi.org/10.1109/CVPR.1994.323794
7. Suzuki, S., be, K.: Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing **30**(1), 32 – 46 (1985). https://doi.org/https://doi.org/10.1016/0734-189X(85)90016-7, http://www.sciencedirect.com/science/article/pii/0734189X85900167
8. Zivkovic, Z., van der Heijden, F.: Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognition Letters **27**(7), 773 – 780 (2006). https://doi.org/https://doi.org/10.1016/j.patrec.2005.11.005, http://www.sciencedirect.com/science/article/pii/S0167865505003521