Object detection and tracking in video from multiple surveillance cameras

TSBB11 CDIO, Technical report

Hanna Hamrell, hanha664@student.liu.se
Klara Hellgren, klahe156@student.liu.se
Denise Härnström, denha296@student.liu.se
Helena Kihlström, helki570@student.liu.se
Axel Nyström, axeny846@student.liu.se

December 17, 2018

Contents

1 Introduction

This project is about developing an analysis module for surveillance camera videos to minimize manual work when e.g. looking for persons fitting into a specific description or re-identifying a person from one video in other videos.

1.1 Problem Description

The client wanted an automatic analysis module for videos from surveillance cameras. The wishes of the client was that it should be able to detect and track people and e.g. cars or bags in surveillance video cameras. Another wish was that it should be possible to match a person detection in one video with person detections in other videos. A third wish was that all found and tracked objects should be saved in json format and that it should be possible to search among the objects based on a signalement in text form. In addition, the customer required a study of good algorithms for detection and classification as well as tracking. After approval from the customer, the requirements in table 1 were decided upon.

Within the scope of the project, all surveillance video cameras are stationary and no camera calibrations are possible. The surveillance videos of interest are from the Stockholm metro. The positions of the cameras are known.

Table 1: Requirement table.

Requirement nr	**Change**	**Requirement description**	**Priority**
Nr 1	Original	Detect people in a video sequence.	1
Nr 2	Original	Assign a unique ID for every person within the scene.	1
Nr 3	Original	Track a detected person in a video sequence.	1
Nr 4	Original	Output movement position in pixel coordinates for every tracked person.	1
Nr 5	Original	Save detected objects in JSON format.	1
Nr 6	Original	Produce a system user manual.	1
Nr 7	Original	Document the time it takes to run the system.	1
Nr 8	Original	Visualize how a person has moved through a video sequence.	2
Nr 9	Original	For a person with a specific object ID in one video sequence, identify persons in other video sequences that are probable to be the same person.	2
Nr 10	Original	Provide the customer with the results of a literature survey of existing algorithms for detection and classification.	2
Nr 11	Original	Provide the customer with the results of a literature survey of existing algorithms for tracking.	2
Nr 12	Original	Detect object attributes such as clothing colors, bags, shoes etc.	3
Nr 13	Original	Pair detected bags with the person carrying them, when they are carried.	3
Nr 14	Original	Find possible person matches in all video sequences based on specific attributes input in text format.	3
Nr 14	Original	For all persons, detect when their faces are shown.	3
Nr 16	Original	For a person with a specific object ID, visualize all images where the face is shown.	3

1.2 Project Overview

The goal of the project was to develop an analysis module for surveillance video cameras with focus on the cameras from Stockholm metro stations. The system should be able to detect, classify and track persons and assign the same ID to the same person throughout the sequence where the person is visible. The result from the tracking should be saved in json format. The system should also be able to find a selected person from one video sequence in other video sequences. Lastly, the system should be able to detect attributes, such as shirt or jacket color etc., to make the result searchable based on attributes.

1.3 Client

The client is Nationellt Forensiskt Centrum (NFC). NFC is the department of the Swedish Police in charge of assisting the Police with image expertise in preliminary crime investigations. The Police as a whole have to manually go through huge amounts of surveillance videos to solve crimes and are in need of a way to reduce this manual work and find possible culprits faster.

1.4 Limitations

The client asked for the analysis module to be developed under MIT license and said that it could be developed on either Windows and Linux and using open source code. A dataset with surveillance videos from the Stockholm metro was available to use for development of the module.

A limitation for the report was the fact that images from the metro surveillance dataset were not allowed in the report due to juridical reasons. Therefore, the results that are presented visually in the report are from another dataset than the one for which the system was developed.

2 Related Work

Below, related works in detection, tracking and attribute segmentation are briefly discussed.

2.1 Tracking

There are several ways to do tracking in a video sequence. Especially suited for sometimes very crowded scenes such as metro surveillance videos is multi object tracking (MOT) using tracking-by-detection. Here all objects first are detected in each frame using an object detector and the detections are then associated between frames using object location and appearance. Two examples of traditional methods that have been revisited in a tracking-by-detection scenario are Multiple Hypothesis Tracking (MHT) [1] and Joint Probabilistic Data Association Filter (JPDAF) [2]. These traditional tracking-by-detection methods are usually very complex and computationally heavy. However, with better detections enabled by recent object detectors, simple tracking models can be used.

Two recent open source trackers are SORT and Deep SORT. SORT stands for Simple Online Real Time Tracker and the tracker uses Kalman filtering and the Hungarian method to handle motion prediction and data association. SORT is fast and simple and have high precision and accuracy but it cannot handle occlusion very well [3].

Deep SORT, is an extension of SORT, which improves the matching procedure and greatly reduces the number of ID switches by using a visual appearance descriptor for each detected object [4].

2.1.1 Multi-video Tracking

Multivideo tracking is the process of finding the same object in different cameras. This can be done by comparing different appearance descriptors for the objects. CNNs such as ResNet[5] can be used to extract these descriptors.

An example is the recently introduced method called AlignedReID [6]. AlignedReID extracts both global and local feature vectors. The local feature vectors are aligned when comparing objects so that the model learns to account for differences in camera angles and object size.

2.2 Detection

Three of the currently best performing open source and publicly available detectors are Yolov3 [7], Mask R-CNN [8] and RetinaNet [9]. All of these object detectors takes an image or a video frame as an input and outputs coordinates of the bounding boxes for each detected object as well as object class for each bounding box and with what confidence the detection is made [7][8][9]. The Mask R-CNN also outputs segmentation masks and it is possible to apply the network to detect instance-specific poses [8].

Comparing the different Average Precision (AP) values using COCOs average mean AP metric [10], currently the most accurate detector is RetinaNet, but the other two detectors

are not far off (see table 2). The network speed is not easily comparable only by reading their respective papers. However, Yolov3 is claimed to be faster than RetinaNet [7].

Table 2: AP values using COCOs mean Average Precision for different object detectors [10].

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Yolov3 [7]	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9
Mask R-CNN [11]	ResNetXt-101-FPN	37.1	60.0	39.4	16.9	39.9	**53.5**
RetinaNet [12]	ResNetXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2

2.3 Human Parsing

Human parsing can be used in for example surveillance, for better person identification and in the fashion and clothing industry. Since there are a lot of different areas with interest in clothes detection and parsing, previous work on the subject has been done to meet the requirements in those fields. In fashion there has been a demand for clothing segmentation in rather high quality fashion still images. The Fashionista dataset was created for this purpose [13], and [14] uses a Condition Random Field model to improve clothes parsing on this dataset.

For video surveillance the parsing has to be done on a sequence of images that can be of lower quality and contain more people. The Crowd Instance-level Human Parsing (CIHP) dataset was created to deal with some of the problems with the previous dataset, and includes more images and more instances of persons in one image [15]. The authors of [15] solves the segmentation problem with what they call a Part Grouping Network (PGN), that is based on fully convolutional networks (FCN), and handles both detection and segmentation of human parts.

3 Theoretical Background

This section describes the theoretical background on detection, tracking, attribute segmentation and re-identification.

3.1 Detection

For the detection, Yolov3 [7] was used. Yolov3 is a detector applying a single neural network to a full image. The network predicts bounding boxes and probabilities for different regions of the image and the bounding boxes are weighted using predicted probabilities. It predicts detections across three different scales and for each bounding box, the class which it might contain is predicted using multi-lable classification [7].

3.2 Tracking

The general aim for a MOT tracker is to associate detections across frames by localizing and identifying all objects of interest. An ideal tracker should provide a constant ID for each of the objects within the scene by keeping track of objects even when the detections are missing or false positive. The MOT problem is challenging since objects can be occluded or temporarily leave the field of view. The appearance of an object can change within the scene because of scale, rotation and illumination variance.

In surveillance tracking both performance and speed are of interest. Real-time tracking requires fast online models. In online tracking only information from current and past frames are presented to the tracker.

3.2.1 SORT

The SORT algorithm keeps track of each object by estimating an object model for every frame. The object model contains current spatial information about object position, scale and bounding box ratio. The object model also contains motion prediction for the next frame that is estimated using Kalman filtering.

The data association determines which detection belongs to which object. Since objects can enter or leave the scene, be occluded or correspond to false detections, the data association problem can be hard to solve. The SORT algorithm solves the problem by calculating the bounding box similarity between objects and detections. This is done by calculating the bounding box IoU distance as

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cup B|} \tag{1}$$

where A is the current bounding box for the previously detected object and B is the current bounding box for the new detection. For the previously detected objects, the current bounding box is estimated using the motion prediction. After calculating the IoU distance, the final assignment problem is solved by using the Hungarian method [3].

3.2.2 SORT with Deep Association Metric

This algorithm will be referred to as *Deep SORT*, and it is an extension of the SORT algorithm described in the previous section. SORT is fast and simple, while it performs very well in terms of precision and accuracy. However, it also delivers a relatively high number of identity switches. This motivates an improvement using descriptors for the visual appearance of the detected objects, which are used when matching detected objects from one frame to another to keep track of identities throughout a video sequence.

The descriptor used in Deep SORT is obtained from a convolutional neural network (CNN) that has been pre-trained on a large re-identification dataset with the purpose to discriminate pedestrians. This means that the network has been trained to produce descriptor vectors that are far apart for detected pedestrians with different identities, and very close apart for images of the same person [4]. In deep metric learning methods, the notion of similarity is included directly in the training objective. The feature vectors that are generated for re-identification of the persons in the scene using Deep SORT are based on cosine similarity [16].

The Kalman filtering handles occlusion, but when an object has been occluded for several frames, the prediction becomes more uncertain. Hence, the probability mass spreads out in the state space and there is a risk that a larger uncertainty is prioritized because of the reduced distance in standard deviations of any detection towards the projected track mean. Deep SORT solves this problem by using a matching cascade that prioritizes more frequently seen objects [4].

As previously stated; the Deep SORT algorithm is an *extension* of the SORT algorithm. Comparing the cosine distance between the feature vectors is a complement to measuring the IoU distance and the distances between Kalman state estimations. Because of this improvement, Deep SORT obtains better accuracy on standard benchmarks [4].

3.3 Object Re-Identification

Re-identification is the process of recognizing a specific object in different images. In the context of this project it means being able to assign the same ID to people in multiple cameras. A paper [6] from 2018 introduces an algorithm called AlignedReID which the authors claim performs person re-identification better than human annotators. AlignedReID uses a CNN to jointly learn global and local features to represent a person image.

A feature map of size $C \times H \times W$ is taken from the last convolutional layer of a CNN, for example ResNet50 [5]. A global feature vector is then created by using global max pooling, i.e. pooling with a $H \times W$ kernel, which gives a C-dimensional global feature vector. Local feature vectors are then created by first horizontally max pooling the original feature map to create H different local feature maps and then convolving each local feature map with a 1×1 kernel to reduce the number of channels from C to c. After this a person image is represented by a C-dimensional global vector and H different c-dimensional local vectors, where each c-dimensional vector represent a row of the person image.

The local distance between two person images is calculated by finding the alignment

of local features which gives the smallest total distance. It is done by first creating the distance matrix D containing elements $d_{i,j}$. The element $d_{i,j}$ is a normalized distance between local feature vector f_i from one image and local feature vector g_j from another image. The normalizing transformation is done as in equation 2 below.

$$d_{i,j} = \frac{e^{||f_i - g_j||_2} - 1}{e^{||f_i - g_j||_2} + 1} \qquad i, j \in 1, 2, .., H \tag{2}$$

The local distance between two images is then calculated as the shortest path from $(1, 1)$ to (H, H) in matrix D. The global distance is calculated as the L2 distance between the global feature vectors of the images. Finally the total distance between two person images is simply the sum of the local and global distances.

The procedure above with both global and local features is used during the training stage. However, during the inference stage only the global features are used to compare similarity between person images. The authors of AlignedReID found that only using the global feature during inference worked almost as well as the combined global and local features. In [6] they speculate that the reason for this is that the structure prior of the person image in the learning stage makes the model learn better global features and that the local matching makes the global feature pay more attention to the person instead of the background.

AlignedReID uses TriHard [17] loss as metric learning loss and hard samples are mined using the global distance only. Mutual learning is used during training, details about the mutual learning used can be found in [6] and [18].

3.4 Human Parsing

Part Grouping network (PGN) was introduced by [15] as a method to solve the multi-person human parsing problem. In order to solve this task, semantic part segmentation and instance-aware edge detection is done. Instead of training two networks to solve these tasks separately, PGN uses shared layers to learn common features, and then branches out to solve the separate tasks [15].

The network used to extract the shared feature maps is ResNet-101. The feature maps are extracted from the three last layers and used in the different branches. Two parallel branches are trained, one to assign every pixel with a semantic part label and one to perform edge detecting. The last branch uses the output from the first two branches to refine both the edge and segmentation predictions [15].

The edges and parts can then be used to to do instance-level human parsing, if the parts are connected to a certain instance of a person as described in [15].

4 Method

Here, the system construction is presented.

4.1 System Overview

The system is split into two modules; one main module for detection, classification, tracking, and PGN segmentation, and one multi-video module for multi-video matching – re-identification in other videos. Figure 1 shows an illustration of the system.

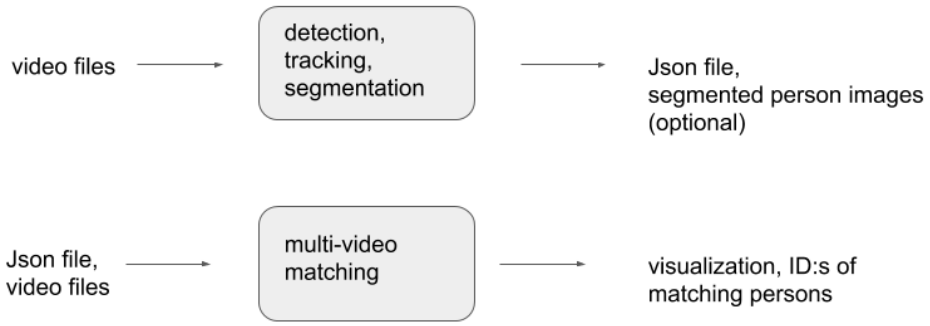


Figure 1: Overview of the system.

4.2 Detection

For our system, the object detection is done using an implementation of Yolov3 in Pytorch [19]. The specific Yolov3 model used in the system is trained to be able to detect 80 different kinds of objects, among these 'persons', 'handbag' and 'backpack' are included. Per default, the final system only forwards detections of object type 'person'. In an effort to speed up the detections, the Tiny Yolov3 model and weights were evaluated.

4.3 Tracking

For the tracking, an open source implementation of the Deep SORT algorithm was used[20], with some adaptations to the system.

This implementation is divided into two steps; feature generation and tracking. A 1x128 feature vector is extracted for every object before the tracking is performed, using the network described in section 3.2.2. This is done for the cut-out bounding box for each detection in each frame, obtained from the detection step described in section 4.2.

After adding feature vectors to all objects the multi object tracking is performed. For each pair of frames, the cosine feature distance, the IoU distance and the Kalman state distance are calculated between all pairs of detections between the frames. Using motion predictions from the Kalman filter the tracking bounding boxes are updated. The cosine feature distance and the IoU distance are used in the matching procedure. The IDs from the previous frame are assigned to the detections in the next frame according to the matches that generate the lowest costs. The IDs will only be matched if they have an IoU distance greater than IoU_{max} and cosine feature distance greater than $cosine_{max}$. Detections with confidences that are too low ($< c_{score}$) are disregarded.

Newly created tracks will not be initiated as an object instantly; a track will remain in the initialization phase until enough evidence have been collected. Tracks that have not been associated with a detection for a time will be deleted and removed from the set of active tracks. The initialization and removal of objects is controlled by the n_{int} and A_{max} parameters.

When the tracking is done, information about each detected object is written to a json file. For every object, the json file will contain the object ID and the following information for every frame where the object is visible: frame number, position and bounding box width and height .

4.4 Person Re-Identification (Multi-video Matching)

The multi-video matching module requires a set of video sequences, on which the tracking had been performed. The task is to find a desired person from one video sequence in the other sequences.

4.4.1 Re-identification using AlignedReID

AlignedReID is the main algorithm used for person re-identification. A pre-trained ResNet50 trained on the Market1501 [21] dataset is used as CNN, the pre-trained model was taken from [22].

AlignedReID needs person images in order to compare people found in different cameras. Information about the bounding boxes from the tracking was used to extract a person image for each frame in which the person could be seen. This means that there is now one person image for each frame in which the person is seen. However only a single person image per person is wanted since it is not computationally feasible to compare multiple bounding boxes per person. This means that a single bounding box has to be extracted for each person, this bounding box should represent the general appearance of the person as close as possible.

The bounding boxes are extracted in the following way: The bounding boxes are first added to a list and sorted according to size so that the smallest bounding box is first.

1. If the person is visible in less than 10 frames, the median sized bounding box is simply taken as the final bounding box representing the person.

2. If the person is visible in more than 10 frames a mean bounding box is calculated. This mean bounding box consists of the set of all the bounding boxes from the

median sized bounding box to the median of the upper half of the list. Then all the boxes in the set are compared to the mean bounding box individually and the bounding box in the set which is most similar to the mean bounding box is taken as the final bounding box representing the person. Figure 2 below shows how the mean bounding box is extracted.

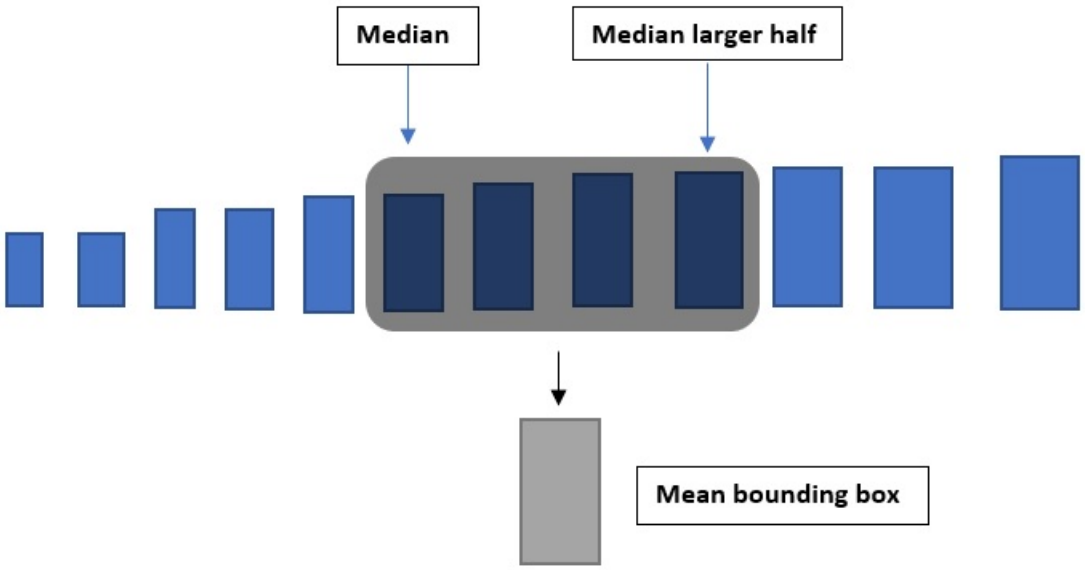


Figure 2: Calculation of mean bounding box

The implementation of AlignedReID which is used works as described in section 3.3. All person images are resized to size (256, 128) before feature maps are extracted using the pre-trained ResNet50. The extracted feature maps have dimensions $2048 \times 8 \times 4$ which means that the global feature vector is a 2048-dimensional vector. A convolution with a 1×1 kernel is done on the feature map to reduce the number of channels from 2048 to 512 before the local feature vectors are extracted. This means that there are in total 8 different 512-dimensional local feature vectors per person image.

Unlike the original AlignedReID paper, both the global and local distances are used in the inference stage when comparing two person images. The reason for this is that no significant difference in computation time could be observed when only global distance was calculated compared to when both global and local distances were calculated.

4.4.2 Re-identification using Features from Deep SORT

The feature vectors that were used in the (frame-by-frame) tracking step were also investigated for use in re-identification between video sequences. For each detected person, the feature vectors throughout the sequence are stored for this purpose.

For a chosen ID in one of the video sequences, the mean feature vector for the person with this ID is calculated. Thereafter, this mean feature vector is compared to the mean feature vector of every other person in other video sequences. The IDs of the ones with the smallest cosine distance to the one belonging to the desired person are returned as candidates.

To obtain a more reasonable set of candidates, there is a possibility to use time information to filter out candidates that appears a lot earlier or later than the desired person. A motivation to this is the data from the client; metro surveillance footage – a person rarely stays within the underground area for longer than a few minutes.

4.5 Human Parsing

The goal with the human parsing was to find attributes of a person (e.g. "person with hat") and also to connect the attribute with one or several colors (e.g "person with red coat"), in order to simplify searching for a specific person. This was done in several steps: the attributes were detected and segmented out, the dominant colors of the pixels belonging to an attribute was calculated and the color code was mapped to a color name in English (e.g [255, 0, 0] in RGB space to the word "red"). Since the input to the system is a video sequence and each person may appear in several frames, the image that is used for the attribute detection for each person also needs to be chosen.

For the segmentation of the attributes, the PGN network, trained with the previously described CIHP dataset was used. Unlike how it was first used by the authors, who did both human part detection and segmentation in one go, information about where each person is seen in the sequence is already available and can be used. For every detected object, as defined after the tracking, one image of a bounding box is chosen for attribute detection and segmentation. This image is chosen as described in section 4.4.1.

This means that for every detected object, the bounding box image is used as input to the PGN network, after doing the same preprocessing as previously described. This outputs label masks, of the different human parts of the object. For every part, or attribute, of the object, the two dominant colors of all the pixels are calculated with k-means clustering. The dominant colors are mapped to a color name by comparing the Euclidean distance to a predefined set of colors in CIELAB color space. The color space chosen because differences in color and luminance are more perceptually uniform [23]. Since the predefined set of colors contains a lot of colors with different names, these names were simplified (e.g. "DodgerBlue3" to "Blue").

Both the color code and the color name is saved to each attribute belonging to each detected object. This is written to a json file along with all other information of the object.

5 Evaluation and Result

In this section the result of the project will be presented. Since no ground truth is provided, the result will be qualitative in forms of figures. All figures are generated from the Laboratory sequences.

5.1 Evaluation Data

When developing the system, actual surveillance videos from the Stockholm metro was used. Due to juridical reasons, videos 1-4 in sequence 2 from CVLab's Laboratory sequences[1] was used to evaluate the final system and the results from these final evaluations are what will mainly be presented in this report. Each of the 4 videos is around 2 minutes long and with a frame rate of 25 frames per second, each frame with a size of 360x288 pixels.

5.2 Runtime Comparison

The measured speed for when the system is run on a Dell XPS 15 9560 laptop with a NVIDIA GeForce GTX 1050 graphics card. The specified time is for running all four of the videos in Laboratory Sequences with six persons. The runtime speed is shown in table 3.

Table 3: Runtime for different parts of the system when processing the four videos in Laboratory Sequences

Module	Time spent	Frames per second
Entire module	**68 min and 8 sec**	**2.89**
-Detection and classification module	32 min and 18 sec	6.09
-Entire tracking module	9 min and 5 sec	21.7
-Extracting features	8 min and 17 sec	23.7
-Deep-sort	40 sec	295
-PGN segmentation	26 min and 45 sec	7.35

5.3 Detection

Figure 3 shows a few examples of the bounding boxes generated by the detection module. There are successes with detection altough there is partial occlusion as in figure 3a and 3c. There are also glitches in the detection as seen in figure 3b.

When processing the client's video dataset from the Stockholm metro, Yolov3 rarely detected any bags. The few actual bag detections are a few backpacks in random single frames when seen straight from behind. It was generally good at detecting persons, except for when they were very far away from the cameras or very close and only partly visible. A few times, weird misinterpretations such as a train or a whole platform labeled

[1] `https://cvlab.epfl.ch/data/data-pom-index-php/`

as a person occurs even at confidence levels as high as 0.85.

(a) Example 1. (b) Example 2. (c) Example 3.

Figure 3: A few examples of detections by Yolov3 in the Laboratory Sequences.

As seen in table 3, it took 32 minutes and 18 seconds to run the detection module of the system, making it the most time consuming process in the complete module. The effort to speed up the system by using Tiny Yolov3 did make the detection module exceptionally fast, but accurate detections decreased to a minimal, as most persons where classified as airplanes.

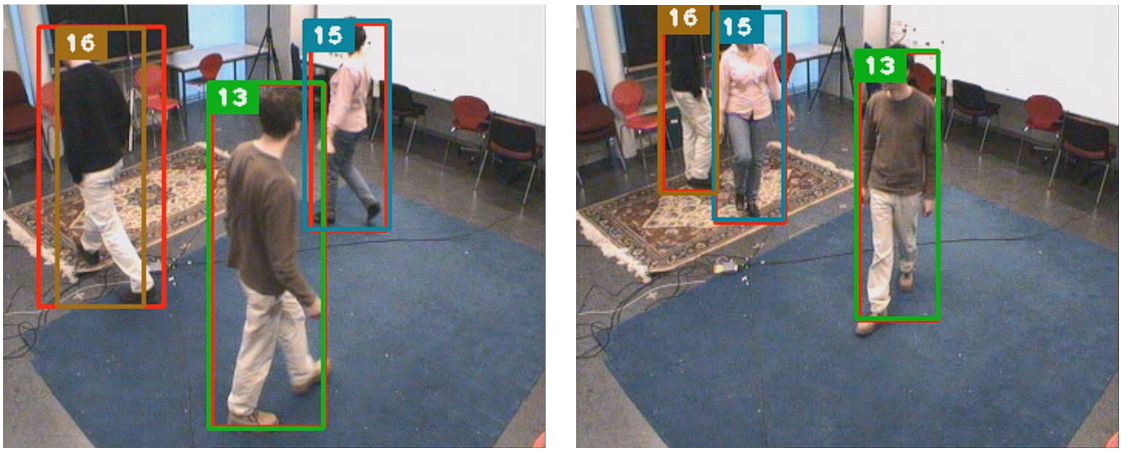
5.4 Tracking

The tracking is performed using the tracking parameters given in table 4.

Table 4: Tracking parameters.

Parameter	**Value**	**Description**
IoU_{max}	0.7	Maximun IoU distance.
$cosine_{max}$	0.2	Gating threshold for cosine distance metric.
c_{score}	0.85	Detection confidence threshold.
n_{int}	3	Max number of missing matches before a track is deleted.
A_{max}	30	Number of frames that a track remain in initialization phase.

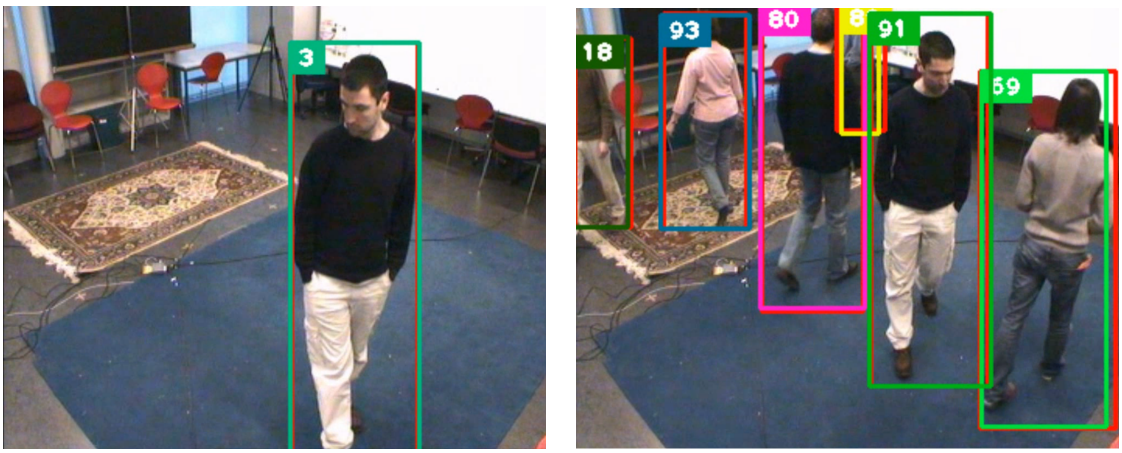
The Deep SORT tracker is fast and effective and the performance is quite good. Results of Deep SORT is shown in figures 4, 5 and 6b. Even if the objects have similar appearance, the tracker manages to keep the objects apart and throughout the sequence, there are only a few ID switches.



(a) Frame 1 (b) Frame 2

Figure 4: Two arbitrarily chosen frames from the result of Deep SORT, whith bounding boxes and ID:s for each person. For each object there are two bounding boxes, the red box are the detection in the current frame and the colored box are the predicted bounding box given by the Kalman filter.

Even if Deep SORT uses both motion and visual appearance, the algorithm struggles when objects are occluded. In the Laboratory Sequences the objects move back and forth in circles and their motion pattern is irregular and complex. When an object passes another object and is fully occluded, the tracker mostly assigns a new ID to the object when it is visible again. Several IDs can therefore be assigned to the same object throught the sequence. An illustration of this is shown in figure 5.

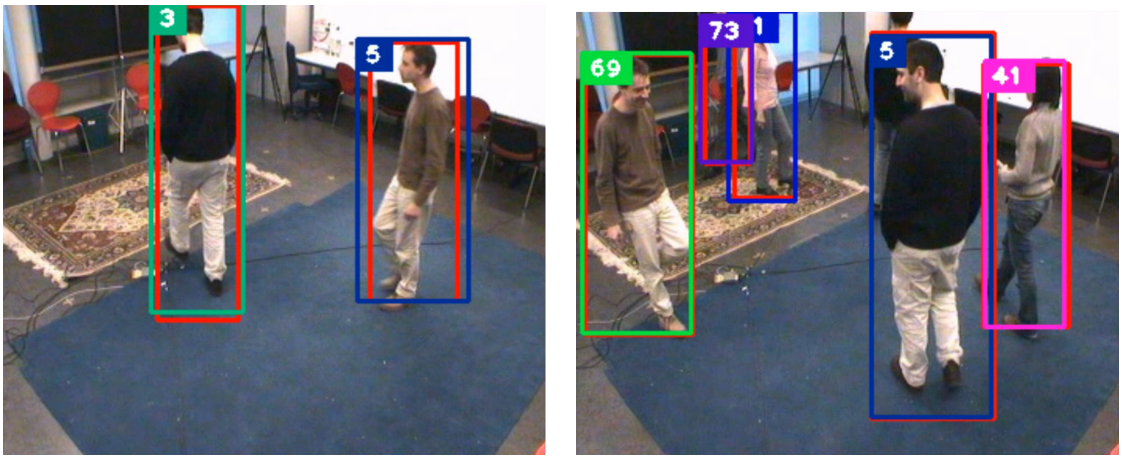


(a) Frame 1 (b) Frame 2

Figure 5: Tracking result from camera 0, the same object is assigned to different IDs through the sequence. For each object there are two bounding boxes, the red box are the detection in the current frame and the colored box are the predicted bounding box given by the Kalman filter.

If $cosine_{max}$ is increased the occlusion is handled better and some objects are tracked through almost the whole sequence with the same ID. Feature vectors before and after occlusion are often dissimilar since part of the object can be occluded. A higher $cosine_{max}$ accepts matches that are more dissimilar and will therefor handle occlusion better. The

drawback is that a higher $cosine_{max}$ also contributes to more ID switches since the differences in visual appearance have less impact.



(a) Frame 1 (b) Frame 2

Figure 6: Tracking result from camera 0 using $cosine_{max}$=0.4, the increase in $cosine_{max}$ causes an id switch for ID=5.

5.5 Person Re-identification

In this section, the results from the re-identification module are presented.

5.5.1 Runtime Comparison

The runtime for two re-identification submodules, one using AlignedReID and another using Deep SORT features, is shown in table 5. The implementation of AlignedReID requires the system to save pictures of all bounding boxes, this has to be done after the whole module with detection and tracking has been run. The user can specify if they want to delete these bounding boxes after AlignedReID has been run. Extracting the bounding boxes, i.e. saving pictures of every bounding box, takes a while which is why timing was done for both when the bounding boxes already exist and when they don't exist. Deep SORT feature vectors are extracted during the tracking part which means that there is no need to save bounding boxes for matching with Deep SORT feature vectors.

Runtime for the multi-video matching will be dependent on the total number of bounding boxes which the original bounding box is compared to. Hence, the number of bounding boxes which the original box has been compared to is included in the result.

Table 5: Time spent on re-identification

Submodule	Time spent	Bounding boxes
AlignedReID with boxes	18 sec	187
AlignedReID without boxes	7 min 56 sec	187
Feature vectors from Deep SORT	6.4 sec	-

5.5.2 Re-identification using AlignedReID

Results from multi-video matching using AlignedReID are shown below. Figure 7 shows the original image and figure 8 shows the matched images in other cameras.

Figure 7: The desired person in camera 0.

(a) Camera 1. (b) Camera 2. (c) Camera 3.

Figure 8: The top 5 matches in each of the other sequences using AlignedReID.

AlignedReID also performed quite well on the metro dataset. It worked especially well when good bounding boxes could be extracted.

5.5.3 Re-identification using Features from Deep SORT

Below, the result from an execution of the Deep SORT feature based re-identification in the lab sequences is shown. Figure 9 shows the top 5 matches for the person in figure 7.

(a) Camera 1. (b) Camera 2. (c) Camera 3.

Figure 9: The top 5 matches in each of the other sequences using features from Deep SORT.

Overall, the matching performed fairly well on the lab dataset for different IDs. On the metro dataset, the matching worked very well for persons with colorful clothes, but worse for some of the other persons.

5.6 Human Parsing

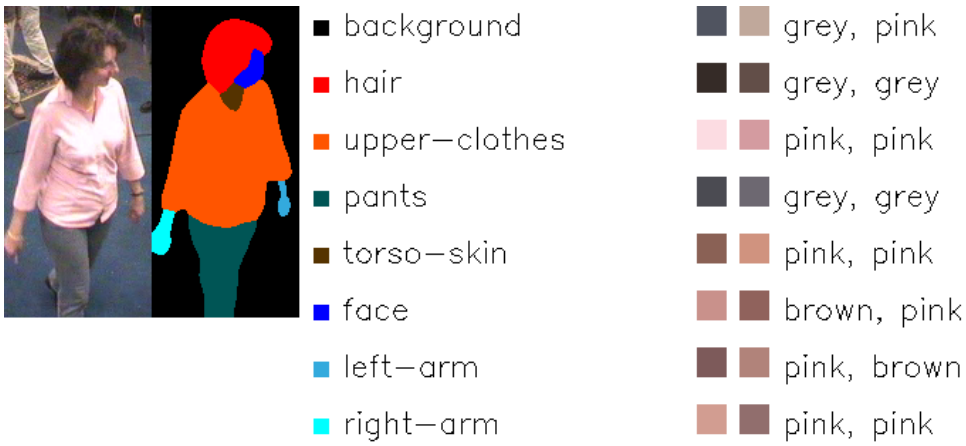


Figure 10: Example of output from human parsing module.

Examples of the result from the human parsing and color mapping can be seen in figures 10 and 11. Figure 10 is an example of when parsing with the Part Grouping Network has worked well. The module has also been able to correctly find the color of the upper-clothes. In figure 11 the parsing hasn't worked as well. The outline of the upper-clothes has been classified as 'upper-clothes' but the rest has been classified as 'coat'.

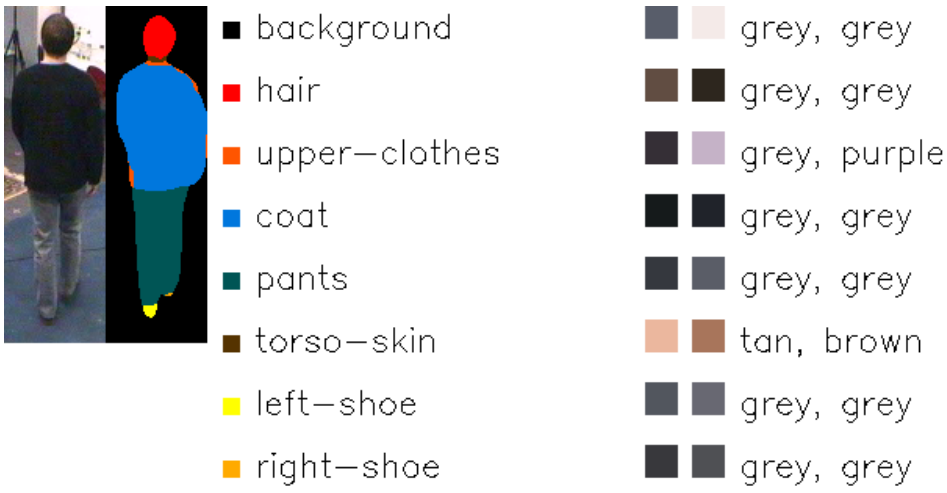


Figure 11: Example of output from human parsing module.

6 Discussion

The complete system seems to perform quite well on the lab dataset. Below, a discussion is given on the different parts of the system.

6.1 Detection

Since Yolov3 is probably the fastest detector with competitive performance – we can draw the conclusion that it has been beneficial to use Yolov3 considering system runtime. However, certain accuracy sacrifices might have been done, which in turn might have had a negative effect on the tracking. I.e. there is a speed versus accuracy trade-off.

To improve the system, a Yolov3 model that has been trained only on persons, bags and other objects of interest, could be used to improve the accuracy of the detection part of the system, as well as generating better bag detections.

6.2 Tracking

The Deep SORT tracker is fast, has few ID switches and is easy to use. The main drawback of the algorithm is the occlusion handling. Even if the tracker uses both motion and appearance for the association, the tracker handles occlusion poorly in the Laboratory sequence. The feature vectors before and after occlusion are often too dissimilar to generate a match.

The confidence threshold that was used was larger than the threshold used in the original Deep SORT algorithm. A larger confidence threshold can potentially improve the performance but the confidence value needs to be changed with care. A higher threshold can improve the performance since detections with low confidence might correspond to erroneously detected objects. However, if the threshold is too large, too many detections will be disregarded and the performance will be even worse since Deep SORT is a relatively simple tracker that demands detections in most frames.

6.3 Person Re-identification

As shown in table 5, one can save a lot of time by re-using the feature vectors from Deep Sort for re-identification. However, it is at the cost of a possibly lower accuracy than when using AlignedReID. This is, of course, only beneficial when the features already have been extracted in the tracking step – a process that also is time consuming, see table 3.

In the lab sequences, the accuracy does not seem to differ so much between the two methods. However, in the metro surveillance dataset the AlignedReID implementation performed better. Therefore, it makes sense to prioritize AlignedReID in this implementation, with the Deep SORT feature comparison as a faster alternative. This may however differ between different datasets.

If the feature representation used in AlignedReID was already used in the tracking instead of the current Deep SORT representation, the multi-video re-identification step

would be a lot faster when using AlignedReID. This would, however, require a lot longer computation time in the tracking step. Even though one feature representation might be better for re-identification in the same sequence while another representation is better for the multi-video case, it would, in some cases, make sense to use the same representation both in the tracking and in the multi-video matching in order to have a faster system.

6.4 Human Parsing

Overall the parsing gives good results, as long as the image is not too small or does not only contain a small part of a person. If there are several persons in the bounding box, the color outputs might be misleading since the attributes belonging to different persons will be added to one single detected object. This is something that could be handled by connecting the parts to an instance of a person by using both the part segmentation and edge output from PGN, in a similar manner as [15] did.

The mapping of color code to a name of a color does not always work well. There might be several reasons for this. First of all, there is a lot of colors in the predefined set of colors, having fewer might yield better results. Secondly the surveillance videos used are often greyish in nature, and a lot of the colors will come back as grey. This could be handled by comparing the hue of the colors instead, but then the problem is the definition of colors like grey, white and black in that color space.

Since PGN does both segmentation and detection, and can group parts to an instance of a person, it could be used earlier in the system and improve the detections, if speed is not an issue. This could also solve the problem of the parsing when the images of the bounding boxes are either too small or only contains a small part of a person, since the parsing would not be dependent on bounding box output from the tracking module. The whole human parsing module could also be used in order to improve tracking, both in one video and in several videos.

6.5 Future Work

One might consider trying out another network, e.g. Mask R-CNN or RetinaNet. This would require a longer runtime, but it would most likely result in better accuracy.

If the processing speed is not a problem, one interesting extension regarding the tracking and re-identification would be to investigate the performance when using the feature representation from AlignedReID also in the tracking step. This would add a significant complexity to the tracking, but it would probably generate more accurate results.

Another improvement that would give a better accuracy, but also require a lot more processing, would be to use the attribute segmentation in the detection step. This could also give better feature vectors in the tracking because the background would not have to be used when extracting features.

By detecting individual clothings, e.g. hats or jackets, this information could also be used as a huge improvement of the re-identification module. By knowing the types and colors of every detected person's clothes, one could build a text-based search module. One

could e.g. search for a person with a blue jacket and a white hat, and the search module would quickly find the matching persons in the sequences. Combined with the comparison of the feature representations of the persons, this could be a powerful re-identification module.

7 Conclusions

This project managed to fulfill most of the specified requirements with the exception for some of the priority 3 requirements. The performance of the system varies depending on which data is used, however the final system has an overall decent performance on the dataset which was provided by the client.

The system runs reasonably fast on a standard modern laptop, but with more powerful hardware, some further improvements could be made as suggested in the discussion. This would most likely give more accurate results.

The tracking-by-detection which was used is heavily dependant on the detections in the detections stage. Yolov3 which was used worked good but it is possible that a better result could be achieved if another detector was used. Better detection and tracking would also lead to better person attribute segmentation and better person re-identification.

References

[1] K. F. Chanho, L. A. Ciptadi, and J. Rehg. "Multiple Hypothesis Tracking Revisited". In: (2018). DOI: 10.1109/WACV.2018.00087.

[2] S. H. Rezatofighi et al. "Joint Probabilistic Data Association Revisited". In: (2015). DOI: 10.1109/ICCV.2015.349.

[3] A. Bewley et al. "Simple online and realtime tracking". In: *2016 IEEE International Conference on Image Processing (ICIP)*. 2016, pp. 3464–3468. DOI: 10.1109/ICIP.2016.7533003.

[4] N. Wojke, A. Bewley, and P. Dietrich. "Simple Online and Realtime Tracking with a Deep Association Metric". In: (2017), pp. 3645–3649. DOI: 10.1109/ICIP.2017.8296962.

[5] K. He et al. "Deep Residual Learning for Image Recognition". In: *arXiv preprint arXiv:1512.03385* (2015).

[6] X. Zhang et al. "Alignedreid: Surpassing human-level performance in person re-identification". In: *arXiv preprint arXiv:1711.08184* (2017).

[7] J. Redmon and A. Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018).

[8] W. Abdulla. *Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow*. https://github.com/matterport/Mask_RCNN. 2017.

[9] *Keras RetinaNet*. https://github.com/fizyr/keras-retinanet.

[10] T. Lin et al. "Microsoft COCO: Common Objects in Context". In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312.

[11] K. He et al. "Mask R-CNN". In: *CoRR* abs/1703.06870 (2017). arXiv: 1703.06870. URL: http://arxiv.org/abs/1703.06870.

[12] T. Lin et al. "Focal Loss for Dense Object Detection". In: *CoRR* abs/1708.02002 (2017). arXiv: 1708.02002. URL: http://arxiv.org/abs/1708.02002.

[13] K. Yamaguchi et al. "Parsing clothing in fashion photographs". In: June 2012, pp. 3570–3577. ISBN: 978-1-4673-1226-4. DOI: 10.1109/CVPR.2012.6248101.

[14] E. Simo-Serra et al. "A High Performance CRF Model for Clothes Parsing". In: *Proceedings of the Asian Conference on Computer Vision (2014)*. 2014.

[15] K. Gong et al. "Instance-level Human Parsing via Part Grouping Network". In: *ArXiv e-prints* (July 2018). arXiv: 1808.00157 [cs.CV].

[16] N. Wojke and A. Bewley. "Deep Cosine Metric Learning for Person Re-identification". In: (2018), pp. 748–756. DOI: 10.1109/WACV.2018.00087.

[17] A. Hermans, L. Beyer, and B. Leibe. "In Defense of the Triplet Loss for Person Re-Identification". In: *arXiv preprint arXiv:1703.07737* (2017).

[18] Y. Zhang et al. "Deep Mutual Learning". In: *CVPR*. 2018.

[19] *pytorch-yolo3*. https://github.com/marvis/pytorch-yolo3.

[20] N. Wojke. *Simple Online Realtime Tracking with a Deep Association Metric (deep-sort)*. https://github.com/nwojke/deep_sort. 2017.

[21] L. Zheng et al. “Scalable Person Re-identification: A Benchmark”. In: *Computer Vision, IEEE International Conference on*. 2015.

[22] H. Luo. *Alignedreid+: Dynamically Matching Local Information for Person Re-Identification*. `https://github.com/michuanhaohao/AlignedReID`. 2018.

[23] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.