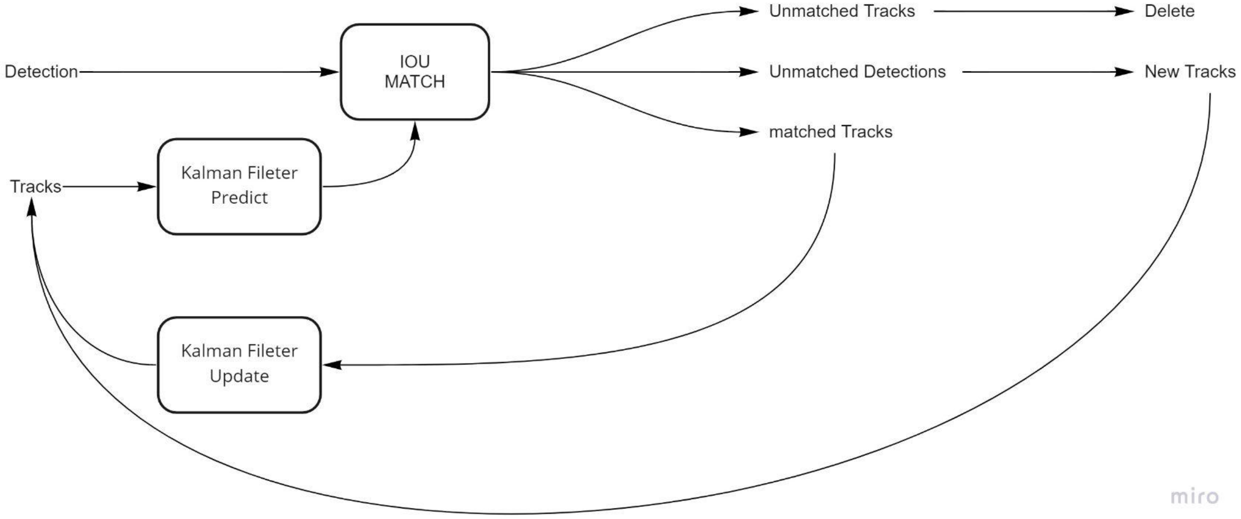
1. Multi-Object Tracking (MOT)

   Multi-Object Tracking can be analyzed through five main procedures as following:

   - Determine the original frames.
   - Obtain object detection bounding by using object detectors such as Faster R-CNN, YOLO, SSD.
   - Choose objects from the corresponding object bounding then select the characteristic.
   - Compute similarity, that is, the matching degree between previous and current frames.
   - Link the data and assign identity to objects.

2. Simple Online and Realtime Tracking (SORT)



SORT is an object detection algorithm based on Faster R-CNN, combined with Kalman filter and Hungarian methods to improve tracking speed of multi objects sharply and approaches the accuracy of SOTA (State of the Art). The algorithm defines the movement status as eight vectors of normal distribution.

The Kalman filter algorithm consists of two procedures, predicting and updating. After the movement of objects, predicting is to forecast object location and speed of object in current frame compared with parameters such as location and speed of previous frame. Updating is designed to renew current status by linear computation of the prediction and observation.

The Hungarian algorithm addresses the assignment problem by computing the similarity to obtain a similarity matrix. The solution of the similarity matrix determines the truly matching objects between previous and current frames.

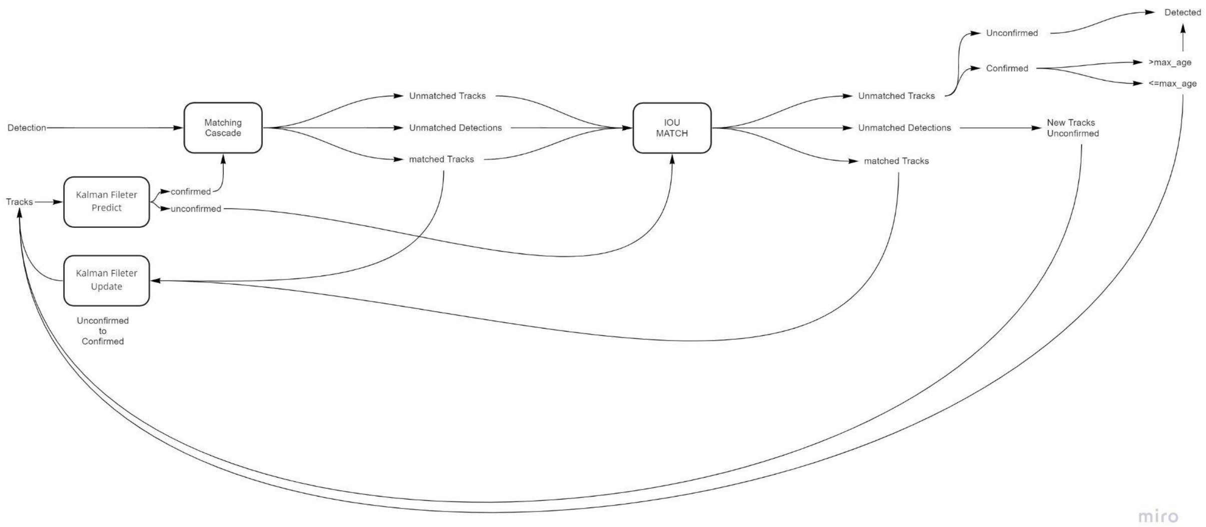
SORT builds the similarity matrix with intersection over union (IOU) to achieve better performance on time.

Detections are object boundings from object detectors. Track is a section of locus. The core of the SORT algorithm shown in the figure above includes matching procedure, predicting and updating procedures of Kalman filter.

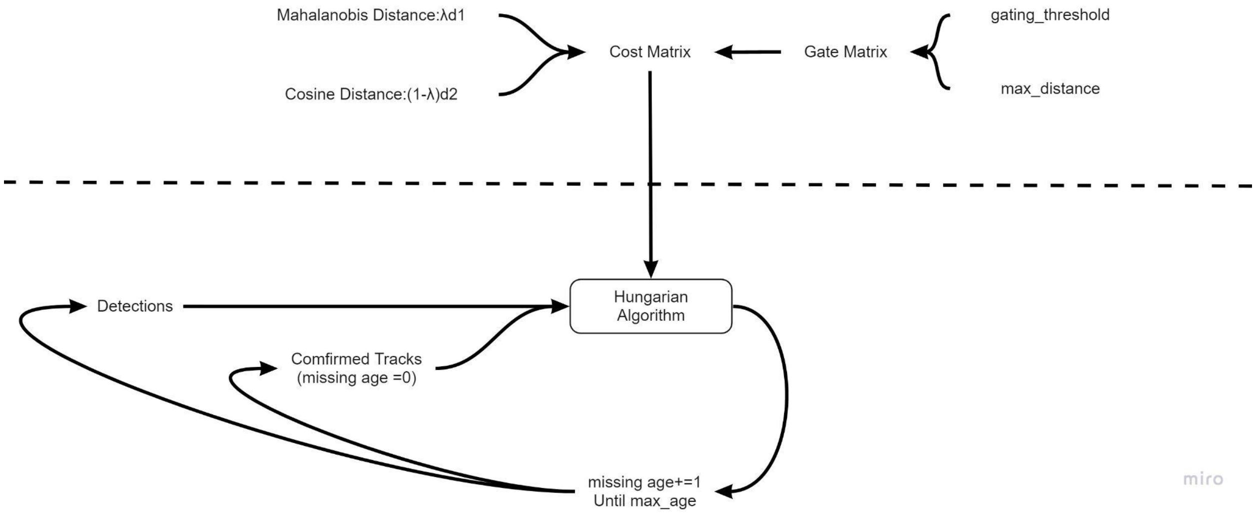
The result from the IOU matching between detections from detector and tracks from Kalman filters is divided into following parts:

- Unmatched Tracks, which means the detection fails to match any tracks. The identity of unmatched track will be discarded if unmatched track status sustains T_{lost}frames.
- Unmatched Detections, which means all tracks fail to match detection and the detection will be assigned a new track.
- Matched Track.

3. Deep SORT
   Deep SORT decreases identity switch times by adding appearance information and ReID domain model to extract characteristics.



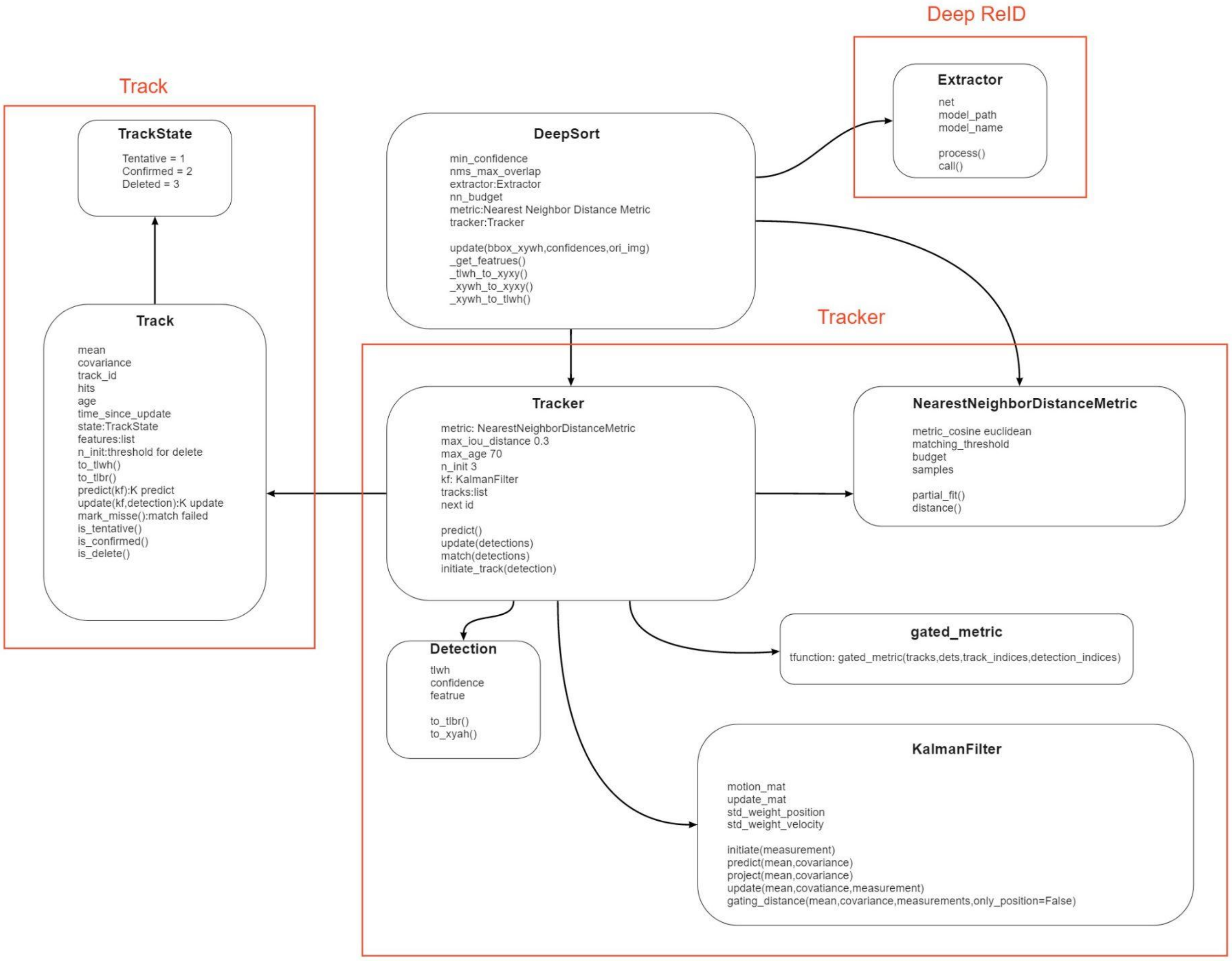
From the chart, we can see that Deep SORT just adds matching cascade and new track confirmation based on SORT, that is, using Hungarian algorithm to match predicted tracks to detections in the frame (cascade matching and IOU matching) between Kalman's predicting and updating process.



The part above the dotted line is to obtain the cost matrix gate matrix. The cost matrix measures similarity using the appearance model and movement model expressed by Cosine distance and Mahalanobis distance. Gate matrix is to limit the extremely large value of the cost matrix.
The part below the dotted line is to realize the matching cascade, which is a cycle matching tracks to detections from the tracks that missing age equals from 0 to 70. The track without loss will be matched priorly, before the track which lost for a while. The cycle can retrieve the obscured objects and decrease the times of ID switch of reappearing objects.
The results of matching can be discussed as below:

1. Matched Tracks, that is, detections match tracks. It is common to the continuously tracked objects that can be matched by either previous or current frame.
2. Unmatched Detections, that is, there is a new object, where detection can not be found in the previous track.
3. Unmatched Tracks, that is, the continuous tracked objects exceed the figure boundaries so that track fails to match any detections.
4. There is a special situation left, which is the occlusion. The track of occluded objects also can not match detection due to temporary disappearance. After reappearance of occluded objects, the ID should not be changed, which decreases ID switch by cascade matching.

4. Deep SORT algorithm



4.1. Deep SORT is the main module calling for other modules. There are three modules invoked by Deep SORT: ReID module, extracting appearance characteristics, Track module, storing track status information as a unit, Tracker module, containing Kalman filter and Hungarian algorithm.

4.2. Classes

4.2.1. Class Detection is designed for storing the detection boundings obtained from object detectors, including upper left corner location, the height and width of boundings, confidence of corresponding bounding box, the embedding obtained from ReID.

4.2.2. Class Track is used to save track information. Mean and covariance represents the location and speed information of the bounding box, and track_id is the identity assigned to the track. There are three status of state as below:

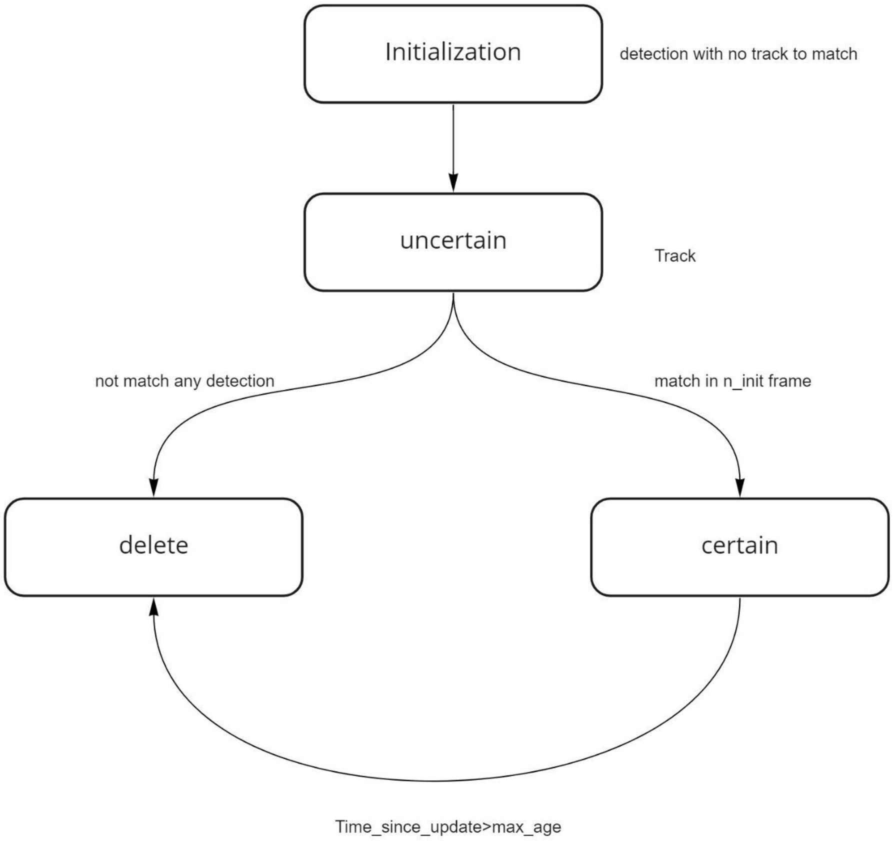
- Tentative: Uncertainty. It is assigned when a track initialization happens. If the object is detected continuously for n_init frames, the status is transferred into confirmed status. Otherwise, if the object under tentative status can not match any detection, the status will be transferred to deleted.

- Confirmed: Certainty. It represents the track is in matched status, but the status will turn to deleted if matching failure continues max age frames.
- Deleted: The track is invalid.

Max_age represents the life span of a track. Time_since_update is a counter which plus one with invoking predict function and becomes zero with invoking update function. If time_since_update exceeds max age, this track will be deleted from the tracker list.

Hits records the calling times in transferring from tentative to confirmed, which plus one with invoking update function. If hits is larger than n_init, that is, there are continuous n_init frames satisfying matching, the status will be transferred from tentative to confirmed.

Feature lists is designed to address characteristics matching in reappearance problems. Budget variable is to limit the list length, just saving the latest budget number of features and discarding the older ones.



4.2.3. Class ReID

ReID net is an independent object detection and tracking module, which is used to extract features from corresponding bounding boxes to obtain embedding with fixed dimensions for similarity computation.

4.2.4. Class NearNeighborDistanceMetric

4.2.5. Class Tracker

Tracker stores all the track information. This class is responsible for first frame's initialization, Kalman filter predicting and updating, cascade matching, IOU matching and so on.

4.3. Gate matrix

Gate matrix is to limit the cost matrix measuring the distance between Kalman filter state distribution and observation. The distance in the cost matrix is the appearance similarity between Track and Detection. It is the solution to avoid the mistake of using a single track to match two similar appearance detection that takes two detection respectively with the track into calculation of Mahalanobis distance with gating_threshold and discarding the detection with larger distance.

4.4. Kalman Filter

The mean and covariance of track are required in Deep SORT. The mean is expressed as a eight-dimension vector $(x, y, a, h, v_x, v_y, v_a, v_h)$. The (x,y) is the center of bounding, a is aspect ratio, h is height, and the last four values are velocity in the corresponding directions. The covariance measures the uncertainty of the object location, expressed as an eight by eight diagonal matrix. The larger the value of the matrix, the more uncertain the object is.

The main procedures of Kalman filter are as follows:

1. Predict the status of the next frame using the status information of the current frame.
2. Obtain the observation, that is, the detection bounding from the object detector.
3. Update the prediction and observation.

Prediction contains two formulas as follows:

1. $x' = Fx$, which F is the status transfer matrix as follows:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & dt & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & dt & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & dt & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & dt \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

2. $P' = FPF^T + Q$, which P is the covariance of the current frame and Q is the motion estimation error of the Kalman filter, representing the uncertainty.

Formulas in updating are as follows:

$$y = z - Hx'$$

, where z is the mean of detection without changing values, the status is [cx,cy,a,h], H is the observation matrix mapping the mean vector of track into detection space. The result is the mean error of detection and track.

$$S = HP'H^T + R$$

, where R is a four by four noise matrix of the object detector. The values on the diagonal are the noise of the center coordinates, width, heights.

$$K = P'H^T S^{-1}$$

, which calculates the Kalman amplifier, that is, the weight measuring estimation error.

$$x = x' + Ky$$

, which is used to update the mean vector.

$$P = (I - KH)P'$$

, which is used to update the covariance matrix.