Spatial Hold Comparative Analysis

Erick Platero (1812570; eeplater@cougarnet.uh.edu)
Department of Computer Science, University of Houston

1. Introduction

The task of Multi-object Tracking (MOT) has undergone many advances: SORT formulated MOT as a data association problem that associates present bounding box detections with previously detected tracks using Intersection over Union (IoU) and Kalman Filter [1]; then, Deep SORT introduced a re-identification embedding to recover tracks that were temporarily occluded [6], and since then, multiple formulations, each with their own unique scheme, have been formed by leveraging IoU overlap and the re-identification embedding to make associations [3, 5–8]. While most tracking methods focus on the tracking-by-detection paradigm, recent efforts have focused in creating a tracking module that performs detection and tracking in a single step [2, 8]. The benefits of such tracking is that they are able to perform tracking in real-time at the expense of lower accuracy.

Improvements in creating new MOT methods often focuses on three areas: (i) the architecture of the model, (ii) the motion algorithm o the model, and (iii) answering what detections will be associated with what tracks. While innovation in each of these areas has seen improvements, there is one area that is often overlooked: the association algorithm itself. In MOT literature, by far the most pervasive association method is the Hungarian algorithm. This algorithms attempts to globally minimize the association cost. This is an intuitive choice when we are associating re-identification embeddings as we are attempting to make associations that minimize a distance. However, when it comes to making associations based on proximity of a current bounding box with a previous bounding box as is done in SORT and DeepSORT when embedding associations do not meet a threshold, then attempting a global minimization may not be optimal. This is because making an association that minimizes local distances at the expense of global minimization may be more suitable to recover short-term occlusions. We have developed an algorithm called Spatial Hold that just does this. In this work, we conduct a comparative analysis on the effects between implementing the Hungarian and Spatial Hold algorithm on in-the-wild crowded scenes. All code associated for this project is found in `https://github.com/eplatero97/SpatialHold/blob/main/README.md`.

2. Spatial Hold

Spatial Hold is a module that tracks objects by leveraging spatial data of bounding box detections. It can be used as a standalone tracking module or as an extension module that can run on top of any tracking algorithm (e.g., [6, 8]). In either case, Spatial Hold solves the association problem between previously seen tracks and current tracks. The selection of tracks that are associated is what differs between the standalone and extension version. As a standalone, Spatial Hold associates the tracks between every two adjacent frames in a video. As an extension, Spatial Hold classifies tracks into four categories: new, lost, found, and dead (see fig. 1). Then, if any tracks exist in the new and lost set, then these tracks are associated. More of the specific details of the algorithm are described in the Supplemental section.

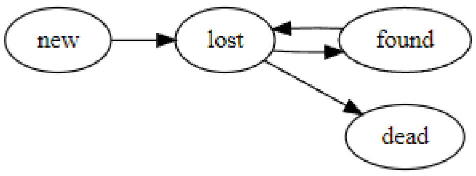

Fig. 1: Spatial Hold Extension track schema.

2.0.1. Formulation

Formally, as a standalone, Spatial Hold generates a bipartite graph every two adjacent frames $G_t = (V, W)$ where $V = V_{t-1} \cup V_t$ are the bipartite vertices of the previous and current frame such that $|V_{t-1}| = |T_{t-1}| = N_{t-1}$ and $|V_t| = |T_t| = N_t$. W is a distance matrix between the centroids of each bounding box in the previous track T_{t-1} and the current track T_t. Then, we formulate our track association as a minimum-weight matching problem between the distances of W. Overall, we go through a three-step filtering process. First, we initiate the matching as the ordered array $P = [v_t^j | argmin_1(W)]$ where $argmin_1$ outputs the vertice in the current frame v_t^j (indicated as a column in W) that minimizes the distance w.r.t. each vertice in the previous frame (indicated as a row in W) (e.g., if $P[i] = v_t^j$, then the associated pair is (v_{t-1}^i, v_t^j)). If any elements in P repeat, then we say that there is a conflict since there exists two or more vertices in the previous frame that have been associated with the same vertice in the current frame. To resolve this, we keep the pair that minimizes the overall distance. We do this by modifying all values of W that contain the row and column values of the associated pair to infinity, except the indices of the associated pair (see eq. (1)). Once calculated, we repeat the first step until no more conflicts arise or when the number of repetitions equals the minimum dimension of W. Second, we calculate the radius r of each vertice in the previous frame by setting it equal to $r_{t-1}^k = \frac{min(w_{t-1}^k, h_{t-1}^k)}{2}$. If the radius associated with each pair is less-than the paired distance, then we filter out the pair. Third, we repeat the first step with the vertices that remain to assert that no two pairs contain the same vertice in the current frame. Lastly, for all the paired vertices that remain, we link the tracklet ID of the previous vertice with the corresponding vertice in the current frame. For all other vertices in the current frame that did not receive a pairing, they are given unique tracklet ID.

$$W_{i,j} = \begin{cases} W_{i,j} & \text{if (i,j) = (k,l)} \\ \infty & \text{else if i = k or j = l} \end{cases}$$

Algorithm 1 Spatial Hold (Tracking)

```
Require: Track indices $T = \{T_0, ..., T_N\}$
  for $t \in 0..N$ do
    if t = 0 then
      $IDs = \{1, ..., |T_0|\}$
      continue
    end if
    $a \leftarrow T_{t-1}$
    $b \leftarrow T_t$
    $W \leftarrow pairwise\_distances(a, b)$
    $rows, cols = min\_cost\_matching(W)$
    $IDs = reassign\_trackIDs(rows, cols, IDs)$
    assert $|IDs| = |T_t|$
```

As an extension, Spatial Hold classifies each tracklet as lost, dead, new, or found. A ”lost” tracklet is defined as any individual track that was in the previous frame $b_{t-1}^k \in T_{t-1}$ but is not on the current frame $b_{t-1}^k \notin T_t$. The set of lost tracklets is defined as $L = \{b_{t-m}^k, ..., b_{t-i}^l\}$ where k,l represent two tracks at times $t-m$ and $t-i$. A ”dead” tracklet is defined as all lost tracklets whose time $t-i$ exceeds the user defined $LIFE \in Z^+$. We represent these tracklets as $D = \{b_{t-i}^l | (t-i) > \text{LIFE}\}$. A ”new” tracklet is the opposite of a lost tracklet as it is a track that was not in the previous frame $b_t \notin T_{t-1}$ but is on the current frame $b_t \in T_t$. Lastly, a ”found” tracklet is any lost tracklet that was associated with a new tracklet. As an extension, Spatial Hold generates a bipartite graph every two adjacent frames that contain lost and new tracklets $|L| > 0$ and $|N| > 0$. The reasoning behind this is that Spatial Hold is re-linking tracks that have received new tracklet IDs but have close spatial proximity with a track in the lost category.

Algorithm 2 Spatial Hold (Extension)

```
Require: Track indices $T = \{T_0, ..., T_N\}$, $LIFE \in Z^+$
  for $t \in 0..N$ do
    if t = 0 then
      $IDs = \{1, ..., |T_0|\}$
      continue
    end if
    $a \leftarrow T_{t-1}$
    $b \leftarrow T_t$
    $W \leftarrow pairwise\_distances(a, b)$
    $rows, cols = min\_cost\_matching(W)$
    $IDs = reassign\_trackIDs(rows, cols, IDs)$
    assert $|IDs| = |T_t|$
```

2.1. *Applications*

Spatial Hold was originally created to track passengers in static environments (dataset is proprietary and thus not made public). When Spatial Hold was implemented to make associations instead of other association algorithms like the Hungarian algorithm, Spatial Hold created significant tracking improvements when using on a pre-trained DeepSORT. The reason for the improvement was that associating bounding boxes based on local minimization was a strong indicator as to whether two bounding boxes belonged to the same track.

While Spatial Hold has proven its benefits on tracking passengers in a static environment, it has not been tested on tracking passengers in a dynamic environment. As such, this work seeks to evaluate the limitations (and possible benefits) of Spatial Hold when used on crowded scenes.

3. Dataset

To compare our association algorithms, we will harness the training data and detections provided by MOT17 [4]. The MOT benchmark is often the standard at which the MOT literature evaluates the performance of its algorithms because it contains multiple crowded scene videos of pedestrians walking in-the-wild. For our comparative analysis, since neither the Hungarian method nor Spatial Hold require trainable parameters, we use the three public model detections provided by MOT17: DPM, FRCNN, SDP. Further, since the testing dataset component of MOT17 does not provide the annotations, we split each of the training videos in half: the first half is the training partition and the latter half is the validation partition. More information on the MOT17 Training dataset is shown in table 1.

4. Comparative Analysis

For our comparative analysis, we analyze performance in two models: SORT and DeepSORT. By default, both of these models leverage the Hungarian algorithm to make associations. DeepSORT is really an extension of SORT where the authors incorporate an association of tracks and bounding boxes through appearance cues. If the re-identification embedding does not meet a user-defined threshold, then DeepSORT effectively becomes SORT as it uses the same logic of associating bounding boxes based on the Kalman Filter.

4.1. *Metrics*

To evaluate models, we calculated the following metrics: Partial Tracking (PT), FragMentation (FM), Recall (Rcll), Precision (Prcn), False Negatives (FN), False Positives (FP), Higher-Order Tracking Accuracy (HOTA), global ID min-cost Recall (IDR), global ID min-cost precision (IDP), mean Average Precision (mAP), Multi-Object Tracking Accuracy (MOTA), ID F1-score (IDF1), ID switches (IDs), ID transfer (IDt), ID migration (IDm), ID ascencion (IDa), Mostly Lost (ML), Mostly Tracked (MT), and Multi-Object Tracking Precision (MOTP). These metrics were calculated using the py-motmetrics framework (`https://github.com/cheind/py-motmetrics`) and not the official matlab devkit and thus, metrics will not be completely aligned with official results and the user must be aware of known errors in the computation by the framework.

4.2. *SORT*

SORT utilizes the Hungarian algorithm to associate bounding boxes with tracks by calculating the pairwise distances between bounding boxes in the current frame and that of previous. To evaluate the difference in performance between the Hungarian algorithm and Spatial Hold, we used a pre-trained SORT algorithm that was trained on the first half of the training videos of MOT17. To assess performance, we separated metrics into ones that need to be maximized (MOTA, MOTP, MT, IDF1, HOTA, Precision, mAP) and ones that need to be minimized (FM, FN, FP, IDa, IDm, IDs, IDt, ML). The results of these metrics are shown on table 2. For each of these metrics, both of the methods had the exact same scores and thus only one value was given for both methods in the table.

Table 1: MOT17 Training Dataset

Name	**FPS**	**Resolution**	**Length**	**Tracks**	**Boxes**
MOT17-13	25	1920x1080	750	110	11642
MOT17-11	30	1920x1080	900	75	9436
MOT17-10	30	1920x1080	654	57	12839
MOT17-09	30	1920x1080	525	26	5325
MOT17-05	14	640x480	837	133	6917
MOT17-04	30	1920x1080	1050	83	47557
MOT17-02	30	1920x1080	600	62	18581

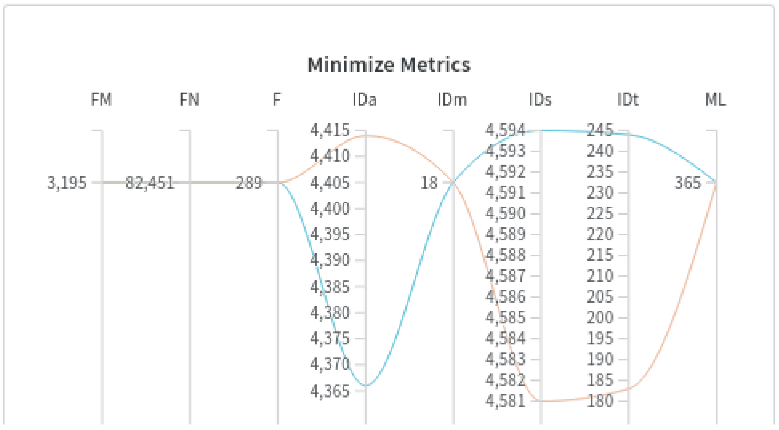

Fig. 2: SORT Validation Performance of metrics that need to be minimized. Blue: SORT with Spatial Hold; Orange: SORT with Hungarian

The MT metric is the only metric that has a score that is neither 0 nor 1. We reason that the rest of the metrics were not computed accurately as a MOTA, MOTP, IDF1, and HOTA vale of 0 would indicate that we had zero False Positives or False Negatives (this is not shown to be true on the minimization metrics). Further, we see both metrics had the same score for the MT metric. On the other hand, the metrics that need to be minimized are shown in fig. 2. The ranges of these computations show that these metrics were actually computed correctly. From this graph, since we see that both methods had the same value for FM, FN, FP, IDm, and ML metrics. This reveals that the performance between these two models is actually very similar. This may be that pehaps performing sequential local minimization does not come out to be as different as performing a global minimization. As for metric IDa, we notice that SORT hungarian contains 48 more ID ascends, which are a subset of ID switches that happen when the new hypothesis ID is a new ID. However, from the overall ID switches, we see that SORT Hungarian contains 13 less total ID switches. This means that although Spatial Hold performs better when it comes to assigning less new IDs of targets that have been previously seen, it performs worse at assigning previously seen targets to previous tracks. This is confirmed by the IDt metric which shows that Spatial Hold is more vulnerable to assigning previous tracks to new targets. This is a surprising result into the difference of these two and it makes sense in a crowded scene: if targets are close to each other, then Spatial Hold will likely contain more ID transfers from one target to another.

4.3. DeepSORT

For DeepSORT, we used a pre-trained DeepSORT model that was trained on the first half of the training videos of MOT17. By default, since DeepSORT uses the Hungarian algorithm to associate re-identification embeddings and bounding boxes, we conducted four experiments: (i) Hungarian algorithm for both tasks, (ii) Hungarian for embedding and Spatial Hold for bounding boxes, (iii) Spatial Hold for embedding and Hungarian for bounding boxes, and (iv) Spatial Hold for both tasks. In reality, the only tests we are really interested are the the first and second since Spatial Hold was originally made to associate bounding boxes in close temporal distances. There is no expectation that this will add any benefits when associating embeddings. For all experiments, we recorded 11 primary metrics: MOTA, MT, IDF1, HOTA, IDR, IDP, FP ,FN, ID Switches, ML, and FM. The metrics are plotted using parallel coordinates as shown in fig. 3. To represent the different combinations in associations, we have the following scheme: track_hung_embed_sh means bounding box (track) was associated with Hungar-

Table 2: MOT17 Training Dataset

Metric	Value
MOTA	0
MOTP	0
MT	208
IDF1	0
HOTA	0
Prcn	1
mAP	1

ian and embedding with Spatial Hold; track_embed_sh means both associations were done with Spatial Hold; track_sh_embed_hung means bounding box was done with Spatial Hold and embedding was done with Hungarian; lastly, track_embed_hung means bounding box and embedding was done with Hungarian. From the top graph, the bottom line observation is an overlay between track_embed_sh and track_hung_embed_sh while the top is an overlay between track_sh_embed_hung and track_embed_hung. This plot reveals that the combinations of using Hungarian for both associations and its counterpart of using Spatial Hold for bounding box and Hungarian for embedding are extremely similar to each other and were able to give the best performance on the metrics that needed to be maximized. The other two follow the same pattern but with the most degraded performance. As for minimization of metrics, we see a slight more complicated story where only the combinations of track_sh_embed_hung and track_embed_hung had an overlay while the rest of the rest of the patterns were different. The combination overlay seems to have the best overall performance, except on the Mostly Lost (ML) metric, which is minimized by the track_hung_embed_sh combination. This metric counts the number of predicted trajectories that matched a ground-truth trajectory for a maximum of 20%. This reveals something surprising: using Spatial Hold on the embedding component actually minimized the amount of ML trajectories. This may imply that performing a series of local associations (as is done in Spatial Hold) may actually be beneficial in tracking ground-truth trajections through occlusions or changes in appearance. As an overall scheme, we notice that track_sh_embed_hung and track_embed_hung had the best performance on all metrics except ML. Then, track_hung_embed_sh had the following best pattern, followed by track_embed_sh.

5. Conclusion

From our analysis, we were able to tell that for SORT, both methods behave more similar than different, while their changes are more prominent when it comes to ID switches. In all, we saw that Spatial Hold performed bettter on IDa while the Hungarian performed better on IIDs and IDt. Given these results, Spatial Hold does not seem to provide any significant improvement on tracking in-the-wild crowded scenes and thus, the status-quo of using the Hungarian algorithm should be maintained. As for DeepSORT, we noticed that whether we use the Hungarian algorithm for embedding and bounding or just for embedding (and Spatial Hold on the detection) does not make any significant effect in performance. As such, both methods could be used as substitutions to each other. As for the other combinations, they all performed worse than the first aforementioned combination, except for the ML metric on the track_hung_embed_sh combination, in which it only had a small difference of one, which is not at all significant). Given these results, we cannot make any conclusions as whether Spatial Hold really improves performance in in-the-wild crowded scenes. However, to be able to talk more accurately whether significant differences exist, a study needs to be conducted in how these methods perform on the test data of MOT17, which would require an official submission. Further, analysis would need to be conducted on the differences in performance between other models so that we are better able to evaluate whether certain differential magnitudes between metrics are significant or not. Lastly, a further analysis that needs to be made is an asymptotic one in which we record the latency performance of each algorithm as complexity (or the dimension of input matrix) increases. Without accounting these limitations, this study concludes that whether we use Spatial Hold to associate detection bounding boxes does not have significant difference on the status-quo of just using the Hungarian algorithm.

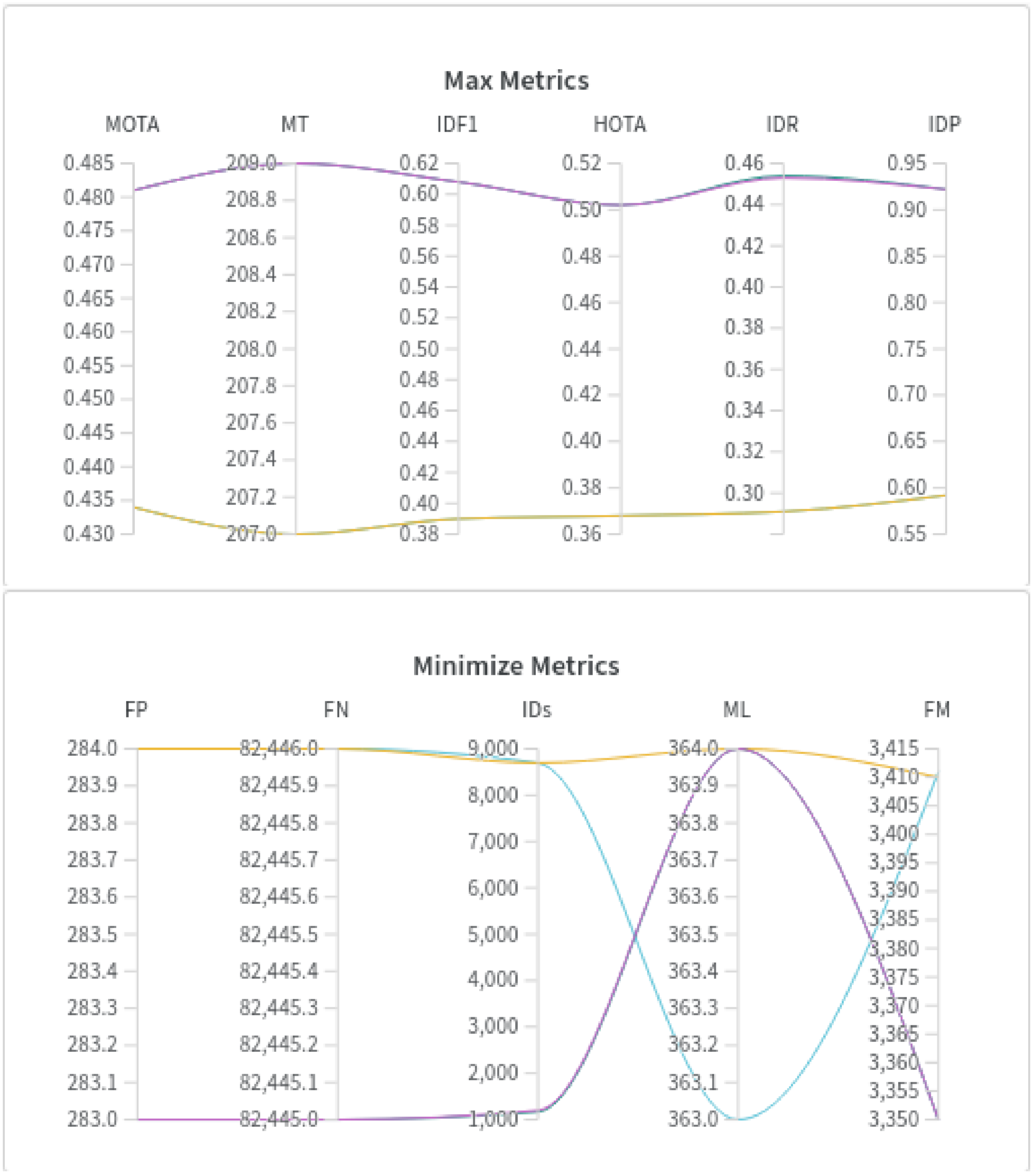

Fig. 3: Validation Performance Separated by metrics that need to be maximized (top) and metrics that need to be minimized (bottom). Green: track_hung_embed_sh; Orange: track_embed_sh; Blue: track_sh_embed_hung; Red: track_embed_hung

References

1. Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.
2. Mohamed Chaabane, Peter Zhang, Ross Beveridge, and Stephen O'Hara. Deft: Detection embeddings for tracking. *arXiv preprint arXiv:2102.02267*, 2021.
3. Chen Long, Ai Haizhou, Zhuang Zijie, and Shang Chong. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*, 2018.
4. ShiJie Sun, Naveed Akhtar, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Deep affinity network for multiple object tracking. *CoRR*, abs/1810.11780, 2018.
5. Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. *The European Conference on Computer Vision (ECCV)*, 2020.
6. Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649. IEEE, 2017.
7. Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. *arXiv preprint arXiv:2110.06864*, 2021.
8. Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision*, 129:3069–3087, 2021.