Module 2: 3D Object Tracking Motion Forecasting

Deng Juan, Yu Hau Chen
University of Toronto

1 Introduction

2 Part A: Object Tracking

2.1 Two-Frame Tracking

Part 1: Cost matrix construction

Part 2: Association

Greedy Association

Question The greedy algorithm does not always guarantee minimal total cost. Please provide an example cost matrix where the greedy algorithm does not lead to optimal total cost.

Answer One example might be:

```
cost_matrix = np.array(
        [
            [1000, 1001, 1002, 1003],
            [1004, 1005, 1006, 1007],
            [1008, 1010, 1011, 1012],
        ]
    )
```

Since in the first iteration, $S_1 = 0$ and $S_2 = 0$ are depleted, we only consider $S_1 = \{1, ...\}$ and $S_2 = \{1, ...\}$ in the later iterations. However, the 3 smallest values in this matrix are in the first rows, thus greedy will not generate a minimal cost.

Question Can you also come up with a scenario with two sets of bounding boxes where the resulting cost matrix does not lead to optimal assignment with the greedy algorithm?

Answer The scenario might be that the min(N, M) smallest values are in the same row or column, which means after the first deletion in the first iteration, we are no longer have access to these values.

Hungarian Association

Part 3: Post-processing.

2.2 Multi-Frame Tracking

2.3 Evaluation

Part 1: Multiple Object Tracking Precision (MOTP)

Part 2: Multiple Object Tracking Accuracy (MOTA)

Part 3: Mostly Tracked (MT), Least Tracked (LT), Partially Tracked (PT)

Part 4: Evaluation and analysis

Question In other words, include the output of the python -m tracking.main evaluate in your report.

Table 1 and 2 shows the result for Hungarian Association. Figure 1 shows the visualization result for Hungarian Association

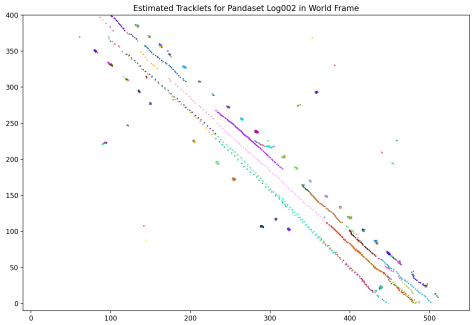
Sequence	mota	motp	mostly_tracked	mostly_lost	partially_tracked
002	0.4757	0.5720	0.6761	0.3036	0.0202
003	0.3520	0.5691	0.7560	0.2439	0.0
004	0.2575	0.4848	0.3410	0.6124	0.0465
005	0.1572	0.6282	0.3426	0.6404	0.0168
017	0.5284	0.6533	0.6716	0.3171	0.0111
019	0.375	0.6163	0.5215	0.4741	0.0043
021	0.2452	0.6150	0.7658	0.2025	0.0316
028	0.4031	0.6923	0.6197	0.3661	0.0140
032	0.2397	0.6631	0.5670	0.4024	0.0304
033	0.2631	0.6308	0.6462	0.3254	0.0283
034	0.3256	0.6794	0.5786	0.3836	0.0377
035	0.2681	0.6093	0.5514	0.4392	0.0093
mean	0.3242	0.6178	0.5865	0.3926	0.0208
median	0.2969	0.6222	0.5991	0.3749	0.0185

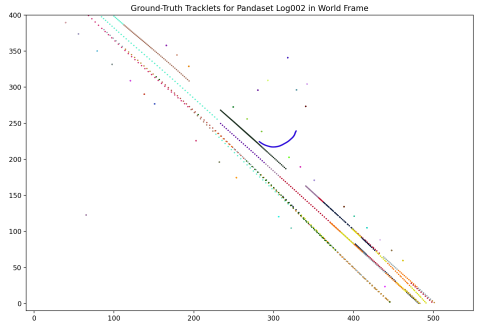
Table 1: Results for Hungarian Association

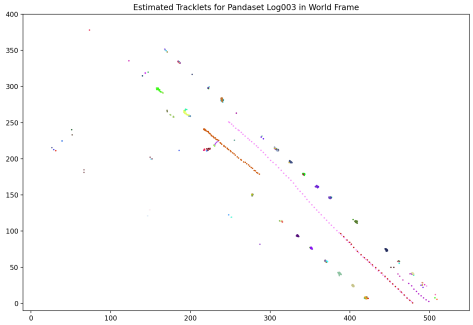
Sequence	missed	false positives	mismatches	matches
002	1553	359	55	2199
003	2142	114	6	1349
004	1216	477	8	1075
005	1530	259	60	664
017	1677	209	31	2388
019	1007	253	85	1145
021	2923	123	53	1183
028	1694	403	4	1826
032	1977	278	6	997
033	2849	331	15	1487
034	2452	204	3	1491
035	1900	474	3	1348

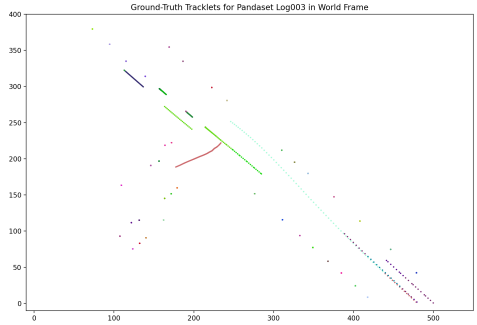
Table 2: number for Hungarian Association

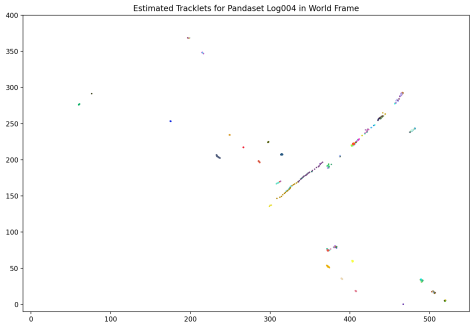
Table 3 and 4 shows the result for Greedy Association. Figure 2 shows the visualization result for Greedy Association

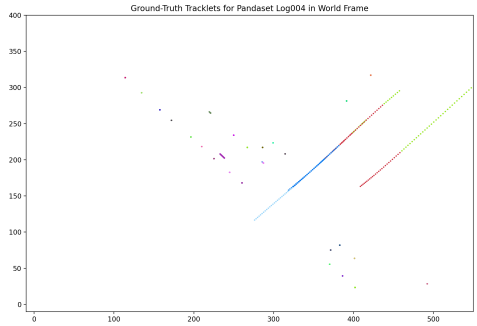

Estimated Tracklets for Pandaset Log002 in World Frame

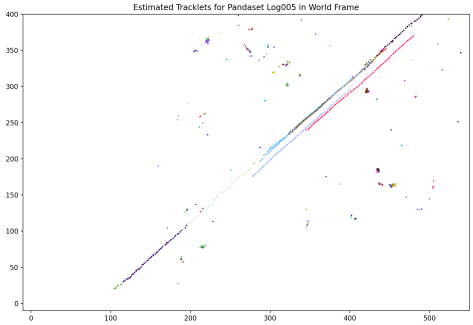

Ground-Truth Tracklets for Pandaset Log002 in World Frame

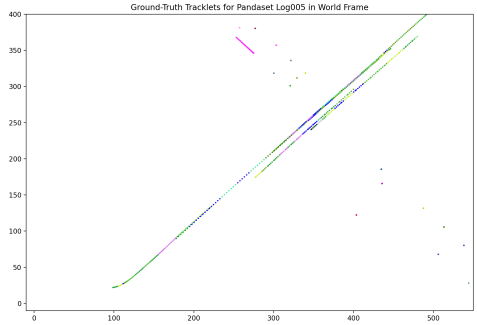

Estimated Tracklets for Pandaset Log003 in World Frame

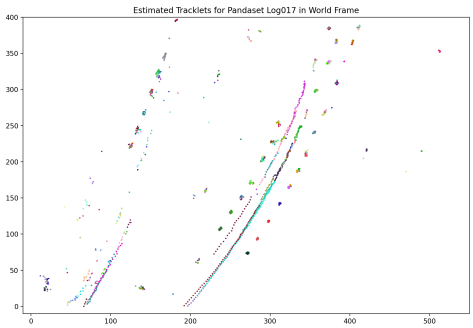

Ground-Truth Tracklets for Pandaset Log003 in World Frame

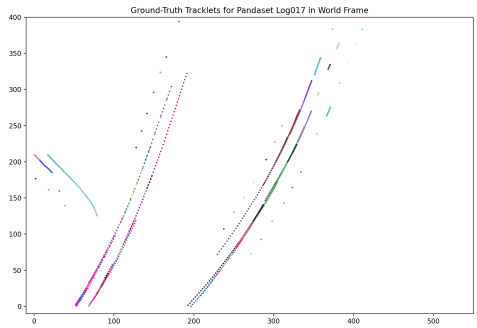

Estimated Tracklets for Pandaset Log004 in World Frame

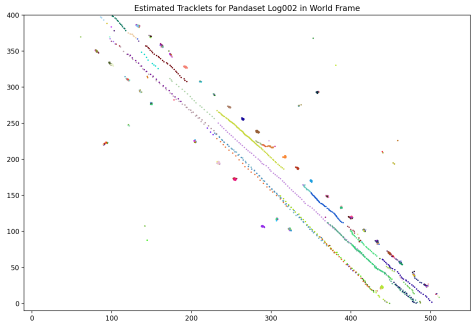

Ground-Truth Tracklets for Pandaset Log004 in World Frame

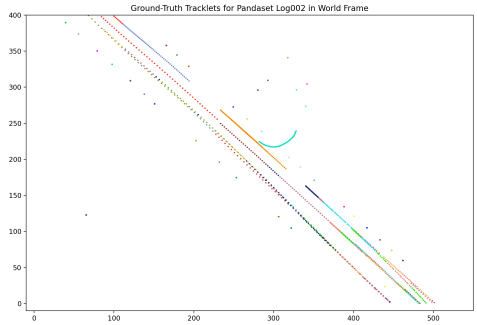

Estimated Tracklets for Pandaset Log005 in World Frame

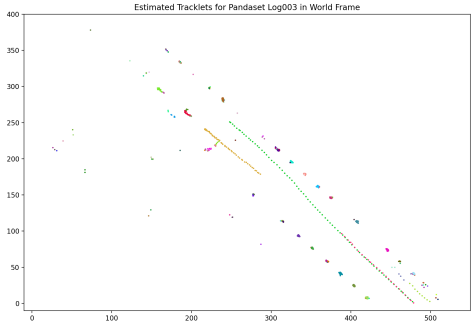

Ground-Truth Tracklets for Pandaset Log005 in World Frame

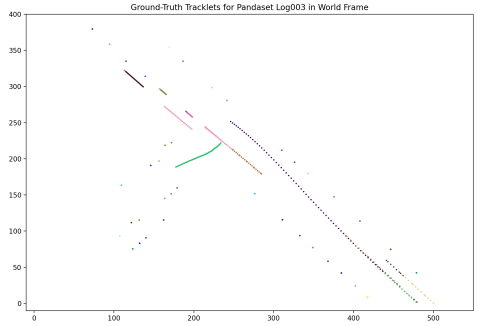

Estimated Tracklets for Pandaset Log017 in World Frame

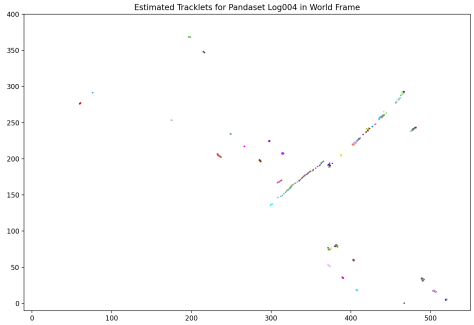

Ground-Truth Tracklets for Pandaset Log017 in World Frame

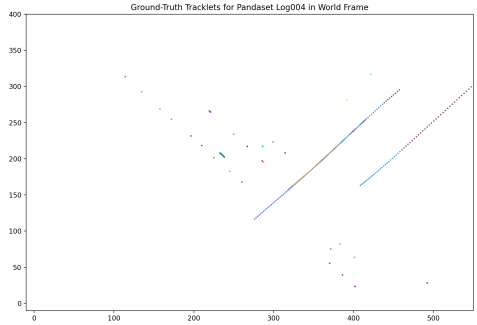

Estimated Tracklets for Pandaset Log002 in World Frame

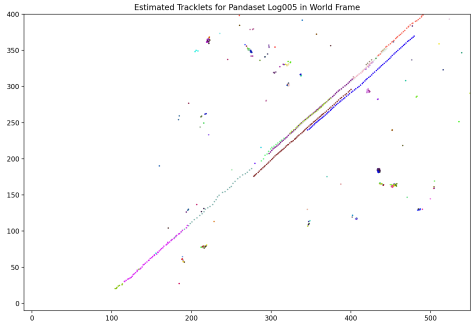

Ground-Truth Tracklets for Pandaset Log002 in World Frame

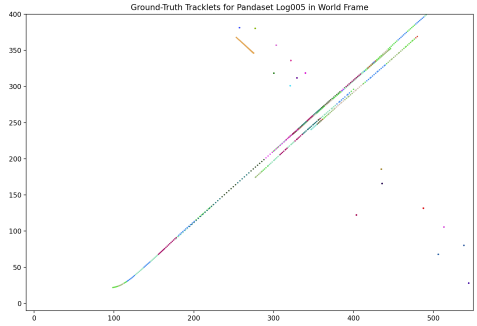

Estimated Tracklets for Pandaset Log003 in World Frame

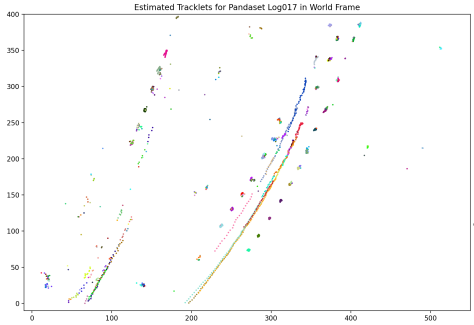

Ground-Truth Tracklets for Pandaset Log003 in World Frame

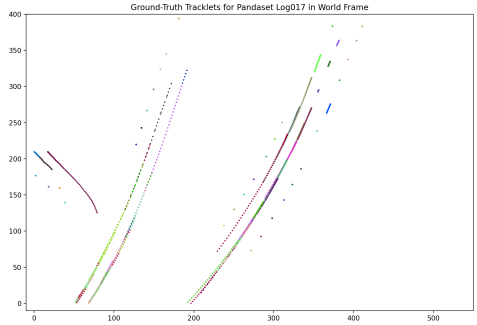

Estimated Tracklets for Pandaset Log004 in World Frame


Ground-Truth Tracklets for Pandaset Log004 in World Frame


Estimated Tracklets for Pandaset Log005 in World Frame


Ground-Truth Tracklets for Pandaset Log005 in World Frame


Estimated Tracklets for Pandaset Log017 in World Frame


Ground-Truth Tracklets for Pandaset Log017 in World Frame

Sequence	mota	motp	mostly_tracked	mostly_lost	partially_tracked
002	0.4757	0.5720	0.6761	0.3036	0.0202
003	0.3520	0.5691	0.7560	0.2439	0.0
004	0.2575	0.4848	0.3410	0.6124	0.0465
005	0.1572	0.6282	0.3426	0.6404	0.0168
017	0.5284	0.6533	0.6716	0.3171	0.0111
019	0.375	0.6163	0.5215	0.4741	0.0043
021	0.2452	0.6150	0.7658	0.2025	0.0316
028	0.4031	0.6923	0.6197	0.3661	0.0140
032	0.2397	0.6631	0.5670	0.4024	0.0304
033	0.2631	0.6308	0.6462	0.3254	0.0283
034	0.3256	0.6794	0.5786	0.3836	0.0377
035	0.2681	0.6093	0.5514	0.4392	0.0093
mean	0.3242	0.6178	0.5865	0.3926	0.0208
median	0.2969	0.6222	0.5991	0.3749	0.0185

Table 3: Results for Greedy Association

Sequence	missed	false positives	mismatches	matches
002	1554	360	53	2198
003	2142	114	6	1349
004	1216	477	8	1075
005	1530	259	60	664
017	1677	209	31	2388
019	1007	253	85	1145
021	2923	123	53	1183
028	1694	403	4	1826
032	1977	278	6	997
033	2849	331	15	1487
034	2452	204	3	1491
035	1900	474	3	1348

Table 4: number for Greedy Association

Question How does the greedy association compare with the Hungarian association on our dataset? The Greedy association and the Hungarian association have the same performance on our dataset, since they are the same value of mean mota, mean motp and median mota, median motp. They also have the same performance in the visualization result.

2.4 Improved Tracker

Part 1: More sophisticated cost functions

2.4.1: Motivation

The basic tracker uses the simple Intersection over Union (IoU) to compute the cost matrix and use this cost matrix to find the associated tracklet to a unique object. However, the simple IoU only measures the overlapping area between two bounding boxes, but it does not consider the influence of rotation or velocity of two bounding boxes. Therefore, we may want to improve our cost function using geometry distance and motion feature.

(a) Geometry distance: We consider the generalized IoU [4], which will maximize the overlapping area of the ground truth bounding box and the detected bounding box by finding the smallest box covering the detected bounding box and the ground truth bounding box.

(a) Motion feature: We consider the distance IoU [7], which will compute the normalized distance between the centroid point of the ground truth box and the detected bounding box and the normalized

distance between the smallest x and largest x over the centroid point of the ground truth box and the detected bounding box.

2.4.2: Techniques

Given bounding box 1 (x1, y1, x2, y2) where x1 < x2 and y1 < y2 and bounding box 2 (x3, y3, x4, y4) where x3 < x4 and y3 < y4.

For generalized IoU, the basic step is:

Step 1: Find the smallest covering box C is given by min(x1, x3), min(y1, y3), max(x2, x4), max(y2, y4)

Step 2: Compute the area Area of C is: (max(x2, x4) - min(x1, x3)) * (max(y2, y4) - min(y1, y3))

Step 3: Compute generalized IoU: g_iou = 1 - iou + abs(Area - Union) / Area

For distance IoU, the basic step is:

Step 1: Find the center of bounding box 1 Center1 and bounding box 2 Center2: Center1 ((x2 + x1) / 2, (y2 + y1) / 2) and Center2 ((x3 + x4) / 2, (y3 + y4) / 2)

Step 2: Compute the normalized distance d^2 between center: d^2 = (Center1x - Center2x)^2 + (Center1y - Center2y)^2

Step 3: Compute the normalized distance c^2 between smallest x and largest x: c^2 = (max(x2, x4) - min(x1, x3))^2 + (max(y2, y4) - min(y1, y3))^2

Step 3: Compute distance IoU: d_iou = 1 - iou + d^2 / c^2

2.4.3: Evaluation

Table 5 and 6 shows the result for Generalized IoU.

Sequence	mota	motp	mostly_tracked	mostly_lost	partially_tracked
002	0.4757	0.5720	0.6761	0.3036	0.0202
003	0.3520	0.5691	0.7560	0.2439	0.0
004	0.2575	0.4848	0.3410	0.6124	0.0465
005	0.1572	0.6282	0.3426	0.6404	0.0168
017	0.5284	0.6533	0.6716	0.3171	0.0111
019	0.375	0.6163	0.5215	0.4741	0.0043
021	0.2452	0.6150	0.7658	0.2025	0.0316
028	0.4031	0.6923	0.6197	0.3661	0.0140
032	0.2397	0.6631	0.5670	0.4024	0.0304
033	0.2631	0.6308	0.6462	0.3254	0.0283
034	0.3256	0.6794	0.5786	0.3836	0.0377
035	0.2681	0.6093	0.5514	0.4392	0.0093
mean	0.3242	0.6178	0.5865	0.3926	0.0208
median	0.2969	0.6222	0.5991	0.3749	0.0185

Table 5: Results for generalized IoU

Table 7 and 8 shows the result for Distance IoU.

Using these results, we find that although distance iou and generalized iou do not improve mota, they improve the number of mostly_tracked and motp, which means the our algorithm generated more relevant data than irrelevant data.

Sequence	missed	false positives	mismatches	matches
002	1554	360	53	2198
003	2142	114	6	1349
004	1216	477	8	1075
005	1530	259	60	664
017	1677	209	31	2388
019	1007	253	85	1145
021	2923	123	53	1183
028	1694	403	4	1826
032	1977	278	6	997
033	2849	331	15	1487
034	2452	204	3	1491
035	1900	474	3	1348

Table 6: number for generalized IoU

Sequence	mota	motp	mostly_tracked	mostly_lost	partially_tracked
002	0.4720	0.5808	0.7003	0.2846	0.0149
003	0.3517	0.5760	0.7580	0.2419	0.0
004	0.2575	0.5065	0.3461	0.6076	0.0461
005	0.1567	0.6348	0.3516	0.6373	0.0109
017	0.5276	0.6605	0.6727	0.3161	0.0110
019	0.3736	0.6235	0.5289	0.4669	0.0041
021	0.2433	0.6336	0.7758	0.1954	0.02873
028	0.4031	0.7037	0.6357	0.3509	0.0132
032	0.2397	0.6789	0.5636	0.4060	0.0303
033	0.2629	0.6447	0.6467	0.3348	0.0183
034	0.3248	0.6894	0.5818	0.3818	0.0363
035	0.2681	0.6213	0.5534	0.4418	0.0046
mean	0.3234	0.6295	0.5929	0.3888	0.0182
median	0.2965	0.6342	0.6087	0.3664	0.0141

Table 7: Results for distance IoU

Sequence	missed	false positives	mismatches	matches
002	1553	359	69	2198
003	2142	114	7	1349
004	1216	477	8	1075
005	1530	259	61	664
017	1677	209	34	2388
019	1007	253	88	1145
021	2923	123	61	1183
028	1694	403	4	1826
032	1977	278	6	997
033	2849	331	16	1487
034	2452	204	6	1491
035	1900	474	3	1348

Table 8: number for distance IoU

2.4.4: Limitations

- Failed to track the rotated bounding box. After the rotation, both the smallest covering box and the distance between the smallest x and the largest x will change, but our current implementation is hard to detect these changes.

- Future path: Complete IoU [6], which will compute the overlapping area and using the central point distance to generate the IoU. Since CIoU uses both geometric distance and motion feature, it will improve precision and recall at the same time.

Part 2: Occlusion handling

2.4.1: Motivation

The basic tracker can partition the given detected bounding boxes into tracklets, where each tracklet is associated to a unique object. However, this basic tracker does not good where our detected bounding boxes heavily occlude and overlap with each other and we cannot determine which bounding box is the best detection for which object. Therefore, we may want to improve our tracker such that it can select the best-fitting detected bounding box out of a set of overlapping boxes.

In part A, we used Intersection over Union (IoU) to compute the ratio of the overlapped area between the detected bounding box and the ground truth bounding box to the total combined area of these two bounding boxes. Since there might be many detection boxes with high overlaps corresponding to the same object, these bounding boxes needed to be grouped and reduced to only one box. Non Maximum Suppression (NMS) [3] is a technique to select the best bounding box out of many overlapping detected boxes.

2.4.2: Techniques

Given a sequence of detected bounding boxes, the basic tracker will construct a cost matrix and an assignment matrix and use Greedy Association and Hungarian Association.

The basic idea behind NMS is that for any two sequences of bounding boxes (bboxes1 and bboxes2), for each bounding box in bboxes1, we will compare the overlap (intersection over union) of this box with other bounding boxes in bboxes2 and find the bounding box with the highest IoU score.

The pseudocode is as following:

Algorithm 1 NMS using two sequences

```
for i = 1, 2, ...len(bboxes1) do
    for j = 1, 2, ...len(bboxes2) do
        Compute IoU between bboxes1[i] and bboxes2[j]
    end for
    highest← bboxes2_with_highest_IoU_for_bboxes[i]
end for
```

For a given sequence of bounding boxes, we will loop over this sequence and all other frames. For each bounding box in this sequence, we will find a list of the bounding boxes with the highest IoU scores from each other frame. Then, we will find the bounding box with the highest IoU score from this list.

The pseudocode is as following:

Algorithm 2 NMS using all frames

```
for i = 1, 2, ...len(frames) do
    for j = 1, 2, ...len(frames) do
        highest[i] = NMS using two sequences(frame[i], frame[j])
    end for
    target← bboxes_with_highest_IoU_in_list_highest
end for
```

2.4.3: Evaluation

Table 9 shows the mota, motp, mostly_tracked, mostly_lost and partially_tracked value of each sequence (total 12 sequences), and mean mota, mean motp, mean mostly_tracked, mean mostly_lost and mean partially_tracked value and median mota, median motp, median mostly_tracked, median mostly_lost and median partially_tracked values of these sequences.

From the previous parts, we know the results of greedy association and hungarian association are similar. Compared to the results of greedy/hungarian, we observe that the improved approaches bring some improvements to the object tracking. The value of mostly_tracked increases while the value of mostly_lost and partially_tracked decreases, NMS has a better performance on selecting the best-fit bounding box from overlapped boxes for each object.

However, NMS has a lower mota and a lower motp. We also notice that we have a negative mota, which can onlyoccur when the number of errors is larger than the number of targets in our implementation. Lower motp means the NMS returns more irrelevant results than relevant ones. To help analyze the change in mota, figure 10 shows the number of missed, false positive, mismatched, matched for each sequence. Using these numbers, we find the number of mismatched increased. Therefore, since our implementation of NMS does consider the influence from the rotation, the generated bounding box might not be the target box but be the rotated box.

Sequence	mota	motp	mostly_tracked	mostly_lost	partially_tracked
002	-0.0509	0.5745	0.8596	0.1403	-2.775e-17
003	-0.0032	0.5696	0.9221	0.0779	2.7756e-17
004	-0.1829	0.4908	0.6927	0.3073	0.0
005	-0.0816	0.6288	0.7193	0.2806	-5.5511e-17
017	-0.0064	0.6568	0.9195	0.0804	-4.1633e-17
019	-0.0441	0.6164	0.8190	0.1810	-5.5511e-17
021	0.0153	0.6278	0.9058	0.0942	-2.7755-17
028	-0.0866	0.6939	0.8192	0.1808	2.7755e-17
032	-0.0612	0.6641	0.7819	0.2180	-2.7755e-17
033	-0.0396	0.6331	0.8190	0.1809	0.0
034	-0.0271	0.6799	0.8796	0.1203	-5.5511e-17
035	-0.1065	0.6105	0.7398	0.2601	-5.5511e-17
mean	-0.0562	0.6205	0.8231	0.1768	-1.5034e-17
median	-0.0475	0.6283	0.8191	0.1808	-2.7755e-17

Table 9: Results for NMS

Sequence	missed	false positives	mismatches	matches
002	1553	359	2031	2199
003	2142	114	1246	1349
004	1216	477	1017	1075
005	1530	259	584	664
017	1677	209	2205	2388
019	1007	253	987	1145
021	2923	123	997	1183
028	1694	403	1728	1826
032	1977	278	901	997
033	2847	329	1332	1489
034	2452	204	1394	1491
035	1900	474	1220	1348

Table 10: number for NMS

2.4.4: Limitations

- Failed to track the rotated bounding box. While finding the bounding box with the highest IoU score, if the only change from timestep t to timestep t+1 is that the bounding box rotated at angle θ, it may have a very large IoU score. To further solve this problem, we may need to consider the heading angle distance.

- Ignored the measure the overlapping relationship between each object. If there are many objects highly overlapping with each other, then our implementation will fail to tell which bounding box with the highest IoU belongs to which object.

- Future path:
Soft NMS [3], which will change the detection scores between the object and the bounding box to a continuous function of overlapping between object and the bounding box.

3 Part B: Motion Forecasting

3.1 Baseline Motion Forecaster

3.1.1 Encoding Past Trajectory Info (Prediction Encoder)

Question The code also contains an option to feed the current and past yaw as input to the model, in addition to x and y. Does adding yaw information help? Why / why not?

Answer Adding yaw information helps. The current and past yaw provides the model information of the heading of the actor, which has an effect on the actor's future trajectory.

3.1.2 Predicting Future Timesteps (Prediction Decoder)

Question Right now our prediction has a similar output space to our detector - it predicts a box location at each future time. Can you think of other output parametrizations which can be used? What are their pros and cons?

Other output parametrizations can be used is instead of only output a box location at each future time, we can output a Gaussian heatmap that showcases probabilistic outputs. The advantage of this is that since the future of each actor's action is uncertain, a Gaussian heatmap explains the uncertainty better. The cons of this is that the predictions would need to have a higher dimension output space, which requires higher complexity of the model.

3.1.3 Implement the loss function in prediction/modules/loss_function.py, in the forward method of PredictionLossFunction and leverage it in the training loop from main.py

3.1.4 Overfitting

Figure 3 showcases the visualization of the resulting motion forecasting after overfitting. We observe that the predictions are able to perfectly matches the actors' actual trajectories.

3.1.5 Analysis

For the purpose of finding the configuration that produces the best outcome on the validation set, the hyper-parameters to vary are learning rates and encoder architecture. Specifically, the learning rates to vary are 1e-5, 1e-4, and 1e-3, and the encoder architecture to vary are a simple linear layer, a deep neural network with linear layers and ReLU, and finally added a probabilistic layer similar to variational autoencoder [2].

Given an input $\chi \in \mathbb{R}^{W \cdot F}$, where $W = 10$ is the number of time step and $F = 3$ is the number of features per time step, and the dimension of the latent vector of neural net activation to be $D_{FEAT} = 128$, the different encoders are described rigorously below:

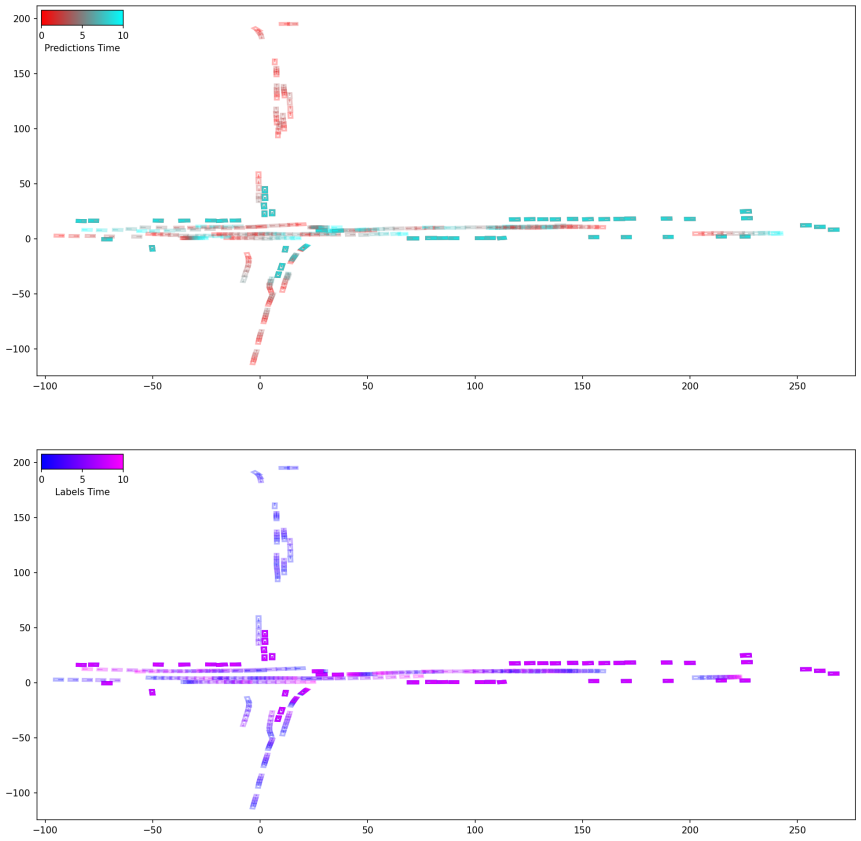


Figure 3: Overfitted Motion Forecast

- Simple Linear Layer: Applies linear transformation $f : \mathbb{R}^{W \cdot F} \rightarrow \mathbb{R}^{D_{FEAT}}$
- Deep Neural Network with Linear Layers and ReLU: A neural network with 3 sequential layers, each sequential contains a linear layer and a ReLU layer, and the hidden units are 32, 64, and $D_{FEAT} = 128$, respectively.
- Add a probabilistic Layer: Expanding from the architecture above, let $x \in \mathbb{R}^{D_{FEAT}}$ be the output, we have $\mu = W_\mu x + b \in \mathbb{R}^{D_{FEAT}}$ and $\sigma = W_\sigma x + b \in \mathbb{R}^{D_{FEAT}}$. The latent vector is obtained by $\mu + \sigma\epsilon$ where $\epsilon \sim \mathbb{N}(0, 1)$

We compare the average displacement error (ADE) and the final displacement error (FDE) of the different configurations, as shown in Table 12.

We observe that the simple linear layer encoder has the highest ADE and FDE for all learning rate, this suggest that the model performs better with a higher complexity encoder. The deep neural network with linear layers and ReLU (Deep NN) performs the best for all learning rate, and the learning rate of 1e-3 has the lowest FDE, while the learning rate of 1e-4 has the lowest ADE. This suggests that while Deep NN with a learning rate of 1e-4 generally has a better predicted trajectories, Deep NN with a learning rate of 1e-3 has a better endpoint of the predicted trajectories.

Encoder	Learning Rate	ADE	FDE
Simple Linear Layer	1e-3	0.6491	1.4011
Deep NN	1e-3	0.5928	1.2489
Deep NN + Probabilistic Layer	1e-3	0.6079	1.3361
Simple Linear Layer	1e-4	0.6144	1.3559
Deep NN	1e-4	0.5754	1.2740
Deep NN + Probabilistic Layer	1e-4	0.6192	1.3621
Simple Linear Layer	1e-5	0.6705	1.4412
Deep NN	1e-5	0.6180	1.3574
Deep NN + Probabilistic Layer	1e-5	0.6205	1.3577

Table 11: Hyperparameter Tuning Result

4 Improved Motion Forecaster - Prababilistic Prediction

4.1 Motivation

The baseline motion forecaster predicts the actor's future trajectory as a point estimate, i.e. the actor's location at time t is generated deterministically. However, the future trajectory of the actors is stochastic, and therefore we want the motion forecaster to be able to generate probabilistic waypoints for future trajectory. As a result, the self-driving vehicle can plan a safe route better.

The relational behaviour forecasting stage of the spatially-aware graph neural network [1] produces socially coherent probabilistic estimates of future trajectories by utilizing the interactions between different actors. Inspired by the model, we model the marginal of each way waypoint to be normally distributed.

4.2 Techniques

The baseline motion forecaster predicts an deterministic output (x, y) for each time point for each actor. The forecaster improves by generating a probability distribution $p(x, y)$ where p follows a multivariate normal distribution.

Originally, the model outputs $(x, y) \in \mathbb{R}^2$ for each future waypoint for each actor. To enable probabilistic future trajectories, the model outputs $\mu \in \mathbb{R}^2$ and $\Sigma \in \mathbb{R}^{2\times 2}$ instead. We will test out the outcome of two approaches: 1. p follows a factorized gaussian distribution, i.e. $\Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}$, 2. The covariance matrix is not diagonal, i.e. $\Sigma = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}$. Lastly, the network architecture was trained with ℓ_1 error between predictions and the targets, which measures the difference of the two locations. To enable the model learning probabilistic future trajectories, a negative log-likelihood (NLL) loss is used instead:

$$\mathcal{L}_{NLL} = \sum_{i=1}^{N}\sum_{t=1}^{T}\big[\frac{1}{2}log|\Sigma_{i,t}| + \frac{1}{2}(x_{i,t} - \mu_{i,t})^T\Sigma_{i,t}^{-1}(x_{i,t} - \mu_{i,t})\big] \tag{1}$$

where $x_{i,t}$ represents actor i's ground truth future position at time t

4.3 Evaluation

Figure 4 shows the motion forecast of the overfitted improved method, specifically the NLL with non diagonal coviarance matrix. We immediately spots that there are a number of flaws of the method. The uncertainty of the predictions does not increase with respect to predictions time, and there are a lot of errors in the trajectories, especially for high predictions time.

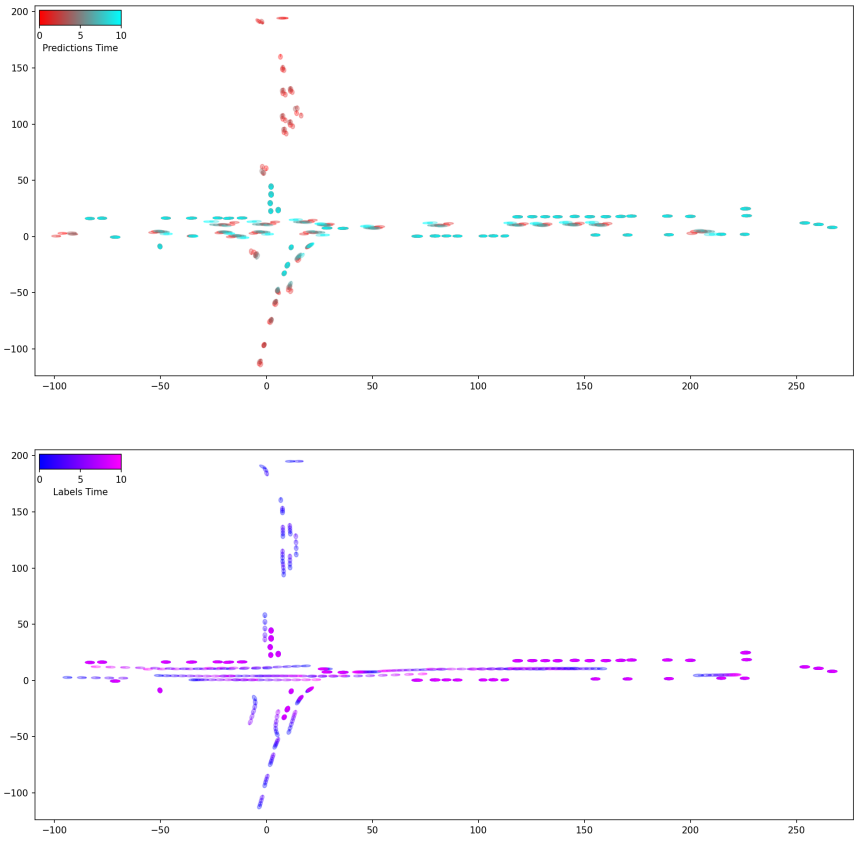


Figure 4: Improved Overfitted Motion Forecast

Table 12 shows that value of ADE and FDE for our original approach (ℓ_1-error) and the improved approaches. We observe that the improved approaches does not bring improvements to the predictions of the location (x, y) of future trajectories. While NLL with non diagonal covariance matrix has better predicted trajectories compare to diagonal matrix, it does not perform well on predicting the endpoints (high FDE).

Moreoever, by comparing the mean error (m) by timestep for each methods in 5, we can see clearly that while both methods using NLL performs well for early timesteps, the predictions are off and unstable for later timesteps.

Loss Function	ADE	FDE
ℓ_1-error	0.5754	1.2740
NLL with Diagonal Covariance Matrix	2.2791	2.7500
NLL with non Diagonal Covariance Matrix	2.5516	5.3007

Table 12: Result of different Forecasters

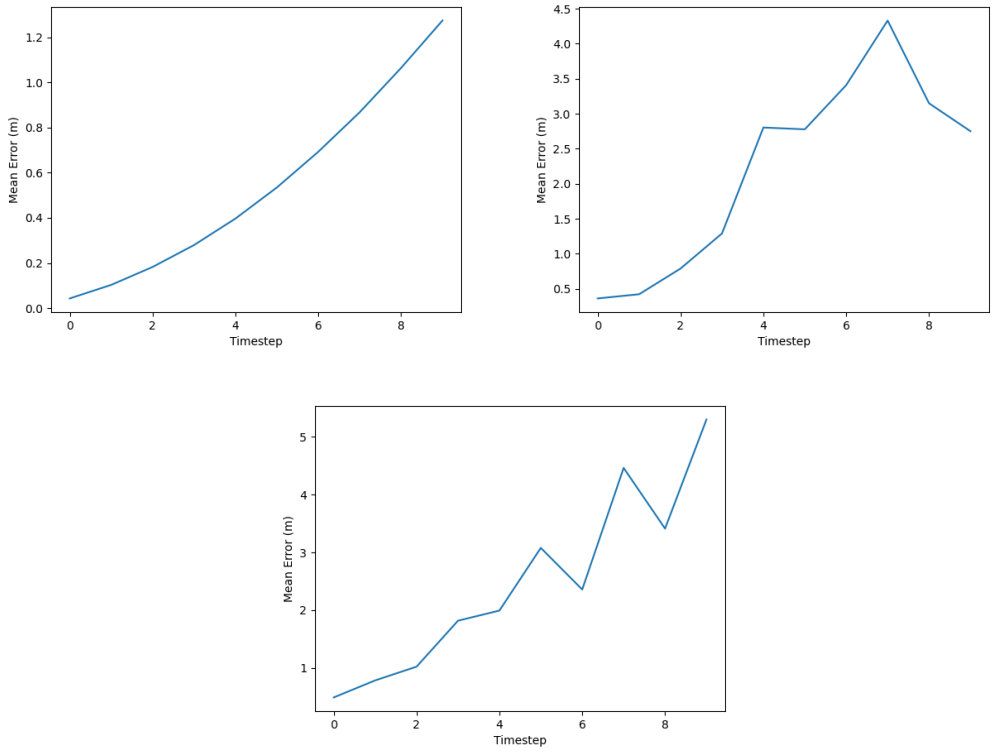


Figure 5: Mean Error (m) by Timestep

4.4 Limitations

There are some limitations that we want to recognize. Our approach does not perform well on predicting trajectories at later timesteps. Moreover, the direction of the predicted trajectories are not represented as a probabilistic variable, but only the location. Some potential paths of future work to improve our approach are first, since the problem is essentially a sequence-to-sequence modelling problem, Recurrent Neural Network architectures such as LSTM and GRU would potentially improves our predictions. Futhermore, the actions of the drivers has an effect of other drivers on the road. Our model does not take into account of interactions between the agents. The graphical neural network [5] can be used to utilize interactions between the agents.

References

[1] Sergio Casas, Cole Gulino, Renjie Liao, and Raquel Urtasun. Spatially-aware graph neural networks for relational behavior forecasting from sensor data, 2019.

[2] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[3] Zekun Luo, Zheng Fang, Sixiao Zheng, Yabiao Wang, and Yanwei Fu. Nms-loss: Learning with non-maximum suppression for crowded pedestrian detection, 2021.

[4] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression, 2019.

[5] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[6] Dongwei Ren Wei Liu Rongguang Ye Qinghua Hu Wangmeng Zuo Zhaohui Zheng, Ping Wang. Enhancing geometric factors in model learning and inference for object detection and instance segmentation, 2020.

[7] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression, 2019.