Airborne Object Tracking Challenge Solution

Dmytro Poplavskiy

Background

Software Engineer at Topcon Positioning systems

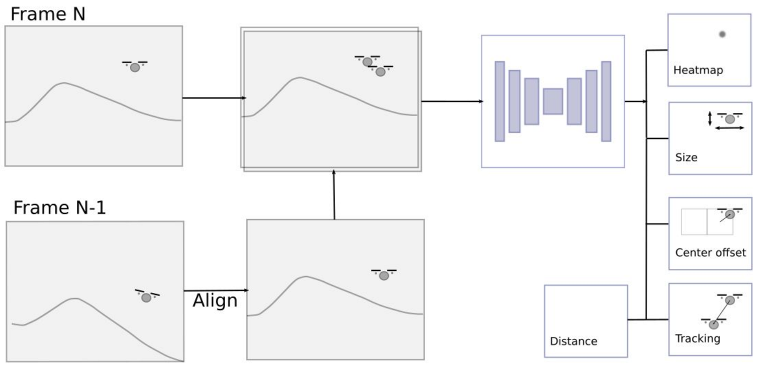
Automation and control of road construction machines.

Interest in Machine Learning

Kaggle Grandmaster: 12 competitions, 11 gold + 1 silver medals

Solution overview

- Align previous frames to the current frame to compensate for drone movement
- Combine the current frame and aligned previous frames as inputs to the segmentation network, predicting at 1/8th of the original resolution the object's center, size, center offset, tracking vectors, and distance.
- Associate the detected objects using the predicted tracking vectors



Frame alignment

The first approach: OpenCV

- goodFeaturesToTrack()
- calcOpticalFlowPyrLK()
- estimateAffinePartial2D()

Frame alignment

The first approach: OpenCV

Drawbacks:

- Slow
- Not always reliable

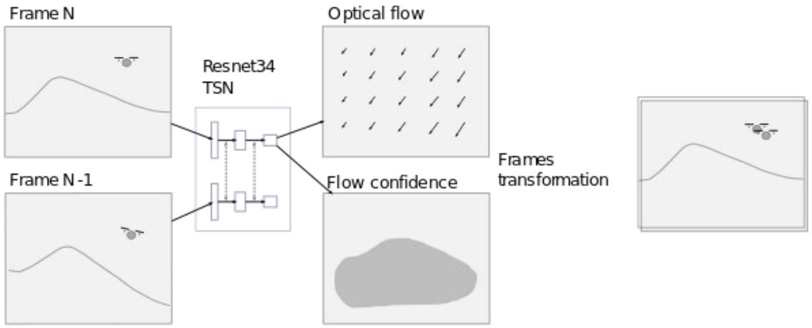
Run at ½ resolution, if failed, fallback to ¼ resolution

Frame alignment model

Inputs: The current and previous frames

Outputs:

- Optical flow at 1/32th resolution grid
- Confidence of the flow prediction



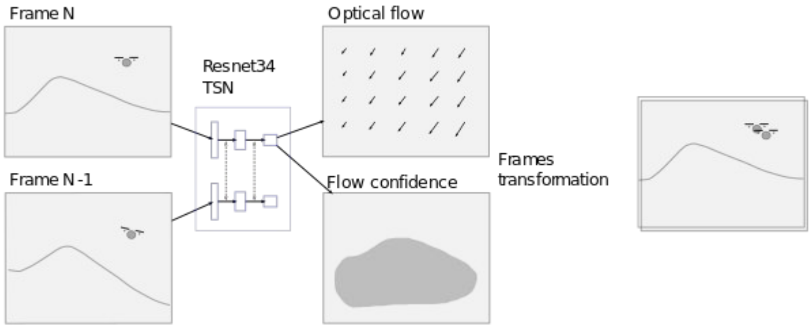
Frame alignment model

The flow confidence:

Normalized exp(3*sigmoid(x))

During training the central 1024x1024 crop with the MSE(error * confidence) loss.

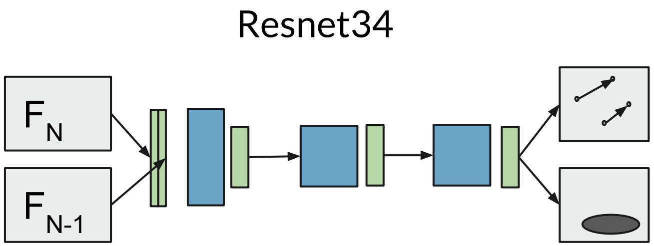
Inference: Use predicted flow to estimate the transformation matrix with the weighted linear regression

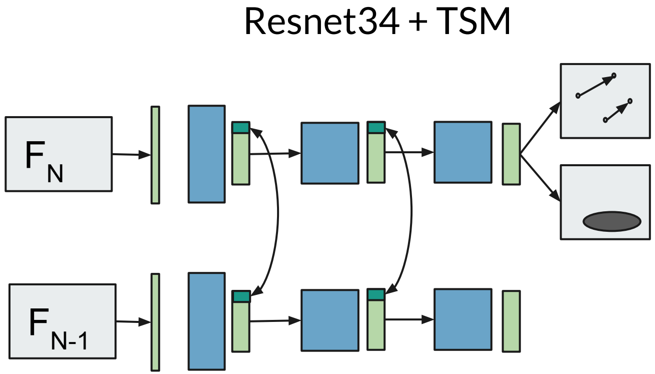


Frame alignment model

Different approaches:

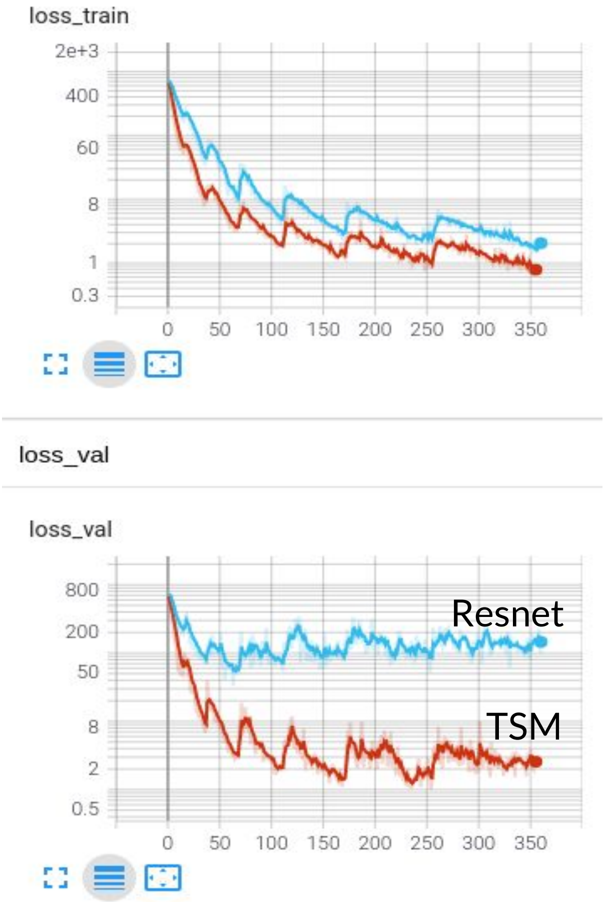
- Combine both frames as different input channels
- Two separate models with shared weights and part of activation values exchanged in residual blocks. Inspired by the “Temporal Shift Module” approach.





Frame alignment model, Resnet34 vs Resnet34+TSM

TSM: 50x improvement of the MSE metric on the validation set

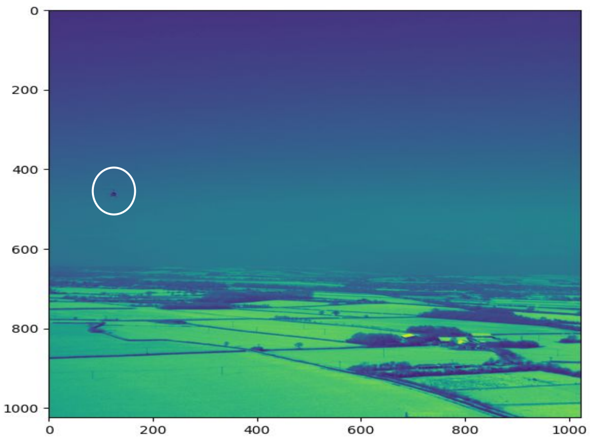


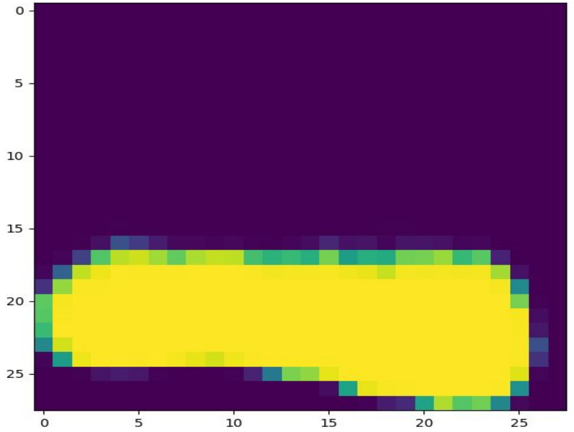
Frame alignment model

Model trained on:

- 75% random transformations
- 25% transformation estimated by OpenCV

Learned to ignore moving objects





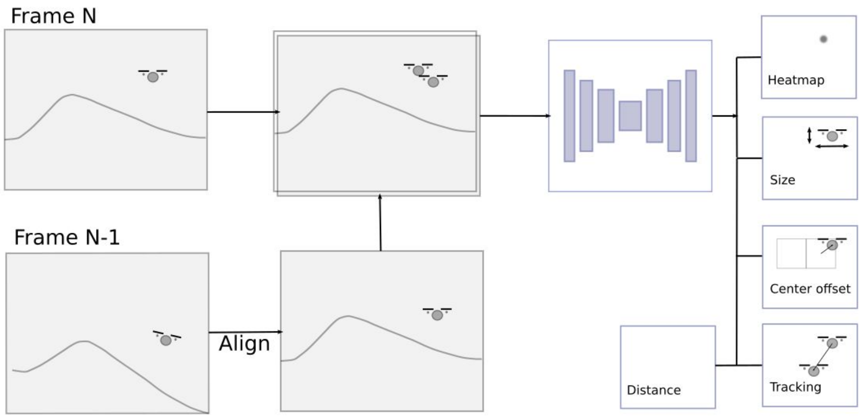
Frame alignment model, potential improvements

- Use Gyro and the camera calibration: much faster
- Exchange activations only in one direction: previous -> current frame
  - 2x latency improvement
  - Information from multiple previous frames
- Correct parallax due to the drone movement using the known elevation map.

Detection and Tracking Model

Simple CenterTrack inspired approach:

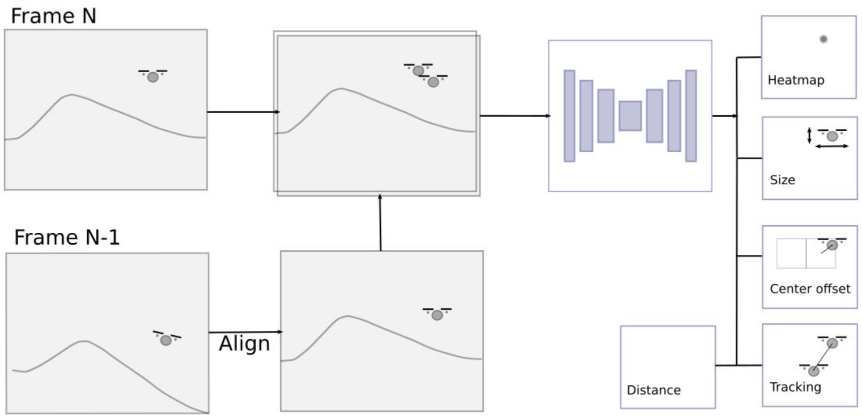
- The current and aligned previous frames as input channels
- Segmentation like model, with ⅛ resolution output: HRNet, EfficientNet+BiFPN, DLA, etc.
- Multiple heads to predict heatmap, size, distance, tracking, center offset
- Unlike CenterTrack, no input of previous heatmap



Detection and Tracking Model

Simple CenterTrack inspired approach:

- Easy to implement, train and debug
- Simple visualisation of model predictions
- Easy to swap backbone to achieve the required latency
- More likely to be useful after the competition



Heatmap encoding

CenterTrack, FairMOT heatmap:

- Gaussian kernel, depending on the object size
- Would be the single pixel for the very small distant planes

Detections $\hat{Y}^{(t)}$

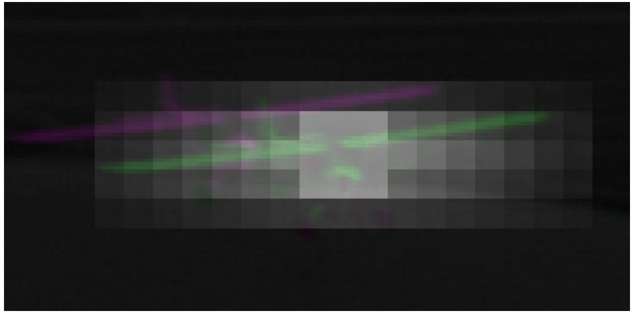
Heatmap encoding

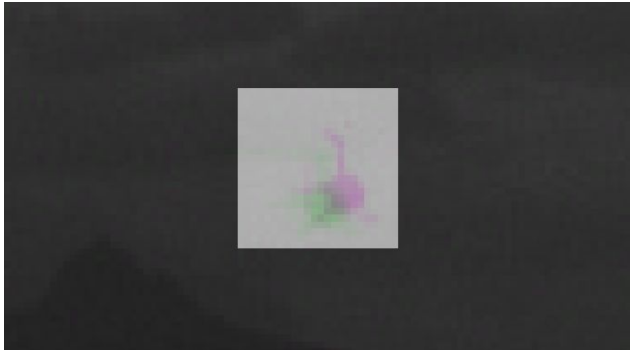
Single pixel heatmap does not work well for prediction

Due to center position uncertainty, the predicted heatmap is distributed between surrounding pixels

One possible approach: sum of the surrounding pixels

Better solution: always use at least 3x3 box, in addition to the gaussian kernel: even with the box position uncertainty, the central pixel would always be predicted with the correct confidence





Heatmap encoding: Which objects to include?

Metric objective details:

True positive: planned airborne objects within 700m
“Don’t care”: planned airborne objects at range over 700m and unplanned objects

Which object to include to the training loss?

- All objects?
  - Doesn’t work well, many predicted false positives.
  - Possible workaround: use predicted distance as a filter, generate the distance pseudo-labels.
- Only planned up to certain distance?
  - Some labels for the model would be wrong, would learn the incorrect negative labels.
- Only planned and large unplanned?

Heatmap encoding: Combined heatmap loss

Suggested loss to mimic the competition objective:

Generate two heatmaps:

1. For planned objects within some distance: $H_{planned}$
2. Heatmap for all objects: H_{all}

$$L = \min_{pixelwise}(FocalLoss(Y, H_{planned}), FocalLoss(Y, H_{all}))$$

So for unplanned objects, positive or negative prediction would not be penalized.

Does it perform better: inconclusive, worked better in the early experiments comparing to predicting only planned objects, worse in the test after the challenge.

Regression heads

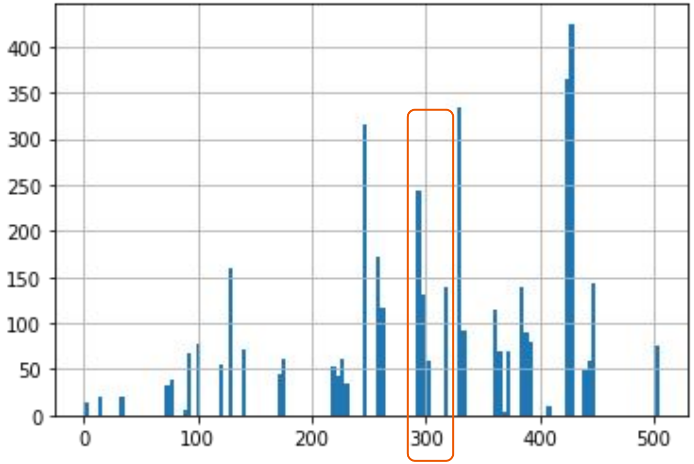
Distance:
Large range, important values around 500-1000m. Model predicts log(distance)
Loss: MSE(log(distance))

Size:
Large range, for IOU metric the relative size is important.
Model predicts log(w), log(h) loss: MSE(log(w))+MSE(log(h))
After the competition, tested instead MAE(w)+MAE(h) as used in CenterTrack, the result was the same

Offsets, Tracking:
Model predicts value, Loss: MSE

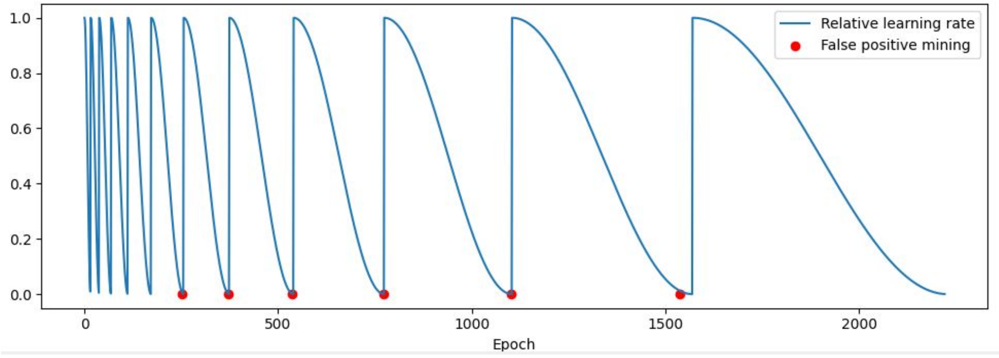
Train/Validation Split

Flights, days since the first flight:



572 Validation Flights

Training Procedure



Madgrad or SGD

CosineAnnealingWarmRestarts LR scheduler

512x512 crops

After each LR cycle, mined false positive samples

Sampling is important:

- 68% random crops, including flights without planned objects
- 25% samples with known false positive area
- 6.25% samples with the planned object within 1km

Ensembling

The objective/metric is very sensitive to false positives

Simple heatmap averaging from multiple models allows to reduce the number of “accidental” model specific false positive predictions.

Possible to ensemble with very little performance overhead:

- Predict with the fast model on the full 2496x2048 resolution
- Predict heavier models on 512x512 crops around the top N predicted objects only, around 20x lower area comparing to the full resolution.

Results from individual models and from ensemble

Model	AFDR at 0.0002 FPPI	AFDR at 0.0005 FPPI
hrnet32	78.2	81.2
hrnet48	81.7	84.7
gernet_m	77.8	79.7
dla60	79.1	81.6
dla34	76.2	78.1
hrnet32 + hrnet48 ensemble	84.3	85.9
hrnet32 + hrnet48 + gernet_m + dla60 ensemble	85.5	87.1

Impact from the negative sample mining and multiple frames

Model	AFDR at 0.0005	AFDR at 0.0002
gernet_m with HNM	79.7	77.8
gernet_m	76.1	71.5
gernet_m with HNM, with the current frame used twice instead of the previous + the current frames	55.7	50.4
gernet_m with HNM, re-trained on the single frame input	57.2	49.0

Very large impact of the previous frames

Detection/track association

Simple track association, based on the predicted tracking vector/offset and the maximum distance threshold (40 pixels used, or scaled relatively to plane size for larger planes).

Extra penalty for low confidence predictions or size mismatch.

Check a few steps of the frames history if match not found in the previous frame.

Simple post-processing to retort track of at least 8 frames size, the same as the baseline solution.

Competition Results

AIRBORNE DETECTION AND TRACKING BENCHMARK
(FINAL LEADERBOARD: UNRATED)

Δ	Submission ID	Participants	EDR	HFAR	
•	153567	dmytro_poplavskiy	0.95146	0.3045685279	View
•	153579	dmytro_poplavskiy	0.91262	0.2284263959	View
•	153739	Colab-BUAA	0.84466	0.5329949239	View
•	153438	Colab-BUAA	0.83495	0.4568527919	View
•	153155	aiforward	0.82524	0.6852791878	View
•	148512	yauhen_babakhin	0.74757	1.065989848	View
•	149835	yauhen_babakhin	0.73786	1.065989848	View
		Baseline SiamMot: Siamese Multi-Object Tracking *Fork to make your submission*	0.72816	0.6091370558	View

FRAME-LEVEL AIRBORNE OBJECT DETECTION BENCHMARK
(FINAL LEADERBOARD: UNRATED)

Δ	Submission ID	Participants	AFDR	FPPI	
•	153611	dmytro_poplav...	0.890504	0.000079	View
•	153579	dmytro_poplavskiy	0.851511	0.000040	View
•	153739	Colab-BUAA	0.807986	0.000929	View
•	153438	Colab-BUAA	0.803381	0.000367	View
•	153699	yauhen_babakhin	0.735252	0.000919	View
•	150067	yauhen_babakhin	0.732158	0.000853	View
•	153577	aiforward	0.687554	0.001091	View
		Baseline SiamMot: Siamese Multi-Object Tracking *Fork to make your submission*	0.678417	0.000526	View

Thank You!