Localising objects with drones:

A case study on the localisation of fisher boats in restricted areas

P5 Presentation

Lisa Geers

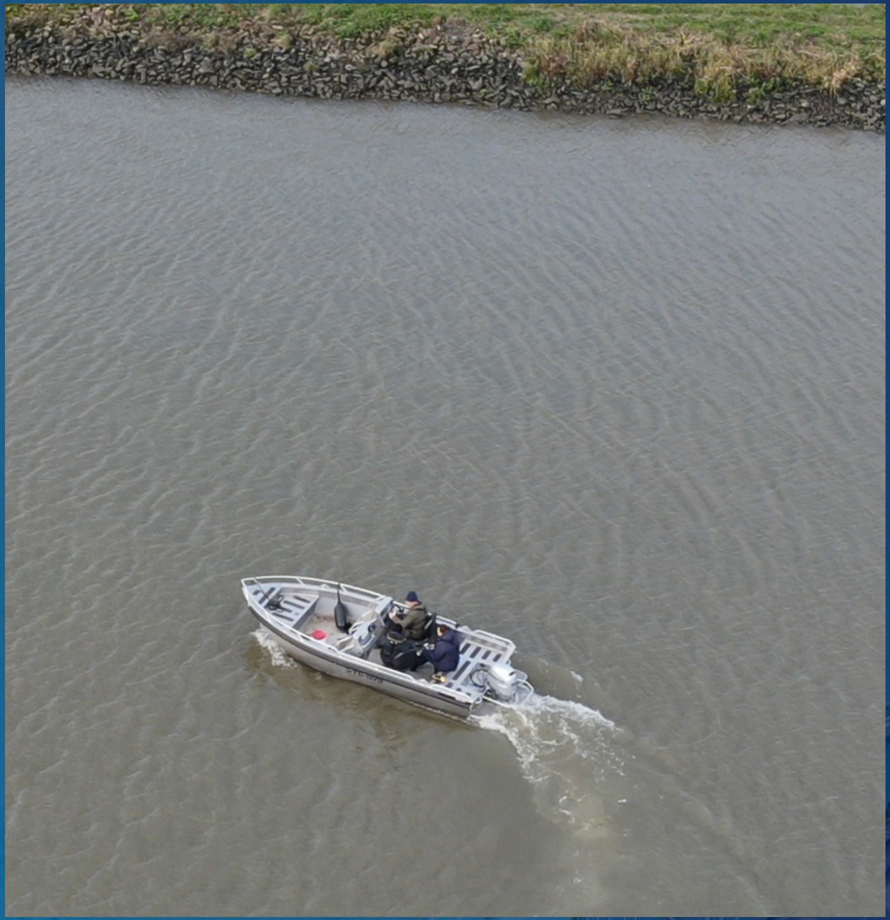
5351421

January 17, 2023

Table of contents

Introduction inspections

- Monitoring in-person time consuming
- Develop integrated method for more efficiency



Inspection boat

Why drones?

- High resolution imagery
- Agile data collection
- Unmanned inspections

drone inspection

Research goals

- Integrate different components into prototype
- To what extent can drones be used to localise objects in real time?
  - Deep learning
  - Positioning
  - Real time

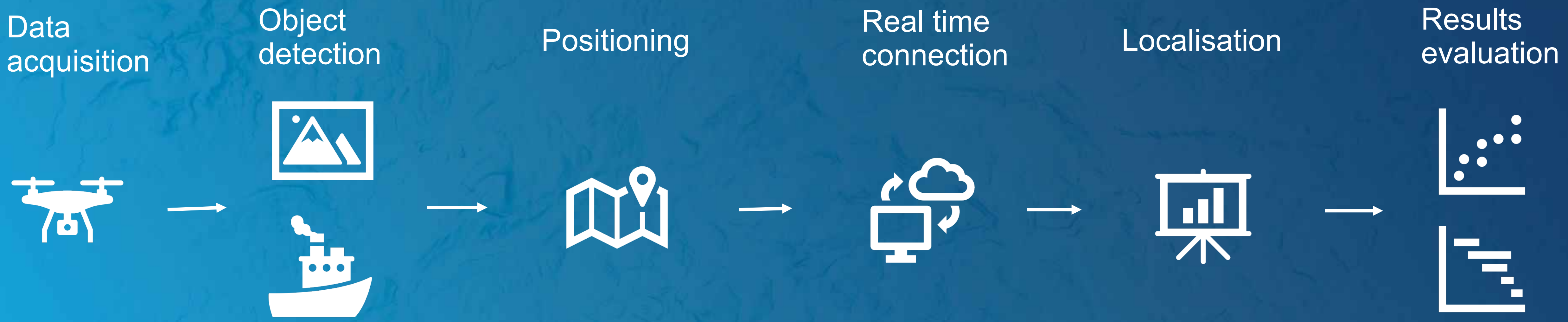
Theory & Related work

- Object detection widely researched
  - AI
  - Deep learning models
  - Object detection

- Localisation vs Positioning
  - Positioning: numerical coordinates
  - Localisation: coordinate context

- Aim is how to integrate into one prototype

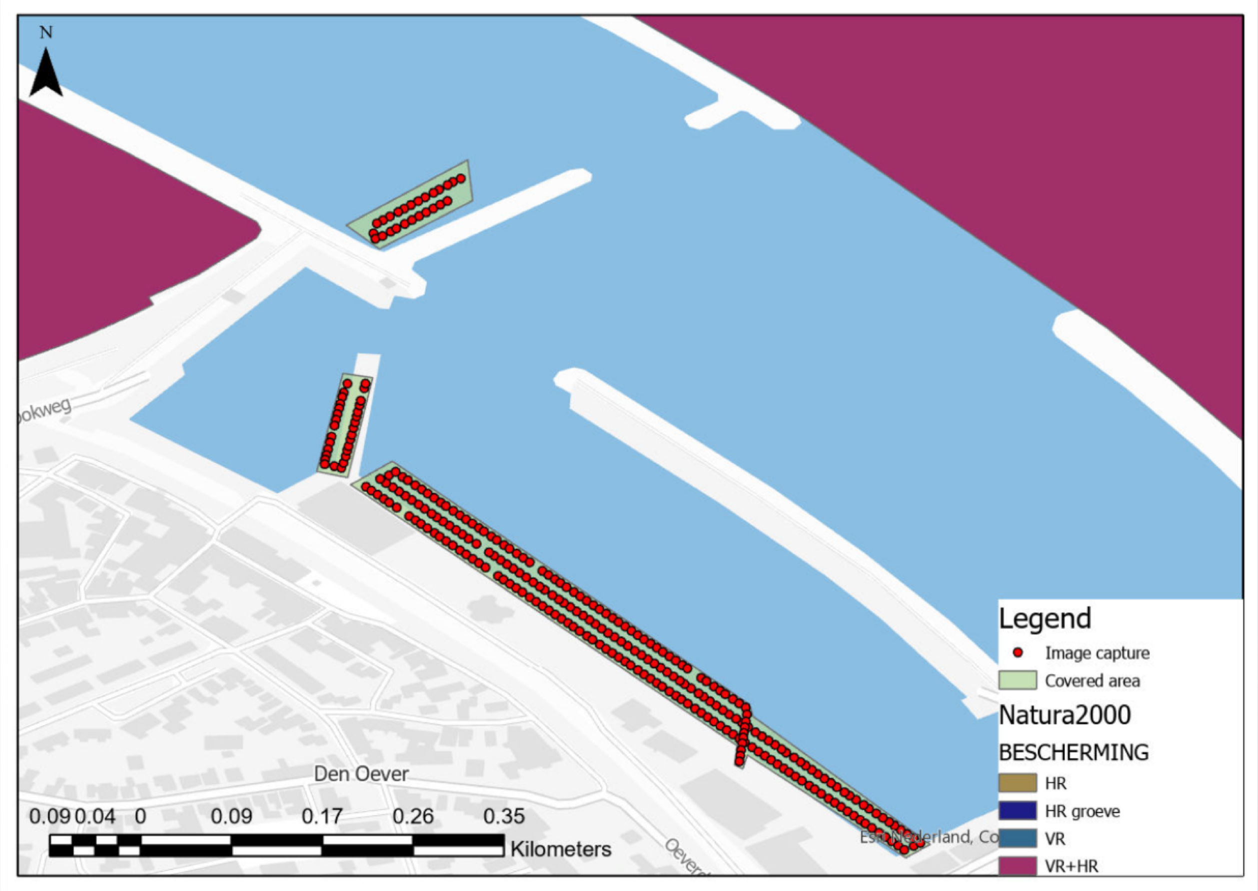
Methodology



Data collection

- Den Oever
  - Nadir
  - Predefined path

DJI Mavic 2 Enterprise Advanced

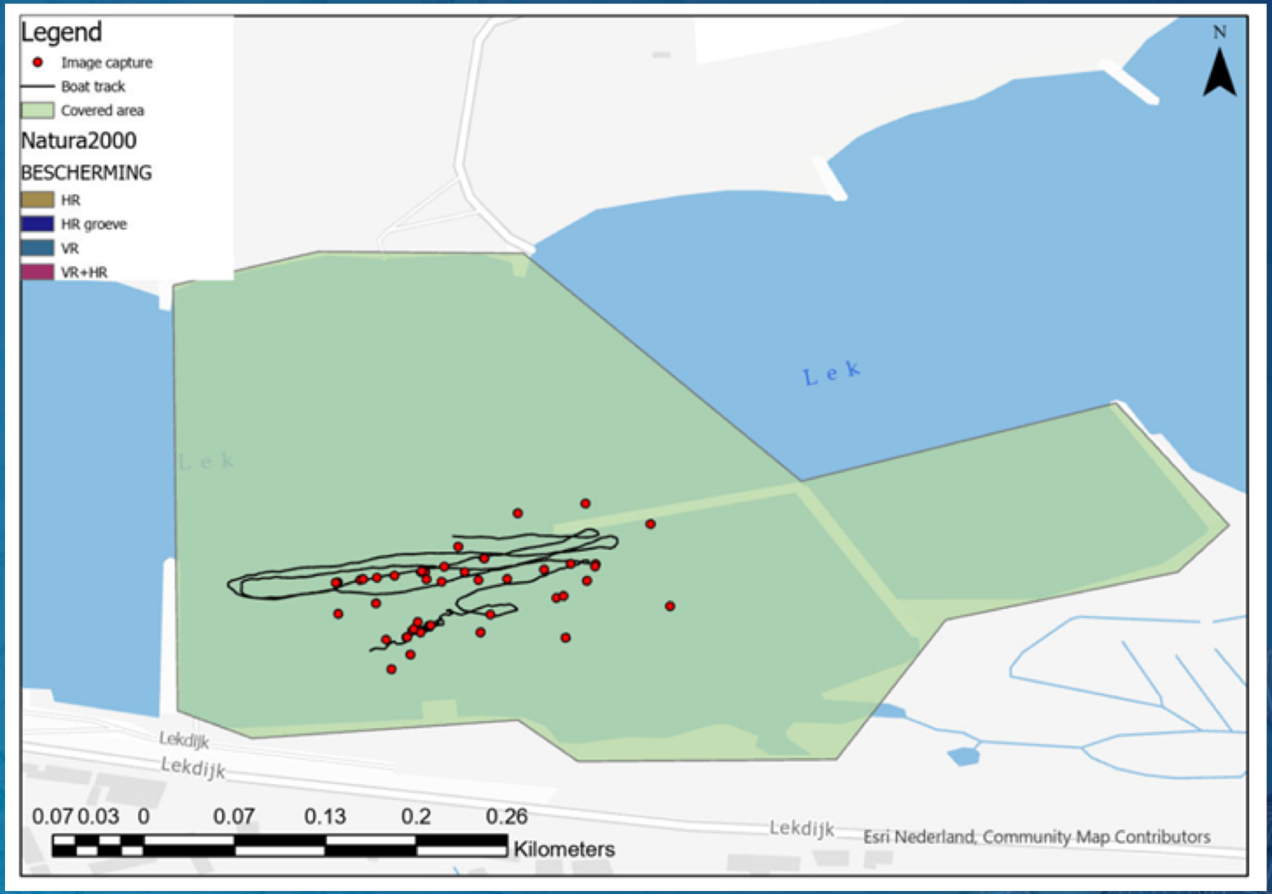


Den Oever data collection area

Data collection

- Ameide
  - Nadir & oblique
  - Boat movement
  - GPS tracker

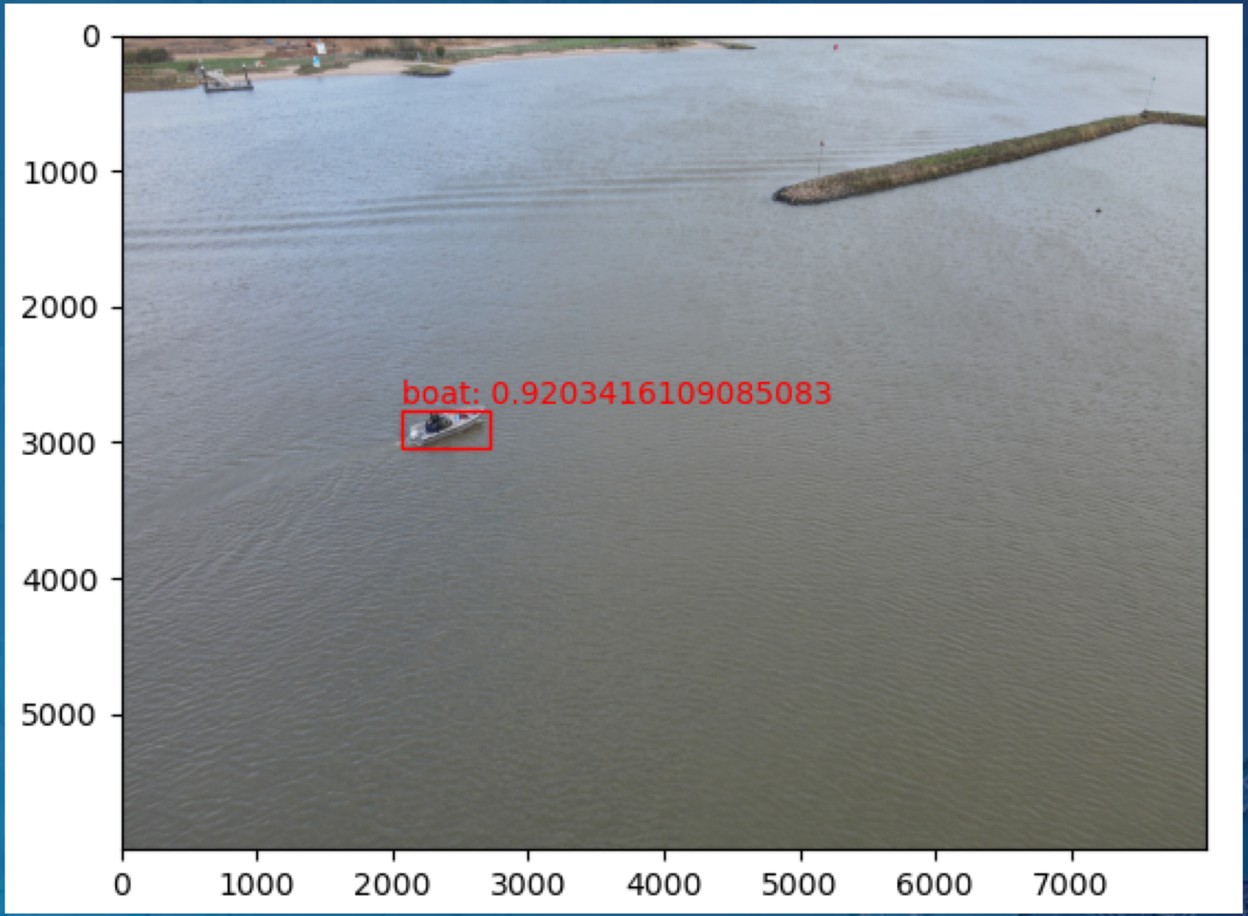
DJI Mavic 2 Enterprise Advanced



Ameide data collection area

Object detection

- YOLOv3 detection model
- ArcGIS Python API
- Pretrained on COCO dataset

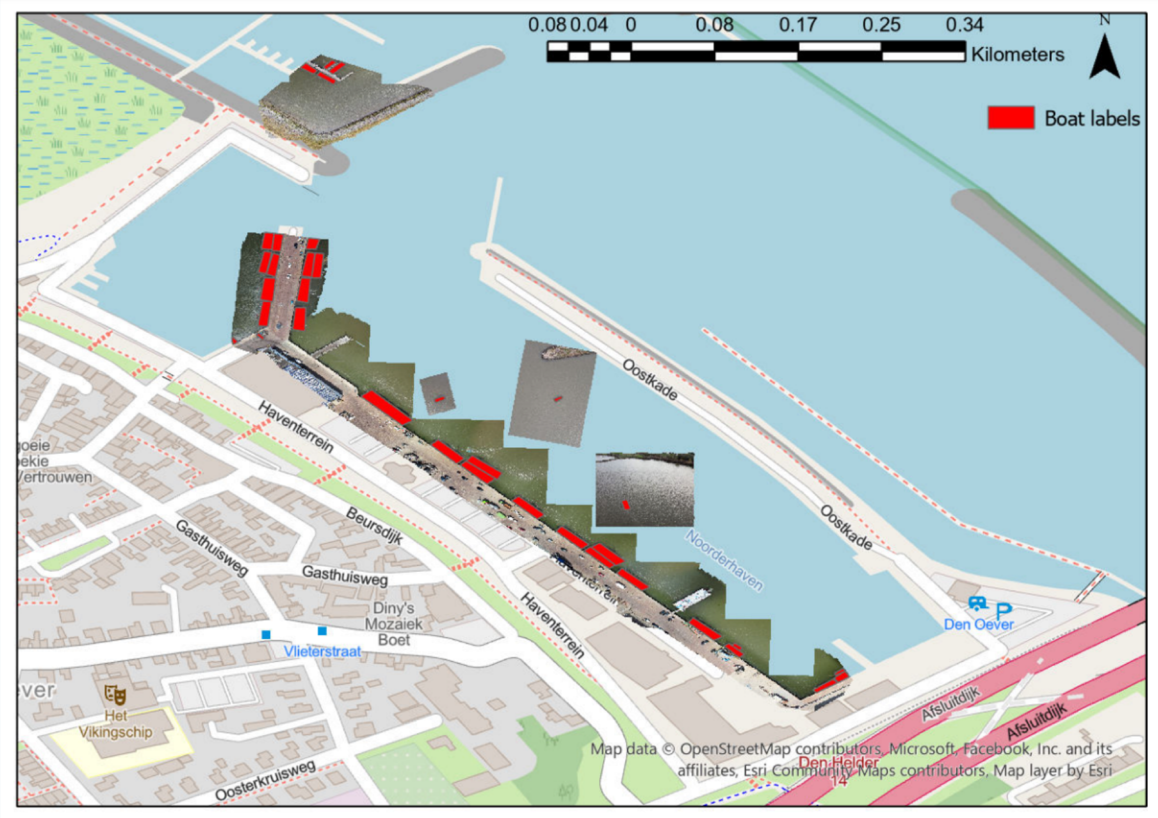


Output YOLOv3 model

Training the detection model

Generate raster → Label objects → Generate image chips → Train deep learning model

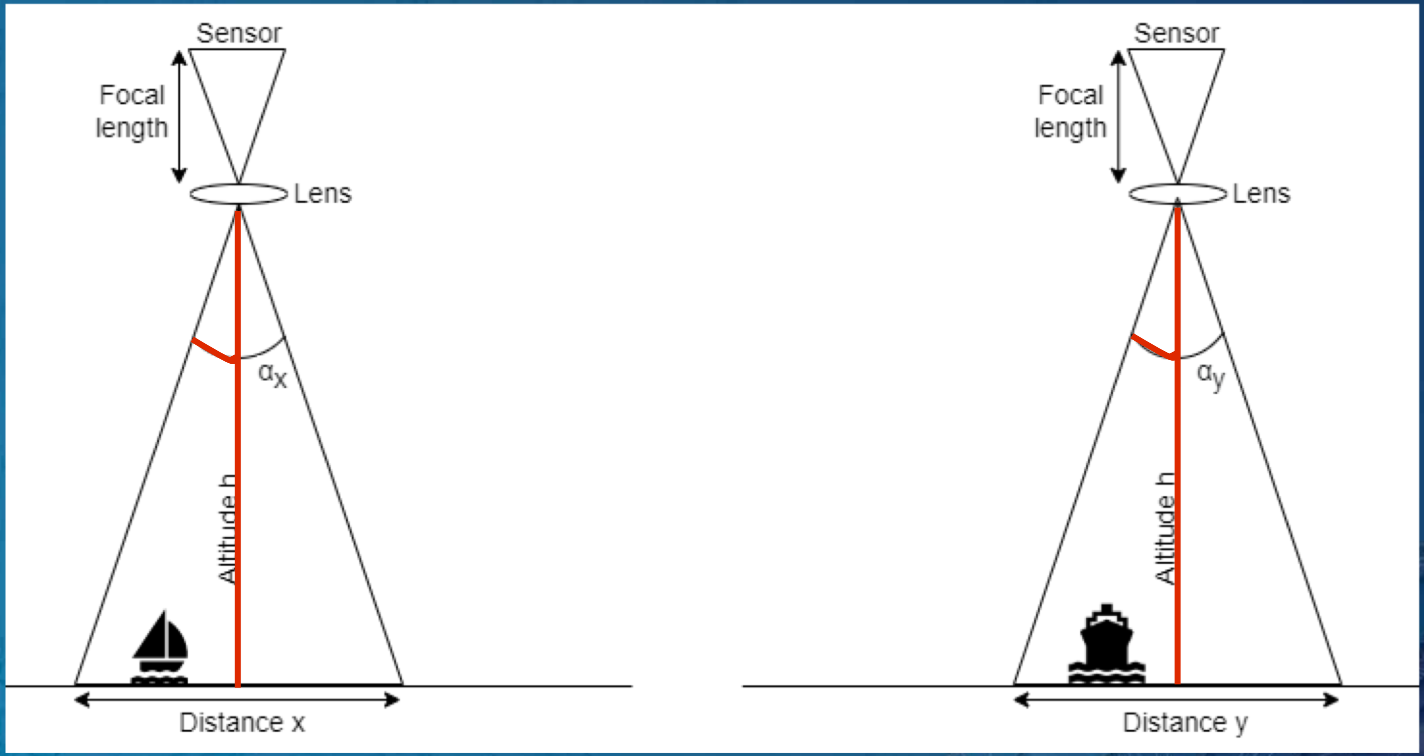
- Train pretrained YOLOv3 model
- Imagery input



Input training

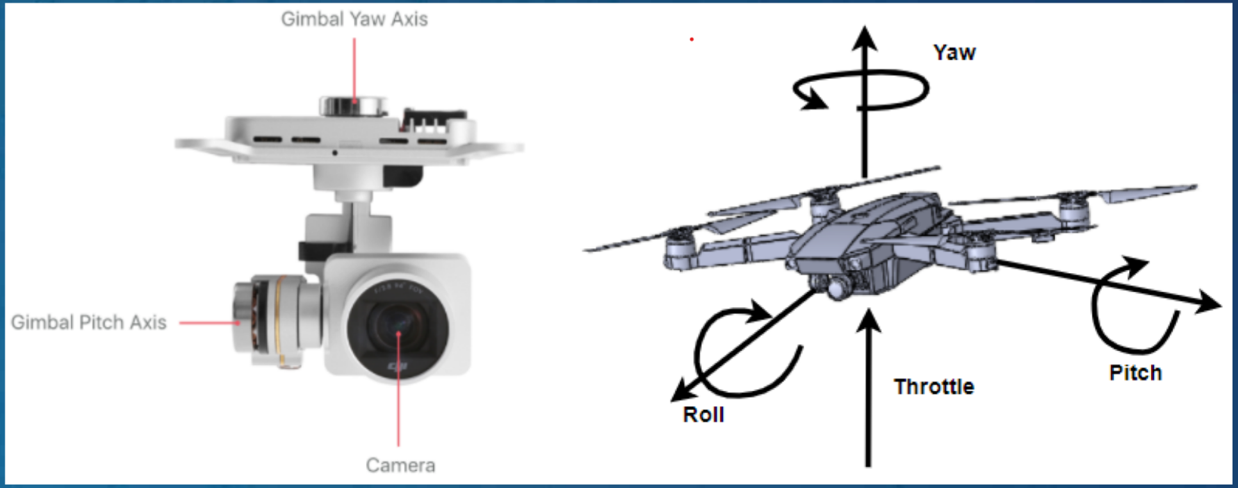
Nadir positioning

- Metadata: camera pose and drone position
- Meters per pixel

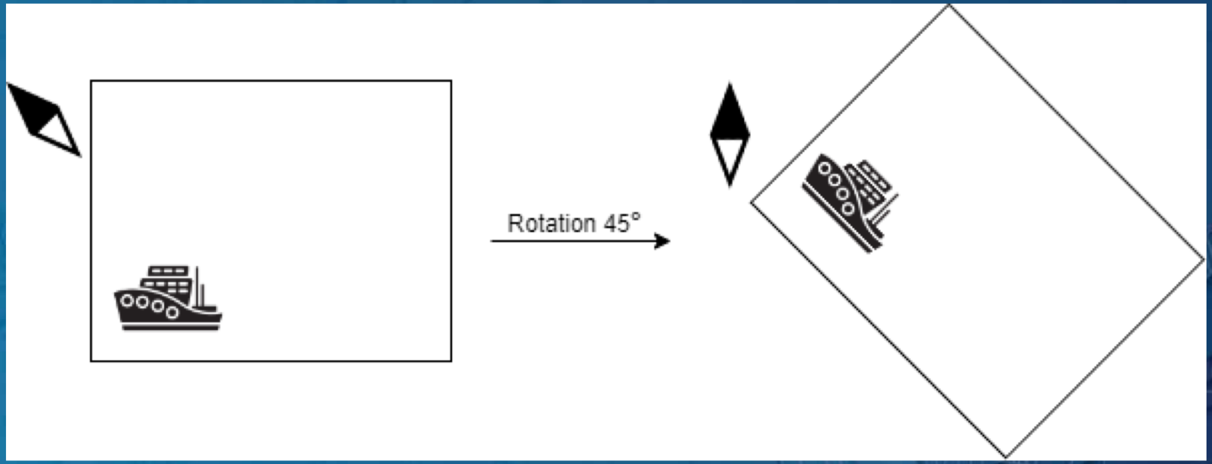


Nadir positioning

- Metadata: camera pose and drone position
- Meters per pixel
- Rotation with yaw

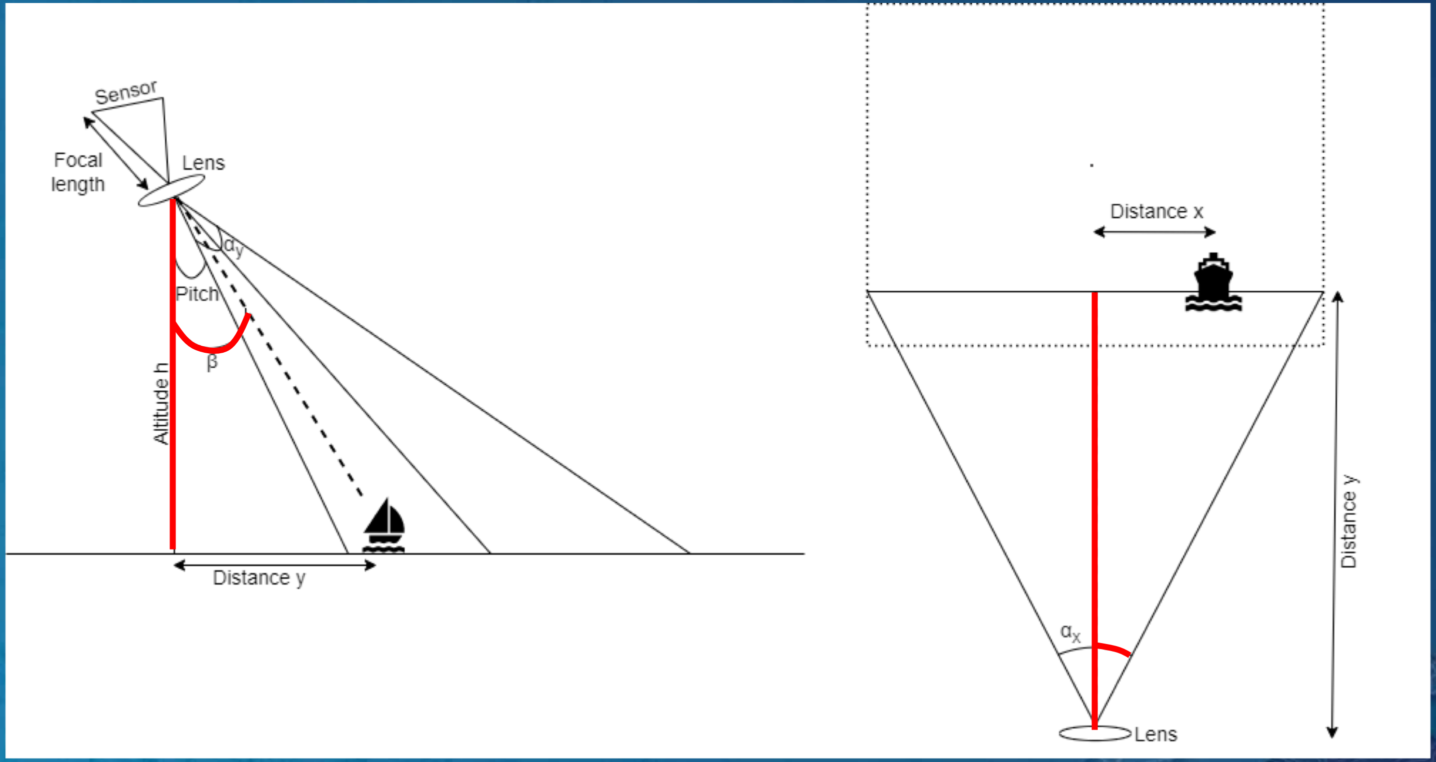


Camera and drone principal axes



Oblique positioning

- Added pitch parameter
- Different approach x and y-axis

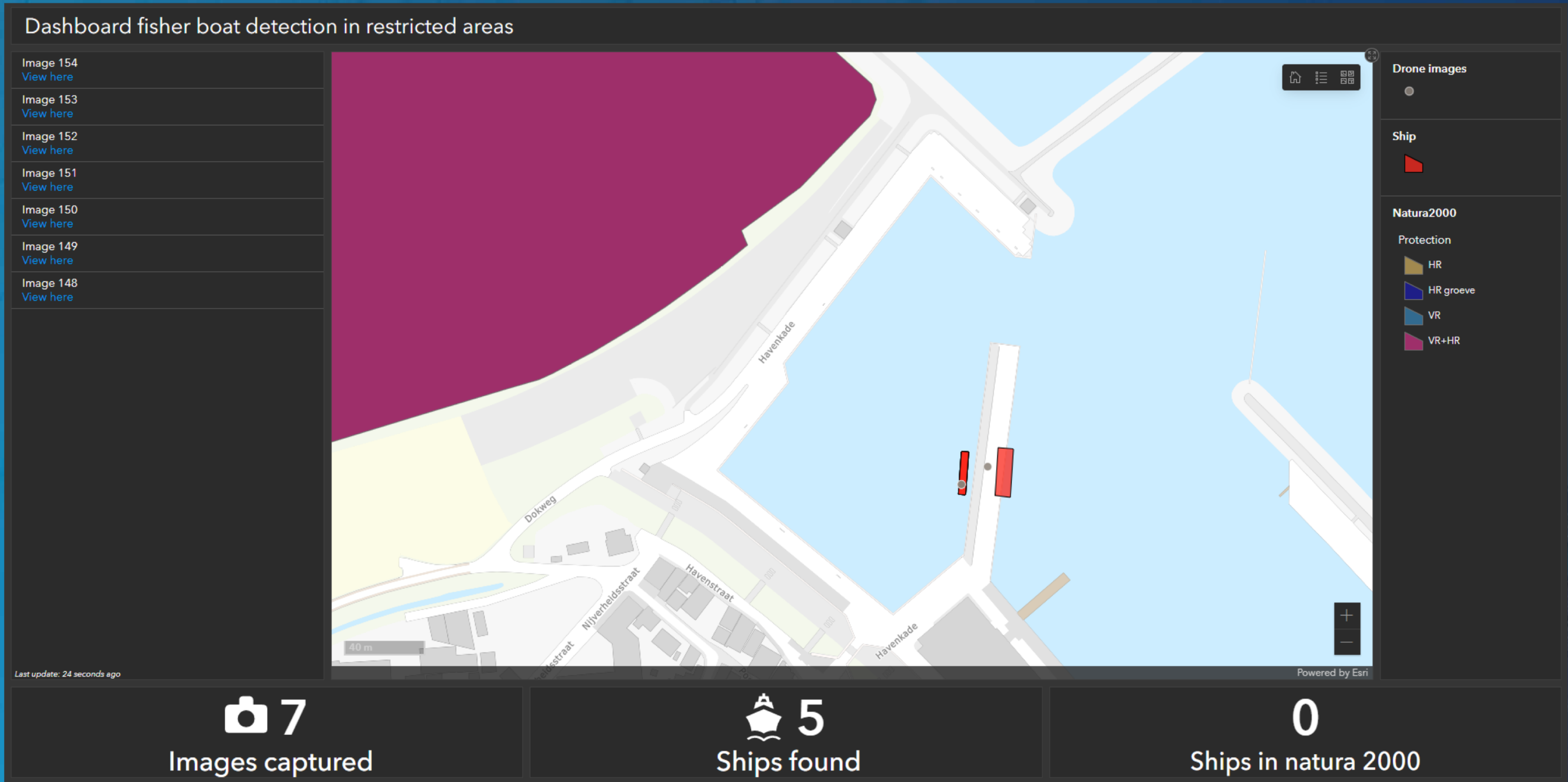


Real time connection

- Controller connection to cloud
- Python connection to cloud

Localisation with a dashboard

- Map with polygons and restricted areas
- Statistics
- Updates



Prototype

Input: link to Google Drive
Output:

$M \leftarrow$ deep learning model
Fill stack with image IDs

Prototype

Input: link to Google Drive
Output:

    $M \leftarrow$ deep learning model
    Fill stack with image IDs
    while IDs in stack **do**
        Get ID from stack
        Download image bytes *img* with ID

Prototype

Input: link to Google Drive
Output:
$M \leftarrow$ deep learning model
Fill stack with image IDs
while IDs in stack **do**
Get ID from stack
Download image bytes *img* with ID
$p \leftarrow$ inference using M and *img*
filter boat features b from p

Prototype

Input: link to Google Drive
Output:
    $M \leftarrow$ deep learning model
    Fill stack with image IDs
    while IDs in stack **do**
        Get ID from stack
        Download image bytes *img* with ID
        $p \leftarrow$ inference using M and *img*
        filter boat features b from p
        for each b **do**
            $c \leftarrow$ positioning b

Prototype

Input: link to Google Drive
Output:
$M \leftarrow$ deep learning model
Fill stack with image IDs
while IDs in stack **do**
Get ID from stack
Download image bytes *img* with ID
$p \leftarrow$ inference using M and *img*
filter boat features b from p
for each b **do**
$c \leftarrow$ positioning b
write c into online layer
end for

Prototype

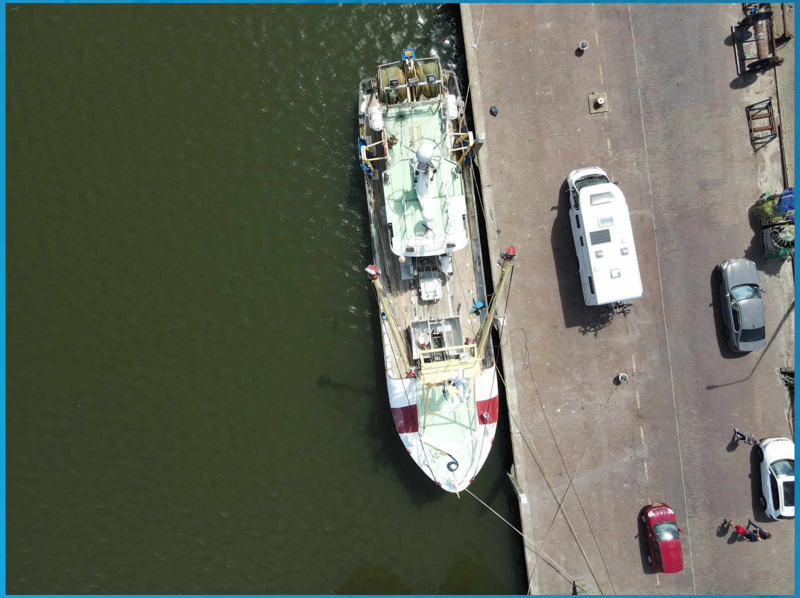
Input: link to Google Drive
Output:
$M \leftarrow$ deep learning model
Fill stack with image IDs
while IDs in stack **do**
Get ID from stack
Download image bytes *img* with ID
$p \leftarrow$ inference using M and *img*
filter boat features b from p
for each b **do**
$c \leftarrow$ positioning b
write c into online layer
end for
remove current ID from stack
add new image IDs from drive to stack
end while

Experiments

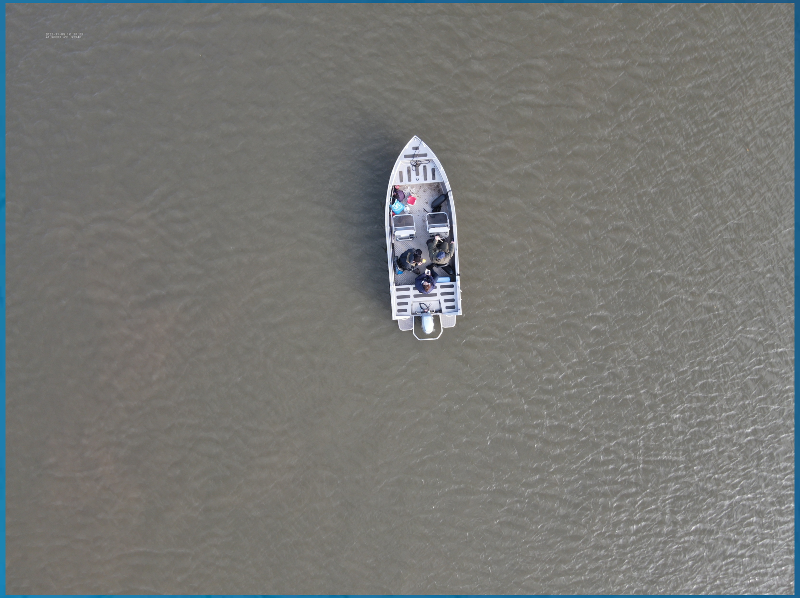
- 3 main components
  - Detection models
  - Positioning algorithms
  - Speed real time connection

Experiments - Detection

- Difference ground truth and detected labels
- Ground truth drawn manually



Den Oever

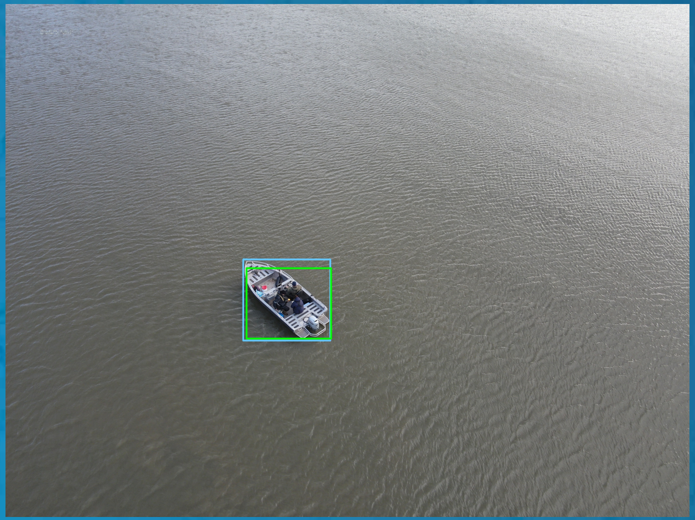


Ameide - nadir

Ameide - oblique

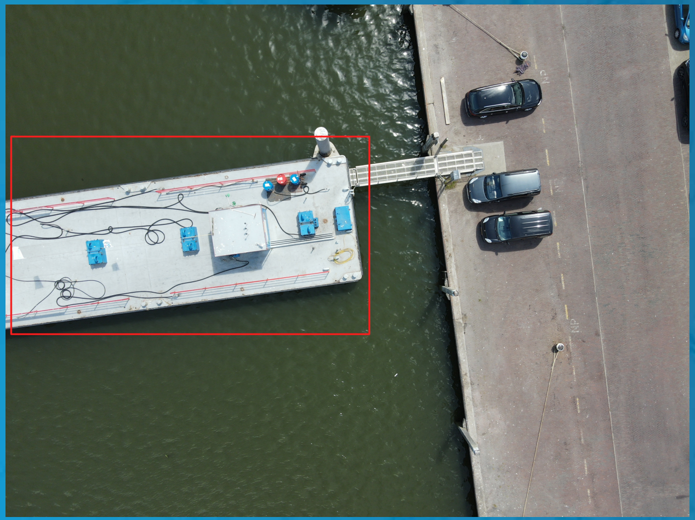
Experiments - Detection

True Positive

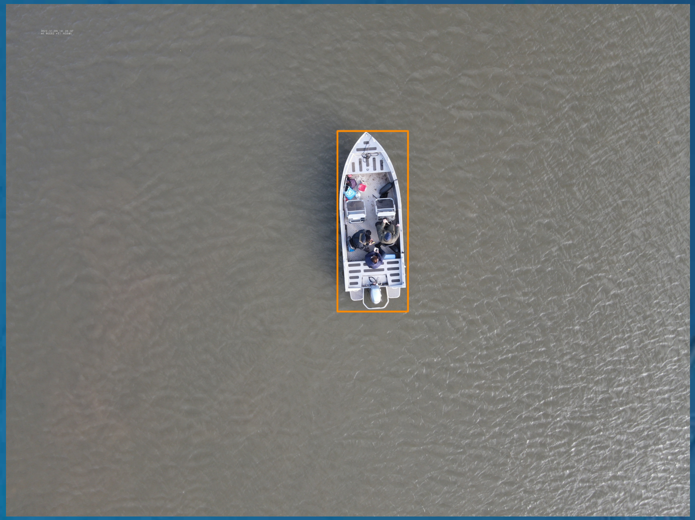


True Negative

False Positive

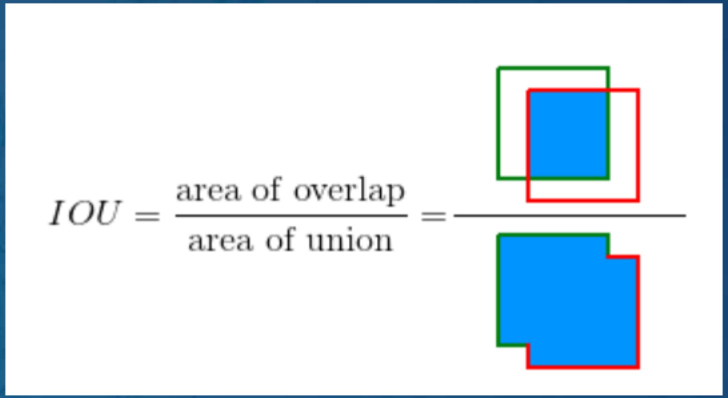


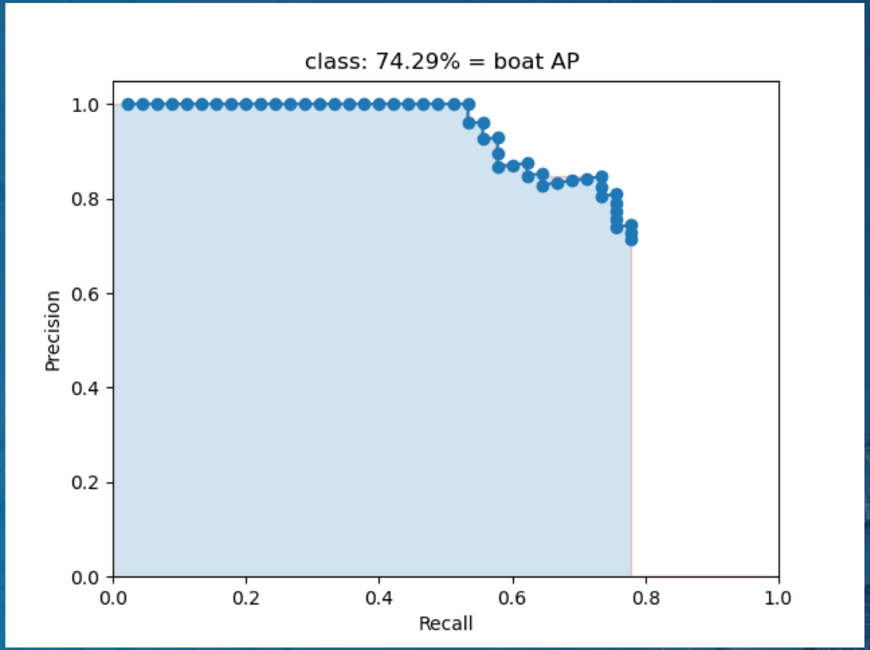
False Negative



Experiments - Detection

- Recall = TP / (TP + FN)
- Precision = TP / (TP + FP)
- AP = Area under the curve



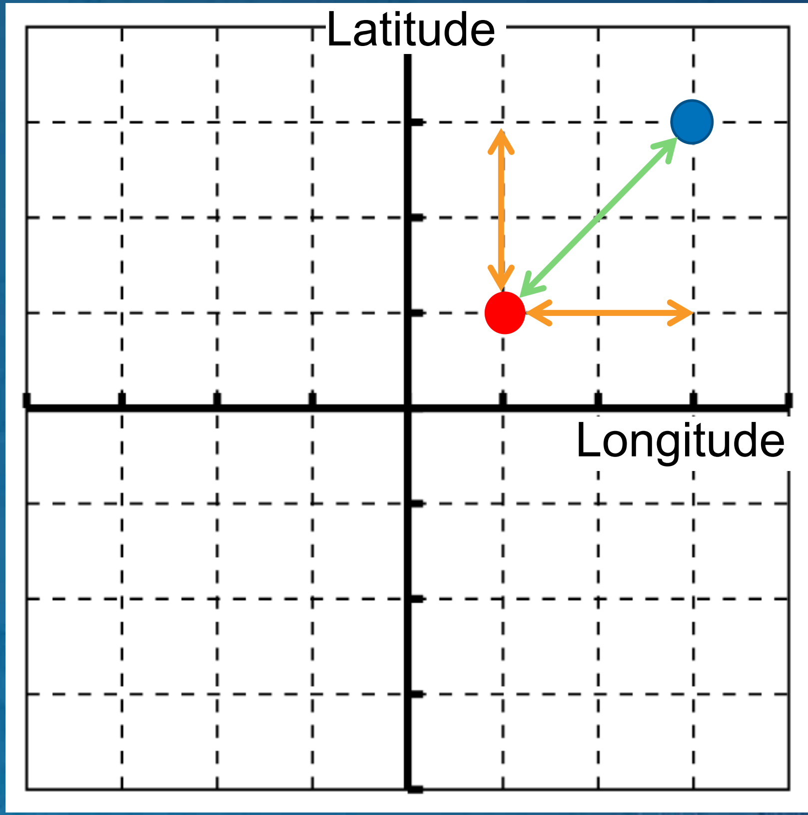


Example precision-recall curve

Experiments - Positioning

- Difference ground truth and positioned coordinates
- Conversion to meters

Collection ground truth data

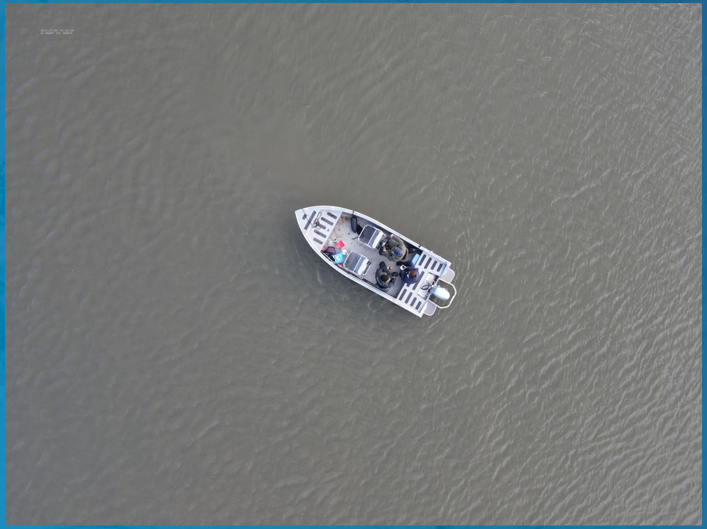


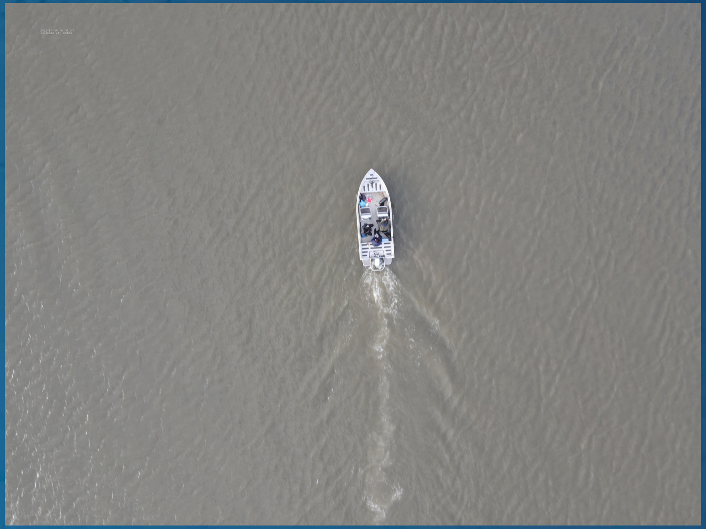
Experiments - Positioning

Motionless

Moving

Nadir





Oblique

Experiments – real time connection

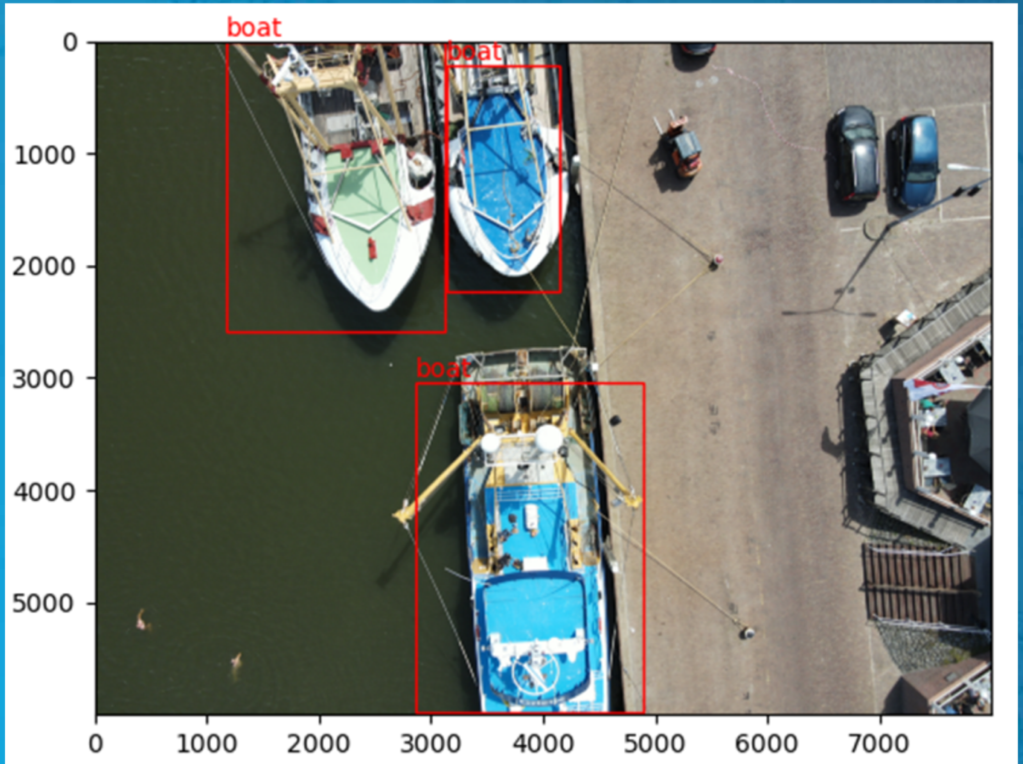
- Measure time main components
- Stopwatch and Python

Used stopwatch

Results – pretrained detection model

- Difference nadir & oblique datasets
- Den Oever boat parts

Dataset	Average Precision
Den Oever	25.15%
Ameide – nadir	6.06%
Ameide – oblique	74.29%

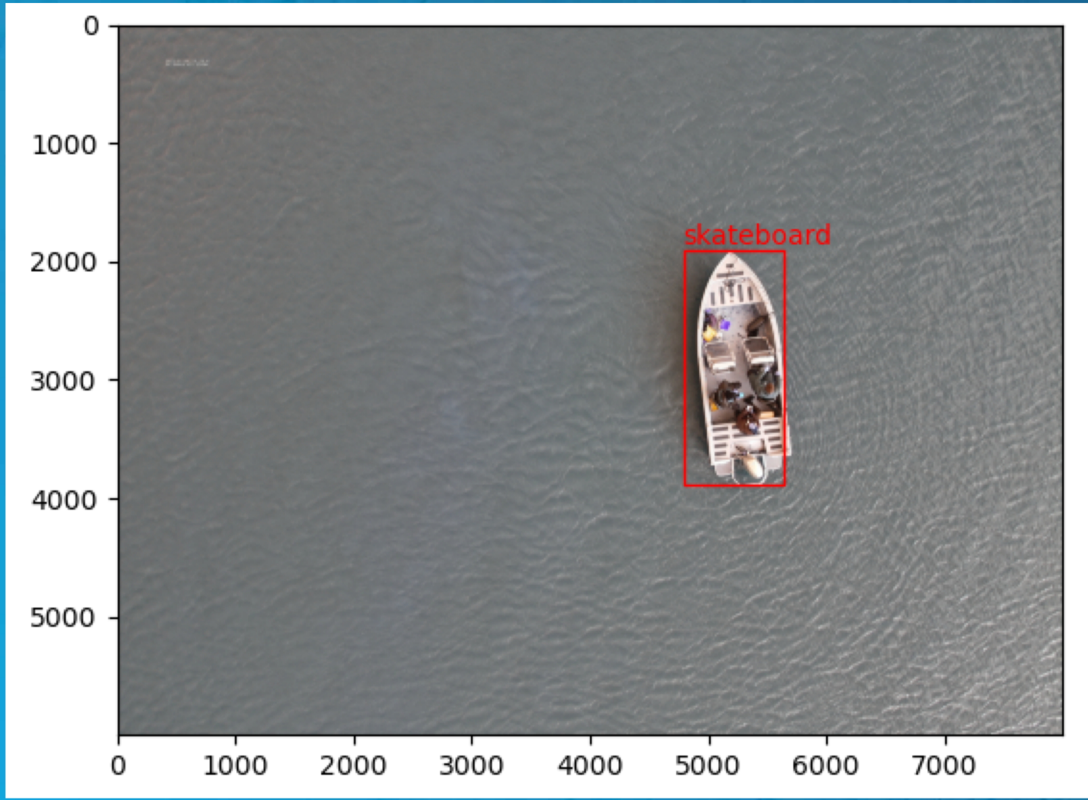


Detection result Den Oever

Results – pretrained detection model, all classes

- Increase in true positive and false positives
- Proves misclassification in pretrained model

Dataset	Average Precision
Den Oever	23.84%
Ameide – nadir	50.47%
Ameide – oblique	76.91%

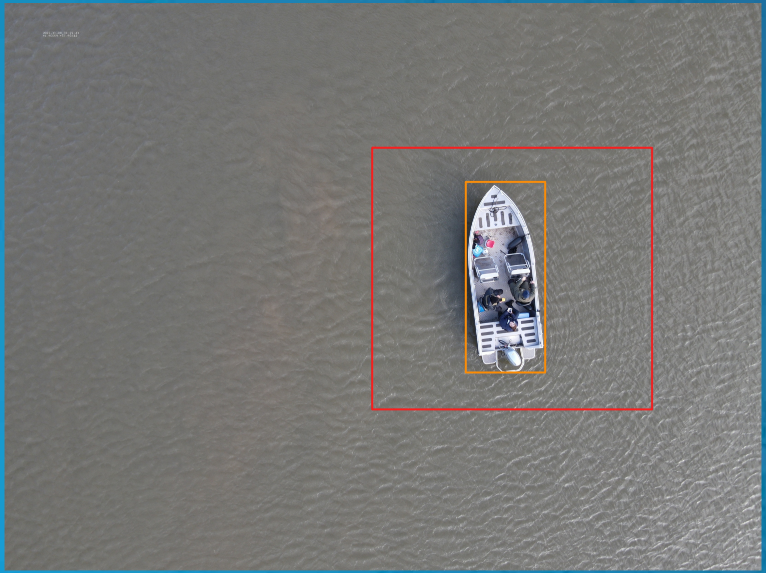


Misclassification Ameide dataset

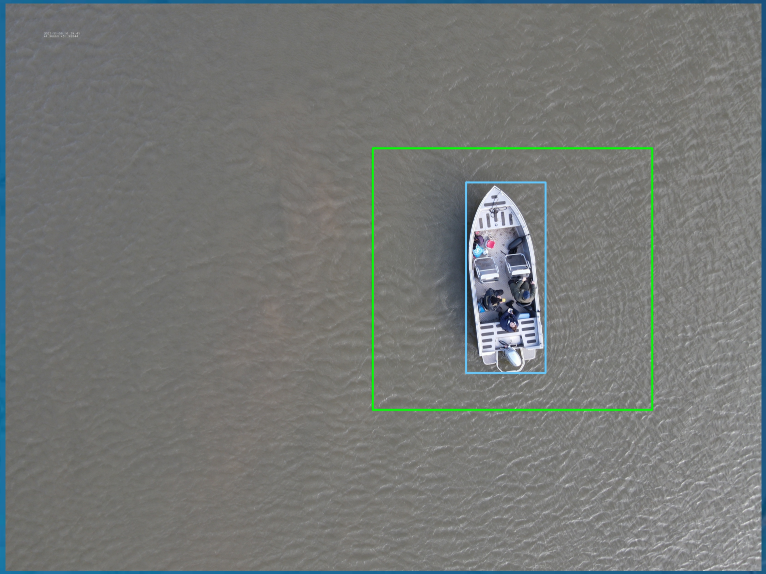
Results – trained detection model

- Predicted bounding boxes too big
- Training increases accuracy

Dataset	IOU	Average Precision
Den Oever	0.5	42.83%
	0.1	68.55%
Ameide – nadir	0.5	12.12%
	0.1	42.42%
Ameide – oblique	0.5	4.44%
	0.1	4.44%



Detection result IOU: 0.5



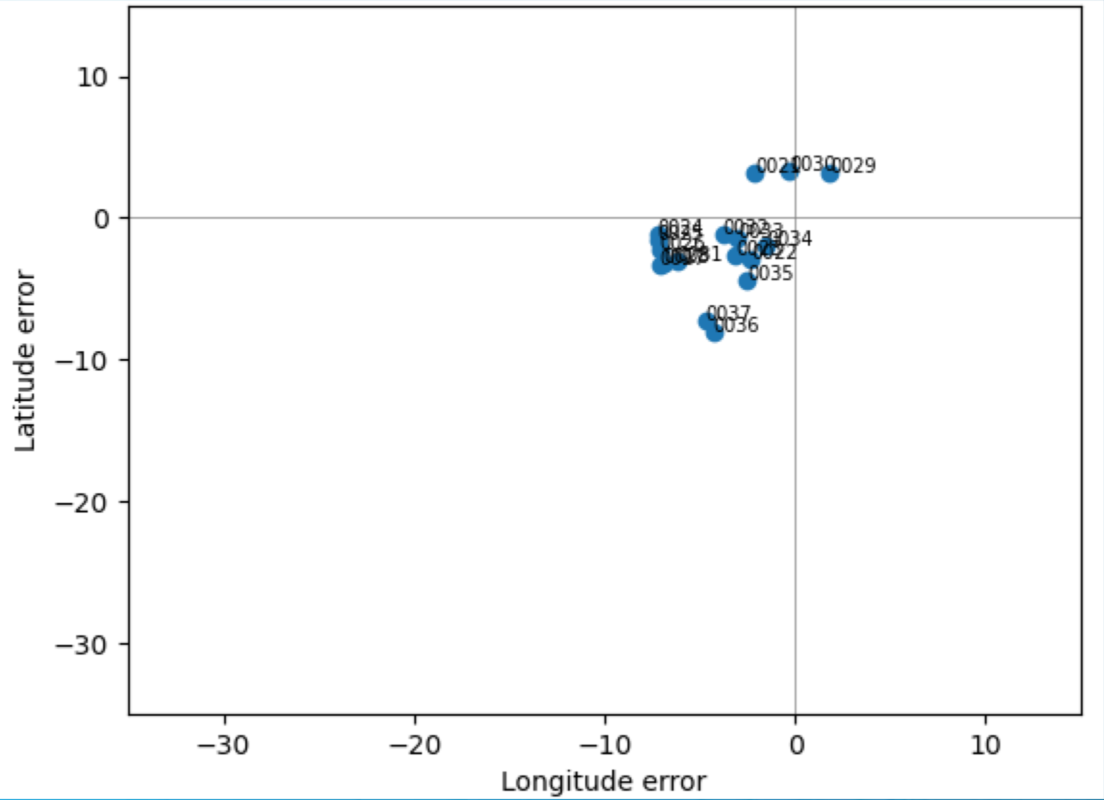
Detection result IOU: 0.1

Results - Positioning

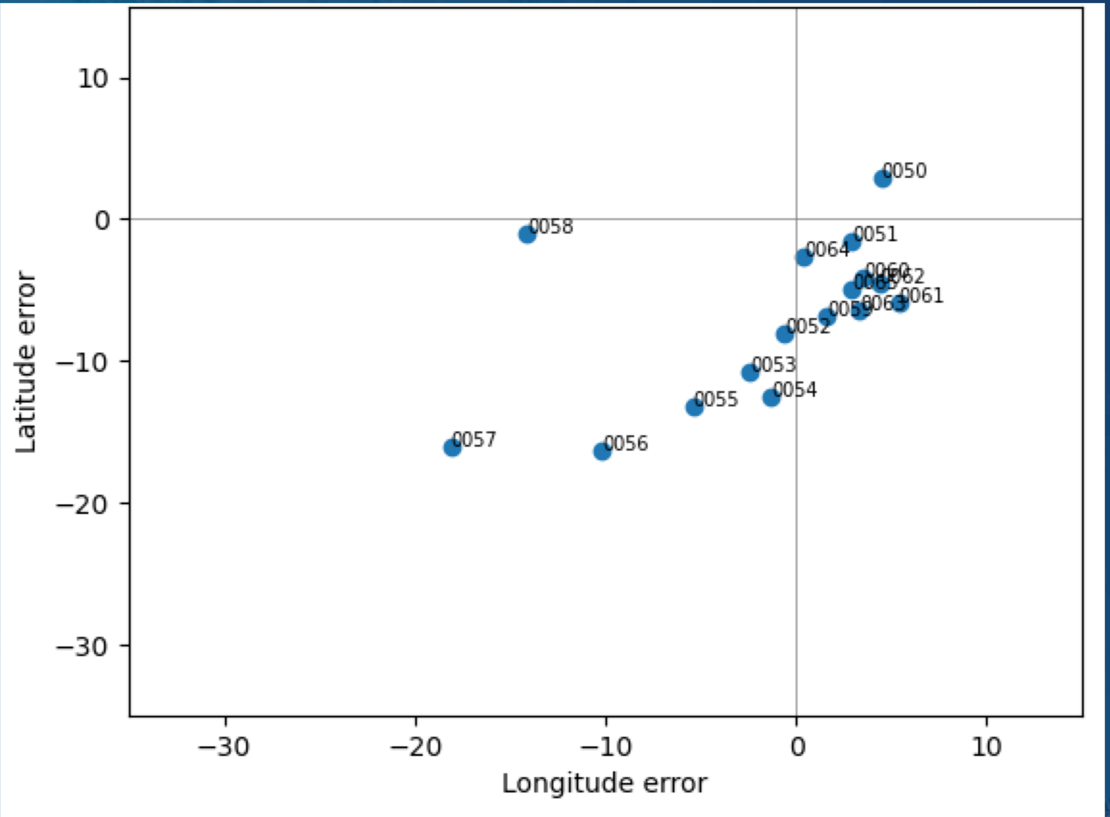
- Motionless & nadir perform better

Euclidean absolute average error in meters		
	Motionless	Moving
Ameide – nadir	5.6	9.7
Ameide – oblique	8.2	19.6

Results – nadir positioning

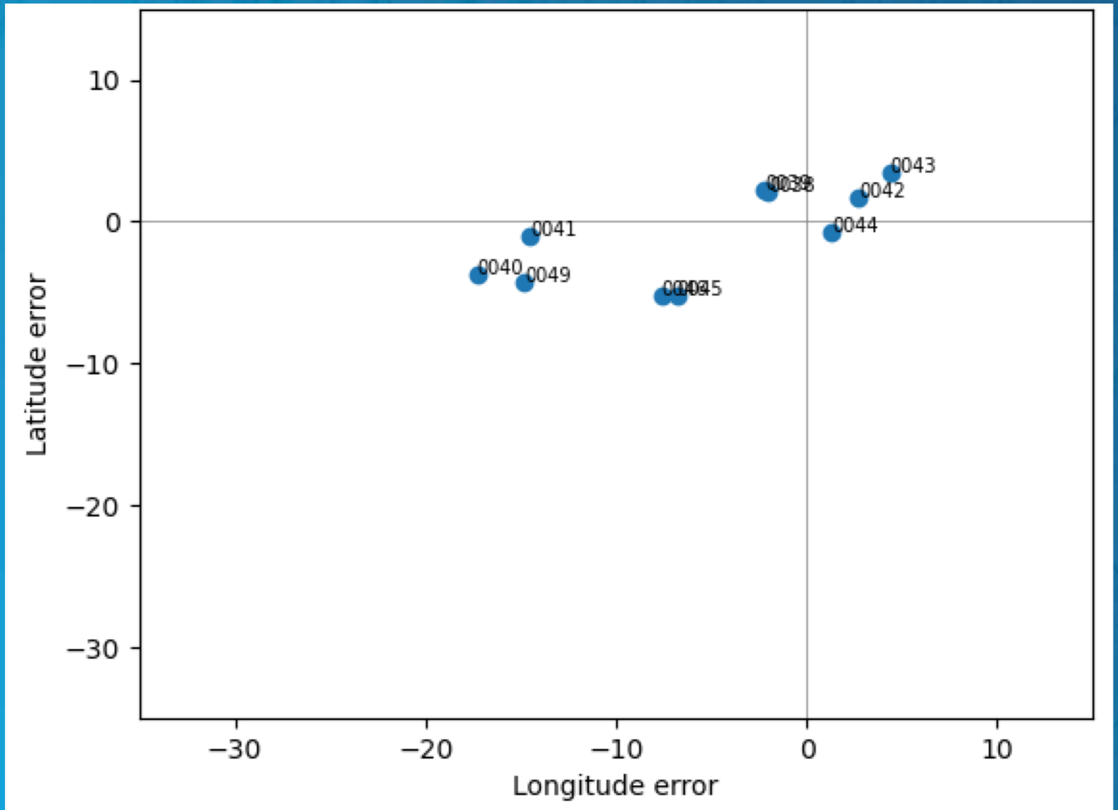


Nadir images – motionless boat

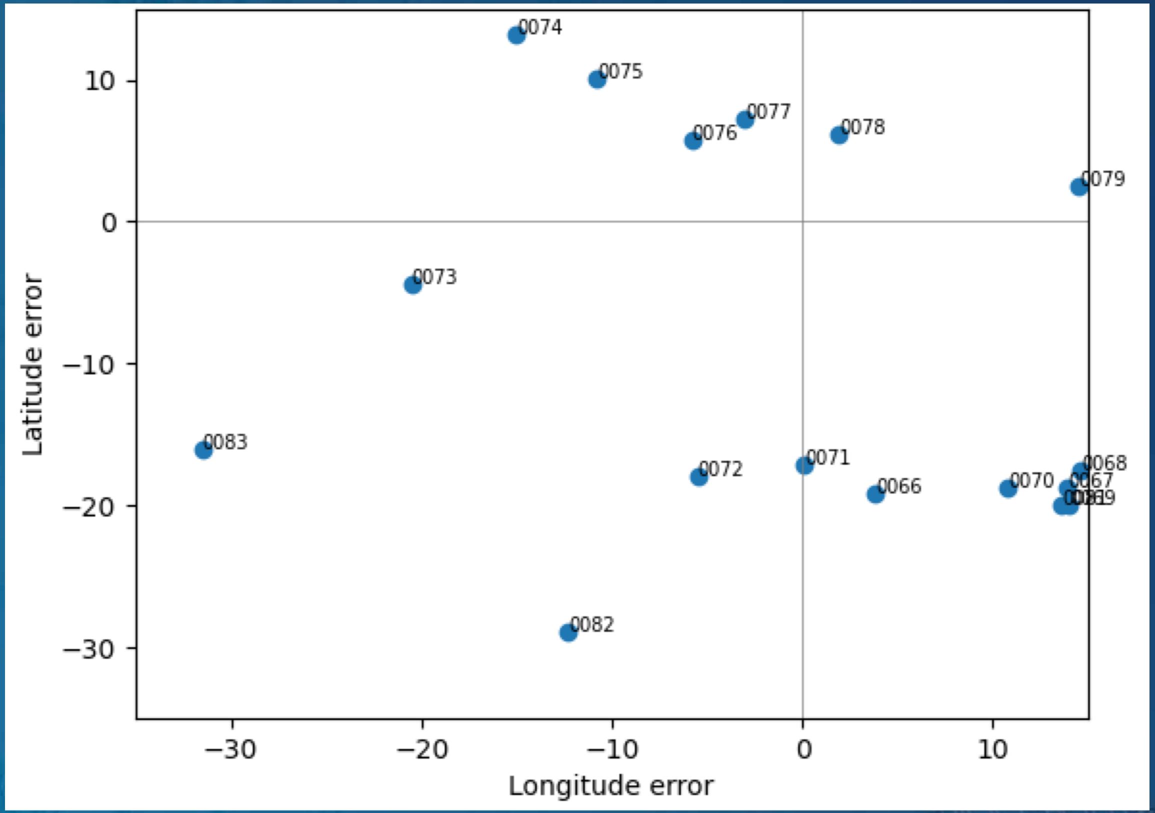


Nadir images – moving boat

Results – oblique positioning



Oblique images – motionless boat



Oblique images – moving boat

Results – real time

- 25.72 seconds one image
- Downloading and uploading time consuming

Processing time in seconds						
Download full size	To Drive	From Drive	Detection	Positioning	Write to file	Code total
2.53	9.97	0.31	1.01	0.0	3.49	13.21

Discussion

- Detection improvement with training
- Positioning improvement with better metadata
- Real time improvement with processing on drone or controller
- Privacy

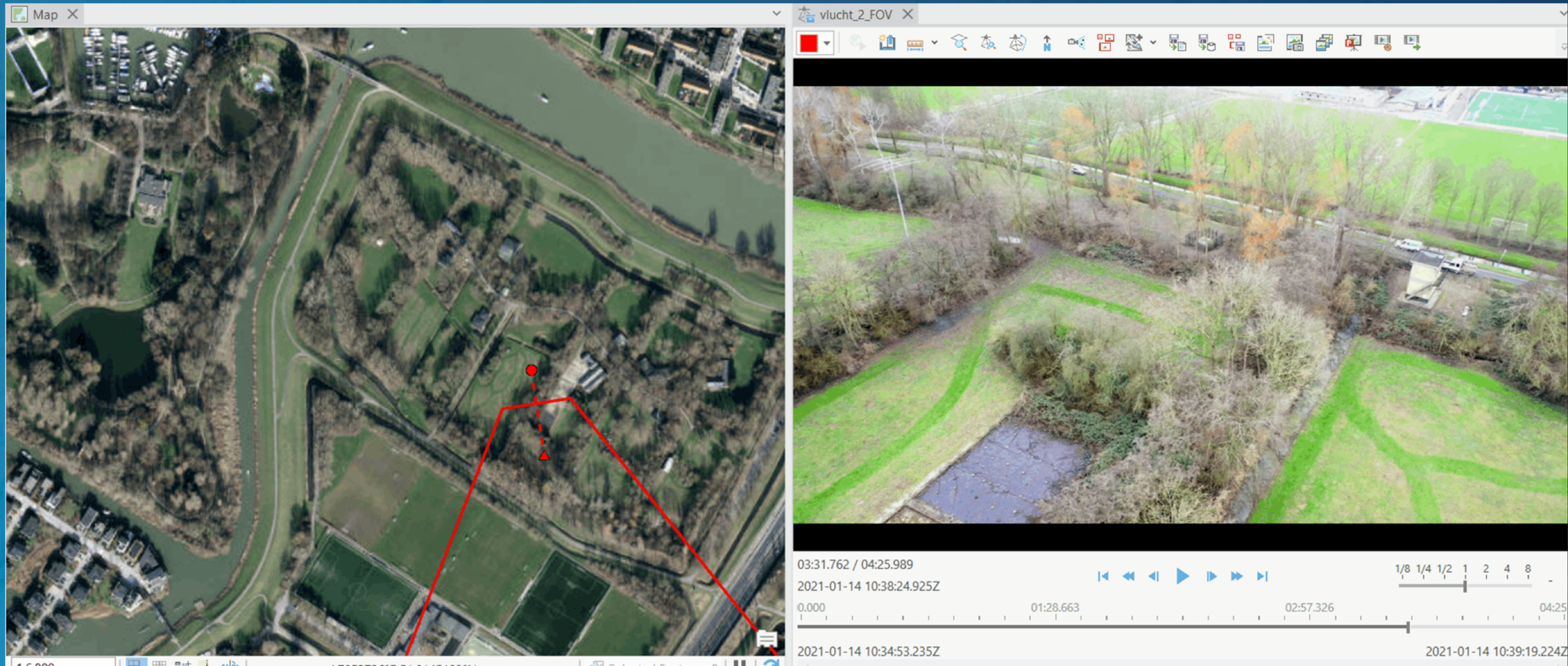
Conclusion

- How can deep learning be used to detect objects on drone images?
- How can detected objects be automatically positioned in a geographical coordinate system?
- What hardware and software is needed for this method to be carried out in real time?

- ***To what extent can drones be used to localise objects in real time?***

Future Work

- Tracking
- Full motion video
- Processing on drone or controller



Full motion video example

Thank you for your attention!