Object Detection, Tracking & Prediction

Object Detection & Tracking = ODT
Perhaps rename to general-purpose distributed inference system (GP-DIS)?
ODT Platform = a service, or a set of microservices running on a number of devices, that can perform on-request, or continuous deep learning inference.

Hoe worden applications gepackaget?!

Constraints ivm I/O types worden bij static, micro-placement naar coordinator gepushed

Tips

We see multiple constraints here (input type, output type, device, specific model, …), perhaps unify these into a single “constraint” mechanism, that allows for selecting the correct models

Developer time is extremely valuable, so make sure to allow for easy inspection of your system, and develop debugging tools when necessary

Automation is key if you want to test distributed system scalability, make sure you can launch new instances of your system in an automated way, and send (random) requests in an automated way, to squish all the bugs that occur in the tiny little edge cases

Schema-based validation of data for formats like JSON and YAML is a very easy way to check for correctness, some formats like XML even allow you to define data-transformations (using things like XSLT).

Requirements

All items have an associated priority, going from 100 to 0, with 100 being the highest priority, and 0 being the lowest priority.

P100	Support for multiple input types
Description	
The ODT system should be able to deal with different types of input for the neural networks it manages. Some of the possibilities are: ● Still camera images (P100) ● RGB-D Camera Images (P100) ● LIDAR Point Clouds (P100) ● Intermediate Featuremaps (P75) ● Video feeds (P50) ● SONAR Data (P???) ● RADAR Data (P???) ● Graph Data (GCNs) (P0) ● Random Latent Space Noise (GANs) (P0) ● Arbitrary RL Data (P0) When the input type of a network, and that of a given input match (for example, both are images), it’s important that the ODT platform exactly matches these, (Such as rescaling the size of the input image to match the size of the network), this should happen in an automatic fashion.	
External Dependencies	
Niels + Thomas: SONAR + RADAR: Overleggen wat feasible is in eerste versie	

<table>
<tr><td>P100</td><td rowspan="2">Support for multiple output types</td></tr>
<tr><td><b>Description</b></td></tr>
<tr><td colspan="2">The ODT system should be able to deal with different types of tasks that need to be performed, some examples of this are:
● Image Classification (P100)
● Object Detection (P100)
● Image Segmentation (P50)
● General Classification (P50)
● General Regression (P50)
When working with things like classifiers, it’s important to ensure that the classes the network outputs match the classes the output expects.</td></tr>
<tr><td><b>External Dependencies</b></td><td></td></tr>
<tr><td colspan="2"></td></tr>
</table>

<table>
<tr><td>P100</td><td rowspan="2">Cross-Platform Support</td></tr>
<tr><td><b>Description</b></td></tr>
<tr><td colspan="2">The ODT platform will initially be running on ARM-based NVIDIA devices, but might get deployed to different types of devices later on, the platform should thus be developed in a cross-platform manner, and should make as little assumptions as possible about the underlying hardware. This includes:
● The ability to use CUDA accelerators (Cf. Jetson Nano)
● The ability to use TPU accelerators (Cf. EdgeTPU)
● The ability to run networks completely on a CPU</td></tr>
<tr><td><b>External Dependencies</b></td><td></td></tr>
<tr><td colspan="2"></td></tr>
</table>

P100	Supported Model Types
Description	
There are a number of different formats that can be used to store machine learning models: ● ONNX (P100) ● Torch state dicts (P75) ● TensorFlow models (P50) ● TensorFlow Lite models (P50) ● … Our platform should be capable of supporting a number of different model types	
External Dependencies	

P100	Automatic Docker Build System
Description	
The networks will be distributed as Docker container	
External Dependencies	

P50	Profiling
Description	
When returning the result from a request, the ODT platform should always include some basic profiling information: ● Inference Time (P100) ● Memory Use ● Power Use ● …	
External Dependencies	