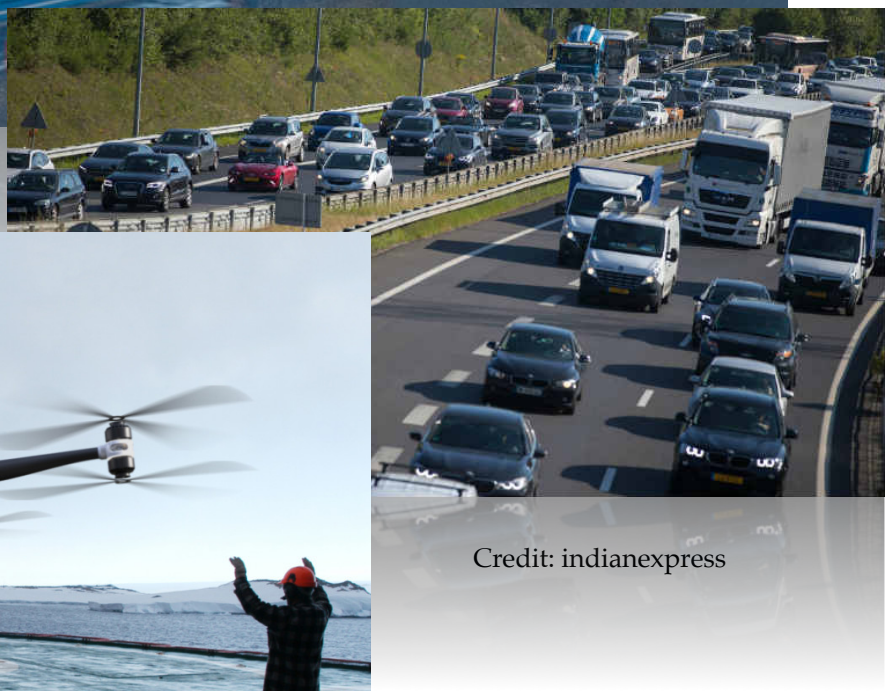
Intelligent Mission Controller

In Autonomous vehicles

By Terje Nilsen

Brandbu, NORWAY - 25 January 2019

Credit: Kongsberg



Credit: indianexpress

Credit: Fedotov

Introduction

We see more and more autonomous vehicles arriving, from small worker-drones carrying only a camera, advanced robots, cars, to huge container ships. A lot of resurge is done on the autopilot, being it a flight controller for drones and airplanes, an autonomous autopilot for ships or a self-driving car. My work is for the “mission controller”, the computer or software module that owns the mission, not sears the vehicle, rather that makes high order decision based on exceptions. This module do NOT consider the motion itself, it does not control the vehicle directly, but it is able to alter the current mission path or goal on-the-fly in real time when an exception occurs.

Examples is, if a ships main axel starts taking in water, then today the sensors will sense it, but what will the autopilot do? Between the sensor and the autopilot sits the mission controller, that analyses all sensor inputs and evaluate all possibilities should something happen. Should we stop, alter course and find the nearest harbor, or drive the ship hard on land to save it from sinking? If a drone loses power, what should it do? Land immediately, continue, engage the emergency parachute ? If a flying drone are ordered to follow a power line until half the energy is used and then fly back the same path, for inspections, and there are hard down-wind that day on the way out, it will never reach back (head-wind), the mission controller will have to override the system at the point-of-no-return, rather than wait for the calculated half way, or is the mission so critical that we do not care and fly beyond point-of-no-return ? There are no humans in the loop, as the radio is dead, humans are captured in a snow avalanche, drone equipment is replaceable humans are not, how can the drone fight the snow storm even if it might never return, only focusing on finding and tagging thaws people.

There is a close to infinite list of eventualities the might happen and that the autonomous system as such cannot do much about. What makes it complicated is that using classical methods one can only cover tows errors that was foreseen, and especially programmed in to the system. This is where search engines comes to play. The idea is to make intelligent decisions should the unforeseen happen, as one day it will. We have seen so many examples with autopilot in airplanes having sensor problems and making bad decisions.

I believe we can use deep Q-Learning to training for what situations we can think of, and then let the algorithm find the best solution even if the situation is never before seen or was classified as impossible to ever happen.

Domain background

Autonomy is being used in everything from robots to drones, cars, airplanes, rockets and lately also container ships. The idea is to make a domain agnostic algorithm, that takes a set of sensors as input and output a set of high-level decisions, not like the autopilot, but that alters the goal for the autopilot, like stop mission and land at new landing point. As this is a machine learning algorithm, we need not program the response to each sensor or combination, but we rather use training, with simulated input. The goal is to have a supervisor function that can be adapted to any autonomy system, and have it taking overall decisions that cannot be expected to be part of the autonomy as such. It is sort of an anomaly detector, that from the anomaly makes decisions as a human would in today's partly autonomous systems.

To be able to design this algorithm in the short time, I will consider discrete values, but to prove the point, maybe add one value function and a continuous value. In the real product I imagine a set of selectable inputs form a .xml file, more like a PLS system. However in this project it will be a simplified version to prove to point.

Problem statement

When an autonomous vehicle experience problems, especially combined problems what we not foreseen, it's desertions might not be optimal, and in some cases disastrous. We need an algorithm that can supervise all sensors and do a sanity check, and then when things breaker down, makes good decisions, like running a ship on land at a safe location rather than sink, if possible. Even a total breakdown can have a best and worst solution.

Datasets and inputs

In this case the dataset has to be simulated. I will make a simulator that randomly train the system on errors that can happen, and then test the system om errors that was not part of the training. As this is a simplified version of the potentially final product, it will be limited on complexity. The input will mainly be Boolean type sensors, that is ether true or false. However I would like to add at least on analog input, as that increases complexity and makes it more real. Some generic sensors might be energy/fuel level, speed, heading etc. and "water sensor" to detect leakage, oil pressure, vibration, etc. Also

payload sensors can be of interest, if the camera fails, do we continue the mission, when the mission is to film? Maybe 10 sensors in this initial project.

Solution statement

My thesis is that I can use Deep Q-Learning with a mix of digital and analog inputs, and the same on the output. Then by training on a set of possible problems with known rewards, and then testing on some non-foreseen problems, I will be able to find if the system makes sane decisions. The goal is to make equal or better decisions than what would have been made by humans under the extreme pressure of a catastrophic event, and make sane decision when smaller event occur, like if the sensor detecting the road suddenly gets a drastically increased variance, may hitting the brake would be a safe solution, rather than continuing…

I do understand that getting an algorithm that makes unforeseen decisions are hard to certify, however today we drive cars steered by deep learning, so the step is not that far.

Benchmark model

We may compare the result by what a traditional autopilot would have done, like the Ardupilot etc. However I might need some help and guidance for this part. Maybe there is an existing project on this already?

Evaluation metrics

As this is a simplified model, the number of devotions are somewhat limited. As of this I believe I can make evaluation metrics based on sane intuition. If the solution for a low fuel level is to turn off all engines to preserve energy in a flying Drone, one can say the system failed. When making the simulation stimuli on would have to classify solutions using sanity.

Project design

I envision a Deep Q-Learning architecture, with the combined sensors as a multi-dimensional state space, It would be nice to make the reward system partly based on responses from the vehicle, like if the solution was turn right hard now, and nothing happens, then the vehicle cannot turn right (for some unforeseen reason), and the reward should be increasingly negative such that the algorithm makes plan B after a while etc.