# AA222 Final Project status: Trajectory Optimization with dynamic obstacles avoidance

The project deals with trajectory optimization and control command of an object along a path where we have to avoid obstacles crossing our path. It is a multi-objective optimization problem: we have to optimize efficiency (time to goal), comfort (minimizing acceleration variations) and safety (avoiding collision).

So we far we did:

1. Litterature review and project objectives definition (cf below)
2. github repository setup
3. In the conitnuity of a previous project , we have a first baseline implementation. A MPC solution based on Julia JuMP.jl and ipopt.jl which is fully functional.

We rely on a direct collocation method where MPC is used: we plan over 20 time steps, for 20 commands, but apply only the first command before taking into account new observations for re-planning. What we plan to do next is:

1. Setup of our own optimizer, without relying on external packages like JuMP and ipopt. We will leverage on project2 work where an Interior Point Method based on log barrier was implemented. We implemented a first order (Conjugate Gradient) and second order (quasi-newton BFGS) optimizer with backtracking line search.
2. Handle an inconsistent constraint of the form $abs(.) \geq something$
   (a) With slack variables
   (b) With a smooth version of abs()
3. Study the influence of various initialization strategies
   (a) Initialization with a trajectory that is dynamically feasible but not safe
   (b) Initialization with a Neural Network heuristic. Based on a previous poroject we already have a Neural Network that can propose a candidate solution. This solution is unsafe in 20% of the cases but could be used as a usefull initializer.
4. If possible, depending on fast we progress with the 3 previous objectives, we will compare optimization over station with optimization over time for 2D paths. In the first case we fix the time steps and derive positions at every time step, while in the second case we fix the spatial positions and derive the time instants at which we go over these spatial positions. In the first case, when dealing with 2D paths, an analytical representation of the path is required while in the second case, a sequence of waypoints is sufficient. But the constraints may be even more non linear and non convex in the later case.

Point 2) issue is avoided in many publications by claiming that a higher level panner decides whether we have to proceed or yield the way. But while this may be a reeasonable strategy when dealing with a single object crossing our path, it does not scale to multiple crossing objects. A more principled approach is required to scale with the complexity of the scene. Point 3) could be a promising combination of techniques: leveraging on offline Neural Network training to speed up online optimization convergence. For Point 4) while some publications focus on one or the other approach, we would like to outline more precisely the pros and cons of these approaches and benchmark their accuracy and convergence speed.

## References

[1] Mykel J. Kochenderfer and Tim A. Wheeler. *Algorithms for Optimization*. The MIT Press, 2019.