About

Meeting note preparation for meeting with TJ on Jan 13, 2023 at ASL

Agenda

1. What I've tried so far
2. Current questions on NPFG
3. Current status (velocity curve formulation)

What I've tried so far

JunwooHwang_WeeklyReports

1. Implemented NPFG without wind handling logic
   a. Simulated PF in wind-less 2D environment in OpenAI Gym
2. Identified a problem with using NPFG on Multicopter purely relying on Track-keeping
   a. where the bearing feasibility interfered with the minimum forward ground speed setpoint
   b. Week 4 - Summary
3. Vector Field visualization has been done on NPFG
   a. 7_NPFG_VectorField_Visualization

Current Questions on NPFG

Summarized in 2_Understanding TJ NPFG (Flying Backwards)

1. Main points
   ~~a. Period and Damping's relationship to vehicle dynamics~~
   ~~b. Period and Damping's relationship to track error boundary~~

Current Status

Summarized in 10_CartesianVelocity RampCurve Visualization

What to optimize for

1. **Time** optimal: Commonly known as jerk-limited trajectory in PX4 (VelocitySmoothing), the bang-bang controller on Jerk ensures that we reach the target (velocity / position) in the least time possible
2. **Jerk** optimal: Given target position & velocity, creates trajectory that minimizes the Mean-Square-Error of the Jerk imposed on the vehicle (smoothest *motion)

Velocity Components

1. **'Vertical'** velocity curve
   a. Fairly straight forward when using **'time**-optimal' jerk limited trajectory, as the target is at a distance of E (track error), with velocity 0 (can be altered, when following curved trajectories)
   b. This results in calculating:
      i. **Track error boundary**: minimum braking distance to bring orthogonal velocity to target velocity, 0 for straight path

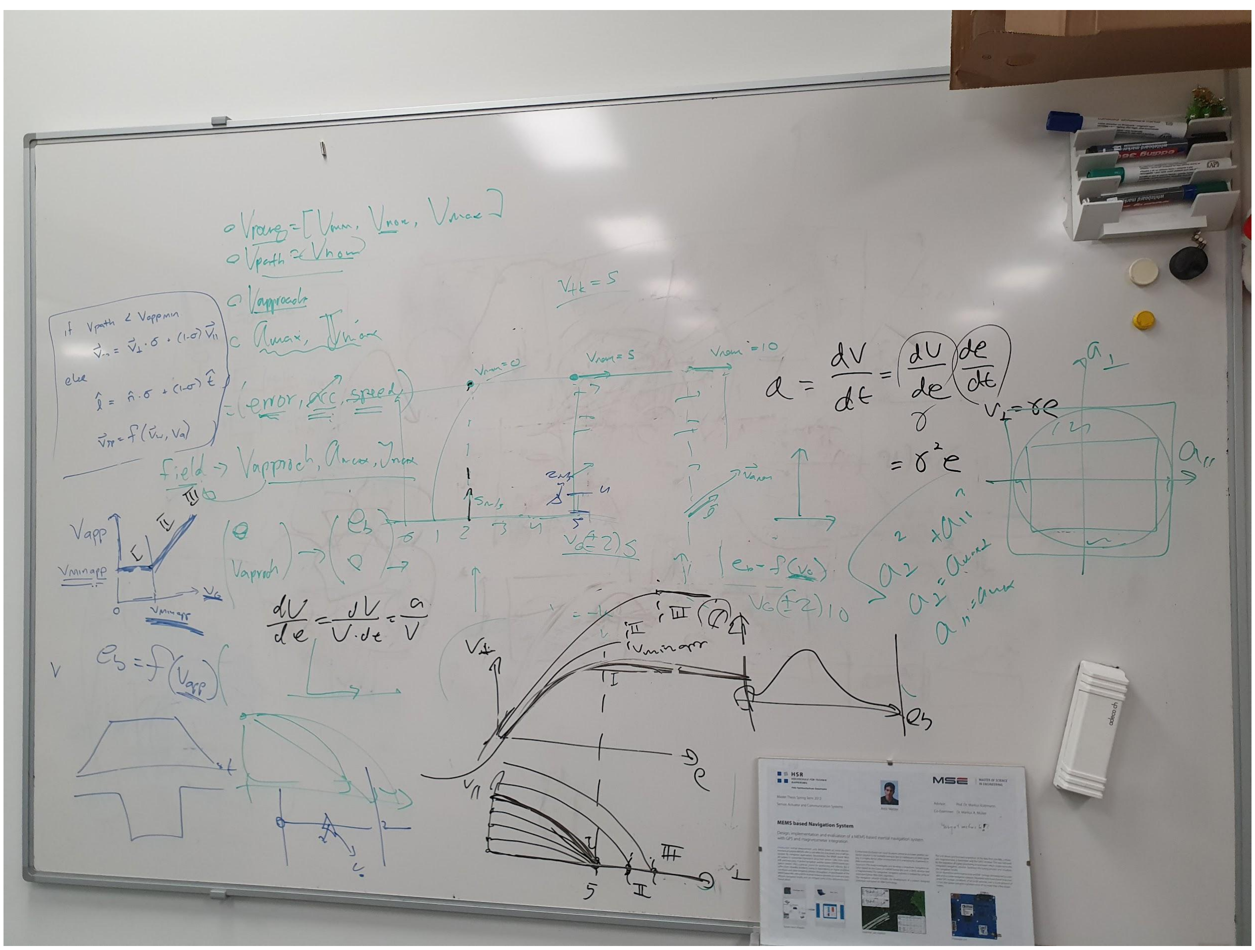
ii. **Position, Velocity, Acceleration, and Jerk** vs. Time curve
1. Which can be used for calculating Vector Field

2. Freedom in **'parallel'** velocity component curve
   a. Unlike 'vertical' position to the path, the parallel position has no solid 'position' setpoint
   b. Therefore, we can have different curve techniques, however:
      i. <u>Syncing the 'position and velocity' target based vertical velocity profile to a purely 'velocity' based horizontal velocity profile will be the challenge</u>
         1. As the duration for those trajectories generated won't match.
2. Parametrization
   a. Approach orthogonal velocity
      i. V_max: Ensures we approach with max speed
      ii. Track keeping velocity:
   b. Parallel velocity
      i. Simply following a less aggressive trajectory matching the timespan of the time-optimal orthogonal velocity curve

Notes

- Having a **'cusp'** on the heading rate, due to the bearing angle formulation around the path (when track error normalized == 0), *probably forced the track error boundary formulation to be linearly scaling to a ground speed (roughly)
- Drawing the '**heading rate (bearing)**' as function of track error will give a view on how the bearing rate changes (and whether it has a 'cusp' around the actual track on path with error == 0)

Conclusions



1. Considering jerk-limited trajectory isn't important now, it should more be focused about having a robust algorithm that can handle Multicopter and Fixed Wing guidance
2. Although the guidance may generate infeasible vector fields, depending on vehicle's state, we should trust the lower level controls to eventually follow the VF, and minimize taking current vehicle's state into PF calculation
3. For now, dealing with 'zero speed on path' case (for Multicopter) would be the ideal case to show that MC can be supported with the new PF algorithm
4. For that, we can specify the **'V_approach_minimum'**, and if vehicle's ground speed towards the path is beneath this speed, we 'saturate' the track error boundary, so that it never reaches 0 (which can cause division by 0, when calculating 'normalized' track error)
    a. If approaching faster than V_approach_minimum:
        i. We can increase the track error boundary, and apply look-ahead-angle variation based bearing setpoint generation
    b. If slower than V_approach_minimum:
        i. We can clip the track error boundary, and form VF that decreases it's orthogonal velocity component smoothly as track error reaches 0
        ii. Also, the V_path can be slowly ramped in, or this can be scaled up/down
5. Track Error Boundary can be considered to be linearly increasing relative to approaching ground speed (which is clipped in it's lower bound by V_approach_minimum)
6. Watch out & observe what happens around when track error is close to 0, there may be 'cusps', that generates aggressive VF, which can cause instability
7. Read the stability proof document, to understand theory behind period & damping ratio calculation of the NPFG algorithm