Model Predictive Control for Autonomous Flights of Drones

- Mathematical Models, System Developments and Tests -

M.Sc. Felix Vortisch and Prof. Dr.-Ing. Reiner Jäger

The presentation is dealing with the development, simulations and tests of a Model Predictive Control (MPC) flight control for the autonomous flight of configurable drones along a defined trajectory.

The computation of the control parameters $\mathbf{u}$(t) - e. g. the propeller rotation rates of a propeller drone - is theoretically based on the availability of the full so-called 19 parameter control deviation $\mathrm{d}\mathbf{y}(\mathrm{t}) = \mathbf{y}(\mathrm{t})_{\mathrm{des}} - \mathbf{y}(\mathrm{t})_{\mathrm{nav}}$ (nav = multisensory navigated) due to the below mentioned differential equations of flight physics. In case of a remote control, $\mathrm{d}\mathbf{y}(\mathrm{t})$ is directly read form the slider-/joystick-based control settings of the pilot. Here the PID control has become a standard to compute continuously the control variables $\mathbf{u}$(t) at any time $\mathrm{t_i}$.

To fly however autonomously along a desired, in first instance only waypoint-based defined trajectory $\bar{\mathbf{y}}(\mathrm{t})_{\mathrm{des}}$, the PID control algorithm becomes inappropriate, as the desired navigation state parameters $\mathbf{y}(\mathrm{t})_{\mathrm{des}}$ contains 19 parameters. The associated desired way position polygon $\bar{\mathbf{y}}(\mathrm{t})_{\mathrm{des}}$ is however only a 3D sub-vector of $\mathbf{y}(\mathrm{t})_{\mathrm{des}}$. So there is an large uncertainty, how to “fill up” the missing parameters $\mathbf{y}(\mathrm{t})_{\mathrm{des,rest}}$, complementary to $\bar{\mathbf{y}}(\mathrm{t})_{\mathrm{des}}$, in order be able to solve the differential equation for getting $\mathbf{u}(\mathrm{t})$. This goes hand in hand with the fact that “any” defined trajectory $\mathbf{y}(\mathrm{t})_{\mathrm{des}}$ is not guaranteed to be flyable, means controllable, due to the physical characteristics of the inertia matrix $\mathbf{J}$ and the motor allocation matrix **M**. **M** is giving the relation (linear in case of propeller drone) between the above controls $\mathbf{u}(\mathrm{t})$ and the resulting thrusts and torques in body frame of the drone within the differential equation below.

The MPC avoids all above complications in an autonomous flight. Common with the PID control, the MPC algorithm is based on the unique first order flight physics differential equation $\dot{\mathbf{y}}(\mathbf{t}) = \mathbf{f}(\mathbf{y}(\mathbf{t}), \mathbf{J}, \mathbf{M}, \mathbf{u}(\mathrm{t}))$ of a flight object to be controlled. The total vector $\mathbf{u}(\mathrm{t}) = [\mathbf{u}(\mathrm{t}_1), .., \mathbf{u}(\mathrm{t_j}), .., \mathbf{u}(\mathrm{t_m})]$ of the control parameters is in the MPC case piecewise predicted along the desired waypoint trajectory $\bar{\mathbf{y}}(\mathrm{t})_{\mathrm{des}}$ in terms of MPC sequences $\mathbf{u}(\mathrm{t_j})$ over control horizons $\Delta\mathrm{T_j} = [\mathrm{t}_1, .., \mathrm{t_i}, .. \mathrm{t_n}]_\mathrm{j}\,, \mathrm{j} = 1, \mathrm{n_j}$ in time spans $\Delta\mathrm{T_j}$. So it holds for the total control vector $\mathbf{u}(\mathrm{t}) = [\mathbf{u}(\mathrm{t}_1),\ .., \mathbf{u}(\mathrm{t_j}), ..., \mathbf{u}(\mathrm{t_m})]$. The above-mentioned uncertainties on the complementary parameters $\mathbf{y}(\mathrm{t})_{\mathrm{des,rest}}$ can be overcome by the MPC method, as well as the problem of flyability of a defined trajectory $\mathbf{y}(\mathrm{t})_{\mathrm{des}}$ **,** in respect to get a complete set of unique control parameters $\mathbf{u}(\mathrm{t}) = [\mathbf{u}(\mathrm{t}_1),\ .., \mathbf{u}(\mathrm{t_j}), ..., \mathbf{u}(\mathrm{t_m})]$ along $\mathbf{y}(\mathrm{t})_{\mathrm{des}}$**.**

In the first part of the presentation, the principle of the MPC algorithm is explained, by the derivation of the mathematical model for the computation of sets of controls $\mathbf{u}(\mathrm{t_j})$ over predicted time-horizon $\Delta\mathrm{T_j}$. The solution for $(\mathrm{t_j})$ leads to the introduction of a quadratic cost function, which has to be minimized to receive the controls $\mathbf{u}(\mathrm{t_j})$. The MPC method leads to a $\mathbf{Q}_\mathrm{i}$-weighted result with theoretical residuals $\mathrm{d}\mathbf{y}(\mathrm{t})$ between the controlled and the desired $[\mathbf{y}(\mathrm{t}_1)_{\mathrm{des}}, ., \mathbf{y}(\mathrm{t_i})_{\mathrm{des}}, ., \mathbf{y}\left(\mathrm{t_{n_j}}\right)_{\mathrm{des}}]_\mathrm{j}\,, \mathrm{j} = 1, \mathrm{n_j}$ and so in $[\bar{\mathbf{y}}(\mathrm{t}_1)_{\mathrm{des}}, ., \bar{\mathbf{y}}(\mathrm{t_i})_{\mathrm{des}}, ., \bar{\mathbf{y}}\left(\mathrm{t_{n_j}}\right)_{\mathrm{des}}]_\mathrm{j}\,, \mathrm{j} = 1, \mathrm{n_j}$ over each time horizon $\Delta\mathrm{T_j}$. Further residuals $\mathrm{d}\mathbf{y}(\mathrm{t})'$ between the desired $\mathbf{y}(\mathrm{t})_{\mathrm{des}}$ and the navigated $\mathbf{y}(\mathrm{t})_{\mathrm{nav}}$ have to be handled from time to time. In the MPC algorithm the matrices **J** and **M** are set as free configurable parameters, so any type of drones can be configured.

The final part of the presentation is dealing with the implementation of the MPC flight control using python and toolkit ACADOS. The implementation of the MPC flight control is validated and tested under simulation conditions by hardware in the loop (HIL) and software in the loop (SIL). The performance and robustness have been tested and discussed, and the validated MPC flight control system has been adapted for a real-world applications.