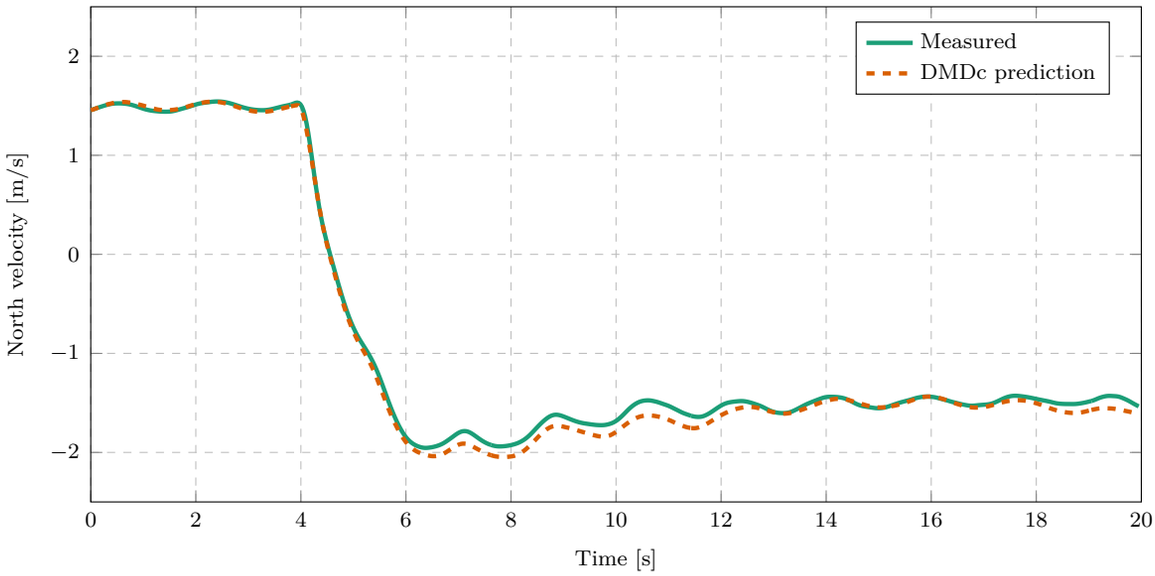

Figure 7.18: Practical flight data and model predictions with an elongated payload for a North velocity step input ($m_1 = 0.2\,\text{kg}$, $l_1 = 0.5\,\text{m}$, $m_2 = 0.1\,\text{kg}$, $l_2 = 0.6\,\text{m}$)

Recall that the prediction is propagated from an initial condition using the given input data only. The model does not use state measurements to readjust after the initial condition is taken. Therefore an accumulation in prediction error is expected as the prediction horizon increases. Because the model prediction matches the shape of the practical testing data so closely, it is expected that this model can be used for a practical MPC implementation on practical data.

7.3. ~~HITL~~ Hardware-In-The-Loop (HITL)

In Section 7.2 it was shown that DMDc can generate accurate state prediction models from practical flight data. It was also shown in Section 5.4 that the DMDc models can be used in an MPC for effective swing-damping control. A HITL simulation can now be used to test the complete control architecture with the final software running on the actual hardware.

As described in Section 6.4, a companion computer runs a ROS node which implements the MPC algorithm. The MPC is based on a DMDc plant model generated from training data from a HITL simulation. The companion computer receives state feedback and sends control signals via a Universal Asynchronous Receiver Transmitter (UART) connection to the Flight Controller (FC). The FC runs the actual PX4 flight stack firmware and executes the control signals received from the MPC node. Sensor values are generated and sent to PX4 by the Gazebo simulator, which runs on a desktop computer connected to the FC via Universal Serial Bus (USB). In this way, we can safely determine whether the practical hardware is suitable for the computational complexity of the control algorithms.

In this section, the results of different HITL simulations will be shown and discussed. Firstly, the effect of the plant model size on the Central Processing Unit (CPU) and Random Access Memory (RAM) consumption of the MPC node will be investigated. The effect of the MPC sample time on the CPU consumption will also be investigated. Finally, it will be shown that an MPC node, which is based on a data-driven system identification model, effectively implements swing-damping control in a HITL simulation.

7.3.1. Effect of hyperparameters on computational requirements

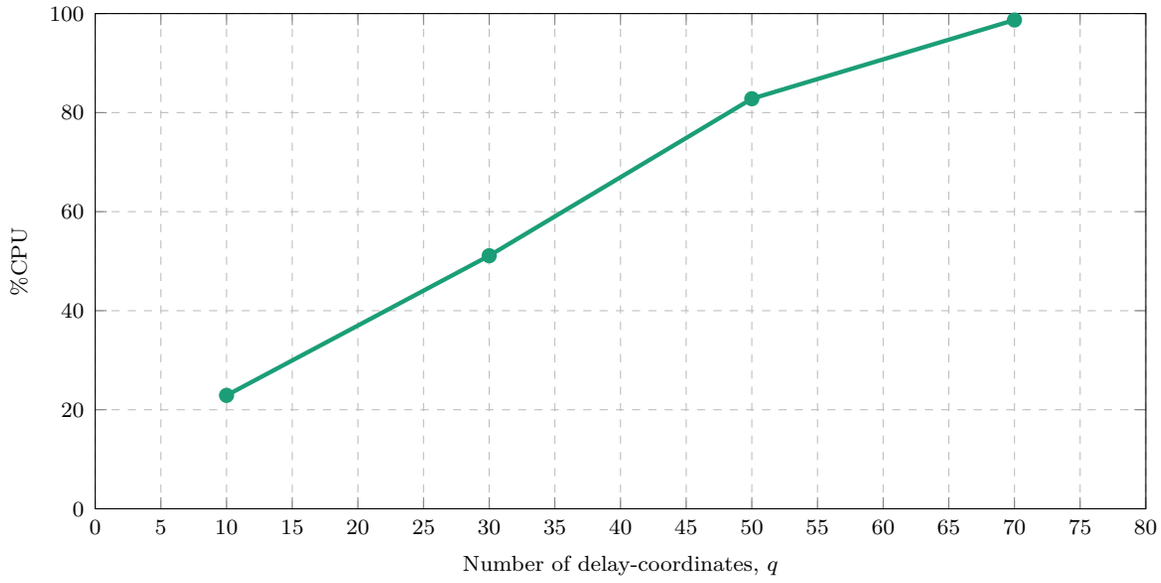

Figure 7.19: Maximum %CPU used by the MPC node for different values of q ($Ts = 0.03\,\mathrm{s}$)

As discussed in Section 5.4, the computational complexity of the MPC optimisation problem increases for larger state-space matrices, and the hyperparameter, q, determines the size of these matrices. Figure 7.19 shows the %CPU consumption of MPC nodes based on models with different q values. The %CPU value represents the average percentage of CPU time used by the MPC node running on the companion computer. From Figure 7.19, it is clear that %CPU increases with increasing values of q. This shows that the computational complexity increases for larger values of q.

In Section 4.6.3, it was shown that $q = 50$ is near the Pareto optimum for practical data and the prediction accuracy does not increase significantly for $q > 50$. As shown in Figure 7.19, an MPC node with $q = 50$ and $T_s = 0.03\,\mathrm{s}$ run at %CPU = 82.8% on the companion computer. The status of the QP solver was also monitored, which showed that the optimisation problem was consistently solved within the given optimisation time for $q = 50$. The MPC node achieves stable, swing-damping control with this model. However,

for $q = 70$ the MPC node uses %CPU = 98.7% and the optimisation problem can not be solved fast enough for stable quadrotor control. This results in an unstable controller and the quadrotor-payload system crashes consistently with this MPC node.

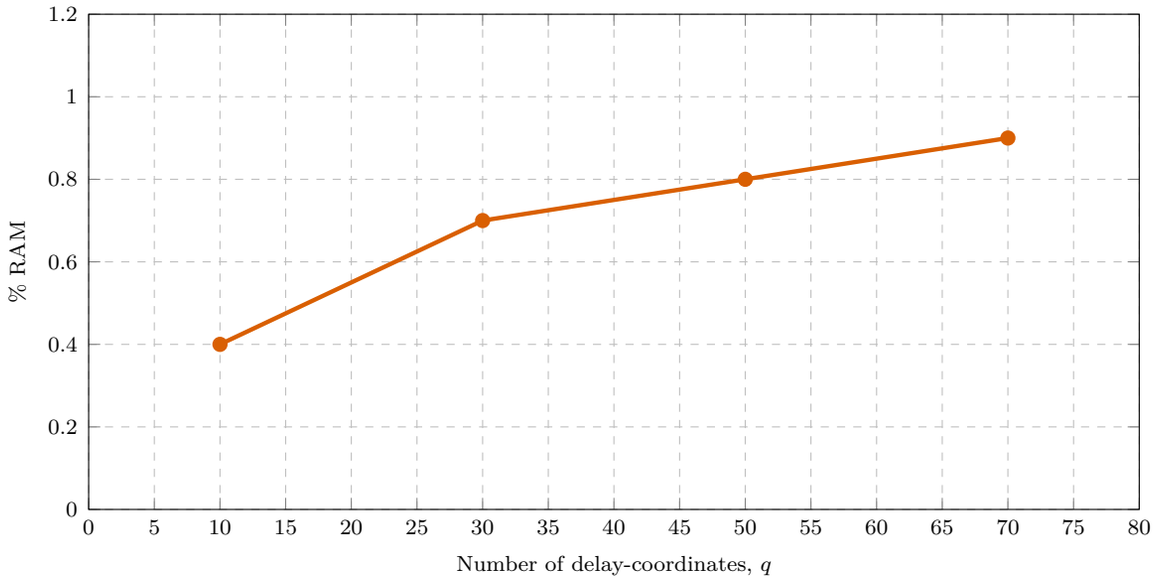

Figure 7.20: Maximum % RAM used by the MPC node for different values of q

Figure 7.20 shows that the %RAM also increases for larger values of q. The %RAM value represents the maximum percentage of RAM space used by the MPC node while running on the companion computer. It is clear that the companion computer has sufficient memory to handle the MPC, since the MPC node uses less than 1% RAM, even for large models with $q = 70$.

7.3.2. Effect of sample time on computational requirements

The sample time of the controller also affects the %CPU consumption of the MPC node. Figure 7.21 shows the measured %CPU of MPC nodes with different sample times using a plant model with $q = 50$. It is clear that %CPU decreases as T_s increases. The default velocity controller in PX4 runs at 50 Hz, which corresponds to $T_s = 20$ ms. However, an MPC node with $T_s = 20$ ms struggles to run fast enough on the given companion computer. This is clear from the high %CPU consumption (98.6%) of the MPC node running at $T_s = 20$ ms. This node results in unstable control because the QP problem cannot be solved within the given optimisation time.

A MPC node with $T_s = 40$ ms runs with a low %CPU, however it also results in unstable control. This is because the controller frequency is too low to provide adequate velocity control of the quadrotor-payload dynamics. However, an MPC node with $T_s = 30$ ms provides stable control of the quadrotor-payload system. The QP problem is solved within

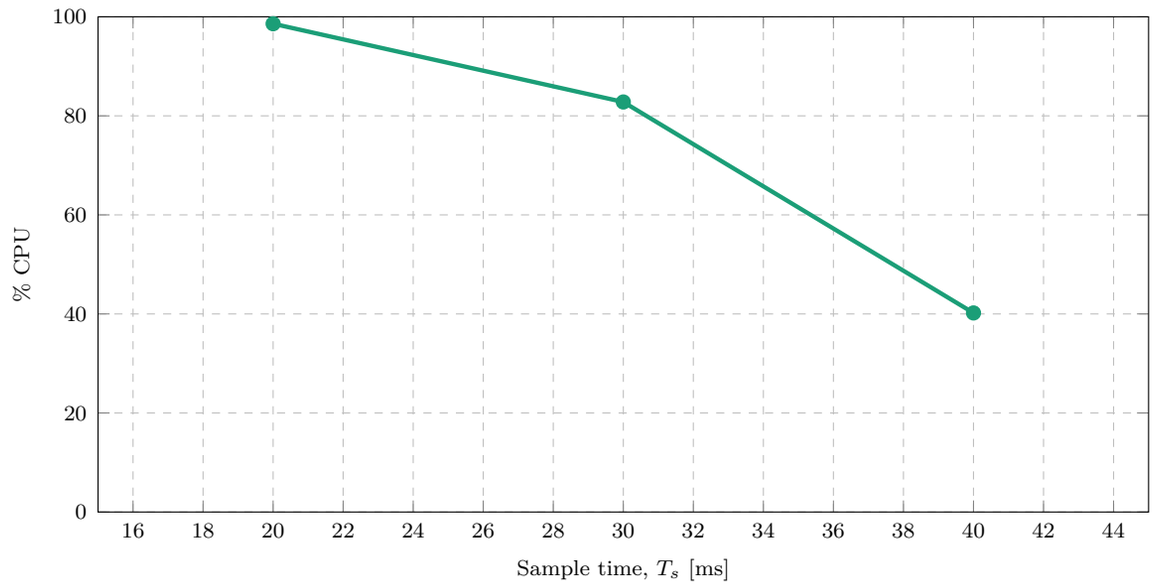

Figure 7.21: Maximum % CPU used by the MPC node for different sample times ($q = 50$)

the given optimisation time and, as shown in the section below, the resulting controller frequency of 33.33 Hz appears to be fast enough to control the quadrotor-payload dynamics.

7.3.3. Velocity step response

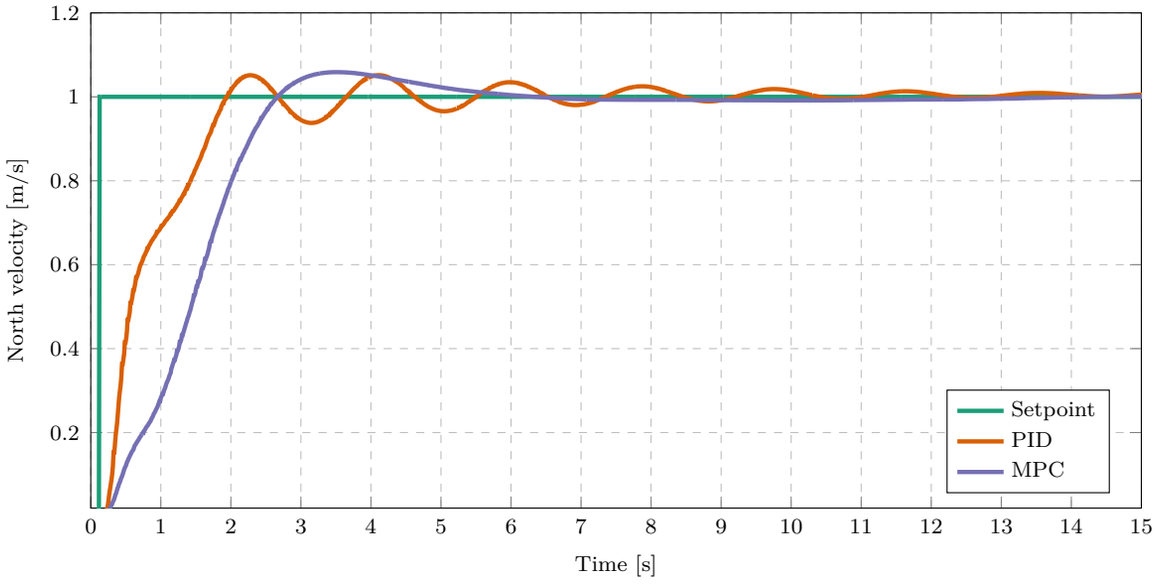

Figure 7.22: Velocity step responses of MPC and PID controllers for HITL simulations ($q = 50$, $T_s = 0.03\,\text{s}$)

Figure 7.22 plots the velocity responses of the PID and MPC controllers in HITL simulations of the quadrotor-payload system. The MPC controller is based on a system identification model with $q = 50$ and ~~has~~ a sample time of $T_s = 30\,\text{ms}$. It is clear from this plot that the

MPC node running on the companion computer provides stable control of the quadrotor payload system and damps the velocity oscillations well.

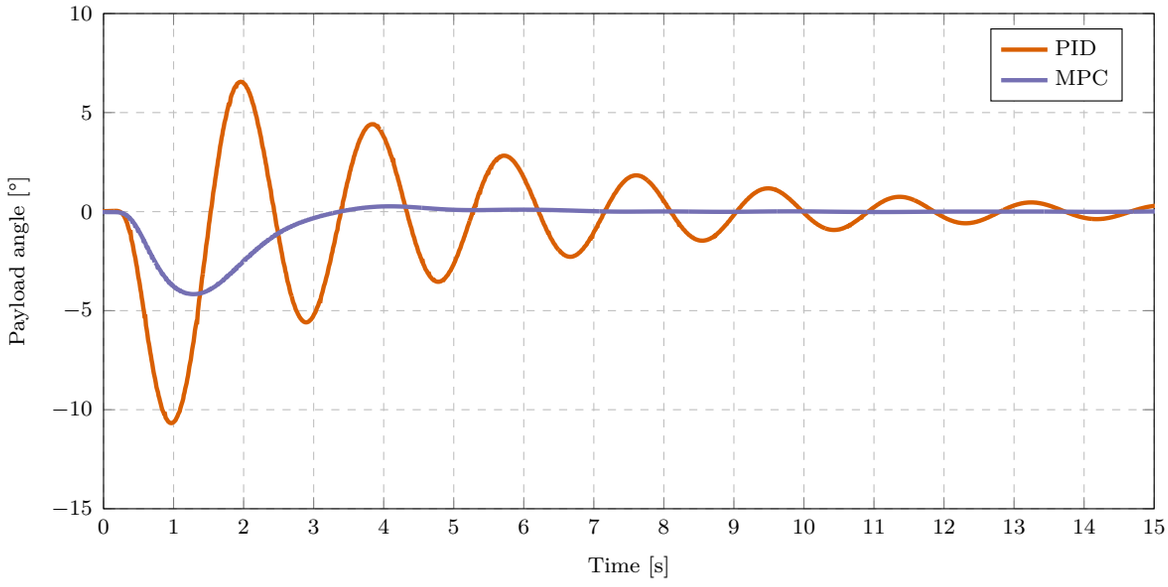

Figure 7.23: Payload angle responses of MPC and PID controllers for HITL simulations ($q = 50$, $T_s = 0.03\,\mathrm{s}$)

Figure 7.23 shows the payload angle data of the PID and MPC controllers during the velocity step response. From this plot, it is visible that the MPC controller damps the payload angle well. Overall, these plots show that the swing-damping control achieved in simulations in Section 5.4, is also achievable with the final controller software running on the actual hardware.

7.4. Summary

In this chapter, the system identification techniques were applied to practical flight data. The cable length estimation technique described in Chapter 4 was applied to data from flights with different payload masses and cable lengths. The results showed that the estimated length converged quickly for each payload configuration and that the estimation error was consistently quite large. The large error was attributed to cable attachment being below the vehicle CoM, and the damping effect of the controller, which were not considered in the parameter estimation algorithm.

It was also shown that the estimated length converged for flight data with different wind conditions and that the estimation error was the largest for the data with the most wind. Furthermore, the cable length estimator was applied to flight data with a double-pendulum payload. The estimated value converged to a length corresponding to the dominant frequency in the data, and the resulting white-box model prediction only captured the

general shape of the data well. However, the white-box model prediction was shown to be very sensitive to initial conditions.

The data-driven system identification techniques were also successfully applied to practical flight data. As expected, the prediction error of the techniques decreased for flights with lower wind speeds. With a reasonable amount of wind, the techniques were able to generate accurate prediction models. However, at very high wind speeds, the resultant models were not usable.

The DMDc and HAVOKc state predictions performed equally well on data from various flights. DMDc was chosen as the preferred method due to its lower complexity. It was shown that DMDc could be applied to data from a wide range of different payload parameters and that it consistently generated accurate prediction models without prior knowledge of the payload configuration. The Pareto front and 'double-descent' trends discussed in Chapter 4 were also identified in these results.

DMDc was also extended to capture the North and East velocity dynamics simultaneously, and the resulting prediction model accurately reconstructed the system dynamics for test data. Furthermore, DMDc was applied to flight data with a dynamic payload representing a double-pendulum system. DMDc accurately reconstructed the dynamics of this system for unseen test data. Unlike the white-box model which only represented the dominant oscillation frequency, the DMDc model also predicted the irregular oscillations well.

Finally, the full data-driven system identification with MPC control architecture was tested in HITL simulations. It was shown that the hardware successfully executes the MPC algorithm at the desired speed for the selected hyperparameters and sample time. The successful velocity step response proved that the various software systems work together seamlessly. It also showed that the control architecture can be implemented on the hardware of an actual multirotor to successfully apply swing-damping control of the quadrotor-payload system.