Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-21 (2021) 210–215

Data-Driven System Identification and Model Predictive Control of a Multirotor with an Unknown Suspended Payload

J.M. Louw * **H.W. Jordaan** **

* *Electrical and Electronic Engineering Department, Stellenbosch University, South Africa (email: 19977476@sun.ac.za)*

** *Electrical and Electronic Engineering Department, Stellenbosch University, South Africa (email: wjordaan@sun.ac.za)*

Abstract: Transporting a suspended payload with a multirotor has many applications. Knowledge of the payload dynamics is required to adjust the control of the system to damp payload oscillations. Often, the dynamics of the payload are unknown and cannot be represented with an a priori model before a flight. This paper proposes a Model Predictive Control (MPC) architecture that controls a multirotor carrying an unknown suspended payload using a plant model from data-driven system identification techniques. Dynamic Mode Decomposition with Control (DMDc) and Hankel Alternative View Of Koopman with Control (HAVOKc) are the regression techniques used to identify system models without relying on modelling assumptions and by using only time series measurements. The standard Hankel Alternative View Of Koopman (HAVOK) is adapted slightly in this work for use with controlled systems. These two techniques are combined in the MPC architecture and compared against a conventional Proportional Integral Derivative (PID) system to control a multirotor with an unknown suspended payload within simulation. The results show that both MPC systems outperform the conventional system and achieve velocity control while simultaneously damping the payload swing angle. The proposed systems also show good adaptability with different payload parameters. Both system identification methods perform well with the presence of measurement noise.

Keywords: Modelling and system identification, Aerospace systems, Robotic systems, Data-driven Control, Model Predictive Control

1. INTRODUCTION

Transporting goods with multirotor vehicles has gained considerable interest over the last few years. Most commercial cases attach a payload rigidly to the vehicle so the payload does not significantly affect the flight dynamics of the multirotor. However, this greatly limits the shape and size of the payload that can be grasped. Attaching the payload with a suspended cable increases the variety of use cases, but changes the flight dynamics considerably, with the suspended payload now free to swing.

Two common ways of dealing with this problem are input-shapers that generate minimum oscillation trajectories, or active vibration damping controllers. Generating minimum oscillation trajectories requires an accurate model of the entire system, including the payload, to be effective. Active damping controllers are more robust to model uncertainty, but they require knowledge of the payload state to actively damp the payload oscillations.

Standard adaptive control uses the difference between the current output and the expected behaviour, and automatically adjusts the controller with some adapting law. In practical tests, these techniques handle unknown parameters well (Dai et al., 2014), but they are still dependent on prior knowledge of the expected system dynamics. A model of the payload can also be used within a Model Predictive Control (MPC) architecture, which uses an optimizer to determine optimal input commands to achieve a specified goal. This was also shown to be feasible for controlling a multirotor with a payload in practical tests (Notter et al., 2016).

Common methods to determine the model of a system from state measurements include the Prediction Error Method, Maximum Likelihood, Least-Squares estimation or the Extended Kalman Filter. Many of these techniques require prior modelling of the dynamics to then estimate parameters to determine a model. However, these modelling assumptions can detract from the accuracy of the estimated model in the presence of unexpected payload complexity (such as an elongated payload, sloshing liquid in a container, or a cable that stretches). Modern data-driven techniques such as Dynamic Mode Decomposition with Control (DMDc) (Proctor et al., 2016), Hankel Alternative View Of Koopman (HAVOK) (Brunton et al., 2017; Champion et al., 2019), Sparse Identification of Nonlinear Dynamics (SINDy) (Brunton et al., 2016) and even Neural Networks, obtain a model without prior modelling assumptions. SINDy, for example, was shown to be an effective non-linear data-driven method for use in a MPC control scheme for an aerial vehicle (Kaiser et al., 2018).

10.1016/j.ifacol.2021.12.036

This paper proposes an MPC architecture with a linear data-driven system identification algorithm for a multirotor carrying an unknown swinging payload. Two methods are compared for determining the unknown model. DMDc is a linear regression technique that determines a model from time series measurements. HAVOK is another linear regression technique with connections to DMDc and Koopman operator theory. In our application, HAVOK is further extended to be useful for the control problem. Within simulations, these two techniques are applied separately in a MPC architecture to control a multirotor and are compared to a conventional PID control system.

2. MODELLING

The simulation models used in this paper are based on a practical quadrotor that was build and commissioned by Erasmus (2020). This multirotor has a mass of 4.5 kg and a motor-to-motor distance of 980 mm. For this paper the multirotor is modelled in two dimensional (2D) space with motion constrained in the xz-plane, as shown in Fig. 1. The North-East-Down (NED) inertial frame is indicated by $\mathcal{I} = \{\boldsymbol{x}_\mathcal{I}, \boldsymbol{x}_\mathcal{I}, \boldsymbol{z}_\mathcal{I}\}$. The vehicle is free to translate in the $\boldsymbol{x}_\mathcal{I}$ and $\boldsymbol{z}_\mathcal{I}$ axes and rotate about the $\boldsymbol{x}_\mathcal{I}$ axis. Two motors on one side of the quadrotor are modelled as a single thrust, T_1, and similarly the other two as T_2.

In Fig. 1, x_Q and z_Q define the position of the multirotor in the inertial frame and θ is the pitch angle of the multirotor. The payload swing angle is denoted by β, the payload mass by m_P, and the link length by l. Important modelling assumptions include that the payload is a point mass, the payload link is massless and rigid, and the link is attached to the Centre of Mass (CoM) of the vehicle. The quadrotor with payload is modelled as a single dynamical system, therefore its state vector is defined as, $\boldsymbol{x} = \begin{bmatrix} x_Q & z_Q & \theta & \beta & \dot{x}_Q & \dot{z}_Q & \dot{\theta} & \dot{\beta} \end{bmatrix}^T$, and its input vector as, $\boldsymbol{u} = [T_1 \ T_2]^T$. The Lagrangian is given by,

$$\mathcal{L} = \frac{1}{2} m_Q \left(\dot{x}_Q^2 + \dot{z}_Q^2\right) + \frac{1}{2} m_P \left(\dot{x}_P^2 + \dot{z}_P^2\right) + m_Q g z_Q + m_P g z_P, \tag{1}$$

where x_P and z_P define the inertial position of the payload, m_P is the payload mass, m_Q is the quadrotor mass, and g is the gravitational acceleration constant. Hereafter, the Euler Lagrange equation is defined as:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{p}}}\right) - \frac{\partial \mathcal{L}}{\partial \boldsymbol{p}} = \begin{bmatrix} -(T_1 + T_2)\sin(\theta) \\ -(T_1 + T_2)\cos(\theta) \\ T_2 r - T_1 r \\ -c\dot{\beta} \end{bmatrix}, \tag{2}$$

where r is the distance from T_1 and T_2 to the CoM of the quadrotor, c is the damping coefficient of the link connection, and $\boldsymbol{p} = [x_Q \ z_Q \ \theta \ \beta]^T$ contains the non-velocity states. For the simulation model, aerodynamic drag is also modelled for the quadrotor and payload, but it is omitted here for simplicity. Finally, Equation (2) is solved to obtain the differential equations, $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$, that describe the dynamics of the system.

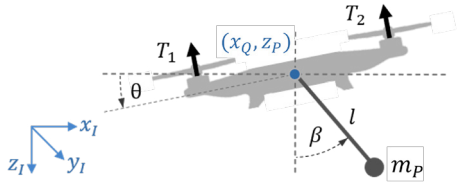

Fig. 1. 2D schematic of a quadrotor with a suspended payload

3. SYSTEM IDENTIFICATION

When the dynamics of a system can be derived a priori, parameter estimation methods can populate a predetermined model for model-based control of an unknown system. This approach was demonstrated for a multirotor with an unknown suspended payload by Erasmus (2020). However, when the actual dynamics of a payload differs significantly from the predetermined model, it may have an detrimental effect on a controller based on this model. Therefore, the system identification technique needs to be manually redesigned for different types of payloads. In contrast, data-driven methods do not rely on a priori modelling and provide an adaptable solution for a wide range of different payloads. DMDc and Hankel Alternative View Of Koopman with Control (HAVOKc) are the data-driven system identification methods investigated in this paper. These are linear regression techniques that produce a linear model to approximate non-linear dynamics. Non-linear data-driven techniques like Neural Networks and SINDy (Brunton et al., 2016) may produce models that are more accurate, however they have greater computational complexity. DMDc and HAVOKc are less computationally complex and their models are suitable for linear MPC, which is significantly faster than non-linear MPC. This is desirable for a multirotor use case, where onboard computational power is limited.

3.1 Dynamic Mode Decomposition with Control

Dynamic Mode Decomposition (DMD) is a linear regression technique that will be used here to approximate a non-linear dynamical system (Tu et al., 2014). It uses only time series measurements to reconstruct the system dynamics, with no prior modelling assumptions. DMDc is an adaptation of DMD that also accounts for control (Proctor et al., 2016).

This section provides an overview of the specific implementation of DMDc used in this paper. Note that DMDc is adapted for this work to include time-delay-embedding of measurement variables. Korda and Mezić (2018) and Arbabi and Mezić (2017) use time-delay-embedding in their DMD adaptions in similar ways.

Only the non-velocity state variables, $\boldsymbol{x} = [x_Q \ z_Q \ \theta \ \beta]^T$, of the 2D multirotor and payload system are used in this implementation. The algorithm produces a discrete state-space model, hence discrete instances, $\boldsymbol{x}_k$, of the continuous time variable, $\boldsymbol{x}(t)$, are used, where $\boldsymbol{x}_k = \boldsymbol{x}(kT_s)$, and T_s is the sampling time of the model. Delay-coordinates (i.e. $\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}$, etc.) are also included in the state-space model to account for input delay and state delay in the system. Therefore, we define a state delay vector as:

$$\boldsymbol{d}_k = \begin{bmatrix} \boldsymbol{x}_{k-1} & \boldsymbol{x}_{k-2} & \cdots & \boldsymbol{x}_{k-q+1} \end{bmatrix}^T, \tag{3}$$

where q is the number of delay-coordinates (including the current time-step) used in the model, and $\boldsymbol{d}_k \in \mathbb{R}^{(n_x)(q-1)}$.

The discrete state-space model is defined as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{A}_d\boldsymbol{d}_k + \boldsymbol{B}\boldsymbol{u}_k, \tag{4}$$

where $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}$ is the system matrix, $\boldsymbol{A}_1 \in \mathbb{R}^{(q-1)\cdot n_x \times (q-1)\cdot n_x}$ is the state delay system matrix and $\boldsymbol{B} \in \mathbb{R}^{n_x \times n_u}$ is the input matrix. The training data consists of discrete measurements, $\boldsymbol{x}_k$, and corresponding inputs, $\boldsymbol{u}_k$, taken at regular intervals of $\Delta t = T_s$, during a simulated flight. Conventional Proportional Integral Derivative (PID) controllers are used for training flights. The training data is collected into the following matrices:

$$\begin{aligned}
\boldsymbol{X'} &= \begin{bmatrix} \boldsymbol{x}_{q+1} & \boldsymbol{x}_{q+2} & \boldsymbol{x}_{q+3} & \cdots & \boldsymbol{x}_{w+q} \end{bmatrix}, \\
\boldsymbol{X} &= \begin{bmatrix} \boldsymbol{x}_{q} & \boldsymbol{x}_{q+1} & \boldsymbol{x}_{q+2} & \cdots & \boldsymbol{x}_{w+q-1} \end{bmatrix}, \\
\boldsymbol{X}_d &= \begin{bmatrix} \boldsymbol{x}_{q-1} & \boldsymbol{x}_{q+0} & \boldsymbol{x}_{q+1} & \cdots & \boldsymbol{x}_{w+q-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{2} & \boldsymbol{x}_{3} & \boldsymbol{x}_{4} & \cdots & \boldsymbol{x}_{w+1} \\ \boldsymbol{x}_{1} & \boldsymbol{x}_{2} & \boldsymbol{x}_{3} & \cdots & \boldsymbol{x}_{w} \end{bmatrix}, \\
\boldsymbol{\Upsilon} &= \begin{bmatrix} \boldsymbol{u}_{q} & \boldsymbol{u}_{q+1} & \boldsymbol{u}_{q+2} & \cdots & \boldsymbol{u}_{w+q-1} \end{bmatrix},
\end{aligned} \tag{5}$$

where w is the number of columns in the matrices, $\boldsymbol{X'}$ is the matrix $\boldsymbol{X}$ shifted forward by one time-step, $\boldsymbol{X}_d$ is the matrix with delay states, and $\boldsymbol{\Upsilon}$ is the matrix of inputs. The equation in (4) can now be combined with the matrices in (5) to produce:

$$\boldsymbol{X'} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{A}_d\boldsymbol{X}_d + \boldsymbol{B}\boldsymbol{\Upsilon}. \tag{6}$$

Note that the primary objective of DMDc is to determine the best fit model matrices, $\boldsymbol{A}$, $\boldsymbol{A}_d$ and $\boldsymbol{B}$, given the data in $\boldsymbol{X'}$, $\boldsymbol{X}$, $\boldsymbol{X}_d$, and $\boldsymbol{\Upsilon}$ (Proctor et al., 2016). In order to group the unknowns into a single matrix, (4) is manipulated into the form,

$$\boldsymbol{X'} = [\boldsymbol{A}\ \boldsymbol{A}_d\ \boldsymbol{B}] \begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{X}_d \\ \boldsymbol{\Upsilon} \end{bmatrix} = \boldsymbol{G}\boldsymbol{\Omega}, \tag{7}$$

where $\boldsymbol{\Omega}$ contains the state and control data, and $\boldsymbol{G}$ represents the system and input matrices. A Singular Value Decomposition (SVD) is then preformed on $\boldsymbol{\Omega}$ resulting in: $\boldsymbol{\Omega} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$. Often, only the first p columns of $\boldsymbol{U}$ and $\boldsymbol{V}$ are required for a good approximation (Brunton et al., 2017), hence the SVD is used in the truncated form:

$$\boldsymbol{\Omega} \approx \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T, \tag{8}$$

where $\tilde{}$ represents rank-p truncation. In many cases, the truncated form results in better approximations from noisy measurements than the exact form, since the effect of measurement noise is mostly captured by the truncated columns of $\boldsymbol{U}$ and $\boldsymbol{V}$. By truncating these columns, the influence of noise in the regression problem is reduced.

Now, by combining (8) with the over-constrained equality in (7), the least-squared solution, $\boldsymbol{G}$, can be found with:

$$\boldsymbol{G} \approx \boldsymbol{X'}\tilde{\boldsymbol{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\boldsymbol{U}}. \tag{9}$$

$\boldsymbol{G}$ can now be decomposed into: $\boldsymbol{G} = [\boldsymbol{A}\ \boldsymbol{A}_d\ \boldsymbol{B}]$ according to the required dimensions of each matrix. Thereby, the state-space model approximated by is complete.

3.2 Hankel Alternative View Of Koopman with Control

HAVOK is a data-driven, regression technique that provides a strong connection between DMD and Koopman operator theory (Brunton et al., 2017; Champion et al., 2019). We have adapted the standard HAVOK algorithm slightly to account for the effect of control. This adaptation will be referred to as HAVOKc and produces a discrete, linear model that approximates a controlled dynamical system. A brief overview of HAVOKc will be provided here.

A defining characteristic of HAVOKc is that it uses multiple delay-coordinates (i.e. $\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}$, etc.) in the system identification process. To fit this into the standard state-space format, an extended state vector is defined as:

$$\boldsymbol{a}_k = \begin{bmatrix} \boldsymbol{x}_{k-(q-1)} & \cdots & \boldsymbol{x}_{k-1} & \boldsymbol{x}_k \end{bmatrix}^T, \tag{10}$$

where $\boldsymbol{a}_k \in \mathbb{R}^{(n_y)(q)}$, and the subscript of $\boldsymbol{a}$ denotes the highest subscript of $\boldsymbol{x}$ in the vector. The resulting discrete state-space model is therefore in the form,

$$\boldsymbol{a}_{k+1} = \boldsymbol{A}_H\boldsymbol{a}_k + \boldsymbol{B}_H\boldsymbol{u}_k, \tag{11}$$

where $\boldsymbol{A}_H \in \mathbb{R}^{(q\cdot n_x)\times(q\cdot n_x)}$ is the system matrix, and $\boldsymbol{B}_H \in \mathbb{R}^{(q\cdot n_x)\times n_u}$ is the input matrix. Here, the subscript of $\boldsymbol{A}_H$ and $\boldsymbol{B}_H$ is used to differentiate these matrices from $\boldsymbol{A}$ and $\boldsymbol{B}$ used in DMDc.

The original HAVOK algorithm, developed by Brunton et al. (2017), constructs a Hankel matrix from output variables only. In order to incorporate the effect of control, an extended Hankel matrix, $\boldsymbol{\Pi}$, is created by appending a matrix of inputs to a Hankel matrix of measurements:

$$\boldsymbol{\Pi} = \begin{bmatrix} \boldsymbol{a}_q & \boldsymbol{a}_{q+1} & \boldsymbol{a}_{q+2} & \cdots & \boldsymbol{a}_{w+q-1} \\ \boldsymbol{u}_q & \boldsymbol{u}_{q+1} & \boldsymbol{u}_{q+2} & \cdots & \boldsymbol{u}_{w+q-1} \end{bmatrix}, \tag{12}$$

where w is the number of columns in $\boldsymbol{\Pi}$. A truncated SVD of this Hankel matrix results in following approximation:

$$\boldsymbol{\Pi} \approx \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T, \tag{13}$$

where $\tilde{}$ represents rank-p truncation. It is important to note that the model extracted by HAVOKc depends on the choice of hyperparameters, p and q. The number of samples in the training data, $N_{train} = w + q - 1$, also influences the accuracy of the model.

The columns of $\tilde{\boldsymbol{V}}$ represent the most significant principal components of the system dynamics (Kamb et al., 2020). This matrix, $\tilde{\boldsymbol{V}}$, can be considered to contain a time-series of the pseudo-state, $\boldsymbol{v}$, such that $\tilde{\boldsymbol{V}}^T = [\boldsymbol{v}_q\ \boldsymbol{v}_{q+1}\ \cdots\ \boldsymbol{v}_w]$, characterises the evolution of the actual dynamics in an eigen-time-delay coordinate system (Brunton et al., 2017). Consider the following discrete, state-space formulation:

$$\boldsymbol{v}_{k+1} = \boldsymbol{\Lambda}\boldsymbol{v}_k. \tag{14}$$

Recall that DMDc finds a best fit linear operator that directly maps $\boldsymbol{a}_k$ to $\boldsymbol{a}_{k+1}$. Similarly, HAVOKc determines the best fit linear operator $\boldsymbol{\Lambda}$ that maps the pseudo-state $\boldsymbol{v}_k$ to $\boldsymbol{v}_{k+1}$. So, to setup an over-determined equality for (14), $\tilde{\boldsymbol{V}}^T$ is divided into two matrices:

$$\begin{aligned}
\boldsymbol{V}_1 &= \begin{bmatrix} \boldsymbol{v}_q & \boldsymbol{v}_{q+1} & ... & \boldsymbol{v}_{w-1} \end{bmatrix}, \\
\boldsymbol{V}_2 &= \begin{bmatrix} \boldsymbol{v}_{q+1} & \boldsymbol{v}_{q+2} & ... & \boldsymbol{v}_{w} \end{bmatrix},
\end{aligned} \tag{15}$$

where $\boldsymbol{V}_2$ is $\boldsymbol{V}_1$ advanced a single step forward in time. The matrices from Equation (15) are now combined with Equation (14) and the best fit $\boldsymbol{\Lambda}$ is determined with the Moore-Penrose pseudoinverse:

$$\boldsymbol{V}_2 = \boldsymbol{\Lambda}\boldsymbol{V}_1 \qquad \Rightarrow \qquad \boldsymbol{\Lambda} \approx \boldsymbol{V}_1\boldsymbol{V}_1^{\dagger} \tag{16}$$

It can be shown from Equation (13) that Equation (14) is transformed from the eigen-time-delay coordinate system to the original coordinate system as the following:

$$\begin{bmatrix} \boldsymbol{a}_{k+1} \\ \boldsymbol{u}_{k+1} \end{bmatrix} = (\tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}})\boldsymbol{\Lambda}(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}})^{\dagger} \begin{bmatrix} \boldsymbol{a}_k \\ \boldsymbol{u}_k \end{bmatrix}. \tag{17}$$

This form is used to extract $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{B}}$ from the matrix, $(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}})\boldsymbol{\Lambda}(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}})^{\dagger}$, in the following way:

$$\begin{bmatrix} \boldsymbol{a}_{k+1} \\ \boldsymbol{u}_{k+1} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{A}} & \tilde{\boldsymbol{B}} \\ (discarded) \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_k \\ \boldsymbol{u}_k \end{bmatrix}. \tag{18}$$

Note that the matrix entries in Equation (18) that map $\boldsymbol{u}_k$ to $\boldsymbol{u}_{k+1}$ are meaningless for our purposes and are discarded. Some matrix entries in $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{B}}$ are known a priori due to the relative positions of delay coordinates. These are forced to 1 or 0 to improve the prediction performance of the model.

4. FLIGHT CONTROLLER

4.1 Cascaded PID

A cascaded control architecture with linear PID controllers is used as a benchmark control system. The same controllers used in the practical multirotor from Erasmus (2020) are implemented in these simulations. These are standard controllers which were designed without a swinging payload. This does not provide active damping of the payload oscillations and is therefore expected to perform worse than the MPC. Fig. 2 illustrates the structure of this controller. The inertial position and velocity of the multirotor are controlled in the outer loops. The inner loops control the attitude of the quadrotor and actuate the motors.

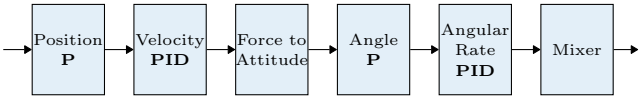

Fig. 2. Cascaded PID architecture.

Fig. 3 shows the simulated velocity ($\dot{x}$) response for a position step reference of $\Delta x_{ref} = 10\,\text{m}$, with and without a suspended payload. The oscillations caused by the payload are clearly visible and take a long time to reach steady-state.

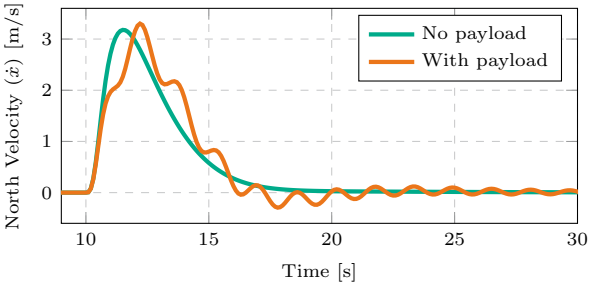

Fig. 3. Velocity response to a position step input with $m = 2\,\text{kg}$, $l = 1\,\text{m}$.

4.2 Model Predictive Control

MPC is a control scheme that uses an internal plant model to determine future control inputs by optimising the predicted trajectory of the model over a receding time horizon. The MPC implementation from MATLAB's Model Predictive Control Toolbox™ is used in this paper. This converts the linear MPC optimization problem to a general form QP problem. The models produced by DMDc and HAVOKc are each used as an internal plant model of the MPC.

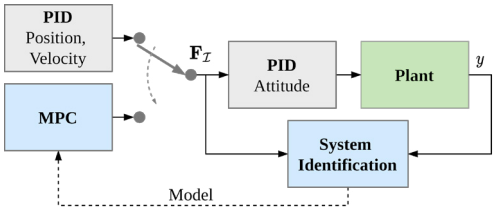

Fig. 4. Control structure with MPC added

For the multirotor with a suspended payload, the MPC is only required for translational control, because the payload has a negligible effect on the rotation of the vehicle. This is because the suspended payload is attached near the CoM of the vehicle. As shown in Fig. 4, the MPC replaces the position and velocity controllers of the Cascaded PID architecture and commands an inertial force reference to the existing PID attitude controllers. This has the advantage that the MPC can run effectively at a lower frequency than if it directly actuated the motors, which is desirable for practical implementation.

The MPC flight performance is heavily dependent on the choice of sampling time, T_s, prediction horizon, N_y, and input horizon, N_u. The final design choice of the MPC parameters are, $T_s = 30\,\text{ms}$, $N_y = 200$, and $N_u = 100$, resulting in prediction periods of $T_y = N_y T_s = 6\,\text{s}$, and $T_u = N_u T_s = 3\,\text{s}$. These values were determined by iteratively tuning the controller.

A large output weighting is placed on the swing angle state in the MPC cost function to ensure that the controller prioritises the damping of the payload oscillations. A great advantage of MPC is that it can also handle input and output constraints. A constraint can be enforced on the payload swing angle to ensure that the multirotor flies in such a way that the swing angle stays within certain limits. A constraint can also be placed on the control values commanded by the MPC to avoid actuator saturation. However, adding constraints increases the computation time of the MPC significantly.

5. SIMULATION RESULTS

In this section, simulation results of three control architectures are discussed, namely: Cascaded PID, MPC with DMDc, and MPC with HAVOKc. These simulations are performed within a MATLAB® and Simulink® environment. For the first part of each simulation, the multirotor is given position setpoints while controlled by the Cascaded PID controller. These setpoints are position step commands in both the $x_{\mathcal{I}}$ and $z_{\mathcal{I}}$ directions. Measurements are taken during this period and used as training data for system identification. The system identification algorithm then generates a model and the multirotor switches from PID control to MPC.

Note that system identification happens during flight and only data from the current flight is used for training. Therefore, the model is not dependent on dynamics previously seen from other payloads. In future work, system identification can be applied recursively to update the model over time, which would result in Adaptive MPC.

The performance of DMDc and HAVOKc is dependent on the flight time covered by the training data, sampling time, measurement noise and the hyperparameters, q and p. A flight period of 200 s and sampling time of $T_s = 30$ ms are used to generate training data during simulations. The same sampling time is used for the MPC. For each simulation with different conditions, a hyperparameter grid-search is performed by running the system identification algorithms with different combinations of q and p values. Each resulting model is used to predict the behaviour of the system over a time period not covered by the training data. A Mean Absolute Error (MAE) of each prediction is calculated and the model that produces the lowest MAE is then used in the MPC.

5.1 Velocity Response

Fig. 5 shows the velocity response for each controller with a position step input of, $\Delta x_{ref} = 10$ m, and payload parameters of, $m_P = 2$ kg and $l = 1$ m. It is clear that the MPC damps the velocity oscillations well. Note that the response of MPC with DMDc is almost identical to the response of MPC with HAVOKc, even though they use different models.

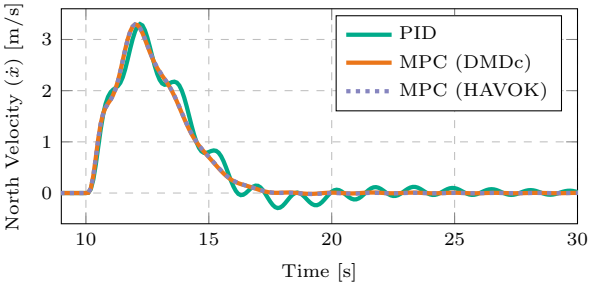

Fig. 5. Velocity response to a position step input with $m = 2$ kg, $l = 1$ m.

5.2 Payload Swing Angle

Fig. 6 shows the swing angle for a position step input of $\Delta x_{ref} = 10$ m. Note that the MPC damps the swing angle oscillations much better than the PID controller as expected. The PID controller causes a large swing angle during the initial move, whereafter the payload continues oscillating for some time. In contrast, the MPC manages to keep the initial swing angle low and damps the oscillations quickly. This is due to an output constraint and a high output weighting applied to the swing angle in the MPC optimisation. When all constraints are omitted, the MPC provides a similar settling time as before but causes an initial swing angle equal to that of the PID controller. It is worth noting that the performance of unconstrained MPC is satisfactory and has a shorter computation time than constrained MPC, which is advantageous for practical implementation.

5.3 Different Payloads

An advantage of the MPC control architecture is that it enables a multirotor to fly well with a range of different payloads. The control scheme was simulated with different payload mass and length combinations. In each case, it

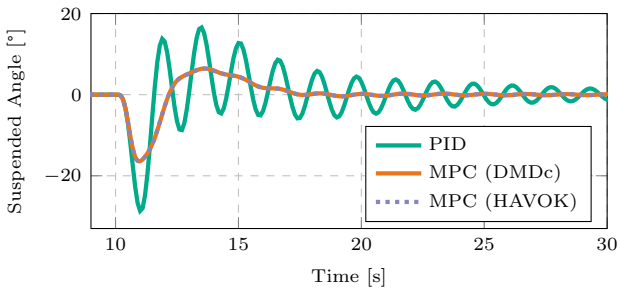

Fig. 6. Swing angle of payload for a position step input

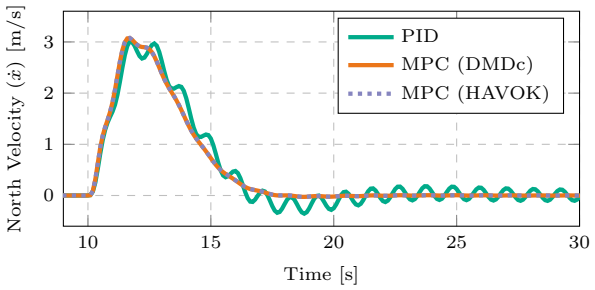

Fig. 7. Velocity response to a position step input with $m = 2.5$ kg, $l = 0.5$ m.

produced a new model and controlled the system successfully. Fig. 7 shows the velocity response of the multirotor with a heavier payload and shorter link than used in Fig. 5. Again a position step reference of $\Delta x_{ref} = 10$ m is used. Note that the oscillations in Fig. 7 have a higher frequency than in Fig. 5, and the system dynamics are therefore considerably different. It is clear that the MPC flies well with different payload parameters, without prior knowledge of those payloads.

5.4 Measurement Noise

DMDc and HAVOKc appear reasonably robust against measurement noise. The simulations were repeated with different amounts of additive band-limited white noise on the state measurements. Noise is added in both the training and control phases. With small amounts of noise, the control performance is nearly identical to the noiseless case. However, as greater noise is added, the performance of the MPC degrades due to lower accuracy models produced by DMDc and HAVOKc. Fig. 8 shows the performance of the MPC with a practical level of band-limited white noise added to the measurements. The noise power values are 6×10^{-8}, 6×10^{-8}, 4×10^{-8}, and 4×10^{-7}J for the angular rate, attitude, velocity and position measurements respectively. These values were determined from practical test flights with the actual multirotor (Erasmus, 2020).

It is clear from Fig. 8 that both MPC implementations damp the velocity oscillations and significantly outperform the PID controller, even when noise is present. For this amount of noise, MPC with HAVOKc performs better than MPC with DMDc, since it causes a lower overshoot in the velocity response with a near-identical settling time. Therefore, the overshoot in the position response (not shown) is also lower for MPC with HAVOKc than for MPC with DMDc. However, the difference in performance is not very significant and both provide satisfactory control.

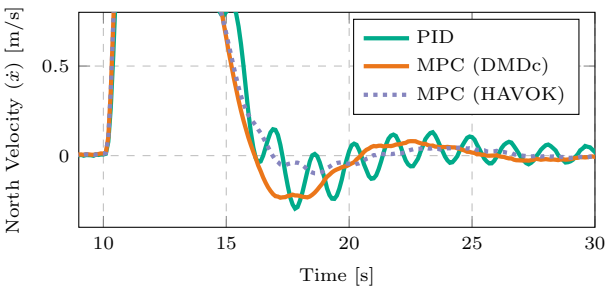

Fig. 8. Velocity response with measurement noise.

Note that the improved noise rejection comes at a cost to computation time. For example, the best performing model from Fig. 8 has a 156×156 system matrix, which is considerably bigger than the 16×16 system matrix used in the noiseless simulation. This increases the computation time of the MPC, which may make it difficult to implement it on a practical multirotor.

The simulation was repeated with noise power twice as large as the practical values. The MPC achieved stable control with the models from both DMDc and HAVOKc and still outperforms Cascaded PID control. However, there is a much larger overshoot in the velocity and position responses than before. HAVOKc produces a lower overshoot than DMDc but has a similar settling time.

6. CONCLUSION

It has been shown that the proposed MPC architecture successfully controls a multirotor with an unknown suspended payload in simulation, without any prior knowledge of the payload dynamics. This control architecture clearly outperforms a standard Cascaded PID controller in response to step inputs by damping oscillations in the velocity response and payload swing angle effectively. It was also shown that constraints can be added to the MPC to generate trajectories that limit the maximum swing angle. This level of control may be desirable when transporting delicate payloads.

Therefore, it appears that DMDc and HAVOKc are effective data-driven, system identification algorithms for use with linear MPC to control a multirotor with a suspended payload. Models of the altered flight dynamics were successfully identified without prior knowledge of the payload. Hence, the method is not bound to suspended payloads but may be feasible for other flight altering payloads too.

Both algorithms are quite robust against measurement noise and perform well with a practical level of noise. In the noiseless case, the algorithms give nearly identical performances, but HAVOKc outperforms DMDc with measurement noise. Both algorithms require larger system matrices for accurate models when dealing with noise, which increases the computation time of the MPC.

Because the non-linear dynamics of the multirotor and payload is approximated with a linear model, linear MPC can be implemented. This is advantageous for practical implementations because it has a lower time complexity than non-linear MPC. DMDc and HAVOKc are more computationally efficient than other data-driven techniques like Neural Networks since they do not require iterative training but rather rely on a simple linear regression operation. Overall, the proposed MPC architecture appears to be an effective control strategy for a multirotor with an unknown suspended payload.

REFERENCES

Arbabi, H. and Mezić, I. (2017). Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4), 2096–2126.

Brunton, S.L., Brunton, B.W., Proctor, J.L., Kaiser, E., and Kutz, J.N. (2017). Chaos as an intermittently forced linear system. *Nature Communications*, 8(19).

Brunton, S.L., Proctor, J.L., and Kutz, J.N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15), 3932–3937.

Champion, K.P., Brunton, S.L., and Kutz, J.N. (2019). Discovery of nonlinear multiscale systems: Sampling strategies and embeddings. *SIAM Journal on Applied Dynamical Systems*, 18(1), 312–333.

Dai, S., Lee, T., and Bernstein, D.S. (2014). Adaptive control of a quadrotor uav transporting a cable-suspended load with unknown mass. In *53rd IEEE Conference on Decision and Control*, 6149–6154.

Erasmus, A.P. (2020). *Stabilization of a Rotary Wing Unmanned Aerial Vehicle with an Unknown Suspended Payload*. Master's thesis, Stellenbosch University, Stellenbosch.

Kaiser, E., Kutz, J.N., and Brunton, S.L. (2018). Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2219), 20180335.

Kamb, M., Kaiser, E., Brunton, S.L., and Kutz, J.N. (2020). Time-delay observables for koopman: Theory and applications. *SIAM Journal on Applied Dynamical Systems*, 19(2), 886–917.

Korda, M. and Mezić, I. (2018). Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93, 149–160.

Notter, S., Heckmann, A., Mcfadyen, A., and Gonzalez, F. (2016). Modelling, simulation and flight test of a model predictive controlled multirotor with heavy slung load. *IFAC-PapersOnLine*, 49(17), 182 – 187. 20th IFAC Symposium on Automatic Control in Aerospace (ACA) 2016.

Proctor, J.L., Brunton, S.L., and Kutz, J.N. (2016). Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1), 142–161.

Tu, J.H., Rowley, C.W., Luchtenburg, D.M., Brunton, S.L., and Kutz, J.N. (2014). On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2), 391–421.