ArUco based Reference Shaping for Real-time Precision Motion Control for Suspended Payloads

Adrian Stein, David Vexler and Tarunraj Singh

***Abstract*— This work presents a real-time time-delay filtering approach for reference shaping of high precision motion control of vibratory systems. The motion of the system is initiated with a judicious (arbitrary) step command and the acquired motion data is used to estimate the modal parameters in real-time. The modal data is subsequently used to synthesize the subsequent step commands to mitigate the residual vibrations. The proposed control algorithm is tested on a gantry crane structure with a suspended payload. Our method estimates the system parameters based on computer vision while tracking an ArUco fiducial marker which is integral with the payload. Computational efficiency is ensured by using C++ to deploy the algorithm. The goal is to minimize the residual energy at the terminal displacement for rest-to-rest maneuvers of a suspended payload with unknown dynamics. An inertial measurement unit is used to track the pendular angular velocity at the end of the maneuver and is not used in the model identification process.**

***Index Terms*— Input Shaper, Computer Vision, ArUco, Gantry Crane, Vibration Control.**

I. INTRODUCTION

Vibration control for high speed rest-to-rest maneuvers has become vital in numerous applications, especially industrial crane control. For example, operating large cranes can often become incredibly difficult due to unforeseen conditions, such as strong winds, which can make moving large payloads very difficult and even dangerous. Implementing an algorithm for vibration suppression into operating cranes of all sizes is easily motivated with benefits including making their use easier, safer, and more efficient.

Previous research regarding pose estimation of crane loads has been done using computer vision implemented with particle filters for determining certain static parameters of crane loads [1]. These static parameters are used for visual motion tracking of crane loads in order to estimate precise load motion that corresponds to crane movement [2], as well as keeping tight control over crane movement in an effort to prevent emergencies, such as detecting collisions of swinging crane loads [3]. In research done in this field, cranes are often modeled as pendulums with moving pivot points [1].

In this paper, we will demonstrate the development of an algorithm for vibration suppression and validate it on a scaled model of a crane instrumented with an inertial measurement unit and an ArUco-based (Augmented Reality University of Cordoba) computer vision system. This approach for precise pose tracking [4], [5] and position estimation [6] is demonstrated on a small-scale gantry crane undergoing point-to-point maneuvers. The algorithm is capable of measuring the relative position of the payload with respect to the gantry crane and synthesizes a reference shaped input to the crane motor which mitigates the swinging. For the case of accurate pose and position estimation, we exploit computer vision using ArUco markers. We incorporated an inertial measurement unit (IMU) to track the swinging of our payload as well.

ArUco is an open-source library that was developed using OpenCV by Muñoz and Garrido [7]. The ArUco library provides algorithms for the robust and fast detection, pose tracking, and position estimation of square fiducial markers with six degrees of freedom. ArUco has been used in autonomous unmanned vehicle control, including aerial vehicles [8], [9], guided vehicles [10], and underwater vehicles [11], 3D scanning [12], tag identification [13], crane pose estimation [3], [1], and many other functions. Using ArUco's real-time tracking capabilities and the algorithm we coded in C++, high frequency ArUco marker position was acquired.

Referencing previous work, tests aimed to compare IMU tracking with ArUco tracking have found the two methods to both be extremely accurate with typically insignificant differences [14]. However, ArUco data has been seen to have much more noise than IMU data, likely because of the fact that ArUco is much more sensitive to exterior conditions than IMU, for instance, background and light conditions. ArUco is heavily dependent on a high resolution camera (In our experiment we used the Intel RealSense D435i with up to 1280 x 720 resolution), sufficient lighting, and effective calibration [14]. Although IMU has also been seen producing inaccuracies in its data from accumulation of minuscule errors in its accelerometer, drift errors in its gyroscope, and present external magnetic fields that affect its magnetometer, these effects have been to a smaller degree compared with ArUco. Some studies have integrated the two methods together for position estimation where each method's are compensated

faults cancel each other out to some degree [15].

This paper will proceed as follows: In Section II, we illustrate the simulation results of our proposed method. In Section III, we describe the experimental setup and results. In Section IV, we provide a brief conclusion.

A. Stein and T. Singh is with the Department of Mechanical and Aerospace Engineering, University at Buffalo, NY 14260, USA. (email: {astein3,tsingh}@buffalo.edu).

D. Vexler is with the Williamsville East High School, East Amherst, NY 14051. (email: david.vexler796@gmail.com).

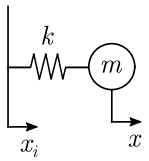

Fig. 1: Spring-Mass model.

II. SIMULATION

In this section we present the mathematical development of a real-time time-delay filter (TDF) using the benchmark spring-mass system as an example. Time-delay filtering of step inputs results in a staircase profiles where the tread of the stairs is a function of frequency of oscillation of the structure. The core idea of the proposed work is to initiate motion of the structure with an arbitrary step command (judiciously selected with an estimate of the damping ratio of oscillation of the suspended payload). In the interval prior to the execution of the next step command, the measured motion of the payload is used to estimate the damping and natural frequency of the structural mode, which results in the specification of the tread width of the current step.

A. Real-time time-delay filtering

Consider a spring-mass model as illustrated in Fig. 1 as an example. The equation of motion for the spring-mass system is:

$$m\ddot{x}(t) + kx(t) - kx_i(t) = 0. \tag{1}$$

We define a TDF as:

$$G(s) = \mathcal{K} + A_1 e^{-sT} + (1 - \mathcal{K} - A_1)e^{-2sT}, \tag{2}$$

where $0 \leq \mathcal{K} \leq 1$ is an arbitrary user-selected gain. The parameters A_1 and T are selected so that a pair of zeros of the TDF cancels the underdamped poles of the system located at $s = -\zeta\omega_n \pm i\omega_n\sqrt{1-\zeta^2}$. Equating the real and imaginary parts of Eqn. (2) to zero leads to:

$$\begin{aligned} \mathcal{K} + A_1 e^{\zeta\omega_n T}\cos(\omega_n T\sqrt{1-\zeta^2})... \\ +(1-\mathcal{K}-A_1)e^{2\zeta\omega_n T}\cos(2\omega_n T\sqrt{1-\zeta^2}) = 0, \end{aligned} \tag{3}$$

$$\begin{aligned} A_1 e^{\zeta\omega_n T}\sin(\omega_n T\sqrt{1-\zeta^2})... \\ +(1-\mathcal{K}-A_1)e^{2\zeta\omega_n T}\sin(2\omega_n T\sqrt{1-\zeta^2}) = 0. \end{aligned} \tag{4}$$

Eqn. (4) can be written as:

$$\begin{aligned} e^{\zeta\omega_n T}\sin(\omega_n T\sqrt{1-\zeta^2})\left(A_1 + 2(1-\mathcal{K}-A_1)e^{\zeta\omega_n T}\right. ... \\ \left. ... \cos(\omega_n T\sqrt{1-\zeta^2})\right) = 0, \end{aligned} \tag{5}$$

whose solution is [16]:

$$T = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}}. \tag{6}$$

Now we can substitute Eqn. (6) into Eqn. (3):

$$\mathcal{K} - A_1 e^{\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} + (1-\mathcal{K}-A_1)e^{\frac{2\pi\zeta}{\sqrt{1-\zeta^2}}} = 0, \tag{7}$$

which simplifies to [16]:

$$A_1 = \frac{\mathcal{K} + (1-\mathcal{K})e^{\frac{2\pi\zeta}{\sqrt{1-\zeta^2}}}}{e^{\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} + e^{\frac{2\pi\zeta}{\sqrt{1-\zeta^2}}}}. \tag{8}$$

The reason why the initial gain of the TDF $G(s)$ is user-selected is to permit the identification of model parameters (ζ and ω_n) over the time window of $[0,\ T]$ which can then be used to determine the delay time T and the amplitude A_1, which is available in closed-form. This will result in a real-time model identification and input shaping for vibration control of underdamped systems. Extending the spring mass model to include an input u, we consider the second order system:

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = \omega_n^2 u. \tag{9}$$

Laplace transform of Equation 9 assuming zero initial conditions and with a step input of magnitude $\mathcal{K}$ results in:

$$s^2X(s) + 2\zeta\omega_n sX(s) + \omega_n^2 X(s) = \frac{\mathcal{K}\omega_n^2}{s}. \tag{10}$$

Dividing Eqn. (10) by s^2 and evaluating the inverse Laplace transform leads to:

$$x(t) + 2\zeta\omega_n\int_0^t x(\tau)d\tau + \omega_n^2\int_0^t\int_0^\sigma (x(\tau)d\tau)d\sigma = \mathcal{K}\omega_n^2\frac{t^2}{2}. \tag{11}$$

Dividing Eqn. (10) by s^3 and performing another inverse Laplace transform, we get:

$$\begin{aligned} \int_0^t x(\tau)d\tau + 2\zeta\omega_n\int_0^t\int_0^\sigma (x(\tau)d\tau)d\sigma... \\ ... + \omega_n^2\int_0^t\int_0^\sigma\int_0^\lambda (x(\tau)d\tau)d\lambda d\sigma = \mathcal{K}\omega_n^2\frac{t^3}{6}. \end{aligned} \tag{12}$$

Combining Eqs. (11) and (12) in a matrix form results in [17]:

$$\mathcal{A}\begin{bmatrix} 2\zeta\omega_n \\ \omega_n^2 \end{bmatrix} = \begin{bmatrix} -x(t) \\ -\int_0^t x(\tau)d\tau \end{bmatrix} \tag{13}$$

$$\mathcal{A} = \begin{bmatrix} \int_0^t x(\tau)d\tau & \int_0^t\int_0^\sigma (x(\tau)d\tau)d\sigma - \mathcal{K}\frac{t^2}{2} \\ \int_0^t\int_0^\sigma (x(\tau)d\tau)d\sigma & \int_0^t\int_0^\sigma\int_0^\lambda (x(\tau)d\tau)d\lambda d\sigma - \mathcal{K}\frac{t^3}{6} \end{bmatrix}, \tag{14}$$

which can easily be solved for ζ and ω_n.

This approach is extended to a system where the commanded input is velocity. Assuming that a pendulum on a gantry crane as shown in Fig. 4 can be approximated as (see [18]):

$$\ddot{x}(t) + 2\zeta\omega_n\dot{x}(t) + \omega_n^2 x(t) - \omega_n^2 x_i(t) = 0 \tag{15}$$

$$\dot{x}_i(t) = v(t). \tag{16}$$

We can transform this into the Laplace domain and combine both equations to result in:

$$X(s)\left[s^2 + 2\zeta\omega_n s + \omega_n^2\right] - \omega_n^2\frac{V(s)}{s} = 0. \tag{17}$$

Applying a step velocity command, $V(s) = \mathcal{K}/s$, we have:

$$X(s)\left[s^2 + 2\zeta\omega_n s + \omega_n^2\right] - \omega_n^2 \frac{\mathcal{K}}{s^2} = 0. \tag{18}$$

We divide Eqn. (18) by s^2 and apply the inverse Laplace transform which results in:

$$x(t) + 2\zeta\omega_n \int_0^t x(\tau)d\tau + \omega_n^2 \int_0^t \int_0^\sigma (x(\tau)d\tau)d\sigma = \frac{\mathcal{K}\omega_n^2 t^3}{6}. \tag{19}$$

Since we have two unknowns (ζ and ω_n) we need to derive another equation, which can simply be done by dividing Eqn. (18) by s^3:

$$\int_0^t x(\tau)d\tau + 2\zeta\omega_n \int_0^t \int_0^\sigma (x(\tau)d\tau)d\sigma ... \\ ... + \omega_n^2 \int_0^t \int_0^\sigma \int_0^\lambda (x(\tau)d\tau)d\sigma d\lambda = \frac{\mathcal{K}\omega_n^2 t^4}{24}. \tag{20}$$

Writing Eqs. (19) and (20) in a matrix form results in:

$$\mathcal{B}\begin{bmatrix} 2\zeta\omega_n \\ \omega_n^2 \end{bmatrix} = \begin{bmatrix} -x(t) \\ -\int_0^t x(\tau)d\tau \end{bmatrix} \tag{21a}$$

$$\mathcal{B} = \begin{bmatrix} \int_0^t x(\tau)d\tau & \int_0^t \int_0^\sigma (x(\tau)d\tau)d\sigma - \mathcal{K}\frac{t^3}{6} \\ \int_0^t \int_0^\sigma (x(\tau)d\tau)d\sigma & \int_0^t \int_0^\sigma \int_0^\lambda (x(\tau)d\tau)d\lambda d\sigma - \mathcal{K}\frac{t^4}{24} \end{bmatrix}, \tag{21b}$$

which can easily be solved for ζ and ω_n. Now, we apply a pulse input, $G_p(s) = 1 - e^{-st_{f,pulse}}$, where $t_{f,pulse}$ is determined by the desired final displacement and the maximum velocity. Note, that we assume that $t_{f,pulse} > 2T$.

B. Numerical results

We simulate the gantry crane system using MATLAB and set the desired final displacement to $x_f = 600$ mm. Assume that $K = A_0 = 0.25$, $v_{max} = 240$ mm/s and the real system has a natural frequency of $\omega_n = 2\pi$ and a damping ratio of $\zeta = 0.1$. The initial guess for the estimated parameters are: $\hat{\omega}_n(0) = 3\pi$ and $\hat{\zeta}(0) = 0.15$. From Eqn. (6) it can easily be seen that $T = 0.5025$ s. Thus, we know that the system identification phase has to happen before T. We set the bounds for the identification as $t_{bounds} = [0.35,\ 0.45]$ s. The justification for waiting to collect data prior to estimating the model parameters is illustrated by the variation of the condition number of the matrix given by Eqn. (21b) in Fig. 3, whose inverse is required to determine ζ and ω_n. Waiting for the condition number to reduce results in a stable estimate of the model parameters.

In Fig. 2, it can be seen that the target of reaching 600 mm is satisfied and that the velocity at the end of the maneuver is 0 mm/s. This implies that there is no residual vibration in the system. The red phase indicates the estimation phase of the natural frequency and damping ratio over time, which spans the time period of 0.1 s. The estimated parameters for the input shaper are calculated as: $\hat{T} = 0.5035$ s and $\hat{A}_1 = 122.4086$. Fig. 3 shows the estimated natural frequency

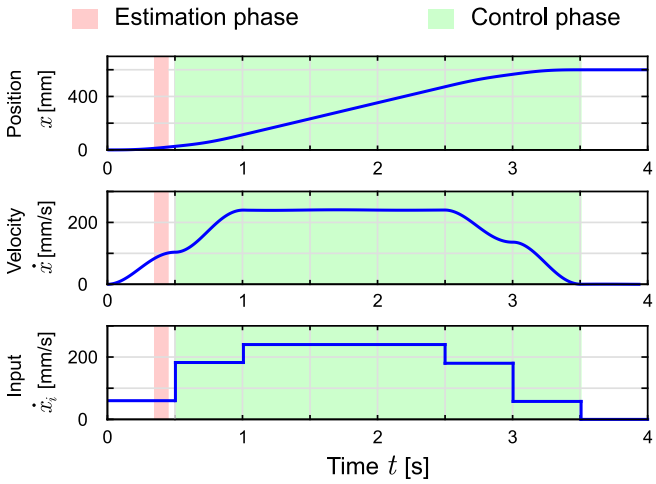

Fig. 2: Position and velocity of the payload. Velocity input of the trolley.

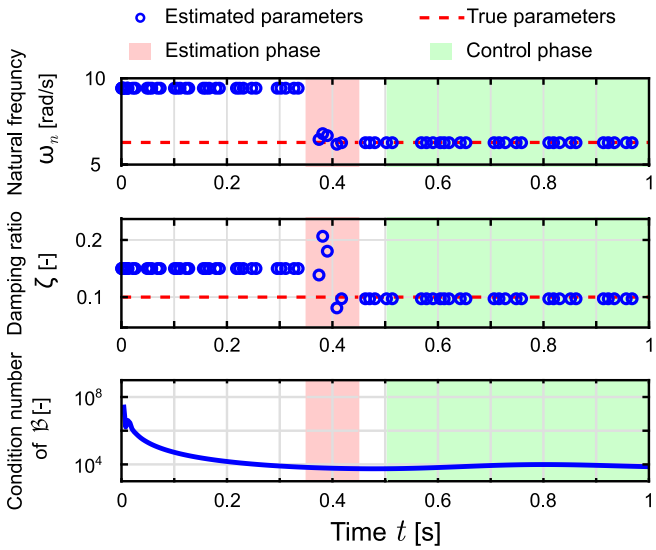

Fig. 3: Estimated natural frequency, damping ratio and condition number of $\mathcal{B}$ in Eqn. (21b).

and damping ratio over time. It can be seen that the parameters converge rather quickly towards the true parameters of the system. At the end of the estimation phase, the parameters are estimated as: $\hat{\omega}_n(0.45) \approx 6.2692$ rad/s (true: $\omega_n = 6.2832$ rad/s) and $\hat{\zeta}(0.45) = 0.0971$ (true: $\zeta = 0.1$), which has error of 0.2223% and 2.8962% respectively.

III. Experiment

This section describes the experimental part of the proposed algorithm.

A. Experimental setup

The experiment is conducted on a 8'x4'x3' Gantry Crane system where the trolley is attached to a stepper motor via a timing belt, which can be seen in Fig. 4. The inset shows the payload with the IMU, transmitter, and power bank. The

IMU is an MPU-6050 which is used to track the angular velocity around the y-axis. It is attached to the payload and transfers a timestamp and the angular velocity to an nRF24L01 transmitter with the help of an Arduino Nano. The IMU, transmitter, and Arduino Nano are all attached to the payload and are powered by a power bank. The IMU's sampling frequency is on average 48 Hz. The motion of the trolley is constrained by the maximum number of steps per time instant (velocity constraint). With the driver set to 400 steps per revolution, the trolley displaces by 40 mm per revolution. For this setup, it is found that the maximum velocity of the trolley is $v_{max} = 240$ mm/s. Furthermore, we use the Intel RealSense Depth Camera

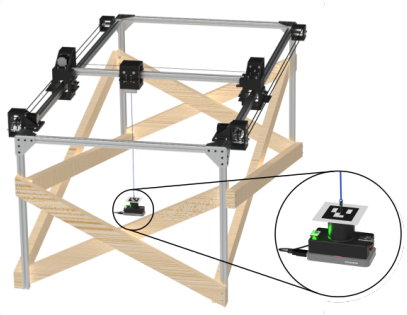

Fig. 4: Experimental setup of the gantry crane.

D435i. To achieve high computational speed and capture a sufficiently wide image, we use a 848×100 resolution. This provides the highest frames per second and matches the purpose of our experiment. Since we actuate the crane trolley only in one axis, the payload will swing only around one axis, which solely requires a long and thin frame. The infrared camera can capture up to 300 frames per second (fps) while the RGB camera module is limited to 30 fps. We decided to use the left infrared camera of the Intel RealSense D435i, which is sufficient to detect the 6×6 ArUco marker. During the experiments, the infrared emitter is disabled. The intrinsic properties of the Intel RealSense D435i infrared camera are [19]: $[f_x, f_y] = [427.3, 427.3]$ and $[c_x, c_y] = [422.4, 43.8]$. For the depth calculation and therefore the Cartesian position estimation, both infrared cameras are being used. Our tests showed that using the Intel RealSense D435i in a Python framework results in 30 fps while using C++ results in 200 fps when purely recording images without the detection algorithm. Therefore, the ArUco detection algorithm is implemented in C++ to achieve a higher frame rate per second during the estimation phase. The flowchart in Fig. 5 illustrates the sequence of tasks starting with the capture of the ArUco marker and leading to the ultimate stepper motor control. The initial guess of $\hat{\omega}_n(0) = 7$ rad/s in the experimental implementation. We set the estimated damping ratio to 0 for all time.

The computation was performed on a Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz 2.40GHz with 8GB RAM using Visual Studio 2022 and OpenCV 4.6.0. It should be mentioned that there is a 55 mm offset in the x-axis between the

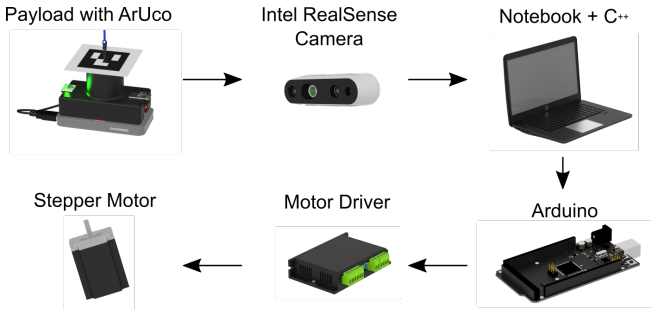

Fig. 5: Flowchart of data processing and control.

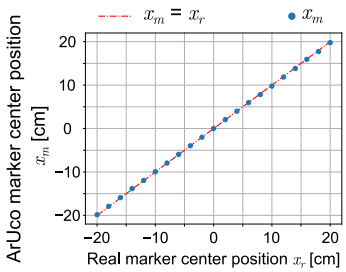

Fig. 6: Estimated position measurement in x-axis with camera versus real position using an ArUco marker.

center of the infrared camera and the center of the ArUco marker.

B. Experimental results

We place the Intel Realsense D435i on the trolley and test the intrinsic properties of the camera. We choose a distance between the ArUco marker and the camera module of 53.3 cm. Then, a 6×6 AruCo marker gets displaced only along the x-axis from -20 to 20 cm in 21 equally spaced increments, which are shown by the blue dots in Fig. 6. The red-dashed line shows the position which would be an exact match between the actual (real) displacement and the estimated position by the camera. The blue dots align with the red-dashed line and we conclude that the chosen intrinsic properties of the camera module are satisfying the position tracking along the x-axis and can be used for our proposed algorithm. Fig. 7 shows an image of the 848×100 resolution image from C++ and is displayed to the user. The authors are aware that displaying the image during the real-time system identification has an impact on the sampling frequency of the algorithm. Alg. 1 explains the procedure for the tracking of the ArUco marker, the real-time system identification, and the control via the stepper motor. When integrating with Runge Kutta Dormand–Prince method, we are using linear interpolation to get the marker

Fig. 7: Camera view on the payload with ArUco marker.

position between two time instances. We decided to move

Algorithm 1: Position and parameter estimation with ArUco and crane control.

Prerequisites: //Identification of intrinsic camera properties;
//Using infrared channel in 848×100 resolution with 300 fps setup;
Ensure: //Initialize states x to be 0;
$K = 60,\ x_f = 600,\ off_m = 55,\ t = 0;$
$i = 0,\ j = 0,\ k = 1;$
//global $t_0,\ t_1,\ x_0,\ x_1;$
while *(Marker detected == True)* **do**
  //Apply velocity K to crane;
  //Assign marker center $\rightarrow x_m$ & system clock $\mathcal{T}$;
  if $(i == 0)$ **then**
    $t_0 = t_1 = t;$
    $x_0 = x_1 = x_m + off_m = 0;$
  else
    $t_0 = t_1,\ t_1 = \mathcal{T};$
    $x_0 = x_1,\ x_1 = x_m + off_m + (t - t_1 - t_0)K;$
    $x[0] = x_0$ (Payload's position);
    x = fct_integration(x, t, $t_1 - t_0$);
    $t = t + t_1 - t_0;$
    //Estimation;
    if $(t \geq 0.35\ \&\ t \leq 0.6)$ **then**
      // Use Eqs. (21),(6),(8);
    else if $(t > 0.6\ \&\ k == 1)$ **then**
      $k = 0;$
      //Apply control algorithm to crane;
  $i = i + 1;$
  //Plot the image with recognized marker (Fig. 7)

the trolley 600 mm for 3 different cases: 1) Pulse input; 2) closed-form based TDF for the 1st mode; 3) ArUco based TDF for the 1st mode. Fig. 8 illustrates the velocity about the y-axis 10 s after the completion of the maneuver. It can be seen that the response to a pulse input causes the largest oscillation in the 1st mode and slight oscillations in the 2nd mode (top graph). The closed-form based TDF for the 1st mode cancels the vibration completely for the 1st mode but causes large oscillations in the 2nd mode due to the aggressive maneuver (middle graph). Our proposed controller, which is designed based on the ArUco marker (bottom graph), almost cancels the 1st mode completely and only excites the 2nd mode slightly. To reflect the magnitude of the oscillations for each controller case, we make use of the fast-fourier-transformation, which is also shown in Fig. 8 as a single sided amplitude spectrum. It can be seen that the ArUco proposed algorithm is minimizing the residual vibration compared to the pulse input tremendously. The reduction of the amplitude in the 1st mode is 92.81% compared to a pure pulse input.

The controller that was designed by the

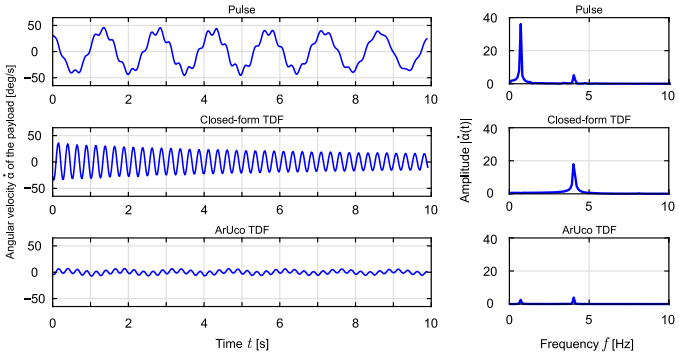

Fig. 8: Angular velocity $\dot{\alpha}$ of the payload for a displacement of 600 mm after the maneuver is completed and single sided amplitude spectrum of different controllers.

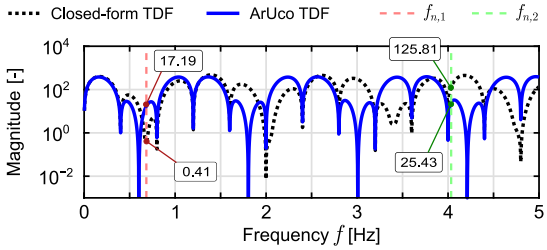

Fig. 9: Bode magnitude plot of closed-form and ArUco TDF.

proposed ArUco algorithm has the structure: Switch times = $[0.831, 1.662, 2.500, 3.331, 4.162]$ s, and Velocities = $[60, 180, 240, 180, 60]$ mm/s. The rope length to the center of the mass of the payload is 53 cm. Thus, the switch times for the closed-form TDF are $[0.735, 1.471, 2.500, 3.235, 3.971]$ s. Fig. 9 shows the bode plots of the controller designed in closed form and by the ArUco marker. It can be seen that the magnitude of the second natural frequency is much higher for the closed-form TDF than for the ArUco one. Thus, the excitation of the second mode is much larger, as it was shown in Fig. 8. The damping ratios are very small, thus only the undamped natural frequencies are illustrated. The ArUco controller takes longer for the maneuver because the switch time is identified as $\hat{T} = 0.831$ s instead of 0.735 s. The actual 1st natural frequency of the system is $\omega_n = 4.273$ rad/s and the estimated one is $\hat{\omega}_n = 3.779$ rad/s, which is an error of 11.56% in the estimation. Note that the proposed ArUco algorithm does not have any knowledge about the system parameters, as it estimates the natural frequency within 0.25 s, which can be found in Table I. It can be seen that the switch time converges from 0.923 to 0.831 s. It should be mentioned that the user needs to approximately know where the switching time lies because if the estimation phase starts too late and the $\hat{T} > T$, the algorithm cannot be successfully implemented. Another burden was to align the payload's IMU with the direction of the motion, which is due to the design of the experiment. Our proposed method can yield the time-optimal results if, for instance, an undampened system is assumed and the parameter A_0 is set to 0.5.

Time	Natural frequency	Estimated payload position	Trolley position	Switch time
t [s]	$\hat{\omega}_n$ [rad/s]	$\hat{x}$ [mm]	x_i [mm]	$\hat{T}$ [s]
0.380	3.402	8.057	22.800	0.923
0.407	3.434	9.677	24.420	0.914
0.425	3.478	10.757	25.500	0.904
0.440	3.516	11.657	26.400	0.893
0.456	3.370	13.684	27.360	0.932
0.472	3.475	14.644	28.320	0.904
0.489	3.578	15.664	29.340	0.878
0.504	3.463	17.631	30.240	0.907
0.520	3.592	18.591	31.200	0.874
0.534	3.693	19.431	32.040	0.850
0.551	3.663	24.252	33.060	0.857
0.590	3.779	24.603	35.400	0.831

TABLE I: Parameters during ArUco estimation phase.

This means, choosing a different initial velocity during the identification phase can lead to a time-optimal solution, so no final maneuver time increase will occur. This will be performed in future work.

The algorithm can be easily extended to identify more than just the first mode of the system and, for instance, reduce the vibration in the 2^{nd}, 3^{rd}, ... modes. The classic method of a vibration control is to identify the system first, design a controller and then apply the control. This will require one experimental run first before the vibration can be suppressed. The errors in the estimation of the natural frequency can be accommodated for with the design of a robust TDF, which can easily be implemented to our algorithm. Usually an IMU with a power source (e.g. power bank) and a transmitter needs to be attached to the payload, while our method only requires a QR code-like marker and a camera which can track the marker. This offers several advantages such as non-invasive sensing of the payload's motion, less equipment, battery-free tracking of the payload, and a simple setup. An application of our method can be used for container hubs, where stickers could be placed on containers, precision motion control in package deliveries with quadcopters, wind mill assemblies onshore or offshore, oil and gas industry for pipeline mounting, or helicopter operations such as heavy load transportation, rescue operations with rope ladders, firefighting operations with water buckets etc.

IV. CONCLUSIONS

This work deals with adaptive precision motion control, while the proposed algorithm estimates the natural frequency on the fly and leads to a reduction in residual vibration of the targeted mode. Experiments were performed as a proof of concept for a suspended payload on a gantry crane system. The algorithm was implemented in C++ to enable a higher sampling rate, thus the authors would like to stress that the identification phase happens in a quarter of a second. Currently, the proposed method requires zero initial conditions but can easily be extended to non-zero initial condition scenarios. Future work will include operations such as lifting/lowering of the payload and deriving a robust version of the real-time time-delay filter structure via Augmented Reality. Tests will be performed to test time-optimal precision motion algorithm and will be extended to 2D motions.

REFERENCES

[1] T. A. Myhre and O. Egeland, Eds., *Estimation of crane load parameters during tracking using expectation-maximization: 2017 American Control Conference (ACC)*, 2017.

[2] T. A. Myhre, Ed., *Static Parameter Estimation on SO(3) using Stochastic Gradient Descent for Visual Tracking: 2019 18th European Control Conference (ECC)*, 2019.

[3] T. A. Myhre and O. Egeland, Eds., *Collision detection for visual tracking of crane loads using a particle filter: IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016.

[4] H. C. Kam, Y. K. Yu, and K. H. Wong, Eds., *An Improvement on ArUco Marker for Pose Tracking Using Kalman Filter: 2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 2018.

[5] P. Oščádal, D. Heczko, A. Vysocký, J. Mlotek, P. Novák, I. Virgala, M. Sukop, and Z. Bobovský, "Improved pose estimation of aruco tags using a novel 3d placement strategy," *Sensors (Basel, Switzerland)*, vol. 20, no. 17, 2020.

[6] S. S. Tørdal and G. Hovland, "Relative vessel motion tracking using sensor fusion, aruco markers, and mru sensors," *Modeling, Identification and Control: A Norwegian Research Bulletin*, vol. 38, no. 2, pp. 79–93, 2017.

[7] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[8] B. H. Y. Alsalam, K. Morton, D. Campbell, and F. Gonzalez, Eds., *Autonomous UAV with vision based on-board decision making for remote sensing and precision agriculture: 2017 IEEE Aerospace Conference*, 2017.

[9] J. Bacik, F. Durovsky, P. Fedor, and D. Perdukova, "Autonomous flying with quadrocopter using fuzzy control and aruco markers," *Intelligent Service Robotics*, vol. 10, no. 3, pp. 185–194, 2017.

[10] V. T and N. U. H. D, Eds., *Vision-Based Automated Guided Vehicle: 2023 International Conference on Smart Systems for applications in Electrical Sciences (ICSSES)*, 2023.

[11] Z. Xu, M. Haroutunian, A. J. Murphy, J. Neasham, and R. Norman, "An underwater visual navigation method based on multiple aruco markers," *Journal of Marine Science and Engineering*, vol. 9, no. 12, p. 1432, 2021.

[12] F. Erich and N. Ando, Eds., *A Framework for 3D Scanning using RGB-D Cameras and an Automated Rotary Table: 2022 IEEE/SICE International Symposium on System Integration (SII)*, 2022.

[13] V. Magnago, L. Palopoli, R. Passerone, D. Fontanelli, and D. Macii, "Effective landmark placement for robot indoor localization with position uncertainty constraints," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 11, pp. 4443–4455, 2019.

[14] N. Elangovan, A. Dwivedi, L. Gerez, C. M. Chang, and M. Liarokapis, Eds., *Employing IMU and ArUco Marker Based Tracking to Decode the Contact Forces Exerted by Adaptive Hands: 2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019.

[15] A. Poulose and D. S. Han, "Hybrid indoor localization using imu sensors and smartphone camera," *Sensors*, vol. 19, no. 23, p. 5084, 2019.

[16] T. Singh, "Optimal reference shaping for dynamical systems: Theory and applications," *Optimal Reference Shaping for Dynamical Systems: Theory and Applications*, 2009.

[17] M. Fliess and H. Sira-Ramízes, "An algebraic framework for linear identification," *ESAIM - Control, Optimisation and Calculus of Variations*, vol. 9, 2003.

[18] A. Stein and T. Singh, "Minimum time control of a gantry crane system with rate constraints," *Mechanical Systems and Signal Processing*, vol. 190, p. 110120, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888327023000274

[19] "https://github.com/intelrealsense/librealsense/issues/11511," https://github.com/IntelRealSense/librealsense/issues/11511, accessed: 2023-09-20.