Delay Robust Model Predictive Control For Whole-Body Torque Control of Humanoids

Rajesh Subburaman, Olivier Stasse

Delay Robust Model Predictive Control For Whole-Body Torque Control of Humanoids

Rajesh Subburaman[1], and Olivier Stasse[1,2]

***Abstract*— Whole body model predictive control (WBMPC) is a powerful tool to generate complex robotics motion. Despite the recent increase in computational capabilities with new processors such as the Apple chipsets (M1, .. M3) or GPUs, WBMPC algorithms need a significant amount of computational time. This induces delay, which, if not properly accounted for, can have detrimental effects on the controller's performance. This paper conducts a detailed study to understand the impact of delay on WBMPC and proposes an efficient solution to handle it effectively. In this regard, a whole-body control task is formulated as an optimal control problem and solved using the Differential Dynamic Programming (DDP) solver. An extensive amount of numerical studies are carried out to understand the nature of the problem and thereby devise an effective solution. The proposed solution is found to be effective numerically, and it has been experimentally verified with the humanoid TALOS. Both numerical and experimental results are presented and discussed in this work to provide valuable incites.**

I. Introduction

Model Predictive Control (MPC) is a renowned feedback control strategy widely used in the robotics community apart from various industrial applications. It has the advantage of dealing with nonlinearities, model inaccuracies, and constraints, making it the prime choice tool for controlling a diverse number of complex and dynamic robotic systems such as quadrotor [1], autonomous racing [2], legged robots [3]–[5], etc. In robotics, MPC [6] is used to solve large-scale optimal control problems (OCP) in a receding finite horizon fashion to optimize a prediction of the robot motion and apply the first instance of the resulting control output while waiting for the next control update [7]. Though MPC was initially restricted to handle simple and dynamically slower systems, thanks to the increase in computation power, it is being widely applied to fairly complex systems such as four-legged robots, collaborative manipulation involving mobile robots [8], quadrotors, etc.

MPC is effectively a trajectory optimization problem wherein the problem is formulated as an OCP considering the task-related goals, system constraints, and limitations. The various objectives and constraints are usually written as non-linear cost functions with individual weights to prioritize them according to their significance. At the instant of receiving a state estimate of the system, the OCP above is solved over a receding finite horizon, and the first instance of the control output trajectory is fed to the low-level controller for execution. This is repeated for each control cycle. Though this seems to work well for linear problems, it poses a huge challenge for non-linear systems involving high degrees of freedom (DoF) and complex tasks such as contact handling, obstacle avoidance, etc., because the solving complexity is $O(m^3 \cdot N)$, where m and N is the optimized variable dimension and prediction horizon, respectively. In such cases, finding a global minimum in real time is hard to achieve. Due to this, MPC usage, until recently, has been predominantly limited to systems represented with reduced models.

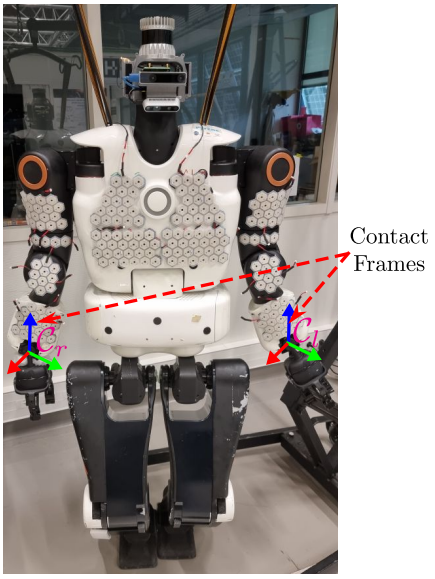


Fig. 1: The humanoid TALOS used in this study is shown here with the predefined contact frames C_l and C_r.

Lately, interesting works have been reported using Differential Dynamic Programming (DDP) [9], a direct shooting type OCP solver. The ability of DDP to exploit the sparsities in OCP, its linear convergence rate, and its linear complexity in the horizon length have resulted in interesting and promising works [3], [10]. The aforementioned traits have made it possible to realize real-time control with full-body dynamic models. The first application of a whole-body MPC using the DDP solver is reported in [4] for an HRP-2 humanoid to reach a desired end-effector target under different circumstances. In the above work, the desired joint state trajectories are sent to the low-level joint controllers for them to track accurately. This is further extended to carry out dynamic whole-body locomotion on TALOS in torque control mode [11]. Notably, in [11], the feed-forward torques are computed at 100 Hz, and together with a linear state feedback controller running at 2 kHz, a stable performance has been reported. The linear state feedback controller naturally arises in the form of Riccati gains when the OCP is solved using DDP, as explained in [12]. This has been exploited in this work.

[1] Gepetto, LAAS-CNRS, Universite de Toulouse, France
[2] Artifical and Natural Intelligence Toulouse Institute, France

Though a successful demonstration of a DDP-based whole-body MPC has been reported in [11], the effect of computation delay on the controller performance and how to deal with it has never been addressed. Since an optimal solution for the control variable is computed after obtaining an estimate of the robot's state, any significant CPU time taken to solve an OCP problem can induce a delay between the state update and the injection of control. This delay can deteriorate the controller's performance and can subsequently lead to an unstable system. This has been extensively analyzed for a non-linear MPC applied on a simple continuous system in [13]. However, such analysis has never been done for a highly dynamic and non-linear system such as a humanoid controlled in torque mode. In this work, we have strived to analyze the effect of the aforementioned computation delay for a DDP-based MPC and proposed a solution to effectively deal with it. Though in [4], it is stated that the delay has been accounted for, the proposed solution is applicable only for a system controlled in position mode since the feed-forward and feedback terms are not considered. Further, no extensive analysis has been reported on the effect of delay on the controller's performance, as done in this work. To summarize, this work makes the following notable contributions: 1) extensive analysis of the detrimental effect of computation delay on a whole-body MPC controller in torque mode, 2) effect and analysis of different control policies proposed to counter the delay, and 3) experimental evaluation of the proposed solution on a torque controlled humanoid TALOS.

The manuscript is structured as follows: Firstly, in Section II, some background information related to the manuscript is briefly presented. This is followed by the succinct presentation of how an MPC is formulated as DDP in Section II-C. The problem that we have strived to address in this manuscript is stated in Section III, and the whole-body control task used to analyze the problem is defined in Section III-A. Following this, the various control policies are defined in Section III-B, and the numerical results obtained with each of those policies are compared and analyzed in Section IV. Finally, experimental results are presented in Section V and appropriate conclusions are drawn in Section VI.

II. Background

In this section, some background information that is required to understand this manuscript is briefly presented. In particular, in Section II-A, the system used for this study is introduced. The mathematical model used to represent the robotic system is succinctly presented in Section II-B. Finally, an OCP problem is defined for the aforementioned system, and how it can be solved in an MPC framework using the DDP solver is explained briefly in Section II-C.

A. System Description

The robotic system considered for the numerical evaluation and experimental verification includes the TALOS humanoid [14], shown in Fig.1. The TALOS humanoid has 32 DoF, weighs 95 kg, is 1.75 m in height, and can be controlled in both position and torque mode. In the presented work, we have controlled TALOS in torque mode for both simulations and experiments. Further, the humanoid is equipped with joint torque sensors and encoders on each joint, and force-torque sensors are fitted to both the wrists and ankles of the robot. In addition, TALOS is equipped with a set of skin patches (white patches) distributed on its upper body. However, these skin patches are not used in the presented work.

B. Contact-Constrained Robot Dynamic Model

The dynamic equation of motion of a rigid body system with n DoF subjected to n_c number of contact constraints can be written as described in [15] as follows:

$$\begin{bmatrix} \mathbf{M} & \mathbf{J}_c^T \\ \mathbf{J}_c & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{S}^T\mathbf{u} - \mathbf{b} \\ -\dot{\mathbf{J}}_c\dot{\mathbf{q}} \end{bmatrix}, \tag{1}$$

where $\mathbf{M} \in \mathbb{R}^{n\times n}$, $\mathbf{b} \in \mathbb{R}^{n\times 1}$, $\mathbf{S} \in \mathbb{R}^{(n+6)\times n}$, and $\mathbf{u} \in \mathbb{R}^{n\times 1}$ are the joint space inertia matrix, nonlinear force vector, selection matrix to apply torques to the actuated joints, and the joint torque vector. In (1), $\mathbf{q} \in \mathbb{SE}(3) \times \mathbb{R}^n$ denotes the configuration of a robotic system and since the system considered here is a floating base system, i.e., a humanoid, the unactuated 6 DoF is represented by $\mathbb{SE}(3)$. The velocity and acceleration of the system are denoted by $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, respectively. $\mathbf{J}_c = [\mathbf{J}_1 \cdots \mathbf{J}_{n_c}]^T$ represents the contact jacobian matrix for n_c contacts and $\mathbf{J}_i \in \mathbb{R}^{6\times n}$ denotes the contact jacobian matrix for i^{th} contact. Similarly, $\lambda_i \in \mathbb{R}^{6\times 1}$ denote the i^{th} contact wrench and $\lambda = [\lambda_i \cdots \lambda_{n_c}]$ includes the contact wrench vector for all n_c contacts. The bottom part in (1) ensures that the n_c number of contacts remain motionless during the general motion of a rigid body system, i.e., contact consistent dynamic motion is generated.

The above equation of motion can be integrated in discrete form with a time period dt as follows:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k), \tag{2}$$

where $\mathbf{x} = [\mathbf{q}, \dot{\mathbf{q}}] \in \mathbb{R}^{n_x}$, $\dot{\mathbf{x}} = [\dot{\mathbf{q}}, \ddot{\mathbf{q}}] \in \mathbb{R}^{n_{dx}}$, and $\mathbf{u}$ denote the state, its derivative and the input vector of the system. n_x and n_{dx} represent the number of state variables and state derivative variables respectively. The state of the system at next time step $(k+1)$ is denoted by $\mathbf{x}_{k+1}$.

C. MPC Formulated as DDP

Using (2), an optimal control problem for the contact-constrained rigid body dynamic system can be written in discrete form as follows:

$$\begin{aligned} \min_{\bar{\mathbf{x}},\bar{\mathbf{u}}} \quad & \sum_{k=0}^{T-1} l(\mathbf{x}_k, \mathbf{u}_k, k) + l_T(\mathbf{x}_T) \\ \text{s.t.} \quad & \mathbf{x}_0 = f_0 \\ \forall t = 0 \cdots T \quad & \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k), \end{aligned} \tag{3}$$

where $\bar{\mathbf{x}} = (\mathbf{x}_k)_{k=0\cdots T}$, and $\bar{\mathbf{u}} = (\mathbf{u}_k)_{k=0\cdots T}$ are the state and control trajectory sequence for T number of knots or time steps. l and l_T are the running and terminal cost functions

used to encode the desired behavior of the robot, and $\mathbf{x}_0$ represents the system's initial state. The second condition in (3) ensures that the optimal state and control sequence respects the system dynamics. By solving (3), an optimal pair of state ($\bar{\mathbf{x}}^\star$) and control ($\bar{\mathbf{u}}^\star$) sequence that is consistent with the system dynamics (2) is obtained.

Now, solving the above OCP problem in an MPC framework, i.e., at time step k, we solve (3) and obtain $\bar{\mathbf{u}}_k^\star$ by taking the system's initial state to be $\mathbf{x}_k$. Applying only the first instance $\mathbf{u}_0$ of $\bar{\mathbf{u}}_k^\star$ for a duration dt results in a new state $\mathbf{x}_{k+1}$. This is used as the initial state for the next time step $k+1$, to obtain a new optimal control sequence $\bar{\mathbf{u}}_{k+1}^\star$, and it is repeated iteratively. This ensures that the generated motion is feasible for the next T steps in the future, also called the prediction horizon. Beyond T, the system's nominal stability is ensured by the terminal cost l_T.

The OCP problem (3) is solved at each iteration of the MPC using the DDP algorithm. One of the advantages of using DDP is that the system dynamics is implicity considered during the forward pass of the algorithm, so the generated motion is always dynamically feasible. However, for floating-base robots that have ≥ 12 DoF, solving an MPC problem at a rate $>$100 Hz is a difficult task due to the current limitation of computational power. In our case, we control 28 DoF of the TALOS humanoid in torque control mode, and we have taken $T = 100$, and the MPC control policy is updated every 15 ms, i.e., 66 Hz. Generally, it is difficult to achieve a stable torque control performance at 100 Hz even with a fixed base manipulator, let alone for a 28 DoF humanoid. Thanks to the DDP algorithm, a stable control has been achieved due to the Ricatti gains ($\mathbf{K}_k$) that arise naturally while solving the OCP problem, in particular, during the backward pass of DDP. The gains $\mathbf{K}_k$ are used to implement a linear state feedback controller that effectively interpolates the MPC control policy. The resulting control law can be formulated as

$$\mathbf{u} = \mathbf{u}_k^{ff} + \mathbf{K}_k(\mathbf{x} - \mathbf{x}_k) \tag{4}$$

where $\mathbf{u}_k^{ff}$ and $\mathbf{K}_k$ are the feedforward torque commands and the Ricatti gains computed by solving the MPC problem at the initial state $\mathbf{x}_k$. The second term in (4) represents the state feedback torque $\mathbf{u}_k^{fb}$, and $\mathbf{x}$ is the current state of the system which is updated at a very high rate. In this manuscript, for both simulations and experiments, $\mathbf{u}_k^{ff}$ and $\mathbf{K}_k$ are updated at 66 Hz, and $\mathbf{x}$ is updated at 1 kHz. The final control command $\mathbf{u}$ is updated at 1 kHz in the simulations, and for the real system, it is implemented at the lower level at 2 kHz. For more details on (4), their implementation and evaluation, please refer to [12].

III. Problem Description and Methodology

The problem due to the delay associated with the whole-body MPC for humanoids is explained in this section using Fig.2. In the figure, $\hat{\mathbf{x}}_k$ denote the state measurement at t_k and $\Pi_{k-1}^k(\mathbf{u}_k, \mathbf{K}_k, \mathbf{x}_k)$ represents the control policy to be applied from t_{k-1} to t_k. $\tau = \tau_c + \tau_n$ is the total time delay, where τ_c and τ_n represent the computation and network

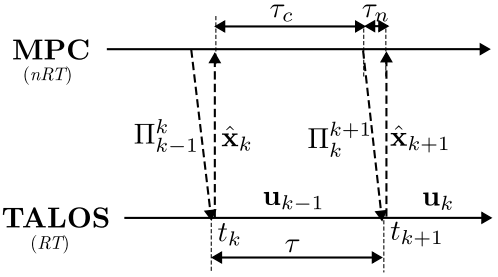


Fig. 2: The MPC delay observed in the classical approach is shown here.

communication delays respectively. Since solving OCP using the DDP solver takes a significant amount of time, it is run separately in a non-real-time thread (*nRT*), and the low-level controllers for the TALOS humanoid are executed at a much higher frequency in a real-time thread (*RT*).

In an ideal scenario, given $\hat{\mathbf{x}}_k$, if the DDP solver can solve (3) in a non-negligible amount of time (τ_c), then τ can be taken as τ_n. Since $\tau_n \approx 1$ ms, the computed $\mathbf{u}_k$ can be applied almost instantaneously, resulting in a much better controller performance. However, in reality, this is not the case. Since humanoid is a highly nonlinear system with $>$ 12 DoF and the tasks that need to be achieved are usually non-convex, the DDP solver takes a considerable amount of time. Assuming the time taken by DDP to solve (3) as τ_c, given $\hat{\mathbf{x}}_k$, it requires $\tau = \tau_c + \tau_n$ amount of time to compute $\mathbf{u}_k$ and apply it on the system. This implies that $\mathbf{u}_k$ can only be applied at t_{k+1} and until then, the system (TALOS) will be running with the control policy computed for the previous cycle $\mathbf{u}_{k-1}$. Though this approach may still yield stable performance of the system in certain cases, such as a relatively less dynamic task/motion, moderate τ, etc., for highly dynamic tasks, which in turn induces a significant τ, the above approach could deteriorate the controller performance. This problem is often overlooked in the simulations because of the instantaneous realization of the commanded torque. This is not the case when it is applied on a real system, as discussed later in Section V. The extent of the above-mentioned performance deterioration and how to mitigate this by adopting alternate approaches has been extensively explored in this work.

A. Whole-Body Control Task

A whole-body control task is devised to analyze the above-mentioned problem. The objective of the task is to maintain balance and track a desired trajectory (cartesian space) given as a reference for predefined contact frames on the arms. Figure 1 shows the contact frames C_l and C_r defined on the left and right arm of the robot. Further, during the motion, it is necessary to ensure that the joint position, velocity, and torque limits are respected. To realize the aforementioned task, a total of six different costs are considered for the running and terminal cost models given in (3). The different costs considered are as follows: i) state, ii) actuation, iii) joint limits, iv) Center of Mass (CoM), v) end-effector pose, and vi) end-effector twist. The first two costs are used to regulate the change in joint state and torque, (iii) is intended to enforce the joint limits through penalties, and (iv) is

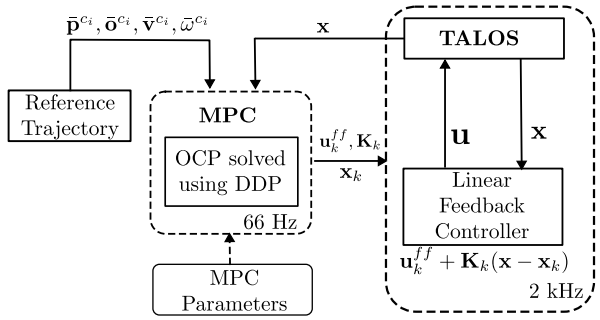


Fig. 3: The MPC control framework considered for the whole-body control task is shown here.

included to maintain the robot's balance. The last two cost terms are intended to track the desired trajectory. The MPC control framework is shown in Fig.3. In the figure, $\bar{\mathbf{p}}^{c_i}$, $\bar{\mathbf{o}}^{c_i}$, $\bar{\mathbf{v}}^{c_i}$, and $\bar{\omega}^{c_i}$ denote the reference trajectories for pose, orientation, linear velocity, and angular velocity of the pre-defined contact c_i. For more details regarding the control framework and costs, please refer to [16].

For this task, wrist joints 5, 6, and 7 are not considered for both arms, and the contact frame location is pre-defined on link 5 for both arms. Even without the wrist joints, a total of 28 DoF is considered for whole-body control. This amounts to a total of 57 state variables ($\mathbf{x}$) and 22 control outputs ($\mathbf{u}$) that will be optimized using (3). Further, the balancing and trajectory tracking tasks make this OCP already a complex one to solve using DDP.

B. Different Control Policies

To analyze and understand the effect of delay on the performance of a DDP-based MPC, six different variations of the control policy defined in (4) are considered as follows:

- $\mathbf{u}_k^1 = \mathbf{u}_k^{ff} + [\mathbf{K}_k \cdot (\mathbf{x} - \mathbf{x}_k)]$
- $\mathbf{u}_k^2 = \mathbf{u}_k^{ff} + [\mathbf{K}_k \cdot (\mathbf{x} - \mathbf{x}_k)]_{lp}$
- $\mathbf{u}_k^3 = \mathbf{u}_k^{ff} + \left[\mathbf{K}_k \cdot (\mathbf{x} - \tilde{\mathbf{x}}_{k,k+1})\right]_{lp}$
- $\mathbf{u}_{k+1}^1 = \mathbf{u}_{k+1}^{ff} + [\mathbf{K}_{k+1} \cdot (\mathbf{x} - \mathbf{x}_{k+1})]$
- $\mathbf{u}_{k+1}^2 = \mathbf{u}_{k+1}^{ff} + [\mathbf{K}_{k+1} \cdot (\mathbf{x} - \mathbf{x}_{k+1})]_{lp}$
- $\mathbf{u}_{k+1}^3 = \mathbf{u}_{k+1}^{ff} + \left(\mathbf{K}_{k+1} \cdot (\mathbf{x} - \tilde{\mathbf{x}}_{k+1,k+2})\right]_{lp}$,

where, $(\cdots)_{lp}$ denote the low-pass filtering of the feedback torque computed using the Ricatti gains; and $\tilde{\mathbf{x}}_{i,j}$ represents the linear interpolation of optimal state trajectories between $\mathbf{x}_i$ and $\mathbf{x}_j$. The first three control policies are variations of $\mathbf{u}_k$, and the last three are variations of $\mathbf{u}_{k+1}$. The reason for introducing low-pass filtering and interpolation in the computation of feedback torque will be explained in the following section. To compare and analyze the performance of the above-listed control policies, a whole-body control task, defined in Section III-A, has been used. In particular, the reference trajectory that is used is a reaching task, which involves the extension of both the arms of TALOS while maintaining its foot contact and balancing its CoM.

IV. Simulation Results and Analysis

In this section, the various control policies defined in Section III-B are executed numerically, and their respective

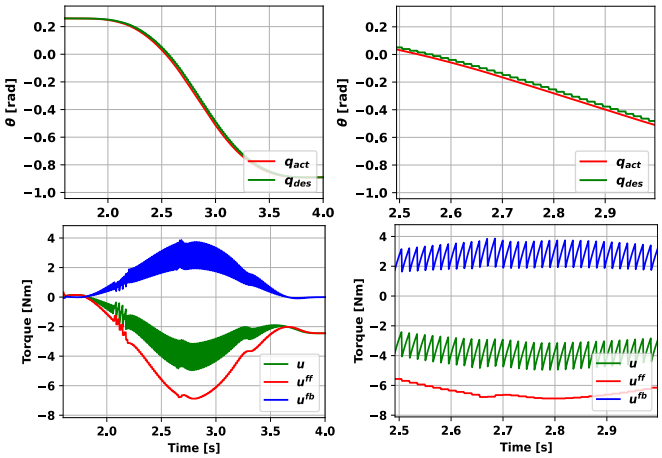


Fig. 4: The left arm joint 1 position and torque plots generated with $\mathbf{u}_k^1$ policy is shown here.

results are analyzed and compared critically. For the simulations, the PyBullet simulator is used with the complete dynamic model of TALOS. The DDP-based MPC controller shown in Fig. 3 is implemented in C++, and Python bindings are used to interact with the pybullet simulator. The linear feedback controller is run at 1kHz instead of 2kHz, owing to the simulator's limitation. For solving the OCP problem (3), the Crocoddyl [17], an optimal control library based on novel and efficient DDP algorithms, is used. In this work, 28 DoF (legs - 6×2, arms - 4×2, torso - 2, and floating base - 6) of TALOS are controlled in torque mode, and the rest of the joints are maintained at a fixed position using position control. Further, contact frames (C_l and C_r) predefined at the wrist of each arm are used to track any given reference trajectory. The whole-body control task defined in Section III-A is solved using DDP, and due to the complexity of the task, a single iteration takes ≈15 ms, and as a result, the MPC control thread is run at ≈66 Hz.

1) Effect of Various $\mathbf{u}_k$*:* Firstly, $\mathbf{u}_k^1$ is applied, and the resulting motion is recorded. The motion involves a stabilization period of 1.5s followed by a reference trajectory duration of 2s. Since the majority of the motion is associated with the arms of TALOS, in particular, the joint 1 of both the arms, the joint position (top) and torque (bottom) plots of the left arm joint 1 are shown in Fig. 4. The top left side plot shows a reasonably good tracking of joint 1. However, it can be seen that the actual value (q_{act}) is leading the desired (q_{des}), and this suggests that $\mathbf{u}_k$ computed at t_k is being applied at t_{k+1} due to τ. Despite this, fairly reasonable tracking is achieved, thanks to the feedback torque ($\mathbf{u}_k^{fb}$), which keeps the system from drifting away. However, this is not desirable since a significant magnitude ($> 50\%$) of $\mathbf{u}_k^{fb}$ is acting against $\mathbf{u}_k^{ff}$ and it is highly jittery as visualized in the bottom right plot of $\mathbf{u}_k^{fb}$. When a jittery $\mathbf{u}_k^{fb}$ is combined with $\mathbf{u}_k^{ff}$, it makes the final torque output ($\mathbf{u}_k$) to be jittery as well. This may not affect the numerical model in a simulator, but for real systems, this can significantly affect the controller's performance by inducing unwanted vibrations in the system. The adverse effects of applying $\mathbf{u}_k^1$ to a real system are

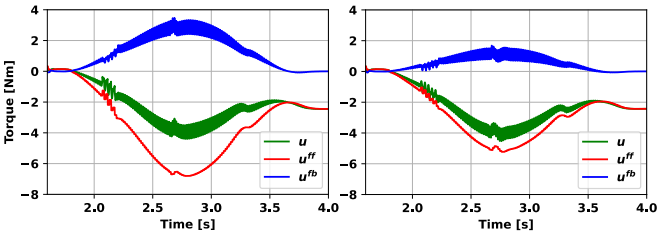


Fig. 5: The torque plots of left arm joint 1 generated with $\mathbf{u}_k^2$ and $\mathbf{u}_k^3$ policies are shown here.

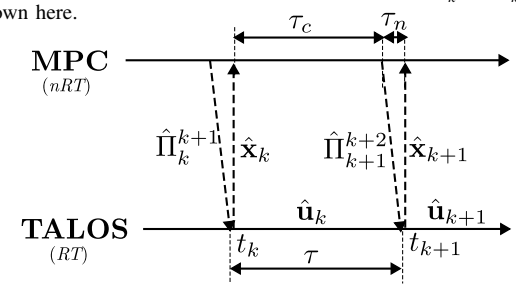


Fig. 6: MPC with delay compensation is shown here.

discussed in Section V.

To reduce the jitteriness in $\mathbf{u}_k^{fb}$, a Butterworth filter with a cut-off frequency of 66Hz is applied to $\mathbf{u}_k^{fb}$. The resulting control policy is denoted as $\mathbf{u}_k^2$, and the left arm joint 1 torque components are shown in the left side plot of Fig. 5. We can notice that the jitteriness and the magnitude have been reduced only slightly. This suggests that the source of jitteriness is something else other than noise. Inspecting the formulation of $\mathbf{u}_k^{fb}$ in (4), it can be understood that $\mathbf{K}$ is a constant gain matrix for a time step dt and it cannot be varying. Whereas, $\mathbf{x}$ being the system's instantaneous state, it continues to evolve until the next control input arrives. In such a case, it is not ideal to keep a constant optimal reference state $\mathbf{x}_k$, as the steps seen in the top right-side plot of Fig.4. Since the OCP is written in an MPC form with a horizon $T = 100$, $\mathbf{x}_{0\cdots T}$ is known at every time step. It is possible to interpolate from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ during a control cycle. Applying the filter and linear interpolation results in $\mathbf{u}_k^3$. The torque plot obtained for left arm joint 1 by executing $\mathbf{u}_k^3$ is shown in the right-side plot of Fig. 5. The magnitude of $\mathbf{u}_k^{fb}$ is reduced significantly but the jitteriness is still present which suggests that $\mathbf{x} - \tilde{\mathbf{x}}_{k,k+1}$ is still erroneous, specifically due to $\tilde{\mathbf{x}}_{k,k+1}$.

2) Effect of Various $\mathbf{u}_{k+1}$: In the previous subsection, we observed a jittery $\mathbf{u}$ even after filtering $\mathbf{u}_k^{fb}$ and interpolating $\tilde{\mathbf{x}}_{k,k+1}$ and this is due to the state difference $\mathbf{x} - \tilde{\mathbf{x}}_{k,k+1}$. This state difference is because $\mathbf{u}_k^3$ computed using $\mathbf{x}_k$ is being applied at t_{k+1} where the system is at $\mathbf{x}_{k+1}$ due to the delay τ_c. This is evident from Fig. 2. To recitfy this, we need to apply $\mathbf{u}_{k+1}$ at t_{k+1} but this is not possible due to τ_c. However, at t_k, when (3) is solved using DDP, we get $\overline{\mathbf{u}_k} = \left[\mathbf{u}_k, \mathbf{u}_{k+1|\hat{\mathbf{x}}_{k+1}} \cdots \mathbf{u}_{k+n|\hat{\mathbf{x}}_{k+n}}\right]$, where $\overline{\mathbf{u}_k}$ is the set of control policies to be applied for $k+T$ steps. Here, $\mathbf{u}_{k+1|\hat{\mathbf{x}}_{k+1}}$ is computed by integrating $\mathbf{x}_k$ using $\mathbf{u}_k$ to obtain $\hat{\mathbf{x}}_{k+1}$ which is then used to determine $\mathbf{u}_{k+1|\hat{\mathbf{x}}_{k+1}}$. Hence, $\mathbf{u}_{k+1|\hat{\mathbf{x}}_{k+1}}$ is an estimate of the actual $\mathbf{u}_{k+1}$ and henceforth, it will be denoted as $\hat{\mathbf{u}}_{k+1}$. Since $\hat{\mathbf{u}}_{k+1}$ is readily available at t_{k+1}, it can be applied instantaneously without any delay. Simultaneously, the computation of $\overline{\mathbf{u}}_{k+1}$ will be initiated with the state measurement $\mathbf{x}_{k+1}$. Euler integrator has been used for both simulations and experiments. The MPC control update with the delay (τ) compensation is shown in Fig. 6. In the figure, $\hat{\Pi}_k^{k+1}$ is a function of $\hat{\mathbf{u}}_k$, $\hat{\mathbf{K}}_k$, $\hat{\mathbf{x}}_k$ and $\hat{\mathbf{x}}_{k+1}$.

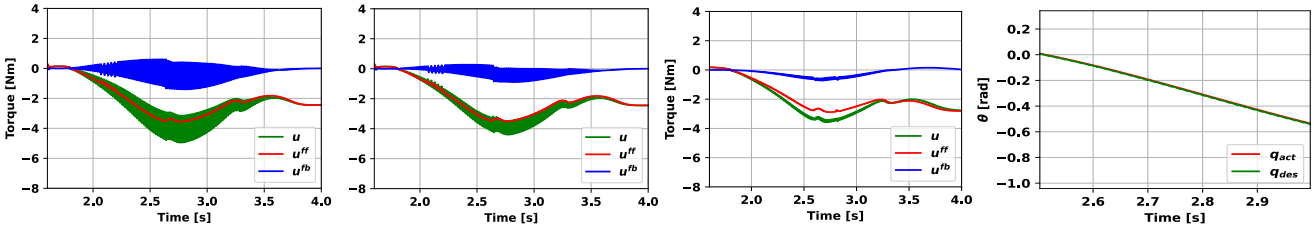
Similar to $\mathbf{u}_k$, three different variations of $\mathbf{u}_{k+1}$, as listed in Section III-B, are used to carry out the whole-body control task, and their results are compared and analyzed critically. The various torque components of left arm joint 1 are shown for $\mathbf{u}_{k+1}^1$ and $\mathbf{u}_{k+1}^2$ in Fig.7a, and Fig.7b shows for the same joint the torque component and joint position tracking plots obtained with $\mathbf{u}_{k+1}^3$ policy. With the application of $\mathbf{u}_{k+1}^1$, it can be seen that there is a significant drop in the magnitude of $\mathbf{u}_{k+1}^{ff}$ and $\mathbf{u}_{k+1}^{fb}$, and hence, the overall control output $\mathbf{u}$ drops too. In addition, we can observe that $\mathbf{u}_{k+1}^{fb}$ fluctuates around zero, which suggests that the desired state $\mathbf{x}_{k+1}$ is in sync with the system's actual values $\mathbf{x}$. However, the fluctuations seen in $\mathbf{u}_{k+1}^{fb}$ is due to having a fixed $\mathbf{x}_{k+1}$ at every time step. The fluctuations seem to be reduced to a certain extent, as seen in the right-side plot of Fig. 7a, with the application of a Butterworth filter on $\mathbf{u}_{k+1}^{fb}$, resulting in $\mathbf{u}_{k+1}^2$. However, in addition to filtering, an optimal state interpolation ($\mathbf{x}_{k+1,k+2}$) yields much better results, as seen in the left-side plot of Fig. 7b. The magnitude of $\mathbf{u}_{k+1}^{fb} \approx 1$ Nm, less fluctuating, and is acting in the direction of the motion, i.e., $\mathbf{u}_{k+1}^{ff}$. This suggests that $\tilde{\mathbf{x}}_{k+1,k+2}$ is leading $\mathbf{x}$ which can be visualized in the right-side plot of Fig. 7b and this is indeed an ideal and safe way of doing trajectory tracking. It is to be noted that even with the application of $\mathbf{u}_{k+1}^3$ there are still some minor fluctuations observed in $\mathbf{u}_{k+1}^{fb}$ and it can be attributed to the fact that $\mathbf{u}_{k+1}^3$ is computed using $\hat{\mathbf{x}}_{k+1}$ which is less accurate than an actual measurement since it is obtained by Euler integrating a simplified model.

To ascertain the significance of state interpolation, the results of $\mathbf{u}_{k+1}^3$ are compared with those of the same policy but without the low-pass filter. The torque results of left arm joint 1 obtained with the aforementioned variants of $\mathbf{u}_{k+1}^3$ are compared in Fig.8. It can be seen that without the low-pass filter (left-side plot), the state interpolation in $\mathbf{u}_{k+1}^3$ results in a significant improvement of $\mathbf{u}_{k+1}^{ff}$, $\mathbf{u}_{k+1}^{fb}$, and $\mathbf{u}$. Adding a low-pass filter to the state interpolation results in only a minor improvement, as seen in the right-side plot of Fig.8. This showcases the relative effect of state interpolation in $\mathbf{u}_{k+1}^3$ over low-pass filters. Despite this, we propose to use $\mathbf{u}_{k+1}^3$ with the filters activated to deal with the system noise, in particular, the joint velocity noise.

V. Experiment Results

In this Section, the different policies discussed in Section III-B are applied on the real TALOS system in torque mode, and the results so obtained are compared and analyzed.

The experiment setup involves the TALOS humanoid controllable in torque mode. Similar to the numerical results, only 22 joints of TALOS are controlled in torque mode, and the rest of the joints (wrist joints 5-7 of both arms) are held at a pre-defined configuration in position mode. For



(a) (b)

Fig. 7: (a) The torque components of left arm joint 1 obtained with $\mathbf{u}^1_{k+1}$ (left) and $\mathbf{u}^2_{k+1}$ (right) are shown here. (b) The torque components (left) and the joint position (right) tracking of left arm joint 1 obtained with $\mathbf{u}^3_{k+1}$ are plotted here.

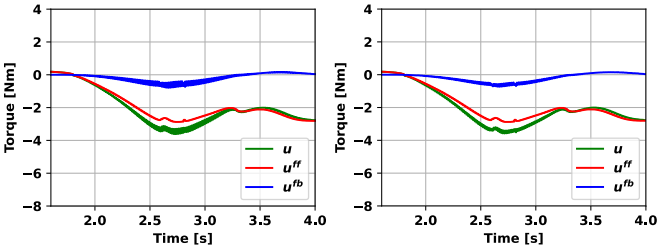
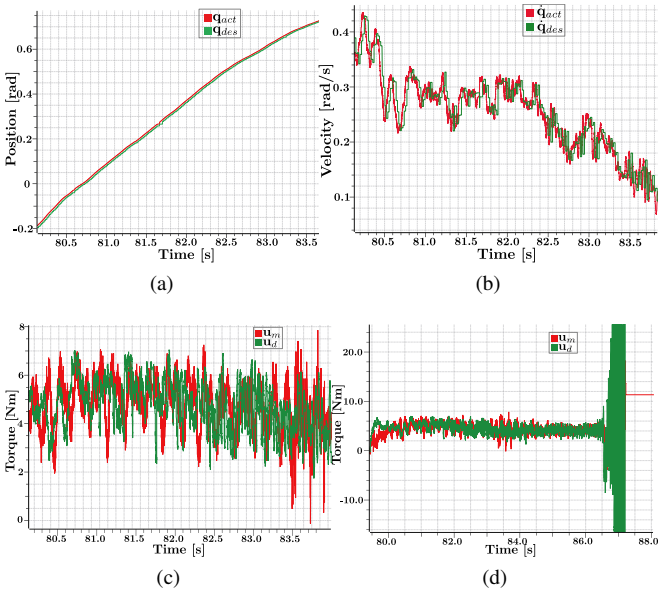


Fig. 8: The torque plots of left arm joint 1 generated with two variants of $\mathbf{u}^3_{k+1}$ are shown here. Variants considered: 1) without low pass filter (left) and 2) low pass filter (right).



(a) (b) (c) (d)

Fig. 9: A part of the position, velocity and torque tracking of right arm joint 1 obtained using $\mathbf{u}^1_k$ on the TALOS system are shown in (a), (b), and (c) respectively. The complete torque tracking plot is showcased in (d).

the evaluation of control policies, a whole-body control task involving a cartesian motion sequence has been considered for the pre-defined contacts on both arms of TALOS. The motion sequence is as follows: i) arm extension (0.4m in the x-direction), ii) grasp action (0.06m in the y-direction), iii) grasp-lift motion (0.1m in the z-direction). The whole-body motion control is carried out as shown in the control architecture Fig.3. The OCP formulated as an MPC is solved using the Crocoddyl library, and it is run in a non-real-time thread on an M1 ultra CPU (20 cores, 3.2GHz). For every single iteration, the Crocoddyl solver running on M1 ultra took $\approx$15 ms to solve, and hence, the MPC is set to run at 66Hz. The low-level linear feedback controller is run at 2 kHz on the onboard system, which uses Ubuntu 18.04. Robot Operating System (ROS) is used as the middleware to communicate with the TALOS system. The whole-body MPC and the linear feedback controller are designed to run as a separate ROS node, and ROS messages are used for communication. The communication between M1-ultra and the onboard system is established by means of an ethernet cable to reduce the transmission delay.

Since some of the policy variants can be destructive to the system, especially the variants of $\mathbf{u}_k$, only $\mathbf{u}^1_k$ and $\mathbf{u}^3_{k+1}$ are applied on the system, and their results are presented in this section. From Section III-B, it is already known that $\mathbf{u}^1_k$ is the original control policy in its standard form (4) and it was applied on TALOS even before we were aware of the instability and other issues instigated by the delay in MPC. In fact, the experimental results of $\mathbf{u}^1_k$ motivated us to carry out a detailed analysis of the effects of delay on an MPC-based whole-body controller.

The experimental results obtained with the TALOS humanoid by applying $\mathbf{u}^1_k$ are shown in Fig. 9. The figure shows the position, velocity, and torque tracking of right arm joint 1. From the joint velocity plot (Fig.9b), it can be seen that due to the delay τ_c, the actual velocity $\dot{\mathbf{q}}_{act}$ continues to advance and it is pulled back by the desired velocity $\dot{\mathbf{q}}_{des}$ commanded by the MPC controller. This is repeated throughout the control sequence and this results in an oscillatory behavior as visualized in the torque tracking plot Fig.9c. The position tracking plot (Fig.9a) shows the actual joint position q_m leading the desired one q_d, and this clearly shows that the applied control is lagging by one time step, just as we observed in the numerical results. The constant advancement of the system and the pulling effort by the controller leads to instability, as seen in Fig.9d around 86.5s. Apart from the instability, there were also a few other issues that were observed during the experiments, such as joint limit violation, undesirable noise in the joint actuators, etc. For more details, please see the video attached along with this manuscript.

The whole-body control task is repeated with the proposed control policy $\mathbf{u}^3_{k+1}$, and the results so obtained are showcased in Fig.10, which includes the torque (left) and joint position tracking (right) of arm joint 1. In the torque plot, apart from $\mathbf{u}_m$ and $\mathbf{u}_d$, the feedback ($\mathbf{u}^{fb}$) and feedforward ($\mathbf{u}^{ff}$) components are also plotted. It can be seen that the oscillation observed in $\mathbf{u}_m$ is relatively minimal but noisy

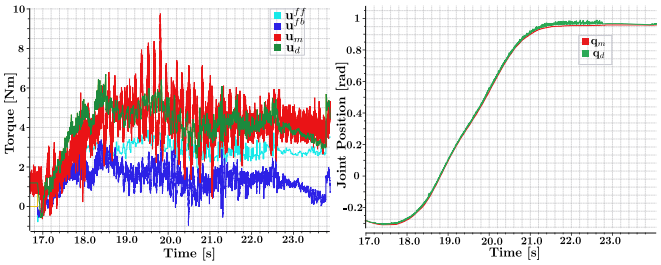


Fig. 10: The torque and position tracking of arm joint 1 obtained by applying $\mathbf{u}^1_{k+1}$ on the TALOS system are shown here.

since it is measured using the torque sensor mounted on the joint. The desired torque $\mathbf{u}_d$ is relatively less noisy and clean due to the filtering and interpolation of optimal state trajectory. The feedback torque $\mathbf{u}^{fb}$ is relatively high and slightly noisy, which could be due to joint friction and the Euler integration error. It is to be noted that in this work, a linear Coloumb-viscous friction model has been used to compensate for the joint friction. However, $\mathbf{u}^{fb}$ is found to be aiding the motion by acting in the same direction as $\mathbf{u}^{ff}$, and this suggests that there is very minimal delay in the applied control policy. Further, the joint position tracking plot shows clearly that q_d is leading q_m, which is not the case with $\mathbf{u}^1_k$. Overall, the proposed control policy $\mathbf{u}^3_{k+1}$ addresses the delay and achieves the desired whole-body control task stably.

VI. Conclusion

In this study, we have carried out a detailed analysis to understand how significant delays can affect the performance of an MPC-based whole-body controller and how it can be dealt with effectively. For the analysis, a whole-body motion task is formulated as an OCP and it is solved using a DDP-based library, Crocoddyl. The numerical results showed that the commanded torque was highly jittery when the classical control policy was applied. This is often overlooked in the simulation studies but it can be detrimental when applied to a real system. Even with low-pass filtering and interpolation of the desired state, the jitteriness was only slightly reduced. Better results were obtained with the proposed control policy $\mathbf{u}^3_{k+1}$, which compensates for the delay and also includes low-pass filtering and interpolation. The optimal command $\mathbf{u}$ yeilded by $\mathbf{u}^3_{k+1}$ was found to be less jittery, and the desired state was found to be leading the actual one, and hence, safer to apply on the real system. The proposed control policy was applied on a real TALOS system and its stable performance has been successfully verified. For future work, we would like to use this delay robust MPC controller to carry out whole-body manipulation of large objects using TALOS which is equipped with artificial skin cells.

Acknowledgement

We wish to thank Pierre Fernback and Maximilien Naveau for their invaluable support with the TALOS system. We would also like to express our sincere gratitude to Côme Perrot for sharing his valuable inputs regarding the usage of the Crocoddyl library. This work was funded in part by HoRoPo (ANR-22-CE92-0029), ANITI (ANR-19-P3IA-0004), and the access to TALOS was provided through ROBOTEX 2.0 (ROBOTEX ANR-10-EQPX-44-01 and TIRREX-ANR-21-ESRE-0015)

References

[1] K. Alexis, C. Papachristos, G. Nikolakopoulos, and A. Tzes, "Model predictive quadrotor indoor position control," in *2011 19th Mediterranean Conference on Control & Automation (MED)*. IEEE, 2011, pp. 1247–1252.

[2] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat, "Mpc-based approach to active steering for autonomous vehicle systems," *International journal of vehicle autonomous systems*, vol. 3, no. 2-4, pp. 265–291, 2005.

[3] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4906–4913.

[4] J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard, "Whole-body model-predictive control applied to the hrp-2 humanoid," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 3346–3351.

[5] F. Farshidian, E. Jelavic, A. Satapathy, M. Giftthaler, and J. Buchli, "Real-time motion planning of legged robots: A model predictive control approach," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 577–584.

[6] M. Alamir, *Stabilization of nonlinear systems using receding-horizon control schemes: a parametrized approach for fast systems*. Springer, 2006, vol. 339.

[7] J. B. Rawlings, D. Q. Mayne, M. Diehl, *et al.*, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2.

[8] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-body mpc for a dynamically stable mobile manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3687–3694, 2019.

[9] D. Q. Mayne, "Differential dynamic programming–a unified approach to the optimization of dynamic systems," in *Control and dynamic systems*. Elsevier, 1973, vol. 10, pp. 179–254.

[10] R. Budhiraja, J. Carpentier, C. Mastalli, and N. Mansard, "Differential dynamic programming for multi-phase rigid contact dynamics," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 1–9.

[11] E. Dantec, M. Naveau, P. Fernbach, N. Villa, G. Saurel, O. Stasse, M. Taix, and N. Mansard, "Whole-body model predictive control for biped locomotion on a torque-controlled humanoid robot," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*. IEEE, 2022, pp. 638–644.

[12] E. Dantec, M. Taix, and N. Mansard, "First order approximation of model predictive control solutions for high frequency feedback," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4448–4455, 2022.

[13] X. Yang and L. T. Biegler, "Advanced-multi-step nonlinear model predictive control," *IFAC Proceedings Volumes*, vol. 45, no. 15, pp. 426–431, 2012.

[14] O. Stasse, T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, J. Mirabel, A. Del Prete, P. Souères, N. Mansard, F. Lamiraux, *et al.*, "Talos: A new humanoid research platform targeted for industrial applications," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 689–695.

[15] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2014.

[16] C. Perrot and O. Stasse, "Step toward deploying the torque-controlled robot talos on industrial operations," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 10 405–10 411.

[17] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, "Crocoddyl: An efficient and versatile framework for multi-contact optimal control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2536–2542.