

Actuator Model, Identification and Differential Dynamic Programming for a TALOS Humanoid Robot

N. Ramuzat^{*†}, F. Forget[†], V. Bonnet[‡], M. Gautier[§], S. Boria^{*}, O. Stasse[†]

^{*} Airbus DVR, Toulouse, France

[†] LAAS-CNRS, Université de Toulouse, France

[‡] LISSI, Université de Paris-Est Créteil, France

[§]LS2N-CNRS, Université de Nantes, France

Abstract—In this experimental paper, we would like to validate a non linear optimal control solver to realize torque control on actuators embedded in a TALOS humanoid robot. The targeted application involves high payload, thus, it is necessary to handle the mechanical limitations of the system. To this extent, we propose a method to model, identify and control the TALOS humanoid actuators. The model includes the actuator drive chain and the corresponding inertial parameters that are identified at once using two experimental dataset. The identified model is then used by a Differential Dynamic Programming (DDP) optimal control solver to take into account the actuator limits. We demonstrated that the DDP can decrease the quality of the tracking to avoid physical limits in angular position, velocity and current in extreme conditions such as carrying large loads. Because of the solver high computational time, we validate our method on one actuator of the robot, the elbow joint, using its main CPU. In the experiments, we charge up to 34 kg on the arm of the robot at 5cm of the elbow joint, corresponding to 16 N at the joint level. The proposed implementation is working on this specific joint at 300μs and provide an effective solution to a real-world control problem. In the future, we will implement it over dedicated and embedded electronics board attached to each actuator.

INTRODUCTION

Having torque control on a humanoid robot is interesting for safe interaction with the environment and for its potential impact on cobotic applications. Historically, most of the humanoid robots are controlled in position and few robots, such as the TORO robot [3] or the DYROS-JET robot [14], are torque controlled. Recently, a new generation of humanoid robots, including the TALOS robot, has shown the possibility to use their actuators either in position or in torque control modes [17]. Several whole-body control architectures have been proposed ([2], [8]) to compute and follow desired joint torques and PAL Robotics demonstrated, during IROS 2018, an accurate whole-body balancing control on TALOS [13]¹. However regulating joint torques is still an opened problem, which requires a low level controller able to follow a desired joint torque. However some humanoid robots do not provide access to their low level controller. In this situation, Khatib et al.[9] proposed a controller which takes in input

a desired torque and provides a desired position to reflect the desired torque at the actuator. In a similar spirit, Del Prete et al.[16] implemented a torque control and inverse-dynamics control on the HRP-2 robot, originally designed for position control. They estimate the joint torques from the force sensors and the IMU by using the method proposed for the iCub robot by Nori et al.[12].

In this paper, we would like to take advantage of the capabilities of the commercially available TALOS robot. Its industrial standard EtherCat bus allows to gather numerous information about the actuators at a very high frequency. Thus, information related to the drive chain, the motor and joint positions or the joint torques can be read simultaneously. We would like to be able to generate extreme but safe motions, i.e. without breaking the motor. This event may occur if a large torque is sustained for an extended period of time or when the robot is carrying a large additional payload. To avoid it, the current drawn in an actuator can be limited by adapting its motion. Moreover, the motor current can be saturated by the joint controller if the desired joint torque is above its limit.

Ideally, this could be done using a model-based predictive controller (MPC). The use of a Differential Dynamic Programming (DDP) as MPC allows to cope with the actuator flexibility while meeting the control loop timing constraints. However, such a controller requires the knowledge of the inertial parameters of the robot, of the motor drive chain and to have a well calibrated joint torque sensor. As said before, these parameters may not be provided by the manufacturer of the robot or may be inaccurate. Indeed, for TALOS, the drive chain parameters are not disclosed, to protect industrial conception, and they do not take into account cabling, glued elements or cover. Fortunately, the literature in system identification of serial manipulators proposes methods that allow to identify the joint drive gains and inertial parameters at once [6, 7].

In this context, the objective of this paper is to push the TALOS robot to its limits during joint torque control. For that purpose, we propose to identify the joint drive chain of the elbow joint and the corresponding inertial parameters and to

¹The present paper is independent from the work in [13]

use a Differential Dynamic Programming approach to protect the system.

I. SYSTEM MODELING AND IDENTIFICATION

A. Mechanical Model

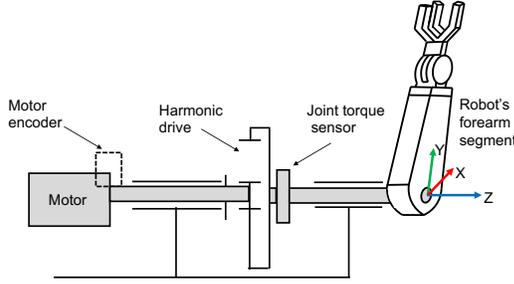


Fig. 1: Scheme of the retained mechanical model.

The inverse dynamics model of the investigated system is used to calculate the motor torque τ_m and joint torques τ_j as a function of the joint and/or motor positions, velocities and accelerations. It is usually calculated using Newton-Euler equations [18]. TALOS actuation chain is composed of a brushless motor, that can be controlled in current, connected to a harmonic drive and a torque sensor attached to its corresponding link. In this study the investigated sub-system is composed of the actuation chain of a single joint as represented in Fig. 1. The motor and joint positions are measured by two high-precision encoders (19 bits or 524,288 counts per revolution) and are considered equal thanks to the harmonic drive high stiffness. Consequently, the variables q , \dot{q} and \ddot{q} will be used to refer to the joint position, velocity and acceleration, respectively. The total inverse dynamics equations of the retained model are:

$$\begin{aligned}\tau_m &= (I_m + I_j)\ddot{q} + g(MY \sin q + MX \cos q) + \tau_{fm} + \tau_{fj} \\ \tau_j &= I_j\ddot{q} + g(MY \sin q + MX \cos q) + \tau_{fj}\end{aligned}\quad (1)$$

and

$$\begin{aligned}\tau_{fm} &= F_{vm}\dot{q} + F_{sm}\text{sign}(\dot{q}) + \text{off}_m \\ \tau_{fj} &= F_{vj}\dot{q} + F_{sj}\text{sign}(\dot{q}) + \text{off}_j\end{aligned}\quad (2)$$

where I_m is the rotor inertia of the motor, I_j is the corresponding link inertia expressed at the joint level, F_{vm} and F_{sm} are the motor viscous and dry frictions, F_{vj} and F_{sj} are the viscous and dry frictions at the joint level, off_m and off_j are the offsets of the torque motor and joint, MX and MY are the first moment of inertia expressed at the joint level and g is the gravity.

The motor torque τ_m is related to the current by:

$$\tau_m = RK_m i_m \quad (3)$$

where R is the gear ratio, K_m is the joint drive of the manufacturer and i_m is the input current of the motor. We can express the simple dynamic of our system as:

$$\begin{aligned}\ddot{q} &= \frac{1}{I}(RK_m i_m - F_v \dot{q} - F_s \text{sign}(\dot{q}) + \text{off}) \\ &\quad - (MY \sin q + MX \cos q) \frac{g}{I}\end{aligned}\quad (4)$$

where $I = I_m + I_j$, $F_v = F_{vm} + F_{vj}$, $F_s = F_{sm} + F_{sj}$ and $\text{off} = \text{off}_m + \text{off}_j$.

B. Identification of drive chain and segment inertial parameters

Usually the segment mass, center of mass and inertia are provided by the manufacturer of the robot with a relatively good accuracy, whereas the friction and drive chain parameters are unknown. The provided current gain drive K_m is known to have 10 to 15% of inaccuracy [6]. Finally, the joint torque sensor should be calibrated prior to be used for control applications. Thus, we propose to use the inverse dynamics model and a total least square approach [7] to identify the system described in Fig. 1 and its drive chain at once. To do so, two experiments were performed with and without using an additional payload.

When the inertial parameters of the dynamic model are expressed at the joint level, the model is linear, thus Eq.(1) becomes:

$$\begin{bmatrix} \tau_m \\ \tau_j \end{bmatrix} = \begin{bmatrix} RK_m i_m \\ K_j \tau_j s \end{bmatrix} = \mathbf{W}\Phi = \begin{bmatrix} \cos(q) & \sin(q) & \mathbf{D} & \mathbf{D} \\ \cos(q) & \sin(q) & \mathbf{0} & \mathbf{D} \end{bmatrix} \Phi \quad (5)$$

where $\mathbf{D} = [\ddot{q} \quad \dot{q} \quad \text{sign}(\dot{q}) \quad 1]$ and \mathbf{W} is the so-called regressor matrix. The vector ϕ containing the inertial parameters to be identified is defined as $\phi = [MX \quad MY \quad I_m \quad F_{vm} \quad F_{sm} \quad \text{off}_m \quad F_{vj} \quad F_{sj} \quad \text{off}_j]$. The TALOS robot embeds motor current and joint torque sensors, thus the regressor matrix \mathbf{W} is full rank and the parameters can be identified separately.

To identify the current drive and the torque sensor gains, it is necessary to insert them into the vector ϕ [7]. To do so, two regressor matrices that contain observations from two experiments have to be considered: \mathbf{W}_0 for the experiment without payload, and \mathbf{W}_1 for the one with a known payload at the end-effector. The Eq.(5) can be reformulated as:

$$\begin{bmatrix} -\mathbf{W}_0 & i_m & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{W}_0 & \mathbf{0} & \tau_{js} & \mathbf{0} & \mathbf{0} \\ -\mathbf{W}_1 & i_m & \mathbf{0} & -\mathbf{W}_{up} & -\mathbf{W}_{kp} \\ -\mathbf{W}_1 & \mathbf{0} & \tau_{js} & -\mathbf{W}_{up} & -\mathbf{W}_{kp} \end{bmatrix} \begin{bmatrix} \phi \\ RK_m \\ K_j \\ \phi_{up} \\ \phi_{kp} \end{bmatrix} = \mathbf{0} \quad (6)$$

$$\hat{\mathbf{W}}_{tot} \hat{\Phi}_{tot} = \mathbf{0} \quad (7)$$

where \mathbf{W}_{up} and \mathbf{W}_{kp} are the observation matrices corresponding respectively to the unknown and known payload inertial parameters.

$\hat{\mathbf{W}}_{tot}$ is the closest rank deficient matrix (Frobenius norm) from \mathbf{W}_{tot} , calculated using the singular value decomposition of $\mathbf{W}_{tot} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ because \mathbf{W}_{tot} is a full rank matrix [7]. The solution $\hat{\Phi}_{tot}$ is given by the last column of \mathbf{V} and is scaled [7] using the known position of the center of mass of the payload along the Y-axis (see Fig.1).

The success of the inertial parameters identification relies on maintaining them excited. Thus, an exciting joint trajectory is defined using a double S-curve spanning the whole range of motion of the elbow joint. The duration of each phase (acceleration, constant velocity and deceleration) of the

S-curve is set to be equal. The velocity plateau is increased by steps of 20% up to its maximal value. Joint derivatives are obtained to fulfill the regressor matrix using centered differences from the measured joint positions. All the recorded data are low-pass filtered at 5Hz (Butterworth filter, zero-phase lag order 5) and the number of sample is decimated to reduce the noise influence [5]. The S-curve motions are considered exciting since the condition number of the base parameter regressor matrix is low ($cond(\mathbf{W}_b) = 36$).

II. DIFFERENTIAL DYNAMIC PROGRAMMING OPTIMAL CONTROL SCHEME

This section presents the optimal control scheme used to find the control sequences required to perform the desired motion. In this paper we use the formulation given in [10, 4], which maximizes the performance of the desired motion with respect to the control under the actuator physical limitations constraints.

A. DDP State space representation

In state space, we denote the state vector $x = [q, \dot{q}]$ and the command vector $u = [\dot{i}_m]$. We then represent the direct dynamics model (Eq.(4)) as the following:

$$\dot{x} = f(x, u) = f(q, \dot{q}, \dot{i}_m) = \begin{bmatrix} \dot{q} \\ \frac{1}{I}(RK_m \dot{i}_m - F_v \dot{q} - F_s \text{sign}(\dot{q}) + of f) \\ - (MY \sin q + MX \cos q) \frac{g}{I} \end{bmatrix} \quad (8)$$

B. Treatment of Constraints

In comparison to the work achieved in [4], we do not solve a box QP problem to bound the command vector. Instead such as the work in [10], we have introduced the joint and actuator mechanical limits as constraints on the state and control space in the cost function formulation. The retained constraints functions are expressed by the following equations, referred as 'exponential barrier', at each time-step t :

$$\begin{aligned} \max(c_t) &= 1 - \lambda(c_{max} - c_t) \\ \min(c_t) &= 1 - \lambda(c_t - c_{min}) \end{aligned} \quad (9)$$

$$C_s(c_t) = e^{\lambda \max(c_t)} + e^{\lambda \min(c_t)}$$

where c_{min}, c_{max} are the corresponding lower and upper mechanical limits on angular position, velocity and torque, i.e. $c_t \in \{x(t), u(t)\}$. λ is a positive constant which defines the smoothness of the function. The higher it is, the quicker the cost will increase when approaching the limits (depending on the difference between the limit and the current state).

With these constraints, the total cost function will increase and reach a very high cost near the limits, keeping the system safe, in its mechanical bounds.

C. Iterative Linear Quadratic Regulator (iLQR)

To mitigate the DDP computational time it is possible to linearize the dynamics and approximate the cost function to quadratic form along the x trajectory. This method is called the iterative LQR (iLQR) approach [11] and we use the same formulation as in [10].

D. State space partial derivatives

This section details the state space partial derivatives needed by the iLQR algorithm, obtained from eq. (8). The $\text{sign}(\cdot)$ function is not differentiable in 0, therefore we have chosen to approximate the $\text{sign}(\dot{q})$ term by an hyperbolic tangent: $\tanh(\mu \dot{q})$, where $\mu = 1000$. We obtain the following new equation for the dynamic of the system:

$$f(x, u) = \begin{bmatrix} \dot{q} \\ \frac{1}{I}(RK_m \dot{i}_m - F_v \dot{q} - F_s \tanh(\mu \dot{q}) + of f) \\ - (MY \sin q + MX \cos q) \frac{g}{I} \end{bmatrix} \quad (10)$$

The partial derivative in u can be directly computed because the function is linearly dependent in \dot{i}_m :

$$f_u = \begin{bmatrix} 0.0 & \frac{1}{I} RK_m \end{bmatrix}^T \quad (11)$$

Concerning the derivative in x , we have non-linear dependencies between \ddot{q} and \dot{q} , and \ddot{q} and q . Moreover the relationship between \dot{q} and q is not explicit. Therefore we use the spatial finite difference discretization of the equation to obtain f_x at each iteration i :

$$x_{i+1} = f(x_i, u_i) \quad (12)$$

$$f_x = \begin{bmatrix} \frac{f\left(\begin{bmatrix} q_i + h/2 \\ \dot{q}_i \end{bmatrix}, u_i\right) - f\left(\begin{bmatrix} q_i - h/2 \\ \dot{q}_i \end{bmatrix}, u_i\right)}{h} \\ \frac{f\left(\begin{bmatrix} q_i \\ \dot{q}_i + h/2 \end{bmatrix}, u_i\right) - f\left(\begin{bmatrix} q_i \\ \dot{q}_i - h/2 \end{bmatrix}, u_i\right)}{h} \end{bmatrix} \quad (13)$$

E. Cost function

This section presents the cost function used in our system and the cost partial derivatives needed by the iLQR algorithm, obtained from the equation (8). Considering the actual state vector x , the desired state vector x^* , the actual command vector u and the actual torque on the elbow motor τ_m , we use the following cost function:

$$J = (x - x^*)^T Q (x - x^*) + u^T R u + C_s^T(x) W C_s(x) + C_s^T(\tau_m) P C_s(\tau_m) \quad (14)$$

$$\begin{aligned} Q &= \begin{bmatrix} 40.0 & 0.0 \\ 0.0 & 0.01 \end{bmatrix} & W &= \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 0.1 \end{bmatrix} \\ R &= 0.0001 & P &= 10.0 \end{aligned} \quad (15)$$

The weighting matrix Q, W, R and P have been chosen to give a hierarchy from the most important error to control to the lesser one. Using the lexicographical order we have: $q \succ \bar{\tau}_m, \tau_m \succ \bar{q}, \underline{q} \succ \bar{\dot{q}}, \underline{\dot{q}} \succ \dot{q} \succ u$. Where $\bar{\cdot}, \underline{\cdot}$ are the corresponding lower and upper mechanical limits. The tracking of the trajectory (in position) is the prioritized task, followed by the one controlling the torque bounds.

We obtain the following cost partial derivatives $l_u = \frac{\partial J(x, u)}{\partial u}$, $l_{uu} = \frac{\partial^2 J(x, u)}{\partial u^2}$, $l_x = \frac{\partial J(x, u)}{\partial x}$, $l_{xx} = \frac{\partial^2 J(x, u)}{\partial x^2}$, $l_{xu} = \frac{\partial^2 J(x, u)}{\partial x \partial u}$:

$$l_u = 2Ru + 2 \frac{dC_s(\tau_m)^T}{du} W C_s(\tau_m) \quad (16)$$

$$l_{uu} = 2R + 2 \left[\frac{d^2 C_s(\tau_m)^T}{du^2} W C_s + \frac{dC_s(\tau_m)^T}{du} W \frac{dC_s(\tau_m)}{du} \right] \quad (17)$$

$$\frac{dC_s(\tau_m)}{du} = \lambda(\tau_m)^2 R K_m (e^{\lambda(\tau_m)max(\tau_m)} - e^{\lambda(\tau_m)min(\tau_m)}) \quad (18)$$

$$\frac{d^2 C_s(\tau_m)}{du^2} = \lambda(\tau_m)^4 (R K_m)^2 C_s(\tau_m) \quad (19)$$

$$l_x = 2Q (x - x^*) + 2 \frac{dC_s(x)^T}{dx} W C_s \quad (20)$$

$$l_{xx} = 2Q + 2 \left[\frac{d^2 C_s(x)^T}{dx^2} C_s + \frac{dC_s(x)^T}{dx} W \frac{dC_s(x)}{dx} \right] \quad (21)$$

$$\frac{dC_s(x)}{dx} = \text{diag} \left(\begin{bmatrix} \lambda(q)^2 (e^{\lambda max(q)} - e^{\lambda min(q)}) \\ \lambda(\dot{q})^2 (e^{\lambda max(\dot{q})} - e^{\lambda min(\dot{q})}) \end{bmatrix} \right) \quad (22)$$

$$\frac{d^2 C_s(x)}{dx^2} = \begin{bmatrix} \begin{bmatrix} \lambda(q)^4 C_s(q) & 0.0 \\ 0.0 & 0.0 \end{bmatrix} \\ \begin{bmatrix} 0.0 & 0.0 \\ 0.0 & \lambda(\dot{q})^4 C_s(\dot{q}) \end{bmatrix} \end{bmatrix} \quad (23)$$

with $\lambda(\tau_m) = 0.5$, $\lambda(q) = 10$ and $\lambda(\dot{q}) = 1$. These parameters have been chosen accordingly to the explanation on the λ of the section II-B. The 'exponential barrier' on the position bounds will be sharper than the two others. It also depends on the difference between the limit and the current state. For instance the maximal difference in torque is 12A whereas the one in position is 2.35rad, then the $\lambda(\tau_m)$ do not need to be as big as the $\lambda(q)$.

III. SIMULATION AND EXPERIMENTS

A. Identifying the drive chain and inertial parameters

Fig. 2 shows the results of the least square identification. It shows the fitting of the measured motor and joint torque when no payload was used. The corresponding Root Mean Square Difference (RMSD) was 0.6 N.m showing an excellent fitting. Fig. 3.c shows the estimate of joint torque from the current of the motor that is of importance for the proposed dynamic controller.

Table 1 details the comparison between the identified parameters and their values as provided by the manufacturer.

TABLE I: Results of the identification process and comparison with manufacturer data.

	CAD	ID	$\sigma\%$
MX	-0.08	-0.11	2.1
MY	1.1	1.1	0.2
I_j	0.34	0.33	9.9
F_{Sj}	0	0.55	1.3
F_{Vj}	0	0.47	10.3
Off_j	0	-0.27	3.4
I_m	0.21	0.45	9.5
F_{Sm}	0	4.0	0.1
F_{Vm}	0	5.1	0.7
Off_m	0	0.86	7.9

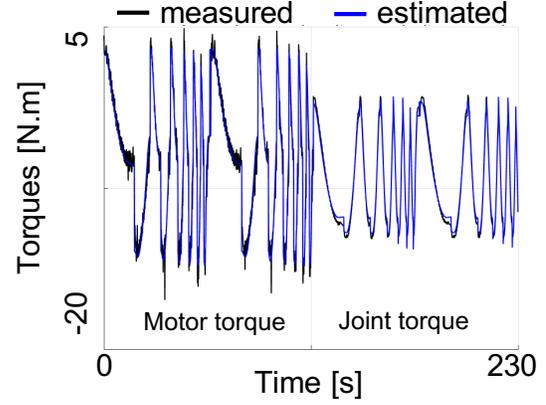


Fig. 2: Results of the fitting of motor and joint torques used for the identification process.

Overall the identified segment inertial parameters are similar to the ones provided by the manufacturer. These parameters can be considered well identified due to their relatively low standard deviation $\sigma\%$. The relative standard deviation gives in % a confidence index on the reliability of the identification of each parameter. See [15] for the detail of $\sigma\%$ calculation. The joint torque sensor was well calibrated since its identified gain was $K_j = 1.015$. The total joint drive gain RK_m value is not disclosed, because of a confidentiality agreement with the robot manufacturer, but a difference of 16% with the value provided by the manufacturer was found.

B. Controlling the actuator

The iLQR is implemented as described in [4] and completed by the 'exponential barrier' constraints in the cost function. The cost functions and model dynamics are evaluated using the identified parameters (Table.I). This implementation is first validated in simulation with the use of Gazebo and then tested on the real TALOS robot. The noticeable contribution of this paper compared to [4] remains in this last point, the torque control is implemented on the robot and achieve a satisfying real time control (with more protections on the mechanical limits of the robot).

The experimental setup is the following: the robot is controlled in torque with high gains except for the elbow actuator. A sinusoidal command for the elbow actuator is sent to the DDP algorithm which computes a resulting signal respecting the limits. This signal is then sent to the robot. A heavy charge is finally put incrementally on the arm of the robot, at 5cm of the elbow joint, until reaching 34kg (see Fig.4). The load moment arm when the load is perpendicular to the robot arm is equal to 16.67 N.m. Otherwise it is expressed similarly to the mass of the arm at the center of mass (see Eq.(8)): $LY \sin q + LX \cos q$ with LY and LX the first moment of inertia of the load.

The DDP algorithm is executed on the robot with a 15ms preview window, a 3.3ms discretization and has an execution time of 300 μ s (using an Intel CPU i7-3612QE @ 2.1 GHz). On a standard laptop such as an Intel CPU

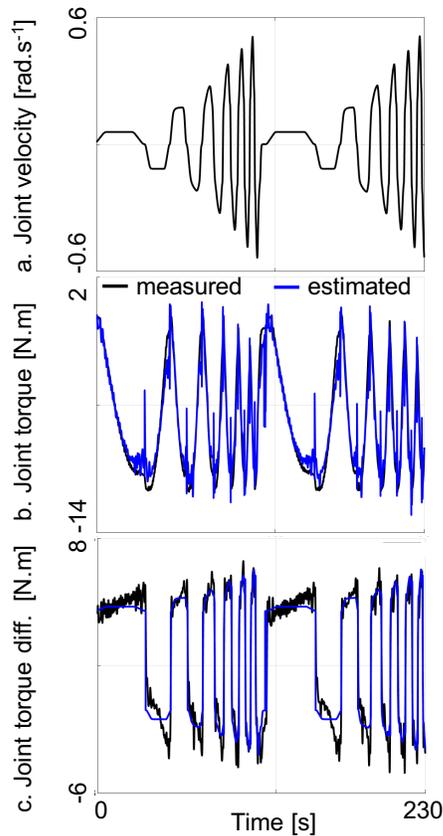


Fig. 3: (a) Velocity profile used to excite the dynamics system. (b) Estimate of the joint torques from motor current and the identified model. (c) Estimation of the difference between the joint and motor torque.

i7-8850H @ 2.6 GHz it can be executed with a longer preview window (100ms) and with a faster control frequency (1ms discretization) in a smaller execution time (200μs).

1) *Simulation*: Fig.5 presents the action of the exponential barriers on the position joint limits (see Eq.9). A sinusoidal desired trajectory is given to track, but reaches the lower angular position limit: $-2.35rad$. The computed trajectory shows a plateau before reaching the limit, demonstrating the activation of the protection in the DDP. We can notice the small oscillations at the top of the sinusoid, due to the dry frictions when the angular velocity reaches zero (see Fig.9). This causes a small delay in the computed trajectory.

Fig.6 depicts the tracking results on two cases: first without additional load and second by adding a 30 kg load to the forearm. In the first case, the trajectory follows the desired one with a small delay ($\sim 0.1s$), which may be caused by the dry friction as explained above. This bias can be removed by increasing the state constraint gains but the system will lose compliance. In the second case, the delay with the desired trajectory increases, in particular during the ascending phase of the sinusoidal command (the robot raises its arm and the load). Indeed, to match the desired trajectory quicker,



Fig. 4: Experiment where TALOS is holding 34kg at 5 cm of the elbow joint.

the torques needed are high and the DDP algorithm limits them. Nevertheless, in both cases the overall motion shape is respected.

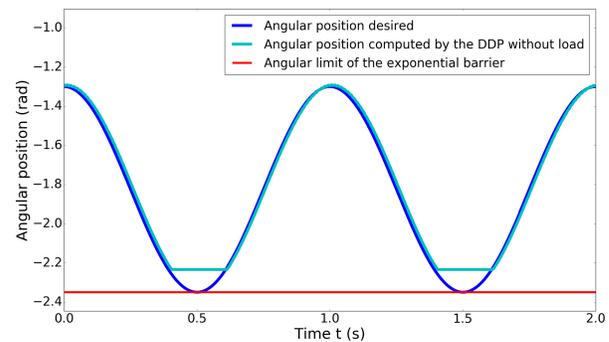


Fig. 5: Simulated state trajectory illustrating the angular exponential barrier.

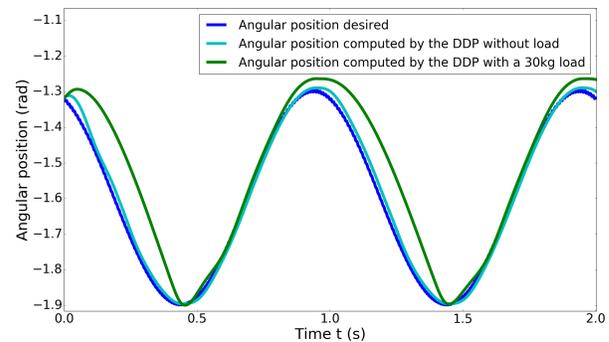


Fig. 6: Comparison between simulated trajectories with and without load.

The control trajectories depicted in Fig.7 show how the limitation of 6 A (set *a-priori*) is implemented with the exponential barrier given by Eq.9. The cyan trajectory represents

the current command when there is no additional load and it does not reach the limit. The green trajectory represents the current command when the robot carries a 30kg load, it shows that the command reaches a plateau (around -5.6A) before the limit. It explains the delay of the computed angular trajectory of Fig.6, the tracking of the desired joint trajectory is degraded because the current is limited. This is due to the DDP action to preserve the actuator, i.e. the exponential barrier is activated by the DDP algorithm.

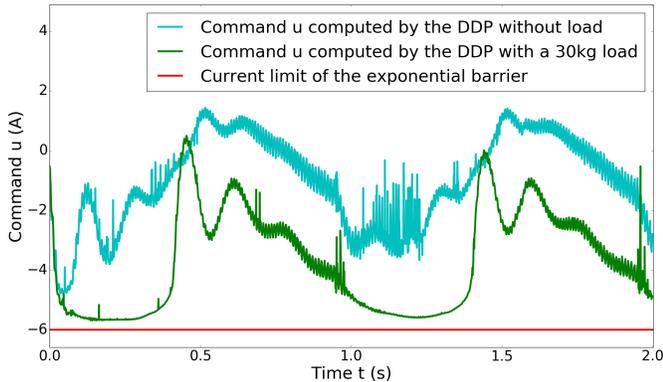


Fig. 7: Simulated control trajectories with and without an additional load.

In comparison, Fig.8-bottom depicts the control going over the current limit in the loaded case when the current exponential barrier is not enabled in simulation. In this configuration the joint position tracking is better, as shown in Fig.8-up, but the actuator reaches its current limit.

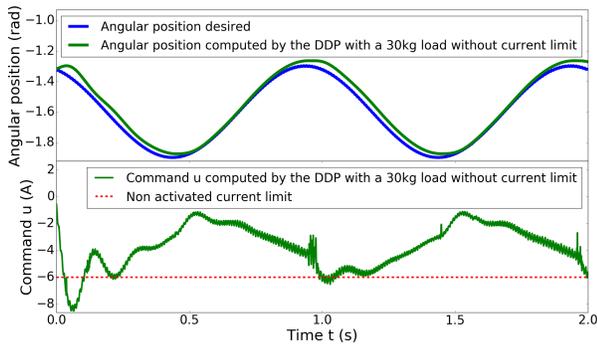


Fig. 8: Simulated state and control trajectories in loaded case but without control input limitation.

2) Experiments: Fig.9 presents the action of the exponential barriers on the position joint limits on the real robot. As in simulation, the desired trajectory reaches the lower angular position limit: $-2.35rad$. The computed trajectory shows a plateau before reaching the limit, demonstrating the efficiency of the protection in the DDP. Notice the oscillations at the beginning of the sinusoidal movement (when the arm is raising), the arm of the robot has difficulties to perform a smooth trajectory. As thought in the simulations, this is the consequence of the dry frictions, when the angular velocity \dot{q}

is approaching zero, the dry coefficient F_s increases to reach the Breakaway friction value (the sum of the Coulomb and static frictions). This behaviour can create the delay noticed in the simulations. This is not currently taken into account in our model but would be in the future.

Fig.10 shows a comparison between the desired joint trajectory and the ones obtained without and with a 34kg load (we increase the load to have a better demonstration of the activation of the current exponential barrier). As expected, the trajectory without any load has a small delay and oscillates when the angular velocity is around zero. The trajectory obtained when the robot is carrying the additional 34 kg displays greater oscillations due to the load movements. It also presents a bigger degradation of the trajectory than in simulation, the sinusoidal movement is reduced (stopped at $-1.7rad$ instead of $-1.9rad$) because large torques are necessary to achieve it (but are prohibited by the DDP algorithm, see Fig.11).

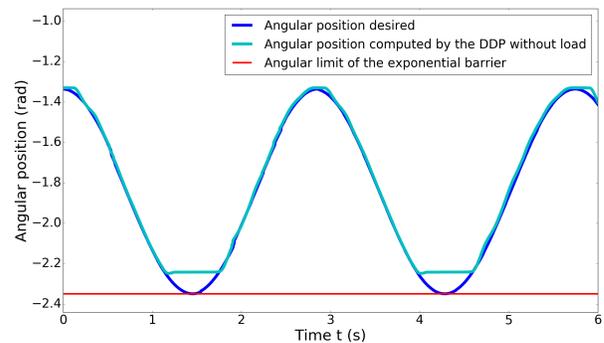


Fig. 9: Experiments - State trajectory illustrating the angular exponential barrier.

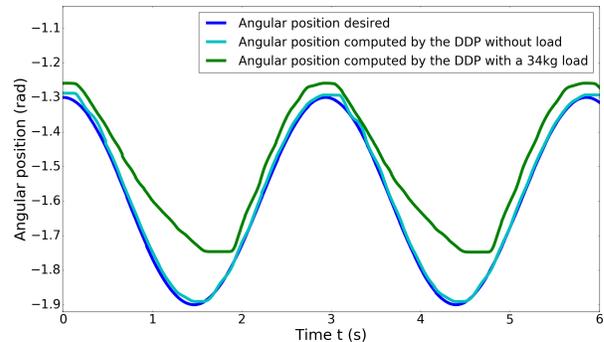


Fig. 10: Experiments - State trajectories with and without a 34kg load.

The Fig.11 presents the control trajectories computed by the DDP without and with the load. In the first case (cyan line), the command is far from the limit and do not activate the barrier. In the second case (green line), as in simulation, the command reaches a plateau before the limit, around -5.3 to $-5.6A$. The current is more reduced than in simulation, inducing a more degraded state trajectory in Fig.10. The plateau is not as smooth as in the simulation, due to the

computation time which is quite high, leading to picks in the calculation. Indeed, in these experiments the computation time of the overall control system is around 1 to 1.5ms. Compared to the control frequency of the robot, which is 1kHz, this duration is large.

A video describing the experiments on the robot can be found at the following location: <https://youtu.be/YNoSnU7w4FY>.

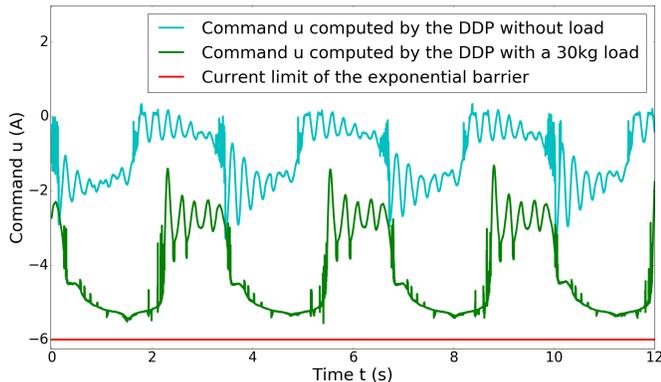


Fig. 11: Experiments - Control trajectories with and without a 34kg load.

IV. CONCLUSION

In this paper we presented the actuator model, the identification and the control of the TALOS robot elbow joint. The identification results of the drive chain and of the inertial parameters at once proved to be accurate with low standard deviation and physical consistency of the parameters. Using the identified model and a DDP approach to avoid reaching its limits, we have demonstrated that the robot is able to carry a load up to 34 kg with a sinusoidal motion at low speed. As expected, it is not possible to use this algorithm in the main CPU as it takes 300μs for one actuator, nevertheless, we validated the efficiency of the solution. The extension of this work would be to identify all the actuators of the robot and to implement the solution over high-performance dedicated and embedded electronics board attached to each actuator. For a short term solution on the TALOS robot, we will use state-of-the-art and less computationally expensive techniques [1].

ACKNOWLEDGMENTS

This work is supported by the cooperation agreement ROB4FAM.

REFERENCES

[1] A. Albu-Schäffer, C. Ott, and G. Hirzinger. A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *The international journal of robotics research*, 2007.

[2] R. Cisneros, M. Benallegue, A. Benallegue, M. Morisawa, H. Audren, P. Gergondet, A. Escande, A. Kheddar, and F. Kanehiro. Robust humanoid control using a qp solver with integral gains. In *IEEE/RSJ IROS*, 2018.

[3] J. Engelsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schäffer. Overview of the torque-controlled humanoid robot toro. In *IEEE/RAS ICHR*, 2014.

[4] F. Forget, K. Giraud-Esclasse, R. Gelin, N. Mansard, and O. Stasse. Implementation, identification and control of an efficient electric actuator for humanoid robots. In *International Conference on Informatics in Control, Automation and Robotics, ICINCO*, 2018.

[5] M. Gautier. Dynamic identification of robots with power model. *ICRA*, 1997.

[6] M. Gautier and S. Briot. Global identification of joint drive gains and dynamic parameters of robots. *ASME Journal of Dynamic Systems*, 2014.

[7] M. Gautier and A. Jubien. Force calibration of KUKA lwr-like robots including embedded joint torque sensors and robot structure. In *IEEE/RSJ IROS*, 2014.

[8] O. Khatib, L. Sentis, J. Park, and J. Warren. Whole-body dynamic behavior and control of human-like robots. *International Journal of Humanoid Robotics*, 2004.

[9] O. Khatib, P. Thaulad, T. Yoshikawa, and J. Park. Torque-position transformer for task control of position controlled robots. In *IEEE/RAS ICRA*, 2008.

[10] G. Kumar Hari Shankar Lal Das, B. Tondu, F. Forget, J. Manhes, O. Stasse, and P. Soueres. Controlling a multi-joint arm actuated by pneumatic muscles with quasi-ddp optimal control. In *IEEE/RSJ IROS*, 2016.

[11] W. Li and E. Todorov. Iterative linearisation methods for approximately optimal control and estimation of non-linear stochastic system. *Int. Jour. of Control*, 2007.

[12] F. Nori, S. Traversaro, J. Eljaik, F. Romano, A. Del Prete, and D. Pucci. icub whole-body control through force regulation on rigid non-coplanar contacts. *Frontiers in Robotics and AI*, 2015.

[13] PAL-Robotics. Talos - torque controlled balancing, 2018. URL <https://www.youtube.com/watch?v=1X6Q0zgFut0>.

[14] S. Park, J. Sim, and J. Park. System design of humanoid robot dyros-jet. In *IEEE/SICE International Symposium on System Integration (SII)*, 2019.

[15] C. Presse and M. Gautier. New criteria of exciting trajectories for robot identification. In *IEEE/RAS ICRA*, 1993.

[16] A. Del Prete, N. Mansard, O. Ponce O., Stasse, and F. Nori. Implementing torque control with high-ratio gear boxes and without joint-torque sensors. *International Journal of Humanoid Robotics*, 2015.

[17] O. Stasse, T. Flayols, R. Budhiraja, K. Giraud-Esclasse, J. Carpentier, and al.. Talos: A new humanoid research platform targeted for industrial applications. In *IEEE/RAS ICHR*, 2017.

[18] E. Dombre W. Khalil. *Modeling, identification and control of robots*. Eds. Hermès Penton, London, United Kingdom, 2002.