2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Automatic Task-specific Model Reduction for Humanoid Robots

Umashankar Nagarajan and Katsu Yamane

Abstract— **Simple inverted pendulum models and their variants are often used to control humanoid robots in order to simplify the control design process. These simple models have significantly fewer degrees of freedom than the full robot model. The design and choice of these simple models are based on the designer's intuition, and the reduced state mapping and the control input mapping are manually chosen. This paper presents an automatic model reduction procedure for humanoid robots, which is task-specific. It also presents an optimization framework that uses the auto-generated task-specific reduced models to control humanoid robots. Successful simulation results of balancing, fast arm swing, and hip rock and roll motion tasks are demonstrated.**

I. Introduction

Control of humanoid robots is a challenging task by virtue of its instability and complexity with large number of degrees of freedom (DOF). Traditionally, researchers have used simplified models like the ones shown in Fig. 1 to design controllers for humanoid robots. These simplified models have significantly fewer DOF than the full robot model, and are often linearized to apply techniques from linear control theory. The linear inverted pendulum model (LIPM) is the most common simplified model used in the literature of humanoid robot control [1], where a lump mass is connected to the ground with a rotational joint. Several other variants such as the spring loaded inverted pendulum (SLIP) model [2], cart-table model [3], reaction mass pendulum (RMP) model [4], double inverted pendulum (DIP) model [5] and linear biped model (LiBM) [6] have been proposed in the literature for humanoid robot control.

One of the main concerns is that the choice and design of the above mentioned simplified models were based on the designer's intuition. The LIPM model was proposed to achieve ankle strategy for balance, while the DIP model was proposed to achieve both ankle and hip strategies [7]. The SLIP model was proposed for hopping and running tasks, while the RMP model was proposed to account for the centroidal angular momentum of the system. The LiBM model was proposed to explicity account for double-support and single-support phases. In [8], Goswami derived and compared the physical properties between planar RMP and compass-gait models. Apart from this work, not much research has been done to investigate how well these simplified models match the dynamics of the original, high-dimensional system. Moreover, in addition to the choice of the model, the designer needs to pick two different mappings, one that maps the state of the full model to the reduced state, and the other that maps the control inputs of the reduced model to those of the full model. The control input mapping is especially tricky since there are infinite possible mappings from a low-dimensional space to a high-dimensional space. Generally, criteria like kinetic energy equivalence or angular momentum equivalence are used to pick these mappings.

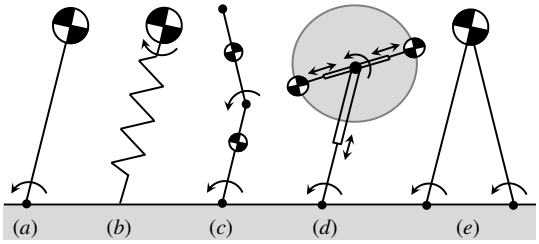


Fig. 1. Simplified planar models used for humanoid robot control: (a) Linear inverted pendulum model (LIPM), (b) Spring-loaded inverted pendulum (SLIP), (c) Double inverted pendulum (DIP), (d) Reaction mass pendulum (RMP), and (e) Linear biped model (LiBM).

The main objective of this paper is to replace the manual, intuitive model simplification process with an automatic model reduction procedure for humanoid robots. One such approach was presented in [9], where the reduced model was automatically computed with the objective of matching the kinetic energy of the system, and the reduced DOF corresponded to the smallest singular values of the inverse of the mass/inertia matrix. However, unlike [9], in this paper, we present a model reduction approach that finds the smallest order statespace model, whose stabilizing controller stabilizes the full humanoid model. Moreover, the model reduction is task-specific because one intuitively understands that a complicated humanoid robot task like manipulation is higher dimensional than a simple balancing task.

This paper builds on existing model reduction techniques from linear control theory like balanced truncation [10] and fractional balanced reduction [11] to develop *Minimum Stable Model Reduction* (MSR) algorithm that finds the smallest reduced order system, whose stabilizing controller stabilizes the original high-dimensional system. The MSR algorithm is made task-specific (TMSR) by formulating the original system with task-specific outputs. This paper presents model reduction results on linear humanoid robot models, which show that the reduced order increases with increasing complexity of the task. Moreover, this paper also presents an optimization framework that uses these reduced order models to successfully control a 34 DOF nonlinear humanoid robot model in simulation for balancing, fast arm swing, and hip rock and roll motion tasks.

U. Nagarajan and K. Yamane are with Disney Research Pittsburgh, PA, 15213 USA umashankar@disneyresearch.com, kyamane@disneyresearch.com

II. BACKGROUND

This section introduces model reduction in a control-theoretic sense and describes a few existing model reduction techniques in linear control theory.

Consider a dynamic system with a state vector $x \in \mathbb{R}^n$, an input vector $u \in \mathbb{R}^m$ and an output vector $y \in \mathbb{R}^p$. Its state space equations are given by:

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx, \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{p\times n}$. Its output transfer function is given by $G(s) = C(sI_n - A)^{-1}B$, where I_n is a $n \times n$ identity matrix.

A reduced order system with fewer states $x_r \in \mathbb{R}^r$ with $r < n$ but with the same inputs and outputs is given by:

$$\begin{aligned} \dot{x}_r &= A_r x_r + B_r u, \\ y &= C_r x_r, \end{aligned} \tag{2}$$

where $A_r \in \mathbb{R}^{r\times r}, B_r \in \mathbb{R}^{r\times m}$ and $C_r \in \mathbb{R}^{p\times r}$. Its output transfer function is given by $G_r(s) = C_r(sI_r - A_r)^{-1}B_r$, where I_r is a $r \times r$ identity matrix.

The objective of model reduction is to find a reduced order model such that $||G - G_r||_\infty$ is minimized [10].

A. Balanced Truncation of Stable Systems

Algorithm 1 presents the square-root method for *Balanced Truncation* (BT), an existing model reduction technique for stable systems [10]. As its name indicates, it consists of two steps: (i) balancing the system, and (ii) truncating the balanced system. A stable system of the form shown in Eq. 1 is said to be *balanced* if its controllability gramian $P \in \mathbb{R}^{n\times n}$ and observability gramian $Q \in \mathbb{R}^{n\times n}$ obtained from Step 2 of Algorithm 1 are equal and diagonal positive definite matrices, *i.e.*, $P = Q = \Sigma > 0$. This implies that each balanced state is equally controllable and observable. The balanced state vector is given by $x_b = T_b x$, where $T_b \in \mathbb{R}^{n\times n}$ is obtained from Step 6 of Algorithm 1.

A measure of controllability and observability of each state can be obtained from its Hankel singular value [12], [13]. The Hankel singular values of a balanced system with controllability and observability gramians P_b and Q_b is given by $\sigma_{HSV} = \sqrt{\lambda(P_b Q_b)}$, where $\lambda(\cdot)$ computes the eigenvalues. The largest Hankel singular values correspond to the most controllable and observable states, whereas the smallest singular values correspond to the least controllable and observable states. Therefore, in order to reduce the system with n states in Eq. 1 to a system with r states in Eq. 2, one can pick the r most controllable and observable states from its balanced state vector x_b. The reduced state vector is given by $x_r = T_r x$, where $T_r \in \mathbb{R}^{r\times n}$ is obtained from Step 9 of Algorithm 1.

B. Fractional Balanced Reduction of Unstable Systems

The balanced truncation algorithm presented in Algorithm 1 cannot be applied to unstable systems because their controllability and observability gramians cannot be computed due to lack of unique solutions to their corresponding Lyapunov equations (Step 2). Since humanoid robots are unstable, we present another existing algorithm called *Fractional Balanced Reduction* (FBR) for unstable systems [11] in Algorithm 2. The FBR algorithm stabilizes the unstable system, balances and truncates the stabilized system, and then retrieves the reduced unstable system.

A system with the statespace realization $\{A, B, C\}$ shown in Eq. 1 is stabilized using the change of input $\bar{u} = Kx - u$,, where $K = -B^T N$ and N is the solution to the algebraic Riccati equation shown in Step 2 of Algorithm 2. The statespace equations for this stabilized system are given by:

$$\begin{aligned} \dot{x} &= \bar{A}x + B\bar{u}, \\ \begin{bmatrix} y \\ u \end{bmatrix} &= \begin{bmatrix} C \\ K \end{bmatrix} x + \begin{bmatrix} 0 \\ I \end{bmatrix} \bar{u}. \end{aligned} \tag{3}$$

The stable state space realization $\{\bar{A}, B, \begin{bmatrix} C \\ K \end{bmatrix}\}$ in Eq. 3 is balanced and truncated to $\{\bar{A}_r, B_r, \begin{bmatrix} C_r \\ K_r \end{bmatrix}\}$ using Algorithm 1. The reduced system $\{A_r, B_r, C_r\}$ of the original unstable system $\{A, B, C\}$ is obtained by undoing the effects of the stabilizing controller K with $A_r = \bar{A}_r - B_r K_r$, as shown in Step 5 of Algorithm 2.

Algorithm 1: Balanced Truncation (BT)

input : System $\{A, B, C\}$, Reduced Order r
output : Reduced System $\{A_r, B_r, C_r\}$, Reduced State Transformation T_r
function: $[A_r, B_r, C_r, T_r, T_n]$ = BT(A, B, C, r)

1 **begin**
2 Solve Lyapunov equations to get controllability and observability gramians P, Q
$AP + PA^T + BB^T = 0$
$A^T Q + QA + C^T C = 0$
3 Find Cholesky factors L_P and L_Q
$L_p = U_p\sqrt{S_p}$, where $P = U_p S_p V_p^T$
$L_q = U_q\sqrt{S_q}$, where $Q = U_q S_q V_q^T$
4 Get singular value decomposition of $L_q^T L_p$
$L_q^T L_p = U_b S_b V_b^T$
5 Get balanced transformation matrices
$T_1 = L_q U_b S_b^{-\frac{1}{2}}$
$T_2 = L_p V_b S_b^{-\frac{1}{2}}$
6 Get balanced state transformation
$T_b = T_2^{-1} \in \mathbb{R}^{n\times n}$
7 Get reduced transformation matrices
$T_3 = T_1(1:n, 1:r) \in \mathbb{R}^{n\times r}$
$T_4 = T_2(1:n, 1:r) \in \mathbb{R}^{n\times r}$
8 Get reduced system
$A_r = T_3^T A T_4$
$B_r = T_3^T B$
$C_r = C T_4$
9 Get reduced state transformation
$T_r = T_b(1:r, 1:n) \in \mathbb{R}^{r\times n}$
10 **end**

Algorithm 2: Fractional Balanced Reduction (FBR)

input : System $\{A, B, C\}$, Reduced Order r
output : Reduced System $\{A_r, B_r, C_r\}$, Reduced State Transformation T_r
function: $[A_r, B_r, C_r, T_r]$ = FBR(A, B, C, r)

1 **begin**
2 Solve algebraic Riccati equation
$A^T N + NA - NBB^T N + C^T C = 0$
3 Get stabilized system $\{\bar{A}, \bar{B}, \bar{C}\}$
$\bar{A} = A + BK$, where $K = -B^T N$
$\bar{B} = B$
$\bar{C} = \begin{bmatrix} C \\ K \end{bmatrix}$
4 Get balanced truncated system $\{\bar{A}_r, \bar{B}_r, \bar{C}_r\}$
$[\bar{A}_r, \bar{B}_r, \bar{C}_r, T_r]$ = BT$(\bar{A}, \bar{B}, \bar{C}, r)$
5 Retrieve the reduced system $\{A_r, B_r, C_r\}$
$K_r = \bar{C}_r(p+1 : p+m, 1 : r)$
$A_r = \bar{A}_r - \bar{B}_r K_r$
$B_r = \bar{B}_r$
$C_r = \bar{C}_r(1 : p, 1 : r)$
6 **end**

III. Minimum Stable Balanced Reduction

The FBR algorithm presented in Algorithm 2 reduces an unstable system in Eq. 1 to a reduced system in Eq. 2 for a given reduced order r. However, the FBR algorithm did not present any approach to pick the reduced order r. This section presents a new algorithm called the *Minimum Stable Balanced Reduction* (MSR), shown in Algorithm 3, which finds the minimum reduced order r_{min} such that the linear quadratic regulator (LQR) that stabilizes the corresponding reduced system also stabilizes the original full system.

The MSR algorithm is an iterative procedure that begins with initializing r to the number of unstable poles of the open-loop system $\{A, B, C\}$, which ensures that the unstable subsytem is retained while reducing the model. In [9], the unstable subsystem was not necessarily retained as the user picked the reduced DOF. At each iteration of r, the system $\{A, B, C\}$ in Eq. 1 is reduced to $\{A_r, B_r, C_r\}$ in Eq. 2 using the FBR algorithm in Algorithm 2 as shown in Step 4 of Algorithm 3. The reduced state transformation T_r is used to transform the symmetric LQR gain matrix on the full states $Q \in \mathbb{R}^{n \times n}$ to a symmetric LQR gain matrix on the reduced states $Q_r \in \mathbb{R}^{r \times r}$ with $Q_r = T_r Q T_r^T$. This transformation allows the user to pick Q for the original states of the system, which is more intuitive than picking the same for the reduced states. Moreover, this allows the model reduction process to be automated since there is no need to pick Q_r for each reduced order r. The LQR gain matrix on the control inputs $R \in \mathbb{R}^{m \times m}$ chosen by the user is used as it is because the control inputs remain the same for the reduced order model as shown in Eq. 2. The stabilizing control law is given by $u = -K_r x_r$, where the control gain matrix $K_r = R^{-1} B_r^T S \in \mathbb{R}^{m \times r}$, and S is obtained by solving the associated Riccati equation in Step 5 of Algorithm 3.

Since the reduced state $x_r = T_r x$, the resulting control law for the full order system shown in Eq. 1 is given by

Algorithm 3: Minimum Stable Balanced Reduction (MSR)

input : System $\{A, B, C\}$, LQR Gains Q, R
output : Minimum Stable Reduced Order r_{min}, Reduced System $\{A_r, B_r, C_r\}$, Reduced State Transformation T_r
function: $[r_{min}, A_r, B_r, C_r, T_r]$ = MSR(A, B, C)

1 **begin**
2 Get the unstable open-loop poles and initialize r
$p_{unstab} = \{\lambda_i | \lambda_i \in \lambda(A) > 0\}$
r = size(p_{unstab})
3 **while** $p_{unstab} \neq \emptyset$ ***and*** $r \leq n$ **do**
4 Get balanced truncated system $\{A_r, B_r, C_r\}$
$[A_r, B_r, C_r, T_r]$ = FBR(A, B, C, r)
5 Get LQR control gain matrix K_r with $Q_r = T_r Q T_r^T$ and $R_r = R$ by solving its associated Riccati equation
$A_r^T S + S A_r - S B_r R_r^{-1} B_r^T S + Q_r = 0$
$K_r = R_r^{-1} B_r^T S$
6 Get the unstable closed-loop poles p_{unstab}
$p_{unstab} = \{\lambda_i | \lambda_i \in \lambda(A – B K_r T_r) > 0\}$
7 $r = r + 1$
8 **end**
9 Get the minimum reduced system$\{A_r, B_r, C_r\}$
$r_{min} = r - 1$
$[A_r, B_r, C_r, T_r]$ = FBR(A, B, C, r_{min})
10 **end**

$u = -K_r T_r x$. The eigenvalues λ of the closed-loop state transition matrix $A_{cl} = \big(A - B K_r T_r\big)$ determine the stability of the closed-loop system, wherein positive eigenvalues form the set of unstable poles p_{unstab} as shown in Step 6 of Algorithm 3. If the closed-loop system is unstable, then the reduced order r is incremented by one, and the whole process repeats until a stable closed-loop system is reached. The minimum stable reduced order r_{min} and the reduced order realization $\{A_r, B_r, C_r\}$ are retrieved as shown in Step 9 of Algorithm 3. If the system $\{A, B, C\}$ with n states is controllable, then $r_{min} \leq n$.

In this work, we are interested in model reduction for humanoid robots, and intuitively, one understands that it must be task-specific. For example, a balancing task for a humanoid robot is lower dimensional than a complex manipulation task. Hence, the balancing task can be achieved with a reduced model whose order is smaller than that required for achieving the manipulation task. Here, we present a task-specific variant of the MSR algorithm called the *Task-specific Minimum Stable Balanced Reduction* (TMSR).

The balanced truncation algorithms (Sec. II) used by the MSR algorithm reduce the system while minimizing the H_∞ norm of the difference in the transfer functions of the full and reduced order systems, $i.e.$ $||G - G_r||_\infty$. Here, the transfer function deals with the effect of the inputs on the outputs. It is important to note that the inputs and outputs for the reduced system in Eq. 2 are the same as the ones for the full system in Eq. 1. Roughly speaking, the balanced truncation algorithms reduce the difference in energy transfer from the inputs to the outputs. This implies that the output

matrix $C \in \mathbb{R}^{p\times n}$ that maps the states to the outputs also plays an important role in model reduction. In order to make model reduction task-specific, we propose to make the output matrix C that influences the transfer function task-specific. Therefore, the TMSR algorithm uses the MSR algorithm shown in Algorithm 3, while picking task-specific output matrices C in the state space realization of the system $\{A, B, C\}$ used for model reduction.

IV. Model Reduction for Humanoid Robots

This section discusses task-specific model reduction of linear humanoid robot models using the MSR and TMSR algorithms presented in Sec. III.

A. Humanoid Robot Model

This section presents a humanoid robot model in double-support as shown in Fig. 2(a). The model has 34 DOF with 7 DOF for each of its legs, 6 DOF for its torso, 4 DOF for each of its arms and 6 DOF for its root joint. The wrist and facial DOF are ignored in this model. The root joint is unactuated, whereas the remaining 28 DOF are actuated. In this work, the humanoid model is constrained to not move its feet and hence, there are six constraints for each foot. The linear equations of motion of the system with the configuration vector $q \in \mathbb{R}^{34}$ can be written as:

$$M\ddot{q} + D\dot{q} + Gq \;=\; F^T\tau + J_c^T f_c, \tag{4}$$

where $M \in \mathbb{R}^{34\times34}$ is the mass/inertia matrix, $D \in \mathbb{R}^{34\times34}$ is the damping matrix, $G \in \mathbb{R}^{34\times34}$ is the stiffness matrix, $F \in \mathbb{R}^{28\times34}$ is the input coupling matrix, $\tau \in \mathbb{R}^{28}$ is the input vector, $J_c \in \mathbb{R}^{12\times34}$ is the contact Jacobian of both feet and $f_c \in \mathbb{R}^{12}$ is the generalized vector of contact forces for the feet. The contact constraints on the feet given by $J_c\dot{q} = 0 \in \mathbb{R}^{12}$ can be differentiated to get

$$J_c\ddot{q} + \dot{J}_c\dot{q} \;=\; 0 \in \mathbb{R}^{12}. \tag{5}$$

The equations of motion in Eq. 4 can be re-written as:

$$\ddot{q} \;=\; -M^{-1}(D\dot{q} + Gq - F^T\tau - J_c^T f_c). \tag{6}$$

Solving for f_c from Eq. 5 and Eq. 6, one gets

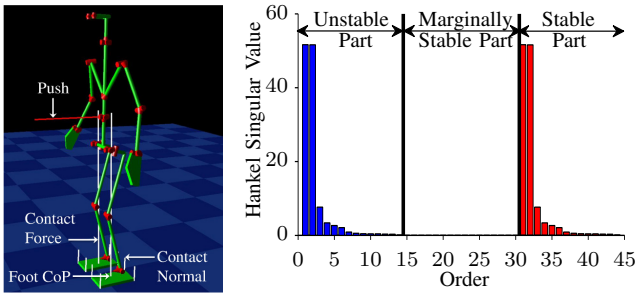
$$f_c = \left(J_cM^{-1}J_c^T\right)^{-1}\left[J_cM^{-1}(D\dot{q}+Gq-F^T\tau)-\dot{J}_c\dot{q}\right]. \tag{7}$$

Solving for $\ddot{q}$ from Eq. 6 and Eq. 7, one gets

$$\ddot{q} = -N_2Gq - \left(N_2D + N_1\dot{J}_c\right)\dot{q} + N_2F^T\tau, \tag{8}$$

where $N_1 = M^{-1}J_c^T\left(J_cM^{-1}J_c^T\right)^{-1}$ and $N_2 = \left(I_{34} - N_1J_c\right)M^{-1}$. Here, I_{34} is a 34×34 identity matrix. The state space matrices A and B of the resulting linear humanoid model with feet constraints and state vector $x = [q^T, \dot{q}^T] \in \mathbb{R}^{68}$ are given by:

$$A \;=\; \begin{bmatrix} 0_{34\times34} & I_{34} \\ -N_2G & -\left(N_2D + N_1\dot{J}_c\right) \end{bmatrix} \in \mathbb{R}^{68\times68},$$
$$B \;=\; \begin{bmatrix} 0_{34\times28} \\ N_2F^T \end{bmatrix} \in \mathbb{R}^{68\times28}, \tag{9}$$



(a) (b)

Fig. 2. (a) Simulated 34 DOF humanoid robot; (b) Hankel singular values of the minimal system with 44 states.

where $0_{a\times b}$ is $a\times b$ matrix with zeros as its elements and I_{34} is an 34×34 identity matrix. The output matrix $C \in \mathbb{R}^{p\times68}$ is chosen based on the task-specific output vector $y \in \mathbb{R}^p$.

It is important to note that the robot has 34 DOF and 12 constraints, which implies that its net DOF is 22. Therefore, the minimal state vector $x_m \in \mathbb{R}^{44}$ is obtained by finding a state transformation U_m such that $x_m = U_mx$, where $U_m \in \mathbb{R}^{44\times68}$. The minimal state space realization is given by $\{A_m, B_m, C_m\} = \{U_mAU_m^T, U_mB, CU_m^T\}$. The minimal realization of a state space system can be obtained using Kalman decomposition [14]. In this work, we use *minreal*() function from MATLAB to obtain the minimal realization.

B. Task-specific Model Reduction Results

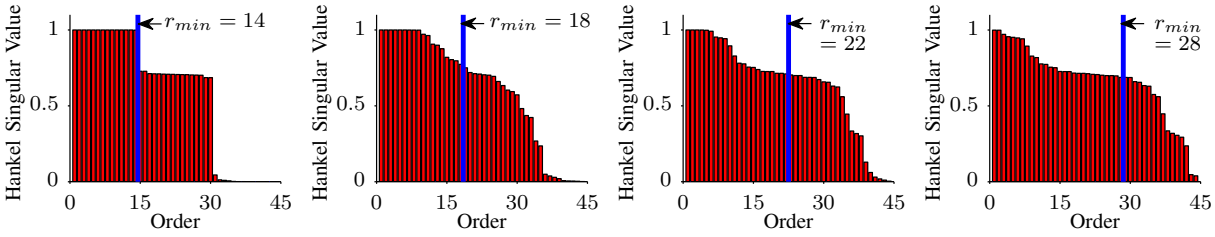
As mentioned in Sec. III, the output matrix C can be chosen to be task-specific. This section presents several task-specific output matrices for the humanoid model in Eq. 9, and it also presents their corresponding minimum reduced orders r_{min} obtained from the TMSR variant of the MSR algorithm presented in Algorithm 3. The task-specific output matrices presented here are manually chosen, and approaches to automate this selection will be explored in the future. For an output matrix $C = I_{68}$, the Hankel singular values (σ_{HSV}) of the minimal system with 44 states are shown in Fig. 2(b). The minimal system has 14 unstable poles and 14 stable poles. The remaining 16 poles lie on the imaginary axis with $\sigma_{HSV} = \infty$ and constitute the set of marginally stable poles as shown in Fig. 2(b). A similar plot is obtained for different choices of the output matrix C.

The minimum stable reduced orders r_{min} obtained from the TMSR algorithm for four different tasks/motions with increasing complexity are listed in Table I along with their corresponding task-specific outputs. For a simple balancing task, the outputs are chosen to be the position and velocity of the center of mass (CoM) of the robot. Therefore, the output matrix C is given by the Jacobian of the CoM state vector $x_{com} \in \mathbb{R}^6$ w.r.t. the states of the original unconstrained system $x \in \mathbb{R}^{68}$, and the minimal output matrix C_m is obtained using *minreal*() in MATLAB. As shown in Table I, for the balancing task, the system can be reduced to a system with 14 states as compared to 44 states of the minimal system. As shown in Fig. 2(b), the minimal system has 14 unstable poles, and interestingly, for the balancing task, the TMSR algorithm reduced the system

TABLE I

TASK-SPECIFIC MINIMUM STABLE REDUCED ORDERS

Model No.	Task/Motion	Task-specific Output	Minimum Stable Reduced Order r_{min}
R1	Balance	6 CoM states	14
R2	Balance + Lower body motion	6 CoM states + 28 Lower body states	18
R3	Balance + Upper body motion	6 CoM states + 28 Upper body states	22
R4	Balance + Whole body motion	6 CoM states + All 68 states	28



(a) Reduced model R1 (b) Reduced model R2 (c) Reduced model R3 (d) Reduced model R4

Fig. 3. Normalized Hankel singular values of the stabilized minimal system before model reduction. The task-specific r_{min} are also marked.

to the minimum order possible as the unstable subsystem should not be removed during model reduction.

However, as the tasks get more complicated, the minimum stable order of the reduced system increases as expected. For example, the task of balancing and moving the lower body requires a minimum of 18 states, while the task of balancing and moving the upper body (including arms) requires a minimum of 22 states. The task of balancing and moving the whole body requires a minimum of 28 states to represent its reduced order model.

As described in Sec. III, the MSR algorithm presented in Algorithm 3 uses the FBR algorithm, which performs balanced truncation on the stabilized system as shown in Step 3−4 in Algorithm 2. Figure 3 shows the normalized Hankel singular values of the stabilized systems in descending order before their balanced truncation. It also shows the task-specific minimum reduced order r_{min} for each task obtained from the TMSR algorithm shown in Table I.

V. HUMANOID ROBOT CONTROL

This section presents the application of using task-specific reduced order models for control of humanoid robots.

A. State Estimation for Reduced Order Models

For all results presented in this paper, an LQR controller was used to stabilize the reduced order model. Since LQR requires full state feedback, a state observer with gain matrix $L_r \in \mathbb{R}^{r\times r}$ was designed using pole-placement technique for the reduced order model. The state space equations for the reduced order model with the observer and its control law are shown in Eq. 10:

$$\begin{aligned} \dot{\hat{x}}_r &= A_r\hat{x}_r + B_r u + L_r(x_r^{meas} - \hat{x}_r), \\ u &= -K_r\hat{x}_r, \end{aligned} \tag{10}$$

where $\hat{x}_r \in \mathbb{R}^r$ is the reduced state estimate, $x_r^{meas} = T_r U_m(x_d - x^{meas}) \in \mathbb{R}^r$ is the reduced state measurement, $x^{meas} \in \mathbb{R}^{68}$ is the full state measurement of the robot, $x_d \in \mathbb{R}^{68}$ is the desired state vector of the robot, $K_r \in \mathbb{R}^{28\times r}$ is the LQR control gain matrix, $T_r \in \mathbb{R}^{r\times 44}$ is the reduced state transformation matrix that transforms the minimal states into the reduced states, and $U_m \in \mathbb{R}^{44\times 68}$ is the minimal state transformation matrix that transforms the original states into the minimal states of the constrained system.

B. Direct Control using Reduced Order Models

Since the reduced order models derived using the MSR algorithm in Algorithm 3 have the same control inputs as the original model as shown in Eq. 2, one can directly use the LQR controller of the reduced order model to control the full model. For the humanoid robot model presented in Sec. IV-A, the control input $\tau \in \mathbb{R}^{28}$ provides joint torques to the 28 actuated joints on the robot. Figure 4 presents the results of successfully balancing a nonlinear simulation of the humanoid robot using the control law in Eq. 10 for the reduced model R4 in Table I with $r = 28$, which was derived for a whole body motion task with all 68 robot states as outputs. The robot was pushed with a forward force of 138 N for 0.1 s, and as shown in Fig. 4, the robot successfully recovered from it. Figure 5 shows the trajectories of the five most relevant reduced states for this motion, with the first reduced state x_{r1} being the most significant one completely capturing the forward motion of the robot due to the push.

However, direct control of the nonlinear simulation of the humanoid robot using the control law in Eq. 10 was not successful while using the other reduced models in Table I. This is due to a number of reasons. The reduced models R1−R3 in Table I were obtained by using only a subset of the robot states as outputs, and hence, the

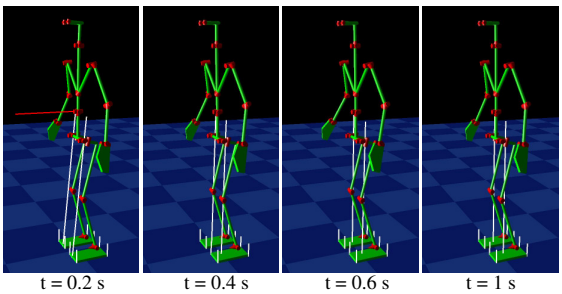


Fig. 4. Direct control of the robot using the reduced order model R4 with $r = 28$ when the robot was pushed with a force of 138 N for 0.1 s.

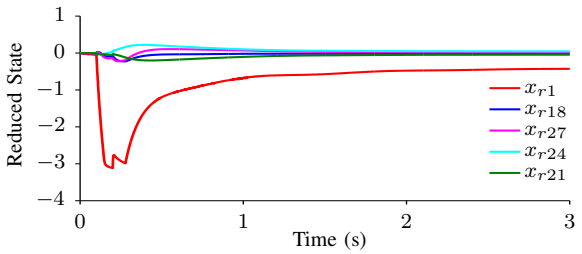


Fig. 5. Trajectories of the five most relevant reduced states for the direct control of the robot using the reduced order model R4 with $r = 28$ when the robot was pushed with a force of 138 N for 0.1 s.

controllers designed to stabilize these models outweigh some robot states over another, which results in poor tracking of reference trajectories for the individual joints. Moreover, they often produce excessive or insufficient contact forces in the feet, and also generate non-zero contact acceleration of the feet, which violates the constraints on the model presented in Sec. IV-A. In order to overcome these issues, an optimization framework is used to find the joint torques τ and contact forces f_c that satisfy the constraints and also achieve the desired task.

C. Control Optimization

This section presents an optimization algorithm that finds the joint torques τ and contact forces f_c shown in Eq. 4, which minimize the following cost function:

$$Z = Z_u + Z_q + Z_c + Z_\tau + Z_c, \tag{11}$$

whose terms are described below.

The cost on the difference between the output of the LQR controller u in Eq. 10 and the joint torques τ is given by:

$$Z_u = \frac{1}{2}(u-\tau)^T W_u (u-\tau), \tag{12}$$

where W_u is a constant weight matrix.

The term Z_q is the cost associated with achieving desired joint accelerations in order to track the reference joint trajectories. The desired joint accelerations are given by:

$$\ddot{q}_d = \ddot{q}_{ref} + K_d(\dot{q}_{ref} - \dot{q}) + K_p(q_{ref} - q), \tag{13}$$

where $K_p, K_d > 0$, and the cost Z_q is given by:

$$Z_q = \frac{1}{2}(\ddot{q}_d - \ddot{q})^T W_q (\ddot{q}_d - \ddot{q}), \tag{14}$$

where W_q is a constant weight matrix. From Eq. 4, Eq. 14 can be re-written as:

$$Z_q = \frac{1}{2}\begin{bmatrix}\tau \\ f_c\end{bmatrix}^T \mathcal{A}_q \begin{bmatrix}\tau \\ f_c\end{bmatrix} - \begin{bmatrix}\tau \\ f_c\end{bmatrix}^T b_q + c_q, \tag{15}$$

where, $\mathcal{A}_q = a_q^T W_q a_q$, $b_q = a_q^T W_q (b_q + \ddot{q}_d)$, $c_q = \frac{1}{2}(b_q - \ddot{q}_d)^T W_q (b_q - \ddot{q}_d)$, $a_q = M^{-1}\left[F^T, J_c^T\right]$ and $b_q = M^{-1}(C\dot{q} + Gq)$.

The cost associated with achieving zero contact acceleration $\ddot{r}_c$ of the feet is given by:

$$Z_c = \frac{1}{2}\ddot{r}_c^T W_c \ddot{r}_c, \tag{16}$$

where $\ddot{r}_c = J_c\ddot{q} + \dot{J}_c\dot{q}$ and W_c is a constant weight matrix. Similar to Eq. 15, Eq. 16 can be re-written as:

$$Z_c = \frac{1}{2}\begin{bmatrix}\tau \\ f_c\end{bmatrix}^T \mathcal{A}_c \begin{bmatrix}\tau \\ f_c\end{bmatrix} - \begin{bmatrix}\tau \\ f_c\end{bmatrix}^T b_c + c_c, \tag{17}$$

where, $\mathcal{A}_c = a_q^T J_c^T W_c J_c a_q$, $b_c = a_q^T J_c^T W_c (J_c b_q - \dot{J}_c\dot{q})$, $c_q = \frac{1}{2}(J_c b_q + \dot{J}_c\dot{q})^T W_c (J_c b_q + \dot{J}_c\dot{q})$, $a_q = M^{-1}\left[F^T, J_c^T\right]$ and $b_q = M^{-1}(C\dot{q} + Gq)$.

The costs associated with minimizing joint torques and contact forces are given by:

$$Z_\tau = \frac{1}{2}\tau^T W_\tau \tau, \tag{18}$$

$$Z_c = \frac{1}{2} f_c^T W_f f_c, \tag{19}$$

where W_τ, W_f are the constant weight matrices.

Using Eqs. 12−19, the cost function Z in Eq. 11 can be re-written in a quadratic form as follows:

$$Z = \frac{1}{2}\begin{bmatrix}\tau \\ f_c\end{bmatrix}^T \mathcal{A} \begin{bmatrix}\tau \\ f_c\end{bmatrix} - \begin{bmatrix}\tau \\ f_c\end{bmatrix}^T b + c, \tag{20}$$

where,

$$\mathcal{A} = \mathcal{A}_q + \mathcal{A}_c + \begin{bmatrix} W_u + W_\tau & 0 \\ 0 & W_f \end{bmatrix} \tag{21}$$

$$b = b_q + b_c + \begin{bmatrix} W_u u \\ 0 \end{bmatrix} \tag{22}$$

$$c = c_q + c_c + u^T W_u u \tag{23}$$

This optimization problem has a simple analytical solution given by:

$$\begin{bmatrix}\tau \\ f_c\end{bmatrix}^* = \mathcal{A}^{-1} b. \tag{24}$$

The optimization presented above does not directly enforce the constraints on the contact forces or other hardware limitations in joint velocities, accelerations and torques. These constraints can be added as inequality constraints to the above optimization. However, for a constrained optimization problem, the solution is no longer a simple matrix computation as shown in Eq. 24, and it is computationally expensive. For all results presented in this paper, the constraints were not added to the optimization, but their corresponding weight matrices in the cost function were adjusted to indirectly enforce the constraints. For example, in order to satisfy the contact force constraints, large values were chosen for the elements of the weight matrix W_f. At any iteration of the optimization algorithm, the weight matrices can also be automatically modified to eliminate violation of the constraints.

D. Simulation Results

This section presents the simulation results of using the reduced order models in Table I with the optimization framework presented in Sec. V-C. For all the results presented in this paper, the weights that form the cost function in Eq. 11 were kept the same, and the only difference between experiments was in the reduced order model used.

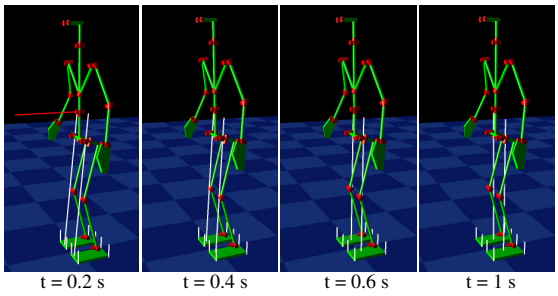


Fig. 6. Balancing control of the robot using the optimization framework and the reduced model R1 with $r = 14$ when the robot was pushed with a force of 143 N for 0.1 s.

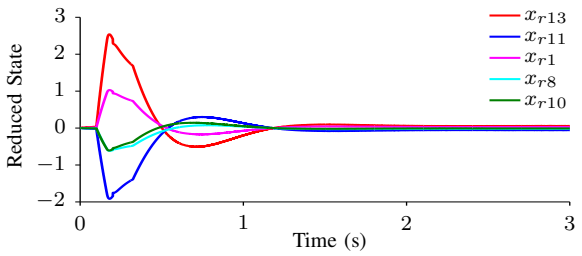


Fig. 7. Trajectories of the five most relevant reduced states using the optimization framework and the reduced order model R1 with $r = 14$ when the robot was pushed with a force of 143 N for 0.1 s.

1) Balancing: Using the optimization framework presented in Sec. V-C, all the reduced order models R1−R4 listed in Table I were able to successfully balance the humanoid robot in simulation while subjected to disturbances. Figure 6 shows snapshots of the robot successfully recovering from a forward push of 143 N for 0.1 s using the reduced model R1 in Table I with $r = 14$, which was derived for the task of balancing with the center of mass (CoM) position and velocity as its outputs. The resulting trajectories of the five most relevant reduced states are shown in Fig. 7. Similar results were obtained for the reduced models R2−R4.

2) Fast Arm Swing Motion: The task of achieving a fast swinging arm motion is considered here. The reference motion moves the arms from 0 rad to 1.25 rad (71.6°) in 0.5 s and back to 0 rad in 0.5 s. Figure 8 shows the snapshots of using the reduced model R1 in Table I with $r = 14$ to achieve this task. As it can be seen from Fig. 8, at time $t = 0.8$ s, the robot loses heel contact with the floor, thereby violating the constraints of the model. At time $t = 1$ s, the robot's heel lands on the floor generating a large contact force that exceeds its limit. This shows that the model R1 with $r = 14$ derived for just the balancing task fails to achieve the fast arm motion. However, it is to be noted that the optimization framework allows the reduced model R1 with $r = 14$ to successfully achieve the same arm motions at slower speeds, for example in 1.5 s rather than 1 s.

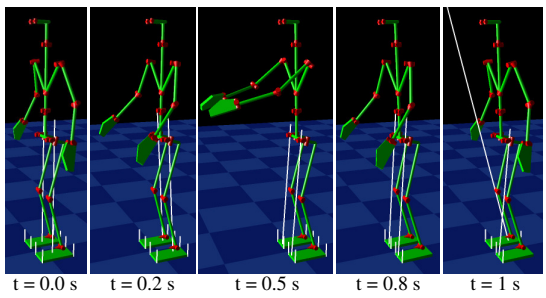


Fig. 8. Failed tracking of the fast reference arm motion using the optimization framework and the reduced order model R1 with $r = 14$.

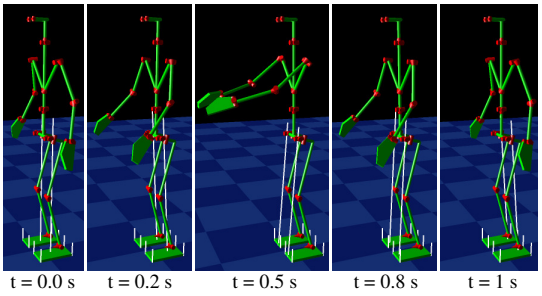


Fig. 9. Successful tracking of the fast reference arm motion using the optimization framework and the reduced order model R3 with $r = 22$.

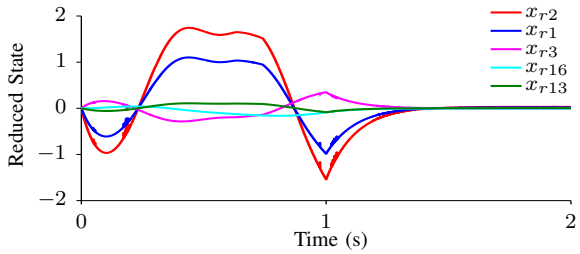


Fig. 10. Trajectories of the five most relevant reduced states while tracking the fast reference arm motion using the optimization framework and the reduced order model R3 with $r = 22$.

Figure 9 shows the snapshots of using the reduced model R3 in Table I with $r = 22$ to achieve the same task. As expected, the model R3 that was generated with the objective of achieving upper body motions successfully achieves the fast arm motion without losing heel contact with the floor and also remains stable. Figure 10 shows the trajectories of the five most relevant reduced states for this motion.

3) Hip Rock and Roll Motion: Figure 11 shows snapshots of the humanoid robot successfully achieving a 4 s rock and roll motion of the hip using the reduced model R2 in Table I with $r = 18$, which was generated for the task of achieving lower body motions. The five most relevant reduced state trajectories for this motion are shown in Fig. 12. Since this is a predominantly lower body motion, the model R2 derived for such a task is successful, whereas the reduced models R1 and R3 fail to achieve the motion as shown in Fig. 13. In both the cases, the robot loses feet contact and generates large

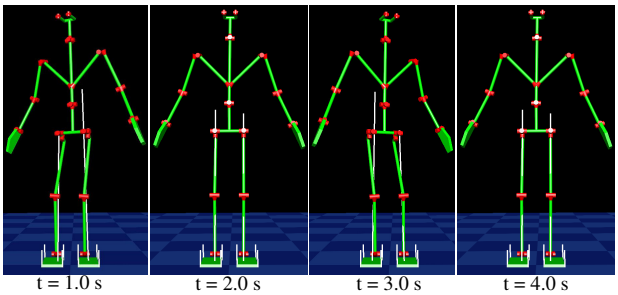


Fig. 11. Successful tracking of the desired hip rock and roll motion using the optimization framework and the reduced order model R2 with $r = 18$.

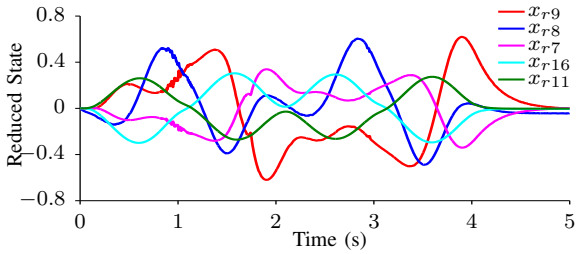


Fig. 12. Trajectories of the five most relevant reduced states while tracking the hip rock and roll motion using the optimization framework and the reduced order model R2 with $r = 18$.

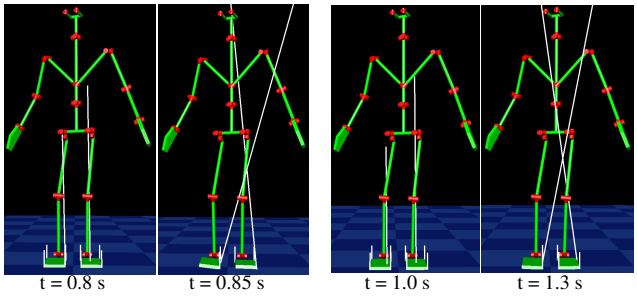
contact forces, which eventually drive the system unstable. It is important to note that even though the reduced model R3 derived for the task of achieving upper body motions has a higher order ($r = 22$) than the reduced model R2 ($r = 18$), it still fails to achieve the hip rock and roll motion. This experiment emphasizes the effectiveness of the task-specific model reduction algorithm presented in this paper.

The videos of all the simulation results presented in this paper are available in the companion video titled "Automatic task-specific model reduction for humanoid robots".

VI. Conclusions and Discussions

This paper presented the minimum stable model reduction (MSR) algorithm, an automatic model reduction algorithm that finds the minimum linear reduced order model whose stabilizing controller stabilizes the full linear system. It also presented its task-specific variant (TMSR), where task-specific output matrices changed the minimum stable reduced order models that were obtained. This paper presented task-specific model reduction results for a 34 DOF humanoid robot model with feet constraints and demonstrated that the order of the task-specific reduced system increased with increase in complexity of the task. Moreover, an optimization framework was presented that empowers the reduced models to be used for control of humanoid robots. Simulation results of a nonlinear robot model successfully achieving balancing, fast arm swing, and hip rock and roll motion tasks using their corresponding task-specific reduced models were also demonstrated. The fast arm swing, and hip rock and roll motion tasks also demonstrated that the reduced models derived for other tasks failed in achieving these tasks even though some were of higher order.

However, the approach presented in this paper has some drawbacks as well. The reduced order models derived using the MSR algorithm do not have any physical meaning and do not represent simple mechanical systems like inverted pendulum models. Moreover, since the model reduction happens in state space, the expressions for energy or momentum cannot be derived for the reduced models, and hence, no physical comparison can be made with the full model. However, one can understand the overall motion represented by a reduced state by visualizing the singular vectors for that reduced state.



(*a*) Using model R1 with $r = 14$ (*b*) Using model R3 with $r = 22$

Fig. 13. Failed tracking of the desired hip rock and roll motion.

VII. Future Works

The simulation results presented in this paper dealt only with a humanoid robot in double support with constraints on both feet. The MSR algorithm can also be used to generate multiple models for tasks like walking, where the model changes between double support and single support. The number of reduced models required to successfully achieve stable walking needs to be explored. In this paper, the task-specific output matrices were provided by the user. Automatic generation of these task-specific output matrices from a different task formulation like the ones in [15] can also be explored.

References

[1] S. Kajita and K. Tani, "Experimental study of biped dynamic walking in the linear inverted pendulum mode," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1995, pp. 2885–2891.

[2] R. Blickhan, "The spring-mass model for running and hopping," *Journal of Biomechanics*, vol. 22, no. 11, pp. 1217–1227, 1989.

[3] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2003, pp. 1620–1626.

[4] S. Lee and A. Goswami, "Reaction mass pendulum (rmp): An explicit model for centroidal angular momentum of humanoid robots," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2007, pp. 4667–4672.

[5] B. Stephens, "Integral control of humanoid balance," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2007, pp. 4020–4027.

[6] B. Stephens and C. Atkeson, "Modeling and control of periodic humanoid balance using the linear biped model," in *Proc. IEEE-RAS Int'l Conf. on Humanoid Robots*, 2009, pp. 379–384.

[7] F. Horak and L. Nashner, "Central programming of postural movements: adaptation to altered support-surface configurations," *Journal of Neurophysiology*, vol. 55, no. 6, pp. 1369–1381, 1986.

[8] A. Goswami, "Kinematic and dynamic analogies between planar biped robots and the reaction mass pendulum (rmp) model," in *Proc. IEEE-RAS Int'l Conf. on Humanoid Robots*, 2008, pp. 182–188.

[9] K. Yamane, "Systematic derivation of simplified dynamics for humanoid robots," in *Proc. IEEE-RAS Int'l Conf. on Humanoid Robots*, November 2012.

[10] M. G. Safonov and R. Y. Chiang, "A schur method for balanced model reduction," *IEEE Trans. on Automatic Control*, vol. 34, no. 7, pp. 729–733, 1989.

[11] D. G. Meyer, "Fractional balanced reduction: Model reduction via fractional representation," *IEEE Trans. on Automatic Control*, vol. 35, no. 12, pp. 1341–1345, 1990.

[12] K. Glover, "All optimal hankel norm approximation of linear multivariable systems and their L^∞-error bounds," *Int'l J. Control*, vol. 39, no. 6, pp. 1145–1193, 1984.

[13] M. G. Safonov, R. Y. Chiang, and D. J. N. Limebeer, "Optimal hankel model reduction for nonminimal systems," *IEEE Trans. on Automatic Control*, vol. 35, no. 4, pp. 496–502, 1990.

[14] W. L. Brogan, *Modern Control Theory, 3rd ed.* Englewood Cliffs, NJ: Prentice Hall, 1991.

[15] A. Safonova, J. K. Hodgins, and N. S. Pollard, "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces," *ACM Trans. on Graphics*, vol. 23, no. 3, pp. 514–521, 2004.