A Prior Reduced Model of Dynamical Systems

Haoran Xie, Zhiqiang Wang, Kazunori Miyata and Ye Zhao

Abstract A reduced model technique for simulating dynamical systems in computer graphics is proposed. Most procedural models of physics-based simulations consist of control parameters in a high-dimensional domain in which the real-time controllability of simulations is an ongoing issue. Therefore, we adopt a separated representation of the model solutions that can be preprocessed offline without relying on the knowledge of the complete solutions. To achieve the functional products in this representation, we utilize an iterative method involving enrichment and projection steps in a tensor formulation. The proposed approaches are successfully applied to different parametric and coupled models.

Keywords Model reduction, dynamical system, separated representation, fixed-point method, tensor product, enrichment step, projection step.

Haoran Xie (Corresponding author)
School of Knowledge Science, Japan Advanced Institute of Science and Technology/JSPS Research Fellow, 1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
e-mail: xiehr@acm.org

Zhiqiang Wang and Ye Zhao
Department of Computer Science, Kent State University,
233 MSB Summit Street, PO Box 5190, Kent, Ohio, 44242, USA
e-mail: zwang22@kent.edu

Ye Zhao
e-mail: zhao@cs.kent.edu

Kazunori Miyata
School of Knowledge Science, Japan Advanced Institute of Science and Technology,
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
e-mail: miyata@jaist.ac.jp

1 Introduction

The simulation of dynamical systems in computer graphics (CG) can be divided into two main categories, i.e., physics-based and data-driven methods. Physics-based methods follow physical principles and have shown remarkable progress recently. The main disadvantages of these methods include high computational cost, low simulation controllability due to numerous control parameters, and an inaccessibility to people who lack expert knowledge in the field. Alternatively, data-driven methods are more efficient and adaptable to complex dynamical systems with which the prerecorded data are largely consumed. One of the limitations of these methods is that the simulation results depend greatly on the prior database or the training data.

A reduced model is a spectacular strategy in data-driven methods, which has been applied successfully to the simulations of deformable bodies [10] and fluids [1]. Most reduced models are based on proper orthogonal decomposition, also known as principal component analysis, which is a posterior method built on a precomputed data field to determine coherent features and reduced basis. Our goal is to introduce a prior reduced model that does not rely on the preprocessed solutions of the problems. The prior reduced model [4] is a developing technique based on separated representations. It has recently been used in different engineering research, including fluids [7, 8] and soft tissue [14]. In contrast to the previous study [3-9], this study describes separated representation in discrete and tensor formulations with high-dimensional dynamical systems and proposes a decoupling approach for coupled problems. In other words, the contribution of this study is the first reported attempt to construct a practical framework for separated representations that can be used in CG applications to achieve realistic simulations at low computational cost.

2 Reduced model

2.1 Problem description

Given a dynamical system $D(U) = G(U,P)$ with unknown state field $U(x_1,x_2,\ldots,x_d)$ where $U = U(t) \in \mathbb{R}^N, t \in [0,T]$, N denotes the degree of freedom (DOF) of the system, G is a source term related to the state U and parameter set $P(p_1,p_2,\ldots)$ and D is represented as a differential operator from the time or parameter dependent ordinary or partial differential equations. The solution of the dynamical systems can be approximated in a high-dimensional domain $(x_1,x_2,\ldots,x_d) \in \Omega_1 \times \Omega_2 \times \cdots \times \Omega_d$ as follows:

$$U(x_1,x_2,...,x_d) = \sum_{i=1}^{N} \alpha_i \prod_{j=1}^{d} U_i^j(x_j). \tag{1}$$

This is also known as a separated representation of the solution [2, 3]. The representation is a sum of N functional products of prior unknown functions $U_i^j(x_j)$

($i = 1,2,\ldots,N$ and $j = 1,2,\ldots,d$ in the following sections) and the normalization coefficients α_i, which are constructed by enrichment steps in an iterative manner. As soon as this representation becomes available, we can obtain the approximated solution with different domains, i.e., temporal and spatial domains, physical parameters, and initial and boundary conditions as extra coordinates. If we assume that we discretize each domain in M nodes, then the representation involves $N \times d \times M$ rather than M^d DOFs in the original problem. For example, if $d = 6, M = 300$, and $N = 15$ (usually, $N \ll M$), the separated representation reduces the DOFs of the dynamical model at a magnitude of 10^{10}. In this sense, the separated representation is a model reduction technique, also known as proper generalized decomposition. In contrast to other reduced models, such as proper orthogonal decomposition, it is a priori model that does not depend on fully precomputed snapshots of the solution. In a two-dimensional problem, the separated representation is similar to singular value decomposition; however, this approach is efficient in high-dimensional dynamical system problems.

2.2 Reduction solver

To determine the functions $U_i^j(x_j)$ and coefficient α_i in the representation Equation (1), we assume that the first $n-1$ separated representation has been obtained at step n. It is straightforward to utilize an iterative process to calculate each $U_n^j(x_j)$. First, we begin from $\alpha_n = 1$, which is then recalculated from a projection process. The solution of the representation at step n is defined as follows:

$$U = \sum_{i=1}^{n-1} \alpha_i \prod_{j=1}^{d} U_i^j(x_j) + \prod_{j=1}^{d} U_n^j(x_j), \tag{2}$$

where $U_n = \{U_n^1(x_1), U_n^2(x_2), \ldots, U_n^d(x_d)\}$ are the test functions that need to be solved next. Then, each term of U_n is projected on the weak form of the dynamical model $D(U) = G$.

$$\langle D(U), U_n^j \rangle_{\Omega_j} = \langle G, U_n^j \rangle_{\Omega_j}, \tag{3}$$

where $\langle , \rangle_{\Omega_j}$ represents the scalar product in L^2 norm on the domain Ω_j. Note that the following residual term r^n is omitted in the weak form which can be used to check the process convergence.

$$r^n = D\Big(\sum_{i=1}^{n-1} \alpha_i \prod_{j=1}^{d} U_i^j(x_j) + \prod_{j=1}^{d} U_n^j(x_j)\Big) - G. \tag{4}$$

To solve each $U_n, 1 \le n \le N$, a simple choice to obtain the enrich term U_n is an iterative method as an alternating directions fixed-point algorithm to solve Equation (3) simultaneously. The idea at p-th iteration for U_n is described as follows. First, u_p^1 is computed with the previously obtained values $(u_{p-1}^2, u_{p-1}^3, \ldots, u_{p-1}^d)$

(Small letter u is distinguished from capital letter U for U_n in a fixed-point iterative process.). Then, for the term $u_p^k, k \in (1,d]$, the updated values and previous values $(u_p^1, \ldots, u_p^{k-1}, u_{p-1}^{k+1}, \ldots, u_{p-1}^d)$ are utilized. After reaching convergence, the U_n values are updated from u.

With the obtained n functional products $\prod_{j=1}^d U_i^j(x_j), 1 \leq i \leq n$, the coefficients α_i is computed by projection of $D(U)$ to each functional product.

$$\langle D(U), \prod_{j=1}^{d} U_i^j(x_j) \rangle = \langle G, \prod_{j=1}^{d} U_i^j(x_j) \rangle. \tag{5}$$

Finally, if the residual term $\|r^n\| < \varepsilon$, ε is a designated threshold value, the entire process is in convergence; otherwise, the computation process returns to the enrichment step in the fixed-point algorithm.

2.3 Discrete formulation

Here, we clarify the scalar products in L^2 using a discrete formulation. For simplicity, a linear differential operator is considered as follows:

$$\frac{du}{dt} + ku = 0. \tag{6}$$

Here, $u(t,k) = \sum_{i=1}^N T_i(t)K_i(k)$ on the domain $\Omega_t \times \Omega_k$ as Equation (1), where the normalization coefficients α_i are omitted for simplicity. From Equation (3), the formulation of the weak form at step n is substituted as follows:

$$\begin{aligned} &\langle \frac{dT_n}{dt}, T_n \rangle \langle K_n, K_n \rangle + \langle T_n, T_n \rangle \langle kK_n, K_n \rangle \\ &\qquad = -\sum_{i=1}^{n-1} \Big(\langle \frac{dT_i}{dt}, T_n \rangle \langle K_i, K_n \rangle + \langle T_i, T_n \rangle \langle kK_i, K_n \rangle \Big). \end{aligned} \tag{7}$$

By adopting finite element discretization techniques in each domain mesh, the equation has the following matrix form (for simplicity, the subscript n is omitted):

$$\begin{aligned} &\mathbf{T}^T \mathbf{P} \mathbf{T} \cdot \mathbf{K}^T \mathbf{M}^k \mathbf{K} + \mathbf{T}^T \mathbf{M}^t \mathbf{T} \cdot \mathbf{K}^T \mathbf{N} \mathbf{K} \\ &\qquad = -\sum_{i=1}^{n-1} (\mathbf{T}_i^T \mathbf{P} \mathbf{T} \cdot \mathbf{K}_i^T \mathbf{M}^k \mathbf{K} + \mathbf{T}_i \mathbf{M}^t \mathbf{T} \cdot K_i^T \mathbf{N} \mathbf{K}), \end{aligned} \tag{8}$$

where $\mathbf{T}$, $\mathbf{K}$, $\mathbf{T}_i$, and $\mathbf{K}_i$ represent the vectors containing the nodal values of the functions T, K, T_i, and K_i. The definitions of $\mathbf{P}$, $\mathbf{N}$, $\mathbf{M}^t$, and $\mathbf{M}^k$ are as follows:

$$\begin{aligned}
\mathbf{P}_{ij} &= \int_{\Omega_t} \frac{dN_i}{dt} N_j dt, \\
\mathbf{N}_{ij} &= \int_{\Omega_k} N_i k N_j dk, \\
\mathbf{M}^t_{ij} &= \int_{\Omega_t} N_i N_j dt, \\
\mathbf{M}^k_{ij} &= \int_{\Omega_k} N_i N_j dk,
\end{aligned} \tag{9}$$

where N_i and N_j are shape functions associated with meshes on Ω_t and Ω_k. Note that the discrete formulation is commonly available for other differential operators, such as gradient and Laplacian.

2.4 Tensor formulation

From the discrete form of a dynamical model as Equation (8), the separated representation can be described in algebraic form with tensor products. For $D(U) = G$:

$$D = \sum_{i=1}^{N_D} D_i^1 \otimes D_i^2 \otimes \cdots \otimes D_i^d, \quad G = \sum_{i=1}^{N_G} G_i^1 \otimes G_i^2 \otimes \cdots \otimes G_i^d, \quad U = \sum_{i=1}^{N} \alpha_i U_i^1 \otimes U_i^2 \otimes \cdots \otimes U_i^d, \tag{10}$$

where $D_i^j, j = 1,2,\ldots,d$, is a $w_j \times w_j$ matrix, and w_j is the number of nodes in domain mesh Ω_j. The sizes of G_i^j and U_i^j are w_j, and they can be obtained directly from the discrete formulation of the dynamical system. We describe the implementation details of the algorithm (Section 2.2) by utilizing a tensor formulation in the next section.

3 Implementation

In this section, the proposed algorithm is discussed comprehensively. The pseudo code is shown in Algorithm 1.

3.1 Enrichment step

As mentioned in Section 2.2, the fixed-point algorithm is adopted to search for the enrich term of test function $U_n = \alpha_n R^1 \otimes R^2 \otimes \cdots \otimes R^d$ using iterative processes. In the dynamical model, we have the following formulation of U_n from Equation (4).

Algorithm 1 Pseudo-code for the prior reduced model of the separated representation.

```
Initialize D, G, and U from Equations (8)(10)
for q = 1 to q_max do // number of coupled equations
    for n = 1 to n_max do // number of enrichments
        for p = 1 to p_max do // fixed-point iteration
            Compute R_p^j from Equation (12) of the q-th equation in
            coupled equations // enrichment step
            Check convergence from Equation (13)
        end for
        Normalize U_n^j from Equation (14)
        Compute coefficients α_i from Equation (15) // projection step
        Update U_n
        Check convergence from Equation (16)
    end for
    Update G
end for
```

$$\sum_{i=1}^{N_D} D_i^1 R^1 \otimes D_i^2 R^2 \otimes \cdots \otimes D_i^d R^d = G - \sum_{i=1}^{N_D} \sum_{k=1}^{n-1} \alpha_k D_i^1 U_k^1 \otimes D_i^2 U_k^2 \otimes \cdots \otimes D_i^d U_k^d. \quad (11)$$

Note that we assume $\alpha_n = 1$. At the p-th step of the fixed-point iteration, $R_p^j, 1 \leq j \leq d$ is obtained from $(R_p^1, \ldots, R_p^{j-1}, R_{p-1}^{j+1}, \ldots, R_{p-1}^d)$. Thus, we obtain the following:

$$\mathbf{E}^j R_p^j = \sum_{i=1}^{N_G} \Big(\prod_{m=1, m\neq j}^{d} (R_p^m)^T G_i^m \Big) G_i^j - \sum_{i=1}^{N_D} \sum_{k=1}^{n-1} \Big(\prod_{m=1, m\neq j}^{d} (R_p^m)^T D_i^m U_k^m \Big) \alpha_k D_i^j U_k^j. \quad (12)$$

Here, the matrix $\mathbf{E}^j = \sum_{i=1}^{N_D} \Big(\prod_{m=1, m\neq j}^{d} (R_p^m)^T D_i^m R_p^m \Big) D_i^j$. After all $(R_p^1, R_p^2, \ldots, R_p^d)$ are obtained at the p-th step. The convergence condition is defined as follows:

$$\| R_p^1 \otimes R_p^2 \otimes \cdots \otimes R_p^d - R_{p-1}^1 \otimes R_{p-1}^2 \otimes \cdots \otimes R_{p-1}^d \| < \varepsilon. \quad (13)$$

Here, ε is set by the user and $\|\cdot\|$ represents L^2-norm. Finally, U_n^j in the formulation of U (Equation (10)) is obtained by the normalization of each R^j.

$$U_n^j = \frac{R^j}{\|R^j\|}, \qquad j = 1, 2, \ldots, d. \quad (14)$$

3.2 Projection step

From the Equation (5), the formulation is modified as follows by using the value of G.

$$\mathbf{BA} = \mathbf{H}, \quad \mathbf{B}_{ij} = \sum_{k=1}^{N_D} \Big(\prod_{e=1}^{d} (U_i^e)^T D_k^e U_j^e \Big), \quad \mathbf{H}_i = \sum_{m=1}^{N_G} \Big(\prod_{e=1}^{d} (U_i^e)^T G_m^e \Big). \tag{15}$$

Here, $\mathbf{A} = [\alpha_1 \alpha_2 \cdots \alpha_n]^T$ and $1 \leq i, j \leq n$. Finally, U_n is updated and the residual term r^n is given as follows:

$$r^n = \sum_{i=1}^{N_D} \sum_{k=1}^{n} \alpha_k D_i^1 U_k^1 \otimes D_i^2 U_k^2 \otimes \cdots \otimes D_i^d U_k^d - G. \tag{16}$$

3.3 Coupled terms

For simultaneous differential equations, their solutions benefit from utilizing a decoupling strategy to reduce computational complexity. We assume that there are q_{max} coupled equations, the reduced solver would be executed iteratively (line 2 in Algorithm 1). If another variable $W = \sum_{i=1}^{N} \beta_i W_i^1 \otimes W_i^2 \otimes \cdots \otimes W_i^d + S^1 \otimes S^2 \otimes \cdots \otimes S^d$ exists, then the coupled equations can be decoupled as follows:

$$\begin{cases} D_U(U, W, R, S = 0) = G_U \\ D_W(U, W, R = 0, S) = G_W \end{cases}, \tag{17}$$

where D_U and D_W are different operators for U and W, respectively. Therefore, there is $(n-1) \times (n-1)$ terms when a multiple term $U \cdot W$ is computed using their previously known values $U_{n-1} \cdot W_{n-1}$. The value of the coupled term is known; therefore, we place all the terms in G as source terms, as shown in Line 14 of Algorithm 1.

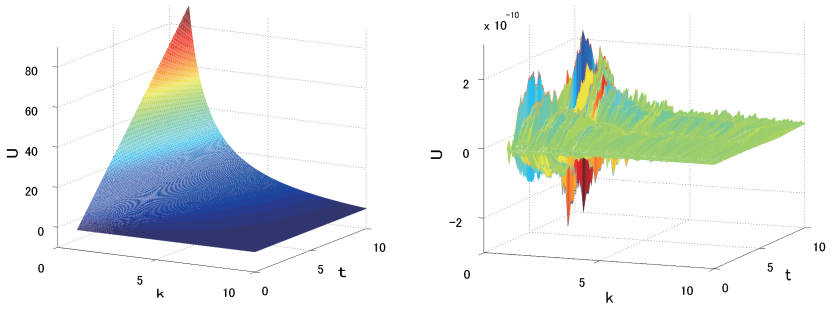
4 Numerical results

Here, we provide numerical examples to verify the efficiency and the accuracy of the proposed prior reduced model of dynamical systems. The examples include a parametric model and two coupled models with unknown initial values. All examples were implemented on a standard PC (Intel Core i7 CPU 2.10 GHz and 8.0 GB RAM), and their reference solutions were obtained from MATLAB ODE solvers.

4.1 Parametric model

In a separated representation, the control parameters can be introduced into the representation as extra coordinates. The following differential equation is considered as an example:

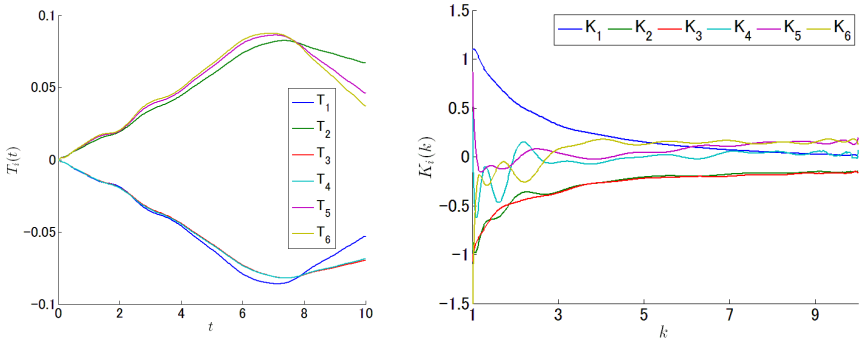
$$k(\dot{u} + 1) = 10, \tag{18}$$



(a) (b)

Fig. 1 (a) Numerical result of the separated representation. (b) Computation error compared with the reference solution.

where $\dot{u}$ represents first time derivative of variable u with initial condition $u(t=0)=0$. The separated representation of this parametric model is given in the following formulation:



(a) (b)

Fig. 2 First six functions of $T_i(t)$ and $K_i(k)$. (Note that the values of functions have been curve fitted by the polynomial curves)

$$u(t,k)=\sum_{i=1}^{n}\alpha_i T_i(t)K_i(k), \tag{19}$$

where $t\in[0,10]$ and $k\in[1,10]$. Figure 1 (a) shows the simulation results obtained using the proposed approach, which is sufficiently accurate with an error level of 10^{-11} (Figure 1 (b)). The reference solutions in the examples are calculated by an

ODE solver, e.g., the Runge-Kutta method. Figure 2 shows the first six functions in the representation, which are obtained by the enrichment steps in Algorithm 1.

4.2 Coupled model

We evaluate the proposed method for dynamical systems with coupled terms using the following differential equations:

$$\begin{cases} \dot{u}_1 + u_2 u_3 = 1 \\ \dot{u}_2 + u_1 u_3 = 2 \\ \dot{u}_3 + u_1 u_2 = 3 \end{cases} . \tag{20}$$

The initial conditions $u_i(t=0) = u_i^0, i = 1,2,3$, are considered unknown in this example. To solve these equations, we introduce new variables $\hat{u}_i = u_i - u_i^0$. Then, Equation (20) becomes a system of the variables $\hat{u}_i$, where the initial conditions are considered as new model coordinates in the separated representation of $\hat{u}_i$.

$$\hat{u}_i(t, u_i^0) = \sum_{j=1}^{n} \alpha_j T_j(t) U_j^1(u_1^0) U_j^2(u_2^0) U_j^3(u_3^0), \tag{21}$$

where the domains are $\Omega_t(t) \times \Omega_1(u_1^0) \times \Omega_2(u_2^0) \times \Omega_3(u_3^0) = [0,1] \times [0,1] \times [0,1] \times [0,1]$. From Figure 3, we determine that our simulation results have good compliance with the reference solutions in this coupled model. The coupled terms in this example are solved by the proposed approach, and the convergence of the solutions can be achieved after eight iterations, as shown in Figure 4 (a).

As per Figure 4 (b), the computation speed of the proposed reduced model is stable and fast compared to a simple iterative procedure for parameter identification. The computation cost of the reference ODE solver increases exponentially, which is known as the "curse of dimensionality" [9]. For example, the computation cost of the proposed method is only 17 ms, i.e., 50 times faster than the reference approach when the node numbers of $u_i^0, i = 1,2,3$ are set to 20, as shown in Figure 4 (b). The computation cost can be reduced remarkably with high DOFs of the dynamical system.

4.3 Complex model

This example considers six DOFs rigid body dynamics in potential flow [13, 17], where the non-linear viscous forces are omitted. The dynamical equations are as follows:

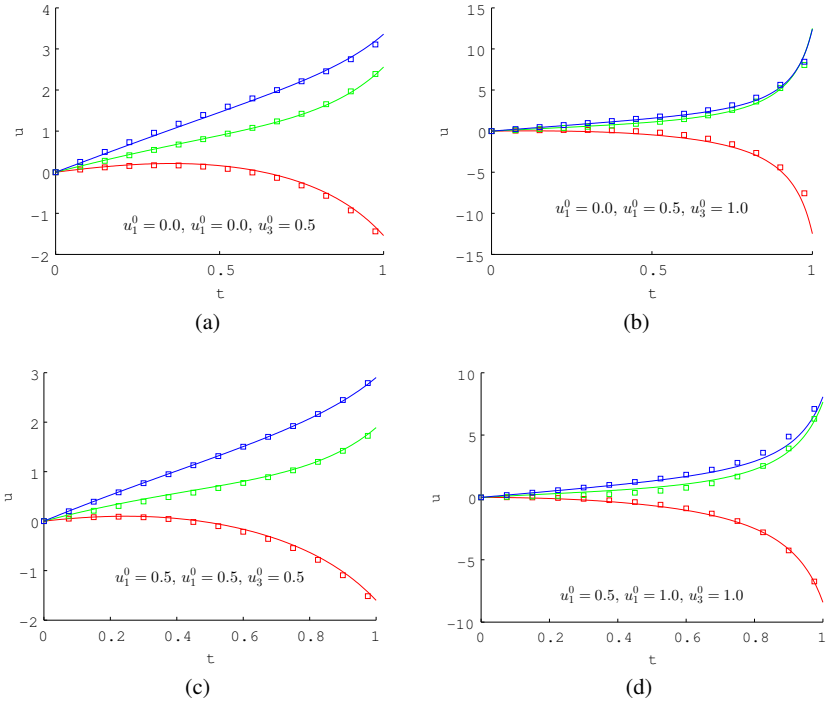


Fig. 3 Comparison with reference results for different values of initial conditions (lines represent reference results; empty squares represent the computation results of our separated representation (red: u_1; blue: u_2; green: u_3))

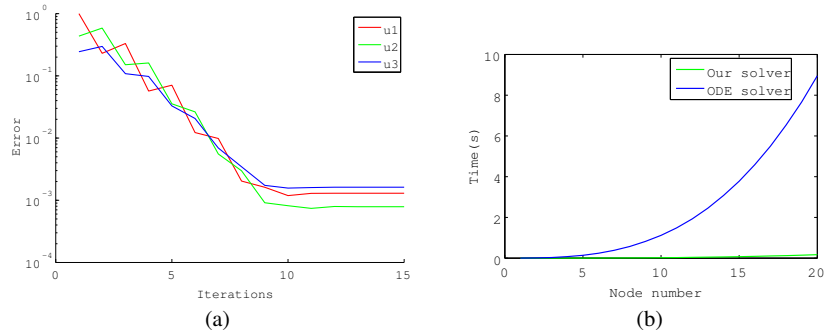


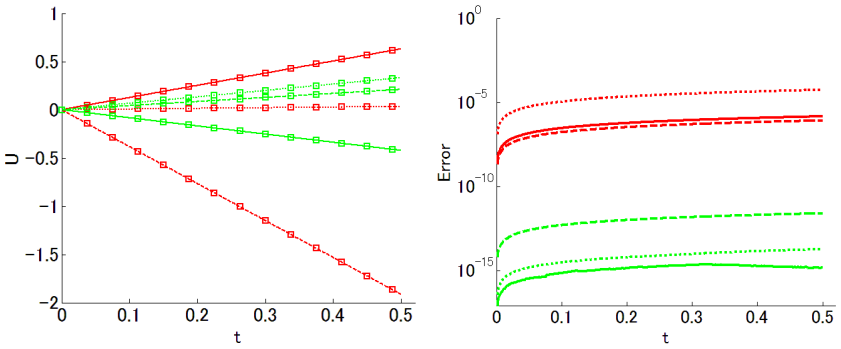
Fig. 4 (a) Convergence of simulation results after iterations in the coupled model with initial conditions corresponding to the case of Figure 3 (d). (b) Computation times compared with a simple iterative procedure.

$$\begin{cases} (mE+M)\dot{u} = (mE+M)u \times \omega + f_g \\ (J+I)\dot{\omega} = (J+I)\omega \times \omega + (Mu) \times u + \tau_g \end{cases}, \tag{22}$$

where E is a unit tensor, I is the moment of inertia of the body, M and J are added mass and added moment of inertia due to the accelerations from the surrounding flow, respectively, and f_g and τ_g are the force and its torque from the buoyancy-corrected gravity in terms of initial velocity state, respectively. In this example, we do not consider the strongly coupled terms due to translational and angular velocities (u, ω). However, it is helpful to evaluate the strongly coupled terms due to initial conditions, where six initial values of $U_0 = (u_0, \omega_0) \in \mathbb{R}^6$ are introduced in the following separated representation as new coordinates.

$$U_k(t, u_k^0) = \sum_{i=1}^{n} \alpha_i T_i(t) \prod_{j=1}^{6} U_i^j(u_j^0), \tag{23}$$

where $1 \leq k \leq 6$. The simulation results of the separated representation match the reference solution, as shown in Figure 5.



(a) (b)

Fig. 5 (a) Simulation result with initial velocities $[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]$ and (b) its computation error (red: translational velocity; green: angular velocity)

5 Conclusion

We have introduced a new prior reduced model based on separated representations that do not require snapshots of complete solutions for dynamical systems. This method can reduce high-dimensional problems and tackle different domains, i.e., temporal and spatial domains, physical parameters, and initial and boundary conditions as extra coordinates. We have proposed a framework for separated repre-

sentation on discrete and tensor formulations and a method to account for coupled terms in the proposed framework. The proposed method utilizes a fixed-point algorithm in an iterative process to control the desired accuracy of the problems under convergence.

The limitation of the proposed method is the difficulty in accounting for nonlinear and coupled terms. For complex models, especially strongly coupled problems, the proposed approach may fail because of the large amount of terms generated in the iterative process, i.e., $O(N^2)$. Other approaches, such as an asymptotic numerical method and discrete empirical interpolation method [5, 6, 16] also suffer the same limitations. A promising solution for this issue is the adoption of nonlinear model reduction techniques, such as piecewise-linear approximation. Furthermore, a promising future work is to decouple the system model and reduce its nonlinearity by combining with POD.

The proposed approach is efficient because the computation of the reduced model is only executed in a precomputed process. The preprocessed data can be saved as a codebook to search solutions for different control parameters and initial and boundary conditions, and the computation cost is only a few milliseconds. The proposed approaches can be applied to motion control, inverse identification, and parameter estimations for various physical simulations in real-time CG applications. The challenge of the physical simulations and their control problem for complex dynamics in CG is related to the physical control parameters, such as the coefficients of restitution and surface normals for rigid bodies [15], stiffness and friction coefficients for deformable bodies [12], drag and lift coefficients for aerodynamics simulations [11], and the Reynolds number for flow simulations, which are usually designated by measured data in constant or curve forms. In the proposed method, all these parameters would be embedded in separated representations as extra coordinates to achieve realistic simulation results at low computational cost. It is difficult to adopt model reduction for instantaneous coupling problems among flow and bodies. However, the parameter estimations of force coefficients with different flow conditions are feasible by the proposed method for one-way coupling issues.

References

1. T. Adrien, L. Andrew, and P. Zoran. Model reduction for real-time fluids. *ACM Trans. Graph.*, 25(3):826–834 (2006)
2. G. Beylkin and M. Mohlenkamp. Algorithms for numerical analysis in high dimensions. *SIAM Journal on Scientific Computing*, 26(6):2133–2159 (2005)
3. F. Chinesta, A. Ammar, and E. Cueto. Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models. *Archives of Computational Methods in Engineering*, 17(4):327–350 (2010)
4. F. Chinesta, R. Keunings, and A. Leygue. The proper generalized decomposition for advanced numerical simulations:a primer. *Springer International Publishing* (2014)
5. F. Chinesta, A. Leygue, M. Beringhier, L. Nguyen, J. Grandidier, B. Schrefler, and F. Pesavento. Towards a framework for non-linear thermal models in shell domains. *International Journal of Numerical Methods for Heat and Fluid Flow*, 23(1):55–73 (2013)

6. F. Chinesta, A. Leygue, F. Bordeu, J.V. Aguado, E. Cueto, D. Gonzalez, I. Alfaro, A. Ammar, and A. Huerta. Pgd-based computational vademecum for efficient design, optimization and control. *Archives of Computational Methods in Engineering*, 20(1):31–59 (2013)
7. A. Dumon, C. Allery, and A. Ammar. Proper general decomposition for the resolution of navier stokes equations. *Journal of Computational Physics*, 230(4):1387–1407 (2011)
8. A. Dumon, C. Allery, and A. Ammar. Proper generalized decomposition method for incompressible navier-stokes equations with a spectral discretization. *Applied Mathematics and Computation*, 219(15):8145–8162 (2013)
9. D. Gonzalez, F. Masson, F. Poulhaon, A. Leygue, E. Cueto, and F. Chinesta. Proper generalized decomposition based dynamic data driven inverse identification. *Mathematics and Computers in Simulation*, 82(9):1677–1695 (2012)
10. D. Jame and F. Kayvon. Precomputing interactive dynamic deformable scenes. *ACM Trans. Graph.*, 22(3):879–887 (2003)
11. E. Ju, J. Won, J. Lee, B. Choi, J. Noh, and M. Choi. Data-driven control of flapping flight. *ACM Trans. Graph.*, 32(5):151:1–151:12 (2013)
12. E. Miguel, R. Tamstorf, D. Bradley, Sara C. Schvartzman, B. Thomaszewski, B. Bickel, W. Matusik, S. Marschner, and Miguel A. Otaduy. Modeling and estimation of internal friction in cloth. *ACM Trans. Graph.*, 32(6):212:1–212:10 (2013)
13. G. Mougin and J. Magnaudet. The generalized kirchhoff equations and their application to the interaction between a rigid body and an arbitrary time-dependent viscous flow. *International Journal of Multiphase Flow*, 28(11):1837–1851 (2002)
14. S. Niroomandi, D. Gonzalez, I. Alfaro, F. Bordeu, A. Leygue, E. Cueto, and F. Chinesta. Real-time simulation of biological soft tissues: a pgd approach. *International Journal for Numerical Methods in Biomedical Engineering*, 29(5):586–600 (2013)
15. J. Popović, Steven M. Seitz, and M. Erdmann. Motion sketching for control of rigid-body simulations. *ACM Trans. Graph.*, 22(4):1034–1054 (2003)
16. E. Pruliere, F. Chinesta, and A. Ammar. On the deterministic solution of multidimensional parametric models using the proper generalized decomposition. *Mathematics and Computers in Simulation*, 81(4):791–810 (2010)
17. H. Xie and K. Miyata. Stochastic modeling of immersed rigid-body dynamics. *SIGGRAPH Asia 2013 Technical Briefs*, pages 12:1–12:4 (2013)