Efficient motion planning of highly articulated chains using physics-based sampling

Russell Gayle, Stephane Redon, Avneesh Sud, Ming C Lin, Dinesh Manocha

Efficient Motion Planning of Highly Articulated Chains using Physics-based Sampling

Russell Gayle[1], Stephane Redon[2], Avneesh Sud[1], Ming C. Lin[1], and Dinesh Manocha[1]
(Video Included)

***Abstract*— We present a novel motion planning algorithm that efficiently generates physics-based samples in a kinematically and dynamically constrained space of a highly articulated chain. Similar to prior kinodynamic planning methods, the sampled nodes in our roadmaps are generated based on dynamic simulation. Moreover, we bias these samples by using constraint forces designed to avoid collisions while moving toward the goal configuration. We adaptively reduce the complexity of the state space by determining a subset of joints that contribute most towards the motion and only simulate these joints. Based on these configurations, we compute a valid path that satisfies non-penetration, kinematic, and dynamics constraints. Our approach can be easily combined with a variety of motion planning algorithms including probabilistic roadmaps (PRMs) and rapidly-exploring random trees (RRTs) and applied to articulated robots with hundreds of joints. We demonstrate the performance of our algorithm on several challenging benchmarks.**

I. Introduction

Highly articulated robots, such as snake or serpentine robots, with many degrees of freedom (DoFs) have received considerable attention recently [1], [2], [3]. Chirikjian and Burdick first introduced the term hyper-redundant robots to describe such robots with a very high number of DoFs [4], [5]. Snake-like robots can serve as suitable alternatives over traditional robotic systems for difficult terrains and challenging scenarios. These include search and rescue missions in complex urban environments, planetary surface exploration, minimally invasive surgery, or inspection of piping and cabling. Highly articulated robots also have many applications in homeland security and national defense, as well as enabling inspection of ships, containers and other structures with narrow, tight workspace. Many computational biology algorithms also model the molecular chains as articulated models with hundreds or thousands of links.

Randomized motion planning research has made great strides toward providing planning solutions in high dimensional configuration spaces. These include probabilistic roadmaps (PRMs) and rapidly-exploring random trees (RRTs), which can solve complex problems with dozens of DoFs [6], [7]. However, they rely upon randomization to map or explore the configuration space, their performance can degrade considerably for robots with much higher dimensionality (e.g. hundreds or thousands) [8]. First, it would be extremely expensive to compute representative samples of the configuration space. Moreover, if the resulting motion needs to maintain other kinematic or dynamics constraints, this requirement further complicates the problem since many planners only consider geometric constraints like non-penetration. For instance, consider the motion generated by a either a PRM or RRT. In most variations of these methods, the links are typically generated as *straight-line* paths in the configuration space using a local planner and only take into account geometric constraints (e.g. collision avoidance). Thus, even if the samples themselves are physically-based and the intermediate configurations are collision-free, they may not satisfy kinematic constraints. On the other hand, if the motion generated was completely physically-based, the path would be more realistic.

Main Results: We present an efficient physics-based motion generation scheme for highly articulated robots, particularly in a serial linkage, or chain-like configuration. We use a dynamics-based motion planning framework to generate samples and bias the search direction. Our framework exploits the coherence between neighboring joints to attempt to reduce the dimensionality of the problem through the use of "adaptive forward dynamics". Based on motion metrics, we prioritize the joints to determine which ones capture the majority of the motion of the chain. Adaptive forward dynamics takes advantage of this formulation by only simulating these joints which effectively produce motion in a reduced dimension and results in improved efficiency. Moreover, we discuss the validity and usability of the resulting solution when compared to earlier planners for the same robots. We demonstrate the application of our planner on several highly articulated robots consisting of 300 to 2500 single degree of freedom joints. Each local sampling using adaptive dynamics for these robots takes on average a few milli-seconds. We have observed up to one order of magnitude performance improvement using our approach as compared to sampling with full dynamics.

Some of the main characteristics of our motion generation method are:

- **Physically-based:** We take into account forward dynamics of articulated joints during motion planning, in addition to the geometric constraints including collision detection, contact handling, kinematic constraints, etc.
- **Efficiency:** We perform *lazy* dynamics update and achieve *sub-linear* running time performance in terms of DoFs when some of the joints do not move much.
- **High DoF robots:** Our algorithm can simulate the forward dynamics and plan the path for a robot with

[1] Department of Computer Science, University of North Carolina at Chapel Hill, {rgayle,sud,lin,dm}@cs.unc.edu
[2] INRIA Rhone-Alpes, France, stephane.redon@inria.fr

very high DoFs in nearly real time, using a progressive refinement framework.

The remainder of the paper is organized as follows. Section II briefly surveys prior work in this area. Section III introduces our notation and gives an overview of our approach. Section IV gives details of our motion planning algorithm. We describe our results and highlight its performance on different benchmarks in Section V.

II. Related Work

There is a rich body of literature on both motion planning for highly articulated (or hyper-redundant robots) and on sampling methods to improve the quality and performance of sample-based planners. We briefly discuss related work here.

A. Articulated Body Motion Planning

In general, much of the existing work in motion planning can be applied to articulated chains. In fact, this is a special case of the more general planning problem for branched or closed loop articulated bodies. For an overview on robot motion planning, we refer the readers to [9], [10], [11].

The simplest algorithms for articulated models are based on randomized motion planning. Most notably, Probabilistic Roadmaps (PRMs) have achieved a great deal of success in solving multiple-query motion planning problems [6]. For single-shot, or single-query, problems, Rapidly-exploring Random Trees (RRTs) have the potential to grow into unexplored regions of space [7]. These have been extended in a number of ways to perform well for objects that are similar to articulated chains, such as pipes, cables, ropes, and flexible wires [12], [13], [14], [15], [16], [17]. Another approach is to determine the principal components using statistical methods to reduce the dimensionality of such a high DoF configuration space [18]. Also, Barraquand et. al used potential fields with random walks for high-dof articulated robots [19]. However, many of these algorithms do not consider the kinematic or dynamics constraints related to these types of structures.

Several algorithms take into account kinematics information to aid in generating samples for PRM-based planners [20], [21], [22]. However, these approaches are targeted towards closed loop linkages and do not take contact responses such as friction into account.

B. Kinodynamic Planning

While more deterministic solutions have been proposed for kinodynamic motion planning, these approaches typically do not scale well to high DOF robots [23]. Many authors have also proposed randomized kinodynamic planners based on PRMs [24] and RRTs [25]. Like their C-space analogues, the performance depends upon the dimensionality of the state space. Alternatively, Ladd and Kavraki pose a tree-based kinodynamic solution where the path segments are the samples rather than states [26].

C. Sampling Strategies for Motion Planning

The problem of sample generation for various planning strategies has received a considerable amount of attention. Several extensions have been proposed for PRMs. For instance, visibility information can be used to generate and store fewer samples [28]; medial-axis based sampling [29], [30], [31] and sampling near the obstacle boundaries [32] can help in dealing with narrow passages. Gaussian weighting can aid to place samples where they would reveal most information about an environment [33].

With respect to RRTs, the standard algorithm already performs extra biasing [7] or try to build more optimal trees [34]. Dynamic domains are used to improve expansions in RRT [36].

III. Overview

In this section, we formalize the notation used throughout the paper and introduce the main ideas behind our work.

A. Dynamics Simulation

Our intended application of this work is an articulated chain, where actuators may exist to control the internal joints. However, the ideas presented in this paper are also applicable to rigid as well as branching articulated structures.

Multi-body systems and forward dynamics are central to simulation of an articulated robot. We make use of a simplified articulated model, much like that proposed by [38], in order to reduce the dimensionality of the configuration space. An impulse-based contact determination and response algorithm for adaptive forward dynamics was described by [39]. Other physical constraints on the joints can be efficiently simulated through *analytical constraints* [40]. These are the necessary components to ensure realistic samples for our algorithm. Due to space limitations, we refer the reader to a recent survey on multi-body simulation [41] for more details.

Our articulated robot is represented using Featherstone's recursive definition of an articulated body; an articulated body is composed of two other articulated bodies that are connected through a *principal joint* (See Fig 1).

At the base level, we have a set of n rigid bodies. For simplicity, we define our joint primitives to be 1-degree of freedom (DoF) revolute joints. More complex 2-DoF revolute joints can be added in a standard way. A rigid body, b_{thin}, with zero (or sufficiently small) width is placed between each pair of bodies, b_i and b_{i+1}. A standard 1-DoF joint will connect b_i and b_{thin}, and another 1-DoF joint rotated 90 degrees about the center of the chain will connect b_{i+1} and b_{thin}.

We then represent the entire articulated body and its *state* at time t as $\mathcal{A}(t) = \{\mathcal{B}, \mathcal{J}, \mathbf{q}(t), \dot{\mathbf{q}}(\mathbf{t})\}$, where $\mathcal{B} = \{b_1, b_2, \ldots, b_{2n-1}\}$ are the rigid bodies, $\mathcal{J} = \{j_1, j_2, \ldots, j_{2n-2}\}$ are the joints, $\mathbf{q}(t)$ is a vector of length $2n-2$ of joint positions, and $\dot{\mathbf{q}}(\mathbf{t})$ is a vector of length $2n-2$ of joint velocities at time t. To build the articulated body, pairs of bodies are connected via principal joints from the bottom up, following the recursive definition. The robot's

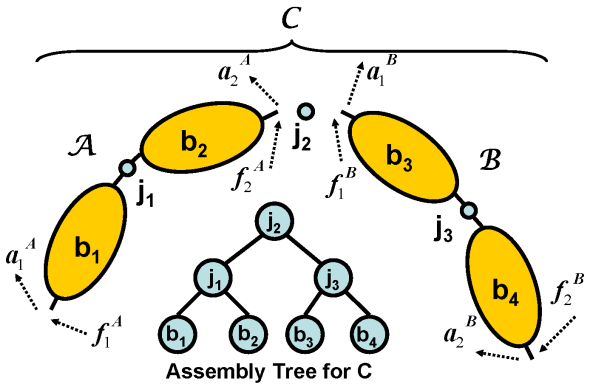

Fig. 1: Construction of an articulated body. An articulated body $\mathcal{A}$ is connected to body $\mathcal{B}$ at the principal joint, j_2, to form body $\mathcal{C}$. The assembly tree for $\mathcal{C}$ is shown beneath the body. Forces and accelerations which govern $\mathcal{C}$'s motion are shown.

principal joint is then the middle joint along the linkage. This process forms an *assembly tree* (See Fig. 1).

Given this construction, Featherstone derives motions of equations that have the same recursive structure, given an external force acting upon the body. Most importantly,

$$\begin{bmatrix} \hat{\mathbf{a}}_1 \\ \hat{\mathbf{a}}_2 \\ \vdots \\ \hat{\mathbf{a}}_m \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi}_1 & \mathbf{\Phi}_{12} & \cdots & \mathbf{\Phi}_{1m} \\ \mathbf{\Phi}_{21} & \mathbf{\Phi}_2 & \cdots & \mathbf{\Phi}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{\Phi}_{m1} & \mathbf{\Phi}_{m2} & \cdots & \mathbf{\Phi}_m \end{bmatrix} \begin{bmatrix} \hat{\mathbf{f}}_1 \\ \hat{\mathbf{f}}_2 \\ \vdots \\ \hat{\mathbf{f}}_m \end{bmatrix} + \begin{bmatrix} \hat{\mathbf{b}}_1 \\ \hat{\mathbf{b}}_2 \\ \vdots \\ \hat{\mathbf{b}}_m \end{bmatrix}, \tag{1}$$

where $\hat{\mathbf{a}}_i$ is the 6×1 spatial acceleration of link i, $\hat{\mathbf{f}}_i$ is the 6×1 spatial force applied to link i, $\hat{\mathbf{b}}_i$ is the 6×1 bias acceleration of link i (the acceleration link i would have if all link forces were zero), $\mathbf{\Phi}_i$ is the 6×6 inverse articulated-body inertia of link i, and $\mathbf{\Phi}_{ij}$ is the 6×6 cross-coupling inverse inertia between links i and j.

With these motion equations, standard numerical integration techniques such as Euler's method or various Runge-Kutta schemes can be used to determine the next state of the system. For complete dynamics simulation, it is necessary to determine how the body responds to interactions with the environment. Standard techniques use analytical constraints and impulse-based dynamics [42], [40].

B. Articulated Chain Planning

Randomized planning algorithms such as PRMs and RRTs have recently been popular due to their success in a wide range of applications and in high-dimensional configuration spaces, or C-spaces. For highly articulated robots, the performance of these planners is dictated by the dimensionality of the configuration space; a large number of samples are needed to adequately cover or explore the space.

C. Planning in state-space

We use kinodynamic planning which generates realistic motions and paths. Specifically, we incorporate both the kinematic and dynamics constraints directly into planning; by working in a *state space*, or *phase space*. This state space maintains the physical state of the robot, often its configuration augmented by trajectory information. However, the kinodynamic planning problem is often considered much more difficult since state spaces are typically at least twice the dimensionality of configuration spaces; for each element in configuration space, at least an associated velocity is typically required. Like configuration-space planners, the complexity for the problem lies mainly in the dimensionality of this space.

D. Sample Generation

The goal of our sampling algorithm is to improve sampling quality, as well as planning performance, when compared to prior approaches. We create a system which efficiently generates joint samples in a directed search direction. Samples are built off one another using integration methods like that of state-space RRTs [25]. Rather than randomly sampling in state or configuration space and moving in that direction, we use workspace distance information in order to generate forces which direct the growth. The ideas behind biasing forces are similar to potential field method for a high-DoF robot. This approach could also be used as the local planner in a PRM or RRT situation, providing solutions even when the configurations or states are relatively far apart. Each step in the algorithm would require running time linear in the number of joints. For extremely high-DoF robots (with thousands of joints), this is a limiting factor.

To improve performance, we use an observation about the robot. In highly articulated chains, it is known that added degrees of freedom actually *constrain* the configuration space. For instance, in a "smooth" configuration, the position of one joint has an influence on the position of the next joint in the chain. Rather than treat every joint as an independent degree of freedom, we allow certain joint angles to be fixed; the bodies connected by this joint effectively become a rigid body. Redon et al. [38] have shown how such ideas can be applied to efficient, error-bounded simulation of multi-body systems, called *adaptive forward dynamics*. Gayle et al. 2006 [39] extended this model to include contacts and collisions. We use this dynamics framework as the core of our sampling algorithm.

IV. Physics-based Sampling and Path Computation

We seek to determine the usefulness of a biased physics-based sampling strategy in the case of highly articulated robots. We consider a "highly" articulated body that contains hundreds to thousands of joints. For efficiency, we exploit the coherence between joint angles via the adaptive foward dynamics framework. Joints are first prioritized based upon well-defined motion metrics. Then, only the most important joints are simulated which effectively reduces the dimensionality of the search space. Potential bias forces are applied to encourage movement towards a goal. In this section, we describe our path generation and sampling algorithm.

A. Adaptive Forward Dynamics

Adaptive Forward Dynamics is used for computing the new state given the current state and a set of forces and

torques acting upon the body. It is necessary for physically-based motion and interactions with the environment.

1) Reduced Articulated Body Simulation: The simulation is based on ideas from Adaptive Forward Dynamics [38] and efficient impulse-based response [39]. Both methods utilize joint coherence to simplify the problem. Depending on the amount of motion a joint, or its associated subbody, generates with respect to the entire body, adaptive foward dynamics selects a subset of joints which to simulate. The remaining joints are simulated with zero joint velocity; their joint angle does not change and they are considered *rigidified*. Once a joint is rigidified, its entire subtree on the assembly tree (described in Section III-A) behaves like a rigid body. With respect to planning, this reduction is equivalent to planning through state spaces of reduced dimensionality. Additionally, the overall performance improves since the number of joints to simulate is less than the total number of joints.

Forward dynamics of the reduced body mimics the steps provided for Featherstone's recursive Divide-And-Conquer articulated body method [43]. The primary difference is that the equations of motion have been modified to allow rigidified subtrees. Since the subtrees behave like a rigid link, recursion terminates at the root of a rigidified subtree nodes in the subtree are not visited. This ensures a run-time that is sub-linear in the number of joints. Due to the space limit, we refer readers to [38] for the detailed approach.

2) Motion Metrics: Adaptive forward dynamics tracks *motion deviation* throughout the simulation. Motion deviation metrics are defined as a function of the joint accelerations, $\ddot{\mathbf{q}}$, and joint velocities $\dot{\mathbf{q}}$ for a subtree. For an articulated body, C, the *acceleration metric value* and the *velocity metric value* are given by

$$\mathcal{A}(C) = \sum_{i \in C} \ddot{\mathbf{q}}_i^T \mathbf{A}_i \ddot{\mathbf{q}}_i \qquad \mathcal{V}(C) = \sum_{i \in C} \dot{\mathbf{q}}_i^T \mathbf{V}_i \dot{\mathbf{q}}_i.$$

where $\mathbf{A}_i$ and $\mathbf{V}_i$, $i \in C$, are symmetric, positive definite (SPD) weight matrices whose dimension is $d_{J_i} \times d_{J_i}$, where d_{J_i} is the number of degrees of freedom of joint J_i. By using 1-DoF revolute joints, these result in simple scalar values. The simplest choice of such matrices is the identity matrix.

Note that the metric is defined over an articulated body, rather than for a specific joint. Each metric value then describes the motion of the entire subtree rooted at that joint. [38] shows how to evaluate the metric value without having to compute the accelerations of joints in C.

3) Prioritization of Joints: The metric value quantifies the amount of motion that would result by allowing the principal joint to move. This provides two ways of reducing the dimensionality, or number of *active* joints, of the articulated robot. One option is to set a motion metric threshold and ensure that the remaining motion is below this threshold. Or, one can specify a desired degree-of-freedom for the chain, and ensure that the most "important" joints are chosen in sampling the configuration space. We determine this set using a prioritized search through the joints.

Consider the case when a motion threshold, ϵ, is specified. The goal would be to simulate a sufficient number of joints such that the amount of total *remaining* motion is below ϵ. To do this, we traverse the assembly tree in a top down manner, starting with the root of the tree. While the total metric value in the queue is greater than ϵ, we pop off the front joint, J, which will be simulated. Of the unexplored joints, this one will result in the greatest motion. The metric values of the children of J are then evaluated and placed on the queue. This process is repeated until the total remaining motion, determined by inspecting the queue, falls below ϵ. In the case of a fixed n-DoF simplification, a similar process is used. We evaluate the front elements in the queue until n joints have been popped off, each time replacing the front node with its children.

4) Collisions: In order to compute accurate physically-based responses, it is necessary to model collisions with the environment. We use an impulse-based dynamics scheme that has the same run-time complexity as the adaptive forward dyanmics algorithm, [39].

The equations of motion for a robot in collision at an end effector is given by,

$$\begin{aligned}\ddot{q}(t) = \; & H^{-1}(q(t)) \left[Q(t) - C(q(t), \dot{q}(t)) \dot{q(t)} - G(q(t)) \right] \\ & + J^T(q(t)) \hat{f}(t),\end{aligned}$$

where H is the joint-space inertia matrix, C describes the Coriolis forces in matrix form, G describes external forces such as gravity, J is the Jacobian of the end effector, $\hat{f}$ is the external spatial force applied to the end effector, and Q is a vector of the magnitudes of forces and torques being applied by the joint actuators. Collisions at other points can be computed by aggregating the forces and computing the Jacobian at the contact location.

The complete impulse-based dynamics solution for this situation requires several passes up and down the articulated chain. [39] modifies this algorithm to make use of the reduced articulated structure. They propose changes to the algorithms coupled with a *Hybrid-Body Jacobian* in order to have a run-time asymptotically equivalent to that of adaptive forward dynamics. This Jacobian is equivalent to the Jacobian that would be computed if the articulated body had rigid links in place of rigidified subtrees and is given by,

$$J_P(q(t)) = \begin{pmatrix} \frac{\delta x}{\delta q_1} & \cdots & \frac{\delta x}{\delta q_{d_n}} \\ \frac{\delta y}{\delta q_1} & \cdots & \frac{\delta y}{\delta q_{d_n}} \\ \frac{\delta z}{\delta q_1} & \cdots & \frac{\delta z}{\delta q_{d_n}} \end{pmatrix}$$

where $P = (x, y, z)$, each q_i is an active joint, and d_n is number of active joints. Similarly, the response computed is correct for the reduced rigid body.

B. Potential Forces

To encourage the system toward a goal, we generate several types of constraint forces and torques. This has the similar effect of a potential-field planner guiding a robot towards a goal configuration. To avoid problems of local minima in the "potential" well, additional forces are introduced to ensure that the robot makes progress toward its goal.

The first force is a repulsive force that helps to keep the robot away from obstacles. If contact occurs with the environment, impulse-based responses are applied to ensure no penetration with obstacles. We can create simple limited radius repulsive force based upon the distance to the link. For each link, we apply a force

$$force_{ob_i}(b_j) = \begin{cases} -(\frac{\delta^2}{d(b_j,ob_i)^2} - 1)\hat{\mathbf{d}} & \text{if } d(b_j, ob_i) < \delta \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where δ is a distance threshold, $d(b_j, ob_i)$ is the distance between rigid link b_j and obstacle ob_i, and $\hat{\mathbf{d}}$ is the direction from b_j to ob_i.

The second is an attraction force to the goal configuration. This force can be applied to the entire body, a subset of links or joints, or alternatively to a single node or joint such as the end effector:

$$force_{goal}(\mathbf{q}) = w_{goal}\frac{(\mathbf{q}_{goal} - \mathbf{q})}{\|\mathbf{q}_{goal} - \mathbf{q}\|}$$

where w_{goal} is a goal node weight, $\mathbf{q}_{goal}$ is the goal configuration.

To escape local minima, we use a third *path-following force* at the cost of generating a path for a point robot in the workspace. This methodology works especially well in situations where the end effector needs to reach a point, and the configuration of the remainder of the robot is less important as long as it is valid. The path following force acts much like the goal attraction force, but instead uses a configuration along the path on the way to the goal.

The fourth is a torque rather than a force. It encourages the robot toward the desired goal configuration. This control is implemented as a linear torsion spring acting on a joint:

$$torque_{j_i} = -k_s(q_i - q_{goal}) - k_d\dot{q}_i$$

where k_s is the angular spring coefficient and k_d is an angular sprint damping coefficient.

C. Motion generation

The framework described here can be used in at least two ways; either as a local planner when coupled with some high-level randomized planner, or potentially as a way to determine the entire path. Pseudocode for the sampling algorithm is given in Algorithm 1.

The method begins by first intializing the robot, $\mathcal{R}$, with the starting configuration, q_{start}. If the robot is sufficiently close to the goal configuration, q_{goal}, then the planner has reached the goal. Otherwise, we apply the potential bias forces to the robot. Next, The adaptive forward dynamics solver is used to determine the joint accelerations and perform numerical integration in order to arrive at the next sample. Finally, prioritization of the joints determines which joints should be simulated; effectively reducing the dimensionality of the problem.

This results in a path or path segment that maintains kinematics and dynamics constraints at every intermediate configuration. With the efficient forward dynamics algorithm, we are able to compute this in a relatively small amount of time. Other probabilistic planners would likely require significantly more time or do not typically take both kinematics and dynamics constraints into account.

Input: Robot $\mathcal{R}$, Starting configuration q_{start}, Goal configuration q_{goal}, obstacles $o_1, \ldots, o_n$
Output: A path of robot states, $\mathcal{S}$

$\mathcal{R}^i.q \leftarrow q_{start}$;
while *d($\mathcal{R}^i$.q, q_{goal}) > GoalReachedThreshold* **do**
  Add $\mathcal{R}^i$ to $\mathcal{S}$;
  Apply potential bias forces to $\mathcal{R}^i$;
  `/* Generate next sample */`
  $\mathcal{R}^{i+1} \leftarrow$ AdaptiveDynamicsSolver($\mathcal{R}^i$);
  `/* Prioritization of Joints */`
  remainingMotion $\leftarrow \mathcal{A}(\mathcal{R}^{i+1}.treeRootNode)$;
  PriorityQueue $\mathcal{P}$.enqueue($\mathcal{R}^{i+1}$.treeRootNode);
  nodeCount = 0;
  while *remainingMotion > motionThreshold* **and** *nodeCount < numActiveNodes* **do**
    $n \leftarrow \mathcal{P}$.front;
    Mark n to be simulated;
    $\mathcal{P}$.enqueue(n.leftchild);
    $\mathcal{P}$.enqueue(n.righchild);
    remainingMotion $\leftarrow \Sigma_{n_i \in \mathcal{P}}\ \mathcal{A}(n_i)$;
    nodeCount $\leftarrow$ nodeCount $+1$;
  end
  $\mathcal{R}^i \leftarrow \mathcal{R}^{i+1}$;
end

Algorithm 1: Physically-based sampling and path computation

Scene	Total Joints	Active Joints	Env. (tri)	Robot (tri)	Sim. Time (s)	Avg. Step Time (s)
Walls	300	50	216	6000	17.5	0.0008
Tunnel	600	150	72	12000	66.92	0.003
Catheter	2500	200	80086	50000	1821	0.0071
Pipes	2000	200	38146	40000	193.6	0.0064
Debris	2000	175	1296	40000	157.3	0.0059

Fig. 2: Performance Table:

V. Results and Analysis

We implemented this algorithm on a Dell M60 Mobile Workstation, with a 2.1GHz Pentium M processor and 1GB of main memory.

A. Planning Results

We tested our algorithm on a number of benchmarks. In each case, a highly articulated robot in a serial linkage must navigate through an environment. The goal of the planning is for the end effector to reach a certain position in the workspace. The base links of the robot, including the thin rigid bodies as described in Section III-A, are cylinders, each represented by 20 triangular primitives.

- **Serial Walls** - This scenario is based on a benchmark created by the Texas A&M Parasol Laboratory. A 300 joint articulated chain must travel through a sequence of walls with holes. (See Fig. 4(a))

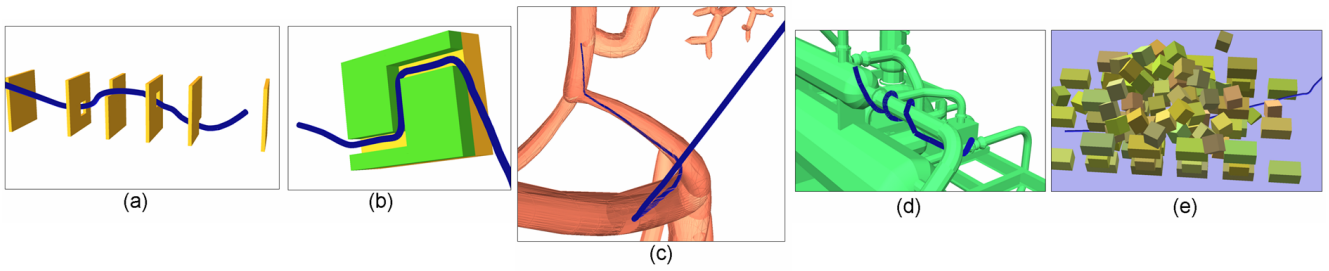

Fig. 4: Benchmark scenarios: (a) Serial Walls; (b) Tunnel; (c) Liver Catheterization; (d) Pipes; (e) Debris.

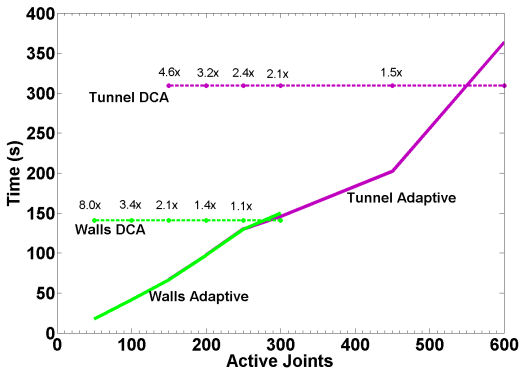

Fig. 3: Relationship between active joints and planning time for the Serial Walls (300 DoF) and Tunnel benchmarks (600 DoF). The horizontal line represents the time taken by using Featherstone's DCA algorithm. The values above it are the speed-ups over the DCA algorithm when that many active joints are simulated.

- **Tunnel** - Also based on a benchmark created by the Texas A&M Parasol Laboratory, the tunnel environment requires a 600 joint articulated chain to move through a tunnel. The tunnel has two right angle bends about half way through the block. (See Fig. 4(b))
- **Catheterization** - This scenario is based upon a medical procedure called *liver chemoembolization*. A thin, flexible catheter, modeled as a 2500-joint articulated robot, must travel through a network of arteries. The goal is to find a tumor in the liver for cancer treatment. (See Fig. 4(c))
- **Pipes** - This situation represents an application of snake-like robots in a pipe inspection task. The 2000-joint robot supports itself by coiling around a pipe while searching for a leak. The coiling behavior was encoded into the goal attraction force. (See Fig. 4(d))
- **Debris** - In this case, the 2000-joint robot aids in a search-and-rescue operation. The goal is to find an opening into the debris pile and to seek a large, open pocket where a survivor or important item may be found, and then to plan a way out. (See Fig. 4(e))

Our sampling method was able to complete the task *without* the explicit need to perform planning in the high-dimensional configuration or state space. Planning performance and sampling time using adaptive dynamics simulation is given in Figure 2. We show results for a fixed number of active joints that determine the reduced dimensionality of the configuration space for the articulated body. The fourth and fifth columns give the geometric complexity of the robot and environment. And, the last two columns give the total planning time followed by the average time it took to generate samples en route to the goal.

As can be seen, our algorithm has very favorable performance. Aside from the complexity of the environment and robot, other factors that affect the performance include the length of the path and any velocity constraints on the robot. Longer paths or slow speeds will increase the time it takes to reach a solution since the robot only travels a finite distance for each step. We observed upto an order of magnitude speed with 10% to 15% of the joints being active. Figure 3 shows the speed-up over simply using Featherstone's algorithm for a fixed number of active joints. The video associated with this work highlights the benefits of dimensionality reduction in planning performance, and demonstrates results for the pipes benchmark.

B. Motion Deviation Analysis

The effectiveness of the state space reduction is dependent upon the allowed motion threshold. To get an idea of how the motion affects dimension reduction, we tested various threshold settings in the Serial Walls environment. We have observed that by halving the motion metric threshold, the dimensionality increases by 9% to 38%. Also, there were certain threshold values which caused a much larger change than others. This is likely due to the fact that even though we reduce motion threshold, the current motion is close enough to the actual motion that it requires a smaller threshold to capture additional motion deviations. Since the total planning time is closely related to the average dimensionality, we noted that halving the threshold increased the planning time by about 10% to 34%.

While accurate simulation is not a focus of this planning work, we like to comment briefly on the kinematic and dynamics differences that results from the reduction. First, we note that quantitatively, motion deviation metrics as defined by adaptive dynamics is very difficult to interpret. Therefore, we consider the joint angle difference between adaptive dynamics and full dynamics simulation over the course of a planning situation. Fig. 5 shows the joint difference throughout the Serial Walls benchmark for varying amounts of motion threshold. Since the difference is bounded on a per time step basis, it has a tendency to slowly accumulate over the course of the planning. But, the potential forces help to reduce the total difference by ensuring that the robot

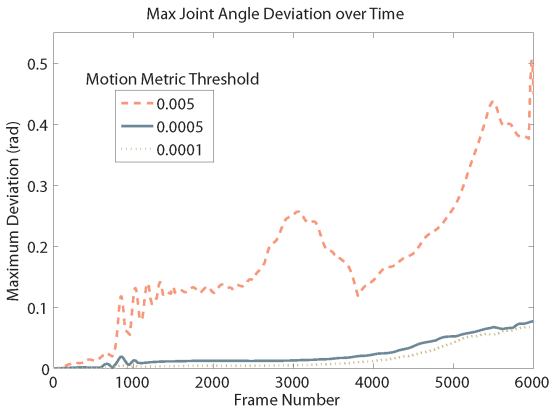

Fig. 5: Joint Angle Difference over Time - Each curve shows the maximum joint error during the Serial Walls benchmark for a fixed motion metric threshold.

follows roughly the same path regardless of the amount of simplification. This accounts for some of the sharp drops seen in the graph.

C. Discussion

There are several advantages and some disadvantages of this method. It is able to perform physically-based local planning in a high-DoF state space in a reasonable amount of time. Its performance makes it a favorable choice as a local planner for either PRM or RRTs to compute link queries. However, as the distance between the start and goal configuration increases, the time to find this path decreases. Additionally, it is possible that this method may fail to find a path, even if one exists. Although, in practice this has not been a problem when adequate forces are applied to the robot to escape local minima due to various constraints.

In the next subsections, we examine the resulting trajectory generated by the various sampling criteria. Note that for a highly articulated body of 300 joints the standard PRM and RRT algorithms were not able to find a solution within a suitable period of time. Thus, the following discussion examines the trajectory if such a path was found.

1) Path Validity: We first mention that paths generated by this algorithm are completely valid. Contact constraints and forces ensure that no penetration occurs between the robot and obstacles. When no dimension reduction occurs, the generated solution will clearly adhere to kinematic and dynamics constraints and joint angle limitations will not be violated. When joints are rigidified, they were already in a valid state and will remain that way if become active again. Therefore, at any time instant, all the joints will be in a valid configuration and the resulting path is valid.

2) Comparison with Configuration-Space Planners: For randomized C-space planners, we compare our method with solutions that might be generated via PRMs or RRTs. It should be noted that in most common implementations, a robot travels between milestones along a *straight-line* path. While this will yield valid results, it does not take both kinematics and dynamics into account. Even when the milestones themselves may be generated through kinematic simulation, the intermediate states are likely not. Thus, the quality of the resulting motion may not be as realistic, though the computed solutions may be adequate to solve the problem.

Arguably, if the space is more completely sampled by a random kinematics based method, then the milestones would be quite close. In that case, the motion generated by the resulting path would be much better. But, sufficient sampling of this space would not be practical.

In comparison, each configuration along a path generated by our method will follow a simplified kinematics and dynamics model. Thus, the resulting motion will look natural with respect to how the world is modeled and the biasing forces being applied.

3) Compared to state space planners: While deterministic kinodynamic solutions exist for some simple problems, many of these will not scale to the dimensionality of our benchmarks. Therefore, one of the randomized kinodynamic planners may be able to generate some solution. But, such an approach is likely to take an extraordinary amount of time and not practical. For instance, it took an RRT-based kinodynamic planner on average over 10 minutes to compute a path for just a rigid body in space (i.e. 12 DoF state space) on an 800 MHz Pentium III computer with 256 MB RAM [25]. While it would be much faster with current computational resources, it also was a very simple model.

Again, in our solution the kinematics and dynamics are modeled for a simplified model. Thus, while the resulting motion may not be identical to the full kinodynamic solution, it will not be much worse, assuming that we set a sufficiently small motion threshold as can be seen in Fig. 5.

VI. Conclusion and Future Directions

In this work, we suggest and analyze the effectiveness of a novel sampling approach or local planner for motion planning of highly articulated chains. In particular, we see that it generates results that respect kinematic and dynamics constraints in a reduced-dimension configuration space. Our method exploits the coherence between joint angles in order to determine which joints have the greatest impact on the overall motion. This method results in improved performance, but may not be fully identical to a kinodynamic solution. However, we argue that it offers better solutions than a completely randomized approach in the configuration space. Thus, our planning results fall between the configuration space planners and kinodynamic planners. This reduced model has direct applications to modular robots in which certain segments simply maintain their current joint angle rather than allowing it to change.

There are several directions for future research in the area. Currently the planning is performed independently of a high-level planner. It would be interesting to see if this work would help a standard PRM or RRT planner in finding a solution within a shorter period of time, or if it would improve the overall quality of the generated motion. It would also be interesting to see if other biologically behaviors can

be encoded by our solution, such as how to get the model to move as a snake might.

VII. Acknowledgements

The authors would like to thank our reviewers for the useful feedback which helped to improve the work. This work was supported by a Department of Energy High-Performance Computer Science Fellowship administered by the Krell Institute and in part by ARO Contracts DAAD19-02-1-0390 and W911NF-04-1-0088, NSF awards 0400134, 0429583 and 0404088, DARPA/RDECOM Contract N61339-04-C-0043 and Disruptive Technology Office.

References

[1] H. Choset and J. Burdick, "Extensibility in local sensor based planning for hyper redundant manipulators (robot snakes)," *AIAA/NASA CIRFFSS*, 1994.

[2] W. Henning, F. Hickman, and H. Choset, "Motion planning for serpentine robots," *Proc. of ASCE Space and Robotics*, 1998.

[3] A. Wolf, H. B. Jr., R. Casciola, A. Costa, M. Schwerin, E. Shammas, and H. Choset, "A mobile hyper redundant mechanism for search and rescue tasks," *Proc. of IROS*, vol. 3, pp. 2889–2895, 2003.

[4] G. S. Chirikjian and J. W. Burdick, "An obstacle avoidance algorithm for hyper-redundant manipulators," *IEEE Transactions of Robotics and Automations*, 1990.

[5] ——, "A modal approach to hyper-redundant manipulator kinematics," *IEEE Transactions on Robotics and Automation*, pp. 343–354, 1994.

[6] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, pp. 12(4):566–580, 1996.

[7] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," *Proceedings of the IEEE Intnational Conference on Robotics and Automation (ICRA)*, April 2000.

[8] E. Plaku and L. Kavraki, "Quantitative analysis of nearest-neighbor search in high-dimensional sample-based motion planning," *Proc. of WAFR*, 2006.

[9] J. Latombe, "Motion planning: A journey of robots, molecules, digital actors, and other artifacts," *International Journal of Robotics Research*, pp. 1119–1128, 1999.

[10] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, 2005.

[11] S. M. LaValle, *Planning Algorithms*. Cambridge University Press (also available at http://msl.cs.uiuc.edu/planning/), 2006.

[12] D. Sun, X. Shi, and Y. Liu, "Modeling and cooperation of two-arm robotic system manipulating a deformable object," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 2346–2351, 1996.

[13] H. Nakagaki and K. Kitagaki, "Study of deformation tasks of a flexible wire," *Proc. of IEEE Int. Conf. on Robotics and Automation*, 1997.

[14] E. Anshelevich, S. Owens, F. Lamiraux, and L. Kavraki, "Deformable volumes in path planning applications," *IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 2290–2295, 2000.

[15] A. Ladd and L. Kavraki, "Using motion planning for knot untangling," *International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 797–808, 2004.

[16] M. Moll and L. Kavraki, "Path planning for variable resolution minimal-energy curves of constant length," *Proceedings of International Conference on Robotics and Automation*, pp. 2143–2147, 2005.

[17] R. Gayle, P. Segars, M. Lin, and D. Manocha, "Path planning for deformable robots in complex environments," University of North Carolina-Chapel Hill, Tech. Rep., 2005, proc. of Robotics: Science and Systems.

[18] M. Teodoro, G. N. J. Phillips, and L. E. Kavraki, "Understanding protein flexibility through dimensionality reduction," *The Journal of Computational Biology*, vol. 10, no. 3-4, pp. 617–634, 2003.

[19] J. Barraquand, "Automatic motion planning of complex articulated bodies," Tech. Rep., 1991.

[20] S. LaValle, J. Yakey, and L. Kavraki, "A probabilistic roadmap approach for systems with closed kinematic chains," *Proceedings of the IEEE Intnational Conference on Robotics and Automation (ICRA)*, 1999.

[21] L. Han and N. M. Amato, "A kinematics-based probabilistic roadmap method for closed chain systems," *Proceedings of the fourth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2000.

[22] J. Cortes, T. Simeon, and J. P. Laumond, "A random loop generator for planning the motions of closed kinematic chains using prm methods," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2002.

[23] B. R. Donald, P. Xavier, J. Canny, and J. H. Reif, "Kinodynamic motion planning," *Journal of ACM*, vol. 40, no. 5, pp. 1048–1066, Nov. 1993.

[24] D. Hsu, R. Kindel, J.-C.Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *International Journal of Robotics Research*, 2002.

[25] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *In Proc. IEEE Int'l Conf. on Robotics and Automation*, 1999.

[26] A. M. Ladd and L. E. Kavraki, "Fast tree-based exploration of state space for robots with dynamics," in *Algorithmic Foundations of Robotics VI*, M. Erdmann, D. Hsu, M. Overmars, and A. F. van der Stappen, Eds. Springer, STAR 17, 2005, pp. 297–312.

[27] S. Rodriguez, J.-M. Lien, and N. M. Amato, "Planning motion in completely deformable environments," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2006.

[28] T. Simeon, J. P. Laumond, and C. Nissoux, "Visibility based probabilistic roadmaps for motion planning," *Advanced Robotics Journal*, vol. 14, no. 6, 2000.

[29] C. Pisula, K. Hoff, M. Lin, and D. Manocha, "Randomized path planning for a rigid body based on hardware accelerated voronoi sampling," in *Proc. of 4th International Workshop on Algorithmic Foundations of Robotics*, 2000.

[30] L. Guibas, C. Holleman, and L. Kavraki, "A probabilistic roadmap planner for flexible objects with a workspace medial-axis-based sampling approach," in *Proc. of IROS*, 1999.

[31] S. Wilmarth, "A probabilistic method for rigid body motion planning using sampling from the medial axis of the free space," Ph.D. dissertation, Texas A&M University, Dec 1999.

[32] N. Amato, O. Bayazit, L. Dale, C. Jones, and D. Vallejo, "Obprm: An obstacle-based prm for 3d workspaces," *Proceedings of WAFR98*, pp. 197–204, 1998.

[33] B. Burns and O. Brock, "Toward optimal configuration space sampling," *Proceedings of Robotics: Science and Systems*, 2005.

[34] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2003.

[35] S. R. Lindemann and S. M. LaValle, "Incrementally reducing dispersion by increasing voronoi bias in RRTs," *Proceedings of the IEEE Conference on Robotics and Automation (ICRA)*, 2005.

[36] A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle, "Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain," in *Proceedings IEEE International Conference on Robotics and Automation*, 2005.

[37] M. Kalisiak and M. van de Panne, "RRT-blossom: RRT with a local flood-fill behavior," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2006.

[38] S. Redon, N. Galoppo, and M. Lin, "Adaptive dynamics of articulated bodies," *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)*, vol. 24, no. 3, 2005.

[39] R. Gayle, M. Lin, and D. Manocha, "Adaptive dynamics with efficient contact handling for articulated robots," *Proc. of Robotics: Systems and Science*, 2006.

[40] V. Kokkevis, "Practical physics for articulated characters," *Proc. of GDC*, 2004.

[41] R. Featherstone and D. E. Orin, "Robot dynamics: Equations and algorithms," *IEEE Int. Conf. Robotics and Automation, pp. 826-834*, 2000.

[42] B. V. Mirtich, "Impulse-based dynamic simulation of rigid body systems," Ph.D. dissertation, University of California, Berkeley, 1996.

[43] R. Featherstone, "A divide-and-conquer articulated body algorithm for parallel o(log(n)) calculation of rigid body dynamics. part 1: Basic algorithm," *International Journal of Robotics Research 18(9):867-875*, 1999.