High-Dimensional Underactuated Motion Planning via Task Space Control

Alexander Shkolnik and Russ Tedrake

***Abstract*— Kinodynamic planning algorithms have the potential to find feasible control trajectories which accomplish a task even in very nonlinear or constrained dynamical systems. Underactuation represents a particular form of a dynamic constraint, inherently present in many machines of interest (e.g., walking robots), and necessitates planning for long-term control solutions. A major limitation in motion planning techniques, especially for real-time implementation, is that they are only practical for relatively low degree-of-freedom problems. Here we present a model-based dimensionality reduction technique based on an extension of partial feedback linearization control into a task-space framework. This allows one to plan motions for a complex underactuated robot directly in a low-dimensional task-space, and to resolve redundancy with lower-priority tasks. We illustrate the potential of this approach with an extremely simple motion planning system which solves the swing-up problem for multi-link underactuated pendula, and discuss extensions to the control of walking.**

I. Introduction

A robotic system is "underactuated" if a control system cannot produce arbitrary accelerations in some of the degrees of freedom at every instant in time. Underactuated systems are typified by robots that have more degrees-of-freedom (DOFs) than actuators, such as the Acrobot [1] or Pendubot [2]. Many important classes of robots are unavoidably underactuated, including most walking, swimming, and flying machines. For instance, walking machines are typically underactuated at the interface between the foot and the ground; the control system cannot produce arbitrary ground reaction forces or moments. This is especially true for point-foot walkers (e.g., [3]), but is also true for flat-foot walkers like Honda's ASIMO's [4] or Kawada's HRP-2 [5]. For machines with large support polygons, the location of the Zero-Moment Point (ZMP) has proven to be a useful metric for avoiding the dynamic constraints of underactuation - when the ZMP is regulated to stay inside a large flat foot, the foot will not roll, and the remaining degrees of freedom can be controlled as if the system is fully actuated. But this ZMP constraint results in overly conservative dynamic trajectories. Because the control systems do not reason about the underactuated dynamics, the humanoid robots do not perform well when walking on any significantly rough or unmodelled terrain, cannot move nearly as quickly as humans, and use dramatically more energy (scaled) than a human [6].

There has been a considerable amount of work on control design for underactuated systems. Such systems are not feedback linearizable, but in many cases it is possible to achieve a *partial feedback linearization* (PFL) [7], in which the dynamics of some subset of the degrees of freedom are replaced (through feedback) with the dynamics of a simple linear system. This subset is not restricted to simply the actuated joints, in fact it is often possible to exploit inertial coupling to linearize the dynamics of the unactuated joints. The machinery developed for partial feedback linearization includes a very compact and straight-forward mechanism for reasoning about, and controlling, the couplings between actuated and unactuated joints in robotic manipulators.

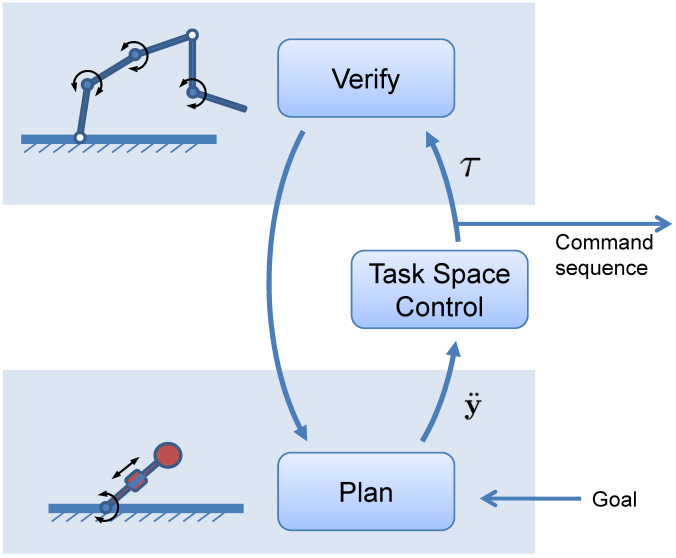

Fig. 1. Schematic of an underactuated manipulator (top left) and the lower-dimensional fully-actuated model (bottom left) that defines the task space of the problem. Plans generated in task space are mapped back to the active joints of the underactuated system, and are verified to satisfy any constraints in the underactuated system. (White circles represent passive joints.)

In this paper, we present an extension of the PFL ideas to control in task space. Task space control [8] has become very popular in recent years, in part, because of the natural mechanism of redundancy resolution and task prioritization (e.g., [8], [9]); we include these ideas in our derivation. Task space control has been widely explored in fully actuated systems, as well as Operation Space control in floating-base systems [10]. However, to accomplish sophisticated tasks, underactuated systems with joint constraints must reason about the possibility that the control system cannot execute arbitrary trajectories in task space. In fact, [2] anticipated an extension of PFL to task space but observed that finding feasible trajectories can be very difficult, [11] developed a Cartesian-space PFL and suggested the need for planning, and [12] used a constrained configuration space planner in an initial attempt to design feasible trajectories for a subclass

A. Shkolnik is a PhD candidate in EECS at the Computer Science and Artificial Intelligence Lab, MIT, 32 Vassar St., Cambridge, MA 02139, USA shkolnik@mit.edu

R. Tedrake is an Assistant Professor of EECS at the Computer Science and Artificial Intelligence Lab, MIT, 32 Vassar St., Cambridge, MA 02139, USA russt@mit.edu

of underactuated systems with brakes.

Recent advances in kinodynamic motion planning algorithms [13], [14], spurred on by randomized motion planners, have matured our understanding of computational methods for designing feasible trajectories in task space. An essential observation in our work is that planning in task space, when done correctly, is a much lower-dimensional planning problem than planning directly in the state space of the robot. In a real sense, the control mapping we provide from task space to (underactuated) joint space is a mechanism for exploiting known properties of the dynamics of the robot in order to dramatically improve the efficiency of search. To be practical, the task-space planner must respect constraints that exist only in the higher-dimensional state space (such as joint and torque limits). The concept of planning in a low dimensional task space and verifying constraints in joints space is illustrated in Figure 1. In this paper we present an extremely simple planning scheme which satisfies this framework, which is only feasible because of the significant reduction in dimensionality in the search. We hope that the success reported here will inspire more work on intelligent planning algorithms which can exploit the task-space mapping, and we expect that fast randomized planners such as Rapidly-exploring random trees (RRTs) in particular, will be amenable to the approach presented.

In addition to developing the framework, we report the results of our experiments on a classically hard motion planning problem - the swing-up task for an underactuated pendulum with an arbitrary number of links (aka, the N-link pendulum). The computational complexity of the algorithm we describe is invariant to the number of links; we focus here on results with a five-link pendula. Finally, we discuss some extensions to motion planning for dynamic locomotion on rough terrain.

II. Templates and Anchors

The concept of "embedding" low-dimensional dynamics into a high-dimensional dynamical system has a long history in control, and robotic locomotion research is no exception (e.g., [15], [16]). [17] gave this idea some broad appeal by describing "templates and anchors", and suggesting that the embedding of simple models (the templates) can describe neuromechanical invariants in real animals (the anchors). One of the hallmark examples of this work is the spring-loaded inverted pendulum (SLIP) model, which explains center-of-pressure to center-of-mass dynamics in an incredible diversity of creatures, and also has been used to design control systems for dynamic locomotion in robots [18]. Most work in templates and anchors (e.g., [19], [20]) use an inverse-dynamics mapping to produce the embedding; consequently when the anchor is underactuated, the template must be chosen carefully to guarantee that dynamics remain invertible. Our approach in this paper is different, as we attempt to embed a fully-actuated template into an underactuated anchor.

The disadvantage of embedding a fully-actuated system into an underactuated system is readily apparent - not all trajectories of the underlying fully-actuated system are feasible due to constraints. This would complicate any analytical controller design process. Our motivation here is different; we mean to exploit computational methods for generating feasible trajectories in the template system. The key element is not an analytical simplification of the control design, but simply a reduction in the dimensionality of the problem designed to facilitate computational solutions via search.

III. Task Space Control of Underactuated Systems

Without loss of generality, the equations of motion of an underactuated system can be broken down in two components, one corresponding to unactuated joints, $\mathbf{q}_1$, and one corresponding to actuated joints, $\mathbf{q}_2$:

$$\mathbf{M}_{11}\ddot{\mathbf{q}}_1 + \mathbf{M}_{12}\ddot{\mathbf{q}}_2 + \mathbf{h}_1 + \boldsymbol{\phi}_1 = 0 \tag{1}$$

$$\mathbf{M}_{21}\ddot{\mathbf{q}}_1 + \mathbf{M}_{22}\ddot{\mathbf{q}}_2 + \mathbf{h}_2 + \boldsymbol{\phi}_2 = \boldsymbol{\tau} \tag{2}$$

with $\mathbf{q} \in \Re^n$, $\mathbf{q}_1 \in \Re^m$, $\mathbf{q}_2 \in \Re^l$, $l = n - m$. Consider an output function of the form,

$$\mathbf{y} = \mathbf{f}(\mathbf{q}),$$

with $\mathbf{y} \in \Re^p$, which defines the task space. Define $\mathbf{J}_1 = \frac{\partial \mathbf{f}}{\partial \mathbf{q}_1}$, $\mathbf{J}_2 = \frac{\partial \mathbf{f}}{\partial \mathbf{q}_2}$, $\mathbf{J} = [\mathbf{J}_1, \mathbf{J}_2]$.

Theorem 1 (Task Space PFL): If the actuated joints are commanded so that

$$\ddot{\mathbf{q}}_2 = \bar{\mathbf{J}}^+ \left[\mathbf{v} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}_1\mathbf{M}_{11}^{-1}(\mathbf{h}_1 + \boldsymbol{\phi}_1) \right], \tag{3}$$

where $\bar{\mathbf{J}} = \mathbf{J}_2 - \mathbf{J}_1\mathbf{M}_{11}^{-1}\mathbf{M}_{12}$. and $\bar{\mathbf{J}}^+$ is the right Moore-Penrose pseudo-inverse,

$$\bar{\mathbf{J}}^+ = \bar{\mathbf{J}}^T(\bar{\mathbf{J}}\bar{\mathbf{J}}^T)^{-1},$$

then we have

$$\ddot{\mathbf{y}} = \mathbf{v}. \tag{4}$$

subject to

$$\text{rank}\left(\bar{\mathbf{J}}\right) = p, \tag{5}$$

Proof: Differentiating the output function we have

$$\dot{\mathbf{y}} = \mathbf{J}\dot{\mathbf{q}}$$

$$\ddot{\mathbf{y}} = \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}_1\ddot{\mathbf{q}}_1 + \mathbf{J}_2\ddot{\mathbf{q}}_2.$$

Solving 1 for the dynamics of the unactuated joints we have:

$$\ddot{\mathbf{q}}_1 = -\mathbf{M}_{11}^{-1}(\mathbf{M}_{12}\ddot{\mathbf{q}}_2 + \mathbf{h}_1 + \boldsymbol{\phi}_1) \tag{6}$$

Substituting, we have

$$\ddot{\mathbf{y}} = \dot{\mathbf{J}}\dot{\mathbf{q}} - \mathbf{J}_1\mathbf{M}_{11}^{-1}(\mathbf{M}_{12}\ddot{\mathbf{q}}_2 + \mathbf{h}_1 + \boldsymbol{\phi}_1) + \mathbf{J}_2\ddot{\mathbf{q}}_2 \tag{7}$$

$$= \dot{\mathbf{J}}\dot{\mathbf{q}} + \bar{\mathbf{J}}\ddot{\mathbf{q}}_2 - \mathbf{J}_1\mathbf{M}_{11}^{-1}(\mathbf{h}_1 + \boldsymbol{\phi}_1) \tag{8}$$

$$= \mathbf{v} \tag{9}$$

Note that the last line required the rank condition (5) on $\bar{\mathbf{J}}$ to ensure that the rows of $\bar{\mathbf{J}}$ are linearly independent, allowing $\bar{\mathbf{J}}\bar{\mathbf{J}}^+ = \mathbf{I}$. ■

In order to execute a task space trajectory one could command

$$\mathbf{v} = \ddot{\mathbf{y}}^d + \mathbf{K}_d(\dot{\mathbf{y}}^d - \dot{\mathbf{y}}) + \mathbf{K}_p(\mathbf{y}^d - \mathbf{y}).$$

Assuming the internal dynamics are stable, this yields converging error dynamics, $(\mathbf{y}_d - \mathbf{y})$, when $\mathbf{K}_p, \mathbf{K}_d > 0$ [21].

For a position control robot, the acceleration command of (3) suffices. Alternatively, a torque command follows by substituting (3) and (6) into (2).

A. Collocated and Non-Collocated PFL

The task space derivation above provides a convenient generalization of the partial feedback linearization (PFL) [7], which emcompasses both the collocated and non-collocated results. If we choose $\mathbf{y} = \mathbf{q}_2$ (collocated), then we have

$$\mathbf{J}_1 = 0, \mathbf{J}_2 = 1, \dot{\mathbf{J}} = 0, \bar{\mathbf{J}} = 1, \bar{\mathbf{J}}^+ = 1.$$

From this, the command in (3) reduces to $\ddot{\mathbf{q}}_2 = v$. The torque command is then

$$\tau = -\mathbf{M}_{21}\mathbf{M}_{11}^{-1}(\mathbf{M}_{12}v + \mathbf{h}_1 + \boldsymbol{\phi}_1) + \mathbf{M}_{22}v + \mathbf{h}_2 + \boldsymbol{\phi}_2,$$

and the rank condition (5) is always met.

If we choose $\mathbf{y} = \mathbf{q}_1$ (non-collocated), we have

$$\mathbf{J}_1 = 1, \mathbf{J}_2 = 0, \dot{\mathbf{J}} = 0, \bar{\mathbf{J}} = -\mathbf{M}_{11}^{-1}\mathbf{M}_{12}.$$

The rank condition (5) requires that $\text{rank}(\mathbf{M}_{12}) = l$, in which case we can write $\bar{\mathbf{J}}^+ = -\mathbf{M}_{12}^+\mathbf{M}_{11}$, reducing the rank condition to precisely the "Strong Inertial Coupling" condition described in [7]. Now the command in (3) reduces to

$$\ddot{\mathbf{q}}_2 = -\mathbf{M}_{12}^+ \left[\mathbf{M}_{11}\mathbf{v} + (\mathbf{h}_1 + \boldsymbol{\phi}_1)\right] \tag{10}$$

The torque command is found by substituting $\ddot{\mathbf{q}}_1 = v$ and (10) into (2), yielding the same results as in [7].

B. Redundancy Resolution

In situations where l is greater than p, there are infinite solutions to achieve the desired task-space command. We may use a Null-space projection to execute a second-priority task in actuator-space:

Theorem 2 (Task Space PFL w/ Redundancy Resolution): If the actuated joints are commanded so that

$$\ddot{\mathbf{q}}_2 = \bar{\mathbf{J}}^+ \left[\mathbf{v} - \dot{\mathbf{J}}\dot{\mathbf{q}} + \mathbf{J}_1\mathbf{M}_{11}^{-1}(\mathbf{h}_1 + \boldsymbol{\phi}_1)\right] + \alpha(\mathbf{I} - \bar{\mathbf{J}}^+\bar{\mathbf{J}})\ddot{\mathbf{q}}_2^{ref}, \tag{11}$$

and the condition (5) is met, then we have

$$\ddot{\mathbf{y}} = \mathbf{v}.$$

Proof: The proof is identical to Theorem 1, because the null-space terms are canceled by the pre-multiplication of $\bar{\mathbf{J}}$, and therefore do not contribute to $\ddot{\mathbf{y}}$. ■

Redundancy resolution is a very important aspect of task space control, particularly for underactuated systems (see e.g. [22]). The command $\ddot{\mathbf{q}}_2^{ref}$ can be thought of as a lower-priority task, which will be executed in the null-space of the high priority task. If the command is zero, the controller will return the solution with the smallest norm. More clever redundancy resolution tasks can help ensure the underactuated system stays dynamically within a regime where there are more actions that can be taken, a characteristic which can improve search performance.

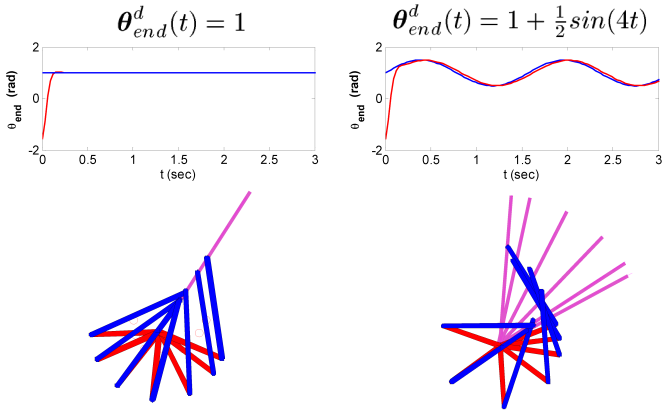

Fig. 2. Acrobot Control: Top, trajectory of control output function (angle from base to end effector); Blue is desired, red is actual. Bottom, snapshots of Acrobot during execution. Red is first link, Blue is second link, Magenta is commanded angle.

IV. Feedback Control of the Acrobot

We begin by exploring task space control on a two link arm with a passive base joint. From herein we will refer to N-link arms with mixtures of Passive and Actuated joints by shorthand notation of combinations of **P** and **A** representing the respective type of joint. This two link system, **PA**, is known as the Acrobot [23]. Direct application of task-space PFL on this system *without planning* clearly demonstrates the power of the feedback linearization and the need for motion planning.

Specifically, we control an Acrobot to execute behaviors where the task space consists of either 1) the angle from the base to the COM, $\boldsymbol{\theta}_{COM}$ or 2) the angle from base to the end effector, $\boldsymbol{\theta}_{end}$. We consider two potential output trajectories: 1) maintain a constant angle of 1 radian; $\boldsymbol{\theta}_d(t) = 1$, or 2) follow an arbitrary desired trajectory - in this case we define it as $\boldsymbol{\theta}_d(t) = 1 + .5\sin(4t)$. When the task space is $\boldsymbol{\theta}_{COM}$ and the torque is unconstrained both output trajectories are followed well, but the controller commands excessive torques and the actuated joint spins. We obtain similar results for controlling the end-effector angle, however in the special case if the second link is substantially longer than the first link, then the link does not spin, and instead the system "oscillates" back and forth along the line specified by $\mathbf{y}_{end_d}(t) = 1$, as shown in figure 2. This is a demonstration of how the choice of task space can drastically affect the system dynamics. If the goal is to perform swing-up behavior, for the Acrobot it may be better to control the end effector. As we may only control one degree of freedom with one actuator, treating the system as a pendulum and controlling the center of mass angle ignores the constantly changing length, which makes the pendulum (angle to COM) a poor choice of template. With redundant degrees of freedom for cases where $N > 2$, however, we may regulate both the angle and length of the pendulum, and so this becomes our task space of choice.

Similar results can be attained with higher DOF systems. These simulation experiments reveal the need for motion planning to enforce joint and torque limits. The dynamic constraint imposed by the rank condition in equation 5 was never actually violated - degeneracy in the Jacobian always

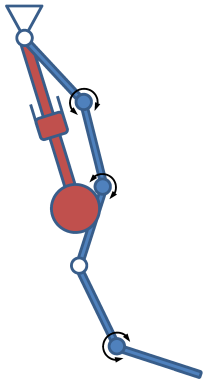

Fig. 3. A low-dimensional template for the underactuated swing-up task.

presented itself with excessive torques before the mapping became truly singular.

V. Planning

In dealing with the dimensionality problem, we suggest planning in a lower dimensional task space. Taking actions in task space ($\ddot{\mathbf{y}}$) confines the possible actions which can be taken from any given state (τ). Constraining potential actions in this way is a feature that many planners can take advantage of. Unfortunately, although we can map actions from task space to joint space, we can not map constraints from joint space to task space. Thus, the planner must consider what happens in joint space when a given task is applied. Typically, this can be accomplished by computing the mapping from task accelerations to torques, simulating the full system given those torques, and verifying that no constraints are violated. The intent is that the reduced set of potential actions is powerful enough to enable the planner to find a path from start to goal. However, reachability to the goal will also depend on the choice of redundancy resolution. We should note that the task controller does not necessarily carve out a manifold within state space. The controller constrains the actions, but the entire feasible state space may still be reachable. Thus, completeness remains a function of the higher-level planner, and is not necessarily constrained by the controller.

If the task space is of low enough dimension, a simple quasi-exhaustive tree search may be performed. More clever algorithms such as RRTs [24] can be utilized and will only improve the performance, but we begin by exploring a near-brute-force search approach illustrated in Algorithm 1, which would clearly not work in higher dimensional systems. This illustrates the effectiveness of the dimensionality reduction. Except for line 23, this is essentially a brute force search algorithm. The pruning step of line 23 keeps the tree from expanding exponentially at each depth.

VI. Generalized Swing Up Control for N-Link Pendula with Passive Joints

A. Simulations of a 5-link pendulum

We consider a general, and classical, problem in underactuated control - the swing-up task (e.g., [25]). Consider a multi-link underactuated pendulum in a gravitational field. The task is to move the center of mass of the pendulum to the vertical configuration. Many of the control solutions

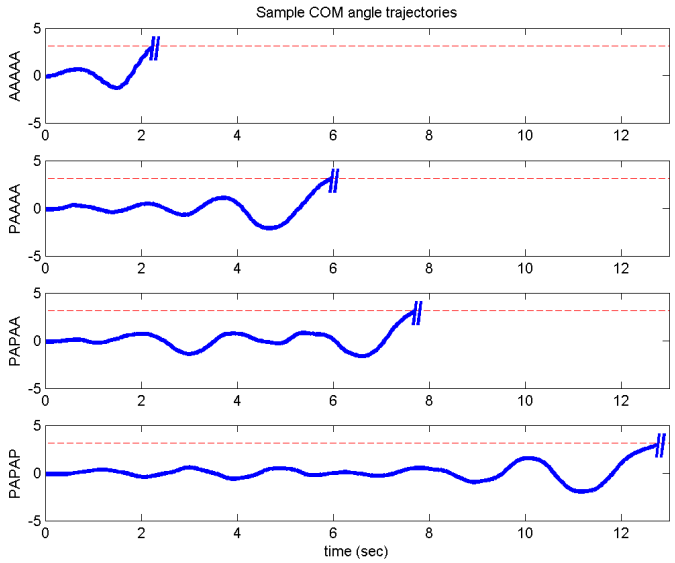

Fig. 4. COM trajectories

investigated in this problem use energy-based methods; for simple pendula, regulating the total energy of the system is enough to place the system on a homoclinic orbit [26]. Most work on pendula with passive joints have focused on cases with 2 or 3 links [23], [27]. With more degrees of freedom, if searching directly for a torque trajectory over the actuators, the problem is crippled by the curse of dimensionality [28]. In addition to increased system dimension, we further complicate the problem by considering limitations of joint positions, and torques.

For reasons alluded to previously, we consider a two-DOF pendulum, with control over angle and length as our template. This template and an example 5-Link pendulum is illustrated in Figure 3. We ran our algorithm on the following systems: AAAAA, PAAAA, PAPAA, and PAPAP. In all cases, we were able to find a trajectory to the goal. Parameters of the system were as follows:

Mass of each link	.2kg
Length	.2m
Moment of Inertia	$.05kg \cdot m^2$
Torque Limitation	2 Nm

In practice, we utilized the method in Algorithm 2 for pruning, where the metric function, M, was based on the energy difference between a given state and the goal, and penalized by a weighted sum of joint space distance away from the straight pose, where joints closest to the base were more heavily weighted and therefore encouraged to be straighter than joints further from the base. This metric encourages pumping energy into the system, but tries to stay away from poses which wrap around. Nodes are probabilistically pruned, biased by the metric, a step that helps to ensure that some feasible paths remain if most high-scoring nodes are actually headed to a dead-end.

Example center of mass trajectories for all four cases are compared in Figure 4. An entire swing-up trajectory is illustrated for the PAAAA case in Figure 6, and the final

Algorithm 1 Simple Planning

Require: discrete set of actions, $V \in \mathbb{R}^{Nxp}$, Depth limit of D, Depth node limit of B

Add-node (x_0, root)
for depth d = 1 to D **do**
  $n \Leftarrow 0$
  for each node, x_n at depth d **do**
    for each $V_i \in V$ **do**
      $tau_H \Leftarrow$ task-control(x_n, V_i)
      if $|tau_H < tau_{LIMIT}|$ **then**
        Simulate: $x_{new} \Leftarrow f(x_n, tau_H, \Delta T)$
        if NO-CONSTRAINTS-VIOLATED(x_{new}) **then**
          Add-node(x_{new}, X_n)
          $n \Leftarrow n + 1$
        end if
      end if
    end for
  end for
  if Goal Reached **then**
    RETURN (GOAL REACHED)
  end if
  if n == 0 **then**
    RETURN (NO PATH FOUND)
  end if
  if $n > B$ **then**
    prune B-n leaves from depth d
  end if
end for

Algorithm 2 Probabilistic Pruning

Require: set of leaves, $\mathbf{X}_{leaves}$, leaf limit B, goal $\mathbf{X}_{goal}$, $0 < \alpha < 1$

Compute a distance metric: $H = M(\mathbf{X}, \mathbf{X}_{goal})$, where H is normalized between 0 (furthest) and 1 (closest). This metric can also account for "'bad poses"' or any other costs on the state.
Sort($(\alpha(H/2) + (1 - \alpha)$)*Random-Number-Generator
Return best N nodes

three swings for the PAPAA case are shown in Figure 7. We omit the first half of the PAPAA trajectory and the PAPAP trajectory for brevity, but the full swing-ups are shown in the accompanying video.

B. Computational Efficiency

It is easy to see that Algorithm 1 is

$$O(D \cdot K \cdot N^{(2p)}).$$

This is substantially better than the case without pruning, which would grow exponentially on D.

To illustrate the time savings offered by the dimensionality reduction, consider the PAAAA example of the preceding section, with 1 passive joint, 4 actuators, and a 2-DOF task space consisting of the angle and length to COM (l = 1, m=4, p=2). Let $K = (B/N^p)$ be the average branching factor per node. In the results presented above, we use K=2, D=65 (ΔT=.15), N=11. Then if we search in the lower dimensional task space where p=2, the search time is $O(D*K*N^{(2p)}) \propto$ 2 million nodes. If we attempted to search with the same discretization factor in the full joint space (note the branching factor blows up to $B = K * N^4$), the equivalent time would be $O(D * K * N^{(2p)}) \propto 30$ billion nodes.

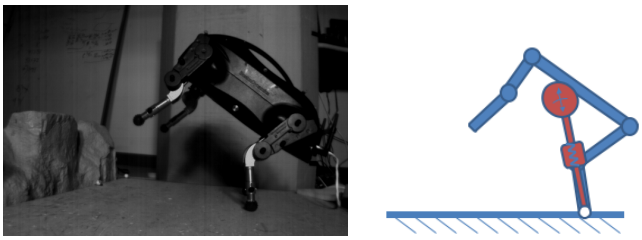

Fig. 5. LittleDog

Although conventional RRTs could potentially solve the 5 link swing-up problem, note that the computational efficiency of the search proposed here is invariant to the number of links.

VII. Discussion

A. Planning

The planning algorithm presented here was intentionally kept simple, and was meant to demonstrate the power of the dimensionality reduction in the search, as a planning algorithm with similar search density in the full state space would have been computationally intractable. The contribution here is in addressing the underactuated control as a search problem, and using task space to constrain the search. We note, however, that any planner can be used if it satisfies the framework of Figure 1. For example, recent work on RRTs has begun to address task-space search. [14] utilizes task space to enable a bidirectional RRT in a grasping problem. A forward RRT grows from the start pose in configuration space, and backward tree grows from the goal in task space; occasionally the forward tree is allowed to grow along the task tree as far as possible. Another related work is the JT-RRT [29] which occasionally has the forward tree grow in task space towards the goal using the Jacobian transpose. Such an approach would be very amenable to the task space controller presented in this paper, and is a future direction we are exploring to enable real-time planning for underactuated systems.

Note that the result of our planning system is a feasible open-loop trajectory of the underactuated system, but that our derivation also provides a feedback stabilization for that trajectory.

B. Choice of template

Our first attempts to generate a swing up controller were done by using a 1-DOF pendulum as a template. It seemed logical that we could stabilize length in the Null space, and focus on increasing energy by controlling only the angle of the COM. In practice, the torques required to keep the 5-link pendulum of constant length were too great, and exhaustive search fails. We find that allowing the length to very acts like a spring, and can relieve tension in the system. When COM angular velocity is high, and inertia is greatest, expanding the lengths helps minimize torques and keeps the system within torque bounds.

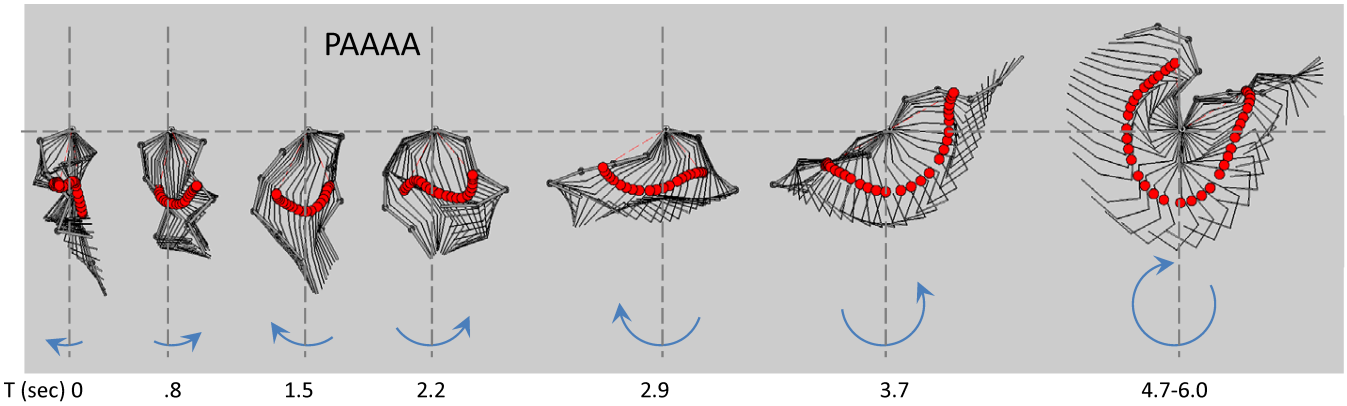

Fig. 6. PAAAA

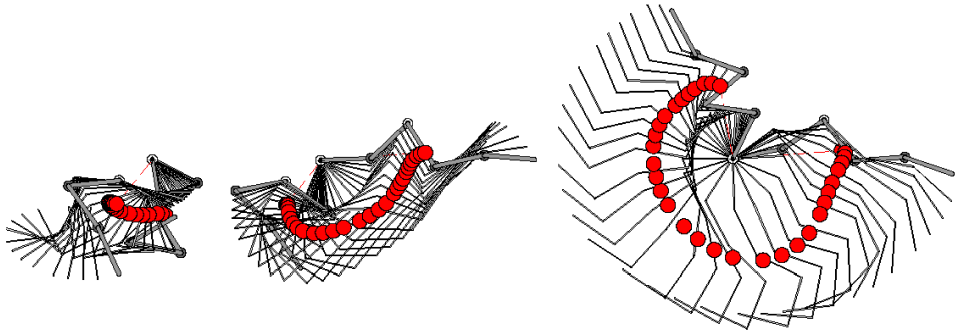

Fig. 7. PAPAA

C. Applications to locomotion

Many legged robots can be modeled by rigid body dynamics. Our lab has been working with the LittleDog robot, a position controlled quadruped robot developed by Boston Dynamics, Inc for the DARPA Learning Locomotion program. The task is to develop a controller for LittleDog to traverse extremely rough terrain. The robot has point feet, and we are currently working on modeling the robot in the sagittal plane as a 5 link PAAAA system. We have had some success generating open loop trajectories which allow the robot to rear-up on its hind legs in an attempt to place its front legs on very high obstacles [30]. Figure 5 shows the robot rearing back, as well as a corresponding 5-link model and 2-DOF template. We are working to utilize the techniques developed in this paper to generate a bounding gait that can be stabilized by the task space linearization. Because the actual robot has 12 actuated DOF, we can utilize the null space for foot placement. Related applications are plentiful, and include stabilizing foot roll in humanoid robots.

VIII. Conclusions

In this work we demonstrated a framework for planning in underactuated systems by utilizing task space control and partial feedback linearization. The power of this approach was demonstrated by application of a feedback controller in the Acrobot (PA), which was able to closely track an arbitrary trajectory of its end-effector. Feedback control can fail due to constraints on the system, including torque limitations, physical obstacles in the environment, joint limits, etc. We proposed a model based motion planner to graphically search for trajectories from start to goal without violating constraints. This planner takes advantage of a reduced action space imposed by the task space control. This planner was successful in producing swingup behavior in 5-link models with anywhere from 0 to 4 passive joints. This toy problem lays the foundation for real-world applications which we are working on, including stabilizing dynamics gaits of a point-food quadruped robot, and roll in a humanoid.

Acknowledgment

This work was supported by the DARPA Learning Locomotion program (AFRL contract # FA8650-05-C-7262)

References

[1] R. M. Murray and J. Hauser, "A case study in approximate linearization: The acrobot example," April 1991.
[2] M. W. Spong, "Underactuated mechanical systems," in *Control Problems in Robotics and Automation*, ser. Lecture notes in control and information sciences 230, B. Siciliano and K. P. Valavanis, Eds. Springer-Verlag, 1997.
[3] A. Goswami, B. Espiau, and A. Keramane, "Limit cycles and their stability in a passive bipedal gait." IEEE International Conference on Robotics and Automation (ICRA), 1996, pp. 246 – 251.
[4] Honda Motor Co., "http://world.honda.com/ASIMO/," July 2004.
[5] Kawada Industries, "http://www.kawada.co.jp/global/ams/hrp_2.html," February 2008.
[6] S. H. Collins, A. Ruina, R. Tedrake, and M. Wisse, "Efficient bipedal robots based on passive-dynamic walkers," *Science*, vol. 307, pp. 1082–1085, February 18 2005.
[7] M. W. Spong, "Partial feedback linearization of underactuated mechanical systems," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, vol. 1, September 1994, pp. 314–321.
[8] Liegeois and A., "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-7, no. 12, pp. 868 – 871, December 1977.

[9] O. Khatib, L. Sentis, J. Park, and J. Warren, "Whole-body dynamic behavior and control of human-like robots," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 29–43, 2004.

[10] L. Sentis and O. Khatib, "Control of free-floating humanoid robots through task prioritization," *ICRA*, 2005.

[11] J.-H. Shin and J.-J. Lee, "Dynamic control of underactuated manipulators with free-swinging passive joints in cartesian space," *ICRA*, 1997.

[12] M. Bergerman, "Dynamics and control of underactuated manipulators," Ph.D. dissertation, Carnegie Mellon University, December 1996.

[13] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[14] R. Diankov, N. Ratliff, D. Ferguson, S. Srinivasa, and J. Kuffner, "Bispace planning: Concurrent multi-space exploration," in *Proceedings of Robotics: Science and Systems IV*, June 2008.

[15] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek, "Zero dynamics of underactuated planar biped walkers," *IFAC-2002, Barcelona, Spain*, pp. 1–6, Jul 2002.

[16] I. Poulakakis and J. Grizzle, "Monopedal running control: Slip embedding and virtual constraint controllers," *IROS*, 2007.

[17] R. Full and D. Koditschek, "Templates and anchors: neuromechanical hypotheses of legged locomotion on land," *Journal of Experimental Biology*, vol. 202, no. 23, pp. 3325–3332, 1999.

[18] R. Altendorfer, U. Saranli, H. Komsoglu, D. Koditschek, H. B. Brown, M. Buehler, N. Moore, D. McMordie, and R. Full, "Evidence for spring loaded inverted pendulum running in a hexapod robot," in *Proceedings of the 7th International Symposium on Experimental Robotics (ISER)*, 2000.

[19] J. Nakanishi, T. Fukuda, and D. Koditschek, "A brachiating robot controller," *Robotics and Automation, IEEE Transactions on*, vol. 16, no. 2, pp. 109–123, Apr 2000.

[20] U. Saranli and D. Koditschek, "Template based control of hexapedal running," in *Proceedings of the IEEE International Conference On Robotics and Automation*, vol. 1, September 2003, pp. 1374–1379.

[21] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*. Prentice Hall, October 1990.

[22] A. Shkolnik and R. Tedrake, "Inverse kinematics for a point-foot quadruped robot with dynamic redundancy resolution," in *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, April 2007.

[23] M. Spong, "The swingup control problem for the acrobot," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 49–55, February 1995.

[24] S. M. LaValle, J. J. Kuffner, and Jr., "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[25] M. W. Spong, "Swing up control of the acrobot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1994, pp. 2356–2361.

[26] I. Fantoni and R. Lozano, *Non-linear Control for Underactuated Mechanical Systems*, ser. Communications and Control Engineering Series. Springer-Verlag, 2002.

[27] K. M. Lynch, N. Shiroma, H. Arai, and K. Tanie, "Collision-free trajectory planning for a 3-dof robot with a passive joint," *International Journal of Robotics Research*, vol. 19, no. 12, pp. 1171–1184, 2000.

[28] R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[29] M. V. Weghe, D. Ferguson, and S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," in *IEEE-RAS International Conference on Humanoid Robots*, November 2007.

[30] K. Byl, A. Shkolnik, S. Prentice, N. Roy, and R. Tedrake, "Reliable dynamic motions for a stiff quadruped," in *Proceedings of the 11th International Symposium on Experimental Robotics (ISER)*, 2008.