

# Hybrid Control for Navigation of Shape-Accelerated Underactuated Balancing Systems

Umashankar Nagarajan, George Kantor and Ralph Hollis

**Abstract**—This paper presents a hybrid control strategy for navigation of shape-accelerated underactuated balancing systems with dynamic constraints. It extends the concept of sequential composition to perform discrete state-based switching between asymptotically convergent control policies to produce a globally asymptotically convergent feedback policy. The individual control policies consists of an external trajectory planner, a shape trajectory planner, an external trajectory tracking controller and a balancing controller. The paper also presents an integrated planning and control procedure, wherein standard graph-search algorithms are used to plan for the sequence of control policies that will help the system achieve a navigation goal. Simulation results of the 3D ballbot system navigating an environment with static obstacles to reach the goal position are also presented.

## I. INTRODUCTION

Underactuated mechanical systems are systems with fewer control inputs than the degrees of freedom [1]. In robotics, balancing (dynamically stable) mobile robots form a special class of underactuated systems. They include wheeled robots like Segway [2], ballbots [3] and legged robots. Balancing robots will play a vital role in realizing the dream of placing robot workers in human environments by virtue of their small footprints and high centers of gravity. Among wheeled balancing systems, ballbots have the advantage of omnidirectional motion that make them more suitable for operation in constrained spaces. These omnidirectional balancing systems, referred to as *shape-accelerated underactuated balancing systems* [4], are of interest here. The interesting and troubling factor in control and planning for such underactuated systems is the constraint on their dynamics by virtue of underactuation. These constraints are second-order nonholonomic [5] constraints, *i.e.*, non-integrable acceleration/dynamic constraints, which restrict the family of trajectories the configurations can follow. Underactuated balancing systems that are destabilized by gravitational forces have to maintain balance, which makes it difficult to track desired configuration trajectories.

Traditionally, motion planning and control for mobile robots have been decoupled. Robot motion planning procedures, generally, account for obstacles in the environment and workspace constraints but do not account for the system dynamics and the constraints on them. They also do not have any knowledge of the details of the controller that is used to achieve these motion plans. On the other hand, the

controller does not have any knowledge of the workspace constraints and obstacles in the environment. Though it is possible to make dynamic, underactuated balancing systems navigate environments using these decoupled procedures, they are often sub-optimal and result in jerky motions, where the controller is fighting with the dynamics of the system to move it around. Moreover, when disturbed, these procedures often either result in collision with the obstacles or drive the system unstable. In order to achieve robust, smooth and collision-free motions, an integrated planning and control procedure is necessary, where both the planner and the controller understand the system dynamics and also understand each other's details.

### A. Related Work

In the last decade, there has been a large body of work on hybrid control techniques that will avoid decoupling between planners and controllers. *Sequential composition*, introduced in [6], is a controller composition technique that connects a palette of controllers and automatically switches between them to generate a globally convergent feedback policy. This technique was successfully applied to a variety of systems [7], [8], [9], [10]. In [11], *sequential composition* was extended to produce an integrated planning and control procedure to achieve global navigation objectives for convex-bodied wheeled mobile robots navigating amongst static obstacles.

### B. Contributions of the Paper

This paper presents a hybrid control framework for navigation of shape-accelerated underactuated balancing systems. *Sequential composition* [6] is used to discretely switch between individual, asymptotically convergent control policies to produce a globally, asymptotically convergent feedback control policy that will achieve the overall navigation goal. The individual control policies are a combination of local planners and controllers, as will be described in Sec. IV. The local planner plans shape trajectories that account for the dynamic constraints of the system in order to effectively track desired external configuration trajectories [12], [4]. Graph-search algorithms like  $A^*$  are used as a high-level planner to plan for the sequence of control policies that will help the system achieve the navigation goal. This paper primarily focuses on the ballbot [3] and simulation results on a 3D model are presented in Sec. V.

U. Nagarajan, G. Kantor and R. Hollis are with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. umashankar@cmu.edu, kantor@ri.cmu.edu, rhollis@cs.cmu.edu

## II. UNDERACTUATED MECHANICAL SYSTEMS

The forced Euler-Lagrange equations of motion for a mechanical system are:

$$\frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q} = F(q)\tau, \quad (1)$$

where,  $q \in \mathbb{R}^n$  is the configuration vector,  $\mathcal{L}(q, \dot{q}) = K(q, \dot{q}) - V(q)$  is the Lagrangian with kinetic energy  $K$  and potential energy  $V$ ,  $\tau \in \mathbb{R}^m$  is the control input and  $F(q)$  is the force matrix.

A mechanical system satisfying Eq. 1 is said to be an *underactuated system* [1] if  $m < n$ , i.e., there are fewer independent control inputs than configuration variables. Eq. 1 for an underactuated system can be written in matrix form as follows:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = F(q)\tau, \quad (2)$$

where,  $M(q)$  is the inertia matrix,  $C(q, \dot{q})$  is the matrix of Coriolis and centrifugal terms and  $G(q)$  is the vector of gravitational forces.

The configuration variables that appear in the inertia matrix are called *shape variables* ( $q_s$ ), whereas, the configuration variables that do not appear in the inertia matrix are called *external variables* ( $q_x$ ), i.e.,  $\partial M(q)/\partial q_x = 0$ . Eq. 2 can be re-written as:

$$\begin{bmatrix} M_{xx}(q_s) & M_{xs}(q_s) \\ M_{sx}(q_s) & M_{ss}(q_s) \end{bmatrix} \begin{bmatrix} \ddot{q}_x \\ \ddot{q}_s \end{bmatrix} + \begin{bmatrix} h_x(q, \dot{q}) \\ h_s(q, \dot{q}) \end{bmatrix} = \begin{bmatrix} F_x(q) \\ F_s(q) \end{bmatrix} \tau, \quad (3)$$

where,  $h(q, \dot{q}) = [h_x(q, \dot{q}), h_s(q, \dot{q})]^T$  is:

$$\begin{bmatrix} h_x(q, \dot{q}) \\ h_s(q, \dot{q}) \end{bmatrix} = \begin{bmatrix} C_{xx}(q, \dot{q}) & C_{xs}(q, \dot{q}) \\ C_{sx}(q, \dot{q}) & C_{ss}(q, \dot{q}) \end{bmatrix} \begin{bmatrix} \dot{q}_x \\ \dot{q}_s \end{bmatrix} + \begin{bmatrix} G_x(q) \\ G_s(q) \end{bmatrix}. \quad (4)$$

The underactuated systems can be classified based on whether the shape variables  $q_s$  are fully actuated, partially actuated or unactuated and based on the presence or lack of input couplings in the force matrix  $F(q)$  [13].

### A. Shape-Accelerated Underactuated Balancing Systems

This paper focuses on shape-accelerated underactuated balancing systems [4], which form a special class of underactuated systems with the following properties: (i) the shape variables are unactuated and there is no input coupling, say,  $F(q) = [I_m, 0]^T$ ; (ii) there are equal number of actuated and unactuated variables, i.e.,  $n = 2m$ ; (iii)  $h(q, \dot{q})$  is independent of both  $q_x$  and  $\dot{q}_x$ . These properties result in equations of motion that are symmetric with respect to the external variables and their first derivatives ( $q_x, \dot{q}_x$ ). More properties for such systems can be found in [4].

The shape-accelerated underactuated balancing systems have equations of motion of the form:

$$\begin{bmatrix} M_{xx} & M_{xs}(q_s) \\ M_{sx}(q_s) & M_{ss}(q_s) \end{bmatrix} \begin{bmatrix} \ddot{q}_x \\ \ddot{q}_s \end{bmatrix} + \begin{bmatrix} h_x(q_s, \dot{q}_s) \\ h_s(q_s, \dot{q}_s) \end{bmatrix} = \begin{bmatrix} \tau \\ 0 \end{bmatrix}, \quad (5)$$

where,

$$\begin{bmatrix} h_x(q_s, \dot{q}_s) \\ h_s(q_s, \dot{q}_s) \end{bmatrix} = \begin{bmatrix} 0 & C_{xs}(q_s, \dot{q}_s) \\ 0 & C_{ss}(q_s, \dot{q}_s) \end{bmatrix} \begin{bmatrix} \dot{q}_x \\ \dot{q}_s \end{bmatrix} + \begin{bmatrix} 0 \\ G_s(q_s) \end{bmatrix}. \quad (6)$$

We can see from Eq. 5 that the equations of motion of these systems are functions of  $(\ddot{q}_x, q_s, \dot{q}_s, \ddot{q}_s)$  and are independent of  $q_x$  and  $\dot{q}_x$ . Some examples of shape-accelerated underactuated balancing systems are planar and 3D cart-pole system with unactuated lean angles, planar wheeled inverted pendulum (e.g., Segway [2] in a plane) and 3D omnidirectional wheeled inverted pendulum (e.g., the ballbot [14], [12]).

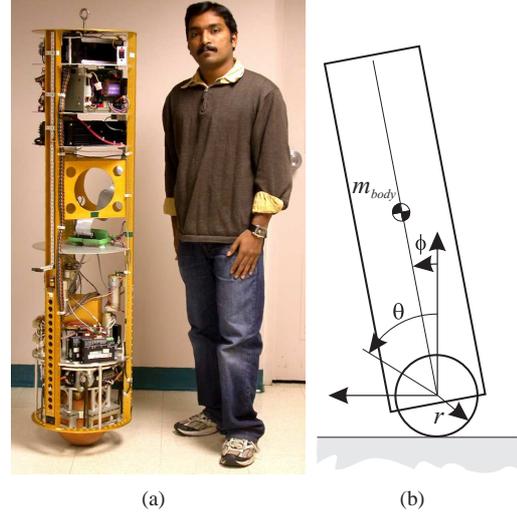


Fig. 1. (a) The ballbot balancing, (b) Planar ballbot model with ball and body configurations shown.

### B. The Ballbot

The ballbot (Fig. 1(a)) is a 3D omni-directional wheeled inverted pendulum robot. It can be modeled as a rigid cylinder on top of a rigid sphere with the following assumptions: (i) there is no slip between the ball and the floor, and (ii) there is no yaw/spinning motion for both the ball and the body, i.e., they have 2-DOF each. For the 3D ballbot model, the ball angles ( $\theta_x, \theta_y$ ), which are algebraically related to the ball position  $(x_w, y_w)$ , constitute the external variables, whereas the body angles ( $\phi_x, \phi_y$ ) constitute the shape variables.

## III. SHAPE TRAJECTORY PLANNING AND CONTROL

The second set of  $m$  equations of motion associated with the unactuated shape variables in Eq. 5 given by

$$M_{sx}(q_s)\ddot{q}_x + M_{ss}(q_s)\ddot{q}_s + h_s(q_s, \dot{q}_s) = 0, \quad (7)$$

which can be written as:

$$\Theta(q_s, \dot{q}_s, \ddot{q}_s, \ddot{q}_x) = 0. \quad (8)$$

Eq. 7 and Eq. 8 are called *second-order nonholonomic constraints*, or *dynamic constraints* because there exists no function  $\Psi$  such that  $\dot{\Psi} = \Theta(q_s, \dot{q}_s, \ddot{q}_s, \ddot{q}_x)$ . The dynamic constraint equations are not even partially integrable, i.e., they cannot be converted into first-order nonholonomic constraints. This is ensured by the fact that the gravitational vector  $G(q_s)$  is not a constant and the inertia matrix  $M(q_s)$  is dependent on the unactuated shape variables  $q_s$ . For a detailed discussion of these conditions, refer to [15].

### A. Optimal Shape Trajectory Planner

In underactuated balancing systems, we are often interested in tracking desired trajectories for the external variables without losing balance. Shape-accelerated underactuated balancing systems in Sec. II-A have constraints on the acceleration of these external variables w.r.t. the shape variables' position, velocity and acceleration as given below:

$$\begin{aligned}\ddot{q}_x &= -M_{sx}(q_s)^{-1}(M_{ss}(q_s)\ddot{q}_s + h_s(q_s, \dot{q}_s)) \\ &= \Gamma(q_s, \dot{q}_s, \ddot{q}_s).\end{aligned}\quad (9)$$

It is to be noted that Eq. 9 holds only if  $M_{sx}(q_s)^{-1}$  exists, which it does in the neighborhood of the origin, a property of shape-accelerated underactuated balancing systems [4].

Using  $a = (q_s, \dot{q}_s, \ddot{q}_s)$  and  $b = \ddot{q}_x$ , Eq. 8 can be written as  $\Theta(a, b) = 0$ . Taking the Jacobian w.r.t.  $b$  at  $(a, b) = (0, 0)$  yields

$$\left. \frac{\partial \Theta(a, b)}{\partial b} \right|_{(a, b) = (0, 0)} = M_{sx}(q_s) \Big|_{q_s=0} \quad (10)$$

From properties of shape-accelerated underactuated balancing systems [4], the Jacobian in Eq. 10 exists and is invertible. Hence, by the implicit function theorem, there exists a map  $\Gamma: a \rightarrow b$  such that  $\Theta(a, \Gamma(a)) = 0$ , which can be seen from Eq. 9. Again from the implicit function theorem, the map  $\Gamma$  is not invertible since the Jacobian  $\partial \Theta(a, b) / \partial a$  at  $(a, b) = (0, 0)$  exists but is not invertible.

In order to track a non-constant, time-varying  $\ddot{q}_x^d(t)$ , there is no function that maps  $\ddot{q}_x^d(t)$  to  $(q_s^d(t), \dot{q}_s^d(t), \ddot{q}_s^d(t))$  such that the dynamic constraints in Eq. 8 are satisfied. So, it is desirable to plan shape configuration trajectories  $(q_s^p(t), \dot{q}_s^p(t), \ddot{q}_s^p(t))$ , which when tracked will result in approximate tracking of  $\ddot{q}_x^d(t)$ . Here, a linear map  $K_{q_x}: \ddot{q}_x^d \rightarrow q_s^p$  is proposed such that

$$K_{q_x} = \underset{K}{\operatorname{argmin}} \|\Gamma(K\ddot{q}_x^d(t), K\dot{q}_x^d(t), K\ddot{q}_x^d(t)) - \ddot{q}_x^d(t)\|_2^2. \quad (11)$$

Here, the planned shape trajectories  $(q_s^p(t), \dot{q}_s^p(t), \ddot{q}_s^p(t)) = (K\ddot{q}_x^d(t), K\dot{q}_x^d(t), K\ddot{q}_x^d(t))$ .

The shape trajectory planning procedure is now an optimization problem with the objective of finding the elements of  $K_{q_x}$  such that the  $\mathbb{L}_2$ -norm of the error in tracking  $\ddot{q}_x^d(t)$  is minimized. It is to be noted that the parameter space is  $m^2$ -dimensional and any optimization algorithm that solves nonlinear least-squares problem can be used.

A good initial guess for  $K_{q_x}$  is obtained from the dynamic constraint given by Eq. 9 with  $(\dot{q}_s, \ddot{q}_s) = (0, 0)$ . In this case, Eq. 9 reduces to

$$\ddot{q}_x = -M_{sx}(q_s)^{-1}G_s(q_s) \quad (12)$$

in the neighborhood of the origin. Jacobian linearization of Eq. 12 w.r.t.  $q_s$  at  $q_s = 0$  gives

$$\begin{aligned}\ddot{q}_x &= \left[ -\frac{\partial(M_{sx}(q_s)^{-1}G_s(q_s))}{\partial q_s} \Big|_{q_s=0} \right] q_s \\ &= \hat{K}_{q_x} q_s,\end{aligned}\quad (13)$$

and  $\hat{K}_{q_x} = \hat{K}_{q_s}^{-1}$ . The inverse,  $\hat{K}_{q_s}^{-1}$ , exists in the neighborhood of the origin due to the properties of shape-accelerated

underactuated systems [4]. For a desired constant acceleration trajectory,  $K_{q_x} = \hat{K}_{q_x}$  ensures optimality, but for any general  $\ddot{q}_x^d(t)$ ,  $K_{q_x} = \hat{K}_{q_x}$  provides a good initial guess for the optimization process. It is to be noted that the optimality here is in tracking error and not in time or path length.

In design of the optimal shape trajectory planner described above, the objective has been approximate tracking of  $\ddot{q}_x^d(t)$ , but in reality, it would be desirable to track some  $q_x^d(t)$ . Under the current procedure, this is possible only if the initial conditions for the external variables are met. In order to approximately track a desired external configuration trajectory  $q_x^d(t)$  using the optimal shape trajectory planner described above, the follow conditions must hold: (i)  $q_x^d(t)$  must be of at least of class  $C^2$ , i.e.,  $\dot{q}_x^d(t)$  and  $\ddot{q}_x^d(t)$  exist and are continuous; (ii) initial conditions for the external variables are met, i.e.,  $q_x^p(0) = q_x^d(0)$  and  $\dot{q}_x^p(0) = \dot{q}_x^d(0)$ . It is to be noted that  $q_x^d(t)$  is preferred to be of class  $C^4$  so that the first four derivatives exist and are continuous.

### B. Balancing and Trajectory Tracking Control

The shape trajectory planning procedure, described in Sec. III-A, assumes that there exists a balancing controller, which has good shape trajectory tracking performance. Similar to [14], [12], this work uses a linear PID controller (Eq. 14) as the balancing controller.

$$\begin{aligned}\tau(t) &= \beta_p(q_s(t) - q_s^d(t)) + \beta_i \int (q_s(t) - q_s^d(t)) \\ &\quad + \beta_d(\dot{q}_s(t) - \dot{q}_s^d(t)),\end{aligned}\quad (14)$$

where,  $\beta_p, \beta_i, \beta_d$  are the proportional, integral and derivative gains respectively.

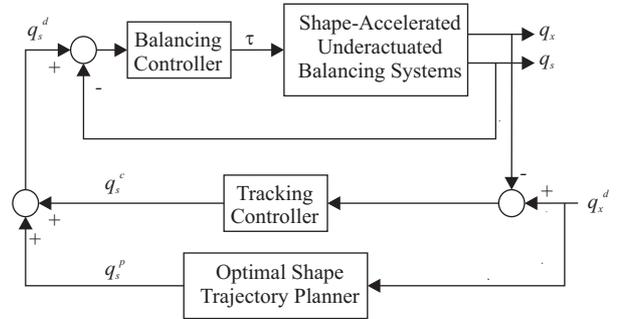


Fig. 2. Control Architecture.

The combination of the balancing controller and the optimal shape trajectory planner provides good approximate tracking of the desired external configuration trajectories under idealized conditions. However, an external trajectory tracking controller is required to achieve tracking in more realistic conditions such as different initial conditions, unmodeled dynamics and disturbances. In this work, a linear PD controller (Eq. 15) is used as the external trajectory tracking controller.

$$\begin{aligned}q_s^d(t) &= q_s^p(t) + q_s^c(t), \\ q_s^c(t) &= \gamma_p(q_x(t) - q_x^d(t)) + \gamma_d(\dot{q}_x(t) - \dot{q}_x^d(t)),\end{aligned}\quad (15)$$

where,  $\gamma_p$ ,  $\gamma_d$  are the proportional and derivative gains respectively.

The resulting control architecture (Fig. 2) has been shown to work well on the experimental ballbot setup [14]. The balancing controller tracks the desired shape trajectories,  $q_s^d(t)$ , which are a sum of planned,  $q_s^p(t)$  and compensation,  $q_s^c(t)$ , shape trajectories. The planned shape trajectories are given by the optimal shape trajectory planner, whereas the compensation shape trajectories are provided by the tracking controller, which tries to compensate for the deviation of external trajectories from the desired ones.

#### IV. HYBRID CONTROL FOR NAVIGATION

This section presents a hybrid control formalism based on *sequential composition* [6] that will enable shape-accelerated underactuated balancing systems, like the ballbot, to navigate an environment with obstacles.

##### A. Sequential Composition

Given a set of control policies  $\mathbb{U} = \{\Phi_1, \dots, \Phi_n\}$ , each with a domain,  $\mathbb{D}(\Phi_i)$  and goal set,  $\mathbb{G}(\Phi_i)$ . It is presumed that the control policy  $\Phi_i$  will take any state in domain  $\mathbb{D}(\Phi_i)$  to  $\mathbb{G}(\Phi_i)$  without leaving  $\mathbb{D}(\Phi_i)$ . It is said that the control policy  $\Phi_1$  *prepares*  $\Phi_2$ , denoted by  $\Phi_1 \succeq \Phi_2$ , if the goal of the first lies inside the domain of the second, *i.e.*,  $\mathbb{G}(\Phi_1) \subset \mathbb{D}(\Phi_2)$ .

A directed graph can be generated for an appropriate set of control policies  $\mathbb{U}$ . If the start state  $S$  belongs to the domain of at least one control policy, *i.e.*,  $\exists i \in [1, n]$ , s.t.  $S \in \mathbb{D}(\Phi_i)$  and the overall goal  $G$  belongs to the goal set of at least one control policy, *i.e.*,  $\exists i \in [1, n]$ , s.t.  $G \in \mathbb{G}(\Phi_i)$ , then the navigation problem becomes a graph search problem, where the optimal sequence of control policies to reach the overall goal can be found.

##### B. Asymptotically Convergent Domains

In the original *sequential composition* approach [6], the policy domains defined are invariant, *i.e.*, under the action of the policy  $\Phi_i$ , the state trajectory starting inside  $\mathbb{D}(\Phi_i)$  remains within the domain until it reaches  $\mathbb{G}(\Phi_i)$ . The control policies that will be defined in the later sections of the paper do have invariant domains (or) domains of attraction that can be determined using Lyapunov-based methods. But they are generally quite complicated and in our attempt to do navigation, it would often be preferable to have smaller subsets of these domains. So, policy domains can be defined with simple geometric shapes that are not necessarily invariant but have asymptotic convergence properties, *i.e.*, under the action of the policy  $\Phi_i$ , the state trajectory starting inside  $\mathbb{D}(\Phi_i)$  remains within domain  $\mathbb{D}'(\Phi_i)$  such that  $\mathbb{D}(\Phi_i) \subseteq \mathbb{D}'(\Phi_i)$  until it reaches  $\mathbb{G}(\Phi_i)$ . These geometric shapes make it simple to determine whether a given state is within a particular policy domain or not.

These domains are generally defined over the state space of the system, whereas in this paper they are defined only over a subset of the state space given by the external variables, *i.e.*,  $(q_x, \dot{q}_x)$ . This reduction in dimensionality for the domains is possible due to the strong coupling between

the shape and external variables. Any changes in shape configurations cause changes in the external configurations and in order to track any desired external configuration trajectory, the shape configurations have to follow a particular trajectory that is stable. The balancing controller (Sec. III-B), which makes this tracking possible, is assumed to have a large enough domain of attraction in the shape state space.

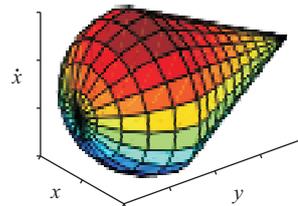


Fig. 3. 3D projection of the 4D ice-cream hypercone.

For the ballbot example, ice-cream shaped hypercones defined in  $(x_w, y_w, \dot{x}_w, \dot{y}_w)$ -space (4D) are used as policy domains. A 3D projection of the ice-cream hypercone is shown in Fig. 3. The ice-cream hypercones are formed by fusing a semi-hyperellipsoid and a hypercone with ellipsoidal cross-section. These ice-cream hypercones are parametrized by the lengths of the semi-principal axes of the hyperellipsoid and the height of the hypercone.

##### C. Palette of Control Policies

This section presents the available control policies to perform hybrid control. There are two control policies with different objectives:

- (i) *Stopping Control Policy*, where the desired goal configuration is to come to rest at the tip of the ice-cream hypercone; and
- (ii) *Moving/Flow-Through Control Policy*, where the desired goal configuration is to continue moving with a desired velocity through the tip of the ice-cream hypercone.

The dynamics of the shape-accelerated underactuated balancing system is invariant to both position and velocity of the external variables and hence these policy domains can be placed anywhere with any orientation in the 4D external variable state space. Moreover, the policy domains, *i.e.*, ice-cream hypercones, are free to be rotated only on the XY-plane, so that the stopping control policies always have zero velocity goal configurations, whereas, the flow-through control policies have the same desired exit speed.

In this work, a control policy consists of: (i) an external trajectory planner, which plans a trajectory from each point in the 4D policy domain to its goal state; (ii) an optimal shape trajectory planner, which plans shape trajectories that ensure optimal tracking of the desired external trajectories; (iii) a tracking controller, which ensures better tracking of the desired external trajectories; and (iv) a balancing controller, which ensures accurate tracking of desired shape trajectories.

Bezier curves are used to plan external trajectories for motion inside the ice-cream hypercone domains. The parametric Bezier curve  $(x(t), y(t))$  can be written as:

$$x(t) = \sum_{i=0}^n x_i b_{i,n}(t), \quad y(t) = \sum_{i=0}^n y_i b_{i,n}(t), \quad (16)$$

where,  $(x_i, y_i)$  are the  $n+1$  control points and the Bernstein polynomial  $b_{i,n}(t)$  is given by:

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{(n-i)} \quad (0 \leq t \leq 1). \quad (17)$$

The  $n+1$  control points are chosen such that the boundary conditions are satisfied, *i.e.*, the initial condition and the desired final goal configuration. When the boundary conditions include conditions on the derivatives, the  $n+1$  control points and, in turn, the Bezier curve is a function of the time duration of the curve. The optimal time duration that minimizes the summed area under the curve of the position, velocity, acceleration and jerk trajectories is used here. A variety of other objective functions can also be optimized to obtain the time duration.

Shape trajectories are planned using the optimal shape trajectory planner described in Sec. III-A for the Bezier curves. As described in Sec. III-B, the external trajectory tracking controller, in combination with the shape trajectory planner, is used to provide desired shape trajectories that the balancing controller will track.

#### D. Prepares Graph

Given an environment and a collection of policy domains distributed in the 4D space, a *prepares* graph can be generated, where each node corresponds to a policy domain and each directed link represents the *prepares* relationship. The task of navigating from a given start state to a desired overall goal state can be performed by the described hybrid control scheme provided the following conditions are satisfied: (i) there is at least one domain that contains the start state; (ii) there is at least one domain whose goal configuration matches the goal state; and (iii) there is a path between these two domains in the *prepares* graph.

Optimal graph search algorithms like  $A^*$  can be used to obtain a sequence of control policies that are optimal w.r.t. some cost function. A variety of heuristic functions can be used to even optimize for time or length of the path. Existing dynamic replanning graph search algorithms like  $D^*$  [16] can be used for automatically replanning the sequence of control policies when the system is disturbed from its current path. An integrated planning and control procedure has been developed, where the high-level planner is planning a sequence of control policies and dynamically updating the sequence based on the system's current state and overall desired goal.

#### E. Switching Control

Given a path *i.e.*, a sequence of control policies to reach the overall goal, a hybrid control strategy is used that sequentially composes asymptotically convergent control policies

with discrete state-based switching between them. To initiate the switching behavior, it is necessary to determine whether the state trajectory has entered a domain or not. Simple geometric domain shapes make it easier to determine whether the external state lies inside the geometric shape by use of analytical equations.

The Bezier curves and the corresponding shape trajectory plans are generated online based on the entering external state values. These trajectories are tracked until the state trajectory enters the next policy domain along the path towards its overall goal. This switching behavior continues until the state trajectory reaches the final policy domain that contains the overall goal.

### V. SIMULATION RESULTS

In this section, we present simulation results of the hybrid control procedure, described in Sec. IV, on the 3D ballbot model. Here, the navigation goal is to move from (0 m, 0 m) to (10 m, 10 m) around two obstacles in the environment shown in Fig. 4(a).

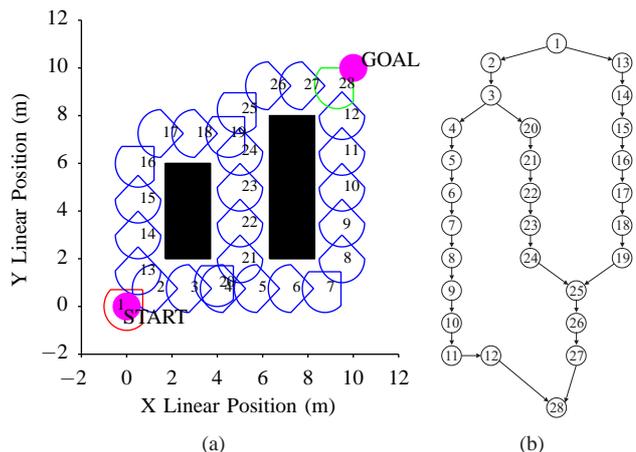


Fig. 4. (a) Environment with start and goal configurations and also the available policy domains; (b) *Prepares* Graph.

For example, suppose that the ice-cream hypercone policy domains are placed in the environment as shown in Fig. 4(a). Note that the policy domains are placed in a 4-dimensional subset of the state space and are projected onto the 2-dimensional workspace for ease of visualization. The resulting *prepares* graph is shown in Fig. 4(b). In the case presented, none of the policy domains collide with the obstacles. The policy domains that intersect the obstacles are removed before generating the *prepares* graph. Moreover, the collision check is done with the outer ice-cream hypercone domains  $\mathbb{D}'(\Phi_i)$  for each control policy  $\Phi_i$  (Fig. 5).

$A^*$ , with Euclidean distance between the goal configurations of domains as the distance metric, was used to determine the optimal path as shown in Fig. 5. The resulting linear position and body angle trajectories are shown in Figs. 6 and 7 respectively.

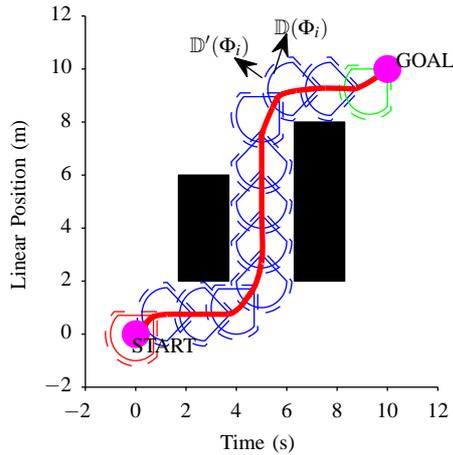


Fig. 5. Navigation among static obstacles using Hybrid Control Scheme.

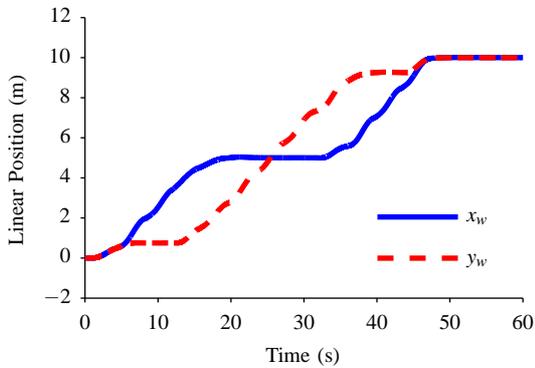


Fig. 6. Linear Position Trajectories.

## VI. CONCLUSIONS

A hybrid control policy for navigation of shape-accelerated underactuated balancing systems was presented. The *sequential composition* [6] technique was extended for the underactuated balancing systems with the policy domains established as geometric shapes in only a subset of the state space, *i.e.*, state space w.r.t. external variables. The concept of designing asymptotically convergent control policies was introduced, with each policy consisting of a combination of planners and controllers. An integrated planning and control procedure was presented that consists of a high-level planner that plans for the sequence of control policies to be followed to achieve a navigation goal. Successful simulation results for a 3D ballbot model navigating an environment with static obstacles were presented.

## VII. FUTURE WORK

As part of the future work, experimental testing of the proposed hybrid control framework has to be done. A variety of other geometric shapes for control policy domains must be analyzed. The use of dynamic replanning algorithms, like  $D^*$ , for replanning control policy sequences when the system is subjected to large disturbances and also for planning in environments with dynamic obstacles can be explored.

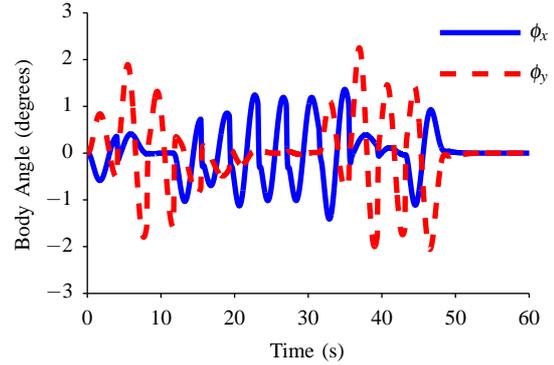


Fig. 7. Body Angle Trajectories.

## VIII. ACKNOWLEDGMENTS

This work was supported in part by NSF grants IIS-0308067 and IIS-0535183.

## REFERENCES

- [1] M. W. Spong, "The control of underactuated mechanical systems," in *First International Conference on Mechatronics*, Mexico City, 1994.
- [2] H. G. Nguyen, J. Morrell, K. Mullens, A. Burmeister, S. Miles, N. Farrington, K. Thomas, and D. Gage, "Segway robotic mobility platform," in *SPIE Proc. 5609: Mobile Robots XVII*, Philadelphia, PA, October 2004.
- [3] R. Hollis, "Ballbots," *Scientific American*, pp. 72–78, October 2006.
- [4] U. Nagarajan, "Dynamic constraint-based optimal shape trajectory planner for shape-accelerated underactuated balancing systems," in *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, 2010.
- [5] J. R. Ray, "Nonholonomic constraints," *American Journal of Physics*, vol. 34, pp. 406–408, 1966.
- [6] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dexterous robot behaviors," *The International Journal of Robotics Research*, vol. 18, no. 6, 1999.
- [7] E. Klavins and D. E. Koditschek, "A formalism for the composition of concurrent robot behaviors," in *Proc. IEEE Int'l. Conf. on Robotics and Automation*, vol. 4, 2000, pp. 3395–3402.
- [8] A. E. Quaid and A. A. Rizzi, "Robust and efficient motion planning for a planar robot using hybrid control," in *Proc. IEEE Int'l. Conf. on Robotics and Automation*, 2000, pp. 4021–4026.
- [9] A. A. Rizzi, J. Gowdy, and R. L. Hollis, "Distributed coordination in modular precision assembly systems," *The International Journal of Robotics Research*, vol. 20, no. 10, 2001.
- [10] G. Kantor and A. A. Rizzi, "Feedback control of underactuated systems via sequential composition: Visually guided control of a unicycle," in *11th International Symposium of Robotics Research*, Siena, Italy, October 2003.
- [11] D. C. Conner, H. Choset, and A. A. Rizzi, "Integrated planning and control for convex-bodied nonholonomic systems using local feedback," in *Proc. Robotics: Science and Systems II*, 2006, pp. 57–64.
- [12] U. Nagarajan, A. Mampetta, G. Kantor, and R. Hollis, "State transition, balancing, station keeping, and yaw control for a dynamically stable single spherical wheel mobile robot," *Proc. IEEE Int'l. Conf. on Robotics and Automation*, May 2009.
- [13] R. Olfati-Saber, "Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles," Ph.D. dissertation, Massachusetts Institute of Technology, February 2001.
- [14] U. Nagarajan, G. Kantor, and R. Hollis, "Trajectory planning and control of an underactuated dynamically stable single spherical wheeled mobile robot," *Proc. IEEE Int'l. Conf. on Robotics and Automation*, May 2009.
- [15] G. Oriolo and Y. Nakamura, "Control of mechanical systems with second-order nonholonomic constraints: Underactuated manipulators," in *In Proc. 30th IEEE Conference on Decision and Control*, vol. 3, 1991, pp. 2398–2403.
- [16] A. Stentz, "The focussed  $d^*$  algorithm for real-time replanning," in *International Joint Conference on Artificial Intelligence*, August 1995.