The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

A New Modular Schema for the Control of Tumbling Robots

Brett Hemes and Nikolaos Papanikolopoulos
Center for Distributed Robtics
University of Minnesota
Minneapolis, MN 55455
{hemes, npapas}@cs.umn.edu

Abstract— **Tumbling is an exciting new area of robotic locomotion that takes advantage of ground-body interactions to achieve rich motions with minimal hardware complexity. The increased mobility of tumbling robots, however, comes at the price of increased control complexity. In this paper, we propose a novel method to handle the issues of tumbling locomotion which takes the problem and separates into locally independent subproblems. Our approach provides an intuitive geometric solution to the tumbling control problem without sacrificing performance. We provide a running example throughout the paper to help solidify the ideas presented.**

I. Introduction

Tumbling robots make up a new and largely unexplored area of robotic locomotion that takes full advantage of the robot's body to aid in locomotion [1]–[3]. This allows tumbling robots to traverse significantly complex terrain while keeping the overall size and complexity of the required hardware small. In this sense we see tumbling as a minimalistic approach to robot locomotion similar to the ideas in [4], [5]. What tumbling robots gain in hardware simplicity, however, they pay for with increased software complexity. The utilization of ground-body interaction, while greatly increasing the potential set of motions, makes control and planning tasks difficult.

The control of tumbling robots in inherently hard due mostly to the nonholonomic nature of tumbling. Additional issues involve high angular velocities during tumbles and the resulting impact along with the fact that many tumbling designs require sliding contacts with the ground, all of which make accurate state estimation and prediction difficult. To the authors' knowledge there are currently no acceptable solutions to the tumbling control problem.

In [6] the authors briefly mention Turbot 2, a Biomorphic two-armed tumbling robot that exhibits phototaxis in generally lit environments. Although the exact details of the control circuit are not mentioned, the authors describe the general Turbot control topology as two chaotic oscillators (one for each arm) weakly coupled by a single analog neuron. In [7] we see a similar two-armed tumbling robot that uses a simple state machine based on relative values from two pairs of photosensors along with motor stall detection. The robot activates one motor at a time based on which of the six possible states it is currently in, resulting in positive phototaxis. The above methods succeed in producing directed tumbling motion, however they require significant expertise to apply, lack the plasticity required by complex tasks, and fail to generalize to other tumbling platforms.

In a previous work [8], we proposed a method for deriving motion primitives for a given tumbling robot by discretizing over the control inputs. The resulting motion of the primitives can then be modeled and stored as a graph allowing the use of traditional search methods for planning. This approach generalizes to other platforms and allows user-defined directional targets, however, it is suboptimal due to discretization of the controls.

In this paper we outline a new approach for control of tumbling locomotion where we break the problem up into separate subcomponents that results in a natural discretization of the overall task. In this manner, we create several subproblems, or modules, that can be solved separately and then combined as a final solution. Specifically, the subproblems involve calculating possible tumbles, predicting future states based on robot/terrain dynamics, and high-level planning using sets of feasible tumbles. We believe that our approach provides an intuitive geometric solution to the tumbling control problem without sacrificing performance.

A. Running Example

For this paper we use the Adelopod [9] (see Figure 1) as a running example to demonstrate our proposed approach. The Adelopod is a small two-armed tumbling robot physically similar to those discussed in [6], [7] with an additional actuated degree of freedom per arm. Electronically the Adelopod is significantly more advanced; it has an onboard 600MHz ARM processor with 128MB of RAM, over 2GB of data storage, and wireless connectivity along with full inertial measurement.

II. Preliminaries

We find it useful to include our definitions of tumbles and tumbling robots before proceeding.

Definition A tumble is a dynamic state of instability during which the robot pivots about an axis formed by two or more contact points with the ground, accelerating downward with gravity and thus behaving as an inverted pendulum.

Definition A tumbling robot is any robot that, by the previous definition, tumbles as its primary means of locomotion with the body playing an active role in achieving such motion.

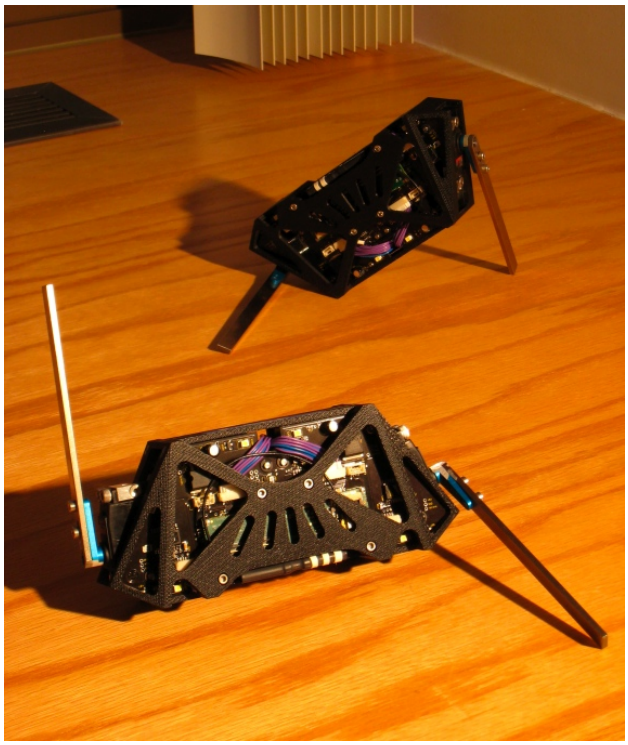


Fig. 1. Two Adelopod tumbling robots shown next to each other.

Additional discussion of the above definitions can be found in [8].

Here we introduce briefly some of the important sets used later in the paper.

- Let $V = \{\boldsymbol{v}_1(\boldsymbol{\phi}), \ldots, \boldsymbol{v}_n(\boldsymbol{\phi})\}$ be an appropriately chosen finite set of vertices that serve as an approximation to the hull of the robot; points in V are, in general, functions of the control inputs. The construction of this set is at the discretion of the implementer, however it should generally include all points of the robot that commonly come in contact with the ground and/or obstacles. For level terrain these points are simply the vertices of the convex hulls of each component (arm, body, etc.) of the robot.
- Let C be the set of vertices from V currently in support. A vertex in support is one that exists in the plane of an arbitrarily chosen face from the convex hull of V, $\mathcal{CH}(V)$.
- Let $S = \{\boldsymbol{s}_1(\boldsymbol{\phi}), \ldots, \boldsymbol{s}_m(\boldsymbol{\phi})\}$ be the set of vertices from V that are the extreme vertices of the support polygon $\mathcal{CH}(C)$.

From the above we have the following relation

$$S \subseteq C \subseteq V. \tag{1}$$

A. Running Example: Choice of V

In this section we describe and motivate our choice of V for the Adelopod. Vertices of the Adelopod can take one of two forms; Each $\boldsymbol{v}_i \in V$ is either a body vertex (static in the robot's frame) or an arm vertex (functions of control inputs).

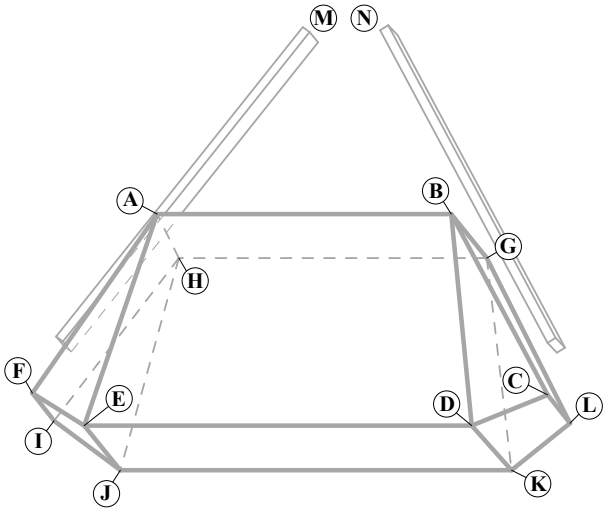


Fig. 2. Selection of points in V for the Adelopod robot.

As mentioned previously, the Adelopod has four actuated degrees of freedom (two per arm). The control vector $\boldsymbol{\phi} \in [-180, 180]^4$ is then

$$\boldsymbol{\phi} = (\phi_1, \phi_2, \phi_3, \phi_4) \tag{2}$$

where ϕ_1 and ϕ_2 are the variables for the left arm and ϕ_3 and ϕ_4 are the variables for the right arm.

The body (static) vertices take the form

$$\boldsymbol{v}_{body} = (v_x, v_y, v_z) \tag{3}$$

where all coordinates are in the robot's body frame. The arm (kinematic) vertices take the form

$$\boldsymbol{v}_{arm} = \begin{bmatrix} v_x^{'} c_i c_j - v_y^{'} c_i s_j - \left(v_z^{'} + d_T\right) s_i + c_i a_i - a_\ell \\ v_x^{'} s_i c_j - v_y^{'} s_i s_j + \left(v_z^{'} + d_T\right) c_i + s_i a_i \\ -v_x^{'} s_j - v_y^{'} c_j \end{bmatrix} \tag{4}$$

where $\boldsymbol{v}_{arm}$ is expressed in the robot's body frame; $(i, j) \in \{(1,2),(3,4)\}$; c_k and s_k represent $\sin(\phi_k)$ and $\cos(\phi_k)$ respectively; $\boldsymbol{v}'$ represents the coordinates of vertex $\boldsymbol{v}_{arm}$ in the arm's coordinate frame; and d and a are Denavit-Hartenberg parameters for the arm.

Our choices of V are depicted graphically in Figure 2 where vertices are labeled with circled letters. In constructing V we have, as suggested previously, included all of the extreme vertices of the convex hull of the body. For the arms, however, we have included only one vertex for each. This counteracts our previous suggestion however this greatly simplifies the math for the purpose of this paper and, in practice, doesn't sacrifice much accuracy on simple terrains.

III. Schema

Tumbling locomotion operates through repeatedly driving the system into states of instability, inducing dynamic motion that brings the system from the current state to another of lower energy, where energy is added through actuation.

The result is a series of tumbles that produce a net displacement of the robot through its environment. We can think of such locomotion as alternating periods of stable and unstable (tumble) states where a majority of the net motion is achieved through tumbles. Therefore we see that, in a tumbling system, instability, or rather planned instability is desirable and even necessary. The fact that tumbling robots leverage periods of instability might seem obvious from the definitions however we find it interesting to note that this is in stark contrast with much of the existing literature on legged systems where stability (static or dynamic) is maintained or even maximized. Although the goals of the two genres are different, the notion of stability remains unchanged, allowing the use of many pre-existing tools.

The main goal of this paper is to motivate a new way of approaching the tumbling robot control problem that addresses previous shortcomings. Effective control for tumbling robots, as with all mobile robots, requires generating appropriate control sequences that bring the robot from its current state to a target global state; the problems can vary greatly depending on the robot kinematics/dynamics, environment, and presence and character of impeding obstacles. Tumbling has its own set of issues that include nonholonomic motion, sliding ground contacts, high angular velocities during tumbles, and the resulting collisions with the ground and/or obstacles. The nonholnomic motion affects planning while the rest of the aforementioned issues complicate state estimation and prediction.

We propose a novel method to handle the issues of tumbling locomotion which takes the problem and separates into locally independent subproblems including calculating feasible tumbles, predicting future states based on robot/terrain dynamics, and high-level planning. Each subproblem is addressed by a separate module; the various modules and their relationships are depicted graphically in Figure 3. The modules are separated into two groups, pre-processing and planning. The overall idea is to produce a set of feasible tumbles in the pre-processing section that are then passed to a planner which assembles sequence of control inputs from the feasible set. By solving for discrete tumbles, the nonholonomic phenomena inherent to tumbling locomotion are effectively removed from the planning problem. Additionally, this method enables the use of discrete planning methods such as $A^\star$ or dynamic programming [10]. A typical implementation of the schema outlined in Figure 3 would proceed as follows:

- For a given state $\boldsymbol{q}_0$, calculate a set of feasible tumbles using information of the robot morphology.
- Propagate all tumbles to the global frame using a dynamic model of the robot. This produces a set of tumble axes in the global frame as well as a set of new states resulting from the tumbles.
- Iterate as desired for increased look-ahead.
- Plan trajectory using set of tumble axes from pre-processing.

In the following subsections we describe in detail each of the modules.

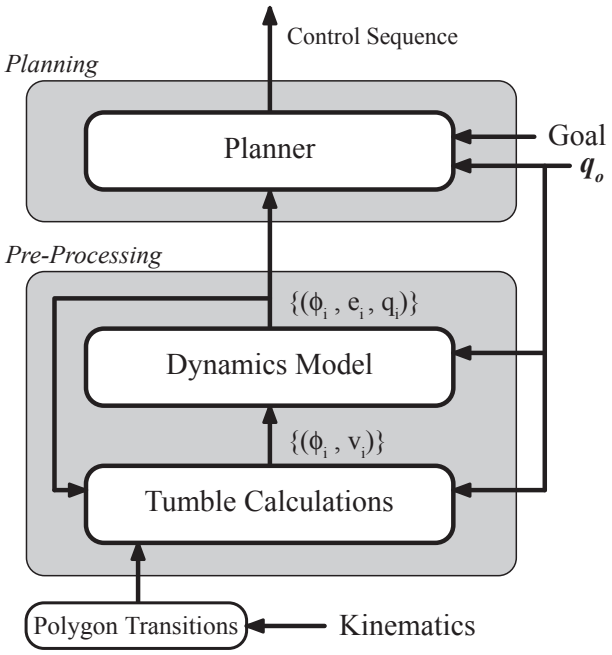


Fig. 3. Flow diagram of proposed modular schema for control of tumbling robots.

A. Tumble Calculations

The tumble calculation module represents the core of our proposed schema. Its functionality consists of building a set of control inputs that will result in a tumble or, in other words, what inputs make the system unstable. Its inputs consist of the state of the robot along with information regarding the robots morphology (this will be discussed in section III-D).

To solve this problem we turn to the theory of stability margins from the field of legged robots. When it comes to measuring stability there are a number of metrics to choose from. For the static case the major metrics include the Static Stability Margin [11], the Energy Stability Margin [12], and the Normalized Energy Stability Margin [13]. The Static Stability Margin relies solely on a projection of the center of gravity (COG) while the latter two methods are based on the amount of energy required to tip the robot. These methods fail to take dynamics into account and are therefore only useful under static or quasi-static (near static) assumptions. Dynamic stability margins include the Effective Mass Center Stability Margin [14] (this is an analog to the Zero Moment Point in the biped literature) and the Normalized Dynamic Energy Stability Margin [15] along with many other momentum-based margins. For a more complete list and detailed comparison of stability margins we refer the reader to [16].

After an appropriate stability margin is chosen, the problem becomes finding where stability vanishes, or more

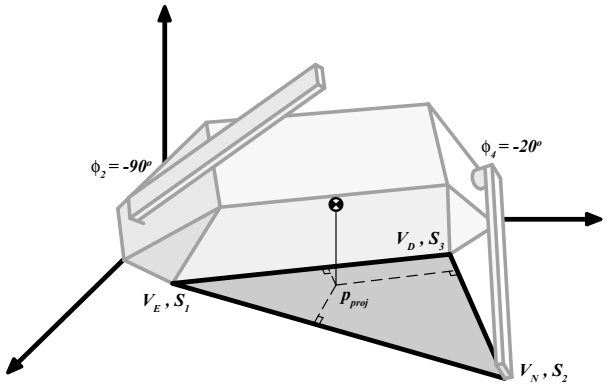


Fig. 4. Sample configuration of the Adelopod tumbling robot with $S = \{v_E, v_D, v_N\}$. The projected barycentric center of mass is depicted as $\boldsymbol{p}_{proj}$. The dashed lines correspond to the stability margin for each edge of the support polygon.

specifically $\{\boldsymbol{\phi} | s = 0\}$ where s is some stability margin. In the following subsection we discuss this calculation for the Adelpod robot.

1) Running Example: Tumble Calculations: In this section we discuss the tumble calculations for the Adelopod tumbling robot. For the purposes of this example we have chosen to use the Static Stability Margin, S_{SM}. From [11] we have the following definition for S_{SM}:

Definition The magnitude of the Static Stability Margin, S_{SM} at time t for an arbitrary support pattern is equal to the shortest distance from the vertical projection of the center of gravity to any point on the boundary of the support polygon. If the pattern is statically stable, the stability margin is positive. Otherwise, it is negative.

For the Adelopod robot on level terrain, the S_{SM} as defined above corresponds to the shortest dashed line in Figure 4. The dashed lines represent the distances from the center of vertical projection of the center of gravity, $\boldsymbol{p}_{proj}$ to each of the three edges ($\overline{\boldsymbol{s}_1\boldsymbol{s}_2}$, $\overline{\boldsymbol{s}_2\boldsymbol{s}_3}$, and $\overline{\boldsymbol{s}_3\boldsymbol{s}_1}$) of the support polygon. Expressing S_{SM} in terms of $\boldsymbol{p}_{proj}$ and S we get the following relation:

$$|S_{SM}| = \min_{\boldsymbol{s}_i, \boldsymbol{s}_j \in S} \sqrt{\|\boldsymbol{p}_{proj} - \boldsymbol{s}_i\|^2 - \langle \boldsymbol{p}_{proj} - \boldsymbol{s}_i, \boldsymbol{s}_j - \boldsymbol{s}_i \rangle^2}, \tag{5}$$

where

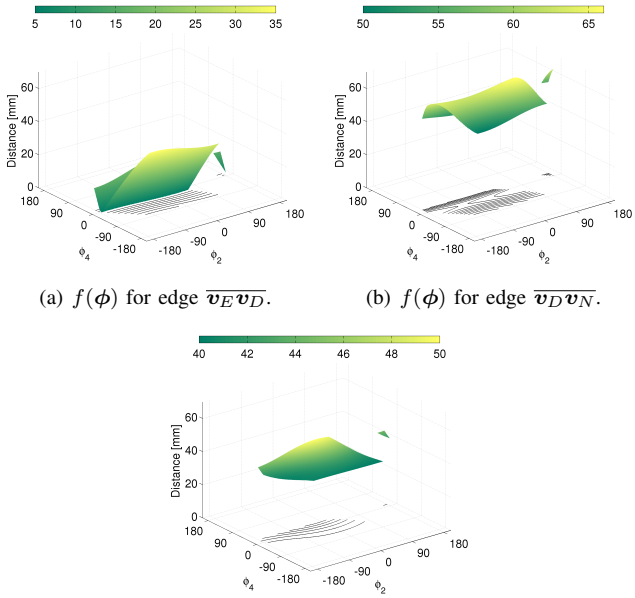
$$\boldsymbol{p}_{proj} = \boldsymbol{p} - \boldsymbol{p}_{\perp} \tag{6}$$

and

$$\boldsymbol{p}_{\perp} = \hat{\boldsymbol{n}} \langle \hat{\boldsymbol{n}}, \boldsymbol{p} - \boldsymbol{s}_1 \rangle \tag{7}$$

$$\hat{\boldsymbol{n}} = \frac{(\boldsymbol{s}_2 - \boldsymbol{s}_1) \times (\boldsymbol{s}_3 - \boldsymbol{s}_1)}{\|(\boldsymbol{s}_2 - \boldsymbol{s}_1) \times (\boldsymbol{s}_3 - \boldsymbol{s}_1)\|}. \tag{8}$$

Note that it is possible to express S_{SM} in terms of its absolute value due to the fact that we are interested only in finding where $S_{SM} = 0$ (this is in contrast to maximizing the stability margin in legged robots).



(a) $f(\boldsymbol{\phi})$ for edge $\overline{\boldsymbol{v}_E\boldsymbol{v}_D}$. (b) $f(\boldsymbol{\phi})$ for edge $\overline{\boldsymbol{v}_D\boldsymbol{v}_N}$.

(c) $f(\boldsymbol{\phi})$ for edge $\overline{\boldsymbol{v}_N\boldsymbol{v}_E}$.

Fig. 5. Distances of projected center of gravity, $\boldsymbol{p}_{proj}$, to specified edges. Tumbles occur when $f(\boldsymbol{\phi}) = 0$.

Because tumbles can only occur over edges, we can examine each edge individually to test for feasible tumbles, thus removing the minimization from Equation (5). On a per edge basis, tumbles occur at the zeros of the following expression:

$$f(\boldsymbol{\phi}) = \|\boldsymbol{p}_{proj} - \boldsymbol{s}_i\|^2 - \langle \boldsymbol{p}_{proj} - \boldsymbol{s}_i, \boldsymbol{s}_j - \boldsymbol{s}_i \rangle^2. \tag{9}$$

It is important to note that the above expression only holds for the given support polygon defined by S. If the support polygon changes, stability must be redefined in terms of the new polygon. Therefore when evaluating $f(\boldsymbol{\phi})$ for a particular S the following constraints must hold (assuming level terrain):

$$\langle \boldsymbol{s}_3 - \boldsymbol{s}_1, (\boldsymbol{s}_2 - \boldsymbol{s}_1) \times (\boldsymbol{s}_i - \boldsymbol{s}_3) \rangle = 0, \quad i = 4, \ldots, m \tag{10}$$

$$\hat{n}(\boldsymbol{v}_i - \boldsymbol{s}_1) < 0, \quad \forall \boldsymbol{v}_i \in V \backslash C \tag{11}$$

$$\hat{n}(\boldsymbol{c}_i - \boldsymbol{s}_1) \leq 0, \quad \forall \boldsymbol{c}_i \in C. \tag{12}$$

The first constraint ensures that all points in S remain coplanar (no vertices leave the support polygon) while the second and third ensure that all points not in C remain above the support polygon (no new vertices are added to the support polygon).

We plot $f(\boldsymbol{\phi})$ from Equation (9) for $S = \{v_E, v_D, v_N\}$ in Figure 5 where each subplot represents an edge of the support polygon. For clarity, we have plotted $f(\boldsymbol{\phi})$ versus only ϕ_2 and ϕ_4; the shoulder variables have been set to the fully contracted position ($\phi_1 = 57^\circ$ and $\phi_3 == 123^\circ$) as seen in Figure 2. Only values for which $\boldsymbol{\phi}$ satisfies the constraints in Equations (10)-(12) are shown. This figure provides insight on the behavior of the robot for the given support polygon. We can see from Subfigure 5(b) that the

robot can tumble over edge $\overline{\boldsymbol{v}_E\boldsymbol{v}_D}$ for appropriate values of ϕ_2 and ϕ_4. Conversely, Subfigures 5(b) and 5(c) indicate that it is not possible to tumble about edges $\overline{\boldsymbol{v}_D\boldsymbol{v}_N}$ or $\overline{\boldsymbol{v}_N\boldsymbol{v}_E}$ from the support polygon defined by $S = \{v_E, v_D, v_N\}$. Additionally we can see that ϕ_4 (rotation of arm in contact) has a much greater impact than ϕ_2 on the stability. In fact, the only effect ϕ_4 has on the stability comes from its influence on the (barycentric) center of gravity for the robot,

$$p = \frac{m_\ell \boldsymbol{p}_\ell + m_r \boldsymbol{p}_r + m_b \boldsymbol{p}_b}{m_\ell + m_r + m_b}, \tag{13}$$

where m_ℓ, m_r, m_b are the component masses of the left arm, right arm, and body respectively. Similarly $\boldsymbol{p}_\ell, \boldsymbol{p}_\ell, \boldsymbol{p}_\ell$ are the component centers of gravity.

It turns out that there is no analytical solution for the zeros of Equation 9 and therefore we must resort to minimization techniques. This is made somewhat easier with the knowledge that our objective function is lower-bounded at zero. Additionally, if we make the assumption that the arms are massless (this could be made feasible by using lighter materials such as carbon fiber for the arms in comparison to the steel used in generating Figure 5), we get the surfaces of Figure 6. Here we see that stability is strictly a function of only the variables appearing in the equations of the kinematic contacts (ϕ_4 for v_N in this case); this is evident from the contour lines parallel to the $\boldsymbol{x}$-axis. This assumption reduces the complexity of the surface and eliminates some local minima for certain support polygons.

With some assumptions such as level terrain, it is often possible to solve for the tumbles offline though minimization (with random starting points) and store them in a lookup table on the robot to be accessed at runtime.

B. Dynamics Propagation

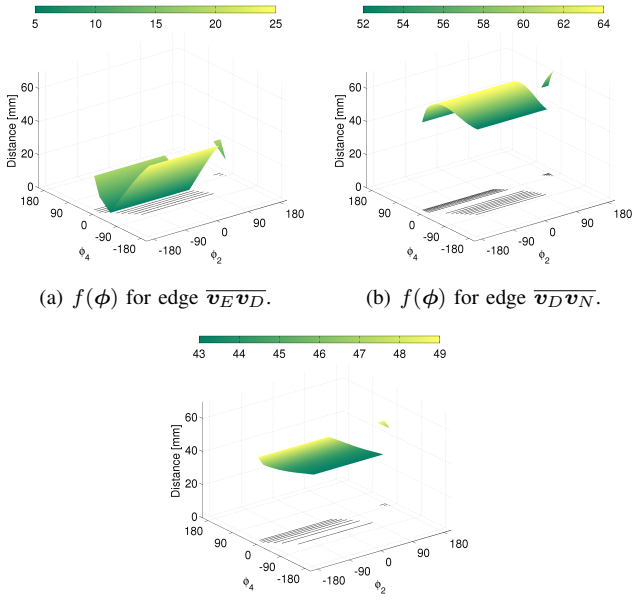
The solutions from the tumble calculation module are relative to the robot's frame and take into account only the robot's kinematics. In order to use these calculations for any useful planning, we must predict how the robot will react in the real world as the control vector $\boldsymbol{\phi}$ changes with time. The purpose of the dynamics propagation module is just that, it is responsible for transforming the feasible tumbles into the global frame, taking into account the robot's interaction with the environment (ground reaction forces and sliding friction). Such behavior is explained by the equations of motion for the robot. The equations of motion can be written in the following generalized form:

$$A(\boldsymbol{q})\,\ddot{\boldsymbol{q}} = B(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{14}$$

where $\ddot{\boldsymbol{q}}$ is the column matrix of generalized accelerations of the system. By modeling the robot as a rigid multibody it is possible to derive matrices A and B by a variety of methods including the application of the principle of virtual power (Jourdain's principle) [17].

C. Planning

By the time control reaches the planning stage much of the difficult work has been done. The planner module takes



(a) $f(\boldsymbol{\phi})$ for edge $\overline{\boldsymbol{v}_E\boldsymbol{v}_D}$. (b) $f(\boldsymbol{\phi})$ for edge $\overline{\boldsymbol{v}_D\boldsymbol{v}_N}$.

(c) $f(\boldsymbol{\phi})$ for edge $\overline{\boldsymbol{v}_N\boldsymbol{v}_E}$.

Fig. 6. Distances of the projected center of gravity, $\boldsymbol{p}_{proj}$, to specified edges assuming massless arms. Resulting surfaces are functions only of arms in contact (ϕ_4 in this case). Tumbles occur when $f(\boldsymbol{\phi}) = 0$.

the set of discrete global tumbles and chooses a sequence accordingly to reduce the distance from the desired goal state. In Figure (7) we show a sample scenario where the dashed lines represent the sequence of tumbles chosen by the planner. Because the tumbles are discrete at this point any form of discrete planning can be used. Due to the issues regarding accurate state propagation in tumbling robots we highly suggest using a receding horizon planning scheme where planning happens periodically during the task. If the horizon is chosen appropriately the inaccuracies of the state estimation can be managed.

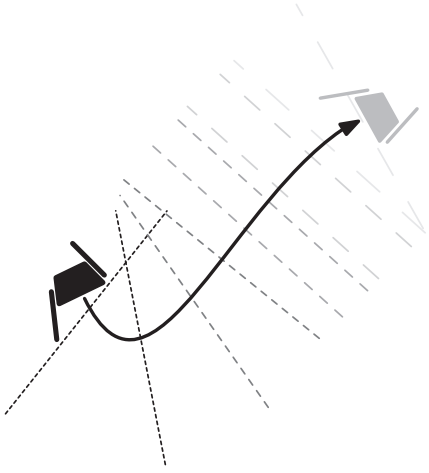


Fig. 7. A sample depiction of tumble axes chosen by the planner.

1) Running Example: Heuristics: We have discovered in our experiences with the Adelopod that rapid progress in

a straight line can be achieved over a variety of terrain by bringing the arms into phase as they rotate about the body. From this we have concluded that, in certain scenarios, planning resources might best be spent trying to align the Adelopod's sagital plane with the target destination. After acceptable alignment has been achieved the forward gait can be executed until external disturbances require replanning. This is just one instance of a useful heuristic applicable to our approach.

D. Polygon Transitions

We have discussed the actual calculation of tumbles and have shown that constraints are necessary to ensure that the solution is valid for the support polygon in question. For almost every support polygon it is possible to alter the support polygon without inducing a tumble (adding or removing contact points so that the cardinality of S remains greater than or equal to 3) extended the set of feasible tumbles. This is necessary for states without tumbles (e.g., $S = \{v_J, v_K, v_D, v_E\}$) and often desirable for others as it increases the number of feasible tumbles at the planner's disposal.

Although this calculation can theoretically be done online, we believe that an adjacency graph of the support polygons would be more suitable for implementation purposes. Currently we are unsure about how to generate such a graph efficiently, however, we believe that the problem is closely related to finding all possible convex hulls of the robot for changing ϕ. Alternate methods include on-the-fly construction where adjacent support polygons are added to the graph as they are encountered. This same effect could also be achieved by a random walk over ϕ in a simulation environment (our current method of choice).

E. Running Example: Adjacent Support Polygons

As previously mentioned, the Adelopod in a state with $S = \{v_J, v_K, v_D, v_E\}$ has no feasible tumbles. By bringing one of the arms into contact however, an adjacent support polygon with feasible tumbles can be obtained. Adjacent support polygons for this scenario include $\{v_E, v_N, v_D\}, \{v_E, v_M, v_D\}, \{v_J, v_K, v_N\}$, and $\{v_J, v_K, v_M\}$ which are achieved by bringing one of the arms into contact with the ground either in front of or behind the robot.

IV. Conclusion

In this paper we outlined a new schema for control of tumbling locomotion where we break the problem up into separate subcomponents, resulting in a natural discretization of the overall task. We discussed each subcomponent (module) and provided examples where applicable using the Adelopod robotic platform. In general, we believe that the proposed schema provides a promising solution to the tumbling control problem. By providing sets of feasible tumbles to the planner we effectively separate the kinematics and dynamics from the planner, significantly reducing complexity and enabling the use of discrete planning methods.

A. Future Work

Currently we are working on completing an implementation of our proposed modular schema on the Adelopod robotic tumbling platform for evaluation. We are using many of the ideas provided in the running example discussed throughout the paper. Additionally we are presently exploring the use of stability margins other than the Static Stability Margin in hopes to find an analytic solution to the tumble calculations. We also believe that an efficient method of calculating the graph of adjacent support polygons from Section III-D is worth exploring. Such a method would greatly benefit our research.

Acknowledgements

This work has been supported in part by National Science Foundation through grants #IIS-0219863, #CNS-0224363, #CNS-0324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, and #CNS-0708344.

References

[1] P. E. Clark, M. L. Rilee, S. A. Curtix, W. Truszkowski, G. Marr, C. Cheung, and M. Rudisill, "Bees for ants: Space mission applications for the autonomous nanotechnology swarm," in *Proceedings of the First AIAA Intelligent Systems Technical Conference*, 2004.

[2] M. Abrahantes, A. Silver, and L. Wendt, "Gait design and modeling of a 12-tetrahedron walker robot," in *Thirty-Ninth Southeastern Symposium on System Theory*, 2007, pp. 21–25.

[3] M. W. Tilden, "Nervous networks and the architecture of "living" machines," Goddard Scientific Colloquium, 2001.

[4] A. Marigo, M. Ceccarelli, S. Piccinocchi, and A. Bicchi, "Planning motions of polyhedral parts by rolling," *Algorithmica*, vol. 26, pp. 560–576, 2000.

[5] A. Bicchi, Y. Chitour, and A. Marigo, "Reachability and steering of rolling polyhedra: A case study in discrete nonholonomy," *IEEE Transactions on Automatic Control*, vol. 49, no. 5, pp. 710–726, 2004.

[6] B. Hasslacher and M. W. Tilden, "Living machines," *Robotics and Autonomous Systems*, vol. 15, pp. 143–169, 1995.

[7] (2008, February) The solarbotics turbot tumbling robot. Solarbotics Ltd. [Online]. Available: http://www.solarbotics.com/assets/documentation/solarbotics_turbot_kit_feb122008.pdf

[8] B. Hemes, D. Fehr, and N. Papanikolopoulos, "Motion primitives for a tumbling robot," in *IEEE/RSJ International Conference on International Conference on Intelligent Robots and Systems*, 2008.

[9] B. Hemes and N. Papanikolopoulos, "The adeolpod tumbling robot," in *IEEE International Conference on Robotics and Automation*, 2009.

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

[11] R. B. McGhee and A. A. Frank, "On the stability properties of quadruped creeping gaits," *Mathematical Bioscience*, vol. 3, pp. 331–351, 1968.

[12] D. Messuri and C. Klein, "Automatic body regulation for maintaining stability of a legged vehicle during rough-terrain locomotion," *IEEE Journal of Robotics and Automation*, vol. 1, no. 3, pp. 132–141, 1985.

[13] S. Hirose, H. Tsukagoshi, and K. Yoneda, "Normalized energy stability margin: Generalized stability criterion for walking vehicles," in *Proceedings of the Inernational Conference on Climbing and Walking Robots*, 1998, pp. 71–76.

[14] D. Kang, Y. Lee, S. Lee, Y. Hong, and Z. Bien, "A study on an adaptive gait for a quadruped walking robot under external forces," in *IEEE International Conference on Robotics and Automation*, 1997, pp. 2777–2782.

[15] E. Garcia, P. G. de Santos, and C. C. de Wit, "An improved energy stability margin for walking machines subject to dynamic effects," *Robotica*, vol. 23, no. 1, pp. 13–20, 2005.

[16] P. G. de Santos, E. Garcia, and J. Estremera, *Quadrupedal Locomotion*. Springer-Verlag, 2006.

[17] J. Wittenburg, *Dynamics of Multibody Systems*. Springer-Verlag, 2008.