Robust Execution of Temporally Flexible Plans for Bipedal Walking Devices

Andreas Hofmann and Brian Williams

Computer Science and Artificial Intelligence Lab, MIT
32 Vassar St., rm. 32 - 275
Cambridge, MA 02139

hofma@csail.mit.edu, williams@mit.edu

Abstract

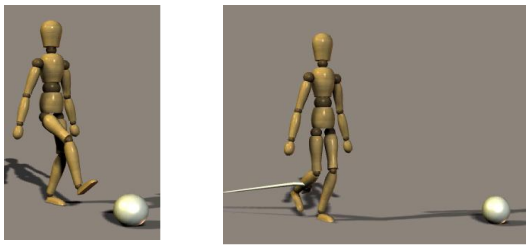
Robotic wheeled rovers have been successfully controlled by activity execution systems that use plans with temporal flexibility to adapt to disturbances. These systems abstract away the detailed dynamic constraints of the controlled device. To control dynamic, devices, such as agile bipeds, we extend this execution paradigm to incorporate detailed dynamic constraints.

Building upon prior work on dispatchable plan execution, we introduce a novel approach to flexible plan execution that achieves robustness by exploiting spatial as well as temporal plan flexibility. To accomplish this, we first transform the high-dimensional system into a set of low dimensional, weakly-coupled systems. Second, to coordinate these systems, we compile a plan into a flow tube description, representing all legal trajectories and their temporal coordination. Finally, the problem of runtime plan dispatching is reduced to maintaining state trajectories in their associated flow tubes. The approach is validated using a high fidelity biped simulation.

Introduction

Effective use of autonomous robots in unstructured human environments requires that they have sufficient autonomy to perform tasks independently, and that they operate robustly in the presence of disturbances. A challenging example of such a robot is a bipedal walking machine. An example task for such a system is to walk to a soccer ball and kick it, as shown in Fig. 1a. Stepping movement must be synchronized with ball movement so that the kick happens when the ball is close enough. If the biped encounters a force disturbance, it must compensate in some way. For example, a trip, shown in Fig. 1b causes disruption of synchronization between the stepping foot, and the overall forward movement of the system's center of mass. To avoid falling, the system must compensate in a way that restores this synchronization. A second example task is walking on a constrained foot path, as shown in Fig. 2. In these examples, and others like them, the key challenge is to move a complex, dynamic system to the right place at the right time, despite actuation limits, and despite disturbances.

Existing activity execution systems [Muscettola et al., 1998] are capable of executing complex task sequences, while observing temporal constraints, and are able to deal with disturbances, but they do not take into account detailed dynamic limitations. For a biped, such limits cannot be ignored. A biped's limited base of support limits the acceleration of its center of mass. This limit must be taken into consideration for tasks requiring agility. Existing systems for biped motion planning and control [[Hirai et al., 1998] do take dynamic limitations into account. However, these methods produce very detailed and inflexible reference trajectories, so they cannot exploit flexibility in task requirements to provide robustness. Our system combines the advantages of the activity execution systems with those of the biped motion planning systems in that it utilizes plan flexibility to compensate for significant disturbances, but also takes into account detailed dynamic limitations.



a. b.

Fig. 1 – a. kicking soccer ball, b. trip disturbance.

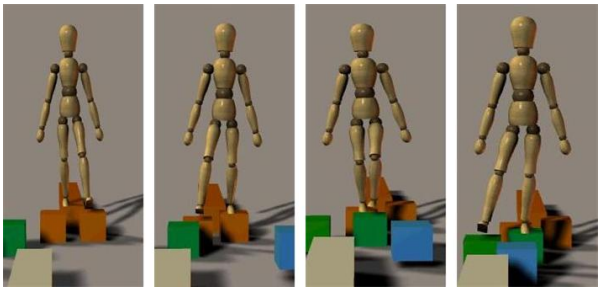


Fig. 2 – Walking with foot placement constraints.

To address the challenge presented by the biped's nonlinearity, and tight coupling, we linearize and decouple the biped system into a set of independent, linear, single-input single-output second-order systems, resulting in an

abstraction of the biped that is easier to control. We accomplish this through a novel controller called a *dynamic virtual model controller* [Hofmann, et. al., 2004]. This controller allows the biped to be controlled by virtual spring-damper elements attached at key *reaction points*, such as the biped's center of mass and stepping foot, as shown in Fig. 3.

In order to project the future evolution of the biped's state, we compute *flow tubes* [Bradley and Zhao, 1993] that define valid operating regions. The flow tubes represent bundles of valid state trajectories that take into account dynamic limitations, and also satisfy plan requirements. The flow tubes observe the discontinuous changes in actuation constraints due to ground contact events. Because generation of flow tubes is computationally intensive, we perform this generation off-line. At runtime, the system monitors task execution by monitoring the abstracted biped's state, and checking whether each state trajectory is in its tube. If a disturbance occurs, the system adjusts the control parameter settings in the abstracted biped, so that the trajectory remains inside the tube, as shown in Fig. 4a. If the disturbance is too large, it may push the trajectory outside its tube, and no adjustment of control parameter settings will bring it back, as shown in Fig. 4b.

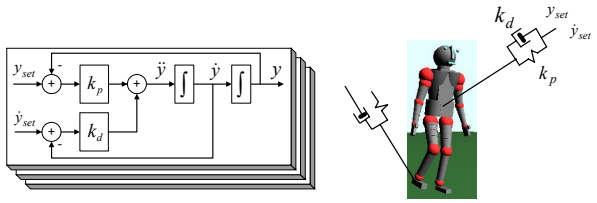


Fig. 3 – The dynamic virtual model controller provides an abstraction of the biped whose reaction points move as if they were controlled by virtual elements.

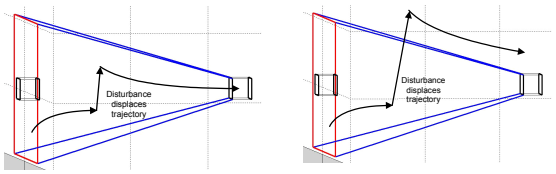


Fig. 4 – a. If a disturbance is not too large, the trajectory remains in its tube. b. If a disturbance is too large, it pushes the trajectory outside its tube, and execution fails.

Model-Based Executive

We extend the capabilities of the activity execution systems to incorporate observance of state-space as well as temporal constraints. We use a *model-based executive* [Williams and Nayak, 1997] to interpret task goals, monitor biped state, and compute joint torque inputs for the biped, as shown in Fig. 5.

The input to the model-based executive is a *Qualitative State Plan* (QSP), which specifies state-space and temporal requirements. The output of the executive is the joint torques for the biped. The plan compiler component compiles the QSP, generating the flow tubes and the dispatchable graph [Muscettola et al., 1998]. This compiled form is output as a *Qualitative Control Plan* (QCP). The QCP is executed by the dispatcher, which schedules activity start times using a method similar to that used in the activity execution systems, and executes activities by keeping trajectories in their flow tubes.

Qualitative State Plan A walking task is most naturally specified as a sequence of qualitatively similar states. A *qualitative state* indicates which feet are on the ground, and includes constraints on foot position. It may also include state space constraints on the biped's center of mass, and temporal constraints specifying time ranges by which the state space goals must be achieved. A sequence of qualitative states represents intermediate goals that lead to the final overall task goal, as shown in Fig. 6.

A QSP has a set of *activities* representing constraints on desired state evolution. In Fig. 6, the activity *left foot ground 1* is for the left foot, *right foot ground 1*, is for the right foot, and CM1 – 4 are for the center of mass. Events define the boundaries of qualitative states. For example, the right toe-off event defines the end of the first qualitative state (double support), and the beginning of the second qualitative state (left single support). The qualitative state plan in Fig. 6 has a temporal constraint between the start and finish events.

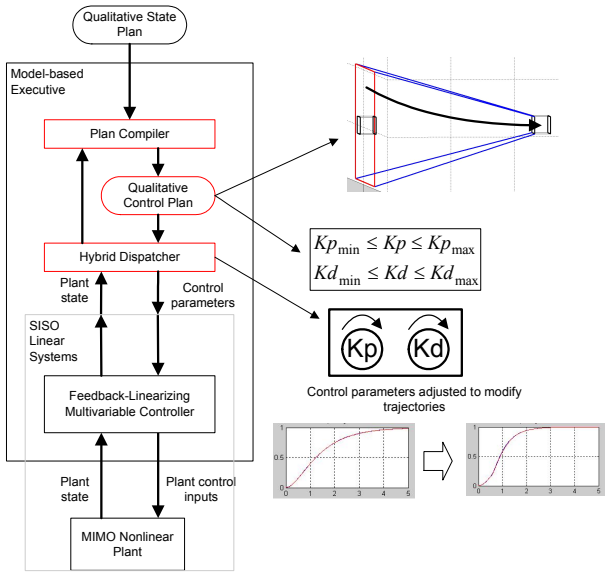


Fig. 5 – The Model-based executive consists of a plan compiler, a dispatcher, and a DVMC controller.

An activity may also have operating region constraints that specify valid regions in state-space where the trajectory must be over the entire duration of the activity. For example, CM movement is represented by four separate activities: CM1 – CM4, and each of these activities has different operating regions. This is due to the

discontinuous changes in the base of support resulting from the foot contact events; the base of support in double support is very different from the one in single support.

Abstracted Plant The hybrid task-level executive does not control the biped directly, but rather, a linearized abstraction, which is easier to control than the actual biped. This *abstracted plant* is provided by the dynamic virtual model controller. The controller transforms the tightly-coupled nonlinear plant into a set of seemingly independent, linear, 2^{nd}-order single-input single-output (SISO) systems, as shown in Fig. 3. Furthermore, the controller performs a state transformation from the robot's *joint* state space, to a more convenient *workspace* state space, where the state vector consists of variables relevant to balance control, such as center of mass position and velocity. The QSP state space constraints are expressed in terms of these workspace state variables.

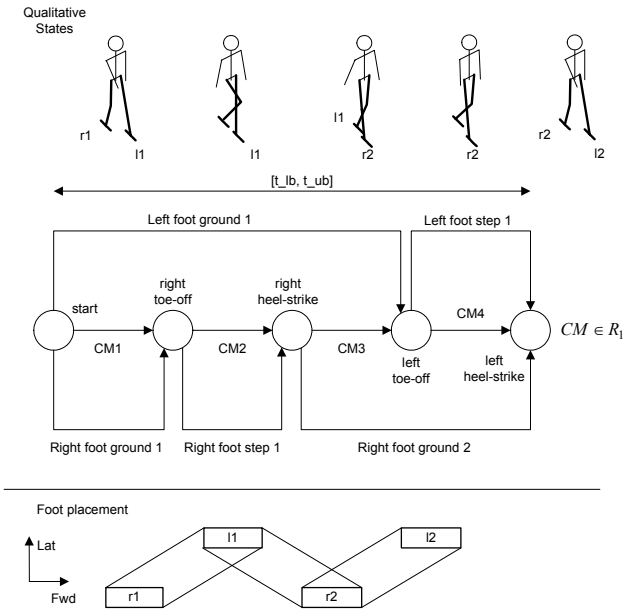


Fig. 6 - Example qualitative state plan for walking gait cycle. Circles represent events, and horizontal arrows represent activities. Activities may have associated state space constraints, such as the goal region constraint $CM \in R_1$. Foot placement constraints are indicated at the bottom; for example, rectangle r1 represents constraints on the first right foot position on the ground, and rectangle l1 on the first left foot position.

Hybrid Executive Successful execution of a QSP by the hybrid executive is defined in terms of a set of SISO plant trajectories and a schedule for events. The executive must find a sequence of control parameter settings for each SISO system such that the state trajectories are consistent with the state space and temporal constraints of the QSP, and also are consistent with the plant dynamics. The schedule of events must be consistent with the temporal constraints specified in the QSP.

Qualitative Control Plan To support temporal flexibility in the QCP, the flow tube approximation must be capable of representing trajectories corresponding to multiple activity durations. Consider an activity with a goal region, R_G. Suppose that we wish to control the duration, d, of this activity such that $l \le d \le u$. We call the interval, $[l,u]$, the *controllable duration* of the activity. The requirement to control the duration in this way is represented by a set of flow tubes, one for each value of d, as shown in Fig. 7.

Consider, now the initial cross sections of these tubes. The intersection of these cross sections, shown in Fig. 8, represents an initial region from which R_G can be achieved in *any* duration, D_1, D_2, or D_3. Thus, in order to support controllable activity durations, it is a requirement that the flow tube approximation be able to represent flow tube sets of the type shown in Figs. 7 and 8.

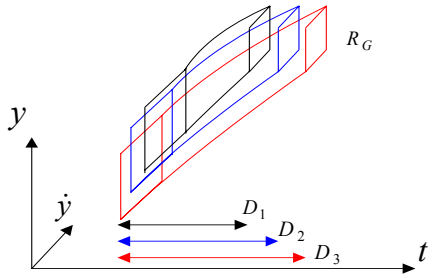


Fig. 7 - Flow tube set for variable duration. The longest flow tube reaches the goal region, R_G, after duration D_3. The shortest flow tube reaches R_G after duration D_1.

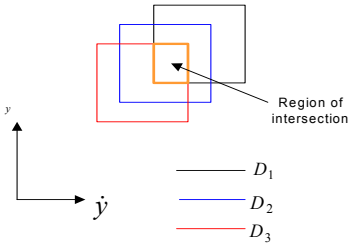


Fig. 8 – The region of intersection is the initial region from which R_G can be achieved in any of the durations.

Hybrid Dispatcher and Plan Compiler The dispatcher executes QCP activities by scheduling start and finish events consistent with the temporal constraints, and by setting control such that the associated trajectory reaches the activity's goal region at an acceptable time. The dispatcher performs three key functions: initialization, monitoring, and transition.

For initialization of activity execution, the dispatcher chooses a goal duration for the activity consistent with its execution window, and sets control parameters for the control activity such that the state trajectory is predicted to be in the activity's goal region at the goal time. The initialization function formulates a small *quadratic*

program (QP) in order to determine these control parameters. Key to this formulation's simplicity is the fact that the analytic solution of the SISO system is used to predict the future state of the SISO system associated with the activity.

After initializing an activity, the dispatcher begins monitoring execution by continually checking whether the state trajectory remains in its flow tube, and hence, is on track. If this is not the case, it attempts to correct by adjusting control parameters, using the QP formulation. If this is unsuccessful, the dispatcher aborts plan execution and requests a new plan. As part of the monitoring function, the dispatcher also continually checks whether a control activity's completion conditions are satisfied. If this is the case, the dispatcher switches to the transition function.

If the control activity being executed has a successor, the transition function invokes the initialization function for this new activity. The transition function also notes the time of the transition event and propagates this through the temporal constraints using an efficient constraint propagation algorithm [Muscettola, 1998. This propagation tightens execution windows of future events. When all activities in the QCP have been executed successfully, execution terminates; all goals have been achieved.

The plan compiler takes a QSP as input and produces a QCP. In order to do this, the plan compiler computes the flow tube approximations for each control activity; it computes each control activity's initial and goal regions, and l and u duration bounds. To compute these parameters, we formulate a constrained optimization problem, where the constraints are state-space and temporal constraints from the QSP, and dynamic limitations of the plant, and the cost function is adjusted to maximize the size of the flow tube approximation in order to maximize robustness.

Results and Discussion

Fig. 2 shows dynamic walking, but with an irregular stepping pattern necessitated by the blocks. The blocks move slowly, so timing of foot placement, as well as the positioning is important. This forces the biped to move at a relatively fast speed of about 0.8 m/s. Fig. 10 shows the CM trajectory and foot placements for this test. The dynamic nature of this task is indicated by the fact that the CM trajectory barely touches the foot placement polygons, and in one case, is 0.1 m away. This indicates that the system is not statically stable, and is relying on subsequent foot placements to maintain balance.

Fig. 11 a. shows the effect of a trip disturbance, if the dispatcher makes no adjustment of control parameters; the biped falls. In order to avoid a fall, the dispatcher's monitor function, detects that the foot has been delayed and will not reach its goal region at the desired time. The dispatcher compensates by adjusting control parameters, causing the stepping foot to move forward more quickly, re-establishing synchronization, as shown in Fig. 11b.

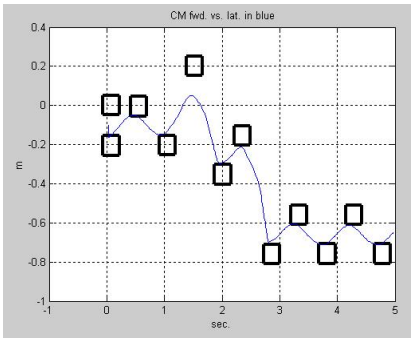


Fig. 10 – Foot placement and CM trajectory for irregular stepping test

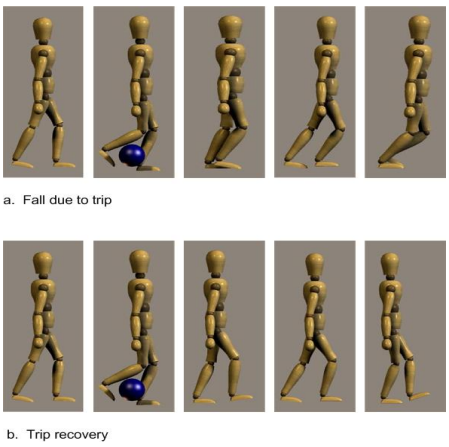


Fig. 11 – Trip disturbance, fall (a), recovery (b)

References

[Bradley and Zhao, 1993] E. Bradley and F. Zhao. Phase-space control system design. *Control Systems*, 13(2),39-46 April, 1993

[Hirai et al., 1998] K. Kirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of Honda humanoid robot. *IEEE International Conference on Robotics and Automation* (ICRA)

[Hofmann et al., 2004] A. Hofmann, S. Massaquoi, M. Popovic, and H. Herr. A sliding controller for bipedal balancing using integrated movement of contact and non-contact limbs. *Proc. International Conference on Intelligent Robots and Systems* (IROS). Sendai, Japan

[Muscettola et al., 1998] N. Muscettola, P. Morris, and I. Tsamardinos. Reformulating temporal plans for efficient execution. *Proc. Of Sixth Int. Conf. On Principles of Knowledge Representation and Reasoning*, 1998

[Williams and Nayak, 1997] B. Williams and P. Nayak. A Reactive Planner for a Model-based Executive. *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI, 1997)