# Safe Control of Hopping in Uneven Terrain

Brian Howley, *Member, AIAA,* and Mark Cutkosky, *Member, IEEE*

***Abstract*— Legged robots have long been proposed as a means of locomotion in unstructured environments. However, much of the analysis to support the design and analysis of legged systems has assumed level or mildly varying terrain. This work determines the maximum terrain variability that a single legged vertical hopping robot can safely negotiate without stumbling or violating joint constraints. The analysis assumes that the step to step variation in terrain height is unknown but bound and casts the hopping problem as a non-cooperative game between robot and environment. The maximum worst case terrain variation that the system can accommodate safely is a measure of the system's "ruggedness". Mechanical properties which maximize ruggedness under closed loop control are identified. Extension of this analysis to more complex systems requires a numerical approach that is a topic of a companion paper.**

***Index Terms*— Hybrid systems, legged robots, safe control, uneven terrain.**

## I. Introduction

LEGGED robots can operate in environments that wheeled or tracked vehicles of similar size cannot. Generally, these machines follow one of two fundamental approaches to the problem of legged locomotion. One approach is to plan foot placement to maintain static stability throughout the step. Examples of statically stable machines include the *Autonomous Suspension Vehicle* [15], *Titan VII* [6], *Dante II* [1], and *ATHLETE*, a six legged wheeled vehicle proposed for lunar exploration [8]. These machines rely on a high degree of motion planning or tele-operation and require knowledge of the environment. Another approach, pioneered by Raibert [12] relies on dynamic stability. Dynamically stable systems run or hop by exchanging kinetic and potential energy in different phases of motion. Recent examples of running machines that include some dynamic stability include *RHex* [13], the *Sprawl* family of robots [3] and [4], and *Tekken* [5]. These robots are relatively fast, from one to several body lengths per second, and rely on passive mechanical properties to stabilize against disturbances or variations in the environment.

Despite impressive experimental results, a comprehensive framework for understanding the interaction between dynamically stabilized legged robots and a variable environment is lacking. The present work takes the position that robot performance is ultimately constrained by safe operating limits and investigates the notion of safety as a design criterion. Here, safety is specified by legal values for the state variables describing robot dynamics. Safe operation is maintained under worst case environmental disturbances. Worst case disturbances and the corresponding optimal control are determined by application of game theory to the safety problem. The approach has several advantages, including: 1) design limitations such as maximum force and joint deflections are treated explicitly, 2) only the general nature of the control (e. g. feedback versus open loop) and not the specific implementation is prescribed so that safety over a wide range of possible control implementations is determined, and 3) the method does not assume small perturbations from a stable, steady state cycle and is appropriate for analysis of motion over highly irregular terrain.

A central problem to studying the effects of uneven terrain is determining a meaningful method of analysis. Large terrain variations preclude steady state running trajectories required for a traditional stability analysis. Furthermore, perfect knowledge of the environment is generally not practical, so the method of analysis must accommodate uncertainty. One approach is to treat the problem as a random process. For non-Gaussian systems this approach requires repeated convolutions of probability density functions and is computationally prohibitive. Even if the approach was feasible, sensitivity to assumptions about the statistics of the terrain variation and sensor and actuator errors would make interpretation of any results difficult. An alternative is to treat the problem as a game between between the robot and its environment. Rather than exist passively, the environment is assumed to act perversely but within some prescribed limits. A game theoretic approach transforms the uncertainty into a deterministic problem by considering worst case conditions [2].

Consider the problem of a single legged robot hopping over an uneven surface as illustrated in Figure 1. The robot must jump high enough to clear each step during its ballistic phase. However, the environment is unknown and the change in elevation may be either higher or lower. If the robot applies maximum thrust and the elevation of the next step is lower, the robot may land with enough velocity to cause damage or lose control. In the context of a game, the robot attempts to stay within a safe operating regime while crossing over the terrain and the environment varies in a way that attempts to force the robot out of safe operation. The robot "wins" if it can negotiate the environment without violating constraints, and the environment "wins" otherwise. Because the robot's position and velocity during one step influence the position and velocity of succeeding steps, the game is played out over a sequence.

Tomlin, et. al. [14] use a game theoretic framework for the controller design of hybrid systems. A hybrid system has both discrete and continuous dynamics. For example, a hopping robot is a hybrid system because there are discrete changes in the robot's dynamics between contact and flight. The control problem is cast as a pursuit–evasion game where the controller wins if the systems remains within a safe subset of states

Manuscript received January 20, 2002; revised November 18, 2002. This work was supported by the IEEE.

Brian Howley is with Lockheed Martin Space Systems Company, Sunnyvale, CA

Mark Cutkosky is with Stanford University, Stanford, CA

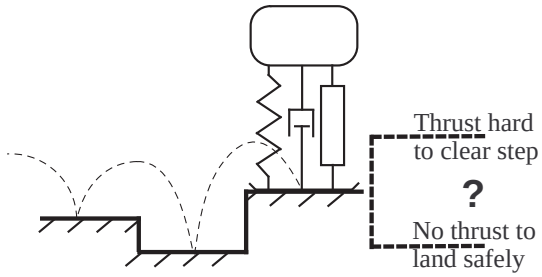



Fig. 1. Hopping over uneven terrain cast as a game. The robot attempts to hop over uneven terrain without violating joint constraints or stumbling over obstacles. The environment is unknown to the robot but is assumed to vary in a way to force an unsafe condition if possible.

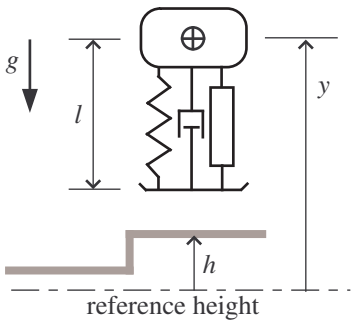



Fig. 2. Schematic representation of a vertical hopper. The hopper dynamics are described the height, $y$, and velocity, $\dot{y}$, above a reference and the height, $h$, of the ground with respect to that reference. Parameters affecting hopper dynamics include unsprung leg length, $l$, spring and damping constants, and maximum thrust capability.

for a specified period of time. The methodology searches for a feedback control law that guarantees that the system remains within the largest possible safe subset of the state space despite disturbances by the environment. The resulting control law is least restrictive in the sense that the control may take on any legal value except at the boundary of the subset where it must act to maintain safety. It can be used in combination with other control laws to achieve performance specifications under normal operating conditions and maintain safety when safety constraints would otherwise be violated [9]. The approach is motivated by high confidence systems such as air traffic management and automated highway systems, but can be applied to other systems where safety can be expressed as a subset of the state space.

Section 2 of this paper describes the dynamics of a single legged vertical hopper including the effects of variable terrain. Section 3 describes the safety problem and recites a general approach for the control synthesis problem given in [14]. This section also describes some issues with the general approach not cited in the previous work. Section 4 describes the results of the analysis method as applied to the robot hopping problem. Section 5 examines the effects of the robot's thrust capability and mechanical properties on terrain hopping ability. A metric for "ruggedness" is defined which is the maximum change in terrain height the system can tolerate between hops and guarantee safe operation. Section 6 discusses extension of the approach to running with single and multiple legs. Running complicates the problem requiring additional states and hybrid mode transitions. Detailed study of the running problem is beyond the scope of the current work but is feasible.

## II. Hopper Dynamics

Figure 2 is a schematic of a single legged vertical hopping robot. The hopper leg consists of a spring, damper, and linear actuator connected in parallel. The unsprung length of the leg from the hopper mass center is $l$. The spring and damper are linear time invariant elements. The linear actuator can exert an upward force on the hopper body between 0 and $f_{max}$. The broken line at the bottom of the figure is a fixed reference. The height of the hopper mass center with respect to the fixed reference is $y$. The height of the ground directly beneath the hopper with respect to the fixed reference is $h$.

The effect of uneven or variable terrain are included by allowing the height of the ground, $h$, to change when the hopper is at the apex of flight. This construct treats the robot as if it were hopping in place while the ground moves underneath. To support this construct, the flight phase of motion is divided into three phases as shown in Figure 3: `ascent`, `descent`, and `step`. The `step` phase is a discrete (instantaneous) phase of motion in which the height of the ground, $h$, changes by an amount over some range, $\Delta h \in [\Delta h_{min} \Delta h_{max}]$. If the robot has insufficient clearance at the time of the step then a collision occurs and the robot stumbles.

Since the ground height, $h$, is constant except at the instant of the `step`, we can simplify the dynamics by adjusting our reference so that $\hat{y} \triangleq (y - h - l)/l$. This incorporates the ground height into the dynamics and normalizes the vertical height so that at $\hat{y} = 0$ the robot leg is just in contact with the ground. The dynamics can be further simplified by dividing by mass and normalizing time by $\sqrt{g/l}$ so that the force from gravity has a non-dimensional value of -1. The non-dimensional equations of motion then become:

$$\hat{y}'' = \begin{cases} -1 & \texttt{ascent, descent} \\ -\hat{k}\hat{y} - \hat{b}\hat{y}' + u - 1 & \texttt{contact} \end{cases} \quad (1)$$

$$\text{where } 0 \le u \le (f_{max}/mg) = u_{max} \text{ in } \texttt{contact}$$
$$\text{and } \hat{y}^+ = \hat{y}^- + \Delta\hat{h} \text{ in } \texttt{step}$$

The primes for $\hat{y}''$ and $\hat{y}'$ denote differentiation with respect to normalized time. $\hat{k}$ and $\hat{b}$ are normalized leg spring and damping coefficients and $u$ is the normalized actuation force. $\hat{y}^+$ is the hopper height above ground just after the change in ground height in the `step` phase and $\hat{y}^-$ is the height above ground immediately prior to `step`. Note that in this problem there are no continuous disturbances, $d = \emptyset$, but that the change in terrain height constitutes a discrete disturbance $\sigma_d = \Delta\hat{h}$.

The hybrid system phases of motion are shown in Figure 3. Once $\hat{y} < 0$ the system remains in contact until either $\hat{y}$ becomes positive again, or, since the ground can't pull down on the suspended mass, the ground reaction force drops below

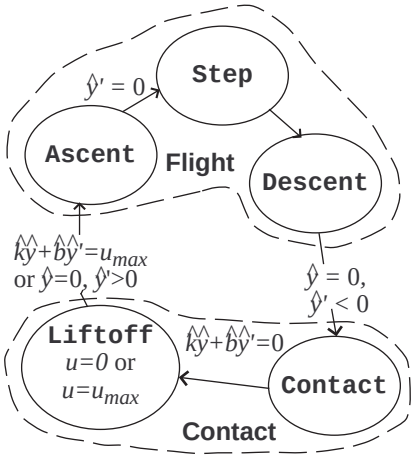



Fig. 3. Hopper flight and contact phases of motion. To accommodate a sudden change in height above ground the `Flight`phase is divided into `Ascent`, `Step`, and `Descent`. Furthermore, the contact phase is divided into normal `Contact` and `Liftoff`. The `Liftoff` phase addresses the problem of zero ground reaction force by fixing the value of the leg thrust. If leg thrust is zero hopper motion is ballistic. If leg thrust is $u_{max}$ the motion remains in contact.

zero. The ground reaction force, which is a function of the thrust, $u$ is given below.

$$\widehat{GRF} = -\hat{k}\hat{y} - \hat{b}\hat{y}' + u \tag{2}$$

The transition from `contact` to `ascent` phases of motion is complicated by the dependence on $u$. For $\hat{y}' > 0$, the motion in the region of state space between $\hat{k}\hat{y} - \hat{b}\hat{y}' > u_{min}$ and $\hat{y} < 0$ is either ballistic or oscillatory. Allowing $u$ to vary within this region may cause undesirable thrashing between modes. We resolve this dilemma by introducing a transition mode, `liftoff`, which fixes the value of $u$. To maximize liftoff vertical velocity we fix $u = u_{max}$. To minimize liftoff velocity we fix $u = u_{min}$. The appropriate value for $u$ is determined as we proceed through the maximal controlled safe invariant algorithm.

## III. Hopper Safety and Control Synthesis Approach

During contact phases of motion, the hopper is safe if the leg remains within joint constraint limits. For this problem we have assumed that the leg can safely compress to 50% of its unsprung length, hence $\hat{y} > -0.5$ is safe. During flight phases of motion the hopper is safe if it clears obstacles and avoids stumbling. Since we adjust the ground height during the `step` phase the hopper remains safe if $\hat{y} > 0$ in `step`. In general, the safe set, $F$, is one or more subsets of the discrete and continuous state space. A complete description of the safe set also requires bounds on position and velocity for `ascent` and `descent` phases of motion but these are not limitations on the hopper.

The ability to remain within the safe set over time depends on the control and disturbance inputs to the system. We consider both continuous time control and disturbance inputs, $u$ and $d$, and discrete event control and disturbance inputs, $\sigma_u$ and $\sigma_d$. The control and disturbance inputs are bound within prescribed limits. Tomlin, et. al. (2000, [14]) pose the following controller synthesis problem:

> *Given a safe set F, determine (i) the maximal invariant safe set contained in F, and (ii) the controller which renders this set invariant*

A subset of the state space is controlled invariant if there exists a controller which guarantees that any execution that begins in the subset remains in that subset.

The algorithm proposed by Tomlin, et. al. starts with an initial safe subset, $W^0 = F$, and goes backwards in time determining the region of state space that will remain within $F$ despite antagonistic disturbance inputs. The algorithm is a form of dynamic programming [2] and iterates until a controlled invariant subset of $F$ is found, if one exists.

The backward time propagation is both continuous and discrete. For discrete event propagation Tomlin, et. al. define two operators: $Pre_1(\cdot)$ and $Pre_2(\cdot)$. $Pre_1(\cdot)$ is the *controllable predecessor* operator. For a given set of continuous and discrete states, $K \subset Q \times X$, $Pre_1(K)$ is the set of states which force a discrete event transition that remains within $K$. The *uncontrollable predecessor* operator, $Pre_2(K^c)$ contains all states in the complement of $K$, $K^c$, as well as all states from which disturbances $(d, \sigma_d)$ can possibly force the state outside of $K$.

For the continuous portion of the algorithm Tomlin, et. al. define the *Reach* operator. $Reach(G, E)$ describes those states from which for all control inputs, $u$, there exist disturbance, $d$, such that the state trajectory can be driven to $G$ without reaching the desired "escape" set $E$. In the operator expression, $G$ is the set of states which are unsafe or can be made unsafe during a discrete transition, *i.e.* $Pre_2(K^c)$, and $E$ is the set of states which remain safe through a discrete transition, *i.e.* $Pre_1(K)$.

The algorithm that satisfies the controller synthesis problem posed by Tomlin, et. al. is given below.

**Maximal controlled safe invariant for hybrid systems**

**initialization:** $W^0 = F, W^{-1} = \emptyset, i = 0.$
**while** $W^i \neq W^{i-1}$ and $W^i \neq \emptyset$ **do**
$W^{i-1} = W^i \setminus Reach(Pre_2((W^i)^c), Pre_1(W^i))$
$i = i - 1$
**end while**

The first iteration through the **while** loop removes all states for which a disturbance can force the system outside of $F$ without first causing a transition between modes that remain within $F$. Subsequent steps iterate to remove states that can be forced outside of $F$ after transitions between modes. The index, $i$, decreases with each step to indicate propagation in backwards time. At each iteration $W^{i-1} \subseteq W^i$ so that the set $W^i$ decreases monotonically. The algorithm continues until either a fixed point, $W^{i-1} = W^i \triangleq W^*$, or a null solution, $W^{i-1} = \emptyset$, is reached.

Implementation of the maximal controlled invariant algorithm requires determination of the $Pre_1(K)$ and $Pre_2(K^c)$

set of states and calculation of the *Reach* operator. $Pre_1(K)$ and $Pre_2(K^c)$ are determined at discrete transitions between phases. The *Reach* operator is determined during the continuous time propagation within a phase of motion. Thus, the algorithm progresses phase by phase going backwards in time.

Tomlin, et. al. solve the *Reach* operation using an interconnected pair of Hamilton–Jacobi equations. The resulting feedback control is least restrictive in the sense that control is only applied to avoid the unsafe set, $(W^i)^c$. Since the hopper example is only second order, the algorithm can be applied semi-analytically and shown graphically. In fact, for a large class of problems, explicit solution of the Hamilton–Jacobi equations is not required [7].

### A. *The* Reach *Operation*

Tomlin, et. al. [14], characterize the reach operation as the solution to two nested optimizations. One optimization is to avoid the unsafe states, $G = Pre_2((W^i)^c)$. The other optimization problem is to achieve the desired escape set, $E = Pre_1(W^i)$. Both optimizations can be cast as pursuit–evasion games.

The optimizations require value functions: $J_G(x,t)$ is the value function for avoiding the set of unsafe states, $G$, and $J_E(x,t)$ is the value function to achieve the desired escape set, $E$.

$$J_G(x,u,d,t) = \phi_G(x(t_f)) \quad (3)$$

$$\text{where} \quad \begin{cases} \phi_G(x) < 0 & \text{for } x \in G \\ \phi_G = 0 & \text{for } x \notin G \end{cases}$$

$$J_E(x,u,d,t) = \phi_E(x(t_f)) \quad (4)$$

$$\text{where} \quad \begin{cases} \phi_E(x) < 0 & \text{for } x \in E \\ \phi_E = 0 & \text{for } x \notin E \end{cases}$$

The value functions depend only on functions of the continuous state vector, $x$, at the terminal time, $t_f$, which is arbitrary but fixed. To enforce transitions between modes we impose the terminal boundary constraints $\psi_{Exit}(x(t_f)) = 0$ and $\psi_{Init}(x(t_0)) = 0$. For $t < t_f$, $J_G$ and $J_E$ are functions of the continuous state vector, $x$, the control and disturbance inputs, $u$ and $d$, respectively, and time, $t$. The state evolves according to $\dot{x} = f_q(x,u,d)$ and we assume that the control and disturbance inputs are separable so that $f_q(x,u,d) = f_{q_1}(x,u) + f_{q_2}(x,d)$.

The hopper wishes to maximize the value for $J_G$ and minimize the value of $J_E$ while the environment wishes the opposite. Optimal values for $J_G$, $J_E$, $u$, and $d$ are denoted with an asterisk.

$$J_G^*(x,t) = \max_{u\in U}\min_{d\in D} J_G(x,u,d,t) \quad (5)$$

$$u^* = \arg\max_{u\in U}\min_{d\in D} J_G(x,u,d,t) \quad (6)$$

$$d^* = \arg\min_{d\in D} J_G(x,u^*,d,t) \quad (7)$$

$$J_E^*(x,t) = \min_{u\in U}\max_{d\in D} J_E(x,u,d,t) \quad (8)$$

$$u^* = \arg\min_{u\in U}\max_{d\in D} J_E(x,u,d,t) \quad (9)$$

$$d^* = \arg\max_{d\in D} J_E(x,u^*,d,t) \quad (10)$$

Assuming $J_G^*$ and $J_E^*$ exist, are continuous, and contain continuous first and second derivatives for all points of interest in the $(x,t)$ space, $J_G^*$ and $J_E^*$ can be found from solving Hamilton–Jacobi partial differential equations. However, as originally formulated $J_G$ and $J_E$ depend only on the value of $x(t)$ at time $t = t_f$. This leaves the possibility that $x(t) \in G$ for $t < t_f$, which is undesirable. To address this possibility Tomlin, et. al. [14] define the interconnected set of optimal Hamiltonians $H_G^*$ and $H_E^*$.

$$H_G^*(x, \frac{\partial J_G^*}{\partial x}) = \begin{cases} 0 \text{ for } \{J_E^*(x,t) < 0\}, \text{ else} \\ \max_{u\in U}\min_{d\in D} \frac{\partial J_G^*}{\partial x} f_q(x,u,d) \end{cases} \quad (11)$$

$$H_E^*(x, \frac{\partial J_E^*}{\partial x}) = \begin{cases} 0 \text{ for } \{J_G^*(x,t) < 0\}, \text{ else} \\ \min_{u\in U}\max_{d\in D} \frac{\partial J_E^*}{\partial x} f_q(x,u,d) \end{cases} \quad (12)$$

$J_E^* < 0$ if $x(t)$ can be forced to reach the escape set, $E$, despite worst case disturbances. Then the optimal Hamiltonian, $H_G^*$, is zero. Similarly, $J_G^* < 0$ if $x(t)$ can be forced to the unsafe condition $G$ despite all efforts by the control, and the optimal Hamiltonian, $H_E^*$, is zero. The interconnected set of Hamilton–Jacobi equations is given below.

$$-\frac{\partial J_G^*(x,t)}{\partial t} = \begin{cases} H_G^*(x, \frac{\partial J_G^*}{\partial x}) \text{ for } J_G^*(x,t) = 0 \\ \min\{0, H_G^*(x, \frac{\partial J_G^*}{\partial x})\}, \text{ otherwise} \end{cases} \quad (13)$$

$$-\frac{\partial J_E^*(x,t)}{\partial t} = \begin{cases} H_E^*(x, \frac{\partial J_E^*}{\partial x}) \text{ for } J_E^*(x,t) = 0 \\ \min\{0, H_E^*(x, \frac{\partial J_E^*}{\partial x})\}, \text{ otherwise} \end{cases} \quad (14)$$

Our approach instead is to consider the $J_G$ and $J_E$ optimizations separately. This is feasible provided that $x(t) \notin G$ for any time prior to the mode transition at time $t_f$. Assume that the boundary of the unsafe states, $\partial G$ is smooth and convex (or can be approximated by a finite set of possibly intersecting convex surfaces). If we assume some optimal trajectory so that at time $t$ the state vector $x^*(t)$ is on the surface of $\partial G$ but does not enter (or emerge) from $G$ over a positive or negative differential time step, then it must be that $f_q(x^*,u^*,d^*)$ is perpendicular to the surface normal, $\frac{\partial G}{\partial x}$. Also at this point $J_G^*(x^*,t) = 0$, and the gradient, $\frac{\partial J_G^*}{\partial x}$, is parallel to the surface normal, $\frac{\partial G}{\partial x}$. From equations 11 and 13 the gradient, $\frac{\partial J_G^*}{\partial x}$, must remain perpendicular to $f_q(x^*,u^*,d^*)$ since the value function is not an explicit function of time and $\frac{\partial J_G^*}{\partial t} = 0$.

Thus, our approach is to first consider the $J_G$ optimization by determining those points along $\partial G$ where $f_q(x^*,u^*,d^*)$ is perpendicular to the surface normal. When $u^* \in \{u_{max}, u_{min}\}$ or $d^* \in \{d_{max}, d_{min}\}$ we have reached an extremum point and must propagate in time. Going backwards in time we pick $u^*$ or $d^*$ to respectively maximize or minimize $H_G^*$ ignoring the $J_E < 0$ condition in equation 11. Since $\frac{\partial J_G^*}{\partial x}$ remains perpendicular to $f_q(x,u^*,d^*)$ we need not calculate $J_G$ but rather determine $\frac{\partial J_G^*}{\partial x}$ as the normal to the propagated trajectory of $x^*(t)$ in state space. For problems of dimension N, we require the constraint boundary to be dimension N-1 and the set of extrema points to form an N-2 dimensional curve on that surface. The propagated trajectory of over that entire curve forms an N-1 dimensional surface so the surface normal, $\frac{\partial J_G^*}{\partial x}$, is easy to determine.

We follow a similar approach for the $J_E$ optimization except that the propagation begins along the border of the desired escape set, $\partial E$ and we use the terminal constraint $\psi(x(t_f)) = 0$ to force a mode transition at time $t_f$. The time propagation is backwards time only and we disregard the $J_G < 0$ condition in equation 12. The combined solution is the intersection of the $J_E$ and $J_G$ optimizations. The approach has the advantage that the $J_G$ optimization need be done only once at initialization and only the $J_E$ optimization is performed during the backwards iteration.

An example solution for the hopper problem is described in the following section. However, the example exposes two issues that the controlled invariant algorithm must accommodate. One issue arises when phase transitions are a function of the continuous time control or disturbance input. Since these inputs are allowed to vary at any instant in time, they can cause thrashing between modes preventing the divergence of time [11]. This problem can be avoided by properly structuring the hybrid dynamics. Another issue arises when the control or disturbance inputs can force continuous time dynamics to a singular point, *e.g.* $\dot{x} = f(x, u, d) = 0$. At a singular point propagation in backwards time is undefined. Again the problem must be structured to avoid this condition, but the best means of doing so is problem specific.

### B. Leg Thrust and Transition to `ascent`

The transition from `contact` to `ascent` occurs when either the leg loses contact with the ground ($\hat{y} = 0$) or when the ground reaction force, given in equation 2, goes to zero. However, the ground reaction force is a function of the control input, $u$. Thus, it is possible over some regime to switch back and forth from `contact` phase motion (a spring-mass-damper system) to `ascent` phase (ballistic motion).

The situation is illustrated in the phase plane diagram in Figure 4. Since $0 \leq u \leq u_{max}$ the transition is bound by the lines at $\hat{k}\hat{y} + \hat{b}\hat{y}' = 0$, $\hat{k}\hat{y} + \hat{b}\hat{y}' = u_{max}$, and $\hat{y} = 0$. The width of this transition region is determined by the strength of the normalized damping coefficient, $\hat{b}$, with respect to the leg's stiffness, $\hat{k}$. Assume we are going forward in time starting from point A1. If $u = 0$ the system switches from `contact` to `ascent` and follows the lower bound trajectory to point A2. If, however, $u = u_{max}$ the system remains in `contact` and follows the upper bound trajectory to point A2'. Intermediate trajectories, depicted by the cross–hatched area, can be achieved by modulating the control. The situation going backward in time is shown starting in `ascent` at point B1. If $u = 0$ the system remains in `ascent` and follows the upper bound trajectory to point B2. If $u = u_{max}$ the system switches to `contact` and follows the lower bound trajectory to B2'. It is clear from the figure that the bounding behavior is at either $u = 0$ or $u = u_{max}$ so that intermediate or varying values for for $u$ need not be considered.

It is useful to define a transitional mode, `liftoff`, to eliminate dependence on the control variable, $u$. The transitional mode "locks" the value of the control to $u \triangleq 0$ or $u \triangleq u_{max}$. The `liftoff` mode transitions are illustrated in Figure **??**. The transition from `contact` to `liftoff` occurs

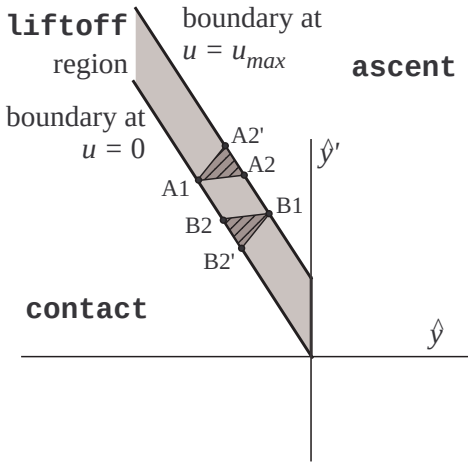



Fig. 4. The transition between contact and ascent phases is influenced by the control, $u$. Going forward in time from point A1 the system will enter ascent at a point between A2 and A2'. Going backwards in time from point B1 the system will enter contact between point B2 and B2'. The bounding behavior for the contact/ascent transition is obtained at either $u = 0$ or $u = u_{max}$.

at the $\hat{k}\hat{y} + \hat{b}\hat{y}' = 0$ manifold. The transition from `liftoff` to `ascent` occurs at the $\hat{k}\hat{y} + \hat{b}\hat{y}' = u_{max}$ for $\hat{y} < 0$ and at $\hat{y} = 0$ otherwise. For $u \triangleq 0$ in `liftoff`, the state trajectory follows the parabolic path of a ballistic system. For $u \triangleq u_{max}$ in `liftoff`, the state trajectory follows the logarithmic spiral of a damped second order system.

The appropriate value for $u$ during `liftoff` depends on context. Referring again to Figure 4, suppose point B1 is the initial velocity required to achieve a minimum height during `ascent`. Then, $u = u_{max}$ maximizes the safe range of exit conditions from `contact` since along the boundary at $u = 0$ all points above B2' will meet this constraint. On the other hand, if B1 is the maximum safe velocity upon entering ascent, then $u = 0$ maximizes the safe range of contact exit conditions since along the boundary at $u = 0$ all points below B2 will meet the maximum height constraint.

### C. Singularity Avoidance

The second difficulty in applying the maximal safe invariant algorithm to the hopper problem is associated with the stable point in the contact region. At $u = 0$ the hopper equilibrium in contact is $\hat{y} = -1/\hat{k}$ and $\hat{y}' = 0$. If the initial descent velocity is low enough and the thrust remains at zero, the system will settle to equilibrium without ever leaving the contact phase. Technically, the system is safe, but it is also inert.

A more fundamental problem is that at the stable point backwards time propagation is undefined. All points in the neighborhood of the stable point eventually reach the stable point so starting from the stable point and propagating backwards in time yields infinite solutions. For this reason, the algorithm for the maximal invariant safe set must take steps to avoid the singularity at $f_q(x, u, v) = 0$.

Figure 5 shows the range of equilibrium points for $0 \leq u \leq u_{max}$ and a position and velocity trajectory for a hopper in

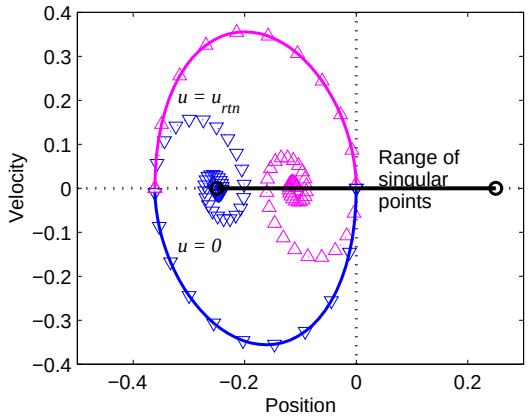



Fig. 5. Singularity Avoidance (SA). The range of singular points from $u = 0$ to $u = u_{max}$ is given by the horizontal bold line. To avoid these points during backward propagation a boundary is defined at (0, 0) with $u = 0$ and projecting forward in time until maximum deflection. The control for the return trajectory, $u_{rtn}$, is set so that the hopper just reaches the transition from `contact` to `ascent`.

contact. In this example the hopper has a normalized spring constant, $\hat{k} = 4$ and a damping coefficient $\hat{b} = 1$, which yields a non-dimensional damping ratio of 0.25. Starting from rest at $\hat{y} = 0$ and $\hat{y}' = 0$, the hopper falls under the influence of gravity with zero thrust and follows the solid line trajectory marked with circles. Without any thrust, the state trajectory would spiral in to the equilibrium position at $u = 0$ marked on the figure. In fact, without any thrust the system will eventually settle to this equilibrium from any initial condition. If, however, a thrust is applied at maximum deflection the equilibrium changes and the hopper follows the solid line trajectory marked with diamonds. The minimum level thrust that will return the hopper back to the starting position, $u_{rtn}$, is a function of the non-dimensional damping ratio, $\zeta$, and is given in equation 15 below.

$$u_{rtn} = \frac{1 - e^{-2\pi\zeta/\sqrt{1-\zeta^2}}}{1 + e^{-\pi\zeta/\sqrt{1-\zeta^2}}}, \text{ where } \zeta = 0.5\hat{b}/\sqrt{\hat{k}} \qquad (15)$$

There are an infinite number of ways of applying thrust to avoid the singular point. The way we take to be 'least restrictive' is to follow the $u = \{0, u_{rtn}\}$ trajectory in Figure 5 once that boundary is reached. The path is least restrictive in the sense that it does not restrict the initial entrance or exit velocity for the `contact` mode (we have not considered `liftoff` mode in this discussion but its effect is assumed to be minor). In effect the boundary in Figure 5 defines a phase transition which constrains the control to follow the boundary path. It is worth emphasizing, however, that many other schemes are possible and perhaps more desirable in practice. For example, rather than follow the boundary path, the control logic could be to use maximum thrust while inside the singularity avoidance zone. This would certainly eliminate singularities but might redefine the singularity avoidance zone boundary.

TABLE I
VERTICAL HOPPER NON-DIMENSIONAL PARAMETERS IN FIGURE 6

| *Parameter* | *Value* |
|---|---|
| stiffness, $\hat{k}$ | 4.0 |
| damping coefficient, $\hat{b}$ | 1.0 |
| damping ratio, $\zeta$ | 0.25 |
| max thrust, $\hat{u}_{max}$ | 2.0 |
| joint constraint, $\hat{y}_{min}$ | -0.5 |
| max step, $\Delta\hat{h}_{max}$ | 0.3 |
| min step, $\Delta\hat{h}_{min}$ | -0.3 |
| max initial height, $\hat{y}_{init-max}$ | 2.0 |
| min initial velocity, $\hat{y}'_{init-min}$ | -3.0 |
| max initial velocity, $\hat{y}'_{init-max}$ | 3.0 |

## IV. RESULTS

Figure 6 is a plot of the maximal controlled invariant set for a vertical hopper. The hopper has a maximum thrust capability equal to twice its weight, and a leg with normalized spring stiffness of 4 and a damping coefficient of 1.0. The hopper does not have a continuous disturbance input, but does have a discrete disturbance input, $\Delta\hat{h}$, during the `Step` phase. The non-dimensional parameters are summarized in Table I. The damping coefficient is equivalent to a damping ratio, $\zeta$, of 0.25. The joint constraint limit is at -0.5 and the maximum and minimum initial height and velocity constraints form a closed boundary for the initial safe set.

The horizontal and vertical axes of the plot in Figure 6 are the normalized height, $\hat{y}$, and the normalized vertical velocity, $\hat{y}'$, respectively. The state space is divided into regions corresponding to the hopper phases of motion. `Liftoff` is the region between the diagonal broken lines in the upper left and bound on the right hand side by the line at $\hat{y} = 0$. The boundaries of the `Liftoff` phase are determined from equation 2 for the ground reaction force at $u = u_{min}$ and at $u = u_{max}$. The `Ascent` phase of motion covers the upper right portion of the state space from the `Liftoff` boundary and bound below by the line at $\hat{y}' = 0$. The `Descent` phase of motion covers the lower right quadrant of the state space bound by $\hat{y} \geq 0$ and $\hat{y}' < 0$. The `Contact` phase of motion covers the lower and center left portion of the state space bound by $\hat{y} \leq 0$ and the broken line at `Liftoff`. Finally, the `Step` phase is not shown explicitly since it is a discrete (instantaneous) phase, but occurs along the line $\hat{y}' = 0$ for $\hat{y} > 0$. In this example, the subsets of the continuous state space for the different phases of motion are disjoint (i.e. do not overlap), but that is not a general requirement.

### A. Initialization

The first step to the algorithm is to determine the initial safe set, $W^0$, or conversely the unsafe states, $G$. The safety constraints are that the hopper clear any obstacle during the `Step` phase and that the hopper avoid the $\hat{y}_{min}$ joint constraint limit. Thus, $W^0$ is the set of states $\hat{y} \geq -0.5$ in `Contact` and `Liftoff` phases, the set of states $\hat{y} > 0$ in `Step` phase and all legal states within `Ascent` and `Descent` phases. The border of the unsafe states, $\partial G$, is the vertical line at $\hat{y} = -0.5$.

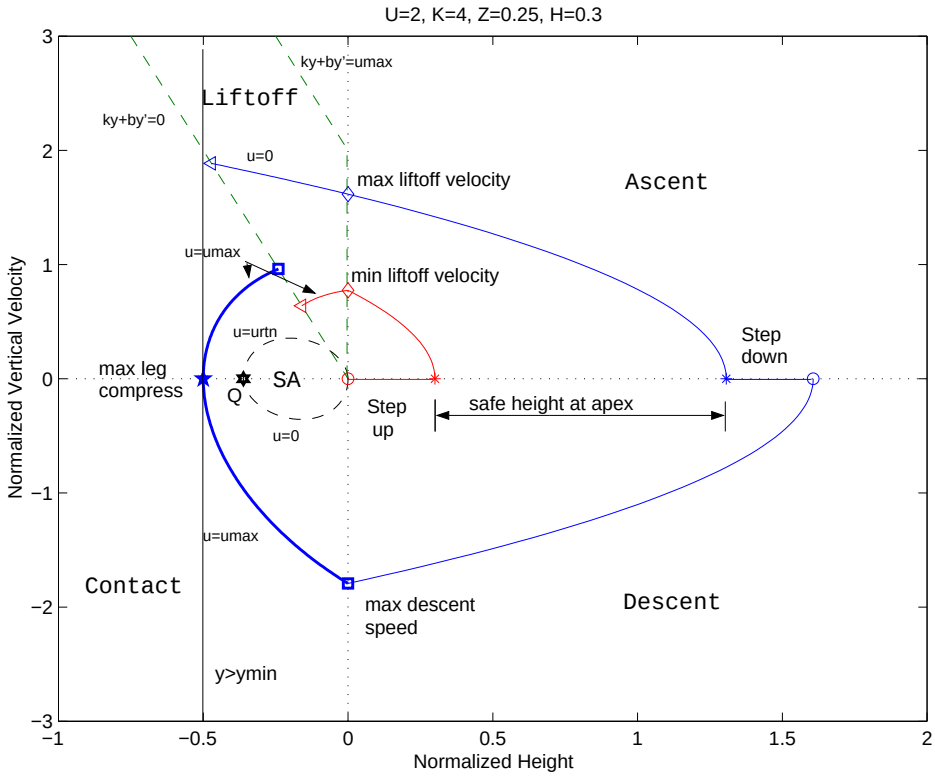



Fig. 6. Maximal safe invariant set for a hopper with $\hat{k} = 4$, $\zeta = 0.25$, and $\Delta\hat{h}_{max} = 0.3$. The diagram shows the `Contact`, `Descent`, `Ascent`, and `Liftoff` phases of motion as well as the singularity avoidance boundary of the `Contact` phase and the maximum compression constraint at $\hat{y} = -0.5$. In this example the algorithm for finding the maximum invariant safe set terminates after a single iteration through the phases of motion.

*1)* ***Contact*** *Phase:* Figure 6 marks the extremum point along $\partial G$ ($\hat{y} = -0.5, \hat{y}' = 0$) with a star. At this point the derivative vector $f(x,u) = [0u]^T$ is perpendicular to the outward normal $fracG\partial x = [10]^T$ for any non-zero value of $u$. The desired escape set, $E$, is the transition to `Liftoff` mode at $\hat{k}\hat{y} + \hat{b}\hat{y}' = 0$. The initial conditions for `Contact` are along the dotted line at $\hat{y} = 0$ and $\hat{y}' \leq 0$. The singularity avoidance region (`SA`) is denoted by a broken line, with the point Q, the point of maximum compression starting from rest, marked by a hexagram symbol.

Starting at the extremum point (the star) we determine the optimal thrust, $u^*$ that maximizes the Hamiltonian $\frac{\partial J_G}{\partial x} f(x,u)$. Over a finite time step we obtain $u^* = u_{max}$. Propagating backwards in time generates the heavy curve from the star at "max leg compression" and terminating at square symbol for "max descent speed" on the `Contact` phase set of initial conditions. The `Contact`states below this curve will propagate to violate the maximum leg compression constraint and therefore belong to *Reach(G,E)*. The contact states on and above this curve remain safe. The set of safe states is maximized if the backwards time propagation is at $u = u_{max}$.

For forward time propagation the sense of the optimization in equation 13 changes and again the optimal control over a finite time step is $u^* = u_{max}$. Propagating forwards in time generates the heavy curve from the star at "max leg compression" to the square symbol at the `Liftoff` boundary. The square denotes the maximum safe velocity entering the `Liftoff` phase. The `Contact`states above this curve can only be reached by passing through $G$ and therefore belong to *Reach(G,E)*. The contact states on and below this curve remain safe. The set of safe states is maximized if the forwards time propagation is at $u = u_{max}$.

The forward and backwards time propagations redefine the safe initial and exit conditions for the `Contact`phase. `Contact` phase safe initial conditions for are those states along the `Contact-Descent` boundary above the"maximum descent speed" point shown in Figure 6. The `Contact` phase safe exit conditions, $E$, are those states along the `Contact-Liftoff` boundary below the square square symbol also shown in Figure 6.

Control, $u$, is unrestricted except at the boundaries of the `Contact` and singularity avoidance regions. For example, if the initial contact velocity is somewhere between 0 and maximum descent speed, the control, $u$, can assume any legal value unless and until the state trajectory intersects the safety or singularity avoidance boundary. At the safety boundary the control is set to $u = u_{max}$ at least until hopper velocity is positive. If the state trajectory intersects the `SA` boundary then control is set to $u = 0$ or $u \geq u_{rtn}$ depending on where the intersect occurs.

### B. Backwards Chaining

The Contact phase is the only continuous time phase with a safety constraint. Completing the *Reach(G,E)* operation on Contactcompletes the initialization phase of the algorithm. However, the *Reach* operation redefined the range of safe initial and exit conditions for Contact. These changes propagate through to other modes during the backward chaining portion of the algorithm.

*1) Descent Phase:* Since the Descent phase immediately precedes the Contact phase, the maximum and minimum safe initial velocities for Contact define the desired escape set, $E$, for Descent. Here we propagate backwards in time from "max descent speed" to the Descent–Step phase boundary. Since Descent is purely ballistic, there is no control and the backwards propagation is straightforward. Figure 6, only shows the propagation from "max descent speed" because the minimum safe exit speed is zero which immediately transitions to Step. Safe initial conditions for Descent are bound by the circle markers on the $\hat{y}' = 0$ line in Figure 6.

*2) Step Phase:* The safe exit conditions for Step are the safe initial conditions from Descent. Step is a discrete mode with no continuous time propagation. However, the environment acts to effect an instantaneous change, $\hat{y}^+ = \hat{y}^- + \Delta\hat{h}$. If $\Delta\hat{h}$ is negative, the height above ground decreases and the environment causes a "step up" change in terrain height. To avoid collision ($\hat{y}^+ < 0$) the hopper must be above the maximum step up height prior to Step. Conversely, if $\Delta\hat{h}$ is positive the environments causes a step down change in terrain height. A step down increases landing speed potentially over stressing the leg. The maximum height just prior to the Step is the height from which the hopper can safely fall assuming a step down change of $\Delta\hat{h}_{max}$. In Figure 6, the safe range for $\hat{y}$ prior to the Step phase is between the asterisks on the $\hat{y}' = 0$ line.

*3) Ascent Phase:* The Ascent exit conditions are equal to the Step phase initial conditions. The Ascent phase initial conditions are the are found by propagating a ballistic trajectory backwards in time. In Figure 6, both upper and lower bound ascent phase trajectories terminate at the Liftoffboundary along $\hat{y} = 0$.

*4) Liftoff Phase:* The Liftoff exit conditions are between the maximum and minimum liftoff velocities at $\hat{y} = 0$ and marked by diamond symbols in Figure 6. From the minimum liftoff velocity, we propagate backwards in time assuming $u = u_{max}$. The intersection at the initial condition boundary of the Liftoff region (lower triangle symbol) marks the minimum velocity in Contact that could safely clear a step up of $\Delta\hat{h}_{max}$. From the maximum liftoff velocity, we propagate backwards in time assuming $u = 0$. The intersection at the initial condition boundary of the Liftoff region (upper triangle symbol) marks the maximum safe velocity in Contact that will avoid the leg compression constraint upon returning to Contact with a worse case drop in the terrain. Since the maximum safe initial velocity for Liftoffis greater than the maximum contact phase velocity (square symbol at the Contact–Liftoff boundary, the maximum Liftoffvelocity is not a constraint on the system and the system has a feasible solution.

### C. Algorithm Termination

We have now taken the algorithm for determining the maximal controlled invariant set through one complete iteration to $W^{-1}$ which is shown in Figure 6. The algorithm continues with a second iteration through the discrete modes again starting with the Contact phase.

Starting at the Contact–Liftoff boundary, the maximum contact velocity, marked by a square symbol along the boundary, remains unchanged. The time propagation from this point is along the heavy solid curve and need not be repeated.

The minimum safe exit velocity for the Contact phase is the left facing triangle associated with the minimum liftoff velocity. Propagating backwards in time from this point the optimal value for $u$ is $u^* = u_{max}$ but the curve intersects the singularity avoidance (SA) boundary (intersection not shown). At the intersection point, the control is set at $u = u_{rtn}$ and the at $u = 0$ to follow the SA boundary backwards in time to Contact phase initial conditions at point $\hat{y} = 0, \hat{y}' = 0$.

Thus, the Contact phase initial conditions are unchanged between the first and second iterations of the maximal safe invariant algorithm. Since the Contact phase initial conditions are the same between iterations, the Descent phase exit and initial conditions are the same. Likewise the Step, Ascent, and Liftoff phases remain unchanged. In this example, the algorithm reaches steady state after a single iteration and terminates.

In general, however, the algorithm will require more than two iterations through the phases of motion prior to termination. Figure 7 is a phase diagram of the same hopper in 6, but with $\Delta\hat{h}_{max} = 0.4$ and $\Delta\hat{h}_{min} = -0.4$ rather than $\pm 0.3$. The first iteration through the phases of motion follows the previous discussion except that the higher step up requires a higher minimum liftoff velocity at the start of the Ascent phase. After the first iteration through Liftoff, the Contact phase minimum safe exit velocity is marked by the lower most triangle on the Contact–Liftoff border. Propagating backwards in time from this point to start the second iteration of the algorithm, one finds that state trajectory just misses the singularity avoidance region. The backwards propagation is at $u^* = u_{max}$ for positive Contact velocities, and at $u^* = 0$ for negative Contact velocities. This control history minimizes the initial contact velocity and maximizes the range between minimum and maximum descent speeds. However, after the second iteration through the Contact phase, the minimum initial contact speed, marked by the topmost right facing triangle along $\hat{y} = 0$ is non-zero. The change in Contact phase initial conditions requires subsequent iteration through the phases of motion.

Continuing backwards in time, the minimum initial height for the Descent phase, marked by a circle in figure 7, moves out to the right from $\hat{y} = 0$. The higher minimum initial Descent height, the higher the minimum initial height at the Step phase and the higher the minimum liftoff velocity. The iterations generate an outward spiral shown in the figure.

Finally, the minimum initial `Liftoff` velocity, marked by left facing triangles at the `Contact-Liftoff` border, exceeds the maximum contact exit velocity, marked by a square in the figure. The algorithm terminates because the invariant safe set is empty.

The iteration illustrated in Figure 7 can be interpreted in forward time as a hopper trying to climb stairs. Assume the hopper starts at the bottom of the stair case in `Descent` at the maximum safe height above ground. The hopper lands at maximum descent speed and immediately applies full thrust (following the bold solid curve) to avoid violating the maximum leg compression constraint. The hopper continues maximum thrust through `Contact` and `Liftoff` phases and enters `Ascent` just below the 'minimum liftoff velocity' marked on the figure. At the apex of flight the hopper's height above ground is indicated by the right most asterisk in the figure. Then the step up causes the hopper's height relative to ground to drop by 0.4 to the point marked by an open circle next to 'Step up' on the figure. At the top of the first step, the hopper lands at a speed just below the 'minimum descent speed'. Back in `Contact`, the hopper waits for maximum leg deflection ($\hat{y}' = 0$) before applying thrust at $u = u_{max}$. However, the liftoff velocity is lower than after initial contact and the hopper lands on the second step with even less velocity than the first. The state space trajectory spirals inward (in forward time) until the hopper fails to clear the fourth step.

Each iteration through the phases of motion reduces the set of safe states. However, the ultimate outcome is not necessarily null. Figure 8 shows a result where the maximal safe invariant converges to a non-empty set. In this example the hopper leg has a non-dimensional stiffness of 10 and a non-dimensional damping ratio of 0.05. The leg is stiffer and less damped than in the previous examples but has the same thrust capability. As in Figure 7, the maximum terrain variation between steps is 40% of the uncompressed leg length ($\Delta\hat{h}_{max} = 0.4$). The hopper must land with sufficient speed to compress the leg so that it can apply thrust long enough to get the minimum required liftoff velocity. The minimum contact speed for the initial safe set of states is zero. However, by iterating through the phases of motion the initial minimum contact speed is increased until the difference between iterations is acceptably small. Starting with a contact speed greater than or equal to the minimum value, the hopper can safely climb an arbitrarily long stair case.

## V. RUGGEDNESS

The term 'ruggedness' is used to describe the maximum environmental disturbance our system can tolerate under worst case conditions and still operate safely. We use 'Ruggedness' rather than 'robustness' to avoid confusion with the many robust control techniques that consider both external and internal disturbances but do not consider worst case conditions in a game theoretic way. The ruggedness of the single legged hopper is a function of leg thrust, stiffness, and damping.

This study assumes $\Delta\hat{h}_{min} = -\Delta\hat{h}_{max}$ so that the terrain is generally level. Note that if the terrain was generally descending then $\Delta\hat{h}_{min} < -\Delta\hat{h}_{max}$ and conversely if terrain is generally ascending then $\Delta\hat{h}_{min} > -\Delta\hat{h}_{max}$. To determine ruggedness, the algorithm for the maximal controlled safe invariant is repeatedly applied but at different values for $\Delta\hat{h}_{max}$. If the safe set is non-empty, $\Delta\hat{h}_{max}$is increased by an increment. If the safe set is empty, $\Delta\hat{h}_{max}$is decreased until a non-empty safe set solution is found. In this study, the ultimate resolution for $\Delta\hat{h}_{max}$is 0.01 or one percent of the uncompressed leg length.

Hopper ruggedness for different leg thrust, stiffness, and damping is shown in Figure 9. The figure shows four subplots each with a different level for maximum leg thrust. The axes for each subplot are maximum terrain variability (that is, ruggedness) versus the non-dimensional damping coefficient and the curves in each subplot are for normalized spring stiffnesses of 2, 4, 6, 10, 14, and 20.

The results in Figure 9 show several general trends. First, ruggedness always increases with increasing leg thrust capability. Second, for a given leg thrust capability ruggedness increases with leg stiffness at low damping. Maximum ruggedness is achieved at high stiffness and near zero damping. But, at high stiffness ruggedness drops off very quickly with damping. Since all mechanical systems have some form of energy loss, this suggests that better performance will be achieved with moderate stiffness. In the hopper example we capture the energy loss with only viscous friction but real systems will have non-zero mass legs and also incur energy loss at ground contact.

An interesting trend in Figure 9 is that for systems with low to moderate stiffness, increasing damping up to an optimum value increases ruggedness. The lower the stiffness, the higher the optimum value for damping. The damping dissipates energy from landing allowing the hopper to tolerate a step down change in terrain height without violating the leg compression constraint. However, as damping increases beyond the optimum value ruggedness is limited by the energy dissipated during thrust when the hopper tries to achieve a minimum liftoff velocity. A final trend observed in Figure 9 is that at higher thrust levels the difference between optimal ruggedness for hoppers with different leg stiffness diminishes. Hopper capabilities become more and more dominated by control rather than passive dynamics.

To further understand the effect of damping, Figure 10 plots hopper ruggedness versus damping for different leg stiffnesses at $u_{max} = 4$. Ruggedness is compared against "obstacle limited" and "compression limited" behaviors which are shown to be limiting values for hopper ruggedness. The "compression limited" curve, shown as a dashed line in the figure, is determined from the maximum height from which the hopper can safely fall. It limits hopper ruggedness at low to moderate leg stiffness and low damping. The "obstacle limited" curve, shown as a dotted line in the figure, is determined from the maximum obstacle height the hopper can clear. It limits hopper ruggedness at moderate to higher levels of damping. Figure 11 shows the maximal safe set in state space for hoppers with a dimensionless spring constant $\hat{k} = 4$ but different damping for maximum ruggedness at the "obstacle limited", "compression limited", and intermediate behaviors.

For compression limited behavior, ruggedness is one half the

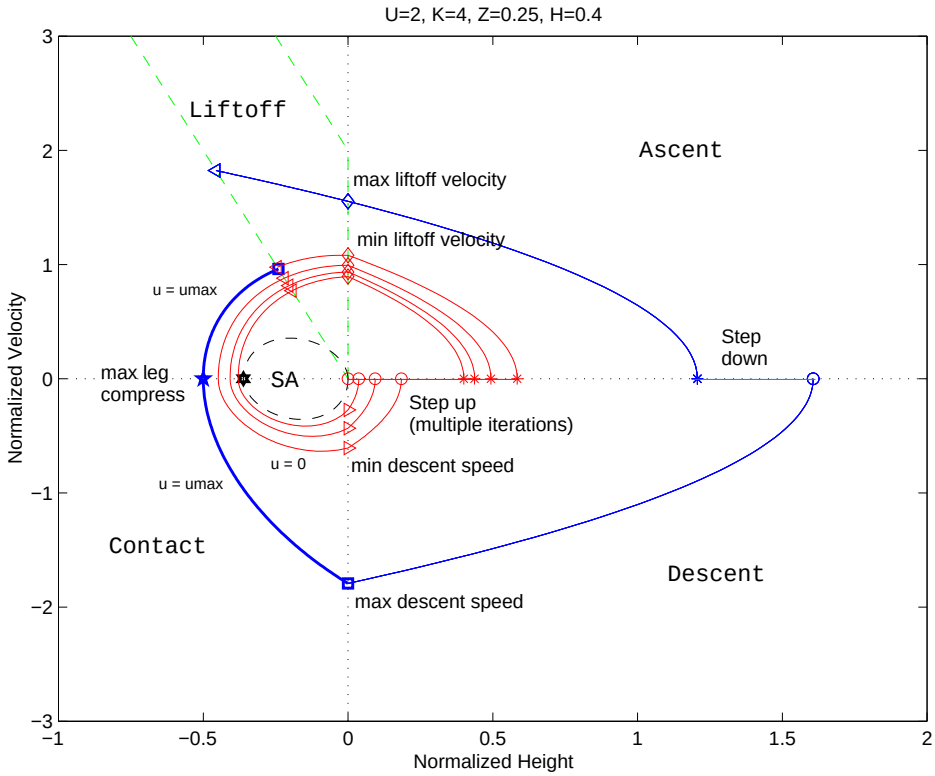



Fig. 7. Maximal safe invariant set for a hopper with $u_{max} = 2$, $\hat{k} = 4$, $\zeta = 0.25$, and $\Delta\hat{h}_{max} = 0.4$. In this example the algorithm for finding the maximum safe invariant set requires multiple iterations through the phases of motion. After four steps up the hopper fails to clear the obstacle and the safe invariant set is empty.

maximum height which the hopper can fall without violating the leg compression constraint. The compression limit is determined for a given set of leg parameters by assuming the leg at maximum compression is at rest and propagating the `contact` equations of motion backwards in time at $u = u_{max}$. This determines the maximum descent speed shown in Figure 11 (a). The maximum height above ground after the `step` phase is determined from the maximum descent speed. $\Delta\hat{h}_{max}$is one half the maximum safe height so a step up reduces height above ground to zero and a step down increases height above ground to the safe limit. Therefore, the maximum and minimum height prior to the `step` are equal. Likewise, the maximum and minimum liftoff velocities prior to `ascent` are equal. Extrapolating backwards in time from liftoff with $u = u_{max}$ (dotted curve between the triangle and hexagram), the state trajectory intersects the singularity avoidance boundary. Following along the singularity avoidance boundary in backwards time, the minimum velocity at contact is zero and the iteration for the maximal safe set terminates. Note that if the step down height were reduced in Figure 11 (a), the hopper could safely handle a larger step up so ruggedness is step down or *compression* limited.

For obstacle limited behavior, ruggedness is the maximum height the hopper can clear from initial contact at zero velocity, as shown in Figure 11 (c). Starting at $\hat{y} = 0$ and $\hat{y}' = 0$ and

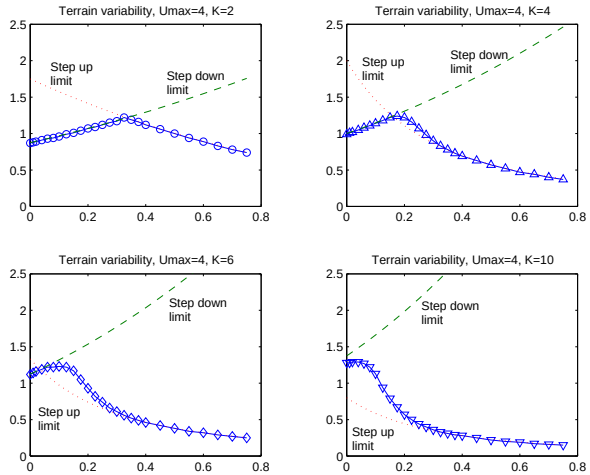



Fig. 10. Hopper ruggedness and "step up" and "step down" limits. The "step down" limit is the maximum drop in height between successive steps. When step down limited the hopper applies maximum thrust at first contact to avoid joint limits and increasing damping increases ruggedness. The "step up" limit is the maximum rise the hopper can sustain over a succession of steps. When step up limited, the hopper applies maximum thrust at full compression to jump as high as possible and decreasing damping increases ruggedness. Optimal damping occurs near where the maximum step height for "step down" and "step up" limited behaviors intersect.

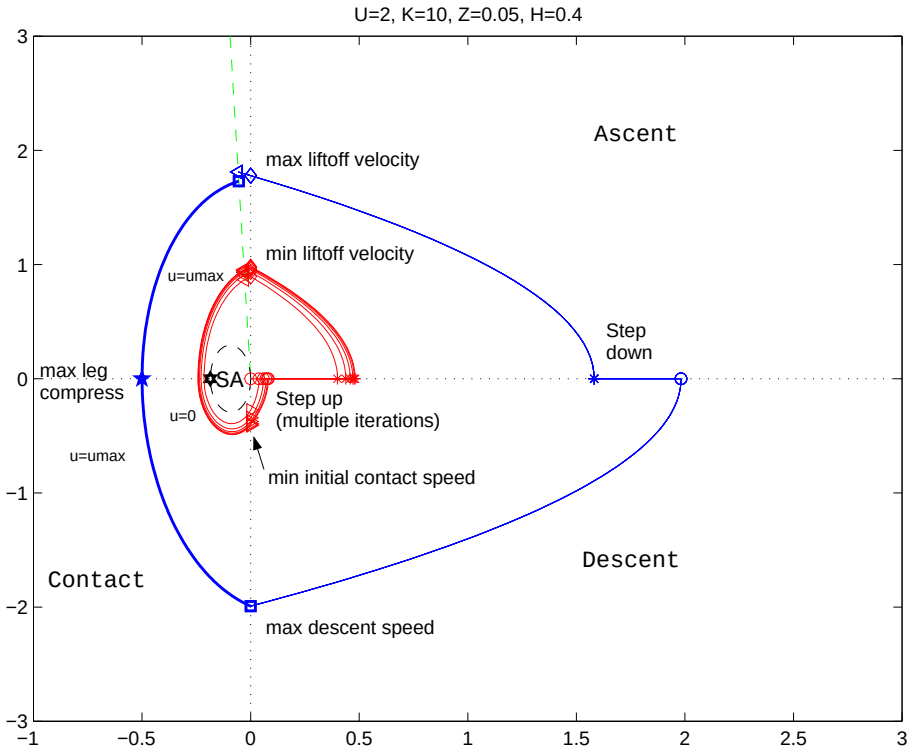



Fig. 8. Maximal safe invariant set for a hopper with $u_{max} = 2$, $\hat{k} = 10$, $\zeta = 0.05$, and $\Delta\hat{h}_{max} = 0.4$. The algorithm for the maximum safe invariant set requires multiple iterations but converges to a non-empty solution space. In this example the hopper requires non-zero landing speeds to compress the leg so that the thrust can act over a long enough distance to achieve the required liftoff velocity.

propagating forward in time with $u = 0$, the hopper falls under gravity and follows the singularity avoidance boundary until maximum compression ($\hat{y}' = 0$). At this point full thrust is applied to achieve the minimum safe liftoff velocity which determines $\Delta\hat{h}_{max}$. After a step up, the hopper contacts the ground at zero velocity and the iteration for the maximal safe set terminates. For larger $\Delta\hat{h}$ the hopper must land with non-zero velocity, but the hopper cannot provide enough thrust to overcome the energy lost to damping so that the hopper lands with lower velocity on subsequent steps and the state space trajectory spirals inward as shown in Figure 7. Note the large distance between maximum and minimum safe height in `ascent` in Figure 11(c). The hopper can safely handle a larger step down so ruggedness is step up or *obstacle* limited.

Figure 10 shows that at moderate levels of damping ruggedness falls somewhere between the obstacle and compression limited behaviors. State space boundaries for the maximal safe set under these conditions are shown in Figure 11 (b). Here, the damping is such that the hopper can sustain stair climbing with a non-zero landing velocity so ruggedness is greater than the step up or "obstacle" limit. However, the gap between maximum and minimum safe height in `ascent` shows that the hopper could safely handle a larger step down so ruggedness is less than the step down or "compression" limited level. For moderate leg stiffness, ruggedness is maximized at this intermediate behavior.

## VI. Extension to Planar Running

The approach demonstrated for the single legged hopper can be extended to more complicated problems. While detailed discussion is beyond the scope of this paper, the intent in this section is to show how the maximal safe invariant algorithm could be applied to planar running and to multi–legged systems. Solutions to these problems requires numerical tools rather than the semi-analytic approach taken here [7].

Extension from simple hopping to planar running complicates the hopper dynamics and the problem formulation for the maximal controlled safe invariant in several ways. First, additional states are required for horizontal and rotational motion and leg angle. If running with multiple legs, additional states are required not only for the angular position of each leg, but also to track the difference in terrain height between legs. Secondly, additional safety constraints are required to satisfy hopper orientation, hip joint limits, and non-slip contact. Finally, the change in terrain cannot be constrained to the apex of flight complicating the hybrid mode dynamics of the problem.

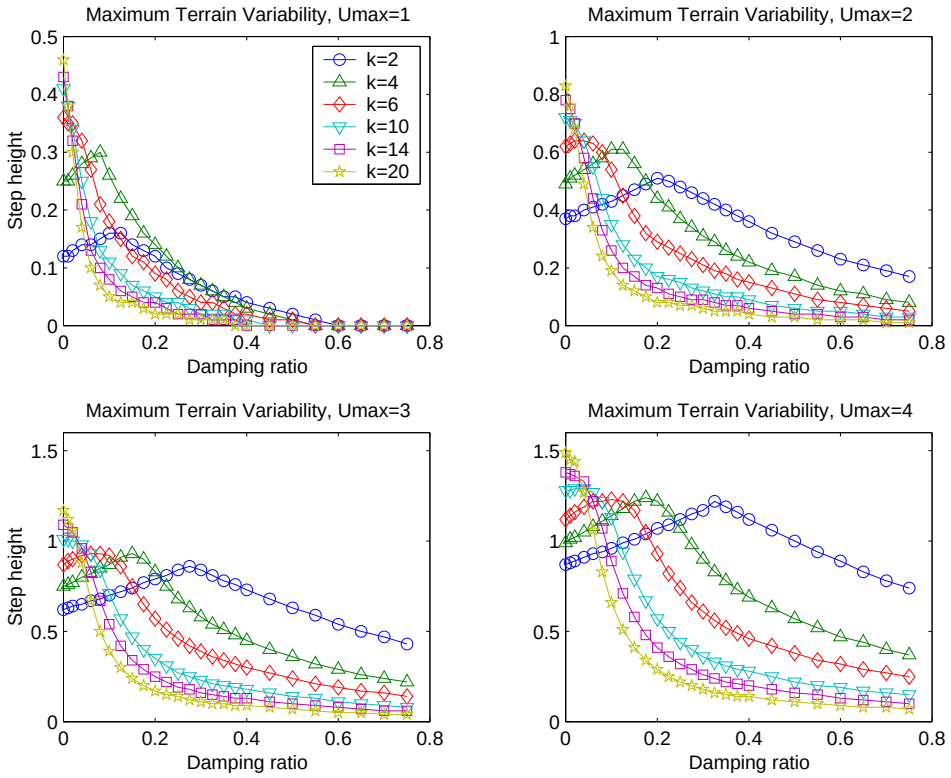
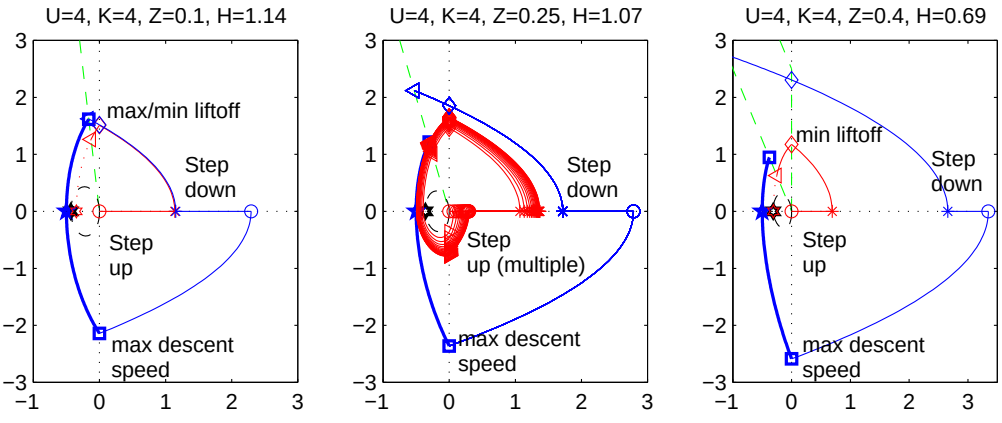



Fig. 9. Hopper ruggedness as a function of leg thrust, stiffness, and damping. Ruggedness is a measure of the maximum variation in terrain height between steps that the system can safely handle. For zero and very low damping then high leg stiffness is optimal. For moderate and higher levels of damping moderate to low leg stiffness is best. Ruggedness increases uniformly with leg thrust capability and sensitivity to mechanical properties decreases with increasing leg thrust.




(a) compression limited (b) intermediate (c) obstacle limited

Fig. 11. Maximal safe invariant for compression limited, obstacle limited and intermediate behaviors. For compression limited behavior (a) the "step up" plus "step down" height is equal to the height from which the hopper can fall without damage. For obstacle limited behavior (c) the maximum step up is the height which the hopper can clear and land at zero velocity and still compress the leg enough to clear the next step up. Intermediate behavior (b) requires a non–zero landing velocity to provide sufficient leg compression for the next step up. The intermediate behavior requires multiple iterations through the maximal safe invariant algorithm to find the minimum safe landing speed.

### A. Running with a Single Leg

The problem of running over variable terrain with a single legged hopper is diagrammed in Figure 12. The figure shows a sequence of positions and orientations over a single hop. As before the hopper leg has stiffness and damping and can provide a thrust. Additionally the hopper body has a rotational degree of freedom and associated inertia and can generate torque about the hip joint. The body is at an angle $\theta$ with respect to horizontal and the hopper leg is at an angle $\phi$ with respect to the body. The terrain is stepped with variable changes in terrain height, $\Delta h$, separated by a fixed distance, $\Delta x$. Changes in terrain height are bound between $\Delta h_{max}$ and $\Delta h_{min}$. Alternative models of the terrain are possible but the stepped terrain is most analogous to the vertical hopper studied previously.

During the flight phase the hopper positions the leg to a desired angle for contact. Once in contact, and assuming no slipping, the toe remains fixed to the surface while the leg compresses and rotates about the hip. Prior to liftoff the leg extends from the combination of spring and any thrusting forces. After liftoff the hopper resumes a ballistic trajectory and repositions the leg for the next contact.

The position of the hopper mass center can be tracked relative to the relevant terrain features by adjusting the reference or origin with a discrete phase transition. Figure 12 marks changes in terrain height with the letters 'A' and 'B'. These features are separated by a set distance, $\Delta x$. During flight, the hopper toe may cross over one or more terrain features as indicated by the broken vertical lines in the figure. The cross over triggers a discrete phase transition that shifts the origin of the reference coordinate system. For example, when the toe crosses feature A, the origin of the reference system becomes point A and the horizontal and vertical location of the mass center with respect to point A is $x_{fA}$, which is negative, and $y_{fA}$. At the time of contact mass center position has changed to $x_{cA}$ and $y_{cA}$, and at the time of lift off mass center position has changed to $x_{lA}$ and $y_{lA}$. The hopper resumes flight and when the toe crosses feature B the origin abruptly shifts to point B and the mass center location becomes $x_{fB}$ and $y_{fB}$. The relationship between points A and B is the distance between features, $\Delta x$, and the change in terrain height, $\Delta h$.

The phases of motion for planar running over variable terrain are shown in Figure 13. The `Contact` and `Liftoff` phases are directly analogous to the vertical hopper phases. The flight phases of motion are divided into `Ascent`, and `Descent`. The `Reposition` and `Cross` phases capture discrete events. The `Cross` phase resets the origin of the coordinate system when the abscissa of the hopper toe reaches $\Delta x$. At this instant terrain height changes and collision may occur. Because the distance traveled in flight varies from hop to hop, the phasing between terrain changes and the flight phase is also variable. Thus, a `Cross` may occur one or more times during a hop, or not at all. Also during flight the hopper must reposition the leg from the angle at liftoff to the angle for landing. Since the leg is massless the reposition can occur instantaneously and has no affect on hopper dynamics in flight unless a collision occurs. For simplicity the leg reposition

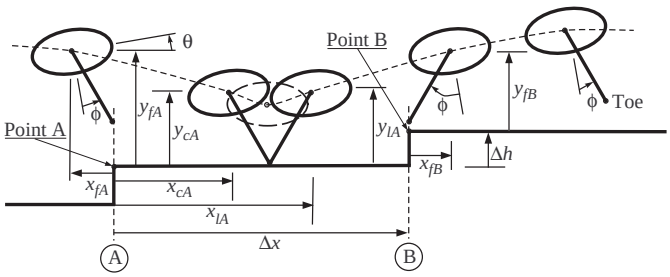



Fig. 12. Diagram of planar running with a single leg. The environment is stepped with variable changes in terrain height, $\Delta h$, separated by a fixed distance, $\Delta x$. To track hopper position relative to terrain features a discrete phase transition shifts the origin of the coordinate system when the hopper toe crosses over feature locations. The planar problem also includes rotational degrees of freedom for the body attitude, $\theta$, and the hopper leg angle, $\phi$, which is measured relative to the hopper body.

is modeled as an instantaneous slew at the apex of flight. However, it should be noted that the leg angle at contact is an important control variable. The `Reposition` phase should allow a range of feasible leg angles and results obtained during the backward chaining sequence of the algorithm will determine the safe subset of reposition angles. Also during `Reposition` checks are made for any `Cross` event or possible collision.

If feature distance, $\Delta x$, is small compared to the length of a hop then more than one `Cross` event may occur in flight. The phase diagram on the left side of Figure 13 indicates multiple `Cross` events during `Ascent` and `Descent` with the two way arrow between mode transitions. The two way transitions create internal loops which must be iterated through during the backward chaining sequence. However, if feature distance is large compared to the length of a hop, then no more than one `Cross` event will occur per hopping cycle. This constraint can be enforced by adding modes and modifying predecessor, successor relationships as shown in the right side of Figure 13. Despite the additional modes and more complex looking phase diagram, execution through the backward chaining sequence is simplified.

The collision during a `Cross` event is a safety constraint on the system. Additional safety constraints include the minimum leg length constraint and constraints on the hip angle. To avoid singularities it is useful and reasonable to impose constraints on the body angle, $\theta$, so that the hopper doesn't perform cartwheels.

A final constraint on the system is non-slip contact at the toe. It can be shown that slipping constraints are linear functions of the control variables $u_1$ and $u_2$ [7]. The constraint boundary is a function of the control which is problematic. However, since the constraints are linear functions, we know that the bounding constraint conditions must be at the extreme values of the control variables. Thus, the constraint for positive slipping is bound by 4 constraint surfaces: one each for the combination of maximum or minimum leg thrust and maximum or minimum hip torque.

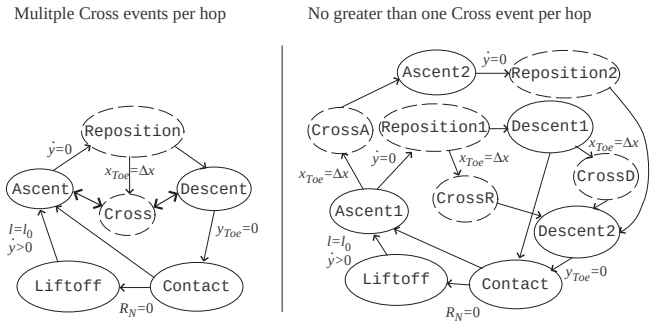



Fig. 13. Phases of motion for planar running with a single leg. The `Ascent`, `Descent`, `Contact`, and `Liftoff` phases are analogous to the like named vertical hopper modes. The `Step` phase is replaced by `Cross` which moves the origin of the coordinate system to accommodate a change in terrain height. `Reposition` slews the leg to a new angle to prepare for landing and may also trigger a `Cross`. If feature distance, $\Delta x$, is small then more than one `Cross` event may occur in flight and the hopper returns to `Ascent` or `Descent` as indicated by the two-way transition to the right. If feature length is large compared to the length of a hop then no more than one `Cross`event will occur per hop which can be enforced by creating new modes with one way predecessor/successor relationships.

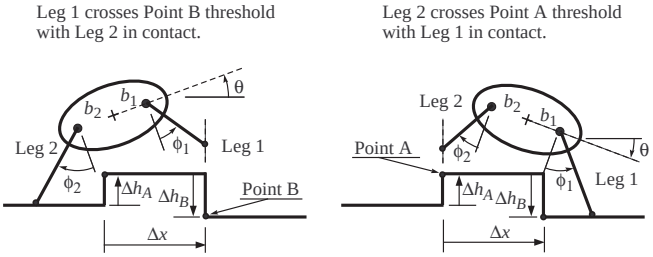



Fig. 14. Planar running with two legs. An additional variable, $\phi_2$, is required to track the position of the second leg relative to the body. Multiple legs also require retaining a "moving window" picture of the environment. When the toe of the front leg reaches a distance $\Delta x$, the environment injects a disturbance,$\Delta h_B$, to change terrain height and the origin of the coordinate systems shifts to the new reference point. When the toe of the back leg reaches $-\Delta x$ the new disturbance replaces the old terrain change, $\Delta h_A$, as a buffered value in the continuous state vector.

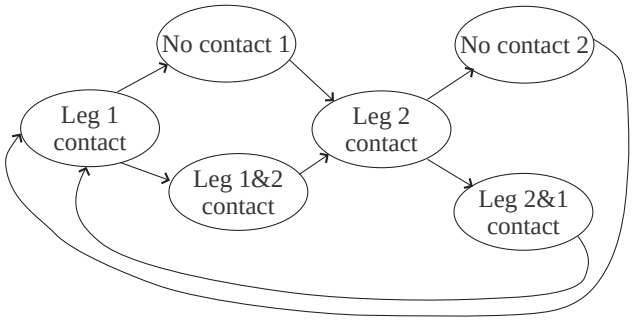



Fig. 15. Simplified phase diagram for running with two legs. Phases between Leg 1 only contact and Leg 2 only contact can be separated by a no contact phase (running gait), or by phases with both legs in contact (walking gait), or some combination of both. Separate no contact and dual leg contact phases enforce alternating legs between steps.

### *B. Running with Multiple Legs*

Applying the maximal controlled safe invariant algorithm to the problem of running with multiple legs approaches a practical application. Interesting applications include the the effect that different robot configurations have on ruggedness. For example, robot legs may be telescoping, such as the *Sprawl* family of robots [3], rigid with flexible hip joints, such as the *RHex* robot [13], or multi-link such as Tekken [16]). Robot posture maybe upright or sprawl. Robot control maybe clock driven or closed loop and include measurements of the environment [7]. Multiple legs also allow multiple gaits. Some of these gaits, such as the alternating tripod adopted by *Sprawl* and *RHex*, keep one or more legs on the ground at all times and could eliminate the collision safety constraint imposed on a single legged hopper.

Figure 14 shows two instances of a two-legged hopper negotiating a step in the terrain. On the left side of the figure the rear leg of the hopper is in contact while the front leg is free. On the right side of the figure the front leg is in contact while the rear leg is free. Other phases of motion include non-contact and dual leg contact.

Figure 15 is a simplified diagram of the phase transitions for two-leg running. Alternating between phases of motions generates different gaits. For example, a running gait may start with Leg 1 in contact then transition to a non-contact phase, then transition to Leg 2 in contact, then transition to another non-contact phase, and repeat. The transition diagram uses two distinct non-contact phases between Leg 1 and Leg 2 contact to force alternating steps between legs. For running, air-borne 'No contact' phases are followed between leg contacts. For walking, both legs are in contact between the 'Leg 1 contact' and 'Leg 2 contact' phases. Note that skipping is permitted in this transition diagram. Single legged hopping could be permitted by adding a transition from 'No contact 1' back to 'Leg 1 contact' and from 'No contact 2' to 'Leg 2 contact'.

Multiple legs complicate the injection of environmental disturbances. Returning to Figure 14, the left hand diagram shows the toe of the front leg of the hopper crossing a change in terrain height, $\Delta h_B$ at Point B. As for the single legged hopper, this change in terrain is assumed to be a worst case disturbance input over a bound range. The right hand side of the figure shows the toe of the rear leg of the hopper cross a change in terrain height at Point A. This feature has already been encountered and the change in height, $\Delta h_A$, was established previously. A change in terrain height at the front leg is fundamentally different then a change in terrain height for the rear leg. Therefore, the system must add dynamic states to track the terrain.

Figure 16 is a more detailed diagram of the phase transitions for two legged running with the addition of discrete modes for front and rear leg crossing events. The number of phases explodes to account for the number of different precedent relationships considered. The figure indicates discrete phases by broken boundaries and front leg crossing events are shaded. For example when leg 2 is in contact and leg 1 is not in

contact (phase B), a front leg crossing event may occur (phase C1), upon which the system transitions back to a continuous phase (phase C2). The transition is to a new phase (C2) rather than the previous phase (B) to avoid an inner loop during backward chaining. From motion with leg 2 in contact (phase B or C2) the system can transition to a phase with both legs in contact (phase D) or with neither leg in contact (phase E). While neither leg is in contact a front leg crossing may occur (phase G1) or a rear leg crossing may occur (phase H1) or both may occur. If both front and rear leg crossings occur with neither leg in contact, the front leg crossing may occur first (G1 $\succ$ G2 $\succ$ H3 $\succ$ G4) or the rear leg crossing may occur first (H1 $\succ$ H2 $\succ$ G3 $\succ$ G4). A similar set of possible mode transitions occurs during the non-contact following leg 1 only contact (phases J, K1-K4, L1-L3).

Figure 16 does not include `liftoff` type transition phases or any explicit `reposition` type phases for the out of contact leg. These phases are omitted to avoid additional clutter. Reposition of the out of contact leg can occur during or upon transition into the single leg contact phases. For example, leg 1 out of contact reposition can occur during phase B and leg 2 out of contact reposition can occur during phase F. Note that any cyclic phase sequence must include phases B and F so that leg repositioning cannot be skipped over.

## VII. Conclusion

This paper describes an analysis approach for determining safe control of a hopping robot operating in uneven and unknown terrain. The approach is based on determining the maximal controlled safe invariant as described by Tomlin, et. al. [14], but simplifies the optimization by assuming that the constraint boundaries are convex (or more generally can be approximated by a finite set of possibly intersecting convex surfaces) and the phase dynamics are non-singular (i.e. $f(x, u, d) \neq 0$) so that boundary of the set of safe states is tangent to and cannot re-enter constraint boundaries. These assumptions apply to a broad class of hybrid systems and permit a geometric solution to the optimization problem.

Results for several robot leg configurations and different environmental variations are provided. The effect of robot thrust and mechanical properties is examined in detail. A metric for "ruggedness" is defined which is the maximum change in terrain height the system can tolerate between hops and guarantee safe operation. For given thrust, maximum ruggedness is achieved at low spring stiffness and at an optimal damping which is high enough to absorb excess energy given a negative change in terrain height, but low enough to permit sustained climbing given positive changes in terrain height.

Extension of the analysis approach to planar running with single or multiple legs is discussed and is feasible but detailed study is considered future work. High dimensional non-linear problems require a numerical tool such as described in [10] or [7]. A numerical solution to a single legged hopper operating under timer based control will be described in a future paper.

Should we put anything in here about Johnathon's work???

S

## Acknowledgment

I could thank Lockheed for supporting me through the Honors Coop Program. If under conclusions and future work we have any reference to RiSE we could also acknowledge that Sponsor (ONR?).

## References

[1] John E. Bares and David S. Wettergreen. Dante II: Technical description, results, and lessons learned. *The International Journal of Robotics Research*, 18(7):621–649, July 1999.

[2] Authur E. Bryson and Yu-Chi Ho. *Applied Optimal Control*. Hemisphere Publishing, 1975.

[3] J. G. Cham, S. A. Bailey, and M. R. Cutkosky. Robust dynamic locomotion through Feedforward–Preflex interaction. In *ASME IMECE Proceedings*, Orlando, Florida, 2000.

[4] J. E. Clark, J. G. Cham, S. A. Bailey, E. M. Froehlich, P. K. Nahata, R. J. Full, and M. R. Cutkosky. Biomimetic design and fabrication of a hexapedal running robot. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3643–3649, 2001.

[5] Yashuhir Fukuoka, Hirosi Kimura, and Avis H. Cohen. Adaptive dynamic waling of a quaddruped robot on irregular terrain based on biological concepts. *The International Journal of Robotics Research*, 22(3–4):187–202, 2003.

[6] Shigeo Hirose, Kan Yoneda, and Hideyuki Tsukagoshi. TITAN VII: Quadruped walking and manipulating robot on a steep slope. In *IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico, 1997.

[7] Brian Howley. *Safe Control Strategies for Hopping Over Uneven Terrain*. PhD thesis, Stanford University, 2006. Department of Mechanical Engineering.

[8] J. C. Latombe K. Hauser, T. Bretl and B. Wilcox. Motion planning for a six–legged lunar robot. In *Workshop on Algorithmic Foundations of Robotics (WAFR)*, New York, New York, July 2006.

[9] John Lygeros, Claire Tomlin, and Shankar Sastry. Multi-objective hybrid controller synthesis. In O. Maler, editor, *Hybrid and Real–Time Systems*, volume 1201 of *Lecture Notes in Computer Science*. Springer Verlag, 1997.

[10] Ian M. Mitchell, Alexandre M. Bayen, and Claire J. Tomlin. A time dependent hamilton–jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957, 2005.

[11] Pieter J. Mosterman, Gautam Biswas, and Janos Szitpanovits. A hybrid modeling and verification paradigm for embedded control systems. In *Control Engineering Practice*. Elsevier Science Ltd., 1998.

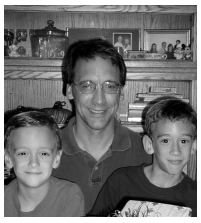
[12] Marc H. Raibert. *Legged Robots that Balance*. MIT Press, 1986.

[13] Uluc Saranli, Martin Bühler, and Daniel E. Koditschek. Design, modeling and preliminary control of a compliant hexapod robot. In *IEEE International Conference on Robotics and Automation*, San Francisco, California, 2000.

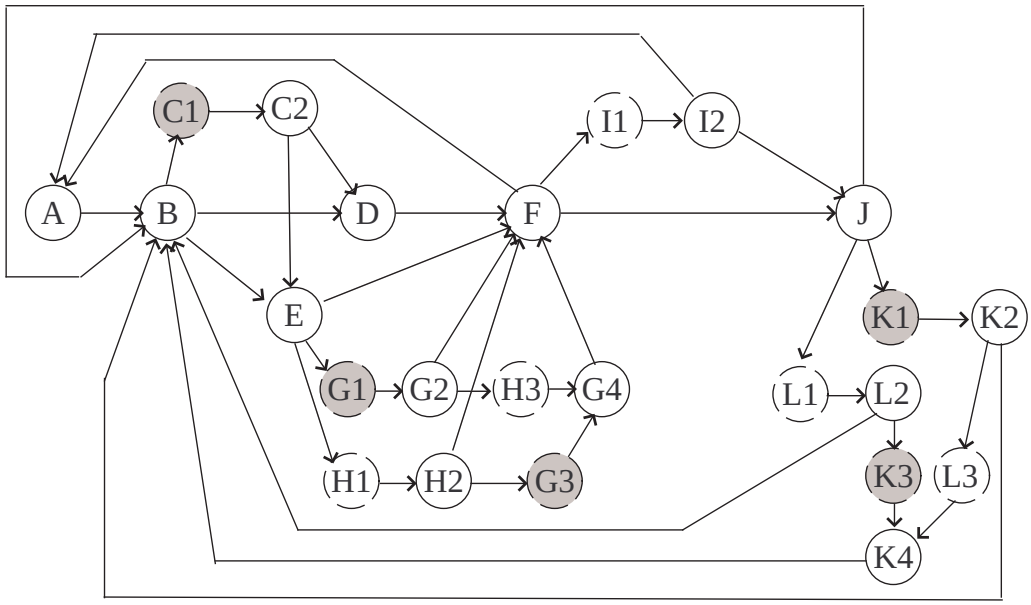
[14] Claire J. Tomlin, John Lygeros, and Shankar Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7), July 2000.

[15] K. J. Waldron, V. J. Perry, and R. B. McGhee. Configuration design of the adaptive suspension vehicle. *The International Journal of Robotics Research*, 3(2), 1984.

[16] Z. G. Zhang, Y. Fukuoka, and H. Kimura. Adaptive running of a quadruped robot using delayed feedback control. In *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005.



**Brian Howley** Senior manager at Lockheed Martin Space Systems Company employee where he has worked for over 23 years. Brian recently completed his PhD in Mechanical Engineering at Stanford University. He is the tall one at the center of the accompanying photo.




A: Leg 1&2 contact
B: Leg 2 contact only
C1: Leg 1 cross after (B)
C2: Leg 2 contact after (C1)
D: Leg 2&1 contact
E: No contact after (B)
F: L1 contact only
G1: Leg 1 cross after (E)
G2: No contact after (G1)
G3: Leg 1 cross after (H2)
G4: No contact after (G3, H3)
H1: Leg 2 cross after (E)
H2: No contact after (H1)
H3: Leg 2 cross after (G2)
I1: Leg 2 cross after (F)
I2: Leg 1 contact after (I1)
J: No contact after (F)
K1: Leg 1 cross after (J)
K2: No contact after (K1)
K3: Leg 1 cross after (L2)
K4: No contact after (K3, L3)
L1: Leg 2 cross after (J)
L2: No contact after (L1)
L3: Leg 2 cross after (K2)

Fig. 16. Detailed phase diagram for two legged running. The number of modes explodes to track the number of possible predecessor/successor relationships.

PLACE PHOTO HERE

**Mark Cutkosky** Biography text here.