# Information-Based Optimization Approaches to Dynamical System Safety Verification

Todd W. Neller*

Knowledge Systems Laboratory, Stanford University
e-mail: neller@ksl.stanford.edu

**Abstract.** Given a heuristic estimate of the relative safety of a hybrid dynamical system trajectory, we transform the initial safety problem for dynamical systems into a global optimization problem. We introduce `MLLO-IQ` and `MLLO-RIQ`, two new information-based optimization algorithms. After demonstrating their strengths and weaknesses, we describe the class of problems for which different optimization methods are best-suited.
The transformation of an initial safety problem for dynamical systems into a global optimization problem is accomplished through construction of a heuristic function which simulates a system trajectory and returns a heuristic evaluation of the relative safety of that trajectory. Since each heuristic function evaluation may be computationally expensive, it becomes desirable to invest more computational effort in intelligent use of function evaluation information to reduce the average number of evaluations needed. To this end, we've developed `MLLO-IQ` and `MLLO-RIQ`, information-based methods which approximate optimal optimization decision procedures.

* This work was supported by the Defense Advanced Research Projects Agency and the National Institute of Standards and Technology under Cooperative Agreement 70NANB6H0075, "Model-Based Support of Distributed Collaborative Design". Author's address: Knowledge Systems Laboratory, Gates Building 2A, Stanford University, Stanford CA 94305-9020.

## 1 Introduction

Given a simulated hybrid dynamical system $S$, a set of possible initial states $I$, and a set of "unsafe" states $U$, we wish to verify nonexistence of an $S$-trajectory from $I$ to $U$ within $t_{\max}$ time units. We call this the *initial safety problem*. Suppose we are given an approximate measure of the relative safety of a trajectory. More specifically, let $f$ be a function taking an initial state $i$ as input, and evaluating the $S$ trajectory from $i$ such that $f(i) = 0$ if and only if the $S$-trajectory from $i$ enters $U$ within $t_{\max}$ time units, and $f(i) > 0$ otherwise. Then verification of the initial safety problem can be transformed into the global optimization (GO) problem:

$$\min_{i \in I}(f(i)) \overset{?}{>} 0$$

GO methods may therefore terminate when $i$ is found such that $f(i) = 0$. Given that $f$ does not generally have an analytic form, we do not assume the availability of derivatives. Since each evaluation of $f$ may require a computationally expensive simulation, we are particularly interested in GO methods which perform relatively few evaluations of $f$. In this context, we introduce two new information-based optimization methods which use function evaluation information approximately optimally in choosing the next best point for evaluation. We demonstrate that these algorithms generally match or exceed the performance of the best methods from our previous comparative study [1], describe the class of functions for which they are best suited, and finally turn our attention to the trade-off between brute-force function evaluation and intelligent, selective function evaluation.

## 2 Motivation

Our research was largely motivated by the following safety verification task: Given bounds on the system parameters of a stepper motor (e.g. viscous friction, inertial load), bounds on initial conditions (e.g. angular displacement and velocity), and an open-loop motor acceleration control, verify that no scenario exists in which the motor stalls. We model the motor's continuous dynamics using ODEs given in [2]:

$$\begin{aligned}
\dot{\theta} &= \omega \\
\dot{\omega} &= \frac{(-i_{\mathrm{a}} N_{\mathrm{b}} \sin(N\theta) + i_{\mathrm{b}} N_{\mathrm{b}} \cos(N\theta) - D \sin(4N\theta) - F_{\mathrm{v}}\omega - F_{\mathrm{c}}\mathrm{sign}(\omega) - F_{\mathrm{g}})}{(J_{\mathrm{l}} + J_{\mathrm{m}})} \\
\dot{i_{\mathrm{a}}} &= (V_{\mathrm{a}} - i_{\mathrm{a}} R + \omega N_{\mathrm{b}} \sin(N\theta))/L \\
\dot{i_{\mathrm{b}}} &= (V_{\mathrm{b}} - i_{\mathrm{b}} R - \omega N_{\mathrm{b}} \cos(N\theta))/L
\end{aligned}$$

where $\theta$ and $\omega$ are motor shaft angular displacement and velocity, $i_{\mathrm{a}}$ and $i_{\mathrm{b}}$ are coil $A$ and $B$ current, $V_{\mathrm{a}}$ and $V_{\mathrm{b}}$ are coil $A$ and $B$ voltage, $R$ and $L$ are coil resistance and inductance, $N$ is the number of rotor teeth, $N_{\mathrm{b}}$ is the maximum motor torque per amp, $D$ is the maximum detent torque, $F_{\mathrm{v}}$ is the viscous friction, $F_{\mathrm{c}}$ is the Coulomb friction, $F_{\mathrm{g}}$ is the gravitational torque load, and $J_{\mathrm{l}}$ and $J_{\mathrm{m}}$ are load and motor shaft inertia. For this system we classify a stall as deviation of $\frac{\pi}{N}$ or more radians from the current desired $\theta$ equilibrium.

The motor is stepped by reversing polarity of the coil voltages in alternation (see Figure 1). Changes to coil voltages occur on such a small time scale that their continuous simulation is judged unnecessary for modeling dynamics relevant to the verification task. Voltage changes were therefore approximated as discrete events. Our acceleration control is open-loop: At fixed intervals the motor is stepped according to an acceleration table. We can express such a system as a nonlinear hybrid automaton as shown in Figure 2.

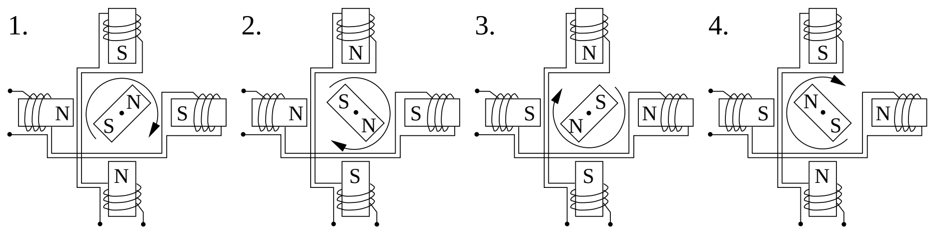



**Fig. 1.** Simple Stepper Motor Stepping

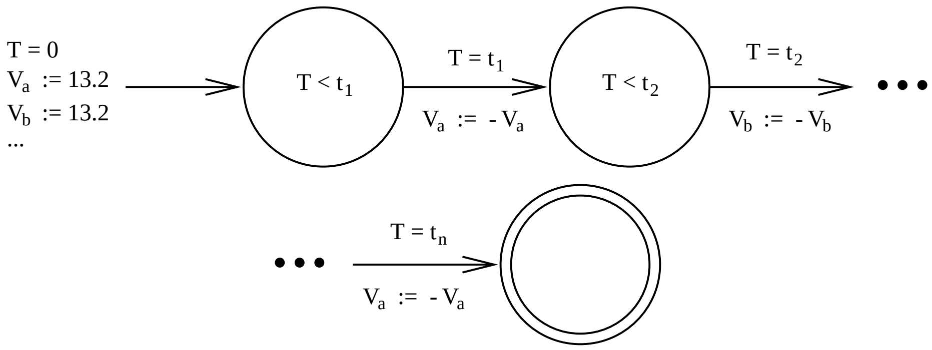



**Fig. 2.** Stepper Motor Nonlinear Hybrid Automaton

First, we note that there is no apparent “geometrically linear hybrid system”[1] approximation with which we could apply the tools of computational geometry, but simulation is feasible. Next, we note that our verification is concerned with

[1] i.e. restricted to constant first derivatives; “geometrically” as opposed to “algebraically”

a fixed initial time interval (i.e. during acceleration) and is therefore an initial safety problem. Finally, we note that we can compute minimum angular displacement from a stall state over all simulation states as a simple heuristic to numerically rate the relative safety of safe trajectories. We can now ask, “For all possible system parameters and initial states, are all simulation trajectories rated safe?” Put another way, “Is the minimum heuristic evaluation of all possible simulations greater than zero?” If we can answer this optimization question positively, we have verified safety of our hybrid system.

One could argue that such optimization is *not* verification, that one cannot exhaustively simulate all possibilities and can therefore have no guarantees. One can only use such optimization for refutation. To this, we offer two responses: First, if one has additional knowledge of characteristics of one's heuristic evaluation function, then an intelligent optimization approach can utilize such characteristics to guarantee a strictly positive minimum (i.e. safety) with enough testing. For example, if one is seeking a zero minimum of a heuristic function which has Lipschitz conditions, and there is no possibility for a zero to occur between previously evaluated points without violating such conditions, one can terminate the optimization having verified safety. Second, if one has no such knowledge about the heuristic (as is the case for our stepper motor problem), the absence of verification techniques well-suited to non-trivial dynamics leaves good global optimization as the best assurance. As has been demonstrated with several NP-hard satisfiability problems [3], refutation through a well-chosen optimization technique, while not complete, can open the door to solving larger classes of problems reliably.

This said, we have endeavored to develop innovative information-based global optimization methods which, under certain assumptions and constraints, make approximately optimal use of information gained in the course of optimization. We next introduce some of these methods.

## 3 Information-Based Global Optimization

From the previous comparative study [1], we noted that most global optimization methods throw away most of the information gained in the course of optimization. For our purposes, each evaluation of $f$ requires a simulation which may be computationally expensive, so we are particularly motivated to make good use of such information in order to reduce the function evaluations needed.

One approach is to characterize properties of the set of functions one wishes to optimize and to use such information to construct an optimal decision procedure for optimization. In the course of optimization, we use our current set of function evaluations to decide on the next best point to evaluate with respect to our function set. Such is the strategy of *Bayesian* or *information* approaches to global optimization [4–7], which have optimal average case behavior over the set of functions for which each is designed. Previous information-based methods have largely been limited to global optimization in one dimension. In this section, we introduce two new information-based optimization methods for multidimensional problems.

We first introduce the decision procedure used by these methods, thus explicating the class of functions for which the decision procedure is biased. Next we discuss the use of *multi-level local optimization* for speeding convergence. Finally, we introduce the information-based optimization algorithms themselves.

### 3.1 Decision Procedure

At each iteration $i$ of our algorithm, we wish to evaluate our heuristic function $f$ at the location $x_i$ for which $f(x_i) = 0$ is most likely to occur. We base our notion of likelihood on characteristics of a class of functions to which $f$ belongs. Our decision procedure is then based on some decision ranking function $g_i$ which computes a ranking corresponding to the relative likelihood of a zero occurring at an unevaluated point $x_i$ given previous $f$-evaluations at $x_1, x_2, \ldots, x_{i-1}$:

$$g_i(x_i) \stackrel{\text{def}}{=} g(x_1, x_2, \ldots, x_{i-1}, x_i)$$

So for each iteration $i$, we could globally optimize $g_i$ to choose the next $x$ for which $f$ is evaluated. However, a reliable global optimization of $g$ for each iteration of a global optimization of $f$ is not only computationally prohibitive, but increasingly very difficult as well. We instead desire to *approximate* an optimal decision with respect to our assumptions about $f$, and we do so by uniformly, randomly sampling $g$, returning the optimum of the samples. We call this `DECISION1` (see Function 1). The computational complexity of this decision procedure grows as the computational complexity of evaluating $g_i$ (which we will see is $O(i^2)$).

**Function 1** Sampling information-based optimization decision function

```
DECISION1(L,lbound,ubound):
% Input: L, a list of [x,f(x)] pairs
%        lbound, lower bounding corner of search space
%        ubound, upper bounding corner of search space

mingx := infinity
for i := 1 to maxpts
  x := uniformly random vector in space bounded by lbound and ubound
  gx := g(L,x)
  if gx < mingx then
    mingx := gx
    minx := x
  end
end for
return minx
```

In order to construct $g$, we must make some assumptions over $f$'s class of functions with regard to where we would most expect to find zeros. One assumption we make is that $f$ is continuous[2]. Another assumption concerns flatness and smoothness preferences: Given a set of points and their f-evaluations, a zero is more likely to occur where it demands less slope between itself and previous points.

A first attempt at constructing $g_i$ might be to create a function which returns

$$g_i(x) = \max_{j=1}^{i-1} \frac{f(x_j)}{||x_j - x||}$$

That is, we could rank the likelihood of $f(x) = 0$ by computing the maximum slope between the hypothetical zero at $x$ and other points we've already evaluated. The lesser the $g$-value, the more likely a zero $f$-value. The global minimum of $g$ would then be the optimal point at which to next evaluate $f$ given previous $f$ evaluations.

Consider Figure 3(a). Suppose we've evaluated the curve at points $a$, $b$, and $c$ and are using such a $g$ as our decision ranking function. Intuitively, we would want $g$ to return point $d$ as the next best point to evaluate. However, the slope between $a$ and $d$ will make $d$ a less preferable decision point than one to the right of $d$ for which a zero would have equal slopes to $a$ and $c$ for this simple function. We would like instead for point $b$ to "shadow" point $d$ from point $a$. Our simple attempt to do so is shown as Function 2. A point $a$ is "shadowed" by point $b$ for function $g$ if $||d-b|| < ||d-a||$ and $|g(a)-g(b)|/||a-b|| > |g(a)-g(d)|/||a-d||$. That is, $a$ is shadowed by $b$ if $b$ is closer to $d$ than $a$, and the slope between $a$ and $b$ on $g$ is greater than the slope between $a$ and $d$ on $g$.

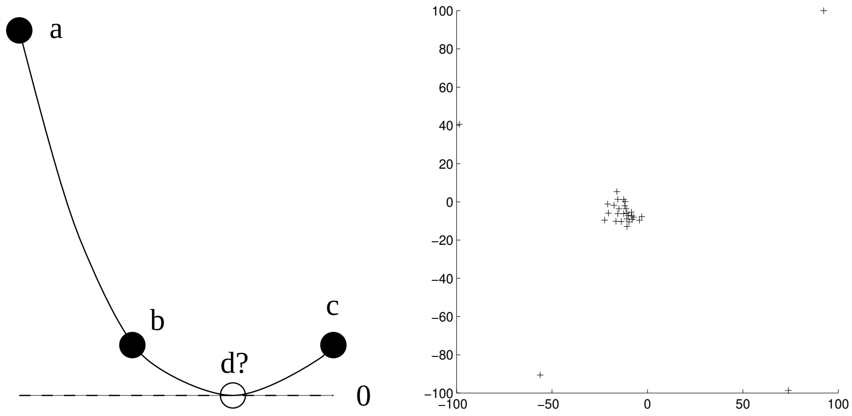
### 3.2 Multi-Level Local Optimization

One might then construct the simple information-based global optimization procedure given in Program 1. However, we note that one ramification of random sampling in our decision procedure is that we do not achieve efficient convergence. This is illustrated in Figure 3(b). From the initial random point in the lower left corner, the procedure then checks points in the upper right, lower right, upper left, and just left of the global minimum at the center. The cluster of 25 points that follows gradually expands towards the center from the fifth point. In practice, where failures do not occur in miniscule regions, this behavior is not a problem. However, we also note that our decision procedure will have to deal with the computational burden of small dense clusters of points which are not very informative globally. We may wish instead to apply a rapidly convergent local optimization procedure and pay attention only to the first and last points of such an optimization.

In our previous comparative study [1], we note that this is a common approach among the most successful methods of the study. A global search phase

---

[2] This is not a trivial assumption for our general application, of course. Our stepper motor system trajectories are continuous in the initial condition. Such continuity is preserved in our choice of $f$.




(a) (b)

**Fig. 3.** (a) Shadowing example, (b) Information-based global optimization of 2-D paraboloid

makes use of a local optimization subroutine so that the global phase is, in effect, searching $f'(x_1) \stackrel{\text{def}}{=} f(x_2)$ where $[x_2, f_{\min}] = \texttt{LO}(f, x_1)$ where LO is a local optimization procedure. In SALO [8] (simulated annealing atop local optimization), for each point evaluation in the global phase, a local optimization takes place and the function value of the local minimum is associated with the original point. The effect can be roughly described as a "flattening" of a search space into many plateaux (with plateaux corresponding to local minimum values). This search paradigm may be generalized to arbitrary levels where each level performs some optimizing transformation of its search landscape to create a "simpler" one for the level above. Obviously, the work done to simplify should be more than compensated by the reduced search effort for the level above. The top level performs a global optimization, and all lower levels perform local optimization. We call this paradigm *Multi-Level Local Optimization* (MLLO). We assert that information-based optimization is particularly well-suited to optimizing coarsely plateaued search landscapes. Now let us consider two information-based applications of MLLO.

### 3.3 MLLO-IQ and MLLO-RIQ

`MLLO-IQ` (Program 2) is a 2-level MLLO with a simple information-based approach (Program 1) atop quasi-Newton local optimization. With each iteration, `MLLO-IQ` chooses a point $x_1$, locally optimizes $f$ from $x_1$ to $x_2$, and associates $f(x_2)$ with both $x_1$ and $x_2$ in order to "plateau" the space. In doing so, we limit the number of function values involved in decision making. Still, we may wish to further limit such growth in computational complexity. By limiting our

**Function 2** $g$, the decision procedure function to be optimized

```
g(L,x):
% Input: L, a list of [x,f(x)] pairs
%        x, current decision point being evaluated

for i := 1 to length(L)
  dx(i) := ||x-first(L(i))||
end for
sort dx in ascending order and permute L accordingly
maxslope := 0
for i := 1 to length(L)
  slope := second(L(i))/dx(i)
  if slope > maxslope then
    newmaxslope := 1
    for j := 1 to i-1
      otherslope := |second(L(i))-second(L(j))|
                    /||first(L(i))-first(L(j))||
      % Note: This otherslope information may be cached.
      if otherslope > slope then
        newmaxslope := 0; break from for loop (j)
    end for
    if newmaxslope then maxslope := slope
  end if
end for
return maxslope
```

information-based search to a hypersphere containing a maximum limit of previously evaluated points, we limit the complexity to a constant. Such is the approach taken in `MLLO-RIQ`.

`MLLO-RIQ` (see Program 3) begins with a locally minimized random point and a maximum search radius. Together these define our initial hypersphere. With each iteration, a decision procedure (`DECISION2`) finds an approximately optimal next point to locally optimize within this hypersphere. If the new point has a lesser function value than the center, it becomes the new center and the distance between the two points becomes the new hypersphere radius. If too many points are being considered in `DECISION2`, a lesser amount of points closest to center are retained and the search radius is adjusted. This information-based local optimization terminates when the number of times the center minimum is found by local optimization exceeds a threshold. Then the process repeats with a new random point. Thus we perform a random search of information-based local optimizations of quasi-Newton local optimizations.

**Program 1** Simple information-based global optimization

```
H = [];
newx := random point in search space
fx := f(newx)
if fx = 0 then terminate with signal UNSAFE
H := append(H,[newx,fx])
loop forever
  newx := DECISION1(H,lbound,ubound)
  fx := f(newx)
  if fx = 0 then terminate with signal UNSAFE
  H := append(H,[newx,fx])
end loop
```

**Program 2** MLLO-IQ

```
H = [];
newx1 := random point in search space
[newx2,fx] := LO(f,newx1)
if fx = 0 then terminate with signal UNSAFE
H := concatenate(H,[[newx1,fx],[newx2,fx]])
loop forever
  newx1 := DECISION1(H,lbound,ubound)
  [newx2,fx] := LO(f,newx1)
  if fx = 0 then terminate with signal UNSAFE
  H := concatenate(H,[[newx1,fx],[newx2,fx]])
end loop
```

## 4 Experimental Results

We now compare our information-based approaches to those considered in our previous comparative study. See [1] for details and references. Our first tests all made use of the same quasi-Newton local optimization method where applicable. 100 optimization trials were performed for each objective function with a maximum of 10000 function evaluations permitted per trial. Each objective function was offset (if necessary) to have a global minimum value of 0. A successful trial was one in which the optimization procedure found a point with function value less than .001 within 10000 function evaluations. This simulates situations where one is seeking a rare failure case in $f$. Each entry in the table of results (Figure 4) shows the number of successful trials (upper left) and the average number of function evaluations for such trials (lower right) for each optimization procedure (rows) and objective function (columns).

Both `MLLO-IQ` and `MLLO-RIQ` perform very well in general. What is most instructive from these results are the cases where the strengths and weaknesses of these methods are most prominently displayed. Let us first consider `RAST`, the Rastrigin function. `RAST` is a 2-D, sinusoidally-modulated, shallow paraboloid

**Program 3** MLLO-RIQ

```
H = []; radius := maxradius
loop forever
  x := random point in search space
  [center,centerval] := LO(f,x)
  if centerval = 0 then terminate with signal UNSAFE
  H := concatenate(H,[[x,centerval],[center,centerval]])
  sort pairs in H in ascending order of ||first(pair)-center||
  H' := up to first (minpts) pairs of H
  centerhits := 0
  while centerhits > maxcenterhits
    recenter := false
    newx1 := DECISION2(H',center,radius)
    [newx2,fx] := LO(f,newx1)
    if fx = 0 then terminate with signal UNSAFE
    if ||newx2-center|| < tolerance1 then
      centerhits := centerhits + 1
    if centerval - fx > tolerance2 then
      radius := min(maxradius, ||newx2-center||)
      center := newx2; centerval := fx; centerhits := 0; recenter := true
    H := concatenate(H,[[newx1,fx],[newx2,fx]])
    H' := concatenate(H,[[newx1,fx],[newx2,fx]])
    if length(H') > maxpts then
      recenter := true
    if recenter then
      sort pairs in H in ascending order of ||first(pair)-center||
      H' := up to first (minpts) pairs of H
  end while
end loop
```

with 49 local minima within the search bounds. The quasi-Newton local optimization layer of MLLO-IQ and MLLO-RIQ effectively transforms this objective function $f$ into a shallow paraboloid of plateaux $f'$. MLLO-IQ's global information-based search of $f'$ finds the lowest plateau very quickly, and the local information-based search of MLLO-RIQ does a focussed descent which leads it to the global minimum with even greater efficiency. This suggests that these searches are particularly well-suited to global optimization of functions with a moderate number of local minima. For functions with fewer local minima (HUMP, G-P, and GW1), there is little to be gained by such extra computation. Random local optimization (RANDLO) will suffice.

Now let us consider the weaknesses of these methods shown in failed cases with GW100. Indeed the performance of these methods is worse than random local optimization. Why? GW100 is a 6-D, sinusoidally-modulated, shallow paraboloid with about $4 \times 10^7$ local minima. For this function, our quasi-Newton local optimization exhibits interesting and unexpected behavior: In all but the lowest points of the surface, local optimization most often leads to local minima that

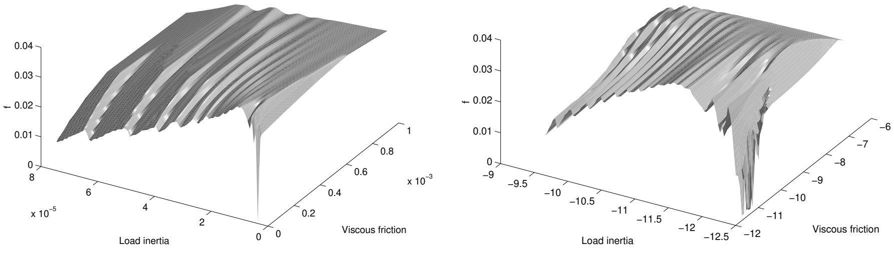
| | `RAST` | `HUMP` | `G-P` | `GW1` | `GW100` | `SWISS` |
|---|---|---|---|---|---|---|
| `AMEBSA` | 16<br>39 | **100**<br>**40** | 90<br>222 | 100<br>86 | 0<br>N/A | 100<br>1340 |
| `ASA` | 100<br>404 | 100<br>225 | 100<br>1042 | 100<br>197 | 2<br>6003 | 100<br>903 |
| `SALO` | 100<br>585 | 100<br>65 | 100<br>97 | 100<br>85 | **95**<br>**4501** | 100<br>163 |
| `LMLSL` | 100<br>847 | 100<br>105 | 100<br>118 | 100<br>96 | 50<br>4508 | 100<br>187 |
| `RANDLO` | 100<br>706 | 100<br>70 | 100<br>96 | 100<br>85 | 58<br>4008 | **100**<br>**146** |
| `MLLO-IQ` | 100<br>286 | 100<br>71 | 100<br>97 | **100**<br>**83** | 57<br>4493 | 100<br>157 |
| `MLLO-RIQ` | **100**<br>**161** | 100<br>57 | **100**<br>**92** | **100**<br>**83** | 46<br>4536 | 100<br>148 |

**Fig. 4.** Successful trials and average function evaluations for each global optimization procedure and test function

are far from those nearby the initial point. In this example, we're reminded that "local" in "local optimization" refers to properties of the optimum itself and not the "nearness" of the optimum location. Without such nearness, the search landscape is not simply transformed into a landscape of plateaux. Our quasi-Newton local optimization didn't optimize to near minima, and so created a landscape which was not suited for information-based global optimization.

`MLLO-RIQ` also has difficulty with `GW100`, but for different reasons. After quickly finding the region containing the global minimum, the method spends much of the remainder of its search effort first searching many points mutually far apart near the boundary of the 6-D hypersphere. Perhaps randomly sampling $f$ or $f'$ within the search hypersphere might encourage convergence. `SALO` remains our best option for functions with a large number of local minima.

While these functions may give a general indication of the relative strengths of these methods (without tuning), the functions share a common property undesirable for our purposes: The *unconstrained* global minimum is never located at or beyond the bounds of the search space. Therefore, our optimization methods need not perform well along the bounds of our search space. It is for this reason that unconstrained quasi-Newton local optimization was suitable for use with such global optimizations. We used this as an opportunity to try two constrained LO procedures `CONSTR` and `YURETMIN` for the 2-D stepper motor test problems `STEP1` and `STEP2` [1] (see Figure 5). `STEP1` takes as input two parameters (viscous friction and load inertia) of the stepper motor model, simulates acceleration of the motor, and performs a simple heuristic evaluation of the trajectory by computing the minimum distance to a stall state (0 if stalled). One could incorporate more sophisticated understanding of a problem domain into one's heuristic function, but computing the minimum distance to an undesirable state is simple and effective for our purposes. `STEP2` is `STEP1` logarithmically scaled so as to focus on the unsafe region of the parameter space. These test functions were chosen for their difficulty. For this testing, we performed 10 trials to find a function value of 0 with a maximum of 1000 function evaluations per trial. The results appear in the tables of figure 6.




(a) `STEP1` (b) `STEP2`

**Fig. 5.** Stepper Motor Test Functions

These results were very pleasing. `MLLO-IQ` is the first technique we've observed that has succeeded in every `STEP1` and `STEP2` trial. It does so with excellent efficiency as well. Since the decision procedure computation time was also dominated by simulation time, it was also easily the fastest algorithm for these trials. `MLLO-RIQ` did surprisingly well considering that most of the search space of these functions slopes downward and away from the corner of the space where the rare failure cases occur.

## 5 Conclusions

A powerful approach to initial safety verification is to transform the problem into an optimization problem and leverage the power of efficient optimization methods. This transformation is accomplished through a heuristic evaluation function $f$ which takes an initial state as input, simulates the corresponding trajectory, and evaluates the trajectory, returning zero if the trajectory is unsafe, or a strictly positive ranking of the relative safety of the trajectory otherwise. Initial safety verification is then a matter of whether or not the global minimum of $f$ for all possible initial states is strictly positive. Our simple heuristic function computes the minimum distance from a trajectory to an unsafe state, but deeper understanding of the problem domain may be incorporated as well.

Although we have not investigated the applicability of optimization to nondeterministic hybrid systems, we believe such techniques are applicable to a

|  | STEP1 | STEP2 |
|---|---|---|
| ASA | 0<br>N/A | 2<br>497 |
| SALO | 10<br>80 | 5<br>202 |
| LMLSL | 10<br>163 | **10**<br>**137** |
| RANDLO | 10<br>78 | 10<br>359 |
| MONTE | 0<br>N/A | 6<br>469 |
| MLLO-IQ | **10**<br>**46** | 10<br>219 |
| MLLO-RIQ | 10<br>60 | 8<br>330 |

(a) CONSTR

|  | STEP1 | STEP2 |
|---|---|---|
| ASA | 0<br>N/A | 2<br>497 |
| SALO | 7<br>387 | 9<br>198 |
| LMLSL | 3<br>389 | 10<br>169 |
| RANDLO | 9<br>501 | 10<br>172 |
| MONTE | 0<br>N/A | 6<br>469 |
| MLLO-IQ | **10**<br>**108** | **10**<br>**109** |
| MLLO-RIQ | 8<br>301 | 9<br>239 |

(b) YURETMIN

**Fig. 6.** Results for STEP1 and STEP2

broader class of deterministic hybrid systems than we have demonstrated. Use of problem domain knowledge to construct a heuristic function and choose global and local optimization techniques should expand the frontier of solvable hybrid system problems. Optimization techniques which are robust with respect to discontinuities should be used for most hybrid system initial safety problems.

From the previous comparative study [1], we noted that most global optimization methods throw away most of the information gained in the course of optimization. For our purposes, each evaluation of $f$ requires a simulation which may be computationally expensive, so we are particularly motivated to make good use of such information in order to reduce the function evaluations needed. To this end, we have introduced two new information-based global optimization methods MLLO-IQ and MLLO-RIQ which, under certain assumptions and constraints, make approximately optimal use of information gained in the course of optimization. Our decision procedure is biased towards approximately optimal average-case behavior for a subclass of continuous heuristic functions.

While no global optimization procedure in our studies was generally dominant, we note that random local optimization seems best suited for heuristic functions with few minima, SALO [8] seems best suited for heuristic functions with very many local minima, and MLLO-IQ and MLLO-RIQ seem best suited for low-dimensional heuristic functions with a moderate number of local minima. MLLO-IQ and MLLO-RIQ appear better suited for problems where the global minima are expected to occur at and within the bounds of the search space respectively.

Finally, we note that the computational effort invested toward efficient optimization should be compensated by reduced overall runtime. For our problem, the computational expense of our simulation justified such effort. But what of initial safety problems for which simulation requires less runtime? Setting `maxpts` $= 0$ for Function 1 yields random decisions. As `maxpts` $\rightarrow \infty$, our decisions approach optimality and the decision-making effort exceeds the search effort it saves. Where is the happy medium in this tradeoff? In future research, we hope to investigate means of dynamically adjusting the level of strategic effort of such information-based algorithms in order to address a larger class of problems efficiently.

## References


1. Todd W. Neller. Heuristic optimization and dynamical system safety verification. In M. Lemmon, editor, *Proceedings of Hybrid Systems V (HS '97)*, pages 51–59, South Bend, IN, USA, 1997. Center for Continuing Education, University of Notre Dame. (available at URL http://www.ksl.stanford.edu/people/neller/pubs.html).
2. Albert C. Leenhouts. *Step Motor System Design Handbook*. Litchfield Engineering, Kingman, Arizona, USA, 1991.
3. Bart Selman, Hector Levesque, and David Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 440–446, Menlo Park, CA, USA, 1992. AAAI Press.
4. Jonas Mockus. *Bayesian Approach to Global Optimization: theory and applications*. Kluwer Academic, Dordrecht, The Netherlands, 1989.
5. Jonas Mockus. Application of bayesian approach to numerical methods of global and stochastic optimization. *J. Global Optimization*, 4:347–365, 1994.
6. Yaroslav D. Sergeyev. An information global optimization algorithm with local tuning. *SIAM J. Optimization*, 5(4):858–870, 1995.
7. Roman G. Strongin. The information approach to multiextremal optimization problems. *Stochastics and Stochastics Reports*, 27:65–82, 1989.
8. Rutvik Desai and Rajendra Patil. SALO: combining simulated annealing and local optimization for efficient global optimization. In J.H. Stewman, editor, *Proceedings of the 9th Florida AI Research Symposium (FLAIRS-'96)*, pages 233–237, St. Petersburg, FL, USA, 1996. Eckerd Coll.