Robotica, in Press.
Accepted January 1998.

Fast Constrained Global Minimax Optimization of Robot Parameters

L. Stocco, S. E. Salcudean and F. Sassani*

Department of Electrical Engineering, *Department of Mechanical Engineering,
The University of British Columbia, Vancouver, British Columbia, Canada V6T 1Z4
leos@ee.ubc.ca, tims@ee.ubc.ca, sassani@mech.ubc.ca

Abstract

A new global isotropy index (GII) is proposed to quantify the configuration independent isotropy of a robot's Jacobian or mass matrix. A new discrete global optimization algorithm is also proposed to optimize either the GII or some local measure without placing any conditions on the objective function. The algorithm is used to establish design guidelines and a globally optimal architecture for a planar haptic interface from both a kinematic and dynamic perspective and to choose the optimum geometry for a 6-DOF Stewart Platform. The algorithm demonstrates consistent effort reductions of up to six orders of magnitude over global searching with low sensitivity to initial conditions.

1 Introduction

Modern robot applications such as haptic interfaces and surgical assistants make performance demands far beyond those of the assembly and repetitive task devices of the past. Designing a robot to uphold a set of performance standards is complicated by the fact that the relationship between the robot's actuators and end-effector varies with position and direction. Only after minimizing this variation, or in other words maximizing the mechanical *isotropy,* can one choose suitable actuators and design a controller. The greatest opportunity for improving isotropy is through geometric parameter selection but making the best choice is no small task. The search space can be made finite through discretization but the order of the optimization problem is compounded by each geometric parameter and by each workspace dimension. Even relatively low dimensional problems are impossible to complete in a reasonable amount of time if an all-inclusive search is attempted. While many efficient search methods exist, most are incompatible with robot design problems which are of the minimax form since an optimum parameter should produce the best worst-case behaviour throughout the robot's workspace, and also because the objective function is, in general, non-linear, non-differentiable, non-convex and may even be discontinuous. Descent algorithms become trapped in local minima, stochastic approaches have uncertain stopping criteria and the results of a global search become increasingly suspicious as the search resolution is decreased.

The kinematic and/or dynamic equations of a robot are often used to describe the relationship between a robot's end-effector and its actuators. The Jacobian matrix determines the required actuator force/torque from a desired end-effector force/torque or the actuator velocity from a desired end-effector velocity. The mass matrix relates actuator force/torque to end-effector acceleration of a device that is starting from rest. Directional independence is represented by a scalar condition index derived from these matrices. Condition indices have been proposed to describe kinematic isotropy **14**, **34** and manipulability **13**, inertial isotropy **3**, **18**, **19** and manipulability **8**, maximum joint velocity **21**, kinematic nonlinearity and redundancy **4**, task completion time **25**, accuracy **13** and stiffness **3**. These condition indices are, however, local measures (i.e. evaluated at a single position) and one is usually interested in behaviour throughout a range of positions. Various methods have been proposed to remove configuration dependence from these measures. Some fix the link lengths and search only for optimal poses **3**, **4**, **34** while others combine the geometry and pose variables and search for optimal geometry/pose pairs **13**, **18**. One approach integrates the performance measure over the workspace **6**, **19**, **23**

while another simply reduces the workspace to a one dimensional trajectory **25**. Similarly, in **21** the parameter set is pared down using a set of predefined trajectories while in **19** the six trajectories along the primary translations and rotations about a central operating point are considered. A coarse minimax optimization is attempted in **8** where the finely discretized workspaces of a few hand-picked 2-DOF device parameters are compared and in **15** where a 3-DOF manipulator is similarly optimized by fixing each degree of freedom individually. The local results of the Matlab™ minimax routine are relied upon in **20** to choose optimum joint arrangements and the number of achievable isotropic poses is analytically maximized in **14**.

Many optimization procedures can be nested to solve minimax problems. Various descent methods exist (see for example **5**) to handle constrained problems. They include gradient, direction, quasi-gradient/direction, penalty, feasible direction and gradient projection methods. Attempts to make these local descent methods global include multi-start **24**, clustering **24**, **30**, decomposition point identification **2**, Rock and Roll **16**, tunneling **33**, transformation of the problem into its concave dual **32** and linear and nonlinear programming **17**, **22**. Some global methods assume a natural model for the objective function such as Bayesian **30**, **31** and Monte-Carlo **30**, **35** methods while others emulate natural processes such as simulated annealing **24**, **30**, **35** and genetic algorithms **7**, **27**. These approaches offer a measure of confidence but no guarantee of global optimality. Other methods such as branch-and-bound **12** guarantees a global optimum but it is not always easy to bound a function and the method is inefficient if the bounds are too conservative. Proposals which address minimax optimization directly include objective function integration over the workspace **10** and tree-searches such as game theory **1**.

A new measure of global isotropy, the “Global Isotropy Index” (GII), is proposed in Section 2 to measure the worst-case consistency of a mechanism in all directions and at any position in its workspace. A new algorithm belonging to the branch-and-bound family of optimization algorithms is presented in Section 3 which finds the globally optimum parameter from a discretized parameter space within a discretized workspace. Unlike descent algorithms, it is unhampered by non-differentiable, non-convex or discontinuous cost functions or those containing local minima. One version is presented for optimizing the GII and another is presented to solve minimax problems in general. The new isotropy index and optimization algorithm are used in Section 4 to design a 2-DOF pantograph robot and again in Section 5 to design a 6-DOF Stewart Platform. The efficiency of the algorithm is discussed in Section 6 and conclusions are drawn in Section 7.

2 The Global Isotropy Index

The Jacobian matrix can be viewed as an effective transmission ratio between the actuators and the end-effector. It transforms joint rates into end-effector velocity or the end-effector force/torque into actuator forces and/or torques. These two uses are shown for a serial manipulator in (1) and (2) where $\dot{x}$ is end-effector velocity, $\dot{q}$ is joint velocity, τ is actuator force/torque and f is end-effector force/torque.

$$\dot{x} = J\dot{q} \qquad (1)$$

$$\tau = J^T f \qquad (2)$$

Using the Jacobian to transform all end-effector forces of unit magnitude and arbitrary direction into actuator torques produces an ellipsoid in the torque domain. Consider the planar elbow manipulator in Figure 1 with the parameter $p = \{l_1 = 5, l_2 = 4\}$ and which applies forces in all directions but whose workspace is constrained to the horizontal trajectory $x \in \{-x_{max}, x_{max}\}$ at $y=2$. A unit circle in the end-effector force domain and its corresponding ellipse in the

actuator torque domain at $x=5$ are shown in Figure 2. The lengths of the major and minor axes of the actuator torque ellipse are equal to the minimum and maximum singular values of the Jacobian, denoted by $\underline{\sigma}(J(p,x))$ and $\tilde{\sigma}(J(p,x))$ respectively and the condition number $\kappa(p,x)$ is defined as the ratio of the these two values as shown in (3). The closer $\kappa(p,x)$ is to unity, the more isotropic the device is at x. Note that while the Jacobian is used in this example, the same can be said about any matrix transformation such as the mass matrix.

$$\kappa(p,x) = \frac{\tilde{\sigma}(J(p,x))}{\underline{\sigma}(J(p,x))} \qquad (3)$$

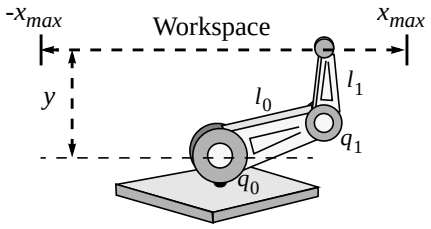

Figure 1: Constrained Planar Elbow Manipulator

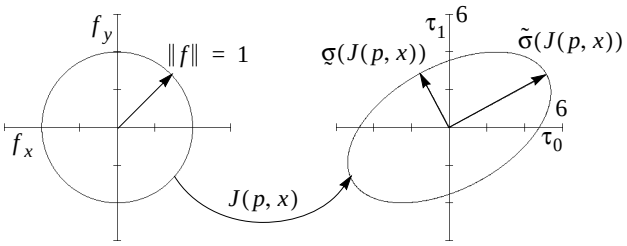

Figure 2: Torque Ellipse at x=5

Because the Jacobian is a function of position, the condition number is a local measure and manipulators that are designed to be isotropic at individual positions may not exhibit similar levels of isotropy throughout their workspaces. The condition number only measures the roundness of an ellipse but does not measure its size. Both of these attributes are, however, important in determining the overall consistency of a device's behaviour since shape is a relative measurement which represents directional isotropy whereas size is an absolute measurement which represents average capabilities. Optimizing the condition number, therefore, does not address the possibility that average capabilities may change as the workspace is navigated. Figure 3 shows the different sizes and shapes of torque ellipses that occur at three different positions for the robot in Figure 1. The ellipses at $x=0$ and $x=5$ have similar shapes ($\kappa(p,0) = 2.17$ and $\kappa(p,\pm5) = 1.81$), but the singular values at $x=5$ are an average of 1.5 times larger than those at $x=0$ ($|\sigma(J(p,0))| = 2.97$ and $|\sigma(J(p,\pm5))| = 4.56$). In other words, even though the manipulator is comparably isotropic at these two positions, it has over one and a half times the average force capabilities in the centre of its workspace than it does at the edges of its workspace. This is hardly uniform behaviour.

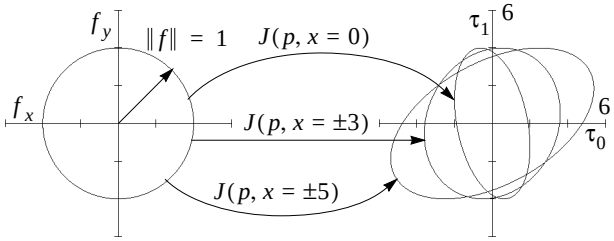

Figure 3: Force/Torque Transformation

In the past **8**, configuration independence has been checked by comparing the products of minimum and maximum singular values at different positions. Here, this secondary, local measure is averted by introducing a global isotropy index called the GII which compares the smallest and largest singular values in the entire workspace (4). The GII is essentially a global, workspace inclusive version of the condition number. Note that the GII is defined as a minimum over a maximum rather than a maximum over a minimum like the condition number. This conveniently makes the GII a normalized performance measure *s(p)* which assigns a value of *1* to perfect isotropy and a value of *0* rather than ∞ to singular behaviour.

$$GII(p) = s(p) = \min_{x_0, x_1 \in W} \frac{\sigma(J(p, x_0))}{\tilde{\sigma}(J(p, x_1))} = \frac{\min_{x_0 \in W} \sigma(J(p, x_0))}{\max_{x_1 \in W} \tilde{\sigma}(J(p, x_1))} \qquad (4)$$

Consider again the elbow manipulator in Figure 1. Local actuator torque ellipses are computed at all values of *x* ranging from *-5* to *+5* and are superimposed upon one another in Figure 4. The GII is the ratio of the radius of the largest circle contained in all of these ellipses to the radius of the smallest circle containing all of these ellipses.

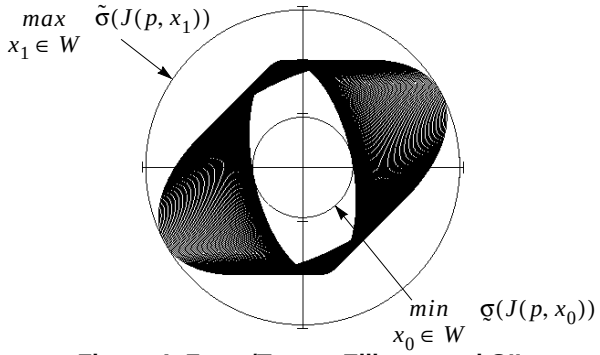

Figure 4: Force/Torque Ellipses and GII

A GII value of *1* implies that a mechanism is not only isotropic (direction insensitive) at each position in its workspace, but also that it behaves consistently at all positions within its workspace. An optimally isotropic robot design parameter p^* is, therefore, one that maximizes the value of the GII as shown in (5).

$$p^* = \arg\max_{p \in P} GII(p) \qquad (5)$$

3 Culling Algorithm

A new algorithm is proposed which is of the branch and bound variety but is specifically designed to solve GII or minimax optimization problems. It identifies non-optimal parameters and culls them from the search space until only the optimum remains. The algorithm optimizes the GII (4) which is defined between *0* and *1* corresponding to poor and ideal performance respectively, over a workspace *W* which is a constrained set of configurations *x* for a parameter *p*. The

optimization goal (5) is to find the parameter p^* with the best "worst-case" behavior throughout the constrained workspace *W*. The algorithm is shown in (6) through (15) and uses the following notation.

List of Symbols

i = looping index
P_i = set of all parameters in parameter space
p_i = design parameter
$\hat{p}_i$ = best known design parameter
W = set of all positions in workspace
x = end-effector position
$\underset{\sim}{x}$ = position with the smallest singular value
$\tilde{x}$ = position with the largest singular value
$\underset{\sim}{\sigma}$ = minimum singular value at a position
$\tilde{\sigma}$ = maximum singular value at a position
$\underset{\sim}{\Sigma}_i : P_i \rightarrow \Re$ = minimum singular value upper bounding function
$\tilde{\Sigma}_i : P_i \rightarrow \Re$ = maximum singular value lower bounding function
s = performance measure; either GII or κ^{-1} as defined in (3)
$\hat{s}$ = performance measure of best known design parameter

GII Culling Algorithm

Set $i = 0, \quad \hat{s}_0 = 0$ (6)

Set $\left\{ \begin{matrix} \underset{\sim}{\Sigma}_0(p) = \infty \\ \tilde{\Sigma}_0(p) = 0 \end{matrix} \right\}; \ \forall p \in P_0$ (7)

Choose $(p_0 = \hat{p}_0) \in P_0$ (8)

REPEAT

Find $\underset{\sim}{x}_i = \underset{x \in W}{\arg min} \ \underset{\sim}{\sigma}(p_i, x), \quad \tilde{x}_i = \underset{x \in W}{\arg max} \ \tilde{\sigma}(p_i, x)$ (9)

if $\left(\hat{s}_{i+1} = \dfrac{\underset{\sim}{\sigma}(p_i, \underset{\sim}{x}_i)}{\tilde{\sigma}(p_i, \tilde{x}_i)} \right) > \hat{s}_i; \ \hat{p}_{i+1} = p_i$ (10)

otherwise $; \ \hat{p}_{i+1} = \hat{p}_i, \hat{s}_{i+1} = \hat{s}_i$

Set $\left\{ \begin{matrix} \underset{\sim}{\Sigma}_{i+1}(p) = min \ \{\underset{\sim}{\Sigma}_i(p), \ \underset{\sim}{\sigma}(p, \underset{\sim}{x}_i)\} \\ \tilde{\Sigma}_{i+1}(p) = max \ \{\tilde{\Sigma}_i(p), \ \tilde{\sigma}(p, \tilde{x}_i)\} \end{matrix} \right\}; \ \forall p \in P_i$ (11)

Set $P_{i+1} = \left\{ p \in P_i \mid \dfrac{\underset{\sim}{\Sigma}_{i+1}(p)}{\tilde{\Sigma}_{i+1}(p)} > \hat{s}_{i+1} \right\}$ (12)

Choose $p_{i+1} \in \underset{p \in P_{i+1}}{\arg max} \ \dfrac{\underset{\sim}{\Sigma}_{i+1}(p)}{\tilde{\Sigma}_{i+1}(p)}$ (13)

$i = i + 1$ (14)

UNTIL $\hat{p}_i = p_i$ (15)

The algorithm starts with a looping index and best known performance measure of zero (6), optimistic bounding functions (7) and an initial p_0 that is chosen from P_0 (8). Minimum and maximum singular values are calculated for p_i at each x in W (9). If p_i produces a better GII than the best known parameter $\hat{p}_i$, p_i becomes the new best known parameter $\hat{p}_{i+1}$ and a new best known performance measure $\hat{s}_{i+1}$ is calculated (10). Singular values are calculated for each p in P_i at $\underset{\sim}{x}_i$ and $\tilde{x}_i$

and the corresponding upper $\underset{\sim}{\Sigma}_i(p)$ and lower $\tilde{\Sigma}_i(p)$ bounds are updated (11). Note that since all singular values are known for $\underset{\sim}{x}_i$ and $\tilde{x}_i$ from (9), bounding may be improved by replacing (11) with (16).

$$\text{Set} \begin{Bmatrix} \underset{\sim}{\Sigma}_{i+1}(p) = min \{\underset{\sim}{\Sigma}_i(p), \ \underset{\sim}{\sigma}(p, \underset{\sim}{x}_i), \ \underset{\sim}{\sigma}(p, \tilde{x}_i)\} \\ \tilde{\Sigma}_{i+1}(p) = max \{\tilde{\Sigma}_i(p), \ \tilde{\sigma}(p, \underset{\sim}{x}_i), \ \tilde{\sigma}(p, \tilde{x}_i)\} \end{Bmatrix}; \ \forall p \in P_i \qquad (16)$$

Although any improvement from using (16) can be primarily attributed to good fortune, it comes from a negligible increment in computational effort and is a worthwhile investment. Also note that one or both updates in steps (11) and (16) can be omitted for all *p* whose ratio of upper and lower bounds is already less (i.e. worse) than $\hat{s}_{i+1}$ since those *p* will be culled from P_i in (12). The *p* with the largest ratio of upper and lower bounds is chosen as the next candidate p_{i+1} (13). (9) through (14) are repeated until $\hat{p}_i$ is the only parameter left in P_i which conclusively identifies $\hat{p}_i$ as the global optimum (15).

Since a parameter is only removed from the search space after it has produced singular values with a ratio worse than that of another parameter $\hat{p}$ for which all singular values have been rigorously computed, the global optimum is guaranteed. Computational savings result from strategically exploring configurations which are likely to simultaneously identify many parameters as sub-optimal. Expected efficiency, however, relies on the presumption that within a continuous, bounded range of parameters, many of them, particularly those in close proximity to each other, will exhibit similarly favourable or poor behaviour at common configurations. This is a presumption that holds well in robot design problems. Consider, for example, a robot that stretches to its reachable limit when visiting a position in the pre-defined workspace which produces a minimum singular value of *0*. A small adjustment to one geometric parameter will usually only slightly affect the robot's reachable limit and it will continue to produce very large and/or small singular values at that position. It and all other neighbors of the original parameter are, therefore, likely candidates for being culled from the parameter space after being evaluated at that position.

While the GII culling algorithm is specifically geared toward optimization of the GII, some worst-case design problems are of the form shown in (17) and can be solved using a similar approach. An optimization criteria of this form is used, for example, by Hayward et. al. **8** with $s(p, x) = \underset{\sim}{\sigma}(D(p, x))/\tilde{\sigma}(D(p, x))$ to optimize the mass matrix *D(p, x)* of a planar pantograph haptic interface. Problems of this form can be solved by the minimax culling algorithm shown in (18) through

(27). Note that in the minimax version of the culling algorithm, $\underset{\sim}{\Sigma}_i$ is an upper bound on the worst-case performance function $s(p, x)$, and no longer contains singular values explicitly.

$$p^* = \underset{p \in P_0}{\arg max} \; \underset{x \in W}{min} \; s(p, x) \tag{17}$$

Minimax Culling Algorithm

Set $i = 0, \;\; \hat{s}_0 = 0$ (18)

Set $\underset{\sim}{\Sigma}_0(p) = 1; \;\; \forall p \in P_0$ (19)

Choose $(p_0 = \hat{p}_0) \in P_0$ (20)

REPEAT

Find $\underset{\sim}{x}_i = \underset{x \in W}{\arg min} \; s(p_i, x)$ (21)

if $(\hat{s}_{i+1} = s(p_i, \underset{\sim}{x}_i)) > \hat{s}_i; \;\; \hat{p}_{i+1} = p_i$
otherwise $; \;\; \hat{p}_{i+1} = \hat{p}_i, \hat{s}_{i+1} = \hat{s}_i$ (22)

Set $\underset{\sim}{\Sigma}_{i+1}(p) = min \; \{\underset{\sim}{\Sigma}_i(p), \;\; s(p, \underset{\sim}{x}_i)\}; \;\; \forall p \in P_i$ (23)

Set $P_{i+1} = \{p \in P_i \mid \Sigma_{i+1}(p) > \hat{s}_{i+1}\}$ (24)

Choose $p_{i+1} \in \underset{p \in P_{i+1}}{\arg max} \; \Sigma_{i+1}(p)$ (25)

$i = i + 1$ (26)

UNTIL $\hat{p}_i = p_i$ (27)

The minimax culling algorithm is illustrated by a step-by-step example using the planar elbow manipulator of Figure 1. The performance index is the ratio of singular values of the Jacobian matrix $s(p, x) = \underset{\sim}{\sigma}(J(p, x)) / \tilde{\sigma}(J(p, x))$ and the parameter space is reduced to a single dimension by calculating the minimum forearm length l_1 from (28) which ensures that the boundaries of the usable and reachable workspaces shown in Figure 5 are separated by a minimum safety margin of length *K*. In other words, l_1 is chosen such that $\{k_0, k_1\} \geq K$.

$$l_1 = max(\left\| \sqrt{x_{max}^2 + y^2} - l_0 \right\|, \left\| y - l_0 \right\|) + K \tag{28}$$

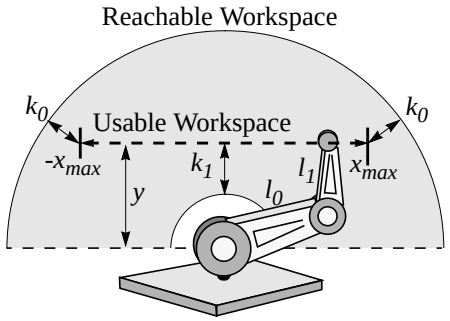

Figure 5: Planar Elbow Manipulator Workspaces

The performance index is non-linear, non-differentiable and contains local minima and maxima in both its operation and parameter spaces which, in this case, can be verified by brute force since the problem has only three dimensions (x, l_0, and

$s(l_0, x) = \varsigma(J(l_0, x))/\tilde{\sigma}(J(l_0, x))$). The performance index $s(l_0, x)$ is plotted against x and l_0 in Figure 6 with $x_{max}=5$, $y=2$, $K = 0.4$ and $l_0 \in \{2, 8\}$.

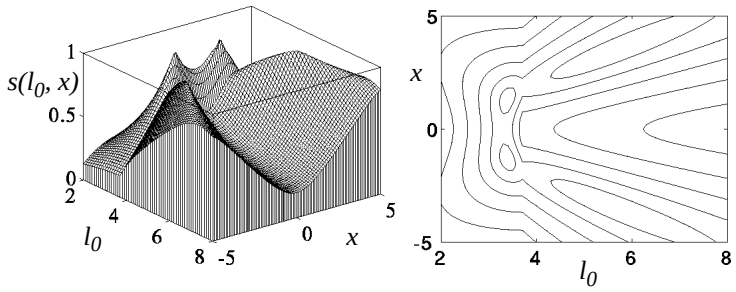

Figure 6: Surface and Contour Plots of Dexterity

A mid-range value of l_0=6 is picked as the initial best known parameter $\hat{p}_0$ (20). The workspace of l_0=6 is searched to find that x_0=0 minimizes $s(l_0, x)$ with a value of $s(6, 0)=0.28$ (21) and the condition index of the best known parameter is updated (22). The parameter space at x_0=0 is searched, the upper bound Σ_1 is updated for each parameter (23) and all sub-optimal parameters are culled (shaded regions in Figure 7) from the parameter space (24). l_0=3.3 has the largest upper bound and is the next candidate p_1 (25).

$$\Sigma_1(l_0) = s(l_0, 0)$$

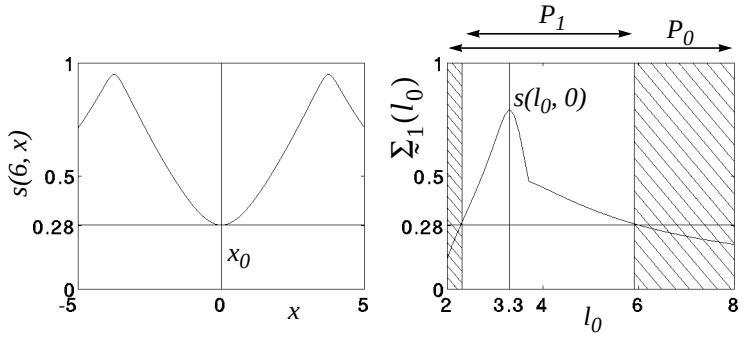

Figure 7: First Culling of Non-Optimal Geometries

The workspace of l_0=3.3 is searched to find that x_1=–5 minimizes $s(l_0, x)$ with a value of $s(3.3, -5)=0.16$ (21) so l_0=6 is still the best known parameter (22). The parameter space $P_1 = \{2.3 \ldots 5.9\}$ at x_1=–5 is searched, the upper bound Σ_2 is

updated for each parameter (23) and all sub-optimal parameters are culled from the parameter space (24). l_0=4.5 has the largest upper bound and is the next candidate p_2 (25) (see Figure 8).

$$\begin{aligned}\tilde{\Sigma}_2(l_0) &= min\{\tilde{\Sigma}_1(l_0), s(l_0, -5)\} \\ &= min\{s(l_0, 0), s(l_0, -5)\}\end{aligned}$$

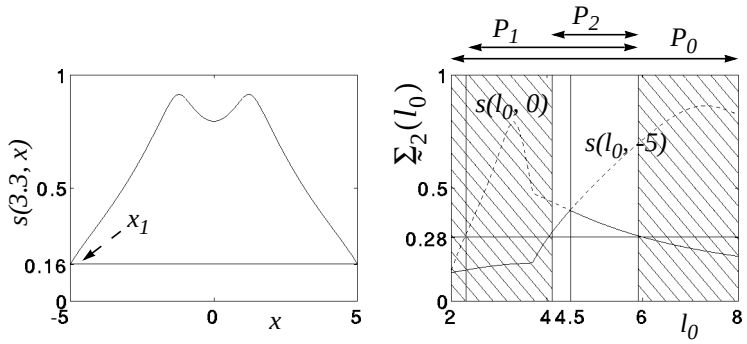

Figure 8: Second Culling of Non-Optimal Geometries

The workspace of l_0=4.5 is searched to find that x_2=0 minimizes $s(l_0, x)$ with a value of $s(4.5, 0)=0.4$ (21) making it the new best known parameter (22). All remaining parameters left in the parameter space $P_2 = \{4.1 \ldots 5.9\}$ have upper bounds that are below this value and are culled from the parameter space (23), (24). Only l_0=4.5 remains and is, therefore, the global optimum p^* (27). This is verified by performing an exhaustive search and plotting the worst-case performance of each geometry. The optimum geometry is shown to be l_0=4.5 in Figure 9.

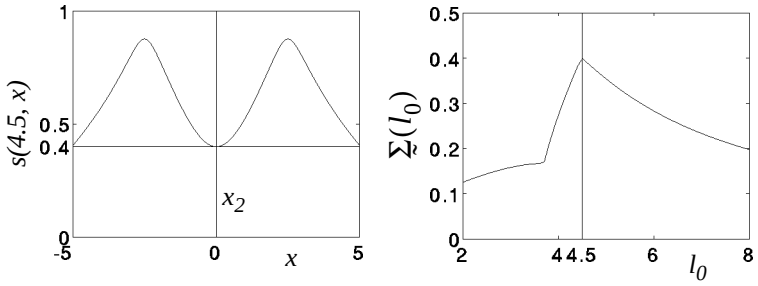

Figure 9: Optimum Solution

The culling algorithm belongs to the branch-and-bound family of optimization algorithms but is unconventional in that it is only useful in solving minimax problems. It also performs all of its bounding through explicit function evaluations and, therefore, does not require any worst-case estimates of the objective function. Each time the condition index of a candidate parameter is rigorously computed, the value is used to push up the lower bound on the performance index of the optimum parameter and each time the condition index is computed for a parameter at a new position, the value is used to push down the upper bound on the performance index of that parameter. If the lower bound on the optimal performance index exceeds the upper bound on the performance index of any parameter, that parameter is culled from the parameter space. This is how the culling algorithm performs bounding. It performs branching by alternating between workspace-inclusive searches for a single parameter and parameter-space inclusive searches for a single point, choosing which parameter or position to search from the results of the previous iteration.

4 Design of a Five-Bar Linkage Based Planar Device

Karadis et. al **11** design a dynamically balanced five-bar linkage for micro-probing while Hayward et. al. **8** optimize a five-bar linkage for use as a planar haptic interface. The culling algorithm and new definition of global isotropy are used to re-examine the five-bar linkage based haptic interface for both kinematic and dynamic conditioning. A general representation of a five-bar linkage with a square workspace is shown in Figure 10 and is used to establish symmetry and positioning guidelines for the device.

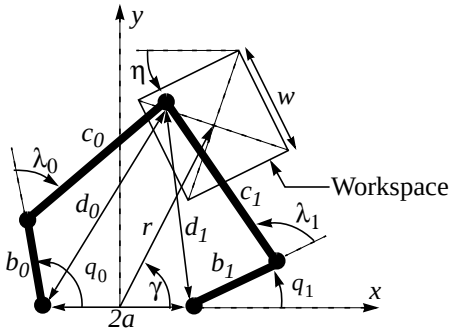

Figure 10: Five-Bar Linkage

The Jacobian matrix (29) of this device is concisely computed as a function of end-point location from equations (30) through (36).

$$\begin{bmatrix}\dot{q}_0\\ \dot{q}_1\end{bmatrix} = J(p,x)\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} \tag{29}$$

$$d_0 = (x+a)^2 + y^2 \tag{30}$$

$$d_1 = (x-a)^2 + y^2 \tag{31}$$

$$J_0 = b_0^2 - c_0^2 - d_0 \tag{32}$$

$$J_1 = b_1^2 - c_1^2 - d_1 \tag{33}$$

$$J_2 = d_0\sqrt{4b_0^2 d_0 - (b_0^2 - c_0^2 + d_0)^2} \tag{34}$$

$$J_3 = d_1\sqrt{4b_1^2 d_1 - (b_1^2 - c_1^2 + d_1)^2} \tag{35}$$

$$J = \begin{bmatrix} \frac{-y}{d_0} + \frac{J_0(x+a)}{J_2} & \frac{x+a}{d_0} + \frac{J_0 y}{J_2} \\ \frac{-y}{d_1} - \frac{J_1(x-a)}{J_3} & \frac{x-a}{d_1} - \frac{J_1 y}{J_3} \end{bmatrix} \tag{36}$$

The kinematic GII is optimized with seven free design parameters (a, b_0, b_1, c_0, c_1, γ and η) and with r and w fixed to avoid a trivial result since isotropy improves as $r\to\infty$ or as $w\to 0$. For $r=w=10$ and $\Delta w=0.1$ (Δw is the discrete sample spacing in the workspace) the search space and solution obtained by the GII culling algorithm are shown in Table I.

Table I: Parameter Space & Optimum

Parameter	Min. Val.	Max. Val.	Resolution	Optimum
a	0	3	0.5	1.5
b_0, b_1	4	10	0.5	7.5
c_0, c_1	7	14	0.5	9.5
γ, η	0	$\pi/2$	$\pi/20$	$\pi/2$

The global optimum has left/right symmetry of both the robot and workspace. Future optimizations, therefore, need only consider three parameters (a, $b=b_0=b_1$ and $c=c_0=c_1$) and half of the workspace ($x\geq 0$). This simplifies the problem sufficiently to allow further generalization. Varying r while constraining the elbow angles $\lambda_0, \lambda_1 \in \{0 \ldots \pi\}$ shows the optimum posture for a range of r values where a negative value of r corresponds to a workspace positioned below the line connecting joints $\{q_0, q_1\}$. Figure 11 shows the optimum geometry, GII and posture for $r \in \{-25 \ldots 25\}$, $\gamma=\eta=\pi/2$ and the parameter space shown in Table II. Note that the search space constrains the robot and workspace to a *30*x*30* square area.

Table II: Reduced Parameter Space

Parameter	Min. Val.	Max. Val.	Resolution
a	0	15	0.2
b, c	5	30	0.2

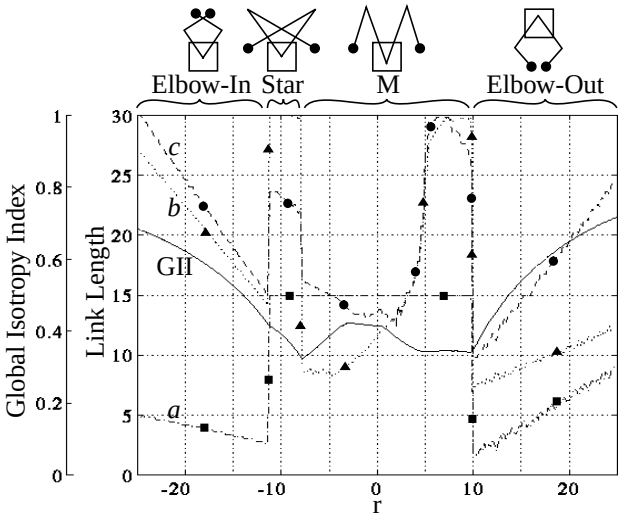

Figure 11: Optimal Postures of Five-Bar Linkage

The GII curve is non-smooth and the optimal parameter curves (a, b & c) are discontinuous in r. Parametric discontinuities occur at the intersections of optimum GII curves for different postures. Consider the region around $r=10$. The optimum GII of the "M" posture is relatively level while the optimum GII of the elbow-out posture increases with r. When the curves intersect, the optimum posture switches from "M" to elbow-out and the parametric curves experience a jump. There are clearly two viable ranges for r. Values of $r \in \{-3 \ldots 1\}$ are acceptable in which case the "M" posture is best with the workspace positioned between the actuators. Magnitudes greater than 10 are also acceptable in which case the elbow-out posture is preferred. While elbow-in achieves GIIs similar to elbow-out for similar magnitudes of r it requires longer physical link lengths (b & c). The inertial implications of this distinguish elbow-out as the better posture.

For all other values (i.e. $r \in \{\{-12 \ldots -3\}, \{1 \ldots 10\}\}$), the optimal postures combine long link lengths with poor GIIs and should be avoided.

For an inertial optimization, it is assumed that the device is held with a light fingertip grip so hand inertia is neglected. It is also assumed that joint and rotor inertia is dominated by the inertia of the linkages which are made from circular cross-section 2024-T4 aluminum tubing of thickness *t*. The mass matrix is obtained by computing the passive joint velocities $\dot{\lambda}_0$ and $\dot{\lambda}_1$ and treating the device like two elbow manipulators joined at the end-effector. The mass matrix of an elbow manipulator as a function of position, geometry and the mass and centre of mass of each link is available in **26**. For uniform tubing, the centre of mass of each tube is its geometric centre and mass per unit length (37) is a function of wall thickness *t* and diameter *d*. Keeping wall thickness constant, the natural frequency ω of the mechanism is conservatively bounded by choosing the diameter (38) that results in the same natural frequency for a cantilever beam the length of all four robot links arranged end to end as calculated in **9**.

$$m = 8.7td \text{ (g/cm)} \tag{37}$$

$$d = 2.55\times10^{-6}(b_0 + b_1 + c_0 + c_1)^2\omega \text{ (cm)} \tag{38}$$

It is debatable whether it is preferable to optimize the mass matrix for isotropy or scale. Since isotropic mass is not particularly important if the magnitude is small, maximum mass is minimized by considering the normalized performance index involving only the maximum singular value of the mass matrix shown in (39). Since this measure can be computed locally, the minimax culling algorithm is used.

$$s(p, x) = \frac{1}{1 + \tilde{\sigma}[D(p, x)]} \tag{39}$$

It was observed that the culling algorithm becomes significantly less efficient when the value obtained from the objective function does not change throughout large, connected portions of the workspace. This often occurs when the desired workspace extends beyond the reachable workspace where $s(p, x)$ is equal to *0*. Efficiency is restored by augmenting the condition index with a separate function for unreachable points $s'(p, x)$ which assigns a value of *0* to a position on the boundary of the reachable workspace and a value of *-1* to a position infinitely far from it (40). The augmented condition index $s''(p, x)$ (41) is still a normalized index since its magnitude is less than or equal to unity (42) with positive values (39) for reachable positions and negative values (40) for unreachable positions.

$$s'(p, x) = \frac{1}{1 + \text{distance from w/s}} - 1 \tag{40}$$

$$s''(p, x) = \begin{cases} s(p, x) & \text{if position is reachable} \\ s'(p, x) & \text{otherwise} \end{cases} \tag{41}$$

$$s''(p, x) \in \{-1 \ldots 1\} \tag{42}$$

Since $s''(p, x)$ can have either a positive or a negative value, a modification must be made to the minimax culling algorithm. The initialization step (18) must be replaced with (43) but the remainder of the algorithm is entirely compatible.

$$\text{Set } \; i = 0, \;\; \hat{s}_0 = -1 \tag{43}$$

Two optimizations are performed, both with $w=10$, $\Delta w=0.1$ and $\omega=200\pi$ (*100* Hz) but searching two different parameter spaces. The first optimization (parameter space "A") attempts to find a design that is both dynamically and kinematically favourable by picking the dynamic optimum from the geometric combinations that are the kinematic optima for different values of *r*. There is only one free parameter *r* while all other geometric parameters (*a*, *b* and *c*) are chosen as a function of *r* from Figure 11. Although this parameter space is very small, it ensures a kinematic GII that is no less that *0.3*. The second optimization (parameter space "B") includes the full cross-section of geometric combinations searched during the kinematic optimizations so all four geometric design parameters *a*, *b*, *c* and *r* are free. Although a trivial result is obtained when *r* is free during kinematic optimization, this does not occur during dynamic optimization. The two parameter spaces and optimum solutions are shown in Table III.

Table III: Inertial Parameter Spaces & Optima

Parameter	Min. Val.	Max. Val.	Resolution	Optimum
Parameter Space A				
a	from table	from table	from table	1.6
b	from table	from table	from table	7.6
c	from table	from table	from table	9.8
r	-25	25	0.2	10.4
Parameter Space B				
a	0	15	0.2	0
b	5	30	0.2	7.2
c	5	30	0.2	8.8
r	-25	25	0.2	9.2

Parameter space "A" has a narrow scope but guarantees a kinematically favourable solution. Parameter space "B" ignores kinematic conditioning but results in the dynamic global optimum. In order to decide which solution is most favourable overall, the two are compared by a sensitivity analysis in Table IV. Since both solutions have similar dynamic performance but solution "A" has significantly better kinematic performance, solution "A" (*a=1.6, b=7.6, c=9.8, r=10.4*) is concluded to be the best overall design.

Table IV: Sensitivity Analysis

Parameter Space	Kinematic GII		Maximum Inertia	
	Value	% Change From Mean	Value	% Change From Mean
Solution A	0.3657	+13.5%	0.9045	-1.2%
Solution B	0.2790	-13.5%	0.9271	+1.2%
Mean	0.3224	0%	0.9158	0%

Special purpose robots such as haptic interfaces are hampered by large changes is singular values as well as by sudden changes in singular values. Smoothness is checked as a secondary measure for the optimal design by plotting the minimum and maximum singular values of the Jacobian and mass matrices over the workspace. As seen in Figure 12, they

are both smooth and even have regions of perfect local isotropy where the minimum and maximum singular value curves intersect (i.e. $\underset{\sim}{\sigma}(J \text{ or } D) = \tilde{\sigma}(J \text{ or } D)$).

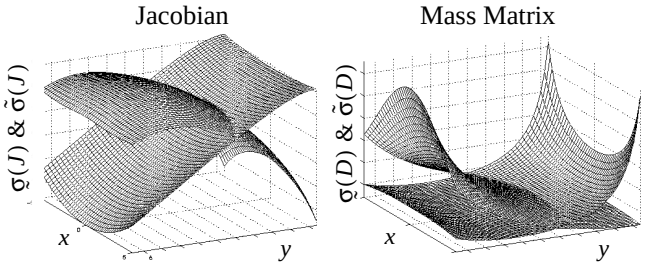

Figure 12: Singular Values of Jacobian & Mass Matrix

Three optimal geometries have resulted from the preceding discussion, each satisfying different design criteria including kinematic conditioning, dynamic conditioning and a combination of the two. Strictly speaking, a kinematic optimum does not exist since isotropy can always be improved by moving the workspace away from the base and increasing the link lengths. An optimum does exist for a fixed workspace position but this is not very helpful in practice since it is unclear where the optimal workspace position is located. A more natural method of keeping device sizes reasonable is to penalize large geometries by including dynamic criteria in the objective function. Unfortunately, a design optimization that relies solely on dynamic conditioning can result in a device with poor static performance. To get the best overall behaviour from a reasonably small device, one should consider both kinematic and dynamic criteria simultaneously. The difficulty of creating a weighted performance function is avoided here by narrowing down the parameter space with one criteria and finding the optimum with the other. It should, however, be noted that approaches such as this are quite computationally expensive.

5 Design of a Stewart Platform Based 6-DOF Device

The GII culling algorithm is next used to attempt a higher dimensional design optimization involving the 6-DOF Stewart Platform manipulator shown in Figure 13.

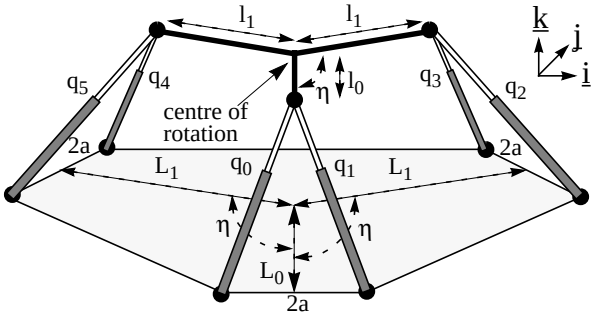

Figure 13: Stewart Platform

The manipulator is designed by exploring combinations of the design parameters a, L_0, L_1, η and the ratio $L_0/l_0=L_1/l_1$. The workspace is a cubic volume with all sides of length *10 cm* centred *25 cm* above the centre of the base. The range of orientations includes a solid angle of 30° traced by the $\underline{k}$ axis of the platform coordinate frame combined with all rotations of up to 30° about that axis. Symmetry of the workspace about the $\underline{jk}$ plane is used to reduce the number of geometric robot parameters by imposing the same symmetry upon the robot. The distances between the centre of the platform and the left and right pairs of actuators are equal (L_1) and the platform is shaped similarly to the base but is scaled by the ratio $L_0/l_0=L_1/l_1$. It is suggested in **29** that the physical units of the Jacobian of such a device can be normalized using a

Characteristic Length of *12 cm* if the device is to be used as a haptic interface. Using this value, the GII of the Jacobian matrix is optimized given the discrete parameter space shown in Table V and the discrete workspace shown in Table VI.

Table V: Stewart Platform Parameter Space

Parameter	Minimum	Maximum	Resolution	Optimum
a	1	20	0.5	16.5
L_0	1	20	0.5	10.0
L_1	1	20	0.5	10.0
L_0/l_0, L_1/l_1	0.5	1.5	0.1	0.7
η	100°	130°	1°	118

Table VI: Stewart Platform Workspace

Dimension	Minimum	Maximum	Resolution	Total
Translation				
i axis	0	5	0.625	9
j axis	-5	5	0.625	17
k axis	-5	5	0.625	17
Rotation				
i, j axes	Uniformly Sampled Solid Angle			168
k axis	-30˚	30˚	5	13

The optimum parameters shown in Table V produce a GII value of *0.281*. To see if the result is sensitive to sample spacing, the discretization resolution is halved for all parameters and workspace dimensions and the optimization is repeated. A GII of 0.285 results from the parameters $a=17$, $L_0=L_1=10$, $\eta=120^\circ$ and $L_0/l_0=L_1/l_1=0.7$. Since the GII fluctuates by only 1.4% and the optimum parameters differ by a maximum of only 3%, it is concluded that the original resolution is adequate and that little improvement can be expected in terms of a better (parameter space resolution) or more trustworthy (workspace resolution) solution from reducing the sample spacing.

6 Efficiency of the Culling Algorithm

The Culling algorithm belongs to the branch and bound family of optimization algorithms where all bounds are determined by explicit function evaluations. It avoids redundant evaluations by eliminating parameters that are shown to be sub-optimal and, therefore, always converges to a global optimum within the discretized parameter space. Each loop iteration removes at least one parameter from contention so the number of potential loop iterations is bounded by the dimension of the parameter space and the stopping criterion is always satisfied in finite time. A worst case scenario of no culling whatsoever results in an exhaustive global search. As with most optimization algorithms, efficiency depends on the objective function and initial conditions. While the algorithm makes no efficiency guarantees, experience with robot design problems has consistently shown dramatic improvement over a global search with low sensitivity to initial

conditions. Table VII compares the number of objective function evaluations performed by the culling algorithm to those required by a global search for the optimizations described in this paper.

Table VII: Culling/Global Search Effort Reduction

Optimization	Workspace Size	Param Space Size	Global Search : Culling Ratio
Table I	1.02×10^{4}	2.66×10^{7}	3670 : 1
Table II	5151	1.21×10^{6}	1910 : 1[A]
Table III - #1	5151	251	82 : 1
Table III - #2	5151	3.03×10^{8}	3500 : 1[B]
Table V	5.68×10^{6}	1.37×10^{7}	1.79×10^{6} : 1

A. Typical value for an optimization conducted for any one value of r

B. The parameter space was divided into 10 parts to overcome hardware (memory) limitations so the reported ratio is an average. Partitioning reduces the efficiency of the culling algorithm so the reported improvement ratio is conservative.

The algorithm is demonstrated to be effective at solving both modest (i.e. few dimensions) and complex (i.e. many dimensions) problems. An example involving the Stewart Platform with a workspace containing over *5 million* elements and a parameter space containing over *13 million* elements resulted in an effort reduction of over *6* orders of magnitude over a global search. This calculation which took just over *4* days to solve on a Sun Sparc 5 workstation using culling is estimated to require over 20,000 years to solve by a global search using a similar machine. By looking at the severity of culling during each loop iteration, we can identify where most of the computation effort takes place. The Stewart Platform optimization described in Table V and Table VI was solved in *5* loop iterations. Table VIII shows for each loop iteration the number of parameters initially present, the number of parameters culled, the maximum number of objective function evaluations required to update the singular value upper and lower bounds of all remaining parameters and the number of objective function evaluations required to search all workspace locations of the candidate parameter. Note that because the GII culling algorithm was used, up to two parameter evaluations (two different workspace locations for each parameter) can occur during each loop iteration.

Table VIII: Effort Breakdown of GII Culling

Loop Iteration	Number of Parameters	Parameters Culled	Parameter Evaluations	Workspace Evaluations
0	13,702,689	12,498,014	27,405,378	5,680,584
1	1,204,675	1,204,049	2,409,350	5,680,584
2	626	353	1,252	5,680,584
3	273	271	546	5,680,584
4	2	1	4	5,680,584
5	1	0	0	0
TOTAL	n/a	13,702,688	29,816,530	28,402,920

It is, however, possible to skip one or both parts of steps (11) and (23) so the number of total parameter space evaluations are bounded from below by the number of parameters in parameter space P_0. The minimum, maximum and actual number of calculations that took place are shown in Table IX.

Table IX: Function Evaluation Summary

	Parameter Evaluations	Workspace Evaluations	Total Evaluations
Minimum Possible	1.37×10^7	2.84×10^7	4.21×10^7
Maximum Possible	2.98×10^7	2.84×10^7	5.82×10^7
Actual	1.52×10^7	2.84×10^7	4.36×10^7

Computational effort is split quite evenly between parameter space and workspace searches. Parameter space searches leave little room for improvement since an average of only one or two positions are visited for each parameter but workspace searches might benefit from a different, perhaps stochastic, searching strategy. For example, workspace positions could be visited in steps (9) and (21) in a random order, stopping when $\hat{s}_{i+1} < \hat{s}_i$ which identifies p_i as suboptimal. If, however, this condition is never met, the workspace search can only be terminate after an exhaustive search has been completed so that the guarantee of global optimality is not compromised. Since the $\tilde{x}_i$ and/or $\underset{\sim}{x}_i$ produced from a truncated search are not expected to be as good as those produced by an exhaustive search, culling is likely to be less severe after the subsequent parameter space search but the net gain from truncating the workspace search could in many cases exceed the net loss from the reduction in parameter culling.

To determine sensitivity to initial conditions, the Stewart Platform optimization was repeated two more times, once with the initial condition set to the optimum solution from Table V and again with the initial condition set to the parameter farthest from the optimal solution (i.e. $a=1.0$, $L_0=L_1=20.0$, $\eta=130^\circ$ and $L_0/l_0=L_1/l_1=1.5$). The number of function evaluations resulting from each of the three trials is presented in Table X and shows that a good first guess can reduce computational effort by up to 12.7% over the mean value. One way of obtaining a good first guess is to increase the sample spacing and pre-optimize the device. The optimization of the Stewart Platform with double the original sample spacing that was described earlier required 8.77×10^5 function evaluations to complete. Adding this initial investment to the required computational effort when starting from the optimal solution reduces its 12.7% gross effort reduction to a 9.9% net effort reduction over the mean value. Since the effort reduction from having a good starting point is only expected to be in the neighborhood of 10%, it is only practical to pursue when the problem is very large and is expected to take a long time to complete. Otherwise, it is probably more practical to choose the starting point arbitrarily.

Table X: Sensitivity to Initial Conditions

Initial Conditions	Number of Evaluations	% Change From Mean
Arbitrary	4.36×10^7	+4.6%
Optimum	3.64×10^7	-12.7%
Farthest from Optimum	4.52×10^7	+8.4%
Mean	4.17×10^7	0%

As a final note, due to the low computational overhead of the culling algorithm, almost all of the processing power is consumed by objective function evaluations. The effort required to calculate a Jacobian and its singular values overwhelms the few conditional checks and assignment statements associated with the algorithm itself, especially in the case of a 6-DOF device. Since the culling algorithm performs blocks of function evaluations for large sets of parameters

and positions where the order of evaluations is of no importance, the algorithm is easily adapted to machines with parallel processing capabilities. One could reasonably expect a linear relationship between completion time and the number of available processors.

7 Conclusions

A new global isotropy index (GII) was proposed which defines isotropy as the ratio between the minimum and maximum singular values in the workspace. It summarizes the workspace inclusive performance of a mechanism by a scalar quantity and can be applied to the Jacobian, mass matrix or any other linear transformation describing a robot's performance. A novel optimization procedure was also proposed which belongs to the branch-and-bound family of optimization algorithms but is specifically designed for either GII or minimax optimization. It repeatedly uses the worst configuration of one parameter to eliminate others from contention until only the optimum remains. The approach guarantees convergence, finite time termination and a global result. The algorithm consistently displays drastic improvements over a global search with demonstrated effort reductions of up to six orders of magnitude. One design example described here took just over *4* days to solve on a Sparc 5 workstation using the culling algorithm. This same example is estimated to require over *20,000* years to solve by a global search on a similar machine. The culling algorithm, therefore, allows one to use unsophisticated computer hardware to solve high dimensional problems that are otherwise too computationally demanding to attempt. When used on more sophisticated hardware, the culling algorithm allows one to solve more complex or finely discretized problems within reasonable time frames and is also easily adapted to exploit the parallel processing capabilities of multi-processor machines.

The culling algorithm is used to optimize the kinematic GII and maximum inertia of a five-bar linkage based planar haptic interface. It is shown that the best overall architecture has left/right symmetry about the robot and workspace and that the robot is best kept in either an "M" or an elbow-out posture. A sensitivity analysis is performed to trade-off kinematic and dynamic performance for an overall optimum design. Another design example involving a 6-DOF Stewart Platform with *5* design parameters demonstrates that the culling algorithm is also very effective at solving large optimization problems with low sensitivity to initial conditions.

Acknowledgements

The authors would like to thank Dr. Philip D. Loewen and Dr. Richard Anstee for their comments during the revision of the manuscript. This work is supported by the Canadian IRIS Network of Centres of Excellence projects, HMI-6 and IS-8 and a scholarship from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] M.S. Campbell, T.A. Marsland, "A Comparison of Minimax Tree Search Algorithms", Artif. Intell., V. 20, pp. 347-367, (1983).

[2] H.D. Chiang, C.C. Chu, "A Systematic Search Method for Obtaining Multiple Local Optimal Solutions of Nonlinear Programming Problems", Proc. IEEE ANNPS 93 2nd Int. Forum on Neural Net. to Power Sys. (Yokohama, Japan), pp. 447-450, (Apr. 19-22, 1993).

[3] K. van den Doel, D.K. Pai, "Constructing Performance Measures for Robotic Manipulators", Proc. IROS '94, IEEE/RSJ/GI Int. Conf. Intell. Robots & Sys., Adv. Robotic Sys. & Real World (Munich, Germany), pp.1601-1607, (Sept. 12-16, 1994).

[4] K. van den Doel, D.K. Pai, "Redundancy and Non-linearity Measures for Robot Manipulators", Proc. IEEE Int. Conf. Robotics & Auto. (San Diego, California), pp. 1873-1880, (May 8-13, 1994).

[5] R.L. Fox, "Optimization Methods for Engineering Design", Addison-Wesley, Reading, Massachusetts, (1971).

[6] C. Gosselin, J. Angeles, “A Global Performance Index for the Kinematic Optimization of Robot Manipulators”, Trans. ASME, J. Mech. Des., V. 113, pp. 220-226, (Sept. 1991).

[7] G.M. Griner, “A Comparison of Simulated Evolution and Genetic Evolution Performance”, Proc. 1st IEEE Conf. on Evolutionary Computation, IEEE World Congress on Computational Intelligence (Orlando, Florida), pp. 374-378, (Jun. 27-29, 1994).

[8] V. Hayward, J. Choksi, G. Lanvin, C. Ramstein, “Design and Multi-Objective Optimization of a Linkage for a Haptic Interface”, Proc. ARK ‘94, 4th Int. Workshop on Adv. in Robot Kin. (Ljubliana, Slovenia), (Jun. 1994).

[9] J.B. Hunt, “Dynamic Vibration Absorbers”, Mechanical Engineering Publications, (1979).

[10] D.C. Jiang, K.L. Teo, W.Y. Yan, “A New Computational Method for the Functional Inequality Constrained Minimax Optimization Problem”, Proc. 34th IEEE Conf. Decision & Ctrl. (New Orleans, Los Angeles), pp. 2310-2315, (Dec. 13-15, 1995).

[11] J.P. Karidis, G. McVicker, J.P. Pawletko, L.C. Zai, M. Goldowsky, R.E. Brown, R.R. Comulada, “The Hummingbird Minipositioner -- Providing Three-Axis Motion at 50 G’s With Low Reactions”, Proc. IEEE Int. Conf. Robotics & Auto. (Nice, France), pp. 685-692, (May 10-15, 1992).

[12] D. Kennedy, “Some Branch and Bound Techniques for Nonlinear Optimization”, Mathematical Prog., V. 42, pp. 147-157, (1988).

[13] J-O. Kim, P.K. Khosla, “Dexterity Measures for Design and Control of Manipulators”, Proc. IROS ‘91, IEEE/RSJ Int. Workshop Intell. Robots & Sys. (Osaka, Japan), pp. 758-763, (Nov. 3-5, 1991).

[14] M.V. Kircanski, “Robotic Isotropy and Optimal Robot Design of Planar Manipulators”, Proc. IEEE Int. Conf. Robotics & Auto. (San Diego, California), pp. 1100-1105, (May 8-13, 1994).

[15] R. Kurtz, V. Hayward, “Multiple-Goal Kinematic Optimization of a Parallel Spherical Mechanism with Actuator Redundancy”, IEEE Trans. Robotics & Auto., V. 8, (Oct. 1992).

[16] J.T-H. Lo, “A New Approach to Global Optimization and its Applications to Neural Networks”, Proc. of IEEE IJCNN Int. Joint Conf. on Neural Net. (Baltimore, Maryland), V. 4, pp, 600-605, (Jun. 7-11, 1992).

[17] D.G. Luenberger, “Introduction to Linear and Nonlinear Programming”, Addison-Wesley, (1973).

[18] O. Ma, J. Angeles, “Optimum Design of Manipulators Under Dynamic Isotropy Conditions”, Proc. IEEE Int. Conf. Robotics & Auto. (Atlanta, Georgia), pp. 470-475, (May 2-6, 1993).

[19] O. Ma, J. Angeles, “The Concept of Dynamic Isotropy and Its Applications to Inverse Kinematics and Trajectory Planning”, Proc. IEEE Int. Conf. Robotics & Auto. (Cincinnati, Ohio), pp. 481-486, (May 13-18, 1990).

[20] R.V. Mayorga, B. Ressa, A.K.C. Wong, “A Kinematic Design Optimization of Robot Manipulators”, Proc. IEEE Int. Conf. Robotics & Auto. (Nice, France), pp. 396-401, (May 10-15, 1992).

[21] J-P. Merlet, “A DEsign MethOdology for the Conception of Robot with parallel ArchiTecture”, Internal Report, INRIA Sophia-Antipolis, (1996).

[22] E.R. Panier, A.L. Tits, “A Globally Convergent Algorithm with Adaptively Refined Discretization for Semi-Infinite Optimization Problems Arising in Engineering Design”, IEEE Trans. Auto. Ctrl., V. 34, No. 8, pp. 903-908, (Aug. 1989).

[23] F.C. Park, R.W. Brockett, “Harmonic Maps and the Optimal Design of Mechanisms”, Proc. 28th IEEE Conf. Decision & Ctrl. (Tampa, Florida), pp. 206-210, (Dec. 1989).

[24] F. Schoen, “Stochastic Techniques for Global Optimization: A Survey of Recent Advances”, J. Global Opt., V. 1, pp. 207-228, (1991).

[25] Z. Shiller, S. Sundar, “Design of Robotic Manipulators for Optimal Dynamic Performance”, Proc. IEEE Int. Conf. Robotics & Auto. (Sacramento, California), pp. 344-349, (Apr. 9-11, 1991).

[26] M.W. Spong, M. Vidyasagar, “Robot Dynamics and Control”, John Wiley & Sons, (1989).

[27] M. Srinivas, L.M. Patnaik, “Genetic Algorithms: A Survey”, Computer, V. 27, Iss. 6, pp. 17-26, (Jun. 1994).

[28] L. Stocco, S.E. Salcudean, “A Coarse-Fine Approach to Force-Reflecting Hand Controller Design”, Proc. IEEE Int. Conf. Robotics & Auto. (Minneapolis, Minnesota), V. 1, pp. 404-410, (Apr. 22-28, 1996).

[29] L. Stocco, S.E. Salcudean, F. Sassani, “Mechanims Design for Global Isotropy with Applications to Haptic Interfaces”, Proc. ASME Winter Annual Meeting, Sixth Annual Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems, Dynamic Systems & Control (Dallas, Texas), V. 61, pp. 115-122, (Nov. 15-21, 1997).

[30] B. Stuckman, E. Easom, “A Comparison of Bayesian/Sampling Global Optimization Techniques”, IEEE Trans. Sys., Man & Cyb., V. 22, No. 5, pp. 1024-1032, (Sept./Oct. 1992).

[31] B. Stuckman, G. Evans, M. Mollaghasemi, “Comparison of Global Search Methods for Design Optimization Using Simulation”, Proc. 1991 IEEE Winter Simulation Conf. (Phoenix, Arizona), pp. 937-944, (Dec. 8-11, 1991).

[32] M. Teboulle, “Nonlinear Perturbations for Linear Semi-Infinite Optimization Problems”, Proc. 29th IEEE Conf. Decision & Ctrl. (Honolulu, Hawaii), pp. 2477-2478, (Dec. 5-7, 1990).

[33] Y. Yao, “Dynamic Tunneling Algorithm for Global Optimization”, IEEE Trans. Sys., Man, & Cyb., V. 19, No. 5, (Sept./Oct. 1989).

[34] T. Yoshikawa, “Manipulability of Robotic Mechanisms”, Int. J. Robotics Res., V. 4, No. 2, pp.3-9, (Summer 1985).

[35] Y. Zheng, W.L. Lewis, “Several Algorithms of Global Optimal Search”, Adv. Eng. Software, V. 21, pp. 87-98, (1994).

List of Captions

Figure 1: Constrained Planar Elbow Manipulator. 3
Figure 2: Torque Ellipse at x=5. 3
Figure 3: Force/Torque Transformation 3
Figure 4: Force/Torque Ellipses and GII. 4
Figure 5: Surface and Contour Plots of Dexterity. 10
Figure 6: First Culling of Non-Optimal Geometries. 11
Figure 7: Second Culling of Non-Optimal Geometries 12
Figure 8: Optimum Solution 13
Figure 9: Generalized Five-Bar Linkage. 14
Figure 10: Optimal Postures of Five-Bar Linkage 16
Figure 11: Singular Values of Jacobian & Mass Matrix. 17
Figure 12: Stewart Platform 12

Table I: Parameter Space & Optimum 17
Table II: Reduced Parameter Space 18
Table III: Inertial Parameter Space & Optima. 19
Table IV: Sensitivity Analysis 20
Table V: Stewart Platform Parameter Space 20
Table VI: Stewart Platform Workspace 20
Table VII: Culling/Global Search Effort Reduction 21
Table VIII: Effort Breakdown of GII Culling 22
Table IX: Function Evaluation Summary 23