Exact interval propagation for the efficient solution of planar linkages

E. Celaya, T. Creemers, and L. Ros*
Institut de Robòtica i Informàtica Industrial (UPC-CSIC)
Llorens Artigas 4-6, 08028 Barcelona, Spain

Abstract— *This paper presents an interval propagation algorithm for variables in single-loop linkages. Given allowed intervals of values for all variables, the algorithm provides, for every variable, the exact interval of values for which the linkage can actually be assembled. We show further how this algorithm can be integrated in a branch-and-bound search scheme, in order to solve the position analysis of general multi-loop linkages. Experimental results are included, comparing the method's performance with that of previous techniques given for the same task.*

Keywords: Interval propagation, planar linkages, box approximation, loop closure equation, position analysis.

I. Introduction

In recent years there has been a growing interest in the use of interval methods in kinematics, specially for the task of finding the solutions of complex linkages which, when performed analytically, may involve either a large number of equations, or high-degree polynomials [1], [2], [3], [4], [5]. When the solution space is continuous, interval methods can provide a discrete approximation of such space as a set of enclosing boxes, which can be refined to the desired precision [6], [7]. Advantages of interval methods include their completeness, the fact that they do not require complex algebraic manipulations, and that the solution set is not populated with imaginary values.

At the core of interval methods there uses to be a branch and bound algorithm: given an initial set of intervals for the involved variables, which determine an initial box where solutions will be sought, a bounding method is applied to reduce the box by eliminating regions that cannot contain a solution. When the bounding method cannot further reduce the box, it is split, usually in two halves (branches), each of which is treated recursively by the same process. The bounding method may be a general interval propagation algorithm applied to the kinematic equations of the linkage [3], [4], [5], or a more specific method suited to the problem [1], [6], [7].

Often, interval methods are, by design, subject to overestimation [8]. This fact prevents reducing a box as much as possible before branching, which may result in important efficiency losses. There is a trade-off between the accuracy of the bounding process and its efficiency: an excessive complexity of the bounding algorithm may overshadow its better accuracy in the reduction of a box, resulting in a globally less efficient algorithm.

In this paper we present an exact, though simple, interval propagation algorithm for variables in planar kinematic loops with revolute pairs. The algorithm provides the exact angle intervals for which a solution exists, assuming the remaining angles can vary within specified ranges. The paper also shows how this algorithm may be embedded within a branch-and-bound process in order to solve complex, multi-loop linkages as described above. Comparisons with alternative general techniques for planar mechanisms show an improvement in performance in many cases.

The paper is organized as follows. Section II considers the case of single-loop linkages and derives the exact intervals enclosing all joint angles for which the loop closes, assuming all joints can freely rotate within the $[0, 2\pi]$ range. The following three sections compute the same intervals, but assuming that further constraints hold on the joint ranges: Section III assumes one joint is held fixed, Section IV assumes one joint can only move within a sub-interval of $[0, 2\pi]$, and Section V assumes all joints move within different sub-intervals each. Based on these results, Section VI considers the case of multi-loop linkages and derives a branch-and-bound algorithm able to isolate all linkage configurations satisfying the required loop closure conditions. Section VII shows some experiments on an implementation of this algorithm and, finally, Section VIII concludes the paper and highlights points deserving further attention.

II. Solution intervals in single loop linkages

For clarity purposes, we limit our analysis here to planar linkages with revolute joints. Similar results can be obtained for planar linkages with both revolute and slider joints, as well as for spherical linkages [9].

Given a single-loop planar linkage with n bars joined by n revolute pairs, the goal is to determine the set of valid values for each joint angle for which there exists a solution (i.e., the linkage closes). We can express the closure condition of an n-bar planar linkage as

$$\mathbf{Rz}(\theta_1)\mathbf{Tx}(l_1)\ldots\mathbf{Rz}(\theta_n)\mathbf{Tx}(l_n) = \mathbf{I}, \tag{1}$$

where $\mathbf{Rz}$ are rotations around the z axis, $\mathbf{Tx}$ are translations along the x axis, l_i are constant lengths, and θ_i are the (exterior) angles between them.

*E-mails:celaya/creemers/ros@iri.upc.edu

The simplest possible loop, with $n = 2$, is trivial and has a solution only if $l_1 = l_2$, in which case we have $\theta_1 = \theta_2 = \pi$.

For $n = 3$ the closure condition is given by the triangle inequalities $|l_a - l_b| \leq l_c \leq l_a + l_b$. By the cosine rule, we have: $l_c^2 = l_a^2 + l_b^2 + 2l_a l_b \cos(\theta_b)$, and thus

$$\theta_b = \arccos\left(\frac{l_c^2 - l_a^2 - l_b^2}{2l_a l_b}\right) \tag{2}$$

When the triangle inequalities hold, eq. (2) provides two solutions for θ_b corresponding to the positive and negative determinations of the arccos function. A single solution exists when the argument of the arccos function is ± 1.

In the general case, for $n \geq 3$, an infinite number of solutions is possible. The closure condition is that the lengths l_i are such that a closed polygon can be formed with them. A way to express this condition that is convenient for our purposes involves the length L_k of the segment connecting joints $(k-1)$ and $(k+1)$, which is a function of θ_k and its adjacent link lengths $l_{(k-1)}$ and l_k (Fig. 1) given by the cosine rule:

$$L_k^2 = l_{(k-1)}^2 + l_k^2 + 2l_{(k-1)} l_k \cos(\theta_k). \tag{3}$$

Clearly, the condition for the polygon to close is that the links $l_i, i \neq (k-1), k$ can be made to reach some length L_k compatible with eq. (3), that is, some value in the interval $I_k = [\,|l_{(k-1)} - l_k|,\, l_{(k-1)} + l_k]$. To determine this, we prove the following:

Lemma: The set $\{L_m\}$ of lengths reachable by m links of lengths l_i joined by revolute joints are all the values in the interval

$$I_m = [\,max(0,\, l_M - \sum_{i=1, i\neq M}^{m} l_i),\ \sum_{i=1}^{m} l_i\,],$$

where l_M is the longest of the l_i: $l_M \geq l_i, i = 1, \ldots, m$.

The proof is by induction: We assume that the result is true for $m - 1$ links and prove it for m links. Clearly the maximum value for L_m is reached when all angles $= 0$, which corresponds to the upper bound of the interval I_m. Varying one or more of the θ_i the length L_m will vary continuously with a minimal value that we will determine next. We distinguish two cases:

a) In the case of $l_M > \sum_{i=1, i\neq M}^{m} l_i$, the lower bound of the interval I_m is just $l_M - \sum_{i=1, i\neq M}^{m} l_i$, and this value can be reached by making all $\theta_i = 0$, except for the two angles next to l_M, for which we make $\theta_M = \theta_{(M-1)} = \pi$.

b) In the case $l_M \leq \sum_{i=1, i\neq M}^{m} l_i$, the lower bound of the interval I_m is 0, and we must prove that it can be reached with the m links. This amounts to proving that the length l_M can be reached with the remaining $m-1$ links. Let $l_{M'}$ be the second longest link: $l_{M'} \geq l_i, i = 1, \ldots, m; i \neq M$. By induction hypothesis, the $m - 1$ links can reach any length in the interval

$$I_{(m-1)} = [max(0, l_{M'} - \sum_{i=1, i\neq M, i\neq M'}^{m} l_i),\ \sum_{i=1, i\neq M}^{m} l_i].$$

Now we have:

$$l_M > 0,$$

$$l_M > l_{M'} > l_{M'} - \sum_{i=1, i\neq M, i\neq M'}^{m} l_i,$$

and

$$l_M \leq \sum_{i=1, i\neq M}^{m} l_i,$$

and therefore $l_M \in I_{(m-1)}$, so that according to the induction hypothesis, it can be effectively reached by the $m - 1$ links.

Next we prove the lemma for $m = 3$ to complete the induction.

It is enough to prove that for $m = 3$, case b) is satisfied. We rename the links so that $l_a \leq l_b \leq l_c$, and we assume that $l_c \leq l_a + l_b$.

We have:

$$|l_b - l_a| \leq l_b \leq l_c \leq l_a + l_b.$$

So, the triangle inequalities, the necessary and sufficient conditions for three sides to form a triangle, are fulfilled by links l_a, l_b, l_c, which means that the distance between the ends of the links can be made equal to zero.

$\square$

Thus, the condition for Equation (1) to have a solution is that the intervals I_k and I_m have a non-empty intersection. In this case, the set S_k of valid values for θ_k can be obtained from the interval $I_k \cap I_m$ of valid values for L_k simply by transforming the bounds of this interval into the corresponding values of θ_k using (3):

$$\theta_k = \arccos\left(\frac{L_k^2 - l_{(k-1)}^2 - l_k^2}{2l_{(k-1)} l_k}\right). \tag{4}$$

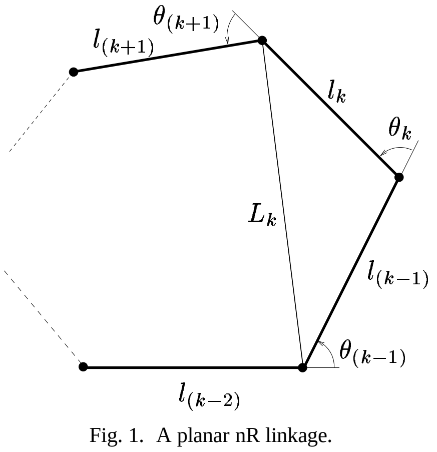


Fig. 1. A planar nR linkage.

Since the arccos function is double-valued, we will obtain two symmetric intervals S_k^+ and S_k^-, corresponding to its positive and negative determinations, respectively. So, in general, S_k is formed by two symmetric intervals that may join into a single interval or even into the whole circumference in the case that these intervals reach the value 0 or π.

III. Propagating a single value to other variables

If one variable θ_i is fixed to a given value, we can readily find the set of compatible values for the remaining variables. We just need to substitute this value in the loop equation and rewrite it so that it takes the form given in Eq. (1). The new equation is equivalent to a new linkage with $n-1$ bars, which can be solved for their variables as explained in the previous section. The basic rule for rewriting the equation is the following: If variable θ_i is fixed to a value α, the loop equation will contain a subchain of the form:

$$...\mathbf{Rz}(\theta_{(i-1)})\mathbf{Tx}(l_{(i-1)})\mathbf{Rz}(\alpha)\mathbf{Tx}(l_i)\mathbf{Rz}(\theta_{(i+1)})... \quad (5)$$

which can be substituted by

$$...\mathbf{Rz}(\theta_{(i-1)}+\beta)\mathbf{Tx}(l_\alpha)\mathbf{Rz}(\theta_{(i+1)}+\gamma)... \quad (6)$$

where

$$l_\alpha = +\sqrt{l_{(i-1)}^2 + l_i^2 + 2l_{(i-1)}l_i cos\alpha}$$

$$\beta = \arctan 2(l_i \sin\alpha, l_{(i-1)} + l_i\cos\alpha)$$

$$\gamma = \alpha - \beta = \arctan 2(l_{(i-1)}\sin\alpha, l_i + l_{(i-1)}\cos\alpha)$$

Note that care must be taken to check whether the new length l_α is 0, in which case the equation should be simplified in a trivial way. We denote by $P_{ik}(\alpha)$ the set of values for θ_k obtained by propagation of the value $\theta_i = \alpha$.

IV. Interval propagation between two variables

In some situations, a given variable may have been constrained to take not just a specific value, but any value inside an interval $I_i = [\alpha, \beta]$. In this case, we are interested in the set of all the allowed values for variable θ_k that are compatible with at least one value $\theta_i \in I_i$. We call this the interval propagation from variable θ_i to variable θ_k in a loop equation. In essence, what we need to obtain is the union of all the intervals obtained for θ_k after substituting variable θ_i for each value in I_i, that is, the set

$$P_{ik}([\alpha,\beta]) = \bigcup_{\phi\in[\alpha,\beta]} P_{ik}(\phi).$$

We next devise an interval propagation algorithm to compute $P_{ik}([\alpha,\beta])$ in an efficient way.

Let us consider the graph of $P_{ik}(\theta_i)$ (Fig. 2). As we have seen, for each value ϕ of θ_i, the set of compatible values of θ_k is composed by two symmetric intervals, which we denote by $P_{ik}^+(\phi)$ and $P_{ik}^-(\phi)$, respectively. In what follows we use the notation $P_{ik}^\sigma(\phi)$, with $\sigma = \pm1$, to denote them.

We define the functions $L_{ik}^\sigma(\theta_i)$ and $U_{ik}^\sigma(\theta_i)$ as the lower and upper values, respectively, of $P_{ik}^\sigma(\theta_i)$. It is clear that, if $S_k^\sigma = [l_k^\sigma, u_k^\sigma]$ is the corresponding determination of the whole feasible set S_k for θ_k, then the minimum value taken by the function $L_{ik}^\sigma(\theta_i)$ is l_k^σ, and the maximum value taken by the function $U_{ik}^\sigma(\theta_i)$ is u_k^σ.

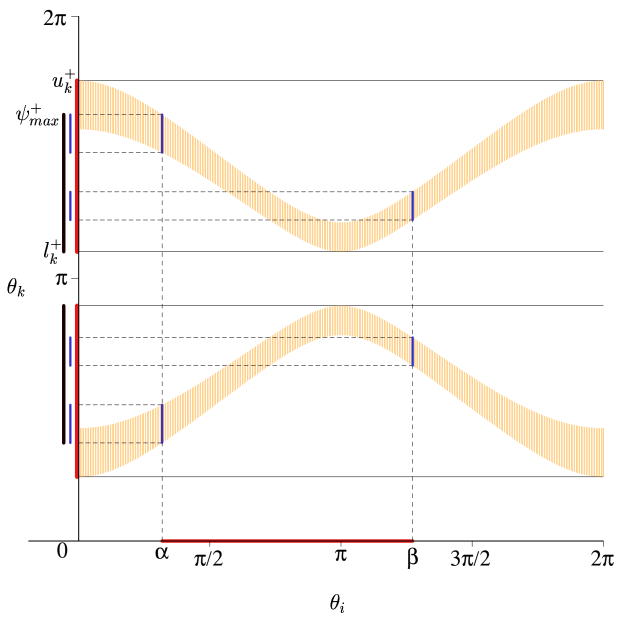


Fig. 2. Graph $P_{ik}(\theta_i)$ for variables $\theta_i = \theta_2$ and $\theta_k = \theta_5$ for the equation $Rz(\theta_1)Tx(4)Rz(\theta_2)Tx(8)Rz(\theta_3)Tx(1)Rz(\theta_4)Tx(8)Rz(\theta_5)Tx(6) = I$, and propagation of an interval $[\alpha,\beta]$.

Now observe that the graph of $P_{ik}(\theta_i)$ can also be interpreted as the graph of the inverse mapping $P_{ki}(\theta_k)$ when looking at it rotated 90°, so, any horizontal slice of the figure must also consist of two (connected or not) symmetric intervals. This fact allows us to establish that $L_{ik}^\sigma(\theta_i)$ (resp. $U_{ik}^\sigma(\theta_i)$) cannot have a local minimum (resp. maximum) with a value different from the global one because, in such case, the inverse mapping would not show the required symmetry. Thus we can assert that if $L_{ik}^\sigma(\theta_i)$ (resp. $U_{ik}^\sigma(\theta_i)$) has a local minimum (resp. maximum), its value is precisely that of the lower bound l_k^σ (resp. the upper bound u_k^σ) of the interval S_k^σ.

Since $P_{ik}^\sigma(\theta_i)$ is a continuous mapping, in the sense that its bounding functions $L_{ik}^\sigma(\theta_i)$ and $U_{ik}^\sigma(\theta_i)$ are continuous, $P_{ik}^\sigma([\alpha,\beta])$ will consist of a single interval $[\psi_{min}^\sigma, \psi_{max}^\sigma]$, whose bounds are given by the extrema of $L_{ik}^\sigma(\theta_i)$ and $U_{ik}^\sigma(\theta_i)$ in the interval $[\alpha,\beta]$:

$$\psi_{min}^\sigma = \min_{[\alpha,\beta]}(L_{ik}^\sigma(\theta_i)); \quad \psi_{max}^\sigma = \max_{[\alpha,\beta]}(U_{ik}^\sigma(\theta_i)).$$

Thus, all we need is to determine these two extreme values. According to the Extreme Value Theorem, the lower value taken by the function $L_{ik}^\sigma(\theta_i)$ in the interval $[\alpha,\beta]$

occurs either at an interior local minimum point or at one bound of the interval. Since we know that the only possible value for a local minimum is l_k^σ, it is enough to check whether this value is included in $P_{ik}^\sigma([\alpha,\beta])$ or not: If it is, then $\psi_{min}^\sigma = l_k^\sigma$, otherwise, $\psi_{min}^\sigma = \min(P_{ik}^\sigma(\alpha), P_{ik}^\sigma(\beta))$. Checking whether l_k^σ is included in $P_{ik}^\sigma([\alpha,\beta])$ can be done by computing its anti-image through the inverse propagation $P_{ki}(l_k^\sigma)$ and checking whether its intersection with $[\alpha,\beta]$ is empty or not. An analogous reasoning can be done for the upper extreme ψ_{max}^σ.

Summarizing, the algorithm for interval propagation from θ_i to θ_k, taking into account the positive and negative determinations, requires finding the solution set for the output variable θ_k according to the method presented in Section II, the propagation of the two bounds of the interval of the input variable θ_i as explained in Section III, and, eventually, some additional propagations of the bounding values of the output variable θ_k.

V. Multiple interval propagation

The propagation algorithm described above is not enough for our purposes. Consider a situation in which two variables, θ_i and θ_j, have been constrained to the intervals I_i and I_j, respectively, and we want to find the set of compatible values for θ_k. Obviously, a compatible value ξ of θ_k must satisfy $\xi \in P_{ik}(I_i) \cap P_{jk}(I_j)$. However, this condition is not sufficient. To show this, let us assume that the equation has two discrete solutions for $\theta_k = \xi$, the first yielding $\{\theta_i = \psi_i^+; \theta_j = \psi_j^+\}$ and the second $\{\theta_i = \psi_i^-; \theta_j = \psi_j^-\}$. If θ_i is constrained to an interval including ψ_i^+, but not ψ_i^-, and θ_j is constrained to an interval including ψ_j^-, but not ψ_j^+, then none of the solutions satisfies both constraints at the same time, and ξ is not compatible with them despite the fact that $\xi \in P_{ik}(I_i) \cap P_{jk}(I_j)$.

What we need in such situations is a *multiple interval propagation* algorithm that provides the set of values of a variable that are simultaneously compatible with all the interval constraints imposed on other variables. The general problem can be posed in this way: given a set of variables[1] $\theta_1, \ldots, \theta_r$ and a set of interval constraints $I_1, \ldots, I_r$ for them, find the set $P_{1\ldots r,k}(I_1, \ldots, I_r)$ of values of θ_k for which there is at least one solution with $\theta_1 \in I_1, \ldots, \theta_r \in I_r$.

To find this set, we next generalize the algorithm for single interval propagation explained in Section IV. In this case, we must consider the graph of $P_{1\ldots r,k}(\theta_1, \ldots, \theta_r)$, which, as before, is composed of two symmetric parts $P_{1\ldots r,k}^\sigma(\theta_1, \ldots, \theta_r)$, with $\sigma = \pm 1$.

Like in the $r = 1$ case, these functions satisfy the important property that all their local extrema are also global extrema. This can be directly proved for the case $r = 2$ corresponding to $P_{ij,k}(\theta_i, \theta_j)$, and, for higher dimensions, by induction on the number of input variables r. The proof, that will not be reproduced here for space limitations, is based on two main facts:
1. The projection of $P_{ij,k}(\theta_i, \theta_j)$ on the θ_i-θ_k plane is the graph of $P_{ik}(\theta_i)$, which we already know satisfies the property.
2. The *2-dimensional slice of the graph* $P_{ij,k}(\theta_i, \theta_j)$ at $\theta_i = \psi$, defined as the function $P_{ij,k}(\psi, \theta_j)$, corresponds to the graph $\overline{P}_{jk}(\theta_j)$ of the linkage that results from fixing $\theta_i = \psi$. Therefore, the slice must also satisfy the property of equivalence between local and global extrema.

In sketch, the proof shows that the existence of an extreme of $P_{ij,k}(\theta_i, \theta_j)$ in a point $(\theta_i, \theta_j) = (\psi, \phi)$ with the value $\theta_k = x$, implies that the 2-dimensional slice of $P_{ij,k}(\theta_i, \theta_j)$ at $\theta_i = \psi$ also shows this extreme in $\theta_j = \phi$, and this implies that the projection $P_{ik}(\theta_i)$ also presents this extreme in $\theta_i = \psi$. It follows that the extreme value x must be a global extreme of $P_{ij,k}(\theta_i, \theta_j)$.

Hence, we can apply the Extreme Value Theorem in the box $I_1 \times \ldots \times I_r$ of $P_{1\ldots r,k}^\sigma(\theta_1, \ldots, \theta_r)$. Exactly as in Section IV, we have to find the extreme values ψ_{min}^σ and ψ_{max}^σ, which must be either l_k^σ, u_k^σ, or the extreme values of $P_{1\ldots r,k}^\sigma(\theta_1, \ldots, \theta_r)$ at the boundary of the box. The only difficulty is that, in this case, the boundary of the box, instead of consisting of the two bounds of an interval, is composed of $2r$ hyperplanes, or boxes of dimension $r - 1$. Thus, the problem reduces to computing the extreme values in each of these $(r-1)$-dimensional boxes. For this, we just need to compute $\overline{P}_{1\ldots(j-1)(j+1)\ldots r,k}^\sigma(I_1, \ldots, I_{(j-1)}, I_{(j+1)}, \ldots, I_r)$, i.e., the propagation of all intervals except I_j in the equation obtained by substituting θ_j with the value that defines this hyperplane. Thus, we can reduce the problem of propagation of r intervals to $2r$ problems of propagation of $r - 1$ intervals. If we apply this reduction recursively, the problem will be reduced to computing a number of single interval propagations, which we already know how to perform.

This multiple interval propagation algorithm will be used below to infer feasibility ranges for n variables, constrained to a given n-dimensional box $I_1 \times \ldots \times I_n$. To this end, for each variable θ_k, a propagation of the intervals for the remaining variables towards θ_k is performed, and the result is then intersected with interval I_k. Note that each process of propagation and intersection can give rise to one or more disjoint intervals.

VI. Solving multi-loop linkages

The preceding section has showed how to compute exact ranges for all angles of a single loop. General linkages, though, may consist of multiple interconnected loops (Fig. 3), and the angles must be compatible with all loop equations simultaneously. No procedure is known to compute the exact ranges in such cases but, as shown next, the previous propagation scheme can be used to estimate them iteratively, and also to isolate all linkage configurations satisfying the equations.

[1] Without loss of generality, and to ease the notation, we rename the input variables with consecutive indices 1 to r

To derive feasibility ranges for all angles we proceed as follows. We start gathering all equations of the form of Eq. (1) for all loops of the linkage[2]. For each equation, we (1) label as *shared* those angles appearing in some other equation, and (2) infer feasibility ranges for all shared angles, as described in Section V above. Any time an interval is reduced as a consequence of such inference, its angle is said to have been "touched". The touching of an angle will trigger all constraints it appears in (except the one that provoked the touch), and all triggered constraints are kept on a "wake-up" stack. To ensure that all possible propagations are performed, we repeatedly select one equation from the stack, infer new ranges for its shared angles, and trigger any other equations involving them, until the stack gets empty. The latter situation is known as a *fixpoint* of the propagation process [11], meaning that all intervals are locally consistent with each equation.

To isolate all feasible configurations, we embed the previous process within a standard branch-and-bound algorithm. Two operators will be employed to this end, TRIM-BOX and SPLIT-BOX. The former prunes portions of a box containing no solution, using the propagation mechanism of the previous paragraph. The latter simply bisects a box into two sub-boxes by dividing its largest interval at its midpoint. Initially, the algorithm applies TRIM-BOX to the whole box $[0, 2\pi]^n$. As mentioned earlier, such reduction may yield several "child" boxes in general, and either one of three actions is applied on each one of them:
1. If the box has an empty interval, then it contains no solution and we label it as *empty*.
2. If the box is "small enough", then we label it as a *solution* box.
3. If the box is not "small enough", then we bisect it in into two sub-boxes using SPLIT-BOX.
If the last case occurs, the whole process is recursively applied to the new sub-boxes until all non-empty boxes are "small enough". A small-enough box is defined as one for which all its intervals are shorter than a given threshold σ.

Note that, on termination, this process will have explored a tree of boxes whose internal nodes are boxes being split at some time, and whose leaves are either solution or empty boxes. Solution boxes form an approximation of the linkage configuration space and are returned as output. Note also that the accuracy of such output can be adjusted through the σ threshold. In other words, the lower the chosen σ, the less the error committed when approximating a solution by any point inside a box.

VII. Experiments

The previous algorithm has been implemented in C++, and all CPU times will be given for a Pentium IV processor at 3.4 GHz, under Windows XP.

Results for two experiments are provided. The first one solves the position analysis of the "double butterfly" linkage (Fig. 3) when θ_3 is a fixed, known angle, which yields a finite number of isolated solutions. The second one solves the same linkage but assuming that θ_3 is a free variable, yielding a 1-dimensional continuum of solutions. The same benchmarks have been used previously to show the performance of elimination [12], [13], continuation [14], [15], and relaxation techniques [7]. We compare our results with those derived by such techniques, and employ the same linkage dimensions used in these papers. Namely:

$$\begin{array}{llll}
a_0 = 7 & b_0 = 13 & \gamma_0 = 36.87^\circ & \\
a_1 = 7 & b_1 = 6 & \gamma_1 = 22.62^\circ & \\
a_2 = 5 & b_2 = 3 & \gamma_2 = 53.13^\circ & \\
a_6 = 3 & b_6 = 2 & \gamma_6 = 36.87^\circ & \\
l_3 = 7 & l_4 = 9 & l_5 = 12 & l_7 = 11
\end{array}$$

A. A rigid butterfly

The number of solutions of the double butterfly linkage varies depending on the choice of driving joint and the angle given to it. If we set $\theta_3 = 75.75^\circ$, the number of obtained solutions is six [7], [12], [13]. They are given in Table I. Actually the cited papers employ absolute orientation angles for the links. Fixing our $\theta_3 = 75.75^\circ$ corresponds to fixing $\theta_6 = 67.38^\circ$ in those works. We note that, while continuation and elimination methods must filter the solutions among the eighteen possible complex roots, the one given here directly provides the six real solutions shown in the table. The obtained solutions are in accordance with those in [7], [12], [13].

Due to the nature of the algorithm all solutions are obtained as intervals that bound them, which allows estimat-

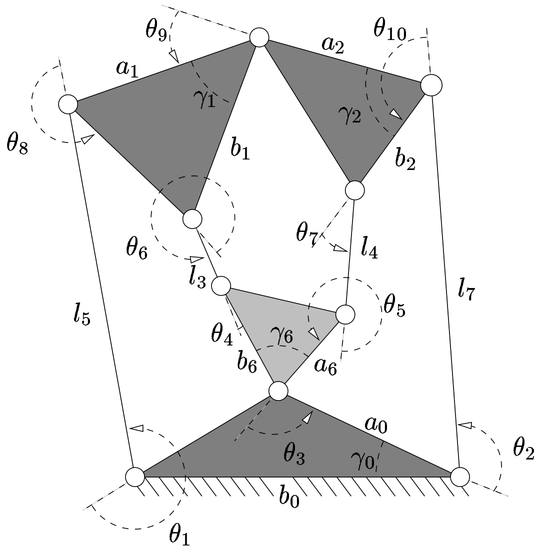


Fig. 3. The double butterfly linkage involves three interconnected loops.

[2] In practice, only the equations of a loop basis of the linkage are sufficient. Note however that the longer a loop is, the less constrained its angles result. Thus, among all possible loop bases, the *minimal* ones are preferred in practice [10], as they keep the loop lengths to a minimum

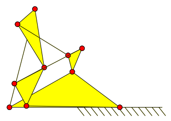
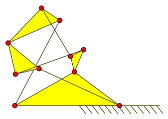
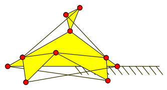
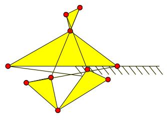
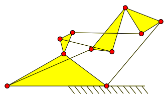
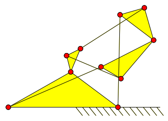
						
θ_1	[3.94335, 3.94335]	[3.71220, 3.71220]	[2.48312, 2.48318]	[2.49296, 2.49301]	[3.03749, 3.03750]	[3.03639, 3.03642]
θ_2	[3.77017, 3.77017]	[3.35355, 3.35356]	[3.95859, 3.95862]	[3.96481, 3.96482]	[1.51266, 1.51266]	[2.19170, 2.19172]
θ_4	[5.51396, 5.51396]	[5.99340, 5.99340]	[2.63872, 2.63877]	[3.02025, 3.02028]	[2.06012, 2.06014]	[2.22075, 2.22080]
θ_5	[3.83643, 3.83643]	[3.97137, 3.97138]	[3.60317, 3.60322]	[3.13912, 3.13917]	[1.19287, 1.19287]	[0.60626, 0.60635]
θ_6	[1.86725, 1.86726]	[2.70201, 2.70202]	[0.68130, 0.68133]	[5.53558, 5.53563]	[3.02443, 3.02445]	[3.27436, 3.27438]
θ_7	[4.69841, 4.69841]	[3.25715, 3.25716]	[5.28944, 5.28951]	[0.97042, 0.97049]	[5.71000, 5.71002]	[3.43216, 3.43218]
θ_8	[2.54508, 2.54508]	[1.46203, 1.46204]	[1.78324, 1.78326]	[2.82075, 2.82078]	[5.74756, 5.74760]	[5.33808, 5.33812]
θ_9	[0.58905, 0.58906]	[4.25173, 4.25174]	[5.00799, 5.00809]	[1.27232, 1.27238]	[1.25375, 1.25376]	[4.26191, 4.26195]
θ_{10}	[5.22246, 5.22246]	[0.66219, 0.66222]	[4.67617, 4.67623]	[3.16983, 3.16988]	[2.82874, 2.82876]	[5.01411, 5.01414]

TABLE I. The six possible configurations of the double butterfly linkage for $\theta_3 = 1.322$ rad ($75.75°$).

ing the error with respect to the exact position of the roots. This error must be equal or less than the chosen σ threshold, which was set to 10^{-4} in this case. The solutions were found in 0.83 sec of CPU time, after processing 11 boxes. From them, only the six shown in Table I were considered as solutions (thus returning the minimum possible number of boxes) and 5 boxes were found to be empty.

It is difficult to tell at this point whether the presented algorithm outperforms the previous methods based on Dixon's resultant [12], [13], mainly because no statistics are given in this respect in those works, and we have found no publicly available package implementing them. We have checked, though, that our method converges in substantially shorter times than those used by the continuation method in [14], [15], using the implementation available from [16], which spent about 8 seconds of CPU time on the same example, running on the same machine. We remark, though, that we are comparing our algorithm with a general-purpose solver targeted to arbitrary systems of algebraic equations, and that a better performance of our algorithm was to be expected, given that we exploit the specific structure of the obtained equations. Moreover, in the experiments done on this and other rigid linkages, the algorithm converges at similar rates than those of relaxation methods [7], [17].

B. *A mobile butterfly*

If we now free θ_3, a one dimensional continuum of solutions is obtained. Figure 4 depicts the projection of the returned boxes onto the θ_3-θ_4 plane, on three different runs of the algorithm, at decreasing values of the σ parameter. If the algorithm is exploring in breadth-first order, the first two plots can also be interpreted as earlier stages of the run for the third case ($\sigma = 0.005$). In every plot we indicate the σ threshold, the CPU time spent (t), the number of solution boxes returned (n_s), and the number of empty boxes found (n_e).

We note that, although from the plots it seems that the different solution branches cross at many points, these are not true bifurcations of the linkage, as revealed by observing other 2D projections of the same output. Actually, four disjoint cyclic paths appear, corresponding to the four possible ways to assemble this mobile mechanism. The labels in Figure 4-(c) help identifying such paths.

Elimination [12], [13] or continuation-based methods [14], [15] cannot be directly applied to this mobile mechanism. Such methods are only applicable when the solution space has a finite number of points, and are complete in such cases. When infinite points are present they can also be used to obtain plots similar to the one in Figure 4-(c) by sampling one angle and solving for the rest iteratively. This approach would yield accurate plots for dense-enough samplings, but it would miss important points in some cases. For example, on linkages having both rigid and mobile assembly modes it would fail to detect the former ones. Being isolated points in configuration space, those modes would never get sampled in general. In other words, elimination or continuation-based methods are not even resolution-complete in such cases. To the authors' knowledge, the only box approximations reported so far for this linkage are those in [7], [17], derived with a linear relaxation technique. A remarkable difference is that such relaxation technique employs absolute orientation angles for the links, while the present technique uses the relative angles between them, what allows a natural definition of constraints in the joints' ranges of movement.

VIII. Conclusions

This work fits within a larger project aimed at providing a general tool for the position analysis of arbitrary multi-loop linkages, using branch-and-bound algorithms of the kind presented. At the core of such algorithms there is always a procedure for bounding the solution set of the linkage within a rectangular box of the search space. The "degree of exactness" of this procedure is crucial to attain efficiency in the search process. Note that, ideally, one would like to generate a search tree without empty boxes, and this can only be assured if the bounding procedure always returns the exact bounds. To the authors' knowledge, this is the first time that an exact bounding procedure is given for planar single-loop linkages. This is clearly an advantage over other related algorithms [6], [7], [17], which are inexact even for such linkages. A limitation of the procedure with respect to [6], [7], [17], however, is that in multi-loop linkages only local consistency is enforced, meaning that the returned intervals are only consistent with each equation separately. This may slow down the convergence to stable boxes and, eventually, produce branches of the search tree containing empty boxes, decreasing the overall efficiency and the quality of the output. The improvement of the convergence and the extension of the technique to spherical and spatial linkages are part of the authors' current efforts.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Education and Science through the I+D project DPI2004-07358, by the CTP under project 2006ITT-10004, and by I3 program funds supporting the third author.

References

[1] E. Celaya, "Geometric reasoning for the determination of the position of objects linked by spatial relationships," Ph.D. dissertation, Technical University of Catalonia, Barcelona, 1992.

[2] A. Castellet, "Solving inverse kinematics problems using interval methods," Ph.D. dissertation, Technical University of Catalonia, 1998.

[3] O. Didrit, M. Petitot, and E. Walter, "Guaranteed solution of direct kinematic problems for general configurations of parallel manipulators," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 2, pp. 259–266, 1998.

[4] R. S. Rao, A. Asaithambi, and S. K. Agrawal, "Inverse kinematic solution of robot manipulators using interval analysis," *ASME Journal of Mechanical Design*, vol. 120, pp. 147–150, 1998.

[5] J.-P. Merlet, "Solving the forward kinematics of a gough-type parallel manipulator with interval analysis," *The International Journal of Robotics Research*, vol. 23, no. 3, pp. 221–235, 2004.

[6] J. M. Porta, L. Ros, and F. Thomas, "Multi-loop position analysis via iterative linear programming," in *Proc. of Robotics, Science, and Systems*, 2006.

[7] J. M. Porta, L. Ros, T. Creemers, and F. Thomas, "Box approximations of planar linkage configuration spaces," *ASME Journal of Mechanical Design*, accepted for publication.

[8] E. Hansen, *Global Optimization Using Interval Analysis*. New York: Dekker, 1992.

[9] E. Celaya and C. Torras, "On finding the set of inverse kinematic solutions for redundant manipulators," in *Computational Kinematics*, J. Angeles, G. Hommel, and P. Kovács, Eds. Kluwer Academic Publishers, 1993, pp. 85–94.

[10] D. M. Chickering, D. Geiger, and D. Heckerman, "On finding a cycle basis with a shortest maximal cycle," *Information Processing Letters*, no. 54, pp. 55–58, 1994.

[11] K. R. Apt, "The essence of constraint propagation," *Theoretical Computer Science*, vol. 221, no. 1-2, pp. 179–210, 1999.

[12] J. Nielsen and B. Roth, "Solving the input/output problem for planar mechanisms," *ASME Journal of Mechanical Design*, vol. 121, pp. 206–211, June 1999.

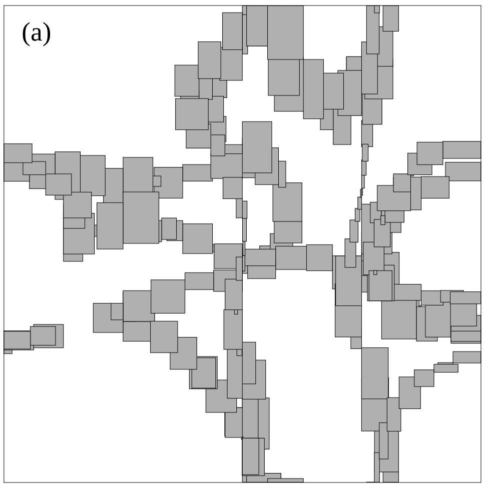
[13] C. W. Wampler, "Solving the kinematics of planar mechanisms by Dixon's determinant and a complex plane formulation," *ASME Journal of Mechanical Design*, vol. 123, pp. 382–387, September 2001.

[14] J. Verschelde, "Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation," *ACM Transactions on Mathematical Software*, vol. 25, no. 2, pp. 251–276, 1999.

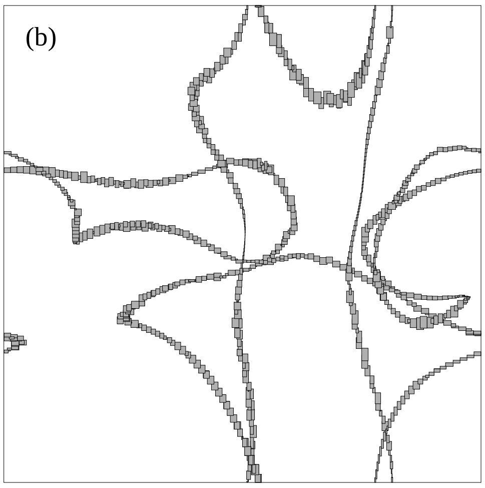
[15] A. J. Sommese, J. Verschelde, and C. W. Wampler, "Advances in polynomial continuation for solving problems in kinematics," *ASME Journal of Mechanical Design*, vol. 126, pp. 262–268, March 2004.

[16] Jan Verschelde's home page, http://www.math.uic.edu/˜jan.

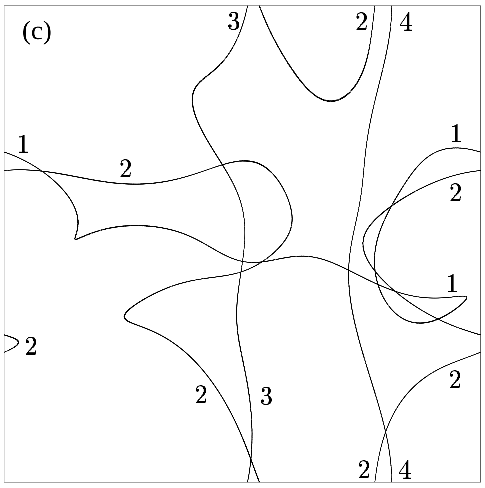
[17] T. Creemers, J. M. Porta, L. Ros, and F. Thomas, "Fast multiresolutive approximations of planar linkage configuration spaces," in *Proc. of the IEEE Int. Conf. on Rob. and Aut.*, 2006, pp. 1511–1517.



$\sigma = 0.5, t = 14.2s, n_s = 184, n_e = 132$



$\sigma = 0.1, t = 80.5s, n_s = 860, n_e = 451$



$\sigma = 0.005, t = 1840, n_s = 18864, n_e = 7931$

Fig. 4. Output boxes at increasing resolution. The horizontal and vertical axes correspond to θ_3 and θ_4, respectively, spanning the range $[0, 2\pi]$ in all cases. Labels in figure (c) help identifying the four connected components of this linkage configuration space.