INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Specific Quantifier Elimination for Inner Box Test in Distance Constraints with Uncertainties

Carlos Grandón — Bertrand Neveu

N° 5883

Avril 2006

Thèmes SYM et NUM

ISSN 0249-6399

A Specific Quantifier Elimination for Inner Box Test in Distance Constraints with Uncertainties

Carlos Grandón* , Bertrand Neveu†

Thèmes SYM et NUM — Systèmes symboliques et Systèmes numériques
Projet Coprin

Rapport de recherche n° 5883 — Avril 2006 — 23 pages

Abstract: This document presents a specific quantifier elimination algorithm for distance constraints with uncertainties represented by constraints with existentially quantified parameters. Generally, the solution set of this type of constraints has a non-null volume, and therefore an interval based solver that implements a branch and prune algorithm will bisect again and again the boxes included inside the solution set, leading to inefficient computation. This situation can be strongly improved using a test for detecting inner boxes. In this work, we show how interval arithmetics can be successfully used for building an inner box test in some types of constraints and how it can be combined with a quantifier elimination algorithm for handling constraints with existentially quantified parameters.

Key-words: Quantifier Elimination, Uncertainty, Distance equations, Interval Analysis, Constraint Programming.

* PhD Scholarship agreement CONICYT - INRIA Sophia Antipolis
† PhD adviser

Une élimination de quantificateur spécifique pour le test de boîte intérieur de contraintes de distance avec incertitudes

Résumé : Ce document présente un algorithme d'élimination de quantificateur spécifique pour des contraintes de distance avec incertitudes, représentées par des contraintes avec des paramètres quantifiés existentiellement. En général, l'ensemble solution de ce type de contraintes a un volume non nul. Par conséquence, un solveur basé sur les intervalles et qui applique un algorithme de type branch and prune divisera un grand nombre de fois les boîtes à l'intérieur de l'ensemble solution, conduisant ainsi à un calcul inefficace. Cette situation peut être fortement améliorée en utilisant un test pour détecter les boîtes intérieures. Dans ce travail, nous montrons comment l'arithmétique par intervalle peut être utilisée pour construire un tel test pour des contraintes particulières, et comment il peut être combiné avec un algorithme d'élimination de quantificateur pour traiter les contraintes avec des paramètres quantifiés existentiellement.

Mots-clés : Elimination de quantificateur, incertitude, équations de distance, analyse par intervalle, programmation par contraintes.

1 Introduction

This document presents a Specific Quantifier Elimination (SQE) algorithm for distance constraints with uncertainties in their parameters. We focus on this type of constraints because of their usefulness and their simplicity. Many problems in practice can be expressed by distance constraints (see [3]), from molecular biology ([1, 18]) to robotics ([19]).

Interval constraint propagation ([2, 5]) is a widely used technique for solving these class of problems. This technique allows one to reduce the domains of variables involved in a constraint without losing any solution. When it is coupled with a bisection algorithm, an accurate reliable outer approximation of the solutions can be achieved ([16]). Many interval constraint-based solvers (e.g. ILOG Solver [15], Numerica [13]) implement this approach. The main advantage of these methods is to generate a set of boxes which *conservatively* enclose each solution with a given precision. However, when applied to problems with non-isolated solutions they bisect again and again the boxes included inside the solution set, leading to inefficient computations and providing enclosures that are either prohibitively verbose or poorly informative.

There are different situations where a problem has non-isolated solutions, e.g. inequality constraints ([22, 18]) or constraints with existentially quantified parameters (e.g. a constraint on variable $x \in \mathbb{R}$ like $(\exists a \in \mathbf{a})(c(a, x))$ where $\mathbf{a}$ is an interval). In particular, a problem formed by distance constraints with uncertainties has non-isolated solutions. That means the solution set of the problem has a non-null volume, and therefore a branch and prune algorithm applied to this problem will bisect again and again the boxes included inside the solution set. This situation can be strongly improved using a test for detecting inner boxes, so that boxes which are proved to lie inside the solution set will not be bisected any more.

Interval arithmetics can be successfully used for building such *inner box test* with some types of constraints. In particular, inequality constraints of the form $c(x) : f(x) \lozenge 0$ (where $\lozenge = \{>, \geq, <, \leq\}$) when the constraint has no existentially quantified parameters. Let ρ_c be the solution set of the constraint $c(x)$, and let $\mathbf{x}$ be a box. Let $\mathbf{f}(\mathbf{x})$ be the *interval evaluation of* $f(x)$ *in* $\mathbf{x}$. The comparison $\mathbf{f}(\mathbf{x}) \lozenge 0$ can be used as a sufficient condition[1] of the inclusion $\mathbf{x} \subseteq \rho_c$, and therefore, as an inner box test for the constraint.

Constraints with existentially quantified parameters are more complicate, because the *interval evaluation* does not take into account these types of quantifiers. Even though the comparison $\mathbf{f}(\mathbf{x}) \lozenge 0$ can be used as inner box test, this test fails in most cases due to existential quantifiers. Distance constraints with uncertainties are clearly one of these classes of constraints.

In this document, we propose a method for checking if a box is included inside the solution set of a distance constraint with existentially quantified parameters. We focus on quantified distance constraints where the variables are the coordinates of a point $x \in \mathbb{R}^n$. As existentially quantified parameters, we have the coordinates of another point $a \in \mathbb{R}^n$ and a distance $r \in \mathbb{R}$. Then, the distance constraint fixes the distance between a and x to be equal

[1] In some particular cases, when the variable x appears once in the function, the comparison can be considered as a necessary and sufficient condition, but in general, this comparison is only a sufficient condition of $\mathbf{x} \subseteq \rho_c$.

to r. In our method, a special quantifier elimination algorithm is used in order to transform a quantified distance constraint into an equivalent non quantified conjunction/disjunction of constraints which can be evaluated using classic interval arithmetics.

The rest of this document is organized as follow: Section 2 presents the problem statement in a more formal way. Section 3 presents the basics related to interval analysis and quantifier elimination. Section 4 presents a Special Quantifier Elimination algorithm in a two and three dimensional space, and some optimization for particular cases. Section 6 comments preliminary results obtained with our implementation. Last, Section 7 summarizes the work's contribution.

Notations Following ([17]), scalars and vectors are denoted by lower case letters. Intervals are denoted by boldface letters. The real values $\underline{x}$ and $\overline{x}$ denote the lower bound and upper bound of an interval $\mathbf{x}$, respectively. Let $\epsilon = \{e_1, ..., e_n\}$ be an ordered set of indices, the vector $(\mathbf{x}_{e_1}, ..., \mathbf{x}_{e_n})$ is denoted by $\mathbf{x}_\epsilon$, so that $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is denoted by $\mathbf{x}_{[1..n]}$. If no confusion is possible, the usual notation $\mathbf{x}$ will be used in place of $\mathbf{x}_{[1..n]}$.

2 Problem Statement

The Euclidean distance between two points $a \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ is defined by

$$d(a, x) = \sqrt{\sum_{k=1}^{n} (x_k - a_k)^2} \tag{1}$$

An Euclidean distance constraint that relates two points a and x with a distance value r between them, can be expressed as following: $c_{a,r,x} : \sum_{k=1}^{n} (x_k - a_k)^2 = r^2$. All values given by a problem are called parameters, and they correspond to known values. Sometimes, these parameters values are not exactly known, but they are bounded by quantities and can be expressed in the form of intervals.

So, given a n-dimensional interval vector $\mathbf{a}$ and an interval $\mathbf{r}$, we are interested in the following quantified distance constraint:

$$c_{\mathbf{a},\mathbf{r}}(x) : (\exists a \in \mathbf{a})(\exists r \in \mathbf{r})(\sum_{k=1}^{n} (x_k - a_k)^2 = r^2) \tag{2}$$

It means, all the points $x \in \mathbb{R}^n$ that are at a distance r from the point a. The set of x which satisfies (2) is denoted by ρ_c. Figure 1 presents some examples of quantified distances constraints and their solutions.

In this work, we are interested in a sufficient condition for the inclusion $\mathbf{x} \subseteq \rho_c$, that means, given a n-dimensional interval vector $\mathbf{x}$, we have $(\forall x \in \mathbf{x})(c_{\mathbf{a},\mathbf{r}}(x))$ is true. Notice that a sufficient condition designed for one quantified distance constraint can also be used for a conjunction of quantified distance constraints $C = \bigwedge_{k \in [1..m]} c^{(k)}_{\mathbf{a}_{(k)},\mathbf{r}_{(k)}}(x)$. If existentially

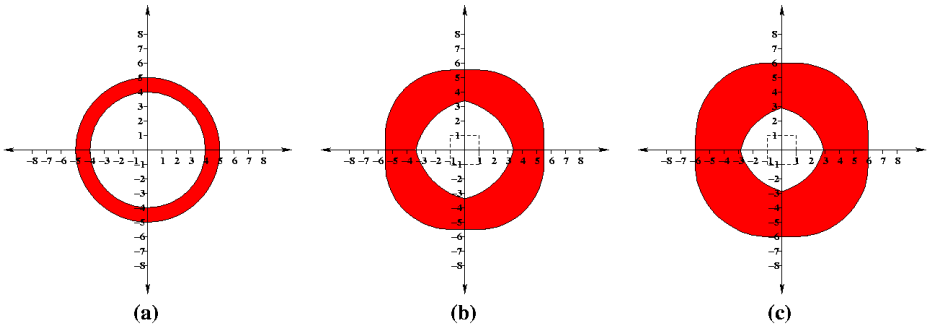

Figure 1: *Some examples of quantified distance constraint and their solutions. In* (a) *only the parameter* r *has an interval value:* $c_{a,\mathbf{r}}(x)$ *with* $a = (0,0)$ *and* $\mathbf{r} = [4,5]$. *In* (b) *only the parameter* a *has an interval value:* $c_{\mathbf{a},r}(x)$ *with* $\mathbf{a} = ([-1,1],[-1,1])$ *and* $r = 4.5$. *In* (c) *both parameters have interval values:* $c_{\mathbf{a},\mathbf{r}}(x)$ *with* $\mathbf{a} = ([-1,1],[-1,1])$ *and* $\mathbf{r} = [4,5]$.

quantified parameters are not shared between different constraints, we have the following implication:

$$\bigwedge_{k\in[1..m]} \mathbf{x} \subseteq \rho_{c^{(k)}} \quad \Longrightarrow \quad \mathbf{x} \subseteq \rho_C \tag{3}$$

3 Background and Definitions

In this section we present a brief introduction to Interval Analysis and Quantifier Elimination. Section 3.1 introduces some concepts as *intervals* and *interval arithmetics*, and relates these concepts with our work. Section 3.2 introduces the *quantifier elimination problem* and how it can be applied to a quantified distance constraint.

3.1 Interval Analysis

Interval Analysis ([20, 12]) was born in the 60's aiming rigorous computations with uncertain quantities and automatic control of the errors in a computed result. Informally, an *interval* is a set of real values represented by two bounds. These bounds are elements of a set $\mathbb{F}$, that is a subset of the real numbers. In a computational implementation, the set $\mathbb{F}$ is generally the set of floating points numbers that the computer can represent. For example, using arithmetics with three significant digits, we can represent a value $\sqrt{2}$ by the interval $[1.414, 1.415]$ and guarantee that the mathematically correct result is inside of the computed interval. In a formal way, an *interval* can be defined as following:

Definition 1 (Interval) *An interval* $\mathbf{x} = [\underline{x}, \overline{x}]$, *with* $\underline{x}$ *and* $\overline{x} \in \mathbb{F}$, *is the set of real numbers* $\{r \in \mathbb{R} \mid \underline{x} \leq r \leq \overline{x}\}$.

The set of intervals with bounds in $\mathbb{F}$, denoted by $\mathbb{IR}$, is partially ordered by set inclusion. A Cartesian product of n intervals $\mathbf{b} = \mathbf{x}_1 \times \cdots \times \mathbf{x}_n$ is called a *box*. Two boxes $\mathbf{b}_1$ and $\mathbf{b}_2$ are said disjoint if $\mathbf{b}_1 \cap \mathbf{b}_2 = \emptyset$.

Interval arithmetics provides a set of operators that allow one to compute interval results. Whenever an operation on reals is specified, the corresponding operation on their intervals is executed. Let $\mathbf{a}$ and $\mathbf{b}$ be two intervals, and let $\Diamond$ denotes one of the four arithmetic operators $\{+, -, *, /\}$. The operation $\mathbf{a}\Diamond\mathbf{b}$ is defined by:

$$\mathbf{a}\Diamond\mathbf{b} = \{a\Diamond b \mid a \in \mathbf{a} \ \wedge \ b \in \mathbf{b}\},$$

with $\mathbf{a}/\mathbf{b}$ undefined if $0 \in \mathbf{b}$. This definition ensures that the resulting interval $\mathbf{a}\Diamond\mathbf{b}$ contains all possible outcomes from applying $\Diamond$ with operands from $\mathbf{a}$ and $\mathbf{b}$, but the definition does not show how to compute it. The resulting interval is easily obtained using the bounds of the involved interval as shown in (4).

$$\begin{aligned} \mathbf{a} \oplus \mathbf{b} &= [\underline{a} + \underline{b}, \underline{a} + \underline{b}] \\ \mathbf{a} \ominus \mathbf{b} &= \left[\underline{a} - \overline{b}, \overline{a} - \underline{b}\right] \\ \mathbf{a} \otimes \mathbf{b} &= \left[min(\underline{ab}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}), max(\underline{ab}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b})\right] \\ \mathbf{a} \oslash \mathbf{b} &= \mathbf{a} * \left[1/\overline{b}, 1/\underline{b}\right] \end{aligned} \tag{4}$$

We used the operators $\{\oplus, \ominus, \otimes, \oslash\}$ here, to emphasize the difference between *interval operations* and *real operations*, but from here on the classical notation $\{+, -, *, /\}$ will be used.

Definition 2 (Hull approximation) *Let S be a subset of $\mathbb{R}$. The Hull approximation of S, denoted $\Box S$, is the smallest interval $\mathbf{x}$ such that $S \subseteq \mathbf{x}$.*

An *interval extension* of $f : \mathbb{R}^n \to \mathbb{R}$ is a mapping $\mathbf{f} : \mathbb{IR}^n \to \mathbb{IR}$ such that for all $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{IR} : x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n \Rightarrow f(x_1, \ldots, x_n) \in \mathbf{f}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$. An *n-ary* interval operation $\blacklozenge$ is called the *natural interval extension* of an *n-ary* real operation $\Diamond$ if for all $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{IR} : \blacklozenge(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \Box(\{\Diamond(x_1, \ldots, x_n) \mid x_1 \in \mathbf{x}_1, \ldots, x_n \in \mathbf{x}_n\}$. The *natural interval extension* of a function $f : \mathbb{R}^n \to \mathbb{R}$ is the interval function obtained from f by replacing each constant r by $\Box r$, each variable by an interval variable, and each operation by its natural interval extension. From here on the *interval evaluation of a function $f(x)$ with $x \in \mathbf{x}$* denotes the evaluation of the natural interval extension of $f(x)$ using the interval $\mathbf{x}$ for the variable x.

Interval arithmetics can be successfully used as a sufficient condition of the inclusion $\mathbf{x} \subseteq \rho$ in some types of quantified constraints. For example, an interval evaluation of a function $f(x, y) : 3x^2 - 2y + 9$ with $x \in [-1, 3]$ and $y \in [-2, 4]$ can be used for proving this inclusion in a constraint $f(x, y) > 0$, because $\mathbf{f}([-1, 3], [-2, 4]) = 3 \cdot [-1, 3]^2 - 2 \cdot [-2, 4] + 9 = [1, 40]$. As the evaluation of the function f is inside the interval $[1, 40]$, that means

$$(\forall x \in [-1, 3])(\forall y \in [-2, 4])(3x^2 - 2y + 9 > 0)$$

In some quantified distance constraints, this inclusion can also be successfully proved, when existentially quantified parameters are not taken into account in the evaluation of the function. For example, consider the following constraint

$$(\exists r \in \mathbf{r})(x_1^2 + x_2^2 = r^2)$$

with $\mathbf{x}_1 = [1,2]$, $\mathbf{x}_2 = [-1,1]$, and $\mathbf{r} = [1,3]$. Let $f(x_1,x_2) = x_1^2 + x_2^2$ be a function and consider the interval evaluation of $\mathbf{f}(\mathbf{x}_1,\mathbf{x}_2) : ([1,2]^2 + [-1,1]^2) = [1,5]$. From the point of view of interval arithmetics, we have the following interpretation:

$$(\forall x_1 \in \mathbf{x}_1)(\forall x_2 \in \mathbf{x}_2)(\exists z \in [1,5])(x_1^2 + x_2^2 = z)$$

In particular, if $[1,5] \subseteq \mathbf{r}^2$ the following proposition is also verified:

$$(\forall x_1 \in \mathbf{x}_1)(\forall x_2 \in \mathbf{x}_2)(\exists r \in \mathbf{r}^2)(x_1^2 + x_2^2 = r^2)$$

So, the inclusion $\mathbf{x}_1^2 + \mathbf{x}_2^2 \subseteq \mathbf{r}^2$ is a sufficient (and necessary, in this case) condition of $\mathbf{x} \subseteq \rho$.

In a more general case (quantified distance constraint like (2)), the interval evaluation is less effective, because existential quantified parameters are inside the evaluated function. For example, consider the constraint

$$(\exists a_1 \in \mathbf{a}_1)(\exists a_2 \in \mathbf{a}_2)(\exists r \in \mathbf{r})((x_1 - a_1)^2 + (x_2 - a_2)^2 = r^2)$$

with $\mathbf{x}_1 = [3,6]$, $\mathbf{x}_2 = [-1,1]$, $\mathbf{a}_1 = [-1,1]$, $\mathbf{a}_2 = [-1,1]$, and $\mathbf{r} = [4,5]$. If we consider the function $f(x_1,x_2,a_1,a_2) = (x_1-a_1)^2+(x_2-a_2)^2$, the interval evaluation of $\mathbf{f}(\mathbf{x}_1,\mathbf{x}_2,\mathbf{a}_1,\mathbf{a}_2)$ is $([3,6]-[-1,1])^2+([-1,1]-[-1,1])^2 = [4,53]$. The inclusion $[4,53] \subseteq \mathbf{r}^2$ is not true, so the test fails. A classic interval evaluation cannot prove that the interval vector $\mathbf{x} = ([3,6],[-1,1])$ is inside of the solution set of the constraint, but it is. This fault is not due to interval arithmetics but to a conceptual problem in its application. For two intervals $\mathbf{x}$ and $\mathbf{a}$, Interval Arithmetic computes an interval result $\mathbf{z}$ that encloses the combinations between any value from $\mathbf{x}$ and any value from $\mathbf{a}$. That means $(\forall x \in \mathbf{x})(\forall a \in \mathbf{a})(x \Diamond a \in \mathbf{z})$. So, the interval a is not considered as an existentially quantified parameter but as a universally quantified variable. In section 3.2 we show how these existentially quantified parameters can be eliminated from the evaluation of a function.

3.2 Quantifier Elimination

Informally, the basic motivation of the quantifier elimination is to *eliminate* unwanted variables from an algebraic description of some situation. These variables may represent parameters of a model, or real quantities that cannot be measured in an exact way (values with uncertainties, for example). Many mathematical problems can be phrased as quantifier elimination problems (see [14, 21]).

The first real quantifier elimination procedure which has been implemented was introduced by Collins in 1975 (see [6]). This method based on *cylindrical algebraic decomposition*

(CAD) is worst-case doubly exponential in the number of variables. Some methods for solving the quantifier elimination problem have been proposed and implemented since the introduction of the CAD based algorithm (see QEPCAD[7], REDLOG[9], and QERRC[8]). A good survey of these methods can be found in ([10]).

3.2.1 Quantifier Elimination Problem

From a more formal point of view, the real *quantifier elimination problem* can be phrased as follows: Given a formula F with quantified variables (universally $\forall$ and/or existentially $\exists$), find a formula $\overline{F}$ in which no variables are quantified, and that both F and $\overline{F}$ are equivalent in the domain of the real numbers. F is called the input formula, and $\overline{F}$ the solution formula.

For example, one solution formula for $F : (\exists x)[ax^2 + bx + c = 0]$ can be:

$$\overline{F} : b^2 - 4ac \geq 0 \ \wedge \ [b \neq 0 \ \vee \ a \neq 0 \ \vee \ c = 0]$$

In the case of distance constraints, a formula $F : (\exists r \in [1, 2])[x^2 + y^2 = r^2]$ is equivalent to the following quantifier-free formula:

$$\overline{F} : (x^2 + y^2 \geq 1) \wedge (x^2 + y^2 \leq 4)$$

Using the QEPCAD implementation of the quantifier elimination algorithm (available in [4]), it is possible to transform a 2D quantified distance constraint into a set of non-quantified constraints in only some seconds, but in the general case, the 3D quantified distance constraint can not be transformed[2]. In the next sections we show that it is possible to transform both types of constraints into a set of non-quantified constraints in less than one second, using a Specific Quantifier Elimination Algorithm based on graphic consideration. Moreover, this set of non-quantified constraints can be evaluated with interval arithmetics in order to build an inner box test.

4 Specific Quantifier Elimination

In this section we propose a specific quantifier elimination algorithm based on graphic considerations for quantified distance constraints. Section 4.1 presents an algorithm to transform a 2D quantified distance constraint into a quantifier-free formula, while section 4.2 presents the same algorithm in a three dimensional case. However, higher dimensions are out of the scope of the specific quantifier elimination proposed here.

4.1 The Two Dimensional Case

Let us consider a quantified distance constraint $c_{\mathbf{a},\mathbf{r}}(x) : f(a, x) = r^2$ as following:

$$c_{\mathbf{a},\mathbf{r}}(x) : (\exists a_1 \in \mathbf{a}_1)(\exists a_2 \in \mathbf{a}_2)(\exists r \in \mathbf{r})((x_1 - a_1)^2 + (x_2 - a_2)^2 = r^2) \tag{5}$$

[2]We use a PentiumIV 3GHz based machine with 512MB RAM and 2GB of swap memory. A general 2D quantified distance constraint is transformed into a quantifier-free formula in 33s, but the calculus for a general 3D quantified distance constraint could not be ended before memory overflow.

Figure 2a shows the graph of this constraint.

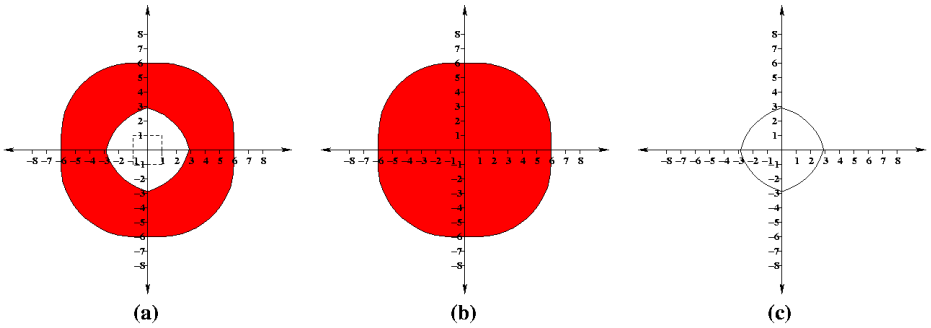

Figure 2: *A two dimensional quantified distance constraint.* (a) *Generic constraint* $c_{\mathbf{a},\mathbf{r}}(x)$. (b) *Internal auxiliary constraint* $c^i_{\mathbf{a},\mathbf{r}}(x)$. (c) *External auxiliary constraint* $c^e_{\mathbf{a},\mathbf{r}}(x)$.

The first step of our algorithm is the decomposition of $c_{\mathbf{a},\mathbf{r}}(x)$ into two auxiliary constraints. These constraints are the result of the elimination of the quantified parameter $\mathbf{r}$ from the constraint:

$$(\exists r \in \mathbf{r})(f(a,x) = r^2) \iff f(a,x) \leq \overline{\mathbf{r}}^2 \wedge f(a,x) \geq \underline{\mathbf{r}}^2 \tag{6}$$

Figure 2b and figure 2c show both auxiliary constraints. Let us call figure 2b the *internal auxiliary constraint* $c^i_{\mathbf{a},\mathbf{r}}(x)$ and figure 2c the *external auxiliary constraint* $c^e_{\mathbf{a},\mathbf{r}}(x)$. So, we clearly have $c_{\mathbf{a},\mathbf{r}}(x) \iff c^i_{\mathbf{a},\mathbf{r}}(x) \wedge \neg c^e_{\mathbf{a},\mathbf{r}}(x)$. These two auxiliary constraints can be characterized using the bounds of the involved interval, as presented in section 4.1.1 and section 4.1.2.

4.1.1 The constraint $c^i_{\mathbf{a},\mathbf{r}}(x)$

Using only the bounds of the intervals in $\mathbf{a}$ we obtain a disjunction of four non-quantified constraints that represent an approximation of $c^i_{\mathbf{a},\mathbf{r}}(x)$:

1. $(x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 \leq \overline{\mathbf{r}}^2$
2. $(x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 \leq \overline{\mathbf{r}}^2$
3. $(x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 \leq \overline{\mathbf{r}}^2$
4. $(x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 \leq \overline{\mathbf{r}}^2$

These constraints do not describe the constraint $c^i_{\mathbf{a},\mathbf{r}}(x)$, because some gaps are present as shown in figure 3a. In order to fill the remaining gaps, we use two constraints that represent boxes as shown in figure 3b. These boxes are characterized in a compact way by the following two interval inclusion constraints:

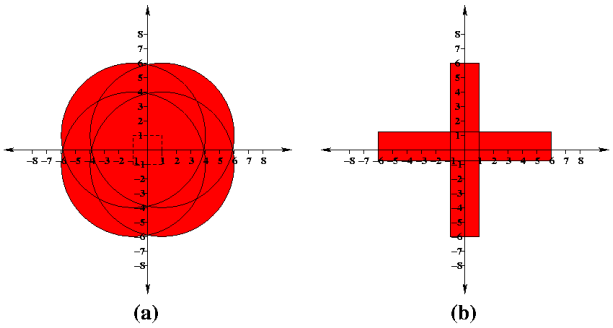

Figure 3: *Reconstruction of* $c^i_{\mathbf{a},\mathbf{r}}(x)$ *using four disks* (a) *and two boxes* (b).

5. $x \in ([\underline{a}_1 - \overline{\mathbf{r}}, \overline{a}_1 + \overline{\mathbf{r}}], \mathbf{a}_2)$
6. $x \in (\mathbf{a}_1, [\underline{a}_2 - \overline{\mathbf{r}}, \overline{a}_2 + \overline{\mathbf{r}}])$

Finally, $c^i_{\mathbf{a},\mathbf{r}}(x)$ is equivalent to the disjunction of the last six non-quantified constraints.

4.1.2 The constraint $c^e_{\mathbf{a},\mathbf{r}}(x)$

The graph of the constraint $c^e_{\mathbf{a},\mathbf{r}}(x)$ is easily obtained by intersecting four open disks. This constraint is represented as a conjunction of the following no quantified constraints:

1. $(x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 < \underline{\mathbf{r}}^2$
2. $(x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 < \underline{\mathbf{r}}^2$
3. $(x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 < \underline{\mathbf{r}}^2$
4. $(x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 < \underline{\mathbf{r}}^2$

Notice that the boundary of the figure 2c is not included in the graph of $c^e_{\mathbf{a},\mathbf{r}}(x)$. The constraint $\neg c^e_{\mathbf{a},\mathbf{r}}(x)$ is then represented as the disjunction of four (non-strict) inequalities.

4.2 The Three Dimensional Case

In the three dimensional case, we use the same decomposition into two auxiliary constraints used in the two dimensional case. The main difference is in the construction of the *internal auxiliary constraint*. Let us consider a quantified distance constraint $c_{\mathbf{a},\mathbf{r}}(x) : f(a,x) = r^2$ as following:

$$c_{\mathbf{a},\mathbf{r}}(x) : (\exists a_1 \in \mathbf{a}_1)(\exists a_2 \in \mathbf{a}_2)(\exists a_3 \in \mathbf{a}_3)(\exists r \in \mathbf{r})(\sum_{i=1}^{3} (x_i - a_i)^2 = r^2) \tag{7}$$

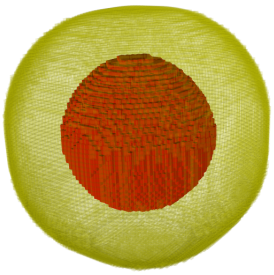

Figure 4: *Generic graph of the constraint $c_{\mathbf{a},\mathbf{r}}(x)$. All values between the the clearest surface and the darkest one are solutions of the constraint. Values inside the darkest surface are not solution.*

The generic graph of the constraint $c_{\mathbf{a},\mathbf{r}}(x)$ is shown in figure 4. The first step of the algorithm replaces the proposition $(\exists r \in \mathbf{r})(f(a,x) = r^2)$ (in the original constraint) by the conjunction $(f(a,x) \leq \overline{\mathbf{r}}^2) \wedge (f(a,x) \geq \underline{\mathbf{r}}^2)$, and builds the auxiliary constraint $c^i_{\mathbf{a},\mathbf{r}}(x)$ and $c^e_{\mathbf{a},\mathbf{r}}(x)$ as shown in figure 5.

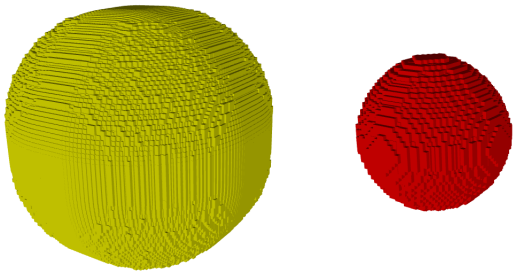

Figure 5: *Decomposition of the constraint $c_{\mathbf{a},\mathbf{r}}(x)$ into two auxiliary constraints: $c^i_{\mathbf{a},\mathbf{r}}(x)$ (left side picture) and $c^e_{\mathbf{a},\mathbf{r}}(x)$ (right side picture).*

Constraint $c^i_{\mathbf{a},\mathbf{r}}(x)$ corresponds to the inequality $(f(a,x) \leq \overline{\mathbf{r}}^2)$, while constraint $c^e_{\mathbf{a},\mathbf{r}}(x)$ corresponds to the inequality $(f(a,x) < \underline{\mathbf{r}}^2)$. The constraint $c_{\mathbf{a},\mathbf{r}}(x)$ is then replaced by the conjunction $c^i_{\mathbf{a},\mathbf{r}}(x) \wedge \neg c^e_{\mathbf{a},\mathbf{r}}(x)$. Sections 4.2.1 and 4.2.2 present a characterization of the constraints $c^i_{\mathbf{a},\mathbf{r}}(x)$ and $c^e_{\mathbf{a},\mathbf{r}}(x)$, respectively, as a conjunction/disjunction of non-quantified constraints.

4.2.1 The constraint $c^i_{\mathbf{a},\mathbf{r}}(x)$

Using the bounds of the intervals in $\mathbf{a}$ we build eight inequalities. The disjunction of these inequalities are a first approximation of $c^i_{\mathbf{a},\mathbf{r}}(x)$:

1. $(x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
2. $(x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
3. $(x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
4. $(x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
5. $(x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
6. $(x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
7. $(x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2$
8. $(x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2$

We notice that many gaps are present in the graph of this approximation. In order to fill these gaps we built a set of constraints based on geometric entities like boxes and cylinders. As in the two dimensional case, we use some boxes characterized by the following three interval inclusion constraints:

9. $x \in ([\underline{a}_1 - \overline{\mathbf{r}}, \overline{a}_1 + \overline{\mathbf{r}}], \mathbf{a}_2, \mathbf{a}_3)$
10. $x \in (\mathbf{a}_1, [\underline{a}_2 - \overline{\mathbf{r}}, \overline{a}_2 + \overline{\mathbf{r}}], \mathbf{a}_3)$
11. $x \in (\mathbf{a}_1, \mathbf{a}_2, [\underline{a}_3 - \overline{\mathbf{r}}, \overline{a}_3 + \overline{\mathbf{r}}])$

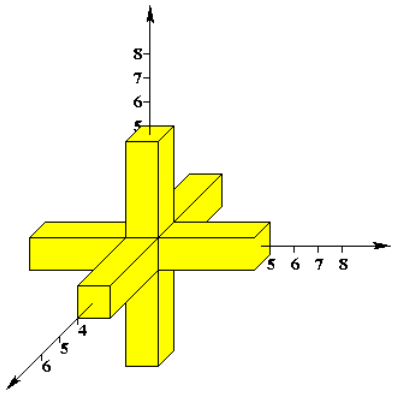

Figure 6: *Graph of some interval inclusion constraints for building* $c^i_{\mathbf{a},\mathbf{r}}(x)$.

The graph of these boxes is shown in figure 6. We notice that some gaps are still present, after the above decomposition. These remained gaps are filled with the following twelve cylindrical constraints (that correspond to the edges of the graph of $c^i_{\mathbf{a},\mathbf{r}}(x)$):

12. $(x_1 \in \mathbf{a}_1) \wedge ((x_2 - \underline{a}_2)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
13. $(x_1 \in \mathbf{a}_1) \wedge ((x_2 - \underline{a}_2)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
14. $(x_1 \in \mathbf{a}_1) \wedge ((x_2 - \overline{a}_2)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
15. $(x_1 \in \mathbf{a}_1) \wedge ((x_2 - \overline{a}_2)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
16. $(x_2 \in \mathbf{a}_2) \wedge ((x_1 - \underline{a}_1)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
17. $(x_2 \in \mathbf{a}_2) \wedge ((x_1 - \underline{a}_1)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
18. $(x_2 \in \mathbf{a}_2) \wedge ((x_1 - \overline{a}_1)^2 + (x_3 - \underline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
19. $(x_2 \in \mathbf{a}_2) \wedge ((x_1 - \overline{a}_1)^2 + (x_3 - \overline{a}_3)^2 \leq \overline{\mathbf{r}}^2)$
20. $(x_3 \in \mathbf{a}_3) \wedge ((x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 \leq \overline{\mathbf{r}}^2)$
21. $(x_3 \in \mathbf{a}_3) \wedge ((x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 \leq \overline{\mathbf{r}}^2)$
22. $(x_3 \in \mathbf{a}_3) \wedge ((x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 \leq \overline{\mathbf{r}}^2)$
23. $(x_3 \in \mathbf{a}_3) \wedge ((x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 \leq \overline{\mathbf{r}}^2)$

Finally, the graph of $c^i_{\mathbf{a},\mathbf{r}}(x)$ is equivalent to the disjunction of the twenty-three non-quantified constraints previously presented.

4.2.2 The constraint $c^e_{\mathbf{a},\mathbf{r}}(x)$

As in the two dimensional case, the graph of the constraint $c^e_{\mathbf{a},\mathbf{r}}(x)$ is easily obtained by intersecting some inequalities. In the three dimensional case, these inequalities represent eight open spheres, and may be characterized as following:

1. $(x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \underline{a}_3)^2 < \underline{\mathbf{r}}^2$
2. $(x_1 - \underline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \overline{a}_3)^2 < \underline{\mathbf{r}}^2$
3. $(x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \underline{a}_3)^2 < \underline{\mathbf{r}}^2$
4. $(x_1 - \underline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \overline{a}_3)^2 < \underline{\mathbf{r}}^2$
5. $(x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \underline{a}_3)^2 < \underline{\mathbf{r}}^2$
6. $(x_1 - \overline{a}_1)^2 + (x_2 - \underline{a}_2)^2 + (x_3 - \overline{a}_3)^2 < \underline{\mathbf{r}}^2$
7. $(x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \underline{a}_3)^2 < \underline{\mathbf{r}}^2$
8. $(x_1 - \overline{a}_1)^2 + (x_2 - \overline{a}_2)^2 + (x_3 - \overline{a}_3)^2 < \underline{\mathbf{r}}^2$

Notice again that the boundary is not included in the graph of $c^e_{\mathbf{a},\mathbf{r}}(x)$. The constraint $\neg c^e_{\mathbf{a},\mathbf{r}}(x)$ is then represented as the disjunction of eight (non-strict) inequalities.

5 Implementation

In our SQE implementation we represent a constraint in the form of a tree. As internal nodes of the tree we have logic operators *and* and *or*. As terminal nodes, we have non-quantified constraints or interval inclusion constraints. Figure 7 shows a generic tree of the quantified distance constraint $c_{\mathbf{a},\mathbf{r}}(x) : (x_1 - \mathbf{a}_1)^2 + (x_2 - \mathbf{a}_2)^2 = \mathbf{r}^2$.

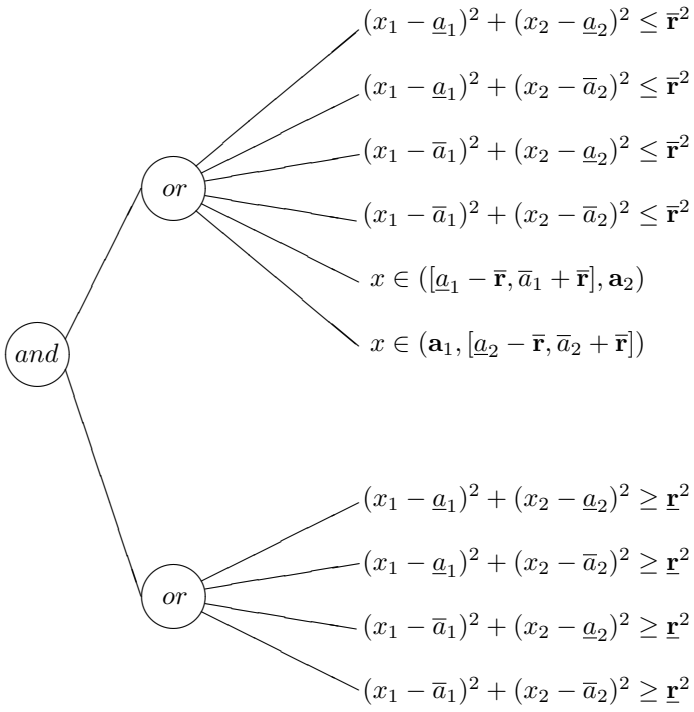

Figure 7: *Generic decomposition tree of the constraint* $c_{\mathbf{a},\mathbf{r}}(x)$.

Given an interval vector $\mathbf{x}$ and a decomposition tree, the algorithm computes an interval evaluation of the left side of each constraint and compares this evaluation with the right side. If at least one constraint is verified in each branch of the node *and*, all points in $\mathbf{x}$ are solution of the constraint $c_{\mathbf{a},\mathbf{r}}(x)$. That means, the box $\mathbf{x}$ is an inner box of $c_{\mathbf{a},\mathbf{r}}(x)$.

5.1 Some Optimizations

In our implementation of the Special Quantifier Elimination algorithm we use some optimization of the decomposition, based on the number of intervals in $\mathbf{a}$ and their values. For example, if only $\mathbf{a}_1$ and $\mathbf{r}$ are intervals and a_2 is a real value, the constraint $c^i_{\mathbf{a},\mathbf{r}}(x)$ is characterized by the disjunction of the following three constraints:

1. $(x_1 - \underline{a}_1)^2 + (x_2 - a_2)^2 \leq \overline{\mathbf{r}}^2$
2. $(x_1 - \overline{a}_1)^2 + (x_2 - a_2)^2 \leq \overline{\mathbf{r}}^2$
3. $x \in (\mathbf{a}_1, [a_2 - \overline{\mathbf{r}}, a_2 + \overline{\mathbf{r}}])$

Moreover, if $(2\underline{\mathbf{r}} \leq \overline{a}_1 - \underline{a}_1)$, then the constraint $c^e_{\mathbf{a},\mathbf{r}}(x)$ is not considered (because the graph of $c_{\mathbf{a},\mathbf{r}}(x)$ has no gap in the middle), and the decomposition is built as a disjunction of only the last three inequalities.

6 Preliminary Results

In this section we present different cases of quantified distance constraints in 2D and 3D, and the results of applying a classic interval solver (without inner box test), an interval solver with a basic test, and an interval solver combining a inner box test based on our SQE algorithm. Table 1 presents a list of quantified distances constraints in 2D.

Constraint	**Variable**	**Parameters**
$c^1_{a,\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10])$	$a = (0,0)$ $\mathbf{r} = [4,5]$
$c^2_{\mathbf{a},r}(x)$	$\mathbf{x} = ([-10,10],[-10,10])$	$\mathbf{a} = ([-1,1],[-1,1])$ $r = 4.5$
$c^3_{\mathbf{a},\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10])$	$\mathbf{a} = ([-1,1],[-1,1])$ $\mathbf{r} = [4,5]$
$c^4_{\mathbf{a},\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10])$	$\mathbf{a} = ([-2,2],[0,0])$ $\mathbf{r} = [2,5]$
$c^5_{\mathbf{a},\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10])$	$\mathbf{a} = ([-0.1,0.1],[-0.1,0.1])$ $\mathbf{r} = [3,5]$

Table 1: *Some examples of quantified distances constraints in 2D.*

The first three constraints correspond to the examples presented in figure 1 (section 2). Constraint $c^4_{\mathbf{a},\mathbf{r}}(x)$ is a special example where the graph of the solution set has no gap in the middle. Finally, constraint $c^5_{\mathbf{a},\mathbf{r}}(x)$ is an example with a big uncertainty in the distance parameter and a little uncertainty in the center parameter. These examples allow us to show the performance of the algorithms in problems with different degree of uncertainties.

Table 2 presents a list of quantified distances constraints in 3D.

Constraint	**Variable**	**Parameters**
$c^6_{\mathbf{a},\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10],[-10,10])$	$\mathbf{a} = ([-0.1,0.1],[-0.1,0.1],[-0.1,0.1])$ $\mathbf{r} = [4,5]$
$c^7_{\mathbf{a},\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10],[-10,10])$	$\mathbf{a} = ([-2,2],[-2,2],[-2,2])$ $\mathbf{r} = [4.4,4.5]$
$c^8_{\mathbf{a},\mathbf{r}}(x)$	$\mathbf{x} = ([-10,10],[-10,10],[-10,10])$	$\mathbf{a} = ([-2,2],[-2,2],[-2,2])$ $\mathbf{r} = [3,6]$

Table 2: *Some examples of quantified distances constraints in 3D.*

A branch and prune algorithm combining filtering and bisection techniques was used for finding the solution set of each constraint. The inner box test was applied each time the filtering phase failed in reducing the domain of the variables. Table 3 shows the computing results[3] of the experimentations without using inner box test (Without Test), using a basic test based on classic interval evaluation (Basic Test), and using the specific quantifier elimination based test (SQE Test).

Row *Time* presents the running time in seconds. Rows *Boxes* and *Inner* present the total number of boxes and the number of inner boxes found, respectively. First of all, it is clear that the use of inner box tests drastically reduces the computing time in all situations. We notice that the SQE Test is the best test in most of cases but in constraint $c^1_{a,\mathbf{r}}(x)$. How we explained in section 3.1, the simple interval evaluation test is a necessary and sufficient condition of $\mathbf{x} \subseteq \rho_c$, so the test based on our SQE cannot improve the computing time. Anyway, the computed results of both tests are the same. Figure 8 shows these results in a graphic way. Left side picture shows the solution set computed without using any inner box test. Right side picture shows the solution set computed using an inner box test.

We notice that only in some particular cases, like constraints $c^1_{a,\mathbf{r}}(x)$, $c^4_{\mathbf{a},\mathbf{r}}(x)$, $c^5_{\mathbf{a},\mathbf{r}}(x)$, and $c^6_{\mathbf{a},\mathbf{r}}(x)$, the inner box test based on classic interval evaluation can detecting inner boxes. In most of these cases, the degree of the uncertainties in the center parameter is low (with respect to the degree of the distance parameter), and the test can detect a few number of boxes. Figure 9 shows a graphic comparison between the solution set computed with the basic inner box test (left side picture) and the results obtained with the test based on the SQE (right side picture). In the first case, the test cannot detect inner boxes in a big surface (due to the existential parameters in the left side of the constraint), and therefore, it splits the boxes inside this zone until arriving to the given precision. On the other hand, the test based on the SQE detects successfully the inner boxes inside this surface, and the number of boxes needed for describing the solution set is substantially lower.

[3]Obtained on a Pentium IV 3GHz with 512MB of RAM and 1.5GB of swap memory, running IcosAlias v0.2b (a tool in development in Coprin project) on a Linux operating system.

Precision $\epsilon = 2e^{-2}$	**Without Test**	**Basic Test**	**SQE Test**
$c^1_{a,\mathbf{r}}(x)$			
Time (s)	4.516	0.358	0.361
Boxes	107,629	7,820	7,820
Inner	–	3,839	3,839
$c^2_{\mathbf{a},r}(x)$			
Time (s)	21.946	22.783	0.505
Boxes	510,307	510,306	9,540
Inner	–	–	4,551
$c^3_{\mathbf{a},\mathbf{r}}(x)$			
Time (s)	31.640	33.151	0.515
Boxes	754,768	754,768	9,559
Inner	–	–	4,556
$c^4_{\mathbf{a},\mathbf{r}}(x)$			
Time (s)	19.767	18.042	0.290
Boxes	470,776	410,412	5,896
Inner	–	1,328	2,776
$c^5_{\mathbf{a},\mathbf{r}}(x)$			
Time (s)	4.407	2.293	0.411
Boxes	170,214	50,338	7,147
Inner	–	2,604	3,590
Precision $\epsilon = 2e^{-1}$	**Without Test**	**Basic Test**	**SQE Test**
$c^6_{\mathbf{a},\mathbf{r}}(x)$			
Time (s)	6.754	5.469	4.420
Boxes	124,832	97,384	48,458
Inner	–	12,616	15,218
$c^7_{\mathbf{a},\mathbf{r}}(x)$			
Time (s)	58.767	62.634	9.805
Boxes	1,162,876	1,162,876	99,827
Inner	–	–	25,467
$c^8_{\mathbf{a},\mathbf{r}}(x)$			
Time (s)	79.263	84.294	13.288
Boxes	1,592,076	1,592,076	147,648
Inner	–	–	37,996

Table 3: *Computing results of the experiments. All constraints in 2D are computed with a precision $\epsilon = 2e^{-2}$. Precision for constraints in 3D was $\epsilon = 2e^{-1}$.*

Similar results can be observed in the three dimensional cases, but the performance of the inner box test is lower. It is important to notice that in a two dimensional space, any

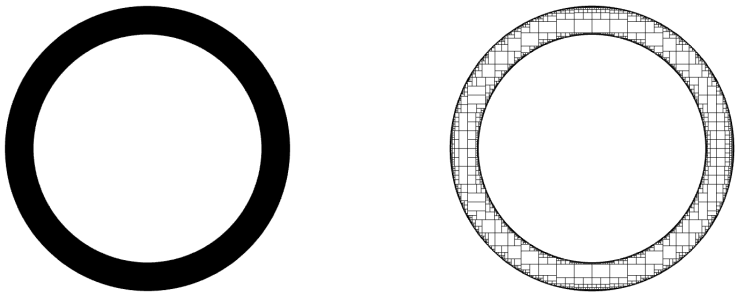

Figure 8: *Graphic representation of the solution set of the constraint* $c^1_{a,\mathbf{r}}(x)$ *computed without using inner box test (left side picture), and using the test for detecting inner boxes (right side picture).*

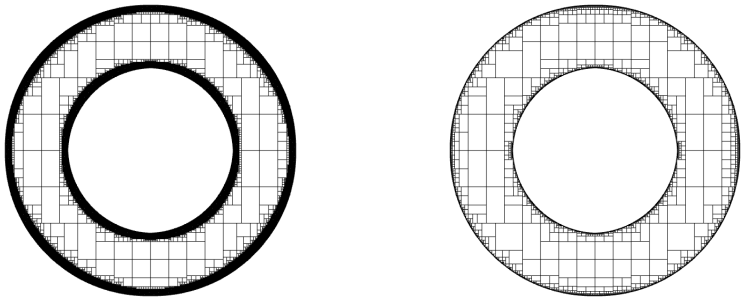

Figure 9: *The solution set of the constraint* $c^5_{\mathbf{a},\mathbf{r}}(x)$ *computed with the basic inner box test (left side picture), and using the test based on SQE (right side picture).*

interval based solver must describe the border of the solution set (a line) with boxes, and these boxes will have the given precision. In a three dimensional space, an interval based solver must describe a surface (the border of the solution set) with the same precision. This is a weakness of the interval based approach, even though an optimal inner box test be applied.

As we have shown in section 2, if existentially quantified parameters are not shared between different constraints, we can use the inner box test for a conjunction of quantified distance constraints. For example, consider the following problem formed by three quantified distance constraints:

$$\begin{aligned} x^2 + y^2 &= [2, 2.25]^2 \\ (x - [3, 3.5])^2 + y^2 &= [2.95, 3.05]^2 \\ (x - [-2.5, -2.25])^2 + (y - 2)^2 &= [3.25, 3.5]^2 \end{aligned}$$

Using a branch and prune algorithm and the inner box test based on SQE we solved this problem in 0.372 seconds with a precision $\epsilon = 1e^{-3}$. Table 4 presents a resume of the results.

	SQE Test
Time (s)	0.372
Total boxes	5,481
Inner boxes	2,550
Total volume	0.21236
Volume inner boxes	0.21103

Table 4: *Computing results of an academic problem formed by quantified distance constraints.*

Row *Total volume* represents the volume of all boxes found. Row *Volume inner boxes* represent the volume of the inner boxes detected. If no inner box test is implemented, the last volume must be described by boxes of the size $10^{-3} \times 10^{-3}$, that is $211,030$ boxes approximatively. Figure 10 shows the above results in a graphic way.

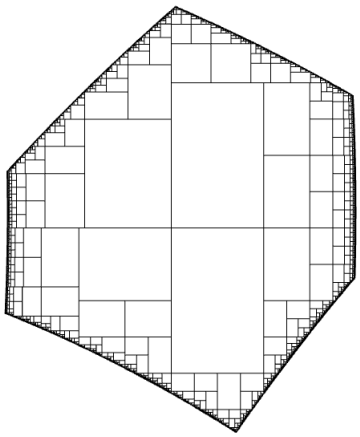

Figure 10: *Graphic results of the algorithm in an academic example.*

7 Conclusions

Constraints with existentially quantified parameters, i.e. constraints like $(\exists a \in \mathbf{a})(f(a,x)=0)$, generally have a non-null volume solution set. Therefore, any bisection algorithm dedicated to the approximation of their solution set should incorporate a test for checking if a box is included inside the solution set, unless it will spend most of the time bisecting again and again boxes included in the solution set.

We have shown how interval arithmetics can be successfully used for detecting inner boxes in some types of constraints and why this arithmetics is less effective in a more general case of quantified constraint. In addition, we presented the quantified elimination problem, and how the solution of this problem can be used with interval arithmetics in order to improve the detection of inner boxes. A special quantified elimination algorithm based on graphic consideration was also proposed. This algorithm allows one to transform quantified distance constraints into a set of non-quantified constraints in less than one second.

Some examples using different types of quantified distance constraints have been presented in order to explain the advantages of this approach. An example with a conjunction of constraints was also presented.

We can name some limitations of this approach. First, the test based on our SQE is a sufficient condition for $\mathbf{x} \subseteq \rho_c$. It is not a necessary condition, because a box can satisfy $\mathbf{x} \subseteq \rho_c$ while it does not satisfy any of the constraints generated by the decomposition. Such a box would intersect several graphs of the generated constraints but would be included in none of them. This situation is called the *decomposition flaw* and it is generally found in the 3D decomposition.

Another limitation of our SQE is the space where the test can be applied. We present a decomposition in two and three dimensional spaces, but higher dimensions are out of the scope of our test.

We are currently working in a new approach for building inner box tests based on *generalized intervals* (intervals whose bounds are not constrained to be ordered [11]). This approach allows one to compute an interval evaluation of a function involving universally and existentially quantified parameters without transforming it in a free-function. This feature is very interesting, because it is possible to verify a constraint with only one interval evaluation instead a set of evaluations given by the decomposition process. A preliminary study shows that a test based on this approach is neither optimum, but a combination of generalized intervals and graphic considerations could overcome this limitation.

References

[1] R. Backofen. Constraint Techniques for Solving the Protein Structure Prediction Problem. *Lecture Notes in Computer Science*, 1520:72–88, 1998.

[2] F. Benhamou and W. Older. Applying Interval Arithmetic to Real, Integer and Boolean Constraints. *Journal of Logic Programming*, 32(1):1–24, 1997.

[3] L.M. Blumenthal. *Theory and Applications of Distance Geometry*. Chelsea, New York, 1970.

[4] C. Brown. Quantifier Elimination by Partial Cylindrical Algebraic Decomposition, `http://www.cs.usna.edu/~qepcad/B/QEPCAD.html`.

[5] H. Collavizza, F. Delobel, and M. Rueher. Comparing partial consistencies. *Reliable Computing*, 1:1–16, 1999.

[6] George Collins. Quantifier elimination for the elementary theory of real closed fields by cynlindrical algebraic decomposition. In *Automata Theory and Formal Languages, Berlin, Germany*, volume LNCS 33, pages 264–274. Springer-Verlag, 1975.

[7] George Collins and Hoon Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *J. Symb. Comput.*, 12(3):299–328, 1991.

[8] A. Dolzmann. *Reelle Quantorenelimination durch parametrisches Zählen von Nullstellen*. PhD thesis, FMI, Universität Passau, D-9403 Passau, Germany, November 1994.

[9] Andreas Dolzmann and Thomas Sturm. Redlog user manual. Technical Report MIP-9616, FMI, Universität Passau, D-94030 Passau, Germany, October 1996.

[10] Andreas Dolzmann, Thomas Sturm, and Volker Weispfenning. Real quantifier elimination in practice. Technical Report MIP9720, FMI, Universität Passau, D-94030 Passau, Germany, December 1997.

[11] A. Goldsztejn. *Définition et Applications des Extensions des Fonctions Réelles aux Intervalles Généralisés*. PhD thesis, Université de Nice-Sophia Antipolis, 2005.

[12] B. Hayes. A Lucid Interval. *American Scientist*, 91(6):484–488, 2003.

[13] Pascal Van Hentenryck. A gentle introduction to NUMERICA. *Artif. Intell.*, 103(1-2):209–235, 1998.

[14] Hoon Hong, Richard Liska, and Stanly Steinberg. Testing stability by quantifier elimination. *Journal of Symbolic Computation*, 24(2):161–187, 1997.

[15] ILOG. Ilog Solver, reference manual, August 2000.

[16] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis with Examples in Parameter and State Estimation, Robust Control and Robotics*. Springer-Verlag, 2001.

[17] R.B. Kearfott. Standardized notation in interval analysis, 2002.

[18] L. Krippahl and P. Barahona. Applying Constraint Programming to Protein Structure Determination. In *Proc. of 5th International Conference on Principles and Practice of Constraint Programming (CP'99)*, volume 1713 of *Lecture Notes in Computer Science*, pages 289–302, 1999.

[19] J.P. Merlet. *Parallel robots*. Kluwer, Dordrecht, 2000.

[20] R. Moore. *Interval Analysis*. Prentice Hall, 1966.

[21] Jacob Schwartz and Micha Sharir. On the 'piano movers' problem i. the case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. Technical Report 39, Department of Computer Science, Courant Institute of Mathematical Sciences, 1981.

[22] M.C. Silaghi, D. Sam-Haroud, and B. Faltings. Search techniques for non-linear constraint satisfaction problems with inequalities. In *Proc. of 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence*, volume 2056 of *Lecture Notes in Computer Science*, pages 183–193, 2001.

Contents

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399