Automated Robot’s Workspace Approximation

Mikhail Posypkin
Dorodnicyn Computing Centre, Federal Research Center Computer Science and Control of Russian Academy of Sciences
E-mail: mposypkin@gmail.com

Abstract. The workspace of a robot is defined as a set of positions that robot or its part can take. The workspace size maximization is often one of the design goals: the larger workspace the bigger area a robot can serve. A manual workspace determination can be an error-prone and time-consuming process. In this work, we propose a method for an automated constructing of the robot’s workspace by a paving composed from boxes. Two different ways of obtaining a paving that covers the solution set of kinematic equations are considered. The first method overestimates the solution set of kinematic equations by comparing interval bounds on the left parts of equations against zero. Though this method constructs a valid paving the approximation can be quite inaccurate. We propose a way for improving the quality of the approximation based on a local solution search method. We compare both ways of constructing the workspace approximation on a real robot.

1. Introduction
A robot’s workspace [1] is one of its key characteristics. It is used for optimal path planning. The size and the shape of the workspace determine the area that the robot can serve. For simple cases, a workspace can be easily constructed manually. However, for more complex cases, this process should be automated.

The topic has been investigated thoroughly for decades, thus an interested reader should be referred to [2, 3]. For instance, in [2] the authors suggest the following classification of the workspace assessment techniques, which can be applied to parallel robots: geometrical, algebraic, and discretization, though the majority of these techniques are intended to be used only for robots within the class they were developed for. The geometric techniques are quite efficient, though they can be applied to relatively simple robots. The techniques that based on discretization can be applied to a broader class of parallel robots. However, these techniques are time-consuming and intended to be used in cases when the forward or the inverse kinematic problems have a simple solution [3, 4].

In [5, 6] authors propose to reduce kinematic equations to a set of inequalities and then use the branch-and-bound technique to construct a set of boxes enclosing the solution set (so-called paving). The drawback of this approach is that it requires additional efforts for obtaining a system of inequalities from kinematic equations. For complex robots, this reduction is rather complicated or even impossible. In this paper, we show how to avoid this step and obtain a paving directly from kinematic equations.

1. This work was supported by the Russian Science Fund, project 16-19-00148
2. The outline of the approach

The problem under consideration consists in finding a projection of a set of points defined by a system of equations. Let \( I \subseteq \{1, \ldots, n\} \) be a subset of indices. Without loss of generality we assume that \( I = \{1, \ldots, k\} \), \( k < n \). Consider a system of equations:

\[
\begin{align*}
  f_1(x) &= 0, \\
  & \quad \vdots \\
  f_m(x) &= 0,
\end{align*}
\]

where \( f_i(x), i = 1, \ldots, m \) are continuous functions defined on \( \mathbb{R}^n \), \( m \leq n \). We assume that parameters satisfy interval constraints:

\[
x_i \in [a_i, b_i], i = 1, \ldots, n.
\]

A set defined by constraints (2) is a box denoted by \( a \) in a sequel. Equations (1) together with interval constraints (2) define a (possibly empty) set \( X \subseteq \mathbb{R}^m \). This set consists of all points \( x \) satisfying (1) and (2). We are interested in finding the projection \( X_I \) of the set \( X \) on a subspace \( \mathbb{R}^k \), spanning the first \( k \) parameters \( x_1, \ldots, x_k \).

**Example 1.** Let \( m = 1, n = 2 \), \( f_1(x) = x_1^2 + x_2^2 - 1 \), \( k = 1 \), \( x_i \in [-2, 2], i = 1, 2 \). Then \( X \) is a circle centered at \((0, 0)\) with the radius 1 and \( X_I \) is an interval \( \{ x_1 : -1 \leq x_1 \leq 1 \} \).

In this paper, we propose an algorithm to construct an outer approximation of the set \( X_I \). An approximation is searched as a set of boxes. Following [5] we call this approximation a paving. More formally a set of boxes \( P \) is called a paving for a set \( S \) if the following two properties hold:

(i) \( S \subseteq \bigcup_{x \in P} \); 
(ii) boxes in \( P \) can’t share common internal points (may have common boundary points).

Pavings have some nice properties. For instance, the volume of a set \( \bigcup_{x \in P} \) is a sum of volumes of boxes from \( P \). When approximating a working set of a robot pavings can be used for path planning [1].

The accuracy \( acc(P) \) of a paving is defined as a Hausdorff distance between the set \( S \) and the paving \( P \):

\[
acc(P) = d_H(S, P).
\]

The maximal width of a box from a paving \( P \) is called granularity and is denoted as \( \text{gran}(P) \).

For a box \( x \) define vectors \( m(x) \) and \( M(x) \) as follows:

\[
m_i(x) = \min_{x \in x} f_i(x), \quad M_i(x) = \max_{x \in x} f_i(x), i = 1, \ldots, m.
\]

The following fact directly follows from the continuity of functions \( f_i(x), i = 1, \ldots, m \).

**Proposition 1.** If \( \max_{i \in \{1, \ldots, n\}} m_i(x) > 0 \) or \( \min_{i \in \{1, \ldots, n\}} M_i(x) < 0 \) then \( x \cap X = \emptyset \).

In practice finding an exact range \([m_i(x), M_i(x)]\) for a function \( f_i(x) \) over a box \( x \) can be problematic. However it is almost always possible to obtain an outer approximation of this range \([\hat{m}_i(x), \hat{M}_i(x)]\) with the help of interval analysis [7] or some other bounding techniques. Values \( \hat{m}_i(x), \hat{M}_i(x) \) obey the following property:

\[
\hat{m}_i(x) \leq \min_{x \in x} f_i(x), \quad \hat{M}_i(x) \geq \max_{x \in x} f_i(x), i = 1, \ldots, m.
\]

In a same way as the Proposition 1 we can prove the following proposition.

**Proposition 2.** If \( \max_{i \in \{1, \ldots, n\}} \hat{m}_i(x) > 0 \) or \( \min_{i \in \{1, \ldots, n\}} \hat{M}_i(x) < 0 \) then \( x \cap X = \emptyset \).
The Proposition 2 leads to the following simple algorithm BASECOV to construct a paving (Algorithm 1). For the sake of clarity we first consider the case $I = \mathbb{1}, n$, i.e. $X = X_I$.

**Input:** $a, \delta$

**Output:** $A$

$L := \{a\}$

$A := \emptyset$

while $L \neq \emptyset$ do

remove a box $x$ from $L$

if $\max_{i \in \mathbb{1}, n} m_i(x) \leq 0$ and $\min_{i \in \mathbb{1}, n} \nabla m_i(x) \geq 0$ then

if $d(x) \leq \delta$ then

$A := A \cup x$

else

split $x$ into $x', x''$

$L := L \cup \{x', x''\}$

end

end

end

**Algorithm 1:** The basic coverage algorithm BASECOV

The algorithm maintains two lists: $L$ for storing processed boxes and $A$ for storing the resulting approximation. The list $A$ is empty at the beginning and the list $L$ initially contains only the box $a$. At each iteration, the main while loop takes a box $x$ from the list $L$. If this box satisfies the proposition 2 it is discarded. Otherwise its diameter $d(x)$ is checked: if it is less than the prescribed accuracy $\delta$ the box is stored in the list $A$ otherwise it is split into two equal boxes across the longest edge. The loop terminates when the list $L$ becomes empty.

The list of boxes $A$ constructed by the BASECOV algorithm satisfies both paving’s properties. Unfortunately, it is difficult to estimate the accuracy of this paving. One of the ways to provide a guaranteed accuracy is to ensure that every box in the approximation contains at least one point from $X$. We’ll call pavings satisfying this property *tight*. The following proposition holds for tight pavings.

**Proposition 3.** Let $P$ be a tight paving with a granularity $\delta$. Then $\text{acc}(P) \leq \delta$.

The algorithm 2 (ADVCOV) constructs a tight paving. The difference between BASECOV and ADVCOV algorithms is that ADVCOV runs a local search for each box with a width less or equal to $\delta$. A local algorithm $L S$ searches for a solution $z$ of a system (1) starting from the center $\text{mid}(x)$ of a box $x$. If $z$ lies inside $x$ then the box is added to the constructed paving $A$. Otherwise the box $x$ is split into two equal boxes $x', x''$ that replace it in the current list of boxes $L$. 
Input: a, δ
Output: A

\[ L := \{a\} \]
\[ A := \emptyset \]

while \( L \neq \emptyset \) do
  remove a box \( x \) from \( L \)
  if \( \max_{i \in T} \hat{m}_i(x) \leq 0 \) and \( \min_{i \in T} \hat{m}_i(x) \geq 0 \) then
    if \( d(x) \leq \delta \) then
      \[ z := LS(mid(x)) \]
      if \( z \in x \) then
        \[ A := A \cup x \]
      else
        split \( x \) into \( x', x'' \)
        \[ L := L \cup \{x', x''\} \]
      end
    else
      split \( x \) into \( x', x'' \)
      \[ L := L \cup \{x', x''\} \]
    end
  end
end

**Algorithm 2:** The advanced coverage algorithm ADVCOV

Any algorithm able to locate a local solution of the system (1) can be taken as a local search method \( LS \). We used a box-constrained coordinate descent method applied to a convolution \( \phi(x) = \sum_{i=1}^{m} f_i(x)^2 \).

Usually, we are interested in a projection of a workspace to a set of coordinates. Since the paving is composed by boxes the projection can be obtained easily as a union of projections of all boxes in it. However, this can be a very resource-demanding process because the number of boxes in a paving can be huge.

3. Experimental evaluation

We applied the proposed algorithm to the problem of finding a working space of a parallel robot called DexTAR [8]. This robot is formed by 4 links \( AB, BC, CD, DE \), connected with passive joints \( B, C, D \) and driven by two rotating actuators \( A, E \) (figure 1). The working tool of a robot is located in \( C \). The workspace is defined as a set of all possible positions of the working tool.

![DexTAR robot’s picture and the scheme](image)

**Figure 1.** The DexTAR robot’s picture and the scheme
We considered the following configuration: \(|AB| = |ED| = 8, |BC| = |CD| = 6, |AE| = 6.\)

The workspace and its approximations obtained by algorithms BASECOV and ADVCOV are depicted at the figure 2. The value of \(\delta\) was set to \(5 \cdot 10^{-2}\) for BASECOV and \(1.25 \cdot 10^{-1}\) for ADVCOV. The precisions were chosen to obtain a comparable number of boxes in pavings. Notice that the actual size of boxes in a paving constructed by ADVCOV can be less due to splitting of boxes where the local solution wasn’t found. With this settings, the numbers of boxes obtained by the BASECOV and ADVCOV methods were 5563 and 3417 respectively. It is visually evident that the paving obtained by ADVCOV is more accurate w.r.t. the paving obtained by BASECOV, though the number of boxes in the latter is bigger.

![Figure 2. The comparison of pavings](image)

4. Conclusions

In this work, we addressed a problem of workspace approximation. An algorithm to construct an approximation of the workspace defined by a set of kinematic equations is proposed. We show that employing the local search significantly improves the quality of the approximation. However, the computational complexity of the proposed method is still significant. In the future we plan to try more efficient local search methods, e.g. Newton method \([9]\) and apply parallel computing.

References

[1] Merlet J P 2006 Parallel robots vol 128 (Springer Science & Business Media)
[2] Jha R, Chablat D, Rouillier F and Moroz G 2016 Robotics and Mechatronics (Springer) pp 121–130
[3] Snyman J, Du Plessis L and Duffy J 2000 Journal of Mechanical Design 122 447–456
[4] Bohigas O, Manubens M and Ros I 2012 IEEE Transactions on Robotics 28 993–1006
[5] Jaulin L 2001 Applied interval analysis: with examples in parameter and state estimation, robust control and robotics vol 1 (Springer Science & Business Media)
[6] Evtushenko Y, Posypkin M, Rybak L and Turkin A 2018 Journal of Global Optimization 71 129–145
[7] Hansen E and Walster G W 2003 Global optimization using interval analysis: revised and expanded vol 264 (CRC Press)
[8] Sergiu-Dan Stan V M and Balan R 2008 Frontiers in Evolutionary Robotics 465–490
[9] Rheinboldt W C 1998 Methods for solving systems of nonlinear equations vol 70 (Siam)