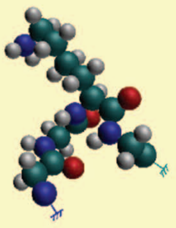
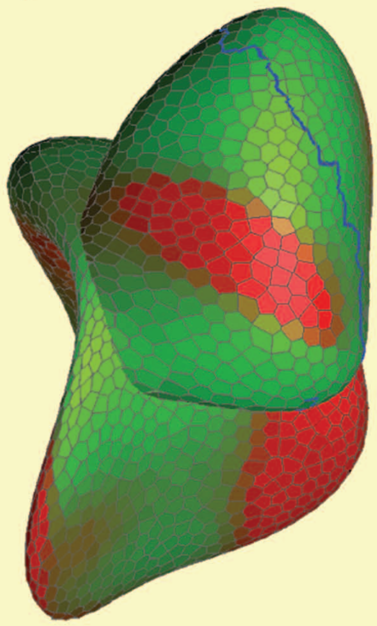
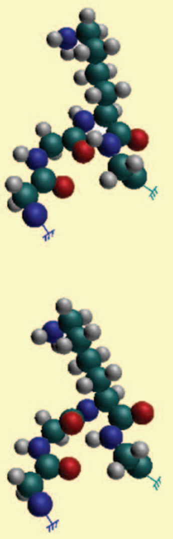
The CUIK Suite

Analyzing the Motion of Closed-Chain Multibody Systems

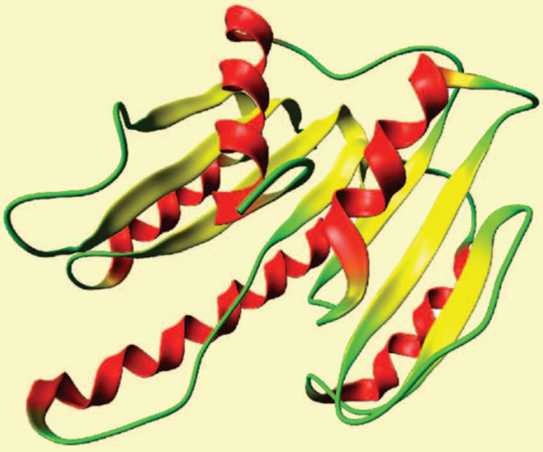
By Josep M. Porta, Lluís Ros, Oriol Bohigas, Montserrat Manubens, Carlos Rosales, and Léonard Jaillet







Many situations in robotics require the analysis of the motions of complex multibody systems. These are sets of articulated bodies arising in a variety of devices, including parallel manipulators, multifingered hands, or reconfigurable mechanisms, but they appear in other domains too as mechanical models of molecular compounds or nanostructures. Closed kinematic chains arise frequently in such systems, either due to their morphology or due to geometric or contact constraints to fulfill during operation, giving rise to configuration spaces of an intricate structure. Despite appearing very often in practice, there is a lack of general software tools to analyze and represent such configuration spaces. Existing packages are oriented either to open-chain systems or to specific robot types, which hinders the analysis and development of innovative manipulators. This article describes the CUIK suite, a software toolbox for the kinematic analysis of general multibody systems. The implemented tools can isolate the valid configurations, determine the motion range of the whole multibody system or of some of its parts, detect singular configurations leading to control or dexterity issues, or find collision- and singularity-free paths between configurations. The toolbox has applications in robot design and programming and is the result of several years of research and development within the Kinematics and Robot Design group at IRI, Barcelona. It is available under GPLv3 license from http://www.iri.upc.edu/cuik.



Motivation and Outlook

The notion of configuration space (C-space) is fundamental in robotics. It allows the design of motion analysis algorithms for broadly defined classes of robots without worrying about their particular geometry or multibody structure. In most robotics courses, this notion is introduced for open-chain robots, where the C-space has an explicit global parameterization. In this way, C-spaces are readily understood and algorithms operating on

Digital Object Identifier 10.1109/MRA.2013.2287462
Date of publication: 10 September 2014

them can be easily defined. In real problems, however, C-spaces can have a more complex structure. On a welding robot, for instance, the set of valid configurations reduces to those where the end-effector is in contact with the surface to be welded. This constraint implicitly defines a nonparametric C-space whose analysis is not trivial in general. Similar problems arise in structural biology, when analyzing the feasible motions of a molecule, or in computer-aided design, when assembling parts using spatial constraints. The CUIK suite provides effective tools aimed at analyzing the C-spaces arising in such a broad range of applications, a point not properly addressed in the related packages available so far.

To deal with larger dimensional spaces, the CUIK suite implements randomized path planning algorithms.

In all of the considered problems, the valid configurations are implicitly defined by a system of kinematic equations encoding the assembly, task, or contact constraints intervening in the problem, and the goal is to analyze the motion capabilities by finding the solutions of such a system. In an extreme case, the valid configurations are isolated points. This is what happens, for example, when solving position analysis problems on parallel or serial manipulators, where one wishes to compute the feasible configurations for given values of the input or output coordinates. Historically, the preferred approach to tackle these problems has been variable elimination: the initial equations are reduced to a resultant univariate polynomial, which is solved using well-established methods. The approach is sound, but it may introduce extraneous roots, and the size and degree of the resultant polynomial rapidly grow with the number of bodies and the complexity of their connection pattern. General elimination packages can be used [8], but they rapidly explode in complexity even on small problems, which explains why no general software for motion analysis has been built using them. The CUIK suite circumvents these issues by adopting an opposite approach. Instead of reducing the initial system of equations to a resultant polynomial, we formulate it as a larger system including only linear and quadratic equations. This particular formulation is then exploited to implement a branch-and-prune

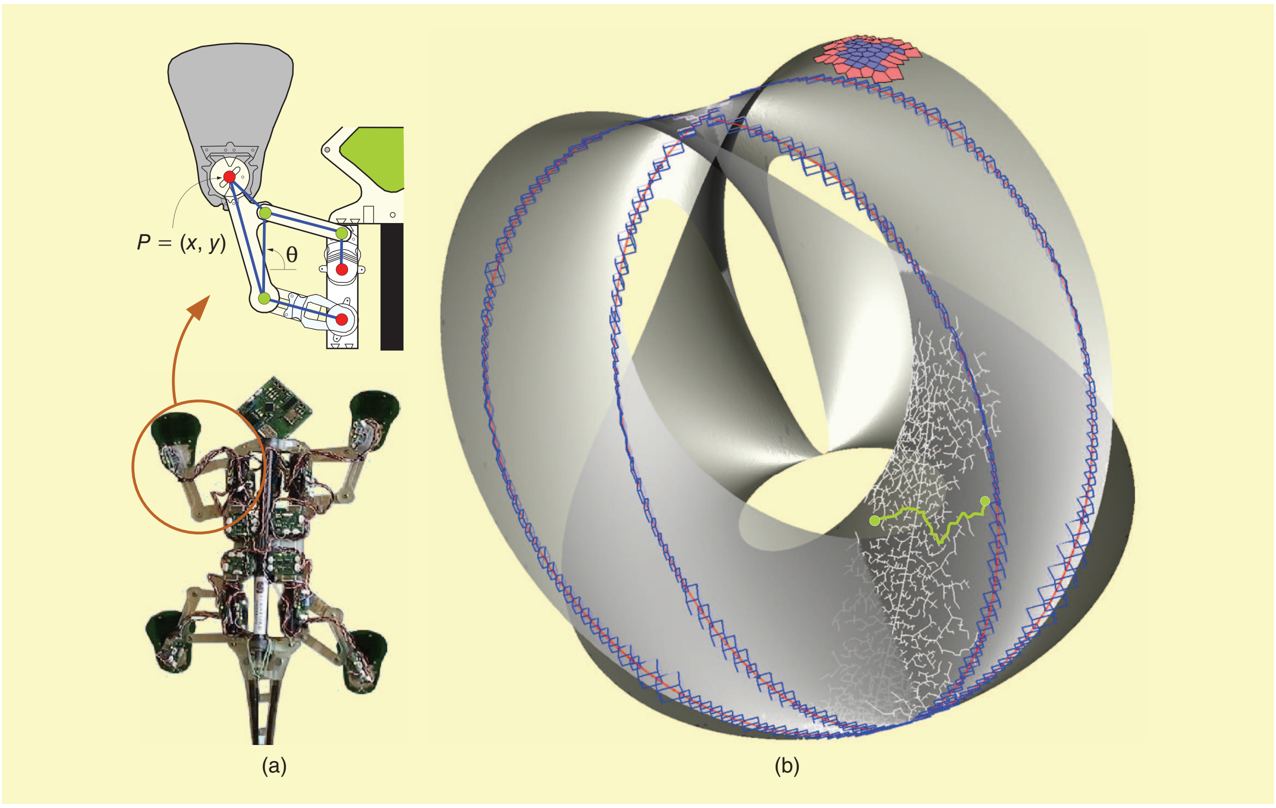


Figure 1. The results of the CUIK suite tools on the Stickybot III robot from the Stanford Biomimetics and Dexterous Manipulation Lab (http://bdml.stanford.edu). (a) Each leg has a five-bar mechanism (with actuated joints in red), (b) whose C-space is shown projected to the (x, y) coordinates of point P and angle θ. The suite allows a comprehensive analysis of the motions of systems of this kind. It can compute the C-space (the translucent gray surface), incrementally produce atlases to solve optimization or path-planning problems (the red and blue mesh at the top), obtain multiresolutive approximations of singularity loci (the blue boxes and the refined red curve), or generate probabilistic road maps to connect configurations (the white tree, with a path highlighted in green). The projection of this surface to the (x, y) plane produces the reachable workspace of point P.

method able to compute the whole solution set. The method departs from an initial box bounding this set and effectively prunes unfeasible portions of the box until all solution points are isolated at the desired accuracy. Software toolboxes, such as ALIAS [16], Bertini [1], and PHC [28], have been successfully applied to kinematics, but they implement general methods for solving systems of algebraic equations. In contrast, the CUIK suite sacrifices generality to gain simplicity and efficiency in the implementation. Opposite to [1] and [28], moreover, it directly isolates the real roots instead of the complex ones, which is beneficial in practice.

While many approaches can only identify the solutions when they are isolated points, the branch-and-prune methods in the CUIK suite can also approximate positive-dimensional solutions. This can be used to compute the whole C-space or some of its subsets, providing global information on the motion capabilities of a manipulator. In particular, by adequately defining the equations passed to the solver, the CUIK suite can isolate the workspace boundaries and any of the singularity sets typically defined [7], becoming the first general tool able to do so, to the best of our knowledge.

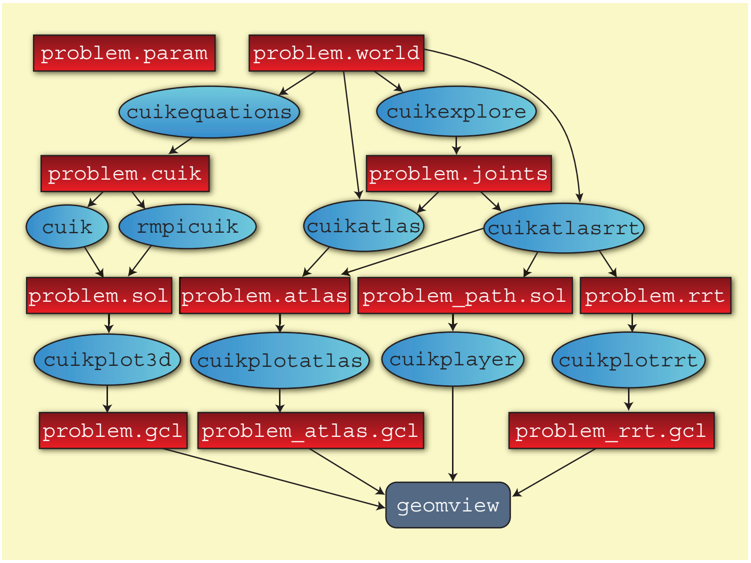


Figure 2. A representative set of the commands implemented in the CUIK suite (ellipses) and their input/output files (rectangles). The parameters file `problem.param` is used by all the commands. The suite implements many other tools not shown here for simplicity. Note: `geomview` is an external visualization program.

Branch-and-prune methods are complete in the sense that they obtain all configurations, irrespective of whether they form one or several connected components. In many problems, however, it may be sufficient to explore only those configurations that are path connected to a given point. To this end, the CUIK suite implements higher dimensional continuation tools to trace arbitrary manifolds [10]. While branch-and-prune methods proceed by discarding nonvalid configurations, continuation techniques march from a given point in all directions identifying new feasible configurations. Local charts of the C-space are constructed and coordinated along the way, defining a global atlas that is suitable to determine feasible motion ranges, optimal configurations, or paths between configurations. Packages to compute the latter exist, but they are oriented to open-chain robots [9], [15], [26], [27] or to specific classes of closed-chain devices [14]. The CUIK suite complements these packages by providing the new methods to deal with the general closed-chain case. For

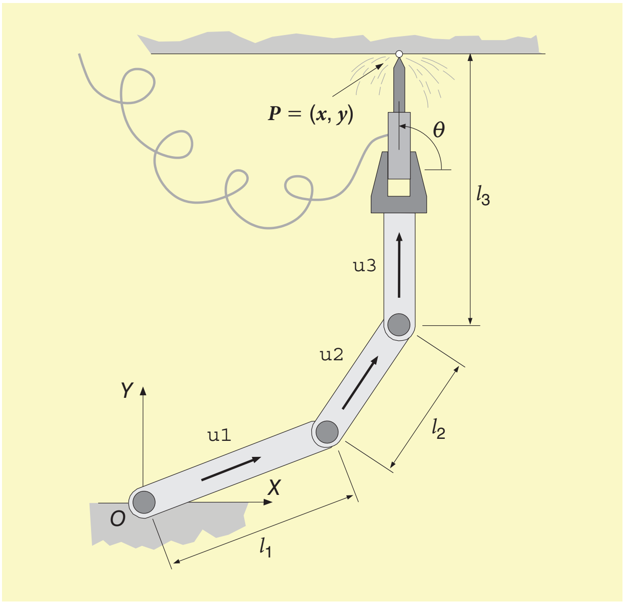


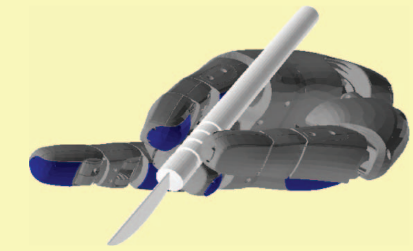
Figure 3. A planar arm welding on a beam. The goal is to weld on a particular point P with orientation θ.

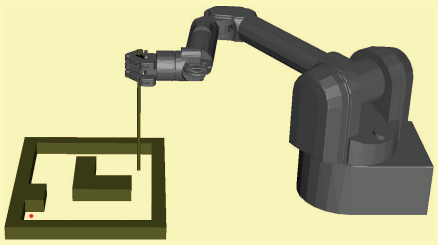
```
[constants]
  l1:= 1
  l2:= 1
  l3:= 1.3
  x:= 0.25
  y:= 2.05
  theta:= pi/2
[system vars]
  u1_x: [-1,1]
  u1_y: [-1,1]
  u2_x: [-1,1]
  u2_y: [-1,1]
  u3_x: [-1,1]
  u3_y: [-1,1]
[system eqs]
  u1_x^2+u1_y^2=1;
  u2_x^2+u2_y^2=1;
  u3_x=cos(theta);
  u3_y=sin(theta);
  l1*u1_x+l2*u2_x+l3*u3_x=x;
  l1*u1_y+l2*u2_y+l3*u3_y=y;
```

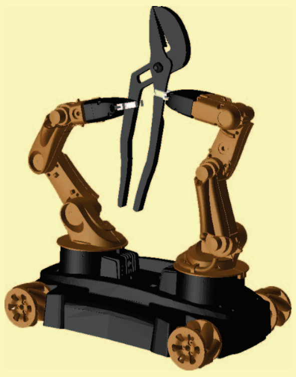
Example 1. The `welding.cuik` file encoding the problem in Figure 3.

```
[constants]
  l1:= 1
  l2:= 1
  l3:= 1.3
  x:= 0.25
  y:= 0.75
  theta:= pi/2
[links]
  environment: body "beam.off" blue
  link1:       body "link1.off" gray
  link2:       body "link2.off" gray
  link3:       body "link3.off" gray
               body "gripper.off" black
[joints]
  revolute: ground (0,0,0) (0,0,1)
            link1 (0,0,0) (0,0,1)
  revolute: link1 (l1,0,0) (l1,0,1)
            link2 (0,0,0) (0,0,1)
  revolute: link2 (l2,0,0) (l2,0,1)
            link3 (0,0,0) (0,0,1)
  fix:      environment
            link3
            Tx(x)*Ty(y)*Rz(theta)
```

Example 2. The `weldingOD.world` file from which a system of equations equivalent to that in Example 1 can be automatically generated.


(a)


(b)


(c)

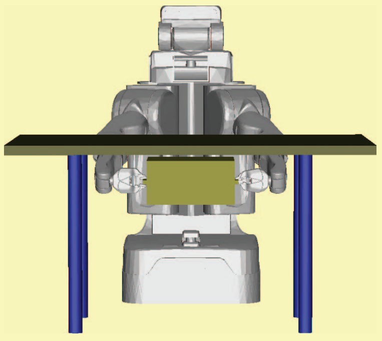

(d)

Figure 4. Some of the robots modeled in the CUIK suite: (a) the Schunk anthropomorphic hand, (b) the Barret hand-arm system, (c) the YouBot two-arm mobile platform, and (d) the PR2 service robot.

C-spaces of moderate dimension, the suite provides strategies to connect start and goal configurations through low-cost, collision- or singularity-free paths. To deal with larger dimensional spaces, the suite implements randomized versions of such tools based on the contruction of rapidly exploring random trees (RRTs) spanning the C-space.

The branch-and-prune methods of the CUIK suite can approximate zero- or positive-dimensional C-spaces.

Figure 1 shows the main outputs of the CUIK suite on a particular example. In the rest of this article, we describe the methods used to produce such outputs and the associated line commands, illustrating them on a simple tutorial example. A representative set of commands and their input/output files are shown in Figure 2. The documentation available online describes all the examples and functionalities in thorough detail.

Branch-and-Prune Methods

A basic example shows how the CUIK suite can address position analysis problems. Consider the planar manipulator in Figure 3, which has three links of lengths l_1, l_2, and l_3 and three revolute (3-RRR) joints. The goal is to compute the arm configurations, allowing it to weld at the indicated point $P = (x, y)$, with orientation θ. Example 1 shows the equations expressing this inverse kinematics problem in the suite. Observe that the arm configurations are represented using three normalized vectors, `(u1_x,u1_y)`, `(u2_x,u2_y)`, and `(u3_x,u3_y)`, encoding the orientation of each link relative to the x-axis of the global frame. The position of the end-effector is given by the last two linear equations, and the `[−1,1]` ranges for the variables

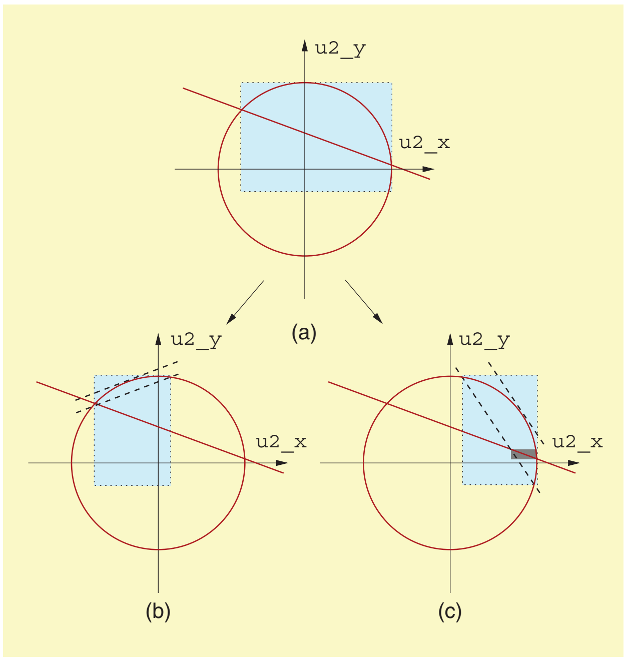


Figure 5. The bisection of (a) the blue box in the plot produces (b) and (c) the child boxes. By considering the linear equation and the relaxed version of the circle equation, we can accurately bound the solution points for the problem in Figure 3.

define a six-dimensional box bounding the location of all the possible solutions of the problem.

To facilitate the generation of the `cuik` files, the suite allows the definition of problems in a high-level form. By executing

```
> cuikequations welding0D.world
```

a set of equations equivalent to that in Example 1 is obtained from the `world` file shown in Example 2. This file describes the links and joints of the multibody system. A link definition includes the name of the link and its geometric shape, and a joint is given by its type, the two links connected, and the position of the joint in the local frame of each link. Figure 4 shows some of the robots modeled with `world` files in the CUIK suite. These predefined `world` files can be used as a starting point to easily define new problems.

By means of trivial manipulations, the equations in Example 1 can be simplified into the following system of equations:

```
0.5*u2_x + 1.5*u2_y = 0.625,
u2_x^2 + u2_y^2 = 1,
```

and the search box can be limited to the ranges $[-0.75, 1]$ and $[-0.25, 1]$ in `u2_x` and `u2_y`, respectively. The graph of these equations and the new search box are represented in Figure 5(a) using the solid and dotted lines, respectively. As shown, the solution points lie in the intersection of a line and a circle in the space of `u2_x` and `u2_y`. To identify such points, the box is first bisected along the `u2_x` axis, obtaining the two blue boxes Figure 5(b) and (c). The circular arc inside each blue box is then approximated by half-planes (shown as dashed lines in the figure). These half-planes are used in

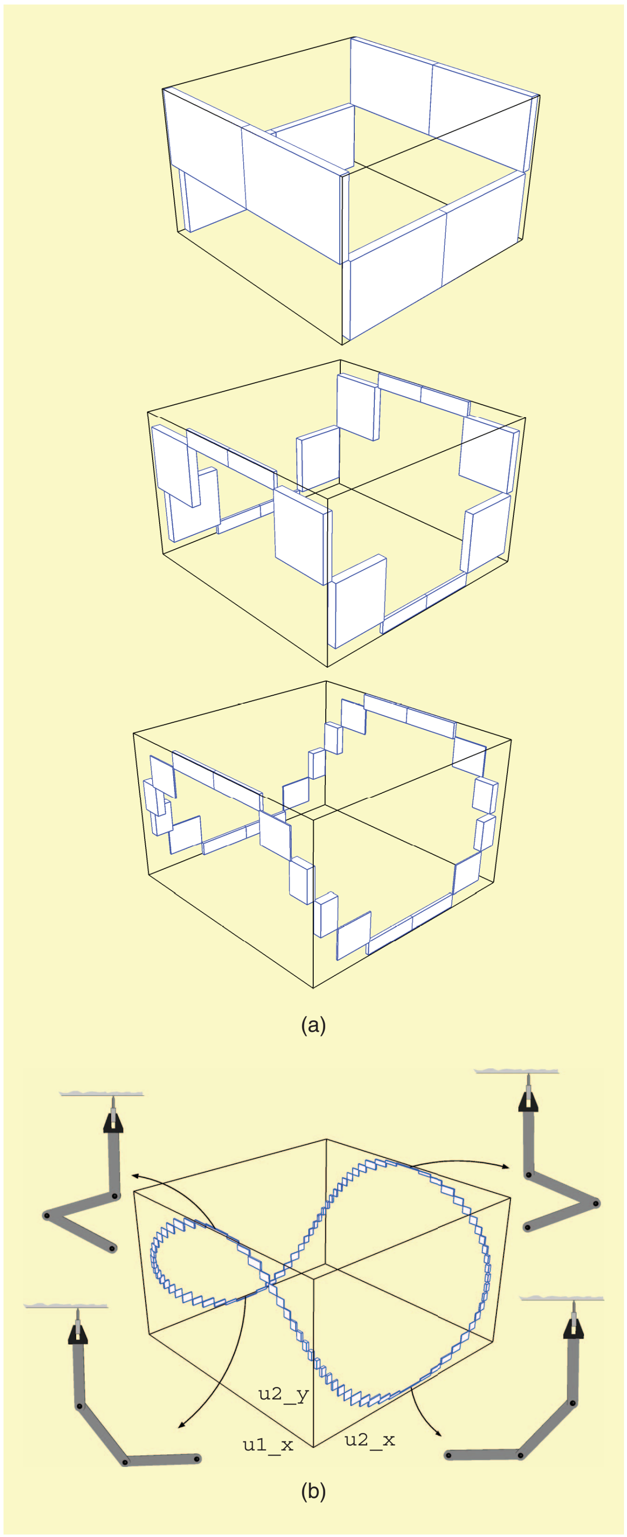


Figure 6. (a) The progress of the branch-and-prune method when computing the C-space of the robot arm in Figure 3, when the end-effector is set to contact the lower line of the beam with a fixed orientation. (b) The arm configurations for some of the solution boxes.

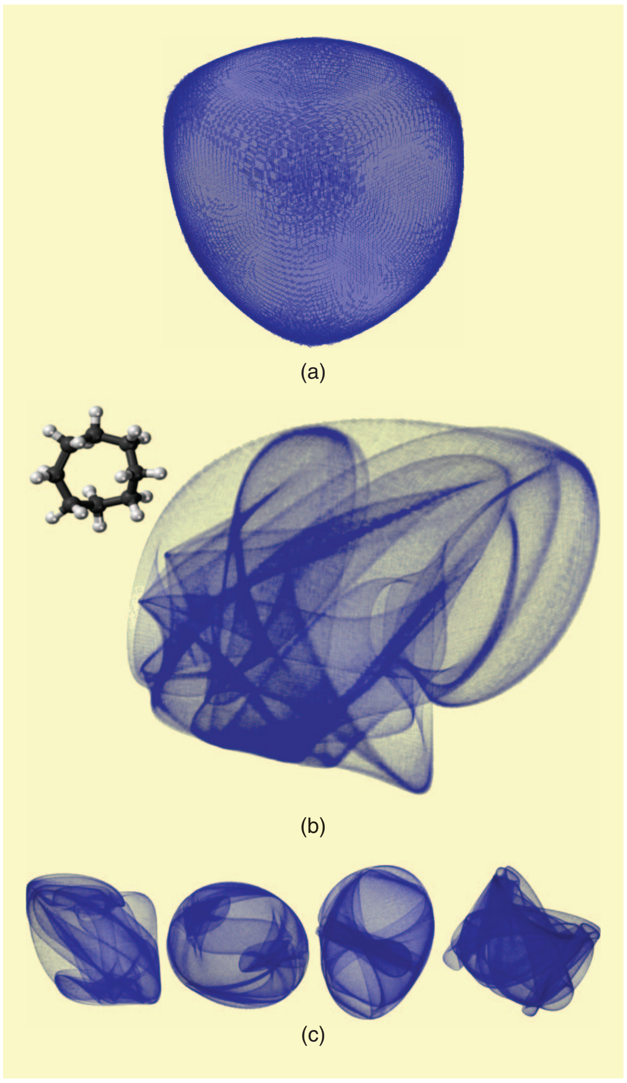


Figure 7. (a) A box approximation of the 2-D C-space of the welding task when the tool can contact the lower part of the beam with any orientation. The approximation contains ~15,000 boxes, which are here rendered with translucent material. (b) The cyclooctane molecule, a ring of eight carbon atoms pairwise connected through rotatable covalent bonds, and a box approximation of its 2-D C-space with ~300,000 boxes projected on three of the problem variables. (c) Different shapes are obtained by projecting the boxes on alternative triplets of variables.

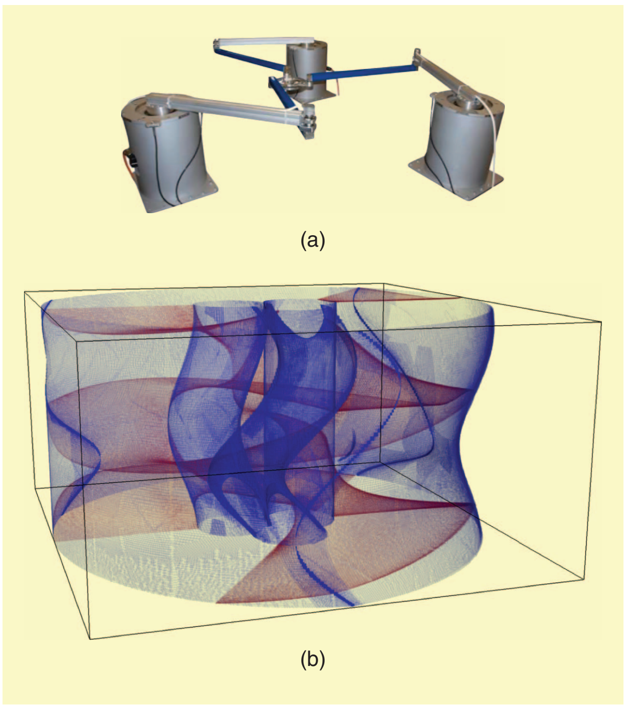


Figure 8. (a) A 3-RRR manipulator from the University of Hannover (photo courtesy of Jens Kotlarski http://www.imes.uni-hannover.de). (b) Typical singularity loci computed for such kind of manipulators [6].

conjunction with the linear equation to derive a tiny box around the solution in Figure 5(b) and a larger dark gray box bounding the solution in Figure 5(c). This pruning operation is implemented using linear programming, and it is repeated for each box until no further significant reduction is possible. In the end, if the largest side of the box is below a given threshold, it is considered a solution box. Otherwise, it is bisected and the process is recursively applied to the newly created subboxes. The following command:

```
> cuik welding0D.cuik
```

automates this process. It reads the `cuik` file, simplifies the equations when possible, and applies the pruning and bisection operations until all solution points are isolated at the desired accuracy. The problem in Figure 3 is simple, but the method is also successful on complex problems involving general 6R robots and Stewart platforms [21] or in mechanisms with more than 12 kinematic loops [20]. To have an idea, elimination or resultant methods are finding their limit in mechanisms of substantially less complexity [23].

The `cuik` command can also be used to isolate C-spaces of positive dimension. For instance, if the robot in Figure 3 has to weld along a line in the lower part of the beam while keeping the tool orientation fixed, then the problem exhibits a one-dimensional manifold of solutions. Figure 6 shows several stages of the branch-and-prune method in this case [Figure 6(a)] and the final box approximation returned as output [Figure 6(b)]. The plots are obtained and visualized by executing

```
> cuik welding1D.cuik
> cuikplot3d welding1D.sol 1 3 4
  welding1D.gcl
> geomview welding1D.gcl
```

which projects the solution boxes to the subspace of the `u1_x`, `u2_x`, and `u2_y` coordinates. These are the variables appearing in the first, third, and fourth positions of the `welding1D.cuik` file describing the problem. The insets in Figure 6(b) show the arm configurations corresponding to some of the boxes.

If the welding task is further relaxed and the tool can contact the beam with any orientation, the problem exhibits a two-dimensional (2-D) C-space. Such a space can be interactively explored using the following command:

```
> cuikexplore welding2D.world
```

and it can again be approximated using the `cuik` command, producing the results in Figure 7(a). The box approximation includes ~15,000 boxes that are computed in just 45 s on a standard desktop computer. To solve more difficult problems in a reasonable amount of time, the CUIK suite implements a parallel version of the solver, which can be invoked through the `rmpicuik` command. With this tool, the branch-and-prune solver can be executed in large computer grids, where a "master" CPU manages the exploration tree and a number of "slave" CPUs apply the pruning operations to the assigned boxes. Using this tool on a cluster with 160 CPUs, it took a few minutes to isolate 2-D C-spaces like the one in Figure 7(b). This process would require costly computations using single-CPU machines or alternative approaches [22].

C-spaces of robotic systems typically exhibit singularity subsets. These are loci of configurations where problematic losses of control or dexterity arise [6], [7]. They also reveal the boundaries of the task and joint workspaces and all motion barriers that may be encountered in their interior [4]. The ability to compute these loci is thus essential, not only to anticipate possible problems during robot operation but also to provide valuable information to the robot designer. The CUIK suite can be employed to isolate all singular configurations of a manipulator by appropriately formulating their equations and passing them to the solver [2], [7]. For example, Figure 8 shows typical forward and inverse singularity loci obtained by the suite tools on a 3-RRR manipulator, shown projected to the pose coordinates of the end effector, x, y, and θ. The blue surface corresponds to the inverse singularities, which delimit the boundary of the (x, y, θ) workspace. The red surface corresponds to the forward singularities, which indicate the configurations in which velocity control issues arise [6].

Continuation Methods

Continuation methods generate atlases of the C-space regions that are connected to a given point. To see how such atlases can be constructed, let us assume that $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ is any system of equations encoding the kinematic constraints of the multibody system, whose solution set constitutes the C-space $\mathcal{C}$ under consideration. Also, let $\mathbf{x}_i$ be an initial configuration satisfying $\mathbf{F}(\mathbf{x}_i) = \mathbf{0}$. The points in $\mathcal{T}_i$, the tangent space of $\mathcal{C}$ at $\mathbf{x}_i$, can be parameterized by

$$\mathbf{x}_i' = \mathbf{x}_i + \mathbf{\Phi}_i \mathbf{u}, \tag{1}$$

where $\mathbf{\Phi}_i$ is a matrix providing an orthonormal basis of $\mathcal{T}_i$, and $\mathbf{u}$ is a parameter vector with the same dimension as $\mathcal{C}$. By choosing a value for $\mathbf{u}$ in (1), we obtain a point $\mathbf{x}_i' \in \mathcal{T}_i$, which can be projected to $\mathbf{x}_j$, the point in $\mathcal{C}$ lying in the normal line through $\mathbf{x}_i'$, by solving the system

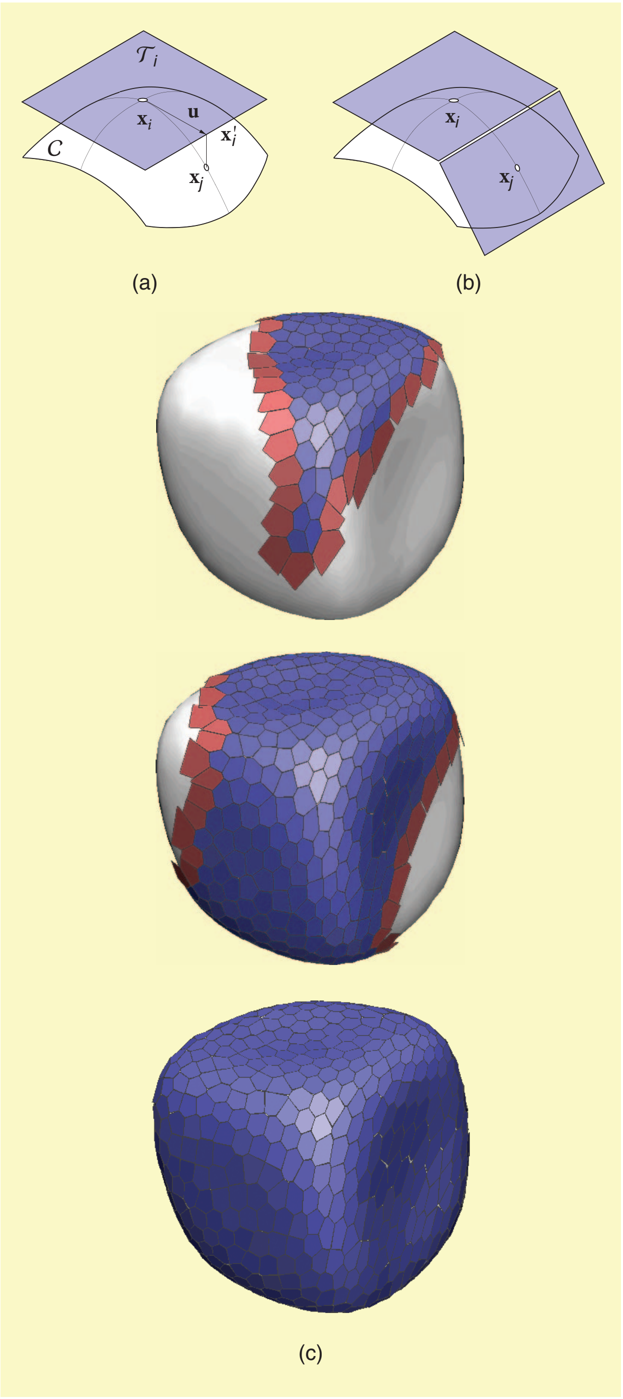


Figure 9. (a) A chart is used to obtain new configurations by projecting points from the tangent space. (b) When a new chart is defined, the chart domains are mutually coordinated to keep track of the explored area. (c) Progress of the atlas construction method on the C-space of the robot arm in Figure 3, assuming that the welding tool can contact the lower part of the beam with any orientation. This C-space is shown projected on three of the problem variables. Red polygons represent the charts to be extended in subsequent iterations.

$$\begin{cases} \mathbf{F}(\mathbf{x}_j) = \mathbf{0}, \\ \mathbf{\Phi}_i^{\top} (\mathbf{x}_j - \mathbf{x}_i') = \mathbf{0}, \end{cases} \tag{2}$$

as shown in Figure 9(a). The new configuration $\mathbf{x}_j$ can be used to define a new chart that can be coordinated with the

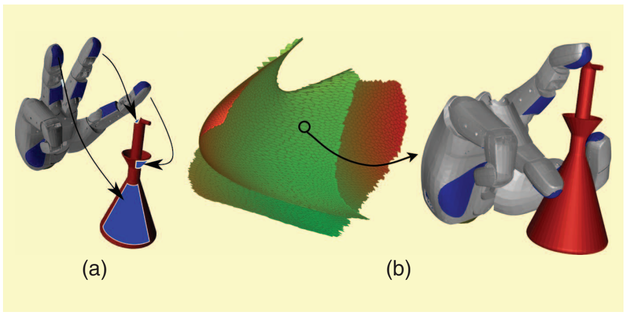


Figure 10. An optimization problem in the context of grasp synthesis. (a) The configuration of the hand-object system must satisfy a number of contact constraints. An initial feasible grasp is computed using the branch-and-prune methods [25] and is used as the initial configuration from which to construct an atlas of the relevant C-space subset [24]. (b) The chart centers of this atlas can be evaluated according to a performance criterion in order to select the optimal grasp. The red and green colors in the atlas correspond to low- and high-quality grasps, respectively.

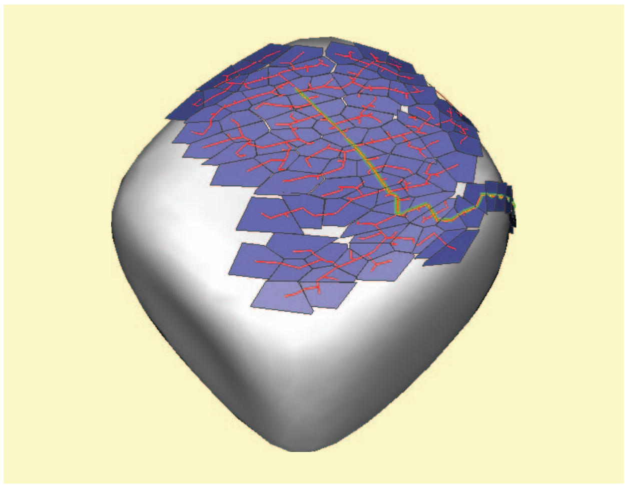


Figure 12. An RRT and a partial atlas solving a path-planning query over the C-space manifold of the welding robot when the tool can contact the lower part of the beam with any orientation. In this problem, the robot has to avoid colliding with the beam. The path over the RRT solving the planning query is shown in green.

previous chart [Figure 9(b)], and the process can be iterated until the component of C reachable from $\mathbf{x}_i$ gets fully covered [Figure 9(c)]. Every time a new chart is defined, the CUIK suite checks for the presence of bifurcations of C and propagates the atlas construction through such bifurcations so as not to leave areas unexplored. The command

```
> cuikatlas welding2D
```

takes the `world` file describing the robot welding problem when the robot can contact the lower part of the beam with any orientation and a `joints` file providing the initial configuration. As output, it returns an `atlas` file including the charts, which can be visualized using

```
> cuikplotatlas welding2D 0 9 18
> geomview welding2D_atlas.gcl
```

where the indices 0, 9, and 18 indicate the three coordinates on which to represent the atlas. The results of these commands are shown in Figure 9(c). In this case, the atlas construction takes 0.1 s on a standard desktop computer, while the isolation of the same space using branch-and-prune techniques requires ~45 s. Such a remarkable speedup comes at the expense of neglecting other connected components but allows for the tackling of difficult optimization problems involving large multibody systems (Figure 10).

To solve path-planning problems, the CUIK suite exploits the fact that an atlas implicitly defines a road map of the C-space, whose nodes and edges are, respectively, the chart centers and collision-free transitions between neighboring charts. The collisions to avoid can be specified in the `world` files. The road map can be readily used to resolve multiple planning queries between different configurations. However, for cases where only one query needs to be resolved, it is better to employ the `cuikatlasAstar` tool, which constructs only those charts leading to the shortest path between the given configurations. With this tool, singularity-free path planners for general closed-chain [3] and cable-driven manipulators [5] have been developed, solving problems for which no alternative satisfactory solution had been given to date (Figure 11). Trading off optimality for efficiency, the `cuikatlasGBF` tool

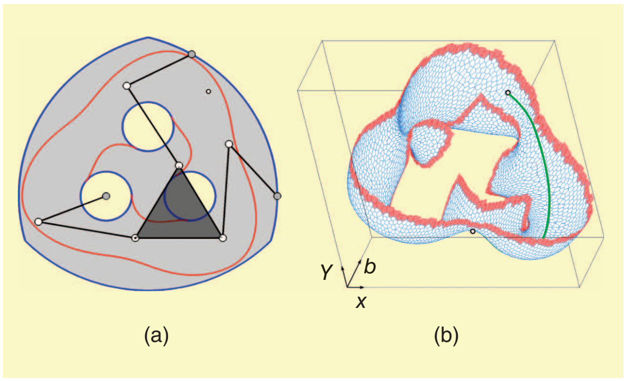


Figure 11. (a) A singularity-free path-planning problem on a 3-RRR manipulator [3], the gray area is the constant-orientation workspace of the manipulator, and the red curves correspond to the forward singularity locus to be avoided during the move. (b) The solution path computed by the CUIK suite, in green, projected on the three variables relevant to the problem. The red charts correspond to configurations that are almost singular.

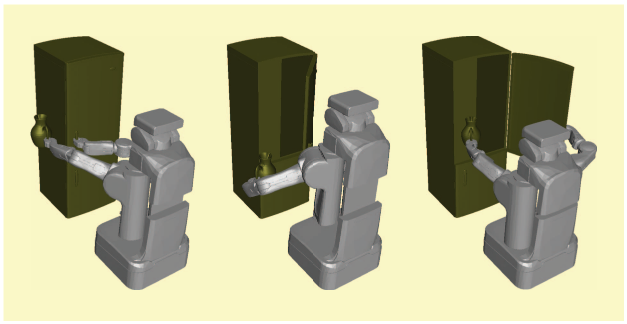


Figure 13. The different stages of the execution of a path allowing the arm to put a pitcher in a fridge that is initially closed.

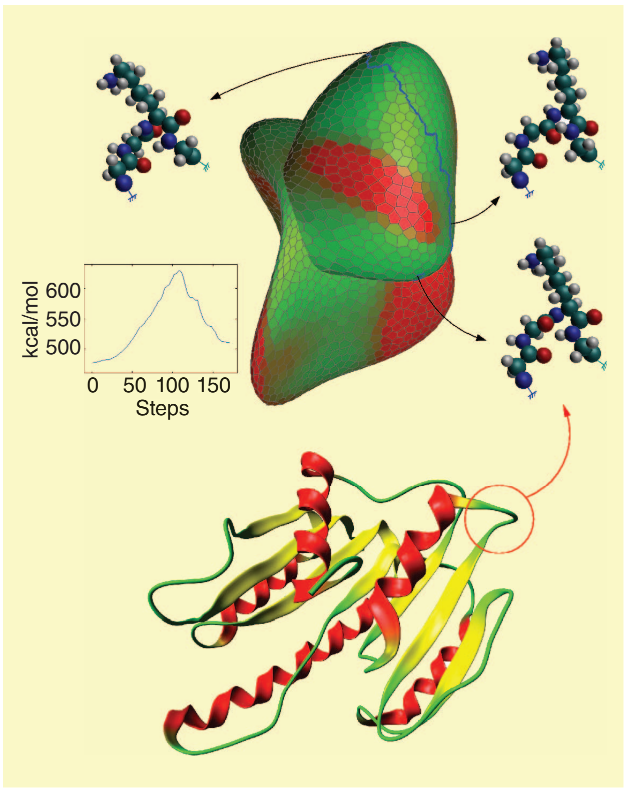


Figure 14. A low-cost path (in blue) computed in the conformational space of a loop of the FTSJ protein of *Escherichia coli* (in the ribbon diagram in the bottom). The cost is the potential energy of each conformation. Inset: the initial conformation, the transition state (i.e., the conformation with the highest potential energy along the path), and the final conformation. Only the atoms in the loop are shown in such conformations. The plot shows the energy profile along the transition path.

provides a path planner that implements a greedy best first strategy to connect the query configurations [19].

Both the `cuikatlasAstar` and the `cuikatlasGBF` tools can be inefficient in cluttered environments, and they do not scale gracefully to higher dimensions. To avoid these weaknesses, the CUIK suite includes the `cuikatlasrrt` tool, which implements a sampling method where a partial atlas is used to extend an RRT, which in turn is exploited to decide expansion directions for the atlas [13]. For instance, after specifying the start and goal configurations in the `joints` file, the sequence of commands

```
> cuikatlasrrt welding2D
> cuikplotatlas welding2D 0 9 18
> cuikplotrrt welding2D 0 9 18
> geomview welding2D_rrt.gcl
  welding2D_atlas.gcl
```

produces an RRT and an atlas like those shown in Figure 12. The motion of the robot along the path solving the planning problem can be visualized executing

```
> cuikplayer welding2D welding2D_path
```

Using this technique, it is possible to solve problems in pretty high dimensions. Figure 13, for example, shows three snapshots of the path computed by this method in a manipulation problem involving a PR2 robot. The robot has to carry a pitcher without tilting it and put it in a fridge that is initially closed. The two arms are open-chain robots, but their configurations are restricted by task and contact constraints, respectively, which make the planning problem rather difficult. The C-space is eight-dimensional, but it takes less than 3s to determine the solution path.

Paths generated with RRT-like algorithms might be costly or unnecessarily long. To address these issues, the CUIK suite includes procedures to generate near-optimal paths when there is a cost function related to the C-space. If the cost is defined for each configuration, one can use the `cuikatlastrrt` tool, which implements an extended version of the T-RRT algorithm in [11]. For instance, Figure 14 shows a low-cost path computed with this method in the case of a short loop of the FTSJ protein of *Escherichia coli*, where the cost function is the potential energy of each protein conformation [18]. If the cost is the length of the path, the `cuikatlasrrtstar` tool can be used instead, which is an adaptation of the RRT* asymptotically optimal path planner to the case of implicitly defined C-spaces [12].

To solve more difficult problems in reasonable times, the CUIK suite implements a parallel version of the solver.

Conclusions

This article described the CUIK suite, a comprehensive set of tools to analyze C-spaces implicitly defined by systems of kinematic constraints. We provided a brief account of the underlying techniques and the commands used to invoke them. Since problems involving kinematic constraints are ubiquitous in robotics, the suite may potentially be used in contexts beyond those described in this article, including mobile robot localization and mapping [17], motion analysis and synthesis of robot formations, tensegrity and deployable structures, or programmable surfaces, to name a few.

In the future, we plan to extend the suite to also accommodate the dynamics of the multibody systems to facilitate a direct interfacing with robots operating under inertia effects. However, the suite is an open-source package under continuous development, and hence, we invite the community to use it and help us to improve it by sending feedback and suggestions on new functionalities to include.

Acknowledgments

The branch-and-prune methods in the CUIK suite have evolved from years of collaboration with Federico Thomas, whose guidance and inspiring comments have certainly shaped the final result. The authors would like to thank Michael E. Henderson for introducing them to the higher dimensional continuation methods and Dimiter Zlatanov for his feedback and support on the analysis and interpretation of mechanism singularities.

This work was partially supported by the Spanish Ministry of Economy and Competitiveness under projects DPI2010-18449 and DPI2011-13208-E, by CSIC project 201250E026, by a Juan de la Cierva fellowship supporting Montserrat Manubens, and by a CSIC JAE-Doc fellowship partially funded by the ESF supporting Léonard Jaillet.

References

[1] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. (2014). The Bertini software. [Online]. Available: http://www3.nd.edu/~sommese/bertini

[2] O. Bohigas. (2013). Numerical computation and avoidance of manipulator singularities. Ph.D. dissertation, Institut de Robòtica i Informàtica Industrial, Universitat Politècnica de Catalunya, Barcelona, Catalonia. [Online]. Available: http://goo.gl/wlR0i

[3] O. Bohigas, M. E. Henderson, L. Ros, M. Manubens, and J. M. Porta, "Planning singularity-free paths on closed-chain manipulators," *IEEE Trans. Robot.*, vol. 29, no. 4, pp. 888–898, 2013.

[4] O. Bohigas, M. Manubens, and L. Ros, "A complete method for workspace boundary determination on general structure manipulators," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 993–1006, 2012.

[5] O. Bohigas, M. Manubens, and L. Ros, "Navigating the wrench-feasible C-space of cable-driven hexapods," in *Proc. Int. Conf. Cable-Driven Parallel Robots*, 2012, pp. 53–68.

[6] O. Bohigas, M. Manubens, and L. Ros, "Singularities of non-redundant manipulators: A short account and a method for their computation in the planar case," *Mech. Mach. Theory*, vol. 68, pp. 1–17, Oct. 2013.

[7] O. Bohigas, D. Zlatanov, L. Ros, M. Manubens, and J. M. Porta, "A general method for the numerical computation of manipulator singularity sets," *IEEE Trans. Robot.*, vol. 30, no. 2, pp. 340–351, 2014.

[8] D. Cox, J. Little, and D. O'Shea, *An Introduction to Computational Algebraic Geometry and Commutative Algebra*. 3rd ed. New York: Springer-Verlag, 2007.

[9] I. A. Şucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robot. Automat. Mag.*, vol. 19, no. 4, pp. 72–82, 2012.

[10] M. E. Henderson, "Multiple parameter continuation: Computing implicitly defined k-manifolds," *Int. J. Bifurcation Chaos*, vol. 12, no. 3, pp. 451–476, 2002.

[11] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robot.*, vol. 26, no. 4, pp. 635–646, 2010.

[12] L. Jaillet and J. M. Porta, "Efficient asymptotically-optimal path planning on manifolds," *Robot. Auton. Syst.*, vol. 61, no. 8, pp. 797–807, 2013.

[13] L. Jaillet and J. M. Porta, "Path planning under kinematic constraints by rapidly exploring manifolds," *IEEE Trans. Robot.*, vol. 29, no. 1, pp. 105–117, 2013.

[14] LAAS-CNRS. (2014, Feb.). Move 3D. [Online]. Available: http://www.openrobots.org/wiki/move3d

[15] S. LaValle. (2014, Feb.). The motion strategy library. [Online]. Available: http://msl.cs.uiuc.edu/msl

[16] J.-P. Merlet. (2014, Feb.). Alias software. [Online]. Available: ftp://ftp-sop.inria.fr/coprin/ALIAS

[17] J. M. Porta, "CuikSLAM: A kinematic-based approach to SLAM," in *Proc. IEEE Int. Conf. Robotics Automation*, 2005, pp. 2425–2431.

[18] J. M. Porta and L. Jaillet, "Exploring the energy landscapes of flexible molecular loops using higher-dimensional continuation," *J. Comput. Chem.*, vol. 34, no. 3, pp. 234–244, 2013.

[19] J. M. Porta, L. Jaillet, and O. Bohigas, "Randomized path planning on manifolds based on higher-dimensional continuation," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 201–215, 2012.

[20] J. M. Porta, L. Ros, T. Creemers, and F. Thomas, "Box approximations of planar linkage configuration spaces," *ASME J. Mech. Des.*, vol. 127, no. 4, pp. 397–405, 2007.

[21] J. M. Porta, L. Ros, and F. Thomas, "A linear relaxation technique for the position analysis of multiloop linkages," *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 225–239, 2009.

[22] J. M. Porta, L. Ros, F. Thomas, F. Corcho, J. Cantó, and J.-J. Pérez, "Complete maps of molecular-loop conformational spaces," *J. Comput. Chem.*, vol. 29, no. 1, pp. 144–155, 2008.

[23] N. Rojas and F. Thomas, "Closed-form solution to the position analysis of Watt–Baranov trusses using the bilateration method," *ASME J. Mech. Robot.*, vol. 3, no. 3, p. 031001, 2011.

[24] C. Rosales, J. M. Porta, and L. Ros, "Grasp optimization under specific contact constraints," *IEEE Trans. Robot.*, vol. 29, no. 3, pp. 746–757, 2013.

[25] C. Rosales, L. Ros, J. M. Porta, and R. Suárez, "Synthesizing grasp configurations with specified contact regions," *Int. J. Robot. Res.*, vol. 30, no. 4, pp. 431–443, 2011.

[26] M. Saha. (2014, Feb.). The motion planning kit. [Online]. Available: http://ai.stanford.edu/~mitul/mpk

[27] N. Vahrenkamp. (2014, Feb.). Simox. [Online]. Available: http://simox.sourceforge.net

[28] J. Verschelde. (2014, Feb.). The PHC software. [Online]. Available: http://homepages.math.uic.edu/~jan/PHCpack/phcpack.html

Josep M. Porta, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. E-mail: porta@iri.upc.edu.

Lluís Ros, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. E-mail: lros@iri.upc.edu.

Oriol Bohigas, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. E-mail: obohigas@iri.upc.edu.

Montserrat Manubens, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. E-mail: mmanuben@iri.upc.edu.

Carlos Rosales, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. E-mail: crosales@iri.upc.edu.

Léonard Jaillet, NANO-D, INRIA Rhône-Alpes, Grenoble, France. E-mail: leonard.jaillet@inria.fr.