# Motion Planning: State of the Art and Perspectives

Jean-Paul Laumond

LAAS-CNRS, 7, Avenue du Colonel Roche, 31077 Toulouse, France
e-mail: jpl@laas.fr , http://www.laas.fr/~jpl

**Abstract**

*This paper presents an overview of algorithmic motion planning techniques together with their current and potential applications. After introducing the main concepts appeared in the late 70's in Robotics, the paper focuses on probabilistic techniques of the 90's which constitute a technological breakthrough with respect to the deterministic approaches developed in the 80's. We then see applications in manufacturing (robot programming), CAD/CAM Design (mechanical part assembly), as well as in Process Engineering (maintenance operation in industrial facilities)*

## 1 PROBLEM STATEMENT

What is a motion planning problem?

- Consider a mechanical assembly consisting of interacting parts. Is a given part movable without moving the other ones?
- Is such a freight movable with a rolling bridge or should I use preferably a crane?
- Should my robot manipulator reconfigure itself between these two welding points?
- How the mobile robot should operate to access this cluttered place?

All these problems are motion planning problems. In all the cases, the problem is to decide if there is a motion for the device (i.e. the part, the freight, the manipulator or the robot) that avoids the obstacles of the environment while achieving the fixed goal.

The inputs of a motion planning problem are:

- an environment
- a device
- a starting configuration and a goal configuration of the device

The output is:

- a collision-free admissible path

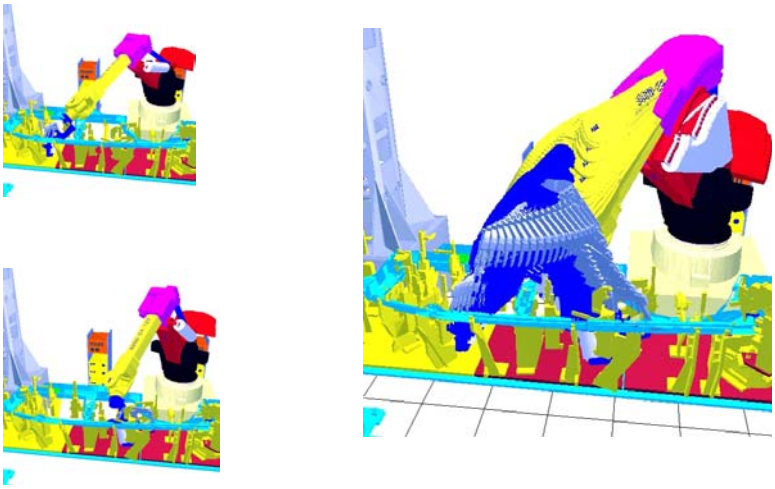


**Figure 1.** In this example the problem is to teach the robot manipulator going automatically from a given welding point to another one while avoiding all the parts of the scene.

The environment is described by a geometric model giving the shape and the position of all the obstacles. The device is the set of bodies to be moved. This can be a single body as in the case of a mechanical part to be removed from an assembly, or a set of articulated bodies linked together by prismatic or rotating joints as for a robot manipulator.

The solution a motion planner has to compute is a path that avoids the obstacles (it should be collision-free) and that accounts for the kinematics of the device (it should be feasible).

## 2 CONFIGURATION SPACE

Motion planning deals with the problem of moving 3-dimensional bodies within a 3-dimensional space. It is known as "the piano mover's problem". Imagine two mover teams. The first one should move a big piano from the outside an apartment to the inside. The second one should move a ping-pong ball. The second problem seems to be easier than the first one... Do not stop reading: both problems are the same as soon as they are expressed in the right space.

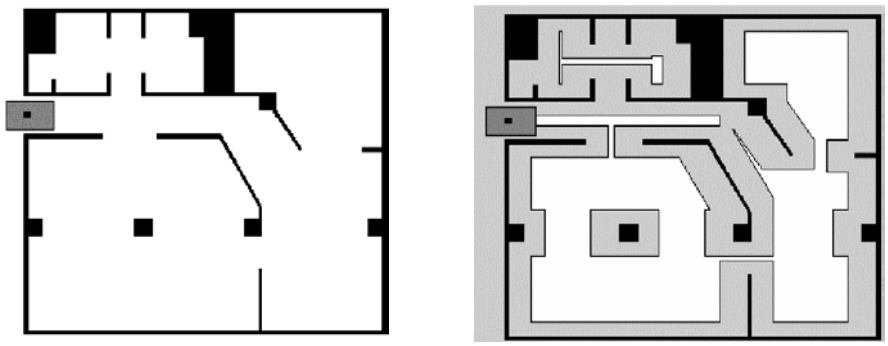
The configuration space is the key concept allowing to transform the problem of moving a body into the problem of moving a point.

### 2.1 TRANSFORMING A TRANSLATING PIANO INTO A POINT

Consider the piano mover's problem in the situation of the apartment depicted in figure 2 below. Assume that the piano cannot rotate for some reason. The problem is to know if it can be moved at the corner of the room down right.

The position of the piano is specified without any ambiguity as soon as one gets the coordinates of its middle point within a frame of the room. Let us move the piano in contact along all the walls and all the obstacles. The middle point describes a boundary splitting the plane into two regions: when the point is located in the first one, the piano is guaranteed to be collision-free with respect to the obstacles, while it is not as soon as the point lies in the second region. The complete information about moving the piano without colliding the obstacles is concentrated within the second region. For instance we see clearly that it is possible to reach the goal assigned to the piano. However there is no possible solution by going through the corridor. The only possibility is to go through to big room.

The region gathering all the collision-free positions has two components. This means that it is not possible to move the piano from any point inside the first region to any point inside the other one. For instance, it is not possible to move the piano from the apartment entrance to the small room on top left.



*The piano at the entrance of the apartment (left). As soon as the reference point of the piano belongs to the white domains of the figure on the right, the piano is guaranteed to be collision-free.*

**Figure 2.** The set of all the piano positions is the *configuration space* of the piano.

## 2.2 TRANSFORMING A TRANSLATING AND ROTATING SQUARE INTO A POINT

While two parameters are sufficient to locate the translating piano above, an additional parameter is required to locate a translating and rotating polygon: this is the direction of the polygon. Therefore the space of all the parameters required to locate the square is 3-dimensional. The set of all collision-free positions then constitutes a 3-dimensional object. Here the *configuration space* has dimension 3.

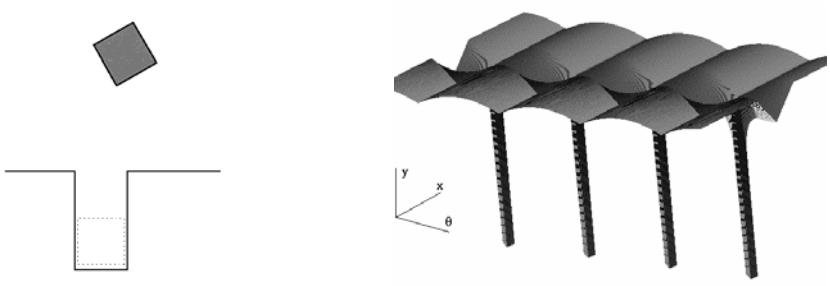



**Figure 3.** A translating and rotating square amidst a polygonal obstacle and its collision-free positions in 3D. Notice that the four "picks" correspond to the four orientations of the square within the hole.

## 2.3 TRANSFORMING A MECHANICAL SYSTEM INTO A POINT

Here is another example of transformation. The figure 4 below represents a simplified 2 degrees of freedom manipulator. The first body is fixed at one of its endpoints and can rotate freely around that point. The second body is fixed to the second endpoint of the first body and can rotate freely around that point. The position of the mechanical system is fully characterized as soon as two angles defining the respective orientations of the ladders are known.

A priori the angles range from 0 to 360 degrees. Therefore the range of each angle may be represented by a circle. A convenient representation of the system is to consider a torus. Indeed there is a one-to-one mapping between the points of a torus and all the pairs of angles. The torus is the *configuration space* of the mechanical system. The motions of the systems then appear as paths on a torus.

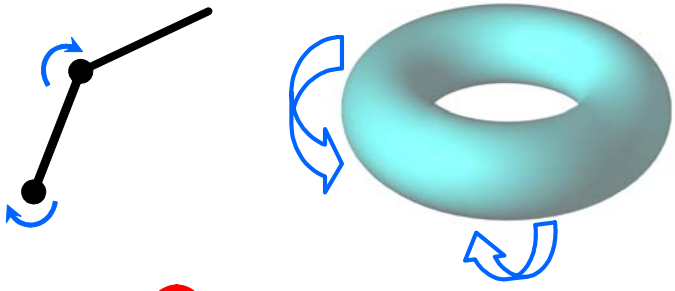


**Figure 4.** The configuration space of this 2 degrees of freedom system is the surface of a torus.

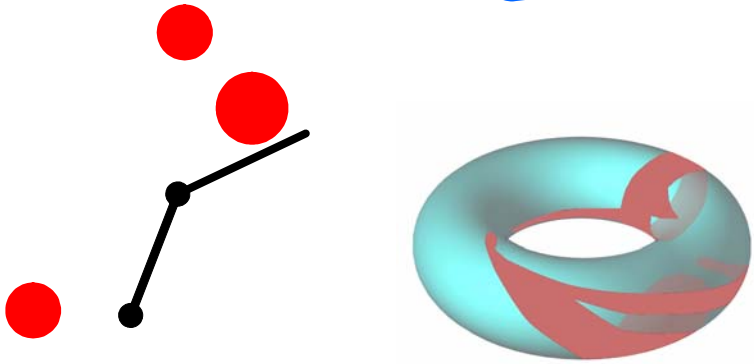


**Figure 5.** The obstacles in the real space create domains to be avoided on the torus. The mapping between the obstacles in the real space and the obstacles in the configuration space is not intuitive. However it can be formally computed.

### 2.4 THE MAIN PROPERTY

The transformations above are general. Any device has its own configuration space. The configuration space of a single body that can freely move in a 3D workspace is the space of all the six parameters locating the device in the space. If the device is a single body moving on a plane, a configuration is composed of two position parameters and one direction parameter. For a mechanical system the configuration space is the set of the parameters necessary to locate precisely the system in its environment. They are the degrees of freedom of a manipulator.

Any path in the configuration space corresponds to a continuous variation of the position parameters and then to a motion for the considered system.

For most systems, reaching a goal from a starting configuration is possible as soon as both start and goal configurations belong to a same connected component of the collision-free configuration space. Therefore *finding a collision-free path for the bodies of a device is equivalent to finding a collision-free path for a point in the configuration space of the device*.

## 3 CONFIGURATION SPACE SEARCHING: THE COMPUTATIONAL CHALLENGE STORY

The figure 6 below shows an abstract configuration space with two obstacles. We "see" that there is a collision-free path between points ***a*** and ***b*** while there is no collision-free path between points ***a*** and ***c***.

A path is a continuous function from some real number interval [0,1] into the configuration space. Searching a path is a problem that deals with the topology of the collision-free configuration space. There is collision-free path between two points if and only if these two points belong to a same connected component of the collision-free space.

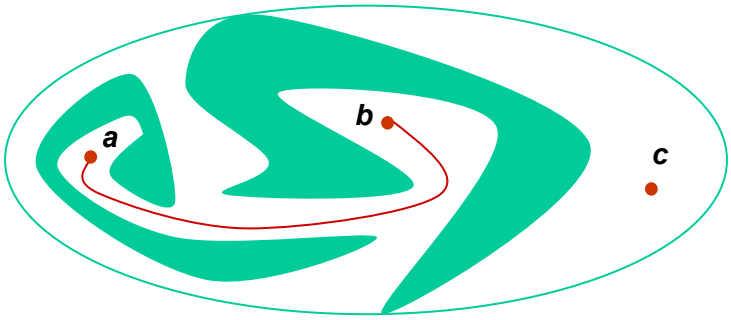



**Figure 6.** How to find a collision-free path for a point moving in a space cluttered with obstacles? How to decide that there is solution path between **a** and **b** and no solution between **a** and **c**?

Finding a path is fundamentally a continuous problem. A computer only computes. *How to transform a continuous problem into a combinatorial one?* This is the core question of the algorithmic motion planning.

Algorithmic motion planning appeared as an active research area at the end of the seventies with the pioneering work of MIT researchers who introduced the notion of configuration space above.

During the eighties researchers from mathematics and computational geometry (e.g. at Current Institute in New York and University of California in Berkeley) attack the problem. The problem is proven to be decidable: there are formal algorithms capable to prove that there is a collision-free path between points *a* and *b* and that there is no path between *a* and *c* in the above example. Such algorithms are said to be exact and complete. Nevertheless their combinatorial complexity is shown to

be very high. There is no hope to get formal generic and practical solutions. Only some dedicated cases that deal mainly with polygonal objects moving on a plane give rise to efficient solutions. Methods are based on exact cell decompositions, visibility graphs and Voronoi diagrams. All these methods work from an *explicit* exact representation of the obstacles in the configuration space. They use sophisticated geometric data structures that give rise to fragile software.

During the same period, researchers in robotics proposed methods based on approximated representations of the obstacles in the configuration space (e.g., grid methods). The problem is solved up to some pre-defined resolution thresholds. Nevertheless such methods remain deterministic and do not overcome the intrinsic combinatorial complexity of the problem. Another interesting approach does not try to build obstacle models. It is based on local information giving the distance of the moving object to the obstacle in the 3D workspace. From this distance a repulsive potential is built. Obstacles then tend to push the moving object away while an attractive potential located at the goal to be reached attracts the object. Following the gradient gives rise to a collision-free path. Nevertheless the presence of local minima in the potential field makes the method fail in some cases.

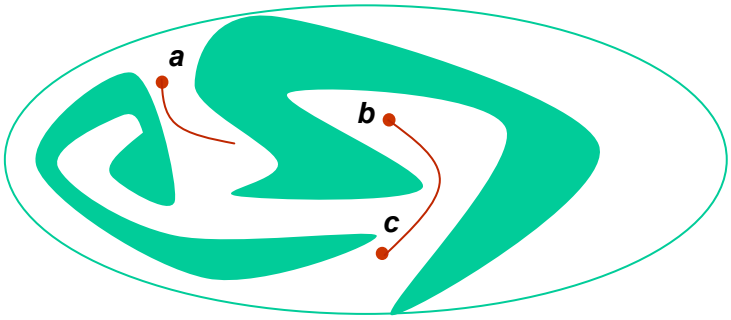



**Figure 7.** The gradient descent provides a collision-free path from point **c** to goal **b**. It fails when it starts at point **a**

At the end of the eighties, the problem is attacked from another point of view (mainly at Stanford University and Utrecht University). While the previous methods required explicit representations of the obstacles in the configuration space, the new methods explore the space on the basis of *implicit* obstacle representations. The completeness property of the algorithms is no more the main objective to be reached. Practical solutions are preferred. They take advantage from the increasing power of the computer. They succeed in solving problems that were out of the scope of the previous methods.

The innovative principle is based on random search. Simple geometric data structures (the so-called roadmaps) are built in the collision-free configuration space. Roadmaps are graphs whose nodes are collision-free configurations and whose edges denote the presence of a collision-free path between two configurations. Collision-free configurations are computed at random. The roadmaps tend to capture both the covering and the connectivity of the space. Several algorithm variants exist. Their formal analysis requires replacing the concept of deterministic completeness by the concept of probabilistic completeness: such algorithms are guaranteed to find a solution path –when it exists-- with a probability equal to one as soon as they run an infinite time. Nevertheless the probability of failure decreases as an exponential function of the computing time.

These new approaches represent a technological breakthrough. Their success has been demonstrated in several research labs around the world (Stanford University, Rice University, Texas A&M University, University of Illinois at Urbana Champaign, Utrecht University, Tokyo University, Singapore University, INRIA and CNRS in France…).

More than that, LAAS-CNRS have engaged several man-years of effort in the late 90's to devise a software platform implementing these approaches in a generic way. In 2000 CNRS transferred the exploitation rights of its technology to a spin-off company (Kineo CAM) that transformed the LAAS-CNRS software prototype into a marketed product which is now used in industry.

## 4 ROADMAP: AN EFFICIENT DATA STRUCTURE

Roadmaps are efficient combinatorial data structures to capture the topology of the collision-free configuration space, both in term of coverage and connectivity.

A roadmap is a finite graph. Nodes of a roadmap are collision-free configurations. Two nodes are adjacent if an elementary collision-free path computed with some given steering method can link both corresponding configurations.

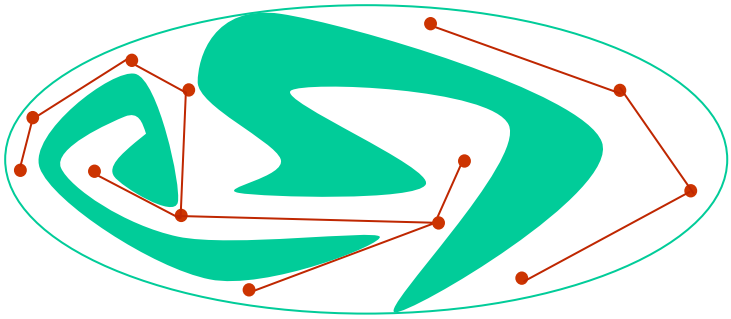


**Figure 8.** A roadmap is a graph tending to capture both coverage and connectivity of the collision-free path.

A roadmap being given, motion planning becomes an easy task. One first checks whether both the start and the goal configurations are "reachable" from nodes of the roadmap. This is made possible as soon as the roadmap covers the entire space. Start and goal are then added to the roadmap as new nodes. A collision-free motion exists between start and goal as soon as a path exists in the roadmap between the corresponding nodes, i.e. start and goal belong to same connected component of the roadmap.

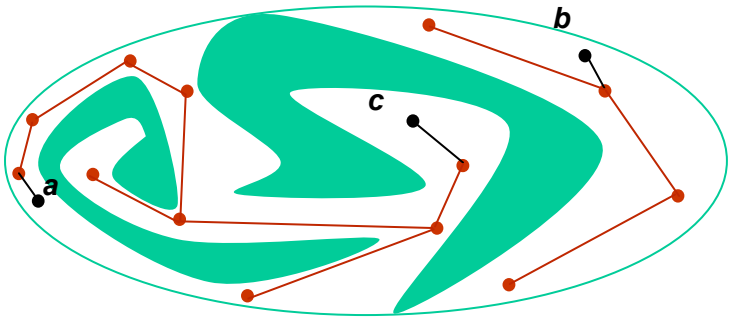



**Figure 9.** **a**, **b** and **c** are added as new nodes in the roadmap. Nodes **a** and **c** belong to a same connected component. There is a collision-free path between **a** and **c**. There is no collision-free path between **a** and **b** and between **c** and **b**

# 5 PROBABILISTIC ROADMAPS

How to compute roadmaps in an automatic way? This section sketches the probabilistic approaches which constituted the technological breakthrough of the 90's.

## 5.1 BASIC INGREDIENTS

The probabilistic roadmap algorithm principle consists in searching the collision-free configuration space without any explicit representation of the obstacles in the configuration space. Three operators are required:

- a collision-checker decides whether a given configuration is collision-free or not. For a given configuration, the device occupies some volume in the working space. This volume is made of geometric primitives. Interference detection is performed between these primitives and the obstacles.
- a steering method computes a direct path that respects the kinematics constraints of the considered system without considering the presence of obstacles, and
- a valid path checker decides whether a path is collision-free or not. It checks if the volume spanned by a device along a given direct path does not interfere with the obstacles. Such operator is built on the interference detection above.

Collision-checker and valid path checker are geometric operators that perform computations in the 3D space. These two operators take more than 90% of the time consumed by the algorithms. They are then critical.

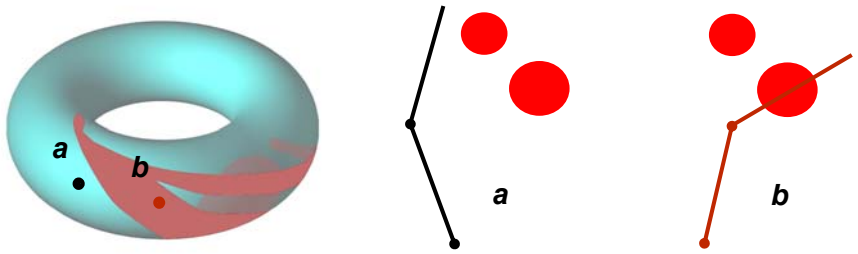



**Figure 10.** The torus is the configuration space of the 2 DOF robot manipulator. To check whether configurations **a** and **b** are collision-free or not, computations are performed in the workspace. The explicit knowledge of the obstacle on the torus (shape in red on the torus) is not necessary.

## 5.2 NOTE ON THE STEERING METHODS

Motion planning mainly deals with obstacle avoidance (and then collision-checking). Most of the time, computing a direct path between two given configurations is not a critical issue: a simple straight line segment in the configuration space usually corresponds to a feasible motion. This is the case of free-flying objects in the framework of mechanical disassembly. This is also the case of holonomic robot manipulators. Nevertheless some systems require paying attention on specific constraints affecting their admissible motion. For instance a car cannot move side-way and any path in the configuration space is not necessarily feasible. Another example is given by a rolling bridge where only one degree of freedom should be moved at the same time. Such constraints are kinematical constraints. That should be taken into account by the steering methods. The choice of the steering method affects the roadmap structure.

Let us consider a 2-dimensional configuration space together with two steering methods. The first one consists in computing straight line segments between two configurations. The second one consists in computing Manhattan paths (only one degree of freedom is moved at once). The picture below shows the accessible domains by paths of a given length for both methods respectively.

**Figure 11.** Direct paths define different topologies according to the considered steering method: in the case of straight line segments (on the left side) the set of points reachable by paths of fixed length is a circle, while it is a diamond in the case of Manhattan paths (on the right side)

Moreover, in the presence of obstacles the set points reachable by collision-free direct paths differ from a steering method to another.

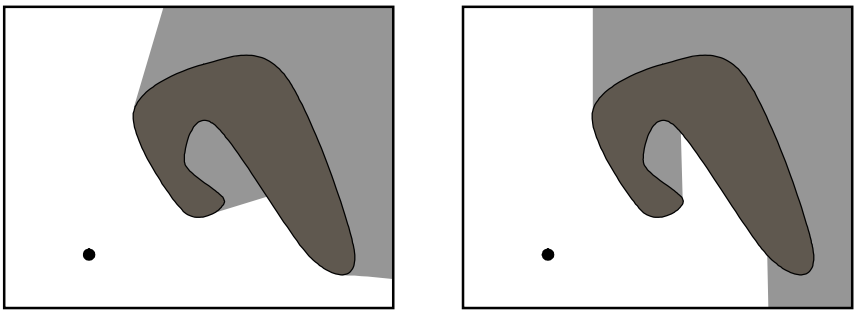


**Figure 12.** The set of points reachable by direct paths in the presence of an obstacle differs from a steering method to another one: straight line segments (left) versus Manhattan paths (right).

As a consequence, different steering methods give rise to different roadmaps in a same environment. The example below shows two roadmaps. The first one (on the left side) has been computed from a steering method using straight-line segments. The second one has been computed with a steering method that uses Manhattan paths.

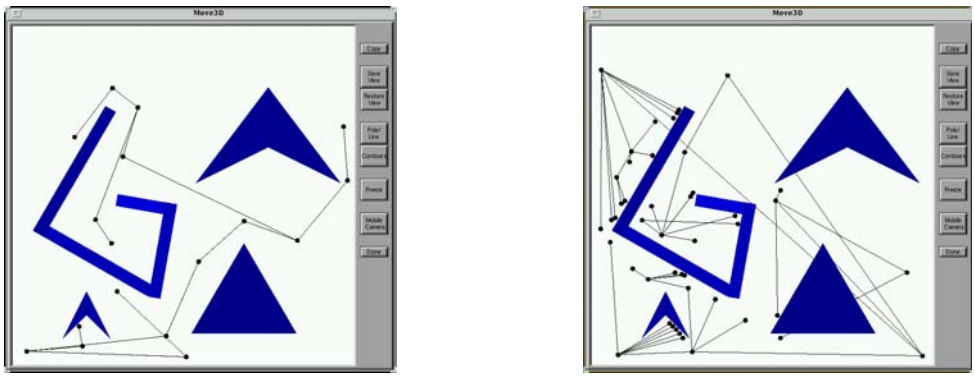


**Figure 13.**

### 5.3 ROADMAP COMPUTATION: SAMPLING AND DIFFUSION

Roadmap computation is done by two types of algorithms. The first one is called *Sampling* : it consists in sampling the collision-free space with configurations at random. Each new configuration is checked to be reachable from the current connected components of the roadmap. It is possible to integrate control parameters allowing to stop the sampling when the quality of the roadmap is good enough (in terms of coverage and connectivity).

The second one is called *Diffusion*: it consists in sampling the collision-free space with only few configurations at random –the roots- and then to diffuse the exploration in the neighbourhoods of the roots in directions chosen at random.

There are two main ways to solve a motion planning problem from probabilistic roadmaps:

- the first one starts with computing a roadmap that covers all the collision-free configuration space (this is the learning phase). Then the roadmap is used to solve a query: the starting configuration and the goal are added as new nodes of the roadmap; both nodes as checked to be reachable from the current connected components of the roadmap. Then a graph search is performed to check if there is path between the start and the goal. This is the learning-based strategy.
- the second one consists in building the roadmap by including both the start and goal configurations at the beginning. In that case the roadmap construction will stop as soon as a solution is found. This is the query-based strategy.

The first strategy is used when several problems of the same type should be solved. In that case the roadmap is stored. For any new instance of a motion planning problem involving the same device within a same environment, the pre-computed roadmap is loaded and the performance of the motion is drastically improved.

The second strategy is used for single query. In that case the performance of the planner is directly dependant from the difficulty of the problem to be solved.

Therefore the user can optimize the performance of the planner according to the context. If he is interested by studying all the accessible positions for a freight carried by a crane in a given environment, he will use the learning-based strategy. If he wants to study the decommissioning of a given part in an industrial installation, he will use the query-based strategy.

### 5.4 PATH SMOOTHING

We have seen that a solution of a motion planning problem by using roadmaps is a connected sequence of direct paths computed with a given steering method. This sequence necessarily goes through nodes of the roadmap.

In fact this sequence constitutes only a first step towards the final solution. Indeed to avoid useless detours via the fixed configurations of the roadmaps, the solution should be optimized.

A first strategy consists in trying to link directly two randomly chosen configurations on the path. This method is simple but may be very efficient. The user just indicates how many tries he wants to do.

The so-called *elastic band method* considers the path as an elastic band in the configuration space. The band is gradually tightened taking into account the non-collision constraint. The result is an optimized path passing round obstacles. To avoid collisions, the band is repulsed from obstacles beyond a given distance.

### 5.5 ANALYSIS: A GOOD ADEQUACY BETWEEN SOFTWARE AND HARDWARE

The theoretical analysis of the effective behaviour of the probabilistic algorithms above is not an easy task. In fact their performance is better noticed than well understood. Practical problems that were out of the scope of the deterministic methods are now solved by probabilistic approaches. Such algorithms did not appear in the 90's at random: the main reason of their success is mainly due to the adequacy between the computational power they require and the available hardware whose power significantly increases in the same period. The same algorithms would not have any interest 20 years before (i.e. computation time would have been too long).

## 6 APPLICATIONS

This section presents few examples of experience feedback in applying the technology on real study cases.

The first one deals with maintenance and operation in nuclear power stations. It has been addressed within the European Esprit project MOLOG (1999-2002) conducted by LAAS-CNRS, with Utrecht University as an academic partner, EDF (Electricité de France) as the end-user and Cadcentre as the provider of the software solution. Figure 14 shows the results of a successful simulation modelling the replacement of a large heat exchanger (350 tonnes, 20 meters high) within a nuclear power station. In this experiment the mechanical system is a polar bridge together with its load. A collision-free solution path with clearance of few centimetres has been *automatically* computed within couple of minutes, then saving several hours of tedious manual 3D interaction with the digital mock-up.

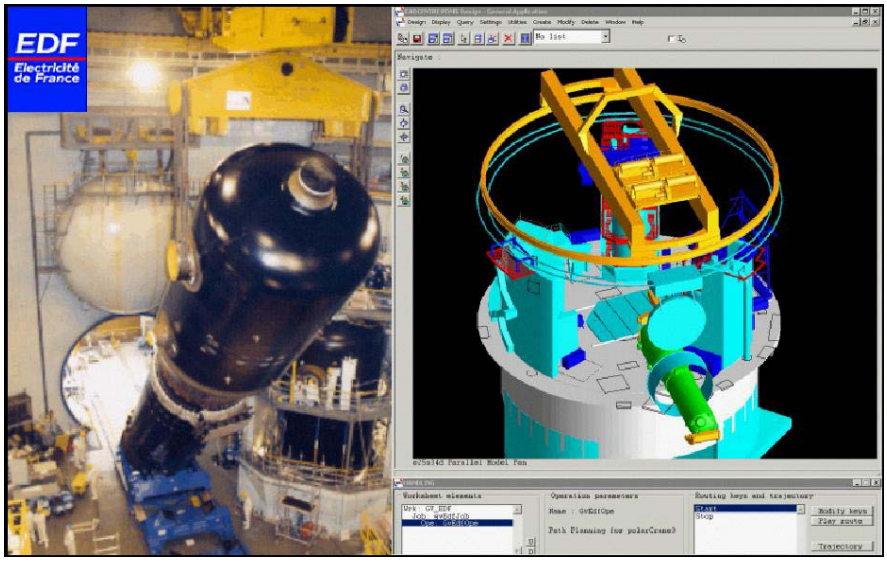



**Figure 14.** Simulation modelling the replacement of a large heat exchanger**.**

The second example comes from the automotive industry. It deals with assembly studies. The example has been submitted by Renault as a difficult problem. The operator never found any solution with the current available CAD tools to prove that the windscreen wiper can be assembled. In this example there is solution path with a clearance greater than 1 millimetre. The technology developed at LAAS-CNRS and marketed by Kineo CAM solved the problem: it *automatically* computed a *continuously safe path* within 90 seconds.

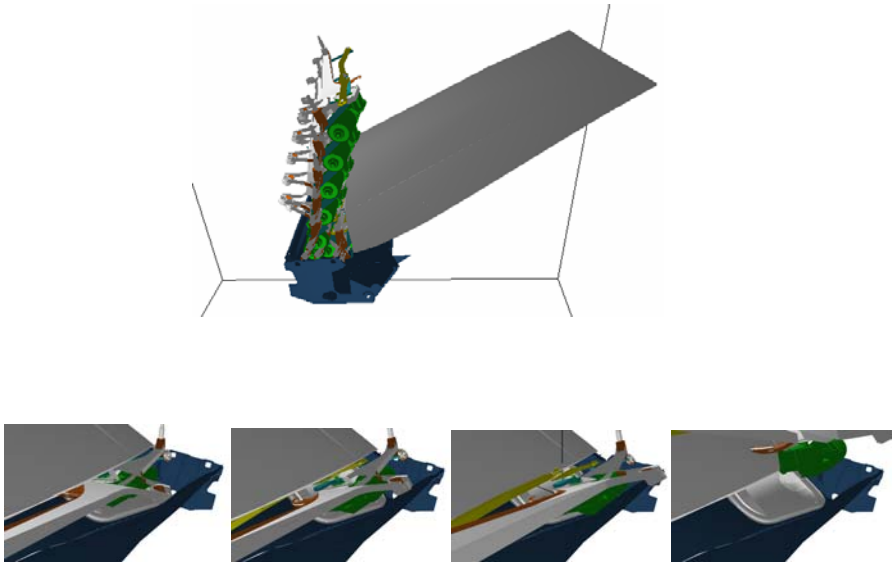


**Figure 15.**

The picture (Figure 1) at the beginning of this paper gives another example in Robotics. Here the main difficulty was to automatically find a solution path that necessarily requires a reconfiguration of the spot welding robot.

## 7 PERSPECTIVES: ADD VIRTUAL MANNEQUINS IN THE LOOP

Mannequins become today part of the digital mock-ups. An active research line tends to extend the previous methods to automatically compute humanlike motions for virtual mannequins evolving in digital factories. Here the goal is to produce and combine different behaviours such as locomotion or manipulation to automatically generate complex animations. The main challenge is to integrate bio-mechanical models of the human body to simulate the feasibility of complex tasks involving human operators. Figure 16 shows the very first results obtained at LAAS-CNRS. In this scenario the mannequin cooperates with a mobile robot to carry a large plate. The automated computation of such an animation involves to process very complex kinematic chains. The configuration space includes more than 60 degrees of freedom. The environment is made of 160000 polygons. The total time to compute the animation sequence is 15 seconds. The study automatically shows that the chosen itinerary necessarily requires the mannequin to bend down his head under the balcony. Such a bend is admissible in the sense that it respects the bio-mechanical constraints on the mannequin backbone.

**Figure 16.**

## 8 BIBLIOGRAPHICAL NOTES

The pioneering book "*Robot Motion Planning*" by J.C. Latombe (Kluwer Academic Press, 1991) is the main reference in algorithmic motion planning. It accounts for the state of the art of the research conduced during the eightieths. The mathematical analysis of the configuration space structure as well as dedicated solutions from Computational Geometry community appear in the book "*Planning, Geometry, and Complexity of Robot Motion*" (J.T. Schwartz, M. Sharir and J. Hopcroft, Eds), Ablex Series in Artificial Intelligence, Ablex Pub., 1987). A state of the art dealing with the research conducted in the ninetieths appears in "*Robot Motion Planning and Control*" (J.P. Laumond, Ed., Lecture Notes in Control and Information Science, Springer Verlag, 1998. The book is freely available at http://www.laas.fr/~jpl). Finally a journal paper by J.C. Latombe « Motion Planning: A Journey of Robots, Molecules, Digital Actors, and Other Artifacts » appeared in *International Journal of Robotics Research* , Special Issue on Robotics at the Millennium -- Part I, 18(11):1119-1128, November 1999.

## ACKNOWLEDGMENTS

This presentation benefits from an active research collaboration with my colleague T. Siméon. I thank E. Ferré (CTO, Kineo CAM) for many discussions and my current PhD students G. Arechavaleta and C. Esteves who are working on motion planning for virtual mannequins. This work is supported by the European project FP5 IST-39250 Movie. Another version of the current paper appears in the proceedings of the isiCAD conference held in Novosibirsk in 2004.