Motion Planning: A Journey of Robots, Molecules, Digital Actors, and Other Artifacts

Jean-Claude Latombe
Stanford University
Stanford, CA 94305, USA

July 1, 1999

Abstract

During the last three decades motion planning has emerged as a crucial and productive research area in robotics. In the mid-80's the most advanced planners were barely able to compute collision-free paths for objects crawling in planar workspaces. Today, planners efficiently deal with robots with many degrees of freedom in complex environments. Techniques also exist to generate quasi-optimal trajectories, coordinate multiple robots, deal with dynamic and kinematic constraints, and handle dynamic environments. This paper describes some of these achievements, presents new problems that have recently emerged, discusses applications likely to motivate future research, and finally gives expectations for the coming years. It stresses the fact that non-robotics applications (*e.g.*, graphic animation, surgical planning, computational biology) are growing in importance and are likely to shape future motion planning research more than robotics itself.

1 Introduction

During the last three decades motion planning has emerged as a crucial and productive research area in robotics [40]. The basic motion planning problem is to find a collision-free path for a robot – a rigid or articulated object – among rigid static obstacles. This is a purely geometric problem which looks deceptively simple; in fact, except for robots with few degrees of freedom (dofs), it is computationally hard. In the mid-80's the most advanced planners were barely able to compute collision-free paths for planar objects translating and rotating in two-dimensional workspaces. Impressive progress has been made since then and today's planners routinely solve complex practical problems, some involving robots with many dofs in complex environments. For instance, the planner in [59] is used by neurosurgeons to compute the motion of an articulated robot equipped with a linear accelerator – the Cyberknife – that performs radiosurgical operations of brain tumors. The planner in [30] successfully computed assembly plans for the guidance section of the Hughes AIM-9X air-to-air missile (an assembly made of 472 parts described by over 55Mb of CAD data). Motion planning has also found applications outside the realm of traditional robotics, such as design for manufacturing, graphic animation and video game software, minimally-invasive surgical planning, and molecule binding and folding. Problems arising from these domains are shaping motion planning research at least as much as robotics problems.

Several extensions of the basic problem have been studied, where, for instance, obstacles are moving, kinematic and dynamic constraints limit robot motions, optimized trajectories must be computed, or multiple robots have to be coordinated. Like the basic problem, some of these problems now have solutions that can be used in practice, but many still require more research. Changes in robotics technology, as well as new applications, will also raise new research problems. For instance, reconfigurable modular robots are now being developed, some with several hundred modules [14, 38, 64]. They will need new types of planners to compute their reconfiguring motions.

A reason for the success of motion planning research is that it is both narrow enough to make it easy to assess progress and deep enough that most results raise new issues motivating additional research. In the future, motion planning research will remain productive by both investigating some fundamental problems (*e.g.*, the interaction between complexity, controllability, and recognizability [41] or motion planning to achieve sensing goals [19]) and studying more specific problems arising from challenging applications. As the number of processors interacting with physical and/or virtual worlds (the so-called "embedded" processors) becomes much larger than that of desk-top computers, applications of motion planning are likely to grow in number, complexity, and diversity. For that to happen, our vision must broaden beyond traditional mobile robots and manipulator arms to encompass many other artifacts interacting with the physical world, ranging from airplanes and automobiles, to surgical tools, to molecules.

In this paper I first describe some important achievements of the last three decades (Section 2). Next I present computational issues and problems that have recently emerged and will grow in importance in the coming years (Section 3). Then I review a number of applications that are likely to motivate some of the research during the next decade (Section 4). In conclusion, I give some of my expectations for the future (Section 5). It is clearly impossible to present a complete panorama of motion planning within a few pages (even one that would be close to complete); by necessity, this paper only presents a limited number of viewpoints.

Throughout this paper the term *path planning* refers to the purely geometric problem of computing a collision-free path for a robot among static obstacles. The term *motion planning* is used for problems involving time, dynamic constraints, object coordination, sensory interaction, etc.

2 Achievements

2.1 Early Work

Motion planning became a topic of significant research during the 70's. As early as 1969, Nilsson [51] described a mobile robot system with motion planning capabilities. He introduced the visibility graph method (combined with the A^* search algorithm) to find the shortest collision-free path for a robot represented by a point amidst polygonal obstacles. This technique has remained enormously popular. The need for more sophisticated motion planners arised as researchers envisioned "task-level" robot programming systems for mechanical assembly [3, 45, 47, 58].

In 1977, Udupa introduced the idea of shrinking a robot to a point for collision avoidance [60]. In 1979, Lozano-Pérez and Wesley exploited this idea in a more general and systematic way and proposed a complete[1] path planner for polygonal/polyhedral robots moving in translation among polygonal/polyhedral obstacles [50]. This work led to the concept of configuration space.

---

[1] A *complete* (or *exact*) path planner is one which returns a collision-free path whenever one exists in the description input to the planner, and indicates that no such path exists otherwise.

2.2 Mathematical Foundations

In the early 80's Lozano-Pérez introduced the concept of a robot's *configuration space* [48], which impacted motion planning more than any other idea. The robot is represented as a point – called a configuration – in a parameter space encoding the robot's dofs – the configuration space. The obstacles in the workspace map as forbidden regions into the configuration space. The complement of these regions is the *free space*. Path planning for a dimensioned robot is thus "reduced" to the problem of planning a path for a point in a space that has as many dimensions as the robot has dofs.

During the 80's, classical tools from differential geometry were used to study the manifold structure of a configuration space, along with its more specific topological, geometric, and algebraic properties. Physical concepts, such as force and friction, were elegantly mapped into this representation. Most of these mathematical results are collected in [40]. They have played a crucial role in helping us understand motion planning problems, especially nonholonomic planning and optimal planning. The semi-algebraic structure of a robot configuration space, combined with techniques to solve decision problems in the first-order theory of the reals, led to the general-purpose path planning algorithms mentioned below.

2.3 Computational Analysis

In the 80's the path planning problem attracted interest from the theoretical computer science community. In 1979, Reif showed that path planning for a 3-D linkage made of polyhedral links is PSPACE-hard [53]. The proof uses the robot's dofs to encode the configuration of a polynomial space bounded Turing machine and design obstacles which force the robot's motions to simulate the computation of this machine. This analysis provides evidence that any complete planner will run in exponential time in the number of dofs.

In 1983, Schwartz and Sharir proposed a complete general-purpose path planning algorithm based on an algebraic decomposition of the robot's configuration space known as the Collins decomposition [54]. This algorithm takes time doubly exponential in the number of dofs. A few years later, Canny developed a singly-exponential algorithm that computes a representation of the robot's free space in the form of a network of 1-D curves [13]. None of these algorithms have been implemented, but they helped calibrating the complexity of path planning and understanding its combinatorial nature.

Complete specific algorithms have also been developed mainly for robots with 2 or 3 dofs. For a k-sided polygonal robot moving freely in a polygonal workspace, the algorithm in [24] takes $O((kn)^{2+\epsilon})$ time, where n is the total number of edges of the workspace. Some of these more specific algorithms have been implemented. But they are usually not robust to floating-point approximations and they have not been used in practice so far.

2.4 Practical Planners

The prohibitive complexity of complete path planners and/or their lack of robustness have motivated the development of heuristic planners [23]. Two popular approaches were introduced in the 80's: approximate cell decomposition, where the free space is represented by a collection of simple cells [12], and potential field [35]. The later was initially proposed for on-line collision avoidance, but it can easily be combined with grid searching techniques to solve planning problems [5].

Both approaches are resolution-complete, if properly implemented: whenever a path exists, they find one if the resolution parameter (the size of the smallest cells or the resolution of the grid) is set fine enough. They can solve complex path planning problems in 2-D and 3-D configuration spaces in a fraction of a second. For example, without any precomputation, the planner in [39] generates navigation paths for virtual characters at the video frame rate (1/30s) in dynamically changing workspaces.

But none of these approaches extends well to robots with more than 4 or 5 dofs. Then either the number of cells becomes too large; or the potential field has local minima. For many-dof robots, several heuristic techniques have been proposed that offer no formal guarantee of performance. The resulting planners have not been reliable (*e.g.*, they often fail to solve seemingly simple problems), and none of them has had significant impact.

In 1991, a randomized planner was introduced [5], which was able to solve complex path planning problems for many-dof robots by alternating "down motions" to track the negated gradient of a potential field and "random motions" to escape local minima. To avoid pathological cases caused by the deterministic potential field, another type of randomized planner was later developed [34], which consists of sampling the configuration space at random and connecting the samples in free space by "local" paths (typically straight paths), thus creating a probabilistic roadmap (PRM). Samples and local paths are checked for collision using a fast collision checker (*e.g.*, [52]), which avoids the prohibitive computation of an explicit representation of the free space. Experiments with PRM planners have been quite successful, showing that they are both fast and reliable even with robots with many dofs. Formal analysis supports this experimental observation by showing that PRM planning is complete in a probabilistic sense. Under reasonable geometric assumptions on the free space (see Subsection 3.2), the probability that a PRM planner fails to find a path while one exists decreases exponentially toward 0 with the number of samples [27]. Interestingly, this analysis relates the number of samples needed to the geometric "goodness" of the free space, rather than to its combinatorial complexity.

PRM planners are also robust to floating-point approximations and quite easy to implement. A number of variants applying different sampling strategies have been recently developed [2, 27, 29, 32, 43]. Somes of these variants solve motion planning problems involving nonholonomic and dynamic constraints, optimization criteria, moving obstacles, and/or flexible robots.

2.5 Other Results

Other motion planning problems have been investigated, some with mixed practical results. We mention a few below.

Nonholonomic planning In the 90's, a fertile research area has been path planning for nonholonomic robots, mainly car-like robots and multi-body tractor-trailer robots with lower-bounded turning radii. Concepts from differential geometry and control theory have been used to study the controllability of these robots and produce effective planning techniques [4, 6]. A nonholonomic robot is said to be locally controllable if, at every configuration, the Control Lie Algebra associated with the valid controls has the same dimension as the robot's configuration space. If a nonholonomic robot R_n is controllable, then any free path of the holonomic robot R_h having the same geometry as R_n can be locally deformed into a free path satisfying the kinematic constraints of R_n. Several practical planners have been developed for locally controllable nonholonomic robots with few dofs, based on the discretized exploration of the free space [6], on the local deformation of a previously computed holonomic path [42], or on PRM-type planning [56]. No practical planner, however, currently exists to compute paths for many-dof nonholonomic robots; the technique in [6] is computationally too costly, while the techniques in [42, 56] lack the "local paths" that are needed for deforming a holonomic path or for connecting the samples of a roadmap. Path planning for robots that are not locally controllable (*e.g.*, a car-like robot that can only move forward) remains an open problem.

Part orientation/positioning An application area that has recently motivated considerable research is part feeding. An algorithm that plans the tilting motions of a tray containing a planar part of known shape to orient it to a desired orientation is presented in [17]. The influential algorithm in [18] computes a sequence of squeezes by a frictionless, sensorless, parallel-jaw gripper to achieve a single orientation of a polygonal part (up to symmetries of the part's convex hull). Programmable vector fields created by arrays of tiny actuators, such as MEMS, have also been proposed to orient and position planar parts [7]. A plan then consists of applying a sequence of vector fields. But it was shown recently that, for most planar parts, a single, carefully crafted vector field suffices to achieve a unique position and orientation of a part of given geometry [8], hence eliminating the need for any planner.

Assembly sequencing Here the goal is to compute both an order in which the components of a product can be assembled, and the corresponding movements of the parts. Early research produced techniques to compute the cone of possible motions of a part in contact with other parts and proposed assembly sequencers which use a trial-and-error approach to compute how a product can be disassembled [26]. However, the inherent combinatorial complexity of trial-and-error restricts its applicability to very simple products and disassembly sequences where a single part is removed at each step. The non-directional blocking graph (NDBG) was proposed to avoid this combinatorial trap [63]. It is a subdivision of the space of allowable motions (*e.g.*, translations) into a finite number of cells such that within each cell the set of blocking relations between all pairs of parts remains fixed. The NDBG is precomputed and then queried to generate assembly sequences. This approach was effectively implemented for several families of motions, including the family of extended rigid-body motions which spans a 5-D sphere in 6-D space (each motion is the composition of a translation along a fixed vector and a rotation around another fixed vector) [21]. The Archimedes system [30] based on the NDBG approach was successfully applied to several products, including the 472-part guidance section of the Hughes AIM-9X air-to-air missile. Nevetheless, as all exact cell decomposition methods, the NDBG approach is sensitive to floating-point approximations which complicate its implementation. Its complexity also grows exponentially with the dimension of the space of allowed motions.

Uncertainty Early in the 70's, motion planning with uncertainty in control and sensing attracted considerable attention, in particular to compute fine motions for delicate assembly operations [47, 58]. Many researchers regarded this problem as one of the most interesting problems in motion planning. An elegant framework – preimage backchaining [49] – was proposed along with reasonably efficient algorithms in low-dimensional spaces [10, 44]. These methods have been applied to mobile robot navigation problems, but their practical impact so far remains limited. Today, most experimental robotics systems deal with uncertainty at execution time (*e.g.*, by sensing whatever environmental features turn visible to maintain a best estimate of the robot location) rather than at planning time. One reason is that planners use simplified models of uncertainty that do not represent reality well; but, on the other hand, using more realistic models seems to prohibitively increase the cost of planning. In other words, what is gained by taking uncertainty into account in a planner is outweighed by the increased complexity and the reduced flexibility of the planner. The choice of an appropriate level of abstraction for representing uncertainty in motion planning still remains an open question.

3 New Issues and Problems

3.1 Enabling Tools and Technology

As motion planning matures, more applications require dealing with complex geometric models. For instance, in virtual prototyping and manufacturing, path planners are already applied to CAD models of robots and obstacles that are made of several 100,000 triangles. Soon millions of triangles, or more, will have to be considered. This trend will require paying more attention to some basic tools than has been done in the past.

Distance computation PRM planners spend 90% or their time, or more, in checking collision or computing distances between robots and obstacles. In most applications, the running time of the planner now depends less on the (fixed) number of dofs of the moving objects than on the geometric complexity of the robots and obstacles. Efficient distance computation algorithms have recently been developed (*e.g.*, [52, 62]), but more needs to be done. Like for many basic operations in graphics, it is likely that this progress will eventually come from hardware implementations.

Robust computation Many planning algorithms in low-dimensional spaces (*e.g.*, assembly sequencing) are based on decomposing that space into cells, such that in each cell a set of certificates (*e.g.*, blocking relations) remain true or false. Cell boundaries are critical (hyper-)surfaces where some certificates may change. Such planning algorithms are very sensitive to floating-point errors. Efficient and robust tools will have to be developed before these algorithms can be widely used in practice.[2]

Dynamic data structures Dealing with many moving objects in complex environments or with complex deformable objects will require appropriate dynamic data structures that can be efficiently updated, for instance, by exploiting the temporal and spatial coherence of the physical world [46]. The kinetic data structure (KDS) in [20] is a significant step in this direction. As objects move, a KDS dynamically maintains a set of assertions that guarantee the correctness of an easy computation for an attribute of interest (*e.g.*, the convex hull of moving objects, the closest pair of objects, etc.). The events that require updating the KDS are the failures of these assertions. It is reasonable to expect that specific hardware will become available in the future to quickly predict and schedule such events.

3.2 Issues with Randomized Planners

Despite their empirical success, randomized planners still raise open issues. Investigating these issues will be critical to make PRM planners even more reliable and faster, and to facilitate their application to harder motion planning problems. *e.g.*, problems requiring the coordination of many moving objects.

Theoretical analysis shows that PRM planners work well in free spaces that satisfy two simple visibility properties, ϵ-goodness and expansiveness. A robot's free space F is ϵ-good (for some $\epsilon \in (0,1]$) if every configuration in F "sees" a subset of F whose total volume is at least an ϵ fraction of F's total volume (two configurations see each other if they can be connected in F by a "local" planner) [33]. Loosely speaking, F is expansive if it does not contain "narrow passages" [27]. Under those two conditions, the probability that a PRM planner fails to find a path while one exists decreases exponentially toward 0 with the number of samples picked in F [27].

This analysis and experiments show that the main outstanding issue with PRM planners is the "narrow passage" problem [28, 61]: if the free space contains narrow passages, the planner must pick a

---

[2] See the CGAL project at `http://www.cs.uu.nl/CGAL/` for a significant research effort in this direction.

prohibitively large number of samples over the entire free space so that enough samples fall in such passages to establish connections through them. When narrow passages can be easily inferred from the workspace geometry, potential field techniques [5] can be used to bias the sampling of the free space. Unfortunately, the robot dofs, the mechanical bounds for each dof, and the workspace obstacles may create narrow passages in free space that cannot be easily derived from the workspace. In such cases, it is proposed in [28] to accept samples that are not in free space, but for which the penetration distance of the robot into the obstacles is small. The effect is to widen the narrow passages, which makes it easier to find them. But this technique has serious drawbacks, including the fact that no general and sufficiently efficient algorithm is currently available to compute penetration distances.

Even in the absence of narrow passages, PRM planning raises another important issue: as long as a planner does not return a path, there is no way to know whether a path exists, or not. The analysis in [27, 33] establishes a relation between the number of samples and the probability of not finding a path, but this relation involves parameters (such as the ϵ of the ϵ-goodness property) that are not known in advance. Evaluating such parameters is likely to be as hard as path planning itself. Perhaps more usable analytical results can be established.

3.3 Physical Models

The most important developments in motion planning are likely to come from using more comprehensive models of the real world, including physical and uncertainty models. Indeed, path planning problems are purely geometric, a fact that considerably limits the applicability of the corresponding algorithms.

Path planning algorithms may be extended to take into account physical models. For instance, once a collision-free geometric path has been computed, this path may be deformed to optimize a criterion (*e.g.*, minimize execution time) under dynamic constraints [29]. PRM planning techniques have also been used to generate quasi-optimal motions (kinodynamic problem) directly, by growing a tree of sample states from the initial configuration of the robot [43]. Each iteration consists of picking a state in the tree and a control at random; integrating the control equations over a short time interval yields a new state if no collision is detected. PRM planning has also been used to generate collision-free motions for deformable objects among rigid obstacles [32]. An energy is associated with each possible deformation and configurations with low energy are prefered over configurations with high energy.

More complex physical models will be needed. For instance, in surgery (*e.g.*, endoscopic operations), it is desirable to precompute minimally invasive paths of surgical tools among soft tissue structures having different elastic properties. In many tasks (*e.g.*, mechanical assembly, medical surgery), motions must be computed for objects sliding in contact with one another, which requires dealing with friction models. Eventually, the complexity of the models may lead to developing new types of planning techniques.

3.4 New Forms of Planning

Most motion planning problems considered so far require one or several moving objects to reach a given final configuration. New forms of planning will be considered in the future. We review two of them below, for which partial results have already been obtained.

Manipulation planning Here, a number of movable objects must reach goal configurations. But none of them is able to move alone. Instead, they must be grasped and displaced by robots. In general, neither the order in which the movable objects must be displaced, nor the robots that should grasp them are specified as inputs. So, the problem is a composite planning problem where one must plan both the motions of the movable objects and those of the robots that will move them. Potentially, this can be very difficult. In particular, avoiding collision among robots, obstacles, and movable objects may require the movable objects to be displaced in some very specific ordering; how can such an ordering be efficiently computed? Several other issues must also be considered, like grasp planning and ensuring that every intermediate arrangement of objects is stable. Stand-alone grasp planning techniques exist, but only certain grasps may allow the robot to carry the movable objects to their goals; how can the manipulation planner efficiently deal with the interaction between grasps and paths? A few manipulation planners have been proposed [1, 37]. But they only address a limited number of issues, and none is widely applicable. Several application areas, such as virtual prototyping to help designers create products that are easy to assemble and service, will motivate the development of more useful manipulation planners.

Planning for sensing Sensors can be used to acquire information about an environment, like building a 3-D model or localizing objects of interest. Active sensing is aimed at moving sensors appropriately to acquire the needed information as quickly and reliably as possible [19]. In this context, a problem which has recently attracted interest is to plan the motion of a mobile robot (or a team of robots) equipped with visual sensors to eventually find a moving intruder hiding in a given environment. Here, the state of the robot at any one time can no longer be reduced to a mere configuration or even a (configuration,velocity) pair; it must also contain information representing what the robot knows. In the intruder example, we can associate a visibility region with each configuration of the robot. In a polygonal environment, this region is a polygon bounded by both obstacle edges and free edges. The information state defines the free edges behind which the intruder may still hide [22]. The planner must compute a trajectory of the robot leading to an information state such that the intruder cannot be hiding behind any of the free edges of the current visibility region. Executing that path guarantees that the intruder will eventually be detected.

3.5 Integration

Most motion planners must eventually interact with other modules in an integrated system. For example, a motion planner for a robot must interact with a controller, which transforms plans into low-level motor commands, and with sensor modules, which localize the robot in its environment. Clearly, integration means much more than mere juxtaposition. For instance, a planner may take dynamic constraints into account and compute motion plans that are easier to execute; or, instead, the controller may include reactive collision avoidance to reduce the computational work of the planner.

There has been significant work on integrating motion planning with control and sensing. This type of integration remains nevertheless rather poorly understood. Some interesting techniques, like the elastic strip [11] or landmark techniques [57], have been proposed, but few widely accepted architectural principles have emerged. As more complex systems are built, we can expect significant progress in this area.

Integration will also require that we develop a better understanding of the limits of what planning can, and cannot do. Consider facial animation for virtual characters, an important topic in computer graphics. One problem is to make a face say a given sentence by coordinating lip motions and sound. Would it make sense to develop a planner that computes the motions of the lips from the input sentence? Very likely not. A system like the one described in [9] solves the problem more efficiently by associating pnonemes to video captured motions and applying morphing techniques to combine these recorded mouth gestures. Similarly, it would probably not be very effective to plan the motion of a humanoid robot to sit on a chair; the motion algorithm could instead be crafted by hand and recorded in a library of specific motions. Motion planning should be reserved for motions that are so diverse that they cannot be anticipated and stored in databases of clipped motions.

4 Applications

Robotics So far, motion planners have not been widely used in industrial robot systems. Most industrial robots perform repetitive tasks and their motions are programmed by hand (*e.g.*, using a teach pendant). However, the situation is changing. Several CAD-based robot programming systems now include motion planners, which may be used to automatically compute robot paths. For some applications, this facility is becoming critical. For example, consider spot welding operations on car frames. Most welding workstations in a body shop contain multiple robots operating concurrently, each performing a sequence of several welding operations. Path coordination is particularly difficult to craft by hand. Automatic path planning greatly simplifies the task of programming the robots. It also makes it possible to compute the optimal location of each robot to minimize the cycle time [29]. Moreover, if test crashes suggest modifications of the welding points after the robots have been programmed, the planner can update the robot trajectories. This kind of application will motivate the development of planners capable of consistently producing quasi-optimal motions in geometrically complex environments.

Obviously, as robots make inroads in other, less structured environments (military, surveillance, construction, space), we can expect the need for on-line, efficient planners to grow.

Virtual Prototyping It is becoming common practice to design complex products using solely CAD systems. The systems that will manufacture and service these products tend to be also designed using CAD systems.

For years CAD systems have been sophisticated, but passive electronic drafting tables equipped with 3-D visualization capabilities. Now, CAD systems evolve to incorporate more active tools, *e.g.*, for checking interferences among parts, computing and simulating motions, or analyzing manufacturability and serviceability. An important function of these tools is to provide feedback to the designers regarding their products and help them make decisions about how objects move or are moved during manufacturing, maintenance, and servicing, as well as during regular operations by the product users.

Virtual prototyping will motivate the development of planners able to compute difficult paths quickly and reliably in complex geometric environments. Some motion planners are already applied in this context. For example, the randomized planner in [15] verifies that parts of an aircraft engine can be removed for inspection. The nonholonomic planner in [25] checks accessibility of a space by a wheelchair in the CAD model of a building.

Graphic Animation During the last few years, a number of robotics techniques have made significant inroads in graphic systems, *e.g.*, dynamic modelling to create physically realistic animation, vision sensing to automatically acquire 3-D models, haptic interaction to "feel" virtual objects, and motion planning to create collision-free motions. Today, the development of video games on the Internet, the development of interactive training systems, the increasing sophistication of websites, and the computer-aided production of movies raise the need for new tools that will facilitate the creation and animation of autonomous virtual characters (or digital actors) in 3-D worlds. Motion planning techniques will be used to direct digital actors at the task-level and to create highly interactive systems.[3] This domain will motivate the creation of fast planners capable of using physically-based models to generate realistic-looking motions. It will require planned motions to be seamlessly combined with motions extracted from large libraries of clipped motions. See [36, 39] for preliminary results in motion planning applied to the animation of digital actors.

Medical Surgery Imaging techniques are now widely available to produce detailed and precise computer representations of 3-D tissue structures. Motion planning will increasingly be used in surgical planning to compute the motion of the surgical tools. For instance, the planner in [59] is now in clinical use by neurosurgeons to compute the successive positions and orientations of a radiation beam to destroy brain tumors without damaging healthy tissues surrounding them, especially highly sensitive and/or critical structures like the optic nerves; the radiation beam is produced by a linear accelerator that is displaced by a 6-dof manipulator robot.

Several researchers are currently working on the creation of realistic models of human soft tissues in order to compute their deformations under external forces, such as those applied by surgical tools when they probe, grasp, pinch, cut, pierce, and suture the tissues [16]. This research will enable motion planning techniques to compute minimally-invasive paths for surgical tools (*e.g.*, endoscopes, scalpels).

Computational Biology Computational biology raises many problems where it is necessary to reason about molecule motions. For example, a drug molecule, called a ligand, is usually a small (20 to 50 atoms), but flexible (5 to 15 torsional dofs) molecule, which acts by binding into the pocket of a large and mostly rigid molecule, called the receptor. The ligand deforms and moves under the influence of forces internal to the ligand and forces caused by the interaction between the ligand and the receptor. Drug design aims at creating a synthesizable ligand that can move and deform to a low-energy configuration, called a conformation, that exhibits geometric and chemical complementarity with the desired binding pocket [31]. A PRM-type planner is described in [55] that verifies the existence of a low-energy path for a given ligand to a binding pocket of a protein. This planner uses the energy field in the configuration space of the ligand to bias the sampling toward low-energy configurations near the receptor's outer surface. Eventually, such a planner could be used to scan large databases of ligands that are publicly available or owned by pharmaceutical companies to extract those ligands which are the most likely to bind at a desired binding site.

5 Expectations

Today path planning for few-dof robots is quasi-instantaneous, even in large and complex environments encountered in practical problems. Within a few years, this will also be true for many-dof robots (including animated figures, such as virtual characters modeled with several dozen dofs, or more) using randomized planning. This trend will encompass more complex planning problems, like optimal planning with kinematic and dynamic constraints, and planning with deformable robots and obstacles, to which PRM-type planning has been shown to extend well.

As several forms of motion planning turn "real-time," the role of planning in integrated systems will change from a costly resource that should be rarely invoked, to a cheap commodity that can be called at will. For example, the planner in [39] is invoked to plan the navigation path of a virtual character whenever this character senses a change in its environment; because the planner is fast enough, there

---

[3] See `http://www.motion-factory.com/` for a commercial system.

is no attempt whatsoever to reuse the previously computed path. Moreover, the same system can easily handle multiple independent characters, without any noticeable slowdown.

Randomized planning will make motion planning less dependent on the number of dofs than on the geometric complexity of the environment. However, progress in enabling technologies, like specific hardware to perform distance computation operations and to maintain dynamic data structures, should make it possible to handle robots and environments described by hundreds of millions of polygons.

Motion planning applications will grow considerably. CAD-based robot programming systems will include planners to compute quasi-optimal motions of robots and to optimize robot layouts. CAD systems will include planners to verify that products or building facilities can be easily manufactured or built, and serviced. In video games, character motions will be (re-)computed on the fly to adapt to user inputs, allowing new forms of games to be created. More generally, video games, movies, animated webpages, and interfaces will converge, with motion planning and motion capture used jointly to produce rich interactive animation of virtual characters. Motion planners will help surgeons plan for minimally-invasive operations. Software packages to assist biochemists in the discovery of new drugs will include motion planners to compute plausible motions of ligands binding against proteins.

Simultaneously, new motion planning problems will be investigated. For instance, reconfigurable robots made up of thousands of modules will require planners to compute their reconfiguration motions [14]; today this problem is mostly open. Robots equipped with sensors to collect information will need planners to decide which are the most efficient motions to obtain pertinent information. Minimizing surgical invasiveness will require planners that reason about the deformations of soft tissues caused by the motions of surgical tools. It is almost certain that the basic path planning problem, which has been the focus of the research in motion planning for more than two decades, will soon loose this status. No other problem, however, seems basic enough to play the same role in the future.

Acknowledgements: The author thanks Lydia Kavraki for her numerous comments on this paper. He also thanks Hector Gonzalez-Baños, David Hsu, James Kuffner, and Steve LaValle who have contributed toward several ideas mentioned in this paper. The author's research is funded by NSF grants IIS-9619625 and IIS-97-11380, ARO MURI grant DAAH04-96-1-007, DARPA/Army Contract DAAE07-98-L027, and gifts from General Motors, Honda R&D Americas, Intel, SUN Microsystems, and Microsoft.

References

[1] Alami, R., Laumond, J.P., and Siméon, T. 1995. Two Manipulation Planning Algorithms. *Algorithmic Foundations of Robotics*, WAFR'94, Goldberg, K. et al. (eds.), A K Peters, Natick, MA, 109-125.

[2] Amato, N.M., Bayazit, O.B., Dale, L.K., Jones, C., and Vallejo, D. 1998. Choosing Good Distance Metrics and Local Planners for Probabilistic Roadmap Methods. *Proc. IEEE Int. Conf. on Robotics and Automation.*

[3] Ambler, A.P. and Popplestone, R.J. 1975. Inferring the Positions of Bodies from Specified Spatial Relationships. *Artificial Intelligence*, 6(2):157-174.

[4] Barraquand, J. and Latombe, J.C. 1989. On Nonholonomic Mobile Robots and Optimal Maneuvering. *Revue d'Intelligence Artificielle*, 3(2):77-103.

[5] Barraquand, J. and Latombe, J.C. 1991. Robot Motion Planning: A Distributed Representation Approach. *Int. J. of Rob. Res.*, 10(6):628-649.

[6] Barraquand, J. and Latombe, J.C. 1993. Nonholonomic Multibody Mobile Robots: Controllability and Motion Planning in the Presence of Obstacles. *Algorithmica*, 10(2-3-4):121-155.

[7] Böhringer, K.F., Donald, B.R., and MacDonald, N.C. 1996. What Programmable Vector Fields Can (and Cannot) Do: Force Field Algorithms for MEMS and Vibratory Parts Feeders. *Proc. IEEE Int. Conf. on Robotics and Automation*, 822-930.

[8] Böhringer, K.F., Donald, B.R., Kavraki, L.E., and Lamiraux, F. 1999. Part Orientation with One or Two Stable Equilibria Using Programmable Vector Fields. *Manuscript.*

[9] Bregler, C., Covell, M., and Slaney, M. 1997. Video Rewrite: Driving Visual Speech with Audio. *Proc. SIGGRAPH'97.*

[10] Briggs, A.J. 1989. An Efficient Algorithm for One-Step Planar Compliant Motion Planning with Uncertainty, *Proc. of the 5th Annual Symp. on Computational Geometry*, Saarbrücken, Germany.

[11] Brock, O. and Khatib, O. 1999. Elastic Strips: A Framework for Integrated Planning and Execution. *Proc. Int. Symp. on Experimental Robotics*, to appear.

[12] Brooks, R. and Lozano-Pérez, T. 1983. A Subdvision Algorithm in Configuration Space for Findpath with Rotation. *Proc. 8th Int. Joint Conf. on Artificial Intelligence (ICAI)*, 799-806.

[13] Canny, J.F. 1988. *The Complexity of Robot Motion Planning.* MIT Press, Cambridge, MA.

[14] Casal, A. and Yim, M. 1999. Self-Reconfiguration Planning for a Class of Modular Robots. *Proc. SPIE Symp. on Intelligent Systems and Advanced Manufacturing.*

[15] Chang, H. and Li, T.Y. 1995. Assembly Maintainability Study with Motion Planning, *Proc. IEEE Int. Conf. on Robotics and Automation*, 1012-1019.

[16] Delingette, H. 1998. *Towards Realistic Soft Tissue Modeling in Medical Simulation.* Rapport de Recherche No. 3506, INRIA, Sophia-Antipolis, France.

[17] Erdmann, M. and Mason, M. 1988. An Exploration of Sensorless Manipulation, *IEEE Tr. on Robotics and Automation*, 4(4):369-379.

[18] Goldberg, K.Y. 1993. Orienting Polygonal Parts Without Sensors. *Algorithmica*, 10:201-225.

[19] Gonzalez-Baños, H.H., Guibas, L.J., Latombe, J.C., LaValle, S.M., Lin, D., Motwani, R., and Tomasi, C. 1998. Motion Planning with Visibility Constraints: Building Autonomous Observers. *Robotics Research - The Eighth Int. Symp.*, Shirai. Y. and Hirose, S. (eds.) Springer, New York, 95-101.

[20] Guibas, L.J. 1998. Kinetic Data Structures: A State of the Art Report. *Robotics: The Algorithmic Perspective*, 1998 WAFR, Agarwal, P.K., Kavraki, L.E., and Mason, M.T. (eds), A K Peters, Natick, MA, 191-209.

[21] Guibas, L.J., Halperin, D., Hirukawa, H. Latombe, J.C., and Wilson, R.H. 1998. Polyhedral Assembly Partitioning Using Maximally Covered Cells in Arrangements of Convex Polygons. *Int. J. of Computational Geometry and its Application*, 8(2):179-199.

[22] Guibas, L.J., Latombe, J.C., LaValle, S.M., Lin, D., and Motwani. , R. 1997. Visibility-Based Pursuit-Evasion in a Polygonal Environment. *Proc. 5th Int. Workshop on Algorithms and Data Structures (WADS'97)*, Dehne, F., Rau-Chaplin, A., Sack, J.R., and Tamassia, R. (eds.), Lecture Notes in Computer Science, Vol. 1272, 17-30.

[23] Gupta, K. and del Pobil, A. (eds.). 1998. *Practical Motion Planning in Robotics: Current Approaches and Future Directions*, John Wiley, West Sussex, England.

[24] Halperin D. and Sharir, M. 1996. A Near-Quadratic Algorithm for Planning the Motion of a Polygon in a Polygonal Environment. *Discrete Comput. Geom.*, 16:121-134.

[25] Han, C.S. 1999. Building Design Services in a Distributed Service Architecture. `http://www-galerkin.stanford.edu/~csh/research.html`.

[26] Homem de Mello, L.S. and Sanderson, A.C. 1991. A Correct and Complete Algorithm for the Generation of Mechanical Assembly Sequences. *IEEE Tr. on Robotics and Automation*, 7(2):228-240.

[27] Hsu, D., Latombe, J.C., and Motwani, R. 1997. Path Planning in Expansive Configuration Spaces. *Proc. IEEE Int. Conf. on Robotics and Autamtion*, Albuquerque, NM, 2719-2726.

[28] Hsu, D., Kavraki, L.E, Latombe, J.C., Motwani, R., and Sorkin, S. 1998. On Finding Narrow Passages with Probabilistic Roadmap Planners. *Robotics: The Algorithmic Perspective*, 1998 WAFR, Agarwal, P.K., Kavraki, L.E., and Mason, M.T. (eds), A K Peters, Natick, MA, 141-153.

[29] Hsu, D., Latombe, J.C., and Sorkin, S. 1999. Placing a Robot Manipulator Amid Obstacles for Optimized Execution. *Proc. IEEE Int. Symp. on Assembly and Task Planning (ISATP'99)*, to appear.

[30] Jones, R.E., Wilson, R.H., and Calton, T.L. 1997. Constraint-Based Interactive Assembly Planning. *Proc. IEEE Int. Conf. on Robotics and Automation*, 913-920.

[31] Kavraki, L.E. 1997. Geometry and the Discovery of New Ligands. *Algorithms for Robotic Motion and Manipulation*, 1996 WAFR, Laumond, J.P. and Overmars, M. (eds.), A K Peters, Natick, MA, 435-448.

[32] Kavraki, L.E., Lamiraux, F., and Holleman, C. 1998. Towards Planning for Elastic Objects. *Robotics: The Algorithmic Perspective*, 1998 WAFR, Agarwal, P.K., Kavraki, L.E., and Mason, M.T. (eds), A K Peters, Natick, MA, 313-326.

[33] Kavraki, L.E., Latombe, J.C., Motwani, R., and Raghavan, P. 1995. Randomized Query Processing in Robot Motion Planning. *Proc. ACM SIGACT Symp. on Theory of Computing (STOC)*, Las Vegas, NV, 353-362.

[34] Kavraki, L.E., Švestka, P., Latombe, J.C., and Overmars, M. 1996. Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE Tr. on Robotics and Automation*, 12(4):566-580.

[35] Khatib, O. 1986. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *Int. J. of Robotics Research*, 5(1), 90-98.

[36] Koga, Y., Kondo, K., Kuffner, J., and Latombe, J.C. 1994. Planning Motions with Intentions. *Proc. SIGGRAPH'94*, ACM, 395-408.

[37] Koga, Y. and Latombe, J.C. 1994. On Multi-Arm Manipulation Planning. *Proc. IEEE Int. Conf. on Robotics and Automation*, 945-952.

[38] Kotay, K., Rus, D., Vona, M., and McGray, C. 1998. The Self-Reconfiguring Robotic Module: Design and Control Algorithms. *Robotics: The Algorithmic Perspective*, 1998 WAFR, Agarwal, P.K., Kavraki, L.E., and Mason, M.T. (eds), A K Peters, Natick, MA, 375-386.

[39] Kuffner, J. and Latombe, J.C. 1999. Fast Synthetic Vision, Memory, and Learning for Virtual Humans. *Proc. Computer Animation'99*, IEEE, 118-127.

[40] Latombe, J.C. 1991. *Robot Motion Planning*, Kluwer Academic Pub., Boston, MA.

[41] Latombe, J.C. 1995. Robot Algorithms. *Algorithmic Foundations of Robotics*, WAFR'94, Goldberg, K. et al. (eds.), A K Peters, Natick, MA, 1-18.

[42] Laumond, J.P., Jacobs, P., Taix, M., and Murray, R. 1994. A Motion Planner for Nonholonomic Robots. *IEEE Tr. on Robotics and Automation*, 10:577-593.

[43] LaValle, S.M. and Kuffner, J.J. 1999. Randomized Kinodynamic Planning. *Proc. IEEE Int. Conf. on Robotics and Automation.*

[44] Lazanas, A. and Latombe, J.C. 1995. Motion Planning with Uncertainty: A Landmark Approach. *Artificial Intelligence*, 76(1-2):285-317.

[45] Liebermann, L.I. and Wesley, M.A. 1977. AUTOPASS: An Automatic Programming System for Computer Controlled Mechanical Assembly. *IBM J. of Res. and Dev.*, 21(4):321-333.

[46] Lin, M.C. and Canny, J.F. 1991. Efficient Algorithms for Incremental Distance Computation. *Proc. IEEE Int. Conf. on Robotics and Automation*, 1008-1014.

[47] Lozano-Pérez, T. 1976. *The Design of a Mechanical Assembly System*, Tech. Rep. AI-TR 397, AI Lab, MIT, Cambridge, MA.

[48] Lozano-Pérez, T. 1983. Statial Planning: A Configuration Space Approach. *IEEE Tr. Computers*, C-32(2):108-120.

[49] Lozano-Pérez, T., Mason, M.T., and Taylor, R.H. 1984. Automatic Synthesis of Fine-Motion Strategies for Robots. *Int. J. of Robotics Research*, 3:3-24.

[50] Lozano-Pérez, T. and Wesley M.A. 1979. An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles. *Comm. ACM*, 22(10):560-570.

[51] Nilsson, N.J. 1969. A Mobile Automaton: An Application of Artificial Intelligence Techniques. *Proc. 1st Int. Joint Conf. on Artificial Intelligence*, Washington D.C., 509-520.

[52] Quinlan, S. 1994. Efficient Distance Computation Between Non-Convex Objects. *Proc. IEEE Int. Conf. on Robotics and Automation*, 3324-3329.

[53] Reif, J.H. 1979. Complexity of the Mover's Problem and Generalizations. *Proc. FOCS*, 421-427.

[54] Schwartz, J.T. and Sharir, M. 1983. On the 'Piano Movers' Problem: II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds. *Advances in Applied Mathematics*, 4:298-351.

[55] Singh, A.P., Latombe, J.C. and Brutlag, D.L. 1999. A Motion Planning Approach to Flexible Ligand Binding. *Proc. ISMB'99.*

[56] Švestka, P. and Overmars, M. 1995. Coordinated Motion Planning for Multiple Car-Like Robots Using Probabilistic Roadmaps. *Proc. IEEE Int. Conf on Robotics and Automation*, 1631-1636.

[57] Takeda, H., Facchinetti, C., and Latombe, J.C. 1994. Planning the Motions of a Mobile Robot in a Sensory Uncertainty Field. *IEEE Tr. on Pattern Analysis and Machine Intelligence*, 16(10):1002-1017.

[58] Taylor, R.H. 1976. *Synthesis of Manipulator Control Programs from Task-Level Specifications.* Ph.D. Dissertation, Dept. of Computer Science, Stanford U., Stanford, CA.

[59] Tombropoulos, R.Z., Adler, J.R., and Latombe, J.C. 1999. CARABEAMER: A Treatment Planner for a Robotic Radiosurgical System with General Kinematics. *Medical Image Analysis*, 3(3):1-28.

[60] Udupa, S. 1977. *Collision Detection and Avoidance in Computer Controlled Manipulators.* Ph.D. Dissertation, Dept. of Electrical Engineering, California Institute of Technology, Pasadena, CA.

[61] Wilmarth, S.A., Amato, N.M., and Stiller, P.F. 1999. Motion Planning for a Rigid Body Using Random Networks on the Medial Axis of the Free Space. *Proc. 15th Annual ACM Symp. on Computational Geometry (SoCG'99).*

[62] Wilson, A., Larsen, E., Lin, M.C., and Manocha, D. 1999. IMMPACT: Partitioning and Handling Massive Models for Interactive Collision Detection. *Proc. Eurographics'99*, to appear.

[63] Wilson, R.H. and Latombe, J.C. 1995. Geometric Reasoning about Mechanical Assembly. *Artificial Intelligence*, 71:371-396.

[64] Yim, M. 1993. A Reconfigurable Modular Robot with Multiple Modes of Locomotion. *Proc. JSME Conf. on Advanced Mechatronics.* Also see `http://www.parc.xerox.com/spl/members/yim/modrobots.html`.