**Oliver Brock**

Laboratory for Perceptual Robotics
Computer Science Department
University of Massachusetts
Amherst, Massachusetts 01003
oli@cs.stanford.edu

**Oussama Khatib**

Robotics Laboratory
Department of Computer Science
Stanford University
Stanford, CA 94305, USA
khatib@cs.stanford.edu

# Elastic Strips: A Framework for Motion Generation in Human Environments

## Abstract

*Robotic applications are expanding into dynamic, unstructured, and populated environments. Mechanisms specifically designed to address the challenges arising in these environments, such as humanoid robots, exhibit high kinematic complexity. This creates the need for new algorithmic approaches to motion generation, capable of performing task execution and real-time obstacle avoidance in high-dimensional configuration spaces. The elastic strip framework presented in this paper enables the execution of a previously planned motion in a dynamic environment for robots with many degrees of freedom. To modify a motion in reaction to changes in the environment, real-time obstacle avoidance is combined with desired posture behavior. The modification of a motion can be performed in a task-consistent manner, leaving task execution unaffected by obstacle avoidance and posture behavior. The elastic strip framework also encompasses methods to suspend task behavior when its execution becomes inconsistent with other constraints imposed on the motion. Task execution is resumed automatically, once those constraints have been removed. Experiments demonstrating these capabilities on a nine-degree-of-freedom mobile manipulator and a 34-degree-of-freedom humanoid robot are presented, proving the elastic strip framework to be a powerful and versatile task-oriented approach to real-time motion generation and motion execution for robots with a large number of degrees of freedom in dynamic environments.*

KEY WORDS—**PLEASE PROVIDE**

The International Journal of Robotics Research
Vol. 21, No. 12, December 2002, pp. xxx-xxx,


## 1. Introduction

Roboticists are beginning to direct their efforts towards applications in unstructured and dynamic environments populated by humans. To perform tasks robustly and reliably in such domains, robots with sophisticated kinematic structures and powerful algorithms in control, planning, and perception are needed. Suitable robotic mechanisms have to overcome a variety of challenges imposed by such environments: locomotion is needed to cover the generally very large work space; legged locomotion might be required in environments designed for humans containing steps or stairs. To perform flexible and dexterous manipulation, potentially using tools designed for humans, mechanisms have to be equipped with multiple, independently controllable end-effectors. Changes in the environment have to be perceived to achieve robust, collision-free motion, imposing the need for actuated on-board vision systems. These diverse requirements have led to the development of humanoid robotic systems (Brooks 1997; Bischoff and Graefe 1999; Adams et al. 2000; Asfour, Berns, and Dillmann 2000; Kagami et al. 2001), which have a complex, branching kinematic structure with a large number of degrees of freedom and are equipped with vision sensors and multiple end-effectors.

As a result of kinematic complexity, new challenges arise in the fields of motion planning and control. For mechanisms with many degrees of freedom, the computational requirements of motion planning become increasingly large. Hence, the ability to replan a motion in reaction to changes in the environment is limited. Nevertheless, the motion executed by the robot has to reflect the dynamic aspects of the environment. In addition to those real-time limitations, due to the kinematic redundancy of human-like mechanisms, the specification of joint positions and trajectories is no longer a practical means

of defining a task. Consequently, the representation of a path or a plan has to encompass both the task and specific joint trajectories, potentially chosen arbitrarily from a set of trajectories consistent with the task. Finally, motion planning and control algorithms have to enable the generation and simultaneous execution of independent motion behavior, for example to allow coordinated manipulation and vision or integrated task execution and obstacle avoidance.

Currently, the areas of motion planning and robot control advance mainly independently of each other. The introduction of probabilistic methods in motion planning (Kavraki et al. 1996) has resulted in significant progress and the resulting methods can be applied to problems of high complexity. Similar success has been achieved in the area of control, where a powerful framework directed towards robots in human environments has been proposed (Khatib et al. 1999). However, a successful approach to task-driven motion generation and execution for human-like robots in dynamic and unstructured environments has to integrate the global aspects of motion planning with the dynamic capabilities and properties of the mechanism, as addressed by control methods. The research presented here is concerned with the integration of motion planning and control methods into a coherent framework of task-based motion generation and execution.

The elastic strip framework (Brock and Khatib 1997; Brock 2000) allows the integration of task-oriented dynamic control and motion coordination (Khatib et al. 1996, 2001) with global motion planning methods (Latombe 1991) and reactive, real-time obstacle avoidance (Khatib 1986). In this framework, tasks can be specified at the object level, leaving redundant degrees of freedom of the robot unspecified. Using those redundant degrees of freedom, elastic strips allow the integration of motion behavior in addition to task execution, such as obstacle avoidance or posture control. These behaviors can be controlled and changed reactively in real time without violating constraints imposed by the task. Thus, elastic strips provide a powerful approach to motion generation and execution, in particular for robots with complex kinematic structure operating in unstructured and dynamic environments.

## 2. Related Work

Research topics related to humanoid robots have recently received much attention. In this section, we survey work applicable to robots with complex kinematic structures, as well as discussing those approaches specifically directed towards human-like robots. In the discussion, we restrict ourselves to algorithms concerned with the generation and execution of motion.

### *2.1. Motion Planning in High-Dimensional Configuration Spaces*

Due to the large number of degrees of freedom required to perform complex tasks in dynamic environments, motion planning algorithms specifically addressing high-dimensional configuration spaces are of particular interest. Since the computational complexity of complete motion planning algorithms is exponential in the dimensionality of the configuration space (Canny 1988), those approaches are not practical for robots with complex kinematics. The introduction of probabilistically complete algorithms (Kavraki et al. 1996) has significantly increased the applicability of motion planning algorithms and remains an active area of research (Amato et al. 1998; LaValle, Yakey, and Kavraki 1999; Hsu et al. 1999; Bohlin and Kavraki 2000; Bohlin 2001). Using probabilistic approaches, motion planning problems in high-dimensional configuration spaces have been solved successfully. Even probabilistic methods, however, are unable to perform planning operations in high-dimensional configuration spaces in real time.

A particular probabilistic approach, the RTT method (LaValle and Kuffner 1999), has been applied to motion planning for humanoid robots with 33 degrees of freedom (Kuffner et al. 2001). In addition to generating motion avoiding static obstacles, balance constraints are taken into account to ensure dynamically stable posture during motion. The approach requires the pre-computation of a set of statically stable postures for the robot. As grasped objects or contact with the environment can change the factors determining stability of the robot, such an approach is limited to planning motion for the manipulation of objects known a priori.

Decomposition-based motion planning (Brock and Kavraki 2001) is another approach to planning in high-dimensional configuration spaces, trading completeness for efficiency. Based on the assumption that the robot will have a minimum clearance to obstacles along the resulting trajectory, it decomposes the original planning problem into two subproblems. One is a simple planning problem in Cartesian space and the other uses the solution to the first to determine a motion in the high-dimensional space of the original planning problem. Initial experimentation was able to demonstrate near real-time performance (Brock and Kavraki 2001).

In addition to involving a high number of degrees of freedom, the motion required for humanoid robots to perform human tasks can be quite complex. Motion planning algorithms, including those discussed above, generally only address the problem of generating a collision-free motion connecting an initial configuration to a goal configuration, ignoring the case in which the motion itself constitutes the task. Consequently, motion planning algorithms only address a part of the overall requirements of complex robots performing human tasks.

### *2.2. Motion Coordination for Redundant Manipulators*

Redundancy, as often encountered in mobile manipulators and human-like structures, poses certain difficulties for the automatic generation of motion. Mobile manipulators generally consist of multiple kinematic structures addressing

manipulation and mobility separately. Many approaches have been proposed for the generation of a coordinated motion of these structures, in particular for mobile manipulators with nonholonomic motion constraints (Seraji 1993; Desai and Kumar 1997; Perrier, Dauchez, and Perrot 1998; Bayle, Fourquet, and Renaud 2000). Given a specific trajectory of the end-effector, methods have been proposed to generate a coordinated motion for the overall system (Mohri, Furuno, and Yamamoto 2001). In all approaches, redundancy resolution schemes are used to determine the trajectory.

Redundancy resolution can also be used to implement different motion behaviors without affecting the execution of the task. This idea has been exploited for singularity avoidance of a mobile manipulator (Tanner and Kyriakopoulos 2000) and for system stability (Huang, Sugano, and Tanie 1998) to prevent a vehicle/arm system from tipping. Another approach to system stability exploits the redundancy of a humanoid robot to control its balance, while optimizing the overall posture for manipulability of the hands (Inoue et al. 2000).

In dynamic environments, coordinated motion has to be modified in real time to accommodate moving obstacles. An event-based motion generation approach for mobile manipulators commands the base of a vehicle/arm system to stop, while task execution is continued according to the work space limitations of the manipulator (Tan and Xi 2001). Given a dynamically generated trajectory of the end-effector, schemes have been devised to move the base of a vehicle/arm system such that the end-effector remains in the center of its work space (Yamamoto and Yun 1994). Another method allows reactive obstacle avoidance with the manipulator arm without affecting the motion of the base (Yamamoto and Yun 1995). A more general, potential field-based approach combines a task potential, causing the end-effector to follow a given trajectory, a coordination potential, using the base to center the end-effector in its work space, and an obstacle avoidance potential, keeping the base at a safe distance from obstacles, to generate motion for a mobile manipulator in a dynamic environment (Ögren, Egerstedt, and Hu 2000).

The approach to motion coordination for robots with kinematically complex structures presented in this paper relies on the operational space formulation (Khatib 1987), using the dynamically consistent force/torque relationship (Khatib 1995) to control the end-effector position and orientation and the redundant degrees of freedom of a mechanism independently. This approach is described in more detail in Section 3.

### *2.3. Integration of Planning and Control*

A successful strategy for motion generation in dynamic environments necessarily needs to combine the global aspects of the task, as generated by a planner, with local constraints imposed on the motion by, for example, unpredictably moving obstacles, balance constraints, work space limitations, or singularities of the mechanism. This insight has resulted in numerous approaches to motion generation (Faverjon and Tournassoud 1987; Barraquand and Latombe 1991; Choi and Latombe 1991; McLean and Cameron 1996; Baginski 1998) attempting to combine the desirable properties of global motion planners (Latombe 1991) with local control methods, such as the potential field approach (Khatib 1986).

A global path planner was developed by applying control algorithms to the motion planning problem (Warren 1989). In this approach, a potential function is associated with the interior of configuration space obstacles. A planning operation is initiated under the assumption that a straight-line path in configuration space connects the initial and the final configuration. The entire path is then exposed to the potential functions defined by the obstacles. Following the negative gradient of those potentials, the path is incrementally modified until it is collision-free. The definition of the potential functions, however, becomes prohibitively difficult, as the dimensionality of the configuration space increases. In addition, the computation of the configuration space obstacles is a very costly operation in those cases.

A very similar approach was taken to address the problem of real-time motion modification. The *elastic band framework* (Quinlan 1994b) represents a previously planned path as a curve in configuration space with properties similar to an elastic band. Obstacles exert repulsive forces, keeping the trajectory free of collision. However, in contrast to the previous approach, proximity information from the work space is used to modify the trajectory, rather than computing the configuration space obstacle. By avoiding the computation of the configuration space obstacles, the approach is computationally more efficient than that previously presented. It can be used to modify a trajectory during its execution in reaction to obstacles moving in the environment (Quinlan 1994b). The proximity information required for motion modification is derived from an estimate of local free space around the trajectory. As the dimensionality of the configuration space and the geometrical complexity of the robot increases, however, this estimate becomes excessively conservative, resulting in loss of real-time performance. In low-dimensional configuration spaces, the approach has been applied successfully and even extended to non-holonomic motion of mobile robots (Khatib 1996).

Combining ideas underlying the two previous approaches, a very fast motion planning method, the *BB method*, was introduced (Baginski 1998). Similar to the aforementioned global path planner, the trajectory is represented as a curve in configuration space and exposed to potentials resulting from obstacles. But, rather than using an explicit representation of those configuration space obstacles, proximity information is used to approximate them, as was the case in the elastic band approach. However, the BB method is subject to local minima and might fail to find a collision-free path, even when one exists. Furthermore, since it is a motion planning method it cannot incorporate motion constraints maintained by control methods.

Previous approaches perform an integration of planning and control to achieve faster motion generation (Faverjon and Tournassoud 1987; Warren 1989; Barraquand and Latombe 1991; McLean and Cameron 1996; Baginski 1998) or to allow motion execution in dynamic environments (Steele and Starr 1988; Choi and Latombe 1991; Quinlan 1994b). The elastic strip framework presented here is, to our knowledge, the first attempt of a general motion generation framework, addressing a wide range of task-driven aspects of robotic motion.

## 3. Task-oriented Control

The operational space formulation (Khatib 1987) serves as the underlying control structure in the elastic strip framework. It decomposes the overall control of a mechanism into *task behavior* and *posture behavior*. The elastic strip framework presented in the subsequent section relies on this decomposition for the integration of various motion behaviors with task execution.

### 3.1. Task Behavior

The joint space dynamics of a manipulator are described by

$$A(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{\Gamma}$$

where $\mathbf{q}$ is the $n$ joint coordinates, $A(\mathbf{q})$ is the $n \times n$ kinetic energy matrix, $\mathbf{b}(\mathbf{q}, \dot{\mathbf{q}})$ is the vector of centrifugal and Coriolis joint forces, $\mathbf{g}(\mathbf{q})$ is the vector of gravity, and $\mathbf{\Gamma}$ is the vector of generalized joint forces.

The operational space formulation (Khatib 1987) provides an effective framework for dynamic modeling and control of branching mechanisms (Russakow, Khatib, and Rock 1995), with multiple operational points. The generalized torque/force relationship (Khatib 1987) provides the decomposition of the total torque $\mathbf{\Gamma}$ into two dynamically decoupled command torque vectors, the torque corresponding to the task behavior command vector and the torque that only affects posture behavior in the nullspace:

$$\mathbf{\Gamma} = \mathbf{\Gamma}_{\text{task}} + \mathbf{\Gamma}_{\text{posture}}. \tag{1}$$

In this section we are only concerned with the torque vector $\mathbf{\Gamma}_{\text{task}}$, corresponding to task behavior. The next section also addresses motion behavior. For a robot with a branching structure of $m$ effectors or operational points, the task is represented by the $6m \times 1$ vector, $\mathbf{x}$, and the $6m \times n$ Jacobian matrix is $J(\mathbf{q})$. This Jacobian matrix is formed by vertically concatenating the $m$ $6 \times n$ Jacobian associated with the $m$ effectors.

The task dynamic behavior *without* posture specification is described by the operational space equations of motion (Khatib 1995)

$$\Lambda(\mathbf{x})\ddot{\mathbf{x}} + \mu(\mathbf{x}, \dot{\mathbf{x}}) + \mathbf{p}(\mathbf{x}) = \mathbf{F},$$

where $\mathbf{x}$ is the vector of the $6m$ operational coordinates describing the position and orientation of the $m$ effectors, and $\Lambda(\mathbf{x})$ is the $6m \times 6m$ kinetic energy matrix associated with the operational space. $\mu(\mathbf{x}, \dot{\mathbf{x}})$, $\mathbf{p}(\mathbf{x})$, and $\mathbf{F}$ are respectively the centrifugal and Coriolis force vector, gravity force vector, and generalized force vector acting in operational space.

The joint torque corresponding to the task command vector $\mathbf{F}$, acting in the operational space, is given by

$$\mathbf{\Gamma}_{\text{task}} = J^{\text{T}}(\mathbf{q})\ \mathbf{F}. \tag{2}$$

The task dynamic decoupling and control is achieved using the control structure

$$\mathbf{F}_{\text{task}} = \widehat{\Lambda}(\mathbf{x})\mathbf{F}^{\star}_{\text{task}} + \widehat{\mu}(\mathbf{x}, \dot{\mathbf{x}}) + \widehat{\mathbf{p}}(\mathbf{x})$$

where $\mathbf{F}^{\star}_{\text{task}}$ represents the inputs to the decoupled system, and $\widehat{\cdot}$ represents estimates of the model parameters. This control structure enables us to control a robot with multiple operational points in terms of the task, rather than the joint coordinates.

### 3.2. Posture Behavior

For non-redundant manipulators the task specification as described in Section 3.1 suffices to control all degrees of freedom of the manipulator. For redundant manipulators, however, this framework becomes incomplete. The specification of the redundant degrees of freedom is given by a desired posture for the robot. Such posture behavior can then be treated separately from the task, allowing intuitive task and posture specifications and effective whole-robot control. The overall control structure for task and posture is

$$\mathbf{\Gamma} = \mathbf{\Gamma}_{\text{task}} + \mathbf{\Gamma}_{\text{posture}},$$

where

$$\mathbf{\Gamma}_{\text{posture}} = N^{\text{T}}(\mathbf{q})\ \mathbf{\Gamma}_{\text{d-posture}}$$

with

$$N(\mathbf{q}) = \left[I - \bar{J}(\mathbf{q})\ J(\mathbf{q})\right]$$

where $\bar{J}(\mathbf{q})$ is the *dynamically consistent generalized inverse* (Khatib 1995), which minimizes the robot kinetic energy,

$$\bar{J}(\mathbf{q}) = A^{-1}(\mathbf{q})\ J^{\text{T}}(\mathbf{q})\ \Lambda(\mathbf{q})$$

and

$$\Lambda(\mathbf{q}) = [J(\mathbf{q})\ A^{-1}(\mathbf{q})\ J^{\text{T}}(\mathbf{q})]^{-1}.$$

$\mathbf{\Gamma}_{\text{d-posture}}$ refers to the desired posture torque. After mapping it into the nullspace we obtain $\mathbf{\Gamma}_{\text{posture}}$, which is the posture torque applied to the robot. The overall control structure is now given by

$$\mathbf{\Gamma} = J^{\text{T}}(\mathbf{q})\ \mathbf{F} + N^{\text{T}}(\mathbf{q})\ \mathbf{\Gamma}_{\text{d-posture}}. \tag{3}$$

This relationship provides a decomposition of joint forces into two control vectors: joint forces corresponding to forces acting at the task, $J^{\mathrm{T}}\mathbf{F}$, and joint forces that only affect the robot posture, $N^{\mathrm{T}}(\mathbf{q})\boldsymbol{\Gamma}_{\text{d-posture}}$. For a given task this control structure produces joint motions that minimize the robot's instantaneous kinetic energy. As a result, a task will be carried out by the combined action of the set of joints that reflect the smallest effective inertial properties.

To control the desired posture of the robot, the vector $\boldsymbol{\Gamma}_{\text{d-posture}}$ can be selected as the gradient of a potential function constructed to meet the desired posture specifications. Later, we see how such a gradient can be defined in terms of proximity to obstacles to realize obstacle avoidance, for example. The interference of posture behavior with the task dynamics is avoided by projecting it into the dynamically consistent nullspace of $J^{\mathrm{T}}(\mathbf{q})$, i.e. $N^{\mathrm{T}}(\mathbf{q})\boldsymbol{\Gamma}_{\text{d-posture}}$.

### *3.3. Branching Mechanisms*

Task-oriented control for one task, represented by a single operational point as described above, has been extended to branching mechanisms, such as human-like robots, with multiple operational points in a straightforward manner (Russakow, Khatib, and Rock 1995; Chang and Khatib 2000). For $m$ end-effectors, $\mathbf{F}$ and $J$ from eq. (3) are obtained by simple concatenation:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_m \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} \mathbf{F}_1 \\ \vdots \\ \mathbf{F}_m \end{bmatrix} \quad \text{and} \quad J = \begin{bmatrix} J_1 \\ \vdots \\ J_m \end{bmatrix},$$

where $\mathbf{x}_i$ represents the coordinates of the $i$th operational point, $\mathbf{F}_i$ represents the forces and moments acting at the $i$th operational point, and $J_i$ is the Jacobian at the $i$th operational point. Using this simple extension, multiple operational points of a branching mechanism can be controlled (Russakow, Khatib, and Rock 1995).

## 4. Elastic Strip Framework

The motion required for a robot to execute a sophisticated task necessarily combines global and local criteria. One example of such a task is the execution of a planned motion to reach a goal configuration (global behavior), while allowing reactive obstacle avoidance to accommodate changes in the environment (local behavior). The elastic strip approach provides a framework for the integration of these behaviors, enabling reactive motion execution while maintaining the global properties of the motion relating to the task. This is accomplished by an incremental modification of the previously planned motion. The method of modification guarantees that the resulting motion maintains the topological properties of the path as well as the constraints imposed by the task, and therefore represents a valid path from the current configuration to the goal configuration consistent with the task.

In this section the elastic strip framework is introduced. First, obstacle avoidance behavior in high-dimensional configuration spaces is demonstrated. The approach differs from purely reactive methods in that it maintains the global properties of the path and thus is not susceptible to local minima. Subsequently, the integration of more sophisticated motion behavior is introduced. This encompasses posture behavior and the concept of task-consistency. In addition, a general method for transitioning between different motion behaviors, based on motion constraints, is presented. Finally, the problem of local replanning is addressed. The integration of these diverse aspects renders the elastic strip framework a very powerful approach to motion generation for complex robots in dynamic environments.

### *4.1. Sets of Homotopic Paths*

The fundamental idea underlying the elastic strip framework is to represent a set of homotopic paths by unions of the work space volumes a robot would sweep out along them. This can be done without exploring the configuration space, by simply considering geometric and kinematic properties of the robot. The prerequisite for the computation of a work space volume representing a set of homotopic paths is the existence of a valid planned motion, called *candidate path*. Assume a planner has generated such a candidate path and it lies entirely in free space. A robot moving along the candidate path sweeps out a certain volume in the work space. This volume can be seen as an alternative representation of the path, which conventionally is viewed as a one-dimensional curve in the configuration space. A slight, continuous modification of the configuration space curve will result in a path homotopic to the candidate path. Again, there is a work space volume associated with this modified path, differing slightly from the volume of the candidate path. We will exploit the property that slight modifications of the curve in configuration space result in a slightly modified work space volume.

Intuitively, we can grow the work space volume swept by the robot along the candidate path to contain the volume swept along a modified path. It is evident that, by growing the free space around the candidate path, the volume will contain the volumes swept by the robot along a whole set of paths. Some paths in this set must be homotopic to the candidate path, as they can be obtained from it by a continuous mapping (Latombe 1991). These homotopic paths are represented implicitly by an approximation of the free space around the candidate path, rather than as a set of curves in the configuration space.

Using such a set of homotopic paths represented by a work space volume, planning and control can be integrated very tightly. The path generated by a motion planner is transformed into a more general representation by augmenting it with an implicit representation of paths homotopic to it. Control algorithms can then be used during execution to efficiently search

that space to find a valid and collision-free trajectory, representing incremental modifications of the candidate path.

### *4.2. Augmented Path Representation*

Let $\mathcal{P}_c$ be a path generated by a planner. This path represents a collision-free motion accomplishing a given task; we will call it *candidate path*. Furthermore, let $V_{\mathcal{P}}^{\mathcal{R}}$ be the work space volume swept by robot $\mathcal{R}$ along trajectory $\mathcal{P}$.[1] This work space volume can be seen as an alternative representation of the one-dimensional curve $\mathcal{P}$ in configuration space. Let $V_{\mathcal{P}}^{\delta}$ be defined as

$$V_{\mathcal{P}}^{\delta} = (V_{\mathcal{P}} \oplus b_{\delta}) \backslash V_{\mathcal{O}},$$

where $\oplus$ is the Minkowski sum operator, $b_{\delta}$ designates a ball of radius $\delta$ centered around the origin, and $V_{\mathcal{O}}$ is the work space volume occupied by obstacles. $V_{\mathcal{P}}^{\delta}$ corresponds to $V_{\mathcal{P}}$ grown by $\delta$ in all directions, excluding the work space volume occupied by obstacles. Assuming that the robot is not in contact with obstacles along the entire path, it is obvious that there exists a path $\mathcal{P}' \neq \mathcal{P}$ homotopic to $\mathcal{P}$ and obtained from $\mathcal{P}$ by a slight modification, such that $V_{\mathcal{P}'} \subset V_{\mathcal{P}}^{\delta}$. Therefore, $V_{\mathcal{P}}^{\delta}$ can be seen as an implicit representation of paths homotopic to the candidate path $\mathcal{P}_c$.

In the elastic strip framework, a work space volume of free space surrounding $V_{\mathcal{P}_c}$ is computed. Since, depending on the method of computation, it might not be identical to $V_{\mathcal{P}_c}^{\delta}$, we will denote it by $\mathcal{T}_{\mathcal{P}_c}$ and call it *elastic tunnel* of free space. This tunnel represents a work space volume implicitly describing a set of paths $P(\mathcal{T}_{\mathcal{P}_c})$ homotopic to $\mathcal{P}_c$:

$$P(\mathcal{T}_{\mathcal{P}_c}) = \left\{ \mathcal{P} \mid V_{\mathcal{P}} \subseteq \mathcal{T}_{\mathcal{P}_c} \text{ and } \mathcal{P} \simeq \mathcal{P}_c \right\},$$

where $\simeq$ denotes the homotopy relation between two paths. The condition $V_{\mathcal{P}_c} \subseteq \mathcal{T}_{\mathcal{P}_c}$ is called the *containment condition*. A criterion to determine the containment of the work space volume $V_{\mathcal{P}}$ swept by the robot along $\mathcal{P}$ in the tunnel $\mathcal{T}_{\mathcal{P}_c}$ for a particular implementation of the elastic strip framework is described in Section 5.3.

Given a candidate path $\mathcal{P}_c$, a corresponding tunnel $\mathcal{T}_{\mathcal{P}_c}$ can be easily computed (Brock 2000) using distance computations in the work space (see Section 5). An elastic strip $\mathcal{S}$ is a tuple consisting of a candidate path and its corresponding elastic tunnel; it is defined as $\mathcal{S} = (\mathcal{P}_c, \mathcal{T}_{\mathcal{P}_c})$.

An example of an elastic tunnel, given a particular scheme of free space computation, is shown in Figure 1. Five configurations of the Stanford Mobile Platform along a given path are displayed. The overlapping, transparent spheres indicate the computed free space. The union of those spheres represents the elastic tunnel. The spherical obstacle in the middle of the trajectory is restricting the size of the tunnel. It can also be seen how the robot lowers its arm to avoid collision with the obstacle. The resulting deformation of the represented free space volume is the reason for calling it the elastic tunnel.

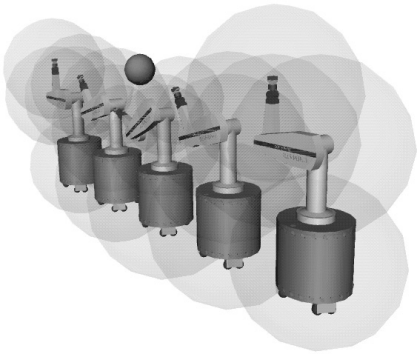


Fig. 1. Elastic tunnel. The union of the local free space, shown as transparent spheres, of five configurations along the elastic strip makes up the elastic tunnel. A spherical obstacle has modified the trajectory and the associated representation of free space.

### *4.3. Obstacle Avoidance Behavior*

Given an elastic strip $\mathcal{S}$, an algorithm is required to efficiently select a new candidate path $\mathcal{P}_c'$, completely contained by the elastic tunnel. Using a potential field-based control algorithm, real-time performance can be achieved for this operation. Rather than exploring the entire configuration space, the algorithm maps proximity information from the environment into the configuration space, using the kinematic structure of the manipulator. The elastic strip framework differs from other reactive approaches in that potential fields are applied to a discretized representation of the entire trajectory and not only to a particular configuration of the robot.

The candidate path $\mathcal{P}_c$ is represented as a discrete set of consecutive configurations. Virtual robots at these configurations are exposed to forces, acting in the work space, incrementally modifying the candidate path to yield a new one. The forces are derived from two potential functions, the external and internal potential, $V_{\text{external}}$ and $V_{\text{internal}}$, respectively.

The external, repulsive potential $V_{\text{external}}$ is defined as a function of proximity to obstacles. Minimizing this potential effectively maximizes the clearance the path has to obstacles in the environment. For a point $p$ on a configuration of the robot along the trajectory, the external potential is defined as

$$V_{\text{external}}(p) = \begin{cases} \frac{1}{2} k_r (d_0 - d(p))^2 & \text{if } d(p) < d_0 \\ 0 & \text{otherwise} \end{cases},$$

where $d(p)$ is the distance from $p$ to the closest obstacle, $d_0$ defines the region of influence around obstacles, and $k_r$ is the

1. For simplicity we will drop the exponent $\mathcal{R}$ in the remainder of this paper. Unless otherwise noted variables refer to the robot $\mathcal{R}$.

repulsion gain. The force resulting from this potential that acts on point $p$ is given by

$$\mathbf{F}_p^{\text{ext}} = -\nabla V_{\text{external}} = k_r \, (d_0 - d(p)) \, \frac{d}{\|d\|},$$

where $d$ is the vector between $p$ and the closest point on an obstacle. Intuitively, the repulsive potential pushes the trajectory away from obstacles, if it is inside their influence region. Using the external forces to select a new candidate path maintains the global properties of the path and local minima can be avoided.

The external potential alone would suffice in most cases to select a new candidate path. The resulting repulsive forces keep the trajectory in free space. If an obstacle deforming a path would recede, however, the path would never shorten. Virtual springs attached to control points on consecutive configurations of the robot along the elastic strip can achieve this effect. Let $p_j^i$ be the position vector of the control point attached to the $j$th link of the robot in configuration $q_i$ along the elastic strip. The internal contraction force acting at control point $p_j^i$, caused by the virtual spring connecting it to the configurations $q_{i-1}$ and $q_{i+1}$ along the elastic strip $\mathcal{S}$, is defined as

$$\mathbf{F}_{i,j}^{\text{internal}} = k_c \left( \frac{d_j^{i-1}}{d_j^{i-1} + d_j^i} (p_j^{i+1} - p_j^{i-1}) - (p_j^i - p_j^{i-1}) \right),$$

where $d_j^i$ is the distance $\| p_j^i - p_j^{i+1} \|$ in the initial, unmodified trajectory and $k_c$ is a constant determining the contraction gain of the elastic strip.

The definition of internal forces causes the tension in the elastic strip to be dependent upon the local curvature of the strip, rather than its length. If the internal tension of the strip were dependent on its length, the distance to obstacles would reduce with elongation. Furthermore, by introducing the scaling factor $\frac{d_j^{i-1}}{d_j^{i-1}+d_j^i}$, the relative distance between consecutive configurations along the elastic strip is preserved. Without this factor, the configurations representing the candidate path would tend to drift away from obstacles into regions of the trajectory where no external forces act.

Exposed to external and internal forces, the elastic strip behaves like a strip of rubber. Obstacles cause it to deform, and as obstacles recede it assumes its previous shape. The forces are mapped to joint displacements using a kinematic model of the manipulator. This effectively replaces configuration space exploration with a directed search, guided by work space forces. The necessary computations are virtually independent of the dimensionality of the configuration space. They mainly depend on the geometric properties of the robot and the environment (Brock 2000). The computation of the kinematic mapping obviously still depends on the number of degrees of freedom of the robot, but in practice its computational requirements are dwarfed by the cost of the free space computation.

The virtual robots at the discrete configurations along the elastic strip are exposed to torques given by eq. (2), which can be extended to

$$\boldsymbol{\Gamma}_{\text{avoidance}} = \sum_{p \in \mathcal{R}} J_p^{\mathrm{T}}(\mathbf{q}) \, \left( \mathbf{F}_p^{internal} + \mathbf{F}_p^{external} \right), \qquad (4)$$

where $J_p(\mathbf{q})$ represents the Jacobian at point $p$ on the robot $\mathcal{R}$ and $\mathbf{F}_p$ designates a force acting at $p$. Forces resulting from external and internal potentials are simply mapped into torques; these torques are then used to simulate the behavior of the robot in a configuration along the elastic strip. It is important to realize that the resulting motion will never be performed by the robot. The robot moves *through* this configuration *along the elastic strip*, but does not perform the motion of the virtual robot representing the elastic strip at discrete configurations. Consequently, it suffices to compute the motion based on kinematics; dynamic effects of the motion can be ignored.

Figure 2 shows the obstacle avoidance behavior, resulting from internal and external potentials for two Stanford Mobile Platforms (Brock and Khatib 1999), consisting of a holonomic base and a PUMA 560 manipulator arm with six degrees of freedom (see also Extension 1). The upper and lower rows of the three pictures show the same experiment from different perspectives. Lines indicate the elastic strip; vertical lines indicate those configurations of the robot which are represented by the elastic strip. Control points on consecutive configurations are connected by horizontal lines to indicate the trajectory. The images show how the trajectory of the robot to the left is incrementally modified as the other manipulator moves into its path. Note how base as well as arm motion is generated by the elastic strip framework to avoid the obstacle.

Figure 3 (Extension 2) shows the same experiment conducted on real robots. The sequence of images shows how the trajectory is modified in real time, based on information about the configuration of the robot representing the obstacle. Perception of the moving obstacle is addressed by querying the robot's configuration at regular intervals and using a geometric model of the robot to update the world model.

The trajectory of the base in the $x/y$ plane and the joint angles as a function of time for the waist, shoulder, and elbow of the PUMA are shown in Figures 4(a) and (b), respectively. Figure 4(a) shows how the base deviates increasingly from the initial straight-line trajectory, as the obstacle approaches. Once the obstacle is passed, a straight-line trajectory to the goal configuration is assumed. The joint trajectories in Figure 4(b) are plotted relative to the final desired arm configuration; only the portion of the overall trajectory is shown, for which the joint angles deviate from that position. The arm moves in a smooth manner to perform obstacle avoidance.

### *4.4. Posture Behavior*

To generate a collision-free motion, the elastic strip framework uses proximity information from the work space in order

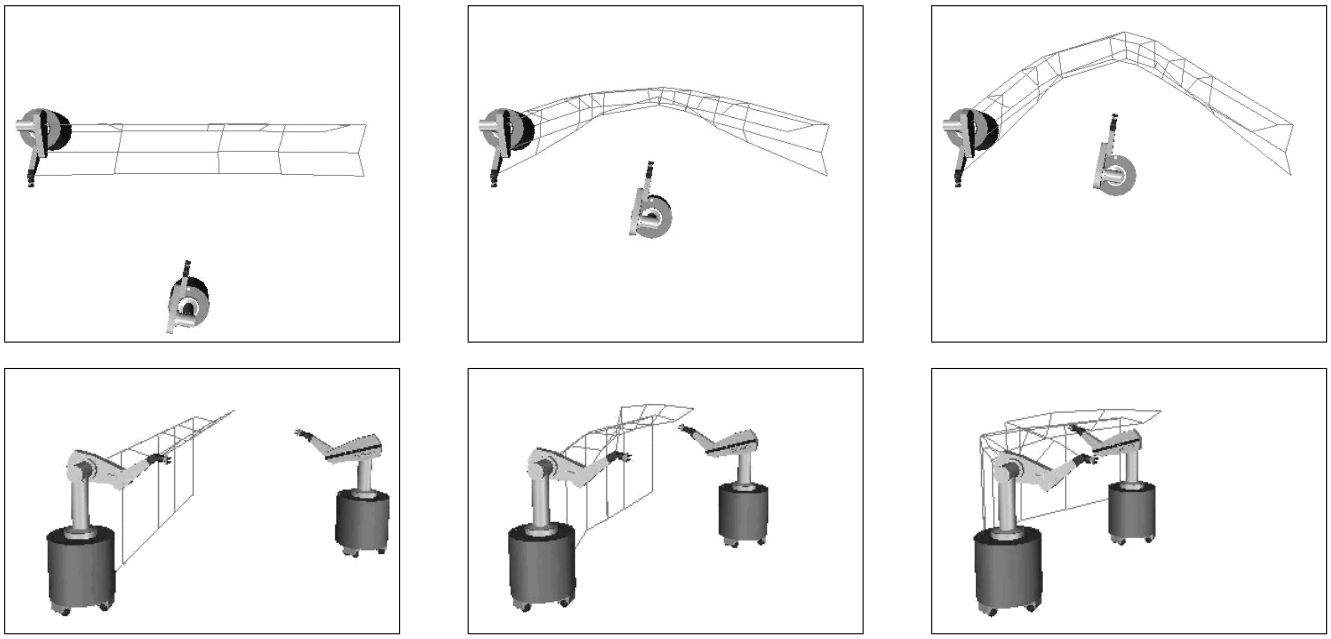
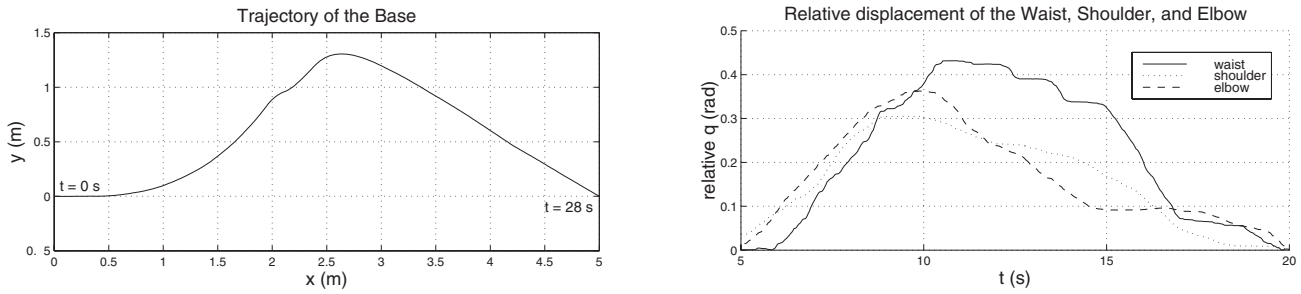


Fig. 2. Obstacle avoidance. The trajectory of a Stanford Robotic Platform, which is indicated by a wire frame representation, is modified in real time as another robot, serving as an obstacle, moves into the path.

Fig. 3. Obstacle avoidance behavior using the elastic strip framework is demonstrated on the Stanford Robotic Platform. The experiment corresponds to that shown in Figure 2.




(a) (b)

Fig. 4. Graphs showing the base trajectory and the joint motion for the waist, shoulder and elbow of the Stanford Robotic Platform during the experiment shown in Figure 3.

to guide the search in the configuration space, as described by eq. (4). When the proximity information is mapped into the configuration space, undesired motion behavior may result. This is illustrated in Figure 5(a) (Extension 3). A humanoid robot with 34 degrees of freedom—seven in each leg and arm and three at the waist and neck—passes underneath a descending beam. Note that the complexity of legged locomotion is ignored; instead the humanoid robot is treated as floating in the plane. In addition, the simulation of its motion is purely kinematic, while respecting given torque limitations. Repulsive forces exerted by the beam result in an unnatural and physically unstable motion. This is avoided by imposing an additional posture potential, which can describe a preferred posture for the robot, keep joint angles in a desired range to avoid joint limits, or prevent self-collision.

For instance, the robot posture can be controlled to maintain the robot total center-of-mass aligned along the vertical $z$-axis of a reference frame attached to the center of the supporting surface of the feet. This posture can be simply implemented with a posture energy function

$$V_{\text{posture}}(\mathbf{q}) = \frac{1}{2}\, k\, (x_{\text{CoM}}^2 + y_{\text{CoM}}^2) \tag{5}$$

where $k$ is a constant gain, and $x_{\text{CoM}}$ and $y_{\text{CoM}}$ are the $x$ and $y$ coordinates of the center of mass. The gradient of this function

$$\mathbf{\Gamma}_{\text{posture}} = J_{\text{CoM}}^{\text{T}}\, (-\nabla V_{\text{posture}}),$$

where $J_{\text{CoM}}$ is the Jacobian associated with the center of mass of the manipulator, provides the required attraction to the $z$-axis of the robot center of mass. The resulting torque can simply be added to the torques resulting from obstacle avoidance, as defined in eq. (4):

$$\mathbf{\Gamma} = \mathbf{\Gamma}_{\text{avoidance}} + \mathbf{\Gamma}_{\text{posture}}. \tag{6}$$

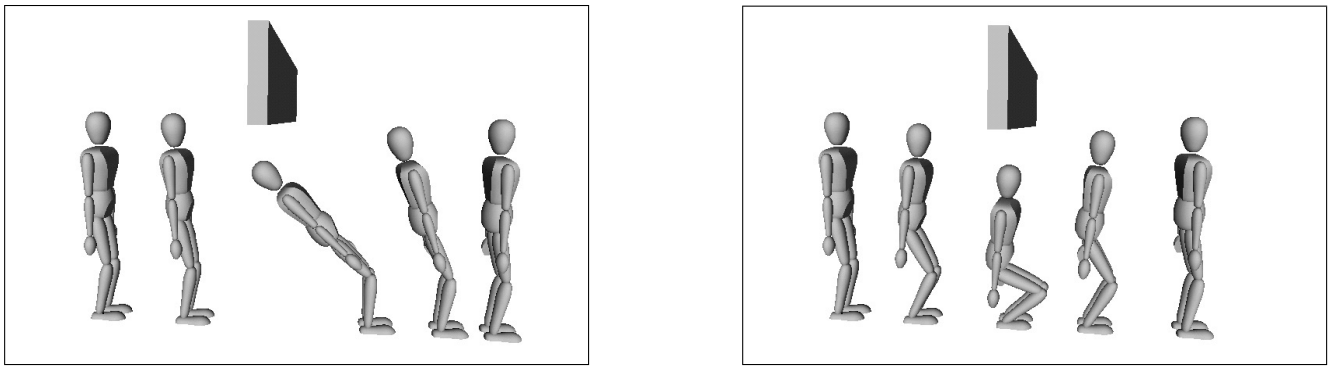
This example is intended to demonstrate the ease of integration of such behaviors into the overarching framework. The center of mass can only guarantee static equilibrium. To ensure dynamic balance, in particular during walking, the notion of a zero moment point (Vukobratović, Frank, and Juričić 1970) needs to be applied.

Imposing a posture as that described in eq. (5) based on the center of mass of the robot would ensure physical feasibility in the static case or in a situation with low accelerations. The resulting behavior could be bending the trunk backwards while reaching forward with the arms. Such a motion would ensure stability, but would not necessarily look natural.

Figure 5(b) (Extension 4) shows the same motion as shown in Figure 5(a) when the posture of the robot is controlled to bend the knees in reaction to a bent upper body. To accomplish this behavior, a torque proportional to the upper body angle is applied to knee, ankle, and hip, causing the figure to bend its knees without changing the orientation of the upper body. In addition, a restoring potential for the upper body maintains an upright posture. Internal forces of the elastic strip will cause the legs to straighten if the obstacle is removed. The resulting motion appears more natural, but these posture potentials are not necessarily suited to ensure dynamic stability.

Note that an arbitrary number of posture energies can be added to the motion by summing the resulting torques. It would thus be possible to combine the natural-looking motion with the physically motivated potential from eq. (5).

Research in the computer graphics community is concerned with automated, natural-looking character animation (Zhao and Badler 1994; Lee and Shin 1999). The elastic strip framework can be applied here, as demonstrated by the example in Figure 6 (Extension 5), where posture control and obstacle avoidance are combined to result in automatically generated, human-like motion of an animated character. A skiing humanoid figure evades a moving snowman and crouches to pass under the banner, which is lowered continuously. Note how the ski poles are moved closer to the body when passing the snowman and the gate. The motion is specified by giving



(a) (b)

Fig. 5. Posture behavior. Motion of a humanoid robot passing under a beam, which is being lowered. (a) shows the motion of the robot without posture control. The motion in (b) is generated with a posture energy causing the upper body to remain upright and the legs to bend.

an initial and a final configuration for the skier. The resulting actual motion is determined in real time based on proximity information to obstacles in the environment and posture energies. This example illustrates how complex motion behavior consisting of obstacle avoidance and posture control can be generated in real time for mechanisms with high kinematic complexity.

Using the sum of torques (see eq. (6)) to combine behaviors, as was the case in the examples presented in this section, the execution of behaviors cannot be guaranteed. Components of the summed torque vectors can cancel each other so that the desired behavior might not be accomplished. In the next section, the elastic strip framework is extended to ensure the execution of the behavior with highest priority, even when performing multiple secondary tasks.

### *4.5. Task-consistent Behavior*

The path modification and posture control performed above employs all degrees of freedom of the manipulator to ensure obstacle avoidance. In other words, obstacle avoidance is regarded as the task of the robot as it moves along the elastic strip towards its goal configuration. For non-redundant manipulators this is necessary, as any deviation from the original motion would necessarily violate task constraints. In the case of redundant manipulators, however, task execution and posture behavior can be performed simultaneously. Since the manipulator is redundant with respect to the task given by forces and moments and designated by $\mathbf{F}$, the nullspace of the associated Jacobian has a non-zero rank. Obstacle avoidance and posture behavior can be performed in that nullspace.

The overall motion behavior and the associated torques can be divided into three different components: the task, constraint satisfaction, and posture behavior. The task relates to desired forces and moments at the end-effectors. Constraints encompass those requirements of the motion, which should be maintained at all times, as their violation could potentially be fatal to the mechanism. They include, for example, obstacle avoidance, joint limit and self-collision avoidance, and balance control. Combining the corresponding torques yields the overall torque applied to the robot:

$$\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_{\text{task}} + \boldsymbol{\Gamma}_{\text{constraints}} + \boldsymbol{\Gamma}_{\text{posture}}.$$

To perform motion behavior in a task-consistent manner, control is then performed using eq. (3)

$$\boldsymbol{\Gamma} = J^{\mathrm{T}}(\mathbf{q})\ \mathbf{F} + N^{\mathrm{T}}(\mathbf{q})\ \left(\boldsymbol{\Gamma}_{\text{d-constraints}} + \boldsymbol{\Gamma}_{\text{d-posture}},\right) \qquad (7)$$

where $\mathbf{F}$ represents an arbitrary task described by forces and moments at the end-effectors. Only considering obstacle avoidance constraints, $\boldsymbol{\Gamma}_{\text{d-constraints}}$ is defined as $\boldsymbol{\Gamma}_{\text{d-avoidance}}$ in eq. (4). The subscript "d-" indicates the desired torques. The actual torques result from mapping the desired quantities into the nullspace associated with the task. The properties of the operational space formulation guarantee that posture behavior mapped into the nullspace will not affect the task. Obstacle avoidance is performed in a *task-consistent* manner. Using this formulation, an arbitrary number of desired posture behaviors, like those described in Section 4.4, can be realized in a task-consistent manner by simply mapping them into the nullspace of the task. Thus, a more general definition of $\boldsymbol{\Gamma}_{\text{d-posture}}$ is given by

$$\boldsymbol{\Gamma}_{\text{d-posture}} = \sum_i k_i\ \boldsymbol{\Gamma}_i,$$

where $k_i$ are constant gains and $\boldsymbol{\Gamma}_i$ is the torque resulting from the forces derived from an arbitrary posture energy function, such as that given in eq. (5).

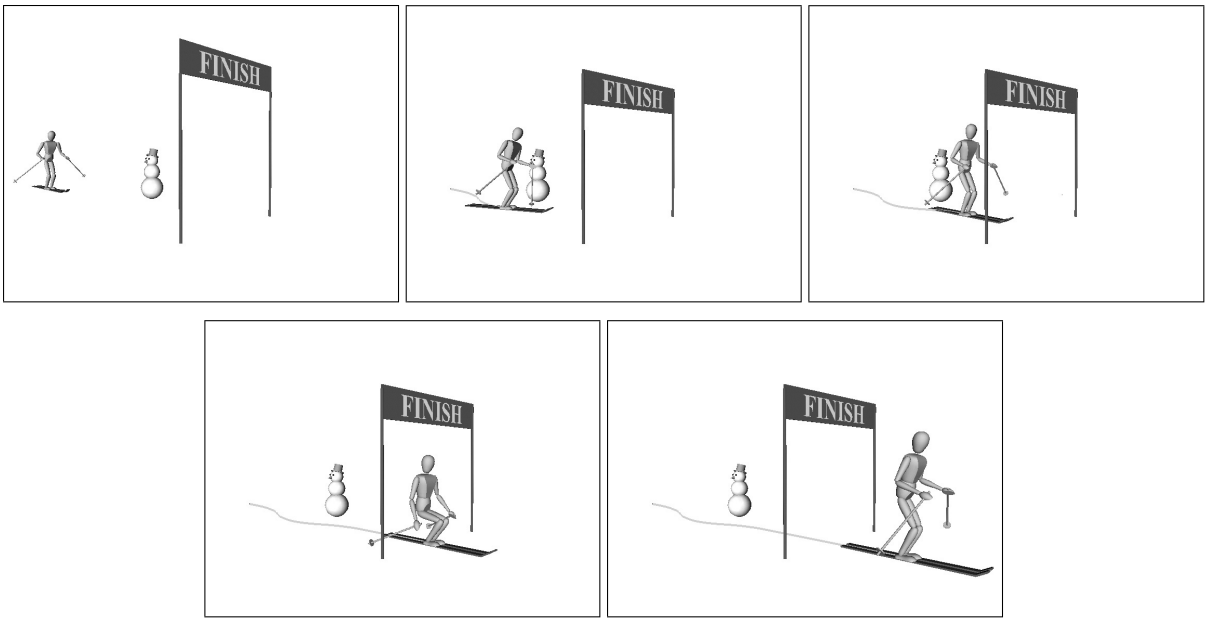



Fig. 6. A human figure skies around an approaching snowman and passes through a gate, while the banner is being lowered. Note how all degrees of freedom of the robot are used to avoid collision in real time, as indicated by the ski poles moving closer to the body when passing the snowman and the gate. Posture energy causes the skier to maintain a human-like posture.

The following experiments demonstrate the integration of task behavior and obstacle avoidance. The exemplary task consists of the end-effector following a straight-line trajectory. Figure 7(a) (Extension 6) shows five consecutive configurations of the Stanford Mobile Platform along a trajectory. The shown trajectory resulted from a straight-line trajectory which was modified by two small mobile robots moving into the path from opposite directions. Lines indicate the trajectory of points on the base, elbow, and end-effector. The end-effector error of this motion with respect to the desired task is shown in Figure 7(b); the end-effector follows the base trajectory in avoiding the two obstacles and results in a large error.

In Figure 7(c) (Extension 7) the obstacle robots perform the identical motion, but the elastic strip is modified in a task-consistent manner, enabling the end-effector to remain on the straight-line trajectory, as indicated by the corresponding end-effector error graph in Figure 7(d).

Task-consistent modification in the elastic strip framework was also demonstrated in experiments on the Stanford Robotic Platform itself, rather than in simulation (see Figure 8 and Extension 8). During these experiments the end-effector error generally did not exceed 2 mm; in some experiments, motion with sub-millimeter error was achieved. The motion of the obstacle, a Scout robot, is perceived using a SICK laser range finder.

### *4.6. Suspending Task Behavior*

The integration of task execution, obstacle avoidance, and posture behavior as described above can only be performed as long as the torques resulting from mapping $\Gamma_{\text{d-constraints}}$ and $\Gamma_{\text{d-posture}}$ into the nullspace (see eq. (7)) yield sufficient motion to accomplish the desired overall behavior. The ability to move inside the nullspace of the task is significantly reduced, for example, when the manipulator reaches the limits of its workspace or a singularity. Furthermore, it is possible for desired posture behavior or obstacle avoidance behavior to directly conflict with the task. In such a situation it is desirable to suspend task execution in order to perform motion necessary to fulfill the required motion constraints. Once the task execution has been suspended, the constraint behavior previously executed in the nullspace of the task can now be executed using all degrees of freedom of the robot. When possible, task execution should be resumed. In this section, we introduce criteria for determining when a transition is required and methods for performing such a transition.

While the task does not conflict with the constraints the robot can be controlled using eq. (7), which maps constraint satisfaction and posture torques into the nullspace:

$$\boldsymbol{\Gamma} = J^{\mathrm{T}}(\mathbf{q})\ \mathbf{F} + N^{\mathrm{T}}(\mathbf{q})\ \left(\boldsymbol{\Gamma}_{\text{d-constraints}} + \boldsymbol{\Gamma}_{\text{d-posture}}\right).$$

The transition criteria determine under which conditions the task is suspended or resumed. The criterion to suspend the

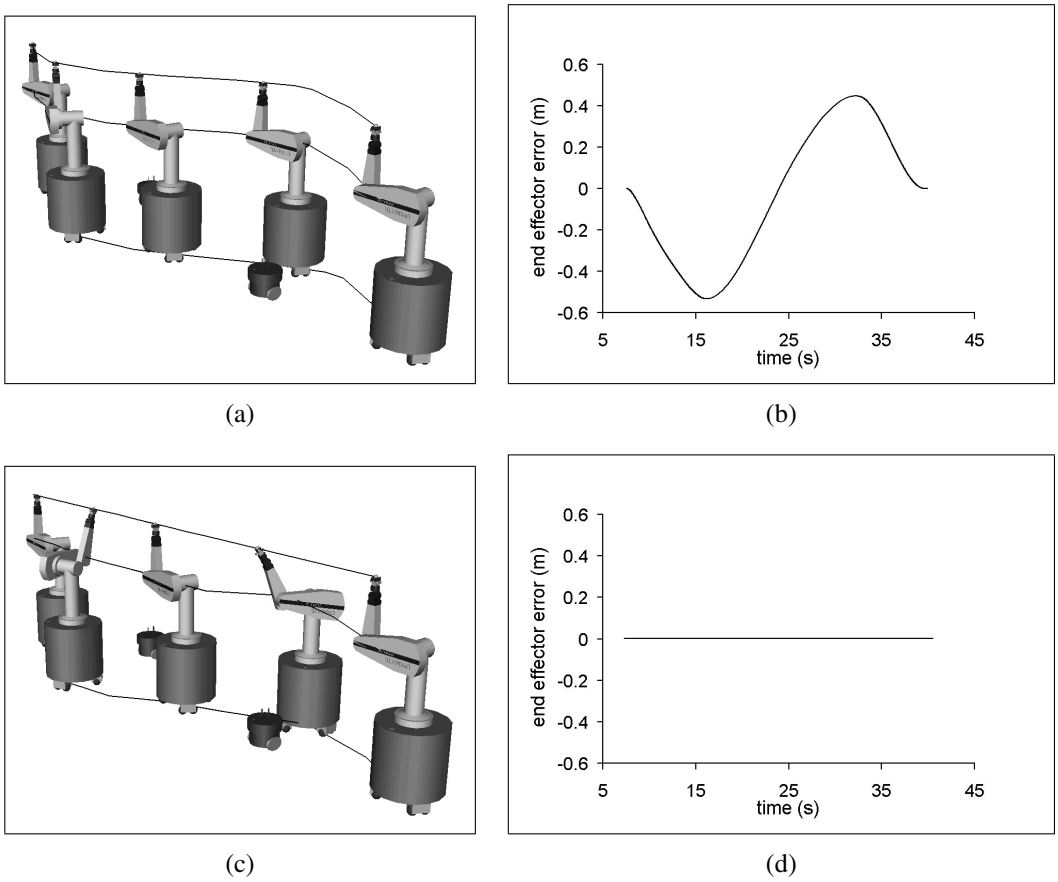



Fig. 7. Task consistency. A comparison of task-inconsistent and task-consistent obstacle avoidance. (a) shows obstacle avoidance of two mobile robots without task consistency by a Stanford Robotic Platform. The graph in (b) depicts the associated end-effector error, relative to the task of following a straight line. The end-effector performs a motion similar to the base, resulting in large error. (c) and (d) show the corresponding image and graph for task-consistent obstacle avoidance. The obstacle performs the same motion as in (a), but the end-effector error is minimal.

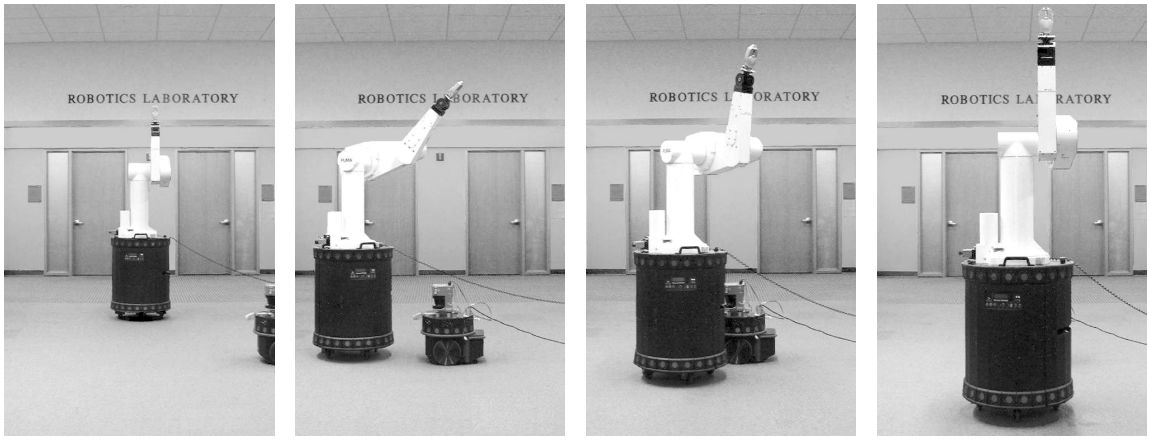



Fig. 8. Demonstrating task-consistent obstacle avoidance with the Stanford Robotic Platform. The end-effector performs a straight-line motion, while the base avoids the obstacle, which is perceived using a laser range finder.

task should maintain task behavior as long as possible, while the criterion to resume the task should be fulfilled as early as possible. Their optimality with respect to those requirements depends on properties of the robot and the environment. Here, we present simple and quite general criteria that do not depend upon a specific mechanism.

Let $N^{\mathrm{T}}(\mathbf{q}) = \left[I - J^{\mathrm{T}}(\mathbf{q})\ \bar{J}^{\mathrm{T}}(\mathbf{q})\right]$ be the dynamically consistent nullspace mapping of the Jacobian $J(\mathbf{q})$ associated with the task. We define the coefficient

$$c = \frac{\|\mathbf{\Gamma}_{\text{constraints}}\|}{\|\mathbf{\Gamma}_{\text{d-constraints}}\|},$$

where $\mathbf{\Gamma}_{\text{constraints}} = N^{\mathrm{T}}(\mathbf{q})\ \mathbf{\Gamma}_{\text{d-constraints}}$, to correspond to the quotient of the magnitude of the mapped and unmapped constraint satisfaction torque vector. The value $c \in [0..1]$ is an indication of how well the behavior represented by $\mathbf{\Gamma}_{\text{d-posture}}$ can be performed inside the nullspace of the task.

To determine when a transition to suspend the task needs to be initiated, we empirically determine a value $c_{\text{suspend}}$ at which it is desirable to suspend task execution in favor of the behavior previously mapped into the nullspace. Once the coefficient $c$ assumes a value $c < c_{\text{suspend}}$, a transition is initiated. During this transition, task and posture behavior at the joint are gradually suspended and constraint satisfaction behavior, previously performed in the nullspace, is now transitioned from the nullspace into the full space, using all degrees of freedom of the manipulator. For large values of $c_{\text{suspend}}$ the task is maintained longer, whereas small values result in a fast suspension of the task. The optimal value depends on the acceleration capabilities of the manipulator with respect to the anticipated velocity of obstacles; if obstacles can move much faster than the manipulator itself, the transition should be initiated early and a lower value for $c_{\text{suspend}}$ should be chosen. To ensure the avoidance of kinematic singularities or collisions with obstacles, i.e., the violation of motion constraints, $c_{\text{suspend}}$ should not be chosen too close to 1. During the experiments presented in Figure 9 a value of $c_{\text{suspend}} = 0.8$ was used.

The reverse transition to resume task behavior is initiated when the magnitude of the vector $\mathbf{F}$, representing the forces and moments applied at the end-effector to resume the task, becomes smaller than a threshold $\epsilon$, $\|\mathbf{F}\| \leq \epsilon$, and simultaneously $c > c_{\text{resume}}$, where $c_{\text{resume}}$ is the threshold for resuming the task. This condition is an indication that the task can be executed while realizing the posture behavior entirely in the nullspace. The value for $c_{\text{resume}}$ is chosen so that $c_{\text{resume}} > c_{\text{suspend}}$; this deadband between $c_{\text{suspend}}$ and $c_{\text{resume}}$ avoids unnecessary transitions.

Since $\mathbf{F}$ is proportional to the distance between the current and the desired end-effector configuration, $\epsilon$ effectively describes a region around the desired configuration within which the task is considered to be fulfilled.

The above criteria determine when to initiate a transition to suspend or resume task behavior at a given joint. The actual transition to suspend the task execution is performed based on the coefficient $c$ and a desired duration $t_{\text{suspend}}$ for the transition. Transitions are characterized by a transition variable $\alpha \in [0, 1]$. When suspending the task, $\alpha_{\text{suspend}}$ for a given degree of freedom $i$ is given by

$$\alpha^{\text{suspend}} = \begin{cases} \min\left(\frac{c}{c_{\text{suspend}}}, 1 - \frac{t - t_0}{t_{\text{suspend}}}\right) & \text{if } t - t_0 < t_{\text{suspend}} \\ 0 & \text{otherwise} \end{cases},$$

where $t$ is the current time, and $t_0$ is the time at which the transition was initiated. The time-based component allows a smooth transition under normal circumstances, whereas considering $c$ permits a more rapid reaction to extreme situations, in which the mapping into the nullspace yields only very minimal torques.

The transition to resume the task is performed entirely based on the desired duration $t_{\text{resume}}$. The transition variable $\alpha_{\text{resume}}$ for a given degree of freedom $i$ is defined as

$$\alpha^{\text{resume}} = \begin{cases} \frac{t - t_0}{t_{\text{resume}}} & \text{if } t - t_0 < t_{\text{resume}} \\ 1 & \text{otherwise} \end{cases}.$$

The parameters $t_{\text{suspend}}$ and $t_{\text{resume}}$ can be chosen based upon the acceleration capabilities of the manipulator and the expected rate of change in the environment. They affect the appearance of the resulting overall motion.

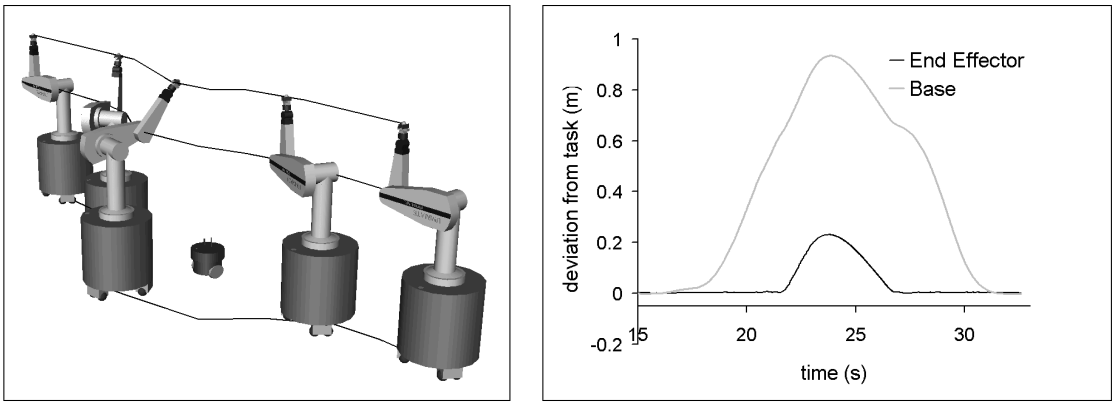
The transitions are performed based on the transition variable $\alpha$ as defined above. This variable is used as an argument to the transition function $f(\cdot)$. By varying $f(\cdot)$ different transition behaviors can easily be accomplished. The simplest choice for $f(\cdot)$ is the identity function, in which case the transition variable $\alpha$ will generate a linear transition over time. Other choices include the sigmoidal function $f(x) = \frac{1}{1+e^{-x}}$, scaled and translated to the interval $[0, 1]$. The condition of $f(\alpha_i) + f(1 - \alpha_i) = 1$ for all $\alpha \in [0, 1]$ is not necessary for a particular choice of $f$; it is desirable, however, to maintain this property at the end points of that interval.

The motion of the manipulator is generated using

$$\begin{aligned} \mathbf{\Gamma} \quad = \quad & f(\alpha) \quad J^{\mathrm{T}}(\mathbf{q})\ \mathbf{F} \\ & + f(\alpha) \quad N^{\mathrm{T}}(\mathbf{q})\ \left(\mathbf{\Gamma}_{\text{d-constraints}} + \mathbf{\Gamma}_{\text{d-posture}}\right) \\ & + f(\bar{\alpha}) \quad \mathbf{\Gamma}_{\text{d-constraints}} \end{aligned}$$

where $\bar{\alpha} = (1 - \alpha)$ is defined as the complement of the transition variable $\alpha$, given by $\alpha_{\text{suspend}}$ or $\alpha_{\text{resume}}$ depending on the transition.

The process of suspending and resuming task behavior is illustrated in in Figure 9(a) (Extension 9). The graph in Figure 9(b) shows the deviation from the task for the base and the end-effector. In the initial portion of the graph, the base starts deviating to avoid the obstacle, while the end-effector continues to execute the task. Once the base deviates approximately 0.6 m, the task is suspended and the end-effector deviated follows the base trajectory up to a deviation of over 0.2 m. After




(a) (b)

Fig. 9. Suspension and resumption of task-consistent obstacle avoidance. The obstacle, a small mobile robot, moves into the path of a Stanford Robotic Platform. (a) shows five configurations along the trajectory; lines indicate the trajectory of the based, elbow, and end-effector. As the graph of (b) indicates, the base deviates significantly from the straight-line trajectory. The end-effector follows the task with negligible error, until the task is suspended. Once the obstacle is passed, the end-effector error is reduced until the task can be resumed.

$t = 24$ s the obstacle is passed and the end-effector deviation is reduced until the task can be resumed. The graph resulted from an experiment on the Stanford Robotic Platform; the obstacle was perceived using a SICK laser range finder. For this particular experiment, the maximum end-effector error during task-consistent execution was 3 mm.

### *4.7. Replanning*

In the elastic strip framework, the modification of a trajectory is accomplished by the application of local potential fields. Like any other method generating global motion based on local information, it can fail. We differentiate two failure modes. One kind of failure results from a topological change of the configuration free space of the robot. The candidate path $\mathcal{P}_c$ of an elastic strip $\mathcal{S}$ represents a curve in configuration free space connecting the initial configuration of the robot with the goal configuration. Should changes in the environment result in the elimination of a passage in the configuration free space through which $\mathcal{P}_c$ passes, the elastic strip framework will fail. This failure cannot generally be avoided, as the topological information provided by the planner in the form of the initial candidate path has become invalid. A planner has to be invoked to compute a new candidate path.

A second kind of failure can occur, when the passage in the configuration free space around $\mathcal{P}_c$ has become so narrow that a valid candidate path cannot be determined by reactive motion modification. We say that a structural local minimum has been encountered. The term structural local minimum refers to a local minimum in the overall potential function, arising only due to the particular configuration of the robot. Such a minimum could be avoided if the robot approached the narrow passage in some other configuration. As an example of a structural local minimum, consider an L-shaped robot passing through a narrow passage. If it enters the passage with one of its "legs" first it can pass by twisting through the narrow passage; if it enters the passage with the bend first, it will get stuck in a structural local minimum.

Such structural local minima are a result of the incompleteness of reactive motion modification as a search method in the configuration space. They do not represent a change the configuration space connectivity. To remedy the situation a planner has to be invoked. It can determine a path passing through the narrow passage in the configuration free space, avoiding the structural local minimum.

In addition to these two failures which can only be addressed by a global planning operation, the elastic strip framework can exhibit suboptimalities of the path, due to the fact that only local information is used to modify the path. Such suboptimalities can arise, for example, when an obstacle moves through the robot's path and continues to deform the path, even after the obstacle has crossed and passed the original trajectory. We consider these to be local minima in the optimality space of all paths, as opposed to local minima in the configuration space of the robot. Rather than preventing the robot from reaching the goal, they cause it to take a suboptimal path to the goal. Initially the candidate path represents a "good" trajectory, based on criteria established by the planning problem. With respect to these criteria the candidate path represents a local minimum close to or identical to the global optimum. As changes in the environment occur, the local minimum containing the candidate path might change to become significantly worse that the global optimum.

A two-dimensional example of such a situation is illustrated in Figure 10. Figure 10(a) shows a straight-line

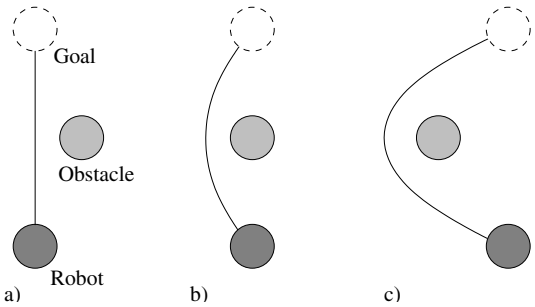



Fig. 10. A candidate path becomes suboptimal after the environment changes.

trajectory for the robot. In Figure 10(b) the path is modified by the approaching obstacle, but it still represents an optimal trajectory. Further motion of the obstacle causes the path to become suboptimal, as shown in Figure 10(c), where a topologically different straight-line trajectory becomes feasible. The original path would have to be transformed into a disconnected, suboptimal one, before it could reach the global maximum of the optimality space again.

This problem of suboptimality due to a passing obstacle can be resolved by the invocation of a global path planner to determine a new candidate path $\mathcal{P}_c$ for the elastic strip. But since such obstacle behavior is likely to occur frequently in dynamic environments, we would like to be able to address it within the framework itself.

In Section 4.3 the internal forces acting on consecutive configurations along the elastic strip were defined in order to keep internal tension and relative distance constant as the strip deforms. If the component of the repulsive force in the opposite direction of the internal force exceeds the magnitude of the external force, two adjacent configurations along the elastic strip start to separate. If this separation results in two disjunct components of the strip, the effect of repulsive forces at the separated configurations is suspended. The obstacle can then pass through the opening and internal forces will "repair" the strip, by joining the two disjunct pieces. Please note that this behavior is a direct consequence of the definition of internal and external forces in Section 4.3 and therefore represents behavior inherent to elastic strips. The process of splitting the elastic strip, letting an obstacle pass through it, and then merging it again is illustrated in Figure 11 (Extension 10).

If an obstacle comes to rest on the optimal path, the internal forces will not be able to reconnect the elastic strip during this procedure. By maintaining two versions of the elastic strip, one which is continuously modified to avoid the obstacle and one which is broken in an attempt to let it pass through, this difficulty can be avoided. This procedure guarantees that, as long as none of the aforementioned failure modes occurs, the elastic strip will represent a valid path, maintaining optimality criteria as closely as possible, while avoiding costly planning operations.

## 5. Implementation

In this section we discuss a particular implementation (Brock 2000) of the elastic strip framework, which was the basis for the experimental results presented in the previous section. Since most of the computational requirements arise from free space computation, special attention is paid to distance computation, rigid body representation, and the geometric criteria employed to determine the containment of a volume swept by the robot along a path within the elastic tunnel.

### *5.1. Rigid Body Representation*

To ensure collision avoidance, the elastic strip framework maintains a representation of free space, called the elastic tunnel, along the entire trajectory. Its computation requires the distance information between rigid bodies in space; as a result, distance computation is the most frequently executed algorithmic primitive in the elastic strip framework. The efficiency of this operation has a crucial impact on the overall performance. Various distance computation methods are presented in the literature (Gilbert, Johnson, and Keerthi 1988; Lin 1993; Mirtich 1997; Ong and Gilbert 1997). For this particular implementation of the elastic strip framework we have chosen a hierarchical bounding sphere method (Quinlan 1994a). The representation of rigid bodies described in this section was chosen to optimize the performance of distance computation using that method.

A rigid body can be approximated by a line segment with associated width, varying linearly along the line segment. We call such a parametrized line segment the *spine* of the rigid body. The width specifies the free space required around the line segment for the body to be free of collision. The volume described by a spine and its width parametrization is a generalized cylinder with caps on its circular faces. This representation is motivated by the resulting simplification of the distance computation; rather than computing the distance between two bodies, the distance between a line and a body is computed. Figure 12 shows the spine of a PUMA 560 link as the black line; the volume associated with the spine encloses the link.

This spine model is only a very coarse approximation of the rigid body. It can be assumed, however, that in most cases the robot will maintain a safe distance to obstacles. The computationally more efficient check against a coarse representation of its volume then suffices to ensure collision avoidance. As a rigid body comes closer to obstacles, the model described above might result in incorrect collision detection. To address this difficulty the representation of rigid bodies with spines can be generalized to describe rigid bodies at an arbitrary level of detail.

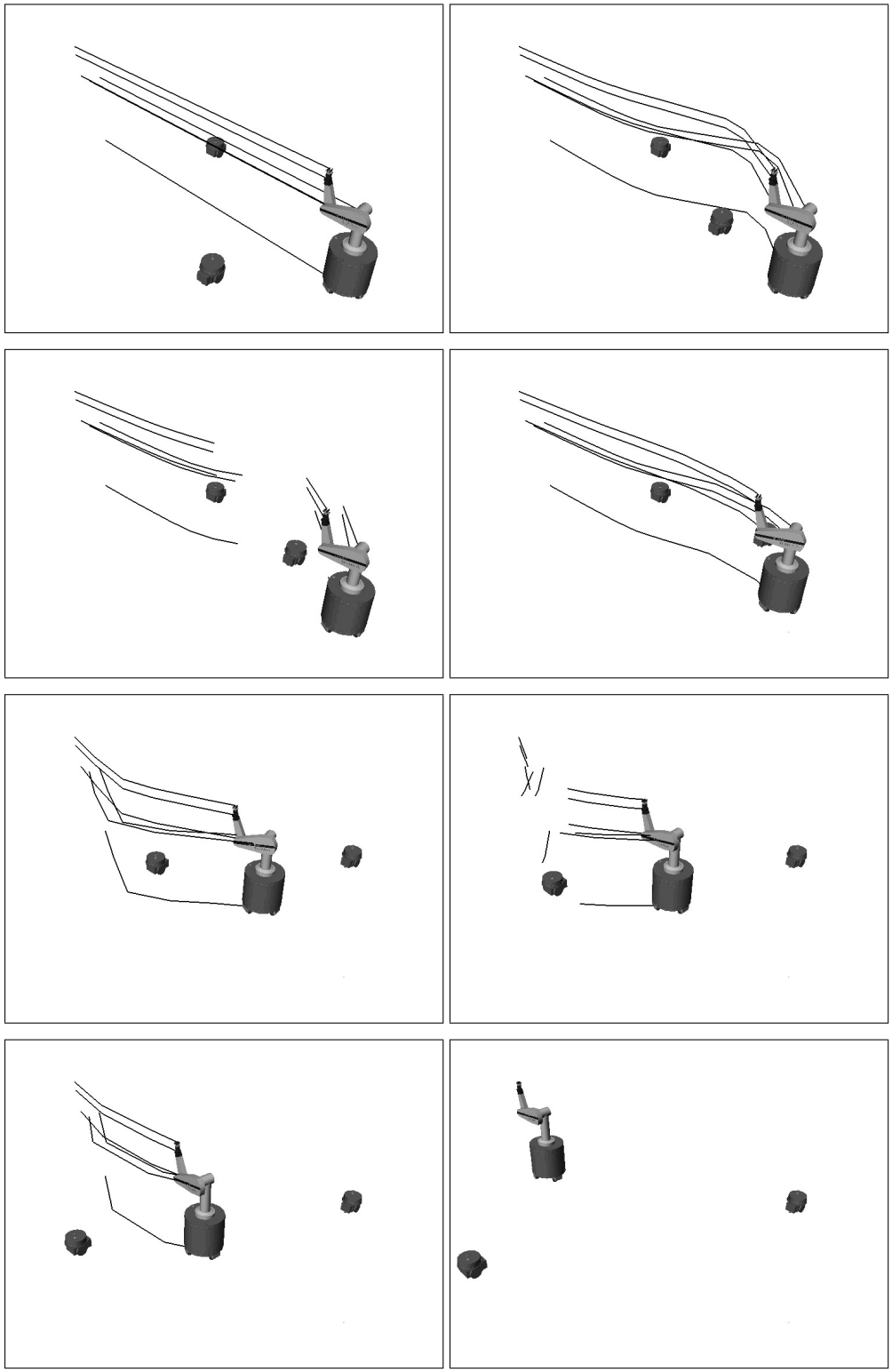


Fig. 11. This series of images shows how two moving obstacle pass through the elastic strip after a given local elongation is exceeded. The images show the elastic strip immediately before the split occurs, during the split, and immediately after the strip has been reconnected by internal forces.

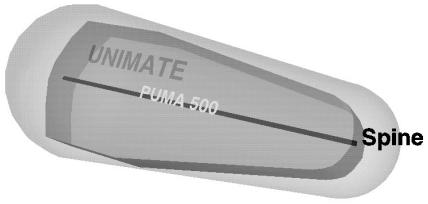



Fig. 12. Illustration of a spine and the associated volume approximating a rigid body. The spine is indicated by the black line along the principal axis of the PUMA 560 link. The transparent cloud indicates the width parametrization of the spine.

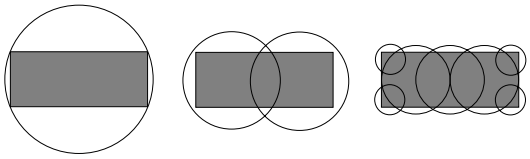


Fig. 13. Covering a body with an increasing number of spines. The figure shows a rigid body and associated hulls viewed along the spines. The number of spines and also the accuracy or representation increase from left to right.

The basic idea of this generalization is to introduce more spines to cover the volume of the rigid body with increasing accuracy. In Figure 13 the rectangular cross section of a body and its circular covering by spines are shown for different resolutions. The spines are situated at the centers of the circles orthogonal to the paper. The radius of the circle indicates the width parametrization.

For the sake of simplicity we assume in the remainder of this paper that the volume of rigid bodies is described by a single spine. The extension to a multi-spine representation is can be accomplished by examining every spine of the body individually.

### *5.2. Free Space Representation*

Given the representation of a rigid body, the elastic strip framework requires the computation and representation of the local free space around that body. The concept of a *bubble* as an efficiently computable representation of local free space was introduced in in conjunction with the elastic band framework (Quinlan 1994b). A bubble captures a spherical region of free space around a given point $p$. Let $d(p)$ be the function that computes the minimum distance from a point $p$ to any obstacle. The *workspace bubble* of free space around $p$ is defined as

$$\mathcal{B}(p) = \{ r : \| p - r \| < d(p) \}.$$

An approximation of the local free space around a rigid body $b$ in configuration $q$ can be computed by generating a set of overlapping workspace bubbles centered on the spine. This set of bubbles is called the *protective hull* $\mathcal{H}_q^b$. It can be computed by computing bubbles at the ends of the line segment associated with the spine and recursively subdividing the portion of the spine not enclosed by the bubbles. The protective hull narrows at the intersection of two adjacent bubbles. To accurately capture local free space around the rigid body, the procedure for computing the protective hull requires the width at such an intersection point to be close to the minimum radius of the adjacent bubbles.

The local free space or protective hull $\mathcal{H}_q$ of a robot $\mathcal{R}$ at a configuration $q$ is described by the union of protective hulls of each rigid body of $\mathcal{R}$,

$$\mathcal{H}_q = \bigcup_{b \in \mathcal{R}} \mathcal{H}_q^b.$$

Figure 14 shows an example of a protective hull for the Stanford Mobile Platform. Note that a single workspace bubble may contain multiple rigid bodies or even the entire robot, implying that for large clearances a simple description of the local free space suffices.

Given a candidate path $\mathcal{P}_c = (q_1, \ldots, q_n)$ of an elastic strip $\mathcal{S}$, the corresponding elastic tunnel $\mathcal{T}_{\mathcal{P}_c}$ is given by

$$\mathcal{T}_{\mathcal{P}_c} = \bigcup_{q_i} \mathcal{H}_{q_i}.$$

An elastic strip is considered *valid*, if the *containment condition*

$$V_{\mathcal{P}_c} \subseteq \mathcal{T}_{\mathcal{P}_c} = \bigcup_{q_i} \mathcal{H}_{q_i} \tag{8}$$

holds; otherwise it is considered invalid. The criterion to determine whether the containment condition holds or not is described in the next section.

### *5.3. Containment Criterion*

To determine if the volume $V_{\mathcal{P}_c}$ swept by a robot along the path $\mathcal{P}_c$ is contained within the corresponding tunnel $\mathcal{T}_{\mathcal{P}_c}$, it suffices to describe a procedure that verifies the existence of a path between two consecutive protective hulls $\mathcal{H}_i^{\mathcal{R}}$ and $\mathcal{H}_{i+1}^{\mathcal{R}}$ along the path. The repeated application of this procedure to all neighboring configurations $q_i$ and $q_{i+1}$ along the path $\mathcal{P}_c$ represents a criterion for the containment condition given by eq. (8).

To determine if the volume swept by a robot when transitioning from $q_i$ to $q_{i+1}$ is contained within the volume

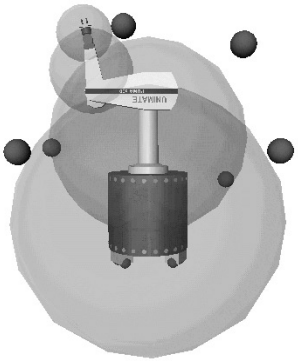


Fig. 14. Protective hull. The union of bubbles describing the local free space around the Stanford Mobile Platform makes up its protective hull. Spherical obstacles limit the size of the bubbles.

$\mathcal{H}_{q_i} \cup \mathcal{H}_{q_{i+1}}$, we make the assumption that every point on the robot moves on a straight line as it transitions from $q_i$ to $q_{i+1}$. This ignores the effect of rotation. However, this effect can be bounded and taken into account at computational expense, when computing the protective hull. This can be achieved by bounding the motion of all points caused by the rotation and including this bound as a safety margin in the free space computation. When the robot is close to an obstacle, two adjacent configurations on the elastic strip will be similar enough for the effect of rotation to be small. Note that this assumption represents simplification but not an inherent limitation of the approach. Using this assumption, the path of each rigid body $b$ of the robot $\mathcal{R}$ can be examined independently. If a trajectory between $q_i$ and $q_{i+1}$ exists for all rigid bodies $b \in \mathcal{R}$, one exists for $\mathcal{R}$.

The existence of a path $\mathcal{P}_{i,i+1}$ for a rigid body $b$ from configuration $q_i$ to $q_{i+1}$ is guaranteed if the volume $V^b_{\mathcal{P}_{i,i+1}}$ swept by $b$ along $\mathcal{P}_{i,i+1}$ is contained within the protective hulls of the configuration $q_i$ and $q_{i+1}$,

$$V^b_{\mathcal{P}_{i,i+1}} \subseteq \left(\mathcal{H}^b_i \cup \mathcal{H}^b_{i+1}\right). \tag{9}$$

To verify condition (9) the union $\mathcal{U} = \mathcal{H}^b_i \cup \mathcal{H}^b_{i+1}$ is examined. If $b$ can pass through $\mathcal{U}$ on a straight-line trajectory from $q_i$ to $q_{i+1}$, the existence of a trajectory $V^b_{\mathcal{P}_{i,i+1}}$ contained within $\mathcal{H}^b_i \cup \mathcal{H}^b_{i+1}$ is guaranteed. To determine if this is the case, we observe that the narrowest passage in passing from one protective hull to the next must lie at the intersection of three bubbles (two bubbles in boundary cases). The width of this intersection between all relevant triplets of bubbles can be determined and compared to the width of the rigid body. If the width of all such triplets exceeds the width of the rigid body at that location in space, the rigid body can pass from $\mathcal{H}^b_i$ to $\mathcal{H}^b_{i+1}$.

Relevant triplets of bubbles are determined by a linear traversal of the protective hulls $\mathcal{H}^b_i$ and $\mathcal{H}^b_{i+1}$. During this traversal a set of three intersecting bubbles is maintained. Each protective hull must contribute at least one bubble to this triplet. The closest bubble on each of the two spines is a candidate to be determined next; the one with the narrowest intersection is chosen and one of the previous bubbles can be deleted from the triplet. This procedure examines every bubble exactly once.

If for all rigid bodies $b \in \mathcal{R}$ the union of their protective hulls $\mathcal{H}^b_i \cup \mathcal{H}^b_{i+1}$ is large enough to allow a straight-line trajectory, we say that two consecutive protective hulls $\mathcal{H}^{\mathcal{R}}_i$ and $\mathcal{H}^{\mathcal{R}}_{i+1}$ are *connected*.

### 5.4. Motion Execution

The elastic strip represents a trajectory as a sequence of discrete configurations. These configurations are changed in real time by various forces, as described above. Before a trajectory can be executed on a robot, however, it needs to be converted into a trajectory, taking into account the current configuration of the robot and its actuation constraints. Since no time requirements are imposed on reaching the goal configuration, this can be accomplished by choosing an adequate time parametrization. The interpolation between two consecutive configurations along the elastic strip has to ensure that at no point an intermediate configuration is not entirely contained within the elastic tunnel. Appropriate methods for trajectory generation have been developed (Brock and Khatib 1999).

When task execution is combined with other motion behavior, the discrete configurations along the elastic strip are modified in a task-consistent manner. Configurations obtained by interpolating between two task-consistent configurations, however, will not necessarily be task-consistent themselves. This can be addressed by mapping the resulting interpolated trajectory into the nullspace of the task, as described in eqs. (3). The control structure will ensure that only motions not affecting task behavior are executed. This approach also smoothes the nullspace motion and specifies a behavior for those degrees of freedom which are neither required to perform the task nor to perform the desired additional behavior.

## 6. Conclusion

The applications of robotic technology are beginning to extend beyond the assembly line into unstructured, dynamic, and populated environments. Robotic systems capable of performing complex tasks in such environments generally require high kinematic complexity, as evidenced by humanoid robots, posing new challenges in the field of robot motion generation and control. We presented the elastic strip framework, which allows the integration of global motion planning, reactive obstacle avoidance, and a task-based control formulation. Using

this framework, globally planned motion for robots with many degrees of freedom can be modified in real time in reaction to changes in the environment, thus enabling the robust execution of collision-free motion in dynamic environments. The generation of this motion can be guided by arbitrary posture behavior to optimize aspects of the resulting motion, such as maintaining stability, avoiding singularities, or preventing self-collision.

Desired task behavior may impose constraints on the motion of the end-effector. The elastic strip framework allows obstacle avoidance and posture behavior to be performed without violating these constraints. This property, called task consistency, guarantees that desired motion behavior does not interfere with task execution. To accomplish this, motion behavior is performed in the nullspace of the task. Should the mechanism, due to obstacle movement, kinematic limitations, or other constraints, become incapable of executing this behavior, the elastic strip framework can automatically suspend task behavior. Execution of the task is resumed, once the prohibiting constraints have been removed.

Experimental applications of the elastic strip framework to the nine-degree-of-freedom Stanford Robotic Platform and to a simulated 34-degree-of-freedom humanoid robot have been presented. Obstacle avoidance, posture behavior, task-consistent obstacle avoidance, and task suspension and resumption have been demonstrated in simulation and on real robots. The results illustrate the effectiveness of the elastic strip framework as a general and powerful task-oriented approach to motion generation for robots with many degrees of freedom, such as humanoid or human-like structures, in dynamic and unstructured environments.

## Appendix: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org.

**Table of Multimedia Extensions**

| Extension | Type | Description |
|---|---|---|
| 1 | Video | Demonstration of real-time path modification using elastic strips. The trajectory of a nine-degree-of-freedom mobile manipulator (left) is modified in real time as another mobile manipulator (bottom) moves into its path. The resulting motion employs all nine degrees of freedom to accomplish obstacle avoidance, as demonstrated by the arm motion. |
| 2 | Video | This video shows the experiment from Extension 1 executed on a real robot. |
| 3 | Video | The trajectory of a 34-degree-of-freedom humanoid robot is modified in real time, as it moves under a lowering beam. The video shows motion modification without posture control, leading to physically infeasible motion. |
| 4 | Video | This video shows the same experiment as Extension 4 with posture control. The resulting motion is not only more natural in its appearance, but also physically feasible. |
| 5 | Video | Real-time path modification can be applied to character animation. Observe how the skiing robot with 34 degrees of freedom avoids the moving snowman and the lowering finish banner. Note how the ski poles are moved to avoid obstacles and how a natural posture is maintained with posture control. |
| 6 | Video | For this experiment, the task consists of following the red line with the end-effector. Obstacle avoidance is accomplished using elastic strips and causes the end-effector to deviate significantly from the task. |
| 7 | Video | This experiment is the same as in Extension 7, except that the elastic strip performs obstacle avoidance in a task-consistent manner. You can see how the end-effector continuously performs the task. The obstacles perform the exact same motion as in the previous experiment. |
| 8 | Video | This segment shows the experiment of Extension 8 executed on the real robot. The end-effector task consists of following the red beam. Obstacle avoidance is accomplished using elastic strips in a task-consistent manner. The obstacle is perceived using a laser range finder. The last segment of this video shows task-consistent obstacle avoidance for a different task: a bottle of water is placed on a tray |

| | | |
|---|---|---|
| | | and the task consists of keeping the tray level during obstacle avoidance. |
| 9 | Video | When constraints render task-consistent obstacle avoidance infeasible, the task can be suspended and subsequently resumed, as demonstrated in this video. The task consists again of following a straight-line trajectory with the end-effector. The second mobile robot moves too far into the path of the mobile manipulator for the task to be maintained. It is suspended and resumed, once the obstacle is passed. |
| 10 | Video | Purely reactive obstacle avoidance can result in highly suboptimal paths. This video shows a simple method of local replanning to avoid the suboptimalities. |

## Acknowledgments

The financial support of Honda Motor Company, Boeing Company, General Motors, Nomadic Technologies, Inc., and NSF (grant IRI-9320017) is gratefully acknowledged. The authors would like to express their gratitude to Sriram Viji for assisting with parts of the implementation, to Kyong-Sok Chang and Bob Holmberg for providing the operational space controller for the Stanford Robotic Platform, to Costas Stratelos for providing software for the SICK laser range finder, to Luis Sentis, Francois Conti, and James Warren for their valuable contributions to the work reported here, and to the reviewers for their helpful comments and suggestions.

## References

Adams, B., Breazeal, C., Brooks, R., and Scassellati, B. 2000. Humanoid robots: A new kind of tool. *IEEE Intelligent Systems* 15(4):25–31.

Amato, N., Bayazid, B., Dale, L., Jones, C., and Vallejo, D. 1998. OBPRM: An obstacle-based PRM for 3D workspaces. In *Robotics: The Algorithmic Perspective*. AK Peters.

Asfour, T., Berns, K., and Dillmann, R. 2000. The humanoid robot ARMAR: Design and control. In *Proceedings of the International Conference on Humanoid Robots*, Cambridge, USA.

Baginski, B. 1998. Motion Planning for Manipulators with Many Degrees of Freedom—The BB Method. Ph.D. thesis, Technische Universität München.

Barraquand, J., and Latombe, J.-C. 1991. Robot motion planning: A distributed representation approach. *International Journal of Robotics Research* 10(6):628–649.

Bayle, B., Fourquet, J.-Y., and Renaud, M. 2000. Generalized path generation for a mobile manipulator. In *Proceedings of the International Conference on Mechanical Design and Production*, Cairo, Egypt, pp. 57–66.

Bischoff, R., and Graefe, V. 1999. Integrating vision, touch and natural language in the control of a situation-oriented behavior-based humanoid robot. In *Proceedings of the International Conference on Systems, Man, and Cybernetics*, Vol. 2, pp. 999–1004.

Bohlin, R. 2001. Path planning in practice: Lazy evaluation on a multi-resolution grid. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Maui, USA, pp. 49–54.

Bohlin, R., and Kavraki, L. E. 2000. Path planning using lazy PRM. In *Proceedings of the International Conference on Robotics and Automation*, San Francisco, USA, Vol. 1, pp. 521–528.

Brock, O. 2000. Generating Robot Motion: The Integration of Planning and Execution. Ph.D. thesis, Stanford University, Stanford University, USA.

Brock, O., and Kavraki, L. E. 2001. Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces. In *Proceedings of the International Conference on Robotics and Automation*, Seoul, Korea, pp. 1469–1474.

Brock, O., and Khatib, O. 1997. Elastic strips: Real-time path modification for mobile manipulation. In *Proceedings of the International Symposium of Robotics Research*, pp. 117–122.

Brock, O., and Khatib, O. 1999. Elastic strips: A framework for integrated planning and execution. In P. Corke and J. Trevelyan (Eds.), *Proceedings of the International Symposium on Experimental Robotics, Volume 250 of Lecture Notes in Control and Information Sciences*, pp. 328–338. Berlin: Springer Verlag.

Brooks, R. 1997. From earwigs to humans. *Robotics and Autonomous Systems* 20(2–4):291–304.

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

Chang, K.-S., and Khatib, O. 2000. Operational space dynamics: Efficient algorithms for modelling and control of branching mechanisms. In *Proceedings of the International Conference on Robotics and Automation*, San Francisco, USA, pp. 850–856.

Choi, W., and Latombe, J.-C. 1991. A reactive architecture for planning and executing robot motions with incomplete knowledge. In *Proceedings of the International Conference on Intelligent Robots and Systems* Vol. 1, pp. 24–29.

Desai, J. P., and Kumar, V. 1997. Nonholonomic motion planning for multiple mobile robots. In *Proceedings of the*

*International Conference on Robotics and Automation*, Albuquerque, USA, Vol. 3409–3414.

Faverjon, B., and Tournassoud, P. 1987. A local based approach for path planning of manipulators with a high number of degrees of freedom. In *Proceedings of the International Conference on Robotics and Automation* Vol. 2, pp. 1152–1159.

Gilbert, E. G., Johnson, D. W., and Keerthi, S. S. 1988. A fast procedure for computing the distance between complex objects in three-dimensional space. *International Journal of Robotics and Automation* 4(2):193–203.

Hsu, D., Kavraki, L. E., Latombe, J.-C., and Motwani, R. 1999. Capturing the connectivity of high-dimensional geometric spaces by parallelizable random sampling techniques. In P. M. Pardalos and S. Rajasekaran (eds.), *Advances in Randomized Parallel Computing*, pp. 159–182. Dordrecht: Kluwer Academic.

Huang, Q., Sugano, S., and Tanie, K. 1998. Motion planning for a mobile manipulator considering stability and task constraints. In *Proceedings of the International Conference on Robotics and Automation*, Leuven, Belgium, pp. 2192–2198.

Inoue, K., Yoshida, H., Arai, T., and Mae, Y. 2000. Mobile manipulation of humanoids—real-time control based on manipulability and stability. In *Proceedings of the International Conference on Robotics and Automation*, San Francisco, USA.

Kagami, S., Nishiwaki, K., Sugihara, T., Kuffner, J., Inaba, M., and Inoue, H. 2001. Design and implementation of software research platform for humanoid robotics: H6. In *Proceedings of the International Conference on Robotics and Automation*, Seoul, Korea, pp. 2431–2436.

Kavraki, L. E., Švestka, P., Latombe, J.-C., and Overmars, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4):566–580.

Khatib, M. 1996. Sensor-Based Motion Control for Mobile Robots. Ph.D. thesis, LAAS-CNRS, Toulouse, France.

Khatib, O. 1986. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research* 5(1):90–98.

Khatib, O. 1987. A unified approach to motion and force control of robot manipulators: The operational space formulation. *International Journal of Robotics and Automation* 3(1):43–53.

Khatib, O. 1995. Inertial properties in robotics manipulation: An object-level framework. *International Journal of Robotics Research* 14(1):19–36.

Khatib, O., Brock, O., Chang, K.-S., Ruspini, D., Sentis, L., and Viji, S. 2001. Human-centered robotics and interactive haptic simulation. In *Proceedings of the International Symposium of Robotics Research*. Preprints.

Khatib, O., Yokoi, K., Brock, O., Chang, K.-S., and Casal, A. 1999. Robots in human environments: Basic autonomous capabilities. *International Journal of Robotics Research* 18(7):684–696.

Khatib, O., Yokoi, K., Chang, K.-S., Ruspini, D., Holmberg, R., and Casal, A. 1996. Coordination and decentralized cooperation of multiple mobile manipulators. *Journal of Robotic Systems* 13(11):755–764.

Kuffner, J. J., Nishiwaki, K., Kagami, S., Inaba, M., and Inoue, H. 2001. Motion planning for humanoid robots under obstacle and dynamic balance constraints. In *Proceedings of the International Conference on Robotics and Automation*, Seoul, Korea, pp. 692–698.

Latombe, J.-C. 1991. *Robot Motion Planning*. Boston: Kluwer Academic.

LaValle, S. M. and Kuffner, J. J. 1999. Randomized kinodynamic planning. In *Proceedings of the International Conference on Robotics and Automation*, Detroit, USA.

LaValle, S. M., Yakey, J. H., and Kavraki, L. E. 1999. A probabilistic roadmap approach for systems with closed kinematic chains. In *Proceedings of the International Conference on Robotics and Automation*, Detroit, USA, pp. 1671–1675.

Lee, J. and Shin, S. Y. 1999. A hierarchical approach for interactive motion editing for human-like figures. In *Proceedings of SIGRAPH* pp. 39–48.

Lin, M. C. 1993. Efficient Collision Detection for Animation and Robotics. Ph.D. thesis, University of California at Berkeley.

McLean, A. and Cameron, S. 1996. The virtual springs method: Path planning and collision avoidance for redundant manipulators. *International Journal of Robotics Research* 15(4):300–319.

Mirtich, B. 1997. V-clip: Fast and robust polyhedral collision detection. Technical Report TR-97-05, Mitsubishi Electric Research Laboratory (MERL).

Mohri, A., Furuno, S., and Yamamoto, M. 2001. Trajectory planning of mobile manipulator with end-effector's specified path. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Maui, USA, pp. 2264–2269.

Ögren, P., Egerstedt, M., and Hu, X. 2000. Reactive mobile manipulation using dynamic trajectory tracking. In *Proceedings of the International Conference on Robotics and Automation*, San Francisco, USA, pp. 3473–3478.

Ong, C. J. and Gilbert, E. G. 1997. The Gilbert–Johnson–Keerthi distance algorithm: A fast version for incremental motions. In *Proceedings of the International Conference on Robotics and Automation*, Vol. 2, pp. 1183–1189.

Perrier, C., Dauchez, P., and Perrot, F. 1998. A global approach for motion generation of non-holonomic mobile manipulators. In *Proceedings of the International Conference on Robotics and Automation*, Leuven, Belgium, pp. 2971–2976.

Quinlan, S. 1994a. Efficient distance computation between non-convex objects. In *Proceedings of the International*

*Conference on Robotics and Automation* Vol. 4, pp. 3324–3329.

Quinlan, S. 1994b. Real-Time Modification of Collision-Free Paths. Ph.D. thesis, Stanford University.

Russakow, J., Khatib, O., and Rock, S. M. 1995. Extended operational space formation for serial-to-parallel chain (branching) manipulators. In *Proceedings of the International Conference on Robotics and Automation* Vol. 1, pp. 1056–1061.

Seraji, H. 1993. An on-line approach to coordinated mobility and manipulation. In *Proceedings of the International Conference on Robotics and Automation* Vol. 1, pp. 28–35.

Steele, J. P. H. and Starr, G. P. 1988. Mobile robot path planning in dynamic environments. In *Proceedings of the International Conference on Systems, Man, and Cybernetics* Vol. 2, pp. 922–925.

Tan, J. and Xi, N. 2001. Unified model approach for planning and control of mobile manipulators. In *Proceedings of the International Conference on Robotics and Automation*, Seoul, Korea, pp. 3145–3152.

Tanner, H. G. and Kyriakopoulos, K. J. 2000. Nonholonomic motion planning for mobile manipulators. In *Proceedings of the International Conference on Robotics and Automation*, San Francisco, USA, pp. 1233–1238.

Vukobratović, M., Frank, A. A., and Juričić, D. 1970. On the stability of biped locomotion. *IEEE Transactions on Biomedical Engineering* 17(1):25–36.

Warren, C. W. 1989. Global path planning using artificial potential fields. In *Proceedings of the International Conference on Robotics and Automation* Vol. 1, pp. 316–321.

Yamamoto, Y. and Yun, X. 1994. Coordinating locomotion and manipulation of a mobile manipulator. *IEEE Transactions on Automatic Control* 39(36):1326–1332.

Yamamoto, Y. and Yun, X. 1995. Coordinated obstacle avoidance of a mobile manipulator. In *Proceedings of the International Conference on Robotics and Automation* Vol. 3, pp. 2255–2260.

Zhao, J. and Badler, N. 1994. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics* 13(4):313–336.