Chapter 7

Potential Field Methods

The planning methods described in the previous three chapters aim at capturing the global connectivity of the robot's free space into a condensed graph that is subsequently searched for a path. The approach presented in this chapter proceeds from a different idea. It treats the robot represented as a point in configuration space as a particle under the influence of an *artificial potential field* U whose local variations are expected to reflect the "structure" of the free space. The potential function is typically (but not necessarily) defined over free space as the sum of an *attractive* potential pulling the robot toward the goal configuration and a *repulsive* potential pushing the robot away from the obstacles. Motion planning is performed in an iterative fashion. At each iteration, the artificial force $\vec{F}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}(\mathbf{q})$ induced by the potential function at the current configuration is regarded as the most promising direction of motion, and path generation proceeds along this direction by some increment.

Potential field was originally developed as an on-line collision avoidance approach, applicable when the robot does not have a prior model of the obstacles, but senses them during motion execution [Khatib, 1980

and 1986]. Emphasis was put on real-time efficiency, rather than on guaranteeing the attainment of the goal. In particular, since an on-line potential field method essentially acts as a fastest descent optimization procedure, it may get stuck at a local minimum of the potential function other than the goal configuration. However, the idea underlying potential field can be combined with graph searching techniques. Then, using a prior model of the workspace, it can be turned into a systematic motion planning approach. This is the approach described in this chapter.

Even when graph searching techniques are used, local minima remain an important cause of inefficiency for potential field methods. Hence, dealing with local minima is *the* major issue that one has to face in designing a planner based on this approach. This issue can be addressed at two levels: (1) in the definition of the potential function, by attempting to specify a function with no or few local minima, and (2) in the design of the search algorithm, by including appropriate techniques for escaping from local minima. A large portion of this chapter describes results and techniques developed at these two levels.

Potential field methods are often referred to as "local methods". This comes from the fact that most potential functions are defined in such a way that their values at any configuration do not depend on the distribution and shapes of the C-obstacles beyond some limited neighborhood around the configuration. However, as mentioned in Chapter 1, if one could construct an "ideal" potential function with a single minimum (at the goal configuration) in the connected subset of the free space containing the goal configuration, this function could be regarded as some kind of "global" information about free space, and the expression "local method" would then be much less relevant.

Most planning methods based on the potential field approach have a strong empirical flavor. They are usually incomplete, i.e. they may fail to find a free path, even if one exists. But, the counterpart is that some of them are particularly fast in a wide range of situations. The strength of the approach is that, with some limited engineering, it makes it possible to construct motion planners which are both quite efficient and reasonably reliable. This explains why they are increasingly popular for implementing practical motion planners.

In Sections 1 and 2 we give a classical definition of a potential function and the induced field of forces. We first consider the case where the robot can only translate (Section 1); then, we treat the case where it can also rotate (Section 2). In Section 3 we describe three simple potential-guided path planning techniques: depth-first, best-first and variational planning. In Section 4 we introduce the notion of *navigation function* (local-minimum-free potential) and we define potential functions that may generate less or smaller local minima than the function of Sections 1 and 2. In Section 5 we combine results presented in previous sections with randomized search techniques for escaping from local minima; we describe a powerful potential-field-based planner that is able to plan free paths in high-dimensional configuration spaces. In Section 6 we describe a simple learning technique that may be used to improve the ability of a potential-field-based planning method to avoid local minima when the geometry of free space remains unchanged over many successive tasks.

1 Potential Field in Translational Case

Most proposed potential functions are based upon the following general idea: the robot should be attracted toward its goal configuration, while being repulsed by the obstacles. In this section we illustrate this idea with the definition of one possible potential function, in the case where $\mathcal{A}$ translates freely at fixed orientation in $\mathcal{W} = \mathbf{R}^N$, with $N = 2$ or 3, i.e. $\mathcal{C} = \mathbf{R}^N$. In the next section we will adapt this definition to the case where $\mathcal{A}$ can both translate and rotate. The function defined below is used in several implemented systems (e.g. see [Khatib, 1986]).

1.1 General Structure

The field of artificial forces $\vec{F}(\mathbf{q})$ in $\mathcal{C}$ is produced by a differentiable potential function $\mathbf{U} : \mathcal{C}_{free} \rightarrow \mathbf{R}$, with:

$$\vec{F}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}(\mathbf{q})$$

where $\vec{\nabla}\mathbf{U}(\mathbf{q})$ denotes the gradient vector of $\mathbf{U}$ at $\mathbf{q}$. In $\mathcal{C} = \mathbf{R}^N$ ($N = 2$ or 3), we can write $\mathbf{q} = (x, y)$ or (x, y, z), and:

$$\vec{\nabla}\mathbf{U} = \begin{pmatrix} \partial\mathbf{U}/\partial x \\ \\ \partial\mathbf{U}/\partial y \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} \partial\mathbf{U}/\partial x \\ \\ \partial\mathbf{U}/\partial y \\ \\ \partial\mathbf{U}/\partial z \end{pmatrix}.$$

In order to make the robot be attracted toward its goal configuration, while being repulsed from the obstacles, U is constructed as the sum of two more elementary potential functions:

$$\mathbf{U}(\mathbf{q}) = \mathbf{U}_{att}(\mathbf{q}) + \mathbf{U}_{rep}(\mathbf{q})$$

where $\mathbf{U}_{att}$ is the **attractive potential** associated with the goal configuration $\mathbf{q}_{goal}$ and $\mathbf{U}_{rep}$ is the **repulsive potential** associated with the C-obstacle region. $\mathbf{U}_{att}$ is independent of the C-obstacle region, while $\mathbf{U}_{rep}$ is independent of the goal configuration.

With these conventions, $\vec{F}$ is the sum of two vectors:

$$\vec{F}_{att} = -\vec{\nabla}\mathbf{U}_{att} \quad \text{and} \quad \vec{F}_{rep} = -\vec{\nabla}\mathbf{U}_{rep}$$

which are called the **attractive** and the **repulsive forces**, respectively.

1.2 Attractive Potential

The attractive potential field $\mathbf{U}_{att}$ can be simply defined as a parabolic well, i.e. :

$$\mathbf{U}_{att}(\mathbf{q}) = \frac{1}{2}\,\xi\,\rho^2_{goal}(\mathbf{q})$$

where ξ is a positive scaling factor and $\rho_{goal}(\mathbf{q})$ denotes the Euclidean distance $\|\mathbf{q} - \mathbf{q}_{goal}\|$. The function $\mathbf{U}_{att}$ is positive or null, and attains its minimum at $\mathbf{q}_{goal}$, where $\mathbf{U}_{att}(\mathbf{q}_{goal}) = 0$.

The function ρ_{goal} is differentiable everywhere in $\mathcal{C}$. At every configuration $\mathbf{q}$, the artificial attractive force $\vec{F}_{att}$ deriving from $\mathbf{U}_{att}$ is:

$$\begin{aligned}\vec{F}_{att}(\mathbf{q}) &= -\vec{\nabla}\mathbf{U}_{att}(\mathbf{q}) \\ &= -\xi\,\rho_{goal}(\mathbf{q})\vec{\nabla}\rho_{goal}(\mathbf{q}) \\ &= -\xi\,(\mathbf{q} - \mathbf{q}_{goal}).\end{aligned}$$

When used for on-line collision avoidance, the parabolic well has good stabilizing characteristics [Khatib, 1986] [Khosla and Volpe, 1988], since it generates a force $\vec{F}_{att}$ that converges linearly toward 0 when the robot's configuration gets closer to the goal configuration. (Asymptotic stabilization of the robot can be achieved by adding dissipative forces proportional to the velocity $\dot{\mathbf{q}}$.)

However, the parabolic-well attractive potential generates a force that augments with the distance to the goal configuration and tends toward infinity when $\rho_{goal}(\mathbf{q}) \to \infty$. Alternatively, one may define $\mathbf{U}_{att}$ as a conic well, i.e. :

$$\mathbf{U}_{att}(\mathbf{q}) = \xi\rho_{goal}(\mathbf{q}).$$

Then, the resulting attractive force is:

$$\begin{aligned}\vec{F}_{att}(\mathbf{q}) &= -\xi\vec{\nabla}\rho_{goal}(\mathbf{q}) \\ &= -\xi(\mathbf{q} - \mathbf{q}_{goal}) \,/\, \|\mathbf{q} - \mathbf{q}_{goal}\|.\end{aligned}$$

The amplitude of $\vec{F}_{att}(\mathbf{q})$ is constant over $\mathcal{C}$, except at $\mathbf{q}_{goal}$, where $\mathbf{U}_{att}$ is singular. Since the amplitude of the force does not tend toward 0 when $\mathbf{q} \to \mathbf{q}_{goal}$, the conic-well potential does not have the stabilizing characteristics of the parabolic-well function.

One way to combine the advantages of both the parabolic and the conic wells is to define the attractive potential as a parabolic well within some distance from the goal configuration and a conic well beyond that distance, with a defined derivative at their juncture.

In the rest of the chapter we will exclusively use parabolic-well attractive potentials (whenever we construct the potential as the sum of an attractive and a repulsive potential).

1.3 Repulsive Potential

The main idea underlying the definition of the repulsive potential is to create a potential barrier around the C-obstacle region that cannot be traversed by the robot's configuration. In addition, it is usually desirable that the repulsive potential not affect the motion of the robot when it is sufficiently far away from the C-obstacles. One way to achieve these constraints is to define the repulsive potential function as follows:

$$\mathbf{U}_{rep}(\mathbf{q}) = \begin{cases} \frac{1}{2}\eta(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0})^2 & \text{if } \rho(\mathbf{q}) \le \rho_0, \\ 0 & \text{if } \rho(\mathbf{q}) > \rho_0, \end{cases}$$

where η is a positive scaling factor, $\rho(\mathbf{q})$ denotes the distance from $\mathbf{q}$ to the C-obstacle region $\mathcal{CB}$, i.e. :

$$\rho(\mathbf{q}) = \min_{\mathbf{q}' \in \mathcal{CB}} \|\mathbf{q} - \mathbf{q}'\|,$$

and ρ_0 is a positive constant called the **distance of influence** of the C-obstacles. The function $\mathbf{U}_{rep}$ is positive or null, tends to infinity as $\mathbf{q}$ gets closer to the C-obstacle region, and is null when the distance of the robot's configuration to the C-obstacle region is greater than ρ_0.

If $\mathcal{CB}$ is a convex region with a piecewise differentiable boundary, ρ is differentiable everywhere in $\mathcal{C}_{free}$. The artificial repulsive force deriving from $\mathbf{U}_{rep}$ is:

$$\begin{aligned} \vec{F}_{rep}(\mathbf{q}) &= -\vec{\nabla}\mathbf{U}_{rep}(\mathbf{q}) \\ &= \begin{cases} \eta(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0})\frac{1}{\rho^2(\mathbf{q})}\vec{\nabla}\rho(\mathbf{q}) & \text{if } \rho(\mathbf{q}) \leq \rho_0, \\ 0 & \text{if } \rho(\mathbf{q}) > \rho_0. \end{cases} \end{aligned}$$

Let $\mathbf{q}_c$ be the unique configuration in $\mathcal{CB}$ that is closest to $\mathbf{q}$, i.e. that achieves $\|\mathbf{q} - \mathbf{q}_c\| = \rho(\mathbf{q})$. The gradient $\vec{\nabla}\rho(\mathbf{q})$ is a unit vector pointing away from $\mathcal{CB}$ and supported by the line passing through $\mathbf{q}_c$ and $\mathbf{q}$.

If we retract the (unrealistic) assumption that $\mathcal{CB}$ is convex, $\rho(\mathbf{q})$ remains differentiable everywhere in $\mathcal{C}_{free}$, except at those configurations $\mathbf{q}$ for which there exist several $\mathbf{q}_c \in \mathcal{CB}$ verifying $\|\mathbf{q} - \mathbf{q}_c\| = \rho(\mathbf{q})$. These configurations form a set of measure zero in $\mathcal{C}$ which is in general locally $(N-1)$-dimensional. The force field $\vec{F}_{rep}$ is defined on both sides of this set, but with differently oriented vector values. This may result in producing paths that oscillate between the two sides of the set.

One way to eliminate this difficulty is to decompose $\mathcal{CB}$ into (possibly overlapping) convex components $\mathcal{CB}_k$, $k = 1, ..., r$, and to associate a repulsive potential $\mathbf{U}_{\mathcal{CB}_k}$ with each component. Then, the repulsive potential is the sum of the repulsive potential fields created by each individual C-obstacle, i.e. :

$$\mathbf{U}_{rep}(\mathbf{q}) = \sum_{k=1}^{r} \mathbf{U}_{\mathcal{CB}_k}(\mathbf{q})$$

with:

$$\mathbf{U}_{\mathcal{CB}_k}(\mathbf{q}) = \begin{cases} \frac{1}{2}\eta(\frac{1}{\rho_k(\mathbf{q})} - \frac{1}{\rho_0})^2 & \text{if } \rho_k(\mathbf{q}) \leq \rho_0, \\ 0 & \text{if } \rho_k(\mathbf{q}) > \rho_0, \end{cases}$$

where $\rho_k(\mathbf{q})$ denotes the distance from $\mathbf{q}$ to $\mathcal{CB}_k$.

The artificial repulsive force deriving from $\mathbf{U}_{rep}$ is:

$$\vec{F}_{rep}(\mathbf{q}) = \sum_{k=1}^{r} \vec{F}_{\mathcal{CB}_k}(\mathbf{q})$$

with:

$$\vec{F}_{\mathcal{CB}_k}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}_{\mathcal{CB}_k}(\mathbf{q}).$$

The $\mathcal{CB}_k$'s may be constructed by first decomposing $\mathcal{A}$ and the workspace obstacles into convex components, and then computing the C-obstacle corresponding to every pair consisting of a component of $\mathcal{A}$ and a component of an obstacle.

The drawback of decomposing the C-obstacle region is that several small components that are close to each other arbitrarily produce a combined repulsive force which is larger than the one produced by a single bigger component. One empirical way to deal with this drawback is to weigh each individual potential by a coefficient depending on the "size" of the C-obstacle component that generates it.

Figure 1 illustrates the above definitions in a two-dimensional configuration space containing two C-obstacles (Figure 1.a). The attractive potential is a parabolic well with its minimum at the goal configuration $\mathbf{q}_{goal}$ (Figure 1.b). The repulsive potential is displayed in Figure 1.c. Since the two C-obstacles are sufficiently far apart from each other, both definitions given above yield the same function. This function tends toward infinity near the C-obstacles' boundary; hence, it is truncated in the figure. The total potential is displayed in Figure 1.d and some of its equipotential contours are shown in Figure 1.e together with a path connecting $\mathbf{q}_{init}$ to $\mathbf{q}_{goal}$. A matrix showing the discretized negated gradient vector of the total potential over free space is shown in Figure 1.f.

In the above definitions of $\mathbf{U}_{rep}(\mathbf{q})$, we use a unique scaling factor η and a unique distance of influence ρ_0. However, there is no difficulty in using different parameters η and ρ_0 for various subsets of the C-obstacle region. In particular, if the goal configuration $\mathbf{q}_{goal}$ is close to $\mathcal{CB}$, the influence distance ρ_0 should be set to a value smaller than the distance from $\mathbf{q}_{goal}$ to $\mathcal{CB}$ for the subset of $\mathcal{CB}$ that is located in the surrounding of $\mathbf{q}_{goal}$, so that the robot is not prevented from achieving $\mathbf{q}_{goal}$ by the

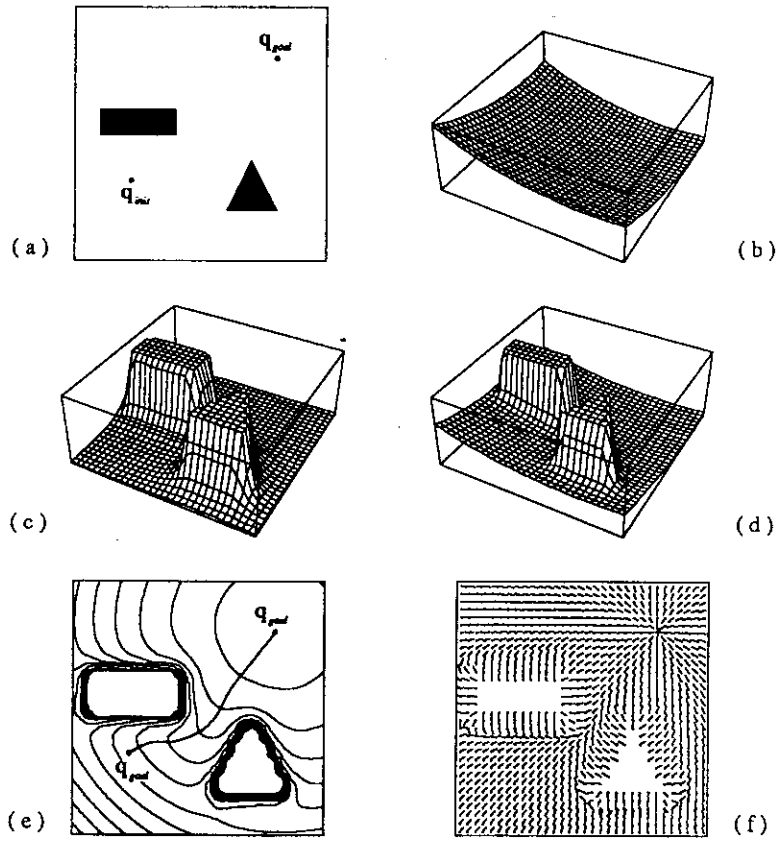

Figure 1. This figure shows an attractive potential field (Figure b), a repulsive potential field (Figure c) and the sum of the two (Figure d) in a two-dimensional configuration space containing two C-obstacles (Figure a). Figure e displays both several equipotential contours of the total potential and a path generated by following the negated gradient of this function. Figure f shows a matrix of the negated gradient vector orientations over free space.

repulsive potential field. (If the repulsive potential is non-zero at $\mathbf{q}_{goal}$, then in general the global minimum of the total potential function is not $\mathbf{q}_{goal}$.) It can be set to a greater value for the rest of $\mathcal{CB}$. From another perspective, if a specific workspace obstacle is known to be more "threatening" than others, it may be appropriate to use larger values of

η and ρ_0 for the corresponding C-obstacle.

The repulsive potential $\mathbf{U}_{\mathcal{CB}}(\mathbf{q})$ and the associated force field $\vec{F}_{\mathcal{CB}}(\mathbf{q})$ can easily be computed when the C-obstacle region is a polygonal or a polyhedral region. Then, if an explicit representation of $\mathcal{CB}$'s boundary has been previously computed, the computation can be done in time $O(n)$, with n being the complexity of $\mathcal{CB}$. Repulsive potential and forces are also easy to compute for some other C-obstacles' shapes (e.g. spheres). In more general cases, however, there may exist no straightforward algorithm. When the C-obstacle region can be decomposed into convex components $\mathcal{CB}_k$, each described as a conjunction of inequalities $f_j^{(k)}(\mathbf{q}) \leq 0$, $j = 1, 2, ...$, the problem of computing $\rho_k(\mathbf{q})$ is a convex optimization problem which can be solved using iterative numerical techniques.

Remark: In two dimensions, the potential field approach with the potential defined as above can be loosely related to retraction (see Section 2 of Chapter 4). The valleys of the repulsive potential form the Voronoi diagram of the free space. The attractive potential acts as a heuristic function guiding the search of the diagram. However, since the two potentials are added in the total potential function, the potential field approach does not restrict the robot to moving along the standardized paths formed by the arcs of the Voronoi diagram. ∎

2 Potential Field in General Case

This section considers the case where $\mathcal{C}$ is the manifold $\mathbf{R}^N \times SO(N)$. The content of the section is not essential to the understanding of the rest of the chapter and may be skipped in a first reading. In particular, the reader who wishes to get an immediate look at potential-guided planning methods can jump to the next section, and later come back to the present section.

2.1 Extension of Previous Notions

The notions introduced in the previous section can be extended to the general case where $\mathcal{C} = \mathbf{R}^N \times SO(N)$. We must however pay some attention to the gradient vector $\vec{\nabla}\mathbf{U}$ whose definition requires us to specify an inner product in the tangent spaces of $\mathcal{C}$.

Let (U, ϕ) be a chart of $\mathcal{C}$, with $\phi : \mathbf{q} \in U \mapsto (x_1(\mathbf{q}), ..., x_m(\mathbf{q})) \in \mathbf{R}^m$.

Let $\vec{t}$ be a vector of the tangent space $T_{\mathbf{q}}(\mathcal{C})$, $[\vec{t}\,]_\beta$ be the m-vector of its components in the base induced by the chart (U, ϕ), and $\mathbf{q}_{\alpha\vec{t}}$ be the configuration defined by $\phi^{-1}((x_1(\mathbf{q}), ..., x_m(\mathbf{q})) + \alpha[\vec{t}\,]_\beta)$. We denote the directional derivative of $\mathbf{U}$ along the direction $\vec{t}$, taken at $\mathbf{q}$, by $D_{\vec{t}}\mathbf{U}(\mathbf{q})$. It is defined as the following linear functional on $T_{\mathbf{q}}(\mathcal{C})$:

$$D_{\vec{t}}\mathbf{U}(\mathbf{q}) = \lim_{\alpha \to 0} \frac{\mathbf{U}(\mathbf{q}_{\alpha\vec{t}}) - \mathbf{U}(\mathbf{q})}{\alpha}.$$

Let $<,>$ designate the inner product in $T_{\mathbf{q}}(\mathcal{C})$. By definition, the gradient vector $\vec{\nabla}\mathbf{U}(\mathbf{q})$ is the vector of $T_{\mathbf{q}}(\mathcal{C})$ such that for any vector $\vec{t} \in T_{\mathbf{q}}(\mathcal{C})$, we have:

$$D_{\vec{t}}\mathbf{U}(\mathbf{q}) \;=\; < \vec{\nabla}\mathbf{U}, \vec{t} > .$$

Notice the similarity with the definition of a force vector in Section 10 of Chapter 2: while the directional derivative of $\mathbf{U}$ (resp. the work of a force) is independent of the choice of the inner product, the representation of the gradient vector (resp. the force vector) depends on that choice.

Let us select the inner product in $T_{\mathbf{q}}(\mathcal{C})$ such that β is an orthonormal basis. Let t_i, $i = 1$ to m, be the components of the vector $\vec{t}$ in β. We have:

$$\mathbf{U}(\mathbf{q}) = \mathbf{U}(x_1(\mathbf{q}), ..., x_m(\mathbf{q}))$$

and:

$$D_{\vec{t}}\mathbf{U}(\mathbf{q}) = \sum_{i=1}^{m} \frac{\partial \mathbf{U}}{\partial x_i}(x_1(\mathbf{q}), ..., x_m(\mathbf{q}))\, t_i.$$

Hence:

$$[\vec{\nabla}\mathbf{U}(\mathbf{q})]_\beta = \left(\frac{\partial \mathbf{U}}{\partial x_1} \;\; ... \;\; \frac{\partial \mathbf{U}}{\partial x_m}\right)^T_{|(x_1(\mathbf{q}),...,x_m(\mathbf{q}))}$$

which coincides with the conventional definition of a gradient vector in the Euclidean space $\mathbf{R}^m$.

Let $\vec{\nabla}_1\mathbf{U}(\mathbf{q})$ and $\vec{\nabla}_2\mathbf{U}(\mathbf{q})$ denote the gradient vectors of $\mathbf{U}$ for two inner products designated by $<,>_1$ and $<,>_2$, respectively. Let β_1 and β_2 be two bases of $T_{\mathbf{q}}(\mathcal{C})$ such that β_i, for any $i \in \{1, 2\}$, is an orthonormal basis when $<,>_i$ is used. Let J be the $m \times m$ matrix of the linear application that maps the m-vector $[\vec{t}\,]_1$ of the components of a vector

$\vec{t} \in T_{\mathbf{q}}(\mathcal{C})$ in β_1 to the m-vector $[\vec{t}\,]_2$ of the components of $\vec{t}$ in β_2. For any $\vec{t} \in T_{\mathbf{q}}(\mathcal{C})$, we have:

$$\begin{aligned} [\vec{\nabla}_1\mathbf{U}(\mathbf{q})]_1^T\, [\vec{t}\,]_1 &= [\vec{\nabla}_2\mathbf{U}(\mathbf{q})]_2^T\, [\vec{t}\,]_2 \\ &= [\vec{\nabla}_2\mathbf{U}(\mathbf{q})]_2^T\, J\, [\vec{t}\,]_1. \end{aligned}$$

Thus:

$$[\vec{\nabla}_1\mathbf{U}(\mathbf{q})]_1 \;=\; J^T\, [\vec{\nabla}_2\mathbf{U}(\mathbf{q})]_2.$$

(Notice the similarity of this relation with the relation established for forces in Section 10 of Chapter 2.)

Given a metric d in $\mathcal{C}$, the attractive potential function and the associated field of forces can be defined as in the previous section by:

$$\begin{aligned} \mathbf{U}_{att}(\mathbf{q}) &= \tfrac{1}{2}\xi\rho^2_{goal}(\mathbf{q}) \\ \vec{F}_{att}(\mathbf{q}) &= -\vec{\nabla}\mathbf{U}_{att}(\mathbf{q}) \\ &= -\xi\rho_{goal}(\mathbf{q})\vec{\nabla}\rho_{goal}(\mathbf{q}) \end{aligned}$$

where $\rho_{goal}(\mathbf{q}) = d(\mathbf{q}, \mathbf{q}_{goal})$. In the same way, the repulsive potential and the associated field of forces are:

$$\mathbf{U}_{rep}(\mathbf{q}) = \begin{cases} \frac{1}{2}\eta(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0})^2 & \text{if } \rho(\mathbf{q}) \le \rho_0, \\ 0 & \text{if } \rho(\mathbf{q}) > \rho_0, \end{cases}$$

and:

$$\begin{aligned} \vec{F}_{rep}(\mathbf{q}) &= -\vec{\nabla}\mathbf{U}_{rep}(\mathbf{q}) \\ &= \begin{cases} \eta(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0})\frac{1}{\rho^2(\mathbf{q})}\vec{\nabla}\rho(\mathbf{q}) & \text{if } \rho(\mathbf{q}) \le \rho_0, \\ 0 & \text{if } \rho(\mathbf{q}) > \rho_0, \end{cases} \end{aligned}$$

where $\rho(\mathbf{q}) = \min_{\mathbf{q}' \in \mathcal{CB}} d(\mathbf{q}, \mathbf{q}')$.

However, this extension yields computational difficulties. Indeed, for a given metric d there is in general no simple and efficient method for computing both the distance $\rho(\mathbf{q})$ from a configuration $\mathbf{q}$ to the C-obstacle region $\mathcal{CB}$ and its gradient $\vec{\nabla}\rho(\mathbf{q})$.

One practical way to proceed is to define the attractive and repulsive potential functions in $\mathcal{W} = \mathbf{R}^N$ and combine their effects at several points of $\mathcal{A}$. This technique is described in the next two subsections.

2.2 Attractive Potential

Let a_j, $j = 1, ..., N$, be points selected in $\mathcal{A}$, with N being the dimension of the workspace. (If $N = 3$, the 3 points a_j should not be aligned in order to uniquely determine the configuration of $\mathcal{A}$.) The points a_j are called the *control points subject to the attractive potential field.*

For each point a_j, $j \in [1, N]$, we define a distinct attractive potential function $\mathbf{V}_{att}^j : \mathcal{W} \rightarrow \mathbf{R}$ with:

$$\mathbf{V}_{att}^j(\mathbf{x}) = \frac{1}{2}\xi \|\mathbf{x} - a_j(\mathbf{q}_{goal})\|^2$$

where ξ is a positive scaling factor.

Each potential $\mathbf{V}_{att}^j$ induces a field of forces $-\vec{\nabla}\mathbf{V}_{att}^j$ over the workspace. Only the control point a_j is "sensitive" to that field. At each configuration $\mathbf{q}$ of $\mathcal{A}$, the artificial force $\vec{\mathbf{F}}_{att}^j(\mathbf{q}) \in \mathbf{R}^N$ defined by:

$$\begin{aligned} \vec{\mathbf{F}}_{att}^j(\mathbf{q}) &= -\left(\vec{\nabla}\mathbf{V}_{att}^j(\mathbf{x})\right)_{|\mathbf{x}=a_j(\mathbf{q})} \\ &= -\xi(a_j(\mathbf{q}) - a_j(\mathbf{q}_{goal})) \end{aligned}$$

is exerted on $\mathcal{A}$ at point a_j. From Section 10 of Chapter 2, and given an inner product in $T_{\mathbf{q}}(\mathcal{C})$, we know how to map this force into a vector $\vec{F}_{att}^j(\mathbf{q})$ of $T_{\mathbf{q}}(\mathcal{C})$.

The total attractive force in $T_{\mathbf{q}}(\mathcal{C})$ is:

$$\vec{F}_{att}(\mathbf{q}) = \sum_{j=1}^{N} \vec{F}_{att}^j(\mathbf{q}).$$

Let:

$$\mathbf{U}_{att}^j(\mathbf{q}) = \mathbf{V}_{att}^j(a_j(\mathbf{q})).$$

One can easily verify that:

$$\vec{F}_{att}^j(\mathbf{q}) = -\vec{\nabla}\mathbf{U}_{att}^j(\mathbf{q}).$$

Thus:

$$\mathbf{U}_{att}(\mathbf{q}) = \sum_{j=1}^{N} \mathbf{V}_{att}^j(a_j(\mathbf{q}))$$

is the attractive potential in $\mathcal{C}$ which produces the attractive force $\vec{F}_{att}(\mathbf{q})$, i.e. :

$$\vec{F}_{att}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}_{att}(\mathbf{q}).$$

Remark 1: The number of points a_j may be smaller than N. Then, specifying the goal positions of these points only partially determines a goal configuration of $\mathcal{A}$, i.e. the goal configurations form a region of $\mathcal{C}$. ∎

Remark 2: If there is more than one control point a_j subject to the attractive potential field, the above definition of the potential $\mathbf{U}_{att}$ makes the control points compete in trying to attain their respective goal positions. This competition may produce local minima of the total potential in narrow passages and cluttered areas of the workspace. A better definition might be:

$$\mathbf{U}_{att}(\mathbf{q}) = \mathbf{V}_{att}^1(a_1(\mathbf{q})) + \varepsilon \sum_{j>1} \mathbf{V}_{att}^j(a_j(\mathbf{q}))$$

where ε is a small positive constant (say, $\varepsilon = 0.1$). This corresponds to choosing a "leader" among the control points (a_1 in the above definition). Along the generated path, this leading point is vigorously pulled toward its goal position, while the small ε coefficient lets the orientation of $\mathcal{A}$ be mainly under the influence of the repulsive potential (defined below). The second term of the attractive potential nevertheless tends to make the robot ultimately achieve a goal orientation. The above function $\mathbf{U}_{att}$ gives good results when the goal configuration is rather far from the C-obstacle region, but on the other hand it may cause a deep local minimum to exist near the goal configuration if this is not the case. ∎

2.3 Repulsive Potential

We can define and compute the repulsive potential in a similar fashion.

For all $\mathbf{x} \in \mathcal{W} \backslash \mathcal{B}$, where $\mathcal{B}$ is the obstacle region in $\mathcal{W}$, we define the repulsive potential in the workspace by:

$$\mathbf{V}_{rep}(\mathbf{x}) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(\mathbf{x})} - \frac{1}{\rho_0}\right)^2 & \text{if } \rho(\mathbf{x}) \leq \rho_0, \\ 0 & \text{if } \rho(\mathbf{x}) > \rho_0. \end{cases}$$

η is a positive scaling factor, $\rho(\mathbf{x})$ denotes the Euclidean distance from the point $\mathbf{x}$ to $\mathcal{B}$, and ρ_0 is the distance of influence of $\mathcal{B}$.

Let a_j, $j = 1, ..., Q$, be points selected in the boundary of $\mathcal{A}$. These Q points are called the *control points subject to the repulsive potential field.* The control points used in the definition of the attractive forces do not have to be among these Q points, although they may.

When $\mathcal{A}$ is at configuration $\mathbf{q}$, $\mathbf{V}_{rep}$ exerts a force $\vec{\mathbf{F}}^j_{rep}(\mathbf{q})$ on $\mathcal{A}$ which is applied at point a_j:

$$\vec{\mathbf{F}}^j_{rep}(\mathbf{q}) = \begin{cases} \eta\left(\frac{1}{\rho(a_j(\mathbf{q}))} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(a_j(\mathbf{q}))}\vec{\nabla}\rho(\mathbf{x})_{|\mathbf{x}=a_j(\mathbf{q})} & \text{if } \rho(a_j(\mathbf{q})) \leq \rho_0, \\ 0 & \text{if } \rho(a_j(\mathbf{q})) > \rho_0. \end{cases}$$

If $\mathcal{B}$ is a convex region with a piecewise differentiable boundary, ρ is differentiable everywhere in $\mathcal{C}_{free}$. If $\mathcal{B}$ is non-convex, we may decompose it into convex components and associate a repulsive potential with each component (see Subsection 1.3).

Given an inner product in $T_{\mathbf{q}}(\mathcal{C})$, we know how to map $\vec{\mathbf{F}}^j_{rep}(\mathbf{q})$ to a vector $\vec{F}^j_{rep}(\mathbf{q})$ of $T_{\mathbf{q}}(\mathcal{C})$. The resulting repulsive force in $T_{\mathbf{q}}(\mathcal{C})$ is:

$$\vec{F}_{rep}(\mathbf{q}) = \sum_{j=1}^{Q} \vec{F}^j_{rep}(\mathbf{q}).$$

Let:

$$\mathbf{U}^j_{rep}(\mathbf{q}) = \mathbf{V}_{rep}(a_j(\mathbf{q})).$$

One can verify that:

$$\vec{F}^j_{rep}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}^j_{rep}(\mathbf{q}).$$

Thus:

$$\mathbf{U}_{rep}(\mathbf{q}) = \sum_{j=1}^{Q} \mathbf{V}_{rep}(a_j(\mathbf{q}))$$

is the repulsive potential in $\mathcal{C}$ that produces the repulsive force $\vec{F}_{rep}(\mathbf{q})$, i.e. :

$$\vec{F}_{rep}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}_{rep}(\mathbf{q}).$$

In order to make sure that $\mathcal{A}$ cannot come close to $\mathcal{B}$ without being repulsed, we may combine a small number of "fixed" control points and a "variable" control point which depends on $\mathcal{A}$'s configuration. The

variable control point is the point a' in $\mathcal{A}$'s boundary that is closest to $\mathcal{B}$ at the current configuration $\mathbf{q}$ of $\mathcal{A}$. Hence, it is the solution of:

$$\min_{b\in\mathcal{B}} \|a'(\mathbf{q}) - b\| = \min_{a\in\mathcal{A}, b\in\mathcal{B}} \|a(\mathbf{q}) - b\|.$$

The repulsive force that is applied at a' is:

$$\vec{\mathbf{F}}'_{rep}(\mathbf{q}) = \begin{cases} \eta\left(\frac{1}{\rho(a'(\mathbf{q}))} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(a'(\mathbf{q}))}\vec{\nabla}\rho(\mathbf{x})_{|\mathbf{x}=a'(\mathbf{q})} & \text{if } \rho(a'(\mathbf{q})) \leq \rho_0, \\ 0 & \text{if } \rho(a'(\mathbf{q})) > \rho_0. \end{cases}$$

If both $\mathcal{A}$ and $\mathcal{B}$ are convex polygons or polyhedra, the computation of $\min_{a\in\mathcal{A}, b\in\mathcal{B}} \|a(\mathbf{q}) - b\|$, together with a pair of points achieving this minimal distance, can be made in time $O(n_{\mathcal{A}} + n_{\mathcal{B}})$, where $n_{\mathcal{A}}$ and $n_{\mathcal{B}}$ are the number of vertices of $\mathcal{A}$ and $\mathcal{B}$, respectively [Dobkin and Kirkpatrick, 1985]. However, for some critical configurations of $\mathcal{A}$, where an edge or a face of $\mathcal{A}$ is parallel to an edge or a face of $\mathcal{B}$, the minimal distance is achieved by more than a single pair of points. At these configurations, the variable control point a' may change discontinuously in $\mathcal{A}$'s boundary, resulting in a discontinuity of the repulsive force $\vec{F}'_{rep}(\mathbf{q})$. This might cause the generation of an oscillating path. One way to reduce the discontinuity of the repulsive force at these configurations is to distribute the repulsive force over several points of $\mathcal{A}(\mathbf{q})$ among those which are closest to $\mathcal{B}$. Let S be the set of points of $\mathcal{A}(\mathbf{q})$ which are closest to $\mathcal{B}$. If S is not a singleton, it is either a line segment or a polygon. The repulsive force $\vec{F}'_{rep}(\mathbf{q})$ can be distributed over the extreme points of S, i.e. the extremities of the line segment or the vertices of the polygon.

If $\mathcal{A}$ and $\mathcal{B}$ are non-convex polygons, they can be decomposed into two sets of convex polygons $\{\mathcal{A}_k\}$ and $\{\mathcal{B}_l\}$. For each pair $(\mathcal{A}_k, \mathcal{B}_l)$ we can define a variable control point a'_{kl} defined as the point in $\mathcal{A}_k$'s boundary that is the closest to $\mathcal{B}_l$. All such points a'_{kl} are simultaneously taken as variable control points subject to the repulsive potential field.

If $\mathcal{A}$ and $\mathcal{B}$ have more complex shapes, but can be decomposed into convex components $\{\mathcal{A}_k\}$ and $\{\mathcal{B}_l\}$, iterative numerical techniques can be used to compute the distance between $\mathcal{A}_k$ and $\mathcal{B}_l$, together with a pair of points achieving this distance (e.g. see [Gilbert, Johnson and Keerthi, 1988]). We can define the repulsive potential field by using several variable control points a'_{kl}, one for each pair $(\mathcal{A}_k, \mathcal{B}_l)$. If either

$\mathcal{A}_k$ or $\mathcal{B}_l$ is strictly convex[1], the other being convex, and if both have piecewise differentiable boundaries, then a'_{kl} is uniquely defined and the repulsive force exerted at a'_{kl} varies continuously.

3 Potential-Guided Path Planning

We now describe simple potential-guided path planning techniques. In principle, these techniques do not assume any specific potential function. Hence, they are applicable with the potential function defined in the previous two sections, and with other potential functions as well.

In its original conception, the potential field approach to motion generation consists of regarding the robot in configuration space as a unit mass particle moving under the influence of the force field $\vec{F} = -\vec{\nabla}\mathbf{U}$. At every configuration $\mathbf{q}$ the artificial force $\vec{F}(\mathbf{q})$ determines the acceleration of the particle. Knowing the dynamic equation of $\mathcal{A}$ and assuming no limitation on the power of the actuators, one can compute the forces/torques that should be delivered by the actuators at each instant so that the robot actually behaves according to the particle metaphor. These forces/torques are the commands sent to the robot's servo controllers [Hogan, 1985] [Khatib, 1986] [Koditschek, 1989].

This way of using the potential function is applicable for generating paths on-line. It is well-suited when the obstacles are not known in advance, but sensed during motion execution. If a prior model of the obstacles is available, the same method can be used to plan a path by simulating the motion of the particle. However, in this case, there exist simpler and more efficient path planning techniques using potential field.

In the following subsections, we present several such techniques. The first of them generates a path in a "depth-first" fashion, without backtracking. Like on-line path generation, it may be very fast in favorable cases, but it may also get stuck at local minima of the potential function. The second technique operates in a "best-first" mode. It deals with local minima by "filling" them up. The third technique consists of optimizing a functional constructed by integrating the potential along a complete

[1] A closed set $S \subset \mathbf{R}^N$ is *strictly convex* if and only if it is homeomorphic to the closed unit ball in $\mathbf{R}^N$ and for every point p in the boundary of S, there exists a line passing through p and intersecting S at p only.

path between the initial and the goal configurations.

3.1 Depth-First Planning

This technique constructs a path as the product of successive path segments starting at the initial configuration $\mathbf{q}_{init}$. Each segment is oriented along the negated gradient of the potential function computed at the configuration attained by the previous segment. The amplitude of the segment is chosen so that the segment lies in free space.

Let $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$ be the origin and end extremities of the i^{th} segment in the path. Let $x_j(\mathbf{q}_i)$, $j = 1, ..., m$, be the coordinates of $\mathbf{q}_i$ in some chart (U, ϕ). We define the inner product in the tangent space $T_{\mathbf{q}}(\mathcal{C})$ so that the basis β induced by this chart in $T_{\mathbf{q}}(\mathcal{C})$ is orthonormal. We then have:

$$[\vec{F}]_\beta = -[\vec{\nabla}\mathbf{U}]_\beta = (-\partial\mathbf{U}/\partial x_1 \ \ ... \ \ -\partial\mathbf{U}/\partial x_m)^T.$$

We denote the components of the unit vector $\vec{t}(\mathbf{q}_i) = \vec{F}(\mathbf{q}_i)/\|\vec{F}(\mathbf{q}_i)\|$ in β by $t_j(\mathbf{q}_i)$. The coordinates of the configuration $\mathbf{q}_{i+1}$ attained at the i^{th} iteration, in (U, ϕ), are:

$$x_j(\mathbf{q}_{i+1}) = x_j(\mathbf{q}_i) + \delta_i t_j(\mathbf{q}_i), \ j = 1, ..., m,$$

with δ_i denoting the length of the i^{th} increment (measured with the Euclidean metric of $\mathbf{R}^m$). The segment $\mathbf{q}_i\mathbf{q}_{i+1}$ is the inverse image in $\mathcal{C}$ of the straight line segment joining $\phi(\mathbf{q}_i)$ to $\phi(\mathbf{q}_{i+1})$ in $\mathbf{R}^m$.

For example, if $\mathcal{A}$ is a planar object moving in $\mathcal{W} = \mathbf{R}^2$, we can parameterize any $\mathbf{q}$ by $(x_1, x_2, x_3) = (x, y, \theta) \in \mathbf{R}^2 \times [0, 2\pi)$, with x and y being the coordinates of $\mathcal{A}$'s reference point $O_{\mathcal{A}}$ at $\mathbf{q}$, and θ being the angle (modulo 2π) between the x-axes of the frames $\mathcal{F}_{\mathcal{W}}$ and $\mathcal{F}_{\mathcal{A}}$ attached to $\mathcal{W}$ and $\mathcal{A}$, respectively. Then:

$$\begin{aligned} x(\mathbf{q}_{i+1}) &= x(\mathbf{q}_i) + \delta_i \frac{\partial \mathbf{U}}{\partial x}(x, y, \theta), \\ y(\mathbf{q}_{i+1}) &= y(\mathbf{q}_i) + \delta_i \frac{\partial \mathbf{U}}{\partial y}(x, y, \theta), \\ \theta(\mathbf{q}_{i+1}) &= \theta(\mathbf{q}_i) + \delta_i \frac{\partial \mathbf{U}}{\partial \theta}(x, y, \theta) \mod 2\pi. \end{aligned}$$

In order to "normalize" the displacements along the θ-axis relative to displacements along the x- and y-axes, one may parameterize $\mathbf{q}$ by

$(x, y, \phi) \in \mathbf{R}^2 \times [0, 2\pi R)$, by posing $\phi = \theta R$, with $R = \max_{a \in \partial \mathcal{A}} \|O_{\mathcal{A}} - a\|$ being the maximal distance between the reference point $O_{\mathcal{A}}$ and $\mathcal{A}$'s boundary. This yields:

$$\phi(\mathbf{q}_{i+1}) = \phi(\mathbf{q}_i) + \delta_i \frac{\partial \mathbf{U}}{\partial \phi}(x, y, \phi) \mod 2\pi R,$$

or, equivalently:

$$\theta(\mathbf{q}_{i+1}) = \theta(\mathbf{q}_i) + \frac{\delta_i}{R} \frac{\partial \mathbf{U}}{\partial \phi}(x, y, \theta R) \mod 2\pi.$$

Several considerations intervene in the choice of a value for δ_i. First, δ_i should be taken small enough so that the direction of the force and the mapping of that direction in the local coordinate system keeps some meaning along the segment $\mathbf{q}_i\mathbf{q}_{i+1}$. Typically, δ_i is taken equal to some small prespecified increment δ, unless other considerations examined below require δ_i to be smaller.

The increment δ_i should also be chosen small enough so that no collision happens along the segment $\mathbf{q}_i\mathbf{q}_{i+1}$. Suppose that δ_i is equal to δ. The motion from $\mathbf{q}_i$ to $\mathbf{q}_{i+1}$ can be expressed as a small rotation about $O_{\mathcal{A}}$, followed by a small translation. The point of $\mathcal{A}$ that moves along the longest distance during the rotation is the point $a \in \partial\mathcal{A}$ that is the most distant from $O_{\mathcal{A}}$. Let l_1 be the length of this displacement and l_2 be the length of the translation. The sum $l = l_1 + l_2$ is an upper bound on the maximal displacement of a point of $\mathcal{A}$ during the motion from $\mathbf{q}_i$ to $\mathbf{q}_{i+1}$. If l is smaller than the minimum distance of $\mathcal{A}$ to the obstacles, then the segment $\mathbf{q}_i\mathbf{q}_{i+1}$ is guaranteed to lie in free space. Otherwise, a smaller value of δ_i, say $\delta/2$, may be selected and checked again for collision.

Finally, the value of the increment should not lead the path to go beyond the goal configuration. Assume that the chart (U, ϕ) used at $\mathbf{q}_i$ also allows us to map $\mathbf{q}_{goal}$. We should take δ_i smaller than the Euclidean distance (in $\mathbf{R}^m$) between the mappings $\phi(\mathbf{q}_i)$ and $\phi(\mathbf{q}_{goal})$.

This technique simply follows the steepest descent of the potential function until the goal configuration is attained. However, it may get stuck at a minimum of the potential other than the goal configuration (Figure 2). Dealing with local minima within depth-first planning is not simple. First, the fact that a local minimum has been attained must be recognized. Since motions are discretized, the planner usually does not stop

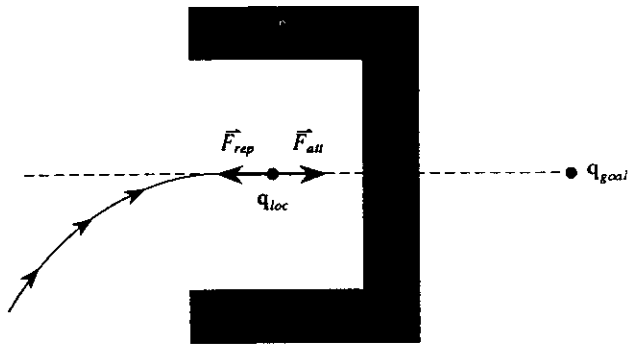

Figure 2. The attractive potential guides the robot's path into a C-obstacle concavity. At some configuration $\mathbf{q}_{loc}$, the repulsive force generated by the C-obstacle cancels exactly the attractive force generated by the goal. This stable zero-force configuration is a local minimum of the total potential function.

exactly at the zero-force configuration; instead, it typically generates segments "looping" around this configuration. This may be detected by checking that several successive configurations $\mathbf{q}_i$'s are not too close to each other. Second, the planner must escape the local minimum. A simple technique consists of moving along a certain direction by some distance before resuming depth-first path generation. A possible direction of motion is any direction in the tangent plane of the repulsive equipotential surface at the local minimum. This corresponds to "moving around" the combination of obstacles that creates the problematic minimum.[2] A more systematic way of dealing with local minima consists of seeing the generated path as a path in a search graph whose nodes are the $\mathbf{q}_i$'s. This leads to presenting best-first planning.

3.2 Best-First Planning

Let us throw a fine regular grid of configurations across $\mathcal{C}$. We denote this grid by $\mathcal{GC}$. $\mathcal{GC}$ can be defined by considering a single chart over

[2] Donald [Donald, 1984 and 1987b] described a variety of heuristics, which he called "local experts", for "moving around" C-obstacles. The principle of these heuristics is to slide over level C-surfaces and/or their intersections. (If $f(\mathbf{q}) = 0$ is the equation of a C-surface (see Chapter 3), $f(\mathbf{q}) = K$, with K constant, is the equation of a level C-surface.) Although Donald did not explicitly use potential functions, his heuristics are directly relevant here.

$\mathcal{C}$ and discretizing each of the m corresponding coordinate axes. For instance, if $\mathcal{A}$ is a free-flying object in $\mathcal{W} = \mathbf{R}^2$, the grid consists of the configurations $(k_x\delta x, k_y\delta y, k_\theta\delta\theta)$, with $k_x, k_y, k_\theta \in \mathbf{Z}$ and modulo 2π arithmetic on θ. The angular increment $\delta\theta$ is an integer fraction of 2π. (As in the previous section, we may use $\phi = \theta R$ rather than θ for the third coordinate, in order to normalize the scale of the angular axis relative to the translational axes.)

Given a configuration $\mathbf{q}$ in the m-dimensional grid $\mathcal{GC}$, its **p-neighbors** ($1 \leq p \leq m$) are defined as all the configurations in $\mathcal{GC}$ having at most p coordinates differing from those of $\mathbf{q}$, the amount of the difference being exactly one increment in absolute value (with appropriate modular arithmetic applied to the angular parameters). There are $2m$ 1-neighbors, $2m^2$ 2-neighbors, ..., and 3^m-1 m-neighbors. In this subsection we consider that two configurations in $\mathcal{GC}$ are neighbors if and only if they are p-neighbors for a predefined $p \in [1, m]$. In practice, we can take $p = 1$ or 2. In addition, for simplifying the presentation, we make the following assumptions:

- Both $\mathbf{q}_{init}$ and $\mathbf{q}_{goal}$ are configurations in $\mathcal{GC}$.
- If two neighbors in $\mathcal{GC}$ are in free space, the straight line segment connecting them in $\mathbf{R}^m$ also lies in free space.
- The grid $\mathcal{GC}$ is bounded and forms a rectangloid (this is achieved by bounding the set of possible positions of $\mathcal{A}$ by a rectangloid).

Best-first planning consists of iteratively constructing a tree T whose nodes are configurations in $\mathcal{GC}$. The root of T is $\mathbf{q}_{init}$. At every iteration, the algorithm examines the neighbors of the leaf of T that has the smallest potential value, retains the neighbors not already in T at which the potential function is less than some large threshold[3] M, and installs the retained neighbors in T as successors of the currently considered leaf. The algorithm terminates when the goal configuration $\mathbf{q}_{goal}$ has been attained (success) or when the free subset of $\mathcal{GC}$ accessible from $\mathbf{q}_{init}$ has been fully explored (failure). Each node in T has a pointer toward its parent. If $\mathbf{q}_{goal}$ is attained, a path is generated by tracing the pointers from $\mathbf{q}_{goal}$ to $\mathbf{q}_{init}$.

[3]We assume here that the potential U(q) is defined such that it grows toward infinity when q gets closer to the C-obstacle region and that it is infinite when $\mathbf{q} \in \mathcal{CB}$.

The procedure BFP given below is a formal expression of the best-first planning algorithm. In addition to the tree T, BFP uses a list OPEN that contains the leaves of T sorted by increasing values of the potential function. OPEN supports the following three operations: FIRST(OPEN), which removes the configuration of OPEN having the smallest potential value and returns it; INSERT(q,OPEN), which inserts the configuration q in OPEN; and EMPTY(OPEN), which returns `true` (resp. `false`) if the list OPEN is empty (resp. non-empty).

```
procedure BFP;
begin
   install q_init in T; [initially, T is the empty tree]
   INSERT(q_init,OPEN); mark q_init visited;
   [initially, all the configurations in GC are marked "unvisited"]
   SUCCESS ← false;
   while ¬ EMPTY(OPEN) and ¬ SUCCESS do
      begin
         q ← FIRST(OPEN);
         for every neighbor q' of q in GC do
            if U(q') < M and q' is not visited then
               begin
                  install q' in T with a pointer toward q;
                  INSERT(q',OPEN); mark q' visited;
                  if q' = q_goal then SUCCESS ← true;
               end;
      end;
   if SUCCESS then
      return the constructed path by tracing the pointers in T
      from q_goal back to q_init;
   else return failure;
end;
```

This procedure follows a discrete approximation of the negated gradient of the potential function until it reaches a local minimum. When this happens, it operates in a way that corresponds to filling the well of this local minimum until a saddle point is attained.

The algorithm is guaranteed to return a free path whenever there exists one in the free subset of the grid $\mathcal{GC}$ and to report failure otherwise. Let $O(r)$ be the number of discretization points along each coordinate

axis. The size of $\mathcal{GC}$ is $O(r^m)$. We can represent the list OPEN as a balanced tree [Aho, Hopcroft and Ullman, 1983] so that each operation FIRST and INSERT takes logarithmic time in the size of OPEN, which is $O(r^m)$ in the worst case. The time complexity of the procedure BFP given above is thus $O(mr^m \log r)$.

Most of the time, the procedure BFP will only explore a small subset of $\mathcal{GC}$. Rather than representing $\mathcal{GC}$ explicitly in a large array and marking its configurations *visited* or *unvisited*, we may only represent the configurations that are installed in T. Then, instead of checking whether a configuration $\mathbf{q}'$ is marked *visited*, we test if it is in T. By storing the configurations in T in a balanced-tree data structure sorted according to the configurations' coordinates, the test takes logarithmic time in the number of configurations in T, which is $O(r^m)$ in the worst case. This modification slightly increases the running time of the procedure (without changing its asymptotic time complexity), but reduces the amount of memory space it needs. It also makes it possible to run the algorithm with a grid $\mathcal{GC}$ of unbounded size (but then the running time is no longer bounded).

BFP is practical only when m is small, say less than 5. In the case of a free-flying object in a two-dimensional workspace ($m = 3$), it provides a means for implementing a very fast and reliable planner with grid resolutions of the order of 256^3 (see [Barraquand and Latombe, 1989a]). It may even be made faster (in average time) by using a pyramid of grids at various resolutions. However, when m becomes larger, filling up local minimum wells is no longer tractable (the number of discretized configurations in a well tends to grow exponentially with the dimension m). Other techniques, e.g. the "randomized planning" technique presented in Section 5, must then be used.

3.3 Variational Planning

Another approach to path planning using a potential field consists of constructing a functional J of a path τ and optimizing this functional over all possible paths. For example, if $\mathbf{U}$ is a potential function that is defined over the whole configuration space, with large values in the C-obstacle region and small values in free space, a possible definition of J is the following:

$$J(\tau) = \int_0^1 \mathbf{U}(\tau(s))ds + \int_0^1 \|\frac{d\tau}{ds}\|ds.$$

The purpose of the first term in J is to make the optimization produce a free path. The second term in J is optional and is aimed at producing shorter paths. In fact, other objectives could be encoded in this form. The optimization of the functional J can be done using standard variational calculus methods. We report the reader to publications describing implementations of this approach for more detail about the construction of both an appropriate potential $\mathbf{U}$ and an appropriate objective functional J, and about the optimization of this functional (e.g. [Buckley, 1985] [Gilbert and Johnson, 1985] [Hwang and Ahuja, 1988] [Suh and Shin, 1988]).

Variational path planning suffers from the same sort of drawback as depth-first planning. Indeed, since it essentially consists of minimizing a functional J by following its negated gradient, it may get stuck at a local minimum of J that does not correspond to a free path. In addition, the optimization of J is conducted over a space of much larger dimension (the number of configurations in the path) and can be quite costly. The advantage of variational path planning is that it allows additional objective criteria to be encoded in J.

4 Other Potential Functions

A variety of potential functions other than those presented in Sections 1 and 2 have been proposed in the literature. The most interesting of them are aimed at either one of these two goals: (1) improving the "local dynamic behavior" of the robot along the generated paths; (2) reducing the number of local minima and/or the size of their attractive wells.

The first goal is important when the potential field method is used to generate a path on-line or in a depth-first fashion (although in the second case, the path can be improved in a postprocessing step) [Tilove, 1990]. One of the proposed function is the "generalized potential field" [Krogh, 1984] which is a function of *both* the robot's configuration and velocity. It is constructed in such a way that the robot is repelled by an obstacle only if it is close to the obstacle *and* its velocity points toward the obstacle. If the robot moves parallel to the obstacle, it does not have to be repulsed by it. Using a similar idea, Faverjon and Tournassoud [Faver-