Autonomous and Mobile Robotics
Prof. Giuseppe Oriolo

Motion Planning 3:
Artificial Potential Fields
on-line planning

• autonomous robots must be able to plan on line, i.e., using partial workspace information collected during the motion via the robot sensors

• incremental workspace information may be integrated in a map and used in a sense-plan-move paradigm (deliberative navigation)

• alternatively, incremental workspace information may be used to plan motions following a memoryless stimulus-response paradigm (reactive navigation)
artificial potential fields

• idea: build potential fields in $\mathcal{C}$ so that the point that represents the robot is attracted by the goal $q_g$ and repelled by the $\mathcal{C}$-obstacle region $\mathcal{CO}$

• the total potential $U$ is the sum of an attractive and a repulsive potential, whose negative gradient $-\nabla U(q)$ indicates the most promising local direction of motion

• the chosen metric in $\mathcal{C}$ plays a role
attractive potential

- **objective**: to guide the robot to the goal $q_g$

- two possibilities; e.g., in $C = \mathbb{R}^2$

paraboloidal

conical
• **paraboloidal:** let \( e = q_g - q \) and choose \( k_a > 0 \)

\[
U_{a1}(q) = \frac{1}{2} k_a e^T(q) e(q) = \frac{1}{2} k_a \|e(q)\|^2
\]

• the resulting attractive force is **linear** in \( e \)

\[
f_{a1}(q) = -\nabla U_{a1}(q) = k_a e(q)
\]

• **conical:**

\[
U_{a2}(q) = k_a \|e(q)\|
\]

• the resulting attractive force is **constant**

\[
f_{a2}(q) = -\nabla U_{a2}(q) = k_a \frac{e(q)}{\|e(q)\|}
\]
• $f_{a1}$ behaves better than $f_{a2}$ in the vicinity of $q_g$ but increases indefinitely with $e$

• a convenient solution is to combine the two profiles: conical away from $q_g$ and paraboloidal close to $q_g$

$$U_a(q) = \begin{cases} 
  \frac{1}{2} k_a \|e(q)\|^2 & \text{if } \|e(q)\| \leq \rho \\
  k_b \|e(q)\| & \text{if } \|e(q)\| > \rho 
\end{cases}$$

continuity of $f_a$ at the transition requires

$$k_a e(q) = k_b \frac{e(q)}{\|e(q)\|} \quad \text{for } \|e(q)\| = \rho$$

i.e., $k_b = \rho k_a$
repulsive potential

- **objective**: keep the robot away from \( CO \)
- assume that \( CO \) has been partitioned in advance in convex components \( CO_i \)
- for each \( CO_i \) define a repulsive field

\[
U_{r,i}(q) = \begin{cases} \frac{k_{r,i}}{\gamma} \left( \frac{1}{\eta_i(q)} - \frac{1}{\eta_{0,i}} \right)^\gamma & \text{if } \eta_i(q) \leq \eta_{0,i} \\ 0 & \text{if } \eta_i(q) > \eta_{0,i} \end{cases}
\]

where \( k_{r,i} > 0; \gamma = 2, 3, \ldots; \eta_{0,i} \) is the range of influence of \( CO_i \) and

\[
\eta_i(q) = \min_{q' \in CO_i} \| q - q' \|
\]
The higher \( \gamma \), the steepest the slope.

\( U_{r,i} \) goes to \( \infty \) at the boundary of \( CO_i \).
• the resulting repulsive force is

\[ f_{r,i}(q) = -\nabla U_{r,i}(q) = \begin{cases} \frac{k_{r,i}}{\eta_i^2(q)} \left( \frac{1}{\eta_i(q)} - \frac{1}{\eta_{0,i}} \right)^{\gamma^{-1}} \nabla \eta_i(q) & \text{if } \eta_i(q) \leq \eta_{0,i} \\ 0 & \text{if } \eta_i(q) > \eta_{0,i} \end{cases} \]

• \( f_{r,i} \) is orthogonal to the equipotential contour passing through \( q \) and points away from the obstacle

• \( f_{r,i} \) is continuous everywhere thanks to the convex decomposition of \( CO \)

• aggregate repulsive potential of \( CO \)

\[ U_r(q) = \sum_{i=1}^{p} U_{r,i}(q) \]
**total potential**

- superposition: \( U_t(q) = U_a(q) + U_r(q) \)

- force field: \( f_t(q) = -\nabla U_t(q) = f_a(q) + \sum_{i=1}^{p} f_{r,i}(q) \)
planning techniques

• three techniques for planning on the basis of $f_t$

1. consider $f_t$ as generalized forces: $\tau = f_t(q)$
   the effect on the robot is filtered by its dynamics
   (generalized accelerations are scaled)

2. consider $f_t$ as generalized accelerations: $\ddot{q} = f_t(q)$
   the effect on the robot is independent on its dynamics
   (generalized forces are scaled)

3. consider $f_t$ as generalized velocities: $\dot{q} = f_t(q)$
   the effect on the robot is independent on its dynamics
   (generalized forces are scaled)
• technique 1 generates smoother movements, while technique 3 is quicker (irrespective of robot dynamics) to realize motion corrections; technique 2 gives intermediate results

• strictly speaking, only technique 3 guarantees (in the absence of local minima) asymptotic stability of $q_g$; velocity damping is necessary to achieve the same with techniques 1 and 2
• **off-line planning**

paths in $\mathcal{C}$ are generated by numerical integration of the dynamic model (if technique 1), of $\dot{q} = f_t(q)$ (if technique 2), of $q = f_t(q)$ (if technique 3)

the most popular choice is 3 and in particular

$$q_{k+1} = q_k + T f_t(q_k)$$

i.e., the algorithm of steepest descent

• **on-line planning** (is actually feedback!)

technique 1 directly provides control inputs, technique 2 too (via inverse dynamics), technique 3 provides reference velocities for low-level control loops

the most popular choice is 3
local minima: a complication

- if a planned path enters the basin of attraction of a local minimum \( q_m \) of \( U_t \), it will reach \( q_m \) and stop there, because \( f_t (q_m) = -\nabla U_t(q_m) = 0 \); whereas saddle points are not an issue.

- repulsive fields generally create local minima, hence motion planning based on artificial potential fields is not complete (the path may not reach \( q_g \) even if a solution exists).

- workarounds exist but consider that artificial potential fields are mainly used for on-line motion planning, where completeness may not be required.
workaround no. 1: best-first algorithm

- build a **discretized representation** (by defect) of $C_{\text{free}}$ using a regular grid, and associate to each free cell of the grid the value of $U_t$ at its centroid.

- build a tree $T$ rooted at $q_s$: at each iteration, select the leaf of $T$ with the **minimum** value of $U_t$ and add as children its adjacent free cells that are not in $T$.

- planning stops when $q_g$ is reached (success) or no further cells can be added to $T$ (failure).

- if success, build a solution path by tracing back the arcs from $q_g$ to $q_s$. 
• best-first evolves as a grid-discretized version of steepest descent until a local minimum is met

• at a local minimum, best-first will “fill” its basin of attraction until it finds a way out

• the best-first algorithm is resolution complete

• its complexity is exponential in the dimension of $C$, hence it is only applicable in low-dimensional spaces

• efficiency improves if random walks are alternated with basin-filling iterations (randomized best-first)
workaround no. 2: navigation functions

• path generated by the best-first algorithm are not efficient (local minima are not avoided)

• a different approach: build navigation functions, i.e., potentials without local minima

• if the $C$-obstacles are star-shaped, one can map $CO$ to a collection of spheres via a diffeomorphism, build a potential in transformed space and map it back to $C$

• another possibility is to define the potential as an harmonic function (solution of Laplace’s equation)
• an efficient alternative: **numerical navigation functions**

• with $C_{\text{free}}$ represented as a gridmap, assign 0 to start cell, 1 to cells adjacent to the 0-cell, 2 to unvisited cells adjacent to 1-cells, ... **(wavefront expansion)**

```
<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>7</td>
<td>8</td>
<td></td>
<td>10</td>
<td>11</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
<td>12</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
<td>12</td>
<td>13</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
</tbody>
</table>
```

**solution path:** steepest descent from the goal
planning for robot manipulators

- complexity of motion planning is **high**, because the configuration space has dimension typically $\geq 4$

- try to **reduce** dimensionality: e.g., in 6-dof robots, replace the wrist with the total volume it can sweep (a conservative approximation)

- both the construction and the shape of $CO$ are complicated by to the presence of **revolute** joints

- **off-line planning**: probabilistic methods are the best choice (although collision checking is heavy)

- **on-line planning**: adaptation of artificial potential fields
artificial potentials for robot manipulators

• to avoid the computation of $CO$ and the “curse of dimensionality”, the potential is built in $W$ (rather than in $C$) and acts on a set of control points $p_1,...,p_P$ distributed on the robot body

• in general, control points include one point per link $(p_1,...,p_{P-1})$ and the end-effector (to which the goal is typically assigned) as $p_P$

• the attractive potential $U_a$ acts on $p_P$ only, while the repulsive potential $U_r$ acts on the whole set $p_1,...,p_P$; hence, $p_P$ is subject to the total $U_t = U_a + U_r$
two techniques for planning with control points:

1. impose to the robot joints the **generalized forces** resulting from the combined action of force fields

\[
\boldsymbol{\tau} = - \sum_{i=1}^{P-1} \mathbf{J}^T_i(q) \nabla U_r(p_i) - \mathbf{J}^T_P(q) \nabla U_t(p_P)
\]

where \(\mathbf{J}_i(q), i = 1,...,P\), is the **Jacobian** matrix of the direct kinematics function associated to \(p_i(q)\)

2. use the above expression as **reference velocities** to be fed to the low-level control loops

\[
\dot{q} = - \sum_{i=1}^{P-1} \mathbf{J}^T_i(q) \nabla U_r(p_i) - \mathbf{J}^T_P(q) \nabla U_t(p_P)
\]
• technique 2 is actually a gradient-based minimization step in \(\mathcal{C}\) of a combined potential in \(\mathcal{W}\); in fact
\[
\nabla q U(p_i) = \left(\frac{\partial U(p_i(q))}{\partial q}\right)^T = \left(\frac{\partial U(p_i)}{\partial p_i} \frac{\partial p_i}{\partial q}\right)^T = J_i^T(q)\nabla U(p_i)
\]

• technique 1 generates smoother movements, while technique 2 is quicker (irrespective of robot dynamics) to realize motion corrections

• both can stop at force equilibria, where the various forces balance each other even if the total potential \(U_t\) is not at a local minimum; hence, this method should be used in conjunction with a best-first algorithm
a force equilibrium between attractive and repulsive forces