


## L06: Continuous Optimal Planning

Planning Algorithms in AI

Prof. Gonzalo Ferrer,

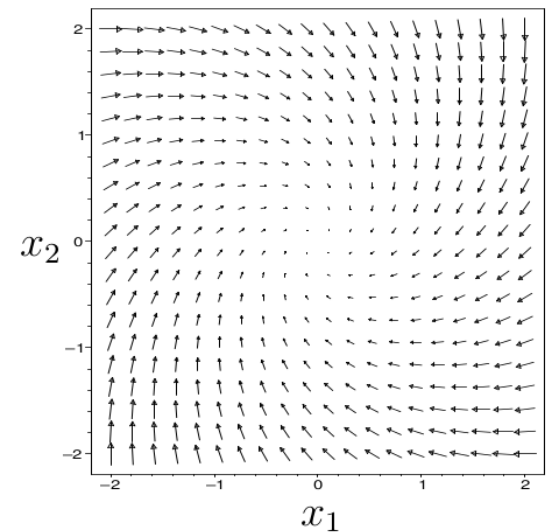
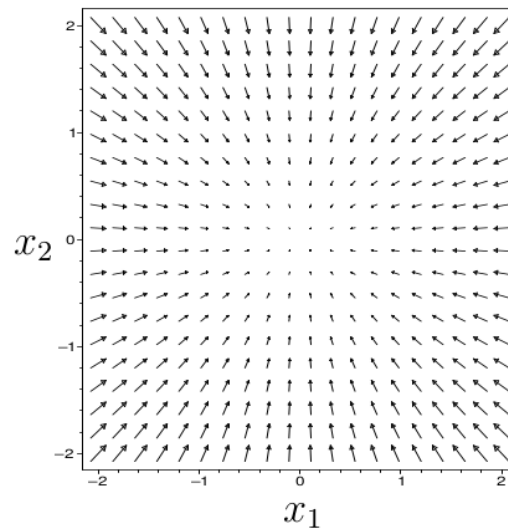
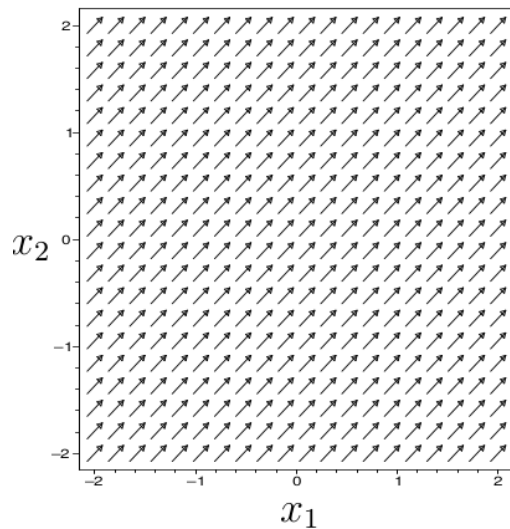
Skoltech, 26 November 2021

# Summary L06, LaValle 8.3-8.5

- ➔ Vector fields
  - ➔ Continuous feedback-based planning
  - ➔ Navigation functions in continuous spaces
  - ➔ (Artificial) Potential fields
  - ➔ Composition of PFs
  - ➔ Cell decomposition and PF
  - ➔ Funnels
  - ➔ Optimal Navigation Functions
  - ➔ Trajectory Optimization
- 

# Vector Field

- Let's continue with the idea of feedback plan, now applied to continuous state spaces (or C-spaces).
- The idea is to specify a “direction” at every point of the state space, as seen in the images.



# Vector Field and Tangent Space

Let's define a vector space, which could represent our state space, C-space, etc.

$$x \in \mathbb{R}^n$$

For every point  $x$ , associate an  $n$ -dimensional space called the **tangent space** at  $x$ , denoted as

$$T_x(\mathbb{R}^n)$$

A **vector field**  $V$  is a function that assigns a vector  $v$  to every point of  $x$  such that

$$v \in T_x(\mathbb{R}^n), \quad \forall x \in \mathbb{R}^n$$

# Vector Fields as Velocity Fields

An interpretation of the *vector field* could be seen as a **velocity field**, where for each coordinate, we simply derive with respect to time:

$$\frac{dx}{dt} = \left[ \frac{dx_1}{dt}, \frac{dx_2}{dt}, \dots, \frac{dx_n}{dt} \right] = [f_1, f_2, \dots, f_n]$$

This enables the function  $f$  to be interpreted as a **velocity field**. Since we are not considering dynamics, the development that follows will consider actions to be directly velocities.

$$\dot{x} = dx/dt = f(x)$$

# Transition function and transition equation

The function  $f$ , has already been presented in the course for describing the transition between states (in discrete form). In the continuous domain, it is known as the *transition equation*.

$$\dot{x} = f(x)$$

In the discrete domain, it is known as the *transition function*:

$$x' = f(x, u)$$

One can build the transition function by integrating the transition equation over a given path, for instance using a constant action over the interval (Euler method)

In Perception In Robotics course we will discuss on L05 how to build some of these function and how to build a probabilistic model.

# Continuous feedback-based planning definitions

1. A continuous **state space**  $X$ . Mostly we are referring to  $C_{free}$
2. For each state  $x \in X$ , an **action space**  $U(x) = T_x(X)$   
The zero velocity is designated as the termination action  $u_T = 0 \in T_x(X)$
3. An unbounded time interval
4. A **state transition equation**  $f$ . For today's lecture, these are velocities.

$$u = \dot{x} = f(x)$$

5. A **goal set**  $X_G \subset X$

Then, a feedback plan can be considered as a vector field on  $X$ , providing an action for every state.

# Continuous feedback-based planning definitions

A **feasible feedback motion plan** provides a solution to the goal set  $X_G$  for any initial state from which  $X_G$  is reachable.

$$\exists \tau : [0, 1] \rightarrow X, \quad \text{for which } \tau(0) = x_I \in X, \tau(1) = x_G \in X_G$$

An **optimal feedback motion plan**, in addition of being a feasible plan, it provides an optimal solution to reach the goal set  $X_G$ , by minimizing the cost functional:

$$L(\text{plan}) = \int_0^{t_F} l(x(t), u(t)) dt + l_F(x(t_F))$$



# Navigation functions

We will redefine the term Navigation function for continuous spaces.

Now, the *local operator* (L05) must convert the potential function (PF) into a vector field such that for every state there is an action:

$$u^* = \arg \min_{u \in U(x)} \{ \phi(x') \}, \quad \text{where } \phi : X \rightarrow \mathbb{R}$$

Recall that the idea was to minimize the potential, therefore, it is reasonable to just follow the gradient of the potential, **artificial potential field\*** approach:

$$\nabla \phi = \left[ \frac{\partial \phi}{\partial x_1}, \frac{\partial \phi}{\partial x_2}, \dots, \frac{\partial \phi}{\partial x_n} \right]$$

The corresponding feedback plan can be defined as  $\pi(x) = -\nabla_x \phi(x)$

\*O. Khatib (1985) “Real-time obstacle avoidance for manipulators and mobile robots”, ICRA

# Potential Fields, borrowed from physics

The motivation comes from physics, for instance, if  $h$  is the height above's Earth's surface in uniform gravitational potential field  $g$ , then the potential energy of a mass  $m$  is

$$\phi(h) = mgh$$

And the resultant force acting on this body, causing the mass fall into the ground:

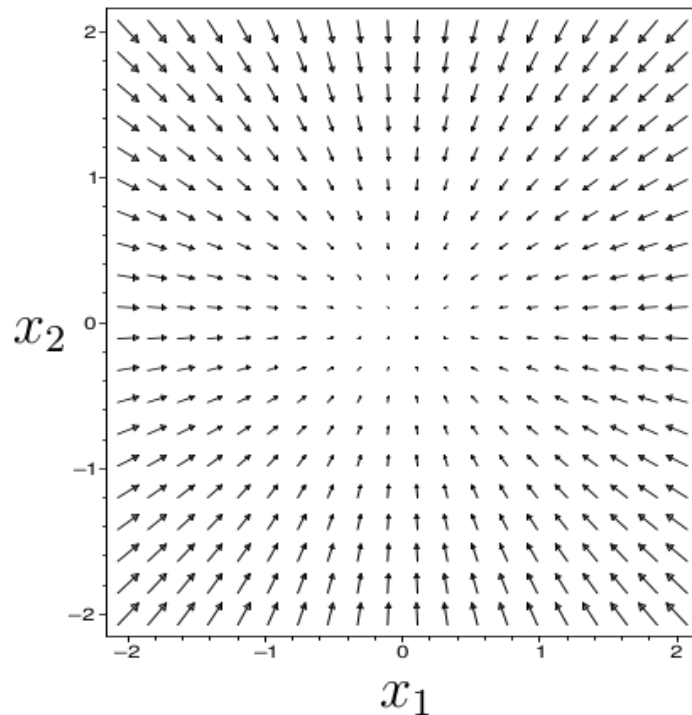
$$F = -\frac{\partial \phi}{\partial h} = -mg$$

On this lecture, we will not deal with dynamics (no forces or torques). APF could be adapted to dynamics easily, but as an introduction for today, we will consider only velocity fields and actions equal to velocities.

# Potential Field

One can propose a potential function. However, it must satisfy the conditions to generate a feasible plan to be considered a *navigation function*.

Example:



$$\phi(x) = \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2$$

$$\pi(x) = -\nabla_x \phi(x) = [-x_1, -x_2]$$

## Relation with Controls

The relation with controls is very narrow. The attractive PF described in the example could be seen as an elementary controller, for a free motion system.

$$\phi(x) = \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2 \quad \pi(x) = -\nabla_x \phi(x) = [-x_1, -x_2]$$

The equivalent of PF is known as a **Proportional** controller, where there is an error signal to minimize and the action is just proportional to the error:

$$\pi(x) = -ke_x = [-x_1, -x_2]$$

PF could also map to a force vector field, in which case the results are identical to controls since we are considering dynamics. Recall that in this lecture (and course) we consider only free moving systems, but the idea is the same for dynamics.

# Potential Field: composition

Can we define a composition of potential functions?

First, the **attractive** PF for reaching the goal, now in the C-space:

$$\phi_G(q) = \frac{1}{2} \|q - g_G\|_{K^{-1}}^2 = \frac{1}{2} (q - g_G)^\top K (q - g_G)$$

$$u_G(q) = -\frac{\partial \phi_G}{\partial q} = -K(q - g_G)$$

And a second **repulsive** PF, that repels the robot from an object B:

$$\phi_B(q) = \frac{k}{2d^2(q, \mathcal{B})}$$

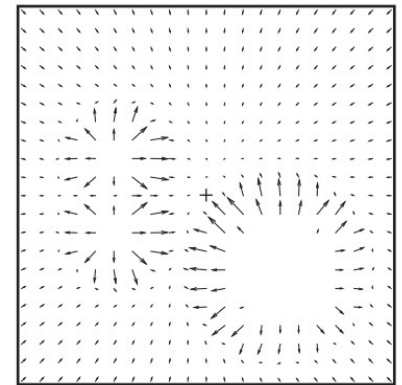
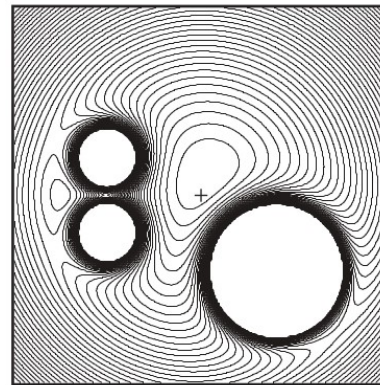
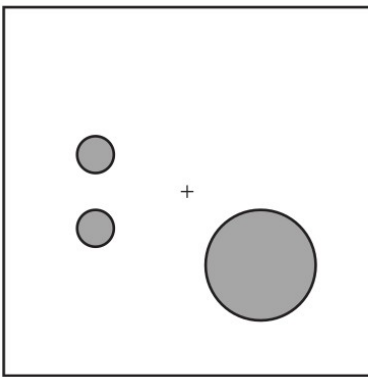
$$u_B(q) = -\frac{\partial \phi_B}{\partial q} = \frac{k}{d^3(q, \mathcal{B})} \frac{\partial d}{\partial q}$$

# Potential Field: composition

Then, the composition follows the principle of superposition and is:

$$\phi(q) = \phi_G(q) + \sum_i \phi_{\mathcal{B}_i}(q)$$
$$u(q) = u_G(q) + \sum_i u_{\mathcal{B}_i}(q)$$

The result, is a plan that simply follows the gradient to the goal.



# PF composition, local minima

- Composing different potential fields into a feedback-based plan can be problematic, since the objectives of the repulsion and attraction are opposed.
- If not correctly designed, PF can lead to **local minima**, and getting trapped before reaching the goal state.

Example:



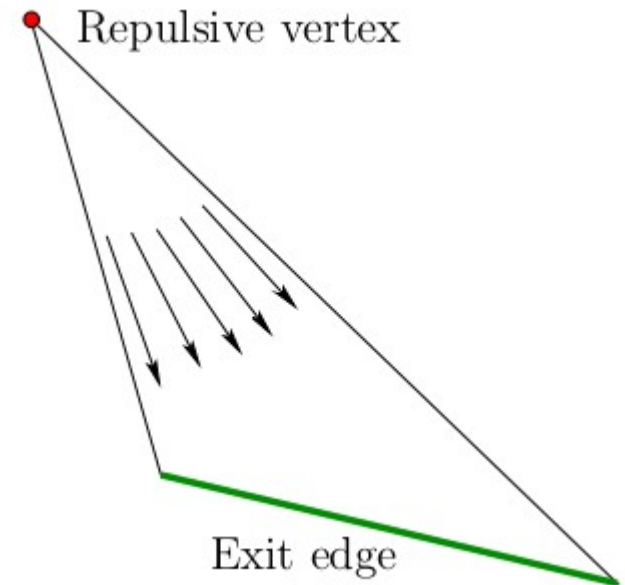
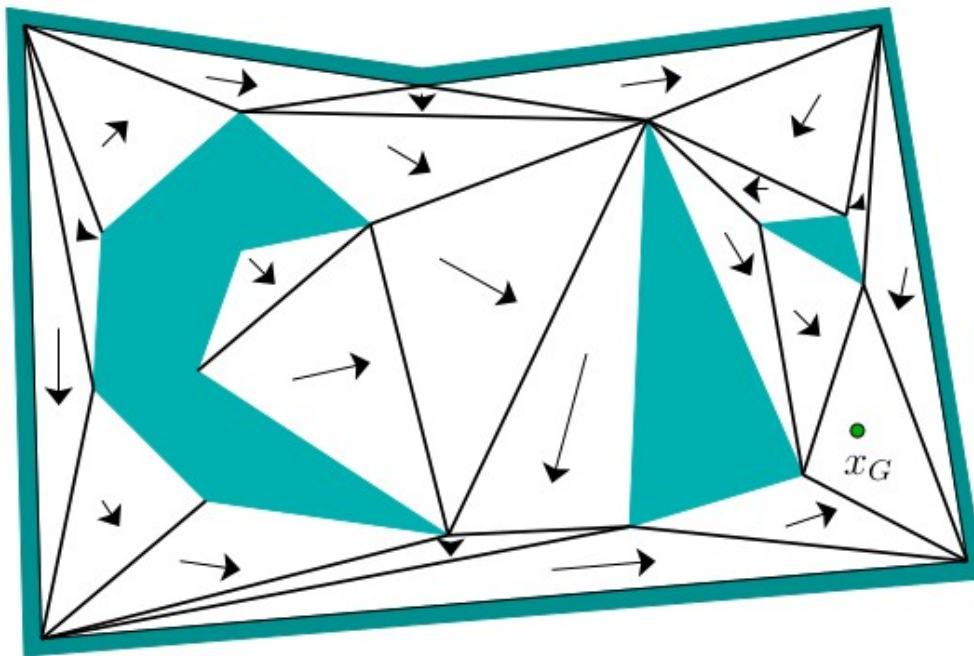
# Potential Fields in practice

- The motion solution can be calculated in real-time, for any state, instead of other planning strategies which require an open loop plan to be followed exhaustively (L02, L04).
- The issue of the local minima can be avoided using different techniques. Most of them focused on 2 strategies
  - Designing better PFs, for instance, optimal cost-to-go or harmonic functions
  - Hierarchical planning solution, for instance with A\* on top of the PF, very similar to having a controller.



# Cell complex and PFs

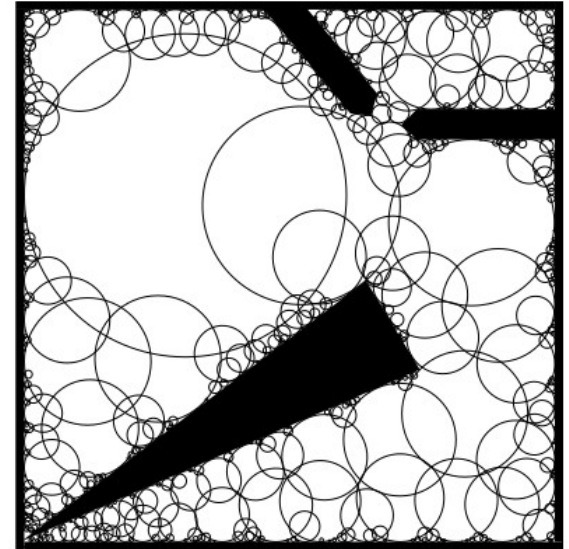
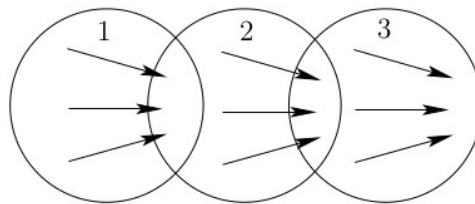
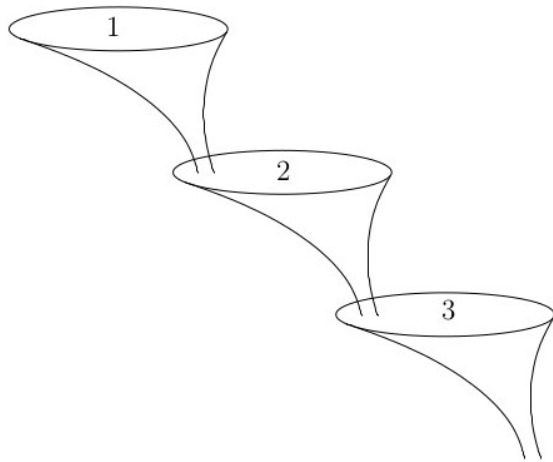
We can divide the space into a cell complex, where each cell has a simple shape to define a vector field. The plan is defined in two stages, discrete planning (sequence of cells) and vector field over each cell.



# Funnels

A navigation function and corresponding vector field can be designed as a composition of funnels. Regardless of the starting position in the funnel, the plan converges to a small goal region.

Then, we need to create a **cover**, the funnel domain to be as close as  $X$ . Sampling techniques could be used.



# Optimal cost-to-go function

We know that if we can calculate the optimal cost-to-go function, then the discrete feedback-based plan will be optimal as well (L05).

$$G_k^*(x_k) = \min_{u_k \in U(x_k)} \{l(x_k, u_k) + G_{k+1}^*(x_{k+1})\}$$

The problem with continuous spaces is that the number of elements is infinite, so how can we store them or compute  $G^*$ ?

- A solution is to using a family of parametric functions to approximate  $G^*$ , and update its parameters at each iteration.
  - Function candidates include polynomials, tensor-train, neural networks, etc.
- A second option is to interpolate neighbors. For grid, based discretization, the number of neighbors grows  $2^n$ . Simplexes alleviate the problem, but make access to elements more challenging.

# Trajectory Optimization

One can simply formulate the problem as a non-linear optimization of a trajectory. The general problem can be written as follows:

$$\begin{array}{ll} \text{find} & u(t), q(t), T \\ \text{minimizing} & J(u(t), q(t), T) \\ \text{subject to} & \dot{x}(t) = f(x(t), u(t)), \quad \forall t \in [0, T], \\ & u(t) \in \mathcal{U}, \quad \forall t \in [0, T], \\ & q(t) \in \mathcal{C}_{\text{free}}, \quad \forall t \in [0, T], \\ & x(0) = x_{\text{start}}, \\ & x(T) = x_{\text{goal}}. \end{array}$$

# Trajectory Optimization

To solve this problem, the control  $u(t)$ , the trajectory  $q(t)$  should be discretized.

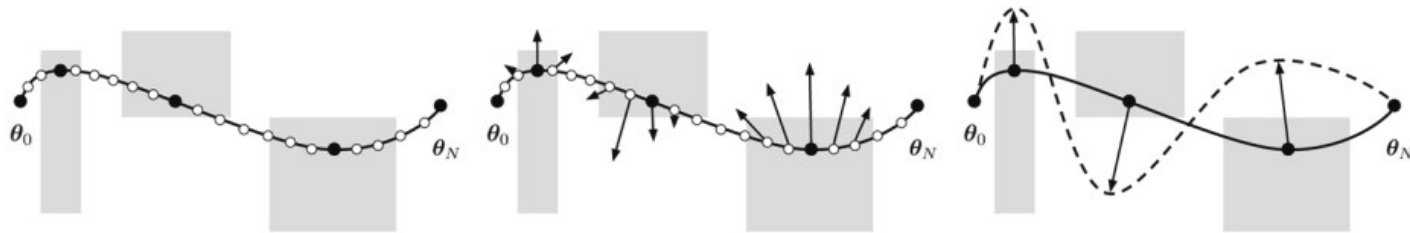
Three options for discretization:

- ➔ Parametrize the trajectory  $q(t)$ . In this case, we can solve the problem directly for the trajectory (parameters). Examples include splines, polynomials, Fourier series, etc. Controls are extracted from the equations of motion.
- ➔ Parametrize the control  $u(t)$ . Solve directly for controls and calculate the trajectory  $q(t)$  integrating the equations of motion.
- ➔ Parametrize both  $q(t)$  and  $u(t)$ . The problem here is how to satisfy the transition equations.

# Optimization as Inference

A recent variant of trajectory optimization, instead of using well-known non-linear solvers, some methods developed for inference problem could be used instead (unconstrained optimization).

This also changes the nature of the problem, now using Factor Graphs and Gaussian Processes to express trajectories.



# Ideas for the final project

We have uploaded to canvas several papers related to trajectory optimization (today's lecture).

Use them as a sample to select final projects.

Warning: most of them are related to dynamics, controls, etc, so choose them only if you are particularly interested.

