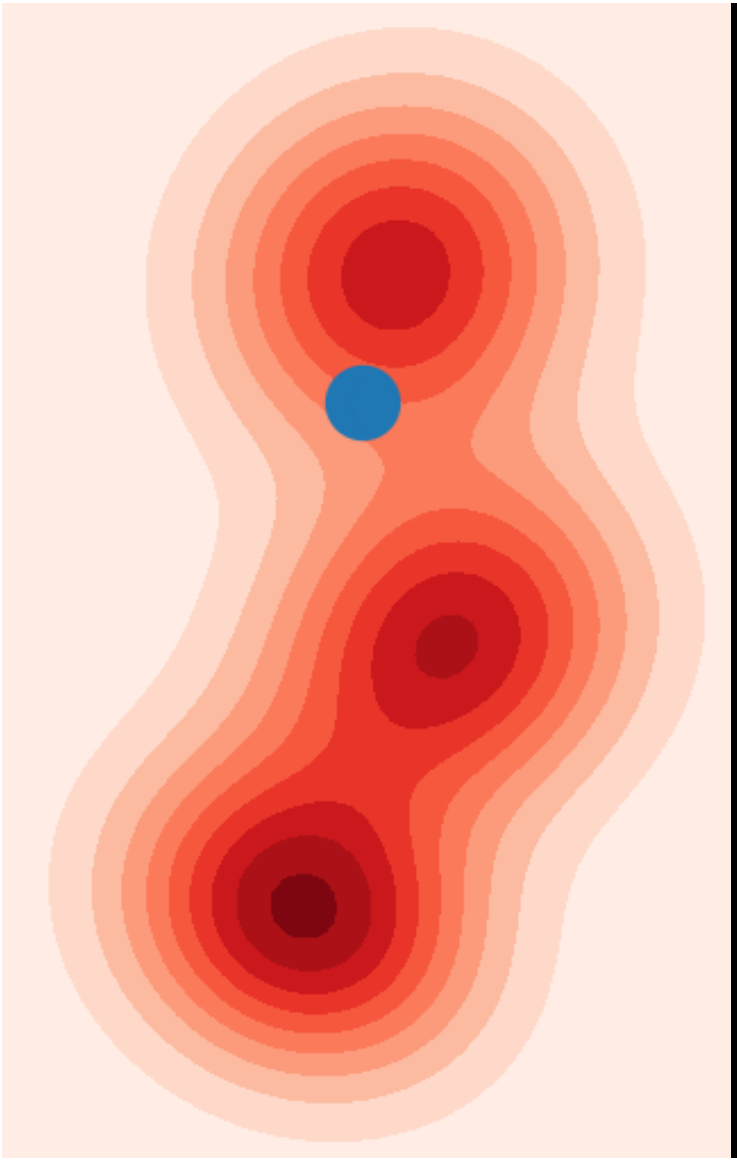


Ergodic Control Sandbox Code

Muchen Sun and Todd Murphey

2024.05.13

ICRA 2024 Tutorial on Ergodic Control

Northwestern

Github Repo:

https://tinyurl.com/ergodic-control

Sandbox Summary:

Introduction to the ergodic metric

Closed-form ergodic control

Trajectory optimization for ergodic control

Ergodic control with kernel functions

Ergodic metric for point cloud registration

Sandbox Summary:

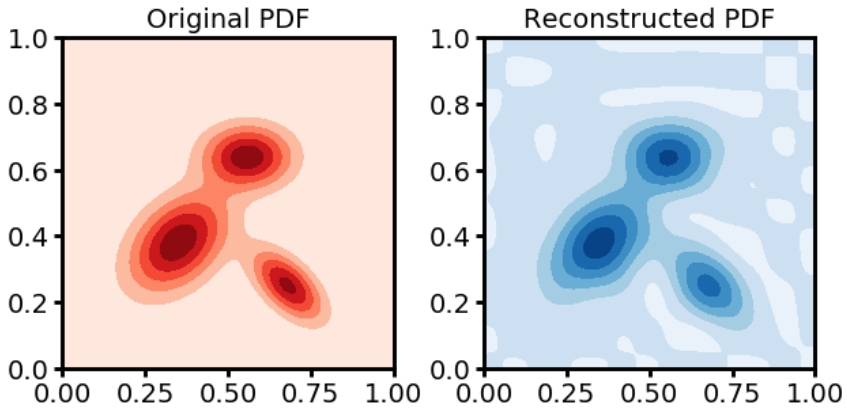
What the sandbox *provides*:

- Easy hands-on experience with ergodic control;
- Intuition behind different ergodic control approaches;
- Template for prototyping ergodic control for your project.

What the sandbox *does not* provide:

- Benchmark-level implementations;
- Application/Hardware-ready implementations;

Tutorial #0: Introduction to the ergodic metric



How to represent a target distribution;

How to define a Fourier basis function;

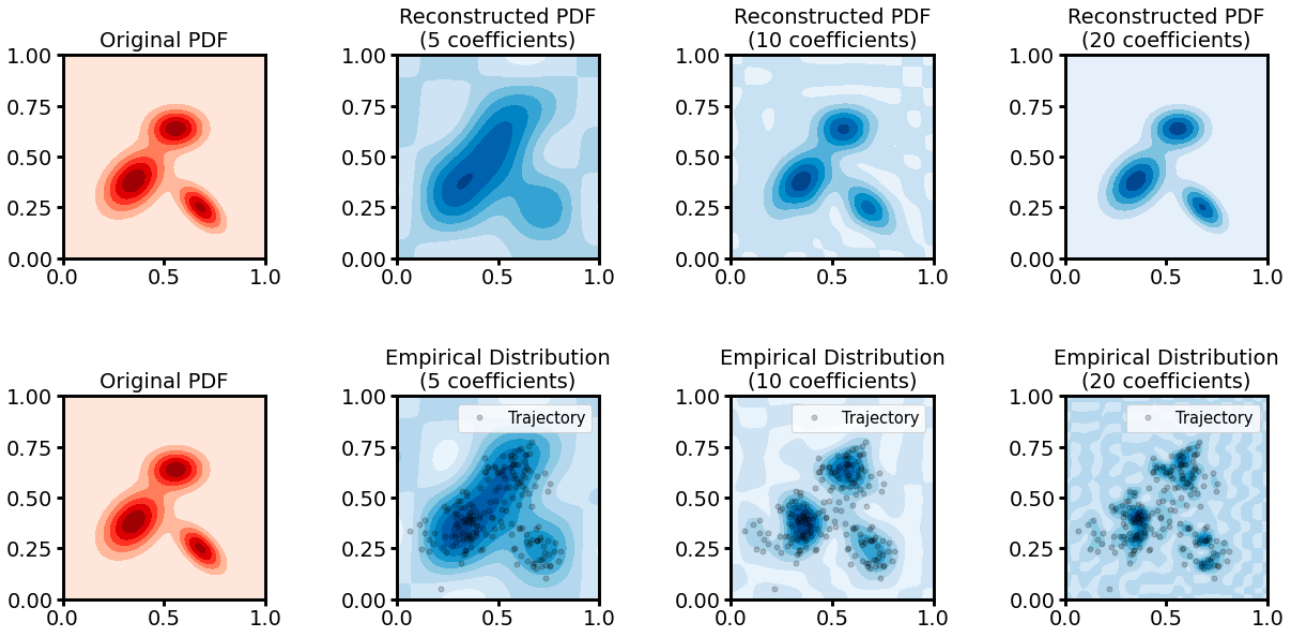
Calculating Fourier coefficients and reconstruct distributions from them;

Calculating the ergodic metric with a given trajectory.

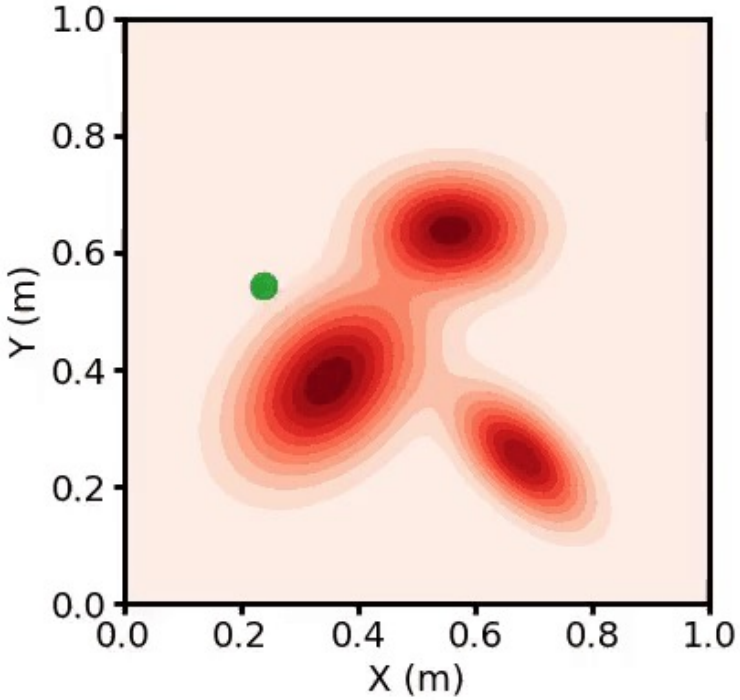
Tutorial #0: Introduction to the ergodic metric

You should play around with:

- Number of coefficients vs. the quality of distribution reconstruction.
- Compute vs. fidelity trade-off.



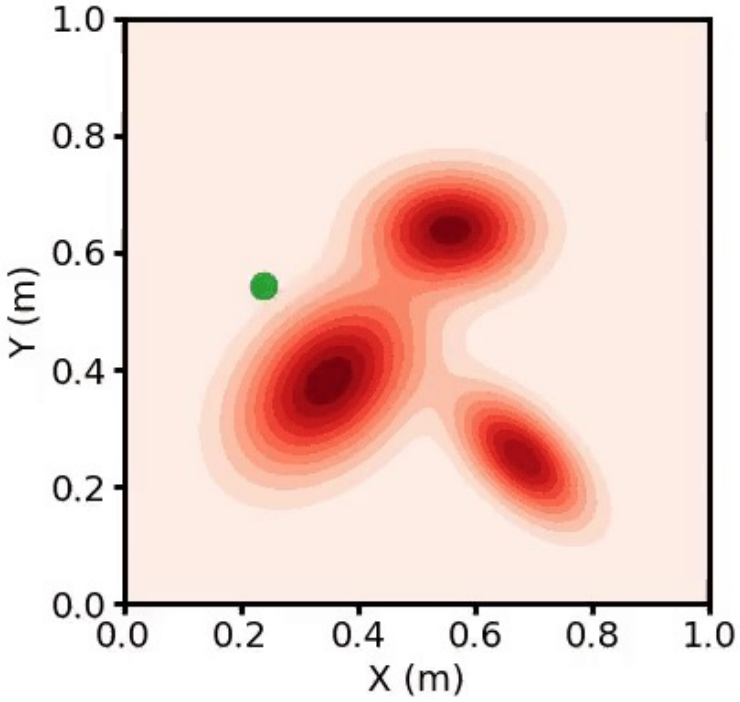
Tutorial #1: Closed-form ergodic control



Fast, closed-form ergodic control for systems with trivial dynamics;

This is where everyone should start with ergodic control.

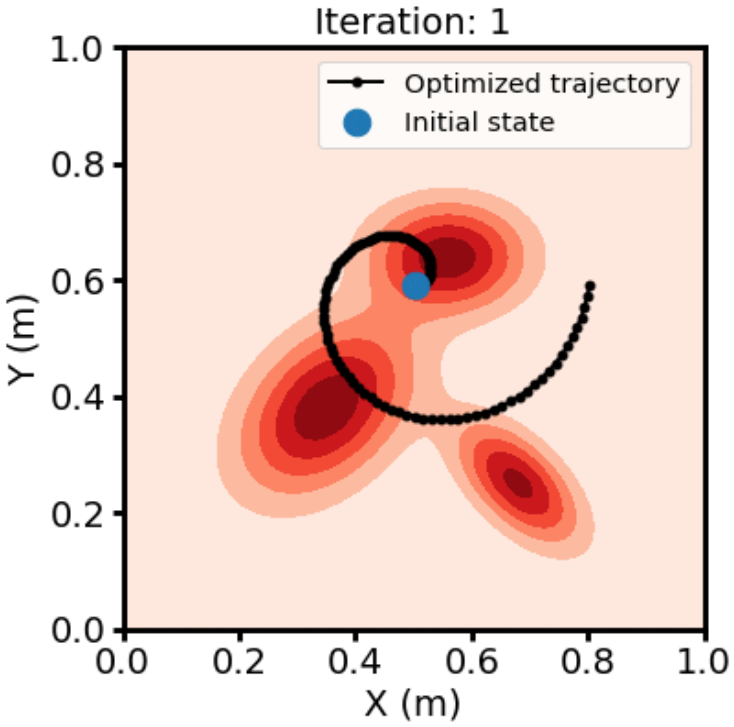
Tutorial #1: Closed-form ergodic control



You should play around with:

- Number of coefficients;
- Exploration time --- can you afford this amount of time in your application?

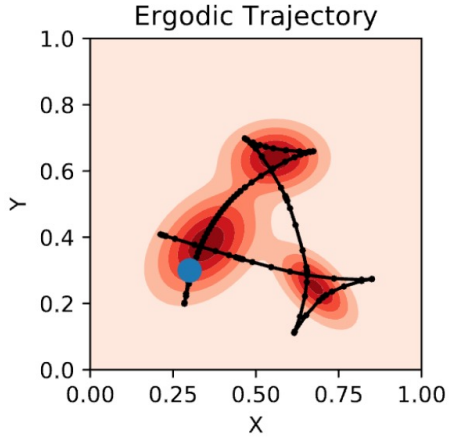
Tutorial #2: Trajectory optimization for ergodic control

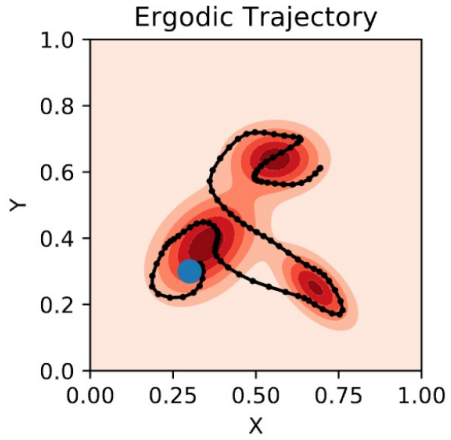


Implement iterative LQR (iLQR) algorithm for ergodic control;

Full iLQR template provided for future extensions;

Tutorial #2: Trajectory optimization for ergodic control

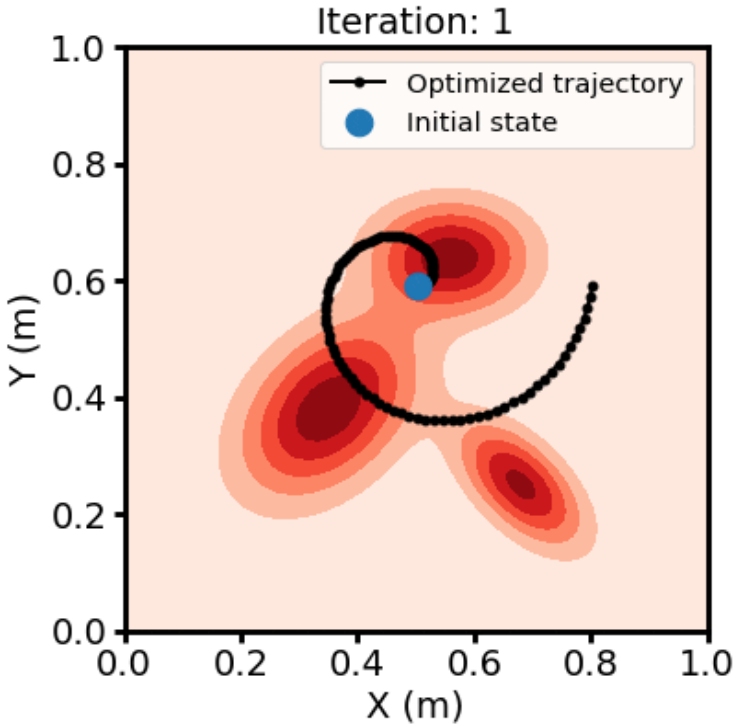




You should play around with:

- Different initial trajectory;
  - iLQR is only locally optimal, but ergodic metric is convex!
- Different dynamics;
- Different time horizons.

Tutorial #3: Ergodic control with kernel functions



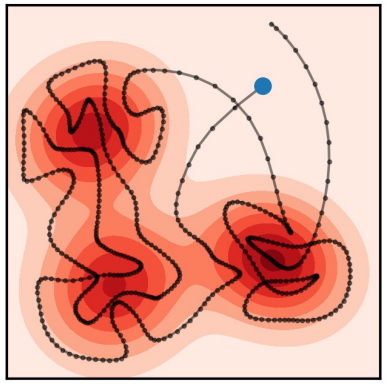
An alternative ergodic metric to the Fourier-based formula;

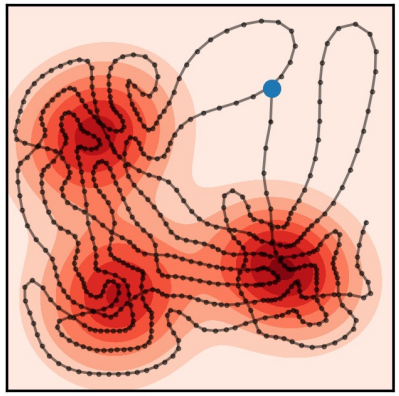
Example of auto-tuning kernel parameter;

Full iLQR implementation.

(This tutorial does not discuss the performance advantages and trade-offs of the kernel formula vs. the original ergodic control formula)

Tutorial #3: Ergodic control with kernel functions

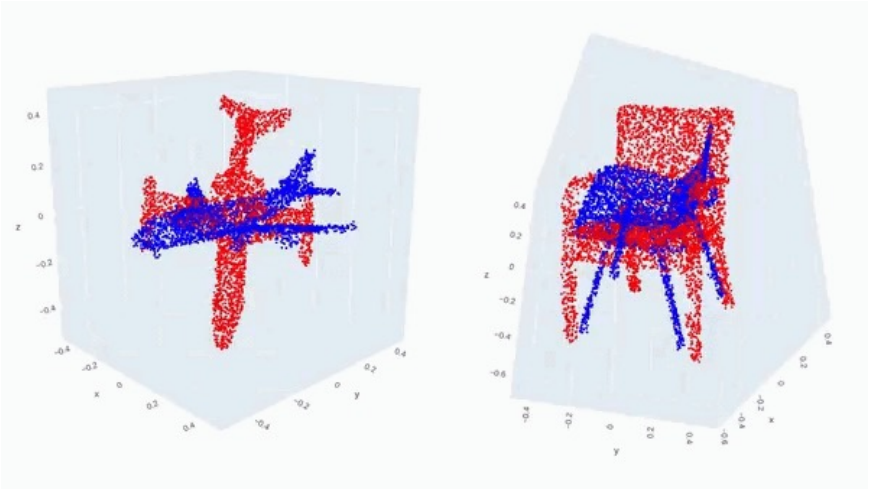




You should play around with:

- Different kernel parameters;
- Different weights between the likelihood maximization term and the uniform coverage term;
- *(You will need barrier functions for these.)*

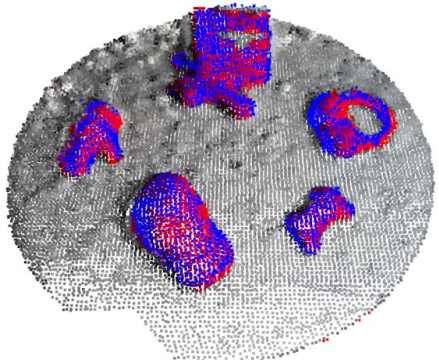
Bonus Tutorial: Ergodic metric for point cloud registration



Use the ergodic metric for robust point cloud registration with unknown correspondences;

Full gradient-descent template on Lie groups;

Bonus Tutorial: Ergodic metric for point cloud registration

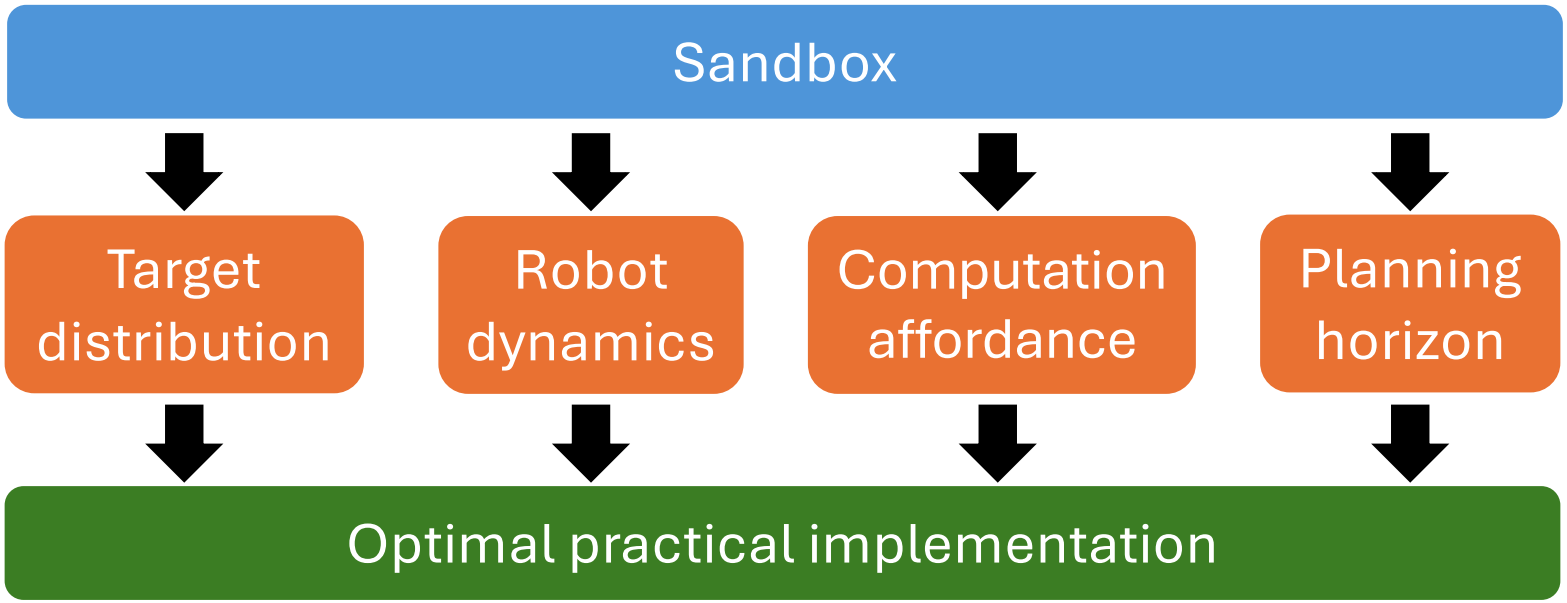


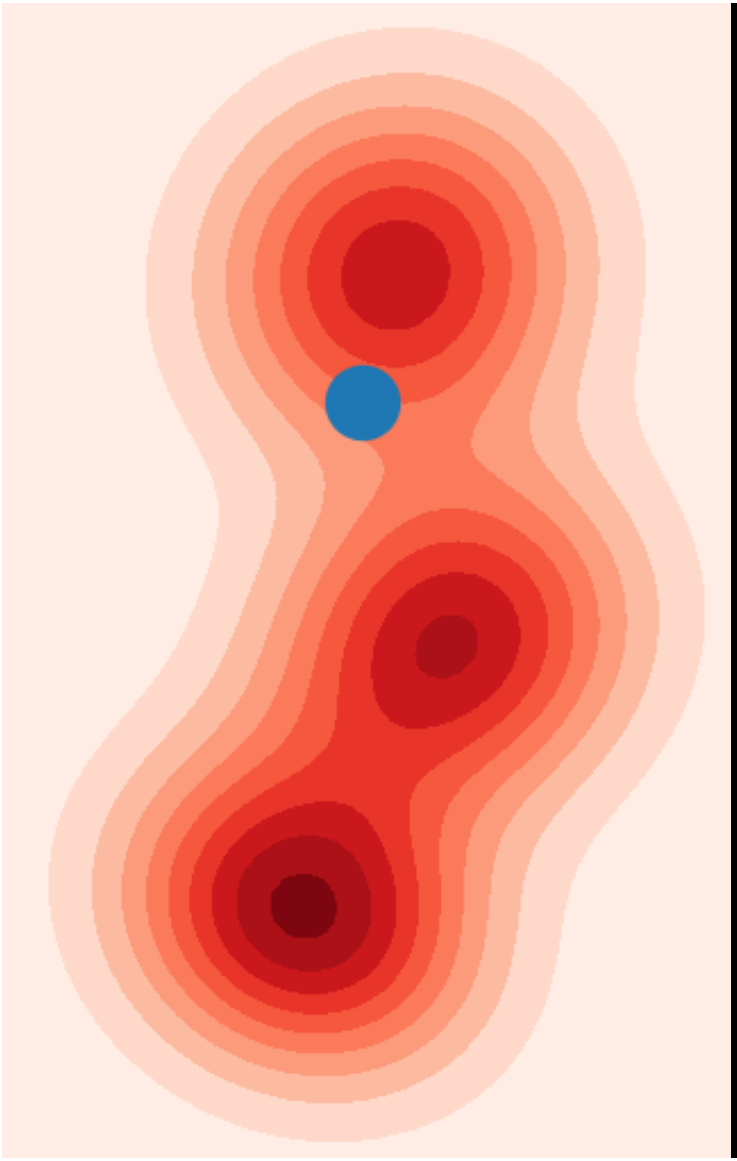
You should play around with:

- Your own point cloud!

Beyond Sandbox: What after for practitioners?

The sandbox serves as a perfect template to prototype different components of ergodic control.





Ergodic Control with Kernel Functions

Muchen Sun and Todd Murphey

2024.05.13

ICRA 2024 Tutorial on Ergodic Control

Northwestern

An alternative formula for ergodic metric

[Key Theorem] A dynamic system can asymptotically reach ergodicity through optimizing controls for the following objective:

$$\mathcal{E}_\phi(s(t)) = -\frac{2}{T}\int_0^T P(s(t))dt + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

An alternative formula for ergodic metric

[Key Theorem] A dynamic system can asymptotically reach ergodicity through optimizing controls for the following objective:

Target Spatial Distribution

$$\mathcal{E}_\phi(s(t)) = -\frac{2}{T}\int_0^T P(s(t))dt + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

An alternative formula for ergodic metric

[Key Theorem] A dynamic system can asymptotically reach ergodicity through optimizing controls for the following objective:

$$\mathcal{E}_\phi(s(t)) = \boxed{-\frac{2}{T}\int_0^T P(s(t))dt} + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

Likelihood maximization

An alternative formula for ergodic metric

[Key Theorem] A dynamic system can asymptotically reach ergodicity through optimizing controls for the following objective:

$$\mathcal{E}_\phi(s(t)) = -\frac{2}{T}\int_0^T P(s(t))dt + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

Kernel Function
(e.g., RBF Kernel)

An alternative formula for ergodic metric

[Key Theorem] A dynamic system can asymptotically reach ergodicity through optimizing controls for the following objective:

$$\mathcal{E}_\phi(s(t)) = -\frac{2}{T}\int_0^T P(s(t))dt + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

Uniform coverage
(entropy maximization)

An alternative formula for ergodic metric

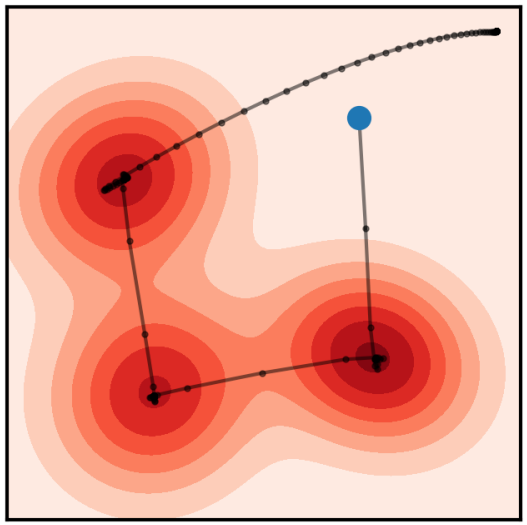
$$\mathcal{E}_\phi(s(t)) = -\frac{2}{T}\int_0^T P(s(t))dt + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

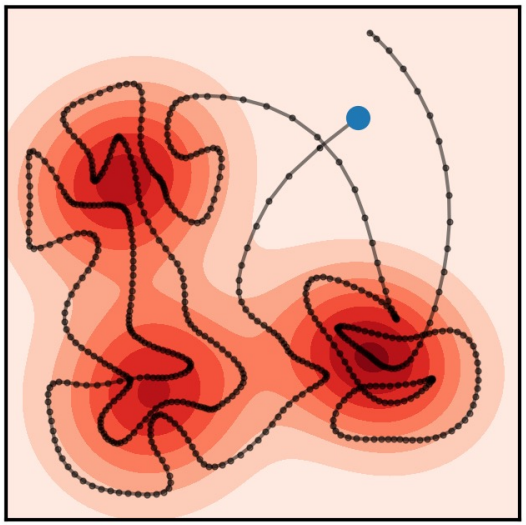
Why it matters: The kernel ergodic metric avoids decomposing the target distribution over the search space, improving the computation efficiency especially in high-dimensional spaces and non-trivial geometrical spaces (e.g., Lie groups).

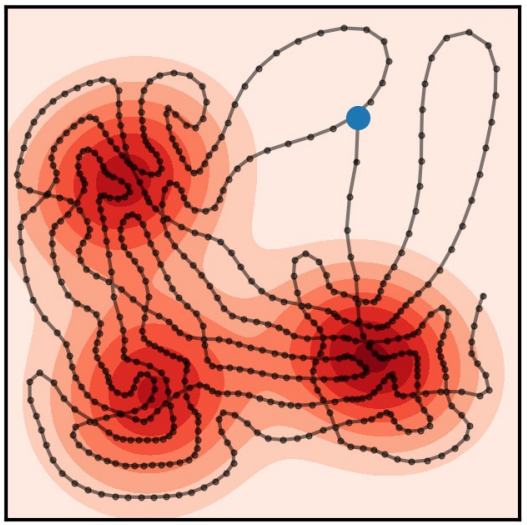
Kernel specification is tricky …

$$\mathcal{E}_\phi(s(t)) = -\frac{2}{T}\int_0^T P(s(t))dt + \frac{1}{T^2}\int_0^T\int_0^T \phi(s(t_1), s(t_2))dt_1 dt_2$$

$$\phi(s, s'; \theta) = \exp\left(-\frac{|s - s'|^2}{\theta}\right)$$



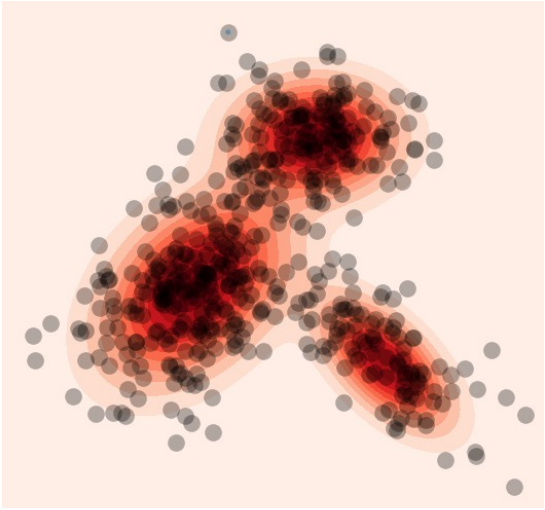




But we can optimize the kernel using samples!

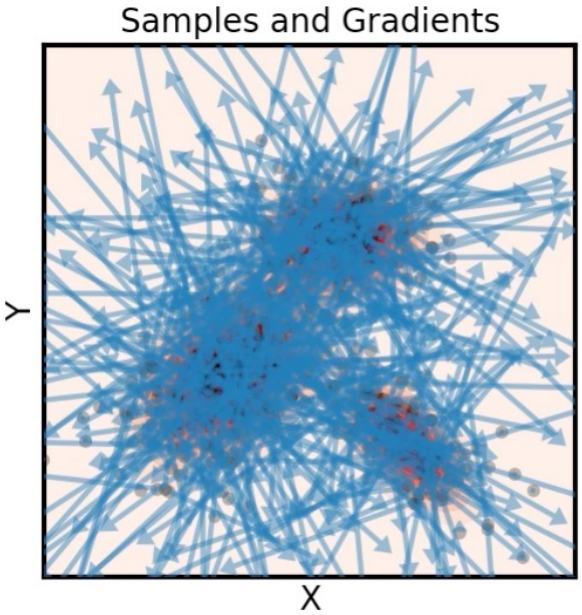
$$\theta^* = \arg\min_\theta \left| \frac{d}{d\mathbf{x}} \left(-\frac{2}{N} \sum_{i=1}^{N} P(x_i) + \frac{1}{N^2} \sum_{i=0}^{N} \sum_{j=0}^{N} \phi(x_i, x_j; \theta) \right) \right|^2$$

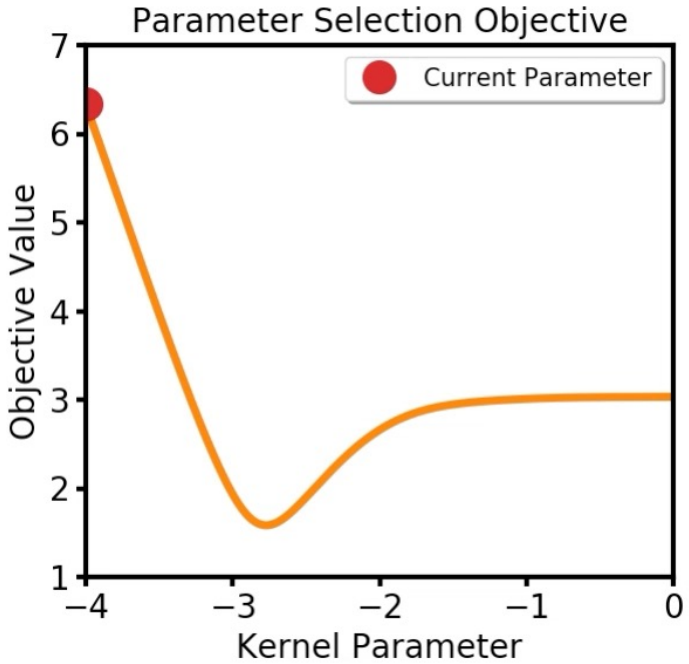
$$\mathbf{x} = [x_1, \ldots, x_N]^\top, x_i \sim P(x)$$

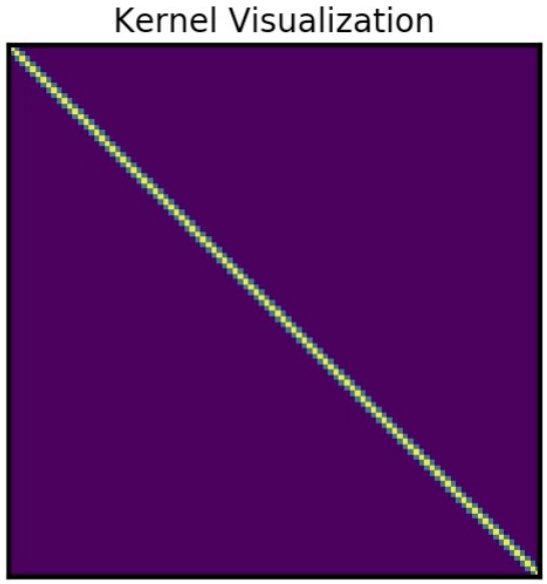


But we can optimize the kernel!

(We only use one parameter in this example, though kernels typically have 2 to 5 parameters in practice.)



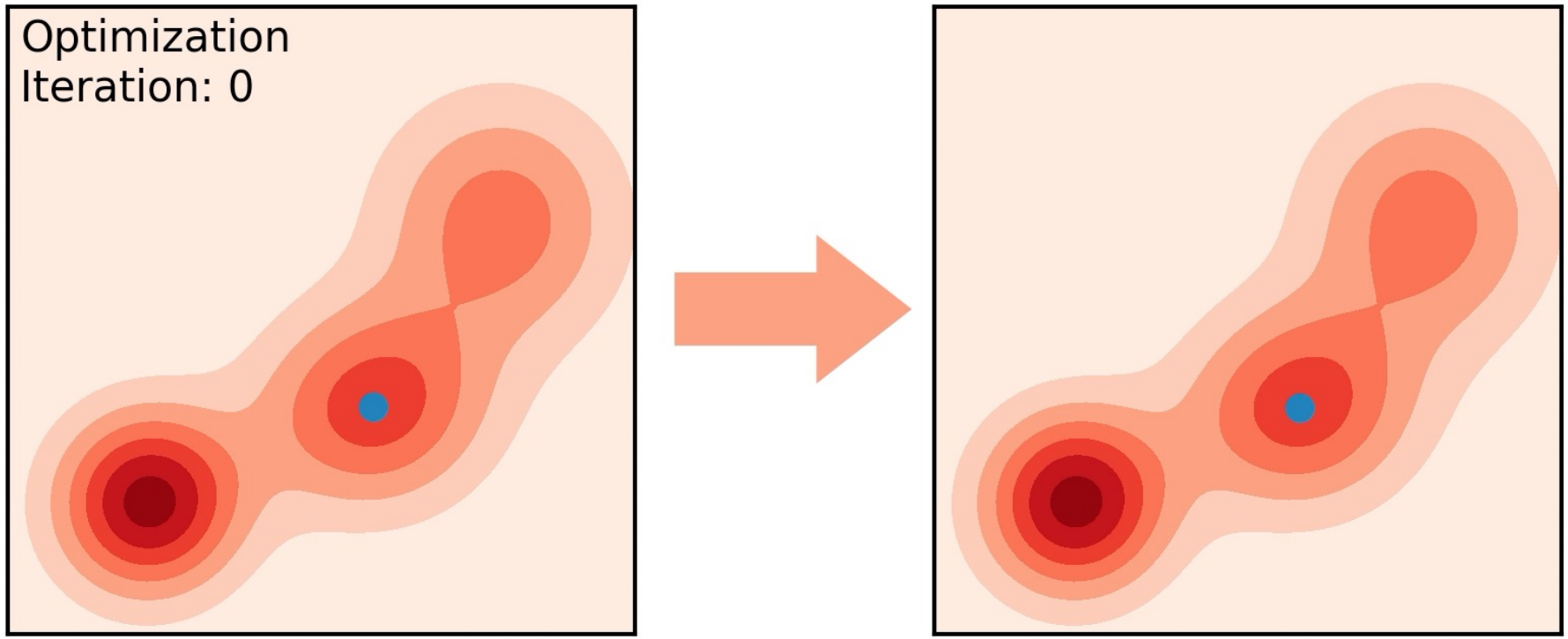




Example of kernel ergodic control

Trajectory optimization
(less than 0.1 second with C++)

Trajectory execution



Where are the samples from?

In general, if we can evaluate the probability density function, we can generate samples from the corresponding probability distribution.

Where are the samples from?

More importantly, there are scenarios where we have the samples (data) first, then extract the target distribution from it, such as in learning from demonstration applications.

Try it out on Google Colab!

arXiv Pre-print:

“*Fast Ergodic Search with Kernel Functions*”