Affine Geometric Heat Flow and Motion Planning for Dynamic Systems

Shenyu Liu, Yinai Fan, Mohamed-Ali Belabbas

Coordinated Science Laboratory,
University of Illinois at Urbana-Champaign, IL, USA

{*sliu113,yfan17,belabbas*}*@illinois.edu*

September 4th, 2019

Overview

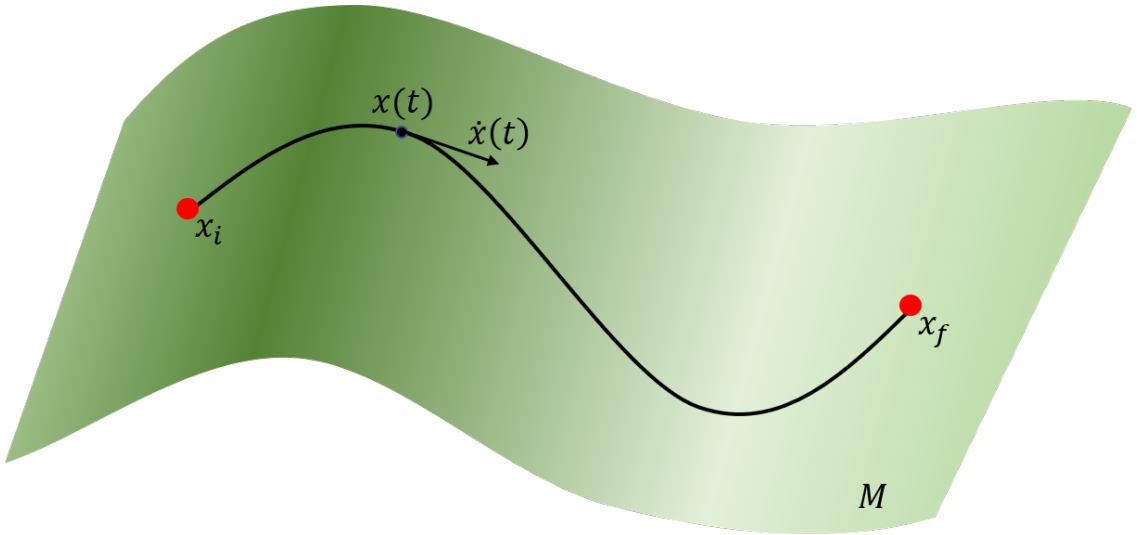
Introduction

Motion planning problem

Given a system

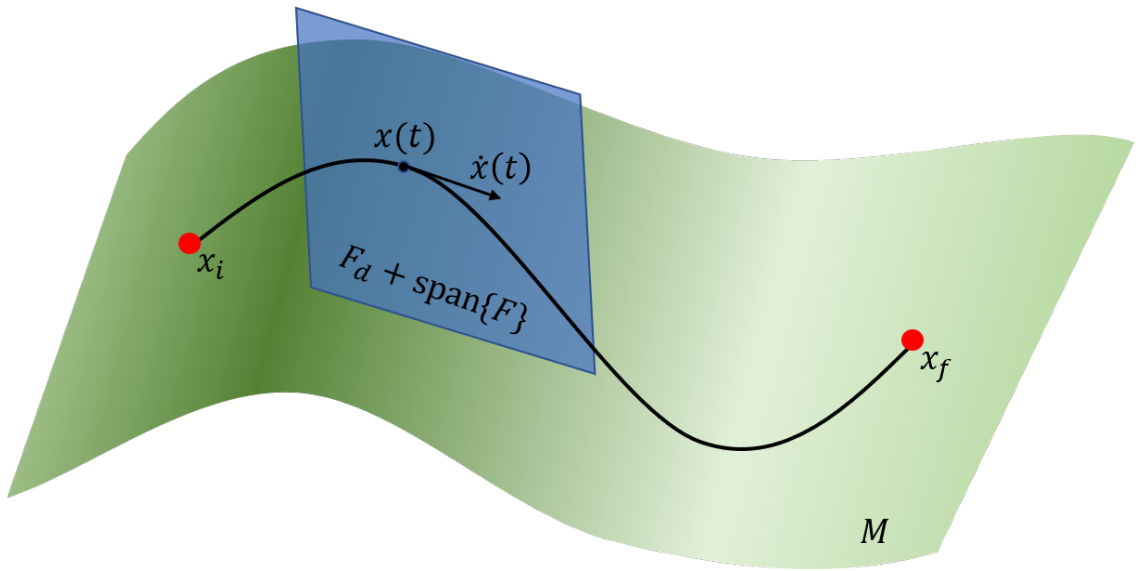
$$\dot{x} = f(x, u) \tag{1}$$

and two points $x_i, x_f \in M$, find a control $u^*(t)$ that steers the system from x_i to x_f in T units of time.



System with affine controls

$$\dot{x} = F_d(x) + F(x)u \tag{2}$$



Assumption A

Both $F_d(x), F(x)$ are assumed to be at least C^2, Lipschitz with constants L_1, L_2 respectively;
$F(x)$ is of rank m almost everywhere on M.

Literature review

- Motion planning has been widely studied (see, e.g., [Laumond, 1998], [LaValle, 2006])
- One of the early control papers which addresses the issue of motion planning for non-holonomic systems is [Brockett, 1982], where motion planning is stated as a sub-Riemannian geodesic problem. See also the monograph [Jean, 2014] for a recent survey of this line of work.
- Other motion planning methods include but are not limited to LQR-tree method [Tedrake et al., 2010], sum-of-square techniques [Majumdar and Tedrake, 2013]. motion primitives [Murphey, 2006] [Woodruff and Lynch, 2017], random sampling-based [Karaman and Frazzoli, 2011], graph-based [Kuffner et al., 2003] and optimization-based approaches [Dai et al., 2014], etc.

Literature review

- Motion planning has been widely studied (see, e.g., [Laumond, 1998], [LaValle, 2006])
- One of the early control papers which addresses the issue of motion planning for non-holonomic systems is [Brockett, 1982], where motion planning is stated as a sub-Riemannian geodesic problem. See also the monograph [Jean, 2014] for a recent survey of this line of work.
- Other motion planning methods include but are not limited to LQR-tree method [Tedrake et al., 2010], sum-of-square techniques [Majumdar and Tedrake, 2013]. motion primitives [Murphey, 2006] [Woodruff and Lynch, 2017], random sampling-based [Karaman and Frazzoli, 2011], graph-based [Kuffner et al., 2003] and optimization-based approaches [Dai et al., 2014], etc.

Literature review

- Motion planning has been widely studied (see, e.g., [Laumond, 1998], [LaValle, 2006])
- One of the early control papers which addresses the issue of motion planning for non-holonomic systems is [Brockett, 1982], where motion planning is stated as a sub-Riemannian geodesic problem. See also the monograph [Jean, 2014] for a recent survey of this line of work.
- Other motion planning methods include but are not limited to LQR-tree method [Tedrake et al., 2010], sum-of-square techniques [Majumdar and Tedrake, 2013]. motion primitives [Murphey, 2006] [Woodruff and Lynch, 2017], random sampling-based [Karaman and Frazzoli, 2011], graph-based [Kuffner et al., 2003] and optimization-based approaches [Dai et al., 2014], etc.

Challenges

Difficulties in motion planning

- non-holonomic dynamics,
- drift,
- constraints on the inputs/states.

The geometric approach proposed in our work can address all three difficulties.

Challenges

Difficulties in motion planning

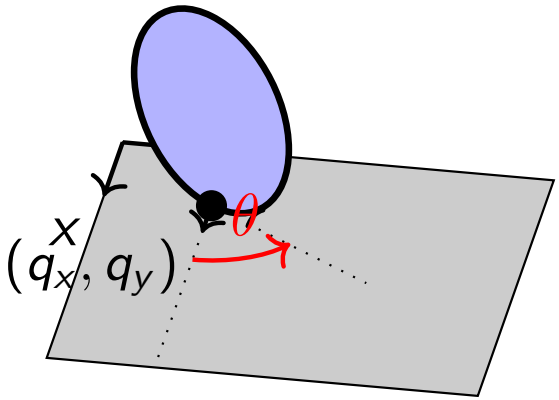
- non-holonomic dynamics,
- drift,
- constraints on the inputs/states.

The geometric approach proposed in our work can address all three difficulties.

Affine geometric heat flow

Core idea

"Deforming" a curve in order to make it feasible.



$$\underbrace{\begin{pmatrix} \dot{q}_x \\ \dot{q}_y \\ \dot{\theta} \end{pmatrix}}_{\dot{x}} = \underbrace{\begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix}}_{f_1} u_1 + \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{f_2} u_2. \qquad (3)$$

Core idea

"Deforming" a curve in order to minimize its "length".

- What is "length"? Answer: Riemannian metric – encodes dynamics, constraints, etc.
- How to "deform"? Answer: Homotopies – achieved by solving PDEs.

Core idea

"Deforming" a curve in order to minimize its "length".

- What is "length"? Answer: Riemannian metric – encodes dynamics, constraints, etc.
- How to "deform"? Answer: Homotopies – achieved by solving PDEs.

Core idea

"Deforming" a curve in order to minimize its "length".

- What is "length"? Answer: Riemannian metric – encodes dynamics, constraints, etc.
- How to "deform"? Answer: Homotopies – achieved by solving PDEs.

Riemannian metric

- A Riemannian metric on M is a family of positive definite matrices $G(x), x \in M$.
- A curve on M has length $\mathcal{L} = \int_0^T \sqrt{\dot{x}^\top G(x)\dot{x}}dt$ w.r.t. Riemannian metric G. e.g., identity matrix $G(x) \equiv I$ gives usual Euclidean length.
- To encode non-holonomic constraints and obstacles, we set

$$G(x) = b(x)(\bar{F}(x)^{-1})^\top D\bar{F}(x)^{-1},$$

where $b(x) \in \mathbb{R}$ is the barrier function, $D = \mathrm{diag}(\underbrace{\lambda, \cdots, \lambda}_{n-m}, \underbrace{1, \cdots, 1}_{m})$ for some large $\lambda > 0$ and $\bar{F}(x) = (F_c(x)|F(x)) \in \mathbb{R}^{n\times n}$ so that it is full rank.

- $\dot{x}^\top G(x)\dot{x} \approx b(x)\left(\left|\mathbf{P}_{\mathcal{F}}\dot{x}\right|^2 + \lambda\left|\mathbf{P}_{\mathcal{F}^\perp}\dot{x}\right|^2\right)$

Riemannian metric

- A Riemannian metric on M is a family of positive definite matrices $G(x), x \in M$.
- A curve on M has length $\mathcal{L} = \int_0^T \sqrt{\dot{x}^\top G(x)\dot{x}}dt$ w.r.t. Riemannian metric G. e.g., identity matrix $G(x) \equiv I$ gives usual Euclidean length.
- To encode non-holonomic constraints and obstacles, we set

$$G(x) = b(x)(\bar{F}(x)^{-1})^\top D\bar{F}(x)^{-1},$$

where $b(x) \in \mathbb{R}$ is the barrier function, $D = \mathrm{diag}(\underbrace{\lambda, \cdots, \lambda}_{n-m}, \underbrace{1, \cdots, 1}_{m})$

for some large $\lambda > 0$ and $\bar{F}(x) = (F_c(x)|F(x)) \in \mathbb{R}^{n\times n}$ so that it is full rank.

- $\dot{x}^\top G(x)\dot{x} \approx b(x)\left(\left|\mathbf{P}_{\mathcal{F}}\dot{x}\right|^2 + \lambda\left|\mathbf{P}_{\mathcal{F}^\perp}\dot{x}\right|^2\right)$

Riemannian metric

- A Riemannian metric on M is a family of positive definite matrices $G(x), x \in M$.
- A curve on M has length $\mathcal{L} = \int_0^T \sqrt{\dot{x}^\top G(x)\dot{x}}dt$ w.r.t. Riemannian metric G. e.g., identity matrix $G(x) \equiv I$ gives usual Euclidean length.
- To encode non-holonomic constraints and obstacles, we set

$$G(x) = b(x)(\bar{F}(x)^{-1})^\top D\bar{F}(x)^{-1},$$

where $b(x) \in \mathbb{R}$ is the barrier function, $D = \mathrm{diag}(\underbrace{\lambda, \cdots, \lambda}_{n-m}, \underbrace{1, \cdots, 1}_{m})$ for some large $\lambda > 0$ and $\bar{F}(x) = (F_c(x)|F(x)) \in \mathbb{R}^{n\times n}$ so that it is full rank.

- $\dot{x}^\top G(x)\dot{x} \approx b(x)\left(\left|\mathbf{P}_{\mathcal{F}}\dot{x}\right|^2 + \lambda\left|\mathbf{P}_{\mathcal{F}^\perp}\dot{x}\right|^2\right)$

Riemannian metric

- A Riemannian metric on M is a family of positive definite matrices $G(x), x \in M$.
- A curve on M has length $\mathcal{L} = \int_0^T \sqrt{\dot{x}^\top G(x)\dot{x}}dt$ w.r.t. Riemannian metric G. e.g., identity matrix $G(x) \equiv I$ gives usual Euclidean length.
- To encode non-holonomic constraints and obstacles, we set

$$G(x) = b(x)(\bar{F}(x)^{-1})^\top D\bar{F}(x)^{-1},$$

where $b(x) \in \mathbb{R}$ is the barrier function, $D = \mathrm{diag}(\underbrace{\lambda, \cdots, \lambda}_{n-m}, \underbrace{1, \cdots, 1}_{m})$ for some large $\lambda > 0$ and $\bar{F}(x) = (F_c(x)|F(x)) \in \mathbb{R}^{n\times n}$ so that it is full rank.

- $\dot{x}^\top G(x)\dot{x} \approx b(x)\left(\left|\mathbf{P}_{\mathcal{F}}\dot{x}\right|^2 + \lambda\left|\mathbf{P}_{\mathcal{F}^\perp}\dot{x}\right|^2\right)$

Homotopies

$$x(t,s) : [0,T] \times [0,\infty) \to M$$

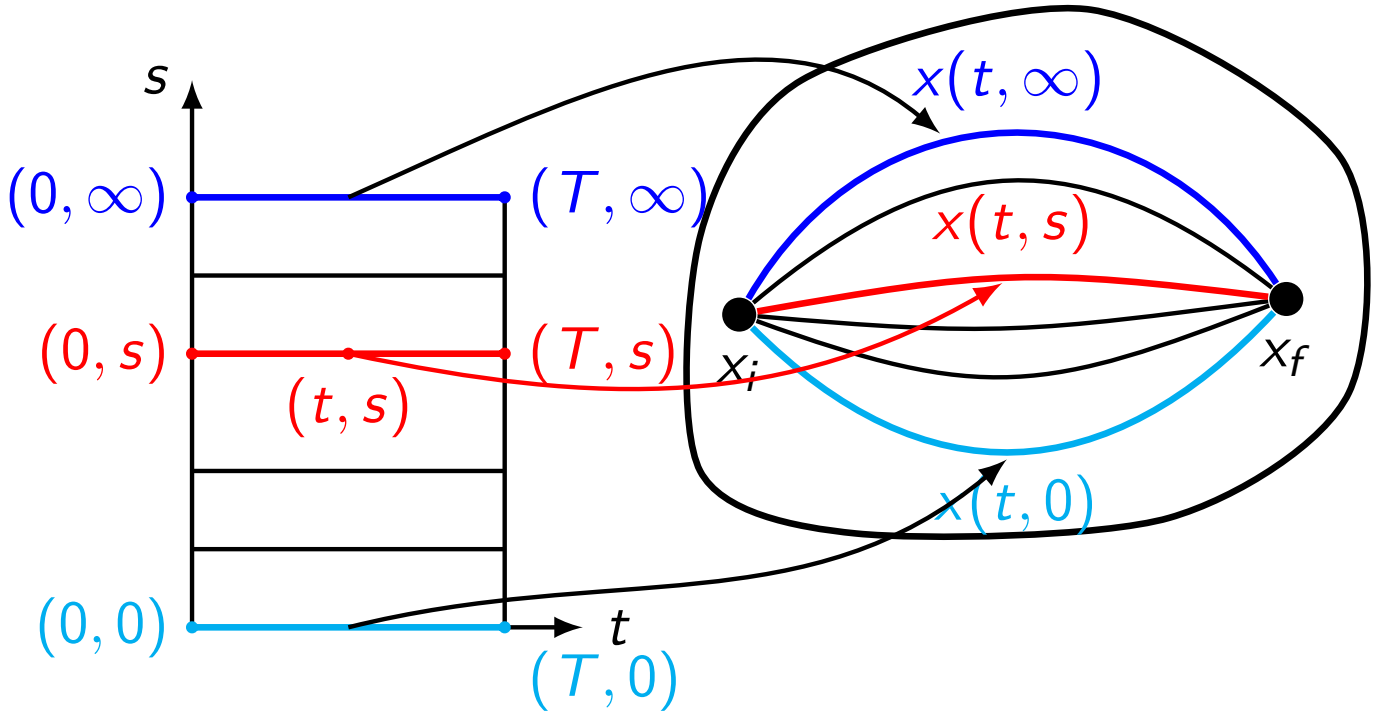


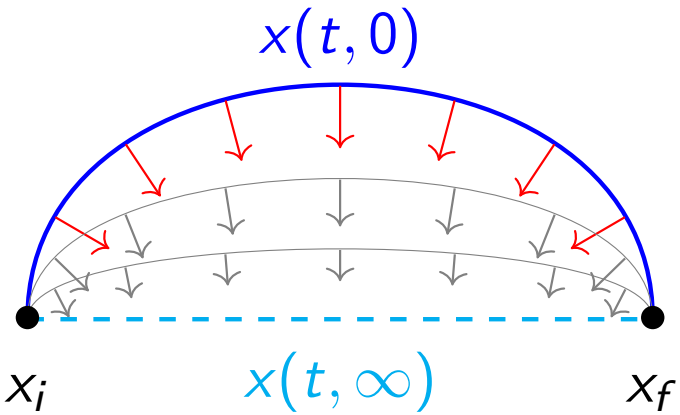
Figure: Homotopy of trajectories joining x_i to x_f in M.

Geometric heat flow (GHF)

For a driftless system,

$$\frac{\partial x(t,s)}{\partial s} = \nabla_{\dot{x}(t,s)} \dot{x}(t,s) \tag{4}$$

yields a curve of minimal length when $s \to \infty$ [Jost, 1995].



$$\nabla_f g := \frac{df}{dt} + \sum_{i,j,k} \Gamma_{ij}^k f_i g_j e^k,$$

$$\Gamma_{jk}^i(x) := \frac{1}{2} \sum_l (G^{-1})_{il} \left(\frac{\partial G_{lj}}{\partial x_k} + \frac{\partial G_{lk}}{\partial x_j} - \frac{\partial G_{jk}}{\partial x_l} \right)$$

Affine geometric heat flow

Taking inspiration from the GHF, we introduce AGHF as below:

$$\frac{\partial x(t,s)}{\partial s} = \nabla_{\dot{x}(t,s)} \left(\dot{x}(t,s) - F_d\right) + r(x(t,s), \dot{x}(t,s)) \tag{5}$$

where

$$r(x,\dot{x}) = G^{-1}\left(\left(\frac{\partial F_d}{\partial x}\right)^{\top} G(\dot{x} - F_d) + \frac{1}{2}\begin{pmatrix} (\dot{x} - F_d)^{\top}\frac{\partial G}{\partial x_1}F_d \\ \vdots \\ (\dot{x} - F_d)^{\top}\frac{\partial G}{\partial x_n}F_d \end{pmatrix}\right)$$

Geometric interpretation

$$\frac{\partial x(t,s)}{\partial s} = \nabla_{\dot{x}(t,s)} \left(\dot{x}(t,s) - F_d \right) + r(x(t,s), \dot{x}(t,s)) \tag{5}$$

- $\nabla_{\dot{x}} \left(\dot{x} - F_d \right)$ is the covariant derivative of $\dot{x} - F_d$ in the direction $\dot{x}$, which updates the curve in the direction of decreasing its "curvature" and hence minimizing the "length".
- $-r(x, \dot{x})$ is the scaled gradient of the point-wise map

$$P_f : M \to \mathbb{R} : x \mapsto \langle F_d(x) - f, F_d(x) \rangle$$

with $f = \dot{x}$ and $\langle f, h \rangle := f^\top G(x) h$. This map reaches its minimal value when $F_d(x)$ is aligned with $\dot{x}$.

Geometric interpretation

$$\frac{\partial x(t,s)}{\partial s} = \nabla_{\dot{x}(t,s)} \left(\dot{x}(t,s) - F_d \right) + r(x(t,s), \dot{x}(t,s)) \tag{5}$$

- $\nabla_{\dot{x}} \left(\dot{x} - F_d \right)$ is the covariant derivative of $\dot{x} - F_d$ in the direction $\dot{x}$, which updates the curve in the direction of decreasing its "curvature" and hence minimizing the "length".
- $-r(x, \dot{x})$ is the scaled gradient of the point-wise map

$$P_f : M \to \mathbb{R} : x \mapsto \langle F_d(x) - f, F_d(x) \rangle$$

with $f = \dot{x}$ and $\langle f, h \rangle := f^\top G(x) h$. This map reaches its minimal value when $F_d(x)$ is aligned with $\dot{x}$.

Geometric interpretation

$$\frac{\partial x(t,s)}{\partial s} = \nabla_{\dot{x}(t,s)} \left(\dot{x}(t,s) - F_d \right) + r(x(t,s), \dot{x}(t,s)) \tag{5}$$

- $\nabla_{\dot{x}} \left(\dot{x} - F_d \right)$ is the covariant derivative of $\dot{x} - F_d$ in the direction $\dot{x}$, which updates the curve in the direction of decreasing its "curvature" and hence minimizing the "length".
- $-r(x, \dot{x})$ is the scaled gradient of the point-wise map

$$P_f : M \to \mathbb{R} : x \mapsto \langle F_d(x) - f, F_d(x) \rangle$$

with $f = \dot{x}$ and $\langle f, h \rangle := f^\top G(x) h$. This map reaches its minimal value when $F_d(x)$ is aligned with $\dot{x}$.

Convergence of AGHF

Our AGHF minimizes the action functional

$$\mathcal{A}(x(\cdot)) := \frac{1}{2}\int_0^T (\dot{x} - F_d(x))^\top G(x)(\dot{x} - F_d(x))dt \tag{6}$$

Lemma

Let $x^*(t)$ be a steady-state solution of the AGHF (5). Then $x^*(t)$ is an extremal curve for $\mathcal{A}$ in (6). Furthermore, $\mathcal{A}$ decreases along the solutions of the AGHF; i.e. if $x(t,s)$ is such a solution, then $\frac{d}{ds}\mathcal{A}(x(\cdot,s)) \leq 0$, and equality holds only if $x(\cdot,s)$ is an extremal curve for $\mathcal{A}$.

Algorithm & theoretical guarantees

Algorithm

Step 1: Encode system dynamics, state constraints into the Riemannian metric G;

Step 2: Solve the AGHF (5) with boundary conditions

$$x(0,s) = x_i, x(T,s) = x_f \quad \forall s \geq 0$$

and an initial condition

$$x(t,0) = y(t), \quad t \in [0,T]$$

for some $y(\cdot) \in \mathcal{X}'$;

Step 3: Evaluate

$$u(t) := F(x(t,s_{\max}))^{\dagger}(\dot{x}(t,s_{\max}) - F_d(x(t,s_{\max})). \tag{7}$$

Output: The control $u(t)$ obtained in (7) is our solution to the motion planning problem. When integrating (2) with initial state x_i and input $u(t)$, the integrated path $\tilde{x}(t)$ approximately ends with $\tilde{x}(T) \approx x_f$.

Algorithm

Step 1: Encode system dynamics, state constraints into the Riemannian metric G;

Step 2: Solve the AGHF (5) with boundary conditions

$$x(0,s) = x_i, x(T,s) = x_f \quad \forall s \geq 0$$

and an initial condition

$$x(t,0) = y(t), \quad t \in [0,T]$$

for some $y(\cdot) \in \mathcal{X}'$;

Step 3: Evaluate

$$u(t) := F(x(t,s_{\max}))^{\dagger}(\dot{x}(t,s_{\max}) - F_d(x(t,s_{\max})). \tag{7}$$

Output: The control $u(t)$ obtained in (7) is our solution to the motion planning problem. When integrating (2) with initial state x_i and input $u(t)$, the integrated path $\tilde{x}(t)$ approximately ends with $\tilde{x}(T) \approx x_f$.

Algorithm

Step 1: Encode system dynamics, state constraints into the Riemannian metric G;

Step 2: Solve the AGHF (5) with boundary conditions

$$x(0,s) = x_i, x(T,s) = x_f \quad \forall s \geq 0$$

and an initial condition

$$x(t,0) = y(t), \quad t \in [0,T]$$

for some $y(\cdot) \in \mathcal{X}'$;

Step 3: Evaluate

$$u(t) := F(x(t,s_{\max}))^{\dagger}(\dot{x}(t,s_{\max}) - F_d(x(t,s_{\max})). \tag{7}$$

Output: The control $u(t)$ obtained in (7) is our solution to the motion planning problem. When integrating (2) with initial state x_i and input $u(t)$, the integrated path $\tilde{x}(t)$ approximately ends with $\tilde{x}(T) \approx x_f$.

Algorithm

Step 1: Encode system dynamics, state constraints into the Riemannian metric G;

Step 2: Solve the AGHF (5) with boundary conditions

$$x(0,s) = x_i, x(T,s) = x_f \quad \forall s \geq 0$$

and an initial condition

$$x(t,0) = y(t), \quad t \in [0,T]$$

for some $y(\cdot) \in \mathcal{X}'$;

Step 3: Evaluate

$$u(t) := F(x(t,s_{\max}))^{\dagger}(\dot{x}(t,s_{\max}) - F_d(x(t,s_{\max})). \qquad (7)$$

Output: The control $u(t)$ obtained in (7) is our solution to the motion planning problem. When integrating (2) with initial state x_i and input $u(t)$, the integrated path $\tilde{x}(t)$ approximately ends with $\tilde{x}(T) \approx x_f$.

Algorithm

Step 1: Encode system dynamics, state constraints into the Riemannian metric G;

Step 2: Solve the AGHF (5) with boundary conditions

$$x(0,s) = x_i, x(T,s) = x_f \quad \forall s \geq 0$$

and an initial condition

$$x(t,0) = y(t), \quad t \in [0,T]$$

for some $y(\cdot) \in \mathcal{X}'$;

Step 3: Evaluate

$$u(t) := F(x(t,s_{\max}))^{\dagger}(\dot{x}(t,s_{\max}) - F_d(x(t,s_{\max})). \qquad (7)$$

Output: The control $u(t)$ obtained in (7) is our solution to the motion planning problem. When integrating (2) with initial state x_i and input $u(t)$, the integrated path $\tilde{x}(t)$ approximately ends with $\tilde{x}(T) \approx x_f$.

On convergence guarantees for motion planning

Theorem

Consider the system (2) *and let* $x_i, x_f \in \mathbb{R}^n$. *Assume that the motion planning problem from* x_i *to* x_f *is feasible and that Assumption A is met. Then there exists* $C > 0$ *such that for any* $\lambda > 0$, *there exists an open set* $\Omega_\lambda \subseteq \mathcal{X}'$ *(with respect to* $\|\cdot\|_{AC}$*) so that as long as the initial curve* $y \in \Omega_\lambda$, *the integrated path* $\tilde{x}(t)$ *from our algorithm with sufficiently large* $s_{\max}$ *has the property that*

$$|\tilde{x}(T) - x_f| \leq \sqrt{\frac{3TC}{\lambda}} \exp\left(\frac{3T}{2}(L_2^2 T + L_1^2 C)\right). \tag{8}$$

Remarks

$$|\tilde{x}(T) - x_f| \leq \sqrt{\frac{3TC}{\lambda}} \exp\left(\frac{3T}{2}(L_2^2 T + L_1^2 C)\right) \tag{8}$$

- Our algorithm gives a solution to the relaxed motion planning problem:

Relaxed motion planning problem

Given $x_i, x_f \in \mathbb{R}^n$, $T > 0, \epsilon > 0$, find an integrable u (potentially continuous u) such that the corresponding solution of (2) with initial condition $x(0) = x_i$ satisfies $|x(T) - x_f| \leq \epsilon$.

Remarks

$$|\tilde{x}(T) - x_f| \leq \sqrt{\frac{3TC}{\lambda}} \exp\left(\frac{3T}{2}(L_2^2 T + L_1^2 C)\right) \tag{8}$$

- Our algorithm gives a solution to the relaxed motion planning problem:

Relaxed motion planning problem

Given $x_i, x_f \in \mathbb{R}^n$, $T > 0, \epsilon > 0$, find an integrable u (potentially continuous u) such that the corresponding solution of (2) with initial condition $x(0) = x_i$ satisfies $|x(T) - x_f| \leq \epsilon$.

Examples

Unicycle with constant linear velocity

$$\underbrace{\begin{pmatrix} \dot{q}_x \\ \dot{q}_y \\ \dot{\theta} \end{pmatrix}}_{\dot{x}} = \underbrace{\begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix}}_{F_d} + \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{F} u. \tag{9}$$

Figure: Two different scenarios for parallel parking. Both start with $x_i = (0, 0, 0)^\top$. Left: $x_f = (0, 1, 0)^\top$. Right: $x_f = (0, 1, 2\pi)^\top$

Dynamic unicycle

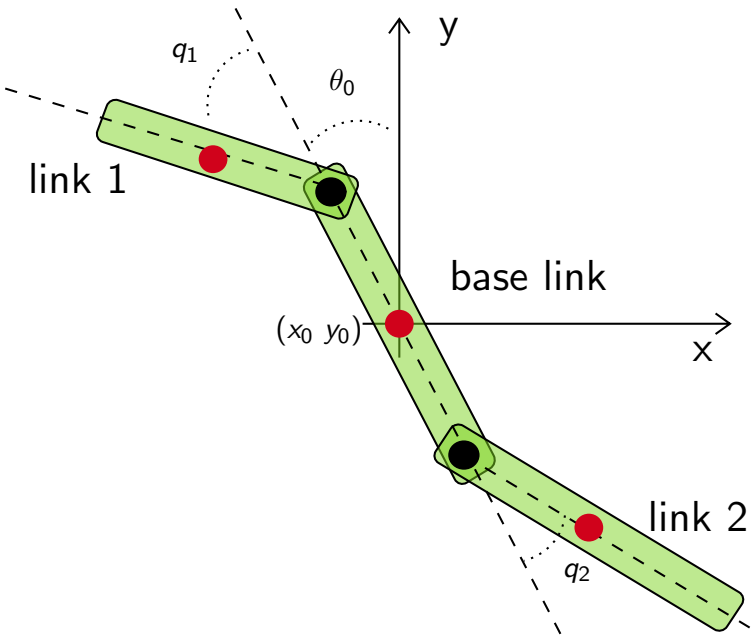
$$\underbrace{\begin{pmatrix} \dot{q}_x \\ \dot{q}_y \\ \dot{\theta} \\ \dot{u}_1 \\ \dot{u}_2 \end{pmatrix}}_{\dot{x}} = \underbrace{\begin{pmatrix} u_1 \cos\theta \\ u_1 \sin\theta \\ u_2 \\ 0 \\ 0 \end{pmatrix}}_{F_d} + \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}}_{F} \underbrace{\begin{pmatrix} v_1 \\ v_2 \end{pmatrix}}_{v} \tag{10}$$

Unicycle with constrained inputs

- Starts with $x = (q_x, q_y, \theta)^\top$, becomes the same as dynamic unicycle (10) after state augmentation;
- for input constraints on magnitude, use the barrier function $b(x) = \frac{1}{(u_i^{\max})^2 - {u_i}^2}, \quad i = 1 \text{ or } 2.$

Figure: Two different scenarios for Input constraints. Left: $u_1^{\max} = 2$. Right: $u_2^{\max} = \frac{\pi}{2}$.

3-link floating robot



$$\begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} = \begin{bmatrix} 0_{3\times 3} & I_{3\times 3} \\ 0_{3\times 3} & -D^{-1}(q)C(q,\dot{q}) \end{bmatrix} \begin{bmatrix} q \\ \dot{q} \end{bmatrix} + \begin{bmatrix} 0_{3\times 3} \\ D^{-1}(q) \end{bmatrix} \begin{bmatrix} 0 \\ u_1 \\ u_2 \end{bmatrix}$$

Conclusion

Conclusion

- In this work we have proposed an innovative motion planning algorithm for dynamical system with drift and affine in controls.
- We have formulated an AGHF equation, obeying which the initial curve is deformed to a curve with locally minimal "length". Controls are extracted from this minimizer and the integrated path is derived by feeding the system with the extracted control, which gives us a solution to the motion planning problem.
- Inspired by the method of using barrier function to deal with state constraints, we developed a similar method for dealing with input constraints via state augmentation.
- Our algorithm is practiced on the models of unicycle with constant linear velocity, dynamic unicycles and unicycle with input constraints. The simulation results of all those examples verify the feasibility of our algorithm and show great potential of its future development.

The End


Brockett, R. W. (1982).

Control Theory and Singular Riemannian Geometry, pages 11–27.

Springer, New York.


Dai, H., Valenzuela, A., and Tedrake, R. (2014).

Whole-body motion planning with centroidal dynamics and full kinematics.

In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 295–302.


Jean, F. (2014).

Control of nonholonomic systems: from sub-Riemannian geometry to motion planning.

Springer.


Jost, J. (1995).

Riemannian geometry and geometric analysis.

Universitext. Springer, Berlin.


Karaman, S. and Frazzoli, E. (2011).

Sampling-based algorithms for optimal motion planning.

The International Journal of Robotics Research, 30(7):846–894.

Kuffner, J., Kagami, S., Nishiwaki, K., Inaba, M., and Inoue, H. (2003).

Online footstep planning for humanoid robots.

In *2003 IEEE International Conference on Robotics and Automation*, volume 1, pages 932–937 vol.1.

Laumond, J. (1998).

Robot motion planning and control.

Lecture notes in control and information sciences. Springer.

LaValle, S. M. (2006).

Planning Algorithms.

Cambridge University Press, Cambridge, U.K.

Available at http://planning.cs.uiuc.edu/.

Majumdar, A. and Tedrake, R. (2013).

Robust online motion planning with regions of finite time invariance.

In *Algorithmic Foundations of Robotics X*, pages 543–558. Springer.

Murphey, T. D. (2006).

Motion planning for kinematically overconstrained vehicles using feedback primitives.

In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1643–1648. IEEE.

Tedrake, R., Manchester, I. R., Tobenkin, M., and Roberts, J. W. (2010).

Lqr-trees: Feedback motion planning via sums-of-squares verification.

The International Journal of Robotics Research, 29(8):1038–1052.

Woodruff, J. Z. and Lynch, K. M. (2017).

Planning and control for dynamic, nonprehensile, and hybrid manipulation tasks.

In *IEEE International Conference on Robotics and Automation*, Singapore.