Improved Action Selection and Path Synthesis using Gradient Sampling

Neil Traft and Ian M. Mitchell

Department of Computer Science
The University of British Columbia

December 2016

ntraft@cs.ubc.ca | mitchell@cs.ubc.ca
http://www.cs.ubc.ca/~mitchell

Shortest Path via the Value Function

- Assume isotropic holonomic vehicle

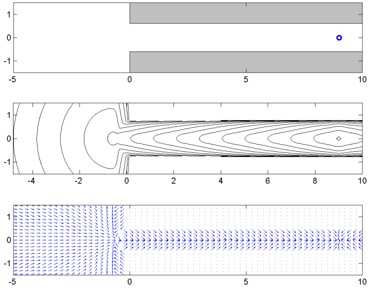
$$\frac{d}{dt}x(t) = \dot{x}(t) = u(t), \quad \|u(t)\| \leq 1.$$

- Plan paths to target set $\mathcal{T}$ optimal by cost metric

$$\psi(x_0) = \inf_{x(\cdot)} \int_{t_0}^{t_f} c(x(s))\, ds,$$
$$t_f = \operatorname{argmin}\{s \mid x(s) \in \mathcal{T}\}.$$

- Value function $\psi(x)$ satisfies Eikonal equation

$$\begin{aligned} \|\nabla \psi(x)\| &= c(x), && \text{for } x \in \Omega \setminus \mathcal{T}; \\ \psi(x) &= 0, && \text{for } x \in \mathcal{T}. \end{aligned}$$



Top: Goal location (blue circle) and obstacles (grey).
Middle: Contours of value function.
Bottom: Gradients of value function (subsampled grid).

Path Extraction from Value Function

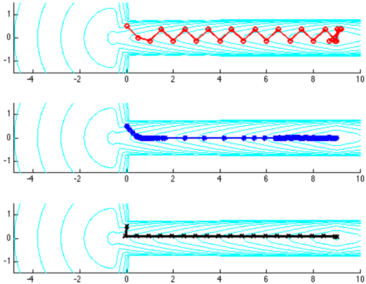
- Given the value function, optimal state feedback action

$$u^*(x) = \frac{\nabla \psi(x)}{\|\nabla \psi(x)\|}.$$

- Typical robot makes decisions on a periodic cycle with period δt so path is given by

$$\begin{aligned} t_{i+1} &= t_i + \Delta t, \\ x(t_{i+1}) &= x(t_i) + \Delta t \; u^*(x(t_i)). \end{aligned}$$

- Even variable step integrators for $\dot{x}(t) = u^*(x(t))$ struggle



Top: Fixed stepsize explicit (forward Euler).
Middle: Adaptive stepsize implicit (`ode15s`).
Bottom: Sampled gradient algorithm.

Outline

1. Motivation: Value Functions and Action / Path Synthesis
2. Background: Gradient Sampling and Particle Filters
3. Gradient Sampling Particle Filter
4. Dealing with Stationary Points
5. Concluding Remarks

Outline

1. Motivation: Value Functions and Action / Path Synthesis
2. Background: Gradient Sampling and Particle Filters
3. Gradient Sampling Particle Filter
4. Dealing with Stationary Points
5. Concluding Remarks

Gradient Sampling for Nonsmooth Optimization I

Gradient sampling algorithm [Burke, Lewis & Overton, SIOPT 2005]

- Evaluate gradient at k random samples within ϵ-ball of current point $x(t_i)$

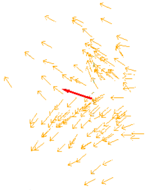
$$x^{(k)}(t_i) = x(t_i) + \epsilon\, \delta x^{(k)},$$
$$p^{(k)}(t_i) = \nabla \psi(x^{(k)}(t_i)).$$

- Determine consensus direction

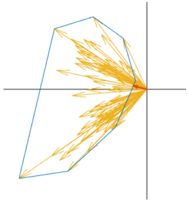
$$p^*(t_i) = \operatorname*{argmin}_{p \in \mathcal{P}(t_i)} \|p\|$$
$$\mathcal{P}(t_i) = \operatorname{conv}\{p^{(1)}(t_i), \ldots, p^{(K)}(t_i)\}.$$

$\mathcal{P}(t_i)$ approximates the Clarke subdifferential at $x(t_i)$.

Gradient samples (yellow) and consensus direction (red).



Plotted in state space.



Plotted in gradient space. Convex hull (blue) also shown.

Gradient Sampling for Nonsmooth Optimization II

Gradient sampling algorithm [Burke, Lewis & Overton, SIOPT 2005]

If $\|p^*(t_i)\| \neq 0$

- Choose step length s by Armijo line search along $p^*(t_i)$.
- Set new point

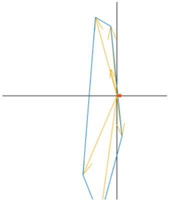
$$x(t_{i+1}) = x(t_i) - s\frac{p^*(t_i)}{\|p^*(t_i)\|}.$$

If $\|p^*(t_i)\| = 0$

- There is a Clarke ϵ-stationary point inside the sampling ball.
- Shrink ϵ and resample.

Gradient samples (yellow) and consensus direction (red).

Plotted in state space.



Plotted in gradient space. Convex hull (blue) also shown.

Particle Filters

Monte Carlo localization (MCL) [Thrun, Burgard & Fox, *Probabilistic Robotics*, 2005] is often used to estimate current state for mobile robots.

- State estimate is a collection of N weighted samples

$$\left\{ (w^{(k)}(t), x^{(k)}(t)) \text{ for } k = 1 \ldots N \right\}.$$

- Predict: Draw new sample state $x^{(k)}(t_{i+1})$ when action $u(t_i)$ is taken

$$x^{(k)}(t_{i+1}) \sim p(x(t_{i+1}) \mid x^{(k)}(t_i), u(t_i)).$$

- Correct: Update weights $w^{(k)}(t_{i+1})$ when sensor reading arrives

$$w^{(k)}(t_{i+1}) = p(\text{sensor reading} \mid x^{(k)}(t_{i+1}))\, w^{(k)}(t_i),$$

- Resample states and reset weights regularly.

We always work with particle cloud after resampling (when all weights are unity).

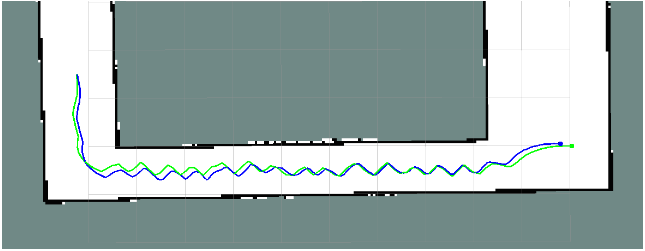
Outline

1. Motivation: Value Functions and Action / Path Synthesis
2. Background: Gradient Sampling and Particle Filters
3. Gradient Sampling Particle Filter
4. Dealing with Stationary Points
5. Concluding Remarks

Narrow Corridor Simulation

Choosing action by AMCL expected state (roughly the mean of particle locations).

- Chattering despite very accurate localization.
- Chattering remains even as step size reduced.

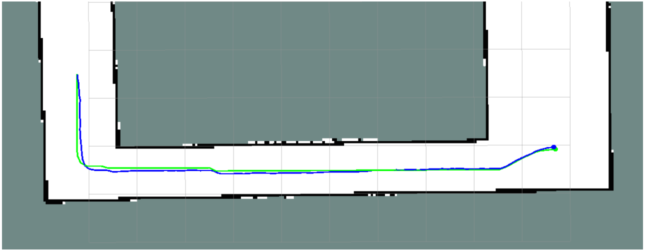


Simulated traversal of a narrow corridor in ROS/Gazebo.
Estimated (blue) and true (green) paths shown.

The Gradient Sampling Particle Filter (GSPF)

Choosing action by GSPF.

- Sample the gradients at the particle locations.
- If $\|p^*(t_i)\| \neq 0$, then $p^*(t_i)$ is a consensus descent direction for current state estimate.



Simulated traversal of a narrow corridor in ROS/Gazebo.
Estimated (blue) and true (green) paths shown.

Outline

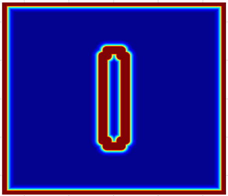
1. Motivation: Value Functions and Action / Path Synthesis
2. Background: Gradient Sampling and Particle Filters
3. Gradient Sampling Particle Filter
4. Dealing with Stationary Points
5. Concluding Remarks

Finite Wall Scenario

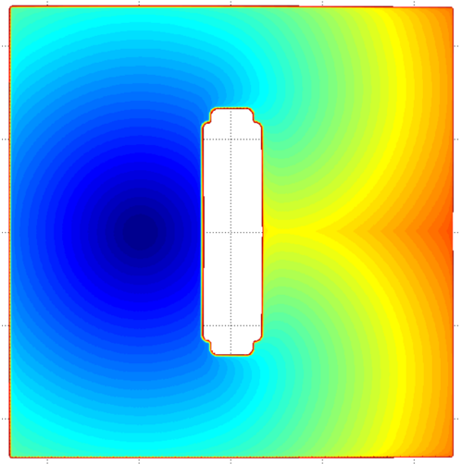
If $\|p^*(t_i)\| = 0$ there is no consensus direction.

Finite wall scenario displays the two typical types of stationary points:

- Minimum (left side): Path is complete(?)
- Saddle point (right side): Seek a descent direction.



Cost.



Value approximation.

Classify the Stationary Point

Quadratic ansatz for value function in neighborhood of samples

$$\bar{\psi}(x) = \tfrac{1}{2}(x - x_c)^T \mathrm{A}(x - x_c) + b^T(x - x_c) + c$$

- Fit to the existing gradient samples

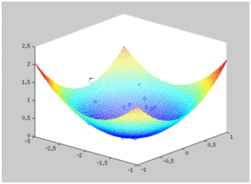
$$\nabla\bar{\psi}(x) = \mathrm{A}(x - x_c) + b.$$

- Solve by least squares

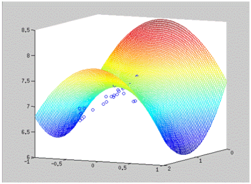
$$\min_{\mathrm{A},b} \|p^{(k)}(t_i) - \mathrm{A}x^{(k)}(t_i) - b\|$$

  and set $x_c = \mathrm{A}^{-1}b$.

- Examine eigenvalues $\{\lambda_j\}_{j=1}^d$ of A
  - If all $\lambda_j > 0$, local minimum.
  - If any $\lambda_j < 0$, corresponding eigenvectors are descent directions.

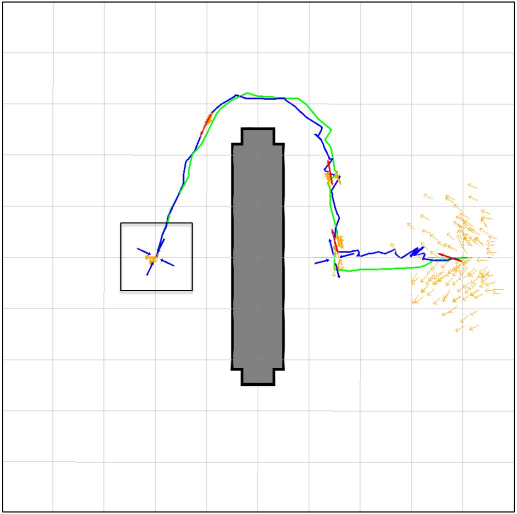


$\bar{\psi}(x)$ at minimum.

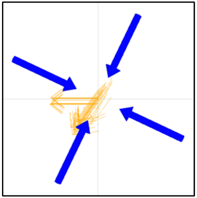


$\bar{\psi}(x)$ at saddle.

Classification Experiments: Minimum

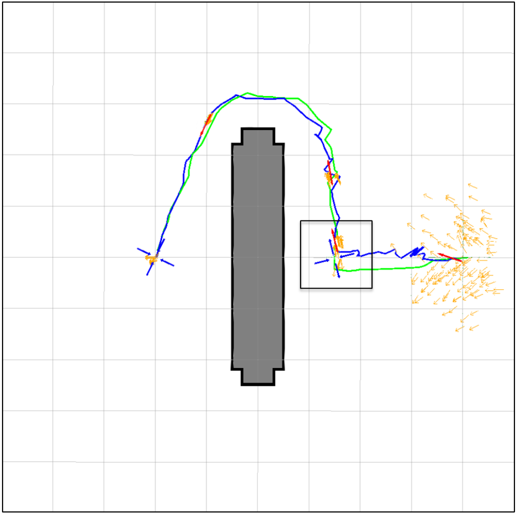


State space view of path.

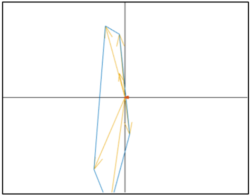


State space gradient samples (gold) and eigenvectors of Hessian of $\bar{\psi}(x)$ (blue). Inward pointing eigenvector arrow pairs correspond to positive eigenvalues.

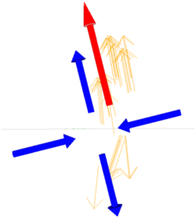
Classification Experiments: Saddle



State space view of path.



Gradient space convex hull

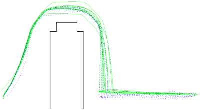


State space eigenvectors of Hessian of $\bar{\psi}(x)$.

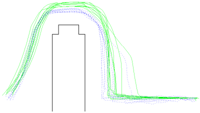
Resolve the Stationary Point

Three potential responses to detection of a stationary point

- Stop: If it is a minimum and localization is sufficiently accurate.
- Reduce sampling radius: Collect additional sensor data to improve localization estimate.
  - Rate and/or quality of sensing can be reduced when consensus direction is available.
  - Localization should be improved by independent sensor data.
- Vote: If it is a saddle and improved localization is infeasible.
  - Let v be the eigenvector associated to a negative eigenvalue and $\alpha = \sum_k \operatorname{sign}(-v^T p^{(k)})$.
  - Travel in direction $\operatorname{sign}(\alpha) v$.



Resolution by using an improved sensor.



Resolution by voting.

Outline

Path Planning Under Uncertainty

- Differential games and the Hamilton-Jacobi-Isaacs equation
  - [Evans & Souganidis, *Indiana University Mathematics Journal*, 1984]
- Robust MPC
  - [Mayne, *Automatica*, 2014]
- Asymptotically optimal sampling-based planners in belief space
  - [Bry & Roy, ICRA 2011]
  - [Luders & How, ACC 2014]
- Efficient POMDP solvers
  - [Pineau, Gordon, & Thrun, IJCAI 2003]
  - [Bai, Hsu, & Lee, IJRR 2014]
- QMDP
  - [Littman, Cassandra, & Kaelbling, ICML 1995]

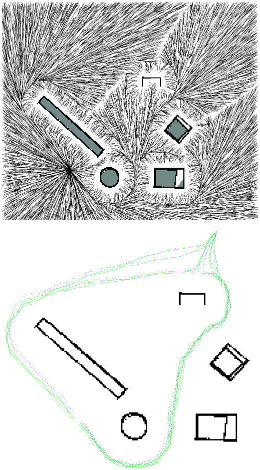
Conclusions

Gradient sampling particle filter (GSPF)

- Utilizes natural uncertainty in system state to reduce chattering due to non-smooth value function and/or numerical approximation.
- Easily implemented on existing planners and state estimation.

Future work

- Nonholonomic dynamics.
- Convergence proof.
- Scalability to more particles.



Actions synthesized by nearest neighbor lookup on RRT* tree. GSPF is not only for value functions.