Machine Learning for Motion Planners

Daniel Beylkin

Summer Undergraduate Research Fellowship

August 3, 2009

CALTECH

The Problem

- Compute realizable trajectories respecting the dynamical and geometric constraints of the system
  - Optimality
  - Real-time implementation
- Problem is a variant of the classical boundary value problem, but such formulation may not be well-suited to handle obstacles
  - Often recast as an optimization problem
  - Overwhelming majority of solution techniques are sampling-based due to differential constraints
- Applicable to autonomous motion planning for aerobots or surface sampling robotic vehicles
  - Also, robotic vacuum cleaners, lawn mowers, humanoid robots designed for home assistance, streamlining warehouse technologies,...

Sampling-Based Motion Planners

- Basic principle: probe state space of mechanical system using a sampling scheme
  - Investigate a state space with infinite number of states using finite number of samples
  - Common choice is random sampling (which is probabilistically complete)
  - Rate of convergence is critical!
- Collision detection
  - Ensure that paths connecting states are entirely collision-free
- Local planning method to connect samples
  - Motion primitives: trajectories are designed by concatenating steady state motions and "well-practiced" maneuvers selected from a finite library
  - Challenge: In many cases we do not have physically accurate or reliable models of the dynamics

Machine Learning Guided Sampling

- Rate of convergence to an optimal solution is critical in random sampling based schemes, especially when sampling in a high-dimensional state space

- Idea: use computed trajectories to guide future sampling
  - Classical methods do not systematically incorporate information about previously computed trajectories and their costs
  - Want to infer some coarse model of the cost as a function of the spatial distribution of the trajectories

- Question: how to handle obstacles?

- Alternative interpretation: construct better initial guesses for trajectory optimization

Which Machine Learning Tool?

- Assumption: paths lying in different regions of the environment are not correlated
  - High correlation values constitute the only meaningful information on the data set
  - Emphasize local structures
- We would like to easily extract extrema information without the need to perform optimization
- Need to consider systems with increasingly high dimensionality
  - Function mapping trajectories to a cost rapidly increase in number of variables
- Locally weighted learning is a good starting point....

Locally Weighted Learning

- Distance function:
  - The only information we have about the trajectories are some nodes along it
  - Distance must not depend on the choice of the nodes on the trajectory
  - Option: area between two trajectories
- Weighting kernel:
  - Must be maximal at zero distance and decay smoothly as distance increases
  - Option: $k(d) = \exp\left[-\alpha \cdot d^2\right]$, where α is optimized using leave-one-out cross validation
- Leave-one-out cross validation: remove the i^{th} data point from the training set and predict the output at that location
  - Minimize the mean squared cross validation error

Locally Weighted Learning

- Weight data directly by defining $\mathbf{Z} = \mathbf{W}\mathbf{X}$ and $\mathbf{v} = \mathbf{W}\mathbf{y}$
  - $\mathbf{X}$ is matrix whose i^{th} row is the i^{th} trajectory in the training set $(\mathbf{x}_i)$
  - $\mathbf{y}$ is vector whose i^{th} element is the cost corresponding to $\mathbf{x}_i$
  - We let $w_i = \sqrt{k(d(\mathbf{x}_i, \mathbf{q}))}$ and define the diagonal matrix $\mathbf{W}_{ii} = w_i$

- Seek global model that is linear in the parameters of β expressed by the overdetermined system $\mathbf{Z} = \beta\mathbf{v}$
  - Determine parameters of β that minimize criterion $C = \|\mathbf{v} - \mathbf{Z}\beta\|^2$
  - Solution given by solving the normal equations $\left(\mathbf{Z}^T\mathbf{Z}\right)\beta = \mathbf{Z}^T\mathbf{v}$

First Go...

- Particle in a plane with simple obstacles and no dynamical constraints

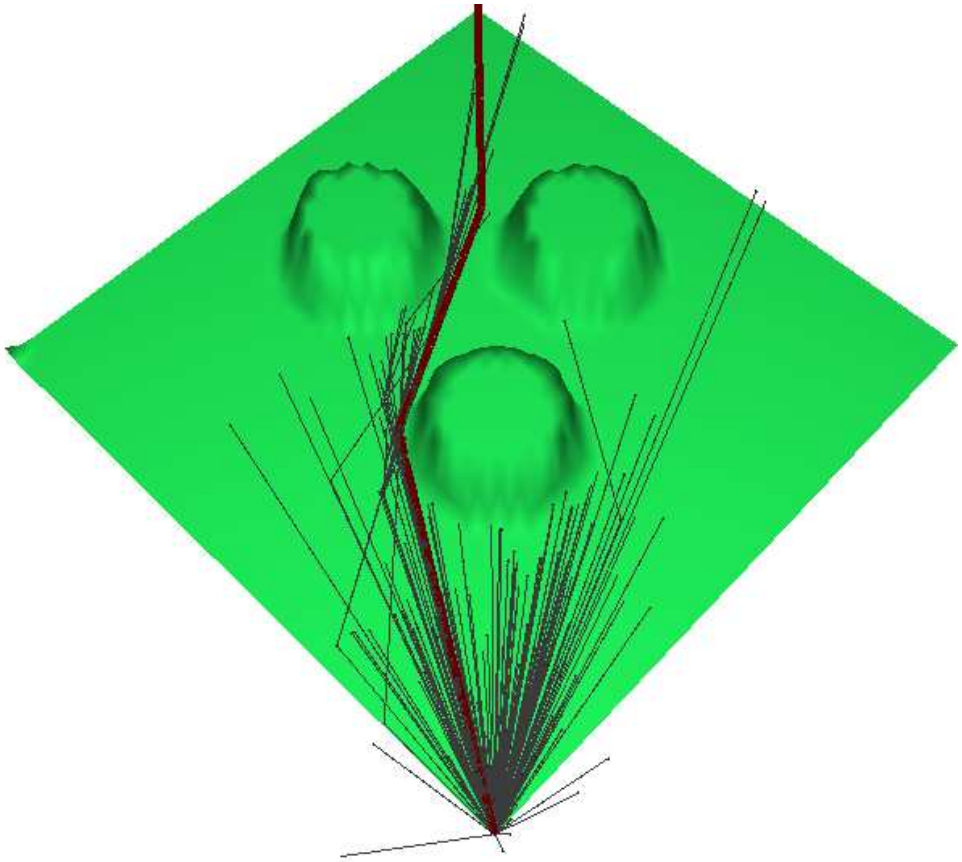

Figure 1: Demonstration of trajectory generation using a random sampling-based scheme

Training and Test Data

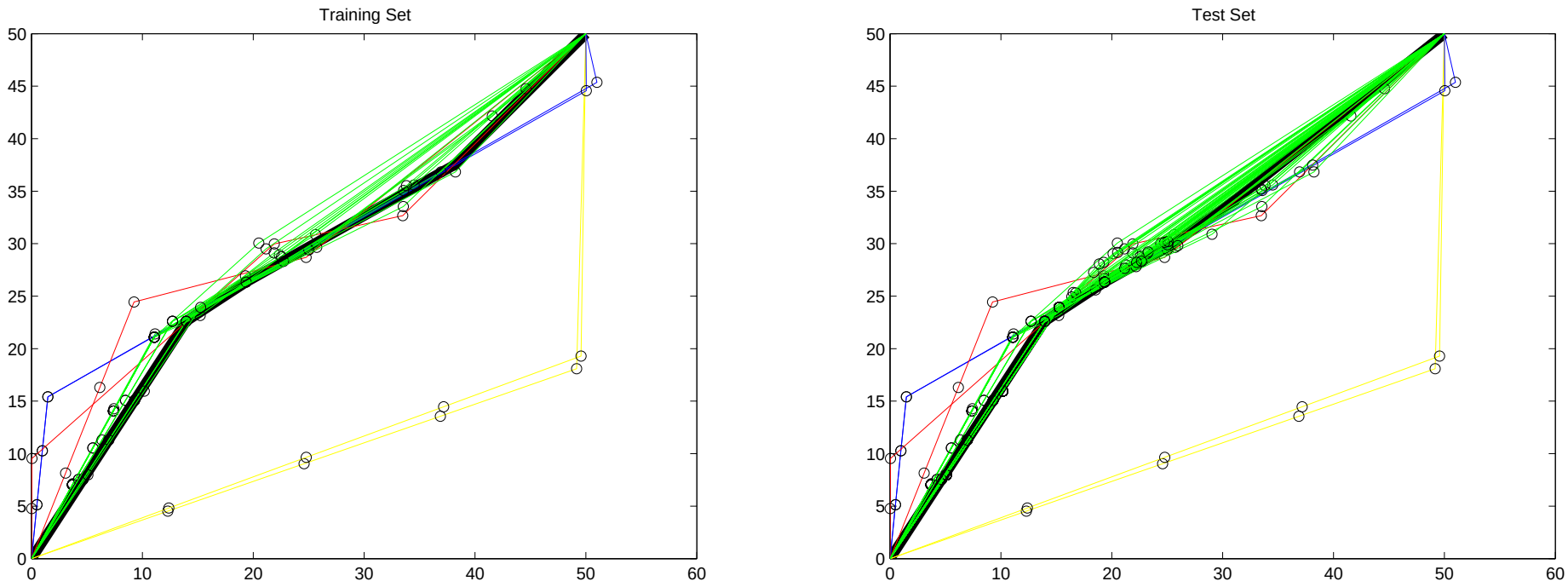

Figure 2: Plot of trajectories in the training set (left), plot of trajectories in the test set (right)

- Trajectories are grouped based on similar costs, with green being the lowest followed by red, blue, magenta, and yellow
  - Bold trajectory indicates optimal one

Prediction

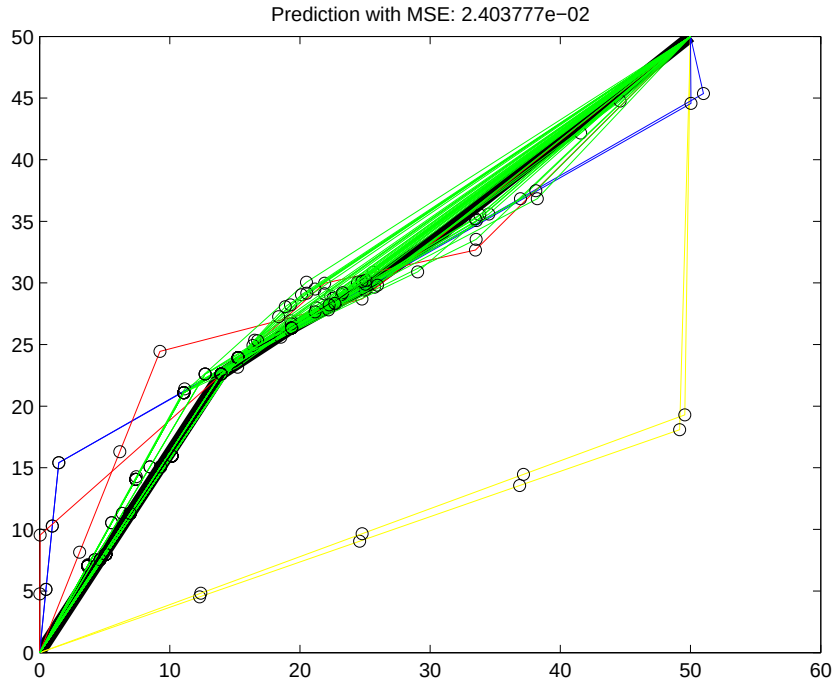

Figure 3: Plot of trajectories grouped in same manner as before using the predicted cost

- Note: mean squared error is roughly 0.024
- We are primarily concerned with general region where extrema are located
  - In this case, it seems to do a good job

Future Goals

- How to extract extrema information explicitely?
- Some preliminary results indicate it is better to handle obstacles in a smooth manner
  - Smoothly increase cost of paths that intersect with obstacles
  - Should not impact extrema information but may help avoid artificial minima inside obstacles
- More complicated systems with dynamics? Systems with high-dimensional state spaces?
  - Locally weighted projection regression? Other options?

Another Project...

- Often, we do not know the physical dynamics of the system, but can we learn the dynamics experimentally?
- We are faced with the problem of learning functions in high dimensional spaces
  - Very hard! Most approaches seem to require some sort of dimensionality reduction
- May bypass curse of dimensionality if we approximate the physical system well using a separable representation

$$f(\mathbf{x}) = \sum_{l=1}^{r} s_l \prod_{i=1}^{d} f_i^l(x_i) + \mathcal{O}(\epsilon),$$

  with small separation rank r and prescribed accuracy ϵ
- We do not fix functions $\left\{f_i^l\right\}_{i=1,\ldots,d}^{l=1,\ldots,r}$ to come from a particular basis set: these functions are determined as part of approximation
  - We switch from linear to non-linear approximation

Illustrative Examples

- Consider $\sin(x_1 + \cdots + x_d)$ which, using usual trigonometric formulas for sums of angles, would have $r = 2^{d-1}$ terms
  - Numerical tests demonstrated that one only needs d terms, which led to a new trigonometric identity

$$\sin\left(\sum_{i=1}^{d} x_i\right) = \sum_{i=1}^{d} \sin(x_i) \prod_{j=1, j\neq i}^{d} \frac{\sin(x_k + \alpha_j - \alpha_i)}{\sin(\alpha_j - \alpha_i)}$$

    for all choices of $\{\alpha_i\}$ such that $\sin(\alpha_j - \alpha_i) \neq 0$ for all $i \neq j$
  - Not unique and there can be ill-conditioned representations even when well-conditioned representations are available
  - Must introduce notion of conditioning for the representation

Illustrative Examples

- Consider additive model $\sum_{i=1}^{d} \phi_i(x_i)$ (where ϕ_i are bounded), which naively has $r = d$:

$$\sum_{i=1}^{d} \phi_i(x_i) = \lim_{h \to 0} \frac{1}{2h} \left(\prod_{i=1}^{d} (1 + h\phi_i(x_i)) - \prod_{i=1}^{d} (1 - h\phi_i(x_i)) \right)$$

  - In the limit, we only need $r = 2$ terms

- Gaussians are separable:

$$\exp\left[-c \left\|\mathbf{x} - \mathbf{z}\right\|^2\right] = \prod_{i=1}^{d} \exp\left[-c(x_i - z_i)^2\right]$$

- It has performed reasonably well on various benchmark data sets (e.g., Friedman) in comparison to other currently available methods

Why?

- Relatively new approach that still may be improved
  - Currently, non-linear approximation is accomplished using alternating least squares
- Quite flexible
- Computation may be done off-line allowing for real-time implementation and on-line to improve the model as new data is gathered
- Current idea: can we use it to learn the dynamics of a double pendulum?
  - Concern: motion is chaotic
- Other ideas: learning solutions of boundary value problems, learning control parameters that produce desired results, etc...

Acknowledgements

- Professor Jerrold Marsden
- Marin Kobilarov
- Professor Martin Mohlenkamp
- Caltech SURF Program
- Carl and Shirley Larson for their generous contribution to SURF