Joint Manifolds and Markov Decision Process in Robot Motion Planning

Authors: Divyanshu Bhartiya (10327250)  
Supervisor: Prof. Amitabha Mukerjee

April 28, 2014
Abstract

Learning trajectories for a robot in unknown environments which can be generalized to different types of robots is a challenging problem. The input space is very high dimensional and any type of algorithm and computation has a very high time complexity and challenging to define problems in such a high state space. Applying low-dimensional reduction to such problems does the trick for us. We can reduce the high dimensional input data to a low dimensional representation and use it to learn and plan paths for a robot.

1. Problem Statement and Idea

In this work, we plan to learn trajectories for a robot, avoiding obstacles and reaching a certain goal keeping the robot constraints like the physical joint constraints and avoidance of occlusions. We expect sample images of various robot’s positions available to us and too from various cameras. We intend to use joint manifold kind of structure to learn the robot’s structure and surroundings. Path planning can be then generalized to unknown environments and unknown obstacles. The idea is that markov Decision processes can also be used to solve robot path planning problem using various poses as states and defining valid transitions depending upon the constraints.

The problem of path planning is intricate due to high dimensional input space. The images are obtained continuously from the cameras, which contribute to large amount of data and that too on a very high dimensional space. The computations on this high dimensional space are very complex and take ample time to do any trivial computations on such a space. To solve such problems, we need a mapping function that will map such high-dimensional data to a low dimensional space. The mapping can be linear or non-linear depending upon the intrinsic manifold structure. The low-dimensional construction of robot’s sensory-motor state space will lead to ease of planning and computations and representations.

2. To do

- Implementation of markov decision process on a robot simulator
- Involving Joint Manifolds to create states of Markov Decision process
- Use of Random Projections in robot path planning

3. Presentations

- Mid-term Presentation: [www.cse.iitk.ac.in/users/dbhartiya/thesis/presentation.pdf](http://www.cse.iitk.ac.in/users/dbhartiya/thesis/presentation.pdf)
- Poster Presentation: [www.cse.iitk.ac.in/users/dbhartiya/thesis/ppt1.pdf](http://www.cse.iitk.ac.in/users/dbhartiya/thesis/ppt1.pdf)
Linear signal models are a result of signals lying on a K-low dimensional subspace of $\mathbb{R}^N$. This can be explained as K components of basis of $\mathbb{R}^N$ having significant coefficients. The vector can be written as a linear combination of these basis. The basis are fixed for a given signal class. Sparse models are a generalization of linear models and every vector can be written as a linear combination of K-basis elements. However these K basis elements varies from signal to signal. A single K-dimensional subspace cant represent all K-sparse signals. Hence the set of all K-sparse signals are a non-linear union of $\binom{N}{K}$ K-dimensional subspaces. Compressed sensing is used as non linear adaptive algorithm for dimensionality reduction. The compressed sensing theory states that every K-sparse signal of $\mathbb{R}^N$ can be recovered from just M linear measurements, where $M = O(K \log(N/K))$. That is $y = \phi(x)$ where $\phi$ is an $M \times N$ matrix. The job in compressed sensing decoding is to recover x. CS recovery is possible because $\phi$ is a homomorphism matrix, that is no two sparse signals in $\mathbb{R}^N$ are mapped to same point in $\mathbb{R}^M$. This mapping is ensured with probability 1 if $\phi$ has independent and identically distributed entries and $M \geq 2K$. However stable embedding is needed for signal recovery, that is the embeddings should be well separated. This is supported by Restricted Isometry Property. JL Lemma provides for stable embedding under a random dimensionality reduction projection. The theorem is:

Let $\mathcal{M}$ be a compact K-dimensional Riemannian submanifold of $\mathbb{R}^N$ having condition number $1/\tau$, volume V and a geodesic covering regularity R. Fix $0 < \epsilon < 1$ and $0 < \rho < 1$. Let $\phi$ be a random orthoprojector from $\mathbb{R}^N$ to $\mathbb{R}^M$ with

$$M = O\left(\frac{K \log(NVR\tau^{-1}\epsilon^{-1}) \log(1/\rho)}{\epsilon^2}\right)$$

If $M \leq N$, then with probability at least $1 - \rho$, the following statement holds: For every pair of points, $x, y \in \mathcal{M}$,

$$(1 - \epsilon)\sqrt{\frac{M}{N}} \leq \frac{||\phi x - \phi y||_2}{||x - y||_2} \leq (1 + \epsilon)\sqrt{\frac{M}{N}}$$


The paper proposes the idea of joint manifolds. The problem framework is of a camera network consisting of J cameras each obtaining N-pixel images of a scene. The total data obtained is $JN$ and hence very large for computations. Hence a manifold is used. A manifold is a k-dimensional space $M$, $M = \{f(\theta) : \theta \in \Theta\}$, where $\Theta$ is the K-dimensional parameter the signal is dependent on. It is assumed that the underlying scene lies on a manifold, parameters differin for each sensor. The aim of the paper is to obtain a simple model that captures the correlation between data points for different manifolds for each camera, that is a concatenation of points indexed by the same parameter $\theta$ from different component manifolds that is $f(\theta) = [f_1(\theta), f_2(\theta), \ldots, f_J(\theta)]$. This is termed as a joint manifold and the joint manifold is also K-dimensional. It is assumed that manifold alignment is present or the manifolds are being aligned using some technique. Since the mapping of joint manifold is non-linear, it depends on the observations from all sensors instead being dependent on one. So their focus is on linear dimensionality reduction methods to obtain $x, y = \phi(x), \phi \in \mathbb{R}^{M \times JN}$, and hence a linear combination of observations. This is compressive data fusion protocol. Instead random projections can also be used to get the manifold. Let $M^*$ be a compact, smooth Riemannian joint manifold in a $JN$-dimensional space with condition number $\frac{1}{\tau}$. Let $\phi$ be an orthogonal linear mapping from $M^*$ to a random M-dimensional subspace of $\mathbb{R}^{JN}$. Let $M = O(K \log(JN/\tau^*)/\epsilon^2)$. Then with high probability, the geodesic distance and euclidean distances between any pair of points are preserved up to distortion $\epsilon$ under $\phi$. Joint manifolds can also be used for classification. Given two manifolds, we wish to determine which manifold generated the data to be classified. We find the distance of the data point from each
manifold and manifold with the smaller distance is the one
classified by the algorithm.

[4] Marco F Duarte, Mark A Davenport, Michael B Wakin, Jason N Laska,
Dharmapal Takhar, Kevin F Kelly, and Richard G Baraniuk. Multiscale
random projections for compressive classification. In *Image Processing,

A set of images in a fixed scene under various domains lie
on a manifold of low dimension. Compressive sensing is re-
constructing a signal from various random measurements.
A K-sparse signal can be recovered by a linear projection
on a random manifold of dimension \(O(K \log(N/K))\), that is
\[ y = \phi(x), \]
where \(y\) is an \(Mx1\) vector, \(\phi\) is an \(MxN\) matrix
and \(x\) is an \(Nx1\) vector. Training is done by sampling points
from different manifolds. The Generalized Likelihood Ratio
Test classifier is based on finding the maximum probability of
a point given its best K dimensional parameters for a given
hypothesis. The best K dimensional parameters can be found
by maximum likelihood principle under that hypothesis.


Let \(M\) be a Riemannian submanifold of \(\mathbb{R}^N\). The condition
number is defined as \(1/\tau\), where \(\tau\) is the largest numsatisfying
the following : the open normal bundle about \(M\) of radius
\(r\) is embedded in \(\mathbb{R}^N\) for all \(r < \tau\). The condition number
defines the smoothness and curvature of the manifold, less
the condition number, less twisted the manifold. The em-
bedding of a high dimensional signal in the low dimensional
space should be stable, that is the embeddings should be well
spaced. The embedding should satisfy the Restricted Isome-
try Property(RIP), which states that for an embedding op-
erator \(P : \mathbb{R}^N \rightarrow \mathbb{R}^M\) to satisfy RIP property, there exist a
positive constant \(\delta\), such that for every \(x, x' \in X, X \subseteq \mathbb{R}^N\),
the following relation hold:

\[
(1 - \delta)||x - x'||_2^2 \leq ||Px - P x'||_2^2 \leq (1 + \epsilon \delta) ||x - x'||_2^2
\]

. To support the existence of en embedding operator which
satisfies this, JL Lemma provides for it.It states: Consider
a dataset \(X = x_1, x_2, \ldots, x_P \subseteq \mathbb{R}^N\).Let \(M \geq O(\delta^{-2} \log P)\).
Construct a matrix \(\phi \in \mathbb{R}^{MxN}\) such that each element of \(\phi\) is
drawn independently from a Gaussian distribution with zero
Then with high probability, the linear operator \( \phi : \mathbb{R}^N \to \mathbb{R}^M \) satisfies RIP on \( \mathcal{X} \). The paper presents a nuclear rank minimization problem for finding a linear embedding on the space of positive semi-definite matrices which satisfies RIP on the secant set of the datapoints. To solve this problem they present a semi-definite algorithm NuMax. They present an algorithm for signal recovery, where try to estimate the independent components of a signal which has noise in it. They use the projection operators for each manifold the independent signal belong to estimate the closest point on the manifold. They presented an algorithm Successive Projection onto Incoherent Manifolds (SPIN) for the above task. The paper also talks about random projections for manifold learning. Intrinsic dimensionality of data is calculated by the Grassberger-Procaccia (GP) algorithm. It computes the scale-dependent correlation dimension of data which is defined as follows: Suppose \( \mathcal{X} = (x_1, x_2, \ldots, x_n) \) is a finite dataset of underlying dimension \( K \). Define

\[
C_n(r) = \frac{1}{n(n-1)} \sum_{i \neq j} I_{||x_i - x_j|| < r}
\]

where \( I \) is the indicator function. The scale dependent correlation dimension of \( \mathcal{X} \) is defined as

\[
D_{corr}(r_1, r_2) = \frac{\log C_n(r_1) - \log C_n(r_2)}{\log r_1 - \log r_2}
\]

They present an algorithm for manifold learning using random projections. The algorithm iterates by generating random orthoprocessor rows for the matrix, and running the isomap algorithm using the intrinsic dimension estimate by the GP algorithm.


This paper describes usage of various dimensionality reduction algorithms and manifold learning to sensory-data time-series from Robonaut-NASA’s humanoid robot. Tests were carried out using principal component analysis, multidimensional scaling and spatio-temporal isomap. They aimed to achieve a mapping between robot’s sensory observations to its motor controls, defined in SSMS(sensorimotor state space).
Sensory-motor data is present as vectors at distinct instances of time $x_k : 1, 2, 3,...N \rightarrow \mathbb{R}^D$. These data points are intrinsically parameterized by lower-dimensional embedding. PCA is used to find the mapping to lower dimensional space by eigen decomposition of the covariance matrix. Similarly MDS preserves pairwise distances instead of covariance. MDS and PCA are similar in euclidean space distance metric. Isomap uses MDS along with Dijkstra’s algorithm. These techniques are not reliable for time dependent data because the data is assumed to be independent of time, whereas the trajectories and the actions are functions of time. Hence windowed MDS was proposed where input vector is extended to a window of observations. This method did find spatially proximal datapoints but failed for corresponding spatially disal datapairs which are as equivalent phases in the temporal process. Hence spatio-temporal isomap (ST-Isomap) was framed to solve both. ST-Isomap finds the nearest neighbours of a datapoint. It proceeds on to find the common temporal neighbors and the adjacent temporal neighbors. It reduces the weights of these neighbors followed by Dijkstra’s algo and MDS. The trajectories obtained using ST-isomap clearly depicts various phases of the temporal process. However the manifolds did not estimate the mapping very well.


This paper describes about path planning with Visual Servoing(VS) for a robotic arm equipped with a camera. Their path planning maintained 4 constraints: 1) target’s continuously within the camera’s field of view, 2) avoiding visual occlusion of target by workspace’s obstacles, robot’s body, 3) avoiding collision with physical obstacles or self collision and 4) joint limits. They have used path planning for IBVS for a 6 degree of freedom robotic arm. They proposed AT-ACE_VS(Alternate Task Space and Configuration space exploration) for exploring task space finding paths satisfying first and second constraints and uses a planner in robot’s C-space to ensure third and fourth constraints. Its an alternate exploration technique in camera and joint space, resulting in
a search tree. The aim is to come up with series of images for the path. Initially camera poses are calculated using AT-ACE algorithm in the camera space, coming up with robot configurations. These conformers are then utilized to get the sequence of images, followed by IBVS technique. Cubic B-splines are used to form $C^2$ trajectories.


They extended the work of robustness of classical image-based visual servoing techniques. The papproach gives $C^1$ smooth trajectories by taking camera dynamics into account along with the four constraints mentioned above. The camera state space is defined as a 12 dimensional vector containing the coordinates, quaternions and linear and angular velocities. The tree is built by generating random camera state using a randomized kinodynamic planner and checking the nearest state in the tree and checking the validity of path formed by applying force and torque between the two states, satisfying the image constraints as well as joint space constraints. Image feature trajectories are built by changing the robot configuration using pseudoinverse of a nominal model of robot jacobian. A greedy extension towards goal is attempted every time a new node is added to tree. In short, conformers are generated randomly and a path is built in the tree satisfying the constraints and then are checked in the image space and joint as well. The set of conformers formed as the path towards goal are then used to form a set of images, on which later a kinodynamic planner is used to generate paths.


The paper proposes to solve MDPs by finding the low dimensional representation of the high dimensional functions of the action space. These functions could be finding the appropriate action to take on a particular state, reward, transition and value functions. Finding low dimensional representation of these functions can lead to ease of computation. A MDP is a 4-tuple $(S, A, P^a_{ss'}, R^a_{ss'})$, where $S$ is a collection of states $S \subseteq \mathbb{R}^d$, $A$ is the set of actions, $P^a_{ss'}$ is the transition matrix which defines the probability of moving from state $s$ to state $s'$ on an action $a$, $R^a_{ss'}$ is the reward matrix which specifies the reward on moving from state $s$ to state $s'$ on an action $a$. The
The aim of an MDP is to maximize the rewards. A discounted MDP is a MDP where we try to maximize the long term reward, \( V_{\Pi}(s) = E_{\Pi}(r_1 + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots | s = s_t) \), \( V \) being the value function \( S \to \mathbb{R} \). The aim in discounted MDP is to come up with an optimal policy. An Average Reward MDP is a MDP where try to maximize the average reward or expected reward per decision. An MDP is solved by policy iteration, value iteration, linear programming, monte carlo method. These algorithms have their complexity in the size of the subspace size. Hence when the state space is large, or if the state space is continuous, exact representation becomes difficult and approximation and short hand representations are needed. To define the low dimensional representation of MDP we need the concept of laplacian of MDP. The Laplacian Matrix of MDP is defined by \( \mathbb{L} = I - P \). Everything about the MDP can be known by the laplacian matrix and its Drazin inverse. The low dimensional representation of a MDP can be built by Diagonalization and Dilation. Diagonalization is finding the eigen vectors of the Laplacian. We find the projections of \( g : V \to \mathbb{R} \) onto \( \mathcal{L}g \) subspace. The eigenfunction of the laplacian can be used to approximate any value function on the graph. The projection \( \langle g, \mathcal{L}g \rangle \) defines the eigen values, \( g \) being the eigenvector and \( \mathcal{L} \) being the normalized Laplacian. Dilation is an operator on the space of functions such that \( Tf(x) = f(2x) \). The Krylov subspace is

\[
K_k^\Pi = \{ P^\Pi, L^\Pi P^\Pi, (L^\Pi)^2 P^\Pi, \ldots, (L^\Pi)^j R^\Pi \}
\]

Similarly, diffusion wavelets are defined as

\[
\{ f, Tf, T^2f, T^4f, \ldots, T^{2^{j-1}}f \}
\]

The top \( k \) components define the basis of the low dimensional representation. The representation policy iteration of the MDP is now defined as first constructing the representation by generating the Krylov basis or Drazin basis or the Diffusion wavelets. Construct the low-dimensional mdp \( P^\Pi_{\phi} = \phi^T P^\Pi \phi \) and \( R^\Pi_{\phi} = \phi^T R^\Pi \).Policy evaluation phase finds the compressed solution \( (I - \gamma P^\Pi_{\phi})w_{\phi} = R^\Pi_{\phi} \). Project the solution back to original state space. Policy improvement phase consist of finding the greedy policy for the next iteration and repeat the process again.

Manifold alignment is used in places where there is some relationship in seemingly distal data sets. Manifold alignment tries to find a mapping between the model of one dataset to another, that is the mapping is not for just only the known data points but instead everywhere. The task of manifold alignment is to find a mapping between two sets to optimally match the points. The algorithm assumes the knowledge of kernel which computes the similarity between data points in the sets. We compute the weight matrices for both the sets using the kernel and compute the laplacian matrices. We compute the eigen vectors of the laplacians obtained from both sets. We find the optimal alignment of the the low dimensional embedding obtained from the above step using SVD. It gives an affine transform on the model.