Consistent Sparsification for Efficient Decision Making Under Uncertainty in High Dimensional State Spaces

by Khen Elimelech, Vadim Indelman

ANPL Autonomous Navigation and Perception Lab

TECHNION Israel Institute of Technology

Problem Definition

The term ***belief*** is used to represent a distribution over a state vector, at certain time, given the performed actions and obtained observations

$$b[X_k] \doteq \mathbb{P}(X_k | a_{0:k-1}, z_{0:k}) \sim \mathcal{N}^{-1}(\times, \Lambda), \qquad X_k \doteq \{x_1, \dots, x_k\}$$

Both transition and observation model contain some Gaussian noise

$$x_{k+1} = g(x_k, a_k) + w_k \quad w_k \sim \mathcal{N}(\times, \mathrm{W_k})$$
$$z_k = h(X_k) + v_k \qquad v_k \sim \mathcal{N}(\times, V_k)$$

When **updating the belief** according to a new action and a predicted observation

$$b^a[X_{k+1}] \doteq \mathbb{P}(X_{k+1} | a_{0:k-1}, z_{0:k}, a, z^a) \sim \mathcal{N}^{-1}(\times, \Lambda_a^+),$$

the information matrix of this new belief can be updated by

$$\Lambda_a^+ = \Lambda + A^T A$$

Where A is a **collective Jacobian**:
- Each action can be described using a Jacobian of this form
- Encapsulates information regarding the transition and its following observation

Entropy is used to ***measure the uncertainty*** in a belief.
For a Gaussian belief b, and its covariance matrix Σ

$$entropy(b) = 0.5 \cdot ln[(2\pi e)^n |\Sigma|]$$

In order to compare the uncertainty in future belief, according to some candidate actions, we can define a ***revenue function***

$$J(b, a) \doteq \frac{1}{|\Sigma_a^+|} = |\Lambda_a^+| = |\Lambda + A^T A|$$

Overall, the ***decision making problem*** is

$$a^* = \underset{a}{\operatorname{argmax}} J(b, a)$$

Expensive! In the general case, for n-dimensional belief, calculating the determinant of the information (covariance) matrix is $O(n^3)$

Do we have to explicitly calculate all possible future revenues?

The Concept ***Find a sparse and action consistent approximation of*** Λ

Definition: Two beliefs are ***action consistent***, if the following applies:

$$J(b, a_i) < J(b, a_j) \Leftrightarrow J(b_s, a_i) < J(b_s, a_j)$$
$$J(b, a_i) = J(b, a_j) \Leftrightarrow J(b_s, a_i) = J(b_s, a_j)$$

- *The order of actions, in terms of posterior revenue, is kept*
- *No meaning for the actual values*
- *Action selection is the same*

Performance Improvement

Keeping **Exact Results**

Sparsification

Per action, variables are involved if they are directly affected by it
Uninvolved variables correspond to columns of zeros in the collective Jacobian
The algorithm considers variables which are uninvolved in all the candidate actions

Inputs:
- A belief $b \sim \mathcal{N}(x, \Lambda^{-1})$
- A list of the variables which are uninvolved for *all* candidate actions in $\mathcal{A}$

Output:
- A sparse belief b_s such that $\gamma(b, b_s) = 0$

Use Cholesky decomposition to find R such that $\Lambda = R^T R$
Calculate $M = R^{-1}$
Generate a sparse M_s according to:

$$(M_s)_{ij} = \begin{cases} M_{ii} & i = j \\ M_{ij} & i \neq j \text{ and the } i\text{-th variable is involved} \\ 0 & i \neq j \text{ and the } i\text{-th variable is never involved} \end{cases}$$

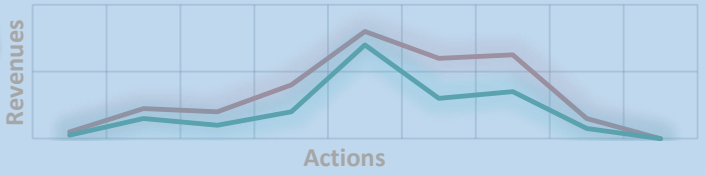
Calculate $R_s = M_s^{-1}$
Calculate $\Lambda_s = R_s^T R_s$
return $b_s \sim \mathcal{N}(x, \Lambda_s^{-1})$

Algorithm 1: Sparsification of the belief

Summary

Initial belief b
↓
Identify uninvolved variables
↓
Find a **sparse** and **action consistent** approximation b_s using our suggested algorithm
↓
Easily calculate all action revenues using b_s
↓
Select best candidate action

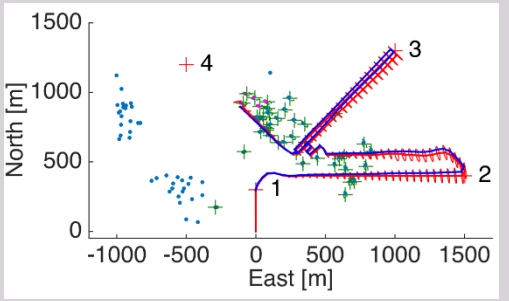
Example: Possible revenues of two action consistent beliefs



SLAM Demo

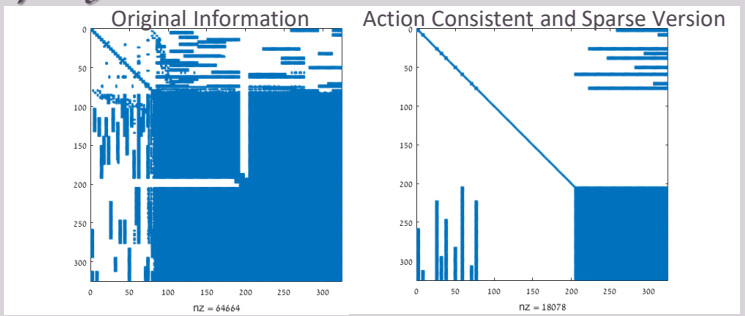
Navigating through several predefined world points, while the state vector maintains the entire trajectory and positions of observed landmarks

The actions refer to taking short paths around the robot

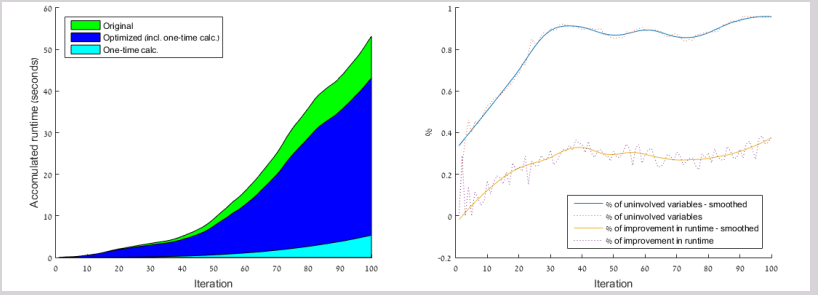


Objective - Keeping the uncertainty low throughout the trajectory by preferring more informative actions

Comparing the Results



In each iteration
- Measured total revenue calculation time, for both methods
- (A one time) Sparsification time added to the total time of the optimized method
- Comparing the time it takes to make each decision in both methods



Accumulation of the measured decision making time

% Uninvolved variables vs % Improvement in runtime