# Monte Carlo Sampling Methods in Motor Control for constraint systems

Oliver Prevenhueber prevenhueber@student.tugraz.at
Institute for Theoretical Computer Science, Inffeldgasse 16b/1 A-8010 Graz, Austria

## Abstract

Stochastic optimal control methods are often applied to motor control problems but have disadvantages in special situations. This paper evaluates the efficiency of sample based methods instead of stochastic optimal control methods solving motor planning problems. Standard methods such as Gibbs sampling, rejection sampling and importance sampling are applied on simple tasks. The experiments reveal specific advantages and disadvantages depending on the applied methods. Importance sampling guarantees a smooth state transition trajectory but probably violates defined boundaries. By contrast, rejection sampling generates a more scattered trajectory, however defined boundaries are not exceeded.

## 1. Introduction

Optimal control using Bayesian inference known as stochastic optimal control is widely used in combination with reinforcement learning for motor planning problems (Toussaint, 2009; Todorov & Li, 2005). The motivation to write this paper follows from the fact that classical stochastic optimal control methods are not optimal in some motor planning cases. For example, the classical algorithms are not able to follow obstacle boundaries tightly and have problems reaching targets in narrow passages. They typically assume quadratic cost functions, which are not able to model hard constraints. The paper evaluates how efficient sample based methods are able to solve motor planning tasks. The following section introduces the applied methods and therefore follow the explanations in (Bishop, 2006). In section 3 the experiments and obtained results are presented. The final section contains concluding remarks.

## 2. Motor planning applying Gibbs sampling

The applied approach is based on Gibbs sampling using in one case importance and in the other case rejection sampling. Sampling methods in general address the problem finding the expectation for a given function with respect to a probability distribution. To compute an optimal approximation of the desired probability distribution the expectation value has to be maximized. In general a probability distribution $p(\mathbf{q})$ of arbitrary complexity from which samples should be drawn is assumed. Due to the fact that modeling $p(\mathbf{q})$ is difficult, a simpler probability distribution $q(\mathbf{q})$, so called *proposal distribution* is used as approximation.

### 2.1. Problem definition

In Figure 1 the task is illustrated. The states in feature space are defined by $\mathbf{q}_{0:T}$, whereas the starting position and the target position are known and therefore highlighted gray. The overall costs are known and represented by $z_{0:T}$ which include obstacle, joint limit and control limit costs. Finding the optimal trajectory $\tau^*$, with the trajectory $\tau$ defined as $\tau = \mathbf{q}_{0:T}$, represents the goal.

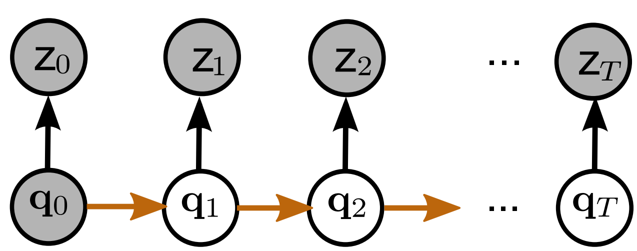


*Figure 1.* Graphical model for the task including obstacles. The states $\mathbf{q}_0$ up to $\mathbf{q}_T$ describe the states in feature space. The states $z_{0:T}$ represent the overall costs including obstacle, joint angle and control limit costs.

**Algorithm 1** Gibbs sampling

**Input:** $\tilde{p}(\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_T)$
**repeat**
  replace $\mathbf{q}_i$ by a sample drawn from $p(\mathbf{q}_i \mid \mathbf{q}_{\setminus i})$
**until** each step is updated
compute expectation value $E[f]$ according to (2)

**Algorithm 2** Rejection sampling

**Input:** $\tilde{p}(\mathbf{q}), q(\mathbf{q}), k$
**repeat**
  Initialize $\mathbf{q}_0$ by drawing randomly from $q(\mathbf{q})$
  Initialize $u_0$ drawing randomly from $[0, kq(\mathbf{q}_0)]$
  **if** $u_0 < \tilde{p}(\mathbf{q}_0)$ **then**
    accept the sample $u_0$
  **end if**
**until** enough accepted samples were drawn

### 2.2. Gibbs sampling

Gibbs sampling represents a Markov chain Monte Carlo algorithm. For this reason the distribution $p(\mathbf{q})$ is written as $p(\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_T)$ whereas $T$ denotes the number of steps. In the $i^{th}$ update step, $\mathbf{q}_i$ is updated by a sample drawn from $p(\mathbf{q}_i \mid \mathbf{q}_{\setminus i})$ using the set $\mathbf{q}_{\setminus i}$ which represents $p(\mathbf{q}_1, ..., \mathbf{q}_{i-1}, \mathbf{q}_{i+1} ..., \mathbf{q}_T)$. This update step is performed by another algorithm, for instance rejection or importance sampling. Gibbs sampling written as pseudo algorithm can be found at 1.

### 2.3. Rejection sampling

Rejection sampling assumes the following association considering a normalizing constant $Z_p$ which is unknown.

$$p(\mathbf{q}) = \frac{1}{Z_p} * \tilde{p}(\mathbf{q}) \tag{1}$$

To draw appropriate samples, $kq(\mathbf{q})$ ($k$ is an optional scaling factor) enclosing the distribution $\tilde{p}(\mathbf{q})$ is a crucial condition. The rejection algorithm is briefly sketched in 2.

### 2.4. Importance sampling

Importance sampling addresses the problem of evaluating the expectation of a function $f(\mathbf{q})$ given the proposal distribution $q(\mathbf{q})$. The proposal distribution is used instead of the desired distribution $p(\mathbf{q})$ due to fact of difficulty in modeling $p(\mathbf{q})$. Therefore weights are introduced to approximate $q(\mathbf{q})$. Hence, the expectation value will be maximized by weight maximization.

$$E[f] \simeq \sum_{l=1}^{L} w_l f(\mathbf{q}^{(l)}) \tag{2}$$

**Algorithm 3** Importance sampling

**Input:** $\tilde{p}(\mathbf{q}), q(\mathbf{q})$
**repeat**
  compute the sample weight according to (3)
**until** the weights for all samples are computed
compute expectation value $E[f]$ according to (2)

with weights defined as

$$w_l = \frac{\tilde{p}(\mathbf{q}^{(l)})/q(\mathbf{q}^{(l)})}{\sum_m \tilde{p}(\mathbf{q}^{(m)})/q(\mathbf{q}^{(m)})} \tag{3}$$

The algorithm is sketched in 3. More detailed informations regarding importance and rejection sampling can be found in (Bishop, 2006).

## 3. Experiments and obtained results

The experiments performed in the project intent to demonstrate advantages of sampling based methods on the one hand and their limitations on the other hand. In the project a simple two-link planar arm without dynamic forces is used. For the experiment the start and target coordinates of the arm end-effector are predefined and all states are initialized with the starting position. The resulting trajectory is computed using Gibbs sampling in combination with rejection sampling in one case and importance sampling in the other case. For this purpose the Gibbs sampling algorithm is evaluated several rounds for a defined number of time-steps. The number of rounds and time-steps must be adapted to the complexity of the task. The precision of the arm is controlled by three different precision matrices. The first precision matrix $W$ is introduced to regularize the distribution for drawing samples. The second precision matrix $C_{trans}$ controls the transition precisions between the time-steps in state space, i.e. the step-size is controlled by this matrix. Finally the third precision matrix $C_{task}$ defines the precision between the last step in state space and the target position in task space. In other words, the third precision matrix controls how accurate the arm has to reach the target position. In the experiments the value of $C_{trans}$ is chosen very small to obtain a smooth trajectory whereas the value of $W$ is a trade-off between the tendency of varying samples (large values) and the inability of moving around obstacles (small values). Furthermore the values of $W$ must be adapted depending on the applied algorithm.

The Gibbs sampling algorithm is applied to a predefined number of rounds were one round equals one complete movement trajectory. To increase the efficiency of Gibbs sampling, updating the samples is started at the last state and operates towards the initial state, which is fixed. Each round includes a predefined amount of time steps in which the algorithm tries to choose the best fitted sample for the actual step. The samples are drawn from a multivariate Gaussian distribution with the angles of the previous round as mean and the variance given by the covariance matrix $W$. In the project the update step is performed by importance or rejection sampling. The number of rounds and steps for optimal results depends on the complexity of the task, the covariance matrices and the chosen algorithm.

For every step of Gibbs sampling the importance sampling algorithm is performed once and therefore a specified number of samples are drawn from the distribution and stored in the corresponding weights. Computing the weights, two cases have to be distinguished whereas the actual time step is defined as t out of T time steps:

$$w_l = \begin{cases} p(\mathbf{q}_T \mid x_T, C_{task}), \text{ if } t = T \\ p(\mathbf{q}_t \mid \mathbf{q}_{t-1}, \mathbf{q}_{t+1}, C_{trans}), \text{ otherwise} \end{cases}$$

In the end the actual sample is updated to the mean value of all new drawn samples rated with their weights. Motor control using importance sampling without obstacles result in a smooth trajectory requiring low computational resources (including the necessary number of rounds, time steps and drawn samples). Introducing obstacles into the task, result in increasing computational costs depending on the degree of interference with the origin trajectory. At a certain level of interference the resulting restrictions due to obstacles can not be further compensated by the algorithm. As shown in Figure 2 and Figure 3 the resulting trajectory is not optimal due to the tendency of samples to vary. In addition, the progress of angles must be piecewise linear for linear tasks to be optimal. To sum up, importance sampling can be used to approximate optimal trajectories but with limitations concerning complexity and it can not be guaranteed that the trajectory will not cross defined boundaries.

Applying the rejection sampling algorithm, the actual sample is updated by the first drawn and not rejected sample. The threshold defines how accurate a sample has to be, for not being rejected. The threshold represents a distribution which is computed by dividing the multivariate Gaussian distribution $\tilde{p}(\mathbf{q})$ by the distribution $kq(\mathbf{q})$. The multivariate Gaussian distribution is defined by the following cases, whereas the

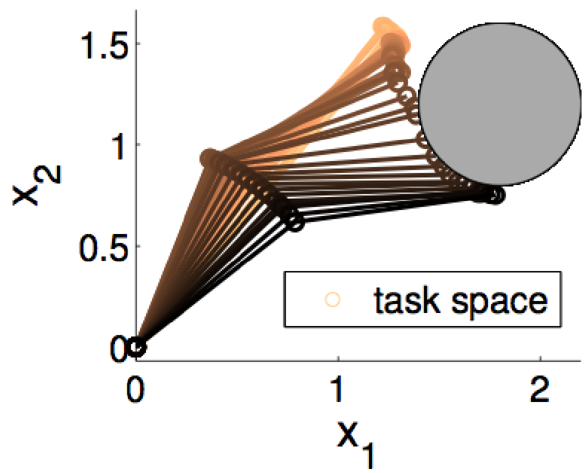


*Figure 2.* Arm movements over all time steps with an obstacle represented by a gray area. The starting position of the arm is indicated by the brightest color and finishes with black color reaching the target position. The illustrated movement was generated using importance sampling.

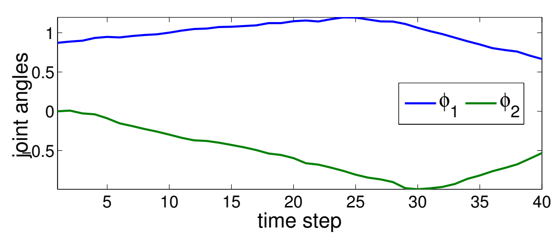


*Figure 3.* Progress of angles depending on the number of time steps using importance sampling. The optimal trajectory would be piecewise straight lines, as we are considering a linear system.

distribution $q(\mathbf{q})$ represents a uniform distribution for simplicity.

$$\tilde{p}(\mathbf{q}) = \begin{cases} p(\mathbf{q}_T \mid x_T, C_{task}), \text{ if } t = T \\ p(\mathbf{q}_t \mid \mathbf{q}_{t-1}, \mathbf{q}_{t+1}, C_{trans}), \text{ otherwise} \end{cases}$$

The experiments emphasize the influence of the distribution $q(\mathbf{q})$ on the quality of the drawn samples. The higher the values of the uniform distribution are chosen the lesser samples are accepted. Whereas small values result in more accepted samples with less quality, i.e. the accepted samples are more scattered. To obtain appropriate samples the uniform distribution must be coordinated with the covariance matrices. For instance, using $W$ with small values and large values of $q(\mathbf{q})$ would result in few accepted samples with high

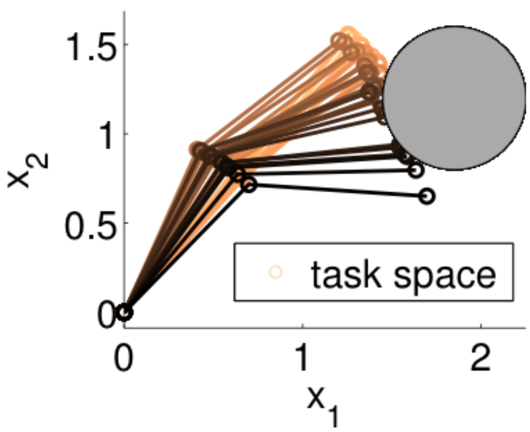


*Figure 4.* Arm movements over all time steps with an obstacle represented by a gray area. The starting position of the arm is indicated by the brightest color and finishes with black color reaching the target position. The illustrated movement was generated using rejection sampling.

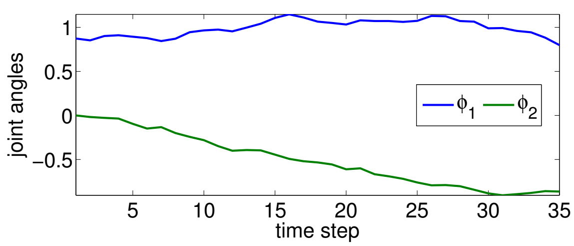


*Figure 5.* Progress of angles depending on the number of time steps using rejection sampling. The optimal trajectory would be piecewise straight lines, as we are considering a linear system. In contrast to importance sampling the trajectory is more jittering.

precision, but the algorithm is not be able to find a trajectory bypassing larger obstacles. In the project hard decisions are applied due to the fact that the first accepted sample will be the update sample and therefore the steps are much more scattered compared to importance sampling. In Figure 4 and Figure 5 the resulting movement using rejection sampling is illustrated. The smoothness of the resulting trajectory can be improved by using the mean of all accepted samples instead of the first one, as update sample. To sum up, finding appropriate parameters is much more difficult applying rejection sampling but due to rejection it can be guaranteed that the trajectory will not cross boundaries.

# 4. Conclusion

The results of the performed experiments have shown that motor planning by sampling methods are an alternative option to stochastic optimal control methods. Although both of the applied algorithms turn out to have some weaknesses. Efficiency can further be improved by dividing complex tasks into simpler subtasks, especially facing large obstacles. To benefit from both algorithms, a combination would be worthwhile. However for solving more complex tasks more advanced methods are required. Investigating more sophisticated methods applied on complex tasks like dynamic motor tasks will be done in further projects.

# References

Bishop, Christopher M. (ed.). *Pattern Recognition and Machine Learning*. Springer Science+Business Media, LLC, 223 Spring Street, NY 10013, USA, 2006.

Todorov, Emanuel and Li, Weiwei. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the American Control Conference 2005*, 2005.

Toussaint, Marc. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, Franklinstr 28/29 FR6-9, 10587 Berlin, 2009.