Unbiased, Scalable Sampling of Closed Kinematic Chains

Yajia Zhang Kris Hauser Jingru Luo

***Abstract*— This paper presents a Monte Carlo technique for sampling configurations of a kinematic chain according to a specified probability density while accounting for loop closure constraints. A key contribution is a method for sampling sub-loops in unbiased fashion using analytical inverse kinematics techniques. Sub-loops are then iterated across the chain to produce samples for the entire chain. The method is demonstrated to scale well to high-dimensional chains (>200DOFs) and is applied to flexible 2D chains, protein molecules, and robots with multiple closed-chains.**

I. Introduction

The movement of kinematic chains is of interest in many areas of science and engineering, including robotics [3], [5], [12], [14] and protein structure prediction [2], [4]. A valuable tool should have the ability to generate (i.e., sample) configurations that satisfies certain hard feasibility constraints, such as kinematic loop closure and collision avoidance, and soft preference constraints, such as low energy and high likelihood. This is computationally challenging: the volume of feasible and favorable configurations is a minuscule subset of the configuration space, whose volume typically shrinks exponentially as dimensionality increases. Moreover, closed-chain constraints, which occur in parallel robots, robot manipulation, legged locomotion, and protein fragments, restrict the feasible set to a lower-dimensional manifold with zero volume.

This paper presents a new Monte Carlo (MC) technique for sampling closed chains according to specified probability densities. It is an advancement upon existing closed-chain sampling methods based on inverse kinematics (IK) [2]–[5], [12], [14], which are unable to sample according to specified probability densities. On the other hand, MC techniques can generate samples according a probability density [1], [10], but have not been applied to closed chains. Our method, Sub-Loop Inverse Kinematics Monte Carlo (SLIKMC), is based on a novel theoretical development that allows us to sample small sub-loops of configuration angles at once in an unbiased manner. These blocks are then iterated across the entire chain to generate an unbiased sequence of configuration samples.

SLIKMC is particularly well-suited for closed chains (Fig. 1) but can also be advantageous for open-ended chains as well (Fig. 2). It is proven to be statistically unbiased and numerical experiments demonstrate that our method generates higher quality samples with lower computational cost than existing approaches. Its use is demonstrated on 2D flexible chains, 3D protein molecules, and multiply-closed articulated robots whose configuration spaces range to >200DOFs.

Yajia Zhang, Kris Hauser and Jingru Luo are with the School of Informatics and Computing, Indiana University Bloomington, USA {zhangyaj,hauserk,luojing}@indiana.edu

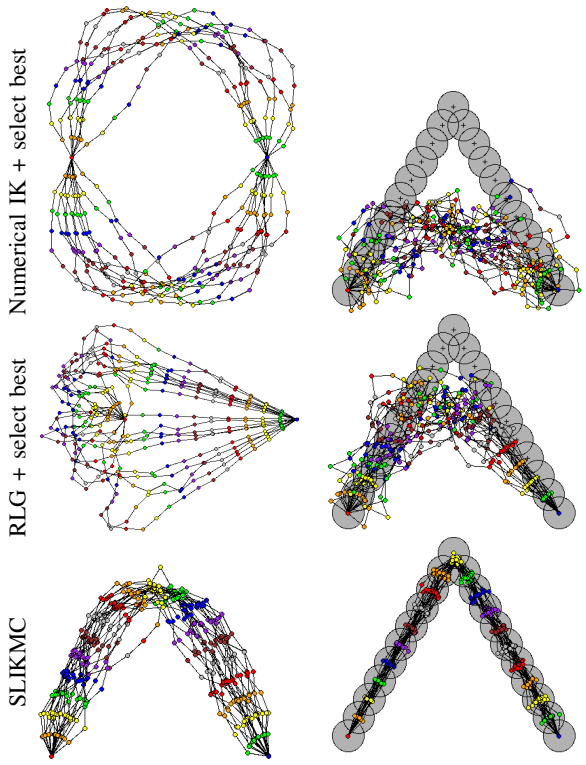

Fig. 1. Three sampling methods for a 20-link closed-loop chain. At left, the prior gives preference to joint angles with small magnitude. At right, the prior gives preference to joint positions in a triangle shaped distribution (crosses: means, shaded circles: 3σ spreads). Top: Sampling joint angles followed by numerical loop closure, best 20/20,000 samples. Middle: Sampling with RLG [3], best 20/20,000 samples. Bottom: Our method, displayed every 40'th sample. These sample sets are generated by our method approximately as fast as RLG and an order of magnitude faster than numerical loop closure.

II. Related Work

The typical approach to sample closed-loop configurations first generates a random sample, and then executes an inverse kinematics (IK) routine to satisfy closed-chain constraints. A variety of techniques can be used for the IK step, including numerical methods such as randomized descent [14], cyclic coordinate descent [2], and Jacobian pseudoinverse methods [11], as well as analytical IK methods [3]–[5]. Techniques have also been proposed so that the initial step yields more configurations that can be successfully closed [3], [12]. Other constraints, such as collision avoidance, are checked after sampling. Though fast, these methods do not

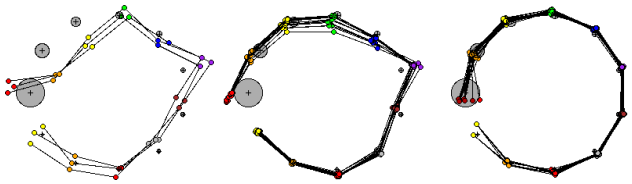

Fig. 2. Comparing MC sampling methods for a free-endpoint chain with heterogeneous prior distribution over joint positions (crosses: means, shaded circles: 3σ spreads). For each method, 400 iterations are run and every 20'th sample is retained, taking $\sim$10 s on a PC. Left: standard Metropolis-Hastings algorithm (MH) takes steps that are too large and only generates 3 unique samples. Middle: MH with step size reduced by 10 has a higher success rate but slower convergence. Note the lack of variance in the leftmost point. Right: Our method.

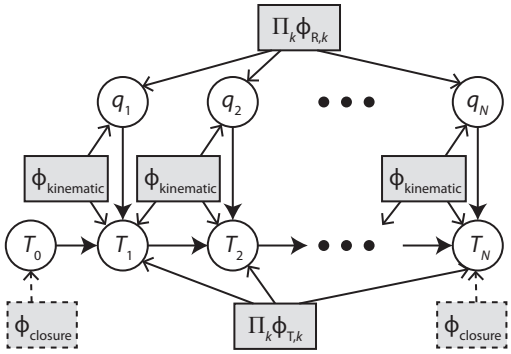

Fig. 3. Sparse graphical model relating N joint angles and link frames.

take specified probability distributions into account during sampling. This is a particular disadvantage in protein motion, because most configurations have high energy. Some authors in computational biology employ a secondary energy optimization step to generate more plausible configurations [7], [9], [13].

In prior methods it is difficult to discuss properties of the sampling *distribution*, which is throughly entangled with the sampling *procedure*, and hence empirical testing is used to argue that a method samples well. Instead, our method provides a unified probabilistic framework for both modeling a desired distribution and sampling from it in a mathematically rigorous fashion. By doing so we extend the formalism of Monte Carlo methods [1], [10], to handle closed chains.

III. Problem Statement

A. Sampling Framework

Let the state variables of the system be denoted $\mathbf{x} = (x_1, \ldots, x_n)$. Hard constraints and soft preferences are encoded into a nonnegative scoring function $\Phi(x_1, \ldots, x_n)$. The score must have finite integral, is zero at states that violate hard constraints, and higher values indicate higher desirability. Φ can be considered as an unnormalized density, and our goal is to generate samples with probability proportional to Φ, i.e. to sample from the normalized density P:

$$P(x_1, \ldots, x_n) = \frac{1}{Z}\Phi(x_1, \ldots, x_n) \tag{1}$$

where Z is a normalization term that ensures that P integrates to 1. It is convenient to represent Φ in *factored* form:

$$\Phi(x_1, \ldots, x_n) = \prod_i \phi_i(S_i) \tag{2}$$

where each ϕ_i is known as a *factor* and each S_i is a subset of $\{x_1, \ldots, x_n\}$ known as the *domain* of the factor ϕ_i. For example, in protein structure prediction, factors will include Ramachandran plots (statistical knowledge about relating angles between subsequent atoms on the protein backbone), atom-atom self collisions, energy functions, and data from X-ray crystallography. For robots, factors may include knowledge about typical joint angles, spatial relationships between points on the robot, and center of mass positions.

Graphical models are inherently factored [6]. A factorization is *sparse* if each variable x_i is involved in only a constant number of factors. Sparseness is important to exploit; for example, when a few variables are changed, the change in Φ can be calculated quickly by only evaluating those factors involved (constant vs. $O(n)$ time).

B. Kinematic Chains

Consider a jointed kinematic chain with reference frames $T_0, T_1, \ldots, T_N$, connected with relative rotational angles $q_1, \ldots, q_N$. For simplicity we first describe our method as applicable to a linear chain. Later we will describe how to apply our method to branched structures (e.g., multi-limbed robots and protein side-chains).

Though states can be unambiguously defined with a minimal set of coordinates, e.g., $\mathbf{x} = (T_0, q_1, \ldots, q_N)$ using standard forward kinematics, this approach eliminates sparsity. A factor defined on T_N will depend on all variables, a factor defined over T_{N-1} will depend on all variables except q_N, and so on.

A key step of our method is to consider an expanded state that incorporates all spatial variables along with the configuration variables: $\mathbf{x} = (q_1, \ldots, q_N, T_0, \ldots, T_N)$. The joint probability density is then defined over angles and reference frames of all links along the chain (Fig. 3):

$$\Phi(\mathbf{x}) = \prod_i \phi_i(S_i) \prod_{j=1}^{N} \phi_{kinematic}(T_j | T_{j-1}, q_j) \tag{3}$$

where each S_i is now a subset of $\{q_1, \ldots, q_n, T_0, \ldots, T_n\}$, and where $\phi_{kinematic}$ is the forward kinematic transform that defines the frame T_j in terms of the prior frame T_{j-1} and the relative angle q_j. Because the transform is deterministic, $\phi_{kinematic}$ is an indicator function. Taking the convention that each frame's origin lies on its joint's axis:

$$\phi_{kinematic}(T_j | T_{j-1}, q_j) = \begin{cases} 1 & \text{if } T_j = T_{j-1} T_j^{rel} R(a_j, q_j) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where T_j^{rel} is the relative transformation of frame j relative to frame $j-1$ and $R(a, q)$ is the rotation of angle q about axis a. Fixed-endpoint constraints are encoded with indicator factors $\phi_{closure}(T_0)$ and $\phi_{closure}(T_n)$ that are zero everywhere except at the fixed frames.

With the new encoding (3) every variable is included in only a few factors and the model is sparse. But we must

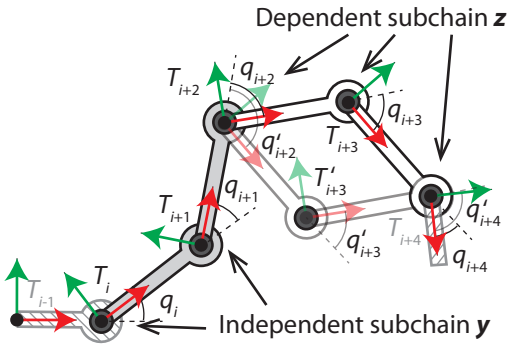

Fig. 4. A 5-angle block for a planar chain with fixed end frames T_{i-1} and T_{i+4}. Two IK solutions exist for the dependent subchain.

now deal with the fact that $\Phi(\mathbf{x}) \neq 0$ only for $\mathbf{x}$ on a lower-dimensional manifold[1] of $\mathbb{R}^n$.

IV. Unbiased Closed Chain Sampling

We describe a Markov Chain Monte Carlo (MCMC) method for generating an unbiased sequence of closed-chain configurations according to an unnormalized probability density given by (1). MCMC methods start from an initial state $\mathbf{x}^{(0)}$ and generate a sequence of states $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$ that forms a random walk whose distribution asymptotically approaches P.

Specifically, the Metropolis-Hastings (MH) algorithm addresses the problem that it is hard to sample directly from P and compute the normalization factor Z explicitly. MH generates the sampling sequence as follows:

1) Samples a candidate move from $\mathbf{x}^{(k)}$ to $\mathbf{x}'$ from a *proposal distribution* $Q(\mathbf{x}'; \mathbf{x}^{(k)})$
2) *Accepts* the move $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}'$ with probability

$$\alpha = \min\left(1, \frac{P(\mathbf{x}')Q(\mathbf{x}^{(k)}; \mathbf{x}')}{P(\mathbf{x}^{(k)})Q(\mathbf{x}'; \mathbf{x}^{(k)})}\right). \quad (5)$$

   The Z term in the numerator and denominator cancel out, so we can use Φ directly instead of P.
3) If the move is rejected, then the current state is maintained: $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)}$.

The remaining question is how to choose a proposal distribution Q that we can sample from and evaluate. Our key contributions is a technique for sampling closed-chain configurations and evaluating Q *exactly*, which lets us evaluate (5) exactly (the so-called *detailed balance* condition) which guarantees convergence to the proper stationary distribution.

A. Closed Chain Sampling Procedure

Let M represent the manifold of loop-closing configurations. A simple random sample has probability zero of generating $\mathbf{x}$ on M; hence, we incorporate inverse kinematics into the sampling procedure. Let us divide the loop variables $\mathbf{x} \equiv (\mathbf{y}, \mathbf{z})$ into those of an n_I-variable *independent* subchain $\mathbf{y}$, and an n_D-variable *dependent* subchain $\mathbf{z}$. This division must be chosen such that for any $\mathbf{y}$, there are a finite number of loop-closing solutions to $\mathbf{z}$. An analytical IK solver must be provided to solve for these solutions. Such a division for a planar chain is depicted in Fig. 4. For 3D chains, a dependent subchain has 6 angles, and all solutions (up to 16) can be determined analytically by finding roots of a polynomial equation [8]. For certain kinematic structures like protein backbones [4] and some robot manipulators, convenient closed-form procedures may be available.

[1]Technically speaking, the density Φ must be considered with respect to a base measure that assigns finite, nonzero density to the manifold.

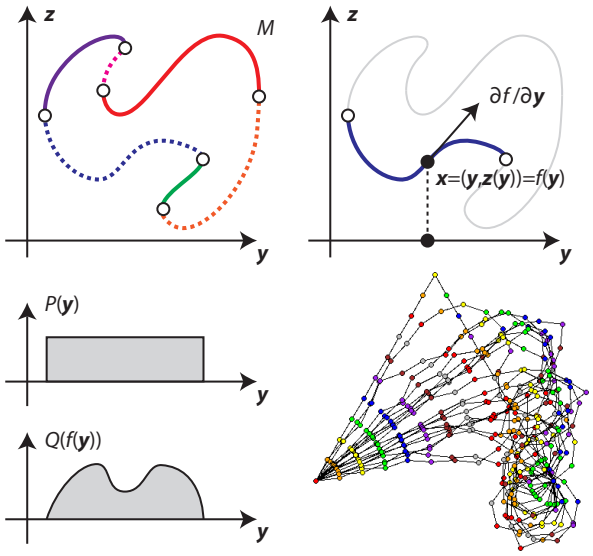

Fig. 5. Analytical IK implicitly decomposes a 1-parameter manifold M into a set of local charts (top left). The jacobian of a chart (top right) must be taken into account when calculating the sampling distribution Q over M (bottom left). Failure to do so leads to a biased sample (bottom right) on the example of Fig. 1, left.

Sample-Loop

1) Sample values $\mathbf{y}$ for the independent subchain at random according to some distribution $Q_y(\mathbf{y})$.
2) Attempt to close the chain by calculating analytical IK solutions for the dependent subchain.
3) If more than one IK solution exists, pick one at random, and if no solution exists, return failure.

B. Closed Chain Sampling Density

We now derive the sampling density $Q(\mathbf{x})$ using concepts from differential geometry. The candidate sample $\mathbf{x}'$ is distributed according to a nonlinear transformation of $Q_y(\mathbf{y})$ onto M. In fact, at non-singular configurations the independent subchain forms a local *chart* of M, which is a local bijection between $\mathbb{R}^{n_I}$ to a neighborhood of M around $\mathbf{x}'$ (Fig. 5).

Since there exists a bijection f between $\mathbf{y}$ and the neighborhood of $\mathbf{x}$ on M, the sampling density is given by:

$$Q(\mathbf{x}) = \frac{Q_y(\mathbf{y})}{s\sqrt{\det G(\mathbf{y})}} \quad (6)$$

where s is the number of IK solutions at $\mathbf{y}$ and G is the *metric tensor* of the chart $\mathbf{x} = f(\mathbf{y})$ (see Appendix). The inclusion of the metric tensor is a natural consequence of transformation of variables. For example, for the case in

which $\mathbf{y} \in \mathbb{R}$, the metric tensor is the squared arc length of the curve $f(\mathbf{y})$ (Fig. 5, bottom). In general, G is given by

$$G(\mathbf{y}) = \left(\frac{\partial f}{\partial \mathbf{y}}(\mathbf{y})\right)^T W \left(\frac{\partial f}{\partial \mathbf{y}}(\mathbf{y})\right) \tag{7}$$

where $\frac{\partial f}{\partial \mathbf{y}}(\mathbf{y})$ is the Jacobian of the function f. Here we have also introduced a positive semidefinite weighting matrix W for the purpose of weighting the relative importance of matching the prior along certain axes. Usually W is the identity, but it can also set a nonuniform diagonal to account for heterogeneous units (e.g., angles vs. positions).

It is often difficult to explicitly compute the Jacobian of the IK function involved in f, i.e., it is difficult to evaluate $\partial \mathbf{z}/\partial \mathbf{y}$. So, we compute it through the implicit form of the constraints $C(\mathbf{x}) = 0$. These vector-valued constraints state that the difference between the terminal frame of the subchain and the desired frame is zero. We have the constraint equation:

$$0 = C(\mathbf{x}) = C(\mathbf{y}, \mathbf{z}) \tag{8}$$

Performing the derivative of both sides of (8) with respect to $\mathbf{y}$ we get:

$$0 = \frac{\partial C}{\partial \mathbf{y}} + \frac{\partial C}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{y}} \tag{9}$$

and hence

$$\frac{\partial \mathbf{z}}{\partial \mathbf{y}} = -\left(\frac{\partial C}{\partial \mathbf{z}}(\mathbf{x})\right)^{-1} \frac{\partial C}{\partial \mathbf{y}}(\mathbf{x}) \tag{10}$$

holds as long as $\frac{\partial C}{\partial \mathbf{z}}$ is invertible, which is true everywhere except at singular configurations. Each derivative of C in the above expression is a submatrix of the Jacobian and is computed using standard techniques.

Finally, since $f(\mathbf{y})^T = \left[\mathbf{y}^T, \mathbf{z}^T\right]$ we obtain the Jacobian $\frac{\partial f}{\partial \mathbf{y}}^T = \left[I, \frac{\partial \mathbf{z}}{\partial \mathbf{y}}^T\right]$ in which I is the identity matrix.

V. SLIKMC Sampler

Based on the principle that small loops have lower computational overhead and higher acceptance probabilities than large loops, the SLIKMC method scales better to large loops by applying the above procedure on small subloops. Sub-loop sampling is repeated to cover the entire chain via the process of *blocked Gibbs sampling*.

A. Blocked Gibbs Sampling

Gibbs sampling is a MCMC method for sampling from high-dimensional probability distributions by repeatedly sampling univariate conditional distributions. Blocked Gibbs sampling is a generalization that samples *blocks* of variables from their conditional distribution. Indices (resp., blocks) are iterated to cover all variables such that the resulting sampling sequence converges to the desired distribution P. The strategy works because it is often easier to sample from a conditional density rather than a large joint density, and is particularly well-suited for sparse factored models.

Given the current sample $\mathbf{x}^{(k)} = (x_1^{(k)}, \ldots, x_n^{(k)})$ at time k, Gibbs sampling generates the next state $\mathbf{x}^{(k+1)}$ by sampling a single variable $x_i^{(k+1)}$ from the conditional density

$$P(x_i | x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x_{i+1}^{(k)}, \ldots, x_n^{(k)}) \tag{11}$$

and keeping the remaining variables fixed. The index i is then incremented from 1 to n in looping fashion to generate a walk across the entire state. Blocked Gibbs sampling is similar, except that a block of variables $B \subseteq \{1, \ldots, n\}$ is chosen at once, and the block move is chosen by sampling $\mathbf{x}_B^{(k+1)}$ from the density $P(\mathbf{x}_B | x_C^{(k)})$ where $x_C^{(k)}$ are the variables indexed by the complement of B.

B. Algorithm

SLIKMC takes as input a set of blocks $B_1, \ldots, B_m$ whose union covers the entire chain. It is assumed that each block yields an appropriate sub-loop for MH sampling.

SLIKMC($\mathbf{x}^{(k)}$):

1) $\mathbf{x} \leftarrow \mathbf{x}^{(k)}$
2) Repeat for $i = 1, \ldots, m$:
3) Sample a move of the sub-loop $\mathbf{x}_{B_i}$ under the distribution Φ_{B_i}. Repeat up to n_B tries or until a successful move is generated.
4) Output the move $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}$.

The key step is Line 3, which uses the method of Sec. IV to sample the sub-loop while keeping variables in the complement $\{1, ..., n\} \setminus B_i$ fixed. The score Φ_{B_i} calculates the product of factors ϕ_j whose domains S_j overlap with B_i, which is more efficient than computing Φ from scratch. The parameter n_B controls how long SLIKMC stays with a particular sub-loop. With n_B large, it is more likely to generate successful moves, but may spend too long in constrained regions of the chain ($n_B = 10$ in our implementation). Assuming sparse probabilistic models and constant sized blocks, each pass of SLIKMC is performed in $O(n)$ time.

C. Block Choice

There is significant flexibility in choosing the blocks. We discuss three issues here.

Block size. Each block must satisfy $n_I \geq 1$, which requires blocks of at least 7 angles for linear 3D chains and 4 angles for 2D chains (n_D is fixed at 6 in 3D and 3 in 2D.). If $n_I = 0$, SLIKMC can only access a finite number of states. It is recommended that $n_I \geq 1$ be chosen as low as possible, because as $n_I + n_D$ grows, the probability of sampling independent variables that admit closure drops off dramatically. Moreover, the cost of each sub-loop sample is $O(n_D^3 + n_I^3)$ because it requires a $n_D \times n_D$ matrix inversion and a $n_I \times n_I$ matrix determinant, and hence it is appropriate to minimize n_I. Our implementations typically use $n_I = 1$, except for proteins which use $n_I = 2$ because even numbers align better with the dihedral angles of the protein backbone.

Block overlap. There is also an issue of whether to overlap blocks, and if so, how much. More overlap requires more work per pass but improve the rate of configuration space exploration. Our implementations overlap all but 1 variable, except for proteins which overlap all but 2.

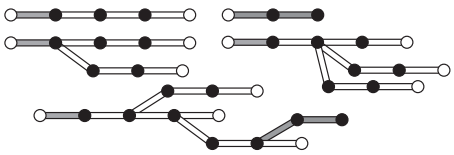

Fig. 6. Several branching structures may be treated as blocks. Independent chains (shaded) must be chosen to parameterize the manifold of configurations satisfying closed chain constraints (open circles).

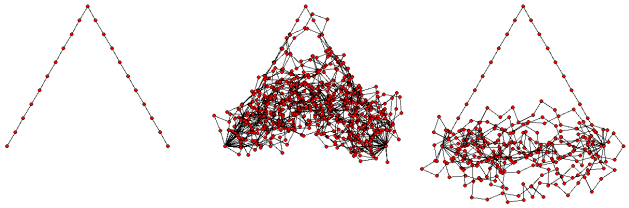

Fig. 7. Sampling configurations of a planar 20-link chain, anchored at the endpoints, with a uniform prior. (Left) Starting from a deliberately bad initial configuration. (Middle) The sequence mixes relatively quickly, but the first 40 samples are biased by the initial configuration and autocorrelate strongly. (Right) A sequence that takes every 40'th sample is effectively independent.

Block topology. Branching structures and free endpoints are treated differently from linear loops. Free-endpoint blocks can be addressed using standard MH. Branches require some care to ensure that independent subchains form a chart of the space of closed-chain configurations (Fig. 6), but are otherwise treated in the same manner using the methods of Sec. IV.

D. Test Order, Mixing, and Autocorrelation

When some factors incur greater computational cost than others, it may be useful to reorder steps in the MH acceptance test. In experiments on a protein, the probability of generating collision-free moves is 78%, but only 0.28% of those moves are accepted. Since collision checking is 60 times more expensive than calculating the remainder of the terms in Φ, we obtain an order of magnitude speedup by performing collision detection *after* determining whether it will be accepted.

In all MCMC methods it is important to empirically examine mixing rate and autocorrelation. Mixing refers to the number of iterations needed to "forget" a poor initialization. For proteins, this is not a significant problem because we initialize the chain with the predicted structure in the Protein Databank (PDB), which is typically quite good. Another concern is autocorrelation, which grows stronger as dimensionality increases (Fig. 7). In practice, one must skip samples in order to obtain *effective independence*, which we define as autocorrelation below a given threshold (0.2 is used in our experiments). Skip lengths are determined empirically.

VI. Experiments

Scalability tests on free-endpoint proteins. Although SLIKMC is designed for closed chains, it can also be applied to open chains as well. We applied it to sample configurations for the 108-residue 1B8C protein (2 joints per residue) according to a distribution including atom-atom collisions,

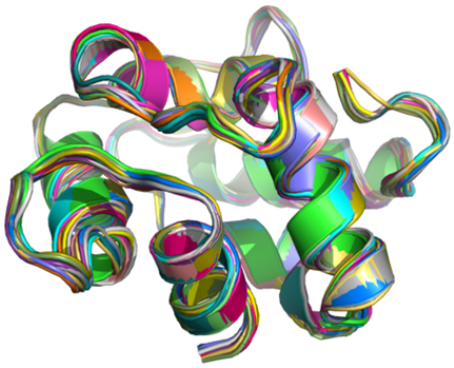

Fig. 8. 17 samples of 1B8C chain A (108 residues) selected from 170 consecutive samples with skip length 10. (Samples drawn in distinct colors)

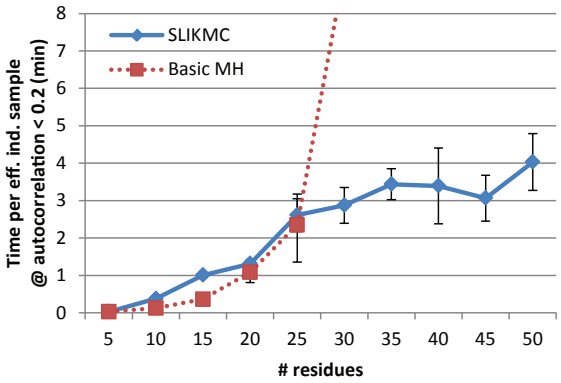

Fig. 9. Time required to obtain one effectively-independent sample on chains with a variety of lengths. Standard MH did not generate even one sample for >30 residues after 30 minutes.

Ramachandran plots, and atom position priors from crystallographic data (Fig. 8). Each sample was generated in approximately 4 s on a Intel i7 2.7 GHz PC with 4 GB RAM

Since standard MH algorithms are applicable to open chains we can compare them to SLIKMC. We sample backbone angles according to a Gaussian proposal distribution with $\sigma = 1°$. Fig. 9 shows the average time needed to obtain one effectively independent sample over ten 30-minute runs for different subchains of 1B8C. Skip lengths are determined empirically. SLIKMC achieves an approximately linear cost per sample, while the cost of standard MH grows exponentially.

Tests on closed chains. We consider a 10-residue protein loop 1AMP181-190. SLIKMC is compared with a sample-then-select method that first samples a set of collision-free configurations and then extracts the top scoring ones. The LoopTK sampler [15] was used here. Both methods generate samples at approximately the same rate. Fig. 10 shows every 100th sample of our method and the top 20 samples of LoopTK after 5 min computation. Clearly, SLIKMC matches the prior information far more closely.

Branched chains. Branching blocks (Fig. 6) are used to capture non-linear chains (e.g., protein side chains and multiple closed chains). We constructed a human-like 2D chain (Fig. 11) with fixed foot positions. Joint angle priors are given, with mean values at a standing configuration, and

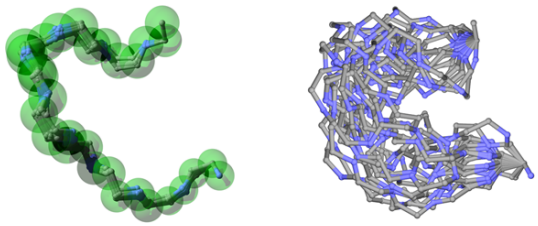

Fig. 10. Left: samples generated by SLIKMC with skip length of 100. Right: samples generated by post-selecting the top 20 scoring samples generated from the LoopTK sampler [15]. Transparent balls depict the 3σ spread of the atom position prior.

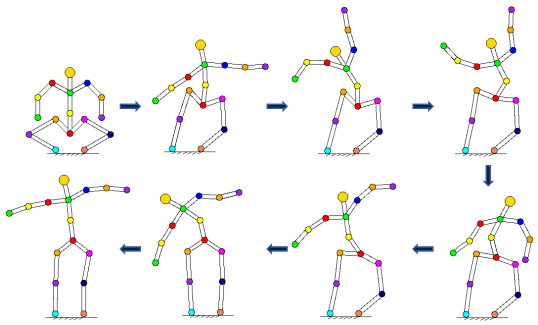

Fig. 11. A sampling sequence for a planar humanoid robot constrained at the feet, with hard joint limit, collision, and stability constraints and a prior whose mode is a standing-up configuration. Every 500th sample is drawn.

angles that exceed joint limits are set to probability zero. We also set the probability of configurations that are self-colliding and unbalanced (i.e., the c.o.m. does not lie over the support polygon) to be zero. The sampling sequence leaves an improbable squatting configuration to reach a more probable standing-up configuration.

VII. Conclusion

We presented a novel Monte Carlo method for sampling closed chains according to a specified probability distribution. We derive the mathematical conditions necessary for this sampling to be unbiased and present an algorithm that is highly scalable. Experiments show that our method can quickly generate closed-chain configurations with over 200 DOFs. In future work, we intend to research the strategy of selecting blocks in order to improve convergence rates and apply our method to more complex examples such as 3D robots with multiple closed loops.

Appendix

We review a fundamental statement about densities under transformations of variables.

Suppose $\mathbf{u} \in \mathbb{R}^m$ *and* $\mathbf{v} \in \mathbb{R}^n$ *are multivariate random variables related by* $\mathbf{v} = f(\mathbf{u})$*, where* f *is differentiable and injective. Denote the image of* $A \subseteq \mathbb{R}^m$ *as* $M = f(A) \subseteq \mathbb{R}^n$*. If* g_u *is a density with support over* A*, then the corresponding density over* M*, with respect to the* m*-volume measure, is*

$$g_v(\mathbf{v}) = g_u(f^{-1}(\mathbf{v}))/\sqrt{\det G(f^{-1}(\mathbf{v}))} \tag{12}$$

where $G(\mathbf{u})$ *is the metric tensor:*

$$G(\mathbf{u}) = \left(\frac{\partial f}{\partial \mathbf{u}}(\mathbf{u})\right)^T \left(\frac{\partial f}{\partial \mathbf{u}}(\mathbf{u})\right). \tag{13}$$

More precisely, g_v *as defined above satisfies:*

$$\int_{f(U)} g_v(\mathbf{v}) d\mu = \int_U g_u(\mathbf{u}) d\mathbf{u} \tag{14}$$

for any $U \subseteq A$*, where* $d\mu$ *is the* m*-volume element of* M*.*

From change of variables we have:

$$\int_{f(U)} g_v(\mathbf{v}) d\mu = \int_U g_v(f(\mathbf{u})) X(\mathbf{u}) d\mathbf{u} \tag{15}$$

where $X(\mathbf{u})$ is the m-volume of the parallelotope spanned by the axes of the coordinate chart f centered at $\mathbf{u}$: $\frac{\partial f}{\partial u_1}(\mathbf{u}), \ldots \frac{\partial f}{\partial u_m}(\mathbf{u})$.

We now use the fact that the m-volume V of the parallelotope spanned by m vectors $\mathbf{v}_1, \ldots, \mathbf{v}_m \in \mathbb{R}^n$ is given by the determinant:

$$V^2 = \det \begin{pmatrix} \mathbf{v}_1^T \mathbf{v}_1 & \cdots & \mathbf{v}_1^T \mathbf{v}_m \\ \vdots & & \vdots \\ \mathbf{v}_m^T \mathbf{v}_1 & \cdots & \mathbf{v}_m^T \mathbf{v}_m \end{pmatrix}. \tag{16}$$

Note that this can be expressed more compactly as $\det(A^T A)$ where A is the matrix with $\mathbf{v}_1, \ldots, \mathbf{v}_m$ as its columns. Hence, $X(\mathbf{u}) = \sqrt{\det G(\mathbf{u})}$. Finally, substituting g_u in the r.h.s. of (15) gives the desired result.

References

[1] D. Bouzida, S. Kumar, and R. H. Swendsen. Efficient monte carlo methods for the computer simulation of biological molecules. *Phys. Rev. A*, 45(12):8894–8901, Jun 1992.

[2] A. Canutescu and R. Dunbrack Jr. Cyclic coordinate descent: A robotics algorithm for protein loop closure. *Protein Science*, 12:963–972, 2003.

[3] J. Cortés, T. Siméon, and J.-P. Laumond. A random loop generator for planning the motions of closed kinematic chains using prm methods. In *IEEE Int. Conf. Rob. Aut.*, Washington, D.C., 2002.

[4] E. Coutsias, C. Soek, M. Jacobson, and K. Dill. A kinematic view of loop closure. *J. Computational Chemistry*, 25:510–528, 2004.

[5] L. Han and N. M. Amato. A kinematics-based probabilistic roadmap method for closed chain systems. In *WAFR*, 2000.

[6] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. The MIT Press, Cambridge, Massachusetts, 2009.

[7] D. Mandell, E. Coutsias, and T. Kortemme. Sub-angstrom accuracy in protein loop reconstruction by robotics-inspired conformational sampling. *Nature Methods*, 6:551–552, 2009.

[8] D. Manocha and J. Canny. Efficient inverse kinematics for general 6r manipulators. *IEEE T. Robotics and Automation*, 10(5), Oct 1994.

[9] A. Shehu, C. Clementi, and L. Kavraki. Modeling protein conformational ensembles: From missing loops to equilibrium fluctuations. *Proteins: Structure, Function, and Bioinformatics*, 65:164–179, 2006.

[10] A. F. M. Smith and G. O. Roberts. Bayesian computation via the gibbs sampler and related markov chain monte carlo methods. *J. Royal Statistical Society. Series B (Methodological)*, 55(1):3–23, 1993.

[11] M. Stilman. Task constrained motion planning in robot joint space. In *IEEE/RSJ Int. Conf. Intel. Rob. Sys.*, pages 3074 – 3081, 2007.

[12] X. Tang, S. Thomas, P. Coleman, and N. M. Amato. Reachable distance space: Efficient sampling-based planning for spatially constrained systems. *Int. J. Rob. Res.*, 29(7):916–934, June 2010.

[13] H. van den Bedem, I. Lotan, J.-C. Latombe, and A. Deacon. Real-space protein-model completion: an inverse-kinematics approach. *Acta Crystallography*, 61(1):2–13, Jan 2005.

[14] J. H. Yakey, S. M. LaValle, and L. E. Kavraki. Randomized path planning for linkages with closed kinematic chains. *IEEE Trans. Robot. and Autom.*, 17(6):951–958, 2001.

[15] P. Yao, A. Dhanik, N. Marz, R. Propper, C. Kou, G. Liu, H. van den Bedem, J.-C. Latombe, I. Halperin-Landsberg, and R. Altman. Efficient algorithms to explore conformation spaces of flexible protein loops. *T. Comp. Biology and Bioinformatics*, 5(4):534 –545, Oct. 2008.