Interleaving Global and Local Search for Protein Motion Computation

Kevin Molloy[1] and Amarda Shehu[1,2,3]

[1]Department of Computer Science, [2]Department of Bioengineering, [3]School of Systems Biology, George Mason University, USA
amarda@gmu.edu, www: http://cs.gmu.edu/~ashehu

Abstract. We propose a novel robotics-inspired algorithm to compute physically-realistic motions connecting thermodynamically-stable and semi-stable structural states in protein molecules. Protein motion computation is a challenging problem due to the high-dimensionality of the search space involved and ruggedness of the potential energy surface underlying the space. To handle the multiple local minima issue, we propose a novel algorithm that is not based on the traditional Molecular Dynamics or Monte Carlo frameworks but instead adapts ideas from robot motion planning. In particular, the algorithm balances computational resources between a global search aimed at obtaining a global view of the network of protein conformations and their connectivity and a detailed local search focused on realizing such connections with physically-realistic models. We present here promising results on a variety of proteins and demonstrate the general utility of the algorithm and its capability to improve the state of the art without employing system-specific insight.

Keywords: protein motion computation, conformational path, roadmap-based algorithm

1 Introduction

Elucidating the detailed motions employed by dynamic protein molecules [1] to switch between different thermodynamically-stable/functional conformations is important to advance our understanding of protein physics and allow drug discovery, protein-based sensor design, and protein engineering [2, 3]. Only computation is capable of providing detailed motions at a microscopic level. However, computational methods are challenged by the size and dimensionality of the protein conformation space, as well as the ruggedness of the underlying protein energy surface. In particular, standard frameworks such as Molecular Dynamics (MD) and Monte Carlo (MC) often get stuck in particular local minima and cannot find conformational paths connecting given functional conformations [4].

Algorithms based on robot motion planning have been proposed over the years, exploiting analogies between protein and robot motions [5]. These algorithms are either limited to small proteins of no more than 100 amino acids when employing no insight on the degrees of freedom (dofs) involved in the motion or heavily employ such insight and in turn have limited applicability [6–16].

Current robotics-inspired methods are tree-based or roadmap-based. Tree-based methods grow a tree search structure in conformation space from a given start to a given goal conformation. The growth of the tree is biased towards the goal. As such, tree-based methods conduct efficient albeit limited sampling of the conformation space. They are limited to finding essentially one path to the goal conformation, a setting known as single-query, and need to be run multiple times to obtain various paths. However, the bias in the growth of the tree causes path correlations among runs. Tree-based methods have successfully been employed to compute motions connecting functional conformations both in small peptides and large proteins of several hundred amino acids [11–16].

Roadmap-based methods can answer multiple queries through graph search algorithms on a constructed graph/roadmap of nearest-neighbor conformations. The conformations are sampled a priori. Such methods have been applied to compute mainly unfolding motions [6–10]. Several challenges limit broad applicability. Sampling conformations in regions of interest is difficult with no a priori knowledge. Once two nearest neighbors are connected with an edge as part of the roadmap construction, the motion represented by that edge needs to be computed or realized through a local search technique known as a local planner. The local planner needs to find intermediate conformations. Doing so is particularly challenging, either because the planner may have to connect vertices of the roadmap far away in conformation space, if the sampling has not been dense, or vertices separated by a high energy barrier. Significant computational time may be spent by local planners to realize all edges in the roadmap before being able to apply simple graph search algorithms to report paths connecting conformations of interest.

We propose here *SPIRAL*, which stands for `S`tochastic `P`rotein mot`I`on `R`oadmap `AL`gorithm. *SPIRAL* is a roadmap-based algorithm that assumes a limited computational budget and spends that budget in a priority-based scheme to realize promising paths. *SPIRAL* balances computational resources between a global search aimed at obtaining a global view of the network of protein conformations and their connectivity and a detailed local search focused on realizing such connections. In particular, *SPIRAL* is an adaptation of the fuzzy probabilistic roadmap method introduced for manipulation planning in robotics [17]. *SPIRAL* is designed to be general and not employ specific insight on where the relevant dofs are. The goal is to provide through *SPIRAL* a first-generation, general algorithm that can be used as a benchmark to further spur research into roadmap-based frameworks for computing protein motions connecting functional conformations arbitrarily far away in conformation space.

2 Methods

SPIRAL consists of two main stages, sampling and roadmap building. The sampling stage generates an ensemble of conformations/samples, Ω, that provide a discrete representation of the conformation space. In roadmap building, the roadmap $G = (V, E)$ first consists of pseudo-edges over nearest neighbors in Ω. A

time-limited iterative interplay between a global search and local search/planners converts pseudo-edges residing in a user-specified number (K) of promising paths into tree search structures of actual edges. At the expiration of time or successful computation of K paths, the roadmap is augmented with conformations and connections obtained by the local planners. All edge weights in the roadmap are recomputed to reflect energetic difficulty, and the resulting roadmap is queried for a specified number of lowest-cost paths. Various types of analyses can be conducted over these paths, whether in terms of energetic profile or proximity to given functional conformations.

2.1 Sampling Stage

SPIRAL extends the usual setting where two functional conformations are given to an arbitrary number of given conformations. The idea is to accommodate applications where a number $\ell \geq 2$ of stable or semi-stable functional conformations are known from experiment or computation for a protein of interest, and the goal is to map out the connectivity them. Let us refer to these conformations as landmarks. The landmarks are used to initialize Ω. The sampling stage then consists of a cycle of selection and perturbation operators. A selection operator selects a conformation within the current ensemble. Once selected, a perturbation operator is then sampled from a set of available ones and applied to the selected conformation to generate a new conformation. The generated conformation is checked for energetic feasibility prior to addition to the ensemble Ω. The process repeats until $|\Omega|$ reaches a pre-determined value.

Selection Operator The selection operator is based on our prior work on tree-based methods for protein motion computation [16] but extended here to deal with an arbitrary number of landmarks. The goal is to promote coverage of the conformation space enclosed by the landmarks. A progress coordinate, $\Delta R(C)_{i,j}$, is defined for each conformation C and a pair of landmarks (C_i, C_j) as in: $\Delta R(C)_{i,j} = \text{lRMSD}(C_i, C) - \text{lRMSD}(C_j, C)$. lRMSD here refers to least root-mean-squared-deviation used to measure the dissimilarity between two conformations after optimal superposition removes differences due to rigid-body motions [18]. The $\Delta R(C)_{i,j}$ coordinate is used to guide sampling towards under-sampled regions. For each pair of landmarks (C_i, C_j), a 1d grid is defined over the range $[-\text{lRMSD}(C_i, C_j) - 2, \text{lRMSD}(C_i, C_j) + 2]$. Each cell in the grid is 1Å wide. All conformations in Ω are projected onto this grid. In this way, each conformation in the growing ensemble Ω has $\binom{\ell}{2}$ projections, one in each of the $\binom{\ell}{2}$ grids. The selection operator proceeds as follows. A pair of landmarks is selected uniformly at random among the $\binom{\ell}{2}$ pairs. This determines the 1d grid, from which a cell is then sampled according to a probability distribution function defined weights w_c associated with cells of a grid. To bias the selection of conformations from under-explored regions of the conformation space, $w_c = \frac{1}{(1+\text{ns})*\text{nc}}$, where ns is the number of times the cell has been selected, and nc is the number

of conformations projected onto that cell. Once a cell is selected, a conformation from that cell is then selected uniformly at random.

Perturbation Operators In the absence of any specific insight onto the dofs of relevance, *SPIRAL* employs a set of perturbation operators in order to make moves of different granularities in conformation space in the sampling stage. Each perturbation operator has to satisfy a set of constraints. One of the constraints enforces energetic feasibility of generated conformations. The energy of a conformation $C^{'}$ generated from a selected conformation C, measured through the Rosetta *score3* function, is compared to the energy of C through the Metropolis criterion (*score3* is the backbone-level energy function, as we employ here only backbone-level representations of protein structure). If this fails, $C^{'}$ is not added to the ensemble. If it passes, $C^{'}$ is checked for satisfaction of distance-based constraints. Additional constraints are introduced on the minimum lRMSD ϵ_{min} of $C^{'}$ to any other conformation in the ensemble Ω and the maximum lRMSD δ of $C^{'}$ to the ℓ landmarks. The first constraint prevents redundant conformations from being added to Ω. The second constraint prevents sampling from veering off in regions of the conformation space deemed far from the landmarks to be useful for participating in paths connecting them. While ϵ_{min} is a parameter that can depend on the specific system under investigation (analysis is provided in section3), a reasonable value for δ is 150% of the maximum lRMSD between any pairs of landmarks.

The idea behind making various perturbation operators available to *SPIRAL* is to allow *SPIRAL* to select the perturbation operator deemed most effective based on features of the conformation space and the specific problem at hand. For instance, when the goal is to connect landmarks that reside far way from one another, a perturbation operator capable of making large moves is first desirable. Afterwards, to be able to make connections between such conformations, other perturbation operators capable of making smaller moves may be more effective. We consider here three perturbation operators, detailed below. An optimal weighting scheme that is responsive to emerging features of the search space is difficult to formulate and beyond the scope of the work here. However, we have been able to empirically determine a weighting scheme that is effective on most protein systems studied here.

Molecular Fragment Replacement Operator This operator is inspired from protein structure prediction, where backbone dihedral angles in a bundle/fragment of f consecutive amino acids are replaced altogether with values from a precompiled library. *SPIRAL* employs $f \in \{3, 9\}$ to balance between large ($f = 9$) and small ($f = 3$) moves.

Single Dihedral Replacement Operator This operator modifies a single backbone dihedral angle at a time to allow small moves. Given a selected backbone dihedral angle in a selected conformation, a new value from it is obtained using a normal distribution $\mathcal{N}(\mu, \sigma)$. The angle to perturb is selected uniformly at random. This operator, gaussian sampling, offers the option of biasing the selection of dihedral angles to promote selection of those that differ most between a

selected conformation and a landmark, though our application of *SPIRAL*here does not make use of biased gaussian sampling in the sampling stage.

Reactive Temperature Scheme: The energetic constraint that determines whether a conformation $C^{'}$ produced from a perturbation operator applied onto a selected conformation C should be added to Ω is based on the Metropolis criterion. Essentially, a probability $e^{-(E_{C'}-E_C)/(K\cdot T)}$, is measured, where K is the Boltzmann constant, and T is temperature. An arbitrary temperature value is both difficult to justify and obtain constraints-satisfying conformations as Ω grows (if T is low). So, as in previous work on tree-based methods [16], we make use of a reactive temperature scheme but extend it to the multiple-landmark setting here. We maintain a temperature value T_c for each cell c of the 1d grids over the progress coordinate. Each cell's temperature is adjusted every s steps (typical value employed is 25). The temperature of a cell, T_c, is increased if the last s selections of that cell have resulted in no conformations being added to Ω. If conformations are added to Ω more than 60% of the time within a window of s steps, T_c is decreased. Increases and decreases occur over adjacent temperature levels per a proportional cooling scheme that starts with very high temperatures in the $2,000$K range and ends with room temperature of 300K.

2.2 Roadmap Building Stage

All conformations in Ω are added to the vertex set V. For each $v \in V$, its k nearest-neighbors are identified, using lRMSD. For each identified neighbor, directional pseudo-edges are added with v. Additional pseudo-edges are added by identifying any vertex $< \epsilon_{max}$ from v that lies in a different connected component from v. Typical values for k and ϵ_{max} are 10 and 5Å, respectively.

The pseudo-edges are assigned a weight to reflect their estimated difficulty of being realizable. At initialization, all pseudo-edges are determined equally difficult with a weight value of 1. A two-layer scheme is then used, which is an iterative interplay between global and local search. The global search, path query, identifies the current most promising/lowest-cost path in the roadmap connecting two given functional conformations. If there are unrealizable edges in the path, these edges are fed to the local search, which launches local planners on unrealized edges, pursuing path realization. The planners are given a limited computational budget, and they report at the end of this budget either a realized edge or a new weight for the unrealized edges. In this iterative interplay between path query and path realization, over time, the pseudo-edges that are most difficult to realize will be assigned high weights and will thus be unlikely to participate in the lowest-cost path pushed to the local planners. This dynamic interplay apportions computational resources in a manner that promotes rapid path discovery. The iterative process continues until a total computational budget is exhausted or a user-specified number of paths is obtained.

Path Query and Path Realization Interplay: A pair of landmarks are selected uniformly at random over the $\ell!$ permutations. The roadmap is then queried for a lowest-cost path, using the assigned pseudo-edge weights. We utilize Yen's K-Shortest path algorithm [19] to identify the lowest non-zero cost path in the

graph and allow us to continue obtaining paths after the first path has been successfully realized. Given an identified path, a local planner is assigned to any of the unrealized edges. The planner is given a fixed computational budget, time T. If the local planner succeeds, the pseudo-edge it has realize is assigned a weight of 0 to indicate the pseudo-edge is resolved. If the local planner fails, the pseudo-edge is reweighted as in $w_e = 0.7 \cdot \text{CallsToPlanner} + 0.3 \cdot (\text{ClosestNode} - \text{RequireResolution})^2$. CallsToPlanner tracks the number of times the planner has been requested to work on a particular pseudo-edge, ClosestNode is the node in the tree constructed by the local planner that is closest to the vertex v in the directed pseudo-edge (u, v). For the planner to be successful, it must also generate a path that is within a user-specified lRMSD of the vertex v, so RequireResolution is also employed.

An additional feature of *SPIRAL* is its ability to learn from failures. When a local planner has failed to complete a path more than RefineLimit times, *SPIRAL* augments the graph with conformations identified by the local planner that are otherwise invisible to the global layer. We now proceed to relate details on the local planner and the augmentation procedure.

Local Planner: The local planner is an adaption of the tree-based method proposed in [16]. The adaptation consists of diversifying the types of perturbation operators employed in the expansion of the tree. The local planner selects through a probabilistic scheme shown in section 3 from the menu of perturbation operators described above. While biased gaussian sampling is not used in the sampling stage in *SPIRAL*, it is used by the local planner.

Roadmap Augmentation: Some regions of conformational space may be challenging to connect through local planners. This can be due to high energetic barriers or inadequate sampling. To address this issue, *SPIRAL* makes use of a feedback mechanism to augment the roadmap. When a local planner encounters difficulty realizing a pseudo-edge connecting given conformations p and r more than RefineLimit times (set at 25), the problem of connecting p to r is considered as a mini-version of the entire motion computation problem. The sampling scheme is repeated, essentially treating p and r as start and goal conformations. The perturbation operators described above are used together with a new one based on straight-line interpolation. The produced conformations are then minimized using the Rosetta *relax* protocol. Only the lowest-energy conformation is considered for addition. Conformations obtained from the perturbation operators are checked for satisfaction of the energetic and geometric constraints also used in the sampling stage. The operators are applied under a probabilistic scheme detailed in section 3 until either 25 conformations have been added to the roadmap or a maximum of 2500 attempts to do so have been made.

Roadmap Analysis: Each edge in the roadmap is reweighted to reflect energetic difficulty per the Metropolis criterion. A room temperature value is used for this purpose. The reweighted graph is queried for one or more lowest-cost paths, which are then analyzed in terms of energetic profile or distance within which they come of the goal landmark structures, as related in section 3.

3 Results

3.1 Systems of Study

Table 1 lists the protein systems selected for testing here. These are carefully gathered from published literature to provide comparisons where possible; not many published methods exist, and many of them either focus on few specific systems or are limited by system size. For most of the collected systems, two functional conformations have been extracted from literature (we consider both directions). The final column in Table 1 shows the lRMSD between the start and goal conformations. Neither size nor the lRMSD between functional conformations do by themselves define system difficulty. We have observed that the larger systems that exhibit smaller motions (less than 4.5Å lRMSD) between the start and goal conformations may require the protein chain to partially unfold before returning to a folded state. The process of unfolding a large, compact structure is computationally costly, as effectively an energy barrier needs to be crossed to get out of the compact state. Indeed, many computational studies avoid computing the motions involved in transitions from a closed to an open structural state because of this challenge.

Table 1. Protein systems for evaluation of performance.

System	Length	Start ↔ Goal	lRMSD(start, goal)
CVN	101	2ezm ↔ 1l5e	16.01 Å
CaM	140	1cfd ↔ 1cll	10.7 Å
		1cfd ↔ 2f3y	9.9 Å
		1cll ↔ 2f3y	13.44 Å
AdK	214	1ake ↔ 4ake	6.96 Å
LAO	238	1laf ↔ 2lao	4.7 Å
DAP	320	1dap ↔ 3dap	4.3 Å
OMP	370	1omp ↔ 3mbp	3.7 Å
BKA	691	1cb6 ↔ 1bka	6.4 Å

3.2 Implementation Details

SPIRAL is implemented in C++. A hard termination criterion is set with regards to the total number of energy evaluations. The sampling stage is terminated if the total number of energy evaluations exceeds $1,000$ times the requested ensemble size. That is, a maximum of 25 attempts are made to obtain a sample. The roadmap building stage is terminated after $10,000$ iterations of the interplay between path query and path realization. This stage may terminate earlier if $K = 250$ paths are obtained for all $\ell!$ landmarks as a way to control computational cost. The analysis stage reports the 50 lowest-cost paths. In terms of CPU time, the computational time demands of all these three stages in *SPIRAL* spans

anywhere from 56 hours on one CPU for protein systems around 100 amino acids long to 300 hours on one CPU for systems around 700 amino acids long.

During sampling, the molecular fragment replacement perturbation operator with $f = 3$ is selected 75% of the time, the operator with $k = 9$ is selected 20% of the time, and the gaussian sampling operator is selected 5% of the time. The reason for this scheme is to make large moves more often than small ones so as to spread out conformations in conformation space during sampling.

During roadmap building, the probabilistic scheme with which local planners and the roadmap augmentation make use of the perturbation operators is different, as shown in Table 2. A local planner can use two different schemes depending on the lRMSD between the two conformation/vertices it is asked to connect by the global layer. These schemes are not fine-tuned; essentially, when the distance is $\leq$ 2.5Å, smaller moves are promoted as opposed to when the distance is $>$ 2.5Å. The reason for basing the decision at 2.5Åis due to prior work on tree-based planners showing that molecular fragment replacement can result in step sizes greater than 2.5Å [16].

Table 2. The perturbation operator set and their weights during roadmap building.

Roadmap Building	Perturbation Operator	Prob.
Local Planner ($>$ 2.5 Å lRMSD)	Molecular Fragment Replacement ($f = 3$)	0.70
	Gaussian Sampling ($\mu = 0$, $\sigma = 15$)	0.15
	Biased Gaussian Sampling ($\mu = 0$, $\sigma = 15$)	0.15
Local Planner ($\leq$ 2.5 Å lRMSD)	Molecular Fragment Replacement ($f = 3$)	0.20
	Gaussian Sampling ($\mu = 0$, $\sigma = 15$)	0.40
	Biased Gaussian Sampling ($\mu = 0$, $\sigma = 15$)	0.40
Augmentation	Molecular Fragment Replacement($f = 3$)	0.20
	Gaussian Sampling ($\mu = 0$, $\sigma = 15$)	0.40
	Biased Gaussian Sampling ($\mu = 0$, $\sigma = 15$)	0.40
	Interpolation-based	0.05

The ϵ_{min} parameter controls how close neighboring conformations will be in the roadmap. Intuitively, smaller ϵ_{min} values would produce a better-quality roadmap. Our analysis indicates that this is not the case. Small values of ϵ_{min} ($<$ 1Å) can result in many small cliques being formed in the roadmap around local minima conformations. This is not surprising, particularly for the broad minima that contain the stable and semi-stable landmarks. On these minima, it is rather easy to sample a very large number of conformations nearby a landmark and thus essentially "get stuck" in the same local minimum. Insisting on a minimum distance separation among sampled conformations forces sampling not to provide refinement or exploitation of a particular local minimum but rather explore the breadth of the conformation space. Not insisting on a minimum distance pushes all the work to obtaining intermediate conformations to bridge local minima to the local planners, which is an ineffective use of computational time. The ϵ_{min}

parameter is set to 2.0Å for systems where the lRMSD between landmarks is > 6Å, 1.5Å for systems where the lRMSD between landmarks is > 4.5 but ≤ 6Å, and 1.0 for systems where the lRMSD between landmarks is ≤ 4.5Å.

3.3 Comparison of Found Paths with Other Methods

We compare *SPIRAL* to published tree-based methods [16, 15, 11, 12]. These methods make use of specific moves. For instance, our tree-based method in [16] uses molecular fragment replacements with $f = 3$, the method in [15] uses moves over low-frequency modes revealed by normal mode analysis, and the method in [11, 12] considers only backbone dihedral angles whose values change between the given functional conformations. The last two methods consider a low-dimensional search space of no more than 30 dimensions.

We report the closest that any path computed by *SPIRAL* comes to the specified goal conformation and compare such values on all protein systems to those reported in other published work. Columns $4-7$ in Table 3 show these values for *SPIRAL* and other published work. Column 3 reports some more details on the path with which *SPIRAL* comes closest to the goal conformation by listing the maximum lRMSD between any two consecutive conformations in the path. *SPIRAL* typically generates paths with conformations closer to the goal conformation than other methods (highlighted in bold where true). A video illustrating the lowest-cost conformational path reported by *SPIRAL*for the CVN protein can be found at http://youtu.be/7P4reYO3k-c.

3.4 Analysis of Energetic Profiles

We show the energetic profile of the lowest-cost path obtained by *SPIRAL* on two selected systems, AdK and CaM. We compare these profiles to those obtained by the interpolation-based planner described in section 2. For this planner, the resolution distance ϵ is set to 1.0 Å, and 50 cycles are performed to obtain a path. This provides a fair comparison, given that we also analyze 50 paths obtained after the analysis stage in *SPIRAL* and report here the lowest-cost one. Figure 1 shows that on proteins, such as AdK, where the distance between the start and goal conformations is large, paths provided by the interpolation-based planner tend to have higher energies than those provided by *SPIRAL*. On systems, such as CaM, where the start-to-goal distance is smaller, an interpolation-based planner can perform comparably to *SPIRAL*.

4 Conclusions

This paper has proposed *SPIRAL*, a novel protein motion computation algorithm capable of handling proteins of various sizes and settings where distances among functional conformations of interest can exceed 16Å. The algorithm is inspired by frameworks used in robot motion planning as opposed to MD- or MC-based frameworks. The main reason for doing so is to address the limited sampling

Table 3. Column 4 reports smallest distance to goal over all paths obtained by *SPIRAL*. Columns $5-7$ shows such distances from tree-based methods. Max Step in column 3 refers to the maximum lRMSD between any two consecutive conformations in the *SPIRAL* path that comes closest to the goal. '-' indicates lack of published data.

System	Start → Goal	Max Step	Dist to Goal (Å)			
			SPIRAL	Tree-based [16]	Cortés[15]	Haspel [11, 12]
CVN	2ezm → 1l5e	1.5	**1.5**	–	2.1	2.1
(101 aa)	1l5e → 2ezm	1.5	1.3	–	–	–
CaM	1cll → 1cfd	3.4	**1.46**	3.35	–	–
(144 aa)	1cfd → 1cll	2.67	**1.12**	3.17	–	–
	1cll → 2f3y	2.77	**1.26**	1.67	–	–
	2f3y → 1cll	3.5	1.12	0.73	–	1.33
	1cfd → 2f3y	3.33	**1.26**	3.5	–	–
	2f3y → 1cfd	3.48	**1.46**	3.2	–	–
AdK	1ake → 4ake	3.0	**1.86**	3.8	2.56	2.2
(214 aa)	4ake → 1ake	3.12	**1.33**	3.6	1.56	–
Lao	2lao → 1laf	2.0	**1.21**	–	1.32	–
(238 aa)	1laf → 2lao	3.2	1.90	–	–	–
DAP	1dap → 3dap	1.42	1.5	–	1.31	–
(320 aa)	3dap → 1dap	1.46	0.92	–	–	–
OMP	1omp → 3mbp	1.04	3.04	–	–	–
(370 aa)	3mbp → 1omp	0.91	3.61	–	–	–
BKA	1bka → 1cb6	3.87	**1.55**	–	2.79	–
(691 aa)	1cb6 → 1bka	3.98	1.69	–	–	–

in MD- or MC-based frameworks, particularly when motions involve disparate time and length scales.

*SPIRAL*exploits no particular information on any protein at hand. It is expected that tunings of the probabilistic scheme or employment of additional perturbation operators and moves based on specific system insight will improve performance. Future work will consider such directions, but the current need of the community is for a powerful, general, baseline method for the purpose of benchmarking.

The results shown here suggest *SPIRAL* produces good-quality paths and can be employed both to extract information on protein motions, possible long-lived intermediate conformations in such motions, as well as to advance algorithmic work in motion computation frameworks. In particular, the inherent prioritization scheme in *SPIRAL* allows the sampling of both low-cost paths and high-cost paths, provided enough computational budget is allocated. The latter paths may highlight possible local unfolding involved in protein motions connecting functional conformations. An executable of *SPIRAL* can be provided to researchers upon demand.

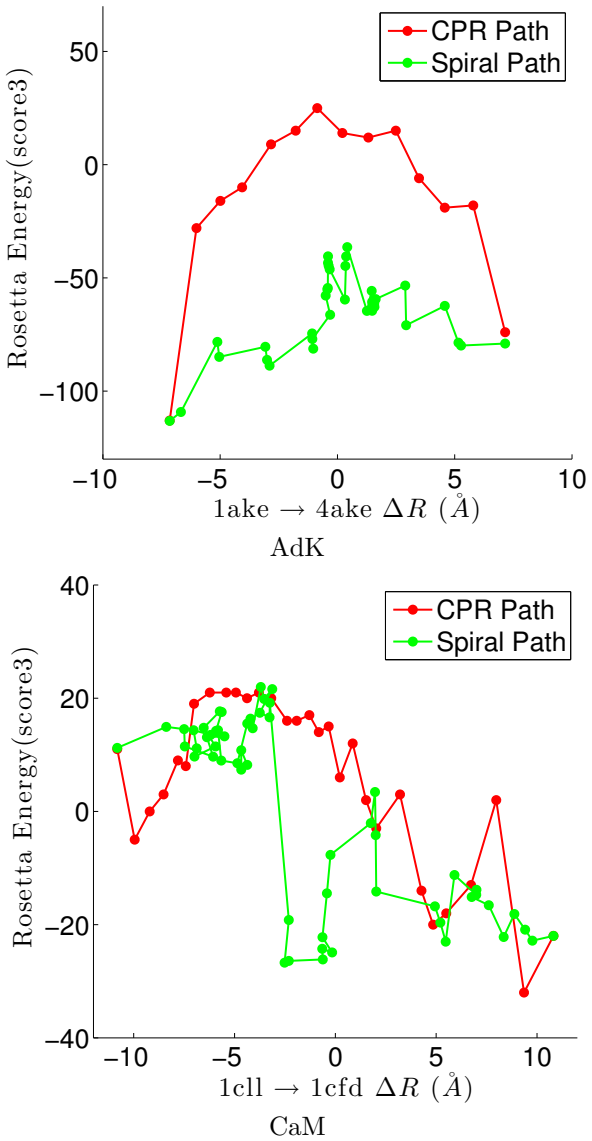

Fig. 1. Energy profiles of conformational paths computed for AdK (top) and of CaM (bottom) by *SPIRAL*(green) and an interpolation-based planner (red).

Acknowledgement

Experiments were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University, VA (URL: http://orc.gmu.edu). Funding for this work is provided in part by the National Science Foundation (Grant No. 1440581 and CAREER Award No. 1144106).

References

1. Jenzler-Wildman, K., Kern, D.: Dynamic personalities of proteins. Nature **450** (2007) 964–972
2. Wong, C.F., A., M.J.: Protein simulation and drug design. Adv. Protein Chem. **66** (2003) 87–121
3. Merkx, M., Golynskiy, M.V., Lindenburg, L.H., Vinkenborg, J.L.: Rational design of FRET sensor proteins based on mutually exclusive domain interactions. Biochem Soc Trans **41** (2013) 128–134
4. Amaro, R.E., Bansai, M.: Editorial overview: Theory and simulation: Tools for solving the insolvable. Curr. Opinion Struct. Biol. **25** (2014) 4–5
5. Singh, A.P., Latombe, J.C., Brutlag, D.L.: A motion planning approach to flexible ligand binding. In Schneider, R., Bork, P., Brutlag, D.L., Glasgow, J.I., Mewes, H.W., Zimmer, R., eds.: Proc Int Conf Intell Sys Mol Biol (ISMB). Volume 7., Heidelberg, Germany, AAAI (1999) 252–261
6. Amato, N.M., Dill, K.A., Song, G.: Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures. J. Comp. Biol. **10** (2002) 239–255
7. Thomas, S., Song, G., Amato, N.: Protein folding by motion planning. Physical Biology (2005) S148–S155
8. Thomas, S., Tang, X., Tapia, L., Amato, N.M.: Simulating protein motions with rigidity analysis. J. Comput. Biol. **14** (2007) 839–855
9. Tapia, L., Tang, X., Thomas, S., Amato, N.: Kinetics analysis methods for approximate folding landscapes. Bioinformatics **23** (2007) i539i548
10. Tapia, L., Thomas, S., Amato, N.: A motion planning approach to studying molecular motions. Communications in Information Systems **10** (2010) 53–68
11. Haspel, N., Moll, M., Baker, M.L., Chiu, W., E., K.L.: Tracing conformational changes in proteins. BMC Struct. Biol. **10** (2010) S1
12. Luo, D., Haspel, N.: Multi-resolution rigidity-based sampling of protein conformational paths. In: CSBW (Computational Structural Bioinformatics Workshop), in proc. of ACM-BCB (ACM International conference on Bioinformatics and Computational Biology). (2013) 787–793
13. Cortés, J., Simeon, T., de Angulo, R., Guieysse, D., Remaud-Simeon, M., Tran, V.: A path planning approach for computing large-amplitude motions of flexible molecules. Bioinformatics **21** (2005) 116–125
14. Jaillet, L., Corcho, F.J., Perez, J.J., Cortés, J.: Randomized tree construction algorithm to explore energy landscapes. J. Comput. Chem. **32** (2011) 3464–3474
15. Al-Bluwi, I., Vaisset, M., Siméon, T., Cortés, J.: Modeling protein conformational transitions by a combination of coarse-grained normal mode analysis and robotics-inspired methods. BMC Structural Biology **13** (2013) S8
16. Molloy, K., Shehu, A.: Elucidating the ensemble of functionally-relevant transitions in protein systems with a robotics-inspired method. BMC Struct Biol **13** (2013) S8
17. Nielsen, C., Kavraki, L.: A two level fuzzy prm for manipulation planning. In: Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on. Volume 3. (2000) 1716–1721 vol.3
18. McLachlan, A.D.: A mathematical procedure for superimposing atomic coordinates of proteins. Acta Crystallogr. A. **26** (1972) 656–657
19. JY, Y.: Finding the k shortest loop less paths in a network. Management Science **17** (1971) 712–716