As-Rigid-As-Possible molecular interpolation paths

Minh Khoa Nguyen, Léonard Jaillet, Stephane Redon

Noname manuscript No.
(will be inserted by the editor)

As-Rigid-As-Possible molecular interpolation paths

Minh Khoa Nguyen · Léonard Jaillet · Stéphane Redon

This is the preprint version. The published version can be viewed at https://doi.org/10.1007/s10822-017-0012-y (doi: 10.1007/s10822-017-0012-y)

Abstract This paper proposes a new method to generate interpolation paths between two given molecular conformations. It relies on the As-Rigid-As-Possible (ARAP) paradigm used in Computer Graphics to manipulate complex meshes while preserving their essential structural characteristics. The adaptation of ARAP approaches to the case of molecular systems is presented in this contribution. Experiments conducted on a large set of benchmarks show how such a strategy can efficiently compute relevant interpolation paths with large conformational rearrangements.

Keywords As-Rigid-As-Possible · morphing · interpolation path · molecular structures

1 Introduction

During the last few decades, important progresses in experimental approaches have allowed to generate massive collections of databases describing three dimensional (3D) structures of biomolecular systems, especially proteins [1,2]. While the quality and diversity of the obtained structures keep growing, each representation only corresponds to an instantaneous snapshot of one of the possible (meta)stable states of the system. However, in many cases, these systems may undergo large conformational changes corresponding to the transi-

We would like to gratefully acknowledge funding from the European Research Council through the ERC Starting Grant No. 307629.

Minh Khoa Nguyen E-mail: minh-khoa.nguyen@inria.fr ·
Léonard Jaillet E-mail: leonard.jaillet@inria.fr ·
Stéphane Redon E-mail: stephane.redon@inria.fr

Inria Grenoble Rhône-Alpes, France

Laboratoire Jean Kuntzmann, France

tions between the existing stable states. A good understanding of these transitions is essential since they are known to play important functional roles [3, 4]. Moreover, transitions are rare events, *i.e.* the transition time between two (meta)stable states is typically very small compared to the time spent in the stable states. Therefore, they are extremely difficult to capture experimentally. To address this difficulty while taking advantage of the increase in computational resources, various simulation methods have attempted to reconstruct such representative conformational changes [5].

Molecular Dynamics (MD) based on Newton's second law is a reference method to study the evolution of a given molecular system [6]. However, such an unbiased approach is typically unable to capture large structural rearrangements. Particularly for proteins, these rare events require the systems to cross over high energy barriers, resulting in a prohibitive computational cost. Therefore, many methods have been designed to facilitate these crossings and/or to reconstruct the transition paths between given conformations [7].

Some methods directly extend the MD principle [8]. Hence, in this category, we find the Transition Path Sampling framework which attempts to model these rare events and characterizes free energy pathways by focusing on the most interesting part of the simulation [9]. Other methods neglect (at least at some stage) the dynamic aspects of the system. For example, Gaussian network models [10–13] use mass-and-spring coarse-grain representations to find the normal modes of collective motions. Low frequency motions, which typically participate in large conformational rearrangements, can then be extracted [14]. The Nudged Elastic Band [15] and String [16] methods search for the Minimum Energy Path (MEP) between two stable states from an initial path. Recently, motion planning methods from Robotics have also been applied to the exploration of conformational spaces in order to predict transition paths [17–20].

The approach presented in this paper is a molecular morphing method [21] which primarily relies on geometrical information. The simplest morphing method to generate a path between two given structures is linear interpolation. However, it tends to create intermediate conformations with artifacts such as artificial scalings, expansions, distortions or unrealistic bond lengths and angles. Hence, more sophisticated methods have been designed to address these problems. The Morph-Pro method corrects the intermediate conformations generated by linear interpolation such that the distances between any two consecutive alpha carbons (C_α) lie within a particular range [22]. The Climber method [23] interpolates the adjacent C_α distances while imposing harmonic restraints among them. The well-known morph server [24] analyzes the difference between two given conformations by locating representative hinges. A path is then generated between these conformations by restrained interpolation using an adiabatic mapping technique. The generated paths are reasonably good in the sense that large chemical distortions are avoided. Their visual representations can help to better understand protein motions. With the FRODA method [25], rigidity analysis, geometric constraint and steric constraint are applied to simulate transition paths. A recent geometric target-

ing method inspired by FRODA but using a different mathematical formalism has also been proposed [26].

In the field of Computer Graphics, powerful tools have been proposed to easily manipulate and animate meshes. This article extends the As-Rigid-As-Possible (ARAP) approaches that are well known to perform such operations. These methods generate new shapes while attempting to preserve the rigidity of the reference shape. The ARAP idea was initially introduced by Alexa et al. for interpolation in two dimensions (2D) [27]. Since then, it has received a lot of attention. Apart from its 2D extensions [28–31], methods for 3D meshes have also been developed for interactive manipulation [32–35] and interpolation [32,36–38]. It is this latter application that inspired the present work. Some alternatives to ARAP are As-Isometric-As-Possible (AIAP) methods that preserve distances within the mesh [39,40], the As-Similar-As-Possible method (ASAP) which enforces the preservation of the mesh tetrahedrals [41], path deformation techniques based on physics [42] or possible dynamics [43], and interpolation methods based on specific surface representations [44–48]. The reader is referred to [49] for a survey of mesh deformation techniques.

This article is organized as follows: Section 2 describes the ARAP methodology and its application to molecular structures. Section 3 shows the results of the ARAP interpolation method on several benchmarks. The pros and cons of the method are discussed in Section 4. Finally, Section 5 concludes this work and gives some suggestions for future improvements and applications.

2 Methodology

This section presents the ARAP methodology to generate molecular paths. It follows the global framework presented in Figure 1. The input is a pair of conformations (intial and target) described by a set of atoms embedded in the 3D cartesian space and connected by covalent bonds. The output is a morphing path comprising intermediate conformations.

As we will see, ARAP interpolations can be efficiently computed and lead to consistent paths which tend to preserve the local rigidity of the models.

2.1 Preprocessing

The ARAP method requires a one-to-one atom mapping (or matching) between the intial and target structures. To obtain such a mapping, a sequence alignment of the residues in both structures is first performed using the MUSCLE method [50]. Atoms of the aligned residues are then mapped by their PDB names [1]. More robust mapping methods could be introduced, but this is outside the scope of this paper. Only matched atoms will be treated by the

[1] More specifically, two atoms are mapped if they share the same name or have equivalent names such as 1HG2 and HG21.

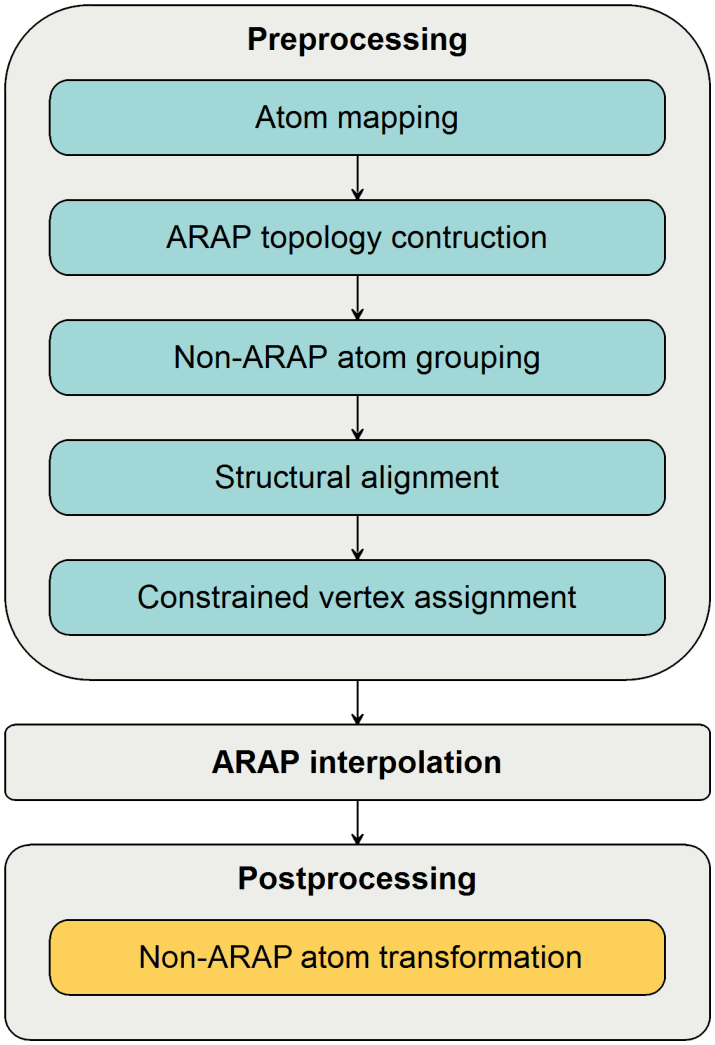

Fig. 1: Global framework to generate molecular paths from an initial to a target conformation using the ARAP interpolation method.

ARAP method and hence labeled as *ARAP atoms*, whereas the other atoms are labeled as *non-ARAP atoms*.

The *ARAP topology* is then built as follows: each ARAP atom is considered as a vertex and each bond between two ARAP atoms in the initial structure as an edge [2]. However, such a simple construction may produce disconnected

[2] Optionally, to build edges, one could consider only the bonds which are present in both the initial and the target structures.

components [3] due to missing residues in the initial structure or a broken connectivity when discarding non-ARAP atoms. Since the ARAP method requires that all vertices belong to one connected component, two additional procedures are done to recover the connectivity. First, if the initial molecular system is itself disconnected due to missing residues, we reconnect the structure by creating an edge between the atom just before and the one just after each missing group of residues [4]. Second, to create connections among ARAP atoms, we use a recursive procedure called *connect*, which operates as follows. For each ARAP atom a, $connect(a, m_i)$ is first called for each neighbor m_i of a. The procedure creates an edge between a and m_i if m_i is also an ARAP atom. If m_i is not an ARAP atom, then $connect(a, n_j)$ is called on each neighbor n_j of m_i. To ensure the termination of the recursive procedure, we check that each atom is visited at most once. The next step forms groups of connected non-ARAP atoms. Each group is associated to a reference ARAP atom that shares a bond with one of the atom in the group. As we will see later, the non-ARAP atom positions are deduced from the ARAP atom positions.

ARAP interpolations may only provide deformations. Hence, we need additional constraints to define the global rotation and translation of the system. To determine these constraints, we perform a 3D structural alignment of the initial and the target structures. This cancels the need for any global rotation, while improving the robustness of the solution obtained, by removing large local rotations during the ARAP interpolation. We use the fast QCP variant developed by Liu et al. [51] of the quaternion superposition methods [52,53] to align both structures. The last step of the preprocessing phase settles the translational part by arbitrarily forcing one of the ARAP atoms to have a linear motion in the final ARAP interpolation.

In summary, the preprocessing phase provides the following inputs to the ARAP algorithm: the (aligned) vertex positions of the initial and target states, the edges that link these vertices together, and the index of the constrained vertex.

2.2 As-Rigid-as-Possible interpolation

As stated in the introduction, several implementations and extensions of the ARAP methodology exist. Here, the proposed formalism follows the original extension for 3D meshes [32].

2.2.1 ARAP formalism

Let us assume a molecular structure described by n vertices $v_1, \ldots, v_n$ connected by edges. Let $\mathbf{p}_i \in \mathbf{R}^3$ be the vertex positions in the initial state $\mathcal{S}$ and $\mathbf{p}'_i$ their positions in the target state $\mathcal{S}'$ for $i \in [1, n]$.

[3] A connected component is a set of vertices and edges such that any vertex can be reached from another vertex by traversing through a series of edges in the same component.

[4] More specifically, the chosen atoms to create the edge are alpha carbon atoms

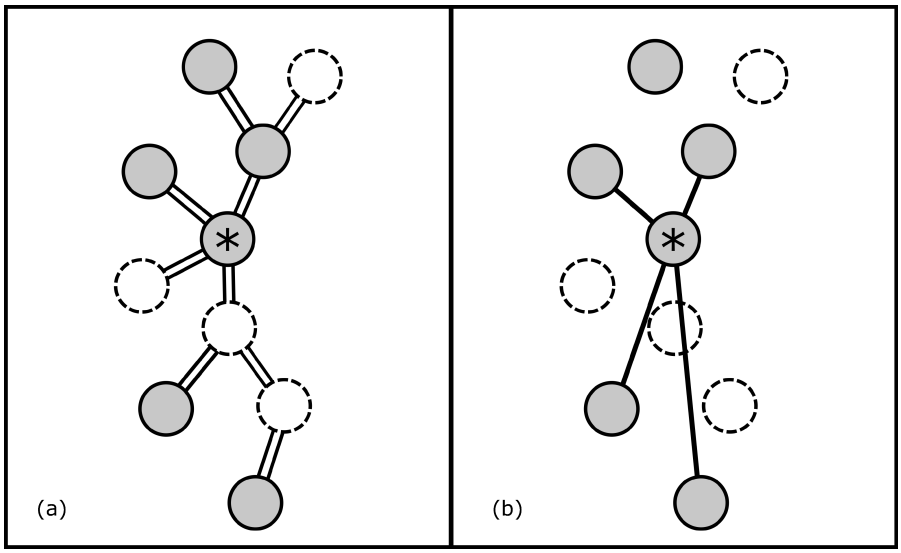

Fig. 2: Example of ARAP connectivity construction starting from the ARAP atom marked with a star. ARAP atoms are grey, while non-ARAP atoms are white. (a) The initial molecular topology. (b) ARAP edges created from the marked ARAP atom and represented by bold lines. When the *connect* procedure stops for the marked atom, the top-left ARAP atom is not connected, since its connection to the marked atom passes through another ARAP atom.

From the molecular topology, one can extract n sets $\mathcal{N}_i$, where each set consists of a central vertex v_i and all the vertices whose topological distances to this vertex is lower than a given value k. The maximum distance used to define a set is typically $k = 1$, thus including only the one-ring neighbors which are the vertices directly connected to the central vertex v_i.

From each topological set, one can define a cell $\mathcal{C}_i$ consisting of the vertex positions of the set $\mathcal{N}_i$ for a given state. An example of cells obtained from a molecular system using the one-ring neighborhood is shown in Figure 3.

Let us now consider $\mathcal{C}_i$ and $\mathcal{C}'_i$, the cells centered on vertex i in the initial and target states, respectively. As stated in [32], if the transformation $\mathcal{C}_i \rightarrow \mathcal{C}'_i$ is rigid, there exists a rotation $\mathbf{R}_i$ such that:

$$\mathbf{p}'_i - \mathbf{p}'_j = \mathbf{R}_i(\mathbf{p}_i - \mathbf{p}_j), \forall j \in \mathcal{N}_i \tag{1}$$

Hence, Sorkine et al. define the approximate rigid transformation between the two cells as the rotation $\mathbf{R}_i$ that minimizes the *cell deformation energy* $E(\mathcal{C}_i, \mathcal{C}'_i)$, also called *ARAP cell energy*:

$$E(\mathcal{C}_i, \mathcal{C}'_i) = \sum_{j \in \mathcal{N}_i} \omega_{ij} ||\mathbf{p}'_i - \mathbf{p}'_j - \mathbf{R}_i(\mathbf{p}_i - \mathbf{p}_j)||^2 \tag{2}$$

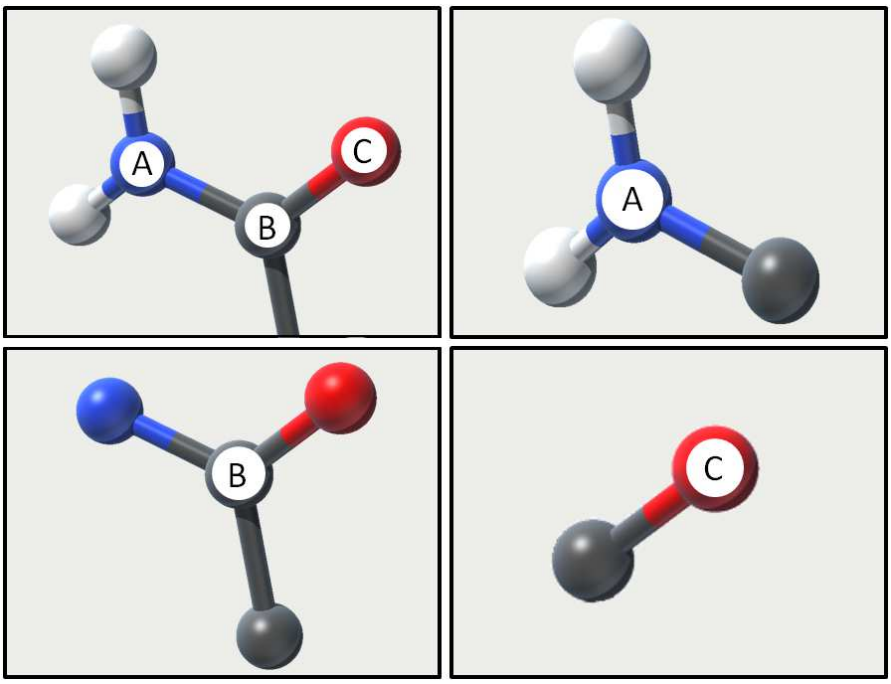

Fig. 3: Some part of a protein (top-left) and examples of cell decomposition for three atoms: nitrogen (A/upper-right), carbon (B/lower-left), and oxygen (C/lower-right).

where ω_{ij} is the weight associated to edge e_{ij} connecting v_i with v_j in $\mathcal{N}(i)$. By default, $\omega_{ij} = \omega_{ji} = 1$. However, it is possible that $\omega_{ij} \neq \omega_{ji}$. Figure 4 shows an example of a rotation $\mathbf{R}_i$ which minimizes the ARAP cell energy.

Considering now all the cells, an ARAP transformation minimizes the total ARAP energy E defined as:

$$\begin{aligned} E &= \sum_i \omega_i E(\mathcal{C}_i, \mathcal{C}'_i) \\ &= \sum_i \omega_i \sum_{j \in \mathcal{N}_i} \omega_{ij} ||\mathbf{p}'_i - \mathbf{p}'_j - \mathbf{R}_i(\mathbf{p}_i - \mathbf{p}_j)||^2 \end{aligned} \tag{3}$$

where ω_i is the cell weight. We will now explain how to compute an interpolation path from such a formulation.

2.2.2 Resolution method

The ARAP interpolation method is described in Algorithm 1. It takes as inputs the vertex positions in the initial and the target states, the desired number of states $\mathcal{L}$ in the interpolation path and the index c of the constrained vertex. First, ARAP cells are generated for the initial and target states (lines 2 and

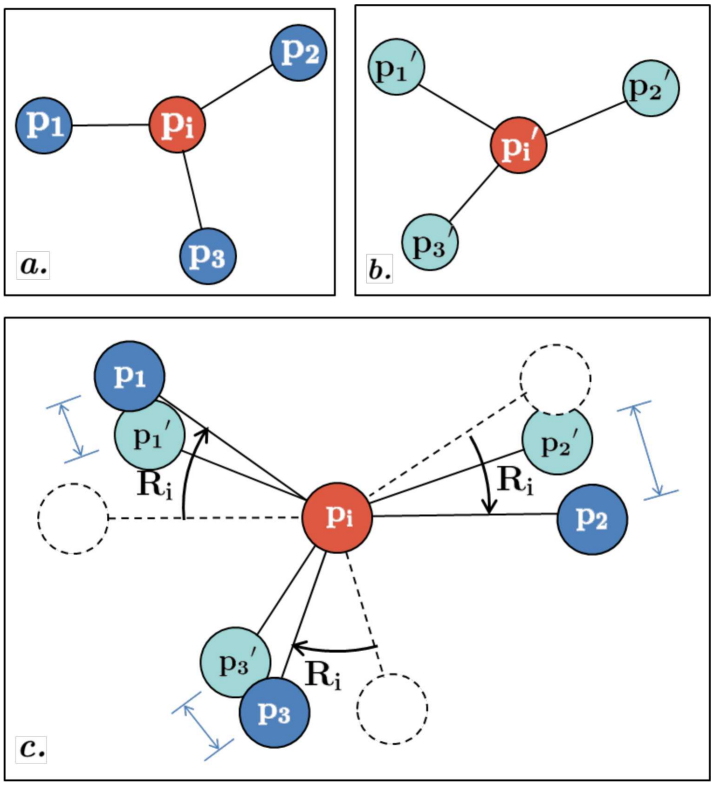

Fig. 4: The local ARAP rotation that maximizes rigidity. **a.** The initial cell. **b.** The target cell. **c.** The optimal rotation $\mathbf{R_i}$ minimizes the sum of the rotated $||\mathbf{p_j} - \mathbf{p'_j}||^2$ distances (arrow bars), after aligning the cells such that $\mathbf{p_i} = \mathbf{p'_i}$.

3). Then, each ARAP rotation $\mathbf{R}_i$ is computed (line 4) by minimizing the cell energy in Equation 2, using the fast QCP variant developed by Liu et al. [51].

Each intermediate state l along the path has a corresponding value t defined at line 6. Hence, $t \in [0, 1]$ and the structure is at the initial and target states when $t = 0$ and $t = 1$, respectively. For each value of t, an interpolated rotation $\mathbf{R}_i(t)$ is evaluated for each cell (line 8). An option to perform such an interpolation is presented in Section 2.2.3. Then, the position of the constrained vertex at t is computed by linearly interpolating its positions in the initial and target states (line 9). Finally, for each intermediate state, we compute the vertex positions $\mathbf{p'_i}(t)$ by minimizing the total ARAP energy in Equation 3 (line 10), *i.e.*

$$\mathbf{p}'_1(t), \ldots, \mathbf{p}'_n(t) = \arg\min \Big(\sum_i \omega_i \sum_{j \in \mathcal{N}_i} \omega_{ij} ||\mathbf{p}'_i(t) - \mathbf{p}'_j(t) - \mathbf{R}_i(t)(\mathbf{p}_i - \mathbf{p}_j)||^2 \Big) \quad (4)$$

Algorithm 1: ARAP molecular interpolation

Input : The initial vertex positions $\mathbf{p}_i$. The target vertex positions $\mathbf{p}'_i$. The edges. The number of states $\mathcal{L}$ in the interpolation path. The index c of the constrained vertex.

Output: A smooth path consisting of $\mathcal{L}$ states.

for $i = 1, \ldots, n$ **do**
  $\mathcal{C}_i \leftarrow \text{ComputeCell}(\mathbf{p}_i)$;
  $\mathcal{C}'_i \leftarrow \text{ComputeCell}(\mathbf{p}'_i)$;
  $\mathbf{R}_i \leftarrow \text{ComputeRotation}(\mathcal{C}_i, \mathcal{C}'_i)$;
for $l = 0, \ldots, \mathcal{L} - 1$ **do**
  $t = \frac{l}{\mathcal{L}-1}$;
  for $i = 1, \ldots, n$ **do**
    $\mathbf{R}_i(t) \leftarrow \text{ComputeInterpolatedRotation}(\mathbf{R}_i, t)$;
  $\mathbf{p}_c(t) \leftarrow (1-t)\mathbf{p}_c + t\mathbf{p}'_c$;
  $\mathbf{p}'_1(t), \ldots, \mathbf{p}'_n(t) \leftarrow \text{ComputePositions}(\mathbf{R}_1(t), \ldots, \mathbf{R}_n(t))$;
return $\mathcal{L}$ sets of vertex positions $\mathbf{p}'_i(t)$;

The method to solve such a system is presented in Section 2.2.4.

2.2.3 Interpolating Rotations

We use the Spherical linear interpolation (Slerp) method to interpolate rotations [54]. This method produces a rotation at a constant angular velocity along the shortest great arc (*i.e.* a geodesic) on a unit quaternion sphere. The formula to compute a Slerp between two normalized quaternions p and q with parameter $t \in [0, 1]$ is:

$$\text{Slerp}(p, q, t) = p(p^* q)^t \tag{5}$$

where p^* is the conjugate of p. This expression can be rewritten without the exponentiation as:

$$\text{Slerp}(p, q, t) = \frac{\sin(1-t)\theta}{\sin\theta} p + \frac{\sin t\theta}{\sin\theta} q \tag{6}$$

where $\cos\theta = p \cdot q$ is the inner product of two quaternions (*i.e.* if $p = [s, (x, y, z)]$ and $q = [s', (x', y', z')]$, then $p \cdot q = ss' + xx' + yy' + zz'$).

2.2.4 Finding ARAP positions

To find $\mathbf{p}'_1(t), \ldots, \mathbf{p}'_n(t)$ in Equation 4, one sets to zero the derivative of the expression inside $\arg\min$ with respect to each unknown $\mathbf{p}'_k(t)$ with $k \in [1, n] \setminus c$, where c is the constrained index. This produces a system of linear equations $\mathbf{LP} = \mathbf{b}$ where $\mathbf{P}$ is a $(n-1) \times 3$ matrix concatenating the unknown vertex positions $\mathbf{p}'_k(t)$. $\mathbf{b}$ has the same size as $\mathbf{P}$. The $(n-1) \times (n-1)$ matrix $\mathbf{L}$ is the discrete Laplace-Beltrami operator applied to $\mathbf{P}$.

Thanks to the one-connected-component property and the constrained vertex, this system always has a unique solution.

The details on constructing these matrices can be found in [32]. However, we also briefly give their derivation here. For the sake of simplification, let us drop t in Equation 4 for the moment. Then, setting to zero the derivative of the expression inside arg min of the equation with respect to $\mathbf{p}'_k$, one obtains

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{p}'_k} =& \frac{\partial}{\partial \mathbf{p}'_k}\Big(\omega_k \sum_{j\in\mathcal{N}_k} \omega_{kj}||\mathbf{p}'_k - \mathbf{p}'_j - \mathbf{R}_k(\mathbf{p}_k - \mathbf{p}_j)||^2 \\ &+ \sum_{j\in\mathcal{N}_k} \omega_j\omega_{jk}||\mathbf{p}'_j - \mathbf{p}'_k - \mathbf{R}_j(\mathbf{p}_j - \mathbf{p}_k)||^2\Big) = 0\end{aligned}$$

$$\begin{aligned}&2\omega_k \sum_{j\in\mathcal{N}_k} \omega_{kj}\left(\mathbf{p}'_k - \mathbf{p}'_j - \mathbf{R}_k(\mathbf{p}_k - \mathbf{p}_j)\right) \\ &- 2\sum_{j\in\mathcal{N}_k} \omega_j\omega_{jk}\left(\mathbf{p}'_j - \mathbf{p}'_k - \mathbf{R}_j(\mathbf{p}_j - \mathbf{p}_k)\right) = 0\end{aligned}$$

$$\begin{aligned}&\left(\sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj} + \omega_j\omega_{jk})\right)\mathbf{p}'_k - \sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj} + \omega_j\omega_{jk})\mathbf{p}'_j \\ &= \sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj}\mathbf{R}_k + \omega_j\omega_{jk}\mathbf{R}_j)(\mathbf{p}_k - \mathbf{p}_j)\end{aligned}$$

We now bring back t into the last equation to obtain,

$$\begin{aligned}&\left(\sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj} + \omega_j\omega_{jk})\right)\mathbf{p}'_k(t) - \sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj} + \omega_j\omega_{jk})\mathbf{p}'_j(t) \\ &= \sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj}\mathbf{R}_k(t) + \omega_j\omega_{jk}\mathbf{R}_j(t))(\mathbf{p}_k - \mathbf{p}_j)\end{aligned} \tag{7}$$

The last equation is used to construct matrices $\mathbf{L}$ and $\mathbf{b}$ when the constrained vertex $c \notin \mathcal{N}_k$. Hence, the diagonal and off-diagonal coefficients of $\mathbf{L}$ are those in front of $\mathbf{p}'_k(t)$ and $\mathbf{p}'_j(t)$ in the equation, respectively. The transpose of the right hand size in Equation 7 constitutes a row in $\mathbf{b}$. When $c \in \mathcal{N}_k$, the following equation is used instead,

$$\begin{aligned}&\left(\sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj} + \omega_j\omega_{jk})\right)\mathbf{p}'_k(t) - \sum_{j\in\mathcal{N}_k\setminus c}(\omega_k\omega_{kj} + \omega_j\omega_{jk})\mathbf{p}'_j(t) \\ &=(\omega_k\omega_{kc} + \omega_c\omega_{ck})\mathbf{p}'_c(t) + \sum_{j\in\mathcal{N}_k}(\omega_k\omega_{kj}\mathbf{R}_k(t) + \omega_j\omega_{jk}\mathbf{R}_j(t))(\mathbf{p}_k - \mathbf{p}_j)\end{aligned} \tag{8}$$

As can be seen, $\mathbf{L}$ is a sparse, symmetric and positive definite matrix, for which we can thus compute a Cholesky decomposition. Because its coefficients are independent of t, this decomposition can be performed only once, and used to compute all intermediate states.

2.2.5 Implementation details

Cell definition We use the same cell definition as in [32], *i.e.* each cell is composed of one vertex and its one-ring neighbors. Also, we set the cell weights $\omega_i = 1$ for all cells and the weights associated to the cell edges $\omega_{ij} = \omega_{ji} = 1$.

Reaching the target Direct application of the ARAP method does not allow to reach the target perfectly, *i.e.* the computed conformation at $t = 1$ does not coincide with the target conformation. To overcome this limitation, we introduce for each edge e_{ij} an extra rotation $\mathbf{R}_{ij}$ and a stretching s_{ij}. Hence, after solving for $\mathbf{R}_i$ by minimizing Equation 2, we find for each set $\mathcal{N}_i$ and each $j \in \mathcal{N}_i$ the rotation $\mathbf{R}_{ij}$ and stretching s_{ij} such that:

$$s_{ij}\mathbf{R}_{ij}\mathbf{R}_i(\mathbf{p}_i - \mathbf{p}_j) = \mathbf{p}'_i - \mathbf{p}'_j \tag{9}$$

The equation to find the atom positions for the intermediate conformations is adapted accordingly:

$$\mathbf{p'_1}(t), \ldots, \mathbf{p'_n}(t) = \arg\min \Big(\sum_i \omega_i \sum_{j \in \mathcal{N}_i} \omega_{ij} ||\mathbf{p'_i}(t) - \mathbf{p'_j}(t) - s_{ij}(t)\mathbf{R}_{ij}(t)\mathbf{R}_i(t)(\mathbf{p_i} - \mathbf{p_j})||^2 \Big) \tag{10}$$

where $s_{ij}(t)$ is linearly interpolated between 1 and s_{ij} (*i.e.* $s_{ij}(t) = (1-t) + ts_{ij}$) and $\mathbf{R}_{ij}(t)$ is computed as in Section 2.2.3.

2.3 Postprocessing

During the preprocessing phase, each non-ARAP atom n was associated to an ARAP atom a_n. Hence, once an intermediate conformation is found at t, the position of the non-ARAP atom $\mathbf{p}_n(t)$ can be deduced from the position of the ARAP atom $\mathbf{p}_{a_n}(t)$ as:

$$\mathbf{p}_n(t) = \mathbf{p}_{a_n}(t) + \mathbf{R}_{a_n}(t)\,(\mathbf{p}_n - \mathbf{p}_{a_n}) \tag{11}$$

where $\mathbf{R}_{a_n}(t)$ is the intermediate rotation of the cell whose central vertex is atom a_n.

3 Results

We have implemented the proposed method in C++ as a module of the SAMSON software platform [55]. The Eigen library is used for solving the linear system of equations [56].

All benchmarks proposed here come from those proposed by the authors of the so-called geometric targeting method in [26]. Since the ARAP method requires connected components, only 13 benchmarks of single-chain molecules

Table 1: Benchmark details and time of execution. A chain id is specified after each pdb code.

#	Protein Name	Initial/Target Structure	No. of non-ARAP and ARAP atoms	CPU Time (ms)				Figure
				Pre-processing	ARAP time per conformation	Post-processing	Total	
1	5'-Nucleotidase	1HP1(A)/1HPU(C)	2/4027	130	22	0	570	10a
2	Adenylate Kinase	4AKE(A)/1AKE(A)	0/1656	47	9	0	227	7a
3	Alcohol Dehydrogenase	8ADH(A)/6ADH(A)	4/2784	87	16	0	407	8a
4	Calmodulin	1CFD(A)/1CFC(A)	1072/1166	49	6	33	202	7b
5	Collagenase	1NQD(A)/1NQJ(B)	0/901	26	5	0	126	9a
6	Dengue 2 Virus Envelope Glycoprotein	1OAN(A)/1OK8(A)	163/2962	95	17	6	441	10d
7	Dihydrofolate Reductase	1RX2(A)/1RX6(A)	1/1268	36	7	0	176	8b
8	Diphtheria Toxin	1DDT(A)/1MDT(A)	0/4021	130	23	0	590	10b
9	DNA Polymerase	1IH7(A)/1IG9(A)	26/7313	261	41	1	1082	10c
10	Pyrophosphokinase	1HKA(A)/1Q0N(A)	0/1267	34	7	0	174	8c
11	Pyruvate Phosphate Dikinase	1KBL(A)/2R82(A)	7/6745	234	38	0	994	10e
12	Spindle Assembly Checkpoint Protein	1DUJ(A)/1KLQ(A)	1479/1509	64	8	46	270	9b

were considered. Among them, Heparin Cofactor II was removed because its target structure has too many missing residues (more than 20 consecutive residues). Therefore, only 12 benchmarks are presented in this article. For each case, the ARAP interpolation method is applied to generate a path composed of 20 conformations. The benchmark names, as well as the computational times, are shown in Table 1. The ARAP time is shown per conformation because it grows linearly with the number of conformations on the path.

As one can see, our methodology has globally a very low computational cost. The highest total time to obtain a path of 20 conformations is 1.082 s for DNA Polymerase which has 7313 ARAP atoms. The interpolation time per conformation per ARAP atom (not shown in the table) is about 0.006 ms for all benchmarks. This value is 13 to 82 times smaller for the same quantity shown in [26]. This is an advantage of the ARAP method, which solves a linear system $\mathbf{LP} = \mathbf{b}$ where the square matrix $\mathbf{L}$ is sparse. Hence, the complexity for computing $\mathbf{P}$ is close to $\mathcal{O}(N)$ where N, the diagonal size of $\mathbf{L}$, is the number of ARAP atoms subtracted by 1 (the constrained vertex).

When visually inspecting the paths obtained by the ARAP method, the results appear reasonable except for Collagenase and Spindle Assembly Checkpoint Protein, where steric clashes occur. We provide in the following a more detailed analysis of the intermediate structures obtained from the ARAP path.

3.1 Bond lengths, bond angles and dihedral angles

To show the structure preservation property of the ARAP interpolation method, we look at how bond lengths, bond angles and dihedral angles in each intermediate conformation deviate from those in the initial conformation. Consider the first benchmark, 5'-Nucleotidase, which has a large rotation (about 90°) in one part of the structure (Figure 10a).

Figure 5a shows the absolute value of the change in bond length during interpolation with our approach. The horizontal axis shows the conformation sequence index along the path, where 1 corresponds to the first conformation and 20 corresponds to the last one. The vertical axis is the change in bond

length in angstrom (Å). The blue line plots the average change in bond length in each conformation. For each conformation, a blue bar shows the standard deviation from the mean value. The dotted and solid red lines plot the maximum and minimum values, respectively, of the change in bond length for each conformation. Therefore, the area between the maximum and minimum curves represents the range of the (absolute) change in bond length. The *deviation area*, *i.e.* the area between the blue bars, is where most values are observed.

The maximum curve in Figure 5a starts at 0 Å for the first conformation because here, it coincides exactly with the initial conformation. The maximum value increases until 0.616 Å (the 11^{th} conformation), then decreases and finally reaches 0.281 Å at the target conformation. Our method does indeed reach the target conformation. However, this final value is not 0 Å because the bond lengths of the target structures deviate from those of the initial structure to some degree. The deviation area is bounded in the range [-0.012, 0.025] Å, indicating that a large amount of bond lengths generated by the ARAP interpolation do not deviate more than 0.025 Å from those in the initial conformation.

The same type of plot for the linear interpolation method [5] is shown in Figure 5d. The maximum values are significantly larger (the peak is around 1.118 Å at the 10^{th} conformation) compared with those of the ARAP interpolation method. The deviation area also has higher mean and standard deviation values, which suggests that the ARAP interpolation method preserves bond lengths much better than the linear interpolation method.

Similarly, Figure 5b and 5e show that ARAP interpolation preserves bond angles better than linear interpolation. On the other hand, the results on the dihedral angle do not differ greatly in both methods as shown in Figure 5c and 5f. This is because the chosen one-ring topology of a cell contains no dihedral angles, and hence dihedral angles are not constrained by the proposed method. Therefore, the method naturally favors global deformation through changes in dihedral angles, which is typically expected in conformational changes of molecular systems. Further discussion on the dihedral angle can be found in Section 4.1.

We have detailed above the results for 5'-Nucleotidase, but the same behavior is observed for all benchmarks. Table 2 shows the maximum mean value across the path regarding the change in bond lengths, bond angles and dihedral angles for each benchmark. For bond lengths and bond angles, the values in the ARAP columns are always lower than those in the Linear columns, *i.e.* the ARAP method preserves these quantities better. However, the results for dihedral angles do not follow this pattern. In fact, maximal changes in dihedral angles occur at the target conformation, which is why the values are the same for both methods in the table.

[5] The path by the linear interpolation method is also generated after the initial and target conformations are 3D structurally aligned.

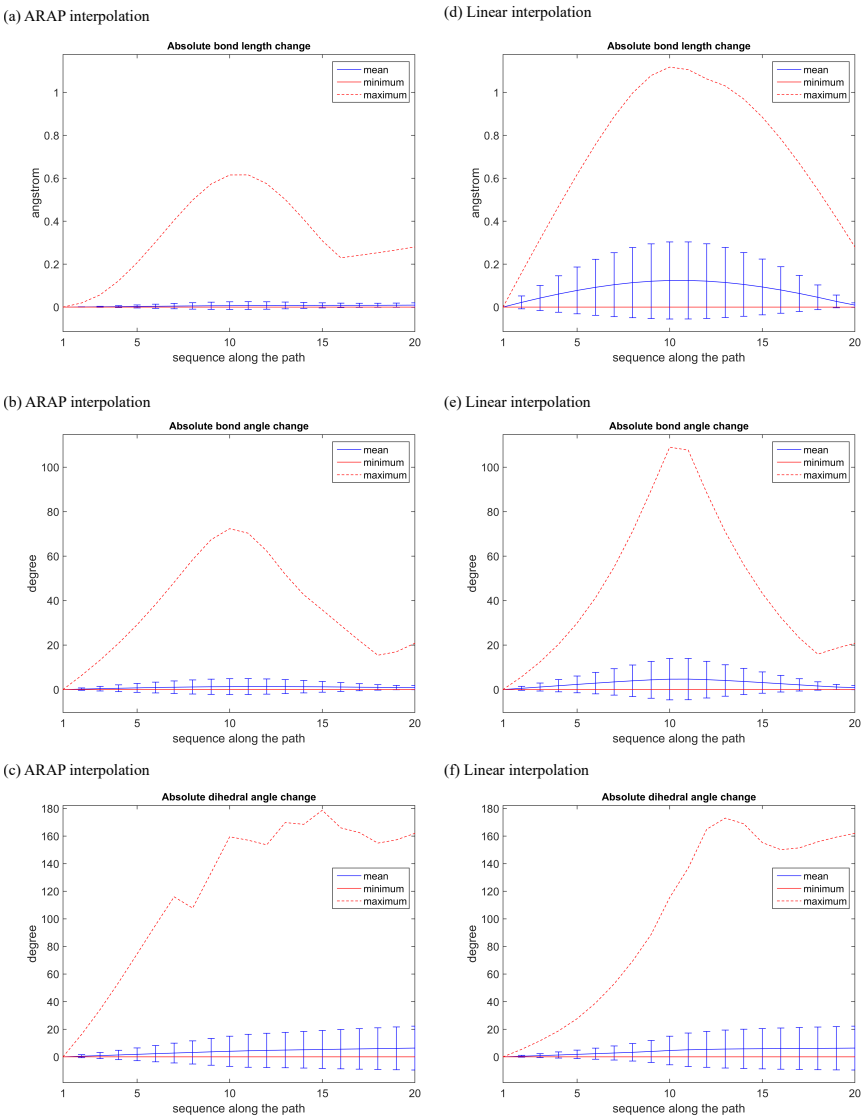

Fig. 5: Statistics of absolute changes in bond length, bond angle and dihedral angle for 5'-Nucleotidase. Results from ARAP interpolation for (a) bond length, (b) bond angle, (c) dihedral angle. Results from linear interpolation for (d) bond length, (e) bond angle, (f) dihedral angle.

Table 2: Comparison of maximum mean values of changes in bond lengths, bond angles, dihedral angles and consecutive C_α distances

#	Protein Name	Bond change ($\times 10^{-3}$Å)		Angle change (°)		Dihedral angle change (°)		Consecutive C_α distance change ($\times 10^{-3}$Å)	
		ARAP	Linear	ARAP	Linear	ARAP	Linear	ARAP	Linear
1	5'-Nucleotidase	9	124	1.4	4.7	6.4	6.4	19	274
2	Adenylate Kinase	16	92	2.7	5.0	14.4	14.4	46	97
3	Alcohol Dehydrogenase	23	81	3.6	5.8	19.8	19.8	53	53
4	Calmodulin	4	71	1.3	2.9	7.2	7.2	7	82
5	Collagenase	18	118	3.5	6.0	14.5	14.5	57	188
6	Dengue 2 Virus Envelope Glycoprotein	11	141	2.9	6.9	18.0	18.0	27	208
7	Dihydrofolate Reductase	23	64	2.4	4.7	9.7	9.7	57	72
8	Diphtheria Toxin	19	216	3.0	10.5	11.1	11.1	40	483
9	DNA Polymerase	6	57	1.6	3.7	10.8	10.8	14	30
10	Pyrophosphokinase	12	80	2.5	5.0	10.0	10.0	22	81
11	Pyruvate Phosphate Dikinase	15	93	1.4	3.6	7.8	7.8	32	227
12	Spindle Assembly Checkpoint Protein	23	200	4.8	10.5	22.8	22.8	46	352

3.2 C_α distance

Most coarse-grain methods use distances among C_α atoms for morphing. In [10], these distances are interpolated between the initial and target structures, whereas in [22] those between consecutive C_α atoms in intermediate structures are constrained to the range [3.7, 3.9] Å. Figure 6a shows how consecutive C_α distances evolve along the path for 5'-Nucleotidase using the proposed method. Although the value range area is within [3.556, 3.822] Å, the deviation area indicates that most values lie in the range [3.775, 3.86] Å. This shows that the ARAP interpolation method tends to preserve consecutive C_α distances as well. The linear interpolation method does not preserve well this quantity as shown inFigure 6b.

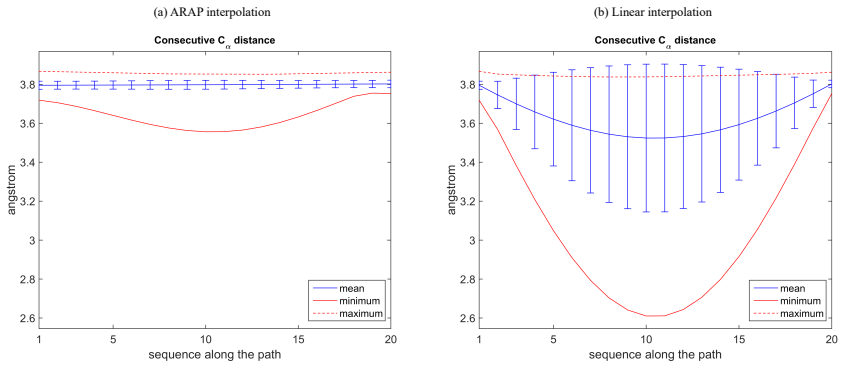

Fig. 6: Statistics of Consecutive C_α distances for 5'-Nucleotidase from (a) ARAP interpolation, (b) Linear interpolation.

As Table 2 indicates, this behavior is also observed for all benchmarks. Similar to bond lengths and bond angles, the changes in consecutive C_α distances are always lower in the ARAP method than those in the linear method.

3.3 Motion of structures along the path

The closing motions of Adenylate Kinase and Calmodulin are captured by the ARAP method in Figure 7.

Figure 8a and 8b show the shear motions of one part of Alcohol Dehydrogenase and Dihydrofolate Reductase, respectively. This type of motion is also observed in Pyrophosphokinase (Figure 8c). A slight change in the loops (green and light blue) on the left is caused by a shear motion of the green helix and light-blue loop.

In Dengue 2 Virus Envelope Glycoprotein (Figure 10d), a closing motion of a hinged domain (red, orange and yellow) and a rotation of about 180° of the small light blue loop are captured.

Our proposed method agrees with [26] for the motion of Diphtheria Toxin, *i.e.* a rotation of about 180° of a large domain (see Figure 10b). Note that this motion was not found by the Yale Morph Server [24].

In Figure 10a, our method shows a rotation of about 90° of one part of 5'-Nucleotidase. Figure 10c (DNA Polymerase) shows the closing motion of a group of green helices and slight rotations of other domains (red and yellow).

For Pyruvate Phosphate Dikinase (Figure 10e), one observes an opening motion carried out by two domains. One domain rotates by 90° (yellow and orange) while another rotates less than 90° (in green and purple).

Some weaknesses of the proposed method are shown in the case of Collagenase and Spindle Assembly Checkpoint Protein. Steric clashes are observed during the formation of the dark blue helix loops for both cases (Figure 9). This is due to the chosen Slerp method for interpolating rotations, which restricts the rotation angle to the range $[0, \pi]$ while the helix formation requires more than this. In the case of Spindle Assembly Checkpoint Protein, one also observes clashes between the red strand and other strands of the beta sheet. This is because the ARAP method does not guarantee the absence of steric clashes between distant (non-bonded) parts of the structure.

4 Discussion

4.1 ARAP energy and dihedral angles

One classical way to represent molecular systems is through the use of dihedral angles (see e.g. [57]). Hence, one can represent the backbone of a protein in terms of ϕ, ψ, ω angles and the mobility of the side chains in terms of dihedral angles χ. This internal coordinate representation is commonly used, since it is known that bond lengths and bond angles vary only slightly at room temperature whereas major conformational rearrangements are often due to dihedral angle variations [58].

There is an interesting connection between the internal coordinates representation and the ARAP energy definition. If a one-ring topology is used

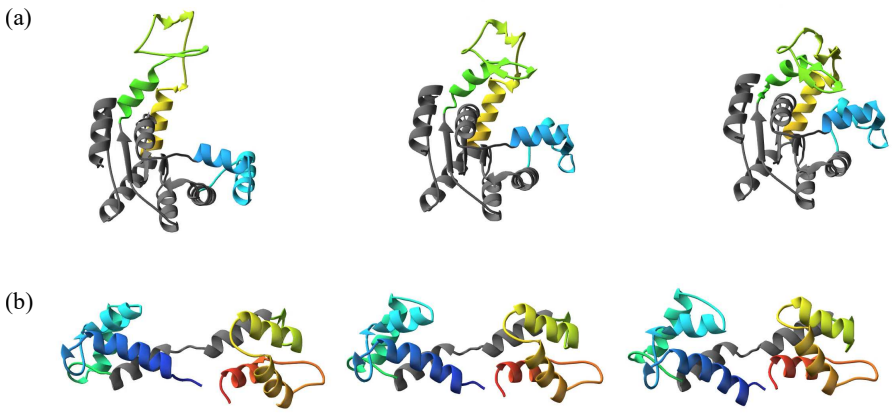

Fig. 7: Open-to-close motions in the colored parts of (a) Adenylate Kinase. (b) Calmodulin.

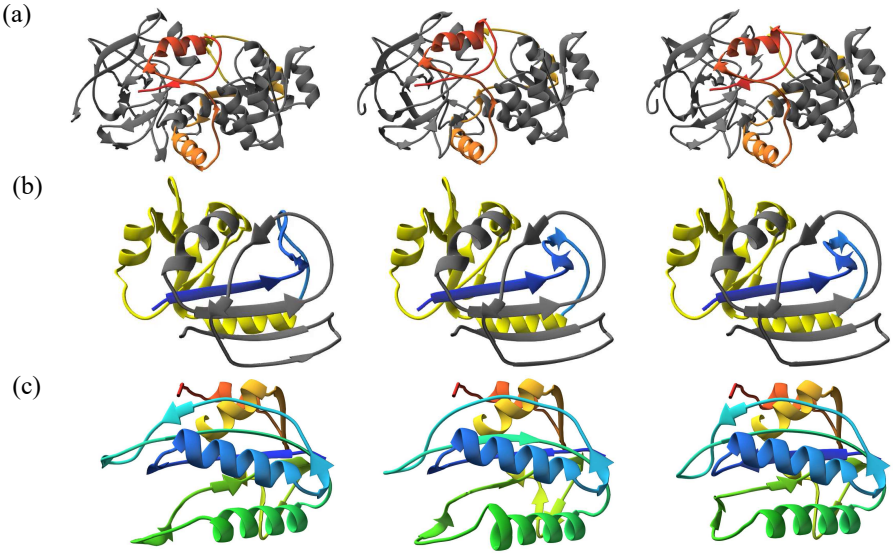

Fig. 8: Shear motions. (a) Alcohol Dehydrogenase: shear motion of the colored part compared to the static grey part. (b) Dihydrofolate Reductase: shear motion of the blue strand of the beta sheet. (c) Pyrophosphokinase: shear motion of the green helix and the light blue loops.

for ARAP cell construction, any move in internal coordinates is just a combination of rigid rotations of cells, which results in zero ARAP energy. On the other hand, conformations produced by ARAP interpolation may not result from changes in internal coordinates only. This is because our solution

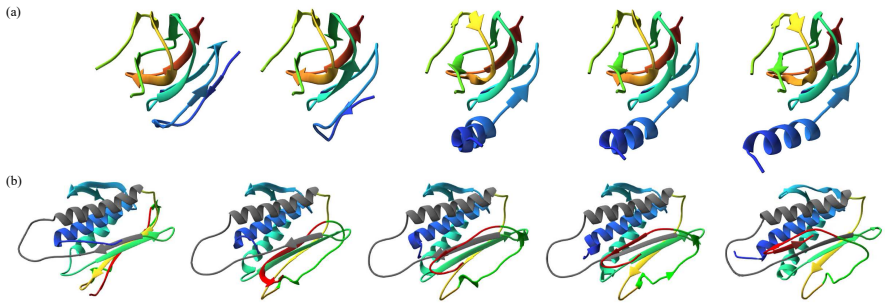

Fig. 9: (a) Collagenase: Steric clashes occur in the formation of the dark blue helix loop. (b) Spindle Assembly Checkpoint Protein: steric clashes occur in the formation of the dark blue helix and the motion of the red loop.

minimizes the ARAP energy, but may not zero it along the path. Therefore, conformations produced by ARAP interpolation may be close to conformations produced by changes in internal coordinates, but may also include changes in bond lengths and bond angles. This helps ARAP interpolation overcome two obstacles faced by internal coordinates moves: the impossibility to reach the target conformation due to fixed bond lengths and bond angles, as well as the loop closure problem [59].

4.2 Mesh quality

In our methodology, ARAP edges are "derived" from covalent bonds in molecular systems (through the recursive *connect* procedure). Hence, the ARAP topology constructed during the preprocessing phase does not form a mesh composed of triangles or tetrahedrons, as is typically the case in Computer Graphics. Despite the simple graph structure that we use, the result section has shown that the method works efficiently and produces paths with adequate geometric properties.

4.3 Large rotations

One limitation of the proposed methodology is that the amplitudes of local rotations are smaller than π. In our case, the initial structural alignment performed during the preprocessing phase alleviates this problem to some degree. However, this may not be sufficient to represent helix formation, as shown for Collagenase and the Spindle Assembly Checkpoint Protein. In 2D Computer Graphics, this issue can be resolved by correcting rotation angles so that the absolute angle differences between adjacent triangles are lower than π [31]. Unfortunately, this cannot be directly applied in 3D because the rotation axes

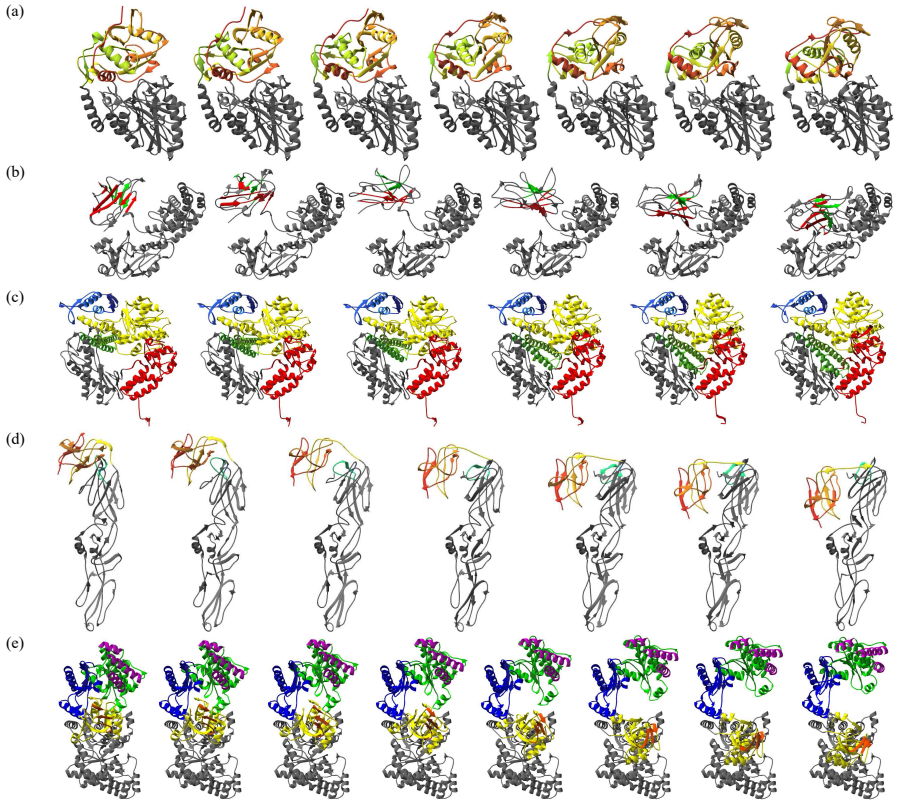

Fig. 10: Motion of (a) 5'-Nucleotidase: Rotation of about 90^o of the colored part can be detected by paying attention to the red helix. (b) Diphtheria Toxin: Rotation of about 180^o of the colored part can be detected by observing the beta sheets, *i.e.* the red sheet goes behind the green one in the final picture compared with the first. (c) DNA Polymerase: Closing motion of the green helices. (d) Dengue 2 Virus Envelope Glycoprotein: The closing motion of the red and yellow domains by rotation. The small light blue loop also rotates by 180^o. (e) Pyruvate Phosphate Dikinase: The opening motion accomplished by the green-purple domain and the yellow-orange domain. Rotation of 90^o of the yellow domain can be seen clearly by the orientation of the orange arrows. Rotation of the green-purple domain is smaller.

of adjacent triangles do not align. In the future, we would like to implement recent approaches to address this problem [60].

4.4 Symmetry

The method presented so far is not symmetric, *i.e.* the result of an interpolation from the target to the initial structure would differ from that of the reverse

direction. However, this property may be interesting for some applications. An easy way to tackle this limitation is to perform ARAP interpolation from both directions and take the average of the results. However, this solution would double the computational cost while in many cases, the result from one direction may be close to the average produced by a bi-directional solution. Hence, such a symmetric version of ARAP has not been implemented yet.

5 Conclusion

We have presented a new morphing method able to generate interpolation paths between two given molecular structures. This method relies on the ARAP paradigm used in Computer Graphics to edit complex meshes while preserving local characteristics of the structure. Despite being a purely geometrical approach, the proposed method generates interpolation paths at a very low computational cost, while preserving bond lengths and bond angles very well. Although the presented method has some limitations discussed above, we believe ARAP interpolation may find numerous uses, and we would like to investigate several extensions.

Obviously, ARAP interpolations may be complemented by an energy minimization step to locally improve the quality of the intermediate conformations (although this might significantly increase the cost of producing a path). In problems requiring more complex paths, one could imagine more sophisticated methods to locally improve the quality of ARAP interpolations. For example, the ARAP method could be combined with Steered MD [61] in order to find in which directions steered MD forces would be applied. ARAP interpolations could also be used to start off umbrella sampling for free energy calculations [62].

There are also many directions we would like to explore in order to extend the ARAP methodology to the field of structural biology. One of them is to improve ARAP interpolations by introducing more edges in the ARAP topology, based on for *e.g.* other pairwise interactions such as hydrogen bonds or dipolar interactions. Another possibility is to reflect chemical properties of molecules in the ARAP edges, by adjusting their weights according to, for *e.g.* known equilibrium bond lengths. We would also like to allow for large internal rotations thanks to recent ARAP improvements such as [60]. Furthermore, the ARAP interpolation computation is highly parallelizable, since intermediate conformations can be computed independently from one another, and we plan to take advantage of this property to improve the efficiency of our implementation.

6 Supplementary Material

Section 3.1 and Section 3.2 presented the results on bond length, bond angle, dihedral angle and consecutive alpha carbon distances for only the first case in

the benchmark list. The same results for all the cases and video clips of their motions can be found in the supplementary material.

References

1. H.M. Berman, T. Battistuz, T.N. Bhat, W.F. Bluhm, P.E. Bourne, K. Burkhardt, Z. Feng, G.L. Gilliland, L. Iype, S. Jain, P. Fagan, J. Marvin, D. Padilla, V. Ravichandran, B. Schneider, N. Thanki, H. Weissig, J.D. Westbrook, C. Zardecki, Acta Crystallographica Section D **58**, 899 (2002)
2. F. Allen, Acta Crystallogr. Sect. B-Struct. Sci. **58**, 380 (2002)
3. J. Yon, D. Perahia, C. Ghelis, Biochimie **80**(1), 33 (1998)
4. D. Kern, E.R. Zuiderweg, Current opinion in structural biology **13**(6), 748 (2003)
5. D. Frenkel, B. Smit, *Understanding molecular simulation: from algorithms to applications*, vol. 1 (Academic press, 2001)
6. J. Haile, Computers in Physics **7**(6), 625 (1993)
7. H. Lei, Y. Duan, Current opinion in structural biology **17**(2), 187 (2007)
8. R.C. Bernardi, M.C. Melo, K. Schulten, Biochimica et Biophysica Acta (BBA)-General Subjects **1850**(5), 872 (2015)
9. C. Dellago, P.G. Bolhuis, (2008)
10. M.K. Kim, R.L. Jernigan, G.S. Chirikjian, Biophys J **83** (2002). DOI 10.1016/S0006-3495(02)73931-3. URL http://dx.doi.org/10.1016/S0006-3495(02)73931-3
11. M.K. Kim, G.S. Chirikjian, R.L. Jernigan, J Mol Graph Model **21** (2002). DOI 10.1016/S1093-3263(02)00143-2. URL http://dx.doi.org/10.1016/S1093-3263(02)00143-2
12. A. Ahmed, H. Gohlke, Proteins **63** (2006). DOI 10.1002/prot.20907. URL http://dx.doi.org/10.1002/prot.20907
13. L. Yang, G. Song, R.L. Jernigan, Biophys J **93** (2007). DOI 10.1529/biophysj.106.095927. URL http://dx.doi.org/10.1529/biophysj.106.095927
14. K.C. Chou, Biophysical chemistry **30**(1), 3 (1988)
15. G. Henkelman, B.P. Uberuaga, H. Jónsson, The Journal of chemical physics **113**(22), 9901 (2000)
16. E. Weinan, W. Ren, E. Vanden-Eijnden, Physical Review B **66**(5), 052301 (2002)
17. N.M. Amato, G. Song, J Comput Biol **9** (2002). DOI 10.1089/10665270252935395. URL http://dx.doi.org/10.1089/10665270252935395
18. M.S. Apaydin, D.L. Brutlag, C. Guestrin, D. Hsu, J.C. Latombe, C. Varma, J Comput Biol **10** (2003). DOI 10.1089/10665270360688011. URL http://dx.doi.org/10.1089/10665270360688011
19. B. Raveh, A. Enosh, O. Schueler-Furma, D. Halperin, PLoS Comput Biol **5** (2009). DOI 10.1371/journal.pcbi.1000295. URL http://dx.doi.org/10.1371/journal.pcbi.1000295
20. L. Jaillet, F.J. Corcho, J.J. Pérez, J. Cortés, Journal of computational chemistry **32**(16), 3464 (2011)
21. Morphs page on proteopedia http://proteopedia.org/wiki/index.php/morphs. URL http://proteopedia.org/wiki/index.php/Morphs
22. N.E. Castellana, A. Lushnikov, P. Rotkiewicz, N. Sefcovic, P.A. Pevzner, A. Godzik, K. Vyatkina, Algorithms for Molecular Biology **8**(1), 1 (2013). DOI 10.1186/1748-7188-8-19. URL http://dx.doi.org/10.1186/1748-7188-8-19
23. D.R. Weiss, M. Levitt, Journal of molecular biology **385**(2), 665 (2009)
24. W.G. Krebs, M. Gerstein, Nucleic Acids Res **28** (2000). DOI 10.1093/nar/28.8.1665. URL http://dx.doi.org/10.1093/nar/28.8.1665
25. S. Wells, S. Menor, B. Hespenheide, M. Thorpe, Physical Biology **2**(4), S127 (2005)
26. D.W. Farrell, K. Speranskiy, M. Thorpe, Proteins: Structure, Function, and Bioinformatics **78**(14), 2908 (2010)
27. M. Alexa, D. Cohen-Or, D. Levin, in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (ACM Press/Addison-Wesley Publishing Co., 2000), pp. 157–164
28. J. Choi, A. Szymczak, (2003)

29. T. Igarashi, T. Moscovich, J.F. Hughes, ACM transactions on Graphics (TOG) **24**(3), 1134 (2005)
30. H. Guo, X. Fu, F. Chen, H. Yang, Y. Wang, H. Li, Journal of Visual Communication and Image Representation **19**(4), 245 (2008)
31. W. Baxter, P. Barla, K.i. Anjyo, in *Proceedings of the 6th international symposium on Non-photorealistic animation and rendering* (ACM, 2008), pp. 59–64
32. O. Sorkine, M. Alexa, in *Symposium on Geometry processing*, vol. 4 (2007), vol. 4
33. A. Cuno, C. Esperança, A. Oliveira, P.R. Cavalcanti, in *Proceedings of the 27th computer graphics international conference* (2007), pp. 115–122
34. P. Borosán, R. Howard, S. Zhang, A. Nealen, in *Proc. of Eurographics*, vol. 2010 (2010), vol. 2010
35. M. Zollhöfer, E. Sert, G. Greiner, J. Süßmuth, in *Eurographics (Short Papers)* (2012), pp. 85–88
36. I. Chao, U. Pinkall, P. Sanan, P. Schröder, in *ACM Transactions on Graphics (TOG)*, vol. 29 (ACM, 2010), vol. 29, p. 38
37. Z. Levi, C. Gotsman, Visualization and Computer Graphics, IEEE Transactions on **21**(2), 264 (2015)
38. Y.S. Liu, H.B. Yan, R.R. Martin, Journal of Computer Science and Technology **26**(3), 548 (2011)
39. M. Kilian, N.J. Mitra, H. Pottmann, ACM Transactions on Graphics (TOG) **26**(3), 64 (2007)
40. Z. Zhang, G. Li, H. Lu, Y. Ouyang, M. Yin, C. Xian, Computers & Graphics **46**, 244 (2015)
41. L. Liu, L. Zhang, Y. Xu, C. Gotsman, S.J. Gortler, in *Computer Graphics Forum*, vol. 27 (Wiley Online Library, 2008), vol. 27, pp. 1495–1504
42. H.B. Yan, S.M. Hu, R.R. Martin, Journal of Computer Science and Technology **22**(1), 147 (2007)
43. J. Huang, Y. Tong, K. Zhou, H. Bao, M. Desbrun, Visualization and Computer Graphics, IEEE Transactions on **17**(7), 983 (2011)
44. O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, H.P. Seidel, in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing* (ACM, 2004), pp. 175–184
45. Y. Lipman, O. Sorkine, D. Levin, D. Cohen-Or, ACM Trans. Graph. **24**(3), 479 (2005). DOI 10.1145/1073204.1073217. URL http://doi.acm.org/10.1145/1073204.1073217
46. D. Xu, H. Zhang, Q. Wang, H. Bao, Graphical Models **68**(3), 268 (2006)
47. M. Botsch, M. Pauly, M.H. Gross, L. Kobbelt, in *Symposium on Geometry Processing* (2006), EPFL-CONF-149310, pp. 11–20
48. N. Paries, P. Degener, R. Klein, in *Computer Graphics and Applications, 2007. PG'07. 15th Pacific Conference on* (IEEE, 2007), pp. 461–464
49. M. Botsch, O. Sorkine, Visualization and Computer Graphics, IEEE Transactions on **14**(1), 213 (2008)
50. R.C. Edgar, BMC bioinformatics **5**(1), 1 (2004)
51. P. Liu, D.K. Agrafiotis, D.L. Theobald, Journal of computational chemistry **31**(7), 1561 (2010)
52. B.K. Horn, JOSA A **4**(4), 629 (1987)
53. S.K. Kearsley, Acta Crystallographica Section A: Foundations of Crystallography **45**(2), 208 (1989)
54. K. Shoemake, in *ACM SIGGRAPH computer graphics*, vol. 19 (ACM, 1985), vol. 19, pp. 245–254
55. Samson software for adaptive modeling and simulation of nanosystems https://www.samson-connect.net/. URL https://www.samson-connect.net/
56. G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org (2010)
57. M. Zhang, L.E. Kavraki, Journal of Chemical Information and Computer Sciences **42**(1), 64 (2002)
58. T. Schlick, *Molecular modeling and simulation: an interdisciplinary guide: an interdisciplinary guide*, vol. 21 (Springer Science & Business Media, 2010)
59. A. Shehu, L.E. Kavraki, Entropy **14**(2), 252 (2012)
60. S. Kaji, in *Mathematical Progress in Expressive Image Synthesis III* (Springer, 2016), pp. 7–19

61. J.C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R.D. Skeel, L. Kale, K. Schulten, Journal of computational chemistry **26**(16), 1781 (2005)
62. G.M. Torrie, J.P. Valleau, Journal of Computational Physics **23**(2), 187 (1977)