

Research Article

Resolution-Free Accurate DNA Contour Length Estimation from Atomic Force Microscopy Images

Peter I. Chang and Ming-Chi Hsaio

Mechanical Engineering, National Taiwan University of Science and Technology, Taiwan

Correspondence should be addressed to Peter I. Chang; itchang@mail.ntust.edu.tw

Received 27 April 2018; Revised 28 January 2019; Accepted 27 February 2019; Published 9 June 2019

Academic Editor: Daniele Passeri

Copyright © 2019 Peter I. Chang and Ming-Chi Hsaio. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This research presented an accurate and efficient contour length estimation method developed for DNA digital curves acquired from Atomic Force Microscopy (AFM) images. This automation method is calibrated against different AFM resolutions and ideal to be extended to all different kinds of biopolymer samples, encompassing all different sample stiffnesses. The methodology considers the digital curve local geometric relationship, as these digital shape segments and pixel connections represent the actual morphology of the biopolymer sample as it is being imaged from the AFM scanning. In order to incorporate the true local geometry relationship that is embedded in the continuous form of the original sample, one needs to find this geometry counterpart in the digitized image. This counterpart is realized by taking the skeleton backbone of the sample contour and by using these digitized pixels’ connection relationship to find its local shape representation. In this research, one uses the 8-connect Freeman Chain Code (CC) to describe the directional connection between DNA image pixels, in order to account for the local shapes of four connected pixels. The result is a novel shape number (SN) system derived from CC, which is a fully automated algorithm that can be applied to DNA samples of any length for accurate estimation, with efficient computational cost. This shape-wise consideration is weighted to modify the local length with great precision, accounting for all the different morphologies of the biopolymer sample, and resulted with accurate length estimation, as the error falls below 0.07%, an order of magnitude improvement compared to previous findings.

1. Introduction

The Atomic Force Microscopy (AFM) system has the ability to probe samples at the nanometer scale, owing to its ability in sensing the sample surface to resolve force interaction at the pico-Newton level [1]. This feature makes AFM systems a useful imaging device in the field of nanotechnology, molecular biology, and many others. It is well known in AFM’s biological application to image biopolymers thanks to its ability to image in liquid, the biopolymer’s natural environment [2].

One very interesting characteristic is the length of a single DNA strand, denoted as $l_c$. This contour length can be applied to identify genome editing results and other application outcomes [3]. And accuracy in getting $l_c$ correct is essential at this scale, as there is small room for error in genome editing, since one base-pair distance for DNA is only 0.34 nm. Thus, AFM images of DNA samples provide means for such $l_c$ studies on accurate DNA length estimation, like the image illustrated here in Figure 1.

There are two ways in finding $l_c$ from AFM images. One is by manual fitting, and the other is by automatic skeleton tracing with image processing. Fitting typically relies on human operators picking specific positions along the DNA contour by examination on the acquired image, which relies on the trained eye of a scholar to map out the contour length $l_c$ [5, 6], as illustrated in Figure 2.

On the other hand, the automatic $l_c$ estimation traces the DNA image along its backbone skeleton. This is done by thinning the strand image to its median position from the...
to modify the coordinate location by considering the three consecutive points with \( X_p = k(x_{p-1} - x_p) + x_p + k(x_{p+1} - x_p) \). This length estimator \( L_p \) calculates \( l_p \) according to the modified \( X_p \) [11].

Another \( l_c \) estimator is designed specifically for DNA strand samples, named \( L_{DNA} \). This estimator introduced a nominal coefficient for different DNA lengths and is defined as \( L_{DNA} = rC_f(n_c + \sqrt{2n_f}) \), where \( C_f \) is inversely calculated from simulated \( l_c \) data, so a table of \( C_f \) helps \( L_{DNA} \) to match the expected value of \( l_c \) [12].

More recently, a machine learning approach utilized a feature extraction to fit different cubic spline segment occurrences with the following: horizontal, vertical, diagonal, perpendicular, varying height and thickness, as defined by \{ \( n_{hor} \), \( n_{vert} \), \( n_{diag} \), \( n_{perp} \), \( n_{thick} \), \( n_{tkcv} \) \} [13]. This machine learning estimator \( L_{ML} \) is trained to generate coefficients considering the abovementioned feature from known DNA \( l_c \).

A summary table in Table 1 provides a quick review of the abovementioned \( l_c \) estimators.

In this paper, the authors propose an estimator based on the DNA imaged contour shape, thus having the name Shape estimator \( L_S \), where \( L_S \) is designed to be robust to image resolution and only uses minimal computational resource. This is achieved by considering the neighboring shape of the original two-pixel connection inspired from \( L_p \), but as all the DNA local morphology shapes are considered for estimating \( l_c \), the resultant accuracy is shown to improve by more than an order of magnitude.

Detailed methodology of the \( L_S \) estimator is explained in Section 2, starting from the general image preprocessing to the identification of twelve local 4-pixel segment configuration shapes. Then, the 12-shape correction coefficients \( k_1 - k_{12} \) are calibrated in Section 3, with different resolutions considered. Finally, the \( l_c \) values for \( L_S \) are compared with \( L_{DNA} \) and \( L_F \) in Section 4.

## 2. Contour Length Estimation with Local Shape Consideration

\( L_S \) estimation essentially takes into account the local shape considerations. As two neighboring pixels are connected together in this AFM image, the overall shape around the two connected pixels represents different local lengths as this DNA morphology is observed. In a tight turn; i.e., a “kink,” this local length will certainly be longer then a smooth linear local profile.

Thus, \( L_S \) considers the two additional pixels extending from the center two-pixel connection and identifies the different 4-pixel segmented shapes surrounding along the DNA skeleton backbone. Then, \( L_S \) makes shape-corrected length adjustments, by multiplying the local shape’s corresponding coefficient to adjust for the estimated \( l_c \). It can be observed that the extension of this segmented elemental shape is not limited to 4 pixels, as with more pixels such as a 5-pixel segment can also be considered. However, due to the trade-off for computational cost and performance, this research investigates the \( L_S \) with 4-pixel elements.
2.1. Pixel Resolution and Image Preprocessing. A standard preprocess extracts the DNA image into the l’s skeleton backbone, by thinning the DNA strand into the centerline of the biopolymer. This research’s automatic image process is illustrated here in Figure 5.

First, the DNA image is prefiltered and mapped into a binary image with thresholding. Then, further, 2-D filters remove isolated pixel islands, ensuring that a single DNA contour is captured. And finally, an iterative debranch thinning morphology is applied to find the skeleton backbone that can be chain-coded [14].

It is well known that AFM systems have a tip broadening effect when imaging, which expands the DNA strand width to a larger value. A repeated thinning preprocess in average converges the single-width pixel contour, towards the mid-point of the DNA strand automatically, given an AFM image with enough resolution across the DNA width [15].
2.2. Identification of 4-Pixel Segment Shape Connectivity. Given the resultant single-width pixels \( P = \{ p_i, i = 0 \sim n \} \) for the contour’s skeleton backbone, its CC \((C = c_i, i = 1 \sim n)\) is coded from one end to the other. Note that this research utilizes the 8-connect chain code, resulting in integers ranging from 0 \sim 7\ for all \( c_i \) and that \( c_i \) is one off from \( p_i \), as there are \( n \) connections between \( n + 1 \) pixels.

With the 4-pixel segment setup, there are up to a total of 64 ways \((4^4)\) to connect the 4 pixels into single-pixel width arrangements. This research paper has fully outlined all the possibilities, and the full table of all 64 different single-width 4-pixel segments is arranged in Figures 6 and 7. They are arranged by the assigned \( k_1 \sim k_{12} \) types, with all the same types grouped together.

It is clear that all the same types of \( k_j \) shape are grouped with the 4-pixel segment’s mirror and rotational images. Take for example the \( k_8 \) shape, where the segment is rotated clockwise/counterclockwise for 90 degrees individually and mirrored on the y-axis, shown here in Figure 8.

Having these \( k_1 \sim k_7 \) segment shapes distinguished, the original inner 2-pixel connection’s distance can now be corrected, by considering the outward extended 4-pixel segment shape. This would take into account the local geometric features according to its categorized shape. Since the skeleton backbone is composed of consecutive 4-pixel segments all along its contour, when tracing from one end to the other, this research makes sure that the \( L_k \) estimator identifies every 4-pixel segment to the twelve unique \( k_j \) shapes, as shown in Figure 9.

2.2.1. Chain Code, Shape Number, and Identifier. In order to identify a skeleton backbone’s different 4-pixel segment shapes along the contour, this research utilizes its chain code, formed as a series of integer number, and developed a novel algorithm called the shape number (SN) identification, labeled \( S \), and uses it to derive an exclusive identifier (ID) number for matching the abovementioned unique \( k_j \sim k_{12} \) shapes.

A typical CC collection, \( C = \{ c_1c_2c_3 \ldots c_{n-2}c_{n-1}c_n \} \), is a series of integers made from 0 \sim 7, provided the \( n + 1 \) single-width skeleton backbone pixels \( P = \{ p_0p_1p_2 \ldots p_{n-1}p_n \} \). Note that \( C \) is one-off from \( P \) and that \( p_S \) is numbered from 0 \sim n. This research emphasizes the general ability to distinguish any skeleton backbone, and while for any given backbone, it creates a set of two distinct CC for every skeleton, due to starting the connection from different ends of the pixel chain. The algorithm will demonstrate the ability to converge on the distinguished 4-pixel segment shapes.

![Figure 4](image-url) (a) 8-connect chain code connectivity from the center pixel, illustrating the connection code 0 \sim 7 associated according to the adjacent pixel location. (b) An example of a single-pixel width biopolymer skeleton backbone used for deriving the chain code. Starting from head to tail codes: 07107565 and tail to head codes: 12134534.

<table>
<thead>
<tr>
<th>Author</th>
<th>Year</th>
<th>Estimator</th>
<th>Algorithm</th>
<th>Specific to DNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spisz et al. [8]</td>
<td>1998</td>
<td>Freeman estimator</td>
<td>( L_E = r(n_e + \sqrt{2n_o}) )</td>
<td></td>
</tr>
<tr>
<td>Rivetti and Codeluppi [9]</td>
<td>2001</td>
<td>Kupla ( L_K ) and corner ( L_C ) estimator</td>
<td>( L_K = r(0.948n_e + 1.343n_o), L_C = r(0.980n_e + 1.406n_o - 0.091n_o) )</td>
<td>✓</td>
</tr>
<tr>
<td>Sanchez-Sevilla et al. [10]</td>
<td>2002</td>
<td>Curvature filtering</td>
<td>Coordinate transform to complex plane with Fourier transform</td>
<td></td>
</tr>
<tr>
<td>Ficarra et al. [11]</td>
<td>2005</td>
<td>Pixel coordinate</td>
<td>Relocation of pixel representative coordinate ( X )</td>
<td></td>
</tr>
<tr>
<td>Rivetti [12]</td>
<td>2009</td>
<td>DNA estimator ( L_{DNA} )</td>
<td>( L_{DNA} = rC_f(n_e + \sqrt{2n_o}), C_f = (l)/L_E )</td>
<td>✓</td>
</tr>
<tr>
<td>Sundstorm et al. [13]</td>
<td>2012</td>
<td>Machine learning</td>
<td>( L_{ML} = \sum { nhz, nvh, n_{diag}, n_{perp}, n_{horz}, n_{vert} } )</td>
<td>✓</td>
</tr>
</tbody>
</table>
As the algorithm needs to continuously identify the 4-pixel segments throughout the contour backbone, a rolling window starts from any end of \( C \) and collects the following \( n \)-segments (Figure 10)

\[
\text{Seg} = \{\text{seg}_1, \text{seg}_2, \ldots, \text{seg}_i, \ldots, \text{seg}_n\}.
\]  

It is clear that each \( \text{seg}_i \) segment is comprised of three consecutive chain codes \( c_{i-1}, c_i, c_{i+1} \), since a 4-pixel segment consists of three connections. With the exception of the first and last segments of the contour skeleton, where there are not enough pixels to form a 4-pixel segment, thus the algorithm just takes the original two connecting pixels, i.e., the original \( c_1 \) or \( c_n \). The pixel/geometric representation of a rolling window CC segmentation is demonstrated in the Figure 11.

Notice that the rolling window in \( C \) moves the 4-pixel segment consecutively from Head to Tail, and each segment can be coded as \( \text{seg} = \{c_{i-1}, c_i, c_{i+1}\} \), from \( i = 2 \sim n - 1 \), excluding Head \( (i = 1) \) and Tail \( (i = n) \).

This research now defines a shape number (SN), derived from each of the rolling segments as

\[
S_{\text{seq}} = \{s_{\text{seg}_2} s_{\text{seg}_3} s_{\text{seg}_4} \ldots s_{\text{seg}_{n-3}} s_{\text{seg}_{n-2}} s_{\text{seg}_{n-1}}\}.
\]  

In short, \( S = \{s_i, i = 2 \sim n - 1\} \) is a collection of the ordered cyclic difference from each segment’s continuous 3 chain codes. Thus, for each segment, SN is composed of three integer numbers as

\[
S_{\text{seg}} = \begin{cases} 
  s_{i-1} = c_{i-1} - c_{i+1}, \\
  s_i = c_i - c_{i-1}, \\
  s_{i+1} = c_{i+1} - c_i,
\end{cases}
\]

Since all CC is comprised of integers from \( 0 \sim 7 \), SNs \( s_i \) are also retained between \( 0 \sim 7 \). Thus, whenever \( s_i \) is derived as negative, we automatically take 8’s compliment to correct it, with \( s_i = s_i + 8 \) if \( s_i < 0 \).

One such example of SN derived is illustrated in the lower part of Figure 11, where the SN is calculated from both directions of the chain code inside each segment: the Start to End Chain Code (SECC) as well as the End to Start Chain Code (ESCC). It is obvious that SECC and ESCC are different; therefore, the resulting Start to End Shape Number (SESN) and End to Start Shape Number (ESSN) are also derived different, albeit representing the exact same segment.

To ensure exclusive identification on the same 4-pixel segment, for both bidirectional CC and SN coding, in addition to all the same shape mirroring and rotational configuration, a simple unique identifier (ID) number is needed to match the rolling 4-pixel segments to the \( k_1 \sim k_{12} \) shapes.

2.2.2. Unique Identifier (ID) Matching. In order to deal with such bidirectional, mirroring, and rotational segment ambiguity, the following rule has been applied to ensure a single SN identifier (ID) to match explicitly one \( k_1 \sim k_{12} \).
shape, for any given random shape number in the lengthy \( l \) contour skeleton backbone.

This is provided by examining all the SESN and ESSN for all the \( k_1 \sim k_{12} \) shapes and capturing the combinatory relative adjacent arrangements from the 4-pixel segment geometry, i.e., reordering the representative numerals of SN to allow for the direct/diagonal connections, to make representation of the given shape.

Since the rolling window segmental SN \( s_{seg,i} \) will fall into the recognizable \( k_1 \sim k_{12} \) shapes, the ID number can be derived from the known segment numbers as specified in Figure 9. Thus, the identifier is a unique number for each of the shape \( k_j, j = 1 \sim 12 \), such that when the rolling window covers a 4-pixel segment, by performing this numeral operation (algorithm), one will find the identifier.

The following Algorithm (1) outlines the unique identifier (ID)’s reorder methodology for all of the \( k_1 \sim k_{12} \) shapes.

The rules for the identifier are stated as follows:

- **Unique**—there exists one unique ID number for each of \( k_5, k_8, k_9, \) and \( k_{12} \) shapes.
- **Common**—there exists common ID numbers for the following pairs: \( \{ k_1, k_2 \}, \{ k_3, k_4 \}, \{ k_6, k_7 \}, \) and \( \{ k_{10}, k_{11} \} \).
- **Distinguish**—the aforementioned sets are discerned by the connection type of the center pixels, [direct or diagonal], by checking its original CC number \( c_i \), with even/odd numbers representing direct/diagonal, respectively.

**Figure 6:** Four-pixel segments of the shapes for \( k_1 \sim k_7 \), with their associated rotation and mirror alternatives.
Take for example the $k_8$SN, as there exist four different SN combinations: 116, 772, 727, and 161 form shaping of the eight different segments, as shown in the last row of Figure 8, due to coding of all the different mirrors/rotations and bidirectional CC.

After performing the abovementioned ID algorithm, we are able to uniquely transform all SN to the same identifier (ID) number: 116, as shown from Table 2.

Finally, the algorithm arrives with $k_5$, $k_8$, $k_9$, and $k_{12}$ matching up with their respective ID numbers: 026, 116,
In addition, the algorithm matches the pairs \(k_1, k_2\), \(k_3, k_4\), \(k_5, k_6\), \(k_7, k_8\), \(k_9, k_{10}\), \(k_{11}, k_{12}\) commonly to \(000, 017, 107, 277\), respectively. In order to distinguish between the pairs, the original \{direct, diagonal\} connection is once again used: by checking if the original \(c_i\) is either even or odd, then it can be trivially matched to the correct shape in the pair Table 3.

All the ID numbers are listed for \(k_1 \sim k_{12}\) shapes here in Figure 9, derived from all the 64 shapes in Figures 6 and 7. Note that the common ID numbers are annotated with (-even/-odd) for distinguishing.

2.3. Parameter Equation Representation. Now that the unique ID number is obtained, it is then ready to amass a collection of the different samples of a given \(l_c\) length, in order to retrieve the correction parameter for the different \(k_j\)s, provided with the same DNA characteristics, i.e., with a fixed \(l_p = 50\) nm.

2.3.1. Length Calculation with Coefficients. This is first done by identification on one individual DNA sample’s contour, by summing up each shape component \(k_j\)’s occurrence contribution for its segment’s connection length. In other words, one identifies along the skeleton backbone and tallies the individual occurrences of the twelve \(k_j\)s, multiplied by the corresponding connection length (1 or \(\sqrt{2}\) pixel length) along with its correction coefficient. This makes the sum of all the length contributions equal to the contour length \(l_c\) as

\[
l_c = r \sum_{j=1}^{12} n_{kj} k_j + (l_{1H} + l_T)
\]

where \(n_{kj}, j = 1 \sim 12\) is the number of occurrences of the type \(k_j\) shapes, provided from the identification along the skeleton backbone. \(k_j\) is the correction coefficient, and \(l_{kj}\) is the connection length (either 1 or \(\sqrt{2}\) according to the \(k_j\) shape). \(l_{1H}\) and \(l_T\) are the head and tail length, respectively, and finally \(r\) is the AFM image pixel resolution.

From Figure 9, \(l_k \sim l_{k_{12}}\) length is ordered in Table 4.

**Figure 9:** Geometric connectivity of the twelve independent \(k_1 \sim k_{12}\) shapes for 4-pixel segments. The inner two direct connections (horizontal/vertical and diagonal) are connected with two additional outward pixels, accounting for the local 2D morphology change.
Figure 11: Consecutive determination of the 4-pixel segment shape number through the rolling window. Two sets of chain code provided HTCC (head to tail) and THCC (tail to head), with their associated local segment CC, also bidirectional: SECC (start to end) and ESCC (end to start), which leads to their derived SESN (start to end) and ESSN (end to start).

**Algorithm 1:** ID number derived from shape number.

```plaintext
Require: $k_j$ shapes, $j = 1,\ldots,12$ and their respective SESN or ESSN $s_{kj} = s_1s_2s_3$
Ensure: Unique ID numbers for all $k_1 \sim k_{12}$ shapes.
1: for $j = 1 \sim 12$ do
2: take the $k_j$'s SN: $s = \{s_1, s_2, s_3\}$
3: if $s_1 \geq 4$ then
4: take 8's complement for this sequence of $s$, for bi-direction and mirroring re-ordering, but
5: if $s_2 = 0$ or $s_3 = 0$ then
6: keep $s_2$ or $s_3$ zero
7: else
8: $s_2 = 8 - s_2$ and $s_3 = 8 - s_3$
9: end if
10: end if
11: Take the new $S$ (with $s_1 \leq 4$), and further re-order $s_2$ and $s_3$, to match the rotational varieties, by
12: if $s_3 > s_3$ then
13: switch $s_2$ with $s_3$
14: else
15: retain original order
16: end if
17: Find final ID number for given SN $s_{kj}$
18: end for
```
with the same class of biopolymers twist/turn, resulting in the same length contribution for the lm for any given backbone skeleton length. Combining equation (4) with the associated lkj, = \sum_{m,k} l_{m,k}\delta_{lj},\quad \forall j = 1,2,...,m_c.

\( l_{m,k}\) values in Table 4, we arrive at

\[ l_{m,k} = r\left[\begin{array}{c} n_{m,k_1} k_1(1) + n_{m,k_2} k_2(\sqrt{2}) + n_{m,k_3} k_3(1) \\
+ n_{m,k_4} k_4(\sqrt{2}) + n_{m,k_5} k_5(\sqrt{2}) + n_{m,k_6} k_6(1) \\
+ n_{m,k_7} k_7(\sqrt{2}) + n_{m,k_8} k_8(\sqrt{2}) + n_{m,k_9} k_9(\sqrt{2}) \\
+ n_{m,k_{10}} k_{10}(1) + n_{m,k_{11}} k_{11}(\sqrt{2}) + n_{m,k_{12}} k_{12}(\sqrt{2}) \\
+ (l_{m,1} + l_{m,2})\end{array}\right], \tag{5}\]

for any given backbone skeleton length \( l_{m,0}\) given the index \( m\)'th sample. Equation (5) can be represented with a matrix form, with

\[ r(NDK + B) = L, \tag{6}\]

where

\[ N = \begin{bmatrix} n_{1,k_1} & n_{2,k_2} & \cdots & n_{1,k_{12}} \\
& n_{2,k_1} & n_{2,k_2} & \cdots & n_{2,k_{12}} \\
& \vdots & \ddots & \vdots & \vdots \\
& n_{m,k_1} & n_{m,k_2} & \cdots & n_{m,k_{12}} \end{bmatrix}, \tag{7}\]

\[ K = \begin{bmatrix} k_1 \\
k_2 \\
\vdots \\
k_{12} \end{bmatrix}, \tag{8}\]

\[ B = \begin{bmatrix} b_1 \\
b_2 \\
\vdots \\
b_m \end{bmatrix}, \tag{9}\]

\[ L = \begin{bmatrix} l_{c,1} \\
l_{c,2} \\
\vdots \\
l_{c,m} \end{bmatrix} \]
Note that $b_m = l_{hm} + l_{mm}$ is the head and tail (boundary) connection length summation. Also note that $N$ is an $m$-by-$12$ matrix, $K$ is 12 by 1, and $B$ and $L$ are both $m$-by-1 matrices.

The final procedure here is to derive matrix $K$ using a standard linear regression and find the best fit for the $k_1 \sim k_{12}$ value. The final results are presented in the next section.

### 3. Contour Parameter Calibration Result

In order to guarantee convergences of the $k_1 - k_{12}$ coefficients, different known values of $l_c$ and $r$ single-pixel-width AFM images were simulated for $k_j$ calibration. Due to the combinations of different $l_c$ and $r$, plus a surplus amount of samples for each $(l_c, r)$ pair, a total of 58,800,000 images were generated. All the simulated images are based on DNA characteristics, as mentioned in Introduction, where all the samples have the same persistence length of $l_p = 50$ nm.

The different lengths calculated ranged from 340 to 1020 nm, for every 34 nm, and the different resolution $r$ is simulated between 5.1 and 7.8 nm/pixel, with a 0.1 nm interval. Thus, there are 21 different $l_c$ scales, along with 28 altering $r$, making a total of $21 \times 28 = 588$ test cases. Each case is studied with 10,000 DNA images, for sufficient representation on $k_j$s. In order words, the test index $m = 10,000$ was used for equation (6).

#### 3.1. Convergence of the $k_j$ Coefficient

This research first checks the convergence of all $k_j$ coefficients, given a growing number of image files, i.e., growing number of $m$ in equation (6). The results are illustrated in Figure 12.

All coefficients verify its convergence when given more than 0.5 million samples and remain constant with fluctuation of less than 0.01% after 1 million samples. This result is verified for all resolutions $r = 5.1 - 7.6$ nm/pixel, showing similar trend for all $k_j$s.

#### 3.2. Linear Variation of $k_j$ Dependence on Resolution $r$

With the convergence for all the $k_j$ coefficients confirmed for all different $r$, the relationship for each $k_j(r)$ as a function of $r$, i.e., the linear fit for $k_j(r)$ results, is found in Figure 13.

It is clear that using the converged $k_j$ values for a specified $r$, the following linear fit equation,

$$k(r) = mr + b,$$

results with a table that contributes to all the twelve different coefficients; it is provided in Table 5.
To demonstrate such performance, the coefficient in equation (10) is used for the shape estimator $L_S$ and compared alongside the DNA estimator $L_{DNA}$ and the Freeman estimator $L_F$.

All estimators $L_S$, $L_{DNA}$, and $L_F$ are compared with different $l_c$ and resolution $r$. All the estimators are applied for the same simulated pixel images and compared against the readily known $l_c$ for error calculation.

Tables 6, 7, and 8 outline the calculation error for different $r$ settings. It shows that the $L_S$ estimator has an averaged relative error maxed at 0.07%, performing with an order of magnitude difference from the $L_{DNA}$ estimator, and two orders of magnitude smaller than $L_F$. The relative error translates to an absolute value of maximum 0.20 nm for the $r = 5.1$ nm/pixel, well below the resolution, making $L_S$ ideal for $l_c$ estimation.

Since the error is averaged amongst the 100,000 samples provided, its standard deviation (STD) in nm is also an indicator for quantitative analysis. The $L_S$ estimator also has a smaller standard deviation compared to both $L_{DNA}$ and $L_F$, against a growing $l_c$ contour estimated.

### 4. Conclusion and Future Direction

This research provided a novel way to estimate digitized contour length, in a general way that is applicable towards all kinds of contour curvature. Utilizing a localized shape

<table>
<thead>
<tr>
<th>$k(r)$</th>
<th>$m$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1$</td>
<td>0.0030698</td>
<td>1.0096</td>
</tr>
<tr>
<td>$k_2$</td>
<td>0.0026365</td>
<td>1.0084</td>
</tr>
<tr>
<td>$k_3$</td>
<td>0.0025729</td>
<td>1.0052</td>
</tr>
<tr>
<td>$k_4$</td>
<td>-0.0037566</td>
<td>0.88691</td>
</tr>
<tr>
<td>$k_5$</td>
<td>-0.00027806</td>
<td>0.7044</td>
</tr>
<tr>
<td>$k_6$</td>
<td>0.010117</td>
<td>0.93777</td>
</tr>
<tr>
<td>$k_7$</td>
<td>0.0045822</td>
<td>0.92897</td>
</tr>
<tr>
<td>$k_8$</td>
<td>-0.0018511</td>
<td>0.79905</td>
</tr>
<tr>
<td>$k_9$</td>
<td>0.013175</td>
<td>0.86184</td>
</tr>
<tr>
<td>$k_{10}$</td>
<td>0.032808</td>
<td>0.7165</td>
</tr>
<tr>
<td>$k_{11}$</td>
<td>0.015803</td>
<td>0.79656</td>
</tr>
<tr>
<td>$k_{12}$</td>
<td>0.042707</td>
<td>0.70788</td>
</tr>
</tbody>
</table>

### 3.3. Performance with Shape Modification Coefficient

The above sections result in the calibration correction in equation (10) and can be used for the unknown $l_c$ estimation. In order to demonstrate such performance, the coefficient in equation (10) is used for the shape estimator $L_S$ and compared alongside the DNA estimator $L_{DNA}$ and the Freeman estimator $L_F$.

All estimators $L_S$, $L_{DNA}$, and $L_F$ are compared with different $l_c$ and resolution $r$. All the estimators are applied for the same simulated pixel images and compared against the readily known $l_c$ for error calculation.

Tables 6, 7, and 8 outline the calculation error for different $r$ settings. It shows that the $L_S$ estimator has an averaged relative error maxed at 0.07%, performing with an order of magnitude difference from the $L_{DNA}$ estimator, and two orders of magnitude smaller than $L_F$. The relative error translates to an absolute value of maximum 0.20 nm for the $r = 5.1$ nm/pixel, well below the resolution, making $L_S$ ideal for $l_c$ estimation.

Since the error is averaged amongst the 100,000 samples provided, its standard deviation (STD) in nm is also an indicator for quantitative analysis. The $L_S$ estimator also has a smaller standard deviation compared to both $L_{DNA}$ and $L_F$, against a growing $l_c$ contour estimated.
connection approach, and correct upon the local connectivity between pixels, this algorithm accounts for both resolution and the sample stiffness.

This research is general in the local 4-pixel segment identification method and extensible towards extension to more pixel elements. The general idea stands that a single-width pixel contour’s digital shape recognition is applicable towards all images acquired from different systems, not only with the AFM family but also optical microscopy systems, electron microscopy systems, and many others.

Experimental verification is also needed for future research, provided with calibrated accurate sample length $l_c$ from DNA samples or other biopolymer samples imaged with AFM systems.

### Data Availability

The data used to support the findings of this study are included within the article.

### Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

### Acknowledgments

The authors would like to thank the funding provided from the Ministry of Science and Technology, Taiwan. The research is supported through grant number MOST 105-2221-E-011-056.

### References


