Supplementary of Polynomial Cauchy Coordinates for Curved Cages

ZHEHUI LIN, University of Science and Technology of China, China
RENJIE CHEN*, University of Science and Technology of China, China

ACM Reference Format:
Zhehui Lin and Renjie Chen*. 2024. Supplementary of Polynomial Cauchy Coordinates for Curved Cages. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 3–6, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3680528.3687654

1 INTRODUCTION

This document serves as supplementary material for the paper "Polynomial Cauchy Coordinates for Curved Cages", providing a comprehensive derivation of our novel coordinate system and addressing computational optimizations for high-degree Bézier curves.

2 DERIVATION OF POLYNOMIAL CAUCHY COORDINATES

Building upon the foundational concepts presented in the main paper's background section, including Cauchy's integral formula and Bézier curve representation, we delve into the mathematical intricacies of our coordinate system. Our analysis incorporates advanced mathematical tools such as binomial theorem expansion and complex integration.

2.1 Our Coordinates

We consider a scenario where the source cage is a polygonal structure represented by a set of closed, counterclockwise oriented line segments. The target cage is a curved structure defined by a Bézier spline. Both cages have M edges or curve segments, respectively. Our goal is to derive a discrete form of Cauchy's integral formula and apply it to each Bézier control point, thereby obtaining the expression for our barycentric coordinates.

The core derivation process for our coordinates has been presented in the main paper. Here, we provide the final expression for our coordinates and offer a detailed derivation of the Integral function:

$$C_{j,m}(z) = \frac{1}{2\pi i}\,\mathrm{Integral}(z, e_j, m, n_j). \tag{1}$$

$$u(z) = \sum_{e_j \in \partial D} \sum_{m=0}^{n_j} C_{j,m}(z)\mathbf{b_{j,m}} \tag{2}$$

Authors' Contact Information: Zhehui Lin, University of Science and Technology of China, Hefei, Anhui, China, linzhehui626@ustc.edu.cn; Renjie Chen*, University of Science and Technology of China, Hefei, Anhui, China, renjiec@ustc.edu.cn.

SA Conference Papers '24, December 3–6, 2024, Tokyo, Japan

ACM ISBN 979-8-4007-1131-2/24/12
https://doi.org/10.1145/3680528.3687654

$$
\begin{aligned}
&\mathrm{Integral}(z, e_j, m, N) \\
&= \int_{e_j} \frac{\binom{N}{m}\left(\frac{w-z_{j-1}}{z_j-z_{j-1}}\right)^m\left(\frac{z_j-w}{z_j-z_{j-1}}\right)^{N-m}}{w-z} dw \\
&= \frac{\binom{N}{m}}{(A_j)^N} \int_{e_j} \frac{(w-z_{j-1})^m (z_j-w)^{N-m}}{w-z} dw \\
&= \frac{\binom{N}{m}}{(A_j)^N} \int_{e_j} \frac{\Big((w-z)-B_{j-1}\Big)^m \Big(B_j-(w-z)\Big)^{N-m}}{w-z} dw \\
&= \frac{\binom{N}{m}}{(A_j)^N} \int_{e_j} \frac{\Big(\sum_{k=0}^{m}\binom{m}{k}(w-z)^k(-B_{j-1})^{m-k}\Big)\Big(\sum_{l=0}^{N-m}\binom{N-m}{l}(B_j)^l\big(-(w-z)\big)^{N-m-l}\Big)}{w-z} dw \\
&= \frac{\binom{N}{m}}{(A_j)^N} \int_{e_j} \sum_{k=0}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}(w-z)^{N-m-l+k-1}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l dw \\
&= \frac{\binom{N}{m}}{(A_j)^N}\left[(-B_{j-1})^m(B_j)^{N-m}\log\frac{B_j}{B_{j-1}} + \sum_{\substack{k=0 \\ N-m-l+k\neq 0}}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}\frac{(w-z)^{N-m-l+k}}{N-m-l+k}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l\Bigg|_{z_{j-1}}^{z_j}\right] \\
&= \frac{\binom{N}{m}}{(A_j)^N}\left[(-B_{j-1})^m(B_j)^{N-m}\log\frac{B_j}{B_{j-1}} + \sum_{\substack{k=0 \\ N-m-l+k\neq 0}}^{m}\sum_{l=0}^{N-m}\frac{\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}}{N-m-l+k}\Big((B_j)^{N-m+k}(B_{j-1})^{m-k}-(B_j)^l(B_{j-1})^{N-l}\Big)\right]
\end{aligned}
$$

The closed-form expression for our coordinates, denoted as $C_{j,m}(z)$, is evaluated for each edge e_j and assigned to the corresponding Bézier control point $\mathbf{b_{j,m}}$. Algorithm 1 details the computation of the `Integral` function.

Algorithm 1: Calculation of $\mathrm{Integral}(z, e_j, m, N)$

Input: Interior point z, edge e_j, index of Bézier control points m, degree of current Bézier curve N
Output: result as $2\pi i \cdot C_{j,m}(z)$
result = 0;
for *k = 0:m* **do**
  for *l = 0:N-m* **do**
    if *N-m-l+k == 0* **then**
      result += $(-B_{j-1})^m(B_j)^{N-m}\log\frac{B_j}{B_{j-1}}$;
    else
      result += $\frac{\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}}{N-m-l+k}\Big((B_j)^{N-m+k}(B_{j-1})^{m-k}-(B_j)^l(B_{j-1})^{N-l}\Big)$;
    end
  end
end
result ×= $\binom{N}{m}/(A_j)^N$;

2.2 Derivatives of our coordinates

For derivative computation, we leverage the infinite differentiability of the holomorphic function u within the interior of D. Utilizing Cauchy's integral formula, we express the derivatives of our coordinates as integral expressions, enabling the derivation of n-th order derivatives.

We first present the first-order derivative $D_{j,m}(z)$:

$$u'(z) = \frac{1}{2\pi i}\int_{\partial D}\frac{f(w)}{(w-z)^2}dw,$$

Since it is a polynomial integration, the derivation is similar to that of the barycentric coordinates. Coefficient $D_{j,m}(z)$ for $\mathbf{b_{j,m}}$ is given by:

$$\begin{aligned}
\frac{1}{2\pi i}\int_{e_j}\frac{B_m^N(\frac{w-z_{j-1}}{z_j-z_{j-1}})}{(w-z)^2}dw =& \frac{1}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\int_{e_j}\frac{\big((w-z)-B_{j-1}\big)^m\big(B_j-(w-z)\big)^{N-m}}{(w-z)^2}dw \\
=& \frac{1}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\int_{e_j}\sum_{k=0}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}(w-z)^{N-m-l+k-2}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l dw \\
=& \frac{1}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\Bigg[-(B_j)^{N-m}(-B_{j-1})^m\Big(\frac{N-m}{B_j}+\frac{m}{B_{j-1}}\Big)\log\frac{B_j}{B_{j-1}}+ \\
& \sum_{\substack{k=0 \\ N-m-l+k\neq 1}}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}\frac{(w-z)^{N-m-l+k-1}}{N-m-l+k-1}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l\Bigg|_{z_{j-1}}^{z_j}\Bigg] \\
=& \frac{1}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\Bigg[-(B_j)^{N-m}(-B_{j-1})^m\Big(\frac{N-m}{B_j}+\frac{m}{B_{j-1}}\Big)\log\frac{B_j}{B_{j-1}}+ \\
& \sum_{\substack{k=0 \\ N-m-l+k\neq 1}}^{m}\sum_{l=0}^{N-m}\frac{\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}}{N-m-l+k-1}\Big((B_j)^{N-m+k-1}(B_{j-1})^{m-k}-(B_j)^l(B_{j-1})^{N-l-1}\Big)\Bigg].
\end{aligned}$$

The n-th order derivative $u^{(n)}(z)$ and coefficients $C^{(n)}_{j,m}(z)$ for $\mathbf{b_{j,m}}$ are given by:

$$u^{(n)}(z) = \frac{n!}{2\pi i}\int_{\partial D}\frac{f(w)}{(w-z)^{n+1}}dw \tag{3}$$

$$C^{(n)}_{j,m}(z) = \frac{n!}{2\pi i}\int_{e_j}\frac{B_m^N(\frac{w-z_{j-1}}{z_j-z_{j-1}})}{(w-z)^{n+1}}dw. \tag{4}$$

$$
\begin{aligned}
&\frac{n!}{2\pi i}\int_{e_j}\frac{B_m^N(\frac{w-z_{j-1}}{z_j-z_{j-1}})}{(w-z)^{n+1}}dw\\
=&\frac{n!}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\int_{e_j}\frac{\big((w-z)-B_{j-1}\big)^m\big(B_j-(w-z)\big)^{N-m}}{(w-z)^{n+1}}dw\\
=&\frac{n!}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\int_{e_j}\sum_{k=0}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}(w-z)^{N-m-l+k-(n+1)}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l dw\\
=&\frac{n!}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\Bigg[\sum_{\substack{k=0\\N-m-l+k=n}}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l\log(w-z)\Bigg|_{z_{j-1}}^{z_j}+\\
&\sum_{\substack{k=0\\N-m-l+k\neq n}}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}\frac{(w-z)^{N-m-l+k-n}}{N-m-l+k-n}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l\Bigg|_{z_{j-1}}^{z_j}\Bigg]\\
=&\frac{n!}{2\pi i}\frac{\binom{N}{m}}{(A_j)^N}\Bigg[\sum_{\substack{k=0\\N-m-l+k=n}}^{m}\sum_{l=0}^{N-m}\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}(B_{j-1})^{m-k}(B_j)^l\log\frac{B_j}{B_{j-1}}+\\
&\sum_{\substack{k=0\\N-m-l+k\neq n}}^{m}\sum_{l=0}^{N-m}\frac{\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}}{N-m-l+k-n}\Big((B_j)^{N-m+k-n}(B_{j-1})^{m-k}-(B_j)^l(B_{j-1})^{N-l-n}\Big)\Bigg]
\end{aligned}
$$

The pseudocode for computing the n-th order derivatives is provided in Algorithm 2. Note that this algorithm can also handle the case of $n=0$, i.e. the barycentric coordinates.

Algorithm 2: Calculation of n-th derivative of our coordinates

Input: Interior point z, edge e_j, index of Bézier control points m, degree of current Bézier curve N, derivative order n
Output: result as $C_{j,m}^{(n)}(z)$
result = 0;
for k = $0{:}m$ **do**
  for l = $0{:}N\text{-}m$ **do**
    if $N\text{-}m\text{-}l\text{+}k == n$ **then**
      result += $\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}(B_j)^l(B_{j-1})^{m-k}\log\frac{B_j}{B_{j-1}}$;
    else
      result += $\frac{\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}}{N-m-l+k-n}\Big((B_j)^{N-m+k-n}(B_{j-1})^{m-k}-(B_j)^l(B_{j-1})^{N-l-n}\Big)$;
    end
  end
  result $\times$= n! $\binom{N}{m}/(A_j)^N/(2\pi i)$;
end

3 ACCELERATION

The above expressions contain two nested summations, as indicated by the Σ symbol. If the Bézier curves is of relatively low order and N is small, the summations will not incur significant overhead, and direct evaluation is sufficient. However, in certain applications with large N, the computational complexity can grow unpredictably. To address this, we will accelerate the computation process to ensure that the time complexity for evaluation each $C_{j,m}^{(n)}(z)$ does not exceed $O(N)$.

3.1 Accelerated Computation for Our Coordinates

We develop an $O(N^2)$ algorithm to compute weights for Bézier control points along a curve segment, allowing an average computation time of $O(N)$ per point. By precomputing combinatorial numbers and $(B_j)^m(B_{j-1})^{N-m}$ terms, we reduce the summation term evaluation to $O(1)$ complexity.

Further optimization is achieved by separating the double summation and precomputing two summations, $T1$ and $T2$:

$$
\begin{aligned}
T1[N-m,k] &= \sum_{l=0}^{N-m} \frac{\binom{N-m}{l}(-1)^{-l}}{N-m-l+k} \qquad k=0,1,...,m \\
T2[m,N-l] &= \sum_{k=0}^{m} \frac{\binom{m}{k}(-1)^{-k}}{N-m-l+k} \qquad l=0,1,...,N-m \\
Mult[m] &= (B_j)^m(B_{j-1})^{N-m} \qquad m=0,1,...,N \\
\sum_{k=0}^{m}\sum_{\substack{l=0 \\ N-m-l+k\neq 0}}^{N-m} ... &= \sum_{k=0}^{m}\binom{m}{k}(-1)^{N-k}T1[N-m,k]Mult[N-m+k] - \sum_{l=0}^{N-m}\binom{N-m}{l}(-1)^{N-l}T2[m,N-l]Mult[l]
\end{aligned}
$$

Matrices $T1$ and $T2$ can be recursively calculated like the combinatorial number:

$$
\begin{aligned}
T1[N-m+1,k] &= \sum_{l=0}^{N-m+1} \frac{\binom{N-m+1}{l}(-1)^{-l}}{N-m+1-l+k} \\
&= \sum_{l=0}^{N-m} \frac{\frac{N-m+1}{N-m+1-l}\binom{N-m}{l}(-1)^{-l}}{N-m+1-l+k} + \frac{(-1)^{-(N-m+1)}}{k} \\
&= \sum_{l=0}^{N-m} \frac{N-m+1}{k}\left(\frac{\binom{N-m}{l}(-1)^{-l}}{N-m+1-l} - \frac{\binom{N-m}{l}(-1)^{-l}}{N-m+1-l+k}\right) + \frac{(-1)^{-(N-m+1)}}{k} \\
&= \frac{N-m+1}{k}\Big(T1[N-m,1] - T1[N-m,k+1]\Big) + \frac{(-1)^{-(N-m+1)}}{k}
\end{aligned}
$$

$$\begin{aligned}
T2[m+1,N-l] &= \sum_{k=0}^{m+1} \frac{\binom{m+1}{k}(-1)^{-k}}{N-m-1-l+k} \\
&= \sum_{k=0}^{m} \frac{\frac{m+1}{m+1-k}\binom{m}{k}(-1)^{-k}}{N-m-1-l+k} + \frac{(-1)^{-(m+1)}}{N-l} \\
&= \sum_{k=0}^{m} \frac{m+1}{N-l}\left(\frac{\binom{m}{k}(-1)^{-k}}{m+1-k} + \frac{\binom{m}{k}(-1)^{-k}}{N-m-1-l+k}\right) + \frac{(-1)^{-(m+1)}}{N-l} \\
&= \frac{m+1}{N-l}\Big(T1[m,1] + T2[m,N-l-1]\Big) + \frac{(-1)^{-(m+1)}}{N-l}
\end{aligned}$$

By appropriately changing the indices, we derive the following recursive formula, which can be used for computation. The total time complexity is $O(N^2)$.

$$\begin{aligned}
T1[N-m,0] &= \sum_{l=0}^{N-m-1} \frac{\binom{N-m}{l}(-1)^{-l}}{N-m-l} \\
T1[N-m,k] &= \frac{N-m}{k}\Big(T1[N-m-1,1] - T1[N-m-1,k+1]\Big) + \frac{(-1)^{-(N-m)}}{k} \qquad k = 1,\ldots,m \\
T2[m,m] &= \sum_{k=1}^{m} \frac{\binom{m}{k}(-1)^{-k}}{k} \\
T2[m,N-l] &= \frac{m}{N-l}\Big(T1[m-1,1] + T2[m-1,N-l-1]\Big) + \frac{(-1)^{-m}}{N-l} \qquad l = 0,1,\ldots,N-m-1
\end{aligned}$$

This approach transforms the computation into an $O(N^2)$ precomputation step and an $O(N)$ per-point coordinate calculation, significantly reducing the complexity from the original $O(N^3)$ per point and $O(N^4)$ per Bézier curve. Algorithm. 3 presents the accelerated method for computing $C_{j,m}(z)$.

Algorithm 3: Accelerated Calculation of our coordinates

Input: Interior point z, edge e_j, index of Bézier control points m, degree of current Bézier curve N, derivative order n
Output: result as $C_{j,m}(z)$
Load precalculated $T1$, $T2$ and combinatorial number $\binom{*}{*}$;
$Mult[0:N] = B_j^{(0:N)} \cdot B_{j-1}^{(N:0)}$;
result = $(-1)^m Mult[N-m] \log \frac{B_j}{B_{j-1}}$;
for k = $0{:}m$ **do**
  result += $\binom{m}{k}(-1)^{N-k} T1[N-m,k] Mult[N-m+k]$;
end
for l = $0{:}N\text{-}m$ **do**
  result -= $\binom{N-m}{l}(-1)^{N-l} T2[m,N-l] Mult[l]$;
end
result ×= $\binom{N}{m}/(A_j)^N/(2\pi i)$;

3.2 Accelerated Computation for the Derivatives of Our Coordinates

The computation for the derivatives of our coordinates can be accelerated similar to the previous section, with the following additional summations:

$$T1[N-m,-1] = \sum_{\substack{l=0 \\ l\neq N-m-1}}^{N-m} \frac{\binom{N-m}{l}(-1)^{-l}}{N-m-l-1} \qquad m = 0,1,...,N$$

$$T2[m,m-1] = \sum_{\substack{k=0 \\ k\neq 1}}^{m} \frac{\binom{m}{k}(-1)^{-k}}{k-1} \qquad m = 0,1,...,N$$

The direct calculation of these summation is also $O(N^2)$, and then the derivative of our coordinates can be computed in $O(N)$ time as well.

$$MultD[m] = (B_j)^m (B_{j-1})^{N-m-1} \qquad m = -1,0,1,...,N$$

$$\sum_{\substack{k=0 \\ N-m-l+k\neq 1}}^{m} \sum_{l=0}^{N-m} ... = \sum_{k=0}^{m} \binom{m}{k}(-1)^{N-k} T1[N-m,k-1]MultD[N-m+k-1] - \sum_{l=0}^{N-m} \binom{N-m}{l}(-1)^{N-l} T2[m,N-l-1]MultD[l]$$

The accelerated algorithm for $D_{j,m}(z)$ is provided in Algorithm. 4. The n-th order derivatives can be accelerated similar to that of the first-order derivative. With precomputation not exceeding $O(N^2)$, we can achieve the same efficiency for the n-th order derivatives. The matrices $T1$, $T2$, and the cage-related data can be precomputed, which can greatly improve the overall computational efficiency.

Algorithm 4: Accelerated Calculation of First-order Derivative of Our coordinates

Input: Interior point z, edge e_j, index of Bézier control points m, degree of current Bézier curve N, derivative order n
Output: result as $D_{j,m}(z)$
Load precalculated $T1$, $T2$ and combinatorial number $\binom{*}{*}$;
$MultD[-1:N] = B_j^{(-1:N)} \cdot B_{j-1}^{(N:-1)}$;
result = $(-1)^{m+1} MultD[N-m]\left((N-m)\frac{B_{j-1}}{B_j} + m\right) \log \frac{B_j}{B_{j-1}}$;
for k = 0:m **do**
  result += $\binom{m}{k}(-1)^{N-k} T1[N-m,k-1]MultD[N-m+k-1]$;
end
for l = 0:N-m **do**
  result -= $\binom{N-m}{l}(-1)^{N-l} T2[m,N-l-1]MultD[l]$;
end
result ×= $\binom{N}{m}/(A_j)^N/(2\pi i)$;