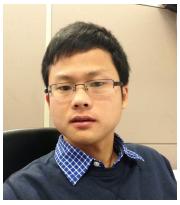
CS6491-2015 P3: Worm

Xiong Ding, Subhajit Das




1 Objective

Our objectives are the following:

1. Compute the median (average) curve between two input curves in a 3D space.
2. Animate a morph between two input curves
3. Compute and display the inflation - minimal tube tangent to both input curves

2 Definitions and input

Given two curves A and B both sharing common start and end points in a 3D space. In order to make the morph between these curves,we will compute the medial curve C by computing the medial points $M_1, M_2, M_3.....M_N$. Subsequently, we would make transverse curves in the form of a series of Arcs spaced at a combined distance of D on each input curve, considering the two input curves can accommodate a sphere or ball of radius r in between them. Here, D is the diameter of the biggest circle tangential to both curves A and B.

3 Approach

3.1 Definition of curves and the tangent vector

Both curves A and B are controlled by 5 control points $\{P_0, P_1, P_2, P_3, P_4\}$ individually to form a quintic Bézier curve, which has an explicit formula:

$$P(t) = (1-t)^4 P_0 + 4(1-t)^3 t P_1 + 6(1-t)^2 t^2 P_2 + 4(1-t)t^3 P_3 + t^4 P_4 \quad (1)$$

It's derivative is

$$P'(t) = 4(1-t)^3(P_1 - P_0) + 12(1-t)^2 t(P_2 - P_1) + 12(1-t)t^2(P_3 - P_2) + 4t^3(P_4 - P_3) \quad (2)$$

The curves are formed by computing the central points of the Ball with diameter D. The points on the curves are obtained by Linear Interpolation.

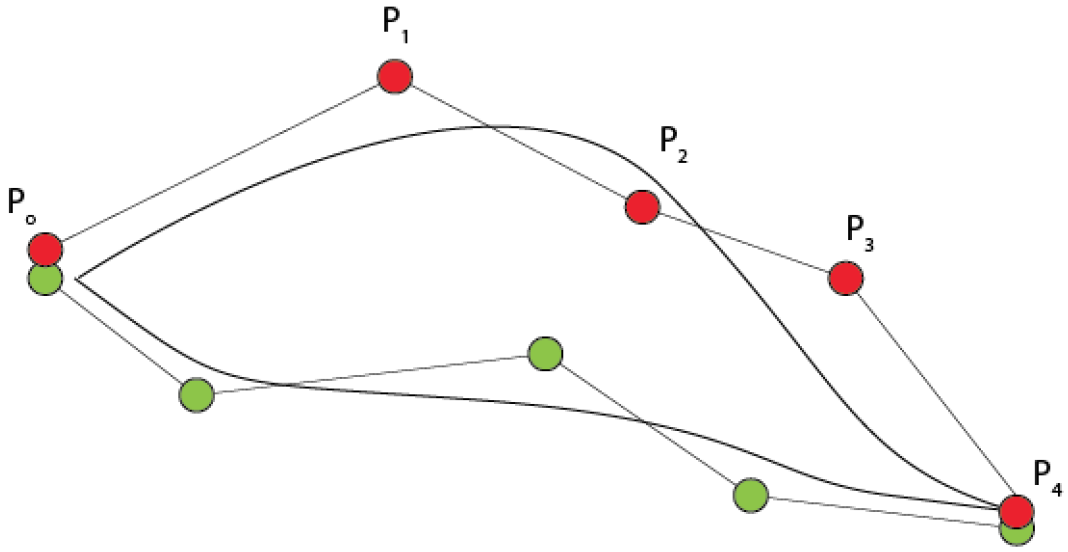


Figure 1: Shows two quintic bezier curve with 5 control points

3.2 Calculate the First Median Point on the Medial Curve

Let A_0 be the start point of both curves A and B and medial curve C. From the parametric equation given in 1 we found two points A_1 and B_1 on curves A and B respectively very close to start point A_0. We first calculate the Angle Bisector Vector BS by summing the vectors A and B and dividing by 2.

Now we need to compute the first median point M_0 on this Vector BS. That is given by the following equation,

$$M_0 = P(A_0, dis, BS) \tag{3}$$

where dis is the minimum value of $D/2$, magnitude($A1$) or magnitude($B1$). Here,D is the diameter of the fitting ball between input curves A and B, such that it is tangential to both the curves.

3.3 Calculate the next median Point by guessing

Calculate the tangent vectors $tanV_1$ and $tanV_2$ at points A_1 and B_1 respectively as below:

$$\begin{aligned} c_1 &= 4 * (1-t)^3 \\ c_2 &= 12 * (1-t)^2 * t \\ c_3 &= 12 * (1-t) * t^2 \\ c_4 &= t^3 \\ Vector\ \text{tanV}_1 &= (\text{c}_1\text{*v1}+c_2\text{*v2}) + (c_3\text{*v3} + c_4\text{*v4}) \end{aligned} \tag{4}$$

where, $v1, v2, v3$ *and* $v4$ are the vectors between the respective control points P_0, P_1, P_2, P_3, P_4.

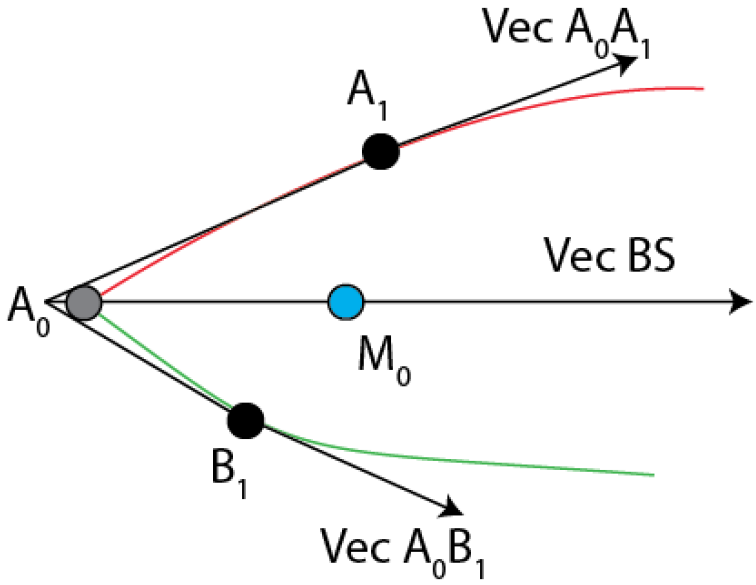


Figure 2: Shows two quintic bezier curve with 5 control points

Here, $tanV_1$ is the calculated tangent vector. We can compute $tanV_2$, the respective tangent vectors at point A_1 and B_1 using aforementioned principles.

We compute the Bisection between the tangent vectors $tanV_1$ and $tanV_2$ and find two points on the bisector by the following calculation:

First, we get the normalized normal Vector $\underline{N}$:

$$\begin{aligned} Vector\underline{\mathrm{N}} &= \mathrm{tanV}_1 \text{ x } tanV_2 \\ Vector\underline{\mathrm{NN}} &= \mathrm{N}/||\mathrm{N}|| \end{aligned} \tag{5}$$

We get the vector $A1B1$ between points A_1 and B_1. Also we compute the normalized perpendicular vector K by taking cross product between $tanV_1$ and vector N. This allows to compute the point $P1$ as below:

$$\begin{aligned} float \text{ val} &= dot(\mathrm{A}_1B_1,K)/\mathrm{dot}(tanV_1,K) \\ Point \text{ P1} &= P(\mathrm{B}_1,val,tanV_2) \end{aligned} \tag{6}$$

We do the same to find the other point $P2$.

We add the points $P1$ and $P2$ to find the middle point $P3$. Also we add the normalized $tanV_1$ vector and normalized $tanV_2$ vector to get vector V. This helps us find $P4$ as below:

$$Point \text{ P4} = P(\mathrm{P3,V}) \tag{7}$$

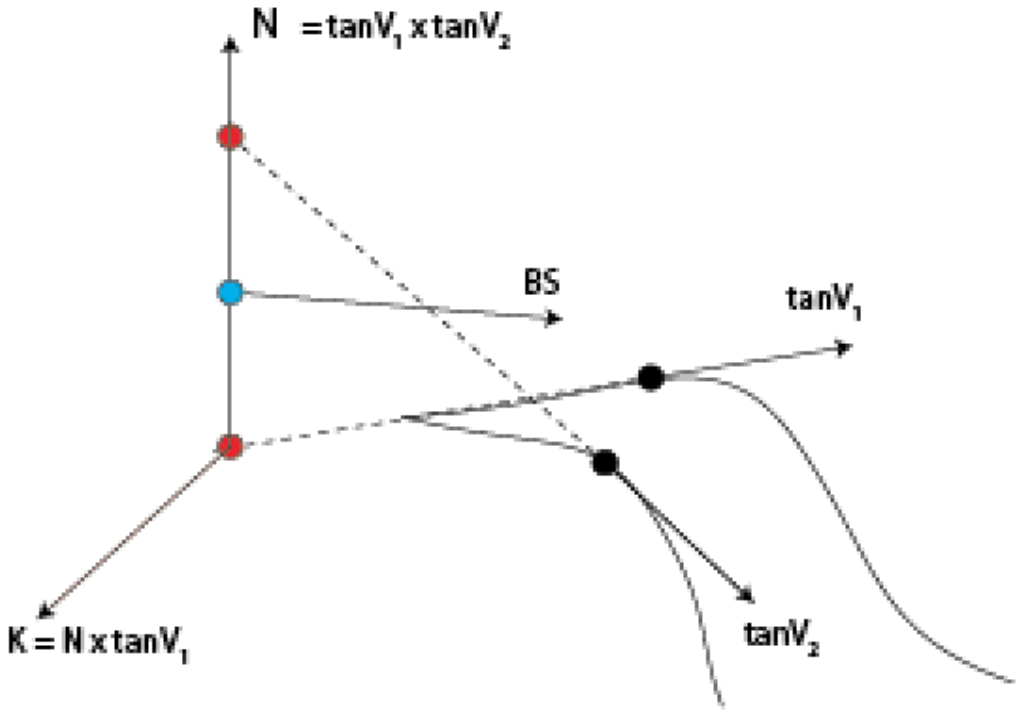


Figure 3: Shows the two bisection point obtained by computing the normal Vector N

Thus the final points we obtained from computing the bisection are $P3$ and $P4$. We use these points to compute the Vector t between them and thus we get the guessed Median Point MG_1 as below:

$$Point\ \mathrm{MG_1} = \mathrm{P}(M_0,D/2,\mathrm{t}) \tag{8}$$

3.4 Update the median point based on the Guess

Even though we have the guessed median point, we know that its not precise. To compute the new median point from the guessed median point, we first find the projection of point MG_1 on curve A and B respectively.

To find the projection we shoot rays to each of the center point of the balls of the curve A and find the ray with least distance $dis1$, to the point MG_1.

Then if it is not the first point on the curve on the curve, then we check the projection side of the point MG_1 to the line between center balls points $c[id-1]$ and $c[id]$. We check to find if its on the line. If it is on the line , then we re compute the projection of point MG_1 on the $line(c[id-1], c[id])$ and compute the new distance $dis2$.

Similarly if it is not the last point on the curve, then we check the projection side of the point MG_1 to the line between center balls points $c[id]$ and $c[id+1]$. We check to find if its on the line.

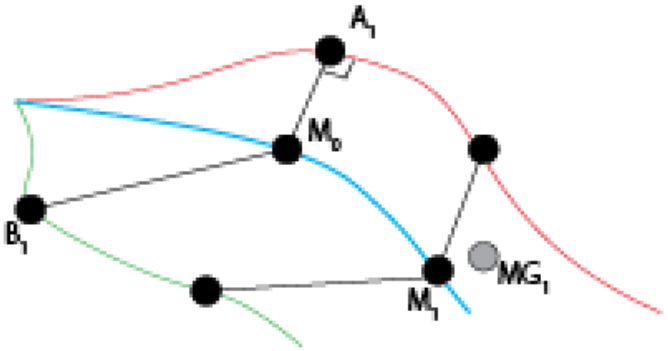


Figure 4: Shows the concept of parallel transport to find the next Median Point based on the current one

If it is on the line , then we re compute the projection of point MG_1 on the $line(c[id], c[id+1])$ and compute the new distance $dis3$.

We compare the three distances $dis1, dis2 dis3$ and get the projected point PA_1 which has the least value.We do the same process for curve B and find the projected point PB_1.

Likewise, we also calculate the tangent vectors $tanPA_1$ and $tanPB_1$ at points PA_1 and PB_1 respectively. At this point with the help of these two points and tangent vectors, we call the bisection method as explained above to compute the two bisected points $BB1$ and $BB2$.

We create a line between $BB1$ and $BB2$ and find the projection of MG_1 on this line which is basically the updated median point as below:

$$Point \text{ M}_1 = \text{project}(MG_1, BB1, BB2) \tag{9}$$

3.5 Calculate the Median Line

Computing the median lines needs computation of all the median points. We can get the First Median Point and the next Median Point as aforementioned. We iterate the computation part of guessing the next median and updating the same point with precision by feeding last computed median point as the first median point at each iteration. We keep a check such that the distance between the last median point and the end center point of the curve should be lesser than $D/2$.

$$\begin{aligned} &while(n(V(last.M, A.C[A.n-1])) > D/2 \quad i < 70) \\ &\quad Point \text{ p= } guessNextMedian() \\ &Point \text{ pnew= } updateMedianPoint() \end{aligned} \tag{10}$$

3.6 Computing the transverse Arcs at each Median Point

At this point, we have stored each of the median points location, M_0, M_1, M_2, M_3.... Also, each of the median points have access to the nearest projection points on the curves A and B. For example, median point M_0 has point A_1 B_1 on curve A and B respectively.

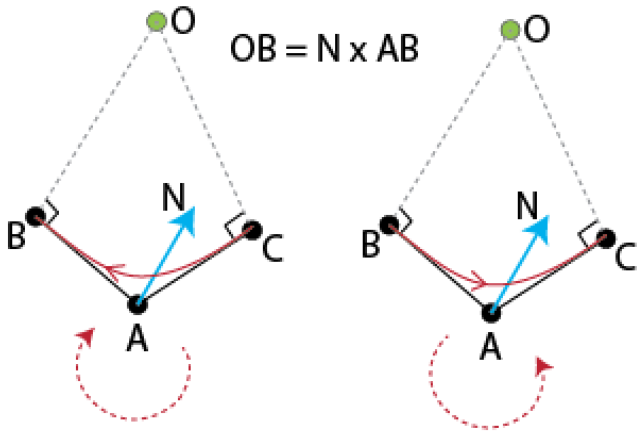


Figure 5: Shows the strategy implemented to computed the Arc Center O

Lets consider any three set of such points namely A, B and C as shown in Figure 5.With these three points we compute the Arc Center point O. We get vector OB and vector OC and compute the angle a between them.To understand if O, B, and C are clockwise or anticlockwise we check if $det(OB, OC)$ is negative or positive respectively.If it is clockwise, we first compute the vector I and J as below:

$$\begin{aligned} \text{vec I} &= normalized(OB) \\ \text{vec J} &= normalized((OCxOB)xOC)) \end{aligned} \tag{11}$$

Then we compute the points on the arc by the following equations:

$$\begin{aligned} \text{vec VN} &= vec(r * (i/n), V(O, pn)) \\ \text{Point OA}_1 &= P(O, VN) \end{aligned} \tag{12}$$

where, OA_1 is one of the computed points on the arc, Point pn is the rotated point around O in the plane of vectors I and J

4 Parallel Transport and Worm

4.1 Median Rope by Parallel Transport

We use the concept of parallel transport to make circles at each median point on the median curve. Parallel transport helps us to keep the normal vector direction aligned with each other for all the median points. Lets say we have three consecutive median control points A, B and C as shown in Figure 6. Let O_1 and O_2 be two points on the median curve , wherein our aim is to keep the Normal vector direction of L_i same as Normal vector direction at point O_2, which is given by L_{i+1}. We computed the vector N_B by taking the cross product of vectors AB and BC. The we decompose the vector L_i in the direction of N_B and another component in another direction by the following equation:

$$\text{L}_i = x^*N_B + y^*(AB \text{ x } N_B) \tag{13}$$

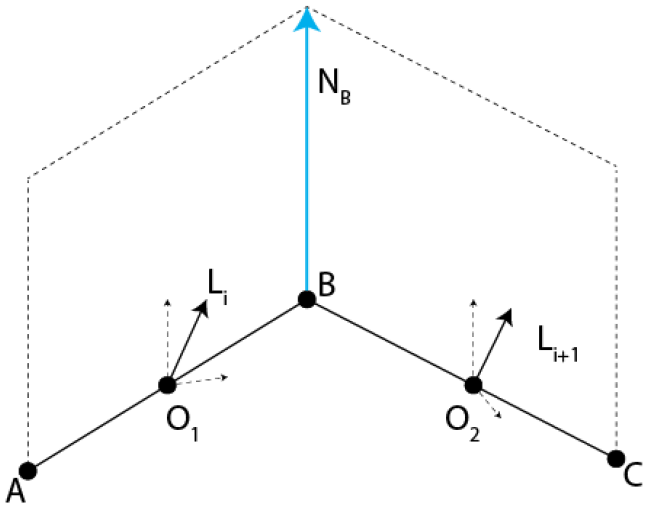


Figure 6: Shows the concept of parallel transport to find the correct normal direction for the next center ball point on the median curve

where x and y are some unknown constants.
Similarly, we computed the vector L_{i+1} as shown below:

$$\mathrm{L}_{i+1} = x^{*}N_B + y^{*}(BC \text{ x } N_B) \tag{14}$$

We solve the above two equations to find the direction of the vector L_{i+1}. We do the same to all the median points to find the normal and the oriented circles. At the next step we form the quads around these circles to form the rope.

4.2 Making of the Worm

We begin by forming a vector C_1C_2 from from first center of ball to the next one on the median curve. We get the cross product of this vector with the vector from the current center of ball to the projected point on Curve B. We call the second vector BB. This cross product gives the vector PP which is perpendicular to bothC_1C_2 and BB. We again take the cross product of vector PP with the vector C_1C_2 to find a vector VF pointing in the plane of the vectors C_1C_2 and vector BB.

We get the number of subdivision of quads, sd from the user and make two dimensional $float$ variables $'c'$ and $'s'$ to store the value of $r * cos(2 * PI * m/sd)$ and $r * sin(2 * PI * m/sd)$, where r is given by the average sum of distance to the projected point on Curve A and B for both the current center ball point and the previous one. We use these values to compute the two perpendicular directors of the center points C_1 and C_2, given by the following equation:

$$\text{Point PT} = \text{Point (p1,c,VF,s, (VF x } \mathrm{C_1}C_2)) \tag{15}$$

where, $p1$ is the summation of points C_1 and C_2.
This was the computation of Point PT for 1st two center ball points on the median curve.

Next we iterate over the rest of the center ball points on the median curve. We compute vector I and IP which are respectively vector between previous center point and current center point and vector between current center point and next one.Further we compute vector IPM by subtracting vector I from vector IP.We also get the normal vector N for vector I and vector IP.

We compute set of points $PT[][]$ similar to point PT computed as shown before by the following equation:

$$\text{Point PT}[][] = \text{Point } ((\text{C}_i + C_{i+1}), c, I, s, J) \tag{16}$$

We use these set of points to form the shape for the worm, which is essentially a series of circular quads having radius r such that both curves A and B are tangential to it.

5 Implementation details and some cautions

5.1 Class Nearest

This Class in rope.pde contains the data structure to store the closest projection in a rope to a certain point. It stores, the projection point, projection distance, the id of the projected point and whether the point is in the middle of an edge.

5.2 Class Median

This Class in rope.pde creates the data structure to save the median point P with two other Nearest Objects, storing the respective nearest points on each Curve A and B.

All the medians are stored in an arraylist *mps*

5.3 Class Rope

The given class rope is used to make a new MPS object which is the median curve. we implement the computation of curve points in the function *calRope*() under rope.pde.Likewise we implemented *calTangent*() to compute the tangent vectors at any point on the curve.

The projection of any point on the curve is implemented in *calProject*() function which returns a Nearest object. It takes the id of the point on the curve and the external point to be projected as input.

5.4 Median Point Implementation

The function $calfirstMedian()$ computes the first median point by taking three points as input.

The function $guessNextMedian()$ returns the next best guessed median point on the median curve by taking the last computed median point as an input

The function $updateMedianPoint()$ returns the precise next median point on the median curve by taking the guessed median point as input along with the id's of the last projected points on the left and right curve.

5.5 Median Curve Implementation

The function *calMedian*() iterates to compute the first median point, guessed median point and updated median point repeatedly, till the length of the median curve cannot fit any new median point to make the curve.

5.6 Parallel Transport Worm Implementation

The function *showQuad*() under Ropes.pde implements the parallel transport. Both full worm and half worm structure is implemented by the *showWorm*() function to be found in Ropes.pde.

5.7 Control methods

We can control the output as below

1. mouse + key 'R or r' = vibrate the output
2. mouse + key 'X or x' = rotate X axis
3. mouse + key 'Y or y' = rotate Y axis
4. key 'A' = show animation of the morph
5. key 'C or c' = show the arcs
6. key 'D or d' = show the net (arcs and curves)
7. key 'F or f' = show the worm
8. key 'G or g' = show the half worm

6 Program Output

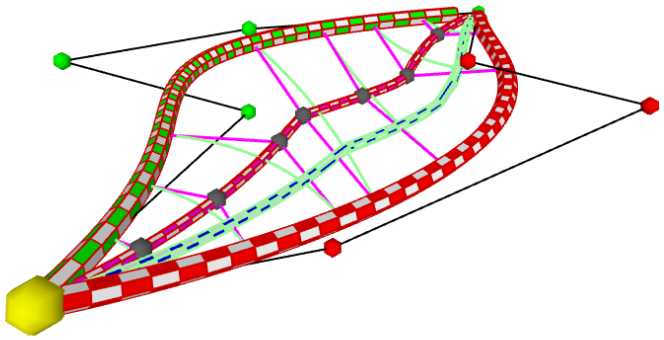


Figure 7: Screenshot of the animating morph via the median curve from Curve A to B

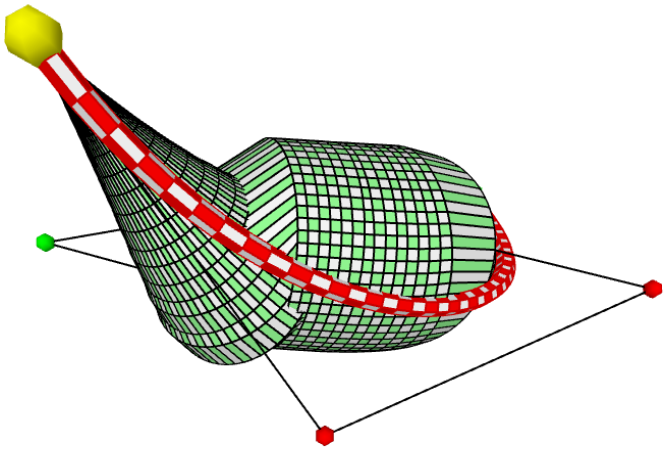


Figure 8: Screenshot of the worm structure between the two tangential rope

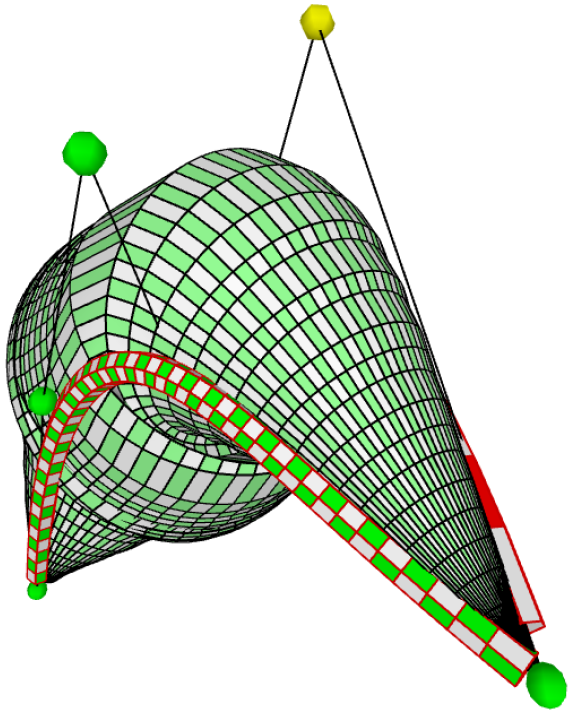


Figure 9: Screenshot of the worm structure between the two tangential rope

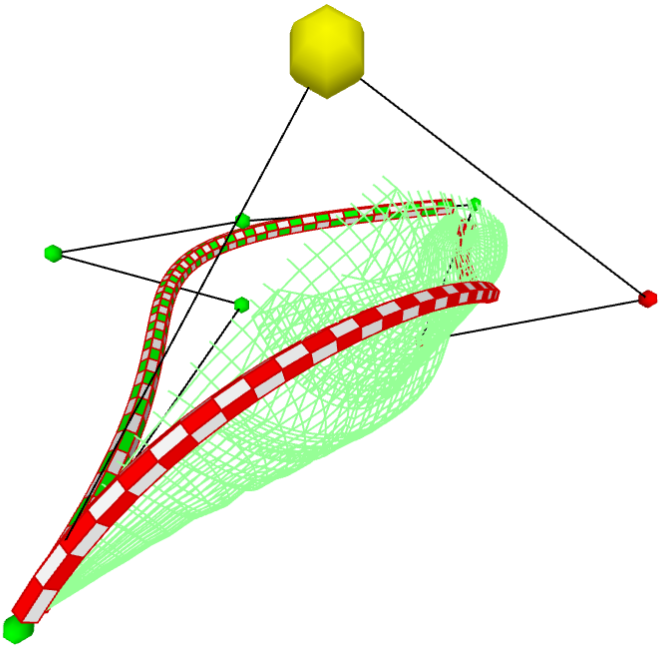


Figure 10: Screenshot of the half worm structure between the two tangential rope

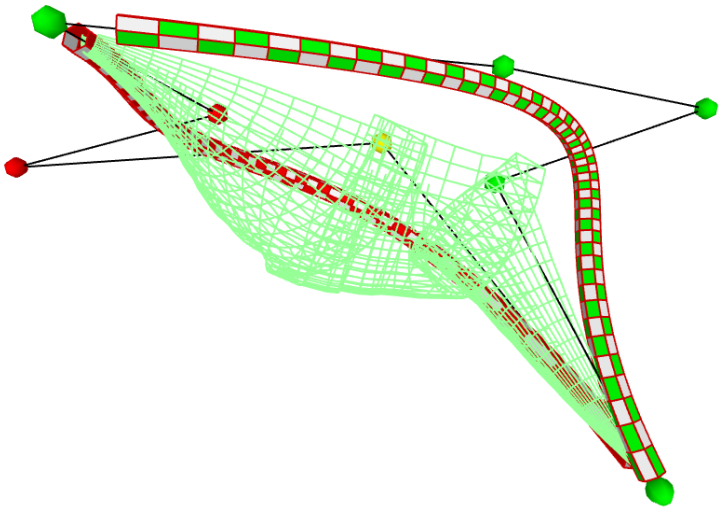


Figure 11: Screenshot of the half worm structure between the two tangential rope

7 Problems and Loose Ends

Though our implementation gives good result for the most part and successfully outputs the intended worm and half worm structure along with the computation of the median curve, it does have its own set of cracks and loose ends. While playing with the input curves and manipulating

in complex ways we figured that at times if the third or fourth control point of one of the input curves is stretched a bit far away from the other input curve, the computed median curve extends beyond the end point and gives erroneous output.

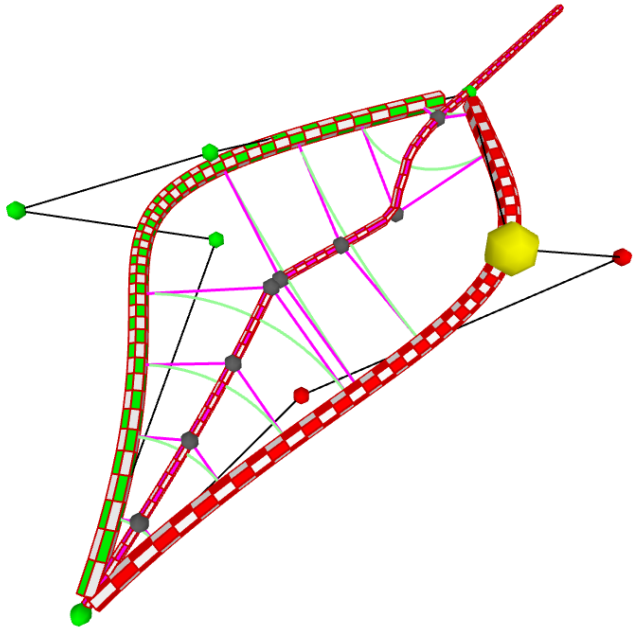


Figure 12: Screenshot of the median curve extends itself outward on one of the ends, if the control points are stretched too far upwards.

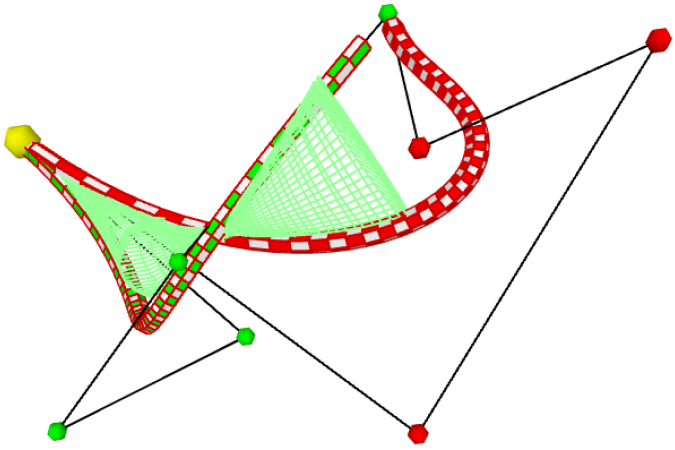


Figure 13: Screenshot of one of the cases for half worm when the two input curves entangle against each other and the half worm ends up not forming itself

Another similar problem was observed in the graphical representation of the halfworm structure.

When the input curves are manipulated such that they overlap each other and entangle, the half worm structure ends up forming only on one half of the area as shown in Figure 13

8 Conclusion, discussion and reliability