Fast and Exact Continuous Collision Detection with Bernstein Sign Classification

Supplementary Material

Min Tang[1*] Ruofeng Tong[1] Zhendong Wang[1] Dinesh Manocha[2†]
1. State Key Lab of CAD&CG, Zhejiang University 2. University of North Carolina at Chapel Hill
http://gamma.cs.unc.edu/BSC/

1 Proofs

In this supplementary document, we provide proofs of various lemmas, theorems and corollaries used in the paper.

Inflection Point Existence Lemma: *For a cubic polynomial $Y(t) = k_0 * B_0^3(t) + k_1 * B_1^3(t) + k_2 * B_2^3(t) + k_3 * B_3^3(t)$, its 2nd order derivative $Y''(t)$ is:*

$$\begin{aligned} Y''(t) &= 6 * (k_2 - 2 * k_1 + k_0) * B_0^1(t) \\ &+ 6 * (k_3 - 2 * k_2 + k_1) * B_1^1(t). \end{aligned}$$

If the two scalars $(k_2 - 2 * k_1 + k_0)$ and $(k_3 - 2 * k_2 + k_1)$ have different signs, there is an inflection point, otherwise there is no inflection point in $t \in [0, 1]$.

Proof. The inflection point corresponds to the root of $Y''(t)$. If the two scalars $(k_2 - 2 * k_1 + k_0)$ and $(k_3 - 2 * k_2 + k_1)$ have different signs, there is one root for $Y''(t)$, i.e. an inflection point, otherwise there is no inflection point in $t \in [0, 1]$. □

Extreme Point Existence Lemma: *For a cubic polynomial $Y(t)$ (defined as above), its 1st derivative $Y'(t)$ is:*

$$\begin{aligned} Y'(t) &= 3 * (k_1 - k_0) * B_0^2(t) + 3 * (k_2 - k_1) * B_1^2(t) \\ &+ 3 * (k_3 - k_2) * B_2^2(t). \end{aligned}$$

If there is no inflection point in its domain and the two scalars ($k_1 - k_0$ and $k_3 - k_2$) have different signs, there is an extreme point, otherwise there is no extreme point in $t \in [0, 1]$.

Proof. If there is no inflection point (i.e. no root for $Y''(t) = 0$) in $[0, 1]$, $Y'(t)$ is monotonic in the domain. It the two scalars ($k_1 - k_0$ and $k_3 - k_2$) have different signs, there is a root of $Y'(t) = 0$ in $[0, 1]$, which corresponds to an extreme point. Otherwise there is no extreme point in $[0, 1]$. □

Polynomial Decomposition Theorem: Let $G(t)$ and $H(t)$ be a cubic polynomial and a quadratic polynomial, respectively:

$$\begin{aligned} &G(t) = i_0 * B_0^3(t) + i_1 * B_1^3(t) + i_2 * B_2^3(t) + i_3 * B_3^3(t), \\ &H(t) = j_0 * B_0^2(t) + j_1 * B_1^2(t) + j_2 * B_2^2(t). \end{aligned} \tag{1}$$

$G(t)$ can be decomposed as:

$$G(t) = L(t) * H(t) + K(t), \tag{2}$$

where $L(t)$ and $K(t)$ are two linear polynomials:

$$\begin{aligned} L(t) &= u_0 * B_0^1(t) + u_1 * B_1^1(t), \\ K(t) &= v_0 * B_0^1(t) + v_1 * B_1^1(t), \end{aligned} \tag{3}$$

where $u_{[0,1]}$ and $v_{[0,1]}$ can be calculated from $i_{[0\ldots3]}$ and $j_{[0\ldots2]}$.

Proof. This can be proven by substituting the algebraic expressions.

$$\begin{aligned} L(t) * H(t) &= (j_0 * B_0^2(t) + j_1 * B_1^2(t) + j_2 * B_2^2(t)) \\ &* (u_0 * B_0^1(t) + u_1 * B_1^1(t)) \\ &= u_0 * j_0 * B_0^3(t) \\ &+ \frac{2 * u_0 * j_1 + u_1 * j_0}{3} * B_1^3(t) \\ &+ \frac{u_0 * j_2 + 2 * u_1 * j_1}{3} * B_2^3(t) \\ &+ u_1 * j_2 * B_3^3(t). \end{aligned} \tag{4}$$

Moreover,

$$\begin{aligned} K(t) &= K(t) * (1 - t + t)^2 \\ &= (v_0 * B_0^1(t) + v_1 * B_1^1(t)) \\ &* (B_0^2(t) + B_1^2(t) + B_2^2(t)) \\ &= v_0 * B_0^3(t) + \frac{2 * v_0 + v_1}{3} * B_1^3(t) \\ &+ \frac{v_0 + 2 * v_1}{3} * B_2^3(t) + v_1 * B_3^3(t). \end{aligned} \tag{5}$$

From Equation (4) and Equation (5), we obtain:

$$\begin{aligned} L(t) &* H(t) + K(t) \\ &= (u_0 * j_0 + v_0) * B_0^3(t) \\ &+ \frac{2 * u_0 * j_1 + u_1 * j_0 + 2 * v_0 + v_1}{3} * B_1^3(t) \\ &+ \frac{u_0 * j_2 + 2 * u_1 * j_1 + v_0 + 2 * v_1}{3} * B_2^3(t) \\ &+ (u_1 * j_2 + v_1) * B_3^3(t). \end{aligned} \tag{6}$$

Based on Equation (1) and Equation (2), we obtain:

$$i_0 = u_0 * j_0 + v_0 \tag{7}$$

$$i_1 = \frac{2 * u_0 * j_1 + u_1 * j_0 + 2 * v_0 + v_1}{3} \tag{8}$$

$$i_2 = \frac{u_0 * j_2 + 2 * u_1 * j_1 + v_0 + 2 * v_1}{3} \tag{9}$$

$$i_3 = u_1 * j_2 + v_1. \tag{10}$$

*e-mail:{tang_m,trf,westernseawolf}@zju.edu.cn
†e-mail:dm@cs.unc.edu

From Equation (7) and Equation (10), we obtain:

$$v_0 = i_0 - u_0 * j_0$$
$$v_1 = i_3 - u_1 * j_2.$$

We substitute these expressions into Equation (8) and Equation (9) and obtain:

$$\begin{aligned} 3*i_1 &= 2*u_0*j_1 + u_1*j_0 + 2*v_0 + v_1 \\ &= 2*u_0*j_1 + u_1*j_0 \\ &+ 2*(i_0 - u_0*j_0) + i_3 - u_1*j_2 \\ 3*i_2 &= u_0*j_2 + 2*u_1*j_1 + v_0 + 2*v_1 \\ &= u_0*j_2 + 2*u_1*j_1 \\ &+ i_0 - u_0*j_0 + 2*(i_3 - u_1*j_2) \end{aligned}$$

After rearranging the equations, we obtain:

$$2*(j_1 - j_0)*u_0 + (j_0 - j_2)*u_1 = 3*i_1 - 2*i_0 - i_3$$
$$(j_2 - j_0)*u_0 + 2*(j_1 - j_2)*u_1 = 3*i_2 - 2*i_3 - i_0$$

This can be expressed as:

$$\begin{aligned} u_0 &= \frac{\begin{vmatrix} 2*(j_1 - j_2) & j_0 - j_2 \\ 3*i_2 - 2i_3 - i_0 & 3*i_1 - 2*i_0 - i_3 \end{vmatrix}}{\begin{vmatrix} 2*(j_1 - j_2) & j_0 - j_2 \\ j_2 - j_0 & 2*(j_1 - j_0) \end{vmatrix}}, \\ u_1 &= \frac{\begin{vmatrix} 2*(j_1 - j_0) & j_2 - j_0 \\ 3*i_1 - 2*i_0 - i_3 & 3*i_2 - 2i_3 - i_0 \end{vmatrix}}{\begin{vmatrix} 2*(j_1 - j_2) & j_0 - j_2 \\ j_2 - j_0 & 2*(j_1 - j_0) \end{vmatrix}}, \\ v_0 &= i_0 - u_0*j_0, \\ v_1 &= i_3 - u_1*j_2. \end{aligned}$$

□

Coplanarity Test Theorem for a VF Pair: *For a deforming triangle, whose initial and final positions are given as (*$\mathbf{a_0}$,$\mathbf{b_0}$, $\mathbf{c_0}$*) and (*$\mathbf{a_1}$, $\mathbf{b_1}$, $\mathbf{c_1}$*) and a vertex with initial and final positions as* $\mathbf{p_0}$ *and* $\mathbf{p_1}$*, the coplanarity test can be formulated as:*

$$\begin{aligned} &Y(t) = (\mathbf{p}_t - \mathbf{a}_t) \cdot \mathbf{n}_t \\ &= k_0 * B_0^3(t) + k_1 * B_1^3(t) + k_2 * B_2^3(t) + k_3 * B_3^3(t), \end{aligned} \quad (11)$$

where $k_{[0..3]}$ *are scalars:*

$$k_0 = (\mathbf{p}_0 - \mathbf{a}_0) \cdot \mathbf{n}_0, \quad k_3 = (\mathbf{p}_1 - \mathbf{a}_1) \cdot \mathbf{n}_1,$$
$$k_1 = (2*(\mathbf{p}_0 - \mathbf{a}_0) \cdot \hat{\mathbf{n}} + (\mathbf{p}_1 - \mathbf{a}_1) \cdot \mathbf{n}_0)/3,$$
$$k_2 = (2*(\mathbf{p}_1 - \mathbf{a}_1) \cdot \hat{\mathbf{n}} + (\mathbf{p}_0 - \mathbf{a}_0) \cdot \mathbf{n_1})/3.$$

and

$$\mathbf{n_0} = (\mathbf{b_0} - \mathbf{a_0}) \times (\mathbf{c_0} - \mathbf{a_0}),\ \mathbf{n_1} = (\mathbf{b_1} - \mathbf{a_1}) \times (\mathbf{c_1} - \mathbf{a_1}),$$
$$\hat{\mathbf{n}} = (\mathbf{n_0} + \mathbf{n_1} - (\mathbf{v_b} - \mathbf{v_a}) \times (\mathbf{v_c} - \mathbf{v_a})) * 0.5,$$
$$\mathbf{v_a} = \mathbf{a_1} - \mathbf{a_0},\ \mathbf{v_b} = \mathbf{b_1} - \mathbf{b_0},\ \mathbf{v_c} = \mathbf{c_1} - \mathbf{c_0}.$$

Proof. The normal vector $\mathbf{n}_t$ of the deforming triangle at time t can be represented as following:

$$\mathbf{n}_t = \mathbf{n_0} * B_0^2(t) + \hat{\mathbf{n}} * B_1^2(t) + \mathbf{n_1} * B_2^2(t), \quad (12)$$

where $B_i^2(t)$ is the i^{th} basis function of the Bernstein polynomials of degree 2.

We define: $\alpha = B_0^2(t) = (1-t)^2$, $\beta = B_1^2(t) = 2*t*(1-t)$, and $\gamma = B_2^2(t) = t^2$. As a result, Equation (12) becomes:

$$\mathbf{n}_t = \mathbf{n_0} * \alpha + \hat{\mathbf{n}} * \beta + \mathbf{n_1} * \gamma.$$

Given the moving vertex $\mathbf{p}_t = \mathbf{p_0} * (1-t) + \mathbf{p_1} * t$ and a vertex of the deforming triangle $\mathbf{a}_t = \mathbf{a_0} * (1-t) + \mathbf{a_1} * t$, their projected distance along $\mathbf{n}_t$ is:

$$\begin{aligned} (\mathbf{p}_t - \mathbf{a}_t) \cdot \mathbf{n}_t &= ((\mathbf{p_0} - \mathbf{a_0}) * (1-t) + (\mathbf{p_1} - \mathbf{a_1}) * t) \cdot \mathbf{n}_t \\ &= ((\mathbf{p_0} - \mathbf{a_0}) * (1-t) + (\mathbf{p_1} - \mathbf{a_1}) * t) \\ &\quad \cdot (\mathbf{n_0} * \alpha + \hat{\mathbf{n}} * \beta + \mathbf{n_1} * \gamma) \\ &= (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_0} * (1-t) * \alpha \\ &+ (\mathbf{p_0} - \mathbf{a_0}) \cdot \hat{\mathbf{n}} * (1-t) * \beta \\ &+ (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_1} * (1-t) * \gamma \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_1} * t * \gamma \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \hat{\mathbf{n}} * t * \beta \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_0} * t * \alpha. \end{aligned}$$

We substitute α, β, and γ and obtain:

$$\begin{aligned} (\mathbf{p}_t - \mathbf{a}_t) \cdot \mathbf{n}_t &= (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_0} * (1-t)^3 \\ &+ (\mathbf{p_0} - \mathbf{a_0}) \cdot \hat{\mathbf{n}} * 2*(1-t)^2 * t \\ &+ (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_1} * (1-t) * t^2 \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_1} * t^3 \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \hat{\mathbf{n}} * 2*t^2 * (1-t) \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_0} * t * (1-t)^2. \end{aligned} \quad (13)$$

Base on Equation (11), we have the k_0, k_1, k_2, and k_3:

$$k_0 = (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_0}, \quad k_3 = (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_1},$$
$$k_1 = (2*(\mathbf{p_0} - \mathbf{a_0}) \cdot \hat{\mathbf{n}} + (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_0})/3,$$
$$k_2 = (2*(\mathbf{p_1} - \mathbf{a_1}) \cdot \hat{\mathbf{n}} + (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_1})/3.$$

□

Inside Test Theorem for a VF Pair: *Given the triangle and the vertex defined by start and end positions over the interval* $[0, 1]$*, the inside test can be formulated as:*

$$\begin{aligned} ((\mathbf{b_t} - \mathbf{p_t}) \times (\mathbf{c_t} - \mathbf{p_t})) \cdot \mathbf{n_t} &= l_0 * B_0^4(t) + l_1 * B_1^4(t) \\ &+ l_2 * B_2^4(t) + *l_3 * B_3^4(t) + l_4 * B_4^4(t), \end{aligned} \quad (14)$$

where $l_{[0...4]}$ *are scalars:*

$$l_0 = \mathbf{m_0} \cdot \mathbf{n_0},\ l_1 = \frac{\mathbf{m_0} \cdot \hat{\mathbf{n}} + \hat{\mathbf{m}} \cdot \mathbf{n_0}}{2},\ l_3 = \frac{\hat{\mathbf{m}} \cdot \mathbf{n_1} + \mathbf{m_1} \cdot \hat{\mathbf{n}}}{2},$$
$$l_2 = \frac{\mathbf{m_0} \cdot \mathbf{n_1} + \mathbf{4} * \hat{\mathbf{m}} \cdot \hat{\mathbf{n}} + \mathbf{m_1} \cdot \mathbf{n_0}}{6},\ l_4 = \mathbf{m_1} \cdot \mathbf{n_1},$$

and

$$\mathbf{n_0} = (\mathbf{b_0} - \mathbf{a_0}) \times (\mathbf{c_0} - \mathbf{a_0}),\ \mathbf{n_1} = (\mathbf{b_1} - \mathbf{a_1}) \times (\mathbf{c_1} - \mathbf{a_1}),$$
$$\hat{\mathbf{n}} = (\mathbf{n_0} + \mathbf{n_1} - (\mathbf{v_b} - \mathbf{v_a}) \times (\mathbf{v_c} - \mathbf{v_a})) * 0.5,$$
$$\mathbf{m_0} = (\mathbf{b_0} - \mathbf{p_0}) \times (\mathbf{c_0} - \mathbf{p_0}),\ \mathbf{m_1} = (\mathbf{b_1} - \mathbf{p_1}) \times (\mathbf{c_1} - \mathbf{p_1}),$$
$$\hat{\mathbf{m}} = (\mathbf{m_0} + \mathbf{m_1} - (\mathbf{v_b} - \mathbf{v_p}) \times (\mathbf{v_c} - \mathbf{v_p})) * 0.5,$$
$$\mathbf{v_a} = \mathbf{a_1} - \mathbf{a_0},\ \mathbf{v_b} = \mathbf{b_1} - \mathbf{b_0},\ \mathbf{v_c} = \mathbf{c_1} - \mathbf{c_0},\ \mathbf{v_p} = \mathbf{p_1} - \mathbf{p_0}.$$

Proof. Its proof is similar to the proof of **Coplanarity Test Theorem for a VF Pair**. We can replace $(\mathbf{p}_t - \mathbf{a}_t)$ with $((\mathbf{b_t} - \mathbf{p_t}) \times (\mathbf{c_t} - \mathbf{p_t}))$. □

Simplified Inside Test Theorem for a VF pair: *Based on combining Inequality $G(t) \geq 0$ with Equation $Y(t) = 0$ and algebraic manipulation, this inside test can be reduced to the following lower degree constraint:*

$$P(t) = p_0 * B_0^2(t) + p_1 * B_1^2(t) + p_2 * B_2^2(t) \geq 0, \quad (15)$$

where:

$$Y(t) = k_0 * B_0^3(t) + k_1 * B_1^3(t) + k_2 * B_2^3(t) + k_3 * B_3^3(t).$$
$$G(t) = l_0 * B_0^4(t) + l_1 * B_1^4(t) + l_2 * B_2^4(t) + *l_3 * B_3^4(t) + l_4 * B_4^4(t)$$

and $p_{[0\ldots2]}$ are scalars, which can be calculated based on $k_{[0\ldots3]}$ and $l_{[0\ldots4]}$.

Proof. We have:

$$Y(t) = k_0 * B_0^3(t) + k_1 * B_1^3(t) + k_2 * B_2^3(t) + k_3 * B_3^3(t) = 0 \quad (16)$$

and:

$$\begin{aligned} Y(t) &= k'_0 * B_0^4(t) + k'_1 * B_1^4(t) + k'_2 * B_2^4(t) \\ &+ k'_3 * B_3^4(t) + k'_4 * B_4^4(t) = 0. \end{aligned} \quad (17)$$

Here:

$$k'_0 = k_0,\ k'_1 = \frac{k_0 + 3 * k_1}{4},\ k'_2 = \frac{k_1 + k_2}{2}$$
$$k'_3 = \frac{3 * k_2 + k_3}{4},\ k'_4 = k_3.$$

We obtain

$$\begin{aligned} G(t) * k'_0 - Y(t) * l_0 &= \\ (l_1 * k'_0 - l_0 * k'_1) * B_1^4(t) &+ \\ (l_2 * k'_0 - l_0 * k'_2) * B_2^4(t) &+ \\ (l_3 * k'_0 - l_0 * k'_3) * B_3^4(t) &+ \\ (l_4 * k'_0 - l_0 * k'_4) * B_4^4(t). & \end{aligned} \quad (18)$$

This Equation can be expressed as:

$$s_0 * B_0^3(t) + s_1 * B_1^3(t) + s_2 * B_2^3(t) + s_3 * B_3^3(t). \quad (19)$$

Here:

$$\begin{aligned} s_0 &= (l_1 * k'_0 - l_0 * k'_1) * 4 \\ s_1 &= (l_2 * k'_0 - l_0 * k'_2) * 2 \\ s_2 &= \frac{(l_3 * k'_0 - l_0 * k'_3) * 4}{3} \\ s_3 &= l_4 * k'_0 - l_0 * k'_4 \end{aligned} \quad (20)$$

And:

$$\begin{aligned} (G(t) * k'_0 - Y(t) * l_0) * k_0 - Y(t) * s_0 &= \\ (s_1 * k_0 - s_0 * k_1) * B_1^3(t) &+ \\ (s_2 * k_0 - s_0 * k_2) * B_2^3(t) &+ \\ (s_3 * k_0 - s_0 * k_3) * B_3^3(t). & \end{aligned} \quad (21)$$

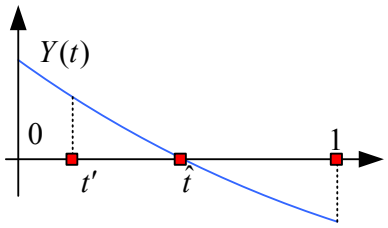


Figure 1: Side Determination Theorem I: *Given a $t' \in [0, 1]$, if $Y(t')$ has the same sign of $Y(0)$, then $t' < \hat{t}$, else $t' > \hat{t}$.*

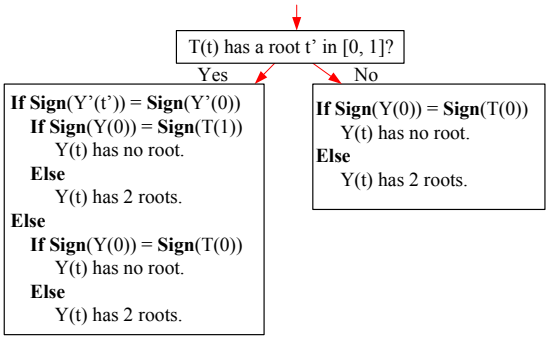


Figure 2: Computing the Number of Roots of $Y(t)$: *We can compute them based on sign evaluations.*

This Equation can be expressed as:

$$p_0 * B_0^2(t) + p_1 * B_1^2(t) + p_2 * B_2^2(t). \quad (22)$$

Here:

$$\begin{aligned} p_0 &= (s_1 * k_0 - s_0 * k_1) * 3 \\ p_1 &= \frac{(s_2 * k_0 - s_0 * k_2) * 3}{2} \\ p_2 &= s_3 * k_0 - s_0 * k_3 \end{aligned} \quad (23)$$

Sine $Y(t) = 0$, we have:

$$\begin{aligned} P(t) &= (G(t) * k'_0 - Y(t) * l_0) * k_0 - Y(t) * s_0 \\ &= G(t) * k'_0 * k_0 = G(t) * k_0 * k_0. \end{aligned}$$

So $P(t)$ has the same sign of $G(t)$. □

Side Determination Theorem I: *Given a polynomial $Y(t)$, which has only one root $\hat{t} \in [0, 1]$ and $Y(\hat{t}) = 0$. $\mathbf{Sign}(Y(0)) \neq \mathbf{Sign}(Y(1))$. Given a $t' \in [0, 1]$, if $Y(t')$ has the same sign as $Y(0)$, then $t' < \hat{t}$, otherwise $t' > \hat{t}$.*

Proof. We prove it using contradiction: If $Y(t')$ has the same sign of $Y(0)$ and $t' > \hat{t}$, then in the interval $[t', 1]$, $\mathbf{Sign}(Y(t')) \neq \mathbf{Sign}(Y(1))$. Since Y(t) is a continuous function, it must have another root in the domain $[t', 1]$. This is contradictory to the fact that $\hat{(t)}$ is the only root in the domain $[0, 1]$. □

Root Finding Lemma: *For a cubic polynomial $Y(t)$ with an extreme point in its domain, its 1st derivative $Y'(t)$ is:*

$$\begin{aligned} Y'(t) &= 3 * (k_1 - k_0) * B_0^2(t) + 3 * (k_2 - k_1) * B_1^2(t) \\ &+ 3 * (k_3 - k_2) * B_2^2(t). \end{aligned}$$

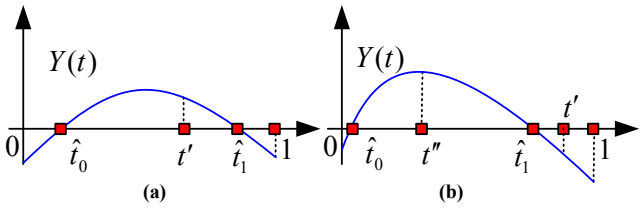


Figure 3: Sign Determination Theorem II: *For a $t' \in [0,1]$, if $Y(t')$ has the same sign of $Y(0)$, then $t' < \hat{t}$, else $t' > \hat{t}$.*

We have $Y(t) = Y'(t) * L(t) + K(t)$, where $L(t)$ and $K(t)$ are two linear polynomials and can be calculated with the **Polynomial Decomposition Theorem**. *We can use the rules in Fig. 2 to compute the number of roots of $Y(t)$.*

Proof. We define t'' as the only root of $Y'(t)$, i.e., $Y'(t'') = 0$. We need to determine the sign of $Y(t'')$. If $\mathbf{Sign}(Y(t'')) = \mathbf{Sign}(Y(0))$ then $Y(t)$ has no root, otherwise $Y(t)$ must have two roots in $[0,1]$. Since $Y(t'') = Y'(t'') * L(t'') + K(t'') = K(t'')$, $\mathbf{Sign}(Y(t'')) = \mathbf{Sign}(K(t''))$. So we need to determine the sign of $K(t'')$.

If $K(t)$ has no root in $[0,1]$, then $\mathbf{Sign}(K(t'')) = \mathbf{Sign}(K(0))$.

Otherwise, let t' be the only root of $K(t)$ in $[0,1]$. Based on the **Side Determination Theorem I**, if $\mathbf{Sign}(Y'(t')) = \mathbf{Sign}(Y'(0))$, $t' < t''$, else $t' > t''$. If $t' < t''$, then $\mathbf{Sign}(K(t'')) = \mathbf{Sign}(K(1))$, else $\mathbf{Sign}(K(t'')) = \mathbf{Sign}(K(0))$. □

Side Determination Theorem II: *For a given polynomial $Y(t)$, in a domain $[0,1]$ which has two roots $\hat{t}_0$ and $\hat{t}_1$. $\mathbf{Sign}(Y(0)) = \mathbf{Sign}(Y(1))$. For $t' \in [0,1]$, if $Y(t')$ has a different sign than $Y(0)$ (Fig. 3(a)), then $\hat{t}_0 < t' < \hat{t}_1$, otherwise $\hat{t}_0, \hat{t}_1$ are on the same side of t' (Fig. 3(b)).*

Proof. If $Y(t')$ has a different sign as compared to $Y(0)$, then $t' \in [\hat{t}_0, \hat{t}_1]$; otherwise:

- If $t' < \hat{t}_0$, $\mathbf{Sign}(Y(t')) \neq \mathbf{Sign}(Y(0))$, there is another root in the interval [0, t'], this contradicts the fact $Y(t)$ has two roots.
- If $t' > \hat{t}_1$, $\mathbf{Sign}(Y(t')) \neq \mathbf{Sign}(Y(1))$, there is another root in the interval [t', 1], this contradicts the fact that $Y(t)$ has two roots.

If $\mathbf{Sign}(Y(t')) = \mathbf{Sign}(Y(0)) = \mathbf{Sign}(Y(1))$, then $t' \ni [\hat{t}_0, \hat{t}_1]$, otherwise, $\mathbf{Sign}(Y(t')) = \mathbf{Sign}(Y(t''))$. This contradicts the fact that $\mathbf{Sign}(Y(0)) \neq \mathbf{Sign}(Y(t''))$, where t'' is the extreme point. □

Sign Determination Theorem I: *Let $L(t)$ be a linear polynomial and $Y(t)$ be a cubic polynomial which corresponds to the Bézier curve of Case (b) (Section 3.2) in the domain $[0,1]$. Let:*

- *$L(t') = 0$, and $t' \in [0,1]$.*
- *$Y(\hat{t}) = 0$, and $\hat{t} \in [0,1]$.*

We have:

If $\mathbf{Sign}(Y(t')) = \mathbf{Sign}(Y(0))$
  $\mathbf{Sign}(L(\hat{t})) = \mathbf{Sign}(L(1))$
Else
  $\mathbf{Sign}(L(\hat{t})) = \mathbf{Sign}(L(0))$
Endif

Proof. With **Side Determination Theorem I**, we have: if $\mathbf{Sign}(Y(t')) = \mathbf{Sign}(Y(0))$, then $t' < \hat{t} \Rightarrow \mathbf{Sign}(L(\hat{t}) = \mathbf{Sign}(L(1))$, else $t' > \hat{t} \Rightarrow \mathbf{Sign}(L(\hat{(t)}) = \mathbf{Sign}(L(0))$. □

Sign Determination Theorem II: *Let $L(t)$ be a linear polynomial and $Y(t)$ be a cubic polynomial which corresponds to the Bézier curve of Case (c) in the domain $[0,1]$. Let:*

- *$L(t') = 0$, and $t' \in [0,1]$.*
- *$Y(\hat{t}_0) = 0$ and $Y(\hat{t}_1) = 0$, and $\hat{t}_0 \in [0,1], \hat{t}_1 \in [0,1], \hat{t}_0 < \hat{t}_1$.*
- *$Y'(t'') = 0$, and $t'' \in [0,1]$. $Y'(t)$ is the 1st order derivative of $Y(t)$.*

We have:

If $\mathbf{Sign}(Y(t')) \neq \mathbf{Sign}(Y(0))$
  $\mathbf{Sign}(L(\hat{t}_0)) = \mathbf{Sign}(L(0))$
  $\mathbf{Sign}(L(\hat{t}_1)) = \mathbf{Sign}(L(1))$
Else
  If $\mathbf{Sign}(Y'(t')) = \mathbf{Sign}(Y'(0))$
    $\mathbf{Sign}(L(\hat{t}_0)) = \mathbf{Sign}(L(1))$
    $\mathbf{Sign}(L(\hat{t}_1)) = \mathbf{Sign}(L(1))$
  Else
    $\mathbf{Sign}(L(\hat{t}_0)) = \mathbf{Sign}(L(0))$
    $\mathbf{Sign}(L(\hat{t}_1)) = \mathbf{Sign}(L(0))$
  Endif
Endif

Proof. With **Side Determination Theorem II**, we have: If $\mathbf{Sign}(Y(t')) \neq \mathbf{Sign}(Y(0))$, then $\hat{t}_0 < t' < \hat{t}_1 \Rightarrow \mathbf{Sign}(L(\hat{t}_0) = \mathbf{Sign}(L(0))$ and $\mathbf{Sign}(L(\hat{t}_1) = \mathbf{Sign}(L(1))$. Otherwise, $\hat{t}_0$, $\hat{t}_1$, and t'' are at the same side of t', and with **Side Determination Theorem I**, we obtain:

if $\mathbf{Sign}(Y'(t')) = \mathbf{Sign}(Y'(0))$, then $t' < t''$, otherwise $t' > t''$.

$t' < t'' \Rightarrow t' < \hat{t}_0$ and $t' < \hat{t}_1 \Rightarrow \mathbf{Sign}(L(\hat{t}_0)) = \mathbf{Sign}(L(1))$ and $\mathbf{Sign}(L(\hat{t}_1)) = \mathbf{Sign}(L(1))$.

$t' > t'' \Rightarrow t' > \hat{t}_0$ and $t' > \hat{t}_1 \Rightarrow \mathbf{Sign}(L(\hat{t}_0)) = \mathbf{Sign}(L(0))$ and $\mathbf{Sign}(L(\hat{t}_1)) = \mathbf{Sign}(L(0))$. □

2 Error Bound for Conservative Culling

The conservative culling algorithm is described in Section 4.3. It uses a floating-point filter and we present an error-bound for that

filter. Since our computation only uses addition, substraction, and multiple operations, it is relatively simple to derive such a bound.

According to the IEEE 754 standard, given an exact arithmetic operator $\times$ and its floating point counterpart $\bigotimes$, $a \times b = ROUND(a \bigotimes b)$ and $|a \bigotimes b - a \times b| \leq |a \times b| * \epsilon$. For the double precision format, $\epsilon = 2^{-52}$.

We use following rules to evaluate the error bounds for addition/subtraction and , repsetively:

Rule I: Error bound for addition/substraction: Give two numbers with rounding errors, i.e. $a + c_1 * \epsilon$ and $b + c_2 * \epsilon$, rounding error of the addition/substration operation on them will be bounded by:

$$\begin{aligned}\Delta &= (a + c_1 * \epsilon) \pm (b + c_2 * \epsilon) \\ &+ ||(a + c_1 * \epsilon) \pm (b + c_2 * \epsilon)|| * \epsilon \\ &= a \pm b + (c_1 \pm c_2) * \epsilon + ||a \pm b||) * \epsilon + O(\epsilon^2) \\ &< a \pm b + (c_1 \pm c_2) * \epsilon + (||a \pm b|| + 1) * \epsilon \end{aligned} \quad (24)$$

The accumulate rounding error is bounded by $(c1 \pm c2 + ||a \pm b|| + 1) * \epsilon$.

Rule II: Error bound for multiply operation: The multiple of $a + c_1 * \epsilon$ and $b + c_2 * \epsilon$ is Δ (with rounding error):

$$\begin{aligned}\Delta &= (a + c_1 * \epsilon) * (b + c_2 * \epsilon) \\ &+ ||(a + c_1 * \epsilon) * (b + c_2 * \epsilon)|| * \epsilon \\ &= a * b + (b * c_1 + a * c_2 + ||a * b||) * \epsilon + O(\epsilon^2) \\ &< a * b + (b * c_1 + a * c_2 + ||a * b|| + 1) * \epsilon \end{aligned} \quad (25)$$

The accumulative rounding error is bounded by $(b * c_1 + a * c_2 + ||a * b|| + 1) * \epsilon$.

In order to perform conservative culling, we need to test the signs of

$$\begin{aligned} k_0 &= (\mathbf{p}_0 - \mathbf{a}_0) \cdot \mathbf{n}_0, \quad k_3 = (\mathbf{p}_1 - \mathbf{a}_1) \cdot \mathbf{n}_1, \\ k_1 &= (2 * (\mathbf{p}_0 - \mathbf{a}_0) \cdot \hat{\mathbf{n}} + (\mathbf{p}_1 - \mathbf{a}_1) \cdot \mathbf{n}_0)/3, \\ k_2 &= (2 * (\mathbf{p}_1 - \mathbf{a}_1) \cdot \hat{\mathbf{n}} + (\mathbf{p}_0 - \mathbf{a}_0) \cdot \mathbf{n_1})/3. \end{aligned}$$

and

$$\begin{aligned} \mathbf{n_0} &= (\mathbf{b_0} - \mathbf{a_0}) \times (\mathbf{c_0} - \mathbf{a_0}), \mathbf{n_1} = (\mathbf{b_1} - \mathbf{a_1}) \times (\mathbf{c_1} - \mathbf{a_1}), \\ \hat{\mathbf{n}} &= (\mathbf{n_0} + \mathbf{n_1} - (\mathbf{v_b} - \mathbf{v_a}) \times (\mathbf{v_c} - \mathbf{v_a})) * 0.5, \\ \mathbf{v_a} &= \mathbf{a_1} - \mathbf{a_0}, \mathbf{v_b} = \mathbf{b_1} - \mathbf{b_0}, \mathbf{v_c} = \mathbf{c_1} - \mathbf{c_0}. \end{aligned}$$

Let $\mathbf{n_v} = (\mathbf{v_b} - \mathbf{v_a}) \times (\mathbf{v_c} - \mathbf{v_a})$, for k_1 and k_2, it is equivalent to testing:

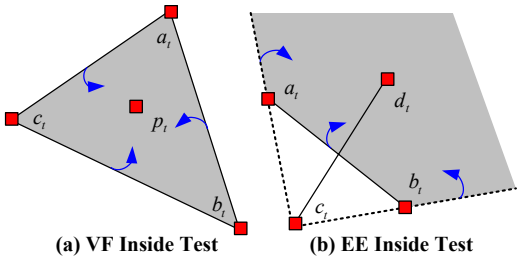
$$\begin{aligned} k_1' &= 2 * (\mathbf{p_0} - \mathbf{a_0}) \cdot \hat{\mathbf{n}} + (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_0} \\ &= (\mathbf{p_0} - \mathbf{a_0}) \cdot (\mathbf{n_0} + \mathbf{n_1} - (\mathbf{v_b} - \mathbf{v_a}) \times (\mathbf{v_c} - \mathbf{v_a})) \\ &+ (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_0}, \\ &= (\mathbf{p_0} - \mathbf{a_0}) \cdot (\mathbf{n_0} + \mathbf{n_1} - \mathbf{n_v}) + (\mathbf{p_1} - \mathbf{a_1}) \cdot \mathbf{n_0}, \end{aligned} \quad (26)$$

Similarly, we have:

$$k_2' = (\mathbf{p_1} - \mathbf{a_1}) \cdot (\mathbf{n_0} + \mathbf{n_1} - \mathbf{n_v}) + (\mathbf{p_0} - \mathbf{a_0}) \cdot \mathbf{n_1}. \quad (27)$$

For k_0, k_1', k_2', k_3, their rounding errors are sum of $\mathbf{i} \times \mathbf{j} \cdot \mathbf{k}$, where $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ are vectors, and

$$\begin{aligned} \mathbf{i} \times \mathbf{j} \cdot \mathbf{k} &= i_y * j_z * k_x - i_z * j_y * k_x \\ &+ i_z * j_x * k_y - i_x * j_z * k_y \\ &+ i_x * j_y * k_z - i_y * j_x * k_z \end{aligned}$$



(a) VF Inside Test **(b) EE Inside Test**

Figure 4: Inside Tests: *For VF and EE pairs, we need to perform inside tests to check if the vertex is inside the triangle, or the two edges are intersecting with each other, when their vertices are coplanar.*

We use the **Rule I** and **Rule II** to accumulate the rounding errors, and compute the error bounds for k_0, k_1', k_2', k_3 on-the-fly. These dynamically computed error bounds are used by the filtering algorithm.

3 EE Query Algorithm

As shown in Fig. 4(b), in order to perform an inside test for a EE pair, we need to perform three one-sided tests to make sure the two edges, $\mathbf{a}_t\mathbf{b}_t$ and $\mathbf{c}_t\mathbf{d}_t$, intersect with each other. This can be expressed based on the following inequalities:

$$((\mathbf{b}_t - \mathbf{d}_t) \times (\mathbf{c}_t - \mathbf{d}_t)) \cdot \mathbf{n}_t \geq 0, \quad (28)$$
$$((\mathbf{c}_t - \mathbf{d}_t) \times (\mathbf{a}_t - \mathbf{d}_t)) \cdot \mathbf{n}_t \geq 0, \quad (29)$$
$$((\mathbf{a}_t - \mathbf{d}_t) \times (\mathbf{b}_t - \mathbf{d}_t)) \cdot \mathbf{n}_t \leq 0. \quad (30)$$

The only different between EE query algorithm vs VF query algorithm is to use these inequalities for the inside tests. The rest of the formulation and algorithm structure is the same.

4 Avoiding Division Operations

A key aspect of the algorithm is that we don't perform any division operations. In practice, division operations are more expensive in the context of extended precision computation and it is harder to obtain tight error bounds for floating-point filters.

For a linear polynomial $L(t) = a * B_0^1(t) + b * B_1^1(t)$, its root is give $t' = \frac{a}{a-b}$. We do not need to perform the division by $(a - b)$, since we only need to:

- Check if $L(t)$ has a root in $[0, 1]$: We can check the signs of a, b. If they have the same sign, there is no root in $[0, 1]$, otherwise, there is 1 root in $[0, 1]$.
- Evaluate $Y(t')$, where $Y(t)$ is a cubic or quadratic polynomial in Bernstein form. In our algorithm, we only need to know the sign of Y(t). For a cubic polynomial: If $(a - b) > 0$, $\mathbf{Sign}(Y(\frac{a}{a-b}) = \mathbf{Sign}(Y(a))$, otherwise $\mathbf{Sign}(Y(\frac{a}{a-b}) = -\mathbf{Sign}(Y(a))$. For a quadratic polynomial, $\mathbf{Sign}(Y(\frac{a}{a-b}) = \mathbf{Sign}(Y(a))$.

In both these cases, we can compute the signs of the expression without explicitly performing a division operations.