[27] D. Lischinski, F. Tampieri, and D. P. Greenberg, "Discontinuity meshing for accurate radiosity," *IEEE Computer Graphics and Applications*, vol. 12, no. 6, pp. 25–39, 1992.
[28] C.-Y. Yao, M.-T. Chi, T.-Y. Lee, and T. Ju, "Region-based line field design using harmonic functions," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 902–913, 2012.
[29] R. Schmidt, C. Grimm, and B. Wyvill, "Interactive decal compositing with discrete exponential maps," in *ACM SIGGRAPH 2006 Papers*, ser. SIGGRAPH '06, 2006, pp. 605–613.

Chih-Yuan Yao Chin-Yuan Yao is an Assistant Professor in the Department of Computer Science and Information Engineering at National Taiwan University of Science and Technology (NTUST), Taipei, Taiwan. He received his M.S. and Ph.D. degrees in Computer Science and Information Engineering from National Cheng-Kung University, Taiwan, in 2003 and 2010, respectively. His research interest is computer graphics, including mesh processing and modeling, and non-photorealistic rendering (NPR)

Kuang-Yi Chen Kuang-Yi Chen received the B.S. and M.S. in Department of Computer Science and Information Engineering from National Taiwan University of Science and Technology, Taipei, R.O.C. in 2011 and 2014 respectively. And He is currently an employee in International Game System. His research interests are computer graphics and GPU shader.

Hong-Nian Guo Hong-Nian Guo received the B.S. in Department of Computer Science and Information Engineering from National Taiwan University of Science and Technology, Taipei, R.O.C. in 2015. And He is currently a MS student in NTUST in Department of Computer Science and Information Engineering. His research interests are computer graphics, vision, and nonphotorealistic rendering (NPR).

Cheng-Chi Li received the B.S. in Department of Computer Science and Information Engineering from National Taiwan University of Science and Technology, Taipei, R.O.C. in 2014. And He is currently a MS student in NTUST in Department of Computer Science and Information Engineering. His research interests are computer graphics, vision, and parallel computing.

Yu-Chi Lai Yu-Chi Lai received the B.S. from National Taiwan University, Taipei, R.O.C., in 1996 in Electrical Engineering Department. He received his M.S. and Ph.D. degrees from University of WisconsinMadison in 2003 and 2009 respectively in Electrical and Computer Engineering and his M.S. and Ph.D. degrees in 2004 and 2010 respectively in Computer Science. He is currently an assistant professor in NTUST and his Research interests are in the area of graphics, vision, and multimedia.

APPENDIX A
CURVE INTERSECTION

For those lines and curves whose bounding boxes overlap each other, their intersections are computed based their respective types:

1) **Two lines**: $\tilde{P}$ and $\tilde{Q}$ represent two lines defined in their normalized implicit form of $a_p x+b_p y+c_p=0$ and $a_q x+b_q y+c_q=0$ where $a^2+b^2=1$. The intersection can be determine by solving $\tilde{P}=\tilde{Q}$.
2) **A line and a curve**: $\tilde{P}$ is a line and $\mathbf{Q}(u)$ is a curve in the form of $\mathbf{Q}(u)=\sum_{i=0}^{n}\overline{P}_i B_i^n(u)$ where $\overline{P}_i=(x_i,y_i)$ are the control points and $B_i^n(t)$ denote the Bernstein basis functions. The distance of a point to $\tilde{P}$ can be expressed as $d(\overline{p})=a\overline{p}_x+b\overline{p}_y+c$ and $d(u)=\sum_{i=0}^{n}d_i B_i^n(u)$ where $d_i=a\overline{P}_{i,x}+b\overline{P}_{i,y}+c$. The intersections are defined as $d(u)=0$ by solving a polynomial equation.
3) **Two curves**: $\mathbf{P}(t)$ and $\mathbf{Q}(u)$ represent two curves. Before determining intersections among curves, we must define the fat line of a Bézier curve which is the convex boundary hull to bound the curve. In order to define the fat line for $\mathbf{Q}(u)$, we define its center line $\tilde{Q}_c$ as the line segment connecting its end points $\overline{Q}_0$ and $\overline{Q}_n$. When applying the convex hull property, a fat line $\hat{Q}$ is defined as a set of lines which parallel to $\tilde{Q}_c$ and can most tightly encloses the control points of a given Bézier curve as

$$\hat{Q}=\{\overline{q}|d_{min}\le\tilde{Q}(\overline{q})\le d_{max}\} \tag{2}$$

where $\tilde{Q}(\overline{q})$ is a line parallel to $\tilde{Q}_c$ and passing through $\overline{q}$, $q_i=\tilde{Q}_c(\overline{Q}_i)$ is the distance from $\overline{Q}_i$ to the center line, and

$$\begin{aligned} d_{min}&=min\{d_0,\cdots,d_n\},\\ d_{max}&=max\{d_0,\cdots,d_n\}. \end{aligned} \tag{3}$$

The possible intersection interval on $\mathbf{P}(t)$ can be identified by $\hat{Q}$.The intersections of $\hat{Q}$ with $\mathbf{P}(t)$ can use the method described previously to find the intersection of $a_q x+b_q y+c_q=d_{min}$ and $a_q x+b_q y+c_q=d_{max}$ with $\mathbf{P}(t)$. The intersection parameters can determine the possible intersection interval of $P(t)$ and the portions of $\mathbf{P}(t)$ outside the possible interval are removed. The left portion of $\mathbf{P}(t)$ can be used to create another fat line to find the possible intersection interval on $\mathbf{Q}(u)$.The process repeats until the area of the hull of $\mathbf{P}(t)$ and $\mathbf{Q}(u)$ is smaller than our selected threshold T_s. Then, the de Casteljau algorithm is applied to find the proper line segment approximation of two curves and those segments are used to determine the intersections.

After determining all intersections, our system uses these intersections to decompose lines and curves into shorter segments whose interior and exterior shading instructions are the same. Fig. 4 shows an exemplar result of path intersection and segmentation.

APPENDIX B
PATHS IMPLICITIZATION

[3] implicitizes a path and shades a pixel using a path inside/outside test for resolution independency, but their algorithm

cannot work for paths with composite coloring operations. Our system overcomes this limitation by decomposing paths in unifying-coloring-operation segments and encoding all possible coloring operations into a color texture to shade a pixel using a similar path inside/outside test along with texture-coordinate-based distance interpolation. Before discussing coloring texture construction, this section will first explain the details of implicitizing a path segment.

All paths can be described as integral curves in the form of $\mathbf{f}(\overline{p}) = \tilde{k}(\overline{p})^3 - \tilde{l}(\overline{p})\tilde{m}(\overline{p})$ where $\tilde{k}$, $\tilde{l}$, and $\tilde{m}$ are lines passing through corresponding inflection points discussed later. Then, we can use $\mathbf{f}(\overline{p})$ and $\tilde{k}(\overline{p})$ to evaluate whether a point is inside/outside the path. Furthermore, when a point $\overline{V}$ is inside the convex hull of a cubic Bézier curve, its location can be expressed as barycentric interpolation of its four control points, $\overline{V}_0 \cdots \overline{V}_3$. Similarly, its distance to a line $\tilde{L} = ax + by + c$ can also be barycentric interpolation of the distance of control points to $\tilde{L}$. Therefore, we can compute and store the distance to $\tilde{k}$, $\tilde{l}$, and $\tilde{m}$ as k, l and m for each control point and use barycentric interpolation of these values for shading a pixel.

[3] shows that the type of a cubic curve is determined by the number of inflection points which are points whose curvature vanishes and can be defined as $I(t,s) = t(3d_1s^2 - 3d_2ts - d_3t^2)$ where $\{d_1,d_2,d_3\} = \{a_1 - 2a_2 + 3a_3, -a_2 + 3a_3, 3a_3\}$, and $\{a_1,a_2,a_3\} = \{\overline{c}_0(\overline{c}_3 \times \overline{c}_2), \overline{c}_1(\overline{c}_0 \times \overline{c}_3), \overline{c}_2(\overline{c}_1 \times \overline{c}_0)\}$. To categorize the curve type, we need to evaluate the discriminant of I as $discr(I) = d_1^2(3d_2^2 - 4d_1d_3)$. According to d_1, d_2, d_3, and I, the path types and $\mathbf{k} = [k_0, \cdots, k_3]^t$, $\mathbf{l} = [l_0, \cdots, l_3]^t$, and $\mathbf{m} = [m_0, \cdots, m_3]^t$ for four control points are determined as follows:

1) If $d_1 \neq 0$ and $3d_2^2 - 4d_1d_3 > 0$, it is a **serpentine** which has three collinear inflection points lying on $\tilde{k}$ with corresponding tangent lines $\tilde{l}$, $\tilde{m}$, and $\tilde{n}$ passing through those inflection points. Three inflection points occur when the parameters of $I(t,s)$ are

$$\begin{aligned}(t_l,s_l) &= \left(d_2 + \frac{1}{\sqrt{3}}\sqrt{3d_2^2 - 4d_1d_3}\ ,\ 2d_1\right)\\ (t_m,s_m) &= \left(d_2 - \frac{1}{\sqrt{3}}\sqrt{3d_2^2 - 4d_1d_3}\ ,\ 2d_1\right)\end{aligned} \tag{4}$$

and $\mathbf{k}$, $\mathbf{l}$, and $\mathbf{m}$ are determined as:

$$\begin{pmatrix} t_l t_m, & & \\ t_l t_m - \frac{1}{3}t_l s_m - \frac{1}{3}t_m s_l & t_l^2(t_l - s_l) & t_m^2(t_m - s_m) \\ \frac{1}{3}((t_l(t_m - 2s_m) + s_l(3t_m - 2s_m)) & (s_l - t_l)^2 t_l & (s_m - t_m)^2 t_m \\ (s_l - t_l)(s_m - t_m) & -(s_l - t_l)^3 & -(s_m - t_m)^3 \end{pmatrix}$$

2) If $d_1 \neq 0$ and $3d_2^2 - 4d_1d_3 < 0$, it is a **loop** which has one inflection point and one double point lying on $\tilde{k}$ and the tangent lines passing through the double point as $\tilde{l}$ and $\tilde{m}$ and the tangent line passing through the inflection point as $\tilde{n}$. The double point occurs when the parameters of $I(t,s)$ are:

$$\begin{aligned}(t_d,s_d) &= \left(d_2 + \sqrt{4d_1d_3 - 3d_2^2}\ ,\ 2d_1\right)\\ (t_e,s_e) &= \left(d_2 - \sqrt{4d_1d_3 - 3d_2^2}\ ,\ 2d_1\right)\end{aligned} \tag{5}$$

and $\mathbf{k}$, $\mathbf{l}$, and $\mathbf{m}$ are determined as:

$$\begin{pmatrix} t_e t_d & t_e^2 t_d & t_e t_d^2 \\ \frac{1}{3}(-t_e s_d - s_e t_d + t_e t_d) & -\frac{1}{3}t_e(t_e(s_d - 3t_d) + 2s_e t_d) & -\frac{1}{3}t_e(t_e(2s_d - 3t_d) + s_e t_d) \\ \frac{1}{3}(s_e(s_d - 2t_d) + t_e(3t_d - 2s_d)) & \frac{1}{3}(s_e - t_e)(t_e(2s_d - 3t_d) + s_e t_d) & \frac{1}{3}(s_d - t_d)(t_e(s_d - 3t_d) + 2t_e t_d) \\ (s_e - t_e)(s_d - td) & -(s_e - t_e)^2(s_d - td) & -(s_e - t_e)(s_d - td)^2 \end{pmatrix}$$

3) If $d_1 \neq 0$ and $3d_2^2 - 4d_1d_3 = 0$, it is a **cusp with inflection at infinity**. Generally, a cusp curve has one inflection point and one cusp point lying on $\tilde{k}$ and $\tilde{l}$ and $\tilde{m}$ are the identical tangent line passing through the cusp point and $\tilde{n}$ is the tangent line passing through the inflection point. The two roots of the quadratic portion of $I(t,s)$ are equal

$$(t_d,s_d) = (d_2\ ,\ 2d_1) \tag{6}$$

and $\mathbf{k}$, $\mathbf{l}$, and $\mathbf{m}$ can determined with same equation with serpentine.

4) If $d_1 = 0,\ d^2 \neq 0$, it is a **cusp with cusp at infinity**. The parameters of $I(t,s)$ for the inflection point is

$$\begin{aligned}(t_l,s_l) &= (d_3\ ,\ 3d_2)\\ (t_m,s_m) &= (1\ ,\ 0)\end{aligned} \tag{7}$$

and has a double root at parametric infinity. $\mathbf{k}$, $\mathbf{l}$, and $\mathbf{m}$ are determined as:

$$\begin{pmatrix} t_l & t_l^3 & 1 \\ t_l - \frac{1}{3}s_l & t_l^2(t_l - s_l) & 1 \\ t_l - \frac{2}{3}s_l & t_l^2(t_l - s_l)^2 & 1 \\ t_l - s_l & (t_l - s_l)^3 & 1 \end{pmatrix} \tag{8}$$

5) If $d_1 = d_2 = 0,\ d_3 \neq 0$, it is a **quadratic curve** and $\mathbf{k}$, $\mathbf{l}$, and $\mathbf{m}$ are expressed as:

$$\begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} \\ \frac{2}{3} & \frac{1}{3} & \frac{2}{3} \\ 1 & 1 & 1 \end{pmatrix} \tag{9}$$

6) If $d_1 = d_2 = d_3 = 0$, it is a **Line** i.e. all four control points are collinear, and we directly align the line with an edge.

We want to always have $\mathbf{f}(\overline{p}) < 0$ when the point is inside the curve. Therefore, after computing k, l, and m for all control points, we select an interior point $\overline{P}$ and compute its corresponding $k_{\overline{P}}$, $l_{\overline{P}}$, and $m_{\overline{P}}$. If $\mathbf{f}(\overline{P}) > 0$, all k's and l's of control points times -1.