Article

Lines of Curvature for Log Aesthetic Surfaces Characteristics Investigation

R.U. Gobithaasan ^{1,*} , Yee Meng Teh ², Kenjiro T. Miura ³ and Wen Eng Ong ² ¹ Special Interest Group on Modelling & Data Analytics, Faculty of Ocean Engineering Technology and Informatics, University Malaysia Terengganu, Kuala Nerus 21030, Malaysia² School of Mathematical Sciences, Universiti Sains Malaysia, Minden 11800, Malaysia; veostly_0405@hotmail.com (Y.M.T.); weneng@usm.my (W.E.O.)³ Graduate School of Science & Technology, Shizuoka University, Hamamatsu, Shizuoka 432-8561, Japan; miura.kenjiro@shizuoka.ac.jp

* Correspondence: gr@umt.edu.my

Abstract: Lines of curvatures (LoCs) are curves on a surface that are derived from the first and second fundamental forms, and have been used for shaping various types of surface. In this paper, we investigated the LoCs of two types of log aesthetic (LA) surfaces; i.e., LA surfaces of revolution and LA swept surfaces. These surfaces are generated with log aesthetic curves (LAC) which comprise various families of curves governed by α . First, since it is impossible to derive the LoCs analytically, we have implemented the LoC computation numerically using the Central Processing Unit (CPU) and General Processing Unit (GPU). The results showed a significant speed up with the latter. Next, we investigated the curvature distributions of the derived LoCs using a Logarithmic Curvature Graph (LCG). In conclusion, the LoCs of LA surface of revolutions are indeed the duplicates of their original profile curves. However, the LoCs of LA swept surfaces are LACs of different shapes. The exception to this is when this type of surface possesses LoCs in the form of circle involutes.

Keywords: log aesthetic curves; log aesthetic surface; LoC; GPU; CUDA

Citation: Gobithaasan, R.; Teh, Y.M.; Miura, K.T.; Ong, W.E. Lines of Curvature for Log Aesthetic Surfaces Characteristics Investigation.

Mathematics **2021**, *9*, 2699. <https://doi.org/10.3390/math9212699>

Academic Editor: Maria Lucia Sampoli

Received: 15 September 2021

Accepted: 20 October 2021

Published: 24 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Curve and surface design for shipbuilding started during Roman times [1]. The progress of representing this in the form of mathematical equations and algorithms increased drastically with the introduction of numerical control (NC) to handle milling machines in the 1950s. In the 1960s, automobile and aircraft designs were the two key fields employing mathematicians to work on curves used for effective and efficient design.

Two classical curves used for computer aided design (CAD) and computer aided manufacturing (CAM) are Bezier curves and B-Splines [2]. Later, Nonuniform Rational B-Spline (NURBS) became the de facto standard for CAD/CAM since it can represent conics analytically [3]. However, these curves cannot be used directly for manufacturing due to their complex forms of curvature; hence, various fairing algorithms were devised to reduce the oscillation in their curvature profile [4–6]. Spirals that have monotonic curvature profiles became a good candidate for use in aesthetic design [7–9]. In 2005, Log Aesthetic Curve (LAC) was first introduced. This curve consists of spiral families, e.g., clothoid, logarithmic spiral, circle involute and Nielsen's spiral [10].

LAC was first proposed by Miura [10], by representing a Logarithmic Curvature of Graph (LCG) as a linear equation with α as its gradient. While the curvature profile involves the second derivatives of curves, LCG involves the third derivatives, hence making it suitable as a higher order shape interrogation tool [11]. The LAC family has a monotonic curvature profile which can be used to represent well known spirals, e.g., clothoid, logarithmic spiral, circle involute and Nielsen's spiral. Research in LAC started to progress rapidly after Yoshida and Saito [12] proposed a G^1 algorithm to interactively render LACs that are

suitable for designing aesthetic shapes. The computational time for rendering LACs can be reduced by representing LAC in the analytic form, i.e., Incomplete Gamma Function [13], computing numerically using the classical Runge-Kutta [14] and adaptive methods [15]. It has been shown that these adaptive methods can render results faster compared to Incomplete Gamma Function [15] whilst maintaining accuracy for Computer-Aided Design (CAD) environments.

Recently, there have been variations in aesthetic curves introduced by numerous researchers. In 2012, Ziatdinov [16] proposed a family of super-spirals by generalizing LACs and developed the super-spiraloid by using the equation for surface of revolution. In 2020, Inoguchi et al. [17] expanded the super-spiral by means of similarity geometry and proposed a new class of planar curves with monotone curvature in terms of Tricomi confluent hypergeometric function. Recently, Crăciun et al. proposed an ε -neutral curve and an ε -aesthetic curve based on the approximation of LCG gradient [18].

Variations of aesthetic surfaces include the work of Inoue et al. [19] who applied LAC as profile curves to generate a log aesthetic (LA) surface. In 2014, Kineri et al. [20] proposed LAC as a planar feature curve to design bi-cubic B-spline surfaces. In 2018, Suzuki et al. [21] proposed a minimum variation log-aesthetic surface as the new definition of a log aesthetic surface with the aid of a log aesthetic filter.

There are many tangent directions on a surface that can be used to generate a curve at a point which has the same surface normal direction. Hence, many possible surface normal curvatures can be obtained in a particular surface normal direction. Of all the tangent directions to the point, the two tangent directions with the maximum or minimum surface normal curvature (except at umbilical and planar points) are called principal directions. In other words, the maximum or minimum surface normal curvatures at that point are called the principal curvatures. A line of curvature (LoC) is a curve on a surface with tangents in the principal direction at that point. Joo et al. [22] presented an algorithm for computing the surface's LoCs and their respective curvature and torsion. Besides using the LoC as a shape interrogation tool [23], it can also function as a guideline or profile curve that can be used to shape surfaces. Examples of work which employs LoCs for design include the fabrication of freeform objects [24], plate forming for shipbuilding [25], and designing carbon fiber reinforced plastic automobile parts and marine propeller blades [26].

LoCs on typical surfaces such as hyperbolic paraboloid, elliptic paraboloid and others can be obtained by solving an initial value problem (IVP) using numerical methods. When it comes to generating LoCs on LA surfaces, there is an additional IVP that has to be solved numerically. The additional IVP of the LAC equation itself increases the computation time needed to generate LoCs for LAC surfaces. One way to reduce the computation time is by means of graphical processing units (GPU) to solve the ordinary differential equations (ODE) [27].

Compute Unified Device Architecture, well known as CUDA, is a parallel computing platform and API invented by NVIDIA for general GPUs. This technology is now readily available in various applications, including in high level scientific computation programs, e.g., Mathematica [28]. There are many success stories of CUDA implementations in the field of CAD, e.g., NURBS computations [29], moment computations [30] and interactive NURBS rendering [31] on GPU. In this work, we implemented CUDA API which is available in Mathematica with a computer specification of Intel® Core™ 2 Quad CPU Q8400 @ 2.66 GHz 2.67 GHz with 12GB RAM and a graphic card GeForce GT 730 with 2.0GB total memory.

The rest of the paper is arranged as follows. The subsequent two sections introduce the necessary LAC and LoC equations used for the study. This is followed by the generation of LA surface of revolution and LA swept surface. Subsequently, we computed the speed of LoC generation in the CPU and GPU, and analyzed the characteristics of LoCs on LA surfaces. Finally, the features of LoCs are elaborated before concluding this work.

2. Log Aesthetic Curves

A linear representation of a Logarithmic Curvature Graph (LCG) is the fundamental equation to describe LACs. The equation for LCG which has a slope, α , is shown below [10]:

$$\log\left(\rho_L \frac{ds}{d\rho}\right) = \alpha \log(\rho_L) + C, \quad (1)$$

where s is the arc-length of LAC, ρ_L is radius of curvature and C is a constant. After differentiation and simplification of Equation (1), we obtain:

$$\frac{ds}{d\rho_L} = \frac{\rho_L^{\alpha-1}}{\Lambda}, \quad (2)$$

where Λ is the extra degree of freedom of LAC and $0 < \Lambda < \infty$. Upon integrating Equation (2), we have:

$$s = \begin{cases} \frac{\log \rho_L - \log \rho_0}{\Lambda} & , \alpha = 0 \\ \frac{(\rho_L - \rho_0)^\alpha}{\alpha \Lambda} & , otherwise' \end{cases} \quad (3)$$

$$\rho_L = \begin{cases} \rho_0 e^{\Lambda s} & , \alpha = 0 \\ (\rho_0^\alpha + \Lambda \alpha s)^{\frac{1}{\alpha}} & , otherwise' \end{cases} \quad (4)$$

where ρ_0 is the initial radius of curvature of LAC, and ρ_L varies from 0 to ∞ . The arc-length, s , has an upper boundary and lower boundary which depends on the α . The curvature of LAC is shown below:

$$\kappa_L = \begin{cases} \frac{1}{\rho_0 e^{\Lambda s}} & , \alpha = 0, \\ (\rho_0^\alpha + \Lambda \alpha s)^{-\frac{1}{\alpha}} & , otherwise \end{cases} \quad (5)$$

By setting torsion, $\tau(s) = 0$ in the case of planar curve, we can compute LACs by solving the Frenet-Serret formula:

$$\begin{aligned} \mathbf{C}'(s) &= \mathbf{t}(s) \\ \mathbf{t}'(s) &= \kappa(s) \mathbf{n}(s) \\ \mathbf{n}'(s) &= -\kappa(s) \mathbf{t}(s) \end{aligned} \quad (6)$$

Table 1 shows the upper boundary and lower boundary of s with respect to α derived by Yoshida & Saito [12]. Since $\rho_0^\alpha \in [0, \infty]$, we must choose a suitable value of α in order to obtain a well-defined LAC segment.

Table 1. The lower boundary and upper boundary of s .

α	Lower Boundary of s	Upper Boundary of s
$\alpha < 0$	-	$-\frac{\rho_0^\alpha}{\alpha \Lambda}$
$\alpha = 0$	-	-
$\alpha > 0$	$-\frac{\rho_0^\alpha}{\alpha \Lambda}$	-

3. Lines of Curvature

Assuming $\mathbf{R}(u, v) = \mathbf{R}(x(u, v), y(u, v), z(u, v))$ is a parametric surface, the first and second fundamental equations of the surface are [12,13]:

$$I(u, v) = E\dot{u}^2 + 2F\dot{u}\dot{v} + G\dot{v}^2, \quad (7)$$

$$II(u, v) = L\dot{u}^2 + 2M\dot{u}\dot{v} + N\dot{v}^2, \quad (8)$$

where $E = \mathbf{R}_u \cdot \mathbf{R}_u$, $F = \mathbf{R}_u \cdot \mathbf{R}_v$, $G = \mathbf{R}_v \cdot \mathbf{R}_v$, $L = \mathbf{N} \cdot \mathbf{R}_{uu}$, $M = \mathbf{N} \cdot \mathbf{R}_{uv}$, $N = \mathbf{N} \cdot \mathbf{R}_{vv}$, $\mathbf{N} = \frac{\mathbf{R}_u \times \mathbf{R}_v}{|\mathbf{R}_u \times \mathbf{R}_v|}$, $\dot{u} = \frac{du}{dt}$ and $\dot{v} = \frac{dv}{dt}$. The Gaussian curvature (K), mean curvature (H), and principal curvature (κ_p) are stated in Equations (9)–(11), respectively.

$$K = \frac{LN - M^2}{EG - F^2}, \quad (9)$$

$$H = \frac{EN + GL - 2FM}{2(EG - F^2)}, \quad (10)$$

$$\kappa_p = H \pm \sqrt{H^2 - K}. \quad (11)$$

To generate LoCs on the surface, we need to solve the IVP below numerically:

$$u'(s) = \frac{dt}{ds} \times \dot{u}(t) = \begin{cases} \eta(M - \kappa_p F) & |L - \kappa_p E| \geq |N - \kappa_p G| \\ \mu(N - \kappa_p G) & |L - \kappa_p E| < |N - \kappa_p G| \end{cases}, \quad (12)$$

$$v'(s) = \frac{dt}{ds} \times \dot{v}(t) = \begin{cases} -\eta(L - \kappa_p E) & |L - \kappa_p E| \geq |N - \kappa_p G| \\ -\mu(M - \kappa_p F) & |L - \kappa_p E| < |N - \kappa_p G| \end{cases}, \quad (13)$$

where κ_p is the principal curvature representing either the maximum principal or the minimum principal curvature. The non-zero factor (η and μ) can be obtained by normalizing the first fundamental equation as shown below:

$$\frac{dt}{ds} = \frac{\pm 1}{\sqrt{E\dot{u}(t)^2 + 2F\dot{u}(t)\dot{v}(t) + G\dot{v}(t)^2}}. \quad (14)$$

The sign of $\frac{dt}{ds}$ can be determined from Equation (14) [22]. LoCs can be generated by solving the IVP (Equation (12) and (13)) using numerical methods such as the Runge-Kutta method and substituting their values into the original surface function. To investigate the characteristics of LoCs, Joo et al. [22] proposed a novel method to compute the curvature and torsion of LoCs. In order to calculate the LoCs, we need first to compute the geodesic curvature of LoCs. Equation (15) represents the proposed method to compute u'' , v'' and the geodesic curvature, κ_g .

$$\begin{bmatrix} u'' \\ v'' \\ \kappa_g \end{bmatrix} = \begin{bmatrix} E & F & -(\mathbf{U} \cdot \mathbf{R}_u) \\ F & G & -(\mathbf{U} \cdot \mathbf{R}_v) \\ -\dot{v}(t) & \dot{u}(t) & 0 \end{bmatrix}^{-1} \begin{bmatrix} -(\gamma_1 \cdot \mathbf{R}_u) \\ -(\gamma_1 \cdot \mathbf{R}_v) \\ \delta_1 \end{bmatrix}, \quad (15)$$

where

$$\mathbf{U} = \mathbf{N} \times \mathbf{t} = \mathbf{N} \times \mathbf{c}'(s),$$

$$\gamma_1 = \mathbf{R}_{uu}u'^2 + 2\mathbf{R}_{uv}u'v' + \mathbf{R}_{vv}v'^2,$$

$$\dot{v}(t) = \begin{cases} -(L - \kappa_p E) & \text{if } |L - \kappa_p E| \geq |N - \kappa_p G| \\ -(M - \kappa_p F) & \text{if } |L - \kappa_p E| < |N - \kappa_p G| \end{cases},$$

$$\dot{u}(t) = \begin{cases} M - \kappa_p F & \text{if } |L - \kappa_p E| \geq |N - \kappa_p G| \\ N - \kappa_p G & \text{if } |L - \kappa_p E| < |N - \kappa_p G| \end{cases},$$

$$\delta_1 = \begin{cases} -(L' - \kappa'_p E - \kappa_p E')u' - (M' - \kappa'_p F - \kappa_p F')v' & \text{if } |L - \kappa_p E| \geq |N - \kappa_p G| \\ -(M' - \kappa'_p F - \kappa_p F')u' - (N' - \kappa'_p G - \kappa_p G')v' & \text{if } |L - \kappa_p E| < |N - \kappa_p G| \end{cases}.$$

The curvature can be calculated using the principal curvature and geodesic curvature, as shown below:

$$\kappa = \sqrt{\kappa_p^2 + \kappa_g^2}. \quad (16)$$

Furthermore, the torsion and first order curvature's derivative of curvature on the surface can be computed using the following equation:

$$\begin{bmatrix} u''' \\ v''' \\ \kappa' \\ \tau \end{bmatrix} = \begin{bmatrix} E & F & -(\mathbf{n} \cdot \mathbf{R}_u) & -\kappa(\mathbf{b} \cdot \mathbf{R}_u) \\ F & G & -(\mathbf{n} \cdot \mathbf{R}_v) & -\kappa(\mathbf{b} \cdot \mathbf{R}_v) \\ (\mathbf{n} \cdot \mathbf{R}_u) & (\mathbf{n} \cdot \mathbf{R}_v) & -1 & 0 \\ -\dot{v}(t) & \dot{u}(t) & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} -(\gamma_2 \cdot \mathbf{R}_u) - \kappa^2(\mathbf{t} \cdot \mathbf{R}_u) \\ -(\gamma_2 \cdot \mathbf{R}_v) - \kappa^2(\mathbf{t} \cdot \mathbf{R}_v) \\ -(\mathbf{n} \cdot \gamma_2) \\ \delta_2 \end{bmatrix} \quad (17)$$

where

$$\mathbf{t} = \mathbf{c}'(s),$$

$$\mathbf{n} = \frac{\mathbf{c}''(s)}{\kappa},$$

$$\begin{aligned}
\mathbf{b} &= \mathbf{t} \times \mathbf{n} = \frac{\mathbf{c}'(s) \times \mathbf{c}''(s)}{\kappa}, \\
\gamma_2 &= \mathbf{R}_{uuu}u'^3 + 3\mathbf{R}_{uuv}u'^2v' + 3\mathbf{R}_{uvv}u'v'^2 + \mathbf{R}_{vvv}v'^3 + 3(\mathbf{R}_{uu}u'u'' + \mathbf{R}_{uv}(u'v'' + u''v') + \mathbf{R}_{vv}v'v''), \\
\delta_2 &= \begin{cases} - \begin{pmatrix} 2(L' - \kappa'_p E - \kappa_p E')u'' + 2w(M' - \kappa'_p F - \kappa_p F')v'' \\ + (L'' - \kappa''_p E - 2\kappa'_p E' - \kappa_p E'')u' + \\ (M'' - \kappa''_p F - 2\kappa'_p F' - \kappa_p F'')v' \end{pmatrix} & \text{if } |L - \kappa_p E| \geq |N - \kappa_p G| \\ - \begin{pmatrix} 2(M' - \kappa'_p F - \kappa_p F')u'' + 2w(N' - \kappa'_p G - \kappa_p G')v'' \\ + (M'' - \kappa''_p F - 2\kappa'_p F' - \kappa_p F'')u' + \\ (N'' - \kappa''_p G - 2\kappa'_p G' - \kappa_p G'')v' \end{pmatrix} & \text{if } |L - \kappa_p E| < |N - \kappa_p G| \end{cases}.
\end{aligned}$$

4. CUDA Programming in Wolfram Mathematica

In this paper, we used parallel programming to speed up the computation time for generating LoCs on the surface. The Mathematica code is a simplified procedure for CUDA implementation, as it internalizes the memory allocation, copies, processes and synchronizes. There are four steps towards CUDA implementation [27]:

1. check CUDA system's availability;
2. the CUDA code is cached to a binary file optimized for the GPU selection;
3. the compiled code is cached to avoid future compilation;
4. a check is performed as to whether the kernel exists in the compiled code.

Hence, the only thing the user needs to do is write the CUDA kernels as shown in the Appendix A. To note, all the codes must be placed into a string (*CUDAKernelCode*) so that it can be executed in Mathematica. Function *func2*(...) is a device function which can be called from *__device__* or *__global__* function but cannot be called from Mathematica's language. Function *func*(float *sum, mint listSize) is a global function which can be called from and returned to CPU. The *sum* is returned in the form of array and *mint* is the Wolfram Language integer that is similar to *int*. The *index* is calculated based on the current thread to be used for execution. If (*index* < *listSize*) is to ensure that the output buffer is not overwritten. Function *func2(index)* is evaluated at *index* and stored at *sum[index]*. Next, *CUDAFunctionLoad* is used to load the program which passes the kernel code as a first argument, the kernel name to be run ("*func*") as a second argument, the function parameters as a third argument and the last argument is the block size. The result is stored in *Kresult*. *CUDAMemoryAllocate* is used to allocate CUDA's memory to store the result. The example shows that 1024 buffers are created to store the result. *KResult[bufferK, 1024]* shows how the kernel is executed and we can retrieve the result by using *CUDAMemoryGet[bufferK]*. The buffer must be freed up by the user manually by using *CUDAMemoryUnload[bufferK]*. The complete implementation of CUDA coding in Mathematica for this paper is readily available in GitHub [32].

5. Log Aesthetic Surface of Revolution

For profile curve $C = \{f(u), g(u)\}$, that is a parametric curve defined in *xz*-plane, the general equation of the surface of revolution, S_{Rs} revolving around *z*-axis is:

$$S_{Rs} = \left\{ \begin{array}{l} f(u) \cos(v), \\ f(u) \sin(v), \\ g(u) \end{array} \right\}, \quad (18)$$

Assume that the radius from the center of the surface of revolution is *m*, then we have the following equation for surface of revolution as:

$$S_T = \left\{ \begin{array}{l} (m + f(u)) \cos(v), \\ (m + f(u)) \sin(v), \\ g(u) \end{array} \right\}. \quad (19)$$

For example, if *C* is a circle, then S_T is a torus with radius *m*.

The first and second fundamental equation's coefficients of this surface are:

$$E = \dot{x}(u)^2 + \dot{y}(u)^2,$$

$$F = 0,$$

$$G = (m + x(u))^2,$$

$$L = (\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)) / \sqrt{\dot{x}(u)^2 + \dot{y}(u)^2},$$

$$M = 0,$$

$$N = ((m + x(u))\dot{y}(u)) / \sqrt{\dot{x}(u)^2 + \dot{y}(u)^2},$$

where $\dot{x}(u) = \frac{dx}{du}$, $\ddot{x}(u) = \frac{d^2x}{du^2}$, $\dot{y}(u) = \frac{dy}{du}$, and $\ddot{y}(u) = \frac{d^2y}{du^2}$, respectively. We can choose any profile curve to generate this surface. The Super-spiraloid produced in [4] also has these settings, if one were to choose super-spiral as their profile curve.

Figure 1 shows LA surfaces of revolution by setting LAC as its profile curve with various types of α . The thick black line is the LAC profile curve, C , used to generate these LA surfaces of revolution. Note that we set $m = 3$, $u = [0, 2]$ and $v = [0, \pi/3]$ for the entire computation. Since $F = M = 0$, the LoCs on LA surface of revolution are meridians (red) and parallel curves (blue) on the surface and they are orthogonal to each other. To reiterate, the red LoCs are duplicates of the original LAC used as the profile curve and the blue LoCs are circles with different radius of curvature.

Figure 1. Curvature lines on LA surfaces of revolution with various types of LAC.

Characteristics of LoCs for LA Surface of Revolution

The numerical methods used to compute the LoCs in CPU are the classical Runge-Kutta method (RK4) [14] with fixed step-size 0.001 and the Dormand-Prince method (DP) [15]. For the GPU, we used the classical Runge-Kutta method (RK4) [14] with a fixed step-size of 0.001. DP is faster for LAC computation when compared to RK4 [15], Gaussian Kronrod [14], and Incomplete Gamma Function [13]. However, the DP's implementation in GPU needs extra effort. Since RK4 is computationally more efficient, we used RK4 for the GPU comparison in this work. Table 2 compares the CPU and GPU performance metrics by calculating the computation time to generate 22 LoCs (11 red and 11 blue lines) on the surface. We repeated these computations five times and the average computation time in CPU and GPU are 167.257 (DP), 1188.91(RK) and 8.28301 (RK) seconds, respectively. It is apparent that GPU greatly reduces the computation time for LoC generation.

Table 2. LA surfaces of revolution and their average computation time (in seconds).

α	Λ	ρ_0	CPU(DP)	CPU(RK4)	GPU(RK4)
2	2	1	161.053	1119.91	7.42628
1	1	1	162.871	1110.93	7.17346
0	0.6	1	162.171	1297.61	7.83301
-1	0.35	1	162.471	1118.24	6.93649
-2	0.15	1	187.718	1297.86	12.0458

Figure 2 shows the characteristics of LoCs of LA surfaces of revolution that correspond to the various types of α in Table 2.

Figure 2. Characteristics of LoCs on LA surfaces of revolution with various types of α : (a) Curvature of LoCs; (b) Derivative of curvature of LoCs.

The red LoCs on the surface are identical to each other since these are the duplicates of the original LAC. Hence, they have the same curvature and torsion profile. Figure 2a,b show that the curvature of the LoCs is indeed the curvature of its respective LACs in xz -plane. When $\alpha = -1$, the curvature of the LoCs is linear, hence its derivative is a constant, which has the same curvature distribution as a clothoid.

Figure 3 depicts the LoCs on LA surface of revolution with $\alpha = 2$ as a square pipe to keep track of the torsion. Figure 3a indicates the Frenet-Serret frames at each point. Figure 3b shows a close-up view of the surface and Figure 3c represents the LoCs on the LA surface of revolution. Based on Figure 3b, there is no twist along the LoCs, which is consistent with the fact that each curvature line has 0 torsion.

Figure 3. Visualization of LoCs as square pipes on LA surface of revolution for $\alpha = 2$. (a) Frenet-Serret frame; (b) Close-up view of LA surface of revolution; (c) Curvature lines on LA surface of revolution.

6. Log Aesthetic Swept Surface

In this section, the characteristics of LoCs of LA swept surface are investigated. First, a LAC is swept along another LAC to form a complete LA swept surface. The general equation of the swept surface is obtained as we sweep along the path curve, $C_1(u)$, with the trajectory of the profile curve, $C_2(v)$ [33]:

$$S_{\text{sweep}}(u, v) = \left(T_1(\theta(u)) C_2(v)^T \right)^T + C_1(u) \quad (20)$$

where T_1 is a rotation matrix. Suppose that the path curve, $C_1(u) = \{x_1(u), y_1(u), 0\}$ and profile curve, $C_2(v) = \{0, -x_2(v), y_2(v)\}$ are LACs, then we sweep curve $C_2(v)$ along curve $C_1(u)$ to form a complete LA surface. The equation of the LA surface can be computed by applying the details into Equation (20) to form:

$$S_{LA_s}(u, v) = \left(\begin{pmatrix} \cos(\theta(u)) & -\sin(\theta(u)) & 0 \\ \sin(\theta(u)) & \cos(\theta(u)) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ -x_2(v) \\ y_2(v) \end{pmatrix} \right)^T + \{x_1(u), y_1(u), 0\} \quad (21)$$

where $\theta(u) = \cos^{-1}(\{1, 0, 0\} \cdot \dot{C}_1(u))$ and $\dot{C}_1(u) = \left\{ \frac{\dot{x}_1(u)}{\sqrt{\dot{x}_1(u)^2 + \dot{y}_1(u)^2}}, \frac{\dot{y}_1(u)}{\sqrt{\dot{x}_1(u)^2 + \dot{y}_1(u)^2}}, 0 \right\} = \{t_{11}(u), t_{12}(u), 0\}$ is the normalized tangent of $C_1(u)$. The general equation of LA surface is

$$S_{LA_s}(u, v) = \left\{ x_1(u) + x_2(v)\sqrt{1 - t_{11}(u)^2}, y_1(u) - x_2(v)t_{11}(u), y_2(v) \right\}. \quad (22)$$

Since $\dot{C}_1(u)$ is a normalized tangent, then we have $t_{11}(u)^2 + t_{12}(u)^2 = 1$. Hence, the LA surface can be simplified as follows:

$$S_{LA_s}(u, v) = \{x_1(u) + x_2(v)t_{12}(u), y_1(u) - x_2(v)t_{11}(u), y_2(v)\}. \quad (23)$$

Next, the first and second fundamental equation's coefficients of LA swept surface are:

$$\begin{aligned} E &= \left(t_{12}(u) - x_2(v)t_{11}(u) \right)^2 + \left(t_{11}(u) + x_2(v)t_{12}(u) \right)^2, \\ F &= x_2(v)t_{21}(v) \left(t_{11}(u)t_{11}(u) + t_{12}(u)t_{12}(u) \right) = 0, \\ G &= t_{11}(u)^2 t_{21}(v)^2 + t_{21}(v)^2 t_{12}(u)^2 + t_{22}(v)^2, \\ S_{LA_{su}} \times S_{LA_{sv}} &= \left\{ \begin{aligned} &t_{22}(v) \left(t_{12}(u) - x_2(v)t_{11}(u) \right), \\ &-t_{22}(v) \left(t_{11}(u) - x_2(v)t_{12}(u) \right), \\ &-t_{21}(v) \left(t_{11}(u)^2 + t_{12}(u) \left(t_{12}(u) - x_2(v)t_{11}(u) \right) + x_2(v)t_{11}(u)t_{12}(u) \right) \end{aligned} \right\}, \\ L &= \frac{t_{22}(v) \left(\left(t_{11}(u) + x_2(v)t_{12}(u) \right) \left(-t_{12}(u) + x_2(v)t_{11}(u) \right) + \left(t_{12}(u) - x_2(v)t_{11}(u) \right) \left(t_{11}(u) + x_2(v)t_{12}(u) \right) \right)}{|S_{LA_{su}} \times S_{LA_{sv}}|}, \\ M &= \frac{t_{21}(v)t_{22}(v) \left(t_{11}(u)t_{11}(u) + t_{12}(u)t_{12}(u) \right)}{|S_{LA_{su}} \times S_{LA_{sv}}|} = 0, \\ N &= \frac{\left(t_{11}(u)^2 + t_{12}(u) \left(t_{12}(u) - x_2(v)t_{11}(u) \right) + x_2(v)t_{11}(u)t_{12}(u) \right) \left(t_{22}(v)t_{21}(v) - t_{21}(v)t_{22}(v) \right)}{|S_{LA_{su}} \times S_{LA_{sv}}|}, \end{aligned}$$

and $\dot{C}_2(v) = \left\{ 0, -\frac{\dot{x}_2(v)}{\sqrt{\dot{x}_2(v)^2 + \dot{y}_2(v)^2}}, \frac{\dot{y}_2(v)}{\sqrt{\dot{x}_2(v)^2 + \dot{y}_2(v)^2}} \right\} = \{0, -t_{21}(u), t_{22}(u)\}$ is also a normalized tangent of $C_2(v)$. Take note that $F = M = 0$, indicating that the LoCs are similar to parametric lines.

As a numerical example, a LA surface is generated by setting $\alpha_1 = 2$, and $\Lambda_1 = 0.1$. As shown in Figure 4, the initial radius of curvature, $\rho_{10} = 1$ for curve $C_1(u)$ and $\alpha_2 = 2$, followed by $\Lambda_2 = 0.1$ and initial radius of curvature, $\rho_{20} = 0.5$ for curve $C_2(v)$. Based on Figure 4b, the green curve represents curve $C_1(u)$ while the black curve represents curve $C_2(v)$. As for Figure 4c, the zebra map of the LA surface indicates a smooth surface.

Figure 4. LA swept surface: (a) Top view; (b) Side view; (c) Zebra map on LA swept surface; (d) One part of LoCs on the LA swept surface.

Characteristics of LoCs for LA Swept Surface

In this section, we investigate one part of the LA swept surface as shown in Figure 4d. The red lines (labeled as L_0 to L_5) are the LoCs in maximum principal direction and the blue lines (labeled as D_0 to D_5) are the LoCs in minimum principal direction. First, we compared the computation time of 12 LoCs (consisting of six lines in the maximum principle direction and six lines in the minimum principle direction) on the CPU and GPU. Figure 5 shows swept LA surfaces with different α and its respective LoCs.

Figure 5. LoCs on various LA swept surfaces.

Figure 6a depicts the LoCs on the LA swept surface ($\alpha = -1$) in square pipe form whereas Figure 6b is the close-up view of this. Since the torsion of LoCs is zero, there is no twist along the LoCs.

Figure 6. Visualization of LoCs on the LA surface ($\alpha = -1$): (a) Curvature lines on LA surface; (b) Close up view of (a).

Table 3 shows the computation time for generating LoCs on various LA swept surfaces. The average computation time for CPU(DP), and GPU is 469.763 s, and 2.38576 s, respectively. The GPU speeds up the LoC computation time by about 197 times faster than the CPU.

Table 3. Various types of LA Surface and their average LoC's computation time (in seconds).

α	CPU (DP)	GPU(RK4)
2	488.052	2.36544
1	469.997	2.22297
0	463.187	2.29104
-1	459.865	2.11680
-2	467.716	2.93257

Figure 7 shows the curvature (κ) and derivative of curvature (κ') for LoCs in the maximum principal direction (L_0 to L_5) for LA swept surfaces, as illustrated in Figure 4d. It is apparent that the curvature profiles and their derivatives of LoCs are monotonic.

We further plotted LCG to verify that these LoCs are indeed LACs as shown in Figure 8. The LCG plots share similar colors with the LoC, as seen in Figure 7. The LCGs are all in the linear form with a fixed LCG gradient Δ_i as stated in the legend. Hence, all L_0 to L_5 LoCs on each LA swept surface are confirmed to be LACs. However, they are dissimilar to their LAC's original profile, except for the case of $\alpha = 2$ which represents the family of circle involutes. For the rest of α , the LCG gradients vary gradually as we sweep along another LAC profile curve.

Figure 9 depicts the curvature and its first derivative of the blue LoCs of LA swept surfaces as shown in Figure 4d. The LoCs (D_0 to D_5) of each LAC swept surface have the same curvature distribution as its original LAC profile curve.

Figure 10 shows its LCG plot which verifies that each set of minimum principal direction of LoCs is indeed similar to its original LAC profile curve.

Figure 7. Curvature and its derivative of curvature for maximum principal direction LoCs (L_0 to L_5) on LA swept surfaces.

Figure 8. LCG plot of LoCs in maximum principal direction on LA swept surfaces for $\alpha = 2$ (left) and $\alpha = -1$ (right).

Figure 9. The Curvature and its derivative in the minimum principal direction of LoCs on LA swept surfaces. (a) Curvature, κ , (b) Derivative of Curvature, κ' .

Figure 10. LCG plot of LoCs in the minimum principal direction on the LA swept surfaces.

7. Conclusions and Future Work

We presented two types of LA surface along with their respective LoCs in this paper. In order to speed up the computation time, we implemented LoC computation on the GPU, which resulted in the GPU performing approximately 20 times faster as compared to the CPU for LA surfaces of revolution. For LA swept surface, the GPU computation is 197 times faster than CPU. Next, we analysed the curvature and its derivatives of LoCs for both types of surfaces. As expected, the LoCs on a LA surface of revolution are also parametric curves in which the meridians are LACs of their profile curve, and the parallels are circular arcs. However, for LA swept surface, the curvature and its derivative of LoCs in the minimum directions are indeed LACs with original LAC profile. However, the LoCs for maximum principal directions are dissimilar except in the case of $\alpha = 2$. This indicates that the LoCs and parametric curves of LAC swept surface for this case are all circle involutes that did not change due to the sweep. Therefore, in the case where the designer wants uniform LoCs of the LAC swept surface, this can be achieved by setting $\alpha = 2$.

Our future work includes the implementation of LA space curves to generate a free form surface design, for example, using Coon's patch, and investigating its LoCs. We are also planning to extend this work by investigating the LoCs for surfaces developed with super-spirals [17].

Author Contributions: Conceptualization, R.G. and Y.M.T.; methodology, Y.M.T. and R.G.; software, Y.M.T.; validation, R.G. and K.T.M.; writing—Y.M.T., R.G. and K.T.M.; supervision, R.G. and W.E.O. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by JST CREST Grant Number JPMJCR1911. It was also supported JSPS Grant-in-Aid for Scientific Research (B) Grant Number 19H02048.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors acknowledge University Malaysia Terengganu/Universiti Sains Malaysia for providing software and facilities which was utilized for this research. Special thanks to the anonymous reviewers for their comments that helped us improve the readability of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Bare Basics of CUDA Implementation in Mathematica

```
CUDAKernelCode = "
__device__ float func2(float z)
{return z;}
__global__ void func(float * sum, mint listSize)
{
    mint index = threadIdx.x + blockIdx.x * blockDim.x;
    if (index < listSize)
        {sum[index] = func2((float) index) + func2((float) index);}
};
KResult = CUDAFunctionLoad[CUDAKernelCode,"func",{{"Float"},_Integer},1024];
bufferK = CUDAMemoryAllocate["Float", 1024];
KResult[bufferK, 1024];
CUDAMemoryGet[bufferK];
CUDAMemoryUnload[bufferK];
```

References

1. Farin, G. A History of Curves and Surfaces in CAGD. Available online: <http://www.farinhansford.com/gerald/papers/history.pdf> (accessed on 18 October 2021).
2. Farin, G. *Curves and Surfaces for Computer Aided Geometric Design*, 5th ed.; Academic Press: Cambridge, MA, USA, 2002.
3. Piegl, L.; Tiller, W. *The NURBS Book*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 1997.
4. Farin, G.; Rein, G.; Sapidis, N.; Worsey, A.J. Fairing cubic b-spline curves. *Comput. Aided Geom. Des.* **1987**, *4*, 91–103. [CrossRef]

5. Farin, G.; Sapidis, N. Curvature and the fairness of curves and surfaces. *IEEE Comput. Graph. Appl.* **1989**, *9*, 52–57. [[CrossRef](#)]
6. Sapidis, N.; Farin, G. Automatic fairing algorithm for b-spline curves. *Comput.-Aided Des.* **1990**, *22*, 121–129. [[CrossRef](#)]
7. Meek, D.S.; Walton, D.J. The use of cornu spirals in drawing planar curves of controlled curvature. *J. Comput. Appl. Math.* **1989**, *25*, 69–78. [[CrossRef](#)]
8. Walton, D.J.; Meek, D.S. A controlled clothoid spline. *Comput. Graph.* **2005**, *29*, 353–363. [[CrossRef](#)]
9. McCrae, J.; Singh, K. Sketching piecewise clothoid curves. *Comput. Graph.* **2009**, *33*, 452–461. [[CrossRef](#)]
10. Miura, K.T. A General Equation of Aesthetic Curves and its Self-Affinity. *Comput.-Aided Des. Applications.* **2006**, *3*, 457–464. [[CrossRef](#)]
11. Gobithaasan, R.U.; Miura, K.T. Logarithmic curvature graph as a shape interrogation tool. *Appl. Math. Sci.* **2014**, *8*, 755–765. [[CrossRef](#)]
12. Yoshida, N.; Saito, T. Interactive Aesthetic Curve Segment. *Vis. Comput.* **2006**, *22*, 896–905. [[CrossRef](#)]
13. Ziatdinov, R.; Yoshida, N.; Tae-wan, K. Analytic parametric equations of Log-aesthetic curves in terms of incomplete gamma functions. *Comput. Aided Geom. Des.* **2012**, *29*, 129–140. [[CrossRef](#)]
14. Gobithaasan, R.U.; Teh, Y.M.; Piah, A.R.; Miura, K.T. Rendering Log Aesthetic Curves via Runge-Kutta Method. *AIP Conf. Proc.* **2014**, *1605*, 319.
15. Gobithaasan, R.U.; Teh, Y.M.; Piah, A.R.; Miura, K.T. Generation of Log-aesthetic curves using adaptive Runge–Kutta methods. *Appl. Math. Comput.* **2014**, *246*, 257–262. [[CrossRef](#)]
16. Ziatdinov, R. Family of superspirals with completely monotonic curvature given in terms of Gauss hypergeometric function. *Comput. Aided Geom. Des.* **2012**, *29*, 510–518. [[CrossRef](#)]
17. Inoguchi, J.-I.; Ziatdinov, R.; Miura, K.T. A Note on Superspirals of Confluent Type. *Mathematics* **2020**, *8*, 762. [[CrossRef](#)]
18. Crăciun, I.; Popa, D.; Serdean, F.; Tudose, L. On Approximate Aesthetic Curves. *Symmetry* **2020**, *12*, 1394. [[CrossRef](#)]
19. Inoue, J.; Harada, T.; Hagihara, T. An Algorithm for Generating Log-Aesthetic Curved Surfaces and the Development of a Curved Surfaces Generation System using VR. In Proceedings of the International Association of Societies of Design Research, IASDR, Seoul, Korea, 18–22 October 2009; pp. 2513–2522.
20. Kineri, Y.; Endo, S.; Maekawa, T. Surface design based on direct curvature editing. *Comput.-Aided Des.* **2014**, *55*, 1–12. [[CrossRef](#)]
21. Suzuki, S.; Gobithaasan, R.U.; Salvi, P.; Usuki, S.; Miura, K.T. Minimum variation log-aesthetic surfaces and their application for smoothing free form shapes. *J. Comput. Des. Eng.* **2018**, *5*, 243–248. [[CrossRef](#)]
22. Joo, H.K.; Yazaki, T.; Takezawa, M.; Maekawa, T. Differential geometry properties of lines of curvature of parametric surfaces and their visualization. *Graph. Models* **2014**, *76*, 224–238. [[CrossRef](#)]
23. Patrikalakis, N.M.; Maekawa, T. Shape Interrogation for Computer Aided Design and Manufacturing. Springer: Berlin/Heidelberg, Germany, 2001.
24. Takezawa, M.; Imai, T.; Shida, K.; Maekawa, T. Fabrication of freeform objects by principal strips. *ACM Trans. Graph.* **2016**, *35*, 1–12. [[CrossRef](#)]
25. Takezawa, M.; Matsuo, K.; Maekawa, T. Control of lines of curvature for plate forming in shipbuilding. *Comput. Aided Geom. Des.* **2019**, *75*, 101785. [[CrossRef](#)]
26. Takezawa, M.; Otoguro, Y.; Matsuo, K.; Shibitani, T.; Sakurai, A.; Maekawa, T. Fabrication of doubly-curved CFRP shell structures with control over fiber directions. *Comput.-Aided Des.* **2021**, *136*, 103028. [[CrossRef](#)]
27. Wolfram Research, Inc. Mathematica. 2021 Ver. 12. Available online: <https://reference.wolfram.com/language/> (accessed on 2 October 2021).
28. Seen, W.M.; Gobithaasan, R.U.; Miura, K.T. GPU acceleration of Runge Kutta-Fehlberg and its comparison with Dormand-Prince method. *AIP Conf. Proc.* **2014**, *1605*, 16.
29. Krishnamurthy, A.; Khardekar, R.; McMains, S. Direct evaluation of NURBS curves and surfaces on the GPU. In Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling, ACM, Beijing, China, 4–6 June 2007.
30. Krishnamurthy, A.; McMains, S. Accurate GPU-accelerated surface integrals for moment computation. *Comput.-Aided Des.* **2011**, *43*, 1284–1295. [[CrossRef](#)]
31. Concheiro, R.; Amor, M.; Padrón, E.J.; Doggett, M. Interactive rendering of NURBS surfaces. *Comput.-Aided Des.* **2014**, *56*, 34–44. [[CrossRef](#)]
32. Meng, T.M.; Gobithaasan, R.U.; Miura, K.T. LoCs of LA Surfaces. Available online: <https://github.com/gob1thaasan/LoC-LASurfaces> (accessed on 2 October 2021).
33. Chang, K.-H. *e-Design*; Elsevier: New York, NY, USA, 2015.