МАТЕМАТИЧНЕ ТА КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ

MATHEMATICAL AND COMPUTER MODELING

UDC 004.94

IMPLICIT CURVES AND SURFACES MODELING WITH PSEUDO-GAUSSIAN INTERPOLATION

Ausheva N. M. – Dr. Sc., Professor, Head of the Department of Digital Technologies in Energy, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine.

Sydorenko Iu. V. – PhD, Associate Professor of the Department of Digital Technologies in Energy, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine.

Kaleniuk O. S. – PhD, Senior lecturer of the Department of Digital Technologies in Energy, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine.

Kardashov O. V. – Post-graduate student of the Department of Digital Technologies in Energy, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine.

Horodetskyi M. V. – Post-graduate student of the Department of Digital Technologies in Energy, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine.

ABSTRACT

Context. With the contemporary development of topological optimization, and parametric and AI-guided design, the problem of implicit surface representation became prominent in additive manufacturing. Although more and more software packages use implicit modeling for design, there is no common standard way of writing, storing, or passing a set of implicit surfaces or curves over the network. The object of the study is one of the possible ways of such representation, specifically: modeling implicit curves and surfaces using pseudo-Gaussian interpolation.

Objective. The goal of the work is the development of a modeling method that improved the accuracy of the implicit object representation wothout significant increase in memory used or processing time spent.

Method. One of the conventional ways to model an implicit surface would be to represent its signed distance function (SDF) with its values defined on a regular grid. Then a continuous SDF could be obtained from the grid values by means of interpolation. What we propose instead is to store not SDF values but the coefficients of a pseudo-Gaussian interpolating function in the grid, which would enable picking the exact interpolation points before the SDF model is written. In this way we achieve better accuracy in the regions we're interested the most in with no additional memory overhead.

Results. The developed method was implemented in software for curves in 2D and validated against several primitive implicit curves of different nanture: circles, sqaures, rectangles with different parameters of the model. The method has shown improved accuaracy in general, but there were several classes of corner cases found for which it deserves further development.

Conclusions. Pseudo-Gaussian interpolation defined as a sum of radial basis functions on a regular grid with points of interpolation defined in the proximity of the grid points generally allows to model an implicit surface more accurately than a voxel model interpolation does. The memory intake or computational toll isn't much different in these two approaches. However, the interpolating points selection strategy and the choice of the best modeling parameters for each particular modeling problem remain an open quesition.

KEYWORDS: surface representation, curve representation, implicit representation, pseudo-Gaussian function, regular grid, implicit surface modeling, implicit surface data format.

ABBREVIATIONS

3MF is a data format called 3D Manufacturing Format;

DICOM is a data format called Digital Imaging and Communications in Medicine;

HDF5 is a data format called Hierarchical Data Format;

HRBF a a Hermite radial basis function;

NRRD is a data format called Nearly Raw Raster Data;

RBF is a radial basis function;

SDF is a signed distance function;

SOP is a Service-Object Pair.

NOMENCLATURE

k is a distance between grid points;

M is a first dimension of a 3D or 2D grid;

n is an order of the pseudo-Gaussian sum continuity;

N is a second dimension of a 3D or 2D grid;

P is a third dimension of a 3D grid;

DOI 10.15588/1607-3274-2025-1-3

$P(x)$ is an algebraic polynomial that models a Gaussian function;

$pg(x)$ is a presudo-Gaussian function of a single variable;

$pgr_{ij}(x, y)$ is a radial basis function based on a pseugo-Gaussian function;

r is a range of pseudo-Gaussian basis functions;

$SDF(x, y)$ is a signed-distance fucntion representing an implicit curve to model;

$Spg(x, y)$ is a pseudo-Gaussian interpolating function;

(x_a, y_b) is an interpolating point in 2D;

(x_a, y_b, z_c) is an interpolating point in 3D;

(x_j, y_i) is a point of a 2D grid;

(x_j, y_i, z_l) is a point of a 3D grid.

INTRODUCTION

With the contemporary development of topological optimization, and parametric and AI-guided design, the problem of implicit surface representation became prominent in additive manufacturing. Although more and more software packages use implicit modeling for design, there is no common standard way of writing, storing, or passing a set of implicit surfaces or curves over the network. The straightforward way to represent an implicit surface would be to write down all the operations necessary for its signed distance function computation, but this would be too tedious to formalize and implement, besides the producer of the model design might not even want to disclose its computation method as it might be by itself a trade secret.

One of the usual ways to represent a signed distance function (SDF) of an implicit surface would be to write down its values defined on a regular grid. This is sometimes called a 3D image, a 3D bitmap, or a voxel model. Then a continuous SDF could be obtained from the grid values by means of interpolation. What we propose instead is to store not SDF values but the coefficients of a pseudo-Gaussian interpolating function, which allows us to pick the exact interpolation points before the SDF model is written. In this way we achieve better accuracy in the regions we're the most interested in (see Fig. 1.) with no additional memory overhead.

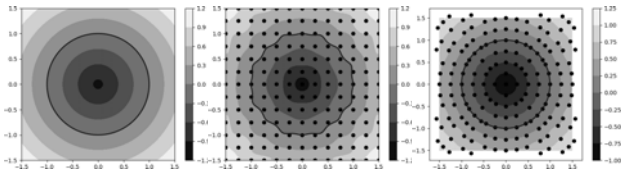

Figure 1 – An SDF of a circle, its model with values set on a regular grid, its model with interpolation points adjusted

The object of study is the modeling of implicit curves and surfaces with data on regular grids.

The subject of study is the pseudo-Gaussian interpolation in the context of implicit curves and surfaces modeling.

The purpose of the work is to increase the accuracy of implicit surface modeling while not increasing the memory intake of the model.

1 PROBLEM STATEMENT

Let $SDF(x, y, z)$ be a function that represents an implicit surface $SDF(x, y, z) = 0$. The value of the function, when not zero, is the signed distance to the abovementioned implicit surface.

Suppose we have a regular grid of points (x_j, y_i, z_l), where $j = 1..M$, $i = 1..N$, $l = 1..P$. Let distance between the closest grid points be k, and the cooridnate-wise minimal point of the grid: (x_0, y_0, z_0).

The problem of the implicit surface modeling $SDF(x, y, z) = 0$ with pseudo-Gaussian interpolation would be to define a pseudo-Gaussial inteprpolation function $Spg(x, y, z)$ that is close to the $SDF(x, y, z)$ within some range of tolerance on the 3D range $[x_0, x_0 + (M–1)k]\times[y_0, y_0 + (N–1)k]\times[z_0, z_0 + (P–1)k]$.

2 REVIEW OF THE LITERATURE

Since normally storing or transmitting the whole computation path of an implicit surface's SDF is impractical, the computation may even require going beyond the algebra of real numbers [1], we have to resort to some sort of an SDF model instead. The common way to represent an SDF with an easily serializable homogeneous data structure fit for transmission would be to use a voxel grid, also known as 3D-image, where each voxel is assigned a value or a set of values that help the receiving side model the initial surface. In this case, the problem of representation concerns the voxel access optimization [2] or the specific data assigned to each voxel, which can not only be an SDF value in a point but a gradient of the SDF in that very point as well [3].

In recent times, another common approach for implicit surface modeling is using a neural network trained to recreate the surface [4], [5]. For this approach, the problem of representation is then reduced to the problem of representation of the model's coefficients. It is not a solved problem, different networks allow users to balance the model precision and the model size differently.

We propose to use an approach that, in a way, combines the previous two. We store the coefficients of the model, but the model is not a neural network but an interpolating function, consisting of weighted radial functions, defined on the regular grid, similar to a voxel grid.

Using radial functions is a common practice in another adjacent problem – the problem of implicit surface reconstruction from a point cloud. For instance, a meta-analysis of implicit surface reconstruction via RBF interpolation methods performed by Mo Jiahui, Shou Huahao, and Chen Wei in 2022 [6] lists 125 published sources. A comparative analysis of RBF approximations has also been performed by Majdisova and Skala [7].

Often, along with the values of modeled functions to approximate, RBFs also take a function's gradient into account. In this case, we usually refer to them as Hermite radial basis functions (HRBF) [8], [9].

DOI 10.15588/1607-3274-2025-1-3

In this work, we use a very specific sort of RBF approximation – Gaussian interpolation [10], [11]. However, to enable localized computation, and thus enable practical application of the global interpolating function, we must resort to a substitution. In this work, we'll use pseudo-Gaussian functions instead of Gaussian functions.

3 MATERIALS AND METHODS

Let's define a pseudo-Gaussian function as a symmetric function $pg(x)$ of a real argument such as:

$$\begin{aligned} &pg(0)=1;\\ &pg(r)=0;\\ &pg(r)^{(i)}=0, i=1..n;\\ &pg(x)=0, x\le -r;\\ &pg(x)=0, x\ge r. \end{aligned} \quad (1)$$

In this way, the function is non-zero on $(-r, r)$, zero elsewhere, and it has its first n derivatives in $-r$ and r, specified as 0 too.

The function is symetric and, as such, mimics the Gaussian function near 0 (see Fig. 2).

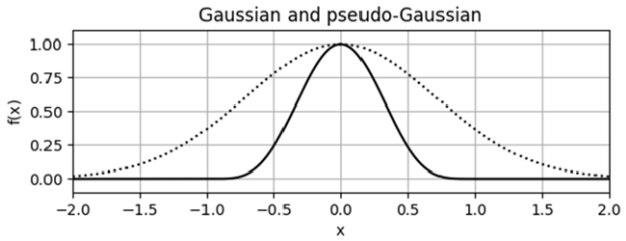

Figure 2 – A Gaussian function (dotted) and a pseudo-Gaussian with n = 4, r = 1 (solid)

The amount of non-zero derivatives on $(-r, r)$ that also become 0 in r and $-r$ condition the continuity class C^n of the pseudo-Gaussian sum and therefore the smoothness degree of the implicit surface (or a contour in 2D) that we're aiming to model.

To specify a pseudo-Gaussian $pg(x)$ for any given r and n, we can use a polynomial representation. A polynomial that represents $pg(x)$ will be a symmetrical polynomial, so it will only consist of even degrees of x, and its degree then should be at least $2(n+2)$. To obtain all the coefficients of a minimal degree polynomial that satisfy the conditions from above, we should comprise a system of equations:

$$\begin{cases} P(0)=1,\\ P(r)=0,\\ P'(r)=0,\\ P''(r)=0,\\ \dots\\ P^{(n)}(r)=0. \end{cases} \quad (2)$$

Since r and n are known constants and we only need to determine the coefficients of the polynomial, the system is linear. We can use any solver to get the coefficients for any degree n and any r.

Interestingly, we might not even have to solve the system at all. The coefficients of the polynomial regardless of r seem to follow the binomial coefficients. For instance, here are the generalized formulas for the $P_n(x)$ for n=1..4:

$$pg_1(x)=1-\frac{2x^2}{r^2}+\frac{x^4}{r^4}, \quad (3)$$

$$pg_2(x)=1-\frac{3x^2}{r^2}+\frac{3x^4}{r^4}-\frac{x^6}{r^6}, \quad (4)$$

$$pg_3(x)=1-\frac{4x^2}{r^2}+\frac{6x^4}{r^4}-\frac{4x^6}{r^6}+\frac{x^8}{r^8}, \quad (5)$$

$$pg_4(x)=1-\frac{5x^2}{r^2}+\frac{10x^4}{r^4}-\frac{10x^6}{r^6}+\frac{5x^8}{r^8}-\frac{x^{10}}{r^{10}}. \quad (6)$$

The core difference between the Gaussian and the pseudo-Gaussian function is that the latter is localized, meaning that all non-zero values of $pg(x)$ lie within the $(-r, r)$ interval. Since we're planning to use this function as a basis radial function determined on a regular grid, this means that for any point in space, we can compute the sum of all the functions in the whole grid by only computing the ones that are defined in the proximity of the point. Every RBF defined further than r from the point of computation will be 0 by definition and, therefore will not contribute to the sum.

This allows us to use a global interpolation function – the pseudo-Gaussian interpolation function – such as it would have been local.

A pseudo-Gaussian interpolating function is a weighted sum of pseudo-Gaussian radial functions determined in all the points of a finite regular grid.

Let's now focus on 2D space for brevity. If we have a grid of size $M\times N$ that starts at a point (x_0, y_0), and has a constant distance k between adjacent points (x_{j+1}, y_i) and (x_j, y_i) as well as between (x_j, y_{i+1}) and (x_j, y_i) for all i = 0..N–2, j = 0..M–2, then the points of this grid would be defined as:

$$(x_j, y_i)=(x_0+jk, y_0+ik), i=0..N-1, j=0..M-1. \quad (7)$$

Let's say we have defined pseudo-Gaussian-based radial basis functions $pgr_{ij}(x, y)$ for all i = 0..n–1, j = 0..m–1:

$$pgr_{ij}(x,y)=pg_n(\|(x,y)-(x_j,y_i)\|). \quad (8)$$

And their respective coefficients a_{ij}. The pseudo-Gaussian interpolating function will then look like this:

DOI 10.15588/1607-3274-2025-1-3

$$Spg(x,y)=\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}a_{ij}\,pgr_{ij}(x,y)\,. \qquad (9)$$

Although here we present a formula as a full sum of all the weighted functions, remember, we don't have to do the full computation for all the basis functions every time we want to compute the $Spg(x, y)$ at a point. Given that the pseudo-Gaussians $pg(x)$ we choose have an effective range of non-zero values r, meaning that $pg(x) = 0$ when $x \le -r$ and $x \ge r$, to compute the interpolating function at a point (x, y) we only need to account for the radial basis functions that are defined in the nodes (x_j, y_i) where $|[x, y] - [x_j, y_i]| < r$. This means that in order to compute an $Spg(x, y)$ at any point, only a relatively small and predefined number of basis functions should be computed, which reduces the computation complexity of such a computation from $O(N^2)$ to $O(1)$.

The coefficients a_{ij} are exactly the data that characterize the $Spg(x, y)$ function. To make the pseudo-Gaussian sum interpolate an arbitrary function we want to model: $SDF(x, y)$, we should select exactly $M \times N$ points (x_a, y_b) where $a = 1..M$, $b = 1..N$ such as they are all defined within the range of the corresponding radial functions meaning that $|(x_a, y_b)–(x_j, y_i)| < r$. Then for the defined grid and defined values $SDF(x_a, y_b)$, we can define a system of equations:

$$Spg(x_a,y_b)=SDF(x_a,y_b)\,. \qquad (10)$$

In regard to the coefficients a_{ij} the system is linear. Moreover, for $N > r/d$ and $M > r/d$, which is expected given the presumed application of this interpolation function, the system will appear sparse. Methods of solution for these types of systems are well known. By solving the system with a chosen method we get the array of a_{ij} coefficients each defined for a grid point (x_j, y_i) and for its radial basis function respectively.

The pseudo-Gaussian interpolation function generalizes easily to 3D. We still use the same radial basis functions, it's only now there are 3 arguments in the function, and the grid is, of course, also 3-dimensional.

If we have a grid of size $M \times N \times P$ that starts at a point (x_0, y_0, z_0), and has a constant distance k between adjacent points (x_{j+1}, y_i, z_l) and (x_j, y_i, z_l), as well as between (x_j, y_{i+1}, z_l) and (x_j, y_i, z_l), and now also (x_j, y_i, z_{l+1}) and (x_j, y_i, z_l), for all $i = 0..N–2$, $j = 0..M–2$, $l = 0..P–2$ then the points of this grid would be:

$$\begin{aligned}&(x_j,y_i,z_l)=(x_0+jk,y_0+ik,z_0+lk),\\&i=0..N-1,j=0..M-1,l=0..P-1.\end{aligned} \qquad (11)$$

The radial basis functions:

$$pgr_{ijl}(x,y,z)=pg_n\left(\left|(x,y,z)-(x_j,y_i,z_l)\right|\right). \qquad (12)$$

And the pseudo-Gaussian interpolating function will then be:

$$Spg(x,y,z)=\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\sum_{l=0}^{p-1}a_{ijl}\,pgr_{ijl}(x,y,z)\,. \qquad (13)$$

The interpolation points are then defined as (x_a, y_b, z_c), where $a = 1..M$, $b = 1..N$, $c = 1..P$.

And the system to get the coefficients a_{ijl} is:

$$Spg(x_a,y_b)=SDF(x_a,y_b)\,. \qquad (14)$$

Note that in both 2D and 3D, the interpolation points may or may not be equal to the grid points. In this work, we argue that selecting interpolation points different from the grid points is a valuable feature we get by storing interpolation function coefficients instead of SDF values.

In both 2D and 3D, the result of interpolation relies heavily on the choice of the interpolation points (x_a, y_b) or (x_a, y_b, z_c) respectively. Of course, other factors impact the interpolation result too: the length of the non-zero range r of the pseudo-Gaussian function $pg(x)$, the distance between the grid points k, and the order of continuity of the basis functions sum n. Still, while all these parameters shape the resulting function differently, and as such deserve studying on their own, they mostly help balance the computational complexity and the accuracy of the model. The interpolation points choice, however, is crucial for successful SDF modeling.

As we mentioned before, we can forfeit the choice to the default and set interpolation points equal to the points of the grid: $(x_a, y_b) = (x_j, y_i)$.

Let's take a 2D SDF of a rotated square (Fig. 3), and show how the result of its interpolation looks like when the interpolation points coincide with the points of the grid (Fig. 4).

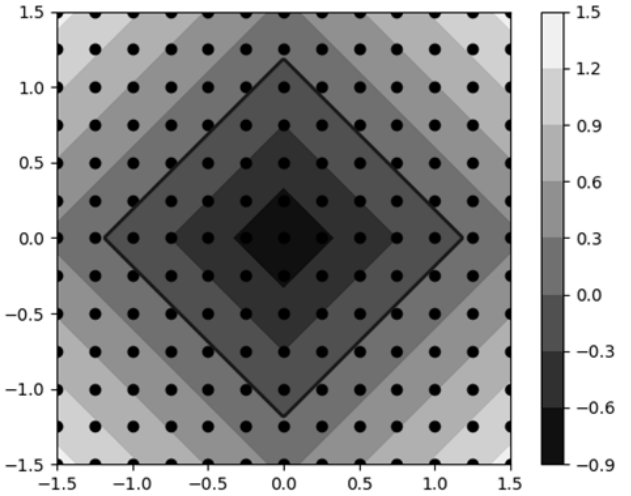

Figure 3 – An SDF of a square rotated by 45 degrees, and a regular 13×13 grid

DOI 10.15588/1607-3274-2025-1-3

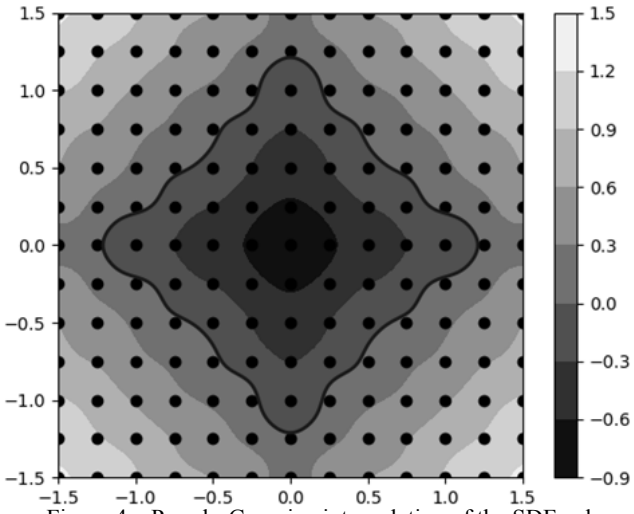

Figure 4 – Pseudo-Gaussian interpolation of the SDF values in grid points

This interpolation models the SDF and we already can store the coefficients of the interpolating function in a 2D image instead of SDF values. The resulting image will take exactly as much memory as the 2D image of SDF values, but we wouldn't have to compute the coefficients afterward.

But if we define each point of interpolation (x_a, y_b) as the point of the isoline $SDF(x, y) = z$, nearest to the grid point (x_j, y_i), where $z = z_{step}q$, q is integer, and the isoline step $z_{step} <= k$, then with this choice of interpolation points, we make our model much more precise around the select isolines, and, what's even more important, around the isoline $SDF(x, y) = 0$ that represents the entity we want to model (see Fig. 5).

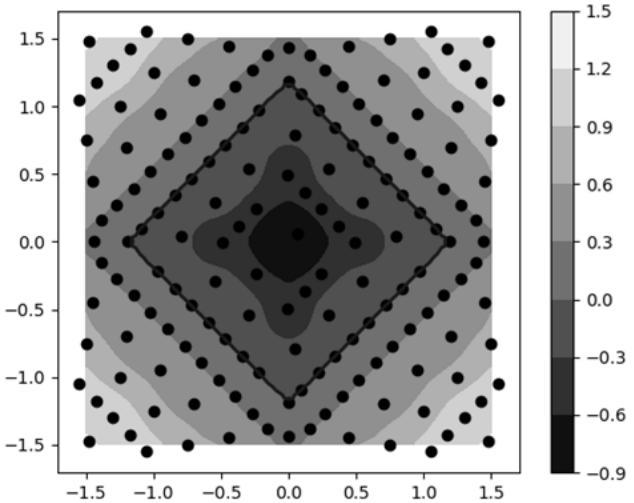

Figure 5 – Model of the rotated square SDF with interpolating points defined on $SDF(x, y) = 0.25q$

The points (x_j, y_i) where the radial functions are defined remain intact for every possible choice of the interpolating points (x_a, y_b). Of course, the choice affects the model, it affects its coefficients a_{ij}, but not the way the interpolating function is computed from these coefficients, so this choice of points brings no impact on the computational complexity of the whole function computation either.

The proposed method of selecting the interpolation points has not been extensively tested in real-world cases. It rather illustrates the idea of point selection itself than serves as a practical guide. The idea itself is interesting since the very possibility of point selection extends our possibilities for more precise modeling of implicit contours in 2D and surfaces in 3D, and doesn't add any memory toll to the resulting data structure.

4 DISCUSSION

With our approach, we can define the interpolating point in any point in between grid points but there are limitations to this too. Any point (x_a, y_b) should be no further from the grid point (x_j, y_i) than r. But what's more important, the total number of these points still can't exceed nm for the regular grid of size $m \times n$. In the context of storing and transmitting SDF models, this means that we can make our model of SDF more accurate in some regions only by forfeiting accuracy in others. This works well with implicit surface (and contour in 2D) representation where we want to focus on the isoline $SDF(x) = 0$, but not necessary in the more general context.

In a way, this reflects how the lossy compression works. We accentuate the important information, and by letting the less important details go, use less data to store essentially the same object as with lossless compression. But what's important and what's not depends solely on the context of the application, not on the information, or the model, itself.

For instance, the consistency of the SDF is important for contouring – turning the SDF into a set of polylines in 2D and triangle meshes in 3D. Common contouring methods like marching cubes, surface nets, or double contouring – all rely on the linearity of the SDF. Marching cubes often use linear interpolation to determine the vertices' positions, and dual contouring also requires Hermite data, so it requires the gradient of the SDF to be consistent with the distance function too.

Another example of a practical problem where the accuracy of the SDF model is equally important as on the isoline $SDF(x) = 0$ as everywhere else is the offset modeling. Signed distance functions, as their name hints are perfect for offset modeling since all you have to do to build an offset of an SDF is to subtract the offset distance from the SDF. Offsets are routinely used in 3D printing both in 3D for solving positioning and supporting problems, and in 2D, to generate toolpaths from the contours of the sliced model. Although our approach allows us to put modeling focus into multiple isolines, we don't necessarily know beforehand which isolines will be used for offsetting so we can't exploit this information to make our model more accurate for the offseting specifically.

The interpolation points selection is a valuable option, but it's not omnipotent. We can make our model more accurate for some applications, but there will always be counterexamples when this rearrangement of data points backfires.

DOI 10.15588/1607-3274-2025-1-3

The other problem is that the control over interpolating points doesn't necessarily grant control over the shape of the modeled body. The exact shape of the surface or contour we achieve doesn't necessarily coincide with the initial shape let alone our expectations. In Fig. 6 you can see an SDF and the pseudo-Gaussian interpolation of this SDF with interpolation points defined on $SDF(x, y) = 0.25q$.

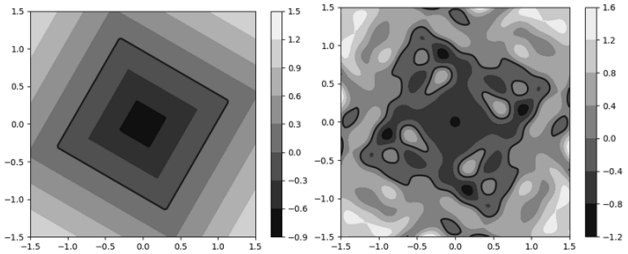

Figure 6 – An SDF of a rotated square and its pseudo-Gaussian interpolation gone wrong

Please note, that 0.25 is the same isoline step as before (see Fig. 5). We also use the same basis functions and the same grid. In Fig. 5 we have shown a similar square rotated slightly differently, and the same strategy for interpolation points selection worked there well.

The result of the modeling depends on four factors:

1. The non-zero range of pseudo-Gaussian basis functions r.
2. The order of the pseudo-Gaussian sum continuity n.
3. The distance between grid points k.
4. And the choice of interpolation points (y_j, y_i).

How exactly these factors affect the result of the modeling has not been yet extensively studied. We have run a series of experiments modeling implicit curves in 2D to determine how r, n, and k affect the model's shape but the results have been inconclusive so far.

Sometimes, when a model of an implicit curve loses its original shape, meaning that there are extra contours in between interpolation points as in Fig. 6, increasing the non-zero range of the pseudo-Gaussian function may help establish its original contour. But for some SDFs, this is not the case. We haven't found the specific pattern here. Also, note that enlarging the r raises the computation cost of the interpolation function since for each point we compute it at, we have to take more basis functions into account. In other words, we can't raise r too much, as this will hinder the applicability of the whole approach.

Also, since at every point the resulting model depends on the contribution of all the radial functions defined within radius r from the point of computation, it is highly recommended to define the interpolating grid in a way that it bounds the zero polyline with at least r/d wide margin (see Fig. 7).

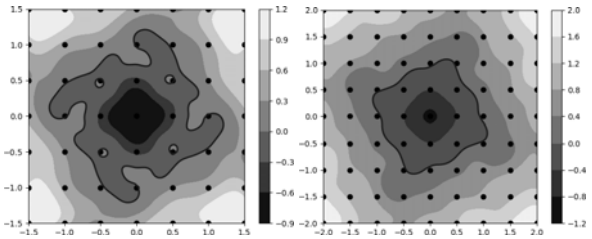

Figure 7 – An extra margin of only one cell helps preserve the shape

Surprisingly, raising the order of pseudo-Gaussian sum continuity n may also help keep the shape of the modeled curve closer to the original although within limits. E. g. if the shape is lost with $n = 2$, raising n to 3 or 4 might help. But if the shape is still lost, adding more members to the pseudo-Gaussian function will most likely elevate the computational cost with no additional benefit. We haven't tested this extensively so far, so this requires further investigation. It might even be worth considering not only experimenting with n, but taking completely different kinds of radial basis functions, or perhaps not even radial, into test.

Managing the distance between points k or, which is the same, regulating the grid density is the common way of trading computational speed and memory footprint for accuracy with traditional voxel-based modeling. Normally, but not necessarily, it works the same with our interpolation-based approach too. The problem is, with linear interpolation, the value between two interpolation points lies in between their respective values. With pseudo-Gaussian interpolation, this is not a given. Besides, the whole point of storing interpolation coefficients instead of SDF values was to increase model precision in the important area without adding extra grid points there. Raising the grid density too much diminishes this advantage.

So while selecting different r, n, and k may or may not help us ensure that the shape of the model is consistent with its original, the only consistent way to do so is by choosing the interpolation points. Moreover, this is the only choice that doesn't affect the computational speed or the memory footprint of the resulting model. Also, this is the only approach that has so far shown itself robust in the experiments.

For the model in Fig. 6 specifically, we have conducted the following computational experiment. We didn't change the method of interpolation points selection itself, we still chose the closest point on isoline $SDF(x, y) = z_{step}q$. Using numerical optimization, however, we found the isoline step value where the difference between the original SDF and its interpolated model is minimal. It was shown that for this particular model, the best z_{step}, computed up to the 6th decimal sign after the decimal point, equals 0.081384. See Fig. 8.

DOI 10.15588/1607-3274-2025-1-3

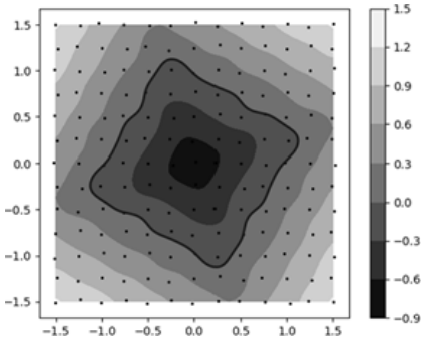

Figure 8 – A model of the square SDF with interpolating points defined on $SDF(x, y) = 0.081384q$

The experiment shows that the interpolation points selection is crucial for every modeling problem in particular. The fact that the optimal z_{step} for the model in the experiment is so low hints that it could be that for this case we need another method of points selection. Something that puts more data into the main isoline $SDF(x, y) = 0$ and not necessarily $SDF(x, y) = z_{stepq}$ where $q \neq 0$.

CONCLUSIONS

Pseudo-Gaussian interpolation defined as a set of radial basis functions on a regular grid with points of interpolation defined in the proximity of the grid points allows us to model an implicit surface more accurately than a voxel model interpolation. At the same time, the memory or computational toll isn't much different in these two approaches. Let's summarize.

Due to the pseudo-Gaussian radial basis function localization, the computational complexity of the whole interpolation function doesn't depend on the grid size and thus can be evaluated as $O(1)$.

Given that the interpolation function is a weighted sum of the radial basis functions defined in the known grid points, and to define such a function we only need to compute, store, and, if necessary transmit, its weight coefficients – 1 per grid point – the memory footprint of this model is exactly the same as a footprint of a voxel grid with SDF values.

At the same time, we retain an option to choose the points of interpolation before the weight coefficients are composed. This allows us to model select isolines more accurately than any generic interpolation of a voxel grid can.

Additionally, since the data structure of the pseudo-Gaussian interpolation function is compatible with a voxel grid, we can already store this type of data in any existing data format that allows voxel grids (also known as 3D images or grey-value floating point images). This includes 3MF, HDF5, NRRD, and even, with the introduction of a user-defined SOP extension, DICOM.

At the same time, the three mechanisms to balance the performance and accuracy of the modeling: selecting the non-zero range of pseudo-Gaussian basis functions r, picking the order of the pseudo-Gaussian sum continuity n, and choosing the distance between grid points k – all deserve additional studying.

But the most important is the problem of the interpolating points selection. The method we proposed in this paper brings only illustrative not practical value. The key to turning the pseudo-Gaussian grid interpolation into a robust technology for implicit surface storage and transmission lies in discovering the best interpolating points selection strategy.

ACKNOWLEDGEMENTS

The study was performed without financial support.

REFERENCES

1. Andrianov I. V., Ausheva N. M., Olevska Yu. B., Olevskyi V. I. Surfaces Modelling Using Isotropic Fractional-Rational Curves, *Journal of Applied Mathematics*. Hindawi, 2019, Vol. 1, pp. 1–13. DOI: 10.1155/2019/5072676
2. Li H., Yang X., Zhai H., Liu Y., Bao H., Zhang G. VoxSurf: Voxel-Based Implicit Surface Representation, *IEEE Transactions on Visualization and Computer Graphics,* 2024, Vol. 30 (3), pp. 1743–1755. DOI: 10.1109/TVCG.2022.3225844
3. Sommer C., Sang L., Schubert D., Cremers D. Gradient-SDF: A Semi-Implicit Surface Representation for 3D Reconstruction, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6280–6289.
4. Michalkiewicz M., Pontes J. K., Jack D., Baktashmotlagh D., Eriksson A. Implicit Surface Representations As Layers in Neural Networks, *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 4743–4752.
5. Xu Q., Wang W., Ceylan D., Mech R., Neumann U. DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction, *Advances in Neural Information Processing Systems,* 2019, Vol. 32. DOI: 10.48550/arXiv.1905.10711
6. Jiahui M., Huahao Sh., Wei Ch. Implicit Surface Reconstruction via RBF Interpolation: A Review, *Recent Patents on Engineering*, 2022, Vol. 16 (5), pp. 49–66. DOI: 10.2174/1872212115666210707110903
7. Majdisova Z., Skala V. Radial basis function approximations: comparison and applications, *Appl. Math. Modelling,* 2017, Vol. 51, pp. 728–743. DOI: 10.1016/j.apm.2017.07.033
8. Macêdo I., Gois J. P., Velho L. Hermite interpolation of implicit surfaces with radial basis functions, *Proceedings of the 2009 XXII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI),* IEEE, 2009, pp. 1–8. DOI: 10.1109/SIBGRAPI.2009.11
9. Pan R., Meng X., Whangbo T. Hermite variational implicit surface reconstruction, *Science in China Series F: Information Sciences,* 2009, Vol. 52, pp. 308–315. DOI: 10.1007/s11432-009-0032-x
10. Sydorenko Yu. V., Horodetskyi M. V. Modification of the algorithm for selecting a variable parameter of the Gaussian interpolation function, *Control Systems and Computers,* 2020, Vol. 6 (290), pp. 21–28. DOI: 10.15407/csc.2020.06.021
11. Sydorenko Yu. V., Onysko A. I., Shaldenko O. V., Horodetskyi M. V. Interpolation of different types of spiral-like curves by gaus-interpolation methods, *Control Systems and Computers,* 2022, Vol. 3 (299), pp. 1–10. DOI: 10.15407/csc.2022.03.003

Received 27.11.2024.
Accepted 28.01.2025.

DOI 10.15588/1607-3274-2025-1-3

УДК 004.94

МОДЕЛЮВАННЯ ІМПЛІЦИТНИХ КРИВИХ І ПОВЕРХОНЬ ПСЕВДОГАУСОВОЮ ІНТЕРПОЛЯЦІЄЮ

Аушева Н. М. – д-р техн. наук, професор, завідувачка кафедри цифрових технологій в енергетиці Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

Сидоренко Ю. В. – канд. техн. наук, доцент, доцент кафедри цифрових технологій в енергетиці Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

Каленюк О. С. – канд. техн. наук, старший викладач кафедри цифрових технологій в енергетиці Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

Кардашов О. В. – аспірант кафедри цифрових технологій в енергетиці Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

Городецький М. В. – аспірант кафедри цифрових технологій в енергетиці Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

АНОТАЦІЯ

Актуальність. Із сучасним розвитком методів топологічної оптимізації, параметричного проектування і проектування на основі штучного інтелекту, проблема запису імпліцитних поверхонь у задачах 3D-друку стала надважливою**.** Хоча все більше програмних продуктів використовують імпліцитні моделі у проектуванні, єдиного стандарту для запису, а значить і для збереження та передачі таких моделей засобами комп'ютерних мереж, не існує. Об'єктом цього дослідження є один з можливих способів такого запису, а саме – моделювання імпліцитних кривих і поверхонь із використанням псевдогаусової інтерполяції.

Мета роботи. Ціль роботи полягає у створенні способа моделювання імпліцитних кривих і поверхонь, із покращенною точністю без значних витрат пам'яті чи часу обчислення.

Метод. Одним із загальноприйнятих способів запису функції знакової відстані (ФЗВ) є запис її значень на регулярній сітці. Такий спосіб ще називається 3D-зображення, 3D-бітмап, або воксельна модель. Неперервна ФЗВ може бути отримана із записаних значень за допомогою інтерполяції. Натомість пропонується записувати не значення ФЗВ, а значення коефіцієнтів псевдогаусової інтерполяційної функції, що дозволяє обирати точки інтерполяції до запису коефіцієнтів моделі. Таким чином можна досягти більш точного моделювання у найважливіших регіонах (див. рисунок 1) без використання додаткової пам'яті.

Результати. Запропонований спосіб був імплементований у вигляді комп'ютерної програми для моделювання пласких імпліцитних кривих і провалідований на декількох примитивних моделях різного походження: колах, квадратах, прямокутниках, – із різними параметрами моделі. Вцілому, порявняно із інтерпольованими значеннями ФЗВ у точках решітки, метод показує кращу точність, але разом із тим має декілька граничних станів, у яких він потребує подальшого вивчення.

Висновки. Псевдогаусова інтерполяція, визначена як сума радіальних базисних функцій на регулярній сітці із точками інтерполяції визначеними у ненульовому околі точок сітки в загальному випадку дозволяє моделювати імпліцитні криві і поверхні точніше ніж інтерполяція воксельної моделі. Разом з тим, оптимальна стратегія визначення точок інтерполяції і інших параметрів моделі для прикладного застосування лишається відкритою проблемою.

КЛЮЧОВІ СЛОВА: представлення поверхонь, представлення кривих, імліцитне представлення, псевдогаусова функція, регулярна сітка, моделювання імпліцитних поверхонь, формат даних для імпліцитних поверхонь.

ЛІТЕРАТУРА

1. Surfaces Modelling Using Isotropic Fractional-Rational Curves / [I. V. Andrianov, N. M. Ausheva, Yu. B. Olevska, V. I. Olevskyi] // Journal of Applied Mathematics. – Hindawi, 2019. – Vol. 1. – P. 1–13. DOI: 10.1155/2019/5072676
2. Vox-Surf: Voxel-Based Implicit Surface Representation / [H. Li, X. Yang, H. Zhai et al.] // IEEE Transactions on Visualization and Computer Graphics. – 2024. – Vol. 30 (3), P. 1743–1755. DOI: 10.1109/TVCG.2022.3225844
3. Gradient-SDF: A Semi-Implicit Surface Representation for 3D Reconstruction / [C. Sommer, L. Sang, D. Schubert, D. Cremers] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). – 2022. – P. 6280–6289.
4. Implicit Surface Representations As Layers in Neural Networks / [M. Michalkiewicz, J. K. Pontes, D. Jack et al.] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). – 2019. – P. 4743–4752.
5. DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction / [Q. Xu, W. Wang, D. Ceylan et al.] // Advances in Neural Information Processing Systems. – 2019. – Vol. 32. DOI: 10.48550/arXiv.1905.10711
6. Jiahui M. Implicit Surface Reconstruction via RBF Interpolation: A Review / M. Jiahui, Sh. Huahao, Ch. Wei // Recent Patents on Engineering. – 2022. – Vol. 16 (5). – P. 49–66. DOI: 10.2174/1872212115666210707110903
7. Majdisova Z. Radial basis function approximations: comparison and applications / Z. Majdisova, V. Skala // Appl. Math. Modelling. – 2017. – Vol. 51. – P. 728–743. DOI: 10.1016/j.apm.2017.07.033
8. Macêdo I. Hermite interpolation of implicit surfaces with radial basis functions / I. Macêdo, J. P. Gois, L. Velho // Proceedings of the 2009 XXII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), IEEE. – 2009. – P. 1–8. DOI: 10.1109/SIBGRAPI.2009.11
9. Pan R. Hermite variational implicit surface reconstruction / R. Pan, X. Meng, T. Whangbo // Science in China Series F: Information Sciences. – 2009. – Vol. 52. – P. 308–315. DOI: 10.1007/s11432-009-0032-x
10. Sydorenko Yu. V. Modification of the algorithm for selecting a variable parameter of the Gaussian interpolation function / Yu. V. Sydorenko, M. V. Horodetskyi // Control Systems and Computers. – 2020. – Vol. 6 (290). – P. 21–28. DOI: 10.15407/csc.2020.06.021
11. Sydorenko Yu. V. Interpolation of different types of spiral-like curves by gaus-interpolation methods / [Yu. V. Sydorenko, A. I. Onysko, O. V. Shaldenko, M. V. Horodetskyi] // Control Systems and Computers. – 2022. – Vol. 3 (299). – P. 1–10. DOI: 10.15407/csc.2022.03.003

DOI 10.15588/1607-3274-2025-1-3