paszynsk@agh.edu.pl

Approximation with B-spline basis functions

Brak plików do pobrania.

In order to write a computer program to calculate this continuous approximation of the terrain, we need to do the following. First, we have to describe our problem in a formal, mathematical way. In particular, we have to choose the mathematical definition of the object - bitmap, and the mathematical definition of our continuous description of the terrain represented by the bitmap.

It seems reasonable to say that a bitmap is a function defined in an area

$$\Omega = [1, maxx] \times [1, maxy] \ni (x, y) \rightarrow BITMAP(x, y) \in [0, 255]$$

where by Ω we mark the entire area where our bitmap is spread.

In turn our continuous representation of the world will be represented by a function u o real values.

$$\Omega = [1, maxx] \times [1, maxy] \ni (x, y) \rightarrow u(x, y) \in [0, 255]$$

We want our function to be "slim" and continuous. Mathematically we write the condition that our function will be of the C^1 class, it means that at each point it will be slender enough to be possible count its derivatives in directions perpendicular to the edges of our area. This practically means that at every point we can apply a "ruler" to the graph of our function perpendicularly to one of the edges of our area, and measure the angle between this ruler and the base (flat surface spread out at zero height). After all, the derivative is nothing but the tangent of this angle. In other words, our function will have no "kinks" or gaps where you would not know how to apply our ruler. At the bends, this ruler would jump from one position to another, and in the case of a hole, it would not be known how to put it. Of course, the ability to measure the derivative (put a ruler) in two directions perpendicular to the edge of the area also means the ability to apply a ruler and measure directional derivatives in any other directions not perpendicular to the edge. So we can smoothly move across the plot of such a slender function in any directions. How to get such a continuous slender function? We have to decide how our function will be constructed. For example, we can divide the area over which the bitmap is spanned into certain elements, and on these elements define a set of many slender functions, from which we then "glue" our function u.

Let us imagine that at the height corresponding to the zero height (equal to the value of the zero pixel) we build a flat two-dimensional mesh, the number of square elements of which is arbitrary. These meshes, according to the adopted nomenclature, will be called finite elements, because each of these elements has a limited finite area. These elements may be less than the pixels in the bitmap, and then several pixels will be spread over each such element. The boundaries between our elements do not have to match the boundaries of the pixels. They can be freely defined on a flat surface. It may also be the case that our elements are more than pixels, and then there will be many such elements in each pixel. Let's assume, however, that our elements are less than pixels. They form a regular grid of

$N_x * N_y$ elementach skończonych.

Now, on each such element, we define a slender function. We can use B-spline functions for this purpose.

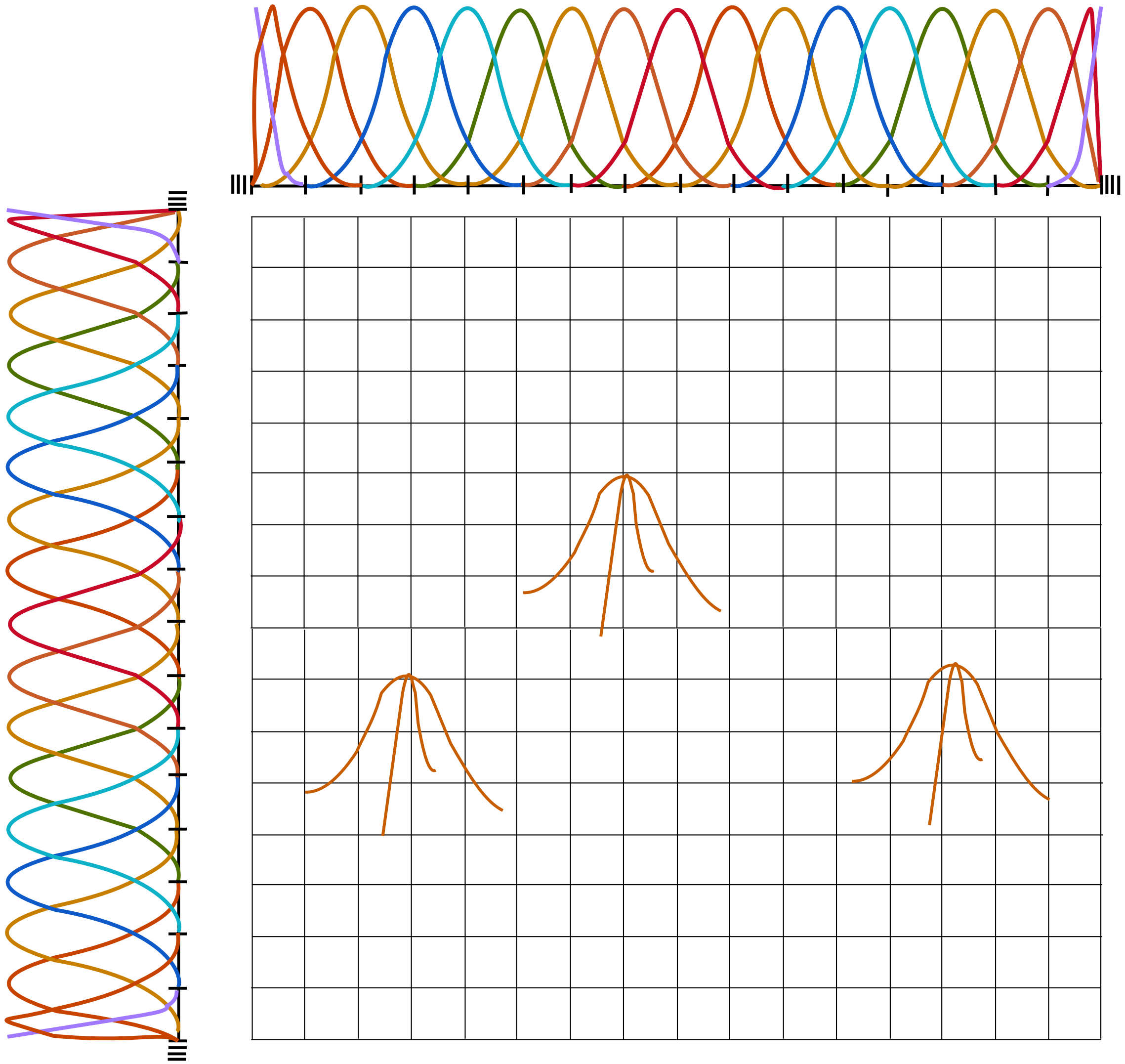
These functions were first introduced by the American mathematician of Romanian origin, Isaac Jakub Schoenberg

[1].

B-spline functions are widely used in computer modeling and simulations thanks to the growing popularity of the field called isogeometric analysis disseminated by prof. T.J.R. Hughes.[2].

The idea behind these methods is to use families of B-spline functions for finite element calculations.

We denote these functions $B_{i,j;2}(x,y)$, where i and j denote the numbering of our functions, and 2 means that they are polynomials with pieces of the second order, class C^1.



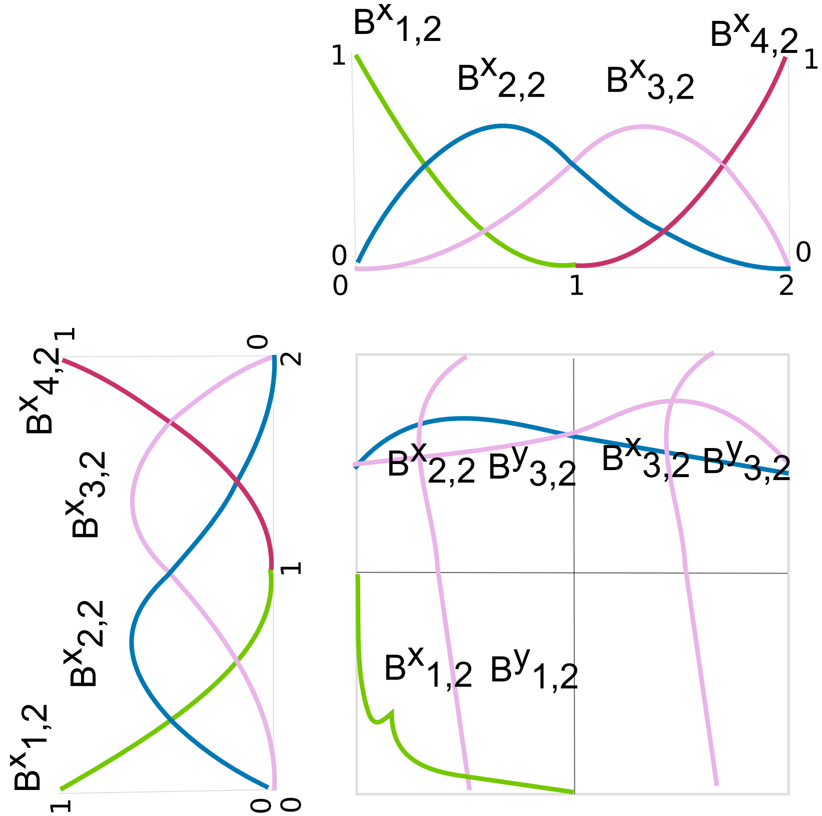
Rysunek 1: Trzy przykładowe funkcje B-spline rozpięte na dwuwymiarowej siatce.

shows three exemplary two-dimensional B-spline functions spanned over a two-dimensional mesh. Each such two-dimensional B-spline function is created by selecting and multiplying by itself two one-dimensional B-spline functions, one selected from the set of one-dimensional B-spline functions spanning along the horizontal edge of the mesh, and the other selected from the set

of one-dimensional B-spline functions spanning along the vertical edge. mesh. These sets are called the bases of one-dimensional B-spline functions.
We denote these one-dimensional Bspline functions in turn $B^x_{i;2}(x)$ and $B^y_{j;2}(y)$ where variables x and y identify the directions (coordinate axes) along which our B-splines (horizontal axis x and vertical axis y), i and j numbering areas of these functions (which next one-dimensional B-spline side we choose from such one-dimensional basis), and 2 means again that they are polynomials with pieces of the second degree, class C^1 (i.e. that we can count the first derivatives of them).
Then the one-dimensional functions are selected multiplied by themselves, which gives us a slender two-dimensional B-spline role. Such a relaxation of a function by multiplying two-dimensional functions is called a tensor product of a function.

$$B_{i,j;2}(x,y) = B^x_{i;2}(x)B^y_{j;2}(y)$$

This is illustrated in . The two-dimensional B-spline functions given in this way have the shape of a slender "hill", similarity on nine adjacent elements. The highest point of such a function - the hill - is in the center of the middle element. These functions go down to the value zero, assumed at the edges of the software defined by the nine elements on which the function is defined. These functions, according to the nomenclature, are called base functions.



Rysunek 2: Dwuwymiarowa funkcja B-spline sklejona za pomocą dwóch jednowymiarowych funkcji B-spline.

Our continuous approximation of the terrain is obtained by adding together many such slender hills - B-splines. Each of them will be scaled (raised up or down) so as to obtain a total of continuous approximation of the terrain. If we correctly select the heights of individual hills, then we will get slender approximations of our terrain as presented in .
Now the question is how to choose the heights to which we will draw our base functions. The first method that comes to mind is to choose the pixel value z $BITMAP(x,y)$ exactly at the highest point of the B-spline (in the middle of the hill).
Unfortunately, this method has several disadvantages. First, if our Bspline base functions are fewer than pixels, then we ignore all neighboring pixels in our Bspline area, choosing only one value from the center of the area. It is possible that our bitmap has some disturbances and that we will hit a local bounce that is a measurement error, or that we will hit a local hole in the terrain, or a local tree or building that has disturbed the measurement of the terrain. Second, notice that our base functions extend over a square of nine elements. Since each such B-spline function can be associated with the center of its element, and each of these

functions extends over nine adjacent elements, it means that there are a total of nine such functions spread on each element, and that the adjacent functions overlap. So if we were to stretch the functions so that their maximum point coincides with the central pixel, and we would sum all these functions together to get our global function u, then on each element, even in the central point, our resulting approximation (the sum of these functions) would be higher than our central pixel, because the adjacent eight functions would also be non-zero on a given element and would raise the value of our approximation at this point up

Bibliografia

1. Isaac Jacob Schoenberg: Cardinal Spline Interpolation, Society for Industrial and Applied Mathematics, SIAM 1973, dostęp:18.10.2019
2. J. Austin Cottrell, T. J. R. Hughes,Yuri Bazilevs: Isogeometric Analysis: Toward Integration of CAD and FEA, John Wiley & Sons, Computational and Numerical Methods 2009, dostęp:18.10.2019

Utworzona przez admin. Ostatnia aktualizacja: Środa 07 z Październik, 2020 07:53:18 UTC przez paszynsk@agh.edu.pl. Autor: Maciej Paszynski

STATUS: W opracowaniu Zgłoś do recenzji Edytuj