Title:
# Image Similarity Computation using B-spline Functions

Authors:
Les A. Piegl, lespiegl@usf.edu, University of South Florida
Zachariah J. Beasley, zjb@usf.edu, University of South Florida

Keywords:
Point cloud, medical images, NURBS, similarity

DOI: 10.14733/cadconfP.2022.145-149

## Introduction:
This paper presents a methodology to measure the degree of similarity between two images quantitatively. The research was motivated by a medical application to search a database of MRI/CT images for comparative medical diagnosis. The proposed algorithm converts the image into a point cloud, fits a NURBS (planar) surface to the points, and uses symbolic algebra on these surfaces to determine which image is most similar to a given query image.

## Proposed Approach:
We propose a multi-stage approach to assist medical diagnosis using images. The raw images are converted to NURBS surfaces in the first stage. Because this will be used for similarity detection, not for visualization, many simplifications can be done. The image should be clipped, the resolution can be reduced, and the intensities compressed. Once the simplification is done, a point cloud is generated: each picture element is represented by a set of points based on the intensity. This step is critical as large point clouds can produce an explosion of NURBS data. A NURBS surface then fits the point cloud, and both the image and the NURBS surface are saved in the database for the next step of the multi-stage process.

Having completed the conversion of images into NURBS forms, searching for similar images comes next. The new image, after simplification, is converted into a point cloud, and the point cloud is aligned to a particular position. A NURBS surface is then fitted to the aligned point cloud to search the database of the NURBS surfaces of the previously converted images. The corresponding information is used for comparative diagnosis if a sufficiently similar image is found. Once the patient is diagnosed and treated, all relevant information is tagged to the image and the NURBS surface for future similarity search and diagnosis.

The NURBS surfaces generated from the images drive the similarity search in a database where images are augmented. The database consists of three main parts. The first contains the raw images, used mainly for visualization. The second part has the NURBS surfaces fitted to the images, and they are used for similarity search. And the third contains all relevant medical data associated with the patient's images, used to assist diagnosis and treatment.

## Image Data Reduction:
While images are used mostly for visualization purposes, when it comes to performing quantitative analysis on them, the redundant data, not necessary for the analysis, needs to be removed. First, images must be clipped by removing the background and other irrelevant parts. Second, the size can be reduced, and third, the full 256 grayscales are not needed either and can be reduced to 128 or 64 (or sometimes even lower).

Point Cloud Generation:
A clustered dot pattern [1] was used to generate a point set for each picture element. For a maximum of 64 intensity levels, the pattern is generated by an 8 x 8 matrix, called the dither matrix [6]. For each picture element, an 8 x 8 dither matrix is used to generate a clustered dot pattern. The picture element is divided uniformly into 8 x 8 cells. For any given intensity, the cell receives a point if the value in the corresponding dither matrix is less than the intensity.

Point Cloud Alignment:
Given two sets of point clouds obtained from converting the images into points, we want to align these points to lay on top of each other.  To do that, we need four transformations: translation, scaling, rotation, and window-to-viewport mapping. Best fitting lines and circles [5] have been used to facilitate the first three, whereas the window-to-viewport mapping needs the bounding boxes only.

Using the line and the circle fits, the point clouds are aligned as follows. A translation is performed so that the circle centers overlap. A scaling is then used to make the two circles identical (same center and radius). Finally, a rotation is employed so that one fitted line is rotated into the other. Denoting the angle between the two lines by alpha, three rotations may be needed to align the point sets: one with the angle alpha, another with 180+alpha, and a third with 90+alpha. The reason for this is that some point sets can be axially symmetric (must try rotation with alpha and 180+alpha) or can be themselves symmetric, e.g., forming a square shape (must try alpha and 90+alpha).

The final step in the alignment process is to bring the bounding boxes together, i.e., to map one point cloud so that its bounding box is identical to the other. In general, this should not be necessary; however, to account for asymmetric scaling and assist the NURBS fitting, the point clouds were subjected to a window-to-viewport transformation. Such a transformation, making minimal changes to prepared medical images, does not impact the similarity of point clouds.

B-spline Surface Fitting:
The two sets of point clouds, brought in alignment in the previous step, are now fitted with two *planar* B-spline surfaces [2]. The fitting algorithm is looking for a B-spline surface that interpolates/approximates the point clouds in the B-spline sense, i.e., for any given data point $P_k$, there is a pair of parameters $(u_i, v_j)$, so that the surface passes through or near that point: $S(u_i, v_j) = P_k \textit{ or } S(u_i, v_j) \approx P_k$. The fitting is performed in two steps: (1) data preparation and (2) surface fitting. To prepare the data for surface fitting, it needs to be arranged to form a rectangular topology. The algorithm is as follows:

**Step 1:** lay a grid over the points so that there is only one point in every cell on average. This can be accomplished by choosing the size of the grid to be: $size = \sqrt{xd \cdot yd / K}$ where $xd$ and $yd$ denote the side lengths of the bounding box and $K$ is the number of points.

**Step 2:** bin the points into the cells while eliminating coincident points.

**Step 3:** add phantom points to empty cells. This step is not required for image data as the background color is also represented with a dot pattern. However, to make the method more general, we added the midpoints of empty cells to the point set. This aids the B-spline fitting; however, it does not alter the similarity (which is based on point distribution, and adding the midpoints of a uniform grid will not change that).

**Step 4:** collect and sort the points column-by-column.

The result of this point processing is a set of rectangularly arranged points in the form:

$$P_{i,j}, i = 0,\cdots,n; j = 0,\cdots,m_i$$

That is, for each column, the number of points can be different. Traditional surface fitting to a non-$N \times M$ data set will not work. However, this kind of data is a perfect setup for a cross-sectional design. It works as follows [3]:

**Step 1:** fit curves of the same degree to the column of data points independently.

**Step 2:** make the curves compatible in the B-spline sense, i.e., merge knot vectors.

**Step 3:** fit a surface through the compatible network of curves.

Because the curves are fitted independently, each one may end up with a different knot vector. This, in turn, can produce a control point explosion during the knot merging process. Assume that there are $n$ columns and that each column has on average $m$ number of internal knots. If all these knots are different, then after merging, the maximum number of control points can be as high as $(m \times n) \times n = mn^2$. Whereas if the internal knots are the same, the number of control points will be in the order of $mn$. That is, even for a simple 100 x 100 image, this adds up to 100 times more control points than desired. Because the cross-sectional curves are all independently obtained, we can never get the ideal $mn$ configuration. However, we can come close to this number by using a master knot vector and fuzzy knots (knot intervals). The method is as follow [4]:

**Step 1:** find the column of points with the largest number of points.

**Step 2:** compute the knot vector to fit this point set and call it the master knot vector.

**Step 3:** create brackets around each master knot $u_i^p$ where $p$ is the degree of the fitting:

$$u_{i-1}^{p-1} < u_i^p < u_i^{p-1}$$

**Step 4:** generate a membership interval $(a,b)$ around each master knot:

$$a = (1 - per) \cdot u_i^p + per \cdot u_{i-1}^{p-1} \quad b = (1 - per) \cdot u_i^p + per \cdot u_i^{p-1}$$

where $per$ denotes the percentage of membership (100% provides maximum flexibility, whereas 0% gives none – a good default is 75%).

**Step 5:** for each column of points, do:

*Step 5.1*: compute the knots for fitting.
*Step 5.2*: if a knot falls into a membership interval, use the master knot.
*Step 5.3*: otherwise, use this knot, and update the master knot vector by inserting this knot.
*Step 5.4*: fit the curve with a mix of master knots and newly computed knots.

This method uses existing knots to minimize the introduction of new knots. As the fitting proceeds from column-by-column, the master knot vector gets dynamically updated to where no new knots need to be used. In the end, the common knot vector is the master knot vector, and all the knot merging process needs to do is add those master knots that are not present in individual curve knot vectors.

## Image Similarity:

The fitting method discussed in the previous section is now used to compute the similarity of two sets of points (obtained from images). Given point sets $P = P_0,\cdots,P_n$ and $Q = Q_0,\cdots,Q_m$ , we first fit surfaces $S_P(u,v)$ and $S_Q(u,v)$ to these points. Then the difference surface is computed via symbolic algebra:

$$D(u,v) = S_P(u,v) - S_Q(u,v)$$

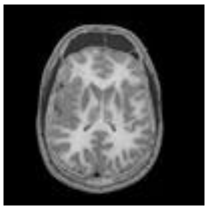
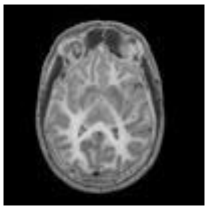
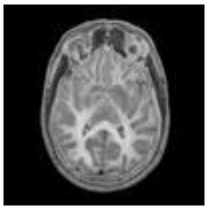
The difference surface is just the origin for identical surfaces, i.e., all the control points of $D(u,v)$ being $R_{i,j} = (0,0,0)$. For surfaces that are not identical, the distances of the control points from the origin are greater than zero. So, the similarity is defined as follows:

$$sim = \frac{Ave \left\| R_{i,j} \right\|}{Diag}$$

where $\left\| R_{i,j} \right\|$ is the vector norm of the control points, $Diag$ is the diagonal of the bounding box of the surfaces, and $Ave$ denotes the average of the control vectors. That is, the similarity is not defined by the largest deviation but rather by the average deviation. Also, the average deviation is scaled by the diagonal of the bounding box to make it independent of the size of the image.

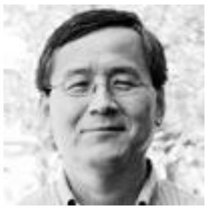
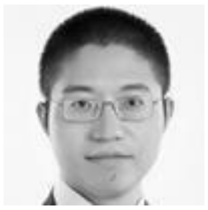
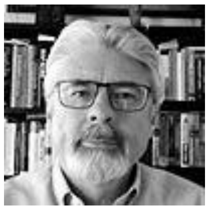
A powerful capability of this similarity method is that it can restrict the computation to the area of interest. When analyzing images, especially medical images, there is always an area of interest, and there are areas that are just background noise. These can be easily excluded by extracting the sub-surface $D_{u_l,u_r \times v_b,v_t}(u,v)$ by specifying the parametric sub-domains $u_l,u_r$ and $v_b,v_t$ in u- and v-direction. Similarity computation is then performed only on this sub-surface.

Examples:
To test the feasibility and effectiveness of the proposed method, we used slices from the Visible Human Dataset (courtesy of the National Library of Medicine, National Institute of Health). If the algorithm works, neighboring slices should be more similar than far apart slices. Table 1 shows comparisons among three slices: 1081, 1093, and 1094. It is assumed that the surfaces are parametrized over the unit square [0,1] x [0,1].

| Areas of interest |  P=1081 |  Q=1093 |  R=1094 |
|---|---|---|---|
| | $P \leftrightarrow Q$ | $Q \leftrightarrow R$ | $P \leftrightarrow R$ |
| [0.0,1.0] x [0.0,1.0] | 0.019363 | **0.011520** | 0.017232 |
| [0.4,0.6] x [0.2,0.8] | 0.031804 | **0.013924** | 0.028479 |
| [0.0,0.2] x [0.0,1.0] | 0.015742 | **0.005033** | 0.014998 |
| [0.0,0.1] x [0.0,0.1] | 0.009528 | **0.002450** | 0.009087 |
| [0.0,0.5] x [0.0,0.5] | 0.021718 | **0.008188** | 0.022999 |
| [0.5,1.0] x [0.5,1.0] | 0.016638 | 0.015238 | **0.011216** |

Tab. 1: Similarities among slices 1081, 1093, and 1094.

Interestingly, the algorithm declares slices 1093 and 1094 the most similar images in all cases, except when the top right corners are compared. The discrepancy can be due to noise or too low sampling. We also tested the method on images of faces, Table 2. The faces shown in the table below do not represent a close similarity. In fact, they seem to be entirely dissimilar. Yet, the question remains: which pair of faces would a human declare the most similar, and which one would the algorithm choose?

| Areas of interest |  P |  Q |  R |
|---|---|---|---|
| | $P \leftrightarrow Q$ | $Q \leftrightarrow R$ | $P \leftrightarrow R$ |
| [0.0,1.0] x [0.0,1.0] | 0.050683 | **0.038743** | 0.065256 |
| [0.4,0.6] x [0.1,0.9] | 0.057151 | **0.032226** | 0.100219 |
| [0.0,0.2] x [0.0,1.0] | 0.059920 | **0.052283** | 0.066503 |
| [0.4,0.6] x [0.4,0.6] | 0.057688 | **0.032273** | 0.100367 |
| [0.45,0.55] x [0.48,0.52] | 0.056100 | **0.032873** | 0.103830 |
| [0.0,1.0] x [0.0,0.2] | 0.049431 | **0.037884** | 0.064119 |

Tab. 2: Similarities of different facial images.

The results are quite compelling: no matter how we zoom in, images Q and R are declared the most similar, and in many cases, by a large margin. There may be a discrepancy between visual appeal and the dynamics that is inherent in the point distribution. Even though the facial characteristics are noticeably different, the method declares Q and R the winners.

## Conclusions:

A point cloud-based method of similarity computation is presented in this paper. The approach addresses a few technical challenges: (1) how to turn an image into a point cloud, (2) what is the optimum number of points to obtain acceptable similarity metrics, (3) how to arrange the points for B-spline surface fitting, and (4) how to use the B-spline surfaces to get a similarity metric. The advantage of the method lies in its simplicity and generality: it works with points, and one can choose the required level of detail (accuracy) by varying the intensities and the resolution.

There are a few disadvantages to the approach. First, it requires a powerful suite of B-spline functions to fit the point cloud. And the second is that the number of points in the cloud can be quite large, making the algorithm run slowly. However, the similarity is computed quickly once the images are converted and stored as B-splines.

## References:

[1] Lau, D. L.; Arce, G. R.: Modern Digital Halftoning, Marcel Dekker, Inc., New York, NY, 2001.

[2] Piegl, L. A.; Tiller, W.: The NURBS Book, Springer-Verlag, Heidelberg, Germany, 1997. https://doi.org/10.1007/978-3-642-59223-2

[3] Piegl, L. A.; Tiller, W.: Surface approximation to scanned data, The Visual Computer, 16(7), 2000, 386-395. https://doi.org/10.1007/PL00013393

[4] Piegl, L. A.; Tiller, W.: Reducing control points in surface interpolation, IEEE Computer Graphics, and Applications, 20(5), 2000, 70-74. https://doi.org/10.1109/38.865883

[5] Piegl, L. A.; Tiller, W.: Fitting NURBS spherical patches to measured data for reverse engineering, Engineering with Computers, 24, 2008, 97-106. https://doi.org/10.1007/s00366-007-0076-8

[6] Ulichney, R.: Digital Halftoning, The MIT Press, Cambridge, MA, 1987.