Intelligent Visual Computing [Spring 2018]

Assignment 4: Surface reconstruction from point clouds

Overview

In this assignment you will implement three implicit surface reconstruction algorithms to approximate a surface represented by scattered point data. The problem can be stated as follows:

Given a set of points $\mathbf{P} = \{\mathbf{p}_1,\mathbf{p}_2,\ldots,\mathbf{p}_n\}$ in a point cloud, we will define an implicit function f(x,y,z) that measures the signed distance to the surface approximated by these points. The surface is extracted at f(x,y,z) = 0 using the marching cubes algorithm. All you need to implement are three implicit functions that measure distance in the following ways: (a) signed distance to tangent plane of the surface point nearest to each point (x,y,z) of the grid storing the implicit function (b) Moving Least Squares (MLS) distance to tangent planes of the K nearest surface points to to each point (x,y,z) of the grid storing the implicit function, (c) Radial Basis Function (RBF) interpolation for approximating the signed distance function. Matlab provides an implementation of marching cubes (function 'isosurface') and K-nearest neighbor point search (function 'knnsearch'). Thus, all you need to do is to fill the code in the specified scripts to implement the above implicit functions. The implicit functions rely on surface normal information per input surface point. In the provided test data files, surface normals are included (the format of the point cloud file is: 'point_x_coordinate' 'point_y_coordinate' 'point_z_coordinate' 'point_normal_x' 'point_normal_y' 'point_normal_z' [newline]). There are three test point cloud (bunny-500.pts, bunny-1000.pts, and sphere.pts) that you will experiment with. Download the starter code and data here: starter_code.zip.

This assignment counts for **15 points** towards your final grade.

What You Need To Do (15 points in total)

[3 points] One way to estimate the signed distance of any point $\boldsymbol{p}=\{x,y,z\}$ of the 3D grid to the sampled surface points $\boldsymbol{p}_i$ is to compute the distance of $\boldsymbol{p}$ to the tangent plane of the surface point $\boldsymbol{p}_i$ that is nearest to $\boldsymbol{p}$. In this case, your signed distance functon is:

$f(\mathbf{p})=\boldsymbol{n}_j\cdot(\boldsymbol{p}-\boldsymbol{p}_j)$ *with* $j=\text{argmin}_i\{\,||\boldsymbol{p}-\boldsymbol{p}_i||\,\}$

Your task: Implement this distance function in the 'naiveReconstruction' script. Show screenshots of the reconstructed bunny (500 and 1000 points) and sphere in your report.

[5 points] The above scheme results in a C^0 surface (i.e., the derivatives of the implicit function are not continuous). To get a smoother result, the Moving Least Squares (MLS) distance from tangent planes is much more preferred. The MLS distance is defined as the weighted sum of the signed distance functions to all points $\boldsymbol{p}_i$:

$f(\mathbf{p})=\boldsymbol{\Sigma}_i d_i(\mathbf{p})\varphi(||\boldsymbol{p}-\boldsymbol{p}_i||)\ /\ \boldsymbol{\Sigma}_i\varphi(||\boldsymbol{p}-\boldsymbol{p}_i||)$

where:

$d_i(\mathbf{p})=\boldsymbol{n}_i\cdot(\boldsymbol{p}-\boldsymbol{p}_i)$

$\varphi(||\boldsymbol{p}-\boldsymbol{p}_i||) = \exp(-||\boldsymbol{p}-\boldsymbol{p}_i||^2/\beta^2)$

Practically, computing the signed distance function to all points $\boldsymbol{p}_i$ is computationally expensive. Since the weights $\varphi(||\boldsymbol{p}-\boldsymbol{p}_i||)$ become very small for surface sample points that are distant to points $\mathbf{p}$ of the grid, in your implementation you will compute the MLS distance to the K=20 nearest surface points for each grid point.

Your task: Implement this distance function in the 'mlsReconstruction.m' script. You will also need to compute an estimate of $(1/\beta^2)$. Set β to be twice the average of the distances between each surface point and its closest neighboring surface point. Show screenshots of the reconstructed bunny (500 and 1000 points) and sphere in your report.

[6 points] A more sophisticated approach is to determine a signed distance function that results in a surface with optimal fairness. This is done by defining the distance function as a weighted sum of RBF basis functions with centers $\boldsymbol{c_k}$:

$f(\mathbf{p})=\boldsymbol{\Sigma_k} w_k \varphi(||\boldsymbol{p}-\boldsymbol{c_k}||)$

where w_k are unknown coefficients of the triharmonic basis functions:

$\varphi(r)=r^3$, $r = ||\boldsymbol{p}-\boldsymbol{c_k}||$

For our purposes, the centers of the basis functions will be the sampled points in addition to offset points: $\{\boldsymbol{p_i}, \boldsymbol{p_i}+\varepsilon\, \boldsymbol{n_i}\}$ (total 2N centers where N is the number of input points). The solution for the weights comes from the following constraints:

$f(\boldsymbol{p_i}) = 0$ [on-surface constraints]

$f(\boldsymbol{p_i}+\varepsilon\, \boldsymbol{n_i}) = \varepsilon$ [off-surface constraints]

where *ε* is a small offset provided by the user.

Given these constraints, we can solve the following linear system:

$\mathbf{M} \cdot \mathbf{w} = \mathbf{d}$

where M is a 2N x 2N matrix whose contents are $\varphi(\ ||\boldsymbol{c_k}-\boldsymbol{c_l}||\ )$, $\mathbf{w}=\{w_k\}$ is the 2N x 1 vector of unknowns, **d** is the 2N x 1 vector of distances.

Your task: Implement the above implicit function in the 'rbfReconstruction.m' script. Include in your report the value of ε yielding the best reconstruction results and show screenshots of the reconstructed bunny (500 and 1000 points) and sphere for that ε value in your report.

[+1 point] Provide the derivation for the optimal scale s given the scale-symmetrical error function used in ICP.

Bonus

[+1 point] Implement the augmented RBF method in a separate script (see p(x) tem in http://www.rbfsrus.com/pdfs/SIGGRAPH_01/carr-beatson-etal-siggraph01.pdf)

Submission

Please follow the **Submission** instructions in the course policy to upload your zip file to Moodle. The zip file should contain all your Matlab code **plus a short PDF report including the results**.

- starter_code.zip

Submission status

Submission status	No attempt
Grading status	Not graded
Due date	Friday, March 23, 2018, 11:55 PM

Time remaining	11 days 10 hours
Last modified	-
Submission comments	Comments (0)

Add submission

Make changes to your submission