METU **Department of Computer Engineering** **Spring 2019**

CENG 789—Digital Geometry Processing

ASSIGNMENT I Mar 1, 2019
Instructor: Y. Sahillioğlu 2 Weeks

Distances and Descriptors on Meshes (start early!)

3D Graphics Programming Setup {*0 points*}
You can use any 3D SDK in this and all other assignments. For OpenGL, OpenMesh, OSG, and libigl setup, see http://youtu.be/D9B_cxUB_LU, http://www.openmesh.org, https://youtu.be/1l5PAVCj2iY, and http://libigl.github.io/libigl, respectively. I prefer an SDK called Open Inventor which is built on top of OpenGL. It comes with a nice viewer that provides a trackball navigation system as well as different rendering modes[1] so that you do not have to implement these features using OpenGL's primitive GLUT or something. Here is how you can set it up on Windows (see also openinventor.com for recent version):

1) Download Coin3D, an independent implementation of API: ceng.metu.edu.tr/∼ys/Coin3D.zip

2) Unzip it to, say, C:\Coin3D

3) Visual Studio → New Project → Visual C++ → Win32 Console Application → Application Settings → Empty Project

4) Place your code file, e.g., ∼ys/HelloWorld.cpp, under Source Files

5) Right click on your Project Name and then hit Properties → C/C++ → General → Additional Include Directories: write C:\Coin3D\include;

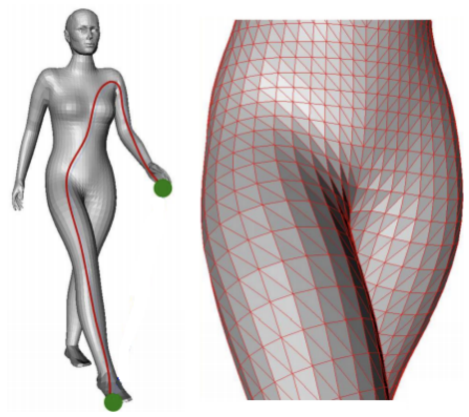
6) C/C++ → Preprocessor: add COIN_DLL;SOWIN_DLL;

7) Linker → General → Additional Library Directories: write C:\Coin3D\lib;

8) Linker → Input → Additional Dependencies: add coin2d.lib;sowin1d.lib;

9) Paste Coin3D\coin2d.dll and sowin1d.dll files into your-work-folder\Debug folder (google other missing files reported when you try to execute the program, if any, and put them in this folder too, e.g., MSVCR71D.dll)

Geodesic Distances on Meshes {*30 points*} Geodesic distance between two mesh vertices is the length of the shortest path along the surface that connects the two, as shown below. Since surface is discretized as an undirected graph, all you have to do is implement the Dijkstra's Shortest Path algorithm discussed in the class. Print the $N \times N$ geodesic distance matrix M to file for the N-vertex input mesh. Visualize the path between two query points, e.g., by redrawing the edges on the path with thicker red lines as below. Report timings with three structures for Q: array, min heap, and Fibonacci heap (disable fprints).

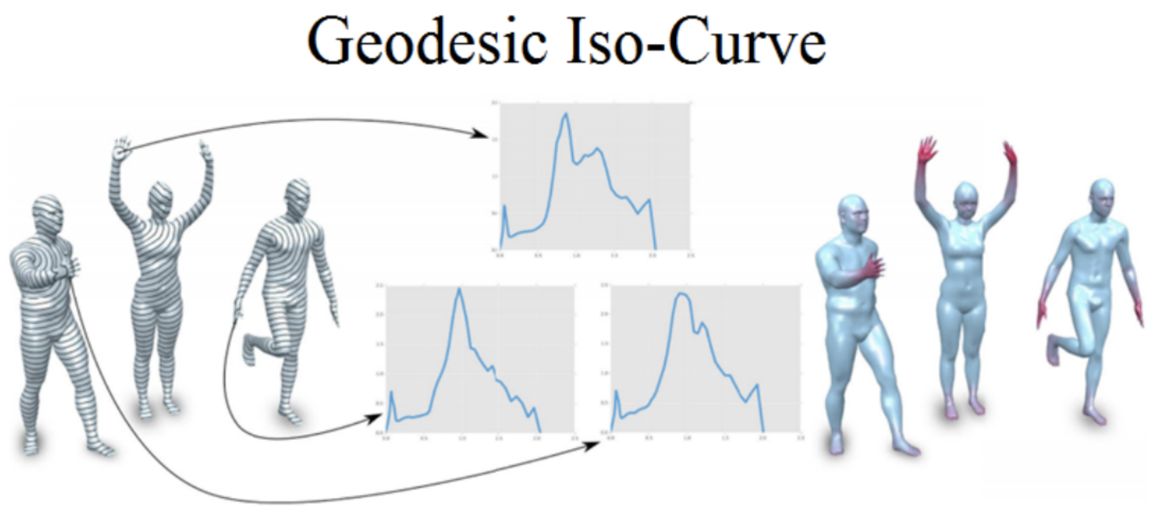


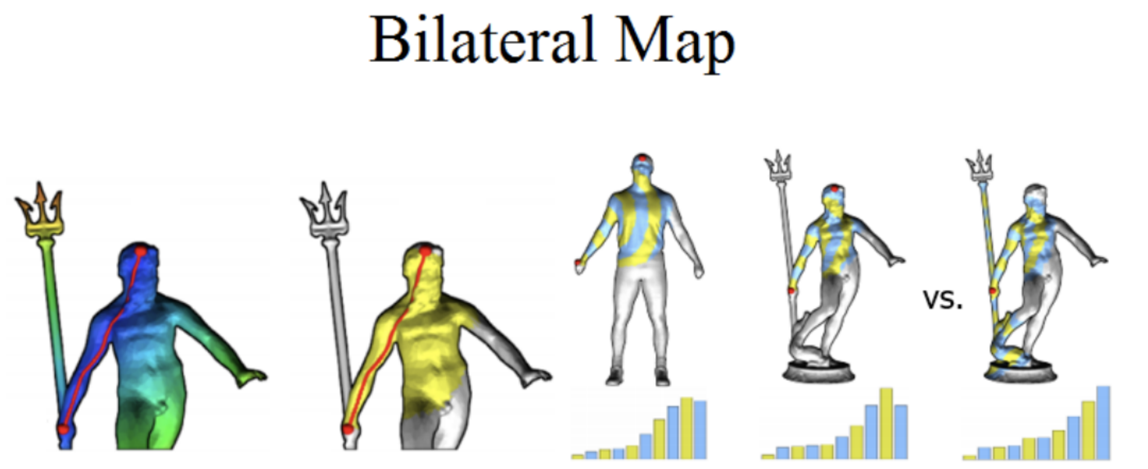
[1]See my short demo at http://youtu.be/lK7aoc1AO8w

Sampling and Descriptors on Meshes {*70 points*} Compute a) Geodesic Iso-Curve Signature (25 points) and b) Bilateral Maps (25 points) descriptors for 100 mesh vertices sampled via FPS (20 points).

For a), draw k iso-curves on mesh and find a way to color the seed vertex based on its histogram. Recall from class that this histogram stores lengths of iso-curves for k different radii. Read Section 4.2 of the paper to compute the lengths.

For b), take 2 seed vertices as input and draw the shortest path p between them. Then color the remaining vertices based on their distances to p. Define your region of interest (ROI) as the set of vertices that are close (τ) to p. For the descriptor on one of the two seeds, say a, create an histogram which stores the total triangle area of the corresponding subregion of ROI. This subregion is at distance d to a, where d gives you the histogram bins. For 33 points bonus (and a potential paper publication), consider 3 seed points to define the ROI. The advantage of this scheme is the elimination of the closeness parameter τ.





Submission This assignment constitutes 20% of your final grade. Use the meshes provided in ~ys/meshes1.zip. Send to ys@ceng.metu.edu.tr your code, executable, output screenshots, and resulting file as well as mynotes.txt file where you mention the encountered problems and interesting observations.