Advanced Topics in Computer Graphics II

Universität Bonn
Institut für Informatik II
January 7, 2020

Winter term 2019
Prof. Dr. Reinhard Klein
Alexander Dieckmann, Lukas Bode

# Sheet G1 - Practical Geometry Project
UPDATE

Project Introduction

During the first part of the lecture you are required to do a practical project in **groups of 2** (and a single group of 3 for an odd number of students). The project shall give you further insight into the process of the geometry processing pipeline.

There will be scheduled meetings and you are expected to be able to present the results regarding the corresponding milestone. Thus, in every meeting at least one member of **each group has to be present** and capable of presenting. Each member of a group has to be able to explain the basics of what every single part of the group's code does. Therefore, we encourage you to talk about your code in the group. You have to **complete every single milestone in time** in order to be allowed to take the exam. The meeting dates and milestones are described below in more detail.

Your pipeline has to be written in **C++ using OpenGL**. We strongly recommend to use **CMake** as build system and **git** as version-control system. You are free to use libraries which provide basic functionality (GLEW, Glad, GLFW, GLM, NanoGUI, Assimp, tinyOBJ, libigl...), however, we expect you to implement the main parts of the pipeline by yourself.

You can find lots of tutorials about C++, OpenGL, and the mentioned libraries on the internet. Regarding C++ and OpenGL in general you can take a look at learnopengl. It covers basically everything you need to know for this lecture and much more. More information about specific parts of C++ can of course be found at cppreference. Cmake and git have good tutorials on the respective websites. The libigl contains a comprehensive tutorial on many geometry processing applications. **If you have enough confidence in your renderer implemented in the rendering part, you might also use your own implementation.**

If you have problems, questions, ... please write an email to the mailinglist (**Vl-atcg2@lists.iai.uni-bonn.de**) or the tutors (**lbode@cs.uni-bonn.de**, **dieckman@cs.uni-bonn.de**).

Schedule (subject to change)

During the next 7 weeks, we will have scheduled meetings **16:00 - 17:30 o'clock in room 3.005**. The dates are:

- Tue, 10.12.19 Intro
- Tue, 17.12.19 Milestone 1
- Tue, 07.01.20 Milestone 2
- Tue, 21.01.20 Milestone 3
- Tue, 28.01.20 Milestone 4 & Showcase

Introduction

Until the intro meeting consider the following: The provided dataset are scanned meshes of plaster casts of the upper and lower teeth rows of 30 different children. Download the dataset from the webpage and try to render the meshes with a simple shading in libigl or your own implementation. Play with libigl to get familiar with how it works, look into the code and do some of the tutorials on their webpage.

Milestones

The following milestones have to be completed in time. For each milestone there will be a detailed description on the webpage describing each milestone.

Milestone 1 As a first milestone, you have to compute the main symmetry plane of a mesh by using a simple icp algorithm. The mirror reflection of a point x with respect to a plane through the origin and with normal vector v is given by:

$$x \mapsto S_v x$$

where $S_v = I - 2vv^T$ and I is the identity matrix. The reflection with respect to a plane through an arbitrary point p and with normal vector v is given by:

$$x \mapsto S_{p,v}x = S_v x + 2dv$$

where $d = \langle p, v \rangle$ is the signed distance between the plane and the origin.

Algorithm:

1. Choose initial reflection plane, given by a point p and a perpendicular vector v. For example, p can be the average (center of mass) of the points in P and $v = (1, 0, \dots, 0)$.
2. Reflect all points $p \in P$:
$$d = \langle p, v \rangle$$
$$x \mapsto S_{p,v}x = S_v x + 2dv$$
to obtain a new set of points Q
3. Register Q and P through a rigid transformation, obtaining a rotation matrix R_0 and a translation vector t. That is the registration transform is given by $x \mapsto R_0 x + t$
4. Compute the eigenvector $\bar{v}$ of the matrix $S_v R_0^T$ corresponding to the eigenvalue -1, where $\bar{v}$ is the vector perpendicular to the symmetry plane.
5. Completely define the symmetry plane by computing a point $\bar{p}$ through which it passes
$$\bar{p} = \frac{1}{2}(R_0(2dv) + t)$$

You should visualize the mesh, the sampled points, the mirrored points and the plane in each frame.

Milestone 2 Extend the symmetry detection algorithm by computing salient feature points on the mesh first. Then pick the random point set only from these feature points. There are different approaches to compute salient points on meshes presented in the slides:

- **mesh saliency**: Let v be vertex of the mesh and $C(v)$ its curvature value.
Define $N(v, \sigma) = x | \|x - v\| < \sigma,$ x mesh point the set of points in a spherical neighborhood around v. Set
$$G(C(v), \sigma) = \sum_{x \in N(v, 2\sigma)} \frac{C(x) \exp - \|x - v\|^2/(2\sigma)^2}{\sum_{x \in N(v, 2\sigma)} C(x) \exp - \|x - v\|^2/(2\sigma)^2}$$

The saliency of vertex v is then

$$S(v) = |G(V(v), \sigma) - G(C(v), 2\sigma)|$$

The scale dependent saliency is

$$S_i(v) = |G(V(v), \sigma_i) - G(C(v), 2\sigma_i)|, \quad \text{with } \sigma_i \in \{2\epsilon, \dots, 6\epsilon\}$$

The value for ϵ can be some percentage of the bounding box of the object.

- **integral invariant signatures**: Fill an occupancy grid with the mesh volume. Mark every grid cell occupied that is inside the mesh.
  Then define a spherical kernel (ball $B_r(p)$ of radius r) and integrate (sum up) all occupied grid cells at vertices p of the mesh:

  $$V_r(p) = \int_{B_r(p) \cap S} dx$$

- **salient geometric features for partial shape matching and similarity**: Fit implicit quadratic surface patch F to locally estimate curvature properties at all vertices of the mesh. First transform vertex v to homogeneous coordinates. The quadratic surface patch is the solution of

  $$F(\begin{bmatrix} x & y & z & 1 \end{bmatrix}) = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} a & e & g & l \\ e & b & f & m \\ g & f & c & n \\ l & m & n & d \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0$$

  Minimize the algebraic distances in local coordinates

  $$\min_{a,b,c,d,e,f,g,l,m,n} [\sum_{[x,y,z] \in N(v,\sigma)} F^2(x, y, z)]$$

  subject to the constraint $a^2 + b^2 + c^2 + d^2 + 2e^2 + 2f^2 + 2g^2 + 2l^2 + 2m^2 + 2n^2 = 1$ which avoids the trivial solution.
  - You can ether minimize the least squares problem using the svd or formulate this as an eigenvalue problem.
  - When you use the eigenvalue problem the solution is found from the components of the eigenvector with the lowest eigenvalue.

  Cluster vertices using a region growing algorithm, starting at vertices with maximum gaussian curvature. Descriptor d from neighboring patches are clustered into cluster C by this region growing algorithm as long as the saliency grade $S(C)$ of the cluster C increases:

  $$S(C) = \sum_{d \in C} W_1 Area(d) Curv(d)^3 + W_2 N(C) Var(C)$$

  where
  - W_1, W_2 are weights, e.g. $= 0.5$
  - $Area(d)$ is the area of patch associated with d relative to the size of its bounding sphere
  - $Curv(d)$ is the curvature associated with d
  - $N(C)$ is the number of minima or maxima
  - $Var(C)$ is the variance of the curvatures in C

- **topology matching for fully automatic similarity estimation of 3d shapes**: For each vertex v on the mesh M compute the geodesic distance field to uniformly distributed point set $S = \{s_1, \dots, s_k\}$

  $$G(v) = \frac{1}{k} \sum_{i=1}^{k} g(v, s_i), \quad v \in M$$

  The value $G(v)$ approximates the average squared geodesic distances from vertex v to all other vertices. The local maxima of G correspond to shape extemities (either convex or concave) while the local minima lie near the center of the shape.

Pick two approaches and compare the produced feature points and their performance in the symmetry algorithm. Feature points are those points which are rare in the models (count the descriptors).

Milestone 3 Segment the meshes by separating the each tooth from the mesh. Visualize the plaster cast by applying a unique color for each tooth. You will need 14 different colors to color each tooth. If you like you can also visualize the individual steps of the segmentation algorithm.

To get a segmentation of the teeth in a plaster cast we propose the following paper as a feasible approach. If you know of any other feasible method, please contact me in advance and we can discuss whether you can implement that approach instead.

Paper: Automatic Tooth Segmentation of Dental Mesh Based on Harmonic Fields

The algorithm works as follows:

a) Identify some stable features on each tooth and assign them to a bin corresponding to the tooth. You need 14 bins per plaster cast.

b) Compute a cutting plane which separates the plaster cast in two parts, one containing all teeth and the other containing the unused rest.

c) Compute the signed mean curvature on the mesh.

d) Set constraints for the computation of a harmonic field, which will separate the teeth from the plaster cast. The constraints need to be set in the following way: The features on a tooth are set to 0 or 1 in such a way that neighboring teeth have a different value (alternating). The cut-boundary from the cutting plane is set to the value 0.5.

e) Construct the linear system as described in the paper and solve for the harmonic field by solving the system using a negative curvature scaled Laplacian.

f) Cut the mesh where the harmonic field has value 0.

This task should provoke some discussion, so i expect you to use the mailing list. Please ask questions and comment on different problems of the task you encounter...

Milestone 4 Find the rigid transformations which align the corresponding tooth meshes between two plaster casts. Given this set of rigid transformations compute the average transformation which minimizes the overall alignment error between two plaster casts, only considering the tooth geometry from the segmentations. Take special care when you compute the average rotations, since this is not a trivial task. This registration process will be used to compute a morphable model of the plaster casts.

Compute a morphable model for the dataset. Visualize the main modes of variation and show some random samplings from the morphable model.

Good luck !