Final Project List

Computer Graphics – Final Project

Xiaoshuang Li (TA)

9/28/2018

Student Information and Topic Selection

39 groups, 20 topics.

At most 2 group can select one topic at the same time.

Selection process will be done through mini program *WeChat voting* at 8 PM of September 29. After selecting, You can see your selection order under the choices, **make sure that you are the first two selectors**. If you are the third or behind, you should change your selection by another click and choose other topics. So making a list for your preferences in advance will help you make quick decision.

Topics field: Reconstruction / AI / VR / Medical Image / Rendering

Final evaluation: Presentation, which should contains **work basics, your contributions and some demonstration of results.** Presentation will be held in a cross-disciplinary forum titled *Artificial Intelligence and Translational Medicine Academic Forum* at the end of December *(12.14-12.26).*

Evaluation

Questions without a reference code are required to be implemented, evaluated based on workload.

Questions with reference code require improvements based on code, evaluated based on workload.

For topics with subtasks, you only need to choose one part such as classification / segmentation / reconstruction tasks.

TA Information

	WeChat ID	Email	Project
Xiaoshuang Li	frost0407	lixiaoshuang@sjtu.edu.cn	SEG, AI, AG, P
Jiefeng Fang	Datoclement	jiefeng.fang@polytechnique.edu	SEG, AI, AG, P
Chao Dai	Daichao002	dc_ecust@qq.com	VR
Jianwei Jiang	avividweixin	vividmailbox@163.com	RD
Yiming Qin	blucewen	lhqym97@163.com	VR

Project List for Computer Graphics – 2018

1. Reconstructing teeth from a CT scan (SEG)

A 3D CT scan consists of a sequence of 2D scanned slices, each of which records the tissue density of a layer of the human body and is represented by gray values. Segmentation tasks are easy to implement for tissues with large differences in density, such as bones and soft tissues. The density difference between tooth and mandible is relatively small, thus the task of segmenting teeth and performing 3D reconstruction from CT scans is more challenging than the usual segmentation work.
Requirements: In this assignment, you will be given dataset of CT scans retrieved from hospital. The goal of this assignment is to reconstruct 3D polygonal meshes that represent the teeth(include roots). Segmentation job can be done using 2D (for example Segnet) or 3D segmentation network, then with **edge extraction** or other image processing methods you may reconstruct a 3D model layer by layer.
Data: CT scans which contains a binary entry file and a set of image file with dcm format. The latter one can be read and processed with the python library 'mudicom'.
Route 1: 2D Segmentation + 3D Reconstruction
Route 2: 3D Clustering + 3D Reconstruction

2. Learning Transformation Matrix of Teeth in Orthodontic Process(AI1)

Taking the orthodontic process as an example. After orthodontics, the state of each tooth can be represented by the initial tooth models multiplied by a rigid transformation matrix.
Requirements: Construct a spatial feature learning network that learns from 3D models and the output of deep network is a 32*16 transformation matrix.
Data: 32 initial tooth crown models and 32*16 matrices, every row of which represents one 4*4 transformation matrix of corresponding tooth.
Reference: https://github.com/zhirongw/3DShapeNets

3. Learning Transformation Matrix of Teeth in Orthodontic Process with CT scans (AI2)

Feature extraction using only teeth crown models can lead to some errors because the external spatial influence factors such as teeth root distribution and jaw bone distribution are not taken into account. Adding CT data as the basis of learning can improve the accuracy of the learned transformation matrix.
Requirements: Construct a spatial feature learning network that learns from multi-source 3D models and the output of deep network is a 32*16 transformation matrix.
Data: 32 initial tooth crown models, CT scans of the same individual and 32*16 matrices, every row of which represents one 4*4 transformation matrix of corresponding tooth.
Reference: https://github.com/zhirongw/3DShapeNets

4. 3D model reconstruction based on multi-view image (AI3)

Data: 3D models of tooth crown, and corresponding multi-view images of tooth.
Route: Construct a spatial feature learning network that learns to extract features from multi-view images to reconstruct 3D models.
Reference: https://github.com/chrischoy/3D-R2N2

5. Classification task of 3D Model of Teeth (AI4)

In this assignment, you need to train a network to make 3 classification decisions when it is fed with a new 3D model of tooth crown. First, whether or not this patient needs tooth extraction (true/false). Second, the place of tooth extraction. Teeth can be marked as 1-32 and the classification results is an array[32] of extraction or not (true/false). Last, whether the patient needs surgery (true/false).
Data: 3D models of tooth crown, and corresponding labels (extraction/position of extraction/surgery) for 3 classification works.
Route: Construct a deep neural network that learns from 3D models of tooth crown and make 3 classification decisions for this models.
Reference: Many 3D classification networks can help in this tasks.

6. Attachment Strategy Learning (Timing and Location) (AI5)

During the orthodontic process, some attachments would be added to teeth, to add a force point to the tooth surface. The strategy of adding attachment contains 3 part of decisions. When to set an attachment, which kind of attachment and which tooth it should be set on. That is, orthodontic steps count *s(5-15),* attachment type *c(0-10)*, And attachment position *p(0-31).* triplet <s,c, p> can be view as 3D coordination in latent space.
Data: initial 3D model of 32 crowns, an array of triplets<s, c, p>
Route: Construct a spatial feature learning network that learns from 3D models and the output of deep network is point cloud.
Reference: https://github.com/charlesq34/pointnet2

7. Conversion between Polygonal Mesh, Point Cloud and Voxel Grid（AG）

Requirements: In this assignment, you need to implement algorithms that deal with the conversion between polygonal mesh, point cloud and voxel grid. You will be given a dataset of 3D models formatted in .obj/.stl/.om. You need to implement algorithms that transform these formats into **voxel** and **point cloud** representation, and also algorithms transform the result **backward to polygonal mesh**. There is already software can solve this problem, please code by yourselves. You will be required to present the result in a demo.
Data: Mesh dataset of 3D models formatted in .obj/.stl/.om

For assignment P1-P7, you should follow the leads from github website, manage to run the program, and make some improvement in whatever direction you want. We will evaluate your work based on the workload.

8.Spatially localized deformation（P1）

Spatially localized deformation components are very useful for shape analysis and synthesis in 3D geometry processing. Several methods have recently been developed, with an aim to extract intuitive and interpretable deformation components. However, these techniques suffer from fundamental limitations especially for meshes with noise or large-scale deformations, and may not always be able to identify important deformation components. In this paper we propose a novel mesh-based autoencoder architecture that is able to cope with meshes with irregular topology. We introduce sparse regularization in this framework, which along with convolutional operations, helps localize deformations. Our framework is capable of extracting localized deformation components from mesh data sets with large-scale deformations and is robust to noise. It also provides a nonlinear approach to reconstruction of meshes using the extracted basis, which is more effective than the current linear combination approach. Extensive experiments show that our method outperforms state-of-the-art methods in both qualitative and quantitative evaluations.

Paper: Mesh-based Autoencoders for Localized Deformation Component Analysis

Code: https://github.com/aldehydecho/convMesh

9.Deep Representation for Volumetric Shapes（P2）

3D shape is a crucial but heavily underutilized cue in today's computer vision systems, mostly due to the lack of a good generic shape representation. With the recent availability of inexpensive 2.5D deep sensors (e.g. Microsoft Kinect), it is becoming increasingly important to have a powerful 3D shape representation in the loop. Apart from category recognition, recovering full 3D shapes from view-based 2.5D deep maps is also a critical part of visual understanding. To this end, we propose to represent a geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid, using a Convolutional Deep Belief Network. Our model, 3D ShapeNets, learns the distribution of complex 3D shapes across different object categories and arbitrary poses from raw CAD data, and discovers hierarchical compositional part representation automatically. It naturally supports joint object recognition and shape completion from 2.5D deep maps, and it enables active object recognition through view planning. To train our 3D deep learning model, we construct ModelNet - a large-scale 3D CAD model dataset. Extensive experiments show that our 3D deep representation enables significant performance improvement over the-state-of-the-arts in a variety of tasks.

Paper: 3D ShapeNets: A Deep Representation for Volumetric Shapes

Code: https://github.com/zhirongw/3DShapeNets

10.Field Probing Neural Networks for 3D Data（P3）

Building discriminative representations for 3D data has been an important task in computer graphics and computer vision research. Lifting convolution operators to 3D (3DCNNs) seems like a plausible and promising next step. Unfortunately, the computational complexity of 3D CNNs grows cubically with respect to voxel resolution. Moreover, since most 3D geometry representations are boundary based, occupied regions do not increase proportionately with the size of the discretization, resulting in wasted computation. In this work, we represent 3D spaces as volumetric fields, and propose a novel design that employs field probing filters to efficiently extract features from them. Each field probing filter is a set of probing points -- sensors that perceive the space. Our learning algorithm optimizes not only the weights associated with the probing points, but also their locations, which deforms the shape of the probing filters and adaptively distributes them in 3D space. The optimized probing points sense the 3D space "intelligently", rather than operating blindly over the entire domain. We show that field probing is significantly more efficient than 3DCNNs, while providing state-of-the-art performance, on classification tasks for 3D object recognition benchmark datasets.
Paper: FPNN: Field Probing Neural Networks for 3D Data

Code: https://github.com/yangyanli/FPNN

11. Octree Generating Networks（P4）

Focus on a deep convolutional decoder architecture that can generate volumetric 3D outputs in a compute- and memory-efficient manner by using an octree representation. The network learns to predict both the structure of the octree, and the occupancy values of individual cells. This makes it a particularly valuable technique for generating 3D shapes. In contrast to standard decoders acting on regular voxel grids, the architecture does not have cubic complexity. This allows representing much higher resolution outputs with a limited memory budget. We demonstrate this in several application domains, including 3D convolutional autoencoders, generation of objects and whole scenes from high-level representations, and shape from a single image.
Paper: Octree Generating Networks: Efficient Convolutional Architectures for High-Resolution 3D Outputs
Code: https://github.com/lmb-freiburg/ogn

12.Deep Learning on Point Sets for 3D Classification and Segmentation（P5）

Point cloud is an important type of geometric data structure. Due to its irregular format, most researchers transform such data to regular 3D voxel grids or collections of images. This, however, renders data unnecessarily voluminous and causes issues. In this paper, we design a novel type of neural network that directly consumes point clouds, which well respects the permutation invariance of points in the input. Our network, named PointNet, provides a unified architecture for applications ranging

from object classification, part segmentation, to scene semantic parsing. Though simple, PointNet is highly efficient and effective.
Paper: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation
Code: https://github.com/charlesq34/pointnet

13.PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space（P6）

PointNet (the v1 model) either transforms features of *individual points* independently or process global features of the *entire point set*. However, in many cases there are well defined distance metrics such as Euclidean distance for 3D point clouds collected by 3D sensors or geodesic distance for manifolds like isometric shape surfaces. In PointNet++ we want to respect *spatial localities* of those point sets. PointNet++ learns hierarchical features with increasing scales of contexts, just like that in convolutional neural networks. Besides, we also observe one challenge that is not present in convnets (with images) -- non-uniform densities in natural point clouds. To deal with those non-uniform densities, we further propose special layers that are able to intelligently aggregate information from different scales.
Paper: Deep Hierarchical Feature Learning on Point Sets in a Metric Space
Code: https://github.com/charlesq34/pointnet2

14.Learning Vector Representation for Objects（P7）

What is a good vector representation of an object? We believe that it should be generative in 3D, in the sense that it can produce new 3D objects; as well as be predictable from 2D, in the sense that it can be perceived from 2D images. We propose a novel architecture, called the TL-embedding network, to learn an embedding space with these properties. The network consists of two components: (a) an autoencoder that ensures the representation is generative; and (b) a convolutional network that ensures the representation is predictable. This enables tackling a number of tasks including voxel prediction from 2D images and 3D model retrieval. Extensive experimental analysis demonstrates the usefulness and versatility of this embedding.
Paper: Learning a Predictable and Generative Vector Representation for Objects
Code: https://github.com/rohitgirdhar/GenerativePredictableVoxels

15.Hair Generation with Single Image （RD1）

This project is aimed at generating 3D hair modelding from a single portrait image, with no user interaction or parameter tuning with high quality. Currently, most of the animated 3D hair is modeled by artists, which is time consuming. We want to give an end-to-end framework by generating a approximate hair style from a single image set. The basic idea is first isolate hair and extract the growth direction from single image using deep learning segmentation network. This data will then be used to match with a large hair database. To conquer this problem, you should have some basic knowledge of deep neural network and computer graphics. Besides, knowing

an 3D software like Maya and Houdini is helpful for data analysis and 3D visualization.
Reference: http://kunzhou.net/zjugaps/autohair/

16.Multiview Dynamic Hair Capturing using Spacetime Optimization （RD2）

Capturing dynamic hair strands movement from multiview is also a challenging task. Different from the "Singe Image Generation" project, the hair is captured with multi camera with a clean environment. The goal of this project to try to build a dynamic system to reconstruct hair movement from multiple synchronized video sequences while keeping the space and time coherance. In this task, you need to develope a motion-path analysing algorithm that can robustly track local hair motions in input videos and transfer the tracking problem into a spacetime optimization problem. Finally, you need to demonstrate your result with 3D animation tools like "mayavi" using python or other 3D softwares.
Reference:
https://www.microsoft.com/en-us/research/video/dynamic-hair-capture-using-spacetime-optimization-2/

17.High Fidelity Human Teeth Simulation and Rendering（RD3）

Traditionally, the teeth are animated skinning mesh techniques. Thus artistis need to control the teeth movement by hands and detailed collisions could not be correctly resolved. Besides, the light scattering of teeth is also challenging, lights can be sub-scattered multiple times for high fidelity teeth rendering. The goal for this project is to develop a simulation framework for teeth simulation or rendering. You should acquire some basic knowledge of ray-tracing techniques and familiar with writing shaders for renderer.

18.Use Unity3D to develop a multiplayer WebVR application（VR1）

With the development of virtual reality technology, more and more people hope to experience VR technology firsthand . WebVR covers a large number of users on the Web, mobile and VR terminals with its multi-platform, convenience features. Unity3d is also a lightweight game development tool with cross-platform and simple development features.
Requirements: Use unity3D to develop a multiplayer WebVR application, where users can perform some simple operations, such as object zooming, object moving, video playing and some other cooperative operations you think are interesting.
Tools: Unity WebVR Assets: https://github.com/mozilla/unity-webvr-export
Photon :Server setup

19.Work on shaders to implement non-uniform mapping.（VR2）

When transmitting information in virtual reality, if rendering images are directly transmitted, some transmission bandwidth will be wasted, so rendering scenes need to be processed.

Requirements: create a scene in unity3D and write shader script by yourself to achieve rendering in the visible region of the camera for normal quality and low-quality rendering in the invisible region.

20.Make a multiplayer interactive application of augmented reality with Unity.（VR3）

Requirements: Multiple users can operate on the same object. While one user is doing this, other users can see the changes in real time, using the form of AR to achieve this effect.

Tools: Unity Vuforia ;Photon