Shape Modeling and Geometry Processing

Exercise 6 - 3D Human Faces

INTERACTIVE GEOMETRY LAB

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Grouping

(1)

- Camilla Casamento Tumeo
- Daniel Sparber
- Jela Kovacevic
- Sebastian Winberg
- Viviane Yang

(2)

- Alessia Paccagnella
- Yingyan Xu
- Xiaojing Xia
- Niklaus Houska
- Alexandre Binninger

(3)

-Ting-Yu Chen
-Wen-Chieh Tung
-Kaiyue Shen
-Zimeng Jiang
-Gene Ting-Chun Kao

(4)

-Tang Jingwei
-Baeza Rojo Irene
-Bartolovic Nemanja
-Jakob Jakob

(5)

-De Keyser Kevin
-Chan Cheuk Yu
-Matti Matthias

(6)

-Lixin Xue
-Dexin Yang
-Kaifeng Zhao
-Minchao Li

(7)

Jan Wiegner	wiegnerj@student.ethz.ch
Jonathan Lehner	lehnerj@student.ethz.ch
Costanza Improta	cimprota@student.ethz.ch
Matej Sladek	msladek@ethz.ch

(8)

Jorel Elmiger	elmigerj@student.ethz.ch
Fabian Ulbricht	fabianu@student.ethz.ch
Peter Gronquist	petergro@student.ethz.ch
Olivier Bitter	bittero@student.ethz.ch
Serquet Manuel	serquetm@student.ethz.ch

(9)

Valentin Weiss	weissva@student.ethz.ch
Philippe Andreu	pandreu@student.ethz.ch
Viturin Zust	vzuest@student.ethz.ch
Yannick Rosskopf	yannicro@student.ethz.ch
Anil Yaris	ayaris@student.ethz.ch

(10)

Hamilton Carrillo Nunez	carrillh@ethz.ch
Allan Benelli	abenelli@student.ethz.ch
Nikola Kovacevic	nikolak@student.ethz.ch
Predrag Krnetic	pkrnetic@student.ethz.ch
Matthias Roshardt	roshardm@student.ethz.ch

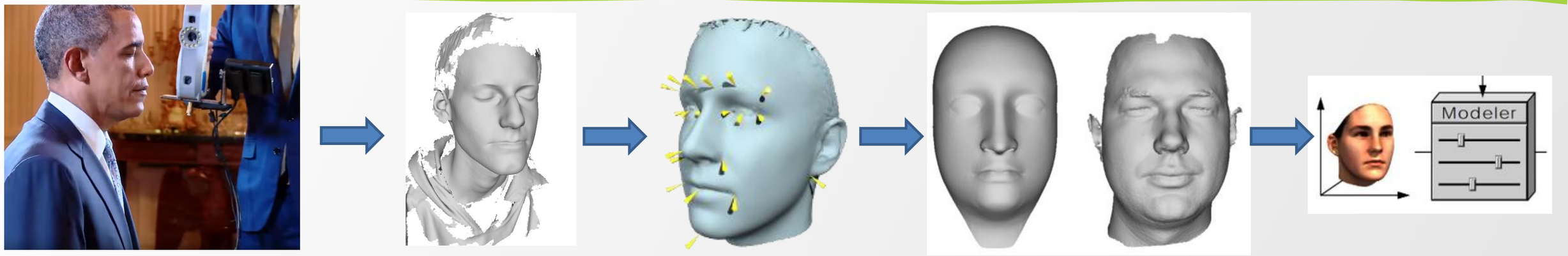
(11)

Maximilian Wolfertz	mwolfert@student.ethz.ch
Anders Hansson	anhansson@ethz.ch
Ioanna Mitropoulou	mitropoulou@arch.ethz.ch
Rudolf Varga	rvarga@student.ethz.ch
Johannes Baureithel	baureitj@student.ethz.ch

(12)

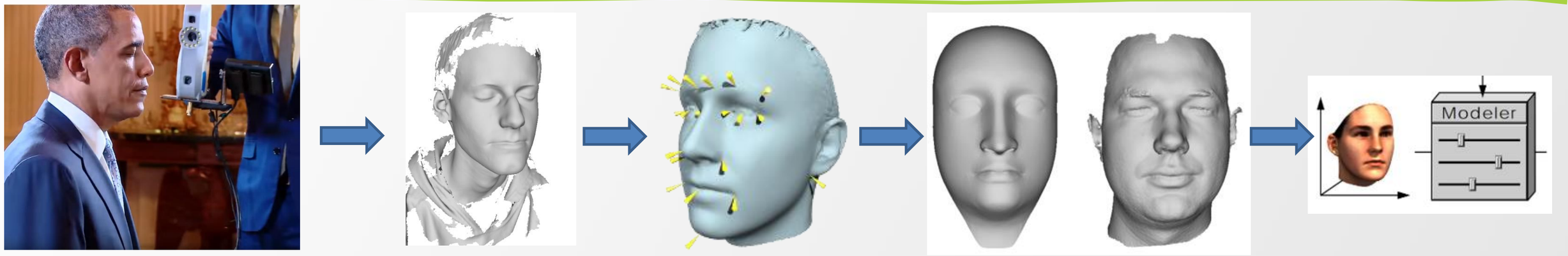
Kenneth Blomqvist	kblomqvist@mavt.ethz.ch
Yuqing Cui	yuqcui@student.ethz.ch
Florin-Alexandru	fvasluianu@student.ethz.ch
Aydin Faraji	afaraji@student.ethz.ch

Goal



- Dealing with high-dimensional data
- Practice data-processing of the whole pipeline
- Practice teamwork
- Learn 3D geometric learning

Overview



Step 0: Decide team leader

Step 1: Download 3D faces.

Step 2: Data preprocessing, face alignment.

Step 3: Compute PCA faces and UI implementation.

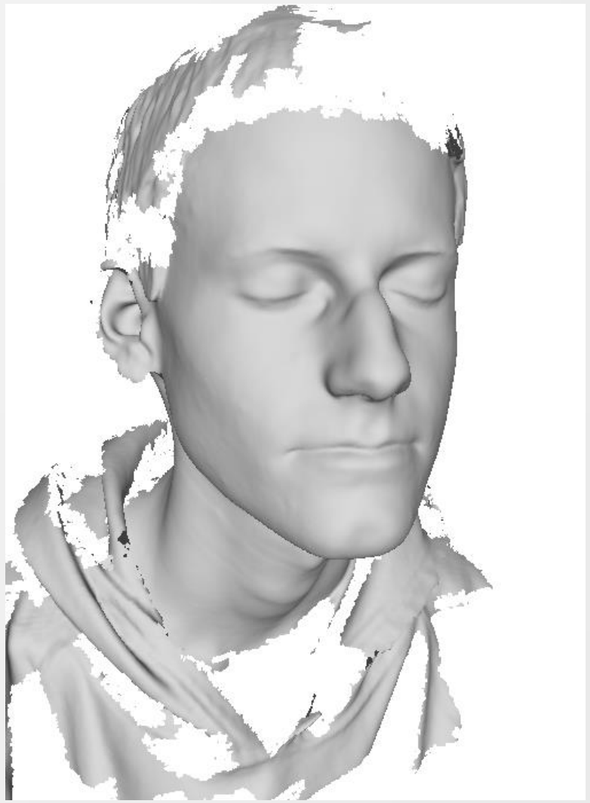
Step 4 (bonus): experiment with different shape space.

Step 5: Demo and final group presentation.

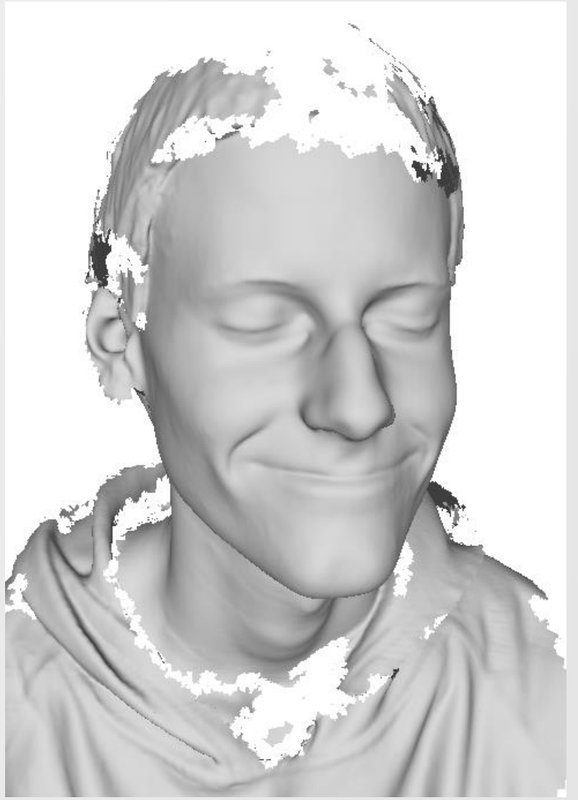
Note: we will provide some example data (scanned faces, face template, aligned faces) so that the implementation of each step should be independent.

Scanning & Preprocessing

- We provide 112 meshes from around 60 different faces, we have cleaned up and removed all the hair and neck parts.
- Better start from a small set
- Inter-group policy



Neutral

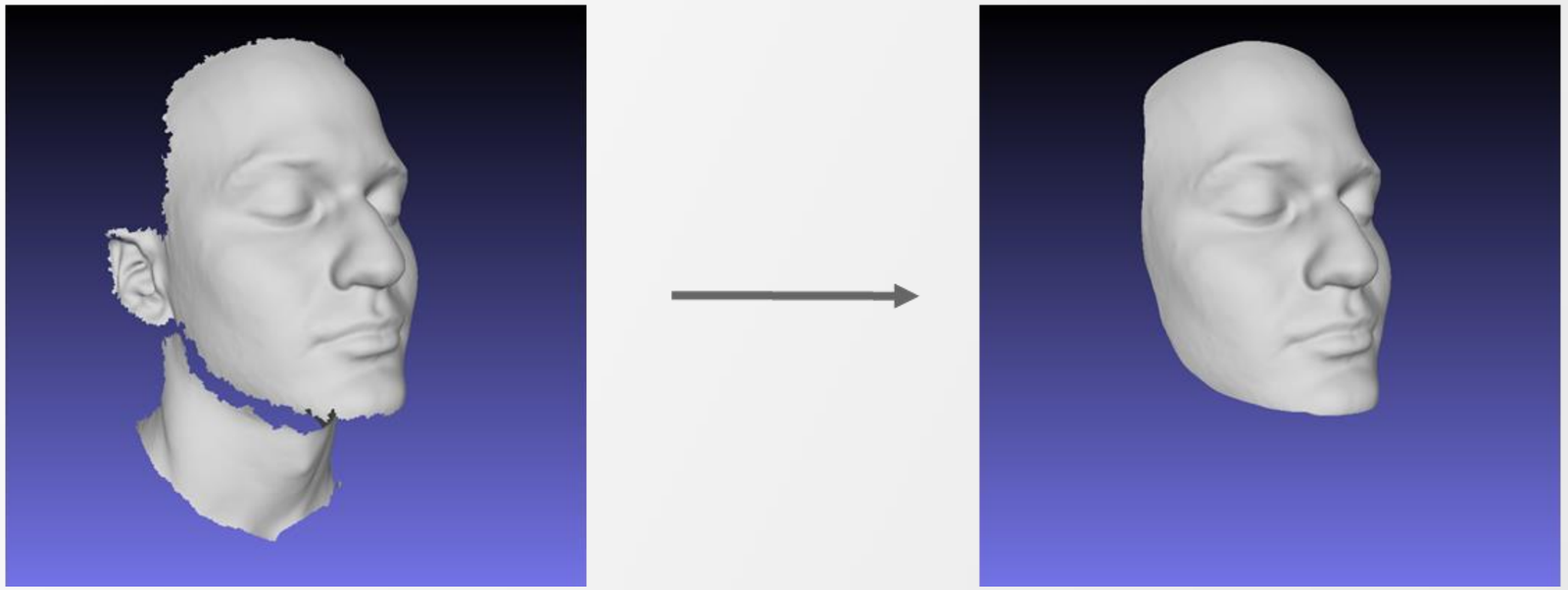


Smile

Scanning & Preprocessing

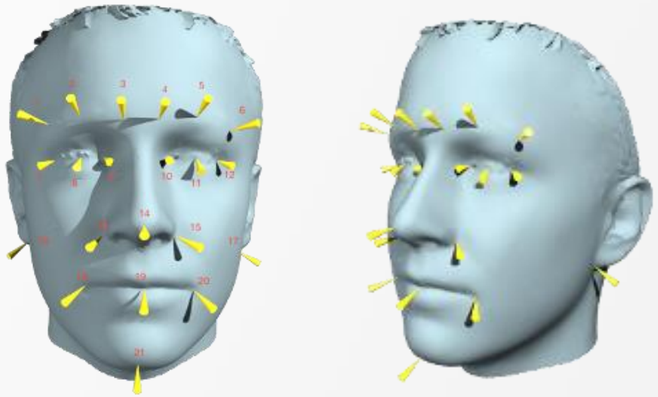
- Clean disconnected components
  (# connected components = 1)

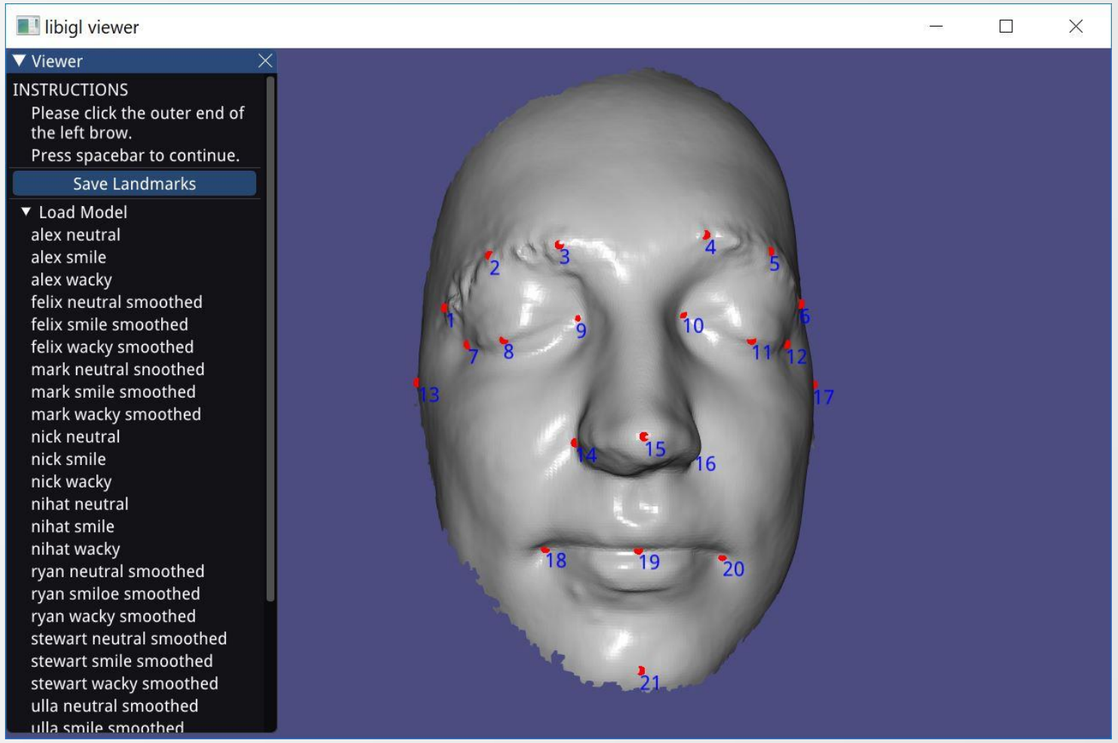
- Standardize face elements for every person
- Smooth mesh boundaries using cotangent Laplacian



Landmarks

- Extract Landmarks
  - Proposed method: manual selection
  - Implement a tool that allows this

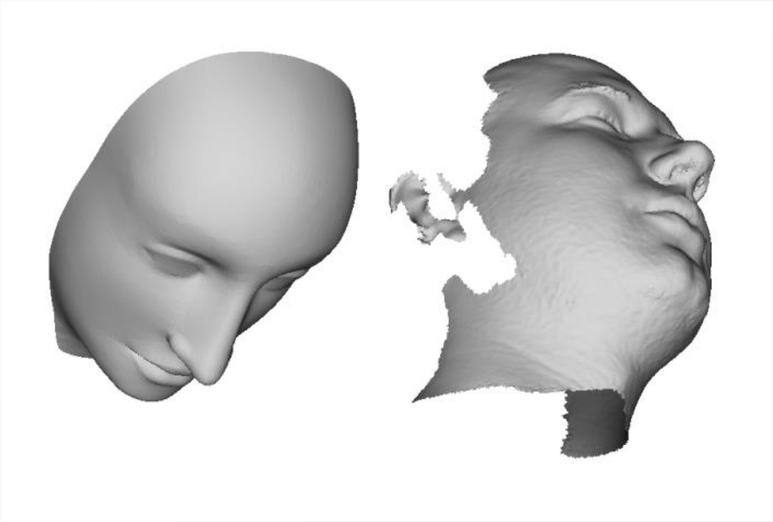




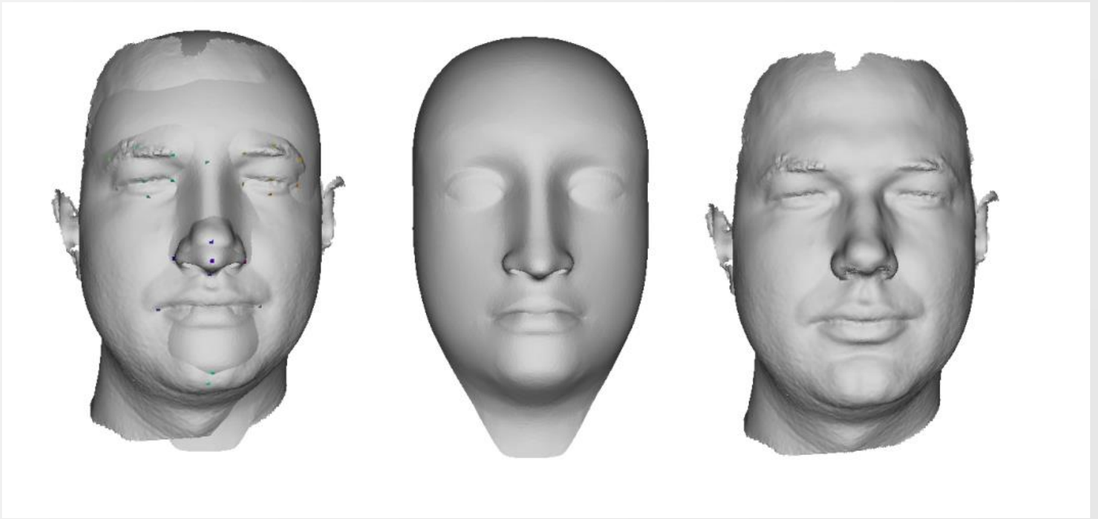
- Get a 3D position from a mouse click (see assignment 5), from libigl

Rigid Alignment

- Rescale template to scan
- Rigidly align scan to template



Template and target



Before and after warping

Notes for Rigid Alignment

2. a) Rigid alignment

- Scale template to scan

  - Don’t rescale scan, or scale all scans by same factor.

  - Scale template s.t. the average distance to mean landmark is the same for scan and template.

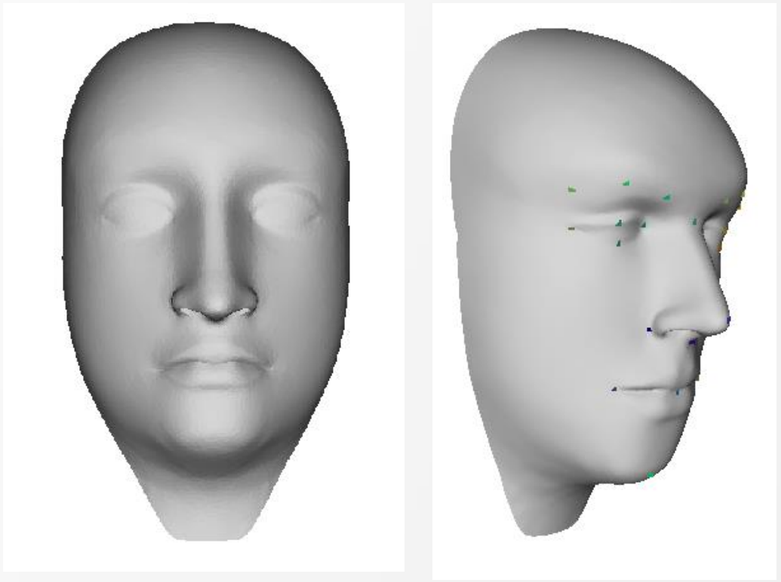
  - Before scaling the template, tranlate it such that the mean of its vertices is (0,0,0)

- Use the correspondences given by the landmarks to find a rigid alignment (e.g., rotation matrix computed via SVD).

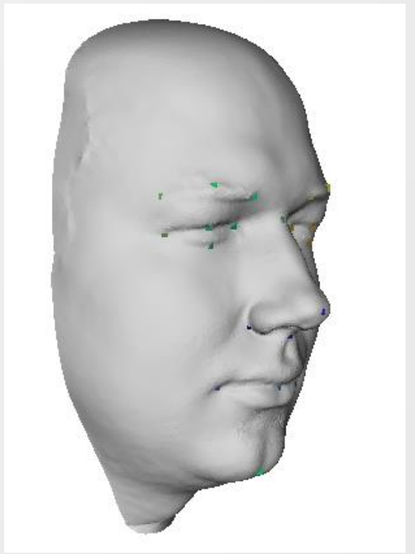
Warping (non-rigid alignment)

2 b). Warping

Goal: warp template to rigidly aligned scan, which provides the common triangulation



Non-rigid warp to match landmarks



Four warpping iterations

Note: find a good resolution (number of faces) of the template face model

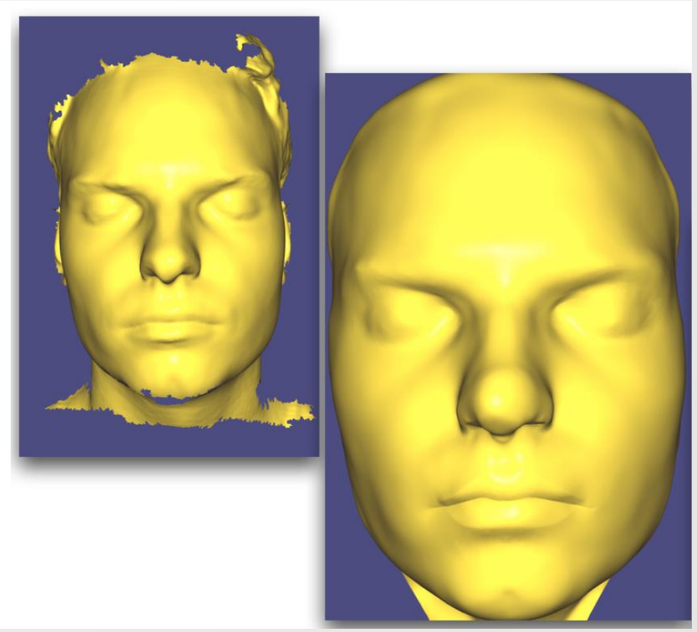
Warping

2.b) Warping: **suggest** method

Use Laplacian as a smoothness constraint

$$E_{warp} = ||Lx' - Lx||^2 + \lambda||Id|_{constr}x' - c||^2$$

X': unknown warped positions,
L: your favorite Laplacian (e.g. cotan weights on boundaries)
Id|const: selects the positions to be constrained
c: target positions

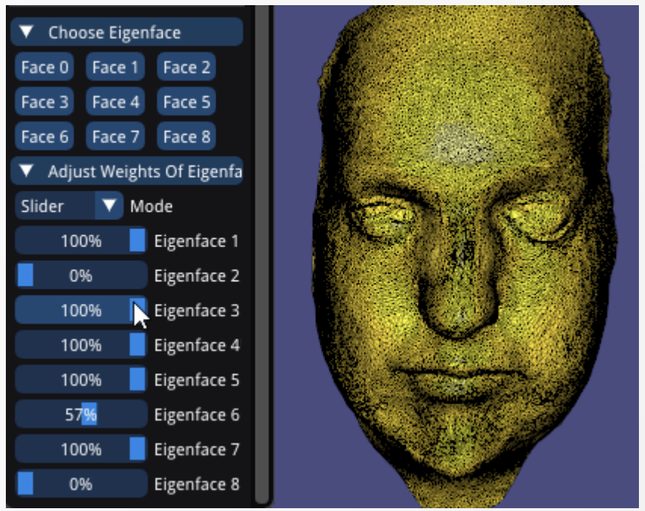


Note:

- Target and src should be rigidly aligned first!
- Keep the boundary fixed (above method works also without doing anything to the boundaries, but it is better for a common triangulation)
- Consider other constraints like those vertices close enough to target face

PCA face (unsupervised learning)

- PCA on face vectors to find our dominant variation / average mesh
- Morphing on eigen-space
  - for example, smile-to-neutral, personA-to-personB



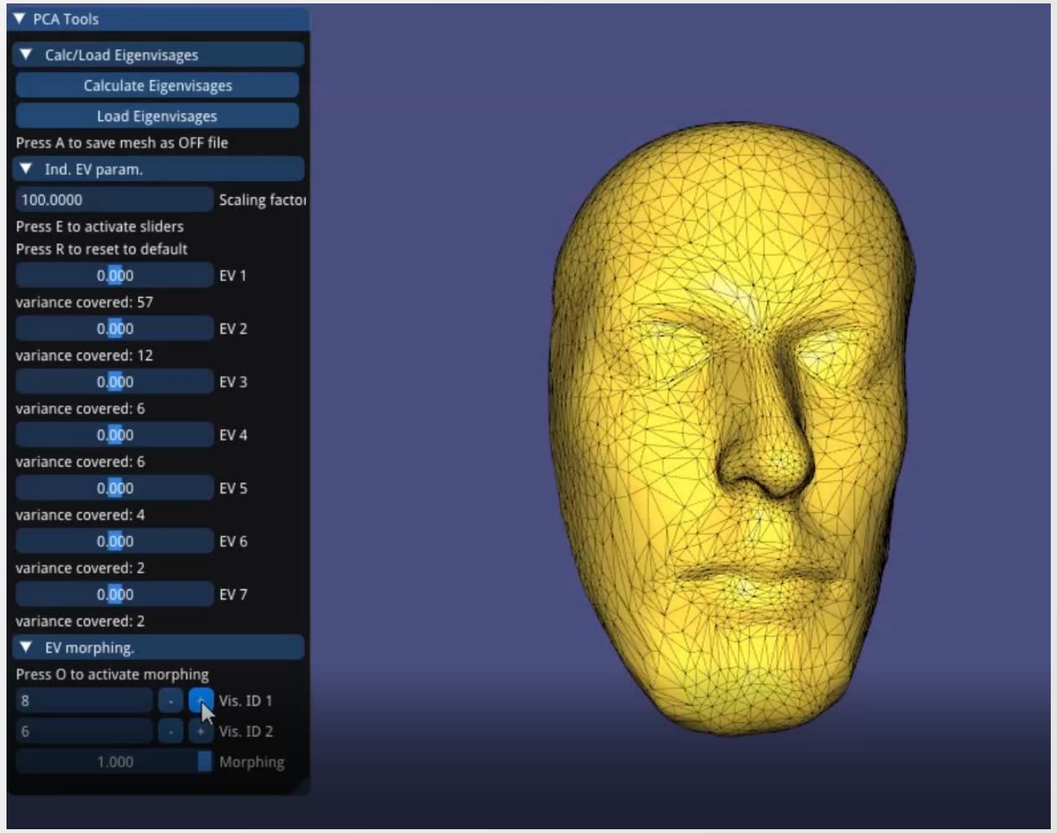
$$f_r = F_m + \sum_{i=0}^{n} e_i * w_i$$

PCA face morphing

- Some literature about PCA 3D face modeling:
  1) A Morphable Model For The Synthesis Of 3D Faces
  2) Singular Value Decomposition, Eigenfaces
  3) 3D Eigenfaces for Face Modeling.
  4) PCA and Face Recognition

$$w_i = (f_i - F_m)^T e_i$$

$$f_{morph} = F_m + \sum_{i=0}^{n} (w_i^{f_1} - m * (w_i^{f_1} - w_i^{f_2})) * e_i$$



Data to get started

- A template and some scanned faces.
  headtemplate.obj
- Landmarks of the scanned faces:
  File Format: vertexIndex<int> labelName<string>\n
- Rigidly aligned of one face:
  template_rigid_aligned.obj, peter_rigid_aligned.obj
  To get started with non-rigid warping and rigid alignment at once.
- Scanned data from last year
- Basel face model (if additional data needed)

Here is the link to the data: https://www.dropbox.com/sh/kvgxcbixbjsolt9/AABM1AHOr1AJnvz-qETJB0K0a?dl=0

Disclaimer: we do **not** hold copyrights of these data,
please use them for study in this class exclusively!

Input/Output Formats

- Output (Scanning):

  Colored obj files.

- Output (Landmark extraction):

  Vertex indices, Label

  As a txt format.

  E.g: label_name<oneword/int> vertex1<int> vertex2<int> vertex3<int> bari1<float> bari2<float> bari3<float>

- Output (common triangulation via template registration)

  Colored obj files, where vertices and faces enumerated the same way

- Result (Applications):

  Interactive GUI allowing to explore Morphing/ Dominant PCA, allow to save STLs or objs of the results.

Grading

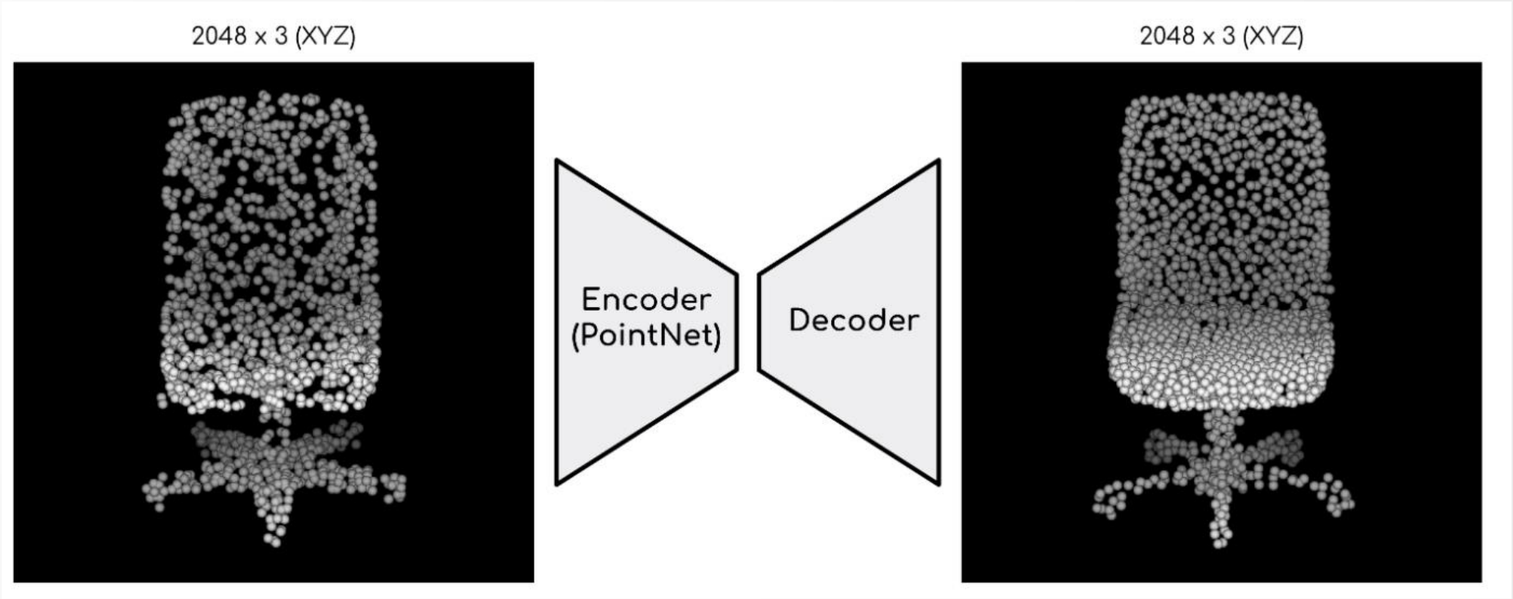
15% : Landmark selection
40% : Face alignment (rigid and non-rigid)
15% : The PCA of faces
10% : The UI (eigenface mixture weight tuning) and interpolation quality
20% : Project presentation and demo
Bonus (15%): Leaning-based face space

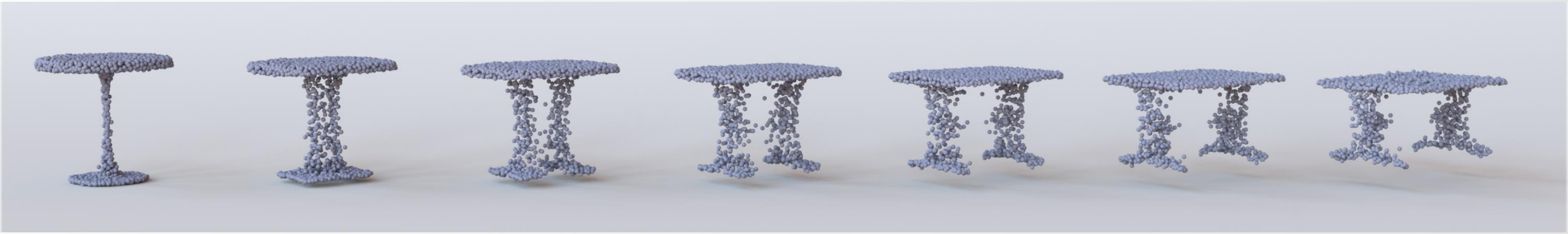
Applying techniques we learned in the class, no base code.

A package per group:
- Code (written with libigl) and data
- Slides, a short report of screenshots and **work division**

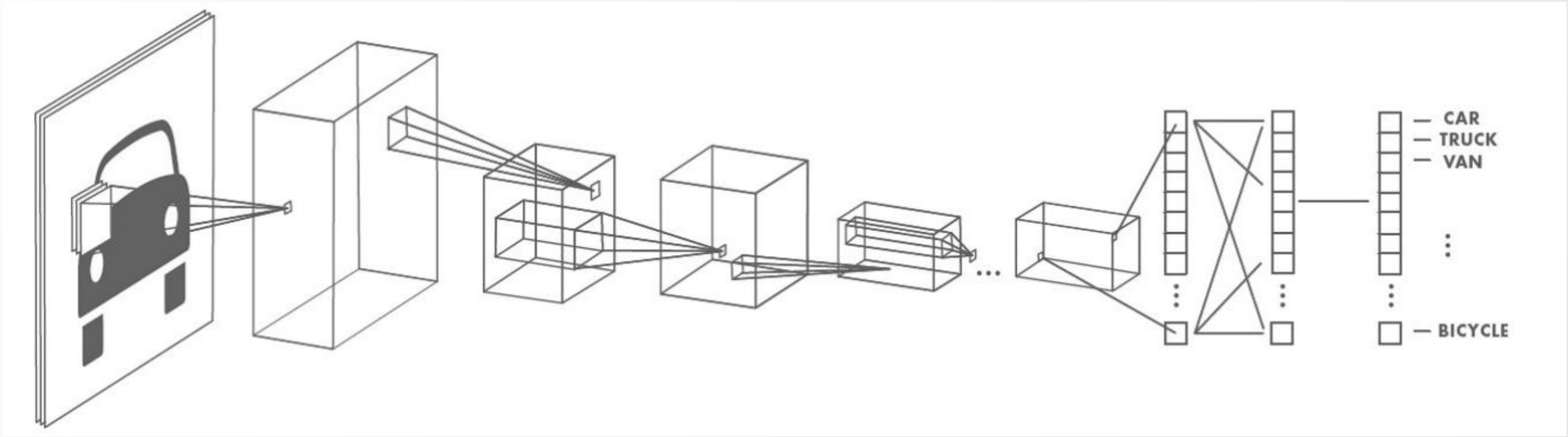
Submission deadline: 29.05.2020

Learning-based face space





Learning-based face space

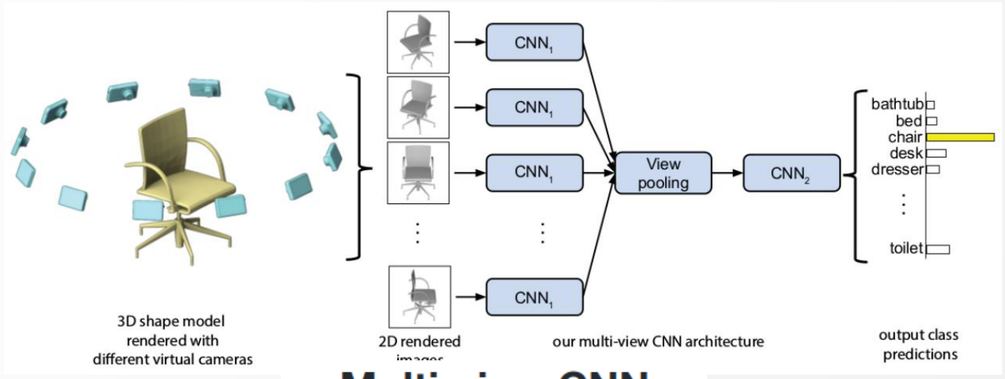


LeCun et al. 1989

Convolutional filters (Translation invariance+Self-similarity)

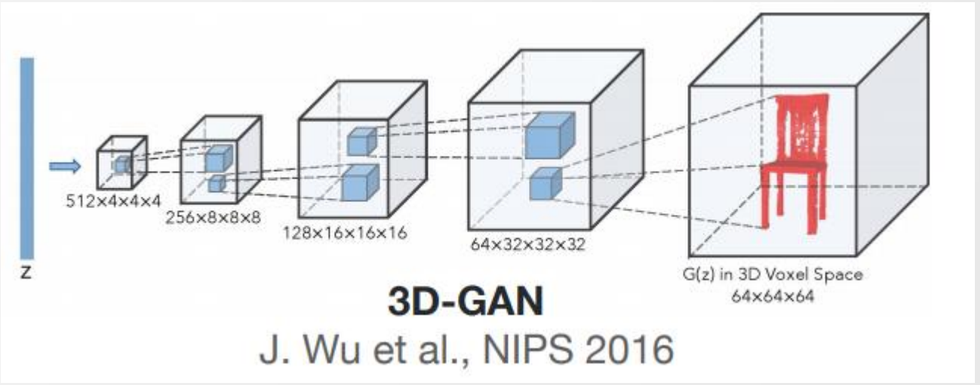
3D shape representation for learning

Image-based



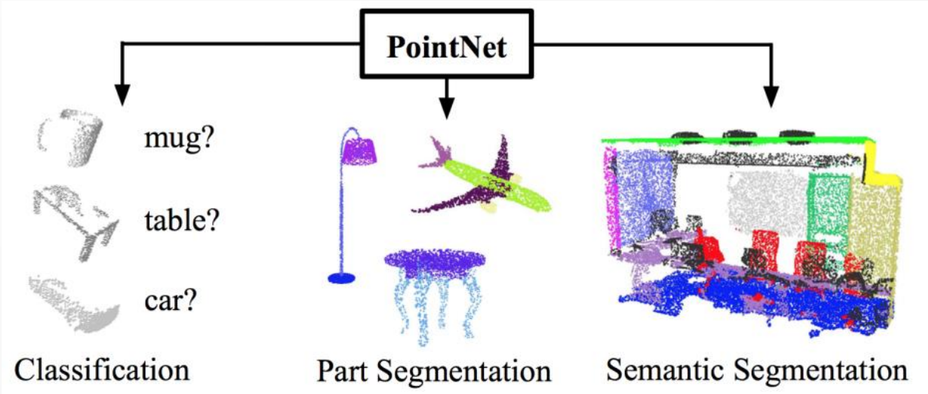
Multi-view CNN
Hang Su et al., ICCV 2015

Volumetric

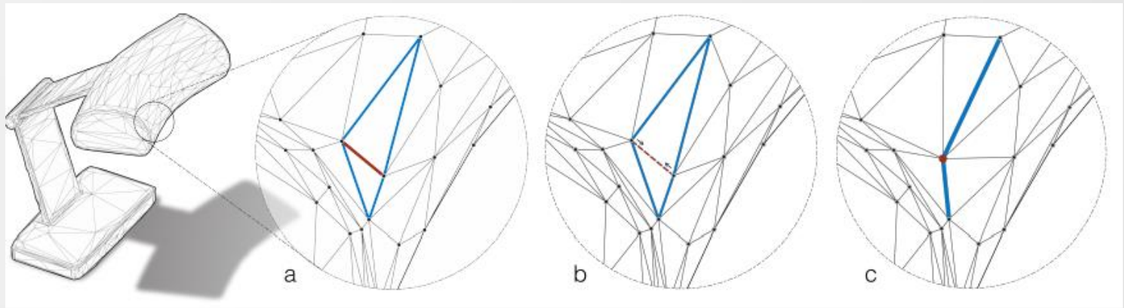


3D-GAN
J. Wu et al., NIPS 2016

Point-based

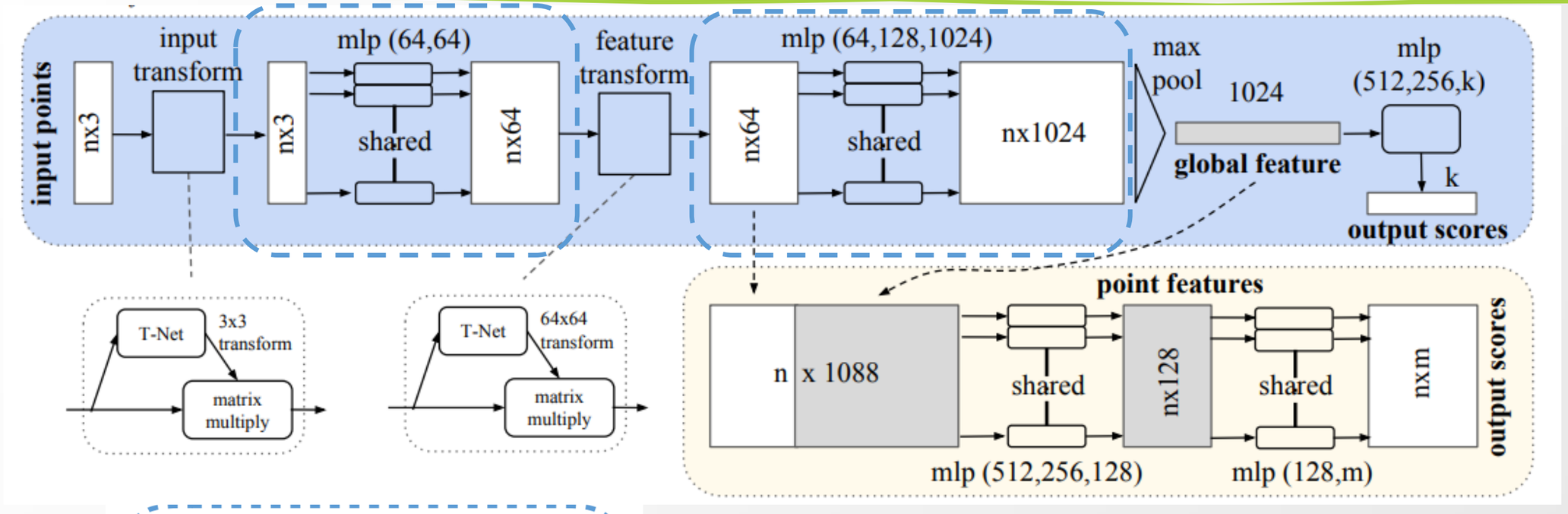


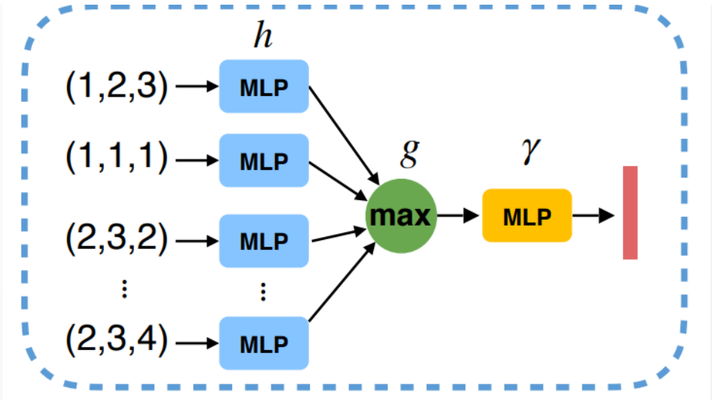
Mesh or Graph based



MeshCNN
Rana et al. SIGGRAPH 2019

PointNet

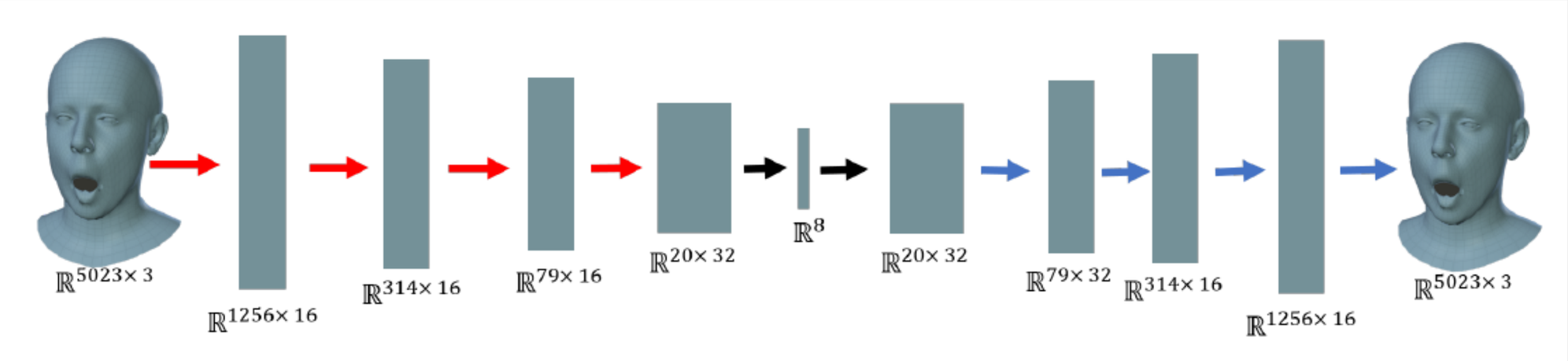

input points
nx3
input transform
nx3
mlp (64,64)
shared
nx64
feature transform
nx64
mlp (64,128,1024)
shared
nx1024
max pool
1024
global feature
mlp (512,256,k)
k
output scores
T-Net
3x3 transform
matrix multiply
T-Net
64x64 transform
matrix multiply
point features
n x 1088
shared
nx128
shared
nxm
output scores
mlp (512,256,128)
mlp (128,m)


h
(1,2,3)
MLP
(1,1,1)
MLP
(2,3,2)
MLP
(2,3,4)
MLP
g
max
γ
MLP

Learning-based face space

Example codes and papers:

- Learning representations and generative models for 3d point clouds
- PointNet implementation in pytorch geometric
- Generating 3D faces using Convolutional Mesh Autoencoders



Questions?

Thank you!