Computer Vision 2 - Assignment 3
2D-to-3D

Wednesday 8th May, 2019

Deadline: Wednesday 29-05-2019, 23:59:59 (Amsterdam time)

General guidelines

Students should work on the assignments in **groups of 3**. Students are supposed to work on this assignment for three weeks. Some minor additions and changes might be done during these three weeks. Students will be informed for these changes via Piazza and Blackboard.
Any questions regarding to the assignment content can be discussed on Piazza `https://piazza.com/class/jf0xj72ryz96k4` (Access code: 52042cov6y).
Students are expected to do this assignment in Python, however students are free to choose other tools. Please provide requirements.txt with all external dependencies listed and README.txt file with description about how to reproduce results.
Source code and report must be handed in together in a zip file (ID1_lastname1_ID2_lastname2_ID3_lastname3.zip) before the deadline in Canvas. For full credit, make sure your report follows these guidelines:

- Include an introduction and a conclusion to your report.
- The maximum number of pages is 10 (single-column, including tables and figures). Please express your thoughts concisely. The number of words does not necessarily correlate with how well you understand the concepts.
- Answer all given questions. Briefly describe what you implemented.
- Try to understand the problem as much as you can. When answering a question, give evidences (qualitative and/or quantitative results, references to papers, etc.) to support your arguments.
- Analyze your results and discuss them, e.g. why algorithm A works better than algorithm B in a certain problem.
- Tables and figures must be accompanied by a brief description. Do not forget to add a number, a title, and if applicable name and unit of variables in a table, name and unit of axes and legends in a figure.

Late submissions are not allowed. Assignments that are submitted after the strict deadline will not be graded. In case of submission conflicts, TAs' system clock is taken as reference. We strongly recommend submitting well in advance, to avoid last minute system failure issues.
Plagiarism note: Keep in mind that plagiarism (submitted materials which are not your work) is a serious crime and any misconduct shall be punished with the university regulations.
Award: The team with highest assignment grades will be 3D scanned and printed. The winner will be announced at the end of this course.

1 Introduction

In this assignment we will make a mini version of an avatar manipulation system similar to Thies et al. [2016] or Nagano et al. [2018]. This assignment will require some knowledge in Python (numpy, Tensorflow). If you are familiar with other tensor manipulation frameworks with gradient-based optimization feel free to use them. If you are not familiar with any, we recommend you to go through the Tensorflow Core tutorial beforehand (`https://www.tensorflow.org/tutorials/eager/custom_training_walkthrough` 2-3 hours).

2 Morphable Model (5 points)

So far we have been working on building a 3D model using depth information (Assignment 1) and feature points from 2D texture images (Assignment 2). Given a prior knowledge about an object, 3D model can be reconstructed from a monocular image. One of the effective approach to capture prior knowledge about an object is to build a statistical model using PCA Cootes [1992]. In the case of face geometry $\mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\delta}) \in \mathbb{R}^{N\times 3}$, we can assume it to be a multilinear PCA:

$$\mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\delta}) = \boldsymbol{\mu}_{id} + \mathbf{E}_{id}[\boldsymbol{\alpha} \cdot \boldsymbol{\sigma}_{id}] + \boldsymbol{\mu}_{exp} + \mathbf{E}_{exp}[\boldsymbol{\delta} \cdot \boldsymbol{\sigma}_{exp}] \tag{1}$$

where $\boldsymbol{\mu}_{id} \in \mathbb{R}^{N\times 3}, \boldsymbol{\mu}_{exp} \in \mathbb{R}^{N\times 3}$ are mean neutral geometry (facial identity) and mean facial expression; $\mathbf{E}_{id} \in \mathbb{R}^{N\times 3\times 30}$ and $\mathbf{E}_{exp} \in \mathbb{R}^{N\times 3\times 20}$ are principal components for neutral geometry and facial expression with their standard deviations $\boldsymbol{\sigma}_{id} \in \mathbb{R}^{30}$, $\boldsymbol{\sigma}_{exp} \in \mathbb{R}^{20}$; $\boldsymbol{\alpha} \in \mathbb{R}^{30}$ and $\boldsymbol{\delta} \in \mathbb{R}^{20}$ are latent parameters we need to estimate. For this assignment we will use first 30 principal components for facial identity and 20 principal components for facial expression.

For this assignment you will need to download Basel Face Model 2017 (`https://faces.dmi.unibas.ch/bfm/bfm2017.html`, `model2017-1_face12_nomouth.h5`). Note that non-commercial license doesn't allow you to distribute the model, consequently each student should download from the website himself / herself. Example code for how to load the h5 model is available in `mesh_to_png.py`.

The PCA model for facial identity can be extracted via keys `shape/model/mean` $\in \mathbb{R}^{3N}$, `shape/model/pcaBasis` $\in \mathbb{R}^{3N\times 199}$, `shape/model/pcaVariance` $\in \mathbb{R}^{199}$ from `model2017-1_face12_nomouth.h5`; PCA model for expression - via keys `expression/model/mean` $\in \mathbb{R}^{3N}$, `expression/model/pcaBasis` $\in \mathbb{R}^{3N\times 100}$, `expression/model/pcaVariance` $\in \mathbb{R}^{100}$, triangle topology - via keys `shape/representer/cells` $\in \mathbb{R}^{3\times K}$ where K is an amount of triangles.

- Extract 30 PC for facial identity and 20 PC for expression and generate a point cloud using Eq. 1. Uniformly sample $\boldsymbol{\alpha} \sim U(-1, 1)$, $\boldsymbol{\delta} \sim U(-1, 1)$. In the report show multiple point cloud samples. Provided `mesh_to_png` function and `Mesh` data structure can be used for visualization. For vertex colour you can use mean face colour available from `color/model/mean`. (**5 points**)

3 Pinhole camera model (5 points)

Given a 3D model from eq. 1 we assume face vertices $\{x, y, z\} \in \mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\delta})$ is rigidly transformed using transformation matrix $\boldsymbol{T} \in \mathbb{R}^{4\times 4}$ in the scene and projected using $\boldsymbol{\Pi} \in \mathbb{R}^{\mathbf{4\times 4}}$ into a camera plane.

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{d} \end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{V}\end{bmatrix} \times \begin{bmatrix}\mathbf{P}\end{bmatrix}}_{\mathbf{\Pi}} \times \underbrace{\begin{bmatrix} \mathbf{R}(\boldsymbol{\omega}) & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}}_{\mathbf{T}} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{2}$$

where $\boldsymbol{V}$ is a viewport matrix (`http://glasnost.itcarlow.ie/~powerk/GeneralGraphicsNotes/projection/viewport_transformation.html`) and $\boldsymbol{P}$ is a perspective projection matrix (`https://bit.ly/300gYmf`). Object transformation is modeled using rotation matrix $\mathbf{R}(\boldsymbol{\omega}) \in \mathbb{R}^{3\times 3}$ and translation $\mathbf{t} \in \mathbb{R}^3$, where $\mathbf{R}(\boldsymbol{\omega})$ is a function over Euler angles $\boldsymbol{\omega} \in \mathbb{R}^3$ (`https://bit.ly/2PQ8glW`).

- Assuming object translation to be 0. Rotate an object 10^o and -10^o around Oy and visualize results. `mesh_to_png` can be used (**2.5 points**)
- Assuming object translation to be 0, and rotation around Oy to be 10^o visualize facial landmark points on the 2D image plane using Eq. 2 (Fig.1 is an example). Vertex indexes annotation are available in the provided file `Landmarks68_model2017-1_face12_nomouth.anl`. (**2.5 points**)

4 Latent parameters estimation (9 points)

So far we have built a pipeline which can infer facial landmarks given facial geometry latent parameters $\boldsymbol{\alpha}$, $\boldsymbol{\delta}$ and object transformation $\boldsymbol{\omega}$, $\boldsymbol{t}$. In this section we will estimate those parameters for a specific 2D image with

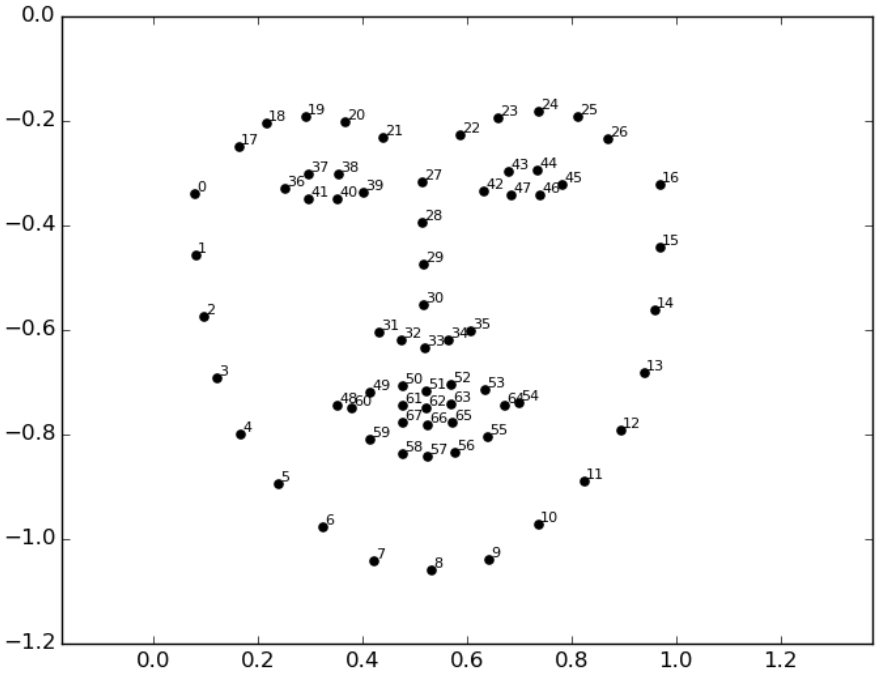

Figure 1: DLib format landmark points. Source: StackOverflow

a human face using Energy minimization. Given 68 ground truth facial landmarks the following energy can be optimized:

$$\mathcal{L}_{fit} = \mathcal{L}_{lan} + \mathcal{L}_{reg} \tag{3}$$

$$\mathcal{L}_{lan} = \sum_{j=1}^{68} \left\| \mathbf{p}_{kib_j} - \mathbf{l}_j \right\|_2^2 \tag{4}$$

where $\mathbf{p}_{k_j}$ is a 2D projection of a landmark point k_j from `Landmarks68_model2017-1_face12_nomouth.anl` and $\mathbf{l}_j$ is its ground truth 2D coordinate. We regularize the model using Tikhonov regularization to enforce the model to predict faces closer to the mean:

$$\mathcal{L}_{reg} = \lambda_{alpha} \sum_{i=1}^{30} \alpha_i^2 + \lambda_{delta} \sum_{i=1}^{20} \delta_i^2 \tag{5}$$

- Take a single picture of yourself or pick random one from the web. Extract ground truth landmarks using Dlib (`http://dlib.net/face_landmark_detection.py.html`). Keep face closer to the frontal and neutral for now. Visualize results. (**1 point**).
- Assuming $\boldsymbol{\alpha}$, $\boldsymbol{\delta}$, $\boldsymbol{\omega}$, $\boldsymbol{t}$ to be latent parameters of your model optimize an Energy described above using Adam optimizer until convergence. Visualize predicted landmarks overlayed on ground truth (**5 points**). Hint: in Tensorflow $\boldsymbol{\alpha}$, $\boldsymbol{\delta}$, $\boldsymbol{\omega}$, $\boldsymbol{t}$ can be declared as `tf.Variable` and ground truth landmarks can be set using `tf.placeholder`, landmark prediction can be extracted using `tf.gather`.
- **Hint**: initializing transformation parameters $\boldsymbol{\omega}$ and $\boldsymbol{t}$ closer to the solution may help with convergence. For example translation over z dimension can be set to be -400 in the case of projection matrix with principal point $\{\frac{W}{2}, \frac{H}{2}\}$ and fovy = 0.5.
- Select hyper parameters such that $\boldsymbol{\alpha}$ and $\boldsymbol{\delta}$ to be obtained in a proper range. Report findings. (**3 points**).

5 Texturing (5 points)

Given latent parameters $\boldsymbol{\alpha}$, $\boldsymbol{\delta}$, $\boldsymbol{\omega}$, $\boldsymbol{t}$ u,v projection coordinates for each point of the 3D model (Eq. 1) can be obtained by bilinear interpolation (`https://en.wikipedia.org/wiki/Bilinear_interpolation`) of neighborhood pixels. We will refer the neutral textured geometry as an avatar.

- Compute 2D projection for each point given estimated latent parameters. For each point extract corresponded RGB value from a 2D image as a vertex colour. Visualize obtained results using `mesh_to_png`. (**5 points**).

6 Energy optimization using multiple frames (5 points)

So far we have built a 2D-to-3D reconstruction algorithm which accepts 1 single monocular images and predicts latent parameters $\boldsymbol{\alpha}$, $\boldsymbol{\delta}$, $\boldsymbol{\omega}$, $\boldsymbol{t}$ for a 3D model. Assuming we have multiple (M) images of the single person we can extend our model to constraint neutral face geometry using multiple frames with corresponding object pose $\boldsymbol{\omega}_i$, $\boldsymbol{t}_i$, i = 1..M.

$$\forall \text{i} = 1..\text{M}\ \mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\delta}) = \boldsymbol{\mu}_{id} + \mathbf{E}_{id}[\boldsymbol{\alpha} \cdot \boldsymbol{\sigma}_{id}] + \boldsymbol{\mu}_{exp} + \mathbf{E}_{exp}[\boldsymbol{\delta}_i \cdot \boldsymbol{\sigma}_{exp}] \tag{6}$$

- Extend the current Energy minimization pipeline (Sections 3, 4) for multiple frames optimization. Report transformed, textured mesh for each frame. (**5 points**)

If you get stuck in that section you can continue with expression manipulation assuming M = 1 frames for identity parameters optimization.

7 Expression manipulation (5 points)

Assuming that you have a video sequence with face (source video) of J frames. Its expression can be easily transferred to a target avatar as following:

- Estimate source identity parameters $\boldsymbol{\alpha}$ using M frames;
- Fix $\boldsymbol{\alpha}$ for the full video sequence;
- $\forall$j = 1..J estimate $\boldsymbol{\delta}_j, \boldsymbol{\omega}_j, \boldsymbol{t}_j$ using energy minimization. Initializing parameters by parameters from previous frame may prevent the model to converge into local optima.
- Apply each $\boldsymbol{\delta}_j$ on the target avatar

- Capture 1 second video sequence with face changing expression from a neutral face. Transfer expression into a previously obtained avatar. Report transferred textured results. The whole frame sequence can be zipped separately for submission. (**5 points**).

8 Discussion (6 points)

In the first three weeks of the Computer Vision 2 course you have implemented a depth based 3D reconstruction method (i.e. ICP). The following 2 weeks you have implemented a texture based 3D reconstruction method (i.e. SFM).

- Please briefly explain what the **advantages** and **disadvantages** of these methods are. (**2 points**)
- Do you think these two methods could be used together to get better 3D reconstruction? (**2 points**)
- Now consider your recently introduced 2D-to-3D reconstruction method, what will be its advantages and disadvantages? How can it be used together with other methods to obtain better 3D reconstruction? (**2 points**)

9 Misc

Please describe briefly in an appendix to the report about contribution of each team member to this assignment. Self-evaluation will be taken into consideration in the case if contributions are significantly unequal.

10 Recommended additional reading

- Cootes (1992) An introduction to active shape models.
- Thies et al. (2016) Face2Face: Real-time Face Capture andReenactment of RGB Videos. Sections 2, 3, 4.
- Paysan et al. (2009) A 3D Face Model for Pose and Illumination Invariant FaceRecognition `https://gravis.dmi.unibas.ch/publications/2009/BFModel09.pdf`

Deadline: Wednesday 29-05-2019, 23:59:59 (Amsterdam time)

Interested in working on similar projects for Project AI or Master Thesis? Drop an email to `l.m.ngo@uva.nl` for more information!

References

T. F. Cootes. An introduction to active shape models . 1992.

K. Nagano, J. Seo, J. Xing, L. Wei, Z. Li, S. Saito, A. Agarwal, J. Fursund, and H. Li. pagan: Real-time avatars using dynamic textures. *ACM Trans. Graph.*, 37(6):258:1–258:12, Dec. 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275075. URL `http://doi.acm.org/10.1145/3272127.3275075`.

J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. Face2Face: Real-time Face Capture and Reenactment of RGB Videos. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2016.