POVa - Computer Vision (in English) - Project

Topic: Facial recognition from RGB-Depth images (identification)

Project Task Definition

Our task is to develop a facial recognition system which uses sensors (Time-Of-Flight or Structured Light Sensors) such as Azure Kinect. This will improve the accuracy of the face detection, alignment and identification.

For the face detection part, we will rely either on Multi-Task CNN or OpenCV's cascade classifier. The system will be based on a reference database of labeled faces so that it can train, we can start with EURECOM Kinect Face Dataset or https://github.com/luxiangju-PersonAI/iCartoonFace to work on unreal/cartoon faces, as most dataset require permissions, which, most of the time are not given to students

Preliminary plan of work

- **General Approach**

Our project will be divided into 3 different parts: face detection, face alignment and face identification, to do so, we will work with two datasets (Depth & Color frames). We aim to enhance face recognition in different contexts and orientations.

- **Methods**

Face Detection : MTCNN or OpenCV to locate initial features.
Face Alignment : Point Distribution Model (PDM)
Face Identification : Trained using EURECOM Kinect Face dataset

- **Software, Tools, Tutorials**

Software and libraries : Python for OpenCV and Face Recognition https://pypi.org/project/face-recognition/, TensorFlow or Pythorch for deep learning frameworks, Scikit-learn to evaluate metrics.
Tools : Azure Kinect SDK to act as a platform between the sensor and our computer. Anaconda for Python's libraries and TensorBoard to visualize in real-time and train our system.
Tutorials & References : TensorFlow tutorials for CNNs, PyImageSearch for OpenCV, and https://github.com/DCGM/mtcnn

Existing Components & Work

- **Existing Components**

MTCNN, OpenCV for face recognition, Azure Kinect SDK to acquire data

- **Work**

Customize existing solutions to adapt them for RGB-D integration, refining of the face obtained by using depth-based adjustments, and creating an user interface to test the solution.

- **Student A**

Collect multiple images with different lighting and angles with Azure Kinect SDK
Implement data preprocessing steps, with landmark detection and alignment with MTCNN/OpenCV then integrate PDM with depth data.

- **Student B**

Work on the color frames of the collected data
Develop and set up a neural network model using TensorFlow/PyTorch
Evaluate the model and the data by using metrics such as IoU/Scikit-learn