The task involved learning some basic image processing techniques in Python using the opencv, numpy, os, sys, pickle, and argparse packages. These techniques help in constructing datasets for deep learning algorithms, as annotated datasets are a crucial component in training such algorithms.
The following libraries were used:
OpenCV: An open-source computer vision and machine learning software library that is commonly used for image processing tasks such as object detection and face recognition.
Numpy: A library for scientific computing in Python, used for processing arrays and matrices in image processing tasks.
OS: A library in Python for interacting with the operating system, such as reading or writing to the file system.
Sys: A library in Python for interacting with the Python interpreter, such as accessing arguments passed to the script.
Pickle: A library in Python for serializing and deserializing Python objects, used for saving and loading the bounding box information for each face in an image. The annotated data consisted of the coordinates of the faces present in each image, which were obtained by selecting the coordinates with mouse clicks. The data was saved as a pickle file, which is a binary file format that can be used to serialize Python objects. The use of Pickle allowed the annotated data to be stored and retrieved easily, making it convenient for future use in training deep learning models. The file name was given the same basename as the corresponding image, with the part after the dot being ".txt". This provided a simple and efficient way to associate the annotated data with the corresponding image, allowing for easy access when needed.
Argparse: A library in Python for parsing command-line arguments, used for allowing the user to specify options for the script. This library allowed for greater flexibility and control over the input parameters, such as the input folder containing the images and the output folder to store the annotations. The parsed arguments were then used to perform the annotation process, where the bounding boxes were drawn around the faces in the images. The use of argparse made it easier to reuse the code for annotation on different sets of images and to easily modify the input parameters for different use cases. It also made the code more user-friendly by allowing the user to specify the parameters from the command line rather than hard-coding them in the code.

The task required us to create our own dataset of images with faces of our group members, annotate the images with rectangular bounding boxes, and blur the faces in the images. We also had to extend the same techniques to videos. Starting with the annotation subtask, we learned how to construct a dataset with faces of our group members and obtained the coordinates of the faces in all the input images by selecting the coordinates with mouse clicks. We also learned to save the coordinates for each face as a pickle file, which was a new concept for us. In the first subtask, we were required to construct a dataset of faces of group members, consisting of at least 2 images per person.
The images should be of various sizes starting from 24x24 and resolutions of type svga, xga, wxga, HD, and ending at no more than Full HD resolution. Additionally, we were required to create one image which did not contain identifying information and place it in the anonymizedData/images folder. The second step was to obtain the coordinates of faces in all the input images by selecting the coordinates with mouse clicks for each image.

We were provided with the function setMouseCallback() to perform this task. The obtained coordinates were then saved as a pickle file with the same basename and the part after the . should be txt (e.g. 01.txt). In the drawing boxes subtask, we read the annotation details provided to us and used them to draw rectangular boxes around the faces present in the image. This subtask allowed us to apply our knowledge of annotation and pickle files. In the blurring faces subtask, we learned how to hide a part of the image using Gaussian blur.

The task of blurring faces within rectangular regions helped us to understand how to apply filters to specific regions of an image. Finally, in the videos subtask, we extended our knowledge of image processing to videos. We prepared a video of 10 seconds containing our team members, loaded the video and annotated it with rectangular regions for faces, and then blurred the rectangular regions specified by the pickle file.

The task of edge and contour detection in an image is an important step in computer vision and image processing. The goal of this task was to use the Canny edge detection method to detect edges in an image and to use the findContours() API to find the contour of a person in an image. The Canny edge detection method was found to be very effective in detecting edges in an image. The process involved converting the image to grayscale and applying a Gaussian blur to the image to reduce noise.

The results obtained were very close to the desired output and in some cases, even better. The output was written to the results folder with the name “edges”. The next step involved using the findContours() API to find the contour of a person in an image. The task was to create a contour for the tall person in the provided image and for a team image with more than one member. The findContours() API proved to be very effective in finding the contour of a person in the image. The output was written to the results folder with the same image name.

In conclusion, edge and contour detection is an important step in computer vision and image processing. The Canny edge detection method and the findContours() API proved to be very effective in detecting edges and contours in an image. The results obtained were very close to the desired output and in some cases, even better.

This task has helped me gain a better understanding of the importance of edge and contour detection in computer vision and image processing. The use of the Canny edge detection algorithm and the findContours function from the OpenCV library demonstrate how computer vision algorithms can be used to identify and analyze important features in an image. Additionally, the annotation of images with rectangular bounding boxes to identify faces highlights the importance of manual labeling in the construction of datasets for deep learning techniques. The task also touches on the importance of collecting large amounts of diverse data to train machine learning models effectively.

Overall, this task was a great learning experience for us as it allowed us to apply our knowledge of Python and image processing in a practical setting. We learned new concepts like pickle files and how to apply filters to specific regions of an image. The task was challenging, but it helped us to improve our skills and confidence in image processing.