**Fundamentals of Computer Vision – Project 1**

**Augmentation system for CV ML algorithms**

1. **Requirements**

Using Python and OpenCV write a program that:

* + Allows user to select a directory on local disk. (e.g. using tkinter library)
  + Read all .jpg images from this directory and, for each of them, apply a set of predefined augmentation algorithms with a set of predefined parameters. (e.g. Rotation with +15 degree).
    - The augmentation algorithms and corresponding parameters to be applied will be loaded when the program starts from a configuration file (plain text, xml etc.)
    - The results of augmentation process will be saved on a new directory (output dir), having the same name with the original one plus the "\_aug" suffix.
* Each augmented image will be saved in the output dir having the name of augmentation algorithm as suffix followed by an incremental number starting with "\_1".

1. **Libraries used**

* **Tkinter** – for selecting the folder from where we take the images
* **OpenCV** – for processing images and applying different augmentation techniques available in the library
* **Os** – for loading the images that need to be augmented
* **Numpy** – for easier manipulation of the arrays in python

1. **Configuration file structure**

* The configuration file is a JSON file that has the schema:

{

"type": "array",

"items": {

"type": "array",

"items": {

"type": "object",

"properties": {

"type": { "type": "string" },

"number1": { "type": "string" },

"number2": { "type": "string" },

"color": { "type": "string" }

}

}

}

}

* The file contains an array of arrays, each of the arrays containing objects that represent the augmentations
* Each augmentation has a type – the name, and 3 other variables, each representing different parameters for the specific augmentation (ex. for Rotate, the number1 parameter represents the degree of rotation)

1. **Augmentation algorithms**

The app allows the user to perform 6 different augmentation algorithms:

* Blur
  + - Function: **blur(image)**
    - Description: Applies a Gaussian blur to the input image.
* Sharpen
  + Function: **sharpen(image)**
  + Description: Applies a sharpening filter to the input image, enhancing edge details.
* Tint
* Function: **tint(image, number, color)**
* Description: Applies a color tint to the input image, adjusting the intensity of a specific color channel.
* Flip
  + Function: **flip(image, number)**
  + Description: Flips the input image horizontally, vertically, or both, altering its orientation.
* Rotate
* Function: **rotate(image, angle)**
* Description: Rotates the input image by a specified angle.
* Translate
* Function: **translate(image, alpha\_x, alpha\_y)**
* Description: Translates (shifts) the input image by specified values in the x and y directions.