**1- Source codes in *"LinearAlgebra in image processing.pdf"***

**Edge detection**

|  |
| --- |
| # -\*- coding: utf-8 -\*-  """  @author: ADMIN  """  **import** numpy **as** np  **import** cv2  # read image and convert to grayscale  img1 **=** cv2**.**imread**(**'anh.png'**,** 0**)**  # compute sobel-x  sobelx **=** cv2**.**Sobel**(**img1**,** cv2**.**CV\_64F**,** 1**,**0**)**  # compute sobely  sobely **=** cv2**.**Sobel**(**img1**,** cv2**.**CV\_64F**,** 0**,**1**)**  # Save results  cv2**.**imwrite**(**'vn\_edge\_x.png'**,** sobelx**)**  cv2**.**imwrite**(**'vn\_edge\_y.png'**,** sobely**)**  cv2**.**imwrite**(**'vn\_grayscale.png'**,** img1**)** |
| **vn\_edge\_x**  **Anh** |
| **vn\_grayscale**  **vn\_edge\_y** |

**Interpolation: image upsampling.**

|  |
| --- |
| # -\*- coding: utf-8 -\*-  """  @author: ADMIN  """  **import** numpy **as** np  **import** cv2  # read a color image  img**=** cv2**.**imread**(**'anh.png'**)**  # get meta-data of the image  height**,** width**,** channels **=** img**.**shape  # new dimension  new\_dim **=** **(**width**\***4**,** height**\***4**)**  # upsampling the image  resize\_nearest **=** cv2**.**resize**(**img**,** new\_dim**,** interpolation **=** cv2**.**INTER\_NEAREST**)**  resize\_bilinear **=** cv2**.**resize**(**img**,** new\_dim**,** interpolation **=** cv2**.**INTER\_LINEAR**)**  # save result  cv2**.**imwrite**(**'resize\_nearest.png'**,** resize\_nearest**)**  cv2**.**imwrite**(**'resize\_bilinear.png'**,** resize\_bilinear**)** |
| # *Result after running:* |

**Add Vector**

|  |
| --- |
| **import** math  # Phép cộng hai vector  **def** add\_vectors**(**vector1**,** vector2**):**  **return** **[**v1**+**v2 **for** v1**,** v2 **in** **zip(**vector1**,** vector2**)]**  vector1 **=** **[**1**,** 4**,** 3**]**  vector2 **=** **[**4**,** 5**,** 6**]**  output **=** add\_vectors**(**vector1**,** vector2**)**  **print(**output**)** |
|  |

**Matrix Multiplication**

|  |
| --- |
| **def** matrix\_multiplication**(**matrix1**,** matrix2**):**  matrix1\_nrows **=** **len(**matrix1**)**  matrix1\_ncols **=** **len(**matrix1**[**0**])**  matrix2\_nrows **=** **len(**matrix2**)**  matrix2\_ncols **=** **len(**matrix2**[**0**])**  # tạo matrix kết quả  result **=** **[[**0**]\***matrix2\_ncols **for** i **in** **range(**matrix1\_nrows**)]**    **for** i **in** **range(**matrix1\_nrows**):**  **for** j **in** **range(**matrix2\_ncols**):**  **for** k **in** **range(**matrix2\_nrows**):**  result**[**i**][**j**]** **+=** matrix1**[**i**][**k**]** **\*** matrix2**[**k**][**j**]**  **return** result    #test case  # 3x3 matrix  matrix1 **=** **[[**1**,** 2**,** 3**],**  **[**4**,** 5**,** 6**],**  **[**7**,** 8**,** 9**]]**  #matrix 3x4  matrix2 **=** **[[**1**,** 1**,** 2**,** 1**],**  **[**1**,** 2**,** 1**,** 1**],**  **[**1**,** 1**,** 1**,** 2**]]**  result **=** matrix\_multiplication**(**matrix1**,** matrix2**)**  **print(**result**[**0**])**  **print(**result**[**1**])**  **print(**result**[**2**])** |
| # *Result after running:* |

**Background Subtraction**

|  |
| --- |
| # *Skip due to lack of information:* |
| # *Result after running:*  *# Try to test also in JupyterLab* |

**Dot Product**

|  |
| --- |
| **def** dot\_product**(**vector1**,** vector2**):**  **return** **sum([**v1**\***v2 **for** v1**,** v2 **in** **zip(**vector1**,** vector2**)])**  #Test case  vector1 **=** **[**1**,** 2**,** 3**]**  vector2 **=** **[**2**,** 3**,** 4**]**  output **=** dot\_product**(**vector1**,** vector2**)**  **print(**output**)** |
| # *Result after running:* |

**Hadamard Product**

|  |
| --- |
| **def** hadamard\_product**(**vector1**,** vector2**):**  **return** **[**v1**\***v2 **for** v1**,** v2 **in** **zip(**vector1**,** vector2**)]**  #test case  vector1 **=** **[**1**,** 2**,** 3**]**  vector2 **=** **[**2**,** 3**,** 4**]**  output **=** hadamard\_product**(**vector1**,** vector2**)**  **print(**output**)** |
| # *Result after running:* |

**Cosine Similarity**

|  |
| --- |
| **import** math  **def** cosine\_similarity**(**vector1**,** vector2**):**  sumxy **=** **sum([**v1**\***v2 **for** v1**,** v2 **in** **zip(**vector1**,** vector2**)])**  sumxx **=** **sum([**v1**\***v2 **for** v1**,** v2 **in** **zip(**vector1**,** vector1**)])**  sumyy **=** **sum([**v1**\***v2 **for** v1**,** v2 **in** **zip(**vector2**,** vector2**)])**    **return** sumxy**/**math**.**sqrt**(**sumxx**\***sumyy**)**  #Test case  vector1 **=** **[**1**,** 2**,** 3**]**  vector2 **=** **[**2**,** 3**,** 4**]**  output **=** cosine\_similarity**(**vector1**,** vector2**)**  **print(**output**)** |
| # *Result after running:* |

**2- Library installation and study**

|  |  |  |  |
| --- | --- | --- | --- |
| **Library name** | **Command to install**  **&**  **Command for version check** | **Library roles** | **Links to references** |
| Numpy | *Pip install numpy*  *Pip list*  *python -m pip show numpy* | NumPy is the fundamental package for scientific computing in Python.  It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. | <https://cs231n.github.io/python-numpy-tutorial/>  <https://www.w3schools.com/python/numpy/default.asp>  <https://cs231n.github.io/python-numpy-tutorial/> |
| Pandas | pip install pandas | Pandas is a Python library used for working with data sets.  It has functions for analyzing, cleaning, exploring, and manipulating data.  The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008 | <https://www.w3schools.com/python/pandas/pandas_intro.asp> |
| Matplotlib | pip install matplotlib | Matplotlib is a low level graph plotting library in python that serves as a visualization utility.  [**Matplotlib**](https://www.geeksforgeeks.org/python-introduction-matplotlib/) is a library in Python and it is numerical – mathematical extension for NumPy library. [Pyplot](https://www.geeksforgeeks.org/pyplot-in-matplotlib/) is a state-based interface to a **Matplotlib** module which provides a MATLAB-like interface. There are various plots which can be used in Pyplot are Line Plot, Contour, Histogram, Scatter, 3D Plot, etc. | <https://www.w3schools.com/python/matplotlib_intro.asp>  [geeksforgeeks](https://www.geeksforgeeks.org/matplotlib-pyplot-plot-function-in-python/#:~:text=Matplotlib%20is%20a%20library%20in,Scatter%2C%203D%20Plot%2C%20etc.) |
| Keras | Pip install tensorflow (included keras) | Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications. | <https://www.tutorialspoint.com/keras/keras_introduction.htm> |
| Torch | pip install torch | PyTorch is an optimized tensor library primarily used for Deep Learning applications using GPUs and CPUs. It is an open-source machine learning library for Python, mainly developed by the Facebook AI Research team. It is one of the widely used Machine learning libraries, others being TensorFlow and Keras. Here is the Google Search Trends which shows that the popularity of the PyTorch library is relatively higher compared to TensorFlow and Keras. | <https://www.analyticsvidhya.com/blog/2021/04/a-gentle-introduction-to-pytorch-library/> |
| torchvision | pip install torchvision | Torchvision is **a library for Computer Vision that goes hand in hand with PyTorch**. It has utilities for efficient Image and Video transformations, some commonly used pre-trained models, and some datasets | <https://medium.com/swlh/understanding-torchvision-functionalities-for-pytorch-391273299dc9> |
| opencv-python #for cv2 | pip install opencv-python | OpenCV is one of the most popular computer vision libraries. If you want to start your journey in the field of computer vision, then a thorough understanding of the concepts of OpenCV is of paramount importance. | <https://www.geeksforgeeks.org/introduction-to-opencv/> |
| pydub | Pip install pydub | Audio files are a widespread means of transferring information. So let’s see how to work with audio files using Python. Python provides a module called **pydub** to work with audio files. **pydub** is a Python library to work with **only .wav** files. By using this library we can play, split, merge, edit our**.**wav audio files. | <https://www.geeksforgeeks.org/working-with-wav-files-in-python-using-pydub/> |
| tensorflow | Pip install tensorflow | TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications. | <https://www.tensorflow.org/> |
|  |  |  |  |