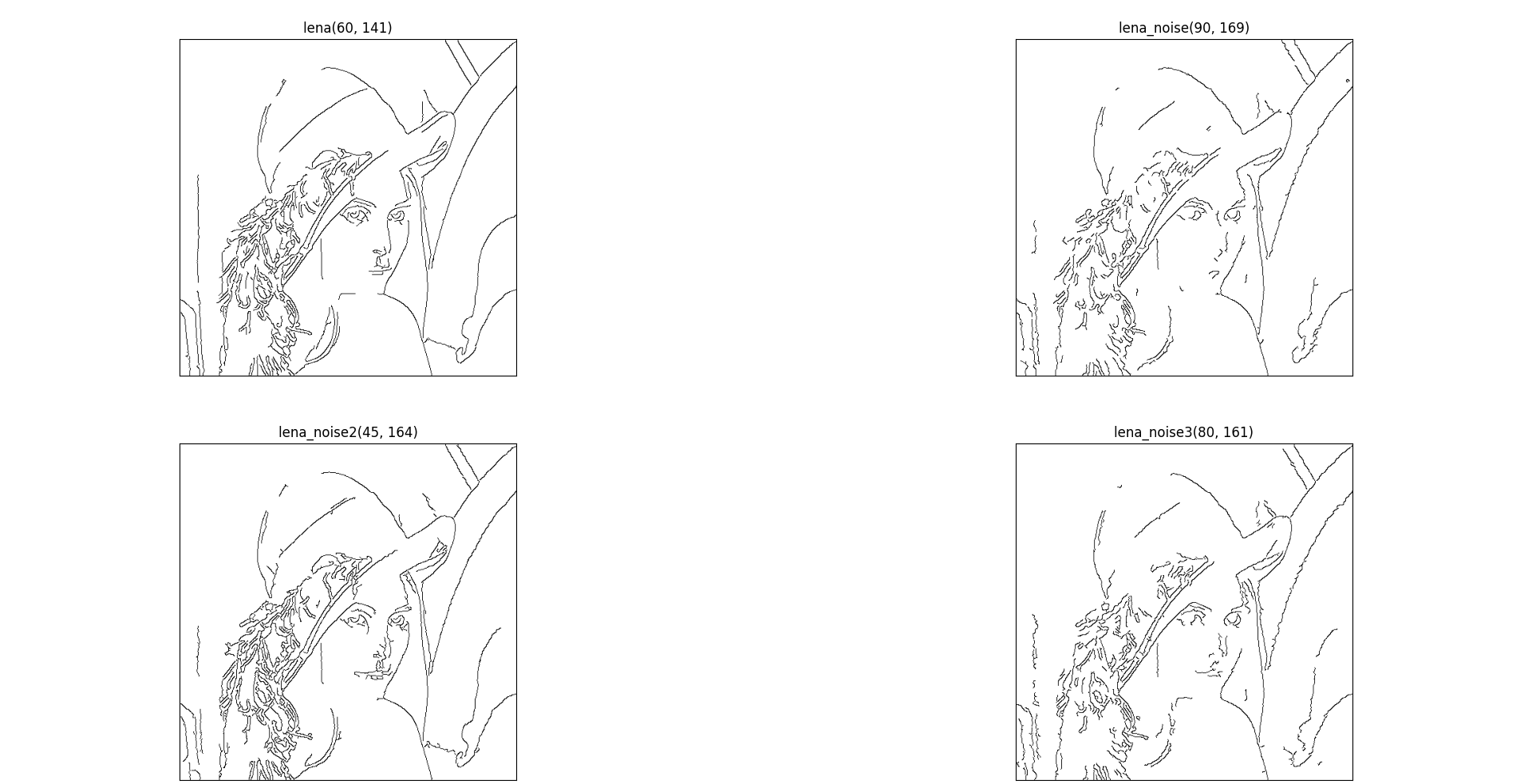
**Computer vision – HW7**

# Source code

|  |
| --- |
| # WeiWen Wu 2023-11-01 15:14:07  **import** cv2 **as** cv  **from** matplotlib **import** pyplot **as** plt  **from** numpy **import** ndarray # type  **def** bgr2gray**(**img\_file**:str)** **->** ndarray**:**  """Convert bgr to gray and remove noise points."""  img **=** cv**.**imread**(**img\_file**)** # Read image.  img **=** cv**.**GaussianBlur**(**img**,** **(**3**,** 3**),**0**)** # GaussianBlur  img **=** cv**.**medianBlur**(**img**,** 3**)** # Median Filtering  img **=** cv**.**GaussianBlur**(**img**,** **(**3**,** 3**),**0**)** # GaussianBlur  img **=** cv**.**medianBlur**(**img**,** 3**)** # Median Filtering  **return** cv**.**cvtColor**(**img**,** cv**.**COLOR\_BGR2GRAY**)** # BGR to gray.  **class** **plot\_from\_matplotlib:**  """  Make pictures from matplotlib.  # Example  ```  plot = plot\_from\_matplotlib("title")  plot("Original",img)  plot\_from\_matplotlib.show()  ```  """  n**:int=**1  **def** \_\_init\_\_**(**self**,**name**:str)** **->** **None:**  fig **=** plt**.**figure**()**  fig**.**canvas**.**manager**.**window**.**setWindowTitle**(**name**)**    **def** \_\_call\_\_**(**self**,**name**:str,**dst**,**cmap**:str=**'gray'**,**save**:bool=**1**):** # Make pictures.  n **=** self**.**n  path **=** \_\_file\_\_**.**split**(**"\\"**)[-**2**]**  plt**.**subplot**(**2**,**2**,**n**),** plt**.**imshow**(**dst**,**cmap**),** plt**.**title**(**name**)**  plt**.**xticks**([]),** plt**.**yticks**([])** # Do not show scale.  **if** save**:** cv**.**imwrite**(**f"./{path}/{name}.png"**,**dst**,[int(**cv**.**IMWRITE\_PNG\_COMPRESSION**),**0**])**  self**.**n**+=**1  *@staticmethod*  **def** show**():**  plt**.**show**()**  **def** canny**(**img**:**ndarray**,min:int=**0**,max:int=**255**)** **->** ndarray**:return** cv**.**bitwise\_not**(**cv**.**Canny**(**img**,min,max))**  **def** canny\_edge**(**img**:**ndarray**)** **->** **tuple[int,int]:**  """Test Canny upper and lower values."""  cv**.**namedWindow**(**"Canny"**)**  ### Create a track bar ### cv2.createTrackbar('Slider name', 'window name', min, max, fn)  cv**.**createTrackbar**(**"minimum"**,** "Canny"**,** 0**,** 255**,** **lambda** \_**:**\_**)**  cv**.**createTrackbar**(**"maximum"**,** "Canny"**,** 0**,** 255**,** **lambda** \_**:**\_**)**  cv**.**setTrackbarPos**(**"minimum"**,** "Canny"**,** 0**)**  cv**.**setTrackbarPos**(**"maximum"**,** "Canny"**,** 255**)**  **while** **True:**  ### Get trackbar position. ### cv2.setTrackbarPos('Slider name', 'window name', default)  **min=** cv**.**getTrackbarPos**(**'minimum'**,**'Canny'**)**  **max=** cv**.**getTrackbarPos**(**'maximum'**,**'Canny'**)**    cv**.**imshow**(**"Canny"**,**canny**(**img**,min,max))**  key **=** cv**.**waitKey**(**1**)**  **if** key **==** **ord(**'q'**)** **or** key **==** 27**:**  **break**  cv**.**destroyAllWindows**()**  **return** **min,max**  ### Convert bgr to gray and remove noise points. ###  lena **=** bgr2gray**(**"lena.jpg"**)** # 60, 141  lena\_n1 **=** bgr2gray**(**"lena\_noise.jpg"**,)** # 90, 169  lena\_n2 **=** bgr2gray**(**"lena\_noise2.jpg"**)** # 45, 164  lena\_n3 **=** bgr2gray**(**"lena\_noise3.jpg"**)** # 80, 161  ### Test Canny upper and lower values. ###  **print(**canny\_edge**(**lena**))**  **print(**canny\_edge**(**lena\_n1**))**  **print(**canny\_edge**(**lena\_n2**))**  **print(**canny\_edge**(**lena\_n3**))**  ### Plot the results. ###  plot **=** plot\_from\_matplotlib**(**"Canny"**)**  plot**(**"lena(60, 141)"**,**canny**(**lena**,**60**,** 141**))**  plot**(**"lena\_noise(90, 169)"**,**canny**(**lena\_n1**,**90**,** 169**))**  plot**(**"lena\_noise2(45, 164)"**,**canny**(**lena\_n2**,**45**,** 164**))**  plot**(**"lena\_noise3(80, 161)"**,**canny**(**lena\_n3**,**80**,** 161**))**  plot\_from\_matplotlib**.**show**()** |

# Result map



# Appendix (original picture)



Figure 1 lena(60, 141)



Figure 2 lena\_noise(90, 169)



Figure 3 lena\_noise2(45, 164)



Figure 4 lena\_noise3(80, 161)