Miroslav Roman Roson

Chapters

Computer Vision

April 10, 2019

1 Computer Vision

1.1 Edge Detection

1.1.1 Describe canny edge detection algorithm?

Here are the steps, which are needed to implement canny edge detection algorithm:

1. Grayscale conversion
2. Gaussian Blur
3. Determine the Intensity Gradients
4. Non Maximum Suppression
5. Double Thresholding
6. Edge Tracking by Hysteresis

Step 1: Convert the image to grayscale

Convert 3 channels into 1.

Original

Grayscaled

Step 2: Apply Gaussian filter to smooth the image in order to remove the noise

Since all edge detection results are easily affected by image noise, it is essential to filter out the noise to prevent false detection caused by noise. To smooth the image, a Gaussian filter is applied to convolve with the image.

Gaussian blurred

Step 3: Find the intensity gradients of the image

An edge in an image may point in a variety of directions, so the Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the blurred image. The edge detection operator (such as Roberts, Prewitt, or Sobel) returns a value for the first derivative in the horizontal direction (G_x) and the vertical direction (G_y). From this the edge gradient and direction can be determined. Here is an example of the sobel operator:

$$G_x = S_x * A = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad G_y = S_y * A = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Using the two operators you can then calculate the magnitude and the directional gradients of the image as follows:

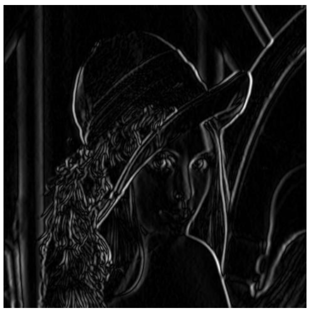
$$|G| = \sqrt(G_x^2 + G_y^2)$$

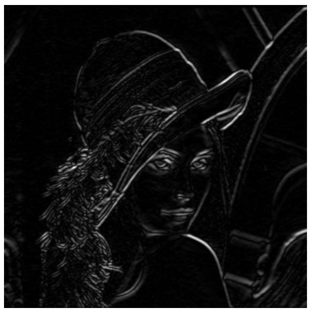
Magnitude

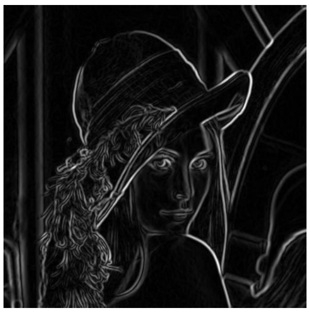
$$\Theta = arctan(G_x/G_y)$$

Directinal gradients

The resulting magnitude looks like:

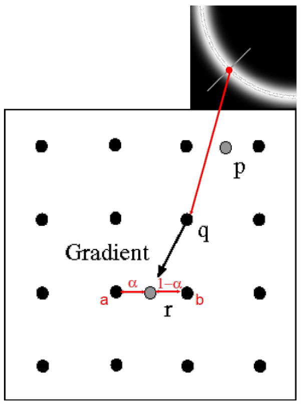

$Sobel_x$


$Sobel_y$


$Magnitude$

Step 4: Apply non-maximum suppression to get rid of spurious response to edge detection

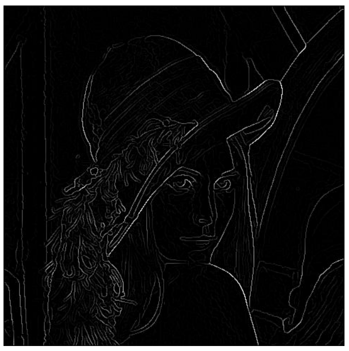
Non-maximum suppression is an edge thinning technique. After applying gradient calculation, the edge extracted from the gradient value is still quite blurred. Ideally, the final image should have only thin edges. Thus non-maximum suppression can help to suppress all the gradient values (by setting them to 0) except the local maxima, which indicate locations with the sharpest change of intensity value. The algorithm for each pixel in the gradient image is:



Non maximum suppression works by finding the pixel with the maximum value in an edge. In the above image, it occurs when pixel q has an intensity that is larger than both p and r where pixels p and r are the pixels in the gradient direction of q. If this condition is true, then we keep the pixel, otherwise we set the pixel to zero (make it a black pixel). Non maximum suppression can be achieved by interpolating the pixels for greater accuracy:

$$r = \alpha * b + (1 - \alpha) * a$$

Here is a resulting image:


Non maximum suppression

Step 5: Apply double threshold to determine potential edges

After application of non-maximum suppression, remaining edge pixels provide a more accurate representation of real edges in an image. However, some edge pixels remain that are caused by noise and color variation. In order to account for these spurious responses, it is essential to filter out edge pixels with a weak gradient value and preserve edge pixels with a high gradient value. This is accomplished by selecting high and low threshold values. If an edge pixel's gradient value is higher than the high threshold value, it is marked as a strong edge pixel. If an edge pixel's gradient value is smaller than the high threshold value and larger than the low threshold value, it is marked as a weak edge pixel. If an edge pixel's value is smaller than the low threshold value, it will be suppressed. The two threshold values are empirically determined and their definition will depend on the content of a given input image.

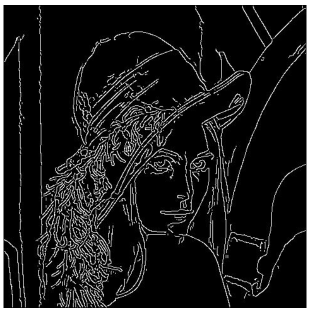
Double thresholding

Step 6: Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges

So far, the strong edge pixels should certainly be involved in the final edge image, as they are extracted from the true edges in the image. However, there will be some debate on the weak edge pixels, as these pixels can either be extracted from the true edge, or the noise/colour variations. To achieve an accurate result, the weak edges caused by the latter reasons should be removed. Usually a weak edge pixel caused from true edges will be connected to a strong edge pixel while noise responses are unconnected. To track the edge connection, blob analysis is applied by looking at a weak edge pixel and its 8-connected neighbourhood pixels. As long as there is one strong edge pixel that is involved in the blob, that weak edge point can be identified as one that should be preserved.

1.1.2 What are the drawbacks of the original canny edge detector?

While traditional Canny edge detection provides relatively simple but precise methodology for edge detection problem, with more demanding requirements on the accuracy and robustness



Final Result from Canny Edge Detection Algorithm

on the detection, the traditional algorithm can no longer handle the challenging edge detection task. The main defects of the traditional algorithm can be summarized as follows:

- Use an adaptive filter to apply different smoothing for noise and edge
- Select an optimal operator
- Use dynamic threshold values
- Use a better method for morphological detection

1. A Gaussian filter is applied to smooth out the noise, but it will also smooth the edge, which is considered as the high frequency feature. This will increase the possibility of missing weak edges, and the appearance of isolated edges in the result.

   As both edge and noise will be identified as high frequency signal, simple Gaussian filter will add smooth effect on both of them. However, in order to reach high accuracy of detection of the real edge, it is expected that more smooth effect should be added to noise and less smooth effect should be added to the edge.

2. For the gradient amplitude calculation, the old Canny edge detection algorithm uses the Centre in a small 2×2 neighbourhood window to calculate the finite difference mean value to represent the gradient amplitude. This method is sensitive to noise and can easily detect false edges and lose real edges.

   The gradient magnitude and direction can be calculated with a variety of different edge detection operators, and the choice of operator can influence the quality of results. A very commonly chosen one is the 3x3 Sobel filter. However, other filters may be better by, such as a 5x5 Sobel filter which will reduce noise or the Scharr filter which has better rotational symmetry. Other common choices are Prewitt (used by

3. In the traditional Canny edge detection algorithm, there will be two fixed global threshold values to filter out the false edges. However, as the image gets complex, different local areas

will need very different threshold values to accurately find the real edges. In addition, the global threshold values are determined manually through experiments in the traditional method, which leads to complexity of calculation when a large number of different images need to be dealt with.

In order to resolve the challenges where it is hard to determine the dual-threshold value empirically, Otsu's method (

4. The result of the traditional detection cannot reach a satisfactory high accuracy of single response for each edge - multi-point responses will appear.

   While the traditional canny edge detection have implemented a good detection result to meet with the first two criteria, it does not meet with the single response per edge strictly. A mathematical morphology to thin the detected edge is developed by

1.1.3 What filter operators do you know?

As a rule if you define your custom operator, the sum of all numbers should yield 0.

Sobel operator

$$G_x = S_x * A = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad G_y = S_y * A = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Roberts cross

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Prewitt operator

$$G_x = S_x * A = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad G_y = S_y * A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$G_{xy} = S_{xy} * A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad G_y = S_y * A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

In addition to the first operator, this one detects 45°-edges

Laplace operator

1.1.4 What is blob Connected-component labeling?

Connected-component labeling is applied in the canny edge algorithm and it is an algorithmic application of graph theory, where subsets of connected components are uniquely labelled based on a given heuristic. In other words, it is used to detect connected regions in binary digital images.

2 C++

2.1 Theoretical Part

2.1.1 Inheritance

Write down an example inheritance class

```cpp
class C: public D {
};

class B: public D {
};

class A: public B, public C {
};
```

2.1.2 Key-Words

2.1.3 General Concepts

2.1.4 Testing

2.2 Practical Part

3 Deep Learning

4 Machine Learning

5 Natural Phenomena

6 Tooling

6.1 Cmder

6.1.1 How do you change lambda promt?

On windows, if you are using anaconda, there might be a problem with activating virtualenv, since conda does't not recognize the lambda character. Thus you have to change the lambda character in the cmder as follows:

In cmder folder: vim vendor/git-for-windows/etc/profile.d/git-prompt.sh

6.2 Anaconda

6.2.1 How do you handle virtual environment in anaconda

```
# create virtual environment
conda create -n myenv python=3.4

# activate the virtual environment
# windows
source activate myenv

# deactivate the virtual environment
conda deactivate

# print all virtual environments
# * indicates the active one
conda info --envs

# delete virtual environment
conda remove --name myenv --all
```

6.3 PyCharm

6.3.1 Pylint setup

pylint is used for static code analysis.

To create a shortcut for the code analysis in pycharm:

a) File → Settings

b) Tools → External Tools

c) Click the "+" Sign to add a new Tool

d) Enter the following:

- Name: pylint
- Description: pylint code inspection
- Program: $PyInterpreterDirectory$.exe
- Arguments: -v –rcfile=.pylintrc "–msg-template='abspath:line:5d,column:2d: msg (symbol)'" –output-format=colorized "$FilePath$"
- Working directory: $ProjectFileDir$
- Advanced Options
  - Syncronize files after execution: True
  - Open Console for tool Output: True
  - Output filters: $FILE_PATH:*LINE*COLUMN:$

pycharm_pylint_setup.png

pycharm$_{pylint_{s}etup.png}$

6.3.2 bandit setup for security testing

Testing security: bandit bandit is used to check for common security vulnerabilities

To create a shortcut for the code analysis in pycharm:

a) File → Settings

b) Tools → External Tools
c) Click the "+" Sign to add a new Tool
d) Enter the following:
  - Name: bandit
  - Description: Linter for security vulnerabilities
  - Program: *PyInterpreterDirectory*
  - Arguments: -r "*FilePath*"
  - Working directory: *ProjectFileDir*
  - Advanced Options
    - Syncronize files after execution: True
    - Open Console for tool Output: True
    - Output filters: *Location:*$FILE_PATH$:*$LINE$

Automatic Code Formatting: yapf yapf is used for automatic code formatting.

To create a shortcut for the code analysis in pycharm:

a) File → Settings
b) Tools → External Tools
c) Click the "+" Sign to add a new Tool
d) Enter the following:
  - Name: yapf in-place
  - Description: Reformat Code in-place
  - Program: *PyInterpreterDirectory*.exe
  - Arguments: –style=yapf.conf -i "*FilePath*"
  - Working directory: *ProjectFileDir*
  - Advanced Options
    - Syncronize files after execution: True
    - Open Console for tool Output: True

If you don't want to replace the code in-place remove the -i argument and add another config for that (Lächeln)

Bibliography