Computer Vision hw_5

By R01922124 許彥彬

In this part the OpenCV-2.4.2 I/O function was included.

A class “Kernel” is used in this homework to denote a kernel, which used to do erosion and dilation. The octagon kernel is used in this homework.

```
class Kernel{
        public:
                int cols,rows;
                int **ele;
                Kernel(int x, int y){
                        rows=x;
                        cols=y;
                        ele = new int *[rows];
                        for(int i=0;i<rows;i++){
                                ele[i]=new int[cols];
                        }
                        for(int i=0;i<rows;i++){
                                for(int j=0;j<cols;j++){
                                        ele[i][j]=0;
                                }
                        }
                }
                void set_ele(int **k){
                        for(int i=0;i<rows;i++){
                                for(int j=0;j<cols;j++){
                                        ele[i][j]=k[i][j];
                                }
                        }
                }
};
```

1. Dilation:
   i. First, load the lena.bmp by gray scale. For all pixel in the gray_scale_lena, mask the octagon kernel and set the pixel to the max gray scale value in the kernel.
   ii. Code:

```
Mat * dilation(Mat *p, Kernel *k){
        Mat *result=new Mat(p->rows,p->cols,0);
        for(int i=0;i<p->rows;i++){
                for(int j=0;j<p->cols;j++){
                        pixel_set(result,i,j,0);
                }
        }
        for(int i=0;i<p->rows;i++){
                for(int j=0;j<p->cols;j++){
                        pixel_set(result,i,j,(int)di_ele(p,k,i,j));
                }
        }
        return result;
}

uchar di_ele(Mat *p, Kernel *k, int x, int y){
        int p_x=x-(k->rows/2);
        int p_y=y-(k->cols/2);
        uchar num=pixel_get(p,x,y);
        for(int i=0;i<k->rows;i++){
                for(int j=0;j<k->cols;j++){
                        if(p_x+i>=0 && p_x+i<p->rows && p_y+j>=0 && p_y+j<p->cols && k->ele[i][j]==1){
                                num=(uchar)max((int)num, (int)pixel_get(p,p_x+i,p_y+j));
                        }
                }
        }
        return num;
}
```

iii. Result:

2. Erosion

i. First, load the lena.bmp by gray scale. For all pixel in the gray_scale_lena, mask the octagon kernel and set the pixel to the min gray scale value in the kernel.

ii. Code:

```
Mat * erosion(Mat *p, Kernel *k){
        Mat *result=new Mat(p->rows,p->cols,0);
        for(int i=0;i<p->rows;i++){
                for(int j=0;j<p->cols;j++){
                        pixel_set(result,i,j,0);
                }
        }
        for(int i=0;i<p->rows;i++){
                for(int j=0;j<p->cols;j++){
                        pixel_set(result,i,j,(int)ero_ele(p,k,i,j));
                }
        }
        return result;
}

uchar ero_ele(Mat *p, Kernel *k, int x, int y){
        int p_x=x-(k->rows/2);
        int p_y=y-(k->cols/2);
        uchar num=pixel_get(p,x,y);
        for(int i=0;i<k->rows;i++){
                for(int j=0;j<k->cols;j++){
                        if(p_x+i>=0 && p_x+i<p->rows && p_y+j>=0 && p_y+j<p->cols && k->ele[i][j]==1){
                                num=(uchar)min((int)num, (int)pixel_get(p,p_x+i,p_y+j));
                        }
                }
        }
        return num;
}
```

iii. Result:

3. Opening

i. We first do the erosion to lena.bmp, and then do the dilation.

ii. Code:

```
Mat * opening(Mat *p, Kernel *k){
        return dilation(erosion(p,k),k);
}
```

iii. Result:

4. Closing
   i. We first do the dilation to lena.bmp, and then do the erosion.
   ii. Code:

```
Mat * closing(Mat *p, Kernel *k){
        return erosion(dilation(p,k),k);
}
```

   iii. Result:

5. Appendix
   i. build_all.sh
      command: “sh build_all.sh” will automatically compile the code
   ii. R01922124_HW5.cpp
      source code
   iii. lena.bmp, di_lena.bmp, ero_lena.bmp, close_ lena.bmp, open_lena.bmp
      results for this homework
   iv. R01922124_HW5.pdf
      report