Tutorial/Lab Session 7

PURPOSE:

1. To practice some Area-Based image processing techniques.

PROCEDURE:

Practice 1: To prepare for doing image convolution.

Step 1: Login to the PC and start the X-window environment.

Step 2: Go to the directory xvision (use "cd xvision").

Step 3: Edit the file "xvision.c" with "emacs" (use "emacs xvision.c").

Step 4: Add in the following contents before the function Test5():

```
static unsigned char  image_out[256*256] ;
static char   v_mask[3*3], h_mask[3*3] ;
void   DoConvolution(char *mask)
{
}

void   GetEdge(int x0, int y0)
{
}
void  Test5()
{
        char   filename[80] ;
        printf("\n> Enter image file : ") ;
        scanf("%s", filename) ;
        ReadImage(filename) ;
        DrawPixmap(oneimage, 256, 256, 0, 0) ;
        /* Do convolution */

        /* Get Edge */
}
```

Step 5: Compile the program (type "make").

Step 6: Execute the program (type "xvision") and click on the button <Test 5>. Enter the image file "image1.img" or "image2.img".

Practice 2: To implement the function DoConvolution() and the function GetEdge():

Step 1: Develop the function DoConvolution() with the following content:

```
void   DoConvolution(char *mask)
{
        int   i, j, s, t , v;
        for (i = 3/2 ; i < 255-3/2; i++)
           for (j=3/2; j < 255-3/2; j++)
           {
              v = 0 ;
              for (s=0; s < 3; s++)
                 for (t = 0; t < 3; t++)
                    v = v + (int) oneimage[(i-3/2+s)*256+j-3/2+t]*
                          (int) mask[s*3+t]) ;
              image_out[i*256+j] = abs(v) ;
           }
}
```

Step 2: Develop the function GetEdge() with the following content:

```
void GetEdge(int x0, int y0)
{
       int  row, col, threshold ;

       printf("\n> Enter threshold value :") ;
       scanf("%d", &threshold) ;
       for (row = 0 : row < 256; row++)
           for (col = 0 ; col < 256 ; col++)
              if ((int) image_out[row*256+col] >
                        threshold)
                             image_out[row*256+col] = 200;
              else
                             image_out[row*256+col]=0 ;
       DrawPixmap(image_out,256,256,x0,y0) ;
}
```

Step 3: Compile the program (type "make").

Step 4: Execute the program (type "xvision") and click on the button <Test 5>. Enter the image file "image1.img" or "image2.img".

Step 5: Play with different threshold values.

Practice 3: To do image convolution with a 3x3 mask (or kernel) that is sensitive to vertical edge.

Step 1: Add in the following content into the function Test5():

```
void  Test5()
{
        char   filename[80] ;
        printf("\n> Enter image file : ") ;
        scanf("%s", filename) ;
        ReadImage(filename) ;
        DrawPixmap(oneimage, 256, 256, 0, 0) ;
        /* Do convolution */
        v_mask[0] = -1 ; v_mask[1] = 0; v_mask[2]=1;
        v_mask[3] = -1 ; v_mask[4] = 0; v_mask[5]=1;
        v_mask[6] = -1 ; v_mask[7] = 0; v_mask[8]=1;
        DoConvolution(v_mask) ;

        /* Get Edge */
        GetEdge(0, 256) ;
}
```

Step 2: Compile the program (type "make").

Step 3: Execute the program (type "xvision") and click on the button <Test 5>. Enter the image file "image1.img" or "image2.img".

Step 4: Play with different threshold values and observe the results.

Practice 4: To do image convolution with a 3x3 mask (or kernel) that is sensitive to horizontal edge.

Step 1: Change the content of the function Test5() to be as follows:

```
void  Test5()
{
        char   filename[80] ;
        printf("\n> Enter image file : ") ;
        scanf("%s", filename) ;
        ReadImage(filename) ;
        DrawPixmap(oneimage, 256, 256, 0, 0) ;
        /* Do convolution */
        h_mask[0] = -1 ; h_mask[1] = -1; h_mask[2]= -1;
        h_mask[3] = 0 ; h_mask[4] = 0; h_mask[5]=0;
        h_mask[6] = 1 ; h_mask[7] = 1; h_mask[8]=1;
        DoConvolution(h_mask) ;

        /* Get Edge */
        GetEdge(256, 256) ;
}
```

Step 2: Compile the program (type "make").

Step 3: Execute the program (type "xvision") and click on the button <Test 5>. Enter the image file "image1.img" or "image2.img".

Step 4: Play with different threshold values and observe the results.

CREATIVE WORK

Combine Practice 3 & 4 to get both vertical and horizontal edge.