

## Tutorial/Lab Session 9

### PURPOSE:

1. To experiment with grouping of pixels (after binarization) according to the criterion of connectivity.

### PROCEDURE:

Practice 1: To prepare for doing pixel grouping

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 and inside the

function Test7():

```
Static unsigned char *labelmap ;
void BinarizeImage(unsigned char *pimage)
{
}
void AssignLabelToPixel(int r, int c, int label)
{
}
void Test7()
{
    char filename[80] ;
    int r, c, label ;
    labelmap = image_out ; /* share memory */
    printf("\n> Enter image file :");
    scanf("%s", filename) ;
    ReadImage(filename) ;
    DrawPixmap(oneimage, 256, 256, 0, 0) ;
    /* Binarize image */
    BinarizeImage(oneimage) ;
    /* Group pixels */
}
```

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

Step 6: Execute the program (type "xvision") and click on the button <Test 7>. Enter the image file: image3.img.

Practice 2: To re-practice image binarization (Have you forgot it already ?)

Step 1: Develop the function `BinarizeImage()` with the following content:

```
void BinarizeImage(unsigned char *pimage)
{
    int r, c, threshold ;

    printf("\n> Enter the threshold value :");
    scanf("%d", &threshold) ;
    for (r = 0 ; r < 256 ; r++)
        for (c = 0 ; c < 256 ; c++)
            if ((int) pimage[r*256+c] > threshold)
                pimage[r*256+c] = 0 ;
            else /* objects are dark */
                pimage[r*256+c] = 1 ;
    DrawPixmap(pimage, 256*256, 256, 0) ;
}
```

Step 2: Compile the program (type “make”).

Step 3: Execute the program (type “xvision”) and click on the button <Test 7>.

Step 4: Enter the image file: image3.img.

Step 5: Enter a threshold value. Observe what happens.

Step 6: Try different threshold values and identify a good one that allows to separate objects from the image background.

Practice 3: To do pixel grouping by 4-connectivity.

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

```
void AssignLabelToPixel(int r, int c, int label)
{
    if (labelmap[r*256+c] == 0)
        labelmap[r*256+c] = label ;
    else
        return ;
    if (oneimage[(r-1)*256+c] == 1) /*north*/
        AssignLabelToPixel(r-1, c, label) ;
    if (oneimage[(r+1)*256+c] == 1) /*south*/
        AssignLabelToPixel(r+1, c, label) ;
    if (oneimage[r*256+c-1] == 1) /*west*/
        AssignLabelToPixel(r, c-1, label) ;
    if (oneimage[r*256+c+1] == 1) /* east */
        AssignLabelToPixel(r, c+1, label) ;
}
```

Step 2: Add in the following content into the function Test7():

```
void Test7()
{
    (same as in Practice 1)

    /* Group pixels */
    label = 20 ;
    for (r = 0 ; r < 256*256; r++) labelmap[r]=0 ;
    for (r = 0 ; r < 256 ; r++)
        for (c = 0 ; c < 256 ; c++)
            if (oneimage[r*256+c] == 1)
                {
                    AssignLabelToPixel(r, c, label) ;
                    label = label + 20 ;
                    if (label > 255) label = 20 ;
                }
    DrawPixmap(labelmap, 256*256, 256, 256);
}
```

Step 3: Compile the program (type “make”).

Step 4: Execute the program (type “xvision”) and click on the button <Test 7>.

Step 5: Enter the image file: image3.img and a good threshold value.

Step 6: Observe the results.

**CREATIVE WORK**

To do pixel grouping by 8-connectivity.

In Practice 3, the `AssignLabelToPixel()` only deals with the 4-connectivity. (North, South, West, East) neighbors are being considered during the assignment of label. Now, add in the other four neighbors (North-West, North-East, South-West, South-East) and verify whether the result will be the same or not.

Write your comments based on what you observe.