Shashi Shivaraju
XID: C88650674

REPORT
ECE 6310 Lab #4 - Region Interaction

Objective:

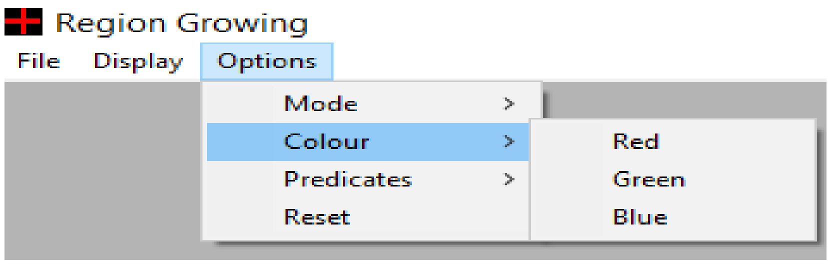
To implement interactive region growing application which allows the user to click any location in an image and visualize the results of growing a region there.

Implementation:

Implementation involved integrating the given queue based **region growing code** with the given **plus application** program which provided the basic image display features and event handling. Using Win32 Interface, new GUI features were added and utilized as listed below:

a) **GUI option to select the colour for pixels that join the region:**

Application allows the user to select from three standard colours (Red, Green, and Blue) .This selection will be used to colour the pixel which will join the region.



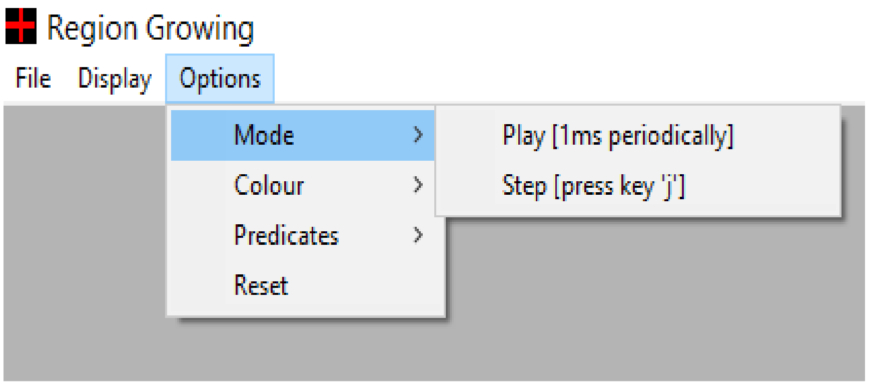
[Code snippet]:

```
/*Calback function of the application */
LRESULT CALLBACK WndProc (HWND hWnd, UINT uMsg,
                          WPARAM wParam, LPARAM lParam)
{
 case WM_COMMAND:
      switch (LOWORD(wParam))
      {
      case ID_COLOUR_RED:
      PixelColour = eRed; /*Global variable for colour set to enum of RED*/
      CheckMenuItem(hMenu,ID_COLOUR_RED,MF_CHECKED);
      CheckMenuItem(hMenu,ID_COLOUR_GREEN,MF_UNCHECKED);
      CheckMenuItem(hMenu,ID_COLOUR_BLUE,MF_UNCHECKED);
      break;
      case ID_COLOUR_GREEN:
      PixelColour = eGreen; /*Global variable for colour set to enum of GREEN*/
      CheckMenuItem(hMenu,ID_COLOUR_RED,MF_UNCHECKED);
      CheckMenuItem(hMenu,ID_COLOUR_GREEN,MF_CHECKED);
      CheckMenuItem(hMenu,ID_COLOUR_BLUE,MF_UNCHECKED);
      break;
      case ID_COLOUR_BLUE:
      PixelColour = eBlue; /*Global variable for colour set to enum of BLUE*/
      CheckMenuItem(hMenu,ID_COLOUR_RED,MF_UNCHECKED);
      CheckMenuItem(hMenu,ID_COLOUR_GREEN,MF_UNCHECKED);
      CheckMenuItem(hMenu,ID_COLOUR_BLUE,MF_CHECKED);
      break;
      }
 break;
}
```

Shashi Shivaraju
XID: C88650674

b) GUI option to select mode as “play” or “step”:

In play mode, a pixel will join the region every 1 ms and in step mode, a pixel will join the region each time the user presses the key “j”. The user can change between modes runtime i.e. even while a region is growing.



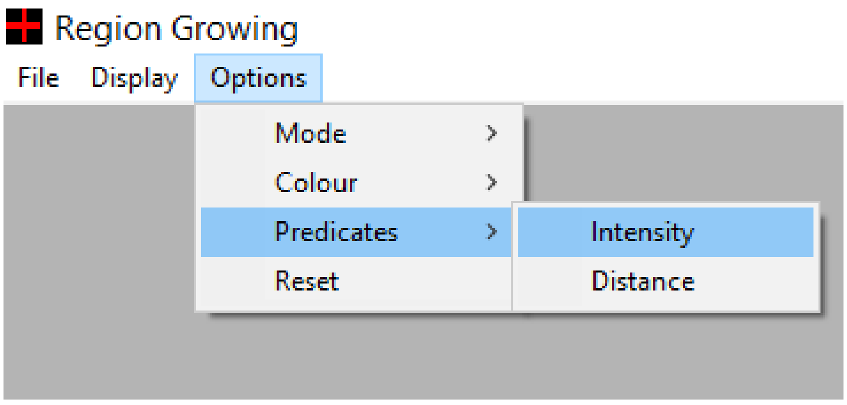
[Code snippet]:

```
/*Calback function of the application */
LRESULT CALLBACK WndProc (HWND hWnd, UINT uMsg,
                          WPARAM wParam, LPARAM lParam)
{
case WM_COMMAND:
      switch (LOWORD(wParam))
      {
      case ID_MODE_PLAY:
      DisplayMode = ePlayMode; /*Global variable for mode set to enum of play*/
      CheckMenuItem(hMenu,ID_MODE_PLAY,MF_CHECKED);
      CheckMenuItem(hMenu,ID_MODE_STEP,MF_UNCHECKED);
      break;
      case ID_MODE_STEP:
      DisplayMode = eStepMode; /*Global variable for mode set to enum of step*/
      CheckMenuItem(hMenu,ID_MODE_PLAY,MF_UNCHECKED);
      CheckMenuItem(hMenu,ID_MODE_STEP,MF_CHECKED);
      break;
      }
break;
case WM_KEYDOWN: /*Callback event when the user will press a key*/
      if ((wParam == 'j'  ||  wParam == 'J')&& (eStepMode == DisplayMode))
      for(i=0;i<255;i++)
      {
      /*If the key pressed is k and displaymode is step,then set the global
      variable maintained for possible thread to TRUE */
            StepModeState[i] = TRUE;
      }
      break;
}
```

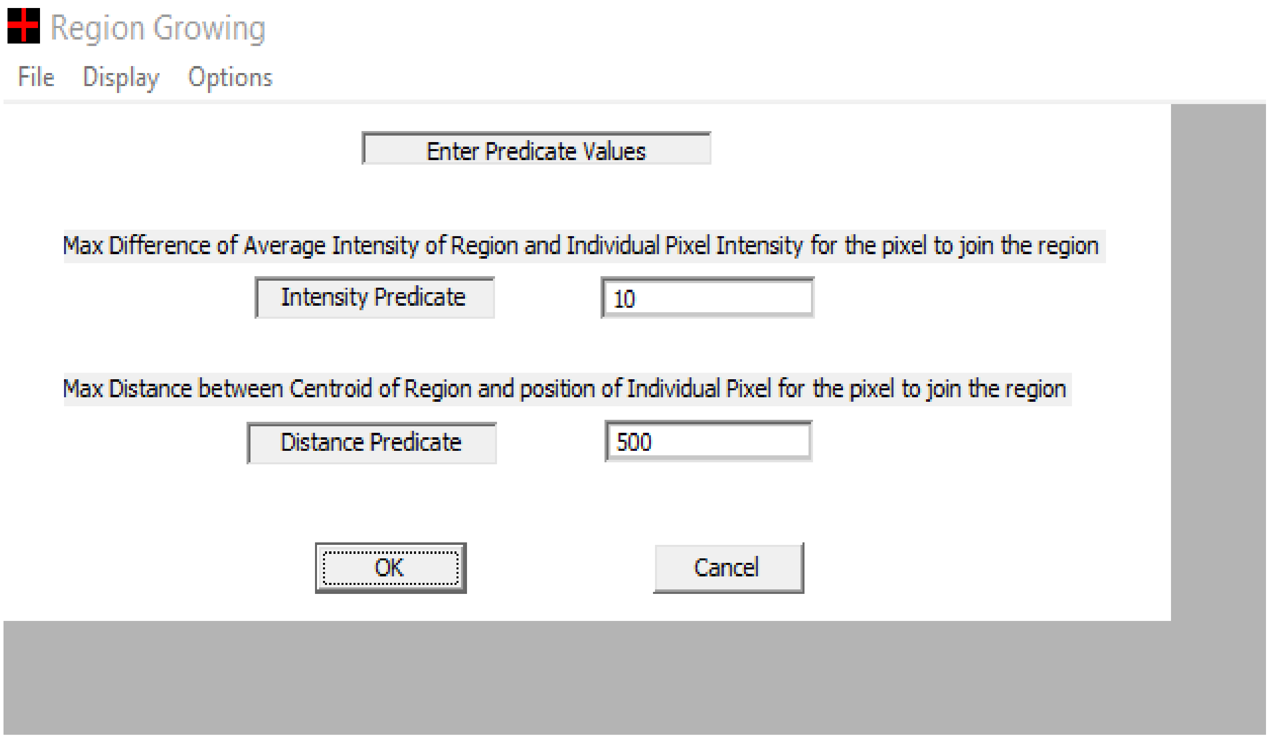
Shashi Shivaraju
XID: C88650674

c) GUI option to set the values for predicates which determine if pixel will join the region or not:

The first predicate is the absolute difference of the pixel intensity to the average intensity of pixels already in the region. (To join, a pixel must be within this range.). The second predicate is the distance of the pixel to the centroid of pixels already in the region. (To join, a pixel must be within this range.). The pixel must satisfy both of the predicates to join the region.



The user can select either Intensity or Distance from the above shown list in Application to update the predicate values. On selecting, a dialog box will appear which will allow the user to update the predicate values .



Max Difference of Average Intensity of Region and Individual Pixel Intensity for the pixel to join the region

Intensity Predicate: 10

Max Distance between Centroid of Region and position of Individual Pixel for the pixel to join the region

Distance Predicate: 500

Shashi Shivaraju
XID: C88650674

[Code snippet]:

```
/*Calback function of the application */
LRESULT CALLBACK WndProc (HWND hWnd, UINT uMsg,
                          WPARAM wParam, LPARAM lParam)
{
 case WM_COMMAND:
      switch (LOWORD(wParam))
      {
      /*On selection of either predicates,display a popup dialog to update the
      Global variables IntensityPredicate and DistancePredicate */
      case ID_PREDICATES_PIXELDISTANCE:
      case ID_PREDICATES_PIXELINTENSITY:
            hPredicateDlg = CreateDialog(GetModuleHandle(NULL),
            MAKEINTRESOURCE(IDD_DIALOG1),hWnd, PredicateDialogProc);
            if(hPredicateDlg != NULL)
            {
            ShowWindow(hPredicateDlg, SW_SHOW);
            SetDlgItemInt(hPredicateDlg, IDC_NUM_INT,IntensityPredicate, FALSE);
            SetDlgItemInt(hPredicateDlg, IDC_NUM_DIST,DistancePredicate, FALSE);
            UpdateWindow(hPredicateDlg);
            }
            else
            {
            MessageBox(hWnd, "CreateDialog returned NULL", "Warning!",
                  MB_OK | MB_ICONINFORMATION);
            }
      break;
      }
 break;
}

/*Callback function of the predicate dialog box */
BOOL CALLBACK PredicateDialogProc(HWND hwnd, UINT Message,
                                  WPARAM wParam, LPARAM lParam)
{
 BOOL bSuccess;
 switch(Message)
 {
   case WM_COMMAND:
      switch(LOWORD(wParam))
      {
      case IDC_BUTTON_OK:/* update the predicate values to new user inputs */
      IntensityPredicate = GetDlgItemInt(hwnd, IDC_NUM_INT, &bSuccess, FALSE);
      DistancePredicate  = GetDlgItemInt(hwnd, IDC_NUM_DIST, &bSuccess, FALSE);
      DestroyWindow(hwnd);
      break;
      case IDC_BUTTON_CANCEL:/*no need to update the predicate values*/
      DestroyWindow(hwnd);
      break;
      }
      break;
  default:
      return TRUE;
 }
 return TRUE;
}
```

Shashi Shivaraju
XID: C88650674

d) Visualization of Region Growing:

The user can click at any location in an image and visualize the results of growing a region from that position.

On detection of a click on image (i.e. left click), a child thread will be created and region growing to be triggered with the all the above discussed user settings (colour ,mode ,predicates) and the user will be able to visualize the region growing by observing the change in colours of the pixel as it joins the region. As stated earlier the user can switch between the play and step mode and visualize the region growing as desired. Additionally the user can click on multiple regions observed in the image and visualize the region growing simultaneously but the user will be displayed a warning popup message if the same region is selected twice.

[Code snippet]:

```
/*Calback function of the application */
LRESULT CALLBACK WndProc (HWND hWnd, UINT uMsg,
                          WPARAM wParam, LPARAM lParam)
{
case WM_LBUTTONDOWN:/*Mouse left click detected*/
      xPos=LOWORD(lParam);yPos=HIWORD(lParam);/*position of the click*/
      if (xPos >= 0  &&  xPos < COLS  &&  yPos >= 0  &&  yPos < ROWS)
      {
            int r=0,c=0,r2=0,c2=0,i=0,RegionSize=0,index =0;
            user_options *options = NULL;
            r = yPos;c = xPos;

            /*Check if the pixel is already in a region*/
            if (labels[r*COLS+c] != 0)
            {
                  MessageBox(hWnd, "Region Already selected;Select new region",
                  "Warning!",MB_OK | MB_ICONINFORMATION);
                  break;
            }
            TotalRegions++;/*Global variable to count number of selected regions*/
            if (TotalRegions == 255)
            {
                  MessageBox(hWnd, "Max selections 255 used  ;Use Reset option",
                  "Warning!",MB_OK | MB_ICONINFORMATION);
                  break;
            }
            /*Current user setting to be sent as argument to child thread*/
            options = (user_options *)malloc(sizeof(user_options));
            options->Count = TotalRegions;
            options->col = c;
            options->row = r;
            options->Colour = PixelColour;
            options->IntensityPredicate = IntensityPredicate ;
            options->DistancePredicate = DistancePredicate;
            ThreadRun[options->Count] = TRUE; /*Thread state to active*/
            /* Create a child thread for new region*/
            _beginthread(RegionGrowThread,0,(void*)options);
      }
break;
}
```

Shashi Shivaraju
XID: C88650674

```
/*Thread function which is invoked for each region selected*/
void RegionGrowThread(void* aoptions)
{
	int RegionSize=0,i=0,xPos=0,yPos=0;
	user_options *options = NULL;
	int	*indices =  NULL,index = 0;
	HDC		hDC;
	options = (user_options *)aoptions;

	/* To store the pixel locations which belong to the region*/
	indices=(int *)calloc(ROWS*COLS,sizeof(int));
	/*Function call to RegionGrow()which provides the location("indices")of all the pixels
	belonging to the region selected by the user and number of pixels in
	region("RegionSize")*/
	RegionGrow(OriginalImage,labels,ROWS,COLS,options->row,options->col,0,options->Count,
		indices,&RegionSize,options->IntensityPredicate,options->DistancePredicate);

	hDC=GetDC(MainWnd);/*Get handle to a device context (DC) for main window*/
	i = 0;
	/*While Thread is active and region is not completely coloured*/
	while(i < RegionSize && ThreadRun[options->Count])
	{
		if(ePlayMode == DisplayMode ||
			(eStepMode == DisplayMode && TRUE == StepModeState[options->Count]))
		{
			index = indices[i];

			yPos = index/COLS;/*position in pixel rows*/
			xPos = index-(yPos*COLS);/*position in pixel rows*/

			switch (options->Colour)
			{
			 case eRed:
				SetPixel(hDC,xPos,yPos,RGB(255,0,0));/* color the pixel red */
				break;
			 case eGreen:
				SetPixel(hDC,xPos,yPos,RGB(0,255,0));/* color the pixel green */
				break;
			 case eBlue:
				SetPixel(hDC,xPos,yPos,RGB(0,0,255));/* color the pixel blue */
				break;
			}
			if(ePlayMode == DisplayMode)
				Sleep(1);/*sleep for 1 ms*/
			else
			/*Change state to false and wait until next instance of j for
				  state to turn TRUE*/
				StepModeState[options->Count] = FALSE;
			i++;
		 }
	}
	ReleaseDC(MainWnd,hDC);
    }
	/*Free allocated memory for location of region pixels*/
	if(indices)
		free(indices);
	/*Free the allocated memory for options from main thread*/
	if(options)
		free(options);
}
```

Shashi Shivaraju
XID: C88650674

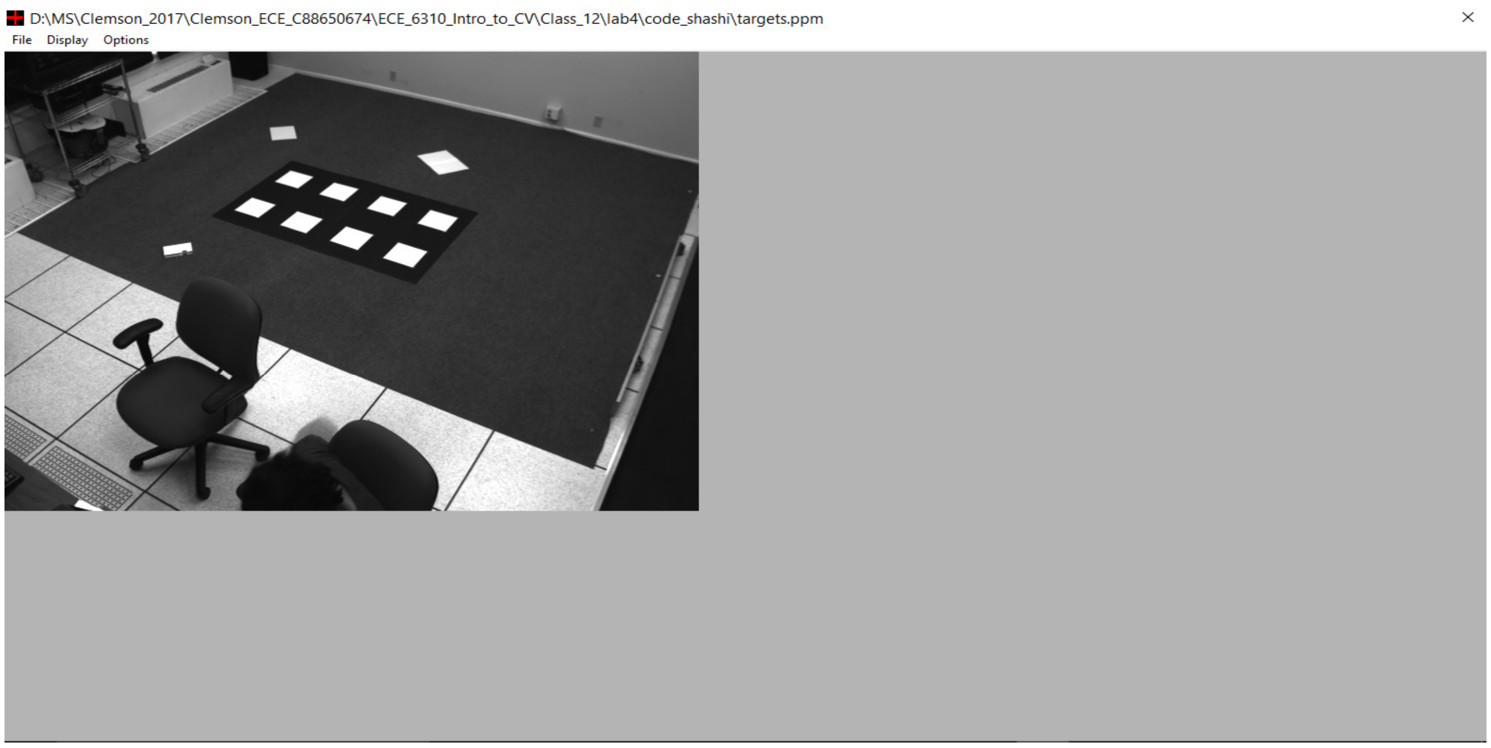
[*Code Modifications to provided RegionGrow() shown in BOLD]

```
/*macro declaration*/
#define MAX_QUEUE 10000 /* max perimeter size (pixels) of border wavefront */

/*
** Given an image, a starting point, and a label, this routine
** paint-fills (8-connected) the area with the given new label
** according to the given criteria (pixels close to the average
** intensity of the growing region are allowed to join).
*/
void RegionGrow(unsigned char *image,     /* image data */
                        unsigned char *labels,  /* segmentation labels */
                        int ROWS,int COLS,      /* size of image */
                        int r,int c,            /* pixel to paint from */
                        int paint_over_label,   /* image label to paint over */
                        int new_label,          /* image label for painting */
                        int *indices,           /* output:  indices of pixels painted */
                        int *count, /* output:  count of pixels painted */
                        int Intensity_Predicate, /*intensity predicate*/
                        int Centroid_Predicate)  /*centroid predicate*/

{
      int   r2,c2;
      int   queue[MAX_QUEUE],qh,qt;
      int   average,total;    /* average and total intensity in growing region */
      double centroid_x_pos=0,centroid_y_pos=0,dist_from_centroid=0; /*centroid varaibles*/
      int x_pos_sum = 0,y_pos_sum = 0,pixel_x_pos = 0,pixel_y_pos = 0;

      *count=0;
      if (labels[r*COLS+c] != paint_over_label)
            return;
      labels[r*COLS+c]=new_label;
      average=total=(int)image[r*COLS+c];

      /*initial centroid values*/
      centroid_x_pos = x_pos_sum = c;
      centroid_y_pos = y_pos_sum = r;

      if (indices != NULL)
            indices[0]=r*COLS+c;
      queue[0]=r*COLS+c;
      qh=1; /* queue head */
      qt=0; /* queue tail */
      (*count)=1;
      while (qt != qh)
      {

            /* recalculate average and centroid after each 50 pixels join */
            if ((*count)%50 == 0)
            {
                  average=total/(*count);
                  centroid_x_pos = x_pos_sum/(*count);
                  centroid_y_pos = y_pos_sum/(*count);
            }
```

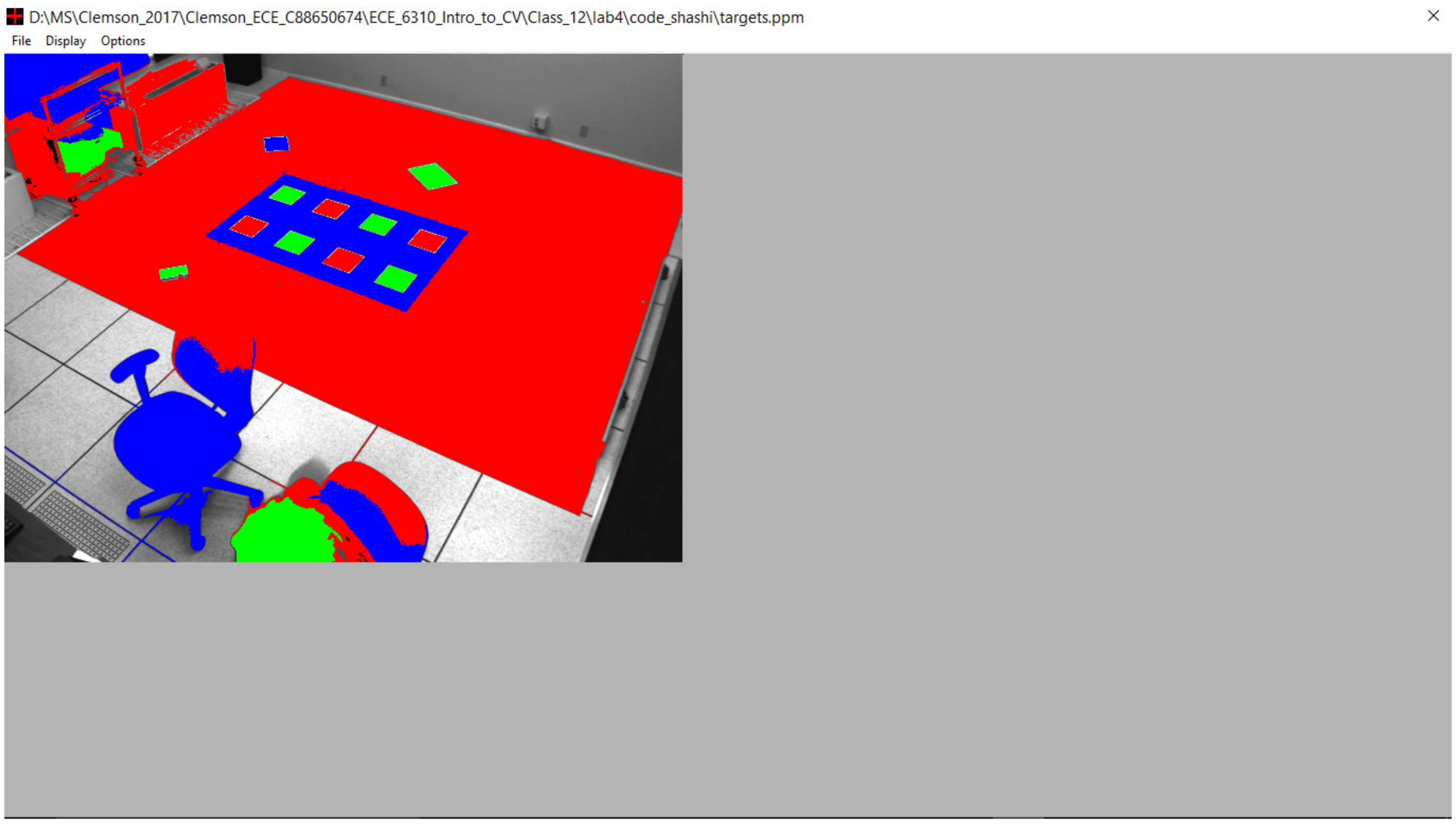
```
        for (r2=-1; r2<=1; r2++)
          for (c2=-1; c2<=1; c2++)
          {
                        if (r2 == 0  &&  c2 == 0)
                            continue;
                        if ((queue[qt]/COLS+r2) < 0  ||  (queue[qt]/COLS+r2) >= ROWS  ||
                            (queue[qt]%COLS+c2) < 0  ||  (queue[qt]%COLS+c2) >= COLS)
                            continue;

          if (labels[(queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2]!=paint_over_label)
                            continue;


          /* Intensity test criterias to join region */
          if (abs((int)(image[(queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2])-average) >
          Intensity_Predicate)
                continue;

          /*Function to obtain location in terms of xpos(col) and ypos(row)*/
          convert_index2height_width((queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2
                                    ,ROWS,COLS,&pixel_x_pos,&pixel_y_pos);
          /* Centroid test criterias to join region */
          dist_from_centroid = sqrt((SQR(centroid_x_pos - pixel_x_pos)+\
          SQR(centroid_y_pos - pixel_y_pos)));
          if(dist_from_centroid > Centroid_Predicate)
          continue;

          /* Both Predicates satisfied,Add pixel to the region*/
          labels[(queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2]=new_label;
          if (indices != NULL)/*Update the location of the added pixel*/
          indices[*count]=(queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2;
          total+=image[(queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2];

          /*For calculating the centroid*/
          x_pos_sum = x_pos_sum + pixel_x_pos;
          y_pos_sum = y_pos_sum + pixel_y_pos;
          (*count)++; /*Region size increaded by 1 pixel*/

          queue[qh]=(queue[qt]/COLS+r2)*COLS+queue[qt]%COLS+c2;
          qh=(qh+1)%MAX_QUEUE;
          if (qh == qt)
          {
                printf("Max queue size exceeded\n");
                exit(0);
          }
        }
        qt=(qt+1)%MAX_QUEUE;
    }
}
```

Shashi Shivaraju
XID: C88650674

[Application MAIN Window with Image "targets.ppm" loaded]:

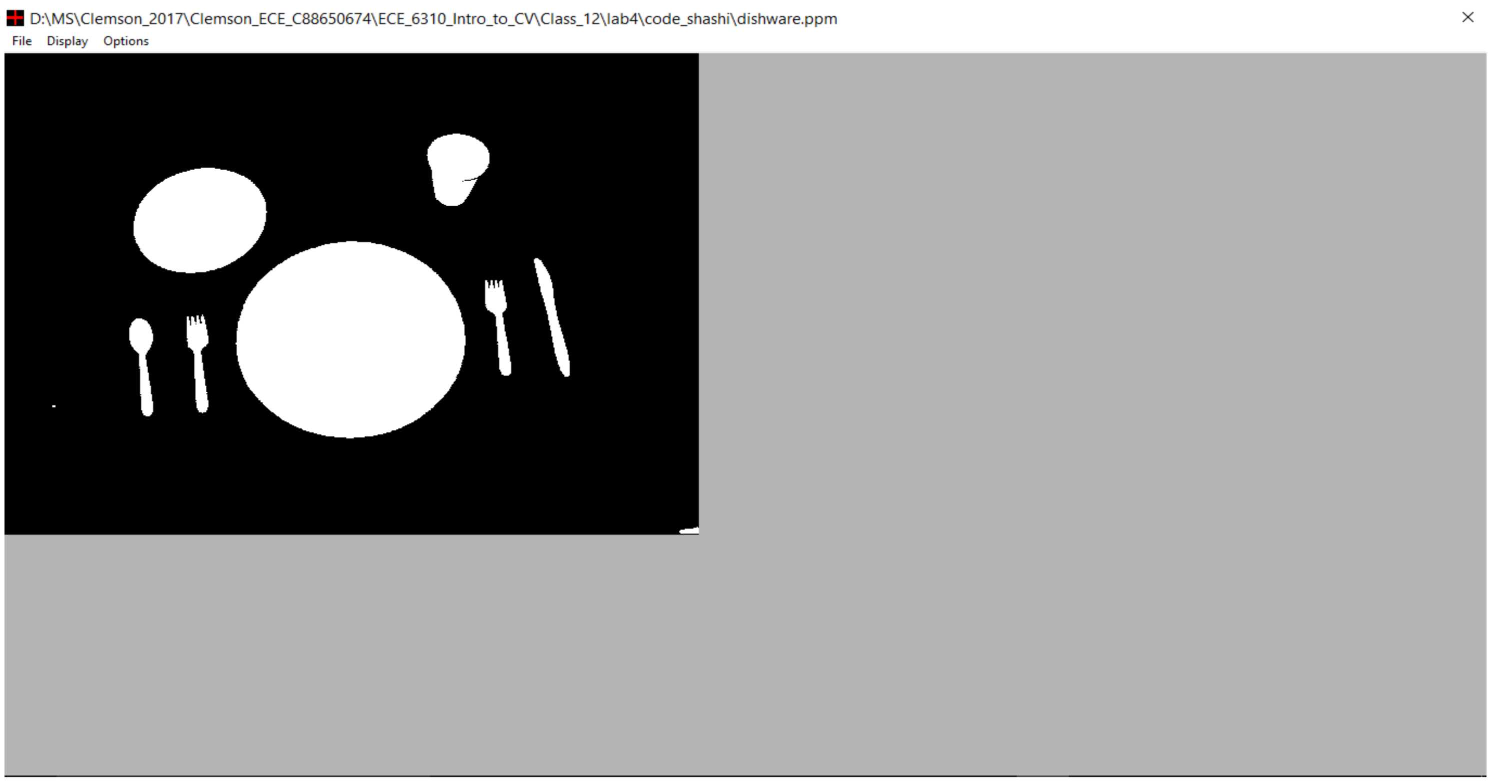

D:\MS\Clemson_2017\Clemson_ECE_C88650674\ECE_6310_Intro_to_CV\Class_12\lab4\code_shashi\targets.ppm
File
Display
Options

[Application MAIN Window with Image "Targets.ppm" loaded and regions selected by user]:

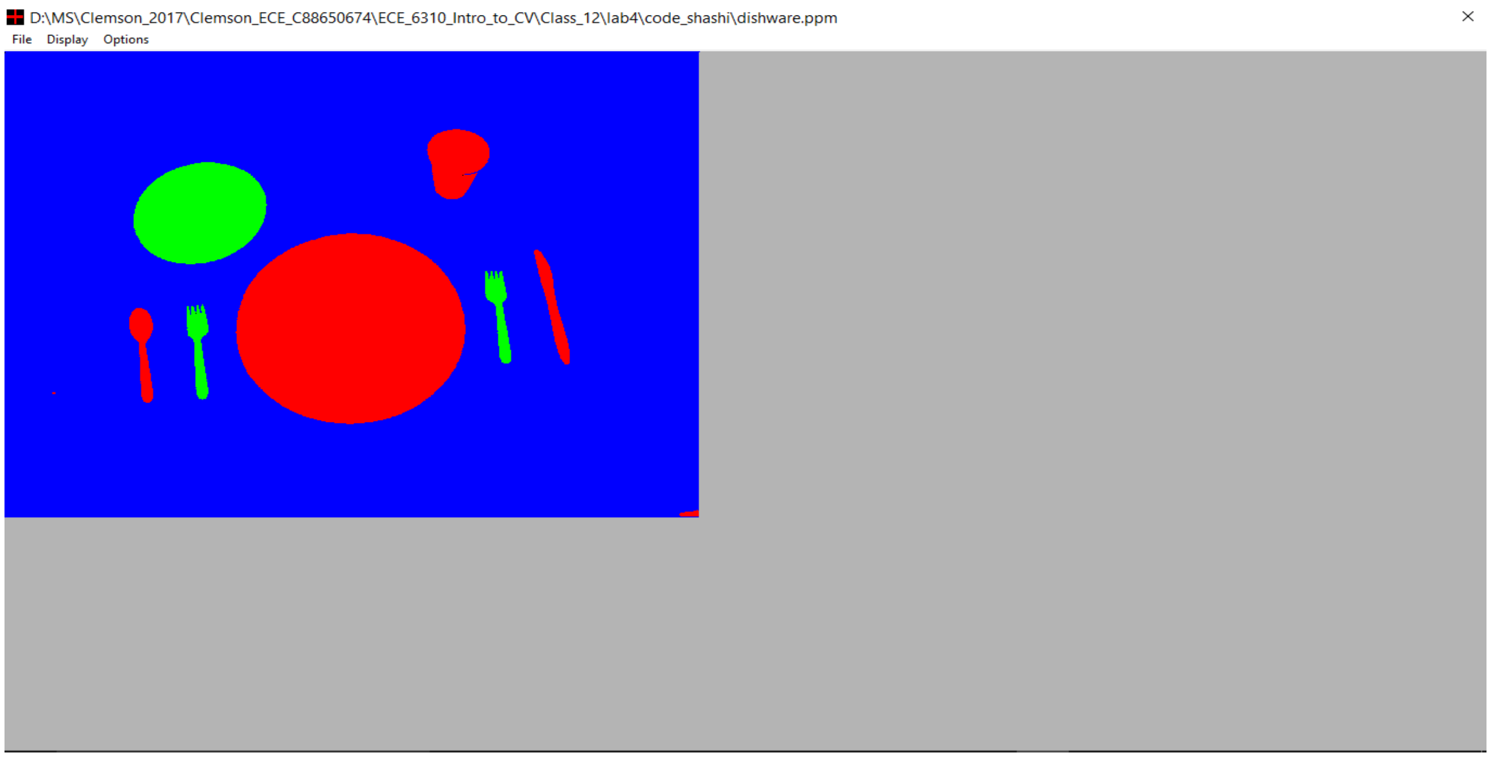

D:\MS\Clemson_2017\Clemson_ECE_C88650674\ECE_6310_Intro_to_CV\Class_12\lab4\code_shashi\targets.ppm
File
Display
Options

Shashi Shivaraju
XID: C88650674

[Application MAIN Window with Image “dishware.ppm” loaded]:

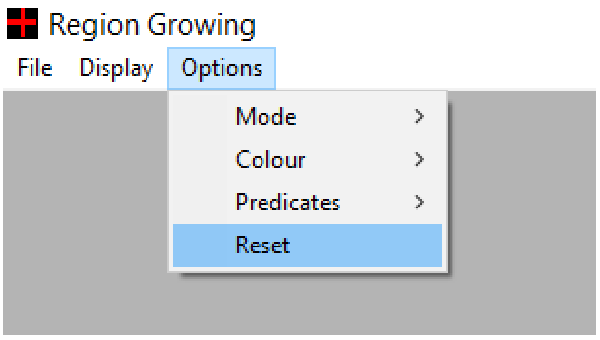

D:\MS\Clemson_2017\Clemson_ECE_C88650674\ECE_6310_Intro_to_CV\Class_12\lab4\code_shashi\dishware.ppm

[Application MAIN Window with Image “dishware.ppm” loaded and regions selected by the user]:


D:\MS\Clemson_2017\Clemson_ECE_C88650674\ECE_6310_Intro_to_CV\Class_12\lab4\code_shashi\dishware.ppm

e) GUI option to reset:

Reset clears the result of a previous region grow settings and display the original image with the default settings of the application. Reset can be applied at any instance of time.



[Code snippet]:

```
/*Calback function of the application */
LRESULT CALLBACK WndProc (HWND hWnd, UINT uMsg,
                          WPARAM wParam, LPARAM lParam)
{
 case WM_COMMAND:
      switch (LOWORD(wParam))
      {
      case ID_OPTIONS_RESET:
            /*exit all the child region growing threads and variables to zero */
            for(i=0;i<255;i++)
            {
                  ThreadRun[i]=0;/*total threads = total regions possible = 255*/
                  StepModeState[i]=0;
            }
            /*reset the lables of previous results,clears all previous regions*/
            if(labels)
            {
                  memset(labels,0,ROWS*COLS);
            }
            ShowPixelCoords = 0;/*default disabled*/
            PixelColour = eRed;/*default red*/
            DisplayMode = ePlay;/*default play*/
            IntensityPredicate = DEFAULT_INTENSITY_PREDICATE;
            DistancePredicate = DEFAULT_CENTROID_DISTANCE_PREDICATE;
            PaintImage();/*function to display the original image*/

            /*Default Application GUI settings*/
            CheckMenuItem(hMenu,ID_SHOWPIXELCOORDS,MF_UNCHECKED);
            CheckMenuItem(hMenu,ID_MODE_PLAY,MF_UNCHECKED);
            CheckMenuItem(hMenu,ID_MODE_STEP,MF_UNCHECKED);
            CheckMenuItem(hMenu,ID_COLOUR_RED,MF_UNCHECKED);
            CheckMenuItem(hMenu,ID_COLOUR_GREEN,MF_UNCHECKED);
            CheckMenuItem(hMenu,ID_COLOUR_BLUE,MF_UNCHECKED);
            break;
      }
     break;
  }
```

<END>