CSE 410: Assignment 3, 2017

Submission deadline: 8th week friday, 11:59pm

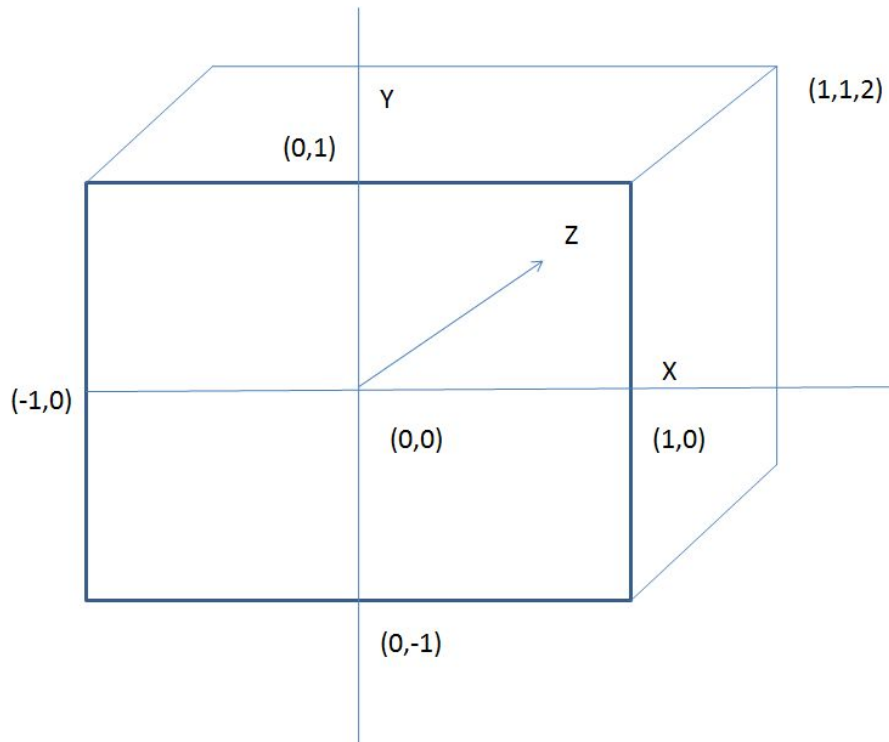# Hidden Surface Removal within the Bounding Box

Problem Definition: (The blue line represents the changes I have done, since your last class)

1. In this assignment you have to implement a very simple Hidden Surface Removal algorithm for the objects within a bounding box.

2. For simplicity we will consider only Triangles.

3. Remember the output `stage3.txt` of your `assignment2`? You have to use that as input of your program. (But during implementation always test with smaller cases)

4. Another input of your program will be from, `config.txt`
   a. The general format of this file is

        ```
         500 500
        -1.00
        -1.00
         0.00 2.00
        ```

   b. First Line of file represents [Screen_Width X Screen_Height]
   c. Second line specify the left limit of X.
      [x_right_limit=-x_left_limit]
   d. Third line specify the bottom limits of Y.
      [y_top_limit=-y_bottom_limit]
   e. Fourth line specify the front and rear limits of Z

5. **Now check the figure below for the above configuration**: imagine all your triangles resides in X, Y, Z space and you only visible volume is bounded by

```
           Y                              (1,1,2)
    (0,1)

                         Z

                              X
(-1,0)
         (0,0)            (1,0)


         (0,-1)
```

      i)   `-1<=X<=1, -1<=Y<=1, 0<=Z<=2`

      ii) Everything outside this volume have to be clipped away and will be out of visibility.

      iii) Now imagine yourself as a parallel-viewer from XY plane.

      iv) You task is to generate the image that can be seen with respect to the XY plane within this bounding volume according to the depth information of triangles.

      v) Also you need to print z_buffer value into a file named **z_buffer.txt. (only those values where z_buffer[row][col]<z_max)**

5. Your must take `stage3.txt` as your input file. As you are already familiar, the input file will contain each triangle information as three lines specifying the coordinates of the three points of the triangle. There will be no invalid cases so rest assured.
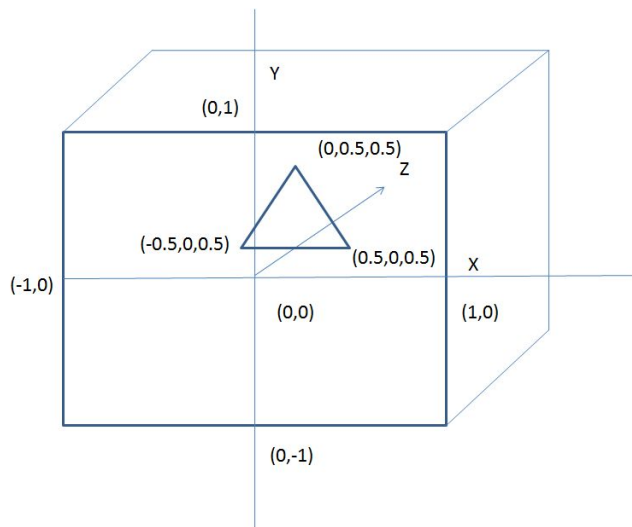
    Your output will be an image defined by `[Screen_Height X Screen_Width]` as you view from parallel to XY plane.
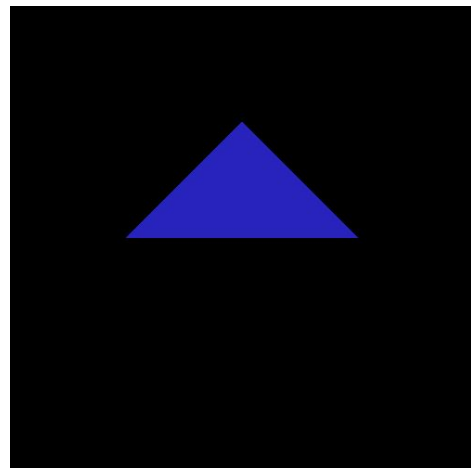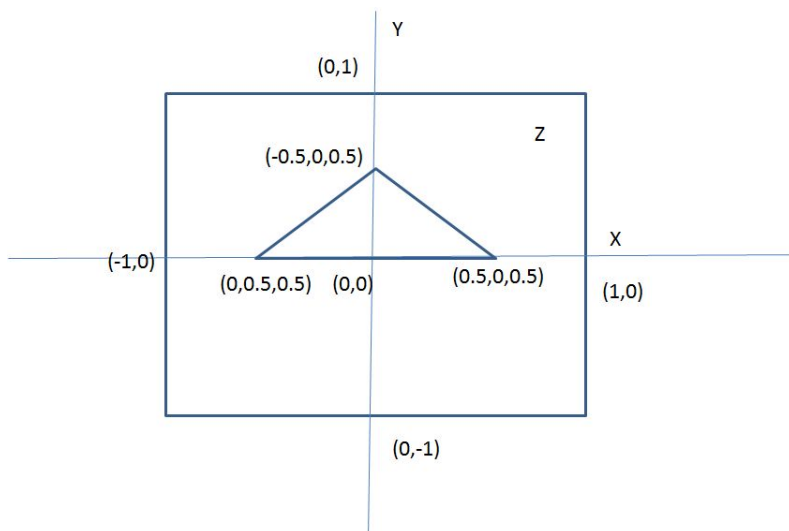
For example:
   Suppose the `stage3.txt` contains only

```
0.50  0.00  0.50
-0.50 0.00  0.50
0.00  0.50  0.50
```

That means there is only one triangle. Now imagine this within the viewing volume the triangle position can be shown by 1st figure. The actual output viewing figure from viewing plane is shown by 2nd figure.

But you have to draw this within the window defined by (Screen_Height, Screen_Width) so the output will be somewhat like,
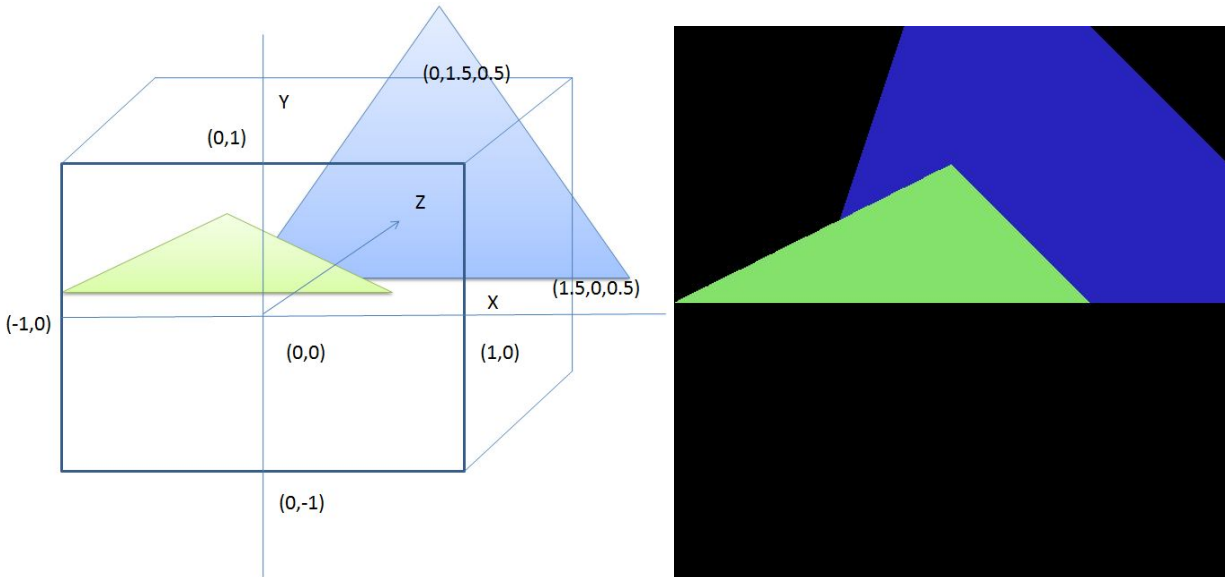

Here during drawing of each triangle we will set its' color randomly.

Another example, `stage3.txt` contains,

```
1.5 0 0.5
-0.5 0 0.5
0 1.5 0.5
```

```
0.5 0 0.25
-1.0 0 0.25
0 0.5 0.25
```

Output will be,

Procedure:

To do the above task please follow the guidelines described below.

1. inside main function
   a. `read_data()`
   b. `initialize_z_buffer_and_frame_buffer()`
   c. `apply_procedure()`
   d. `save()`
   e. `free_memory()`

2. `read_data() :`
   a. Read `config.txt` file and store the values as `Screen_Width, Screen_Height, x_limit, y_limit, z_limit` accordingly.

   b. Read input information from file named `stage3.txt`. In the file, each triangle information will be provided by consecutive three lines where each line will contain three coordinate values x, y, z as double.

c. Use a suitable data structure to hold this information. Also associate a random color value(R, G, B) with each object. RGB values are bounded by 0-255.

d. Example of a triangle object can be,

```
Triangle{
        Point points[3];
        int color[3];
}
```

d. Print and check whether you have correctly read the information from file.

3. `initialize_z_buffer_and_frame_buffer()`:

    a. You have to create a pixel mapping between the x-y range values and the `Screen_Width X Screen_height` range.

        i. To do this we first need to find the values of `dx` and `dy`. For `Screen_Width=8` and `Screen_Height=8,` `dx=` 2/8, dy= 2/8.

        ii. You also need to specify `Top_Y,` and `Left_X` values,

    As during scanning from top to bottom and left to right, we will check for the middle values of each cell. Eg. `Top_Y- r*dy,` `Left_X+c*dx.`

    Top_y= 1-dy/2, Left_x= -1+dx/2
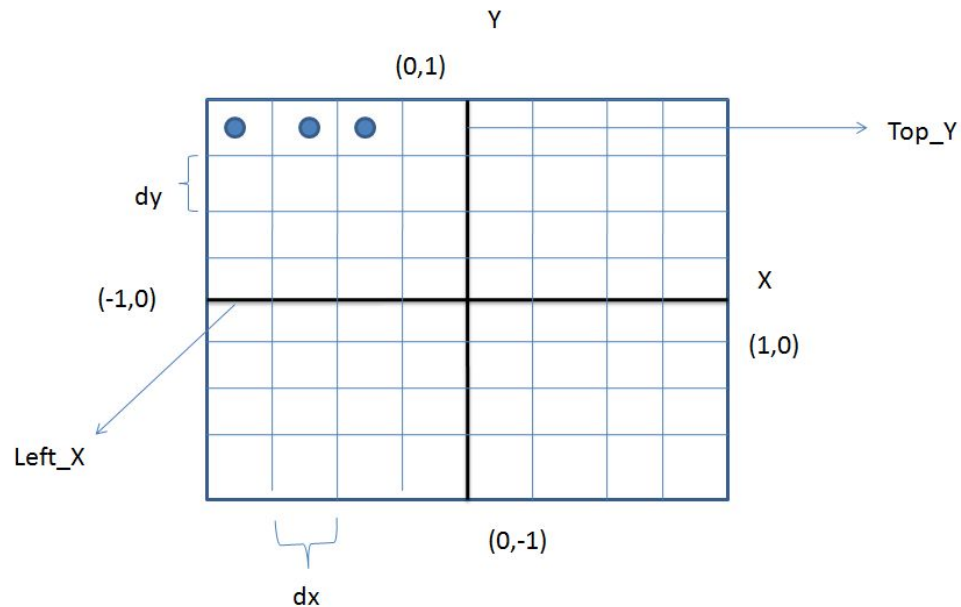
    c. Create a Z_buffer, a two dimensional array of Screen_Width X Screen_Height Dimension and Initialize its value with z_max for all positions. In example case, here z_max =2.0
(You must do this using dynamic memory allocation).

    d. Create a bitmap_image object with Screen_Width X Screen_Height resolution and initialize its background color with black.

NB: Please follow my `image_drawing.cpp` code sample for checking how to utilize the image library.

4. `apply_procedure()`:

    foreach object:Triangles

        Find top_scanline and bottom_scanline after necessary clipping

        for row from top_scanline to bottom_scanline

            Find left_intersecting_column and right_intersecting_column
            after necessary clipping

            for col from left_column to right_column
                Calculate z values
                Compare with z_buffer and z_front_limit and update if
required
                Update pixel information if required
            end
        end

    end

5. `save()`:

    Save the updated image as "output.bmp".
    Save the z_buffer values as "z_buffer.txt"
        For each row, only save those values where
z_buffer[row][col]<z_max. Check the output files

6. `free_memory()`:
    Free objects memory
    Free image memory,
    Free z_buffer memory

Important:

1. In class i have mentioned that no point will be in front of z_front_limit. I have decided to change that. Point can be in front of z_front_limit. You can easily handle this when you are going to update z_buffer just check whether z value is greater than z_front_limit or not. If greater than z_front_limit and less than z_buffer[row][col] then update.

2. **Mark Distribution:**
   https://docs.google.com/spreadsheets/d/19POfJOtJyDPPPoEaZrT06iCjFrfcVLM
   ATPveQ9hDhKM/edit?usp=sharing

3.