Assignment 3 : Moving Matrices

March 2024

1 Problem Statement :

Scene Graphs are often used in the domains of Computer Graphics and Computer Vision. They provide a convenient way of chaining translations/transformations from one component of a scene to multiple other components.

The assignment is a simple exercise to implement a simpler Scene graph using as much parallelism as possible. Given multiple images as matrices (meshes) in gray-scale along with their initial positions, a multi-set of translations, and their relationship to one another, you are to cast a scene of given dimensions.

The initial positions of the images are presented as (x, y) coordinates denoting the position where the top most, left most pixel of the image would be placed in the scene matrix.

The relationships between the images are represented by an edge list. Note that an edge from image a to image b would mean that they translate together. In other words, if image a is translated in some direction on the 2D plane, image b would have to be translated in the same direction by the same amount. However, if image b is moved, this need not be reflected in a. The translations here are transitive. The Test cases will assure that the resultant graph formed by the edges is a tree. The tree will always be rooted at 0.

While the images move, overlaps may happen. On such occasions, the more opaque matrix shall be visible in the overlapping regions (opacity of matrices are provided along with the matrix descriptions in the test cases). If the opacities are same, the matrix closer to node 0 will be displayed.

2 Input and Output Format :

2.1 Input Format :

In each testcase,

- The first line contains n, the number of meshes.
- The second line contains R C, the size of the scene ($R \times C$).
- This is followed by n descriptions for the n meshes respectively. Each of the description,
  - Starts with r c, the size of the mesh along X and Y directions
  - followed by x y, the starting coordinates of the mesh.
  - The next line has the opacity of the mesh.
  - The next r lines have c integers, together defining a mesh ($r \times c$).
- The description of n meshes is followed by the description of edges. The edges description,
  - Starts with number of edges E.
  - Followed by E lines. Each line is of the format a b, which represents an edge from mesh a to b, $a \rightarrow b$.

- Next follows the description of translations to be applied. The description,
  - Starts with T (the number of translations), followed by T lines, one for each translation.
  - Each of the T lines has, N C A, where N is the mesh number on which the translation has to be applied, C is the translation command, A is the amount.

The command C of a translation can be of the following type:

Table 1: Tanslation commands

translation command	translation
0	UP
1	DOWN
2	LEFT
3	RIGHT

2.2 Output Format :

Print out the resultant Scene as an $R \times C$ matrix. (R lines, each containing C integers).

3 Sample Test Case :

3.1 Sample Input :

```
3
11 11
7 5
3 5
1
252 249 121 107 82
20 19 233 226 45
81 142 31 86 8
87 39 167 5 212
208 82 130 119 117
27 153 74 237 88
61 106 82 54 213
2 9
4 7
3
149 95 60 53 181 196 140 221 108
17 50 61 226 180 180 89 207 206
5 6
9 3
2
249 150 252 30 224 102
44 14 123 140 202 48
66 143 188 159 123 206
209 184 177 135 236 138
214 187 46 21 99 14
2
0 1
1 2
2
```

1 2 1
2 2 4

3.2 Sample Output :

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	252	249	121	107	82	0
0	0	0	0	0	20	149	95	60	53	181
0	0	0	0	0	81	17	50	61	226	180
0	0	0	0	0	87	39	167	5	212	0
0	0	0	0	0	208	82	130	119	117	0
0	0	0	0	0	27	153	74	237	88	0
252	30	224	102	0	61	106	82	54	213	0
123	140	202	48	0	0	0	0	0	0	0

3.3 Constraints :

- $n <= 100000$
- $R <= 10000$ and $C <= 10000$
- $r <= 100$ and $c <= 100$
- $T <= 100000$

3.4 Explanation :

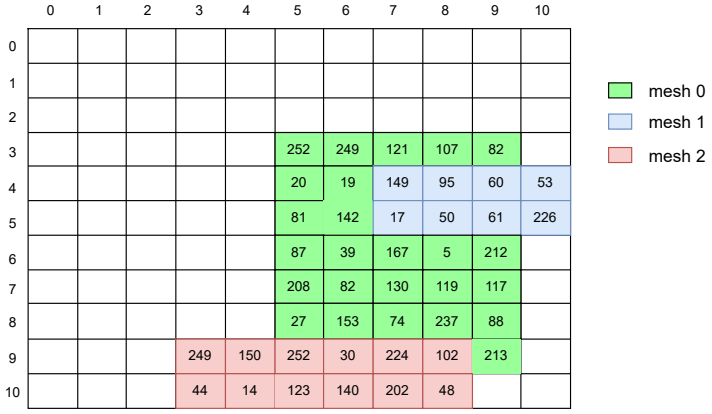

Figure 1: Initial Scene before translations (from the sample test case)

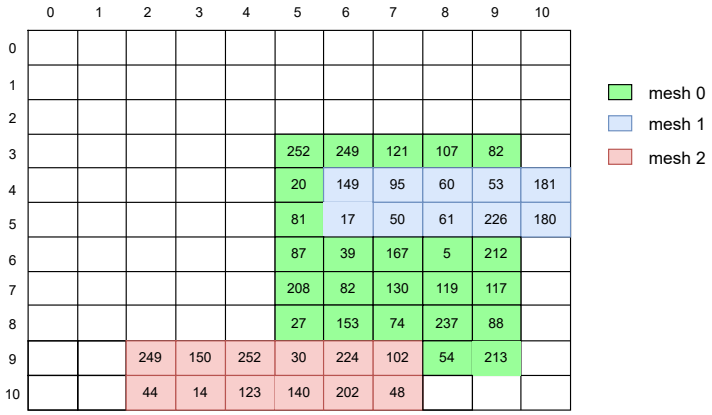

Figure 2: Scene after translation (1 2 1) [mesh 1 LEFT by amount 1]
Note: Because of the edge (1 2), the translation applies to mesh 2 also

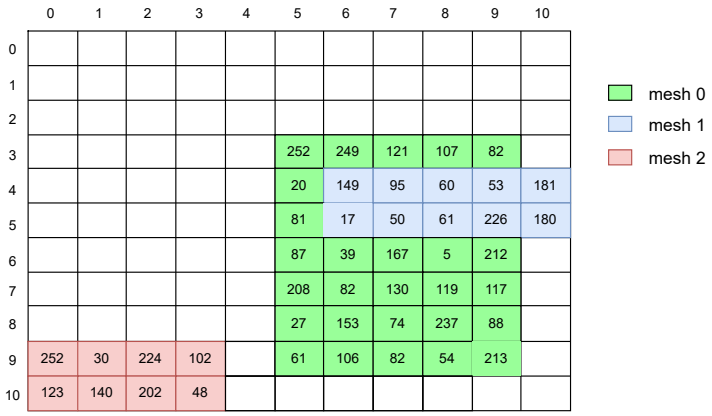

Figure 3: Scene after next translation (2 2 4) [mesh 2 LEFT by amount 4]
Note: There are no edges from mesh 2

4 General Guidelines :

- Driver code is placed inside the submit folder. Edit only the required portion of the main.cu. Do not make any changes to the other files in the submit folder.
- Test cases are to be put in the input folder and their expected outputs in the output folder.
- Run test.sh to compile and test your code. (Use command "bash test.sh" on terminal)
- The main.cu takes two command line arguments, the input file name and the output file name. (In case you'd want to examine the output of your code for debugging purposes.)
- Coordinate X represents the vertical direction (rows of the mesh) and Coordinate Y represents the horizontal direction (columns of the mesh)
- The driver code will parse the input and produce a CSR representation of the relationships between the nodes. It will attach relevant properties to the nodes (The images, their coordinates, ..).
- Execution time will be evaluated for this assignment. +2 points will be awarded for execution time lesser than 0.5*average and -2 points will be penalized for execution times greater than 2*average.
- For more on CSR refer to this slide.
- Sequential Traversal of the CSR will not be accepted as a solution.
- Submitting Sequential Code or indulging in plagiarism will be harshly penalized.

5 Submission Guidlines:

- Fill in your code in the main.cu file present within submit folder. Rename it as your ROLLNO.cu. For example, if your roll number is CS22M056, your file should be named as CS22M056.cu.
- submit zip only the ROLLNO.cu and submit as ROLLNO_A3.zip