Polar Co-ordinate visualization algorithm

Avneet Singh Channa Aranya Aryaman

November 20, 2020

1 Abstract

Affective computing is the study and development of systems and devices that can recognize, interpret, process, and simulate human affects. Affective systems improve user satisfaction and usability by identifying and complementing the affective state of a user at the time of interaction.

Vedinkaksa is a smart classroom app based on affective computing methods intended to improve the quality of teaching in blended classroom environments[1]. After identification of affective states of students in a classroom, the most important step is to provide feedback to the instructor. To provide feedback, there is a need for a visualisation scheme.

Given a classroom seating arrangement and the dimensions of the output panel (in this case a mobile screen), this paper intends to provide a dynamic visualisation scheme for circular classroom arrangements.

2 Previous Work

The earlier implementation for the app was restricted to rectangular classrooms only. The rectangular grid consisted of a grid of smaller elemnts, representing several students. Based on their emotional state calculated by [2], a threshold for criticality score was calculated and the grid was colored red, yellow, or green based on that threshold. In this paper, we extend the visualization scheme to circular/semi-circular classrooms as well. Along with this, we also propose a new grid coloring method to help the teacher understand the attentiveness of students better.

The new visualization works well for rectangular classrooms and is also effective when used in a circular class arrangement (like an auditorium or a lecture hall). Our new proposed visualization scheme uses polar coordinates to represent these cases, making the overall visualization more salient. Now, the teacher can choose between the two schemes according to the class arrangement for better insights into student behaviour.

3 Visualization Scheme

3.1 Two Level Polar Visualization Technique

3.1.1 First Level Visualization

- Mapping the students' physical locations in the classroom to radial elements, calculating the inner radii and polar angles.
- Partitioning the classroom layout based on the device's screen metrics.
- Displaying abstract view of the class.
- Color each sector according to the set threshold.

3.1.2 Second Level Visualization

- On clicking a particular ring sector, 2^{nd} level visualization gets into view
- On tap/click on student's image, the details of that student are displayed

Algorithm 1: Dynamic Radial Sectors Generation for first level visualization

Input : Classroom seating arrangement as an array $\{S_i\}$, $0 \leqslant i \leqslant R$, R = number of circular rows(rings), S_i = number of divisions in row i, D_h, D_w = height and width of output panel in pixels

Output: An array $\{G_i\}$, $0 \leqslant i \leqslant K$, K = number of rings in display output, G_i = number of sectors in i^{th} ring

`// Compute max rings`
MinDimension = $min(D_h, D_w)$
if $MinDimension \geqslant R * 100$ **then**
    K = R
else
    K = $MinDimension/100$
G = [] `// Initialize an empty array to store result`
`// Compute number of elements in ring i`
for $i \leftarrow 0$ **to** $K - 1$ **do**
    MaxDivisions = i+1
    StartRing = $\lfloor \frac{i \times R}{K} \rfloor$
    EndRing = $\lfloor \frac{(i+1) \times R}{K} \rfloor$
    Accumulate all students in $S_{StartRing} \cdots S_{EndRing-1}$ into current output ring
    `// Find largest number of divisions in currently selected rings`
    Divisions = $max(D_{StartRing}, \cdots, D_{EndRing-1})$
    if *Divisions* $\leqslant$ *MaxDivisions* **then**
        NoOfSectors = Divisions
    else
        if *isPrime(Divisions)* **then**
            Find all factors of (Divisions + 1)
        else
            Find all factors of (Divisions)
    NoOfSectors = Largest Factor Less than or equal to MaxDivisions
    if *NoOfSectors* $\times$ *2* $\leqslant$ *MaxDivisions* **then**
        NoOfSectors = MaxDivisions
    StartAngle of j^{th} sector = $\frac{j*\pi}{NoOfSectors}$ radian
    EndAngle of j^{th} sector = $\frac{(j+1)*\pi}{NoOfSectors}$ radian
    Divide students in current output ring into sectors according to polar angle
    G.append(NoOfSectors)
    Return G

Algorithm 2: Grid Coloring

Input : The radial sectors $\{G_i\}$
Output: The colored radial grid

for *each sector in grid* **do**
  Criticality Score = 0
  N = Number of students in the sector
  for *each student in that sector* **do**
    if *Corresponding state is C or LC* **then**
      Criticality Score += 1
  if $N = 0$ **then**
    Mark the sector **GRAY**
  else if *Criticality Score* $\geqslant 0.75 \times N$ **then**
    Mark the sector **RED**
  else if *Criticality Score* $\geqslant 0.50 \times N$ **then**
    Mark the sector **YELLOW**
  else
    Mark the sector **GREEN**
Render the radial grid on the output panel

Algorithm 3: Second Level Visualization

Input : The classroom seating arrangement, Student State, Colored Radial Grid, Interaction Event(Tap/LongPress)
Output: Second Level Display with details of students

while *Interaction on a ring sector in first level* **do**
  Get Location of the element(ring number and sector number)
  Get Details of the students(Name, Image) mapped to that sector from the classroom seating arrangement
  Move the radial colored grid to the bottom one-third of the screen
  Create a new display area covering upper two-thirds of the screen
  for *each student in that sector* **do**
    if *Student's criticality score is C* **then**
      Display the student's image inside a **RED** rectangle within the new display area
    else if *Student's criticality score is LC* **then**
      Display the student's image inside a **YELLOW** rectangle within the new display area
    else
      Display the student's image inside a **GREEN** rectangle within the new display area
while *Interaction in second level display* **do**
  if *Interaction on student image* **then**
    Display the details of the student in a popup display for next 20 seconds
  else if *Interaction on colored grid* **then**
    Return to first level display

4

5 Examples of Visualisation

5.1 Rectangular Classroom

We can also represent a rectangular Classroom in with a Circular Visualisation Algorithm, if required. The classroom is shaped rectangular with length = L and breadth = B. This classroom needs to be mapped to polar co-ordinate system (R,θ). In this visualization, we divide the classroom marking the midpoint of the first row as $(0, 0)$. The corner-most co-ordinates will be $(\frac{L}{2}, B)$ & $(\frac{-L}{2}, B)$. For each point, we calculate the floor of distances w.r.t our origin.

5.1.1 Mapping Technique

For any point (x, y) in the original classroom considering origin was initially same as our visualization. Distance of the point would be $\sqrt{x^2 + y^2}$. Polar angle would be $\theta = tan^{-1}(y/x)$. So, (x, y) will be mapped to (R, θ) under our new visualization scheme.We can now insert (x, y) into $S[(int)R]$ and all the elements of $S[i]$ will be sorted according to θ. We can now feed the array S to our visualisation scheme.

6 Case Study

Let us assume we have a circular classroom with 8 rows. The number of students in each row is given by the array $S = [5, 7, 9, 11, 12, 13, 14, 15]$. This means that 1^{st} row has 5 student, 2^{nd} row has 7 students and so on. We also assume for this example, that height D_h of screen = 1280px and width D_w of screen = 720px.

6.1 Input(s) to the Algorithm

$S = [5, 7, 9, 11, 12, 13, 14, 15]$
$R = 8\ (Length\ of\ S)$
$D_h = 1280px$
$D_w = 720px$

6.2 Details of the Algorithm

The algorithm first computes the following variables

$$MinDimension = min(D_h, D_w) = min(1280, 720) = 720\ px$$
$$K = \frac{MinDimension}{100} = \frac{720}{100} = 7$$
$$G = [\ ]$$

Then, it runs a for loop to calculate max number of sectors in each output ring and stores the result in the array G.

- **Iteration 1 (i=0)**
  MaxDivisions $= i + 1 = 1$
  StartRing $= \text{int}(\frac{0*8}{7}) = 0$
  EndRing $= \text{int}((\frac{1*8}{7}) = 1$
  Divisions $= \max(S_0) = \max(5) = 5$
  NoOfSectors $= 5$
  NoOfSectors $\times$ 2 is less than 1
  G.append(1)

- **Iteration 2 (i=1)**
  MaxDivisions $= i + 1 = 2$
  StartRing $= \text{int}(\frac{1*8}{7}) = 1$
  EndRing $= \text{int}((\frac{2*8}{7}) = 2$
  Divisions $= \max(S_1) = \max(7) = 7$
  NoOfSectors $= 7$
  Factors of $7 + 1 = 1, 2, 4, 8$
  Largest Factor less than or equal to MaxDivisions $= 2$
  G.append(2)

- **Iteration 3 (i=2)**
  MaxDivisions $= i + 1 = 3$
  StartRing $= \text{int}(\frac{2*8}{7}) = 2$
  EndRing $= \text{int}(\frac{3*8}{7}) = 3$
  Divisions $= \max(S_2) = \max(9) = 9$
  Factors of $9 = 1, 3, 9$
  Largest Factor less than or equal to MaxDivisions $= 3$
  NoOfSectors $= 3$
  G.append(3)

- **Iteration 4 (i=3)**
  MaxDivisions $= i + 1 = 4$
  StartRing $= \text{int}(\frac{3*8}{7}) = 3$
  EndRing $= \text{int}((\frac{4*8}{7}) = 4$
  Divisions $= \max(S_3) = \max(11) = 9$
  Factors of $11 + 1 = 1, 2, 3, 4, 6, 12$
  Largest Factor less than or equal to MaxDivisions $= 4$
  NoOfSectors $= 4$
  G.append(4)

- **Iteration 5 (i=4)**
  MaxDivisions $= i + 1 = 5$
  StartRing $= \text{int}(\frac{4*8}{7}) = 4$
  EndRing $= \text{int}((\frac{5*8}{7}) = 5$
  Divisions $= \max(S_4) = \max(12) = 12$
  Factors of $12 = 1, 2, 3, 4, 6, 12$
  Largest Factor less than or equal to MaxDivisions $= 4$
  NoOfSectors $= 4$
  G.append(4)

- **Iteration 6 (i=5)**
  MaxDivisions $= i + 1 = 6$
  StartRing $= \text{int}(\frac{5*8}{7}) = 5$
  EndRing $= \text{int}((\frac{6*8}{7}) = 6$
  Divisions $= \max(S_5) = \max(13) = 13$
  Factors of $13 + 1 = 1, 2, 7, 14$
  Largest Factor less than or equal to MaxDivisions $= 2$
  NoOfSectors $= 2$
  NoOfSectors $\times$ 2 is less than 6
  G.append(6)

- **Iteration 7 (i=6)**
  MaxDivisions $= i + 1 = 7$
  StartRing $= \text{int}(\frac{6*8}{7}) = 6$
  EndRing $= \text{int}((\frac{7*8}{7}) = 8$
  Divisions $= \max(S_6, S_7) = \max(14, 15) = 15$
  Factors of $15 = 1, 3, 5, 15$
  Largest Factor less than or equal to MaxDivisions $= 5$
  NoOfSectors $= 5$
  G.append(5)

Return G and STOP

6.3 Output(s)

Array $G = [1, 2, 3, 4, 4, 6, 5]$. This means that there will be a total of 6 rings, where 1^{st} ring has 1 sector, 2^{nd} ring has 2 sectors, 3^{rd} ring has 3 sectors and so on. The visualization is shown in Figure 1

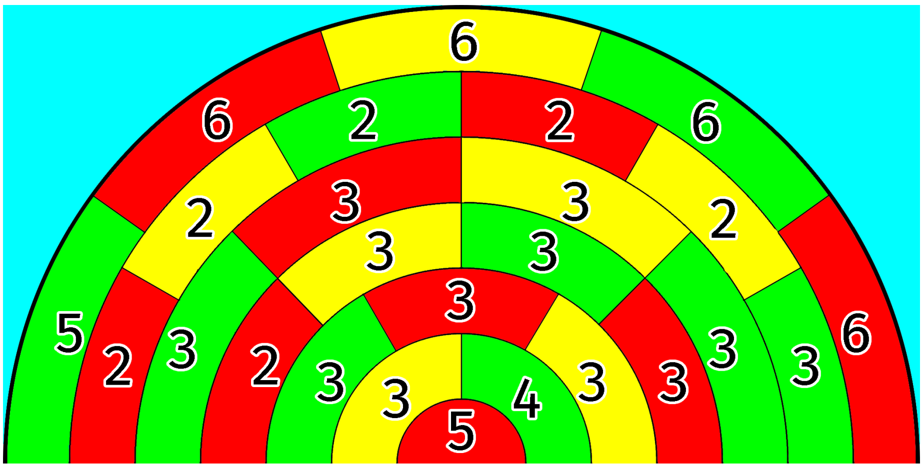

Figure 1: The resulting visualization. The numbers inside each sector correspond to the number of students mapped to that sector. The blue area is not used.

7 Contributions

7.1 Aranya Aryaman

- Method to convert Rectangular Classroom into Circular Classroom and vice-versa
- Based on the new grid-coloring algorithm, devised a new way to implement the second-level of the visualisation scheme
- Case Study to show the implementation of the entire algorithm
- Provided inputs for dynamic grid generation and grid-coloring algorithm

7.2 Avneet Singh Channa

- Dynamic Grid Generation scheme to automatically divide a classroom into radial sectors
- Devised an improved Grid Coloring Algorithm to remove redundancies in the existing algorithm
- Case Study to show the implementation of the entire algorithm
- Provided inputs for second-level visualisation and inter-conversion of co-ordinate system

References

[1] S. Tikadar, S. Bhattacharya, and V. Tamarapalli, "A blended learning platform to improve teaching-learning experience," in *2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT)*, pp. 87–89, July 2018.

[2] S. Tikadar and S. Bhattacharya, "A novel method to build and validate an affective state prediction model from touch-typing," in *Human-Computer Interaction – INTERACT 2019* (D. Lamas, F. Loizides, L. Nacke, H. Petrie, M. Winckler, and P. Zaphiris, eds.), (Cham), pp. 99–119, Springer International Publishing, 2019.