Vectorized Images

2026-27

Mathematics Club

Centre for Innovation
IIT Madras

Instructions

General

- This is not an assignment in the traditional sense. It is meant for you to learn and have fun!
- You are free to use the internet and any other resources for solving the assignment. Please cite whatever reference(s) you use. However, we would suggest refraining from LLM use.
- The bonus goes beyond the purview of what we learned in the session. It is completely optional. If you'd like to explore these topics a little further, you can try those out.
- The goal of the application is for you to learn as much as possible. Resources have been provided in the end for all sections. All the best and have fun!

Assignment Format

- Your solutions can be written manually, digitally, documents or in LaTeX. Submit all your scanned copies with the following format <FullNameSS26_VI.pdf>. For example, if my name is Arjun Arunachalam, I would submit ArjunArunachalamSS26_VI.pdf.

Submission

- The final PDF is to be submitted through this Google Form - Click Here
- The deadline for submission is **31st July 2026, 23:59**.

Contact Us

- Kailash Anand - +91 97909 08101
- Aarav Prakash - +91 81518 42542

Vectorized Images

Mathematics Club

§1 Animating with Beziers

In this section, we will use our knowledge of Beziers and splines to draw a simple circular progress bar animation. Before getting started, here is a sneak-peek of what we will try to build:

In order to complete this section, please click here to open the required Desmos graph.

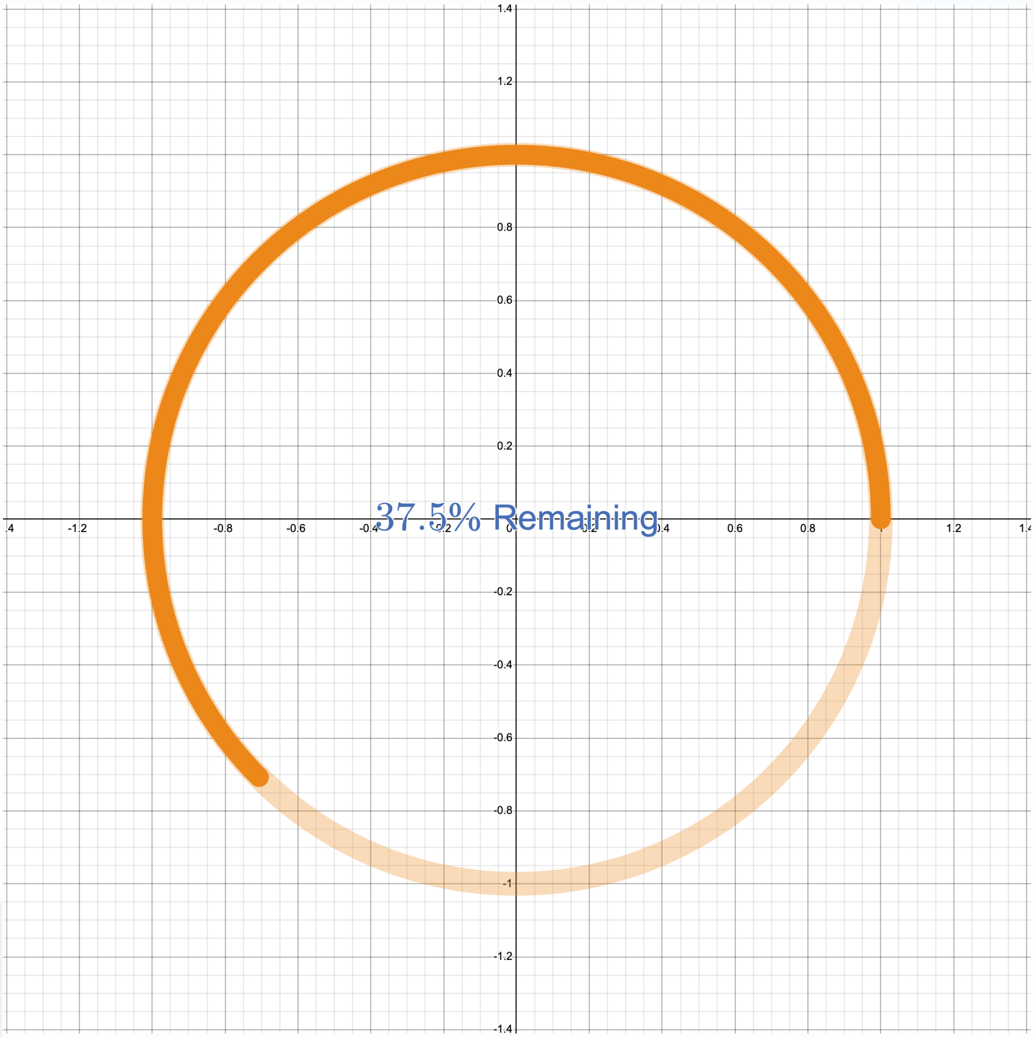

Figure 1: Our goal in this section

I unfortunately couldn't figure out how to get exact timing working in Desmos, so I leave that as an open challenge for you all to try out after this section.

§1.1 You gotta walk before you run: Drawing a quarter circle

IMPORTANT:

If I tell you to use a specific variable name, please use only that. The animation part I've put in Desmos won't work otherwise.

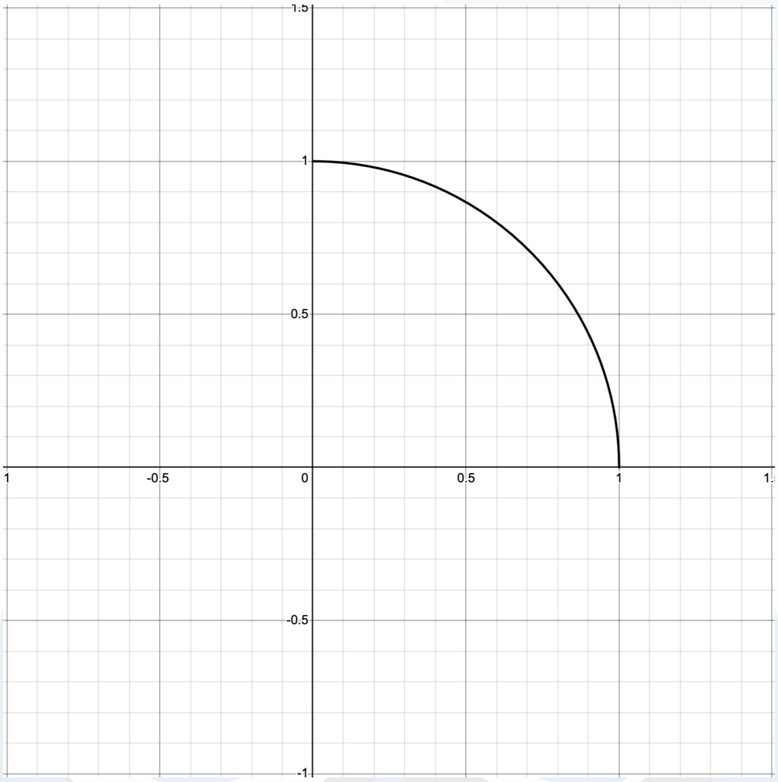

Figure 2: Quarter circle in the first quadrant

Many CAD software and animation engines don't use $x^2 + y^2 = 1$ to draw circles! They instead use Bezier curves. In this part, we will be using a single cubic Bezier to approximate the first quadrant of a circle.

Recall that it is of the form:

$$\mathbf{r}(t) = \sum_{k=0}^{3} \left(\mathbf{p}_k \cdot \binom{3}{k} \cdot t^{3-k}(1-t)^k \right), \forall t \in [0,1]$$

and in matrix form:

$$\begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_2 \\ \mathbf{p}_1 \\ \mathbf{p}_0 \end{bmatrix}$$

Questions

Q1.1 (Bonus) Notice that the characteristic matrix is lower triangular, symmetric about the non-main diagonal, all the elements sum up to 1 and all the rows sum up to 0 except the top row, which sums to 1. What geometric properties of the Cubic Bézier can we infer from these properties? (think about recursivity, symmetry, the basis summing to 1. Feel free to think about other properties. Refrain from getting direct answers from LLMs)

Q1.2 The start and end points of the Bézier clearly need to be $(1, 0)$ and $(0, 1)$ respectively. But what about the control points? We would like the curve to be parallel to the y-axis at $t = 0$, the starting point, and parallel to the x-axis at $t = 1$, the end point. So what should be the form of our control points?
Hint: Think about the derivative w.r.t. t of the curve

You should have gotten that $\mathbf{p}_3 = (1, \alpha)$ and $\mathbf{p}_2 = (\beta, 1)$. This is because for any Bézier curve, by its very definition, the line connecting an end-point to its immediate next control point is a tangent to the curve.

Since a circle is perfectly symmetric, let's enforce symmetry about the line $y = x$. Further, let's make the curve pass through the point $\left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$: the $45°$ point of a unit circle when $t = \frac{1}{2}$.

Questions

Q1.3 To enforce both these conditions, prove that we must choose $\alpha = \beta = \frac{4}{3}(\sqrt{2} - 1)$

You now have all the 4 points required to draw a cubic Bézier curve. Before plotting this directly on Desmos, we must do an intermediate step to make our future a bit better.

Graphing

G1.1 Split the 2D Bézier curve into its x and y components, $Q_x(t)$ and $Q_y(t)$ respectively.

G1.2 Define a new function $Q_1(t) = (Q_x(t), Q_y(t))$. Make sure to type $Q_1(t)$ separately in a new tab to tell Desmos to draw the graph.

The curve should look very close to an actual circular arc! Many CAD and vector graphics software use these very equations and points to draw their circles! Draw an actual quarter circle to really see how well the Bézier approximates the arc!

Graphing

G1.3 To complete the circle, make 3 more functions: $Q_i(t)\ \forall i \in \{2, 3, 4\}$, for each of the 3 remaining quadrants. Use the previously defined functions $Q_x(t)$ and $Q_y(t)$ to write them.

NOTE: After defining each function, make sure to write $Q_i(t)$ separately in another tab to draw the function on the graph. Please keep this in mind for the rest of the assignment, I won't be repeating this hereafter.

Now you should have a complete circle, drawn purely with Béziers! Draw an actual unit circle using $x^2 + y^2 = 1$ to see how closely the Bézier approximates that.

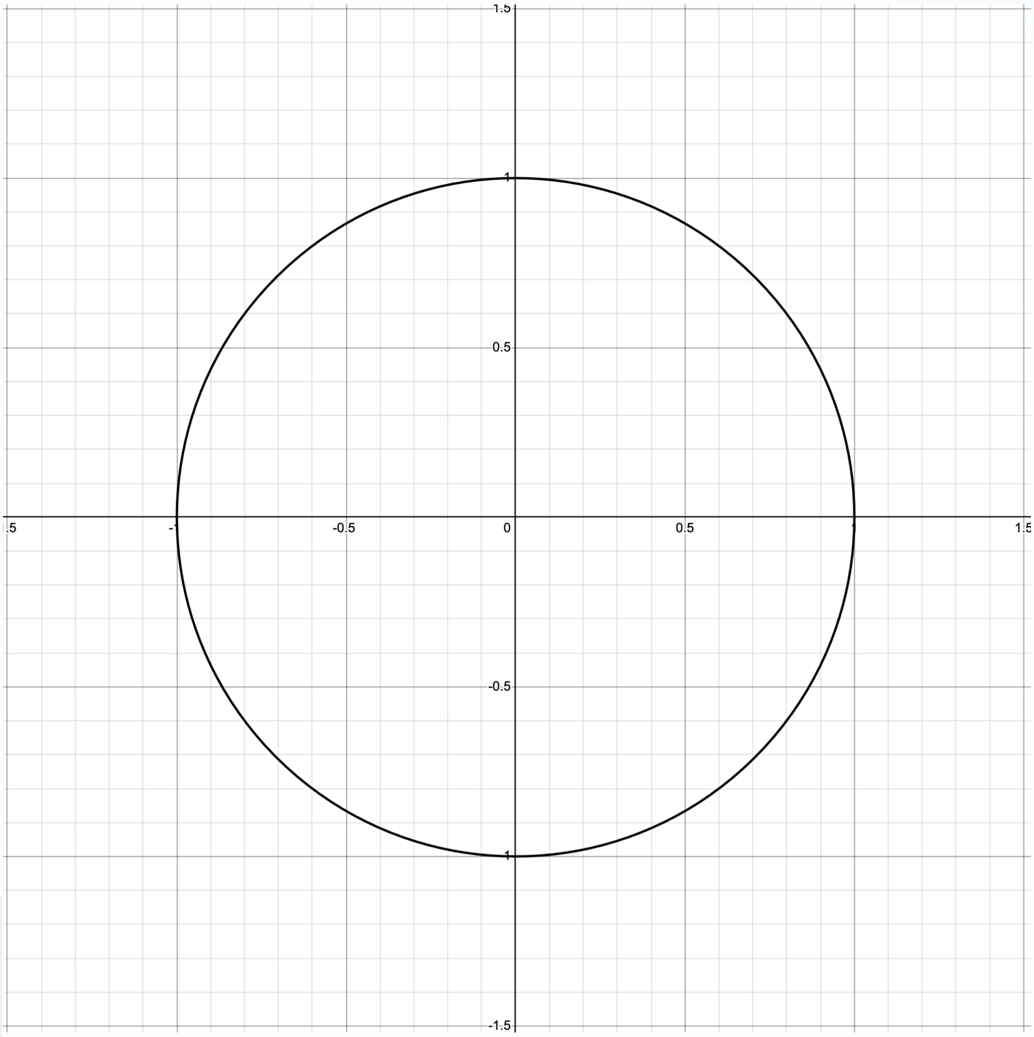

Figure 3: A full circle with 4 cubic Béziers!

§1.2 It's time to ~~run~~ spline

In this section, we will put each of the quadrants into a single circle function $C(t)$.

The Strategy:

As t ranges from $0 \to 4$, we will cover each quadrant. The $\lfloor t \rfloor$-1 will denote which quadrant it is, and $t - \lfloor t \rfloor$ will be the parameter value entered into the corresponding quadrant's equation. This will let us use a single parameter t to cover the entire circle!

For example, $t = 3.44$ would mean we are in the 2nd quadrant, and we need to input a parameter value of 0.44.

Graphing

G1.4 Define a function $C(t)$. It should be piecewise defined for each quadrant: $t \in [0, 1)$ is quadrant-1, $t \in [1, 2)$ is quadrant-2 and so on. As the parameter t runs from 0 to 4, $C(t)$ should transition between each quadrant.

To see if this has been done right, let's do the following:

In a new tab, type $C(s)$, and let the slider for s range from 0 to 4. Hit the play button to see how the point moves. You might notice that at the end of each quadrant, the point suddenly jumps. Why does this happen, and how can we fix this?

§1.3 Fixing the Parametrization

Questions

Q1.4 Remember in the session we talked about reversing motion? We used this as the motivation to introduce symmetry in our basis set. Using this as a clue, can you now fix this issue with the animation? Feel free to refer to the session, by the way.
Hint: Think about what you would expect $Q_2(0)$ to be, and what our current situation gives.

Q1.5 Finally, can you make this loop beyond $s = 4$? That is, when $s = 4.6$, it should behave like $s = 0.6$.
Hint: Use modular arithmetic. This is a pretty good trick to get things to become periodic. Let s range from 0 to 12 to see if your solution works.

§1.4 Introducing the animation tab

There is a folder named: "**ANIMATION (don't open)**"(I will refer to this as the "animation tab" henceforth), activate that. Inside the tab, you will notice a few different variables defined.

Questions

Q1.6 Explain what each of those variables determine, and the logic behind the way they have been defined - what is each variable trying to achieve?

Q1.7 If you let the t_0 slider play, you will notice that the animation is very linear and doesn't have any smoothing. This is because we directly input the t into $C(t)$. We could run t through another function $f(t)$ such that $f(t)$ is monotonic for all $t \in [0, 4]$ and has a range $[0, 4]$. Suggest some functions we could use to change the "smoothing" of our animation. Also explain what features you look for in $f(t)$ to achieve the required kind of smoothing.

§2 (Bonus) Beyond the Land of Béziers

Questions

Q2.1 Can you generate a polynomial basis of degree 3 so that displacing each point by a vector $\mathbf{v}$ would displace the entire curve by $a\mathbf{v}$ for some $a > 0$? Is it possible to make a basis that is resistant to displacement? Meaning, even if I displace all the points by some non-zero vector $\mathbf{p}$, the curve will stay-put and not move.

Q2.2 In this assignment, we made a circle using a set of polynomial functions. If we allow for rational functions of the form $\dfrac{P(t)}{Q(t)}$ where $P(t), Q(t)$ are polynomials, is it possible to achieve an exact circle?
Hint: Think about trigonometric formulas like $\sin 2\theta$, $\cos 2\theta$

Q2.3 Look up B-Splines and NURBS. Can we use those to draw an exact circle? If so, then why would anyone pick NURBS to draw a circle over the much simpler rational functions we discovered in the previous question?

§3 Resources

1. Our Canva presentation: Vectorized Images SS26
2. Our YouTube Livestream: Vectorized Images | Summer School 2026 | Mathematics Club IIT-M
3. A fun website to play with Béziers to draw shapes: The Bézier Game
4. An excellent video covering Béziers and Splining: The Continuity of Béziers and Splines by Freya Holmer
5. A Primer on Bézier Curves: A Primer on Bézier Curves
6. An article on the matrix representation of Cubic Béziers: A Matrix Formulation of the Cubic Bézier Curve
7. A 3Blue1Brown video explaning convolutions (to understand edge detection): But what is a convolution?

And for my dear book lovers, here are some books you could refer to:

8. A Practical Guide to Splines (Revised Edition) by Carl de Boor

9. Curves and Surfaces for Computer Aided Geometric Design (4th Ed) by Gerald Farin