# Assignment 5 - Bezier Curves

Mikkel Willén

January 18, 2023

## Contents

# 1 Introduction

This assignment is about understanding, explaining and implementing different algorithms to visualize Bezier curves. In this report we will look at Hermite curves, which gives a matrix $G$ and a basis matrix $M$. We can use these matrices to get the Bezier curves. We will also look at sampling, forward differencing, subdivision and flatness, which we use to draw Bezier curves.

# 2 Theory

Before we take a look at Bezier curves, we will first look at Hermite curves. Hermite and Bezier curves are to different ways of describing a 3rd degree polynomium. A Hermite curve kan be expressed as a Bezier curve and the other way around.

3rd degree polynomiums is the lowest degree of polynomiums which can be a not-plane. A 3rd degree polynomium can be written in the following way.

$$f(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = \begin{bmatrix} a_x + b_x + c_x + d_x \\ a_y + b_y + c_y + d_y \\ a_z + b_z + c_z + d_z \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix} \quad , \quad 0 \leq t \leq 1$$

We have a coefficient matrix $C$ and a parameter vector $t$. If we want to change the curve, we need to change the $C$ matrix. We rewrite $C$ so it is easier to change the values of it. We split it up into two matrices $G$ and $M$. $G$ is the geometric matrix and desides how the curve looks. $M$ is the basis matrix and decides which type of curve it is. They are defined as follows:

$$f(t) = Ct = GMt$$

$G$ is a 3x4 matrix and $M$ is a 4x4 matrix.

Hermite curves are defined by a starting point $G_1$ and an endpoint $G_2$. These are the entries to the geometric matrix. The 3rd entry $G_3$ is the tagent to the starting point, and the 4th entry $G_4$ is the tagent to the endpoint. It is shown on the figure below. We can write the geometric matrix for Hermite curves:

$$G_H = \begin{bmatrix} G_1 & G_2 & G_3 & G_4 \end{bmatrix}$$

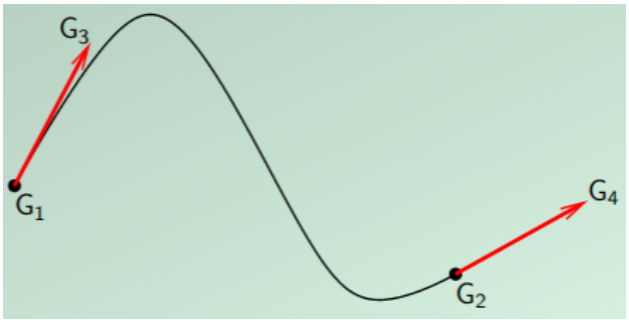


Figure 1: Hermite curve with the points $G_1$, $G_2$, $G_3$, $G_4$

We now need to find the basis matrix $M$ and we have

$$f_H(t) = G_H M_H t$$

We do not know what $M_h$ is, but we know something about the function. We set $t = 0$ as our starting point and $t = 1$ as our endpoint and get

$$f_H(0) = G_1 = G_H M_H \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$f_H(1) = G_2 = G_H M_H \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

To get the tagent of the curve we need $f'(t)$ which is

$$f'_H(t) = G_H M_H \begin{bmatrix} 3t^2 \\ 2t \\ 1 \\ 0 \end{bmatrix}$$

We find the startpoint and endpoint by setting $t = 0$ and $t = 1$ and we get

$$f'_H(0) G_3 = G_H M_H \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$f'_H(1) = G_4 = G_H M_H \begin{bmatrix} 3 \\ 2 \\ 1 \\ 0 \end{bmatrix}$$

We can use these equations to get $G_H$

$$\begin{bmatrix} G_1 & G_2 & G_3 & G_4 \end{bmatrix} = G_H = G_H M_H \begin{bmatrix} 0 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

This is only true, if $M_H$ is the inverse, so we have

$$M_H = \begin{bmatrix} 0 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}^{-1} = \begin{bmatrix} 2 & -3 & 0 & 1 \\ -2 & 3 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}$$

Now we got both matrices and we are able to change them, if we want to.

Bezier curves have a startpoint $G_1$ and an endpoint $G_4$. The points $G_2$ and $G_3$ are controlpoints. Controlpoints do not necessarily lie on the curve, which can also be seen on the figure below. The geometric matrix $G_B$ is written as

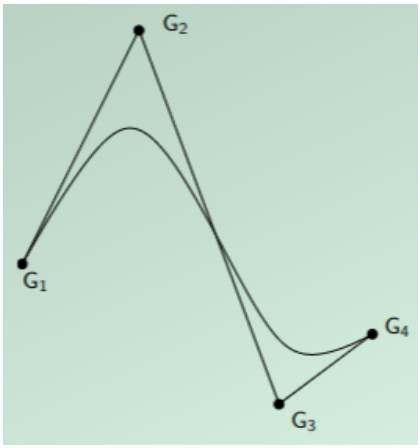


Figure 2: Bezier curves with startpoint, endpoint and controlpoints

$$G_B = \begin{bmatrix} G_1 & G_2 & G_3 & G_4 \end{bmatrix}$$

As with Hermite curves we have a function $f$, which is

$$f(t) = GMt$$

Because the two types of curves and be expressed as eachother, we can write

$$f(t) = GMt = G_H M_H t = G_B M_B t$$

We have the geometric metrices and the basis matrix $M_H$. If we take the vector from $G_1$ to $G_2$ in the Bezier curve, it must be equal to 3 times the tagent vector of the startpoint in the Hermite curve. The same is true for the vector fom $G_3$ to $G_4$ on the Bezier curve with tagent vector of the endpoint of the Hermite curve. This relation can be seen in the equation below:

$$G_H = \begin{bmatrix} G_1 & G_2 & G_3 & G_4 \end{bmatrix} = \begin{bmatrix} G_1 & G_4 & 3(G_2 - G_1) & 3(G_4 - G_3) \end{bmatrix}_B$$

We can write this on matrix form

$$G_H = \begin{bmatrix} G_1 & G_2 & G_3 & G_4 \end{bmatrix}_B \begin{bmatrix} 1 & 0 & -3 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & -3 \\ 0 & 1 & 0 & 3 \end{bmatrix}_{H \leftarrow B} = \begin{bmatrix} G_1 & G_4 & 3(G_2 - G_1) & 3(G_4 - G_3) \end{bmatrix}_B$$

We need to find the basis matrix for the Bezier curve, and we have a matrix taking us from Bezier to Hermite, which means we have

$$G_H = G_B M_{H \leftarrow B}$$

which gives

$$f(t) = G_h M_H t = (G_B M_{H \leftarrow B}) M_H t = G_B (M_{H \leftarrow B} M_H) t$$

$$M_B = M_{H \leftarrow B} M_H = \begin{bmatrix} 1 & 0 & -3 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & -3 \\ 0 & 1 & 0 & 3 \end{bmatrix}_{H \leftarrow B} \begin{bmatrix} 2 & -3 & 0 & 1 \\ -2 & 3 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}_H = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

and we have now found $G_B$ and $M_B$.

Now we can take a look at the 4 ways of draving Bezier curves, sampling, forward differences, sub divsion and flatness. Drawing a curve with sampling we take an intial point $t_0$ and calculate the points $t_0 + \delta$ to $t_0 + n\delta$. The

closer the point or the smaller the $\delta$ the nicer a curve. It is expensive to do sampling, since we have to calculate a 3rd degree polynomium for every point.

Forward differences is less costly to draw curves as Hermite and Bezier. If we multiply the $G$ and $M$ matrices we get the coefficients a, b, c

$$f(t) = at^3 + bt^2 + ct + d$$

In forward differences we do not have to do multiplication. We start with a point $f(t)$ and calculate $\Delta f(t)$. This means we can add our delta values to find the next point, though we have to do the following before hand

$$\Delta f(t) = f(t + \delta) - f(t)$$

$$f(t + \delta) = f(t) + f(t)$$

This can be rewritten as

$$\Delta f(t) = 3a\delta t^2 + (3a\delta^2 + 2b)t + a\delta^3 + b\delta^2 + c\delta$$

Now we have a 2nd degree polynomium, and if we do this 2 more times, we get a constant

$$\Delta^3 f(t) = 6a\delta^3$$

The curve can now be drawn for $f(t) = at^3 + bt^2 + ct + d,\ 0 \leq t \leq 1$ with forward differences with $n$ points where $\delta = \frac{1}{n}$

$$\begin{bmatrix} f_0 \\ \Delta f_0 \\ \Delta^2 f_0 \\ \Delta^3 f_0 \end{bmatrix} = \begin{bmatrix} f(0) \\ \Delta f(0) \\ \Delta^2 f(0) \\ \Delta^3 f(0) \end{bmatrix} = \begin{bmatrix} d \\ a\delta^3 + b\delta^2 + c\delta \\ 6a\delta^3 + 2b\delta^2 \\ 6a\delta^3 \end{bmatrix}$$

The new $f$ values is calculate by adding row $i$ and $i + 1$ together, which can be written as

$$\begin{bmatrix} f_{i+1} \\ \Delta f_{i+1} \\ \Delta^2 f_{i+1} \\ \Delta^3 f_{i+1} \end{bmatrix} = \begin{bmatrix} f_i \\ \Delta f_i \\ \Delta^2 f_i \\ \Delta^3 f_i \end{bmatrix} + \begin{bmatrix} \Delta f_i \\ \Delta^2 f_i \\ \Delta^3 f_0 \\ 0 \end{bmatrix} = \begin{bmatrix} f_i + \Delta f_i \\ \Delta f_i + \Delta^2 f_i \\ \Delta^2 f_i + \Delta^3 f_i \\ \Delta^3 f_0 \end{bmatrix}$$

Now we have 3 additions to get the next point $f_i + 1$ from the point $f_i$, which is how forward differences work.

When drawing a curve with sub division, we say the curve is a good approximation, if the curve is pretty flat. If the curve is not flat enough, we divide the curve into to curves with a new point roughly in the middle. Then the same thing is done for the new lines, until drawing is sufficiently close to the curve.
When doing sub division on Bezier curves, we draw points in between each of the 4 control points so we get $L_2$, $H$ and $R_3$. Then we draw points between the new points. We can express this with the following

$$
\begin{aligned}
L_1 &= G_1 = 8\frac{G_1}{8} \\
L_2 &= \frac{G_1}{2} + \frac{G_2}{2} = \frac{4G_1 + 4G_2}{8} \\
L_3 &= \frac{G_1}{4} + \frac{G_2}{2} + \frac{G_3}{4} = \frac{2G_1 + 4G_2 + 2G_3}{8} \\
L_4 &= \frac{G_1}{8} + 3\frac{G_2}{8} + 3\frac{G_3}{8} + \frac{G_4}{8} = G_1 + 3G_2 + 3G_3 + G_4 8 \\
R_1 &= G_4 = \frac{8G_4}{8} \\
R_2 &= \frac{G_3}{2} + \frac{G_4}{2} = \frac{4G_3 + 4G_4}{8} \\
R_3 &= \frac{G_2}{4} + \frac{G_3}{2} + \frac{G_4}{4} = \frac{2G_2 + 4G_3 + 2G_2}{8} \\
R_4 &= \frac{G_1}{8} + 3\frac{G_2}{8} + 3\frac{G_3}{8} + \frac{G_4}{8} = \frac{G_1 + 3G_2 + 3G_3 + G_4}{8}
\end{aligned}
$$

Inserted into matrices for the left and right side, we have

$$
DLB = \begin{bmatrix} 8 & 4 & 2 & 1 \\ 0 & 4 & 4 & 3 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad DRB = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & 2 & 0 & 0 \\ 3 & 4 & 4 & 0 \\ 1 & 2 & 4 & 8 \end{bmatrix}
$$

The control points can now be calculated recursivly. The control points for the left side by

$$
G_L^B = \begin{bmatrix} L_1 & L_2 & L_3 & L_4 \end{bmatrix} = \begin{bmatrix} G_1 & G_2 & G_3 & G_4 \end{bmatrix} \cdot DLB
$$

and the same thing for the rigth side, but with DRB instead.

When we have the Bezier curve we perform a flatness test, to make sure the

lines are close to what they are supposed to be. A flatness test consists of 7 thing we need to check is not true. These conditions are

$$
\begin{aligned}
||G_1G_4|| &= 0 \\
G_1G_2 \cdot u &< 0 \\
G_4G_3 \cdot u &> 0 \\
G_1G_2 \cdot u &> ||G_1G_4|| \\
G_4G_3 \cdot u &> ||G_1G_4|| \\
d_1 = ||G_1G_2(G_1G_2 \cdot u) \cdot u|| &> \epsilon \\
d_2 = ||G_4G_3(G_4G_3 \cdot u) \cdot u|| &> \epsilon
\end{aligned}
$$

If all these are false, the curve is flat enough.

# 3 Implementation

We have implemented the methods from the theory section. We started by implementing sampling. The first thing we did, was to initialize the $t$ value, $\delta = \frac{1}{n}$ and a vector $T$. Then we push the first point, so it gets drawn. Next thing is to do a loop, where we set $t_1 = t_1 + \delta$, $t_2 = t_1 * t_1$ and $t_3 = t_2 * t_1$. We add these to the vector $T$ and multiply thsi with the geometric matrix and the basis matrix, to get the vertex. After we have been through all the points, we push the endpoint aswell.

We start by initializing the 3 *delta* values when implementing forward differences. Next we initialize the Bezier coefficient by multiplying the geometric matrix with the basis matrix. Then we initialize the forward differences rows by initializing the $\Delta f$ values, and pushing the first point. Then we do a loop, where $F[1] = F[1] + F[2]$, $F[2] = F[2] + F[3]$ and $F[3] = F[3] + F[4]$. We push the $F[1]$ value every time and push the endpoint when the loop is finished.

We implemented sub division recursivly. We have a value $N$ which is how many times we should divide. If this value is 0 we just push the start and end point, if not we call SubDivide twice with

```
SubDivide(G * DLB, N - 1, Vertices);
SubDivide(G * DRB, N - 1, Vertices);
```

This way we find the control points until we do not have to divide any more.

For flatness we implemented the 7 checks from the theory. If all the checks are false, we return true. Else we use subdivision we looks like the first one expect we add an extra thresshold Flatness(G, epsilon).

# 4 Testing

Since all the methods draw the graphs with a very minimal difference, we assume the methods are implemented correctly. Pictures of the graphs are seen below.

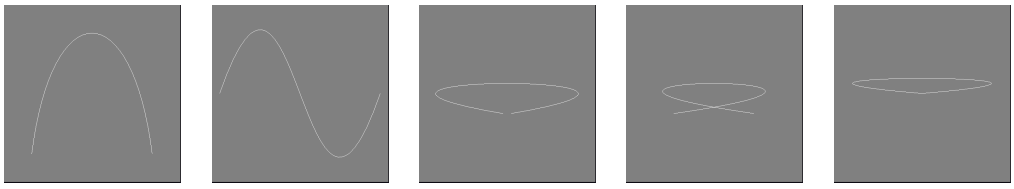

Figure 3: Test of the 5 shapes for sampling

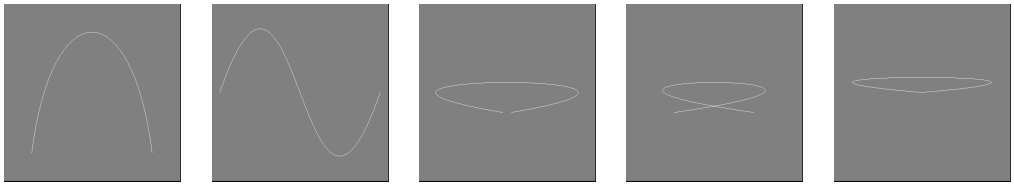

Figure 4: Test of the 5 shapes for forward differences

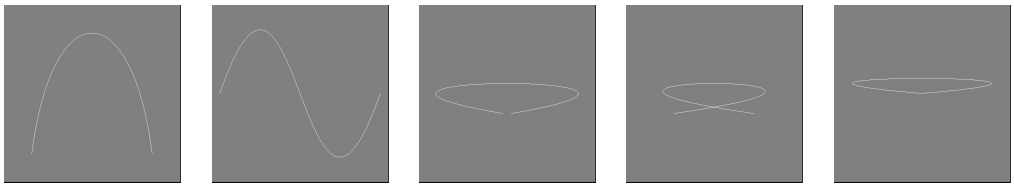

Figure 5: Test of the 5 shapes for sub division

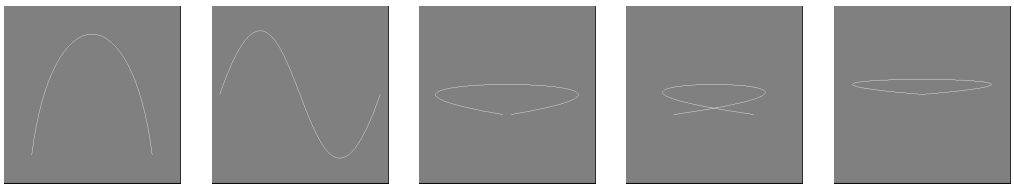

Figure 6: Test of the 5 shapes for flatness

# 5 Conclusion

We have now explained the theory behind and implemented algorithms for drawing Bezier curves. We have explained what a geometric and a basis matrix is and how we calculate them. We have also looked at some of the different ways it is possible to implement bezier curves, namely sampling, forward differences, sub division and flatness. Then we have tested that these implementations work as we wanted to, which is done as seen in the testing section.

# References