Reto 1 AN 2130

Nicolás Barragán, Gabriel de Souza, Pablo Santander

Friday 29th October, 2021

Part I
Reto representación de un solido

1 Introduccion y descripcion del problema

Para el Reto 1 de la clase de análisis numérico del segundo semestre del 2021 se planteó un problema sobre la representación de un sólid. Para este caso se buscar representar de la manera más similar un jarrón y sus curvas de nivel, recreandolo en su completitud.

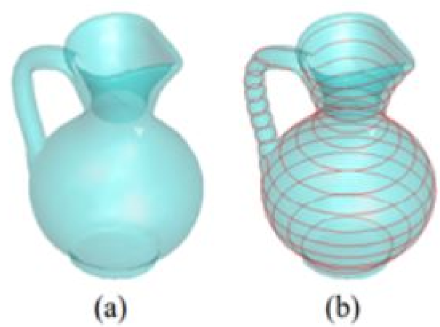


Figura 11: *a. Jarro. b. Curvas de nivel*

Para esto se utilzaron superficies de bezier y el método B-Splines en la herramienta Mathematica de Wolfram. Adicional a estas dos se utilizó Blender para modelar el jarrón inicial y obtener los puntos que conformaban al jarrón.

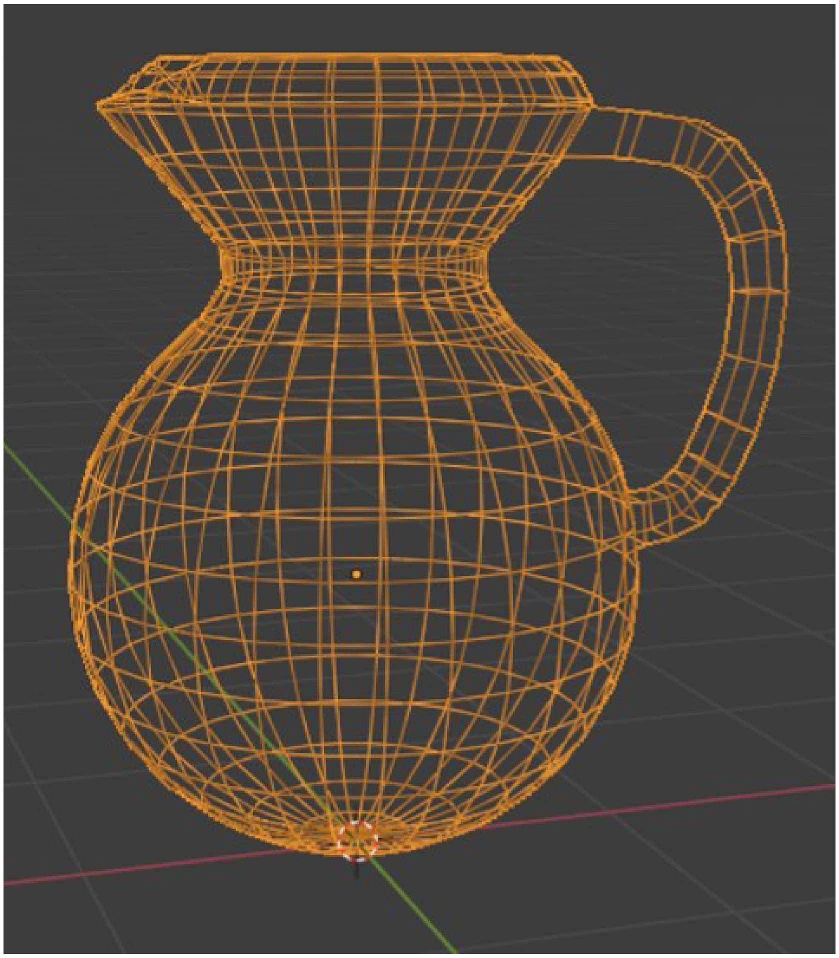
2 Métodos numéricos

1. B-Spline:
   B-Spline es un método que genera curvas a través de un set de puntos dados. Dentro de nuestra implementación se utilizó la función BSplineSurface que hace parte de Mathematica (Herramienta de WolframAlpha). Esta función recibe como parametro una matriz de puntos, en nuestro caso la matriz generada por Blender que importabamos a través de un comando para obtener los puntos como matriz. Una vez aplicada la función el jarrón era interpolado y con la ayuda de Graphics 3D que es también una función de Mathematica pudimos graficar y observar el modelo del jarrón interpolado.

2. Bezier:
   El método por curvas de Bezier consiste en representar el modelo a través de formar distintas curvas basadas en un conjunto de puntos de control. En nuestro caso el método se encontraba dentro de la función BezierFunction que hace parte de mathematica. Esta función recibe como parametro un set de puntos y puntos de control que componen al jarrón y a con base en estos se representaron las curvas para finalmente poder visualizar el jarrón con Graphics3D.

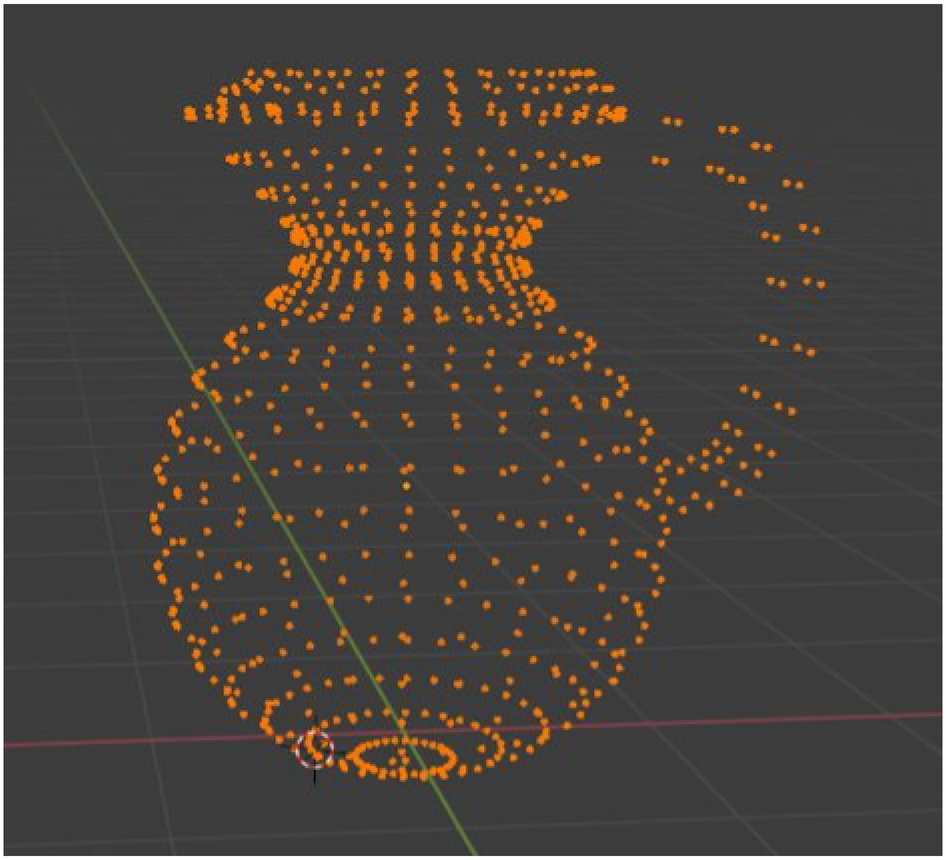
3 Metodología

Respecto a la metodologia realizada para la resolucion de este reto, iniciamos utilizando el software de Blender el cual nos permitio crear el Jarron objetivo, donde lo construimos tal y como se veia en la foto de las pautas del reto, con su respectiva boquilla y manija. El resultado del prototipo inicial fue el siguiente:



Como podemos observar, el Jarron esta compuesto por diferentes circulos que forman una esfera. La construccion de este se baso en un solo circulo colocada en la base del Jarron, el cual fue constantemente duplicado y ampliado uno encima de otro de tal manera que el jarron tomara forma. Ademas al hacerlo de esta manera tenemos una constancia en la figura y una division constante de esta.

Una vez finalizado el Jarron original, procedimos a utilizar una funcion dentro de Blender la cual se llama "Extrude Vertices". Esta funcion nos permite extraer todos los vertices o puntos que forman a la figura, la obtencion de esta informacion es clave para el proceso de la recreacion del jarron, debido a que podremos saber como esta formado en terminos de coordenadas en el plano x, y, z. A continuacion se muestra como quedo el jarron:



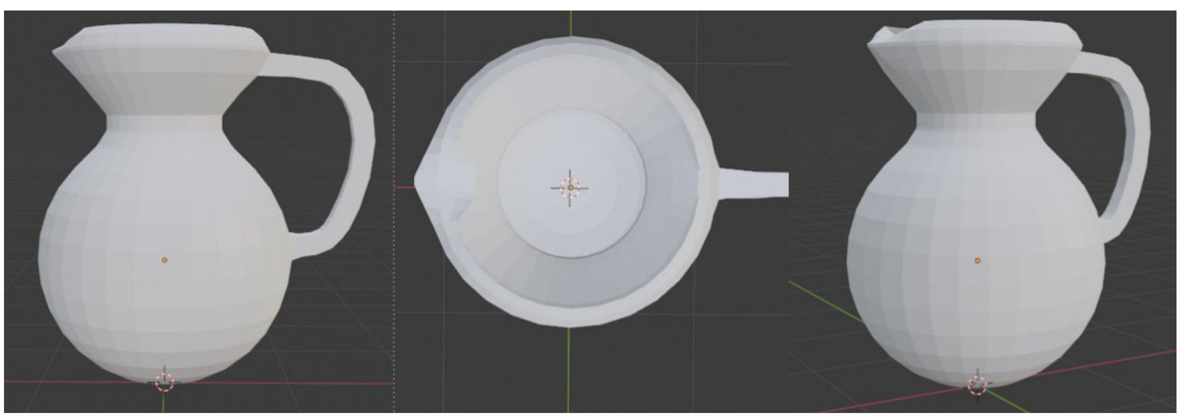
Al tener el jarron de esta manera, investigamos la mejor forma de poder extraer estos datos. Descubrimos entonces los archivos de tipo ".obj", estos son de gran utilidad debido que al exportar un objeto 3D con este tipo, resultara un archivo que contiene todos los puntos de contol que forman la figura, donde cada una de las lineas dentro del archivo empiezan por la letra "v", indicando el vertex o vertice, y consecutivamente las 3 coordenadas que forman ese punto. Una vez finalizado este proceso, utilizamos la plataforma Mathematica, en donde importamos nuestro archivo .obj. Utilizamos la funcion "VertexData" con el fin de extraer toda la data de los puntos dentro del archivo. Mostramos entonces la aplicacion de esto:

```
Import["PuntosJarra.obj", "VertexData"]

{{0.36914, 2.03783, -0.33993}, {0.293265, 2.03783, -0.408248}, ··· 889 ···,

large output | show less | show more | show all | set size limit...
```

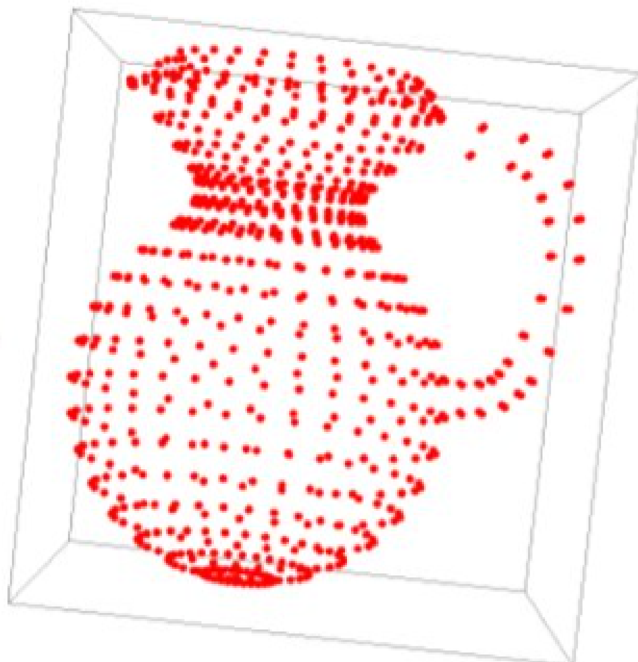
Se conoce entonces que el Jarron esta conformado por un total de 889 puntos de control. Es normal cuestionarse en la razon de por que son esa cantidad de puntos, y no mas o menos. La razon principalmente se basa en el funcionamineto de Blender para representar figuras en su plazo x, y, z. En nuestro caso, cada uno de los circulos usados estaba formada por un total de 30 puntos individuales en Blender, y al usar multiples circulos resulta entendible que sean tantos puntos resultantes.
Adicionalmente, tambien realizamos un proceso de solidificacion y suavizado de nuestro Jarron original, en donde nuestro objetivo fue representarlo de una manera mas realista, en donde no se vieran las lineas amarillas y los circulos que lo formaban. El resultado fue este:



Gracias a la obtencion de puntos en la herramienta Mathematica, como se menciono anteriormente, pudimos entonces aplicar nuestro primer metodo numerico para poder reconstruir nuestro jarron original. Este metodo fue el de BSplineSurface dentro de la herramienta, el cual requiere como parametro un arreglo de puntos de control al cual llamamos "Puntos", responsable de almacenar los 889 puntos obtenidos. Usamos la variable "F" para llamar la funcion y guardar su resultado, luego usamos la funcion Show para poder mostrar la grafica generada, representada por puntos de color rojo tamaño mediano.

```
F = Graphics3D[BSplineSurface[Puntos]];
F = Show[Graphics3D[{PointSize[Medium], Red, Map[Point, Puntos]}], F ]
```

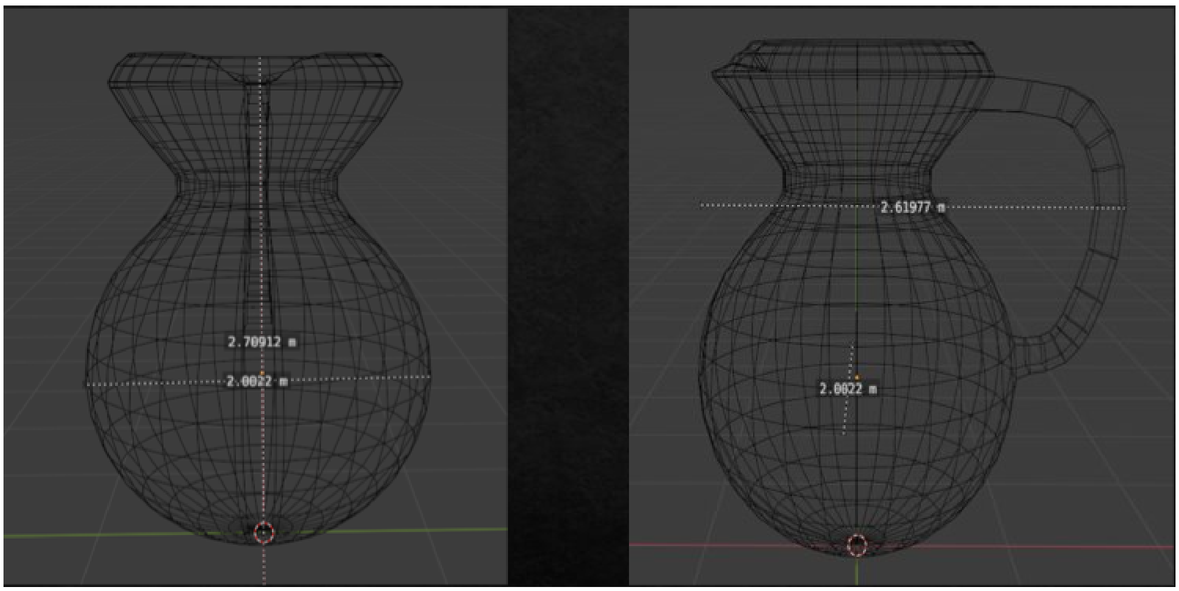
Out[14]=



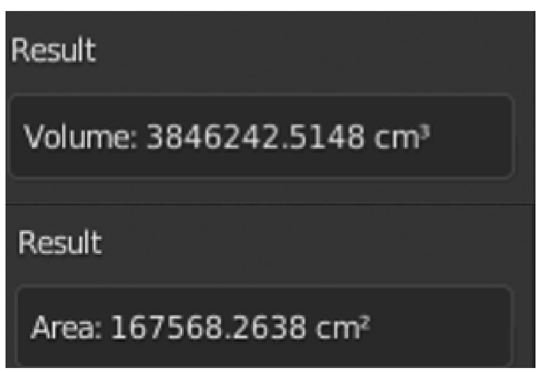
En terminos visuales se puede concluir que el jarron fue reconstruido de una manera bastante satisfactoria mediante este metodo, donde se pueden ver todos los puntos que conforman la figura asi como en la original, junto a su caracteristica boquilla y manija.

4 Mediciones

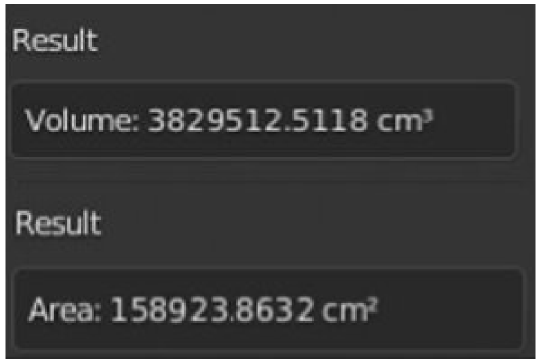
A pesar de obtener una grafica relativamente igual a la original, sabemos que la visualizacion de resultados no es suficiente, y menos en temas de Analisis Numerico, por ende decidimos utilizar la herramienta "Medir" en Blender la cual como su nombre nos dice, permite medir el tamaño del objeto desde cualquier punto hasta cualquier otro, como una regla. Adicionalmente utilizamos la herramienta "3D Tooltip Toolbox", la cual es un add-on de Blender, en donde al seleccionar un objeto esta nos mostrara su calculo respectivo del area y volumen que lo componen. De esta manera comparamos las mediciones de altura, ancho, volumen y area del jarron original versus el jarron interpolado resultante. A continuacion algunas imagenes de los resultados obtenidos en Metros:



Volumen y area en centimetros del Jarron original:



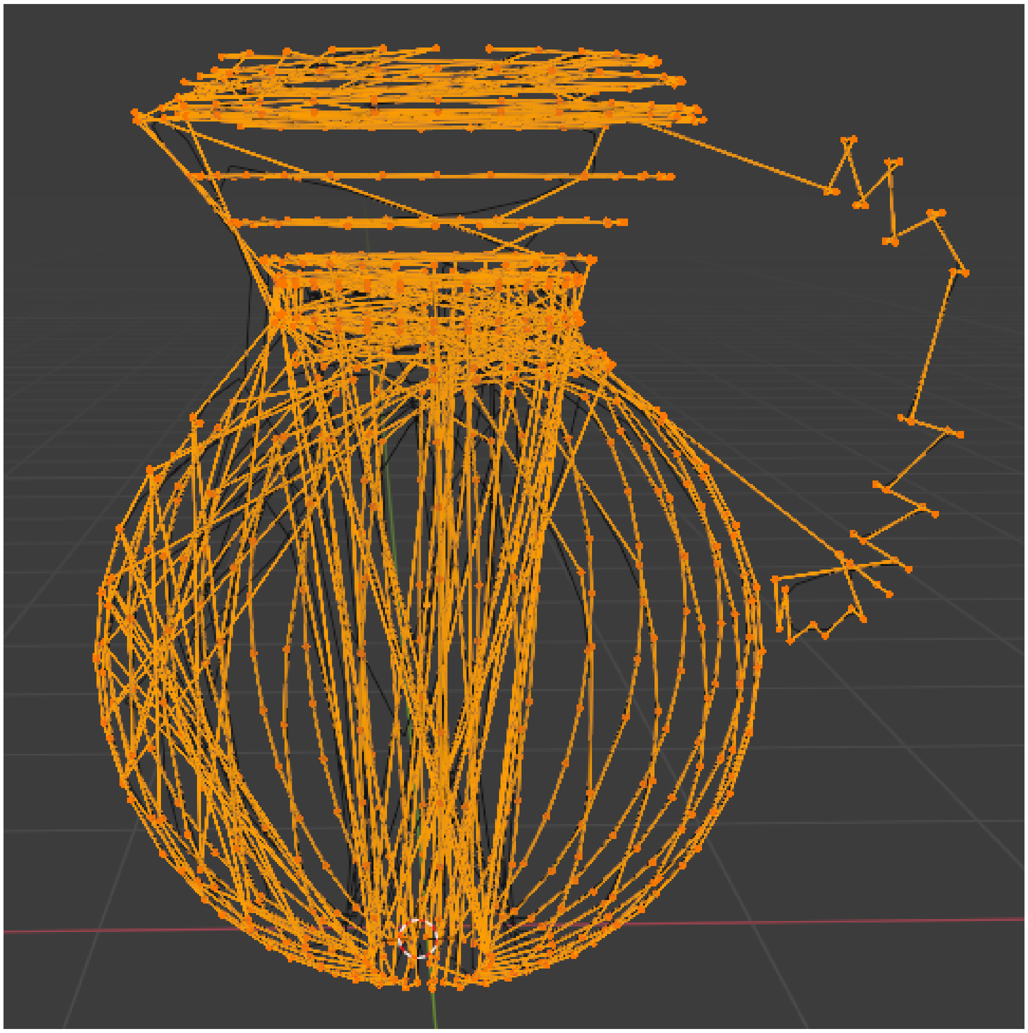
Volumen y area en centimetros del Jarron interpolado:



Los resultados obtenidos decidimos ponerlos en una tabla y hacer algunos calculos, lo cual se expondra en la seccion de Error dentro del documento.

5 Pruebas

Una vez teniamos los diferentes jarrones y resultados, quisimos realizar algunas pruebas para ver que sucedia con los puntos o el jarron. Una de las pruebas fue modificar la anchura del jarron dentro de Blender, de tal manera que todos los puntos que conformaban al jarron se expandieran mas y mas, lo cual genero un incremento en la distancia entre cada punto. Esto no genero ningun cambio en la cantidad de puntos que existian, simplemente se generaron cambios de coordenadas en todos los puntos, y la figura por obvias razones se vio mucho mas ancha. La siguiente prueba que realizamos fue la eliminacion de puntos del jarron, inicialmente habiamos eliminado dos curvas que conformaban la figura, por un total de 64 puntos, en donde repetimos el proceso y reconstruimos la figura en Mathematica. La grafica se pudo reconstruir con los 64 puntos faltantes, en donde claramente se veia la falta de las dos curvas. Sin embargo, la profesora Eddy Herrera nos recomendo que la eliminacion de puntos la realizaramos en distintas zonas del jarron, para ver que pasaba. De tal manera que decidimos eliminar 120 puntos a lo largo del objeto. Lo que pudimos observar en Blender fue que al utilizar la funcion "Bridge edge loops" la cual basicamente une los puntos para poder darle forma a la figura, se generaba un trazado bastante peculiar mostrado a continuacion:



Esto no sucedia en la figura original, en donde se mantenian todos sus puntos sin modificaciones. Parece que al eliminar puntos en diferentes partes de la figura, causa que el algoritmo interno de union haga su mejor esfuerzo al unir puntos con otros en su proximidad, teniendo siempre en cuenta la distancia entre ellos.

6 Error

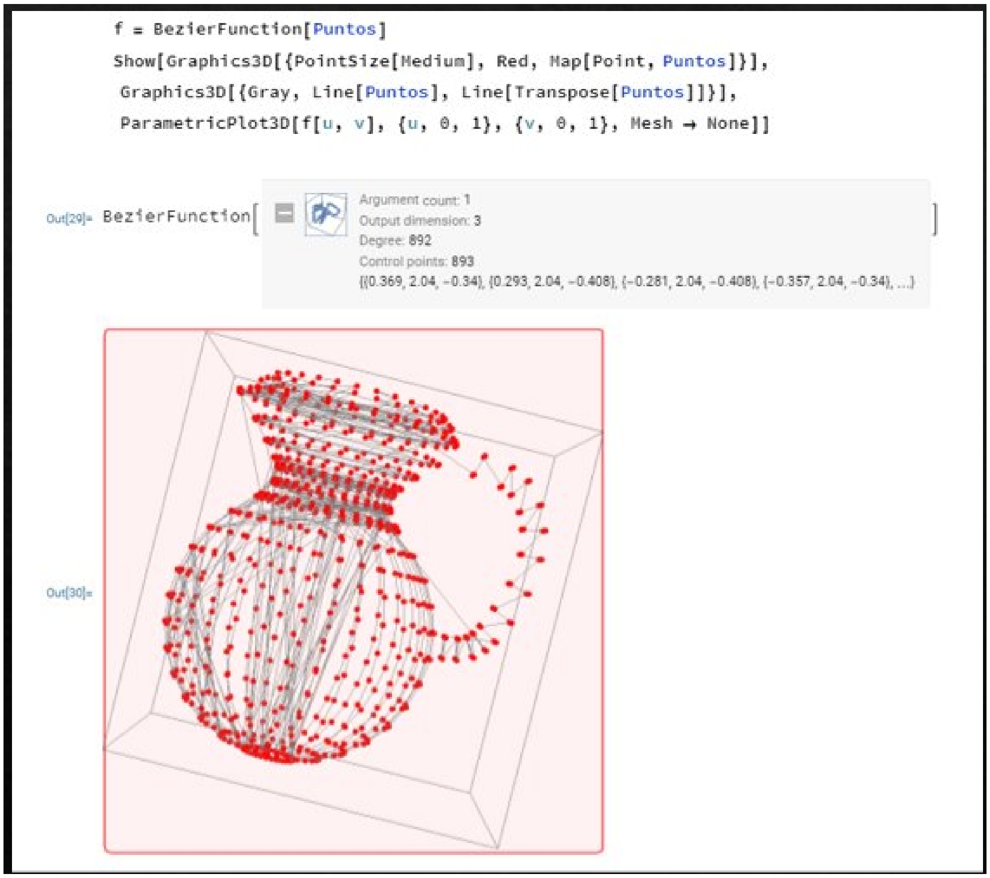
Para estarnos seguro de que la reconstrucción de la jarra se generó de forma correcta, son necesarios los errores, ya que esos nos permiten comparar las medidas de la jarra original y la jarra interpolada. El resultado genero la siguiente tabla:

Medidas	Jarrón Original	Jarrón Interpolado	Error Absoluto	Error relativo
Alto (m)	2.7091	2.6743	0.0348	1.29%
Ancho(m)	2.6198	2.4685	0.1513	5.77%
Area (m^2)	16.7568	15.8923	0.8645	5.16%
Volumen (m^3)	3.8462	3.8295	0.0167	0.44%

Las cifras significativas fueron acortadas y usadas de la casa del metro para la fácil comparación. Fue decidido escoger las cifras que mas hacen sentido para medir una jarra es decir hasta el mm. Se tuvo en cuenta el error absoluto el cual nos indica la desviación del original y el error relativo que nos pone en porcentaje de las medidas del original. Podemos ver en la tabla que el error mas alto fue de 5.77% el cual sigue siendo un error bien bajo, y se puede decir que la reconstrucción esta precisa.

7 Validacion de resultados

Como mencionado en el enunciado del problema, existen varias formas para realizar la reconstrucción del jarrón, como los métodos de BezierFunction o BSplineSurface. Para enriquecer nuestro proyecto y estar seguro de nuestros resultados, fue realizado otra reconstrucción del jarrón original usando la función de Bézier. Este nos dio un total de 3 puntos de control más que el original a una suma de 893. La siguiente imagen muestra este resultado:



Al verificar las medidas, este nos arrojó la siguiente tabla:

Medidas	Jarrón original	Jarrón (Bezier)	Error Absoluto	Error relativo
Alto (m)	2.7091	2.7542	0.0451	2%
Ancho (m)	2.6198	2.6754	0.0556	2%
Area (m^2)	16.7568	17.3816	0.6248	4%
Volumen (m^3)	3.8462	4.1174	0.2712	7%

Como se puede observar el jarrón reconstruido con la función de Bézier es un poco más ancho y alto que el original. El error relativo de sus medidas aún sigue siendo muy bajos con un máximo error de 7% en el volumen. Luego podemos concluir que el jarrón sigue siendo muy preciso al original.

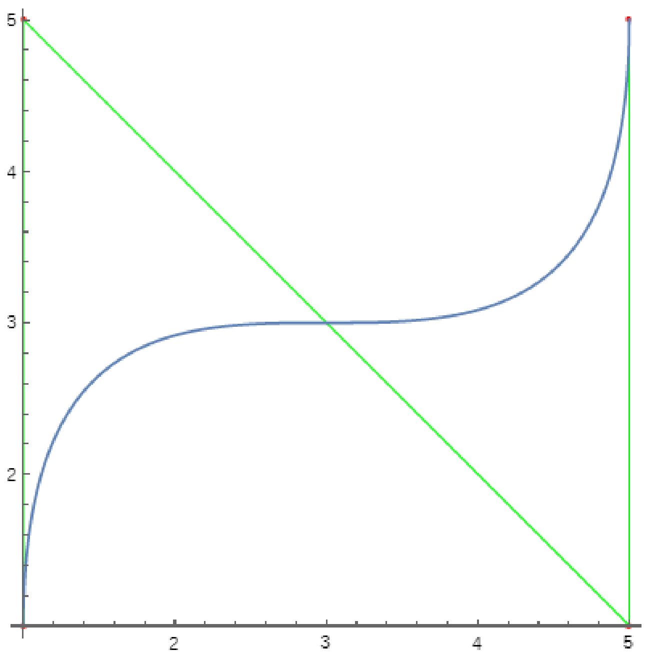
8 Representacion de letra

Por ultimo, se pedia hacer la representacion de alguna de las letras que componia el nombre o apellido de alguno de los integrantes del grupo. Decidimos usar la letra N de Nicolas, para esto se dibujo la letra N en un papel en donde en el plano cartesiano se extrajeron los cuatro puntos que la conformaban, siendo dos de estos los puntos de control. De esta manera usamos la funcion BSpline para recrear la figura y mostrar su curva de Bezier respectiva, el resultado se muestra a continuacion:

```
In[4]:= pts = {{1, 1}, {1, 5}, {5, 1}, {5, 5}};
        f = BSplineFunction[pts]
        Show[Graphics[{Red, Point[pts], Green, Line[pts]}, Axes → True],
          ParametricPlot[f[t], {t, 0, 1}]]

Out[5]= BSplineFunction[ Argument count: 1
                         Output dimension: 2 ]
```

Out[6]=



9 Referencias

1. BSplineSurface—Wolfram Language Documentation. (2020). Retrieved 26 october 2021, from https://reference.wolfram.com/language/ref/BSplineSurface.html

2. B-spline Surfaces: Construction. (2020). Retrieved 26 october 2021, From https://pages.mtu.edu/shene/COURSES/cs3621/NOTES/surface/bspline-construct.html

3. Mathematica. (2020). Retrieved 26 october 2021, From https://es.wikipedia.org/wiki/Mathematica

4. BezierFunction—Wolfram Language Documentation. (n.d.). Wolfram. Retrieved October 29, 2021, from https://reference.wolfram.com/language/ref/BezierFunction.html

5. How to integrate a BÃzier function? | Newbedev. (n.d.). NewbeDEV. Retrieved October 28, 2021, from https://newbedev.com/how-to-integrate-a-bezier-function