Un modelo para visualizar objetos en 4D con el Mathematica

A model to visualize objects in 4D with Mathematica

Ricardo Velezmoro y Robert Ipanaqué

Universidad Nacional de Piura, Urb. Miraflores s/n, Castilla, Piura, Perú.

DOI: https://doi.org/10.33017/RevECIPeru2014.0008/

Resumen

Una variedad de técnicas de gráficos por computadora han permitido la visualización de objetos, que existen en dimensiones más altas, en una pantalla 2D. En este artículo se propone un nuevo modelo a partir de la extensión de una técnica útil en la visualización de objetos en 3D en una pantalla 2D para realizar algo similar con objetos en 4D. Dicha técnica se basa en la definición de una inmersión, en primera instancia, del espacio tridimensional en el espacio bidimensional que luego se toma como referencia para definir otra inmersión, que constituye el modelo propuesto en este artículo, del espacio tetra dimensional en el espacio tridimensional. En teoría la visualización de objetos en 4D en una pantalla 2D se consigue mediante la composición de las dos inmersiones mencionadas, pero en la práctica se aprovechan los comandos incorporados en el sistema de cálculo simbólico *Mathematica* para tal fin.

Descriptores: *objetos 4D, modelo, inmersión*

Abstract

A variety of computer graphics techniques have enabled the display of objects, which exist in higher dimensions, on a 2D screen. In this paper a new model from the extension of a technique useful in visualizing 3D objects on a 2D screen to make something similar with 4D objects is proposed. This technique is based on the definition of a immersion, in the first instance, from the three-dimensional space in two-dimensional space which is then taken as a reference to define another immersion, which is the model proposed in this paper, from the fourdimensional space in three dimensional space. Theoretically the visualization of objects in 4D on a 2D screen is achieved by the composition of the two immersions mentioned, but in practice the incorporated commands into the computer algebra system *Mathematica* for this purpose are utilized.

Keywords: *objects 4D, model, immersion*

1. Introducción

La tarea de visualización de estructuras 4D ha sido explorada por Noll [1]. Noll se vio limitado en su exploración por la tecnología de ese tiempo; su método consistía en la generación de imágenes a través de plotter para luego transferir cada dibujo en una placa. Las películas que produjo brindaron una gran cantidad de información acerca de la estructura de varios objetos tetra dimensionales. Sin embargo, la falta de interacción era un obstáculo significativo.

En 1978, David Brisson [2] presentó híper estereogramas de marcos alámbricos 4D. Estos son híper estereogramas no convencionales en el sentido que el espectador debe girar los híper estereogramas con el fin de resolver el segundo grado de paralaje de las vista 4D. Estos híper estereogramas son muy

difíciles de ver, pero sí ofrecen otro método de la comprensión de las estructuras tetra dimensionales.

A inicios de 1980, Thomas Banchoff (que está muy involucrado en la visualización del espacio tetra dimensional) simuló "cáscaras" de híper esferas ([3] y [4]) que se tradujeron en imágenes hermosas de su rotación en el espacio tetra dimensional.

Varias personas han modelado objetos tetra dimensionales mediante la producción de los cortes tridimensionales del objeto; esto se detalla ampliamente en [4].

En este artículo se propone un nuevo modelo para visualizar objetos en 4D. Para dar a entender este modelo se parte de la definición de una inmersión ϕ: $\mathbb{R}^3 \to \mathbb{R}^2$ (modelo para graficar objetos en 3D) de tal manera que si $\boldsymbol{p}$ está en 3D entonces es posible visualizar $\phi(\boldsymbol{p})$ en una pantalla 2D, como si se tratase de un gráfico en un plano bidimensional. Obviamente, que al aplicar la inmersión ϕ a una curva o superficie (conjuntos de puntos) en 3D, se visualizará la imagen de dicha curva o superficie en una pantalla 2D. En esta parte se visualizan algunos gráficos explicativos a los que no se les puede cambiar el punto de vista en tiempo real y esto nos tiene sin preocupación pues consideramos que no es necesario desarrollar complejos algoritmos para tales cambios de vista, ya que esta técnica está ampliamente desarrollada en la actualidad y los sistemas de cálculo simbólico como el *Mathematica* [5], incorporan comandos específicos (**Graphics3D**, **ParametricPlot3D**, **Plot3D**, etc.) para obtener visualizaciones, en una pantalla 2D, de objetos en 3D. Una vez explicada la idea de partida se procede a definir otra inmersión ψ: $\mathbb{R}^4 \to \mathbb{R}^3$ (modelo para graficar objetos en 4D) de modo que si $\boldsymbol{p}$ está en 4D entonces es posible visualizar $\psi(\boldsymbol{p})$ en un ambiente 3D, como si tratase de un holograma proyectado en el espacio tridimensional (aquí también es aplicable a conjuntos de puntos como curvas, superficies e híper superficies). Sin embargo, al no tener acceso a herramientas de holografía, para las visualizaciones en 3D, se opta por hacer uso de la inmersión compuesta $\phi \circ \psi$: $\mathbb{R}^4 \to \mathbb{R}^2$. De esta manera se visualizan objetos 4D en una pantalla 2D. En realidad, esto es en teoría, ya que en la práctica, tal como se indicó para el caso de visualizar objetos en 3D en una pantalla 2D, se aprovechan los comandos incorporados del *Mathematica* para, con base en éstos, definir nuevos comandos que permitan visualizar objetos en 4D en una pantalla 2D con cambios de vista en tiempo real.

2. Modelo para graficar objetos en 3D

Definición. Una aplicación diferenciable

$$\varphi: M \to N,$$

se dice que es una inmersión si $d\varphi_{\boldsymbol{p}}$ es inyectiva para todo $\boldsymbol{p}$ en M.

En la primera parte de un curso de pregrado de matemática III es común que se aborde las técnicas para representar puntos, curvas y superficies en 3D en una hoja de papel (pantalla 2D). También es común representar los ejes coordenados x, y y z, del sistema cartesiano tridimensional, tal como se muestra en la figura 1.

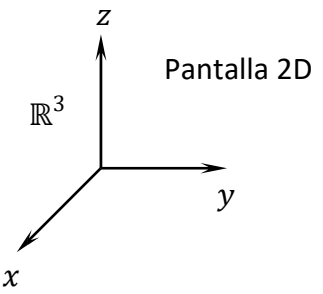

Figura 1: Representación común de los ejes coordenados del sistema cartesiano tridimensional en un plano (pantalla 2D).

Consideremos los vectores unitarios:

$$e_1 = (-\tfrac{1}{\sqrt{2}}, \tfrac{1}{\sqrt{2}}),\ e_2 = (1,0) \text{ y } e_3 = (0,1),$$

en la dirección de los ejes x, y y z (Fig. 2). Usaremos estos vectores para definir la inmersión ϕ: $\mathbb{R}^3 \to \mathbb{R}^2$ tal que

$\phi(\boldsymbol{p}) = p_1 e_1 + p_2 e_2 + p_3 e_3$, para todo $\boldsymbol{p} \in \mathbb{R}^3$. Es decir,

$\phi(\boldsymbol{p}) = \left(-\frac{p_1}{\sqrt{2}} + p_2, -\frac{p_1}{\sqrt{2}} + p_3\right)$, para todo $\boldsymbol{p} \in \mathbb{R}^3$.

Esta inmersión constituye el modelo para graficar objetos en 3D en una pantalla 2D. Para visualizar los resultados en forma rápida implementaremos el comando **phi** en el Mathematica.

$$\{\boldsymbol{e1}, \boldsymbol{e2}, \boldsymbol{e3}\} := \{\{-1,-1\}/\boldsymbol{Sqrt}[2], \{1,0\}, \{0,1\}\}$$
$$\boldsymbol{phi}[\{\boldsymbol{p1\_}, \boldsymbol{p2\_}, \boldsymbol{p3\_}\}] := \boldsymbol{p1e1} + \boldsymbol{p2e2} + \boldsymbol{p3e3}$$

z

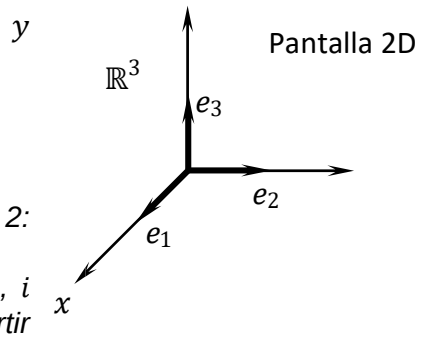

Figura 2: Vectores unitarios, e_i, i = 1,2,3, a partir de los cuales se define la inmersión ϕ.

Ahora, por ejemplo, visualicemos la gráfica de la hélice α: $\mathbb{R} \to \mathbb{R}^3$, tal que

$\alpha(t) = (\cos t\,, \operatorname{sen} t, t/8\,)$, $0 < t < 7\pi$. Asumiendo que sólo contamos con comandos para visualizar objetos 2D, usaremos el comando incorporado en el *Mathematica* **ParametricPlot**.

$\boldsymbol{\alpha[t\_] := \{Cos[t], Sin[t], t/8\}}$
$\boldsymbol{ParametricPlot[phi[\alpha[t]], \{t, 0, 7\pi\}]}$ (Fig. 3)

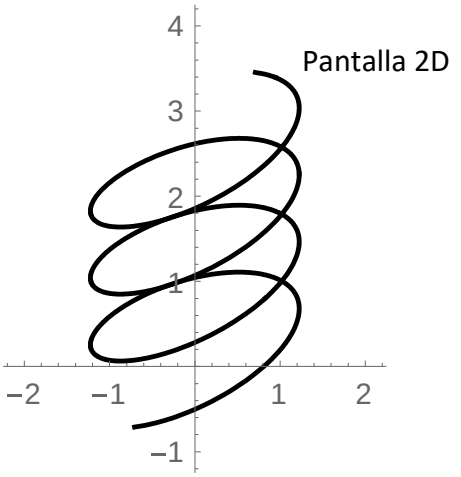

Figura 3: Visualización de la gráfica de la hélice, α en 3D, en una pantalla 2D.

Seguidamente, vamos a visualizar la gráfica de las curvas u −paramétricas y v −paramétricas de la esfera unitaria x: $\mathbb{R}^2 \to \mathbb{R}^3$.

$\boldsymbol{x[u, v] := \{Cos[v]Cos[u], Cos[v]Sin[u], Sin[v]\}}$
$\boldsymbol{upar}$
$\boldsymbol{= Table[phi[x[u, v]], \{v, -\pi/2, \pi/2, \pi/8\}], \{u, 0, 2\pi\}]}$
$\boldsymbol{= ParametricPlot[}$

$\boldsymbol{vpar = ParametricPlot[}$
$\boldsymbol{Table[phi[x[u, v]], \{u, 0, 2\pi, \pi/8\}], \{v, -\pi/2, \pi/2\}]}$
$\boldsymbol{Show[upar, vpar]}$ (Fig. 4)

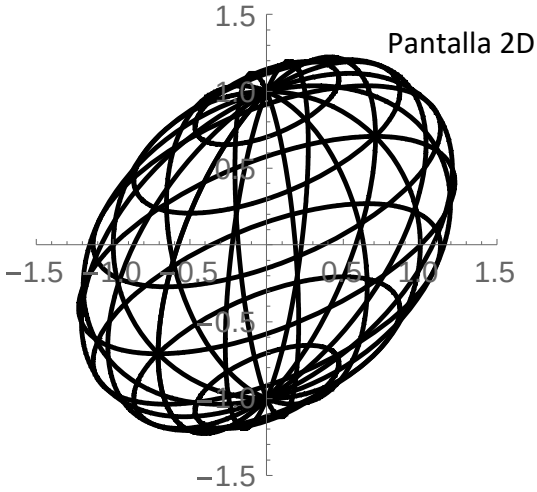

Figura 4: Visualización de las gráficas de las curvas u y v paramétricas de la esfera unitaria, x en 3D, en una pantalla 2D.

Lo anterior nos da una idea clara del funcionamiento del modelo para visualizar gráficos de objetos en 3D en una pantalla 2D, aunque como ya se mencionó anteriormente a las gráficas visualizadas no se les puede cambiar el punto de vista en tiempo real. Tal vez podrían emplearse otras inmersiones para simular otros cambios de puntos de vista y finalmente implementar un programa que abarque a todas las posibles inmersiones para simular el cambio de punto de vista en tiempo real; aunque después surgirían otros problemas como los temas de: las caras ocultas, la iluminación (en el caso de superficies), etc. [6]. Pero, todo esto ya ha sido estudiado y solucionado, es por esto que los sistemas de cálculo simbólico (entre otros softwares) como, por ejemplo, el *Mathematica* cuyos comandos a pesar de devolvernos como salida gráficos que están sobre una pantalla 2D, nos dan la sensación que estamos visualizándolos en 3D.

$\boldsymbol{ParametricPlot3D[x[u,v], \{u,0, 2\pi\},}$
$\boldsymbol{\{v, -\pi/2,\pi/2\}]}$ (Fig. 5)

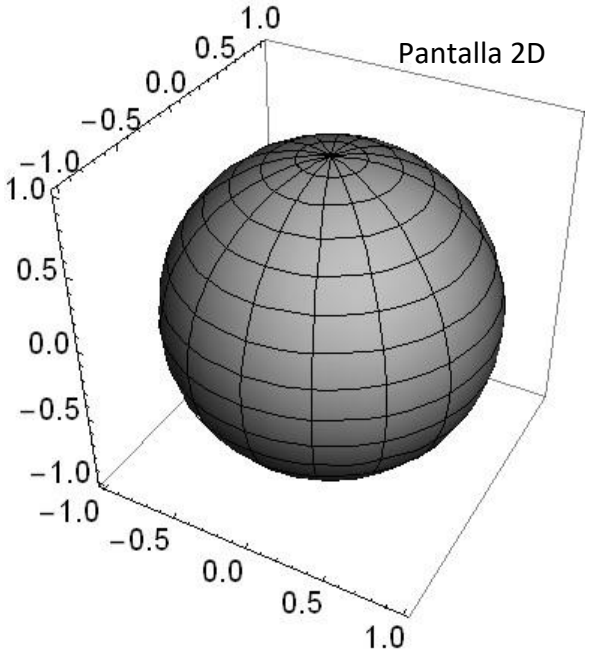

Figura 5: Visualización de la gráfica de la esfera unitaria, x en 3D, en una pantalla 2D.

3. Modelo para graficar objetos en 4D

Teniendo en cuenta que los comandos incorporados en el *Mathematica*, que permiten visualizar objetos en 3D en una pantalla 2D, son altamente eficientes, supondremos que contamos con proyector de hologramas en el espacio tridimensional y procederemos a construir nuestro modelo siguiendo un proceso similar al de la sección previa.
Consideremos un cubo cuyos lados tienen longitud igual a dos unidades, en el espacio tridimensional, cuyo centro de gravedad es el origen de coordenadas del sistema cartesiano tridimensional. Ubiquemos a continuación, en el cubo, cuatro ejes coordenados: x, y, z y w, dispuestos en la forma que se muestra en la figura 6.

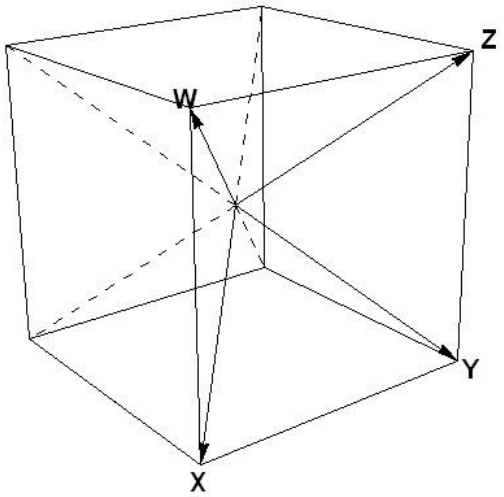

Figura 6: Disposición de los ejes coordenados x, y, z y w en el espacio tridimensional.

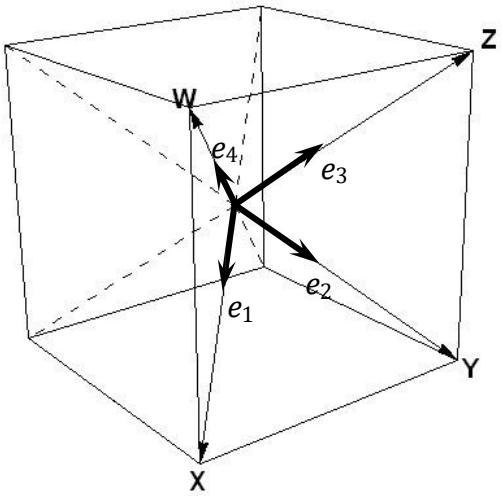

Figura 7: Vectores unitarios, e_i, $i = 1,2,3,4$, a partir de los cuales se define la inmersión ψ.

Luego, tomemos los vectores unitarios:

$$e_1 = \left(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right),\ e_2 = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right),$$
$$e_3 = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right) \text{ y } e_4 = \left(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right),$$

en la dirección de los ejes x, y, z y w (Fig. 7).

A partir de estos vectores se definirá una inmersión ψ que por sí misma constituirá el modelo para graficar objetos en 4D en un ambiente 3D.

4. Resultados y discusión

Definamos la inmersión $\psi: \mathbb{R}^4 \to \mathbb{R}^3$ tal que $\psi(\boldsymbol{p}) = p_1e_1 + p_2e_2 + p_3e_3 + p_4e_4$, para todo $\boldsymbol{p} \in \mathbb{R}^4$.

La inmersión ψ puede escribirse en la forma

$$\psi(\boldsymbol{p}) = \frac{1}{\sqrt{3}}(p_1 + p_2 + p_3 + p_4, -p_1 + p_2 + p_3 - p_4, -p_1 - p_2 + p_3 + p_4)$$ para todo $\boldsymbol{p} \in \mathbb{R}^4$.

Implementaremos el comando **psi**, en el *Mathematica*, para visualizar los resultados en forma rápida.

```
{e1, e2,e3,e4} :=
    {{1, -1,-1}/Sqrt[3] ,{1, 1, -1}/Sqrt[3],
     {1,1, 1}/Sqrt[3] , {1,-1,1}/Sqrt[3]}
psi[{p1_,p2_, p3_, p4_}] :=
p1e1 + p2e2 + p3e3 + p4e4
```

En primera instancia, visualizaremos la gráfica de la curva $\beta: \mathbb{R} \to \mathbb{R}^4$, tal que
$\beta(t) = \left(\cos\left(\frac{s}{\sqrt{5}}\right), \operatorname{sen}\left(\frac{s}{\sqrt{5}}\right), \cos\left(2\frac{s}{\sqrt{5}}\right), \operatorname{sen}\left(2\frac{s}{\sqrt{5}}\right)\right)$,
$0 < t < 5\pi$.

Según lo anotado al inicio de la sección previa, es decir, que se cuenta con un proyector de hologramas en el espacio tridimensional, utilizaremos el comando incorporado en el *Mathematica* **ParametricPlot3D**.

```
β[t_] := {Cos[s/Sqrt[5]],Sin[s/Sqrt[5]],
          Cos[2s/Sqrt[5]],  Sin[2s/Sqrt[5]]}
ParametricPlot3D[psi[β[t]],{t,0, 5π}]   (Fig. 8)
```

Seguidamente, visualizaremos la gráfica de la superficie $y: \mathbb{R}^2 \to \mathbb{R}^4$, tal que $y(u, v) = \{(4 + 2\operatorname{sen}[u])\cos[u],$
$(4 + 2\operatorname{sen}[v])\operatorname{sen}[u], 2\cos[v]\cos[u/2],$
$2\cos[v]\operatorname{sen}[u/2]\}$, $0 < u, v < 2\pi$.

Esta es una superficie no orientable y es conocida como el toro de Klein, o más comúnmente como la botella de Klein.

Para la visualización de la superficie y, de acuerdo con lo establecido previamente, nuevamente se usará el comando **ParametricPlot3D**.

```
y[u_, v_] := {(4 + 2Sin[u])Cos[u],
              (4 + 2Sin[v])Sin[u],2Cos[v]Cos[u/2],
                   2Cos[v]Sin[u/2]}
ParametricPlot3D[psi[y[u,v]],{u, 0, 2π}, {v, 0,2π}]
(Fig. 9)
```

Ahora, visualizaremos la gráfica de la híper superficie $h: \mathbb{R}^3 \to \mathbb{R}^4$ tal que $h(u, v, w) = (u, v, w, u^2 + v^2 + w^2)$, $-1 < u, v, w < 1$.

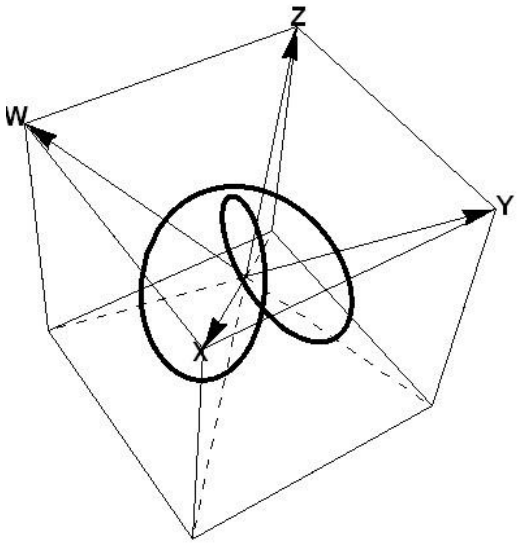

Figura 8: Visualización de la gráfica de la curva, β en 4D, en un "ambiente 3D".

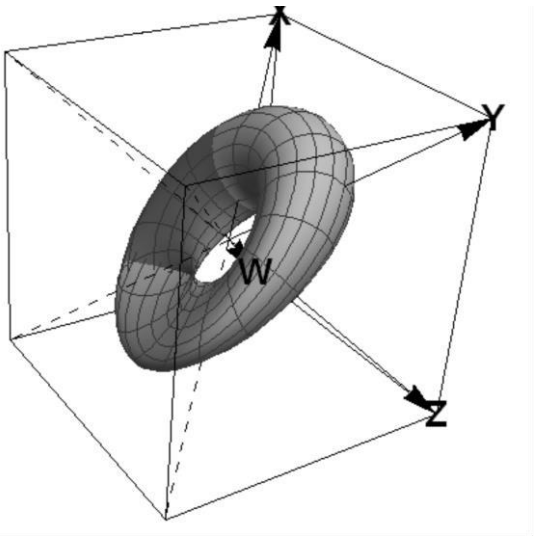

Figura 9: Visualización de la gráfica de la superficie, y en 4D, en un "ambiente 3D".

En este caso ya no puede utilizarse el comando **ParametricPlot3D**; así que, basándonos en el código, concerniente a un comando similar, propuesto por Roman Maeder [7], hemos implementado el nuevo comando **ParametricPlot4D**, el cual utilizaremos de ahora en adelante.

```
ParametricPlot4D[{u, v, w,u^2 + v^2 + w^2},  {u, -1,1,
    0.15},{v, -1,1, 0.15},{w,-1,1, 0.15}]
```

(Fig. 10)

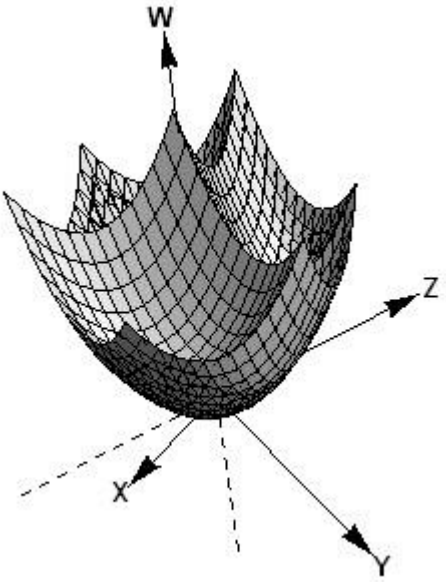

Figura 10: Visualización de la gráfica de la híper superficie, h en 4D, en un "ambiente 3D".

Ahora, definamos la híper superficie $g: \mathbb{R}^3 \to \mathbb{R}^4$, tal que $g(u, v, w) = (u, v, w, u^2 + v^2 - w^2)$, $-1 < u, v, w < 1$.

Para visualizar la gráfica volveremos a hacer uso del nuevo comando **ParamericPlot4D**.

$$\boldsymbol{ParametricPlot4D}[\{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{u^2 + v^2 - w^2}\},\ \{\boldsymbol{u}, \boldsymbol{-1}, \boldsymbol{1}, \boldsymbol{0.15}\}, \{\boldsymbol{v}, \boldsymbol{-1}, \boldsymbol{1}, \boldsymbol{0.15}\}, \{\boldsymbol{w}, \boldsymbol{-1}, \boldsymbol{1}, \boldsymbol{0.15}\}]$$

(Fig. 11)

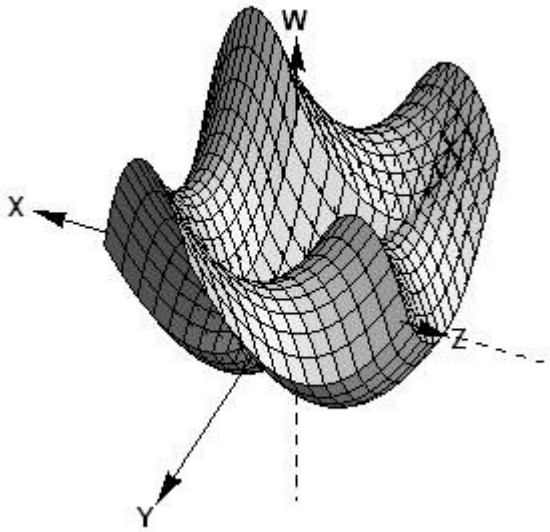

Figura 11: Visualización de la gráfica de la híper superficie, g en 4D, en un "ambiente 3D".

Las híper superficies que se visualizan en las figuras 10 y 11 son sólidos tridimensionales obtenido después de "deformar" la región $-1 < u, v, w < 1$ (cubo) mediante los mapeos $h = (u, v, w, u^2 + v^2 + w^2)$ y $g = (u, v, w, u^2 + v^2 - w^2)$, respectivamente.

Finalmente, en las figuras 12 – 17 se visualizan ciertos elementos asociados a la híper esfera unitaria, $s(u, v, w) = (\cos w \cos u \cos v\,, \cos w \operatorname{sen} u \cos v,$ $\cos w \operatorname{sen} v\,, \operatorname{sen} w)$, $0 < u < 2\pi$, $-\frac{\pi}{2} < v, w < \frac{\pi}{2}$.

Además en la figura 18 se visualizan algunas de las "cáscaras" o "capas" de la híper esfera s, conjuntamente con sus respectivos campos vectoriales normales unitarios.

Todos los gráficos mostrados en las citadas figuras se han generado con el nuevo comando **ParametricPlot4D**.

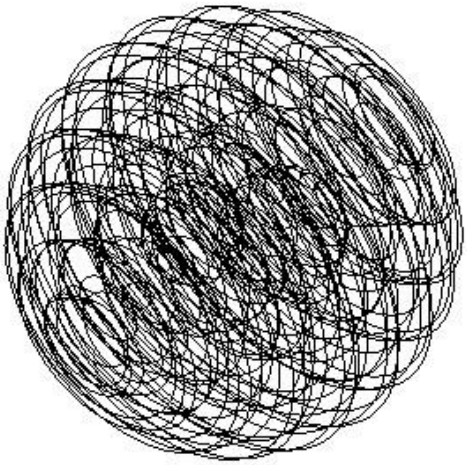

Figura 12: Curvas u-paramétricas asociadas a la híper esfera s.

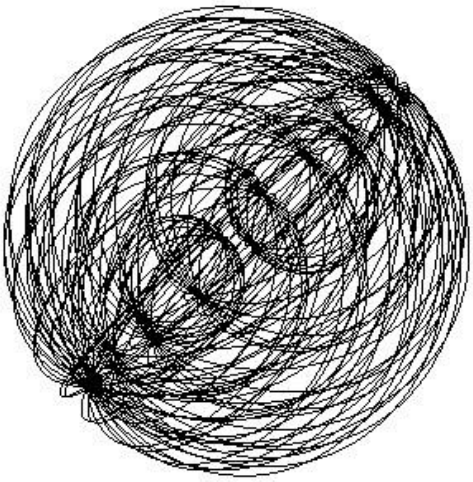

Figura 13: Curvas v-paramétricas asociadas a la híper esfera s.

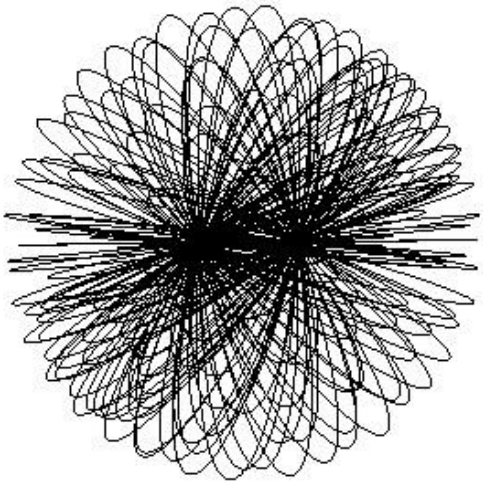

Figura 14: Curvas w-paramétricas asociadas a la híper esfera s.

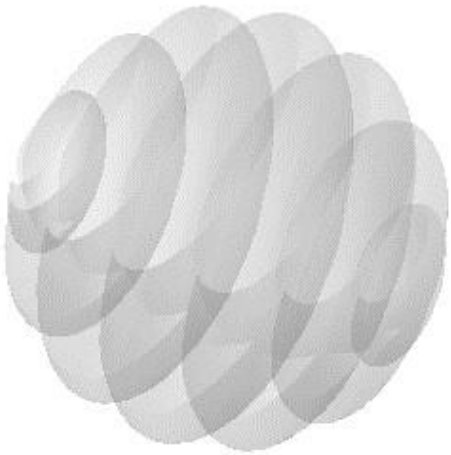

Figura 15: Superficies uv-paramétricas asociadas a la híper esfera s.

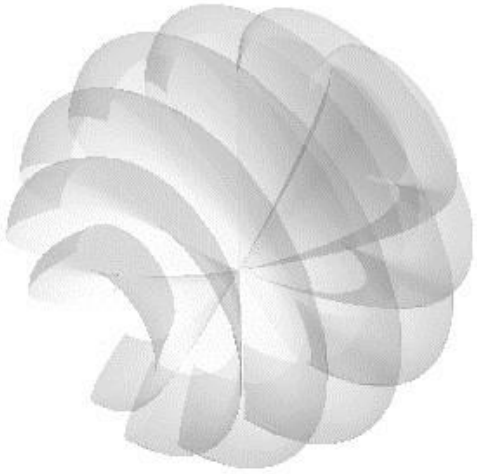

Figura 16: Superficies uw-paramétricas asociadas a la híper esfera s.

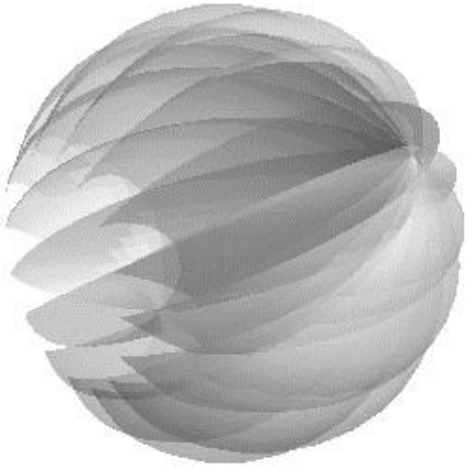

Figura 17: Superficies vw-paramétricas asociadas a la híper esfera s.

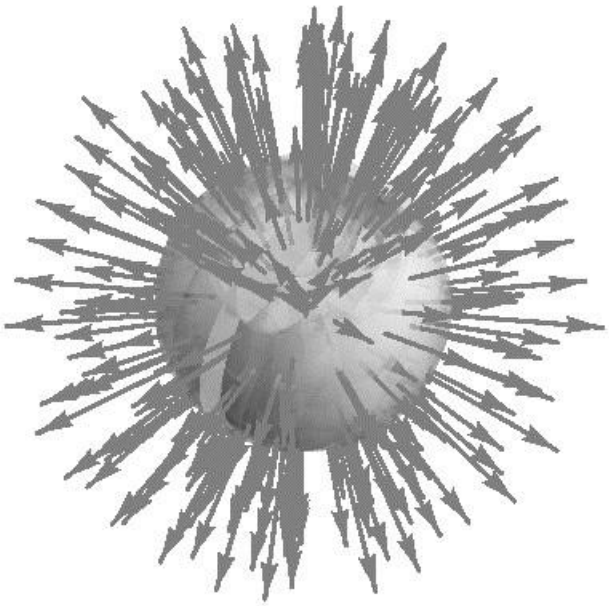

Figura 18: Algunas "capas" de la híper esfera s, acompañadas de sus respectivos campos vectoriales normales unitarios.

5. Conclusiones

Este artículo introduce un modelo para visualizar objetos en 4D con el sistema de cálculo simbólico *Mathematica*. El modelo queda descrito por una inmersión la cual es definida en el lenguaje de programación del *Mathematica* y que es la base sobre la que se implementa el nuevo comando

ParametricPlot4D, el cual facilita la visualización de curvas ($\mathbb{R}$ en $\mathbb{R}^4$), superficies ($\mathbb{R}^2$ en $\mathbb{R}^4$) e híper superficies ($\mathbb{R}^3$ en $\mathbb{R}^4$). Todos los cálculos en este trabajo se han hecho en el *Mathematica* v10.0 en un Procesador Intel Core i3, 2.4 GHz y 4 GB de RAM. En nuestra experiencia, este sistema es una herramienta ideal para esta tarea, ya que nos proporciona una serie de características simbólicas, numéricas y gráficas notables, junto con un lenguaje de programación simple pero potente [8].

Agradecimientos

Los autores expresamos nuestro agradecimiento al Dr. Tomás Recio, Departamento de Matemáticas, Estadística y Computación (Facultad de Ciencias, Universidad de Cantabria), por sus acertadas sugerencias y críticas con respecto a un material previo a la elaboración de este artículo.

Referencias

[1] Michael A. Noll, *A Computer Technique for Displaying n-Dimensional Hyperobjects*. Communications of the ACM 10(8), August 1967, pp 469-473.

[2] David W. Brisson, *Visual Comprehension of nDimensions*. Hypergraphics: Visualizing Complex Relationships in Art, Science & Technology, Westview Press, Boulder CO, 1978, pp 109-146.

[3] A. K. Dewdney, *Computer Recreations*. Scientific American, April 1986, pp 14-23.

[4] Thomas F. Banchoff, *Beyond the Third Dimension*. Scientific American Library, New York NY, 1990.

[5] S. Wolfram, *The Mathematica Book* (4th. edition). Wolfram Media, Champaign, IL & Cambridge University Press, Cambridge, 1999.

[6] Manuel Ujaldón, *Procesadores gráficos para PC*. Editorial Ciencia-3, S. L, Madrid, 2005.

[7] R. Maeder, *Programming in Mathematica* (Second Edition). Addison-Wesley, Redwood City, CA, 1991.

[8] R. Ipanaqué y A. Iglesias, *A Mathematica package for computing and visualizing the Gauss map of surfaces*. Lecture Notes in Computer Science, vol. 3482, 2005, pp. 492501.

E-mail: rvelezmorol@unp.edu.pe, ripanaquec@unp.edu.pe