UNIVERSIDAD CATÓLICA DE VALPARAÍSO - CHILE

ESCUELA DE INGENIERÍA INFORMÁTICA

Visualización de Datos Composicionales en 2 y 3 Dimensiones

Autores: Candy Valencia Torres & Mario de la Cuadra Izquierdo

INFORME FINAL DEL PROYECTO

PARA OPTAR AL TITULO

PROFESIONAL DE INGENIERO

DE EJECUCIÓN EN INFORMÁTICA.

Enero de 1996

UNIVERSIDAD CATÓLICA DE VALPARAÍSO - CHILE

ESCUELA DE INGENIERÍA INFORMÁTICA

Acta de Aprobación

Candy Valencia Torres & Mario de la Cuadra Izquierdo

_____

Christian Sifaqui Merczak.

Profesor Guía

Enero de 1996

RESUMEN

En un comienzo, el proyecto consistió en visualizar los datos composicionales en dos y tres dimensiones, además de ver la región que abarcan los datos . La región que se forma en 2D corresponde a un área que está dentro de un triángulo equilátero; en 3D, a un volumen que está dentro de un tetrahedro. El software final realizado debiera tener una buena interfaz gráfica y fácil interacción del usuario con los objetos visualizados.

Utilizamos software y herramientas gráficas para hacer imágenes de gran calidad visual.

La visualización en 3D, en nuestro proyecto, es una innovación comparado con otro software que sólo visualizaron los datos composicionales en 2 dimensiones.

Además, el proyecto permite que el usuario analice mejor el comportamiento entre las distintas composiciones involucradas.

ABSTRACT

In the begining, the project consisted mainly in draw the compositional data in two and three dimensions, moreover it to see of region that contain the data. The region that had formed in 2D correspond to an area that it were within of a equilateral triangle; in 3D, to a volumen that it were within of a tetrahedron. The final software realized must have a good graphic interfaz and an easy interaction of the user with of objects drawing. We utilize software and graphic tools to make imagens of great visual quality.

The visualization in 3D, in our project, is an innovation compared with another software that only drawing the compositional data in 2 dimensions. Futhermore, the project allow that the user analize major the behavior between diferents compositions involucrated.

Índice

CAPÍTULO 1
PLAN DE PROYECTO

1.1 OBJETIVOS

a. Procesamiento de los datos , utilizando la transformación de Aitchison y los métodos estadísticos.
b. Visualizar composiciones de 3 y 4 componentes.
c. Realizar una buena interacción con el usuario, eso incluye : la interfaz gráfica del usuario y la visualización de los datos.

1.2 TRABAJO RELACIONADO

El CODA es un software para el análisis estadístico de datos composicionales registrado por J. Aitchison (1986) . Este es un programa hecho en lenguaje BASIC que cumple con los objetivos estadísticos, aunque no con una buena interfaz gráfica ni con una buena visualización de gráficos.

El CODA logra realizar una visualización de los datos composicionales sólo en dos dimensiones (, es decir, utilizando 3 componentes).

1.3 FUNCIONES PRINCIPALES

1.3.1 Entrada de Datos.

Los datos y los identificadores se deben leer desde un archivo. Este archivo puede ser de texto, o un archivo propio del programa.

1.3.2 Pre-procesamiento de los datos

Se refiere al procesamiento de validación que tienen los datos para llegar a los cálculos estadísticos.

1.3.3 Representación Tabular.

Es decir, los datos aparecen organizados de igual forma que una planilla de cálculo.

1.3.4 Ordenamiento y selección de los datos.

Los datos se dividen en identificadores y componentes .
Los identificadores se podrán ordenar según la forma ascendente o descendente, y los componentes se pueden seleccionar según el criterio del usuario.

1.3.5 Proceso Estadístico.

El proceso estadístico consiste en la transformación de Aitchison y métodos estadísticos habituales.

1.3.6 Visualización de los datos.

1.3.6.1 Representación Gráfica .

La visualización consiste fundamentalmente mostrar un triángulo equilátero o un tetrahedro .

1.3.6.2 Generar región que agrupa a los datos.

Dibujar la región que agrupa los puntos en el triángulo y en el tetrahedro.
Se identifiquen cuáles puntos están dentro o fuera de la región de agrupación.

1.3.6.3 Imagen en Pov-Ray.

Se desea tener una escena final que será hecha en Pov-Ray por sus características de iluminación global.

1.4 ESTRATEGIAS

1.4.1 Primera Estrategia

Hacer un prototipo inicial y el desarrollo del software será con el modelo de ciclo de vida clásico. Para llevar a cabo el desarrollo del software se debe tener los recursos apropiados para trabajar. Por esta razón, utilizaremos la librería gráfica OpenGL que permitirá la manipulación directa de lo visualizado por parte del usuario. La interfaz gráfica del usuario será construida en base a las herramientas como TK y TCL. El Pov-Ray es el encargado de la imagen fotorrealística final.
La librería para los cálculos estadísticos es *MathEdge.*

1.4.2 Segunda Estrategia

Idem a lo anterior con excepción de que no hay librerías estadísticas, y los cálculos matemáticos se pueden hacer en un programa matemático y/o estadístico.

1.5 EVALUACIÓN

1.5.1 Primera estrategia :

Es improbable la compra de la librería por parte del Instituto de Estadística debido a que *MathEdge* debe ser para estaciones de trabajo. Por ende, ellos no están interesados en comprar una librería que no van a poder utilizar provechosamente. Otra razón secundaria es el precio de esta librería.

1.5.2 Segunda estrategia :

Hasta ahora la estrategia más viable .

1.6 AGENDA

1.6.1 Planificación del primer semestre 1995.

Inicialmente se planteó la siguiente programación :

1.6.1.1 Revisión bibliográfica

Esto comprende la investigación y revisión del material estadístico y de computación gráfica que se va a ocupar. Aquí se especifica los requerimientos del usuario.

1.6.1.2 Revisión del lenguaje y plataforma

Aquí se realiza un estudio del lenguaje y la plataforma a utilizar.

1.6.1.3 Selección del hardware y software

Elección del hardware y software involucrados en el desarrollo del proyecto.

1.6.1.4 Diseño

El diseño consiste en : un diseño arquitectónico y un diseño de la interfaz para el prototipo final del proyecto.

1.6.1.5 Implementación

1.6.1.5.1 Visualización de triángulo equilátero.

Hacer triángulo equilátero con puntos en colores.
Ubicar la posición exacta de los componentes dentro del triángulo.

1.6.1.5.2 Lectura desde el teclado y validación de los datos.

Leer las observaciones desde el teclado, dónde cada observación tiene 3 componentes. Además, realizar la validación que los componentes sean mayores e iguales a cero,

recalcular las componentes si es necesario y la transformación de Aitchison.

1.6.1.5.3 Interfaz 1

Realización de los menus.

1.6.1.5.4 Visualizar tetrahedro

Dibujar el tetrahedro y proveer al modelo de movimiento o animación con la interacción del mouse.

1.6.1.5.5 Lectura de datos desde un archivo

1.6.1.5.5.1 Lectura inicial.

Leer encabezado del archivo y datos para almacenarlo en listas enlazadas.

1.6.1.5.5.2 Ordenación y Selección de datos.

Almacenar los identificadores y componentes en una estructura de datos que facilite la selección y el ordenamiento.

1.6.1.5.6 Interfaz 2

Realización de la ayuda.

1.6.2 Diagrama de línea temporal (Diagrama de Gantt).

Para ver la planificación del primer y segundo semestre hay que ver el apéndice A.

1.7 RECURSOS DEL PROYECTO

1.7.1 Gente involucrada.

El desarrollo del software está a cargo de Candy Valencia T. y Mario de la Cuadra I. con apoyo del profesor José Pascual, del Instituto de Estadística de nuestra universidad y el profesor guía Christian Sifaqui M. de la Escuela de Ingeniería de Informática.

1.7.2 Hardware y software

1.7.2.1 Selección de Hardware

Nuestra plataforma de desarrollo es en equipos IBM RISC/6000 con procesador PowerPC . Esta plataforma es utilizada porque posee una gran capacidad de procesamiento de imágenes.

1.7.2.2 Selección del Software

Lenguaje C.

El lenguaje C utiliza las funciones de la librería OpenGL.

Tk y Tcl

Estas herramientas de programación serán utilizadas para hacer la interfaz gráfica del usuario (GUI).

Esto se debe a que :

a) Permite un desarrollo rápido de aplicaciones.
b) Realiza una fácil aplicación con un lenguaje poderoso.
c) y es de conveniencia para el usuario por la calidad de la interfaz que se genera.

Tcl (Tool command language) provee facilidades de programación genéricas tales como variables, ciclos y procedimientos que son útiles para las aplicaciones.

Los comandos de Tk (Tool kit) permiten crear interfases gráficas de usuario.

Para realizar los gráficos utilizamos la librería OpenGL y Pov-Ray.
Específicamente, OpenGL se usará para interactuar fácilmente con el objeto dibujado en pantalla. En cambio, el Pov-Ray se usará para generar una imagen estática más cercano a la realidad. Al utilizar OpenGL y Pov-Ray, en dónde el primer software genera imágenes con iluminación simple, en cambio, el segundo ; con iluminación global.

Uno de los beneficios de OpenGL es su independencia del :
- sistema operativo, sistema de ventana y de la red.

1.8 MECANISMOS DE SEGUIMIENTO Y CONTROL.

Los mecanismos de control y seguimiento están especificados en la carta Gantt.

CAPÍTULO 2
ESPECIFICACIÓN DE REQUISITOS DEL SOFTWARE.

2.1 INTRODUCCIÓN

2.1.1 Descripción general.

2.1.1.1 Investigación de materia estadística.

Los datos composicionales consisten en proporciones de un total que ha sido dividido en 2 o más partes y están sujetos a la restricción de sumar 1.

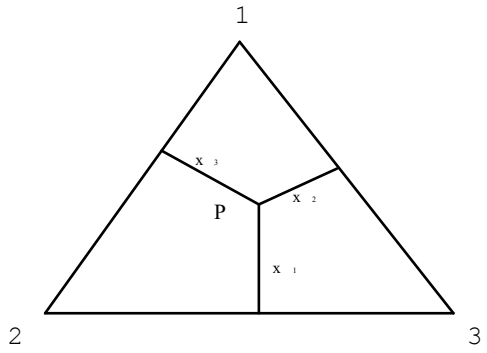
Los componentes de cualquier composición de *D* partes deben satisfacer las condiciones siguientes:

$$x_1 \geq 0, \ldots\ldots, x_D \geq 0$$

$$x_1 + \ldots + x_D = 1$$

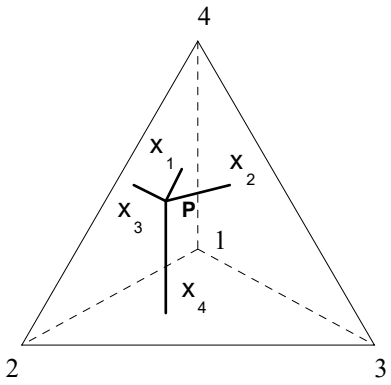
$x_1, \ldots, x_D$ con D partes.

Los datos se dibujan en un triángulo equilátero o en un tetrahedro, dependiendo si son 3 o 4 componentes.



Representación de una composición de tres partes, (x_1, x_2, x_3), en el triángulo de referencia 123.
Triángulo equilátero de altura 1.

Los datos composicionales es necesario que sumen 1. Estos datos se preparan para procesarse por medio de la transformación propuesta por Aitchison [1], luego a los datos resultantes se les aplican los métodos estadísticos y se obtiene una región que agrupa a los puntos que serán dibujados en el triángulo o tetrahedro. En caso de tener 3 componentes la región de agrupación sería un área (2D); con 4 componentes sería una volumen (3D).



Representación de una composición de
4 partes (x_1, x_2, x_3, x_4) en el tetraedro 1234.

El componente Xi corresponde a la perpendicular desde el punto representativo P a la cara triangular opuesta al vértice *i*.

2.1.1.1.1 Subcomposiciones

De una composición de D partes se puede descomponer en composiciones de menor dimensión. Para obtener una composición **x** de *C* componentes ($C \leq D$), se seleccionan las proporciones originales de estos *C* componentes y se transforman de tal manera que ellos sumen 1. A esta última operación la llamamos estandarizar o recalcular las componentes.

A continuación veremos un ejemplo obtenido de una memorista [2] del Instituto de Estadística.

Ejemplo : Considérese la composición química $(x_1, x_2, x_3, x_4, x_5)$ de cereales, (4.2, 0.5, 0.7, 92.4, 2.2) correspondiente a:

1. Proteínas
2. Minerales
3. Lípidos
4. Hidratos de Carbono
5. Otros

Supóngase que se desea estudiar la composición proteínas, minerales y lípidos, es decir, estudiar la composición compuesta por los componentes x_1, x_2, x_3, $\mathbf{x_s} = (x_1, x_2, x_3)$, de la composición $\mathbf{x}$=$(x_1, x_2, x_3, x_4, x_5)$. Entonces, la nueva composición, denotada por (s_1, s_2, s_3) es obtenida por:

$$(s_1, s_2, s_3) = \frac{1}{(x_1 + x_4 + x_5)} \bullet (x_1, x_4, x_5)$$

De esta forma, se tiene que la subcomposición proteínas, minerales y lípidos, para la composición química de cereales del ejemplo, se obtiene de la manera siguiente:

$$(s_1, s_2, s_3) = (4.2,\ 0.5,\ 0.7) \bullet \frac{1}{4.2 + 0.5 + 0.7} = (0.7778,\ 0.0926,\ 0.1296)$$

y multiplicando por 100, se obtiene la composición en forma de porcentajes: (77.78, 9.26, 12.96), es decir 77.78% de proteínas, 9.26% de minerales y 12.96% de lípidos.

2.1.1.1.2 Amalgamas

Una operación que resulta útil en el análisis de composiciones, es la de unir algunos componentes de una composición, dando origen a una nueva composición de menor dimensión. A continuación otro ejemplo sacado de la memoria.

Ejemplo Para los 3 conjuntos de composiciones de residuos sólidos presentados, donde los elementos que las constituyen son

1.	Material Orgánico	4.	Vidrio	7.	Tierra
2.	Papel	5.	Plástico	8.	Textil
3.	Metal	6.	Madera	9.	Otros

se puede realizar una agrupación de acuerdo a las características físicas de las composiciones y obtener para cada composición:

Grupo de materias fermentables	: Materia Orgánica	, 1
Grupo de materias inertes	: Metal, Vidrio, Tierra	, 2
Grupo de materias combustibles	: Papel, Plástico, Madera, Textil	, 3
Otros	: Otros	, 4

Entonces, una manera natural de estudiar la composición (Materia Orgánica, Materia Inerte, Materia Combustible, Otros) es considerando la composición (t_1, t_2, t_3, t_4),

donde:

$$t_1 = x_1$$
$$t_2 = x_3 + x_4 + x_7$$
$$t_3 = x_2 + x_5 + x_6 + x_8$$
$$t_4 = x_9$$

2.1.1.1.3 Visualización de la región

Para dibujar la región que abarca los puntos en un diagrama ternario, las componentes deben ser mayores que cero y se debe realizar la transformación de Aitchison para que los métodos estadísticos puedan ser utilizados. Los datos resultantes de los métodos se devuelven hacia el espacio del símplex, dónde está el diagrama ternario, pasando antes por la transformación de Aitchison.

2.1.1.2 Investigación de software y conceptos de Computación Gráfica

2.1.1.2.1 Tcl y Tk.

Tcl

Se originó debido a que se necesitaba un lenguaje de comando reusable.

Un lenguaje script de propósito general que podría ser construido como una librería en C, entonces esto podría ser reusado para diferentes propósitos en diferentes aplicaciones.

Para utilizar un enfoque basado en componentes requiere ser poderoso y flexible con un "pegado fuerte" para la semblanza de los componentes, y esta característica de pegado la puede proveer un lenguaje de texto compartido.

Tk permite componentes para ser cualquiera de los controles individuales de interfaz de usuario o aplicaciones, en cualquier caso los componentes pueden ser desarrollados independientemente y Tcl puede ser usado para ensamblar los componentes y comunicarlos entre ellos.

Unidos Tcl y Tk proveen un sistema de programación para el desarrollo y usando aplicaciones de interfases de usuario gráficas.

Tcl es un simple lenguaje de script para controlar y extensión de aplicaciones. Tcl provee facilidades de programación genérica, tales como variables, ciclos y procedimientos, que son útiles para una variedad de aplicaciones. Además, Tcl es encajable. Esto es interpretación es una librería de procedimientos en C que pueden fácilmente ser incorporados dentro de aplicaciones.

Uno de los principales extensiones útiles pata Tcl es Tk, la cual es un paquete de herramientas para el Sistema X Windows.
Tk está implementado como una librería de procedimientos en C asi puede ser usado dentro de muchas diferentes aplicaciones.

Tcl y Tk proveen cuatro beneficios para desarrolladores de aplicaciones y usuarios.

El primer beneficio es el desarrollo rápido para el programador. Hay una desventaja para el usuario : es que Tcl es interpretado, por supuesto, esto ejecuta más lentamente que el código compilado en C, mas en las estaciones de trabajo modernas es sorprendentemente rápido.

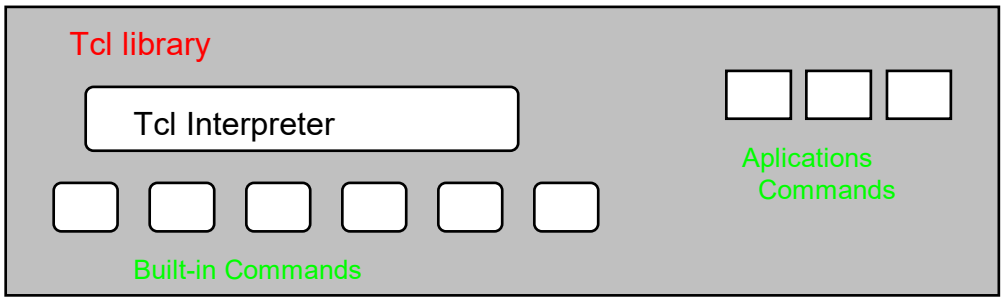
El segundo beneficio es que Tcl elabora esto fácil para aplicaciones que necesitan un lenguaje script poderoso .

El tercer beneficio de Tcl es que esto elabora un excelente "lenguaje de pegado fuerte". Tcl usa de un lenguaje común interpretado entre aplicaciones más poderoso y flexible que los enfoques estáticos, tales como OLE de Microsoft y ToolTalk de Sun Microsystem.

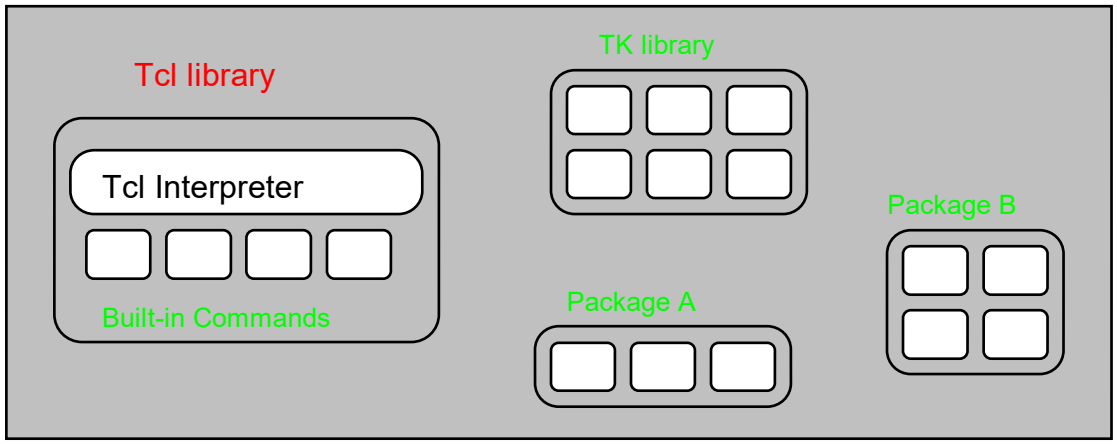
El cuarto beneficio de Tcl es la conveniencia para el usuario.

Para más información consultar libro de Ousterhout [3].

Simple aplicación



Una aplicación compleja



2.1.1.2.2 Nociones de Computación Gráfica en 3D

A continuación, indicamos ciertos conceptos básicos de la computación gráfica en 3D, veáse [4].

2.1.1.2.2.1 Manipulando las estructuras en tres dimensiones.

Un modelo de red poligonal consiste en una estructura de vértices, cada vértice es un punto en el espacio de 3 dimensiones que se llamado Espacio de Coordenadas del Mundo (World Coordinate Space = WCS).

Los objetos son definidos en un WCS la cual es convencionalmente de mano derecha. Algunas veces es conveniente definir objetos en su propio sistema de coordenadas (Local Coordinate System = LCS).

2.1.1.2.2.2 Geometría en la Computación Gráfica.

Un punto V (x,y,z) en el espacio se puede representar en una matriz como :

$$V = \begin{bmatrix} x & 0 & 0 \\ 0 & y & 0 \\ 0 & 0 & z \end{bmatrix}$$

Usando en notación de matrices, un punto V puede ser cambiado desde la traslación, escalamiento y rotación.

$$V' = V + D$$
$$V' = V * S$$
$$V' = V * R$$

D : Matriz de traslación
S : Matriz de escalamiento
R : Matriz de rotación

Estas operaciones son las principalmente usadas en las transformaciones en la computación gráfica.

Para una mayor facilidad de éstas operaciones se utiliza las coordenadas homogéneas.

En un sistema homogéneo un vértice V (x,y,z) es representado como :

$$V\ (X,Y,Z,w)$$

para cualquier factor de escala w ‡ 0.
Las representación de la coordenadas cartesianas en 3 dimensiones es entonces :

$$x = X / w$$
$$y = Y / w$$
$$z = Z / w$$

En la computación gráfica generalmente se utiliza un w = 1 .
Ahora la translación puede ser tratada como multiplicación de matrices.

$$V' = V * T$$

$$\begin{bmatrix} x' & y' & z' \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Tx & Ty & Tz & 1 \end{bmatrix}$$

El escalamiento queda :

$$V' = V * S$$

$$S = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

En la rotación, dónde a es el ángulo en grados.
La rotación se efectúa en contra del sentido de las agujas del reloj.

Rotación por el eje X :

$$V' = V * Rx$$

$$Rx = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(a) & \text{sen}(a) & 0 \\ 0 & -\text{sen}(a) & \cos(a) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotación por el eje Y :

$$V' = V * Ry$$

$$Ry = \begin{bmatrix} \cos(a) & 0 & -\text{sen}(a) & 0 \\ 0 & 1 & 0 & 0 \\ \text{sen}(a) & 0 & \cos(a) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotación por el eje Z :

$$V' = V * Rz$$

$$Rz = \begin{bmatrix} \cos(a) & \text{sen}(a) & 0 & 0 \\ -\text{sen}(a) & \cos(a) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.1.1.2.2.3 Vectores en la Computación Gráfica.

a. Definición y estudio del vector

Un vector es una entidad que posee magnitud y dirección.

V = (v1,v2,v3), dónde cada componente vi es escalar.

Un rayo puede ser especificado por dos puntos o por un punto y un vector. Si dos puntos finales del rayo son (x1,y1,z1) y (x2, y2,z2),

entonces el vector es:

$$V = (x2 - x1, y2 - y1, z2 - z1)$$

b. Sumando vectores

$$X = V + W$$
$$=(x1,x2,x3)$$
$$= (v1+w1,v2+w2,v3+w3)$$

c. Longitud de vectores

La magnitud o longitud de un vector está definida como :

$$|V| = (V_1^2 + V_2^2 + V_3^2)$$

La versión normalizada de U es :

$$U = V / | V |$$

d. Vectores normales y producto cruz.

Un normal vector a un polígono es calculado desde tres vértives (no-colineales) de la figura. Tres vértices definen dos vectores V1 y V2 y la normal al polígono es encontarda tomando el producto cruz de éstos.

$$Np = V1 \times V2$$

El producto cruz de 2 vectores V y W está definida como :

$$X = V \times W$$
$$= (v2\ w3 - v3\ w2)\ i + (v3\ w1 - v1\ w3)\ j + (v1\ w2 - v2\ w1)\ k$$

dónde i, j, k son vectores unitarios estándar.

$$i = (1, 0, 0)$$
$$j = (0, 1, 0)$$
$$k = (0, 0, 1)$$

e. Vectores normales y producto punto.

El producto punto, en computación gráfica, consiste en proveer una medida de el ángulo entre dos vectores.

$$X = V*W$$
$$= v1w1 + v2w2 + v3w3$$

Usando la regla del coseno tenemos :

$$V*W = |V||W|\cos(a)$$

f. Vectores asociados con el vector normal.

Hay tres importantes vectores que están asociados con la normal de la superficie. Ellos son el vector de dirección de la luz, L , el vector de reflectancia o vector de espejo, R, y el vector de visión, V.

2.1.1.2.3 Pov-Ray.

Es un programa que permite al usuario crear imágenes foto-realísticas en cualquier computador. Pov Ray lee archivos de texto ascii, en este se describen las formas, colores, texturas e iluminación y también se simula los rayos de luz moviéndose a través de la escena.

Las características importantes de Pov-Ray [5] son :

- Lenguaje de descripción de la escena, fácil de usar.
- Larga librerías de archivos con escenas de ejemplo.
- Archivos (- include -) que pre-definen formas, colores y texturas.
- Muy alta calidad de archivos de imágen de salida (24 bit color).
- Despliegue de color 15 y 24 bit sobre hardware apropiado.
- Luces proyectores (= spotlight) para la iluminación sofisticada.
- Alta iluminación especular Phong para superficies más realísticas.
- Varios formatos de salida de archivos de imagen incluyendo targa (.tga), deformación (= bump) y fila (.raw).

Amplio rango de formas :

- Primitivas de formas básicas tales como :esfera, caja, quadric, cilindro, icono, triángulo y plano.

- Formas pueden ser fácilmente combinadas para crear nuevas formas complejas. Esta característica es llamada Geometría Constructiva de Sólidos (CSG). La CSG soporta uniones, intersecciones y diferencias entre objetos.

- Primitivas de formas avanzadas tales como :
  Torus (Donut), Hiperboloides, Paraboloides, Bezier, Patch, Campos de altura (montañas), Blobs, Quartics, Triángulos suaves.

- Patrones construidos : Agata, bozo, ajedrez, granito, gradiente, leopardo, mandel, mármol, spotted, radial, madera y archivo de imagen de mapeado.

- Patrones de construcción de superficies de espejo .

2.1.1.2.4 OpenGL

2.1.1.2.4.1 Historia

OpenGL es el primer ambiente para desarrollar aplicaciones en dos y tres dimensiones. La interfaz de programación de aplicación OpenGL es un estándar en las multiplataformas industriales. De hecho su neutralidad ha sido probada por
: Silicon Graphics, AT & T, Cray Research, Digital Equipement, Hitachi, IBM, Intel, Intergraph, Microsoft, NEC, Samsung, Sony, Template Graphics SW, Univel y otras empresas .
OpenGL es directo descendiente de IRIS Graphics Library (IRIS GL), inventado en 1982.

2.1.1.2.4.2 Definición

OpenGL no es un paquete de herramientas, ni un sistema de ventana, ni tampoco está orientado a objetos.

Shamansky [6] expresa que OpenGL es :

- un sistema de modo inmediato
- flexible formato de aplicación
- un sistema procedimental
- funcional lista de despliegue

OpenGL es en sistema modo inmediato, esto es que los comandos son ejecutados inmediatamente.
OpenGL acepta varios tipos de datos, liberando al programador de innecesarias conversiones.

Es procedimental, en que el programador emite específicos comandos para determinar que está actualmente dibujando. Además, soporta una lista de despliegue sub-editable para mejorar la extensibilidad de la red, resultando una mejora de rendimiento a través de aplicaciones cliente-servidor y del rendering directo.

2.1.1.2.4.3 Composición de OpenGL

Consiste en varias piezas [7] :

2.1.1.2.4.3.1 Una librería de rendering.

Capacidades de la librería de OpenGL :

a. Las características básicas de 3 dimensiones :

- Sombreado de Gouraud en puntos, líneas y polígonos.
- Matriz y pilas.
- Iluminación de Phong por vértice
- Almacenando profundidad (Z)
- Despliegue creación de lista y facilidades de manipulación.

b. Puntos, líneas y poligonos anti-aliasing
c. Punto de subpixel exacto, linea y rasterización de polígono.
d. Planos de clipping definidos por el usuario.
e. Soporta dibujado de texturas.

- Texturas.
- Efectos atmosféricos y de niebla.
- Buffer "estarcido" (= stencil). Es útil para la geometría constructiva de sólidos, reflexión e espejo, interferencia y algoritmos de sombreado.
- Buffer "de acumulación". Es útil para el post-procesamiento de la imagen, obscurecer movimientos y efectos de la profundidad de campos.
- Buffer Alfa. Es útil para algoritmos de transparencia.

2.1.1.2.4.3.2 Un paquete de herramientas utilitario.

El paquete de herramientas de utilidad, GLU- graphic librery utility-, que provee las siguientes características :

a. Soporta curvas y superficies Non-Uniform Rational B-spline (NURBS)
b. Capacidades quadric, esfera, cilindro y discos.
c. Capacidades de tesselation de polígonos multicontornos y cóncavos.
d. Manipulación de trazado de texturas, incluyendo texturas escaladas y construcción de MIPMAP's.
e. Creación de tipos de matrices como son : de selección, proyección de perspectiva y ortográfica.
f. Convertir las coordenadas del mundo a coordenadas de la ventana.

2.1.1.2.4.3.3 Un protocolo de red.

Al tener la librería de rendering de OpenGL interpretada con el Sistema de Ventana X11. Esto significa que OpenGL maneja un protocolo de red.

2.1.1.2.4.3.4 Una librería integrada con el Sistema de Ventana, X 11.

La librería GLX provee las siguientes funciones que integran la librería de rendering con X11.

Creación y unión de contextos de dibujo a ventanas. Creación de ventanas que ayudan a través de tipos de mecanismos visuales extendidos. Sincronización de OpenGL con el flujo de datos de X11. Acceso suministrado para letras de X11. Widget 3D, subclases desde Corey XmPrimitives.

2.1.1.2.4.4 Librería Auxiliar

La librería OpenGL no incluye rutinas pra crear y manipular las ventanas, así como también las rutinas para eventos de lectura desde el teclado o mouse . Por lo tanto, éstas rutinas son provistas por la librería auxiliar.

La librería auxiliar también tiene rutinas que frecuentemente son usadas en computación gráfica y modelamiento geométrico para dibujar algunos simples objetos de 3 dimensiones tales como : esfera, cubo, tetrahedro y otros polihedros, un torus, una tetera.

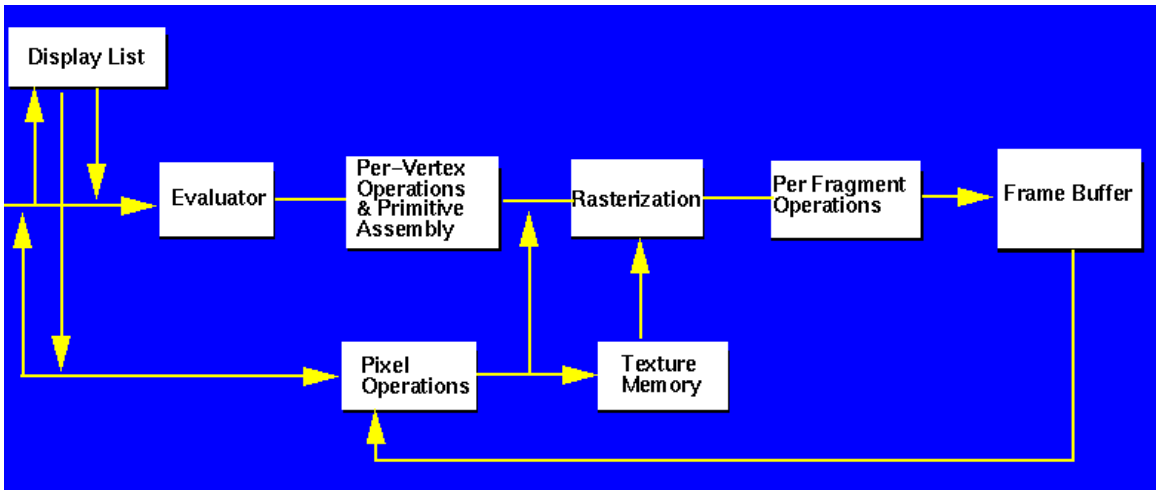
En resumen, hay dos funciones para el flujo de control del programa.

La primera función es auxIdleFunc ( ), especifica la función a ser ejecutada sino hay otro evento para ser procesado.

La segunda función es auxMainLoop ( ), especifica la función para ser invocada cuando la ventana necesita ser adaptada.

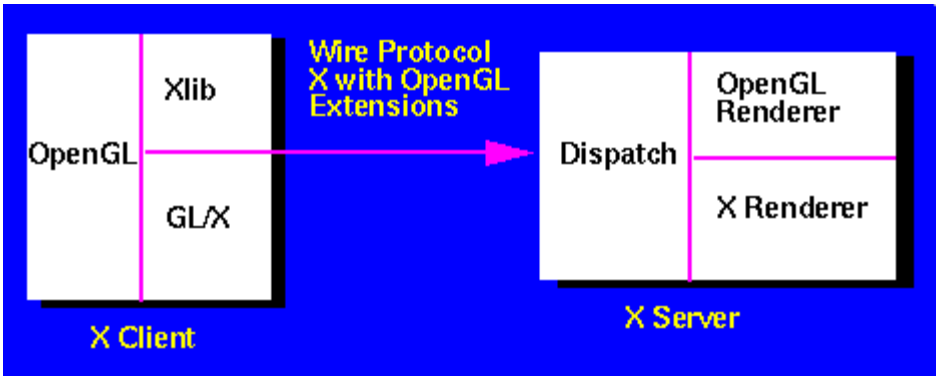
2.1.1.2.4.5 Arquitectura

La maquinaria de OpenGL es un buen modelo para la representación de los problemas gráficos. A continuación presentamos un diagrama de bloques que indica como procesa los datos OpenGL. Los comandos entran desde la izquierda y pueden proceder hacia el conducto de procesamiento. Al igual que todos los elementos de estado de OpenGL, los contenidos de la memoria de textura y el frame buffer pueden ser obtenidos por una aplicación OpenGL.



2.1.1.2.4.6 Modelo Cliente - Servidor

El modelo para la interpretación de comandos de OpenGL es cliente-servidor [6]. Un aplicación cliente emite comandos, los cuáles son interpretados y procesados por un servidor OpenGL. El servidor y el cliente pueden operar sobre diferentes máquinas, por eso OpenGL es transparente en red. Si no va sobre la red, entonces la comunicación cliente-servidor debe ser reemplazado por un rendering local, el cuál es mucho más rápido.

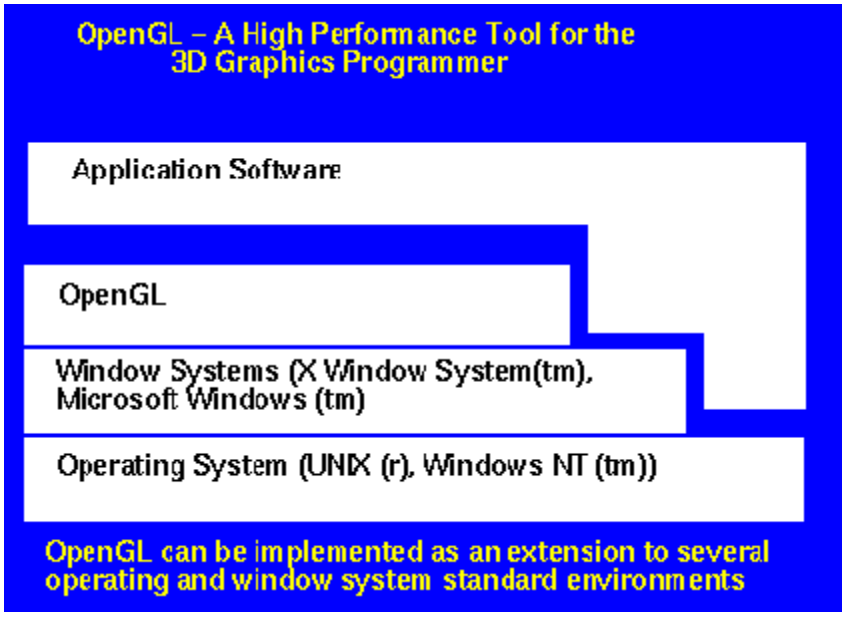


2.1.1.2.4.7 Independencia

OpenGL es independiente de [6] :

a) Sistema de ventanas
b) Sistema Operativo
c) Red

OpenGL es o sería compatible con otras API, tales como Xlib, OSF/Motif, IRIS inventor, Image Vision e IRIS Performer. OpenGL está diseñado para ser llamable desde los lenguajes C, C++, FORTRAN y ADA.



Más detalles de OpenGL se pueden encontar en [8] y [9].

2.1.2 Descripción especifica

2.1.2.1 Descripción inicial

a. Ingreso de datos desde archivos grabados.
Por el programa y por otro programas .Es necesario que se pueda leer los datos desde un archivo de texto.

b. Pre-procesamiento y
c. Ordenación los datos

El programa debe pedir los valores para cada composición. Los datos ingresados deben ser mayores o iguales a cero. Si los componentes de una composición no llegan a sumar 1, entonces se realiza un nuevo cálculo para que sea igual.

Ejemplo de datos composicionales

	Identificadores		*Componentes*		
Casos	Ciudad	Estrato	M. orgánica	M.sólidos	Otros
1	Viña	Alto	77,5	19,5	3,0
2	Viña	Alto	71,9	24,9	3,2
3	Viña	Medio	50,7	36,1	13,2
4	Viña	Medio	52,2	40,9	6,6
5	Valpo	Alto	70,0	26,5	3,5
...			...	...	...
99	Valpo	Medio	2,5	48,0	49,5
100	Valpo	Bajo	2,0	47,8	50,2

Seleccionar los componentes.

Ordenar , de mayor a menor o viceversa , el contenido del identificador.

Realizar amalgama.

d. Visualización Gráfica.
Del triángulo equilátero y el tetrahedro.

e. Transformación de Aitchison y métodos estadísticos habituales.

f. Visualizar Región.

Se verá el comportamiento de los datos composicionales y la región que abarcan éstos.

g. Otros

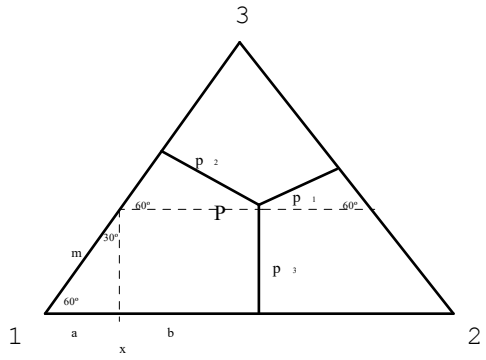
Mostrar imágen en Pov y en OpenGL . La imagen se podrá manipular por el usuario, viendo éste la imagen en distintos ángulos y perspectivas.

h. Ayuda

2.1.2.2 Cálculos Matemáticos

2.1.2.2.1 Ubicación de una composición estandarizada dentro del triángulo equilátero

Teniendo el siguiente diagrama ternario :



Matemáticamente se tiene que :

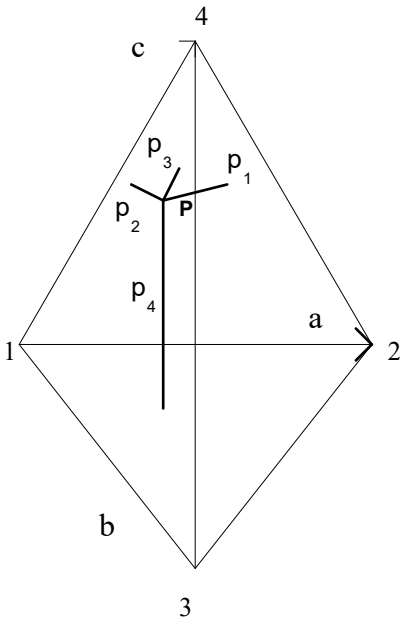
x = a + b
y = p3
sen(30º) = a / m = 1/2 => a = m / 2
sen(60º) = p2 / b = √3 / 2 => b = (2 * p2) / √3
sen(60º) = p3 / m = √3 / 2 => m = (2 * p3) / √3
x = m / 2 + (2 * p2) / √3 = p3 / √3 + (2*p2) / √3 =>

x = (2*p2 + p3) / √3
y = p3

Por lo tanto, para un dato composicional cuyas partes (componentes) son p1, p2, p3 se tiene que dicho punto en el diagrama ternario, se puede ubicar en el plano cartesiano xy.

2.1.2.2.2 Ubicación de una composición estandarizada dentro del tetrahedro.

Calculados ya , la ubicación del tetrahedro, con los siguientes vectores a,b y c:



Representación de una composición de
4 partes (p_1, p_2, p_3, p_4) en el tetraedro 1234

Para calcular la ubicación de la composición en el tetrahedro, aplicamos la fórmula de : la distancia de un punto a un plano. Así debemos calcular la distancia desde un punto $\left(x_p, y_p, z_p\right)$ a cada uno de los planos o caras que conforman el tetrahedro. De ahí que obtengamos cuatro matrices. A cada matriz le aplicamos la determinante , lo que se generan 4 vectores. Estos vectores son normalizados y cada vector es igualado al valor de la componente p_1 o p_2 o p_3 o p_4. Entonces tenemos, 4 ecuaciones distintas con tres incógnitas esto implica que se puede determinar las raíces para x, y , z.

El resultado de los cálculos dió que las coordenadas x_p, y_p, z_p son :

$$x_p = \frac{1}{4}\left(2 * p_2 + p_4 + p_3\right) * \sqrt{6}$$
$$y_p = p_4$$
$$z_p = \frac{1}{4}\sqrt{2}\left(3 * p_3 + p_4\right)$$

dónde $\left(p_1, p_2, p_3, p_4\right)$ es una observación y se debe cumplir que

$$p_1 + p_2 + p_3 + p_4 = 1$$

Ver apéndice C para más detalles de los cálculos.

2.1.2.2.3 Transformación Inicial

Para realizar la transformación se require que las componentes sumen exactamente 1.

En un caso particular de tres componentes, la transformación de Aitchison o transformación de log-razones es de la siguiente manera :

$$x_1 + x_2 + x_3 = 1 \quad , x_i > 0 \quad i = 1 .. 3$$

Si elegimos una de las tres componentes : x_2 , entonces :

$t_1 = (x_1 / x_2)$
$t_2 = (x_3 / x_2)$

$y_1 = LN(t_1)$ y $y_2 = LN(t_2)$

La función LN() : da como resultado del logaritmo natural.
Estos son los nuevos datos que se ocuparán para los métodos estadísticos.

2.1.2.2.4 Cálculos de los métodos estadísticos

Se realizó la lectura del archivo en donde se encuentran los datos composicionales, luego se llevo acabo la transformación de Aitchison, la cual es una transformación en términos de log-razones. Para cada dato composicional ya sea de tres o cuatro componentes se tiene que luego de realizada la transformación se obtiene un dato transformado. Estos datos transformados son almacenados en una matriz para posteriormente realizar el cálculo de los métodos estadísticos.

Inicialmente, cada observación o caso se escribe como un vector x_i con p elementos, es decir, que p es igual al número de componentes menos uno. La constante n es el número total de observaciones o casos.

Dado El promedio de los vectores :

$\bar{x} = \frac{1}{n}\sum_{1}^{n} x_i$, es una matriz de px1.

$$S = \frac{1}{n-1}\sum_{1}^{n}\left(x_i - \bar{x}\right)\left(x_i - \bar{x}\right)'$$

$F_{p,n-p}(\alpha)$: F es una distribución teniendo como numerador y n-p el denominador, n y n-p reflejan los grados de libertad con que se trabaja y α indica la probabilidad.

La ecuación a resolver para μ es :

$$n(\bar{x} - \mu)' S^{-1}\left(\bar{x} - \mu\right) = \frac{p(n-1)}{n-p} F_{n,n-p}(\alpha)$$

A partir de estos cálculos se obtiene una ecuación general de la elipse o elipsoide, elipse ($ax^2 + by^2 + cxy + dx + ey + f = 0$) o elipsoide ($ax^2 + by^2 + cz^2 + dxy + exz + fyz + gx + hy + iz + j = 0$), dependiendo si el número de componentes es igual a 3 o 4 , respectivamente.

2.1.2.2.5 Transformación Final

Por cada punto dibujado del elipsoide o la elipse se aplica la transformación de Aitchison final que es la siguiente :

Particularmente con una elipse

$$i = e^{x}$$
$$j = e^{y}$$

Además,

$$p_1 = \frac{i}{(i + j + 1)}$$
$$p_2 = \frac{j}{(i + j + 1)}$$
$$p_3 = \frac{1}{(i + j + 1)}$$

dónde $p_1, p_2, p_3 > 0$ y $p_1 + p_2 + p_3 = 1$

Cuando hay 4 componentes, se realiza el proceso análogo al de 3 componentes.

2.1.2.3 Visualización de región

2.1.2.3.1 Introducción

A partir de los datos con el número de componentes igual 3 visualizamos un objeto que llamamos área de agrupación de puntos; con el número de componentes igual a 4, un objeto que llamamos volumen final o volumen de agrupación. Tanto al volumen final como el área se le denominan regiones de agrupación.

El modo cómo se visualizan estas regiones es :

En el caso que tengamos datos composicionales, con el número de componentes igual 3, a cada una de las composiciones son transformadas de acuerdo a la fórmula de Aitchison, para luego con estos datos sirven de base para los cálculos estadísticos . Estos cálculos generan una ecuación de una elipse.

Posteriormente, cada punto del elipse se le aplica la transformación de

Aitchison y la ubicación de cada punto dentro del tetrahedro da el volumen de agrupación.

En el caso que tengamos datos composicionales, con el número de componentes igual 4, a cada una de las composiciones son transformadas de acuerdo a la fórmula de Aitchison, para luego con estos datos sirven de base para los cálculos estadísticos . Estos cálculos generan una ecuación de una elipsoide.

Finalmente, cada punto del elipse se le aplica la transformación de Aitchison y la ubicación de de cada punto dentro del triángulo equilátero da el área de agrupación.

2.1.2.3.2 Visualización del área de agrupación en el triángulo equilátero

De la ecuación de la elipse, se dividen en 2 funciones

$$y_1 = f(x) \text{ e } y_2 = f(x)$$

La forma de dibujar el área consistió en ir recorriendo contantemente los valores de acuerdo al intervalo de x, y por cada valor de x obtenemos 2 valores de y de la ecuación de una elipse.

Ver anexo para ver el cálculo del intervalo de x ,y las funciones y_1 e y_2.

Ahí mostramos cómo calculamos el intervalo de x en función de los coeficientes de la ecuación de la elipse y la expresión y en función de x.

2.1.2.3.3 Visualización del volumen de agrupación o volumen final en el tetrahedro.

El visualizar el volumen depende de la ecuación de elipsoidal.

Para dibujar usamos la ecuación :

$$z = f(x,y) \text{ e } y = h(x)$$

Recorremos cada valor del intervalo de x, y obtenemos los valores máximo y mínimo del intervalo de la variable y . Después por cada valor del intervalo y se hallan los dos valores de z. Luego dibujamos el punto.

En el anexo se pueden ver cómo hallar los valores máximo y mínimo del intervalo de x . Los valores de máximo y mínimo son expresados en función de los coeficientes de la ecuación del elipsoide.

Los valores de máximo y mínimo del intervalo y son calculados en función de x, así tenemos $y_{Max} = h(x)$ e $y_{min} = h(x)$. Para hallar el valor z, entonces $z_1 = f(x,y)$ $z_2 = f(x,y)$

Resultado y desventajas

En la visualización del volumen final no se puede ver el interior de éste. Nosotros necesitamos ver que composiciones están dentro o fuera del volumen. Y de la forma que se ésta dibujando no lo podemos ver.

El recorrido de los valores de x e y le asignamos un valor constante en forma arbitraria . Este valor puede ser modificado dependiendo de la ecuación del elipsoide que se haya generado a partir de los datos.

2.1.2.3.3.1 Reducción de la ecuacion elipsoide a la ecuación canónica.

A continuación presentamos los pasos principales de reducción de la ecuación general de un elipsoide a la ecuación canónica

$$\hat{y} = b_0 + \sum_i b_i x_i + \sum_{i<j} \sum_j b_{ij} x_i x_j + \sum_i b_{ii} x_i^2 \qquad (1)$$

$$b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ : \\ b_n \end{bmatrix} \qquad B = \begin{bmatrix} b_{11} & \frac{b12}{2} & \frac{b13}{2} & .. \\ \frac{b21}{2} & b22 & b23/2 & .. \\ \frac{b31}{2} & \frac{b32}{2} & b33 & .. \\ : & : & : & bnn \end{bmatrix}$$

y sabiendo que

$$\hat{y} = b_0 + x'b + x'Bx \qquad (2)$$

Si x_0 es un vector que indica el centro del elipsoide, entonces el vector x lo reemplazamos por x_0. Así tenemos :

$$y_0 = b_0 + x_0'b + x_0'Bx_0 \qquad (3)$$

,derivando por x_0.

$$b + 2Bx_0 = 0 \Rightarrow x_0 = \frac{-b}{2B} \qquad (4)$$

La ecuación canónica es expresada por :

$$\hat{y} = y_0 + \lambda_1 w_1^2 + \lambda_2 w_2^2 + \lambda_3 w_3^2 + ... + \lambda_n w_n^2 \qquad (5)$$

Para el elipsoide queda lo siguiente:

$$\hat{y} = y_0 + \lambda_1 w_1^2 + \lambda_2 w_2^2 + \lambda_3 w_3^2 = 0$$

De la matriz B, podemos saber los valores de los lambdas y por la ecuación (3), hallar el valor de y_0.

$$a = \sqrt{\frac{y_0}{\lambda_1}} \qquad b = \sqrt{\frac{y_0}{\lambda_2}} \qquad c = \sqrt{\frac{y_0}{\lambda_3}} \qquad (6)$$

Hasta el momento, sabemos el tamaño del elipsoide y la traslación del elipsoide con el vector de traslación x_0. Nos queda por calcular la matriz de rotación M.

De la matriz B se obtienen los vectores característicos (ver [10]). Los vectores característicos se normalizan (, es decir , son expresados en vectores unitarios) y los nuevos valores de los vectores son ubicados dentro de la matriz de rotación M.

$$M = \begin{bmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{bmatrix}$$

2.1.2.3.3.2 Forma de dibujar volumen.

En el sistemas de coordenada finales, el elipsoide se representa de la siguiente manera:

$z = Mw$, dónde M es la matriz de rotación y $z = x - x_0$

Por lo tanto,

$$x - x_0 = Mw \Leftrightarrow x = Mw + x_0 \qquad (7)$$

Esta fórmula nos indica como debemos proceder para dibujar el elipsoide.

1°) Se dibuja en el sistema de coordenadas canónico una elipsoide centrada en el origen de acuerdo com la siguiente ecuación paramétrica:

$$x = a * \text{sen}(u) * \cos(v)$$
$$y = b * \text{sen}(u) * \text{sen}(v)$$
$$z = c * \cos(u)$$

$$u \in [0, \pi]$$
$$v \in [0, 2\pi]$$

a , b, c son constantes

2°) Por cada punto del elipsoide se aplica la multiplicación de los valores de la matriz de rotación.

3°) cada punto es trasladado de acuerdo a los valores de x_0.

4°) luego, cada punto del elipsoide se le aplica la transformación de Aitchison.

5°) finalmente, los nuevos componentes son valores que se utilizan en la ecuación de ubicación dentro del tetrahedro.

6°) Se dibuja el punto (x,y,z).

2.1.2.3.3.3 Visualización en malla de alambre

El volumen final dentro del tetrahedro, está formado por un malla de triángulos y así se podrá ver los puntos que agrupa. Para una implementación más óptima del volumen en malla de alambre utilizamos la ecuación paramétrica del volumen.

El proceso de visualización es el siguiente :

- almacenar los vértices del volumen en una estructura de datos .
- almacenar los vértices que corresponden a cada cara del elipsoide en una estructura de datos.
- con estas dos estructuras de datos azamos las líneas que van a conformar volumen final.

2.1.2.3.3.4 Visualización de "llenado".

Esta visualización consiste en ver cómo es la superficie del volumen final.
Para visualizar la superficie del volumen, tomamos cada cara y obtenemos la normal al plano.

El proceso de visualización es muy parecido al anterior :

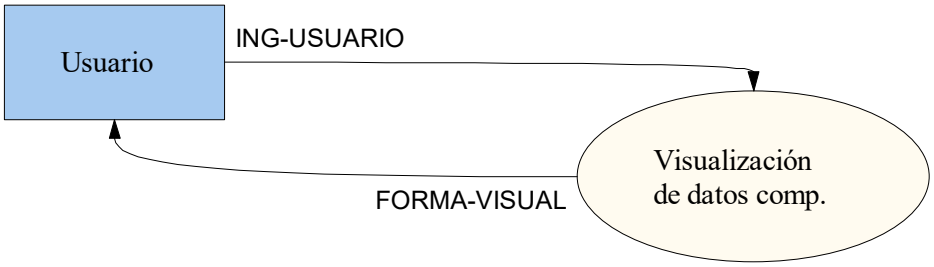
- almacenar los vértices del volumen en una estructura de datos .
- almacenar los vértices que corresponden a cada cara del elipsoide en una estructura de datos.
- con estas dos estructuras de datos conformamos un polígono y le asignamos la normal correspondiente de cada cara. Cada cara está definida como el plano de tres vértices.

2.2 DESCRIPCIÓN DE LA INFORMACIÓN.

2.2.1 Representación del flujo de información

2.2.1.1. Diagrama de Flujo de datos

Diagrama de Contexto



Visualización de datos comp. Diagrama 0

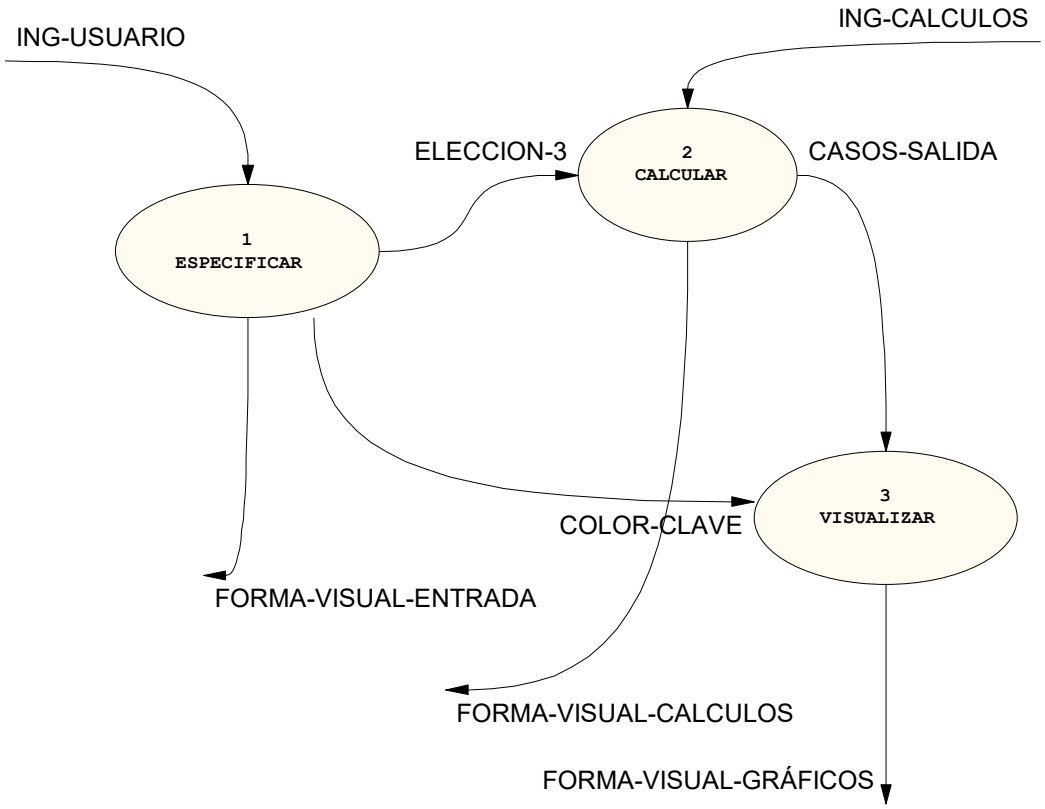
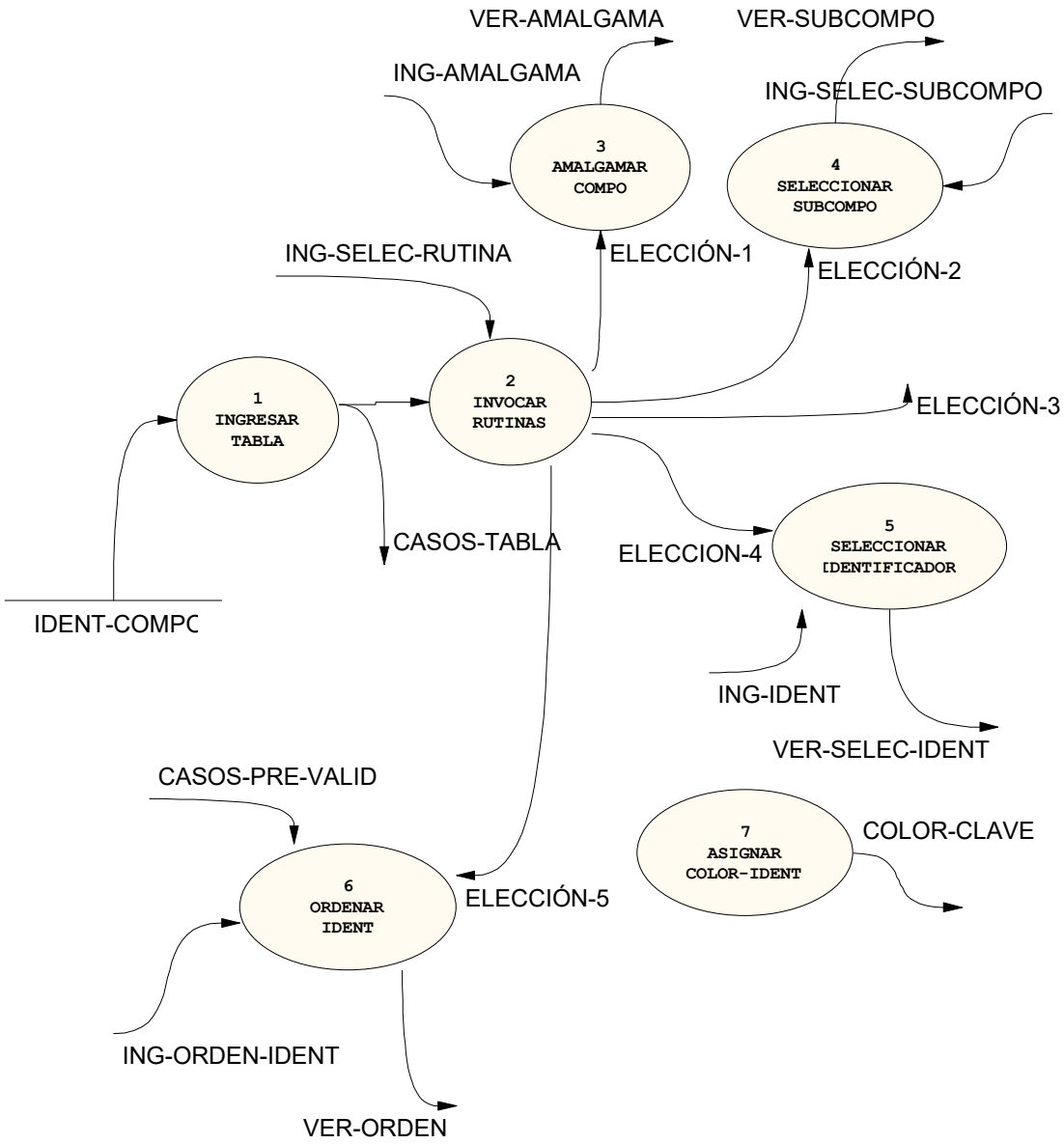
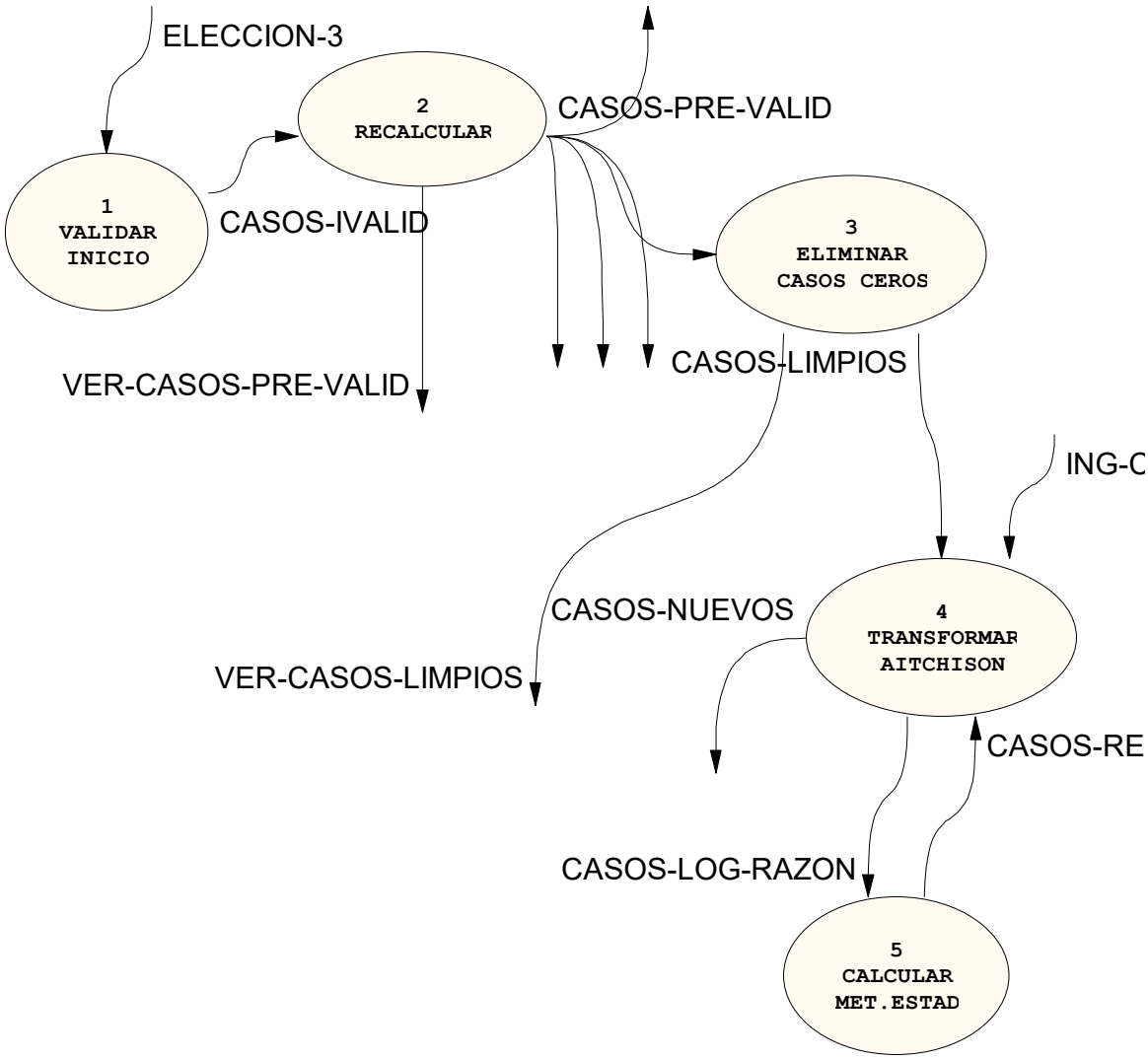


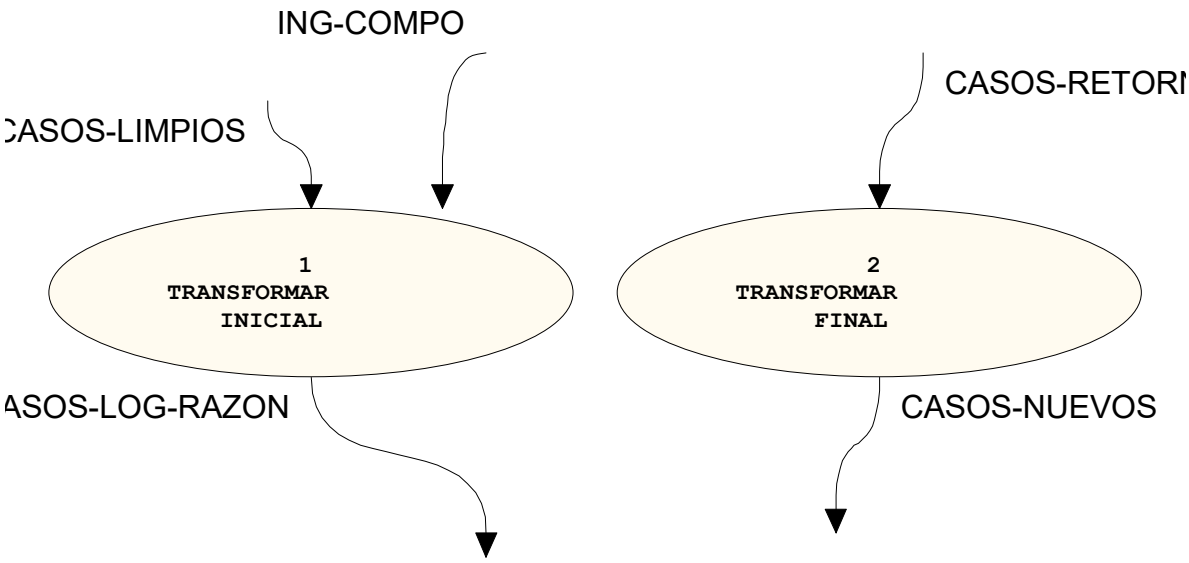
Diagrama 1
Especificar



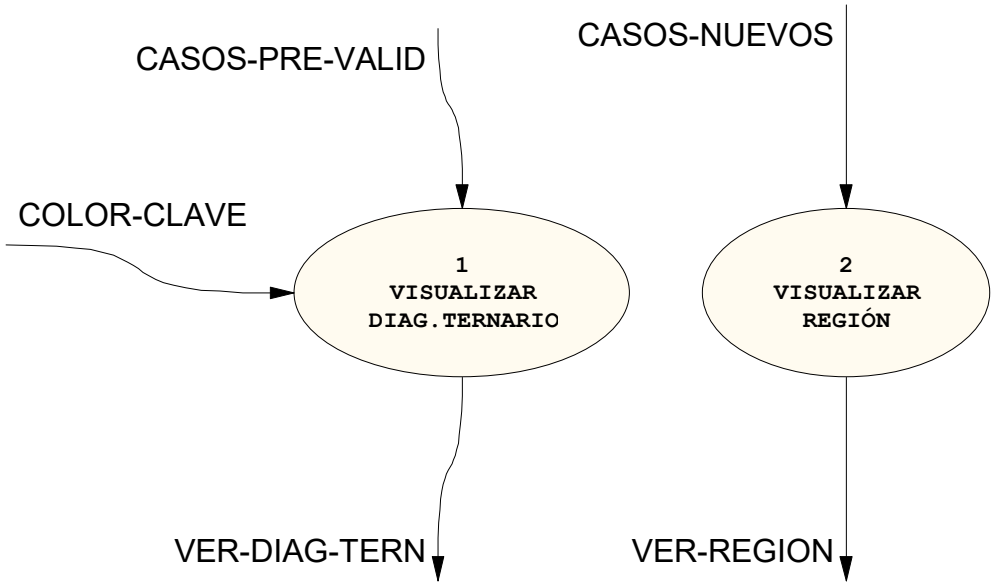
Calcular
Diagrama
2



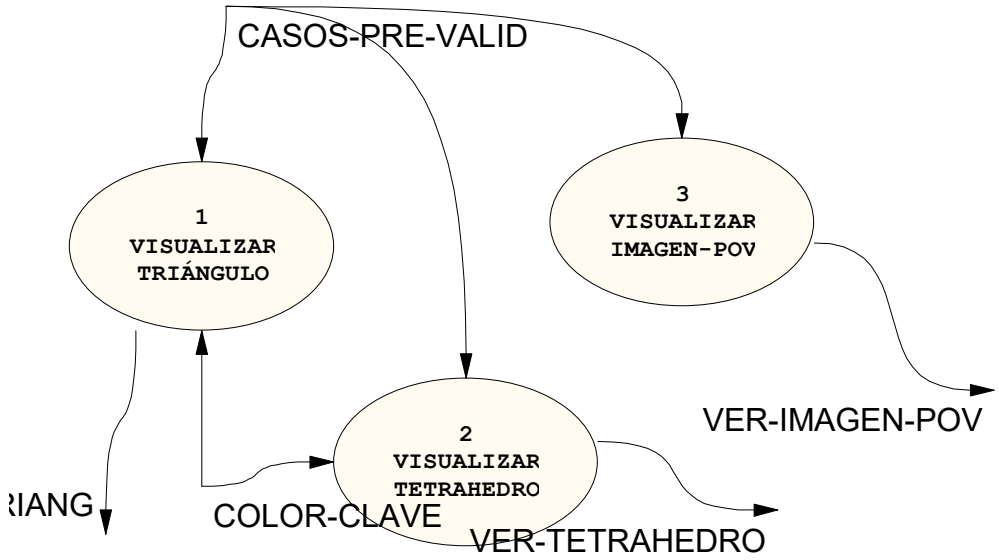
Transformar
Aitchison
Diagrama
2.4



Visualizar
Diagrama
3



Visualizar
Diag.Ternario
Diagrama
3.1



Visualizar
Región
Diagrama
3.2

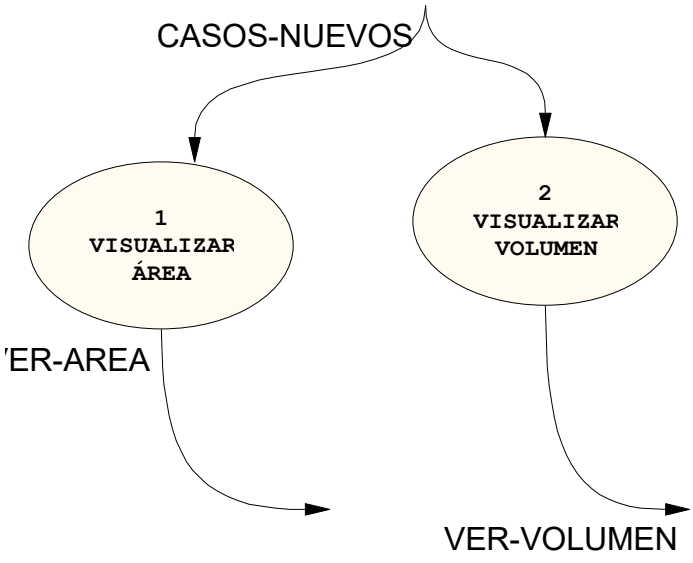
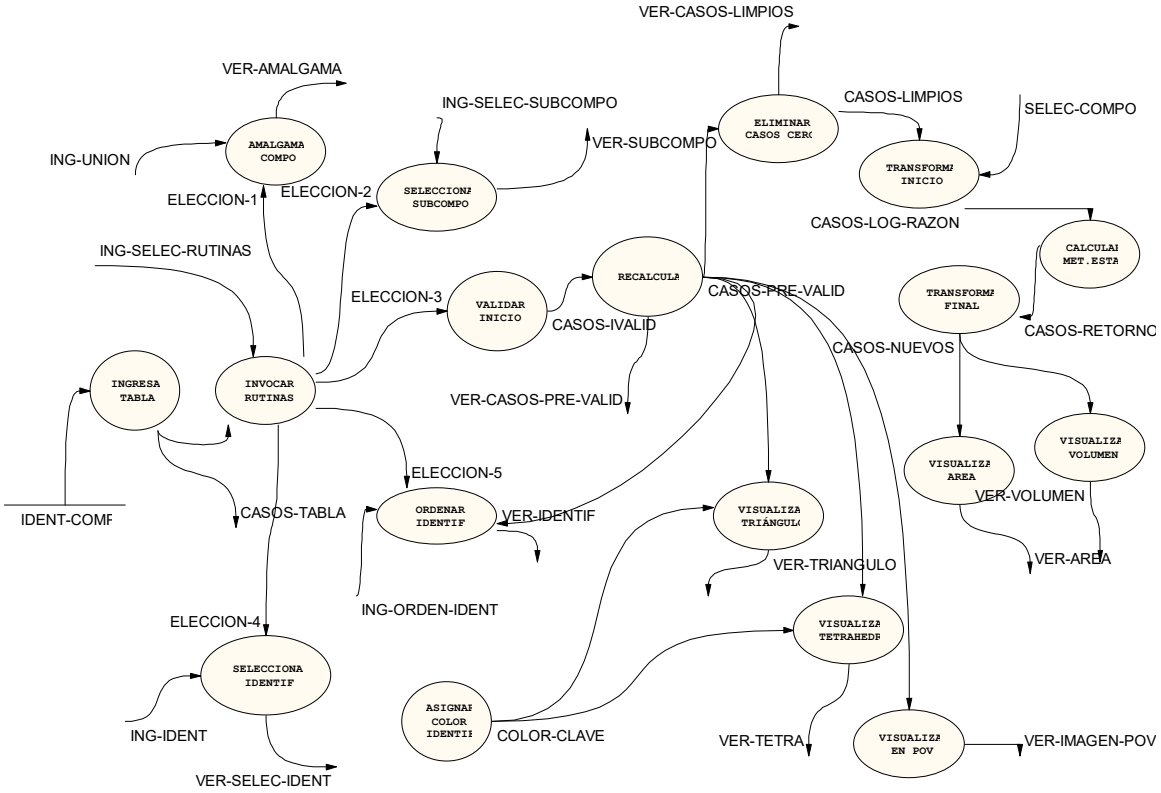


Diagrama
de últimos nivele



2.2.2 Representación del contenido de la información.

nComp : Número de componentes (= D partes)
nIdent : Número de identificadores.
nCasos : Número total de composiciones o casos.

Nombre : Identificador
Alias : IDENT
Dónde se usa / cómo se usa :
es de tipo de cadenas de caracteres
Descripción : IDENT es información elemental

Nombre : Componente
Alias : COMPO
Dónde se usa / cómo se usa :
es de tipo decimal
Descripción : COMPO es información elemental.

Nombre : Encabezado
Alias : ENCABEZADO
Dónde se usa / cómo se usa :
entrada: datos leidos desde un archivo.
salida : Ídem a la información anterior.
Descripción : ENCABEZADO = [({ENCAB-IDEN}^nIdent +)

{ENCAB-COMP}^nComp]

Nombre : Encabezado de identificador
Alias : ENCAB-IDEN
Descripción : ENCAB-IDEN es información elemental ; del tipo caracteres.

Nombre : Encabezado de componente
Alias : ENCAB-COMPO
Descripción : ENCAB-COMPO es información elemental ; del tipo caracteres.

Nombre : Casos de tabla
Alias : CASOS-TABLA
Dónde se usa / cómo se usa :
entrada: INGRESAR-TABLA .
salida : INVOCAR-RUTINAS..
Descripción : CASOS-TABLA son observaciones que están almacenados en una estructura de datos.

Nombre : Forma de entrada visual
Alias : FORMA-VISUAL-ENTRADA
Dónde se usa :
entrada: ESPECIFICAR los datos de entrada
salida : informe-visual por pantalla al usuario
Descripción : FORMA-VISUAL-ENTRADA =
VER-ORDEN + VER-AMALGAMA +
VER-ELECCION + VER-SUBCOMPO
+ CASOS-TABLA.

Nombre : Ingreso de datos por el usuario
Alias : ING-USUARIO
Dónde se usa
entrada : Ingreso del usuario por entrada estándar.
salida : Hacia las diferentes funciones de ESPECIFICAR y
CALCULAR.
Descripción : ING-USUARIO = ING-ESPECIFICA +
ING-CALCULOS.

Nombre : Ingreso de especificación
Alias : ING-ESPECIFICA
Dónde se usa :
entrada: Ingreso de datos por el usuario
salida : Hacia ESPECIFICAR
Descripción : ING-ESPECIFICA =
ING-SELEC-RUTINAS + ING-AMALGAMA +
ING-ORDEN-IDENT + ING-SELEC-
SUBCOMPO.

Nombre : Ingreso de datos en cálculos
Alias : ING-CALCULOS
Dónde se usa
entrada: Datos ingresados por el usuario.
salida : CALCULAR datos
Descripción : ING-CALCULOS = ING-COMPO

Nombre : Ingreso de componente
Alias : ING-COMPO
Dónde se usa
entrada: Datos ingresados por el usuario.
salida : TRANSFORMAR INICIAL
Descripción : ING-COMPO = ENCAB-COMPO

Nombre : Forma visual de presentar los procesos de cálculos
Alias : FORMA-VISUAL-CALCULOS
Dónde se usa
entrada: CALCULAR datos
salida : informe de salida en pantalla
Descripción : FORMA-VISUAL-CALCULOS = VER-CASOS- IVALID + VER-CASOS-PRE-VALID + VER-CASOS-LIMPIOS.

Nombre : Forma visual gráfica
Alias : FORMA-VISUAL-GRÁFICOS
Dónde se usa
entrada: procesos de VISUALIZAR los datos
salida : USUARIO
Descripción : FORMA-VISUAL-GRÁFICOS = VER-DIAG-TERN + VER-REGIÓN.

Nombre : Ingreso de datos para ordenar según el identificador
Alias : ING-ORDEN-IDENT
Dónde se usa
entrada: USUARIO
salida : ORDENAR-IDENTificador
Descripción : ING-ORDEN-IDENT = ING-IDENT + ING-FORMA

Nombre : Ingresar identificadores
Alias : ING-IDENT
Dónde / Cómo se usa
entrada: USUARIO
salida : INGRESAR-ORDEN-IDENT
ING-IDENT es de tipo caracter.
Descripción : ING-IDENT , información elemental.

Nombre : Ingreso en forma se ordenará el identificador.
Alias : ING-FORMA
Dónde se usa
entrada: USUARIO
salida : INGRESAR-ORDEN-IDENT
Descripción : ING-FORMA indica lo [ASCENDENTE / DESCENDENTE]

Nombre : Ingresar datos para amalgama o unión.
Alias : ING-UNION
Dónde se usa
entrada: USUARIO
salida : AMALGAMAR COMPOnentes
Descripción : ING-UNION = { NUEVA-COMPO +
{ENCAB-COMPO}^mCompo }^xGrupo

Nombre : Nuevo componente
Alias : NUEVO-COMPO
Dónde se usa
entrada: USUARIO
salida : AMALGAMAR-COMPOnentes
Descripción : NUEVO-COMPO información elemental, de tipo caracter.

Nombre : Ingresar subcomponente a seleccionar.
Alias : ING-SELEC-SUBCOMPO
Dónde se usa
entrada: USUARIO
salida : SELECCIONAR-SUBCOMPO
Descripción : ING-SELEC-SUBCOMPO = {ENCAB-COMPO}^m

Nombre : Casos validados
Alias : CASOS-PRE-VALID
Dónde se usa / cómo se usa :
entrada : son datos son estandarizados, es decir, todas la componentes de una caso particular suman 1.(RECALCULAR)
salida : VISUALIZAR-DIAG-TERN y ELIMINAR-CEROS.
CASOS-
Descripción : CASOS-PRE-VALID = {{COMPO}^nComp} ^nCasos

Nombre : Casos validados inicialmente
Alias : CASOS-IVALID
Dónde se usa / cómo se usa :
entrada: VALIDAR-INICIO
salida : RECALCULAR las componentes si es necesario.
Descripción : CASOS-IVALID = {{COMPO}^nComp }^nCasos

Nombre : Casos limpios
Alias : CASOS-LIMPIOS = {{COMPO}^nComp}^nCasos
Dónde se usa / cómo se usa
entrada: ELIMINAR-CASOS-CEROS
salida : Transformación de Aitchison (TRANSFORMAR AITCHISON)
Descripción : CASOS-LIMPIOS = {{COMPO}^nComp }^nCasos

Nombre : Casos se les aplica fórmula de razones de logaritmos.
Alias : CASOS-LOG-RAZON
Dónde se usa / cómo se usa
entrada: Transformación de Aitchison
salida : hacia los cálculos estadísticos (CALCULAR MET. ESTADÍSTICOS)
Descripción : CASOS-LOG-RAZON= {{COMPO}^nComp }^nCasos

Nombre : Resultado de los cálculos estadísticos
Alias : CASOS-RETORNO-A-LOG-RAZON
Dónde se usa / cómo se usa
entrada: CALCULAR MET.ESTADÍSTICOS
salida : TRANSFORMAR FINAL
Descripción : CASOS-INV-LOG-RAZON = { {COMPO}^nComp }^nCasos

Nombre : Resultados de los cálculos matemáticos
Alias : CASOS-NUEVOS
Dónde se usa / cómo se usa :
entrada: TRANSFORMAR FINAL
salida : VISUALIZAR REGIÓN
Descripción : CASOS-NUEVOS = { {COMPO}^nComp }^nCasos

Nombre : Imagen de Región
Alias : VER-REGIÓN
Dónde se usa / cómo se usa
entrada : VISUALIZAR-REGIÓN
salida : USUARIO
Descripción : VER-REGIÓN = [VER-ÁREA / VER-VOLUMEN]

Nombre : Imagen de Diagrama Ternario
Alias : VER-DIAG-TERN
Dónde se usa / cómo se usa :
entrada: VISUALIZAR-DIAG-TERN
salida : USUARIO
Descripción : VER-DIAG-TERN = [VER-TRIÁNGULO / VER-TETRAHEDRO + VER-IMAGEN-POV].

Nombre : Seleccionar componente
Alias : ING-COMPO
Dónde se usa / cómo se usa :
entrada: USUARIO
salida : Transformación de Aitchison
Descripción : ING-COMPO es una información elemental; del tipo numérico o caracter.

Nombre : Salida Visual
Alias : FORMA-VISUAL
Dónde se usa / cómo se usa :
entrada: USUARIO.
salida : SISTEMA DE VISUALIZACIÓN DE DATOS COMPOSICIONALES.
Descripción : FORMA-VISUAL = VER-VISUAL-ENTRADA + VER-VISUAL-CALCULOS + VER-VISUAL-GRÁFICOS.

Nombre : Visualización del tetrahedro
Alias : VER-TETRAHEDRO
Dónde se usa
entrada: VISUALIZAR-TETRAHEDRO
salida : USUARIO
Descripción : VER-TETRAHEDRO; Imagen visual

Nombre : Visualización de triángulo equilátero
Alias : VER-TRIÁNGULO.
Dónde se usa
entrada: VISUALIZAR-TRIÁNGULO
salida : USUARIO
Descripción : VER-TRIÁNGULO; Imagen visual.

Nombre : Visualización de área
Alias : VER-ÁREA
Dónde / Cómo se usa
entrada: VISUALIZAR-ÁREA
salida : USUARIO
Descripción : VER-ÁREA es una imagen visual.

Nombre : Visualización gráfica del volumen que agrupa los puntos
Alias : VER-VOLUMEN
Dónde se usa
entrada : VISUALIZAR-VOLUMEN
salida : USUARIO
Descripción : VER-VOLUMEN es una imagen visual.

Nombre : Visualización más real del tetrahedro
Alias : VER-IMAGEN-POV
Dónde / Cómo se usa
entrada: VISUALIZAR-IMAGEN-POV
salida : USUARIO
Descripción : VER-IMAGEN-POV.

Nombre : Primera elección
Alias : ELECCIÓN-1
Dónde se usa
entrada: INVOCAR-RUTINAS
salida : AMALGAMAR
Descripción : ELECCIÓN-1 = CASOS-TABLA + NUM-ELECCION

Nombre : Segunda elección
Alias : ELECCIÓN-2
Dónde se usa
entrada: INVOCAR-RUTINAS
salida : SELECCIONAR-SUBCOMPO
Descripción : ELECCIÓN-2 = CASOS-TABLA + NUM-ELECCION

Nombre : Tercera elección
Alias : ELECCIÓN-3
Dónde se usa
entrada : INVOCAR-RUTINAS
salida : VALID-INICIO
Descripción : ELECCIÓN-3 = CASOS-TABLA + NUM-ELECCION

Nombre : Cuarta elección
Alias : ELECCIÓN-4 = CASOS-TABLA + NUM-ELECCION
Dónde se usa
entrada : INVOCAR-RUTINAS
salida : SELECCIONAR-SUBCOMPO
Descripción : ELECCIÓN-4 = CASOS-TABLA + NUM-ELECCION

Nombre : Quinta elección
Alias : ELECCIÓN-5
Dónde se usa
entrada: INVOCAR-RUTINAS
salida : ORDENAR-IDENTificador
Descripción : ELECCIÓN-5 = NUM-ELECCION

Nombre : Sexta elección
Alias : ELECCIÓN-6
Dónde se usa
entrada : INVOCAR-RUTINAS
salida : ASIGNAR-COLOR-IDENTificador
Descripción : ELECCIÓN-6 = CASOS-TABLA + NUM-ELECCION

Nombre : Número de elección
Alias : NUM-ELECCION
Cómo se usa : Es de tipo numérico.
Descripción : NUM-ELECCION es información elemental.

Nombre : Color asignado a cierto identificador o identificadores
Alias : COLOR-CLAVE
Dónde se usa
entrada: ASIGNAR-COLOR-IDENT
salida : VISUALIZAR-TRIÁNGULO y VISUALIZAR-TETRAHEDRO
Descripción : COLOR-CLAVE = COLOR + {ENCAB-IDENT}^nIdent

Nombre : Ingreso del color para cierto identificador
Alias : ING-CLAVE-COLOR
Dónde se usa
entrada: USUARIO
salida : ASIGNAR-COLOR-IDENT
Descripción : ING-COLOR-CLAVE = COLOR + {ENCAB-IDENT}^nIdent

Nombre : Color
Alias : COLOR
Cómo se usa Es de tipo numérico.
si el COLOR = 1, entonces color rojo
si el COLOR = 2, entonces color verde
si el COLOR = 3, entonces color azul.
Descripción : Información elemental

2.3 DESCRIPCIÓN FUNCIONAL

2.3.1 Narrativa de procesamiento.

2.3.1.1. ESPECIFICAR

Es dónde se ingresan los datos y se clasifican, seleccionan y ordenan para su posterior procesamiento.

2.3.1.1.1 INGRESAR TABLA

Cumple la función de ingresar los datos desde un archivo y los datos se almacenan en una estructura de datos.

2.3.1.1.2 INVOCAR RUTINAS

Se seleccionan las rutinas al gusto del cliente.

2.3.1.1.3 AMALGAMAR-COMPO

Se realizan las amalgamas de componentes.

2.3.1.1.4 SELECCIONAR-SUBCOMPO

De las n componentes se seleccionan m (m <= n).

2.3.1.1.5 ORDENAR-IDENT

Se seleccionan los identificadores que se quieren analizar para mostrarlos en un orden establecido.

2.3.1.1.6 SELECCIONAR-IDENT

De las w identificadores se seleccionan t (t<=w).

2.3.1.2 CALCULAR

Se realizarán todos los cálculos matemáticos y estadísticos para preparar para :

2.3.1.2.1 VALIDAR-INICIO

Verifica que todos casos u observaciones sean mayores o iguales a cero.

2.3.1.2.2 RECALCULAR

Verifica que la suma de los componentes en cada observación sumen uno.

2.3.1.2.3 ELIMINAR-CASOS-CEROS

Eliminación de los casos que tengan algún componente igual a cero.

2.3.1.2.4 TRANSFORMAR-AITCHISON

Se realiza la transformación de Aitchison para las observaciones.

2.3.1.2.4.1 TRANSFORMAR-INICIO

Se hace la transformación de Aitchison según el ingreso de elección de una componente y de los casos cuyas componentes sean mayores que cero y sumen uno.

2.3.1.2.4.2 TRANSFORMAR-FINAL

Se hace la transformación de Aitchison ingresando los datos en el sentido inverso al que se realizó en la primera entrada. Los datos que se aplican en la transformación vienen de la función que se encarga de los cálculos estadísticos.

2.3.1.2.5 CALCULAR MÉTODOS ESTADÍSTICOS

Realizar cálculos matemáticos y estadísticos con los datos.

2.3.1.3 VISUALIZAR los datos

Visualización de los datos en el diagrama ternario (triángulo equilátero o tetrahedro). También se visualiza la región que agrupa los puntos en el diagrama ternario. En el caso de un triángulo equilátero es un área ; en el tetrahedro, un volumen.

2.3.1.3.1 VISUALIZAR-GRÁFICA

Es dónde se visualiza el diagrama ternario. Según la cantidad de componentes se dibuja un triángulo equilátero o un tetrahedro.

2.3.1.3.1.1 VISUALIZAR-TRIÁNGULO

Consiste en la visualización del triángulo equilátero con los puntos en el interior. Para obtener está visualización, necesariamente deberá pasar por el proceso de : VALIDAR-INICIO, RECALCULAR y que el número de componentes sea igual a tres.

2.3.1.3.1.2 VISUALIZAR-TETRAHEDRO

Consiste en la visualización de los casos dentro del tetrahedro. Para obtener esta visualización se requieren los datos pasar por : VALIDAR-INICIO, RECALCULAR y que el número de componentes sea igual a 4.

2.3.1.3.1.3 VISUALIZAR-IMAGEN-POV

La imagen final producida en la función VISUALIZAR-TETRAHEDRO.

2.3.1.3.2 VISUALIZAR REGIÓN

Después de los cálculos estadísticos y transformación de Aitchison, los datos resultantes se emplean para dibujar la región que abarca los puntos en el diagrama ternario.

2.3.1.3.2.1 VISUALIZAR-ÁREA

Se necesitan que los datos vengan procesados por los cálculos estadísticos y de Aitchison para: y en base a esto realizar el área que abarca los puntos.

2.3.1.3.2.2 VISUALIZAR-VOLUMEN

De los datos provenientes de la transformación de los datos se realiza : El volumen que agrupa los datos.

Para realizar esta visualización se requiere que el número de componentes sea igual a cuatro

2.4 CRITERIOS DE VALIDACIÓN.

Respuesta esperada del software.

Se espera que el software interactúe rápidamente en la visualización de los objetos, la interfaz gráfica y del procesamiento de los datos.

CAPÍTULO 3
ESPECIFICACIÓN DEL DISEÑO

3.1 DESCRIPCIÓN DEL DISEÑO

3.1.1 Descomposición de las burbujas en flujos de transacción y transformación.

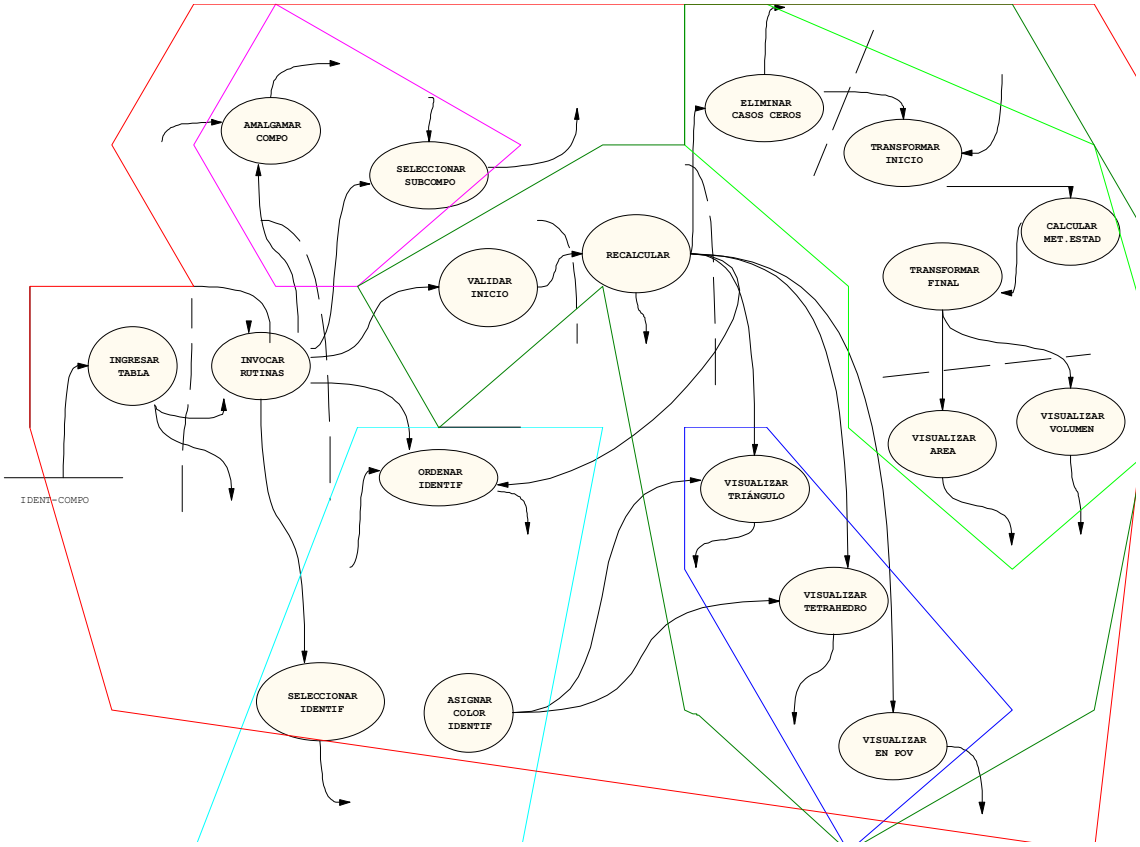


Figura A

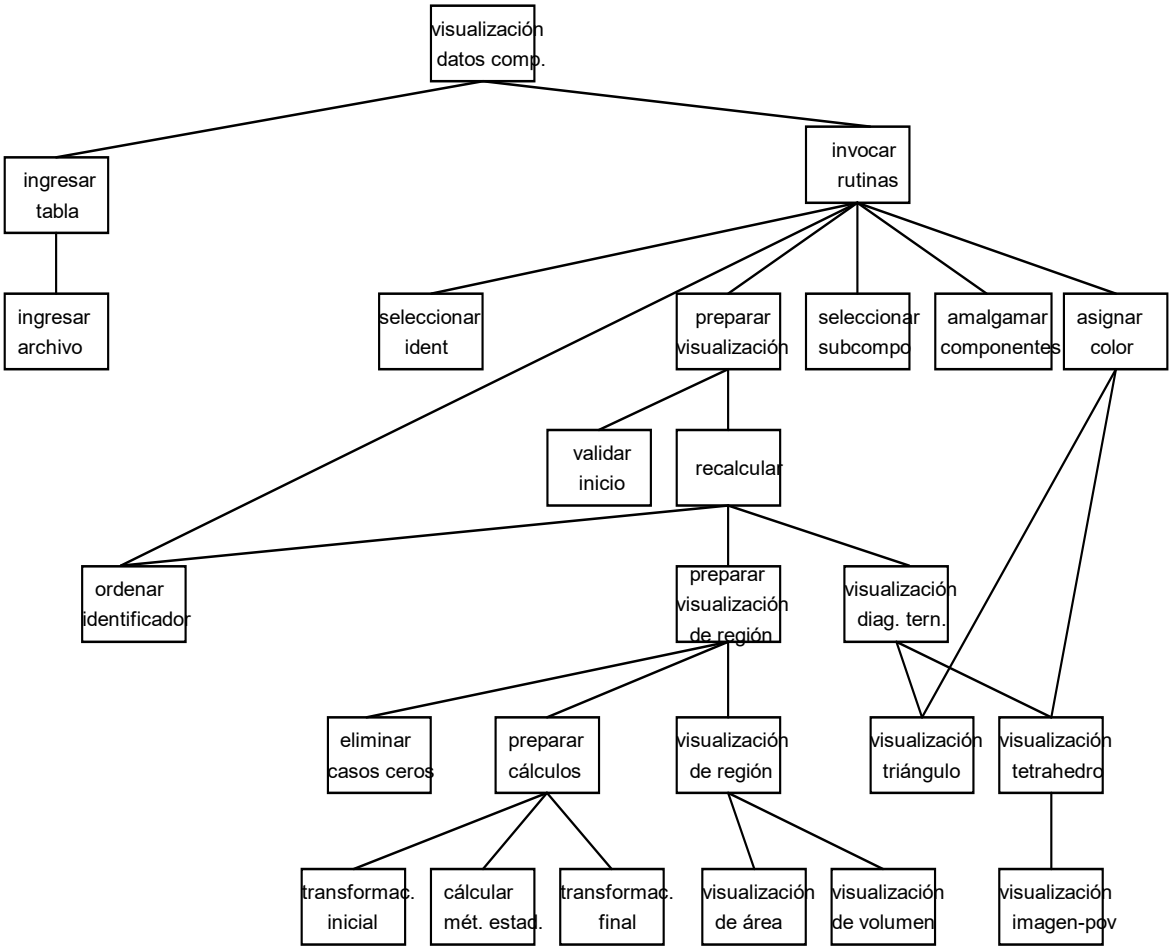
Las líneas rojas representan la región R0.
Las líneas celestes representan la región R1.
Las líneas rosados representan la región R2.
Las líneas verdes representan la región R3.
Las líneas azules representan la región R4.
Las líneas verdes claro representan la región R5

Los procesos que están dentro de la región R5 serán implementados en el programa matemático MAPLE. Los procesos que están dentro de la región R4 serán implementados con el uso del lenguaje C y la librería gráfica OpenGL. Los procesos restantes serán implementados con la utilización de el lenguaje de comando, TCL y un paquete de herramientas como es TK.

En la figura podemos distinguir 6 regiones y cada región tiene un flujo de cierto tipo [11] .
Así tenemos que :

La región R0 tiene características de un flujo de transacción.
La región R3 tiene características de un flujo de transacción.
La región R5 tiene características de un flujo de transformación.

3.2 DISEÑO ARQUITECTÓNICO



3.3 TABLA DE DECISIÓN

Condiciones	**Reglas**			
Número de componente = 3	v	f	v	f
Número de componente = 4	f	v	f	v
Validación	v	v	v	v
Calcular m. estad	f	f	v	v
Acciones				
Visualizar Triángulo	x			
Visualizar Tetrahedro		x		
Visualizar Área			x	
Visualizar Volumen				x

La función Validación implica el proceso de PRE-VALID y RECALCULAR las composiciones.

3.4 ESTRUCTURA DE DATOS Y ARCHIVOS

3.4.1 Estructura de datos

Los datos que son leídos desde un archivo se almacenan en listas (enlazadas).

3.4.2 Estructura del Archivo

Se exigirá que el archivo de entrada tenga una cabecera con el siguiente formato :

IDENT.1	IDENT.2	IDENT.3	COMP.1	COMP.2	COMP3

Los identificadores van separados por un espacio, al igual los componentes van separados por un espacio. Los identificadores y los componentes van separados por un tab.

Este archivo es de acceso secuencial y de dimensiones variables ya que pueden haber *n* identificadores y *n* componentes.

Se denominará identificadores a la naturaleza de los datos, es decir las distintas procedencias de los datos, por ejemplo que provienen de la ciudad de Viña del Mar, del estrato alto, etc.

Los componentes son los que conforman los datos composicionales que están sujetos a la restricción de ser mayores que cero .

CAPÍTULO 4
DISEÑO DE LA INTERFAZ.

4.1 MODELO DE USUARIO

El usuario que va a usar este software debe tener un mínimo conocimiento estadístico para comprender los procesos que están ocurriendo. El programa pretende que lo utilicen usuarios frecuentes y no tan frecuentes. Las personas que pueden utilizar el software son los estudiantes universitarios y profesores.

4.2 LA PERCEPCIÓN DEL SISTEMA.

El usuario final desea que :

Al tener los datos presentados en la pantalla, mediante un tipo de planilla de cálculo, se puedan seleccionar las componentes y los identificadores que se van a graficar. Este gráfico puede ser en 2 o 3 dimensiones dependiendo de la cantidad de los componentes seleccionados. Una de las opciones que especifica el usuario es la de ordenación de los identificadores, éstos se pueden ordenar según cierto nombre o descripción.

Además, según el tipo de identificador tiene un color particular.

4.3 MODELO DE DISEÑO.

En un inicio la interfaz del usuario tendrá un menú principal en la parte superior de la pantalla. Estas opciones del menú principal también pueden indicarse mediante los botones del ratón, haciendo que la interacción del software con el usuario sea mayor, por lo tanto hay un ahorro de tiempo que redunda en una mayor productividad del usuario.

4.4 MENÚS PRINCIPALES DEL SOFTWARE.

A continuación identificamos los menús principales de la interfaz gráfica.

4.4.1 Archivo

Abrir Archivo Existente

Consiste en ver el contenido de un archivo elegido por el usuario. Los datos se presentan en forma tabular, es decir, los datos se distribuyen como una una planilla de cálculo.

Guardar como ...

Almacena los datos actuales en un archivo con un nombre especificado por el usuario.

Cerrar

Cierra todas las ventanas activas del programa.

Salir

Salir del programa.

4.4.2 Selección

Consiste en lo relacionado con los identificadores y componentes.
Se divide en las siguientes tareas :

a) Seleccionar Identificador.

Selecciona el identificador o los identificadores. Cuando seleccionamos los identificadores estamos eligiendo las filas en la planilla de cálculo.

b) Seleccionar Componentes.

Selecciona las componentes, es decir, está eligiendo las columnas de planilla de cálculo.

4.4.3 Utilidad

Ordenar Identificador

Se ordenan los datos según el nombre de un identificador particular. Los datos pueden ser ordenados en la forma ascendente o descendente.

4.4.4 *Procesos*

Son los procesos que el programa va haciendo para graficar los datos , y que el usuario puede seleccionarlos para ver lo que hace.

La función Procesos tiene las siguientes tareas :

a) Subcomposición

Si la cantidad de componentes es mayor que 4 , las observaciones o casos se pueden dividir en varias composiciones de 3 y 4 componentes.

b) Amalgama

Es dónde se juntan o agrupan componentes en una sola.

c) Visualización

Esta opción tendrá la opciones de la visualización de las composiciones y de la región dentro del diagrama ternario (el triángulo equilátero o el tetrahedro).

4.4.5 Ayuda

Indice

Autores

CAPÍTULO 5
CODIFICACIÓN

El código fuente se puede ver en los apéndices : C, D, E y F. El apéndice C tiene el código hecho en Maple; el D, en Tk/Tcl; el E , en Pov-Ray; y el F, en el lenguaje C y la librería OpenGL. Durante a fase de Codificación se consultó a los siguientes libros [3,5,12,13,14,15,16,17,18]

CAPÍTULO 6
CONCLUSIONES

Actualmente, el software realizado de Visualización de datos composicionales en 2D y 3D opera en una estación de trabajo RISC/6000 con SO AIX, sistema de ventana X y el sistema administrador de ventanas Motif .

Para que el software se ejecute en ambiente MS-DOS y con Windows 3.x se necesita tener los siguientes programas :

Una eventual portabilidad al sistema operativo MS-DOS con Windows 3.x sería utilizar el siguiente software :

- Mesa. El Mesa es una librería para gráficos en 3D. Esta librería no es oficial de la *Silicon Graphics* y no da total garantía de funcionamiento .

Para compilar y ejecutar aplicaciones uno debe tener las librerías WinG y las extensiones Win32s para Windows. Para compilar el Mesa y sus ejemplos hay que tener el *Symantec C++* o también se puede utilizar el *Borland C++ 4.5*

-Pov-Ray

-Tcl/Tk : una opción es instalar estos Tcl y Tk para DOS/Windows ;

la segunda opción, convertir los programas de Tcl/Tk en programas escritos en lenguaje C .

El software se ejecuta en UNIX dónde a través del programa se leen los datos iniciales que luego son manipulados y visualizados en la pantalla. Los datos son almacenados en un archivo con extensión TXT .

Para ver la región que agrupa a los datos, el archivo con extensión TXT se lleva al PC y es leído por el Maple que realiza los cálculos matemáticos y estadístico dando como resultado los parámetros o ecuación de la región. Estos parámetros son ingresados el los menús que hay en el software que se ejecuta bajo UNIX y así se puede ver la región. Según, lo explicado anteriormente, para visualizar la región se debe ir a otro hardware y ocupar un software matemático, esto hace que el software de Visualización de Datos Composicionales en 2D y 3D no sea fácil de usar y haya una mayor facilidad de un posible error de

información al hacer el traspaso de los datos desde la estación de trabajo al PC y viceversa.

La solución para resolver este problema es realizar los cálculos matemáticos con la librería *MathEdge* o con alguna librería estadística o matemática que provea de operaciones con matrices y vectores, funciones trigonométricas, tabla de distribución F y de resolver ecuaciones cuadráticas.

Otra de las mejoras para el software radica en visualizar imágenes estereoscópicas, dónde el usuario por medio de lentes puede "percibir" qué está viendo en 3D. Para realizar imágenes 3D hay tres áreas claves que son : la composición de las imágenes, las técnicas de cámara en 3D y montaje&visión [19].

El software para la *Visualización de Datos Composicionales* permite manipular - seleccionar y ordenar - los datos para ver sus diferentes comportamientos que se originan al visualizarlos.

El software llamado CODA realiza una visualización de los datos composicionales en 2 dimensiones, en cambio nuestro software innova con la visualización de los datos composicionales en 3D y logra que el usuario pueda ver de distintos puntos de vista los datos, el diagrama ternario y regiones con ayuda del mouse y el teclado. El mouse entre sus funciones está en hacer girar, acercar o alejar el diagrama ternario. Las anteriores características del software permiten que el usuario realice un mejor análisis y una interpretación del comportamiento de los datos composicionales que sólo se obtiene: visualmente.

CAPÍTULO 7 BIBLIOGRAFÍA

[1] Aitchison, John. *The Satistical Analysis of Compositional Data*, Londres , Inglaterra, Chapman and Hall, 1986.

[2] Contreras Bernal , Linda. Memoria para el título en Estadística, 1994.

[3] Ousterhout, John. *Tcl and Tk Toolkit* , professional computing series, Reading, Mass: USA, Addison-Wesley, 1994.

[4] Watt, Alan. *3D Computer Graphics*, Workingham, Inglaterra, Addison-Wesley, 2da edición, 1993.

[5] Manual Persistence of Vision Ray Tracer (Pov-Ray), User's Documentation, Pov-Ray-Team, versión 2.0 1993.

[6] Harry Shamansky , "OpenGL-The Integration of Windowing and 3D-graphics", http://hertz.eng.ohio-state.edu/~hts/opengl/article.html

[7] Linas Vepstas and Chandrasekhar Narayanaswami, "Programming with OpenGL" http://www.austin_ibm.com/services/vendors/tech/aixpert/feb94/ aixpert_feb94_opengl.html

[8] OpenGL Center , http://www.sgi.com/Technology/openGL/

[9] OpenGL(tm), http://www.sgi.com/Products/Dev_environ_ds.html

[10] N.V. Yemifov, *Quadratic Forms & Matrices*, Londres, Inglaterra , Academic Press Inc.,Tercera impresión 1968.

[11] Pressman, Roger. *Ingeniería de Software*. , México, McGraw-Hill, 1993.

[12] Quercia, Valerie and O'Reilly, Tim. *X Window System. User's Guide for X11 R5*, Sebastopol, California U.S.A., O'Reilly, 1993.

[13] Sobell, Mark. *UNIX, System V. A practical guide*, Wilmington, USA; Addison-Wesley, 1995.

[14] Kernighan, Brian and Ritchie, Dennis. *El lenguaje de programación C*., México, Prentice-Hall Hispanoamericana, 1986.

[15] *OpenGL Programming Guide*, The Official Guide to Learning OpenGL, OpenGL Architecture Review Board, jackie Neider, Tomas Davis and Mason Woo, USA : Addison-Wesley , Release 1 1993.

[16] *OpenGL Reference Manual*, The Official Reference Document for OpenGL, USA: Addison-Wesley , 1992.

[17] MORAY V1.5, User Manual. Monty, the Modeller, Munich, Germany. February 1994.

[18] Welch, Brent. *Practical Programming in Tcl and Tk* , Update for Tcl 7.4 and Tk 4.0., del tipo DRAFT el 13 de Enero de 1995.

[19] Keith Rule, "3D Stereo Pairs with PovRay", PovZine, March/April, Volume 1, Issue2, 1995. Derechos registrados por Keith Rule. keithr@hevanet.com