

**TÉCNICAS DE POST-PROCESAMIENTO DE SOLUCIONES ESCALARES  
EN DOS Y TRES DIMENSIONES**

Marcelo J. Vénere  
Enzo A. Dari  
División Mecánica Computacional  
Centro Atómico Bariloche - CNEA  
Bariloche - Argentina

**RESUMEN**

En este trabajo se describe la implementación de algunas técnicas de visualización de soluciones escalares. En el caso de dos dimensiones, se tratan: graficación de líneas de nivel, zonas de nivel, y superficies espaciales; y para el caso de tres dimensiones: líneas de nivel y zonas de nivel sobre la superficie de la geometría, y sobre cortes de la misma.

Sobre cada una de estas técnicas se analiza además su eficiencia desde el punto de vista del tiempo de procesamiento y de la facilidad para dar información al usuario; se estudian algunas variantes y los principales problemas para su implementación.

**ABSTRACT**

The implementation of some techniques for visualization of scalar fields is described. Graphics of iso-lines, colour filling between iso-lines and 3-D surfaces are implemented for 2-D solutions, while for 3-D cases both iso-lines and colour-filled contour plots on the surface and on cut sections are presented.

The algorithms were chosen on basis of low CPU time and user-friendliness. A discussion on other options and on implementation difficulties follows.

## INTRODUCCIÓN

Cuando se trabaja con programas de cálculo que utilizan técnicas como Diferencias Finitas o Elementos Finitos, los resultados suelen ser un gran volumen de números, y para su análisis se hace imprescindible recurrir a la graficación. A modo de ejemplo, un problema de fluidos no muy grande en dos dimensiones puede tener 1000 incógnitas, y en tres dimensiones pueden superarse las 100000.

Las facilidades con que cuenta una terminal gráfica en la actualidad, hace factible la visualización de estos resultados en formas muy variadas, algunas de las cuales eran inviables algunos años atrás; como por ejemplo, la graficación de una superficie en el espacio iluminada desde distintos puntos de luz, o efectos de transparencias. Este es un trabajo netamente interactivo, y estas técnicas requieren un importante volumen de cálculo y la transferencia de una gran cantidad de datos entre la CPU y la pantalla. Es necesario, por lo tanto, que el equipo gráfico pueda realizar estas operaciones con una alta velocidad, que cuente con una adecuada resolución y un mínimo número de colores. Estas características sólo son satisfechas por las modernas estaciones gráficas, las que están especialmente diseñadas para estas tareas; incluso en algunas de ellas se han implementado en hardware las operaciones más costosas (traslación, rotación y eliminación de líneas y superficies ocultas).

Una consideración adicional a tener en cuenta, es la tendencia a la utilización de mallas no estructuradas en los programas de cálculo, ya que éstas permiten un mejor modelado de geometrías complejas y de singularidades; por lo cual es necesario que el post-procesador sea lo suficientemente general para poder tratarlas.

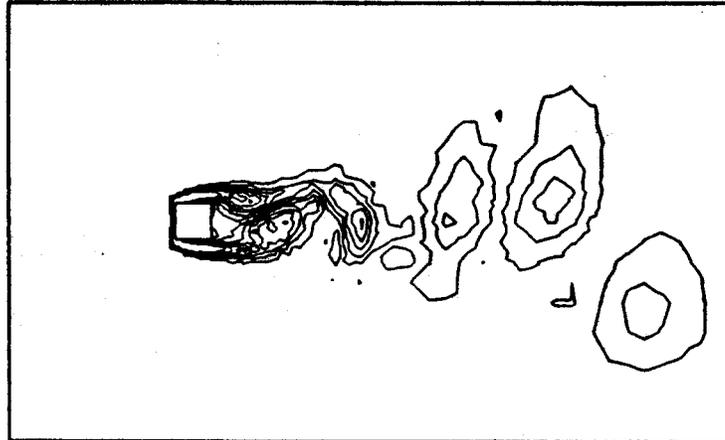
Las dos formas más utilizadas para visualizar soluciones escalares en 2-D son: la graficación de líneas de nivel y vistas espaciales de la superficie:  $Z = f(X, Y)$ . A su vez existen variantes de cada una de éstas, como por ejemplo zonas de nivel, que consiste en colorear la superficie comprendida entre dos líneas de nivel, o también combinaciones de ambas, como la graficación de zonas de nivel sobre una superficie espacial.

Con la graficación de líneas de nivel es posible obtener información precisa de la solución en cualquier punto del dominio, sin embargo, en ocasiones es difícil la interpretación global del resultado, especialmente cuando se tienen picos positivos y negativos simultáneamente. Con las superficies espaciales, en cambio, se tiene una muy buena percepción de la solución general, pero no valores numéricos precisos. Ejemplo de esto se puede ver en la Fig. 1, donde se muestra una misma solución utilizando estos dos métodos.

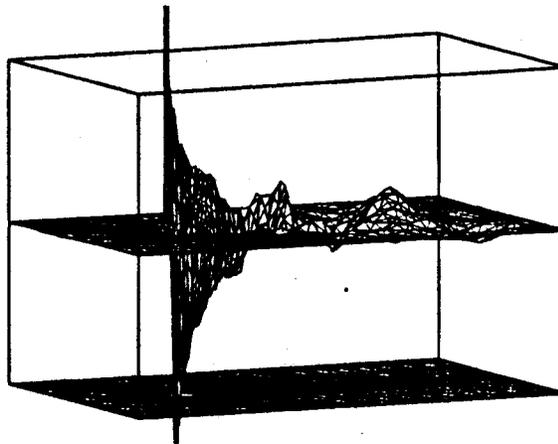
En ambos casos resulta conveniente disponer de una terminal gráfica con un número mínimo de colores, ya sea para distinguir una línea de nivel de otra como para permitir el sombreado de la superficie espacial, para dar una buena sensación de profundidad.

La visualización de soluciones escalares en 3-D es

bastante más dificultosa en comparación con 2-D, ya que sólo es posible graficar la solución sobre cortes o la superficie visible de la pieza, pero nunca sobre la totalidad de ésta. Esto lleva necesariamente a un trabajo considerablemente mayor y mucho más técnico de parte del usuario.



(a)



(b)

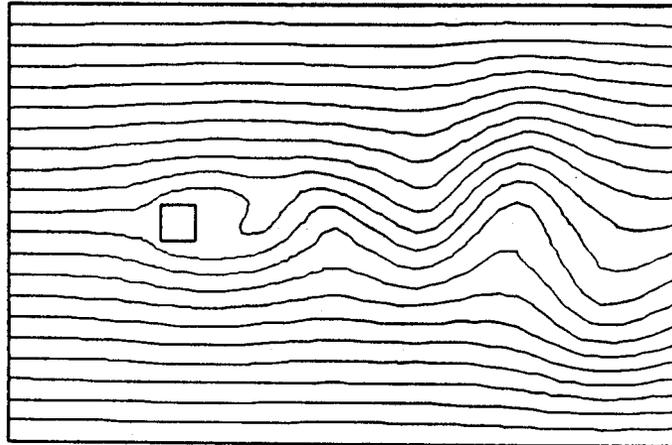
Figura 1 : Comparación entre distintos métodos de visualización de soluciones

#### SOLUCIÓN EN 2-D COMO LINEAS DE NIVEL

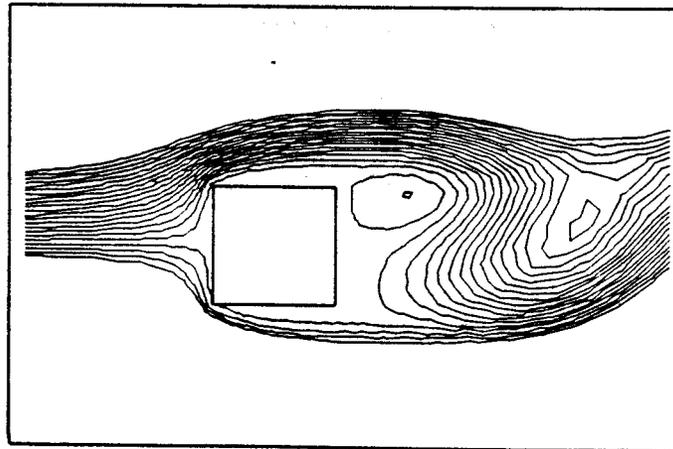
La idea es graficar las líneas sobre las cuales la solución tiene determinado valor. Es sin ninguna duda la forma más utilizada para la visualización de soluciones escalares en dos dimensiones, ya que puede dar información cuantitativa en forma precisa.

Esta forma de visualización en un marco interactivo permite en primer lugar graficar un cierto número de líneas, solicitado por el usuario, con valores equi-espaciados entre el máximo y el mínimo de la solución; de esta forma es posible tener una idea de los gradientes de la misma. Es conveniente que cada línea tenga adjudicado un color en particular y disponer de una tabla con los valores de las mismas. En segundo lugar es imprescindible poder graficar el borde de la pieza ya que graficar la totalidad de la red produce figuras excesivamente cargadas.

Además es deseable que el usuario pueda pedir la graficación de una o varias líneas de nivel con valores particulares, y que éstas se destaquen de las demás. También es muy



(a)



(b)

Figura 2 : Líneas de corriente del flujo en un canal con un obstáculo

útil poder cambiar los máximos y mínimos de la función, ya que de esa forma es posible comparar distintos gráficos o ver en detalle determinado rango de la solución. Por ejemplo en la Fig. 2 (a) se ven las líneas de corriente de un flujo pasando a través de un canal con un obstáculo, y en la Fig. 2 (b) el mismo caso pero donde se ha ampliado la zona del obstáculo y se han cambiado los extremos de la función corriente de forma de ver con detalle la zona con vórtices.

Finalmente, otra información que puede ser de utilidad es poder conocer las coordenadas y el valor de la función en determinado punto de la pieza, para ello el usuario debe poder ubicar un cursor en el punto de interés y solicitar

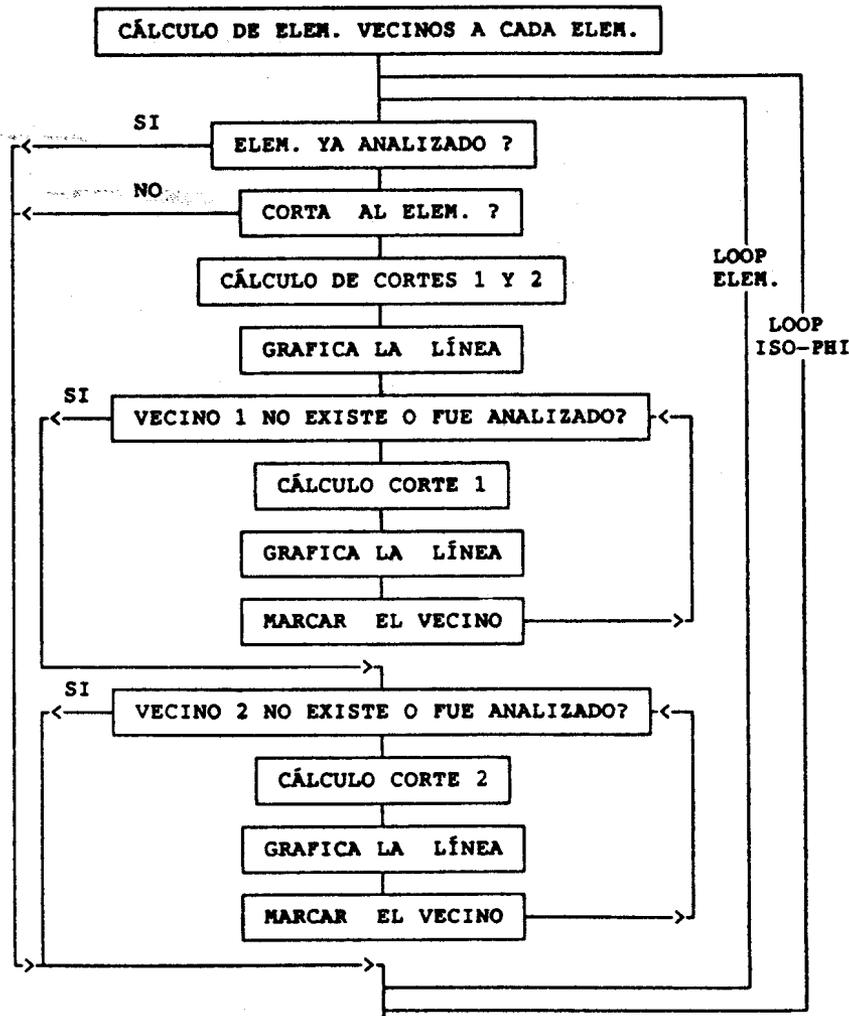


Figura 3 : Esquema para graficar líneas de nivel

esta información.

Para implementar estas opciones sólo es necesario disponer de una rutina que grafique una línea de nivel en particular, y otra que busque y grafique el borde de la red que describe la pieza. Además para poder dar el valor de la solución en un punto de la pieza va a ser necesario una rutina que dado un punto en el plano, encuentre a que elemento de la red pertenece, para luego hacer la interpolación correspondiente.

La graficación de la línea de nivel  $Z^o$  requiere la búsqueda de los elementos que cortan al plano  $Z = Z^o$ , en toda la red. Esta búsqueda requiere un número de operaciones proporcional a NEL (Nro de elementos). Es posible reducir a la mitad el número de operaciones utilizando un esquema como el que se muestra en la Fig. 3, ya que de esta forma el corte del plano con una arista se calcula una sola vez.

#### SOLUCIÓN EN 2-D COMO ZONAS DE NIVEL

Para la implementación de este método de visualización, es necesario contar con una terminal que soporte la instrucción gráfica "fill" (llenado de un polígono).

El algoritmo más simple se muestra en la Fig. 4, se puede ver que es necesario invertir el loop de elementos con el de líneas de nivel. Es posible construir algoritmos más eficientes, pero de implementación mucho más compleja.

Al igual que en líneas de nivel, resulta útil disponer de una tabla con los valores de la solución correspondiente a cada color, poder graficar una línea de nivel en particular, conocer el valor de la solución en un punto y graficar el borde de la pieza. Lo mismo vale para el cambio de los valores extremos de la solución.

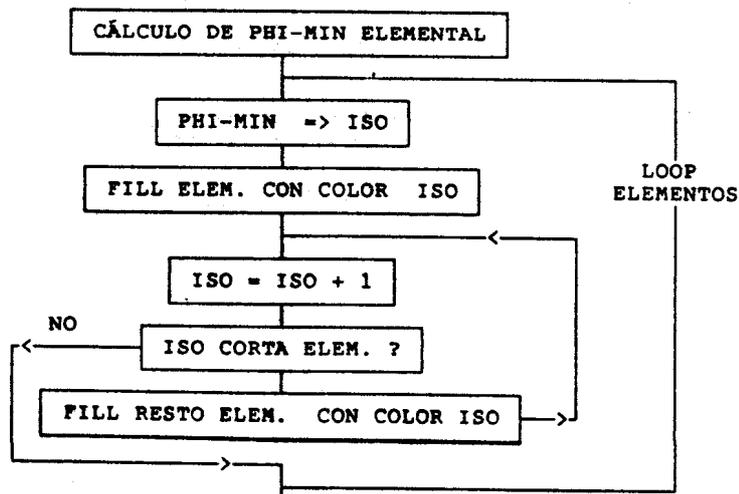


Figura 4 : Esquema para la graficación de zonas de nivel

Si la terminal grafica dispone de suficiente cantidad de colores, tomando una paleta adecuada se consiguen imagenes de muy buena calidad, en algunos casos similares a lo que sería el color real de la pieza.

#### SOLUCIÓN EN 2-D COMO SUPERFICIE ESPACIAL

Este método consiste en tomar a la solución como coordenada Z, y graficar la superficie resultante, como si se la observara desde un punto de vista dado.

Para graficar esta "red deformada" en el espacio, se debe, en primer lugar efectuar una rotación de las coordenadas de los nodos  $(X,Y,Z) \rightarrow (U,V,W)$ , de modo tal que  $(U,V)$  es el plano de la pantalla, y  $-W$  indica la profundidad. Hay dos tipos de rotación posible, una en la que el punto de vista se considera en el infinito (solo importa la dirección de vista y no cuan cerca esté del objeto), y otra en la que el objeto se deformará según la distancia de la que lo miremos. Entendemos que desde el punto de vista del post-procesamiento de cálculos numéricos, la primera es más útil.

A continuación es necesario eliminar las partes de la superficie que deberían quedar ocultas. Para ello existen varias posibilidades, discutiremos aquí dos de ellas:

-Eliminación de líneas ocultas: Dado el conjunto de aristas y caras que conforman la red, este paso consiste en la determinación de los trozos de aristas visibles.

-Eliminación de superficies ocultas: Consiste en el "pintado" de las porciones de caras visibles.

Independientemente del método empleado, va a ser útil poder graficar la terna de ejes o los planos cartesianos, o el cubo extremo que contiene a la superficie, para poder dar al usuario un marco de referencia. En nuestra experiencia la combinación del cubo extremo mas la "sombra" de la superficie (proyección al plano X-Y) produce el mejor efecto.

La solución al ser tomada como coordenada Z va a tener que ser necesariamente puesta a escala; si bien esto debería ser hecho automáticamente va a ser útil que el usuario pueda variar los extremos de la función para poder comparar distintos gráficos o ver en detalle una parte de la solución.

#### Eliminación de líneas ocultas:

El problema de la eliminación de líneas ocultas es uno de los problemas básicos que han surgido en la generación de gráficos tridimensionales por computadora.

Dado un objeto tridimensional, deseamos determinar cuáles aristas o porciones de aristas son visibles desde un punto de vista dado. Aunque el planteo del problema es muy sencillo, su implementación no lo es. Además estos algoritmos requieren un volumen importante de operaciones de punto flotante y consumen mucho de tiempo de CPU, lo cual ha impulsado el desarrollo de numerosas variantes [1,2].

La determinación de la porción de una arista tapada por una cara requiere varias operaciones de punto flotante, las

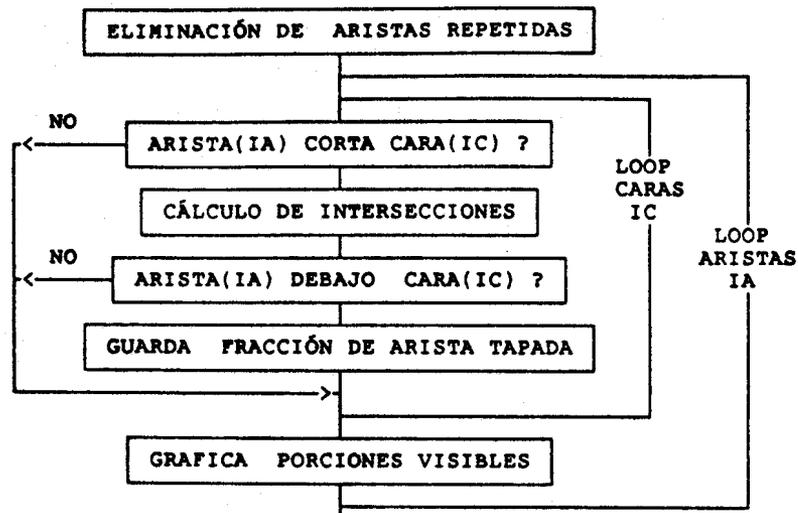


Figura 5 : Esquema 1 para la eliminación de líneas ocultas

cuales son muy costosas. Si se hiciera este "test de corte", para todas las caras, por cada arista, (esquema de la Fig. 5) el número de operaciones necesario es orden (Nro. aristas)<sup>2</sup>. Para la visualización como superficie espacial de una solución escalar en 2-D, con una red de unos 1500 elementos y 900 nodos, se tienen unas 2500 aristas y 1500 caras (elementos). Considerando que son necesarias 12 operaciones de punto flotante por cada test de corte, se llega a  $10^8$  operaciones; en una máquina que trabaje a 0.2 Mflops (e.g. VAX 11-780) resultan unos 6 minutos de CPU. El tiempo real de espera del usuario es aún mayor si se trata de un sistema de tiempo compartido, con lo cual difícilmente podría hablarse de un programa "interactivo". Además, dado el orden cuadrático del algoritmo, problemas de mayor número de incógnitas se hacen intratables en casi cualquier equipo.

Una disminución importante del tiempo de máquina puede lograrse calculando las coordenadas extremas de cada tapa en el plano de la pantalla y tenerlas almacenadas en memoria. Cuando se analiza cada arista, es posible descartar un gran número de caras; aquellas que cumplen:

$$Umáx (cara) < Umín (arista) \text{ ó } Umín (cara) > Umáx (arista)$$

o similares condiciones para las coordenadas V. Este esquema se muestra en la Fig. 6. Con este método se evita un gran número de operaciones de punto flotante, reemplazándolas con tests lógicos, lo cual hace que el algoritmo siga siendo del mismo orden, aunque su costo disminuya notablemente.

Para poder disminuir el orden del algoritmo se trata de que el loop más interno, que originalmente es de 1 a Nro. de caras, tenga un valor constante (independiente del Nro. de

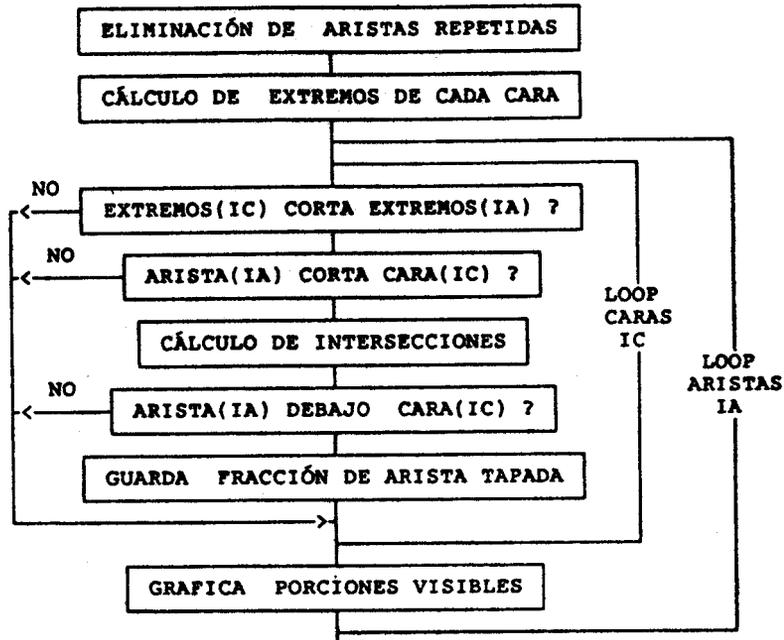


Figura 6 : Esquema 2 para la eliminación de líneas ocultas

caras). Esto se logra realizando una preclasificación de las caras de la manera siguiente (según la propuesta de LO en [2]):

-Se crea una malla regular (en el plano de la pantalla:  $U_{mín}$ ,  $U_{máx}$ ,  $V_{mín}$ ,  $V_{máx}$ ,  $N_u$ ,  $N_v$ ), tratando que el área de cada celda sea aproximadamente el área de una cara.

-Para cada cara se almacenan las celdas de la red regular con las cuales está relacionada (Aquel conjunto de celdas que formen un rectángulo que la contenga totalmente).

-Se reordena la información, para obtener las caras relacionadas con cada celda.

Una vez que se tiene esta información, el procedimiento para hallar las secciones cubiertas de una arista es:

-Determinación de las celdas de la red regular que intersectan a la arista (en nuestra implementación se toma el rectángulo que contiene totalmente a la arista)

-Test de corte de la arista con las caras relacionadas con estas celdas.

Si la red, la solución y el punto de vista son tales que las caras tienen tamaños no muy dispares, y cubren una fracción importante de la pantalla, el número de celdas promedio que intersectan a una arista es pequeño e independiente del tamaño de la red, al igual que el número de caras asociado a cada celda, con lo cual el loop más interno se redujo a un número independiente del tamaño del problema. Este algoritmo, por lo tanto es orden Nro. de aristas.

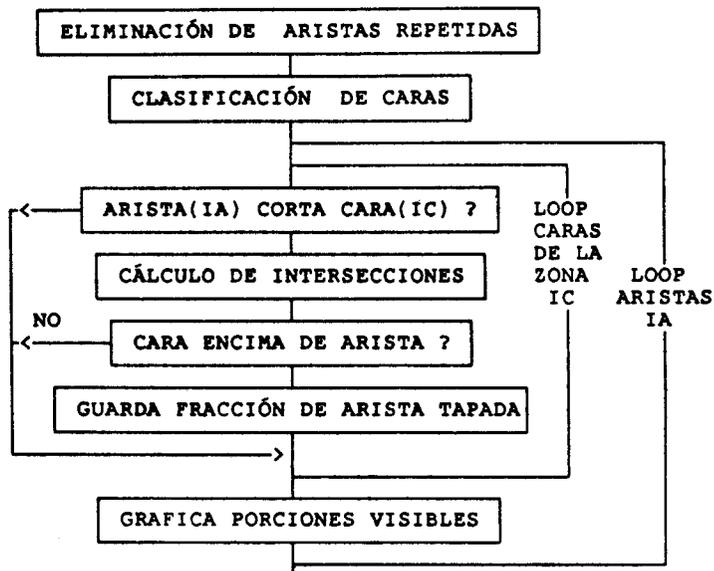


Figura 7 : Esquema 3 para la eliminación de líneas ocultas

Cabe destacar que la idea de reducir el orden de un algoritmo de búsqueda mediante una preclasificación aprovechando las propiedades de una red estructurada ya fue planteada con anterioridad en [3], donde se presenta este método aplicado a la determinación del elemento que contiene a un punto en una red de elementos finitos general.

Por último, una mejora adicional es calcular previamente la coordenada W máxima de cada cara y previamente a analizar si la cara corta a la arista, verificar que:

$$W_{\max}(\text{cara}) < W_{\min}(\text{arista})$$

Se realizaron algunas experiencias comparando estos

TABLA I : Comparación de distintos esquemas para la eliminación de líneas ocultas  
Nro. de aristas = 1900

ESQUEMA	TIEMPO (VAX 11-780)	ORDEN
1	254.4 SEG.	NA <sup>2</sup>
2	31.5 SEG.	NA <sup>2</sup>
3	14.0 SEG.	NA
4	11.0 SEG.	NA

métodos. En la Tabla I se muestran los resultados para una red bidimensional de 1200 elementos triangulares (1900 aristas). El tiempo requerido para el preprocesamiento en los casos 2, 3 y 4 está en el orden del segundo.

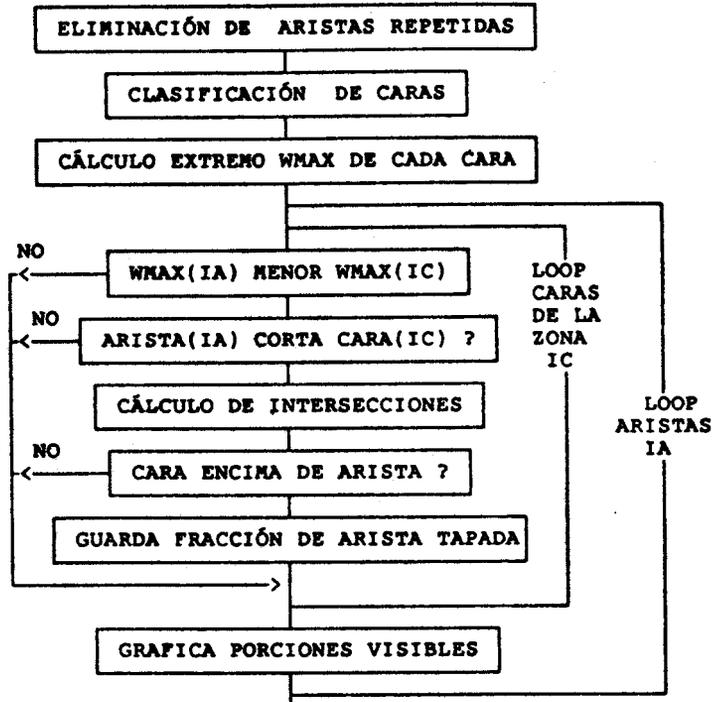


Figura 8 : Esquema 4 para la eliminación de líneas ocultas

#### Eliminación de superficies ocultas:

Un algoritmo muy simple y eficiente es graficar las caras en orden creciente de altura ( $W$ ), para ello se le adjudica a cada cara un valor  $W_c$  como promedio de los valores nodales. Si bien este método no es completamente general, funciona correctamente cuando la solución no tiene picos muy fuertes.

Pintando las caras en este orden, finalmente se obtendrá la vista deseada, con sólo las caras visibles, evitándose de esta forma la costosa eliminación de líneas o superficies ocultas.

Una alternativa interesante es pintar cada cara con un color según el ángulo que forme con un punto de luz determinado, obteniéndose así una mejor sensación de profundidad. Además en este caso se debe tener en cuenta que si el punto de luz está del otro lado de la cara respecto al punto de vista, ésta debe quedar en sombra (color oscuro). En resumen

la iluminación según el punto de luz y de vista debe ser:

- Si  $ANG(n,l) > 90^\circ$  y  $ANG(n,v) < 90^\circ \Rightarrow$  Sombra
- Si  $ANG(n,l) < 90^\circ$  y  $ANG(n,v) > 90^\circ \Rightarrow$  Sombra
- Si  $ANG(n,l) > 90^\circ$  y  $ANG(n,v) > 90^\circ \Rightarrow$  Luz según  $ANG(n,l)$
- Si  $ANG(n,l) < 90^\circ$  y  $ANG(n,v) < 90^\circ \Rightarrow$  Luz según  $ANG(n,l)$

Donde:    n : Normal positiva de la cara  
          l : Dirección de la luz  
          v : Dirección del punto de vista

Otra posibilidad es usar el mismo algoritmo que en zonas de nivel, sobre esta red ordenada y rotada, obteniéndose la superficie coloreada con bandas que indican el valor de la solución. Esto combina la buena apreciación cualitativa de la solución como superficie espacial, con la apreciación cuantitativa de la misma como zonas de nivel.

Si se dispone de suficiente cantidad de colores es posible además combinar la superficie con zonas de nivel con el efecto de iluminación, generando así gráficos muy realistas.

Cabe aclarar que estas son las implementaciones más sencillas en la graficación de superficies espaciales, con las que es posible trabajar en equipos no demasiados poderosos, existiendo posibilidades muy superiores, pero que requieren de buenas "work-stations".

#### GRAFICACIÓN DE UNA SOLUCIÓN EN 3-D SOBRE CORTES DE LA PIEZA

Esta idea es la más utilizada para visualizar soluciones en 3-D. Consiste en hacer un corte de la red tridimensional por un plano. Cualquiera sean los elementos de la red 3-D, siempre es posible reducirla a una red de tetraedros, por ello analizaremos sólo este caso.

Como el plano de corte no será en general un plano particular de la red 3-D, la red en 2-D resultante del corte tendrá elementos tipo triángulos y cuadriláteros simultáneamente, y será necesario hacer una interpolación tanto de las coordenadas como de la solución.

Una vez obtenida la red 2-D y la solución sobre la misma, es posible utilizar todas las herramientas desarrolladas para soluciones en 2-D. Puede ser atractivo que la red 2-D obtenida no sea vista necesariamente desde su normal positiva, sino desde cualquier punto de vista, para dar al usuario una idea más clara de qué corte está realizando.

#### Cortes de una red de tetradros por un plano general

La ecuación de un plano en 3-D puede considerarse como la superficie de nivel 0 de la función:

$$F(X,Y,Z) = A1 * X + A2 * Y + A3 * Z + A4$$

Con la evaluación de esta función para los 4 vértices de un tetraedro es posible decidir si el mismo es cortado por el plano. Asimismo este valor calculado para cada nodo es

proporcional a su distancia al plano, por lo cual, si una arista corta al plano, es posible calcular el valor de la función en el punto de corte por interpolación lineal:

$$s(X_c, Y_c, Z_c) = (F^2 * s(X^1, Y^1, Z^1) - F^1 * s(X^2, Y^2, Z^2)) / (F^2 - F^1)$$

Donde :  $F^1 = F(X^1, Y^1, Z^1)$

$F^2 = F(X^2, Y^2, Z^2)$

$^1$  y  $^2$  indican los puntos extremos de la arista.

Las coordenadas del punto de intersección se calculan en forma idéntica a  $s$ , por interpolación lineal.

Los elementos de la red bidimensional se van creando cuando se encuentra un tetraedro que corta al plano; los nodos, cuando se encuentra una arista cortada por el plano. Para obtener una red bidimensional de Elementos Finitos válida, los elementos provenientes del corte de dos tetraedros vecinos deben ser vecinos también en la red bidimensional, esto es, el nodo proveniente del corte de una arista de un tetraedro debe tener la misma numeración que el nodo proveniente del corte de la misma arista, pero analizada como perteneciente a otro tetraedro; para evitar esta duplicación de nodos, al crearse cada uno se debe guardar la referencia de la numeración de los nodos extremos de la arista en la red 3-D que le dio origen, verificándose antes de la creación de un nuevo nodo que no haya sido generado con anterioridad.

En la Fig. 9 se puede ver una red en 3-D y la red en 2-D resultante al realizarse un corte por un plano general.

Este método fue implementado en el post-procesador de resultados 3-D PITU3D [6], y también como un programa inde-

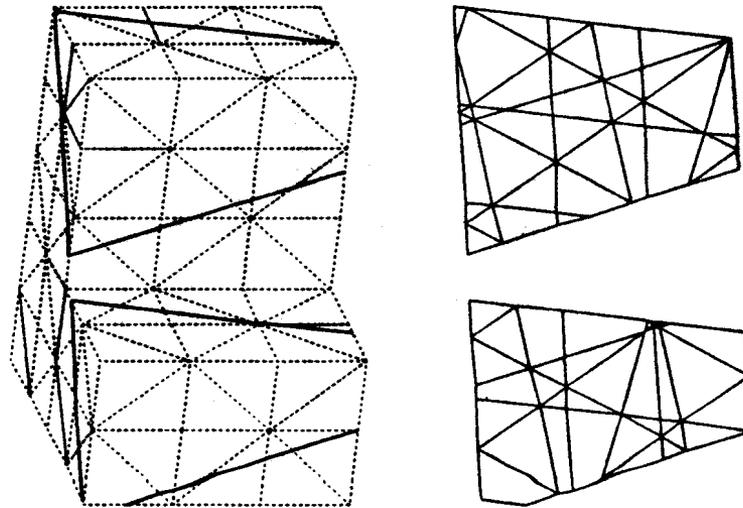


Figura 9: Red 3-D de elementos tipo tetraedros y red 2-D resultante del corte por un plano general

pendiente, que genera la red bidimensional con la solución interpolada para ser manejada por un post-procesador en 2-D estándar.

#### LÍNEAS DE NIVEL SOBRE LA SUPERFICIE DE UNA RED 3-D

La dificultad principal de esta forma de visualizar los resultados es, como en cualquier visualización de un cuerpo tridimensional, la eliminación de líneas ocultas. Sin embargo, la graficación de la superficie de una red 3-D tiene una ventaja importante con respecto a la graficación de una superficie general: se trata de una superficie cerrada, pudiéndose descartar las caras de esta que "miren hacia atrás".

El algoritmo general para la graficación de la superficie de una red 3-D consta de los siguientes pasos:

- Búsqueda de las caras externas de la red 3-D, se obtendrá una red bidimensional en el espacio, cerrada.
- Se rotan las coordenadas de los nodos según el punto de vista deseado.  $((X,Y,Z) \rightarrow (U,V,W)$ , con U,V: plano de la pantalla, -W: profundidad).
- Dado que la superficie a graficar es cerrada, se pueden descartar aquellas caras para las cuales la componente W de su normal exterior es negativa.
- Con las caras restantes se obtienen las aristas a graficar (Eliminación de aristas repetidas).
- Dadas las caras y aristas a graficar se eliminan las líneas ocultas

Las iso-líneas sobre la superficie, se calculan como en 2-D, sobre los elementos de la red bidimensional reducida. Antes de graficarlas será necesario hacer una eliminación de líneas ocultas entre los trozos de iso-líneas y las caras.

Un ejemplo de la visualización de una solución con éste método puede verse en la Fig. 10, el mismo fue obtenido con el post-procesador PITU3D.

En este caso suele ser muy útil poder borrar elementos de la red 3-D para poder ver los valores en el interior de la pieza. En la Fig. 10 se puede observar que han sido borrados todos los elementos de una esquina de la plancha.

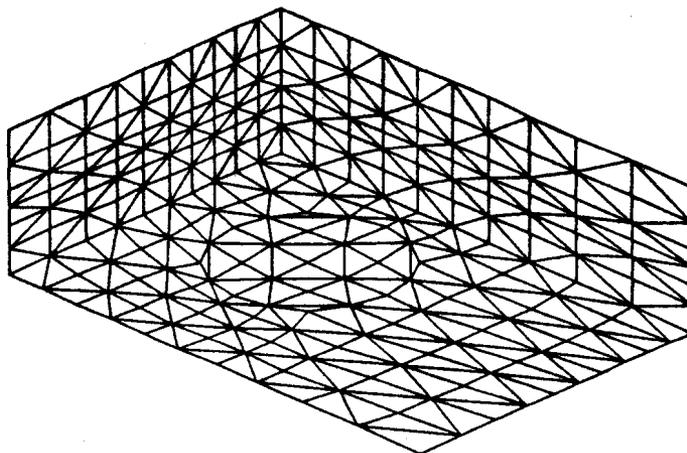
#### ZONAS DE NIVEL SOBRE LA SUPERFICIE DE UNA RED 3-D

Este método es completamente análogo a la visualización de una solución escalar bidimensional como superficie espacial con zonas de nivel, una vez determinada la red 2-D que forma la frontera de la red 3-D. En este caso puede aprovecharse el hecho de que la red bidimensional a graficar es cerrada, con lo cual pueden descartarse a-priori algunas caras.

Los pasos a seguir para visualizar los resultados de esta forma son los siguientes:

- Determinación de las caras externas de la red.
- Rotación de las coordenadas de los nodos.
- Descarte de las caras a-priori no visibles.

- Ordenamiento de las caras restantes según su altura media.
- Pintado de las caras, en forma idéntica a zonas de nivel en 2-D, y en orden de altura creciente (con lo cual se evita la eliminación de líneas ocultas).



MODELO TRIDIMENSIONAL

1850 Nodos

5848 Elementos

1	0.00000E+02
2	0.11224E+03
3	0.12450E+03
4	0.13695E+03
5	0.14930E+03
6	0.16165E+03
7	0.17400E+03
8	0.18635E+03
9	0.19870E+03
10	0.21105E+03

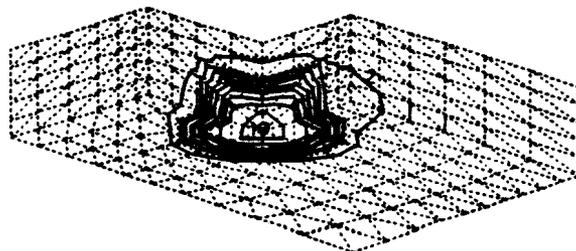


Figura 10: Distribución de temperatura en una pastilla de UO2 encastrada en una plancha de aluminio.

REFERENCIAS

- [1] J. R. RANKIN - "A geometric hidden-line processing algorithm" Computers & Graphics Vol 11, No 1, pp 11-19 (1987)
- [2] S. H. LO - "Perspective projection of non-convex polyhedra" International Journal for Numerical Methods in Engineering Vol 26, pp 1485-1506 (1988)
- [3] S. PISSANETZKY and F. G. BASOMBRÍO - "Efficient calculation of numerical values of a polyhedral function" International Journal for Numerical Methods in Engineering Vol 17, pp 231-237 (1981)
- [4] M. J. VÉNERE - "PITUCO. Post-procesador de resultados en dos dimensiones" presentado en MECOM'88, IX Congreso Latinoamericano e Ibérico sobre Métodos Computacionales para Ingeniería, Córdoba, 1988
- [5] P. BUNNING and J. L. STEGER - "Graphics and flow visualization in Computational Fluid Dynamics" AIAA-CP-85-1507 (1985)
- [6] E. A. DARI y M. J. VÉNERE - "PITU3D: Post-procesador de resultados de Elementos Finitos en 3-D. Manual del Usuario" En preparación.