Asociación Argentina de Mecánica Computacional

Mecánica Computacional Vol XXIX, págs. 3847-3864 (artículo completo)
Eduardo Dvorkin, Marcela Goldschmit, Mario Storti (Eds.)
Buenos Aires, Argentina, 15-18 Noviembre 2010

SIMULACIÓN DE FLUIDOS EN TIEMPO REAL USANDO SPH

Pablo S. Rojas Fredini y Alejandro C. Limache

International Center of Computational Methods in Engineering (CIMEC) INTEC-CONICET. Santa Fe, Argentina., http://www.cimec.gov.ar/

Palabras Clave: smoothed particle hydrodynamics, ghost-particles, interactive simulation, multi-threading

Resumen. Este trabajo presenta las principales características de una formulación computacional desarrollada por los autores y basada en el método llamado Smoothed Particle Hydrodynamics. La formulación resuelve numéricamente las ecuaciones de Navier-Stokes permitiendo la simulación de dinámica de fluidos, tanto compresibles como casi-incompresibles. El método es simple, explícito, computacionalmente rápido y apto para la computación en paralelo. Estas características, junto con el empleo de técnicas avanzadas de computación y visualización han sido utilizadas para el desarrollo de una plataforma de simulación virtual de dinámica de fluidos con la que se puede cambiar interactivamente propiedades físicas del fluido, condiciones de contorno como el movimiento de paredes o la aparición de fuerzas externas, así como también parámetros del método computacional (nivel de viscosidad artificial, tipo de integrador temporal, etc.). La mencionada interacción con el usuario ocurre en tiempo real y mientras transcurre la simulación. La velocidad de cómputo y la capacidad de interacción permiten resolver problemas de manera dinámica y con mayor rápidez, aprovechando que se puede ver y estudiar en tiempo real la respuesta del fluido a cambios de diseño o de configuración del problema físico a resolver.

1. INTRODUCCIÓN

Smoothed particle hydrodynamics (SPH) es un método de partículas que no utiliza mallas (meshless) basado en la convolución de propiedades puntuales de un campo y un kernel de interpolación. Fue inventado para simular fenómenos astrofísicos (Lucy, 1977; Gingold y Monaghan, 1977) y su utilidad fue demostrada en diversos trabajos incluyendo simulaciones de colisiones estelares (Benz, 1988, 1990; Monaghan, 1992; Frederic y James, 1999), formación de galaxias (Monaghan y Lattanzio, 1991; Berczik y Kolesnik, 1993, 1998; Berczik, 2000), agujeros negros (Lee, 1998, 2000). Inclusive la evolución del universo ha sido estudiada (Monaghan, 1990). Con el correr de los años las áreas de aplicación del método han ido creciendo y ha llegado a establecerse en la mecánica fluidos debido a dos características fundamentales: sencillez de formulación y la habilidad de incorporar complicados efectos físicos que de otra manera serían muy difíciles de simular. Liu y Liu (2003); Monaghan (1992, 2005) presentan una introducción muy completa al método y ejemplos de aplicacion en diferentes problemas.

Diversas variantes del método han sido introducidas para solucionar algunos de sus problemas inherentes y extender sus áreas de aplicación. Entre las más destacadas podemos nombrar formulaciones como ISPH (Shobeyri et al., 2007; Cumins y Rudman, 1999; Thiagarajan y Rafiee, 2009) y DSHP (Liu y Liu, 2003), en las cuales se intenta solucionar el problema de la incompresibilidad y las discontinuidades respectivamente. También se han realizado con éxito simulaciones de interacción fluidos-sólidos (Thiagarajan y Rafiee, 2009; Solenthaler et al., 2007; Schlafli, 2005). Varios modelos de viscosidad han sido propuestos y validados (Liu y Liu, 2003; Monaghan, 2005; Cleary, 1998, 1996, 1997). También distintos kernels han sido analizados en la literatura (Vesterlund, 2004; Liu y Liu, 2003; Desbrun y Gascuel, 1996).

En su concepción el método no tenía como objetivo permitir simulaciones interactivas o inclusive en tiempo real. Sin embargo el creciente poder de cálculo de las computadoras y la introducción de arquitecturas paralelas tanto en las CPU como en las GPU han permitido que SPH evolucione y permita llevar a cabo simulaciones interactivas (Surgery et al., 2003; Charypar et al., 2003; Harada et al., 2007; Losasso et al., 2008). La posibilidad de realizar simulaciones interactivas en tiempo real tiene infinidad de aplicaciones, desde simuladores para entrenamiento de profesionales (en ingeniería, medicina, etc.), videojuegos, control de procesos, diseño de prototipos, etc.

En este trabajo se describe una formulación de SPH orientada a resolver las ecuaciones de Navier-Stokes para fluidos compresibles y pseudo-incompresibles que han desarrollado los autores. Durante el proceso de desarrollo del método se construyó una plataforma de visualización que ha evolucionado hasta convertirse en una plataforma interactiva de gran utilidad. La plataforma ha permitido presentar nuevos experimentos que validan y dan soporte al método. En particular se ha procedido a realizar validaciones contra soluciones analíticas existentes en flujos en cavidades circulares. Los experimentos muestran la flexibilidad de la herramienta desarrollada.

2. FORMULACIÓN SPH

La mayoría de los conceptos presentados en esta sección son desarrollados con mayor detalle en el libro de SPH de Liu y Liu (2003), el reciente artículo de Thiagarajan y Rafiee (2009), el review de Monaghan (2005), el reporte de Cleary (1996) y el reciente trabajo de Limache y Rojas-Fredini (2010).

SPH se basa en aplicar dos conceptos fundamentales a los términos de una ecuación diferencial: representación integral y aproximación de partículas. El primero se refiere a la repre-

sentación de una función mediante una convolución con una función de suavizado. El segundo concepto consiste en la aproximación que se lleva a cabo para llevar al mundo discreto la representación integral.

2.1. Representación Integral

Una función $f(x)$ puede ser aproximada utilizando una función de suavizado W -también denominada kernel- de la siguiente manera:

$$< f(x) >= \int_V f(x')W(x-x',h)dV' \tag{1}$$

donde h es la distancia entre partículas. Si el kernel W se elige igual a la función Delta de Dirac $\delta(x-x')$ entonces la representación es exacta:

$$< f(x) >= \int_V f(x)\delta(x-x')dV' = f(x) \tag{2}$$

De la misma forma podemos aproximar la derivada espacial de una función:

$$< \nabla f(x) >= \int_V [\nabla' f(x')]W(x-x',h)dV' \tag{3}$$

Integrando por partes el lado derecho obtenemos su expresión alternativa:

$$< \nabla f(x) >= -\int_V f(x') \left[\nabla' W(x-x',h)\right] dV' + \int_S f(x')W(x-x',h)n'dS' \tag{4}$$

donde n es el vector normal unitario de la superficie de frontera S. Para aquellos puntos cuyo soporte esta contenido totalmente en el volumen V la integral de superficie de la Ec.(4) es nula y la expresión se reduce a:

$$< \nabla f(x) >= -\int_V f(x') \left[\nabla' W(x-x',h)\right] dV' \tag{5}$$

2.2. Aproximación de partículas

En SPH los fluidos se representan como un conjunto finito de partículas donde cada partícula P_i posee una masa m_i y ocupa un volumen dV_i. Por conservación de masa dicho volumen puede ser representado en función de la densidad alrededor de P_i de la siguiente manera:

$$dV_i = \frac{m_i}{\rho_i} \tag{6}$$

La aproximación de partículas consiste en calcular las representaciones integrales de la sección 2.1 como una sumatoria de las contribuciones de los volúmenes discretos alrededor de cada partícula. Entonces las versiones discretas de las Ecs. (1) y (5) evaluadas sobre P_i se calculan como:

$$< f_i >=< f(x_i) >= \sum_{j=1}^{N} m_j \frac{f_j}{\rho_j} W_{ij} \tag{7}$$

$$< \nabla_i f >=< \nabla f(x_i) >= \sum_{j=1}^{N} m_j \frac{f_j}{\rho_j} \nabla_i W_{ij}, \tag{8}$$

respectivamente donde:

$$f_j = f(x_j) \tag{9}$$

$$W_{ij} = W(x_i - x_j, h) \tag{10}$$

$$\nabla_i W_{ij} = \frac{\partial}{\partial x_i} \left[W(x_i - x_j, h) \right] \tag{11}$$

Usando la misma aproximación se pueden obtener expresiones discretas para otros operadores diferenciales como el laplaciano. A continuación se listan algunas comúnmente usadas:

$$< \nabla_i f >= -\sum_{j=1}^{N} m_j \frac{f_{ij}}{\rho_j} \nabla_i W_{ij} \tag{12}$$

$$< \nabla_i f >= \sum_{j=1}^{N} m_j \left(\frac{f_i + f_j}{\rho_j} \right) \nabla_i W_{ij} \tag{13}$$

$$< \frac{1}{\rho_i} \nabla_i f >= \sum_{j=1}^{N} m_j \left(\frac{f_i}{\rho_i^2} + \frac{f_j}{\rho_j^2} \right) \nabla_i W_{ij} \tag{14}$$

$$< \frac{1}{\rho_i} \nabla_i^2 f >= \sum_{j=1}^{N} m_j \frac{1}{\rho_j} \left(\frac{16}{\rho_i + \rho_j} \right) \left(\frac{f_{ij} r_{ij} \cdot \nabla_i W_{ij}}{|r_{ij}|^2 + \eta^2} \right) \tag{15}$$

donde η es un parámetro generalmente elegido como $h/10$ y donde -de ahora en adelante- una cantidad f con dos índices de partícula indica diferencia:

$$f_{ij} = f(x_i) - f(x_j) \tag{16}$$

mientras que una cantidad f con una barra y doble índice indica promedio:

$$\bar{f}_{ij} = \frac{f(x_i) + f(x_j)}{2} \tag{17}$$

2.3. Funciones de Kernel

Existen varias alternativas para la elección del kernel. Cuanto más parecido sea a la función Delta más correcta será nuestra aproximación ya que como se mostró en Monaghan (2005) el error crece cuadráticamente con h.

Para el presente trabajo se utilizó el kernel conocido como *spiky*. El mismo se define como:

$$W(r,h) = \frac{K_d}{h^{n_d}} \begin{cases} (2-q)^3, & 0 \le q \le 2 \\ 0, & 2 < q \end{cases} \tag{18}$$

donde en 2 dimensiones la constante de normalización es $K_d = \frac{5}{16\pi}$.

Otro kernel muy popular en la literatura es el spline cúbico que se define de la siguiente manera:

$$W(r,h) = \frac{K_d}{h^{n_d}} \begin{cases} (2-q)^3 - 4(1-q)^3, & 0 \le q \le 1 \\ (2-q)^3, & 1 \le q \le 2 \\ 0, & 2 < q \end{cases} \tag{19}$$

donde q se define como $\frac{r}{h}$. En dos dimensiones ($n_d = 2$) la constante de normalización es igual a $K_d = \frac{5}{(14\pi)}$. El soporte de ambos kernels es de $2h$.

3. APLICACION DE SPH A LAS ECUACIONES DE NAVIER-STOKES

Usando una descripción lagrangiana, la ecuación de conservación de masa resulta:

$$\frac{D\rho}{Dt} + \rho\nabla \cdot v = 0 \tag{20}$$

mientras que la ecuación de momento puede ser escrita en forma de divergencia:

$$\rho\frac{Dv}{Dt} = \nabla \cdot \sigma \tag{21}$$

o en forma laplaciana:

$$\rho\frac{Dv}{Dt} = -\nabla p_{+}\mu\nabla^2 v + \mu[1 - \frac{2}{n_d}]\nabla(\nabla \cdot v) \tag{22}$$

dónde n_d es la dimensión del espacio del problema.Ver Limache et al. (2008) para mas detalles.

3.1. Aproximación de la densidad

Si se aplica la aproximación de la Ec. (7) a la densidad se obtiene:

$$\rho_i = \sum_{j=1}^{N} m_j W_{ij} \tag{23}$$

Alternativamente, aplicando la Ec.(12) a la Ec.(20):

$$\frac{D\rho_i}{Dt} = \rho_i \sum_{j=1}^{N} \frac{m_j}{\rho_j} v_{ij}^{\beta} \frac{\partial W_{ij}}{\partial x_i^{\beta}} \tag{24}$$

la cual puede ser integrada utilizando esquemas temporales sencillos.

3.2. Aproximación de la ecuación de momento

La ecuación de momento puede ser aproximada en cualquiera de sus dos formas: Ec. (21) o Ec. (22).

Si se utiliza la Ec. (21) usando (13) para computar la divergencia se obtiene:

$$\frac{Dv_i^{\alpha}}{Dt} = \sum_{j=1}^{N} m_j \frac{\sigma_i^{\alpha\beta} + \sigma_j^{\alpha\beta}}{\rho_i\rho_j} \frac{\partial W_{ij}}{\partial x_i^{\beta}} \tag{25}$$

donde el gradiente de velocidad $\frac{\partial v^{\alpha}}{\partial x^{\beta}}$ necesario para calcular $\sigma_i^{\alpha\beta}$ se computa usando la aproximación (12)

$$\frac{\partial v_i^{\alpha}}{\partial x_i^{\beta}} = -\sum_{j=1}^{N} m_j \frac{v_{ij}^{\alpha}}{\rho_j} \frac{\partial W_{ij}}{\partial x_i^{\beta}} \tag{26}$$

Por otro lado usando Ecs. (13) y (15) para calcular el gradiente y el laplaciano de Ec. (22) se obtiene:

$$\frac{Dv_i^{\alpha}}{Dt} = -\sum_{j=1}^{N} m_j \frac{p_i + p_j}{\rho_i\rho_j} \frac{\partial W_{ij}}{\partial x_i^{\alpha}} + \sum_{j=1}^{N} m_j \frac{\mu}{\rho_j} \left(\frac{16}{\rho_i + \rho_j}\right) v_{ij}^{\alpha} \frac{r_{ij}^{\beta}}{|r_{ij}|^2 + \eta^2} \frac{\partial W_{ij}}{\partial x_i^{\beta}} \tag{27}$$

3.3. Viscosidad artificial

En SPH es común agregar un término de viscosidad artificial a la ecuación de momento. Basado en los trabajos de Cleary (1996) y Monaghan (2005) este término se define como:

$$f_i^\alpha = -\sum_{j=1}^{N} m_j \Pi_{ij} \frac{\partial W_{ij}}{\partial x_i^\alpha} \tag{28}$$

con Π_{ij} dado por

$$\Pi_{ij} = -\nu \left(\frac{v_{ij} \cdot r_{ij}}{|r_{ij}|^2 + \eta^2} \right) \tag{29}$$

dónde

$$\nu = \frac{\bar{h}_{ij}}{\bar{\rho}_{ij}} \left(\alpha c_s - \beta \frac{\bar{h}_{ij} v_{ij} \cdot r_{ij}}{|r_{ij}|^2 + \eta^2} \right) \tag{30}$$

3.4. Condicion de borde

En SPH generalmente se utilizan partículas artificiales para simular las condiciones de borde (Liu y Liu, 2003). Para evitar la interpenetración algunos autores utilizan una fuerza artificial ejercida por las partículas de borde sobre las de fluido (Monaghan, 2005; Becker y Teschner, 2007). Otros, en cambio, utilizan el incremento de presión debido a las partículas de borde (Thiagarajan y Rafiee, 2009) para satisfacer de forma natural la condición dicha. En este último caso las partículas ficticias deben tener una configuración apropiada para evitar saltos de densidad y presión en los bordes. Los autores del presente trabajo han preferido seguir el segundo camino para evitar el uso de fuerzas sin significado físico.

Existen diferentes métodos para implementar las partículas de borde. Los autores implementaron partículas ficticias reflejando las partículas reales usando conceptos similares a los utilizados en el método de las imágenes en electroestática. Es decir, se realiza un espejado a lo largo de la frontera de manera tal que las partículas fuera de la misma sean un espejo tanto en posición como en propiedades de las partículas internas. La densidad de las partículas ficticias o *ghost* se calcula como :

$$\rho_{ghost} = \rho_a$$

siendo ρ_a la densidad de la partícula real. Mientras que su velocidad v_{ghost} se fija de acuerdo a la siguiente extrapolación:

$$v_{ghost} = 2u_{wall} - v_a$$

dónde u_{wall} es la velocidad local de la pared y v_a es la velocidad de la partícula real. Si la pared se encuentra en reposo entonces: $u_{wall} = 0$.

4. FLUJO EN UNA CAVIDAD CIRCULAR

El flujo en una cavidad circular es un problema clásico de la mecánica de fluidos, también conocido como Taylor-Couette Flow. Es posible obtener soluciones analíticas del mismo en el caso de flujo laminar estacionario bidimensional por lo que resulta útil para validar métodos numéricos (Limache et al., 2008; Limache y Rojas-Fredini, 2010). En esta Sección se describen dichas soluciones las que serán usadas luego, en la Sección 6, como tests de validación del método SPH.

El problema consiste en la circulacion de un fluido entre dos cilindros concéntricos A y B cuyos radios son r_A y r_B respectivamente como se muestra en la Fig. 1. Para la pared se impone una condición de no deslizamiento.

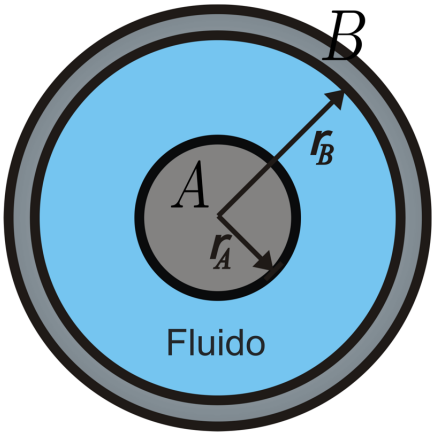

Figura 1: Flujo en una cavidad anular

4.1. Ecuación de estado

En fluidos compresibles y pseudo-incompresibles la presión y la densidad se encuentran relacionadas a través de la ecuación de estado. En este trabajo se utilizó la siguiente forma para dicha ecuación:

$$p = \kappa \left(\left(\frac{\rho}{\rho_0} \right)^{\gamma} - 1 \right) \tag{31}$$

donde κ se define en función de la velocidad del sonido c_s como:

$$\kappa = \frac{\rho_0 c_s^2}{\gamma} \tag{32}$$

Nótese que la inversa de dicha ecuación es:

$$\rho = \rho_0 \left(\frac{p}{\kappa} + 1 \right)^{1/\gamma} \tag{33}$$

Esta ecuación de estado es comúnmente utilizada en las formulaciones SPH.

4.2. Campo de velocidad

Si el cilindro exterior se encuentra rotando con una velocidad angular ω_b y el interno con una velocidad ω_a entonces el campo de velocidades predicho teóricamente (Limache et al., 2008; Limache y Rojas-Fredini, 2010) viene dado por:

$$v_r = 0 \quad ; \quad v_\theta = \frac{a}{r} + br \tag{34}$$

con

$$a = -\frac{r_B^2 r_A^2}{(r_B^2 - r_A^2)} (\omega_B - \omega_A) \quad ; \quad b = \frac{\omega_B r_B^2 - \omega_A r_A^2}{(r_B^2 - r_A^2)} \tag{35}$$

Nótese que este perfil de velocidades es independiente de la compresibilidad del fluido y de su viscosidad.

4.3. Campo de presion

Usando la Ec.(33) y la Ec.(34) en la componente r de la ecuación de momento se puede demostrar que:

- Si $\gamma \neq 1$, la solución general de la presión viene dada por

$$p = \kappa \left[\frac{\Gamma \rho_0}{\kappa} \Psi\left(r\right) + \left(\frac{p_A}{\kappa} + 1 \right)^{\Gamma} \right]^{\frac{1}{\Gamma}} - \kappa \tag{36}$$

donde

$$\Psi(r) = \frac{1}{2} \left[-a^2 \left(\frac{1}{r^2} - \frac{1}{r_A^2} \right) + 4ab \ln(r/r_A) + b^2 \left(r^2 - r_A^2 \right) \right] \tag{37}$$

y

$$\Gamma = \frac{\gamma - 1}{\gamma} \tag{38}$$

- Si $\gamma = 1$:

$$p = \kappa e^{\left[\frac{\rho_0}{\kappa} \Psi(r) + \ln\left(\frac{p_A}{\kappa} + 1 \right) \right]} - \kappa \tag{39}$$

- Por otro lado la solución para el campo de presiones si el fluido es incompresible esta dado por:

$$p = \rho_0 \Psi\left(r\right) + p_A \tag{40}$$

Para detalles de esto ver Limache y Rojas-Fredini (2010)

5. DESARROLLO DEL SIMULADOR DE FLUIDOS

5.1. Descripción general

Se desarrolló un simulador producto de la interconexión de tres componentes fundamentales: El motor de cálculo, el sistema de visualización y el sistema de interfaz de usuario. En el motor de cálculo se procesan las ecuaciones de movimiento y se llevan a cabo las integraciones temporales. Existen dos factores claves para alcanzar velocidades de cómputo del orden de tiempo real. En primer lugar es necesaria una formulación sencilla explícita y que pueda ser objeto de paralelización. Este es requerimiento fundamental y el motivo principal por el que se trabajó en una formulación tipo SPH. El segundo factor tiene que ver con la eficiencia de los algoritmos usados y las técnicas de paralelismo empleadas, junto a un correcto diseño de software.

El sistema de visualización se encuentra acoplado al motor de cálculo y permite observar el movimiento de las partículas durante todo el transcurso de la simulación. Provee facilidades para seguir determinadas partículas, constatar su soporte, y visualizar los distintos campos físicos (densidad, presión, masa, etc.) a través de modulaciones de color. Cabe destacar que todo el proceso de visualización ocurre en tiempo real y no como un post-proceso. También se incorporaron al visualizador scripts de ploteo de manera de poder obtener gráficos instantáneos generados a partir de las variables físicas. Esto permite observar por ejemplo los valores de densidad a lo largo de un corte en la geometría.

El tercer componente que debe ser tenido en cuenta es la interfaz de usuario. La posibilidad de ver resultados y cambiar parámetros a medida que la simulación se ejecuta pone en perspectiva un nuevo horizonte para la mecánica de fluidos. El usuario podría por ejemplo ver como reacciona un fluido ante cambios de viscosidad, cambios de fuerzas externas. Qué sucedería si el fluido de repente se vuelve menos denso, etc. Pero para permitirle tal grado de control a un usuario es necesaria una interfaz que posea un balance justo entre flexibilidad y sencillez de

uso. Para conseguir dicho objetivo se utilizan tecnologías de scripting embebidas en software compilado nativo para la plataforma en la cual se ejecuta.

Para la medición de tiempos y operaciones tensoriales se utilizó la librería LTensor (Rojas-Fredini y Limache, 2009). En las próximas secciones se hará una breve descripción de los aspectos más destacados de los componentes mencionados.

5.2. Scripting

Diseñar interfaces de usuario es un aspecto complejo que a simple vista puede parecer sencillo. Existen dos problemas principales: flexibilidad de uso y flexibilidad ante los cambios. Cuando se habla de flexibilidad de uso se hace referencia a que la interfaz debe ser amigable pero a su vez potente para darle la posibilidad de controlar todos los aspectos del simulador. Las interfaces de usuario tradicionales basadas en ventanas y controles gráficos son muy sencillas de usar y elegantes visualmente, pero incorporar absolutamente todos los parámetros del simulador en dichas interfaces suele ser una tarea extensa y muchas veces injustificada.

Por otro lado, en un entorno de investigación y desarrollo es muy común que los parámetros del simulador varíen. Se agregue funcionalidad o se quite. Todo esto impacta en la interfaz. De manera que si tuviera el simulador una interfaz tradicional, la misma debería ser modificada cada vez que se realiza un cambio en el simulador.

Para evitar las dos complicaciones expuestas anteriormente de una manera natural se recurre a los lenguajes de scripting. Estos lenguajes permiten realizar extensiones de programas compilados. Los lenguajes de scripting son interpretados, por lo tanto mucho mas lentos. Esto implica que se debe ser cuidadoso a la hora de utilizarlos. En particular para el simulador desarrollado se escogío Lua (PUC-Rio, 2010) que es un lenguaje muy popular en los entornos dónde la performance del scripting es crítica: Los videojuegos. El lenguaje es mucho mas acotado que otros como Python pero esto lo convierte en una alternativa muy veloz para extensión de aplicaciones.

Para implementar la extensión por scripting se incorporó una consola al simulador donde el usuario puede enviar sus comandos de script. Lua sólo se utiliza para setear variables y llamar funciones implementadas en código nativo, esto asegura que el impacto sobre la eficiencia es mínimo.

Para conectar el simulador implementado en C++ con la consola que ejecuta una máquina virtual de Lua es necesario escribir funciones para adaptar los tipos de datos y convenciones de llamadas entre ambos lenguajes. Luabind (Rasterbar-Software, 2010) es una librería basada en *expression templates* que permite generar dicho código de manera automática con unas sencillas llamadas a sus rutinas.

Esta consola permite al usuario acceder absolutamente a todos los parámetros configurables del simulador y cambiarlos con el simulador en ejecución.

5.3. Multi-threading

SPH es un método paralelizable por naturaleza, ya que se basa en cálculos locales de cada partícula. Cada partícula sólo depende de los estados del paso de tiempo anterior de sus vecinas. Es decir el algoritmo es explícito. Esto implica que el cálculo de las partículas puede ser desacoplado y llevado a cabo de manera independiente.

En el presente trabajo no se utilizó paralelismo a nivel de GPU, pero sí a nivel de CPU, explotando las nuevas características de los procesadores. El simulador utiliza $n+1$ hilos de ejecución o *thread*, donde n es la cantidad de hilos por hardware que soporta el microprocesador. El hilo principal se encarga de sincronizar los hilos de cálculo y llevar a cabo la visualización. Los n

hilos restantes se dividen de manera equitativa el trabajo de cálculo. Los hilos de cálculo esperan una señal de sincronismo dada por el hilo principal, de manera tal que dicho hilo controla la frecuencia de actualización del simulador.

La cantidad de partículas puede variar de un paso de tiempo a otro debido a las partículas *ghost*. Éstas se crean en cada paso de tiempo por lo tanto es imposible de antemano saber cuántas partículas habrá en total en el paso de tiempo siguiente. Es por ello que se necesita una política dinámica para un correcto balance de carga de las partículas en los procesadores. Las partículas se alojan en un espacio de memoria común a todos los hilos de ejecución, y cada hilo es autosuficiente en el sentido que reconoce automáticamente el segmento que le corresponde calcular. Por supuesto esta división se hace en conjuntos disjuntos de manera tal que un hilo de ejecución nunca modifique una partícula que no tiene asignada. Esto permite disminuir al máximo los mecanismo de sincronicación necesarios propios del ambiente multitarea tales como las barreras y los *locks*.

El diagrama de sequencia de la Fig. 2 muestra el loop principal de la aplicación. Allí se observa como el hilo principal crea de manera asíncrona los hilos de ejecución. Luego el hilo principal una vez en el loop envia una señal (*Signal()*) a través de una barrera y queda en espera de que terminen su trabajo los n hilos de cálculo. Cuando los hilos de cálculo reciben la señal realizan su tarea y al terminar envian una señal de finalización (*SignalFinish()*) también mediante una barrera. Este proceso se repite para cada paso de la simulación.

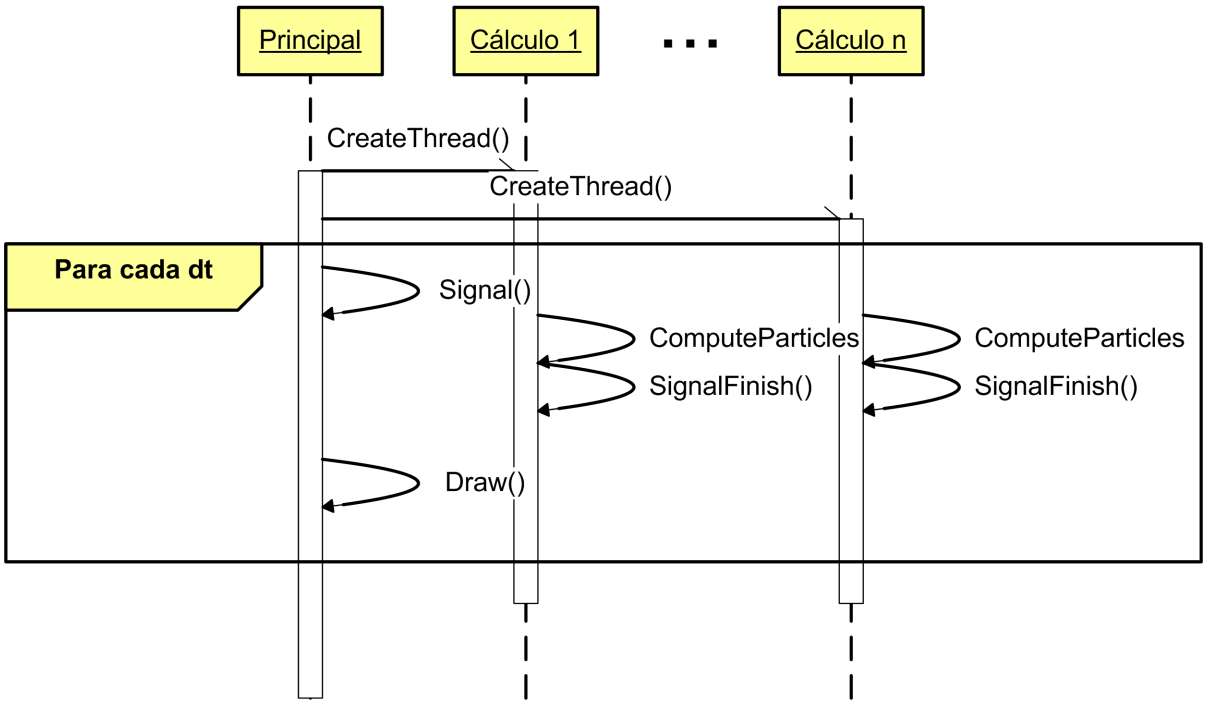

Figura 2: Diagrama de secuencia del loop principal

Para la implementación se utilizó la popular librería multiplataforma *boost::thread* perteneciente a las librerías boost (Boost.org, 2010) para la funcionalidad de *threads*. Los hilos no se crean ni se destruyen en cada paso de tiempo ya que es una tarea costosa.

5.4. Ordenamiento Espacial

Si bien el método es *meshless* para realizar a cabo el cálculo de una partícula es necesario conocer cuáles son sus vecinas espaciales que se encuentran dentro del soporte del kernel. Cómo se realiza esta tarea es determinante para la performance del simulador.

Existen distintas alternativas de ordenamiento espacial como *quadtress*, *btrees*, spatial hashing, etc. Spatial hashing fue la técnica implementada en el simulador. La misma consiste en dividir el espacio en celdas de tamaño constante dónde cada partícula pertenece a una y solo una celda y una celda puede contener más de una partícula. Este tipo de estructura es útil cuando se tiene una distribución uniforme en el dominio de las partículas. En la Fig. 3 se puede apreciar un ejemplo de dicha estructura.

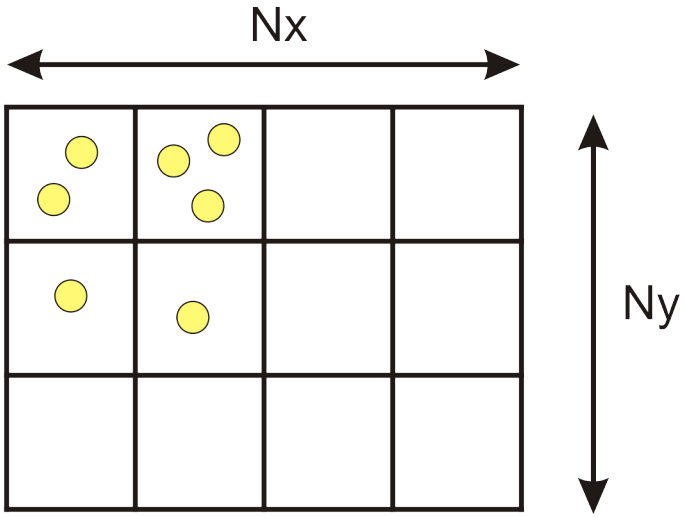

Figura 3: Grilla del spatial hashing

Si bien es cierto que esta estructura debe ser reconstruida en cada paso de tiempo, la tarea de rescontruirla no es demasiado costosa. Para saber en qué celda se ubica una partícula simplemente se calcula un producto y luego se asignan punteros. No hay operaciones de movimiento de memoria. Por el contrario, cada celda de la grilla se comporta como la cabecera de una lista simplemente enlazada. Es decir, en las celdas se almacenan un puntero a la primera partícula, la cual a su vez enlaza a la siguiente. Este comportamiento se muestra en la Fig. 4 .

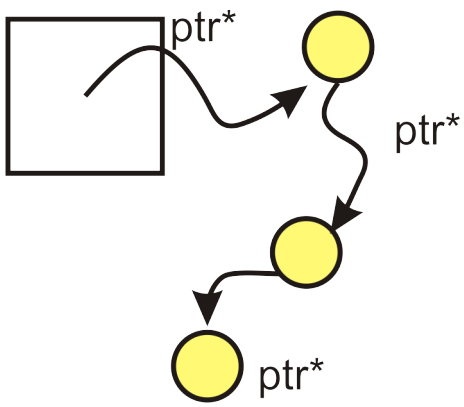

Figura 4: Listas simplemente enlazadas

No se llevaron adelante estudios de coherencia de caché pero los experimentos no mostraron síntomas de que dicho problema sea un cuello de botella en el simulador.

6. EXPERIMENTOS

A continuación se mostrara por completitud una comparación entre la solución analítica y la solución de SPH para el problema de la cavidad circular introducido en la sección 4. Luego se mostrarán ventajas y ejemplos de interactividad de la plataforma desarrollada.

6.1. Comparación con la solución analítica

Para el experimento numérico se eligió $C_s = 5{,}0$, $\gamma = 1$. La cantidad de partículas en el soporte de cálculo fue de 25 con una densidad deseada de $\rho_0 = 1000$. El total de partículas reales fue de 5000 más las *ghost*. La viscosidad dinámica elegida fue $\mu = 100$.

El cilindro interior se configuró a una velocidad $\omega_a = -1$ y el exterior a $\omega_b = 2$. La condición de pared como se dijo anteriormente fue de no deslizamiento. En la Fig. 5 se puede observar el simulador en plena ejecución.

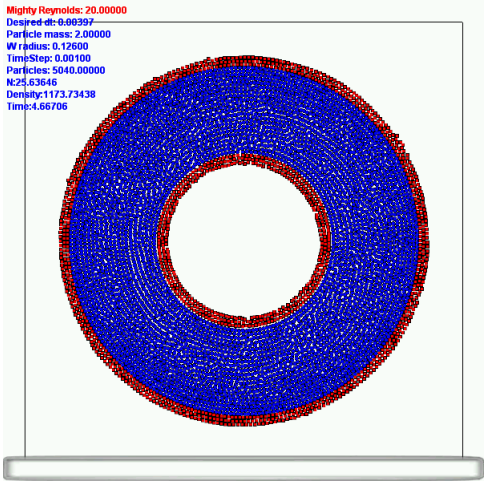

Figura 5: Simulación cavidad circular

En la Fig. 6 se retrata la comparación entre el perfil de SPH y el perfil analítico para la velocidad tangencial a lo largo de un corte radial. Se puede apreciar la excelente concordancia entre ambos.

De la misma manera la Fig. 7 muestra el campo de presiones analítico y el obtenido mediante SPH. Nótese que en promedio la solución coincide con la curva analítica, la dispersión en la presión es una consecuencia natural de la oscilación de las partículas.

6.2. Variación de gravedad

Para mostrar la flexibilidad de la herramienta desarrollada es que se llevó a cabo el siguiente experimento: Se utilizó la geometría de la cavidad circular presentada anteriormente y se llenó con partículas un cuarto de la misma aproximadamente. De esta manera es posible visualizar cómo reacciona el fluido antes los cambios de parámetros. Las partículas se vieron sometidas a una fuerza de gravedad con $g = -9{,}8$ y se las dejó alcanzar el reposo inicialmente, como se muestra en la Fig. 8a. Los parámetros del fluido son $\mu = 100$, $C_s = 5$ y la cantidad de partículas fue aproximadamente 500.

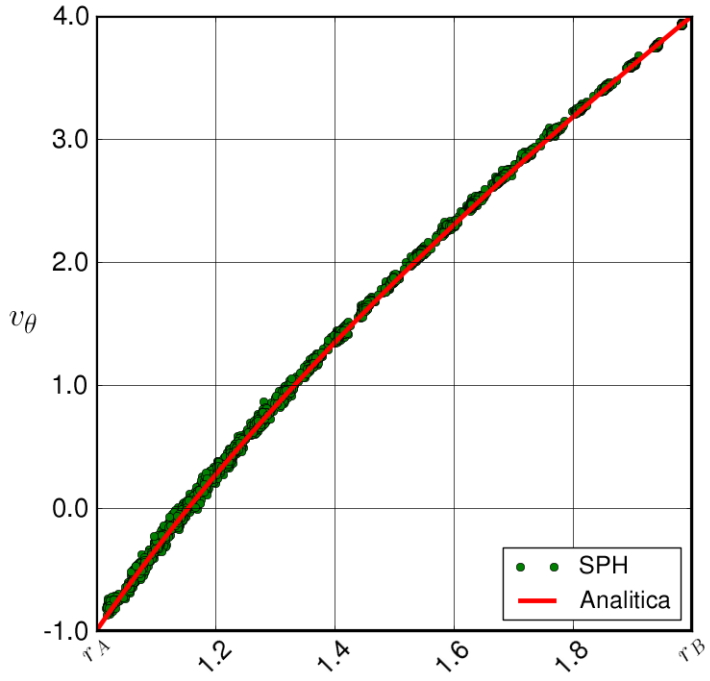

Figura 6: Perfil de velocidad

Cuando las partículas alcanzaron el reposo, mediante la consola de scripting integrada, se invirtió el sentido de la gravedad g de -9.8 a 9.8. En la Fig. 8 se puede observar cómo el fluido reacciona ante el cambio de la fuerza externa y se reacomoda hasta alcanzar nuevamente el equilibrio bajo las nuevas condiciones impuestas. Este cambio se realizó sin detener ni pausar la simulación permitiendo ver interactivamente las reacciones.

6.3. Variación de compresibilidad

Siguiendo la misma metodología del experimento previo, se llenó tres cuartos aproximadamente de la cavidad circular con partículas. Los parámetros iniciales fueron : $\mu = 100$ y $C_s = 10$. Recuérdese que C_s es una medidad de la compresibilidad del fluido. El fluido fue sometido a una gravedad $g = -9{,}8$.

Una vez que el fluido alcanzó el reposo (Fig. 9a) se cambió la velocidad del sonido C_s para hacerlo más compresible con $C_s = 5$. Debido a la acción de la gravedad, el fluido se comprimió. En la Fig. 9f se puede observar la configuración de equilibrio final. Allí se puede apreciar que la misma cantidad de partículas ocupan menos volumen que antes debido a que se encuentran más comprimidas.

La secuencia de la Fig. 9 muestra algunos estados intermedios entre las dos posiciones de equilibrio. La compresibilidad hace oscilar verticalmente a las partículas hasta alcanzar la posición de equilibrio. En las Fig. 9d y 9e se puede observar dicho fenómeno.

El cambio de compresibilidad se llevó a cabo interactivamente. Todas las imágenes fueron capturadas del simulador desarrollado, y no fueron modificadas de ninguna forma mediante un postproceso. Allí se puede observar en la parte inferior de la ventana la consola utilizada para interactuar.

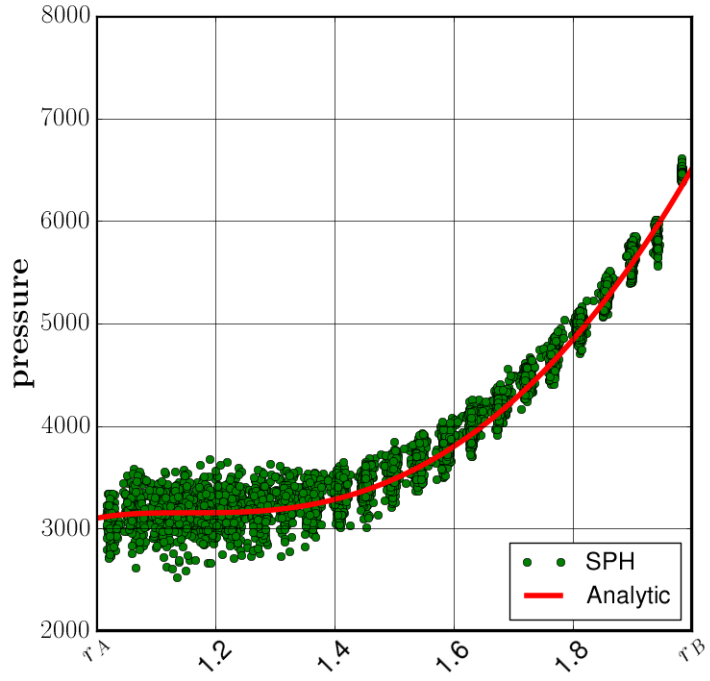

Figura 7: Perfil de presión

7. CONCLUSIONES

Se presentó una formulación basada en SPH para simular fluidos compresibles y pseudo-incompresibles. La misma es explícita y simple. Dichas características permiten realizar los cálculos en tiempo real y desarrollar herramientas de simulación que brinden resultados instantáneos.

Se llevó a cabo la simulación del problema de un fluido en una cavidad circular y se validó contra los resultados analíticos mostrando muy buena correspondencia. También se presentó una herramienta desarrollada para hacer simulaciones interactivas y se mostraron dos casos dónde se variaban la compresibilidad y la gravedad respectivamente. Se mostró cómo el usuario puede cambiar el comportamiento del simulador y cómo el fluído reacciona ante dichos cambios.

Se introdujeron nuevos conceptos en el desarrollo de un simulador de fluidos en tiempo real. Estos conceptos incluyen técnicas para realizar el paralelismo de los cálculos explotando las nuevas arquitecturas de CPU y técnicas de optimización más generales para poder llegar a la meta propuesta. También se mostraron las ventajas de la utilización de lenguajes de scripting en el ámbito de la simulación y las limitaciones del mismo.

REFERENCIAS

Becker M. y Teschner M. Weakly compressible sph for free surface flows. In J. Popovic y D. Metaxas, editores, *SIGGRAPH Symposium on Computer Animation*. 2007.

Benz W. Applications of smoothed particle hydrodynamics (sph) to astrophysical problems. *Computer Physics Communications*, 48:97–105, 1988.

Benz W. Smoothed particle hydrodynamics: a review. *In Numerical Modeling of Non-linear Stellar Pulsation: Problems and Prospects*, 1990.

Berczik P. Modeling the star formation in galaxies using the chemodynamical sph code. *Astronomy and Astrophysics*, 360:76–84, 2000.

Berczik P. y Kolesnik I.G. Smoothed particle hydrodynamics and its applications to astrophysical problems. *Kinematics and Physics of Celestial Bodies*, 9:1–11, 1993.

Berczik P. y Kolesnik I.G. Gas dynamical model of the triaxial protogalaxy collapse. *Astronomy and Astrophysical transactions*, 16:163–185, 1998.

Boost.org. Boost-libraries. www.boost.org, 2010.

Charypar D., Müller M., y Gross M. Particle-based fluid simulation for interactive applications. In M. Lin y D. Breen, editores, *SIGGRAPH Symposium on Computer Animation*. 2003.

Cleary P.W. Sph technical note #8. new implementation of viscosity: Tests with couette flows. Informe Técnico, CSIRO Division of Mathematics and Statistics, 1996.

Cleary P.W. Modelling confined multi-material heat and mass flows using sph. In *Inter. Conf. on CFD in Mineral & Metal Processing and Power Generation CSIRO*, páginas 1–23. 1997.

Cleary P.W. Modelling confined multi-material heat and mass flows using sph. *Applied Mathematical Modelling*, 22:981–993, 1998.

Cumins S.J. y Rudman M. An sph projection method. *Journal of Cumputational Physics*, 152:584–607, 1999.

Desbrun M. y Gascuel M. Smoothed particles: A new paradigm for animating highly deformable bodies. *Eurographics Workshop on Computer Animation and Simulation, EGCAS '96*, páginas 61–76, 1996.

Frederic A.R. y James C.L. Smoothed particle hydrodynamics calculations of stellar interactions. *Journal of Computational and Applied Mathematics*, 109:213–230, 1999.

Gingold R.A. y Monaghan J.J. Smoothed particle hydrodynamics: Theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181:375–389, 1977.

Harada T., Koshizuka S., y Kawaguchi Y. Real-time fluid simulation coupled with cloth. *Theory and Practice of Computer Graphics*, 2007.

Lee W.H. Newtonian hydrodynamics of the coalescence of black holes with neutron stars ii, tidally locked binaries with a soft equation of state. *Monthly Notices of the Royal Astronomical Society*, 308:780–794, 1998.

Lee W.H. Newtonian hydrodynamics of the coalescence of black holes with neutron stars iii, irrotational binaries with a stiff equation of state. *Monthly Notices of the Royal Astronomical Society*, 318:606–624, 2000.

Limache A. y Rojas-Fredini P. Validation and comparison of sph methods against analytical solutions of annular cavity flows. *International Journal for Numerical Methods in Fluids*, submitted, 2010.

Limache A., Sanchez P., Dalcin L., y Idelsohn S. Objectivity tests for navier-stokes simulations: the revealing of non-physical solutions produced by laplace formulations. *Computer Methods in Applied Mechanics and Engineering*, 197:1703–1759, 2008.

Liu M. y Liu G. *Smoothed Particle Hydrodynamics: A Meshfree Particle Method*. World Scientific Publishing Co. Pte. Ltd., 2003.

Losasso F., N. J., Kwatra, y Fedkiw R. Two-way coupled sph and particle level set fluid simulation. *IEEE TVCG*, 14:797–804, 2008.

Lucy L.B. Numerical approach to testing the fission hypothesis,. *Astronomical Journal*, 82:1013–1024, 1977.

Monaghan J. Smoothed particle hydrodynamics. *Annu. Rev. Astron. Astrophys.*, 30:543–74, 1992.

Monaghan J. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68:1703–

1759, 2005.

Monaghan J.J. Modeling the universe. *Proceedings of the Astronomical*, 18:233–237, 1990.

Monaghan J.J. y Lattanzio J.C. A simulation of the collapse and fragmentation of cooling molecular clouds. *Astrophysical Journal*, 375:177–189, 1991.

PUC-Rio. Lua. http://www.lua.org, 2010.

Rasterbar-Software. Luabind. http://www.rasterbar.com/products/luabind.html, 2010.

Rojas-Fredini P.S. y Limache A.C. Ltensor: a high performance library. http://code.google.com/p/ltensor/, 2009.

Schlafli J. *Simulation of Fluid-Solid Interaction*. Tesis de Doctorado, VMML. University of Zurich, 2005.

Shobeyri G., Farhadi L., y Ataie-Ashtiani B. Modified incompressible sph method for simulating free surface problems. *Fluid Dynamics Research*, 40:637–661, 2007.

Solenthaler B., Schläfli J., y Pajarola R. A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds*, 18:67–82, 2007.

Surgery F.V., Müller M., Schirm S., y Teschner M. Interactive blood simulation. *Technology and Health Care*, 12:25–31, 2003.

Thiagarajan K.P. y Rafiee A. An sph projection method for simulating fluid-hypoelastic structure interaction. *Comput. Methods Appl. Mecha. Engrg.*, 198:2785–2795, 2009.

Vesterlund M. *Simulation and Rendering of a Viscous Fluid Using Smoothed Particle Hydrodynamic*. Tesis de Doctorado, VRLab, Umea University, 2004.

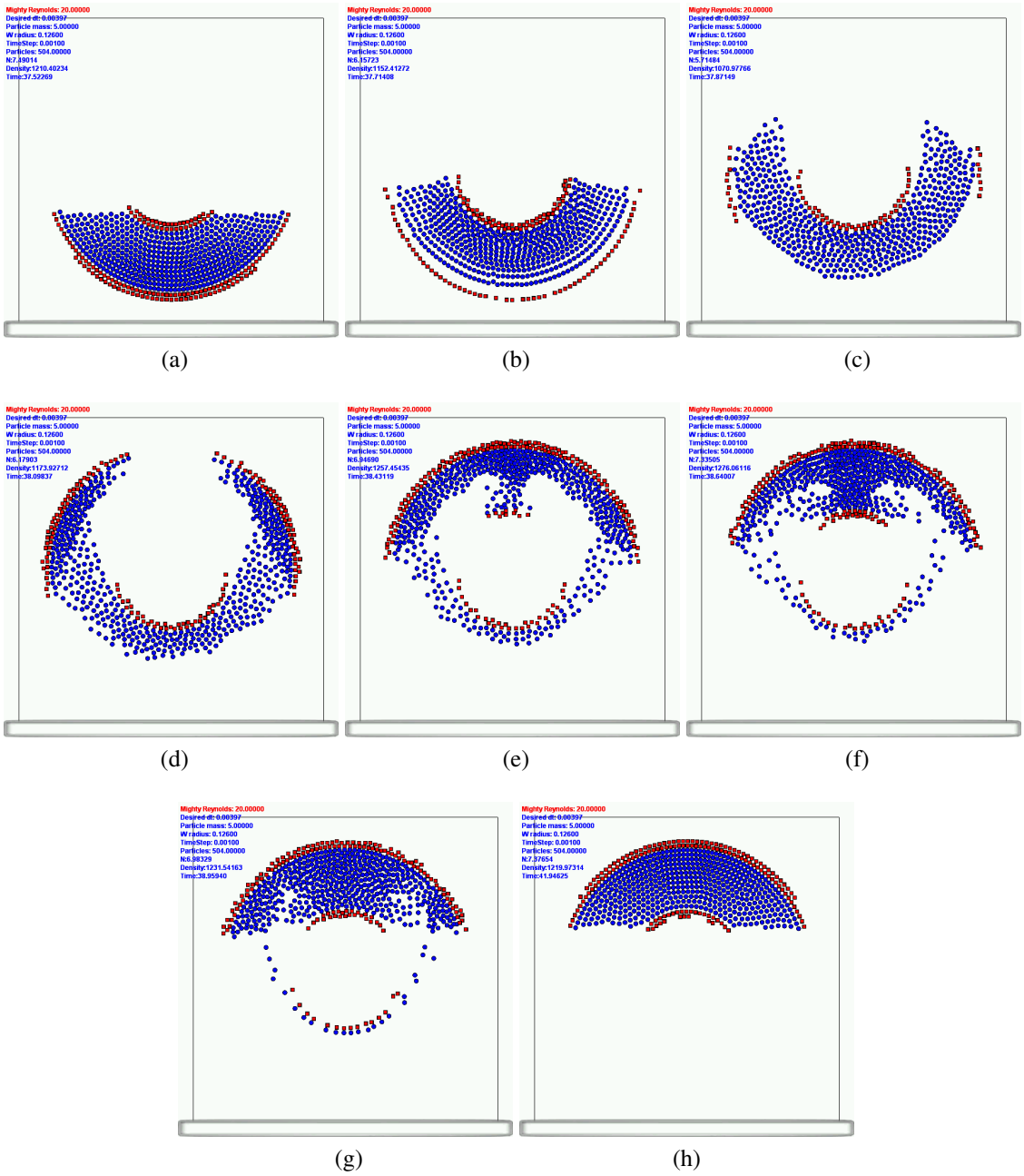

Figura 8: Secuencia cambio de gravedad

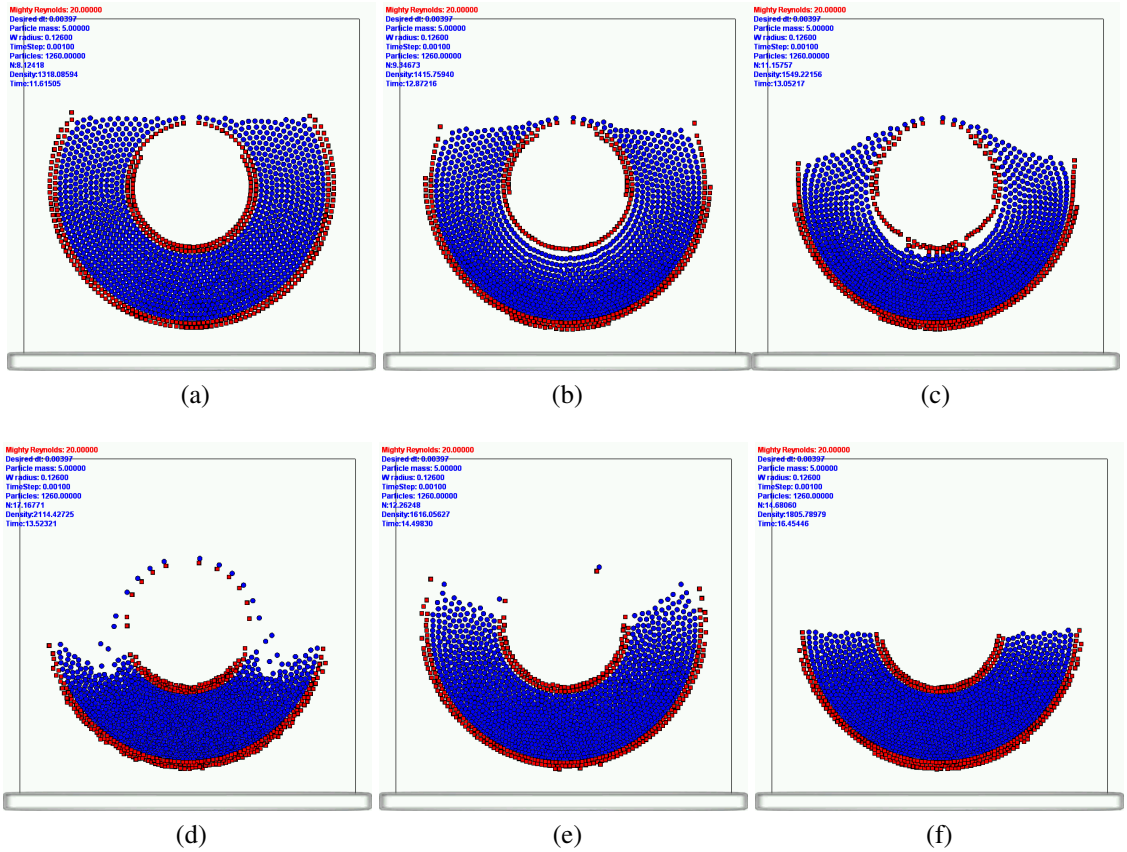

Figura 9: Secuencia cambio de compresibilidad