Fluid Animation Using Sketching, Diffusion-Reaction and Lattice Boltzmann Models

S. Judice, and G. Giraldi

***Abstract*—This work presents a methodology for two-dimensional fluid animation that is based on sketching techniques, along with diffusion-reaction equations and fluid simulation using the Lattice Boltzmann Method (LBM). In the initial stage, the framework converts the drawings made by the users into streamlines. From these lines it is generated a local velocity field, which is widespread throughout the fluid domain through diffusion-reaction techniques. The velocity field obtained at the end of the process is used to initialize the fluid simulation performed with LBM method. Our target applications are visual effects and computer graphics. In this sense, we present visual results involving theoretical and user interface issues which demonstrate the potential of the methodology developed for computational fluid animation.**

***Index Terms*—Fluid Animation, Sketch-Based Modeling, Diffusion-Reaction, Lattice Boltzmann Method.**

I. Introdução

NESTE trabalho, nosso foco é animação via técnicas de modelagem baseadas em física e simulação computacional. No caso particular de cenários envolvendo animações de fluidos, a utilização de métodos em dinâmica de fluidos computacional (DFC) tem se mostrado um recurso valioso para os animadores, diminuindo o custo, bem como o tempo de produção de filmes [1]. Os métodos tradicionais encontrados na literatura para animação de fluidos usando modelos de DFC, são fundamentados nas equações de Navier-Stokes, com técnicas de discretização baseadas em diferenças finitas implícitas e explícitas, bem como em métodos Lagrangianos, tais como SPH (*Smoothed Particle Hidrodynamics*) [2]. Recentemente, o método de *Lattice Boltzmann* (LBM) mostrou-se uma técnica alternativa e promissora para animação e simulação de fluidos [3,4]. Este método não é baseado na discretização das equações que representam um sistema contínuo, uma vez que o mesmo trabalha com uma versão simplificada da equação de Boltzmann para obter o comportamento macroscópico desejado [4].

Este trabalho foi desenvolvido no Laboratório Nacional de Computação Científica (LNCC), Petrópolis, Brasil e financiado pelo CNPq e pelo Ministério de Ciência, Tecnologia, Inovações e Comunicações do Brasil.

S. F. P. P. Judice: Doutora em modelagem computacional. Atualmente faz Pós-Doutorado na Universidade de Calgary, Canadá (siciliajudice@gmail.com).

G. A. Giraldi: Pesquisador do LNCC, Petrópolis, Brasil (gilson@lncc.br).

Corresponding author: G. A. Giraldi.

Do ponto de vista das aplicações em computação gráfica, uma questão importante é a etapa de inicialização do fluido, ou seja, como definir o campo de velocidades no instante inicial. Uma inicialização apropriada permite que o resultado visual desejado seja obtido mais eficientemente a partir dos dados numéricos gerados na simulação. Seguindo esta filosofia, em [5,6] apresentamos a ideia de permitir que um usuário forneça um rascunho do comportamento do fluido, de maneira que ele possa informar graficamente as características desejadas para o escoamento. Este rascunho, denominado *sketch*, é considerado um ponto de partida, que deverá passar por etapas de processamento, compostas por filtragem de curvas e técnicas de difusão-reação, com o objetivo de gerar um campo inicial com as propriedades matemáticas necessárias para ser usado no modelo de simulação do fluido. Em [7] encontramos outra metodologia baseada em *sketching* para inicialização e controle de fluidos, a qual inspirou o sistema apresentado em [8] que implementa métodos para rascunhar vasos e doenças vasculares usando uma interface intuitiva. Porém, estes métodos não utilizam técnicas de difusão-reação para garantir que o campo inicial tenha as propriedades necessárias, não utilizam LBM para a simulação e não apresentam nenhum estudo sobre a preservação da topologia inicial definida pelo usuário.

Neste trabalho, partimos da metodologia descrita em [5,6], e apresentamos: (a) Interface com o usuário; (b) Conceitos teóricos envolvidos na conversão do *sketching* em linhas de corrente; (c) Preservação de singularidades pelos métodos de difusão-reação; (d) Comparação entre dois métodos de difusão-reação. O foco de aplicação está nas áreas de efeitos visuais e computação gráfica. Neste sentido, são apresentados resultados visuais que demonstram a potencialidade da metodologia apresentada e permitem discutir os conceitos teóricos envolvidos.

O texto está organizado da seguinte forma. Na seção II, são descritos os modelos utilizados. Em seguida, na seção III, apresentamos a metodologia desenvolvida para animação de fluidos. Na seção IV descrevemos o sistema computacional implementado. Finalmente, os resultados e conclusões são discutidos nas seções V e VI, respectivamente.

II. Fundamentação teórica

Do ponto de vista teórico, a metodologia para animação de fluidos deste trabalho está baseada em técnicas de *sketching*, filtragem de curvas, LBM e equações de difusão-reação.

A modelagem baseada em *sketching* é o processo onde especifica-se um modelo através de desenhos livres realizados pelo usuário [9]. O modelo gerado pode ser bidimensional ou

tridimensional, de acordo com o problema a ser tratado. O processo inicia-se com a aquisição de traços feitos pelo usuário no dispositivo de entrada (tela do computador, *tablet*, etc.). Neste trabalho, cada traço é representado por uma poligonal, definida por uma sequência de pontos $c=(P_1,P_2,\cdots,P_N)$, onde $c(t)=P_t\in\Re^2$. O processo de aquisição sofre de problemas como a presença de ruído e amostragem irregular devido a imperfeições do movimento do cursor e variação da velocidade com que o usuário faz o desenho. Assim, surge a necessidade de um processo de filtragem para melhorar a qualidade destes dados [10]. No caso de filtros lineares, as técnicas são baseadas na convolução da curva *c* com um núcleo *h*, definida pela equação:

$$(c*h)(t)=\sum_{m=-k_1}^{k_2}c(t-m)h(m). \qquad (1)$$

onde $h=(h_{-k_1},h_{-k_1+1},\cdots,h_{-1},h_0,h_1,\cdots,h_{k_2-1},h_{k_2})$, com $h(t)=h_t\in\Re$.

A sequência *h* define o tipo de filtro, o qual, neste trabalho, pode ser o filtro da média ou subdivisão Chaikin, ambos utilizados para suavização, embora o segundo produza também o aumento da quantidade de pontos na curva (detalhes em [11], seção 3.2). O modelo implementado utiliza também o filtro de super-amostragem, definido pela seguinte operação não-linear: dados dois pontos $P_i,P_{i+1}\in c$ e $0<\delta<1$, são gerados novos pontos através da expressão:

$$Q_j=\frac{P_{i+1}-P_i}{\|P_{i+1}-P_i\|}j\delta+P_i,\quad j=1,2,\cdots K, \qquad (2)$$

onde K é o maior natural tal que $K\delta<\|P_{i+1}-P_i\|$ e $\|\cdot\|$ representa a norma Euclidiana.

No método LBM implementado [4,11], o domínio de interesse é discretizado em uma malha regular (*lattice*) com células quadradas (Fig. 1), e o meio é representado por distribuições de partículas, denotadas por f_i, $i=0,1,\ldots,8$, indicando movimento de partículas com velocidades ao longo das oito direções da malha, mostradas na Fig. 1, bem como partículas em repouso, no caso de f_0.

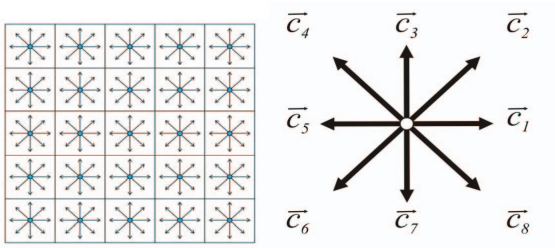


Fig. 1. Esquerda: Malha usada pelo LBM. Direita: Direções de movimento nas arestas da malha.

Além disso, há colisão entre partículas e quantidades físicas de interesse, associadas aos nós da malha, são atualizadas a cada passo de tempo da simulação. A dinâmica em cada nó *P* depende dos nós vizinhos no instante de tempo anterior. Neste trabalho, a dinâmica do LBM é dada pela equação geral:

$$f_i(P+\Delta P,t+\Delta t)=$$
$$f_i(P,t)A(\tau)+f_i^{eq}(\rho,\vec{u})B(\tau)+H_i(t,\Delta t), \qquad (3)$$

onde *A*, *B* são funções do parâmetro de relaxação τ, f_i^{eq} (função de equilíbrio) e H_i são funções inerentes ao modelo físico a ser simulado, $\rho,\vec{u}$ denotam a densidade e velocidade macroscópicas, Δt é o passo de tempo da simulação e ΔP é a distância entre dois nós vizinhos na malha. A partir das distribuições f_i e das direções $\vec{c}_i$ (Fig. 1) são computadas as quantidades macroscópicas de densidade ρ e velocidade $\vec{u}$, usando as equações:

$$\rho(P,t)=\sum_{i=0}^{8}f_i(P,t);\quad \rho(P,t)\vec{u}(P,t)=\sum_{i=1}^{8}\vec{c}_i f_i(P,t). \qquad (4)$$

Neste trabalho, utilizamos o LBM para simular fluidos e difusão-reação, cujos modelos serão denotados por LBMF e LBMDR, respectivamente. O método LBMF utilizado é dado pela equação (3), onde $\tau=1.2$, $A(\tau)=1-\tau^{-1}$, $B(\tau)=\tau^{-1}$ e $H_i=0$. A distribuição de equilíbrio é dada por [4]:

$$f_i^{eq}(\rho,\vec{u})=\rho\omega_i\left[1+3\vec{u}\cdot\vec{c}_i+\frac{9}{2}(\vec{u}\cdot\vec{c}_i)^2-\frac{3}{2}\vec{u}\cdot\vec{u}\right],$$

onde $\omega_0=4/9$, $\omega_i=1/9$ para $i=1,3,5,7$ e $\omega_i=1/36$ se $i=2,4,6,8$.

O modelo LBMDR é definido pela expressão geral (3), com $\tau=1.2$, $A(\tau)=1-\tau$ e $B(\tau)=\tau$. Porém, neste caso, as distribuições de partículas possuem duas componentes $f_i\to\vec{f}_i=(f_i^x,f_i^y)$, o mesmo ocorrendo com as demais funções do método, as quais são computadas por: $\vec{H}_i(t,\Delta t)=-\Delta t\omega_i\|\vec{u}_0\|_2^2(\vec{u}-\vec{u}_0)$ e $\vec{f}_i^{eq}(\rho,\vec{u})=\omega_i\vec{u}(P,t)$. Os valores dos pesos são: $\omega_0=4/9$, $\omega_i=1/9$ para $i=1,2,3,4$ e $\omega_i=1/36$ se $i=5,6,78$ [12].

A inicialização do LBMF e LBMDR é feita percorrendo toda a malha e atribuindo um valor inicial a cada uma das $f_{i's}$, em cada nó da malha. Neste trabalho, o campo de velocidades inicial $\vec{u}(P,0)$ é computado a partir do *sketch*, e a densidade inicial $\rho(P,0)$ é fornecida pelo usuário. A partir

destes campos são computadas as distribuições $f_{i's}$ em $t=0$, pela expressão:

$$f_i(P,0) = f_i^{eq}(\rho(P,0),\vec{u}(P,0)). \quad (5)$$

Além do LBM, outro modelo fundamental para nossa metodologia é o GVF (*Gradient Vector Flow*), o qual foi proposto em [13] como um termo de força externa para modelos de contorno ativo utilizados em processamento de imagens. No caso $2D$, o GVF pode ser formulado como o campo vetorial $\vec{u}(x,y) = (u_1(x,y), u_2(x,y)),\ (x,y) \in \Re^2$, que é a solução estacionária da equação de difusão-reação:

$$\frac{\partial \vec{u}}{\partial t} = \mu \Delta \vec{u} - (\vec{u} - \vec{u}_0)\|\vec{u}_0\|_2^2, \quad (6)$$

onde $\vec{u} = \vec{u}(x,y,t)$, $\vec{u}_0 = \vec{u}(x,y,0)$ é a condição inicial, μ é a constante de difusão, e $\Delta = (\partial^2/\partial x^2, \partial^2/\partial y^2)$ é o operador Laplaciano, aplicado a cada componente de $\vec{u}$ separadamente. Em [13], a equação (6) é resolvida numericamente usando diferenças finitas centrais, no espaço e no tempo. Neste trabalho, consideramos o GVF e o LBMDR como alternativas para difusão-reação para fins de comparação entre um modelo macroscópico, baseado em equações diferencias parciais (GVF) e um modelo que trabalha na escala mesoscópica [4,12], no caso, o LBMDR.

III. Metodologia Desenvolvida

A Fig. 2 mostra o fluxo de dados entre as etapas da metodologia para animação de fluidos desenvolvida. Na primeira etapa, o usuário desenha um rascunho (*sketch*) do fluxo desejado, usando algum dispositivo de entrada conveniente. Cada curva desenhada pelo usuário é interpretada como uma linha de corrente do campo. Portanto, no pré-processamento, precisamos estabelecer um critério para a definição do campo de velocidades sobre o *sketch* do usuário, de forma eficiente e matematicamente correta.

Fig. 2. *Pipeline* da metodologia para animação de fluidos bidimensionais.

Ao desenhar a curva, os pontos que a representam são armazenados em sequência, na ordem definida pelo traçado do usuário, definindo uma poligonal $c = (P_1, P_2, \cdots, P_N)$. Desta forma, o campo vetorial tangente é computado através da sequência de pontos sobre a curva sendo determinado pela expressão: $\vec{v}_i = \alpha_i(P_{i+1} - P_i)$, com $i = 1,2,\ldots,N-1$ e α_i sendo um fator de escala para controlar a intensidade da velocidade. Se a curva desenhada pelo usuário for uma curva fechada, calculamos a média de todas as velocidades sobre a curva e usamos o valor obtido como velocidade constante para todos os pontos. Se a curva for aberta, computamos α_i usando uma Gaussiana para garantir uma distribuição suave das velocidades. O resultado do pré-processamento é utilizado para instanciar os métodos de difusão-reação. Os algoritmos correspondentes utilizam como estrutura subjacente uma malha regular *M*, com resolução $R_x \times R_y$, previamente definida, e espaçamento $\Delta x = \Delta y = 1$. Assim, o campo de velocidades sobre cada linha de corrente deve ser projetado sobre a malha *M*. Para isto, aplicamos o filtro de super-amostragem (equação (2)). A partir deste resultado, podemos determinar todas as células cortadas pela linha de corrente e definir a velocidade inicial nos nós das mesmas, como ilustra a Fig. 3.

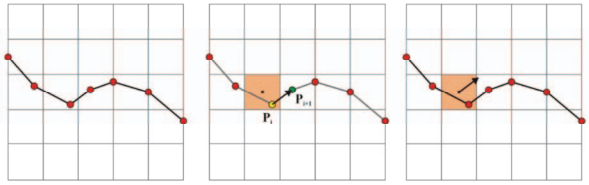


Fig. 3. Projeção do campo de velocidades sobre a malha regular. (a) Linha de corrente. (b) Identificação das células. (c) Projeção da velocidade.

O processo acima gera um campo de velocidades não-nulo somente na vizinhança das linhas de corrente. Esta descontinuidade é fonte de instabilidade numérica para os métodos de difusão-reação. Uma forma de contornar este problema é aplicar convolução Gaussiana *2D* sobre o campo obtido acima, de forma a gerar um novo campo com a suavidade necessária. O campo obtido será utilizado como o campo vetorial $\vec{u}_0 = \vec{u}(P,0)$, nas expressões (5) ou (6), dependendo do método de difusão-reação escolhido. Esta etapa é importante para espalhar o campo de velocidade para todo o domínio, gerando um campo $\vec{u}$ diferenciável e que preserva a topologia definida pelo traçado do usuário. Finalmente, o campo de velocidades $\vec{u}$ é utilizado para inicializar o LBMF para simulação do fluido.

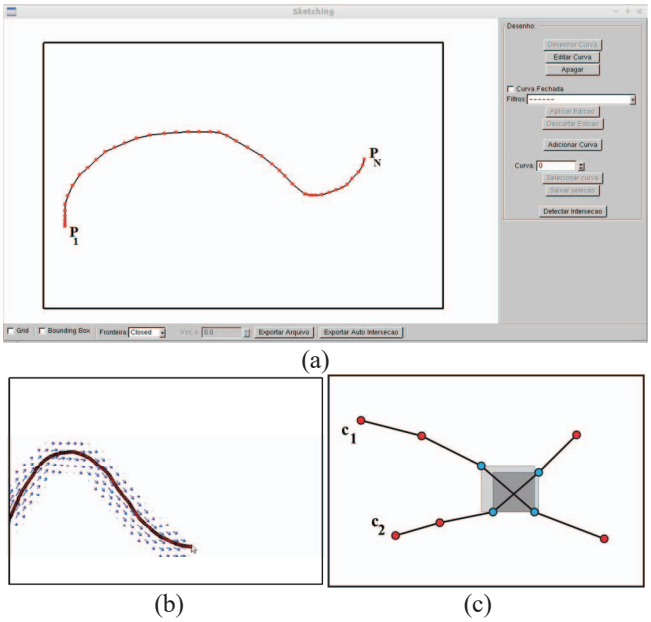
IV. Sistema Computacional

Para o desenvolvimento de software foi utilizada a linguagem de programação C/C++, além do sistema Paraview [14] para a visualização dos dados numéricos, GLUT para sistemas de janelas e GLUI para interface gráfica. A modelagem baseada em *sketch* foi implementada usando a biblioteca descrita em [15].

A Fig. 4(a) mostra o protótipo desenvolvido neste trabalho. Neste protótipo, o usuário utiliza o *mouse* como dispositivo de entrada para desenhar as curvas que vão representar as linhas de corrente de interesse. Neste processo, o usuário pressiona o botão esquerdo do *mouse* no local onde deve se localizar o ponto inicial da curva (P_1), em seguida mantém o botão pressionado e arrasta o *mouse* até obter a geometria desejada. Ao liberar o botão, o sistema interpreta que uma curva foi criada.

Porém, existe um ponto importante relacionado a topologia de campos vetoriais. A Fig. 4(a) mostra uma

situação típica, onde o usuário desenhou uma curva aberta, cujos pontos extremos P_1 e P_N pertencem ao interior do domínio, delimitado pelo retângulo da Fig. 4(a). Se o campo de velocidades for não-nulo nestes pontos, então teremos um campo descontínuo, uma vez que a velocidade é nula fora da linha de corrente. Porém, a equação (6) necessita de campos de classe C^2. Uma forma de acomodar esta situação seria definir P_1 e P_N como singularidades do campo de velocidades, ou seja, $\vec{u}(P_1)=\vec{u}(P_2)=\vec{0}$. Entretanto, tal interpretação pode gerar um campo totalmente diferente daquele que o usuário tem em mente.



(a)

(b) (c)

Fig. 4. (a) Interface gráfica do protótipo desenvolvido. (b) Pré-visualização do campo de velocidades ao longo do traçado usuário. (c) Identificação de interseção de poligonais.

Outra possibilidade seria prolongarmos cada extremidade do desenho do usuário até a borda do domínio do fluido mais próxima, atendendo as condições de fronteira nos pontos de interseção entre o prolongamento e a borda correspondente. Do ponto de vista visual, este procedimento é mais intuitivo para o usuário sendo aquele adotado neste trabalho.

Entretanto, é interessante fornecer para o usuário um *feedback* do campo de velocidades que será gerado. Para isto, o sistema faz uso do método GVF (equação (6)) para difusão-reação, com o intuito de fornecer uma pré-visualização do campo de velocidades que está sendo desenhado pelo usuário. A Fig. 4(b) mostra um exemplo dessa pré-visualização. Nela, temos uma configuração do desenho de uma linha de corrente. À medida que o usuário arrasta o *mouse* para desenhar a curva, é possível ver na vizinhança do traço o desenho de vetores indicando o comportamento que o campo de velocidades irá assumir, fornecendo um retorno visual ao usuário do campo que ele está desenhando. Isto é feito, resolvendo a equação (6) em algumas células na vizinhança da curva.

Outro evento que precisa ser tratado são intersecções (ou auto-intersecções) entre curvas. Sejam então duas curvas distintas representadas por seus respectivos traços, a saber, c_1 e c_2, como ilustrado a Fig. 4(c). Para cada segmento de reta da curva c_1, formado por dois pontos consecutivos, define-se a caixa envolvente que contém esse segmento e compara-se com todas as caixas envolventes de todos os segmentos de reta que compõem a curva c_2, com o objetivo de identificar caixas envolventes que se sobrepõem. Uma vez identificada duas caixas envolventes sobrepostas (região em cinza na Fig. 4(c)), temos então dois segmentos de reta com potencial para intersecção, sendo feita a verificação de intersecção através da equação paramétrica da reta. Na interface o protótipo provê, um botão chamado *Detectar Intersecção*, que ao ser pressionado, executa todo o processo de detecção explicado. As interseções detectadas deverão ser consideradas como pontos singulares (velocidade nula) para evitar inconsistência na definição do campo de velocidades. O usuário deve definir o tipo de singularidade desejada: ponto de sela ou foco estável, por exemplo [16].

Uma vez resolvidas questões topológicas e tratamento de fronteira sobre o traço do usuário, a interface implementada permite aplicar sobre a curva os filtros descritos na seção II, para obter um resultado mais suave (filtro da média ou subdivisão *Chaikin*) ou aumentar a quantidade de pontos (equação (2)), ou mesmo editar a curva para fazer alterações locais usando recursos da interface gráfica.

O protótipo permite a escolha da condição de fronteira. O usuário pode optar por realizar uma simulação em uma condição de fronteira fechada. Neste caso, a simulação de fluido irá tratar a fronteira com condição de não escorregamento em todo o domínio (velocidade nula na fronteira). O usuário pode ainda escolher uma condição de fronteira periódica. Neste caso, a condição de fronteira para a simulação de fluido será a de não escorregamento para as paredes do topo e da base do retângulo na Fig. 4(a), e condição periódica para as paredes laterais. Na versão atual, o método de difusão reação disponível é o GVF, definido pela equação (6).

Resumindo, para gerar uma animação, o usuário deve: (a) usar o *mouse* para traçar as linhas que formam o *sketch* do fluxo; (b) Editar o traçado para aplicar filtros ou executar alterações locais nas curvas; (c) Executar o GVF para computar o campo inicial $\vec{u}(P,0)$; (d) Definir a densidade inicial $\rho(P,0)$; (e) Efetuar a simulação do fluido usando o LBMF, com inicialização dada pela equação (5); (f) Exportar resultado para visualização no Paraview [14].

V. Resultados Computacionais

Mostraremos os resultados da metodologia desenvolvida para animação de fluidos, representada na Fig. 2, com os seguintes objetivos: (a) Comparar os métodos de difusão-reação implementados; (b) Analisar o comportamento da metodologia, do ponto de vista da preservação e/ou criação de

singularidades e geração de vórtices (regiões em rotação no fluido).

A condição de parada para os métodos de difusão-reação (GVF e LBMDR) e simulação de fluidos (LBMF) é dada por:

$$\max_{i,j}\left\|\vec{v}_{i,j}^{k+1}-\vec{v}_{i,j}^{k}\right\|_2<\varepsilon, \tag{7}$$

onde ε é um limiar previamente estabelecido, *k* indica um instante da simulação e (i,j) é um nó genérico da malha *M* utilizada.

A etapa de difusão-reação é inicializada com o campo vetorial oriundo do pré-processamento do *sketch*, como explicado na seção III. A inicialização do fluido é feita pela equação (5), onde o campo $\vec{u}(P,0)$ é oriundo da etapa de difusão-reação e $\rho(P,0)=1.0$, constante. As imagens de linhas de corrente foram geradas pelo programa Paraview [14]. Para todos os exemplos, as malhas utilizadas pelo GVF, LBMDR e LBMF têm resolução 220×140, sendo utilizada a condição de fronteira de não deslizamento (velocidade nula).

Para comparar o resultado obtido pelos métodos de difusão-reação, vamos usar nossa metodologia em quatro exemplos distintos, a saber: (a) Uma linha de corrente simples; (b) Uma linha de corrente fechada; (c) Um *sketch* envolvendo singularidade; (d) Múltiplas linhas de corrente. A Fig. 5 mostra o resultado dos primeiros passos do processamento do *sketch,* com aplicação dos filtros da média subdivisão Chaikin para suavizar o traço inicial e o filtro de super-amostragem para aumentar a densidade dos pontos. Na Fig. 5, o vetor $\vec{u}$ indicar o sentido do desenho inicial.

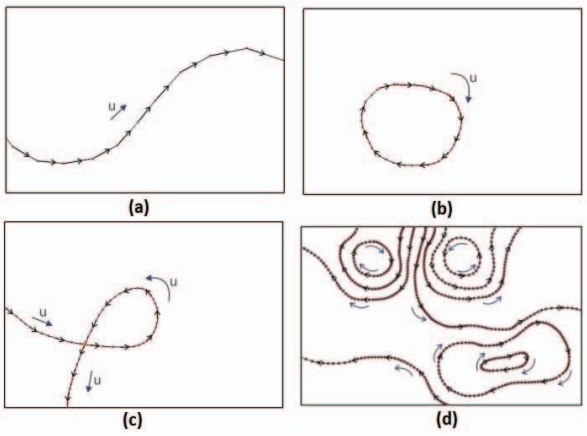


Fig. 5. (a) Linha de corrente simples; (b) Linha de corrente fechada; (c) Problema de singularidade; (d) Múltiplas linhas de corrente.

Nas Figs. 5(a),(c),(d) foi realizado também o prolongamento do traçado inicial do usuário para que os pontos terminais das linhas de corrente abertas estejam sobre a fronteira. A velocidade sobre cada curva é definida de acordo com a metodologia explicada na seção III. Vale ressaltar que os testes realizados em [11] mostram que a metodologia é robusta com relação à variação do parâmetro α_i, usado para escalar a intensidade do campo de velocidades.

No caso da Fig. 5(c), o ponto de intersecção é detectado e definido como um ponto singular na curva, um ponto de sela no caso. O mapeamento das linhas de corrente da Fig. 5 para a malha dos métodos de difusão-reação é feito como explicado na seção III, onde a projeção das velocidades sobre a malha é ilustrada na Fig. 3. O resultado é suavizado via convolução Gaussiana, para ser utilizado na etapa de difusão-reação.

Primeiramente, vamos comparar os resultados do GVF e do LBMDR. A condição de parada de ambos os métodos é dada pela expressão (7), e nestas simulações usamos o parâmetro $\varepsilon=10^{-4}$. No modelo do GVF o parâmetro de difusão usado na equação (6) tem valor $\mu=0.2$.

Os resultados dos métodos de difusão-reação para as linhas de corrente da Fig. 5, podem ser visualizados na Fig. 6, onde o azul mais escuro indica velocidades mais baixas. Os campos das Figs. 6(a)-(d) são as soluções do GVF correspondentes às linhas de corrente das Figs. 5(a)-(d), respectivamente. Analogamente para as Figs. 6(e)-(h), porem estas são soluções obtidas pelo LBMDR.

A observação visual mostra que a topologia definida pelo usuário no *sketch* foi preservada tanto pelo GVF quanto pelo LBMDR, não sendo possível notar diferenças significativas entre os dois métodos. No caso da linha de corrente fechada (Fig. 5(b)), o usuário não indicou explicitamente a singularidade no desenho. Porém, do ponto de vista da topologia de campos vetoriais, a única possibilidade de acomodar um campo vetorial C^2 neste caso será gerando uma região de baixa velocidade no centro da curva fechada, a qual deve conter uma singularidade. Os resultados apresentam este comportamento, uma vez que regiões mais escuras possuem baixa velocidade, estando assim em conformidade com a topologia esperada neste caso. É importante observar que, dentro do conhecimento dos autores, o problema de preservação de singularidades por métodos de difusão-reação é um campo novo de pesquisa na área de equações diferenciais parciais. Como um primeiro passo nesta direção, demonstramos em [11] que, dada a solução estacionária da equação (6) com $\mu\neq0$, então uma singularidade em *P* é preservada pelo GVF se $\Delta\vec{u}(P)=0$.

A Tabela 1 mostra os resultados obtidos na análise do número de iterações e do tempo computacional da etapa de difusão-reação. As linhas 'S' e 'T' na tabela correspondem, respectivamente, ao número de passos e ao tempo de CPU, em segundos, necessários para cada simulação atingir a condição de parada. Os Casos 1, 2, 3 e 4 citados na Tabela 1, correspondentes aos resultados da Fig. 6, do topo para a base, respectivamente. As sub-colunas 'GVF' e 'LBMDR' dizem respeito aos resultados obtidos pelo GVF e pelo LBMDR. Embora a complexidade dos dois métodos seja da ordem da resolução da malha $(O(R_x\cdot R_y))$, como demonstrado em [11], a simples observação da Tabela 1 mostra que o GVF foi mais eficiente que o LBMDR nestes testes.

Dos quatro casos da Fig. 5, os três primeiros tratam de exemplos com apenas uma curva desenhada. O último exemplo é o caso que apresenta múltiplas linhas de corrente.

Tendo isto em mente, podemos observar que os dados obtidos para os três primeiros casos estão dentro de uma mesma faixa de valores, com pequenas variações entre si, tanto para a análise de número de iterações (faixa de valores

TABELA I
CUSTO COMPUTACIONAL DO GVF E LBMDR: NÚMERO DE ITERAÇÕES (S) E TEMPO DE CPU EM SEGUNDOS (T)

	Caso 1		Caso 2		Caso 3		Caso 4	
	GVF	LBMDR	GVF	LBMDR	GVF	LBMDR	GVF	LBMDR
S	216	250	253	287	232	262	298	492
T	0,391	1,248	0,450	1,383	0,471	1,488	0,462	2,742

de 0 a 300 passos) quanto para a análise de tempo computacional (faixa de valores de 0 a 1,5 segundos). Entretanto, a faixa de valores para o quarto caso muda consideravelmente no caso do LBMDR: 492 passos e 2,742 segundos. Em contrapartida, os dados obtidos pelo método GVF no caso com múltiplas curvas se mantém próximo aos dados obtidos nos casos com uma curva apenas. Esse resultado reforça o potencial do método GVF em comparação com o método LBMDR. Considerando os resultados da Fig. 6 e os dados da Tabela 1, fica claro que, nos testes apresentados, o GVF foi equivalente ao LBMDR em relação as propriedades do campo gerado sendo computacionalmente mais eficiente. Assim, este foi o método implementado no sistema computacional descrito na seção IV.

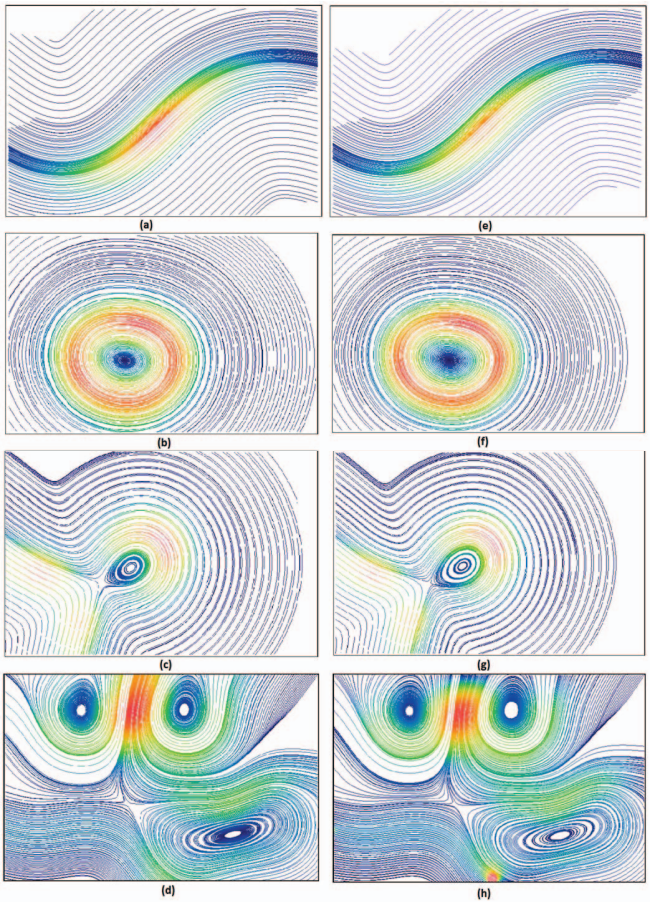


Fig. 6. Linhas de corrente do campo gerado pelo GVF (coluna da esquerda) e LBMDR (coluna da direita).

Seguindo o fluxograma da Fig. 2, uma vez obtido o campo inicial de velocidades pelo processo de difusão-reação, executamos a simulação do fluido usando o método LBMF. As simulações foram executadas até 400 passos de iterações, sendo os campos iniciais aqueles mostrados na Fig. 6(a)-(d). As Figs. 7(a)-(d) mostram os quadros finais da simulação de fluido para estes casos, respectivamente. As simulações completas dos casos 7(a)-(c) estão disponíveis em [17].

No caso da simulação correspondente à linha de corrente simples (Fig. 7(a)), um aspecto importante a ser observado é a presença de regiões em rotação (vórtices) ao longo da simulação. Este fato indica geração de vorticidade, definida pelo rotacional da velocidade (Apêndice B em [11]), criando novos pontos singulares que alteram totalmente a topologia do fluxo [18]. Este resultado é natural na etapa de simulação, uma vez que a topologia inicial do campo de velocidades pode ser radicalmente alterada durante a evolução do fluido.

No caso da linha de corrente fechada, o objetivo é gerar uma região de confinamento de fluido dentro do domínio, onde as linhas de corrente resultantes serão concêntricas ao *sketch* do usuário. Ao observarmos a Fig. 7(b) percebemos que este objetivo foi alcançado. Este exemplo representa bem o potencial da metodologia apresentada neste trabalho, uma vez que não é trivial obter tal comportamento apenas impondo condições na fronteira. Além disso, a região que contém o centro da curva fechada tem velocidades com baixa intensidade comparada à região próxima à curva original. Isso pode ser observado pela coloração na figura, onde as cores mais escuras indicam menor intensidade de velocidade no campo. Este comportamento está correto do ponto de vista dinâmico, pois a intensidade da velocidade deve diminuir à medida que nos aproximamos do centro do vórtice para evitar inconsistência do campo.

O caso de múltiplas linhas de corrente apresenta as singularidades esperadas nas curvas fechadas e regiões de vórtices, como mostra a Fig. 7(d).

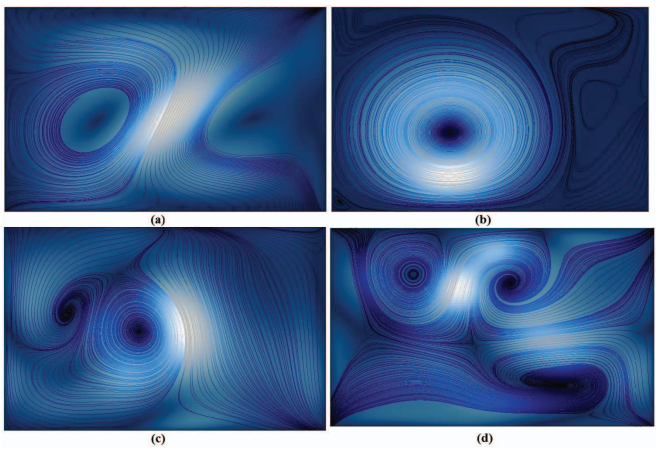


Fig. 7. Passo 400 para as simulações com inicializações dadas pelos campos da Figura 6(a)-(d), respectivamente.

O próximo exemplo envolve o problema clássico da cavidade quadrada e mostra outra vantagem da nossa metodologia ao permite obter a solução de problemas conhecidos mais rapidamente. Assim, na Fig. 8(a) representamos o problema da cavidade quadrada, onde a parede superior se desloca com velocidade constante $\vec{U}$, enquanto as outras paredes permanecem com velocidade nula [19].

O fluido está confinado na cavidade e o escoamento possui solução estacionária com topologia definida por três vórtices, um central e dois outro nos cantos inferiores do domínio, como mostrado na Fig. 8(a). A Fig. 8(b) mostra o *sketch* utilizado, o qual é composto por uma curva fechada na região do vórtice central. O resultado do GVF para este caso é mostrado na Fig. 8(c). A Fig. 8(d) mostra o resultado do LBMF para $\varepsilon = 10^{-4}$ na equação (7). O sistema necessitou 2435 passos de iteração sem a utilização do *sketch* e 2193 passos com o *sketch* para atingir a condição de parada, gerando a solução visualizada na Fig. 8(d). Porém, se $\varepsilon = 10^{-5}$ então são necessários 12908 para atingir a condição de parada sem o *sketch* contra 9766 quando o *sketch* é utilizado, ficando mais realçada a redução do esforço computacional.

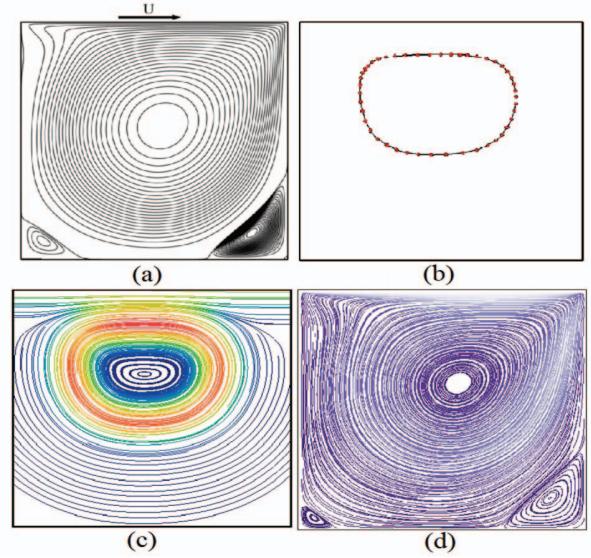


Fig. 8. (a) Solução estacionária para cavidade quadrada. (b) *Sketch* para problema da cavidade quadrada (c) GVF após 226 passos de iteração. (d) Resultado da simulação via LBMF.

VI. Conclusões

Apresentamos neste trabalho uma metodologia de animação de fluidos bidimensionais usando técnicas de *sketching* juntamente com equações de difusão-reação e modelo LBM. A proposta apresentada, juntamente com o software implementado, permite ao usuário fornecer um rascunho do comportamento inicial do escoamento, de maneira que ele possa informar as características desejadas, tais como a existência de um vórtice e a topologia inicial do fluxo.

Foram testadas duas técnicas de difusão-reação, uma baseada em equações diferenciais (GVF) e outra em modelos do tipo LBM. Nos testes realizados, o GVF foi computacionalmente mais eficiente gerando um campo inicial que preserva a topologia do fluxo fornecida pelo usuário. Como trabalho futuro, vamos estender a metodologia para 3D, a partir do estudo inicial realizado em [6].

Agradecimentos

O presente trabalho foi realizado com o apoio financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e do Ministério de Ciência, Tecnologia, Inovações e Comunicações do Brasil.

Referencias

[1] Pixar On-Line Library. Available: http://graphics.pixar.com/library/.
[2] ACM Trans. Graph., vol. 34, no. 4, 2015.
[3] S. F. Judice, B. B. S. Coutinho, and G. A. Giraldi. “Lattice Methods for Fluid Animation in Games,” *Computers in Entertainment*, vol. 7, no. 4, pp. 1-29, 2009.
[4] L.-S. Luo, M. Krafczyk, and W. Shyy, *Lattice Boltzmann Method for Computational Fluid Dynamics*. John Wiley & Sons, Ltd, 2010.
[5] S. F. Judice and G. A. Giraldi. “Sketching Fluid Flows - Combining Sketch-Based Techniques and Gradient Vector Flow for Lattice-Boltzmann Initialization,” in *GRAPP/VISIGRAPP*, Rome, Italy, 2012, pp. 328-337.
[6] S. Judice, J. G. Mayworm, P. Azevedo, and G. Giraldi. Perspectives for Sketching Fluids Using Sketch-Based Techniques and Gradient Vector Flow for 3D LBM Initialization. *Computer Vision, Imaging and Computer Graphics: Theory and Application*, Springer Berlin Heidelberg, vol. 359, 2013, pp. 127-141.
[7] Bo Zhu, M. Iwata, R. Haraguchi, T. Ashihara, N. Umetani, T. Igarashi, and K. Nakazawa. “Sketch-Based Dynamic Illustration of Fluid Systems,” in *Proc. of the SIGGRAPH Asia Conference*. Hong Kong, China, 2011, pages 134-141.
[8] P. Saalfeld P, A. Baer, U. Preim, B. Preim, and K. Lawonn. “Sketching 2D Vessels and Vascular Diseases with Integrated Blood Flow,” in *Proc. of the 10th Int. Conf. on Comp. Graph. Theory and Applications*. Berlin, Germany, 2015, pages 379–390.
[9] L. Olsen, F. F. Samavati, M. C. Sousa, J. A. Jorge. “Sketch-Based Modeling: A Survey,” *Computers & Graphics*, Volume 33, No. 1, Feb. 2009, pp. 85-103.
[10] *Sketch-Based Interfaces and Modeling*. Joaquim Jorge, Faramarz Samavati (Editors), *Springer*, 2011.
[11] S. F. P. P. Judice. “Modelagem e Simulação de Fluidos via Técnicas de Sketching, Modelos de Difusão-Reação e Método de Lattice Boltzmann,” Tese de Doutorado, Laboratório Nacional de Computação Científica, 2016. Available: tede.lncc.br/bitstream/tede/241/2/thesis-sicilia.pdf.
[12] J. Zhang and G. Yan. “A Lattice Boltzmann Model for the Reaction-Diffusion Equations with Higher Order Accuracy,” *Journal of Scientific Computing*, Vol. 52, No. 1, pp. 1-16, 2012.
[13] C. Xu and J. L. Prince. “Snakes, Shapes, and Gradiente Vector Flow,” *IEEE Trans. on Image Processing*, vol, 7, no. 3, pp. 359-369, 1998.
[14] ParaView. An Open-Source, Multi-Platform Data Analysis and Visualization Application. Available: http://www.paraview.org/.
[15] E. A. V Brazil. “On Sketches for Modeling,” Tese de Doutorado, IMPA – Instituto de Matemática Pura e Aplicada, Rio de Janeiro, 2011.
[16] J. Sotomayor, *Lições de Equações Diferenciais Ordinárias*, Projeto Euclides, Gráfica Editora Hamburgo Ltda, São Paulo, 1979.
[17] Simulações LBMF. Available: virtual01.lncc.br/~giraldi/lbmf
[18] P. G. Saffman. *Vortex Dynamics*. Cambridge Monographs on Mechanics and Applied Mathematics, Cambridge University Press, 1995.
[19] R. Schreiber and H. B. Keller. “Driven Cavity Flows by Efficient Numerical Techniques,” *Journal of Computational Physics*, vol. 49, pp. 310–333, 1983.

Sicilia Ferreira Ponce Pasini Judice possui graduação em Ciência da Computação pela Universidade Cândido Mendes (RJ). Em 2009 concluiu o Mestrado em Modelagem Computacional no Laboratório Nacional de Computação Científica (LNCC), Petrópolis – RJ, na área de animação de fluidos. Finalizou o Doutorado também no LNCC em 2016, na área de animação de fluidos via método de Lattice Boltzmann e técnicas de sketching. Atualmente faz Pós-Doutorado na Universidade de Calgary, Canadá, na área de visualização científica para modelagem e simulação de reservatórios de petróleo.

Gilson Antonio Giraldi possui graduação em Matemática pela Pontifícia Universidade Católica de Campinas (1986), mestrado em Matemática Aplicada pela Universidade Estadual de Campinas (1993) e doutorado em Engenharia de Sistemas e Computação pela Universidade Federal do Rio de Janeiro (2000). Atualmente é pesquisador do Laboratório Nacional de Computação Científica (LNCC). Além disso, é Bolsista de Produtividade em Pesquisa do CNPq. Tem experiência na área de Ciência da Computação, com ênfase em Matemática da Computação, atuando principalmente nos seguintes temas: reconhecimento padrão, modelos deformáveis, processamento e visualização de imagens e animação de fluidos.