# A stable tensor-based method for controlled fluid simulations

Marcelo Caniato Renhe[a,*], Marcelo Bernardes Vieira[a], Claudio Esperança[b]

[a] *Departamento de Ciência da Computação, Universidade Federal de Juiz de Fora, Rua José Lourenço Kelmer, s/n - Campus Universitário, Bairro São Pedro - Juiz de Fora - MG, Brazil*

[b] *Cidade Universitária, Centro de Tecnologia, Bloco H, Laboratório de Computação Gráfica, Rio de Janeiro - RJ, Brazil*

**Abstract**

The association between fluids and tensors can be observed in some practical situations, such as diffusion tensor imaging and permeable flow. For simulation purposes, tensors may be used to constrain the fluid flow along specific directions. This requires a customized mathematical model for describing fluid motion influenced by tensors. In this work, we propose a formulation for fluid dynamics to locally change momentum, deflecting the fluid along intended paths. Building upon classical computer graphics approaches for fluid simulation, we discuss the new formulation in detail and we adapt the numerical method to accommodate it. Gaining control over fluid diffusion can also aid on visualization of tensor fields, where the detection and highlighting of paths of interest is often desired. Experiments show that the fluid adequately follows meaningful paths induced by the underlying tensor field, resulting in a method that is numerically stable and suitable for visualization and animation purposes.

*Keywords:* Fluid simulation, Tensor field, Fluid control

## 1. Introduction

Methods for controlling fluid simulations have been an active research topic since [1], which employed user-defined keyframe images to guide the simulation. Many of the works related to this subject [2, 3, 4, 5, 6, 7] are target-driven. This means that the fluid follows arbitrary paths in order to reach and form a pre-specified target shape. Fattal and Lischinski [8] followed this pattern, but, differently from others, used external forces to direct the fluid. A similar approach was presented in [9], using a distance function as the external force.

The works mentioned above are mainly concerned with the final state of the fluid, letting the fluid flow freely during the intermediate steps. Here, we are interested in controlling fluid flow throughout the entire simulation. Most methods related to this problem use external forces to restrict fluid flow to specific paths [10, 11, 12]. There are also methods based on deformation of the underlying grid [13, 14], adapting the velocity fields according to the deformed grid. Interpolation of flow fields were also attempted in [15] in order to generate incompressible fluid animations.

Despite the seeming artificiality of these control mechanisms, the idea of a constrained or directed fluid flow can be observed in certain real scenarios, which makes these methods applicable outside an animation context. For example, if we look into porous media flow, fluid motion is restricted by the medium. The amount of fluid that can flow through the pores can be measured by a physical quantity called permeability, which is defined as a tensor. Interestingly, in diffusion tensor imaging, tensor fields obtained from DT-MRI, a magnetic resonance imaging technique, describe the fiber patterns of water inside organic tissue. These diffusion tensors represent the probability of water flowing in specific directions. So, given this relationship between anisotropic flow and tensors, a natural question arises: would it be viable to use tensors to control and direct a fluid along paths of interest? How can we provide a straightforward way of simulating anisotropic transport in an easy-to-use and stable manner?

This work intends to demonstrate that tensors can be successfully used to control fluids, whether it be for simulation or animation applications. In our approach, fluid flow is controlled by tensor fields, which are interpreted

*Corresponding author

*Email addresses:* marcelo.caniato@ice.ufjf.br (Marcelo Caniato Renhe), marcelo.bernardes@ice.ufjf.br (Marcelo Bernardes Vieira), esperanc@cos.ufrj.br (Claudio Esperança)

as an inherent part of the medium. The fluid in our simulations is not rigidly bound to a path, as in typical force-based methods, nor is deformed by precomputed velocity fields. We leverage on the probabilistic nature of tensors to smoothly deviate fluid towards paths of interest. Similar rationale have been used before in visualization contexts, where fluids are influenced by tensors to follow specific paths, whether by altering viscosity and pressure [16] or by molding advection and introducing tensor-based external forces [17].

Our focus is on a special category of tensors defined by Westin et al. [18], namely the orientation tensors. These are symmetric positive semidefinite rank-2 tensors usually related to covariance estimations. Mathematically, a local orientation tensor can be defined as follows:

$$\mathbf{B} = \sum_{n}^{i=1} \lambda_i \mathbf{e}_i \mathbf{e}_i^T. \tag{1}$$

In this work, we use a scaled version of the tensor $\mathbf{B}$, denoted as $\mathbf{T} = \beta\mathbf{B}$, where $\beta$ is a boosting factor.

To achieve our purposes, we adapted a common fluid simulation method, first proposed by Stam [19], by customizing the mathematical model behind it so that the tensor information is used to locally alter fluid momentum. A new discretization for the basic simulation steps of Stam's approach is also proposed in order to reflect the adapted equations. The main contributions of this work are summarized as follows:

1. a system of partial differential equations whose solution leads to a tensor-based customized description of fluid flow,
2. a stable numerical method for anisotropic transport defined by the aforementioned system of equations,
3. a method that allows for constructing tensor fields aimed at controlling fluids,
4. a method potentially applicable to the visualization of diffusion tensor fields.

Stam's method is divided in 3 steps: advection, diffusion and projection. We used tensors to introduce anisotropy information in each one of them. For the diffusion part, we employed some discretizations proposed by Günter et al. [20], which used them to model heat diffusion in magnetised plasmas. Anisotropic diffusion is a challenging problem to which many physics researchers have proposed various solutions. Mimetic Finite Difference (MFD) methods [21] are commonly employed in this context. The number of available MFD methods is quite large, so a good reference for the interested reader is the review article by Lipnikov et al. [22].

The next section introduces our method and subsequently defines and discusses the proposed customized system of equations (Section 2.1). Sections 2.2, 2.3 and 2.4 discuss the adaptation made to each of the 3 steps of the simulation. In Section 2.5, we focus on demonstrating that our method still maintains the unconditional stability property of Stam's method after the introduction of tensor information. Next, we present a number of experimental results, separated into comparison experiments (Section 3.1), design of tensor fields for simulation (Section 3.2) and 3D experiments (Section 3.3). Section 3.4 discusses some issues related to our method. Lastly, Section 4 presents our concluding remarks.

## 2. Proposed method

As in Stam's approach, we used an Eulerian representation of the fluid, aligning the grid cells with the field. Each tensor is located at the center of its corresponding cell. We work on the idea of the tensor acting as something that deflects velocities passing through its cell, locally reducing or amplifying momentum in the process. Depending on the degree of alignment between tensor and velocities, the tensor could be physically interpreted as a pump (alignment to eigenvectors with high eigenvalues, for instance) or a sink (velocities perpendicular to main eigenvector with low eigenvalue). Figure 1 shows some examples.

In [17], tensors were used to directly transform velocities during the interpolation step of the semi-Lagrangian advection, forcing velocity vectors to abide by the tensor field characteristics. While it could definitely produce good visual results, that was a rather arbitrary approach. It would be interesting to produce the same effect using a more physically-motivated formulation leading to a certain level of numerical stability. Thus, we propose to use the tensor as a local momentum modulator for the fluid as it moves along the field. So, we begin by introducing the following differential equation:

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{K}\mathbf{u}, \tag{2}$$

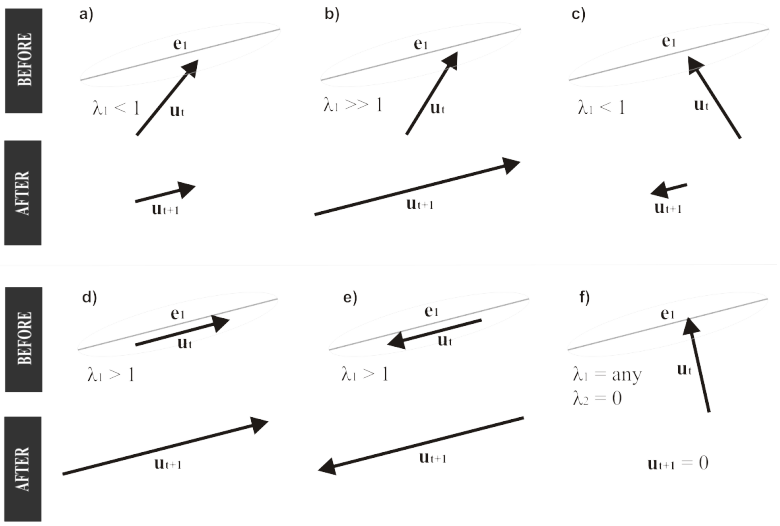



Figure 1: Six cases of a velocity vector being transformed by a tensor. Each case shows a different situation of tensor-vector alignment, producing deflected vectors that may be scaled or shrinked, according to the tensor eigenvalues.

where $\mathbf{u}$ is measured in $m/s$ and $\mathbf{K}$ is a matrix measured in $s^{-1}$. We propose $\mathbf{K} = \mathbf{T} - b\mathbf{I}$, where $b$ is a scaling factor, to transform the velocity $\mathbf{u}$ in the following form:

$$\frac{\partial \mathbf{u}}{\partial t} = (\mathbf{T} - b\mathbf{I})\mathbf{u},$$

which, setting $b = 1$ and putting physical units aside, can be viewed as:

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{T}\mathbf{u} - \mathbf{u}. \tag{3}$$

This is the version we will use for the remainder of this text, always considering the units specified in Eq. 2: $[\mathbf{T}] = \text{s}^{-1}, [b] = \text{s}^{-1}, [\mathbf{u}] = \text{m s}^{-1}$. The dimensional equation for Eq. 3 can be written as $[L^1T^{-2}] = [T^{-1}][L^1T^{-1}] - [T^{-1}][L^1T^{-1}]$. We include this last equation as an additional equation to the original Navier-Stokes formulation, responsible for describing fluid motion. It works as a restriction on the velocity. The main insight behind this restriction is that a tensor should decelerate velocities that are not aligned with its eigenvectors, while also bending them into alignment. If we look at Fig. 2, we see that the transformation of a velocity vector $\mathbf{u}$ by an orientation tensor $\mathbf{T}$ results in a new velocity $\mathbf{Tu}$, whose direction and norm are affected by $\mathbf{T}$'s eigensystem. The vector $\mathbf{Tu} - \mathbf{u}$, which takes $\mathbf{u}$ to $\mathbf{Tu}$, can be interpreted as an acceleration vector. Depending on the eigenvalues of $\mathbf{T}$ and on the angle between its eigenvectors and $\mathbf{u}$, $\mathbf{Tu}$ could locally accelerate or decelerate it, changing fluid momentum in the nearby region.

Eq. 3 is a homogeneous differential equation with constant coefficients in time, i.e., $\partial T/\partial t = 0$. It governs fluid acceleration when coupled with the Navier-Stokes equations. But how do we enforce this restriction on the system? There are different ways to achieve that, and it depends on how we interpret the tensor role in the system. One could think of adding the restriction to the system as an external force, multiplying the resulting acceleration by the fluid density. This would be described by the equation below:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu\nabla^2\mathbf{u} + \mathbf{T}\mathbf{u} - \mathbf{u}. \tag{4}$$

The main advantage of this approach is that it can be directly applied to the Navier-Stokes equations. However, there are a number of disadvantages: it requires very small time steps to remain stable and fluid control is only exerted during instantaneous force applications. So, we can say that using Equation 3 as a force does not establish a strong link between the tensor field and the fluid. If it were applied as a force, the tensor effect would cease as soon as the force stopped acting, since the advection would be able to transport quantities without restrictions. On the other hand,

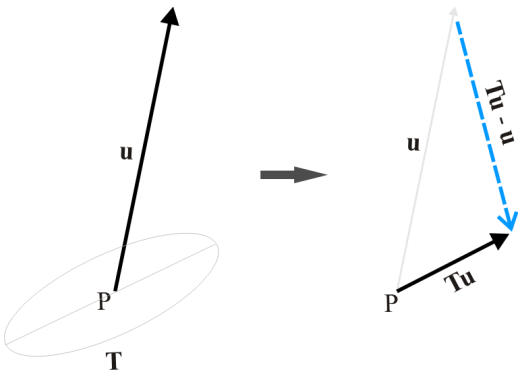



Figure 2: Geometrical interpretation of Equation 3 in $\mathbb{R}^2$.

if we take as a premise that the tensor field is an inherent part of the environment, thinking of it as a property of the medium, it becomes an integral part of fluid motion. The fact that tensor fields are kept static during our simulations also favors the adoption of this point of view. By choosing this approach, what we are saying is that all aspects of the fluid dynamics are bound to the tensor field characteristics. This places a strong bias on fluid motion, making it likely to follow a predefined path. The drawback of this approach is that care must be taken on how to insert the tensor information into the equations, since they govern fluid motion in a coupled way. Also, it is desirable that the tensor field be continuous everywhere, since the presence of null tensors could cripple the simulation. However, for certain discontinuous cases, this approach showed to produce good results.

### *2.1. Proposed continuous formulation*

In order to consider the tensor as a property of the medium through which the fluid flows, we need to adapt the Navier-Stokes equations to account for the tensor information. The Navier-Stokes system of equations can usually be written as:

$$\begin{cases} \frac{\partial q}{\partial t} = -(\mathbf{u} \cdot \nabla)q - \frac{1}{\rho}\nabla p + \phi \nabla^2 q + \mathbf{f} \\ \nabla \cdot \mathbf{u} = 0 \end{cases},$$

where $q$ is the transported quantity (the velocity $\mathbf{u}$ or the density $\rho$, for our purposes), $p$ is the pressure, $\phi$ is the diffusion coefficient (which, for velocities, becomes the viscosity $\nu$) and $\mathbf{f}$ is the sum of external forces. We propose the usage of the following customized system of equations for describing the dynamics of fluids. Notice the addition of Eq. 3:

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \nabla \cdot (\mathbf{T}\nabla \mathbf{u}) + \mathbf{f} \\ \frac{\partial \mathbf{u}}{\partial t} = \mathbf{T}\mathbf{u} - \mathbf{u} \qquad \text{(advective-only)} \\ \nabla \cdot \mathbf{u} = 0 \\ \frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho + \nabla \cdot (\mathbf{T}\nabla \rho) - \alpha\rho + S \\ \mathbf{u}(0) = \mathbf{u}_0 \end{cases} \tag{5}$$

The system of equations above accounts for both the velocity and the density update. The acceleration equation presented in the previous section enters here as a purely advective equation, working as a restriction to fluid motion. Since the advection term is the one responsible for the main transport of the fluid, the addition of the restriction is significative. Eq. 3 should, considering the underlying tensor field, discourage fluid motion along certain regions and deflect the fluid towards intended paths. We do this by adapting the numerical method that solves the advection part. This will be addressed in Section 2.2.

The tensors also affect the diffusive terms for velocities and densities. The tensor is introduced in the equation by multiplying the gradient of the fluid quantity being diffused. This results in a diffusion which is no longer isotropic. Thus, in our formulation, the viscosity $\nu$ and the diffusion constant for densities are both replaced by the tensor $\mathbf{T}$. It is the tensor who determines the degree of diffusiveness applied in each direction for the quantity being evaluated.

This is a classic formulation for anisotropic diffusion. Our contribution is related to how we handle this term, by precomputing Laplacian masks derived from the tensor information. Section 2.3 discusses the details of this proposal.

Finally, in the density equation, we added a source term $S$, for inserting fluid in the system, and a dissipation term $-\alpha\rho$, as used in [19], to avoid the excessive accumulation of densities outside the regions of interest of the tensor field. Since the tensor advection tends to reduce motion in these regions, densities who eventually fall outside the controlled paths will hardly be able to return to the tensor-controlled regions. This could be reduced by using smaller time steps, but the dissipation function provides a good alternative instead.

### *2.2. Customized advection*

The advection is the major means of transport in a fluid. Consider again Eq. 3, coupled with the initial condition $\mathbf{u}(0) = \mathbf{u}_0$ included in the system of equations above. If we put it in the form $\mathbf{u}' + \mathbf{A}\mathbf{u} = 0$, which is a homogeneous equation with constant coefficients, we obtain:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{I} - \mathbf{T})\mathbf{u} = 0.$$

Assuming $\mathbf{A} = \mathbf{I} - \mathbf{T}$ is a diagonalizable matrix, then there is $\mathbf{D}$ such that $\mathbf{D} = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$, the diagonal matrix of $A$'s eigenvalues, and there is a matrix $\mathbf{P} = [\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n]$ such that $\mathbf{P}$ is invertible and $\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$.

Let us define $e^{-t\mathbf{D}} = \text{diag}(e^{-t\lambda_1}, \cdots, e^{-t\lambda_n})$ and $e^{-t\mathbf{A}} = \mathbf{P}e^{-t\mathbf{D}}\mathbf{P}^{-1}$. Then, the solution for Eq. 3 is:

$$\mathbf{u}(t) = e^{-t\mathbf{A}}\mathbf{u}_0 = e^{-t(\mathbf{I}-\mathbf{T})}\mathbf{u}_0. \tag{6}$$

Our goal is to use this equation directly in a backtracing approach. But firstly we obtain a discretized version of Eq. 3, using the operator splitting technique [23]. Since Eq. 3 is composed of two terms, we can solve it in two steps:

$$\tilde{\mathbf{u}} = \mathbf{u}^t + \Delta t\mathbf{T}\mathbf{u}^t,$$
$$\mathbf{u}^{t+1} = \tilde{\mathbf{u}} - \Delta t\tilde{\mathbf{u}},$$

where $t$ indicates a time instant, or an iteration, which rearranged yields the expression for $\mathbf{u}^{t+1}$:

$$\mathbf{u}^{t+1} = \mathbf{u}^t + \Delta t[\mathbf{T}\mathbf{u}^t(1 - \Delta t) - \mathbf{u}^t]. \tag{7}$$

This equation defines how velocities are transported between time $t$ and $t + 1$. So, we want to evaluate this equation whenever velocities suffer any variation during the advection. The original semi-Lagrangian advection proposed by [19] consists of a particle in cell location $\mathbf{x}_G$ propagated backwards in search of its previous position $\mathbf{x}_P$. The velocity in $\mathbf{x}_P$ is then interpolated using its neighbors, and this new value is used to update the velocity in $\mathbf{x}_G$. The backtraced position $\mathbf{x}_P$ in time $t$ is given by

$$\mathbf{x}_P = \mathbf{x}_G - u^t_G\Delta t,$$

that may not fall exactly at the center of a cell. Thus, we interpolate any needed value at $x_P$ from its closest neighbors

$$\mathbf{u}' = \text{interp}(\mathbf{u}^t_P, \mathbf{x}_P), \mathbf{T}' = \text{interp}(\mathbf{T}^t_P, \mathbf{x}_P), \tag{8}$$

where $\text{interp}(\cdot, \mathbf{x})$ is a bilinear or trilinear interpolation function. We could simply plug these interpolated values into Eq. 7 to obtain the approximation $\mathbf{u}^{t+1}_G = \mathbf{u}' + \Delta t[\mathbf{T}'\mathbf{u}'(1 - \Delta t) - \mathbf{u}']$. The problem with this solution is the strong dependency on the time step that should be really small in order to smoothly deflect the local fluid.

An alternative solution is to use Eq. 6 directly in the advection process but keeping the backtracing approach:

$$\mathbf{u}^{t+1}_G = e^{-\Delta t(\mathbf{I}-\mathbf{T}')}\mathbf{u}', \tag{9}$$

where the interpolated tensor affects the current velocity integrating with the analytical solution along the whole $\Delta t$. The overall process is shown in Figure 3. The position $\mathbf{x}_P$ is encountered by moving the particle back in time. With $\mathbf{x}_P$, the new velocity value used to update $\mathbf{u}(\mathbf{x}_G)$ at time $t + 1$ is the result of a direct application of Eq. 6. The resulting velocity field is not divergent-free, as is the classical method's, and must be projected. Our approach can be adapted for more sophisticated advection methods such as MacCormack [24], FLIP [25], APIC [26], among others.

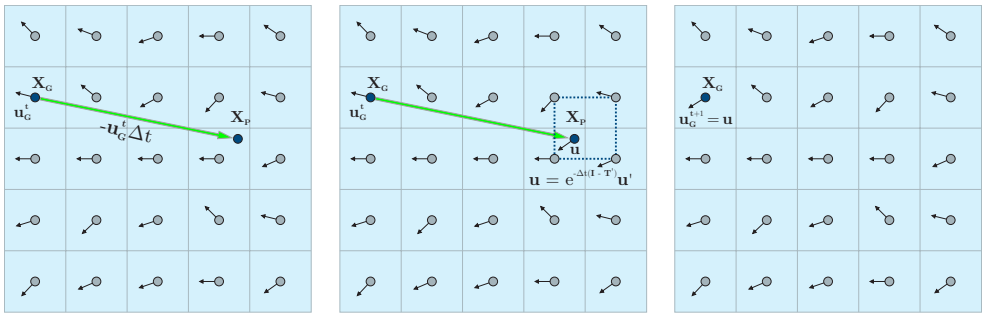



Figure 3: Illustration of the enhanced customized advection, now directly using Eq. 6.

### *2.3. Customized diffusion*

In [19] the diffusion step of the simulation is solved implicitly to improve stability. The implicit equation for the diffusion of any fluid-related quantity can be defined as:

$$(\mathbf{I} - \phi \Delta t \nabla^2) q'(x) = q(x), \tag{10}$$

where $\phi$ is the diffusion coefficient, and $q'(x)$ and $q(x)$ are the diffused and original quantities, respectively. Stam [27] details the reasons why the implicit formulation should be used. The idea is similar to the semi-Lagrangian advection, i.e., we look back in time to find which values $q^{t+1}$, when diffused backwards, produce the value $q^t$. This leaves us with a linear system of unknowns $q_{i,j}^{t+1}$ that can be rearranged and written in the form of the following update equation:

$$q_{i,j}^{t+1} = \frac{q_{i,j}^t + \phi \Delta t \Delta x \Delta y (q_{i-1,j}^{t+1} + q_{i+1,j}^{t+1} + q_{i,j-1}^{t+1} + q_{i,j+1}^{t+1})}{1 + 4\phi \Delta t \Delta x \Delta y}. \tag{11}$$

This can be solved by an iterative solver like the Gauss-Seidel relaxation. But the above formulation is only applicable for an essentially isotropic process. For a generic quantity $q$, the anisotropic diffusion can be defined as:

$$\frac{\partial q}{\partial t} = \nabla \cdot (\mathbf{T} \nabla q)$$

We used the asymmetric and symmetric discretization schemes proposed by Günter et al. [20], which are illustrated in Fig. 4. Each one of these schemes has advantages and disadvantages. For example, the symmetric scheme works better for tensors oriented in the diagonal, while these tensors have poor result for axis-aligned tensors. The symmetric approach also fails with isotropic tensors, forming a checkerboard pattern due to the lack of exchanges between the nearest neighbors (Fig. 5). Actually, the checkerboard pattern plagues this scheme for most tensors with low levels of anisotropy. The asymmetric scheme, however, works fine for both isotropic and highly anisotropic axis-aligned tensors. In the diagonal, the asymmetric still offers good results, but produces some artifacts as shown in Figure 6.

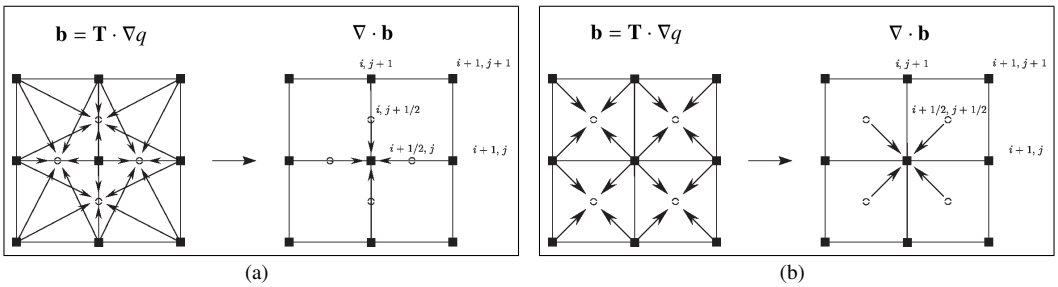



Figure 4: (a) Asymmetric and (b) symmetric finite difference schemes for 2D laplacian computation [28].

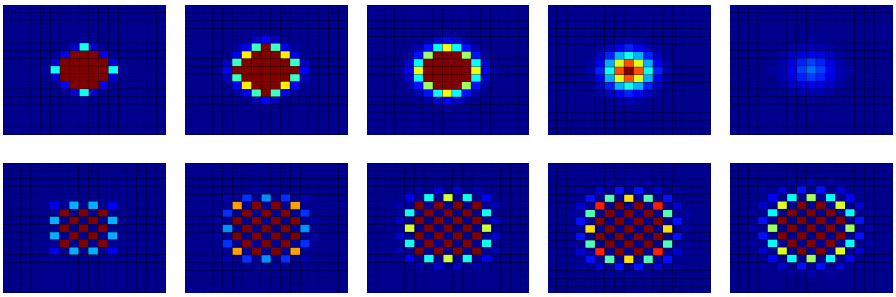


Figure 5: Two sequences of frames for diffusion in the presence of an isotropic tensor field: the top sequence used the asymmetric scheme, while the bottom sequence used the symmetric one.

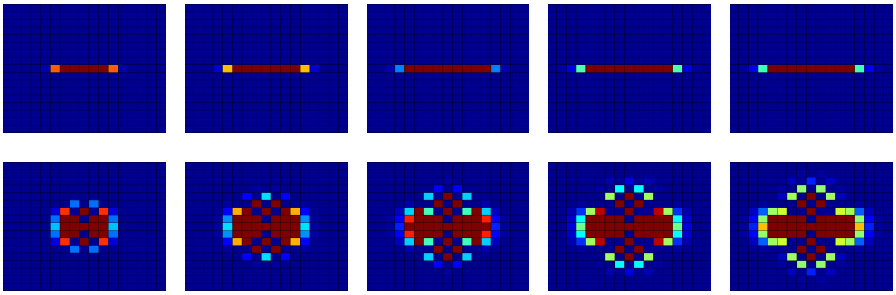


Figure 6: Two sequences of frames for diffusion in the presence of a linear tensor field, oriented horizontally: the top sequence used the asymmetric scheme, while the bottom sequence used the symmetric one.

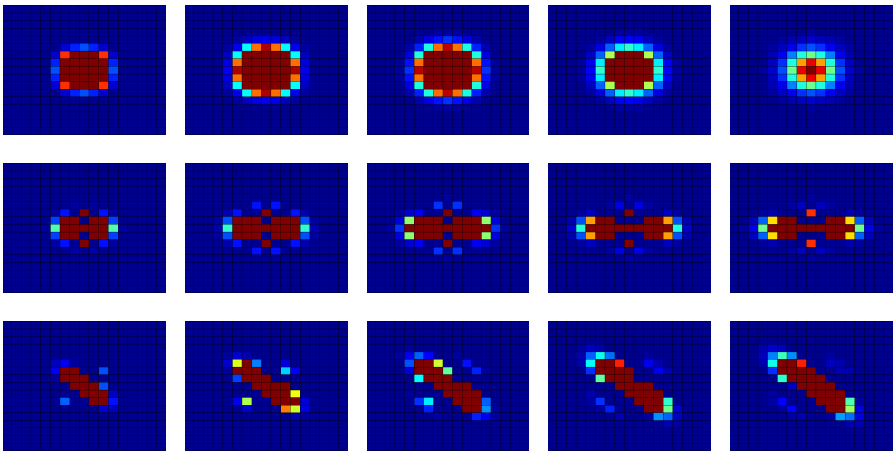


Figure 7: Sequences of frames for the averaged scheme. The topmost sequence is for an isotropic tensor field, the central sequence is an horizontal linear tensor field and the bottom sequence is a diagonal linear tensor field.

Figures 5 and 6 show two different configurations of a Matlab simulation of heat transfer using these two schemes. As shown, no scheme alone is ideal for tensors oriented arbitrarily. Thus, we propose to use a combination of these two schemes. Experiments showed that a simple average between these two is enough to enhance the results. One reason is that the majority of the tensors in practice are not purely linear. So, the symmetric scheme in the average serves as a way of compensating for the few situations where the asymmetric presents issues. Figure 7 shows the outcome of the averaged scheme.

As the tensor fields are static throughout the simulation, we can obtain Laplacian weighting masks to be directly applied during the diffusion step. This allows us to reutilize the diffusion solution used in the classic fluid simulation,

simply replacing the isotropic Laplacian mask by anisotropic masks computed from the differential operators proposed in [20]. The update equation for the anisotropic Gauss-Seidel relaxation technique in $\mathbb{R}^2$ is as follows:

$$q_{i,j}^{t+1} = \frac{q_{i,j}^{t} + \sum_{a=1}^{3}\sum_{b=1}^{3} M_{a,b} Q_{a,b}}{1 - M_{2,2}}, \tag{12}$$

where $M_{i,j}$ are the components of $M(\mathbf{T})$, the anisotropic Laplacian mask for local tensor $\mathbf{T}$, and $Q$ is the matrix containing the values of $q$ for the neighbors:

$$Q = \begin{bmatrix} q_{i-1,j-1}^{t+1} & q_{i-1,j}^{t+1} & q_{i-1,j+1}^{t+1} \\ q_{i,j-1}^{t+1} & 0 & q_{i,j+1}^{t+1} \\ q_{i+1,j-1}^{t+1} & q_{i+1,j}^{t+1} & q_{i+1,j+1}^{t+1} \end{bmatrix}.$$

The mask $M(\mathbf{T})$ is the average of the symmetric mask $S(\mathbf{T})$ and the asymmetric one $A(\mathbf{T})$. We use the discrete Laplacian operators defined in [20] and illustrated in Fig. 4 to construct matrices representing $S$ and $T$. The mask $S$ is defined as:

$$S(\mathbf{T}) = \begin{bmatrix} \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{1}) & \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{2}) & \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{3}) \\ \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{4}) & \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{5}) & \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{6}) \\ \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{7}) & \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{8}) & \nabla\cdot(\mathbf{T}\nabla L_{0,0}^{9}) \end{bmatrix}, \tag{13}$$

where the operator computation follows the symmetric scheme in Fig. 4 and $L_{i,j}^{k}$ is a 3×3 scalar matrix, with $1 \leq k \leq 9$ and elements $l_{i,j}$, $i = \{-1, 0, 1\}$, $j = \{-1, 0, 1\}$, and:

$$l_{i,j} = \begin{cases} 1, & \text{if } 3(i+1) + j + 2 = k \\ 0, & \text{otherwise,} \end{cases}$$

from which the anisotropic Laplacian is taken at central point $L_{0,0}^{k}$.

$A(\mathbf{T})$ is computed in the same way, but using the asymmetric scheme instead. This method is easily adaptable to $\mathbb{R}^3$, and the assymetric mask alone proved to be fairly sufficient for 3D simulations. Notice that the diffusion coefficient $\phi$ is not used in this work. The tensor acts now as the local diffusion coefficient. Isotropic tensors locally act the same way as the classic approach. The tensor-based diffusion will be stronger or weaker depending on the orientation of the tensor eigenvectors and on its eigenvalues. This anisotropic diffusion complements the idea of a tensor as a property of the medium. Depending on the quantity $q$, the tensor will acquire a different meaning. For velocities, for instance, it becomes the viscosity tensor, reducing or amplifying momentum in the specific directions according to the tensor orientation.

An important concern about anisotropic diffusion, discussed in [28], is the possibility of large numerical errors introduced in the system due to the presence of highly anisotropic tensors. However, the diffusion is dissipative, locally reducing the error in a monotonic fashion at each iteration. Besides, the anisotropic Laplacian operator always has a negative value at its center, which forces the denominator in the update equation to always be greater than 1, ensuring the dissipation and the convergence.

### *2.4. Customized projection*

The projection step consists in the solution of a Poisson problem that keeps the velocity field divergence-free, a prerequisite for mass conservation in the fluid. Even though we defined smoke as our fluid of choice for all simulation experiments, we can consider smoke as an incompressible fluid [29]. However, the original pressure solution, if used in our tensor-based context, may soften the tensor influence provided by the previous steps of the simulation. Furthermore, it makes sense that the pressure also be subject to the tensor field, since we interpret it as a property of the medium. Thus, the pressure effect over the fluid should not be isotropic. The solution to the Poisson problem is now a bit more complex, since a new restriction was added to the system, as follows:

$$\nabla\cdot(\hat{\mathbf{T}}\nabla p) = \frac{1}{\Delta t}\nabla\cdot\mathbf{u}, \tag{14}$$

where $\hat{\mathbf{T}}$ is a suitable normalized version of $\mathbf{T}$. The normalization here is due to the fact that the projection is just a step to enforce incompressibility, so it does not make sense for it to be subject to any kind of gain or loss of momentum.

We can use the same Laplacian mask approach of the diffusion step for the projection as well. Instead of using a matrix $Q$ with all the values of the generic quantity $q$ for the neighbors, we will use a matrix $P$ for the neighboring pressures. The equation for calculating the pressure is then defined as follows:

$$p_{i,j} = \frac{\nabla \cdot \mathbf{u}^t_{i,j} + \sum_{a=1}^{3} \sum_{b=1}^{3} M_{a,b} P_{a,b}}{M_{2,2}}, \tag{15}$$

whose solution can be used to obtain a divergence-free velocity field $\mathbf{u}^t = \mathbf{u}^t - \hat{\mathbf{T}}\nabla p$.

Discontinuities in the tensor field can negatively affect the anisotropic projection. The tensor imposes a bigger restriction to the Poisson solver. Now, fluid exchanges due to pressure cannot occur in some directions, which may lead, in certain cases, to a system that has no solution. This problem can be worse if there are null tensors in the field, i.e., locations in space where pressure-driven exchanges are not allowed. Therefore, a tensor-based projection is only recommended when the tensor field is positive definite everywhere with smoothly changing eigenvectors. The inability of correctly calculating pressures would result in a velocity field that cannot be made divergence-free. Bridson [29] recommends that advection be always performed in a divergence-free velocity field, otherwise serious problems can occur with advection, producing some small artifacts or even severely hampering the simulation.

### *2.5. Stability discussion*

Adding tensors to the simulation could, in theory, disavow the unconditional stability claim of the original method. Dealing with normalized tensor fields would not present an issue regarding stability. However, we apply the scaling factor $\beta$ to an input normalized tensor field in order to stimulate motion, so that the maximum amplification for fluid velocities is exactly $\beta$. The boosting factor poses a valid concern on the possibility of the simulation eventually blowing up. Even though we can satisfactorily control the simulation by carefully choosing the scaling factor, we attempt to ensure that stability holds for our customized advection by following standard literature on that matter and applying the CFL (Courant–Friedrichs–Lewy) condition, which limits time step to a maximum value based on grid spacing. Although the CFL condition is not a stability condition *per se*, it truly reduces the probability of a simulation blowing up [29]:

$$\Delta t \leq \frac{C\Delta x}{|\mathbf{u}|}.$$

The velocity $\mathbf{u}$ in the above equation is the maximum velocity encountered in the fluid in a single iteration. Considering the tensor effect in our velocities and the analytical solution presented in Eq. 6, the CFL condition assumes the following form:

$$\Delta t \leq \frac{C\Delta x}{||\mathrm{e}^{-\Delta t(\mathbf{I}-\mathbf{T})}\mathbf{u}||}. \tag{16}$$

So, differently from the typical fluid simulation, we now have a CFL condition with a denominator which is dependent on a tensor $\mathbf{T}$. What we want to show is that this transformation does not undermine stability. In our case, the worst-case scenario is when the velocity vector is fully aligned with the tensor's main eigenvector, and this tensor has its main eigenvalue greater than 1. This could significantly magnify the velocity. So, for our CFL condition, we consider the maximum velocity as the one that, when transformed by the tensor through Eq. 6, assumes the biggest norm in the grid. Now, consider a valid vector norm $||\cdot||$ that induces a matrix norm defined as:

$$||\mathbf{A}|| = \max_{\mathbf{x}\in\mathbb{R}^n, \mathbf{x}\neq 0} \frac{||\mathbf{A}\mathbf{x}||}{||\mathbf{x}||}, \qquad \text{or, equivalently,} \qquad ||\mathbf{A}|| = \max_{\mathbf{x}\in\mathbb{R}^n, ||\mathbf{x}||=1} ||\mathbf{A}\mathbf{x}||.$$

Given such matrix norm, it can be proven that:

$$||\mathbf{A}\mathbf{x}|| \leq ||\mathbf{A}|| \cdot ||\mathbf{x}||. \tag{17}$$

Let $\mathbf{A}$ be a symmetric matrix that defines an operator in $\mathbb{R}^n$. We can also prove the following statement:

$$||\mathbf{A}|| = \max_j ||\lambda_j||, \tag{18}$$

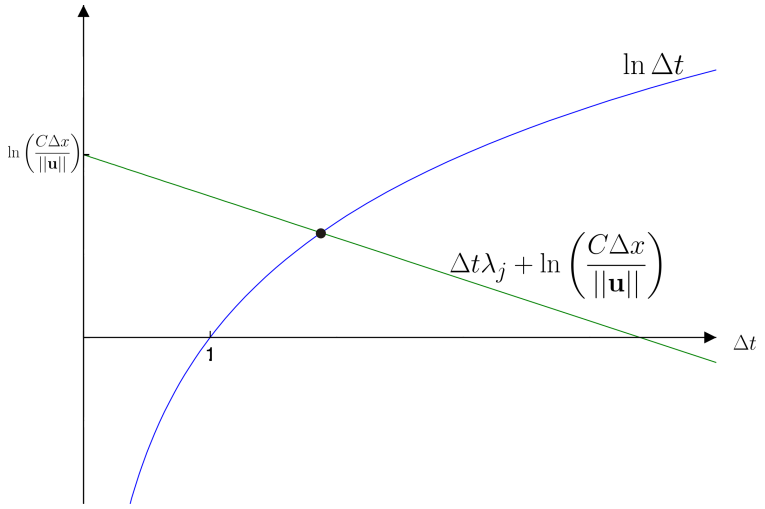



Figure 8: As $\Delta t$ approaches 0, $\ln \Delta t$ goes to infinity. Thus, we can guarantee there is always a valid time step for our advection. When $\lambda_j > 0$, the linear function is increasing, so there is no limit on $\Delta t$. This makes sense, since a positive $\lambda_j$ indicates a purely dissipative tensor.

where $\lambda_j$ are the eigenvalues of $A$. So, considering the stability condition in Eq. 16 and Eq. 17, we know that:

$$||\mathrm{e}^{-\Delta t(\mathbf{I}-\mathbf{T})}\mathbf{u}|| \leq ||\mathrm{e}^{-\Delta t(\mathbf{I}-\mathbf{T})}|| \cdot ||\mathbf{u}||.$$

From Eq. 18, we also know that:

$$\Delta t|\lambda_{max}| \leq \frac{C\Delta x}{||\mathbf{u}||},$$

where $\lambda_{max}$ is the largest eigenvalue of $\mathrm{e}^{\Delta t(\mathbf{I}-\mathbf{T})} = \mathbf{P}\mathrm{e}^{\Delta t\mathbf{D}}\mathbf{P}^{-1}$, $\mathbf{P}$ is the matrix of eigenvectors of $(\mathbf{I}-\mathbf{T})$ and $\mathbf{D} = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ are the eigenvalues of $(\mathbf{I}-\mathbf{T})$. So, we obtain:

$$\Delta t|\mathrm{e}^{-\Delta t\lambda_j}| \leq \frac{C\Delta x}{||\mathbf{u}||},$$

where $\lambda_j$ is the smallest eigenvalue in $\mathbf{D}$. Thus, applying the ln operator and rearranging, we have:

$$\ln \Delta t \leq \Delta t\lambda_j + \ln\left(\frac{C\Delta x}{||\mathbf{u}||}\right), \tag{19}$$

which does not have a closed form solution. Here, the L.H.S. is a logarithmic function of $\Delta t$ and the R.H.S. is a linear function of $\Delta t$. To find the intersection between these two functions is not complicated and can be done efficiently. The smallest $\Delta t$ can be easily obtained as the intersection of the L.H.S. and the R.H.S., for $||\mathbf{u}|| \neq 0$. Notice that: $\lim_{\Delta t\to 0} \ln \Delta t = -\infty$, while the R.H.S., when $\Delta t = 0$, is $\ln \frac{C\Delta x}{||\mathbf{u}||}$, which is constant. Thus, it means there is always a $\Delta t$ which satisfies Inequality 19. So, at each iteration, it is easy to find the maximum $\Delta t$ that guarantees stability. This is illustrated in Fig. 8. Foster and Fedkiw [30] suggests, based on experience, a value around $C = 5$ for the CFL, so that the backtracing particle trajectory length is no longer than five grid cells.

Regarding diffusion, Bridson [29] demonstrates the stability of the implicit treatment of diffusion for the classic method by analyzing an unidimensional version of the problem. He shows that the weights in the update equation presented in Eq. 11 are all positive and that they sum to 1. This guarantees that the new value for $q$ is bounded by its minimum and maximum old values. We reproduce this analysis for our tensor-based diffusion. In our case, we consider the 2D version of this problem, since our problem is at least bidimensional.

First, let us analyze the symmetric scheme. Let $a$ be the product $\phi\Delta t\Delta x\Delta y$. Discretizing $\nabla \cdot (\mathbf{T}\nabla q)$ and putting the

weights in evidence, we get to the following equation for $q_{i,j}$:

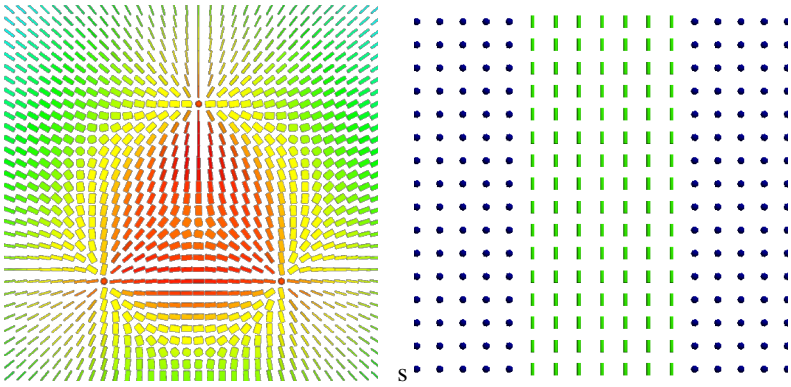
$$q_{i,j}^{t+1} = \left(\frac{1}{1+a(T_{xx}+T_{yy})}\right) q_{i,j}^{t} + \left(\frac{a(T_{xx}+T_{xy}+T_{yx}+T_{yy})}{4(1+a(T_{xx}+T_{yy}))}\right)(q_{i+1,j+1}^{t+1} + q_{i-1,j-1}^{t+1})$$
$$+\left(\frac{a(T_{xx}-T_{xy}-T_{yx}+T_{yy})}{4(1+a(T_{xx}+T_{yy}))}\right)(q_{i+1,j-1}^{t+1} + q_{i-1,j+1}^{t+1}) + \left(\frac{a(2T_{xx}-2T_{yy})}{4(1+a(T_{xx}+T_{yy}))}\right)(q_{i+1,j}^{t+1} + q_{i-1,j}^{t+1}) \quad (20)$$
$$+\left(\frac{a(2T_{yy}-2T_{xx})}{4(1+a(T_{xx}+T_{yy}))}\right)(q_{i,j+1}^{t+1} + q_{i,j-1}^{t+1}).$$

Notice that we now have 8 neighbors contributing to the new value. Considering that each weight appears twice in the equation (each one multiplying two different neighbors), if we sum them all and simplify the opposing terms, we obtain:

$$\frac{1}{1+a(T_{xx}+T_{yy})} + \frac{4a(T_{xx}+T_{yy})}{4(1+a(T_{xx}+T_{yy}))} = \frac{1+a(T_{xx}+T_{yy})}{1+a(T_{xx}+T_{yy})} = 1$$

If we discretize the anisotropic Laplacian for the asymmetric scheme, we obtain the same result for the sum of the weights. Naturally, this also extends to our averaged scheme. So, we can ensure that our diffusion method is bounded by the old values of $q$, being stable no matter how large is the diffusion coefficient $\phi$ and the tensor boost $\beta$. A more complete demonstration can be found in [31].

If we look at the results of the computation of the Laplacian masks, we will also notice that, in all cases, the central element is negative, compensating for the contributions of its neighbors. Thus, there is no possibility whatsoever that our customized diffusion method can hinder the stability present in the original method. Moreover, the matrix of the system solved by the Gauss-Seidel iterative method is diagonally-dominant, which means it always converge to the solution. Our method is stable even for tensors with null eigenvalues. This could impact the exchanges between neighbors, since some cells would be able to share energy without receiving it. From a formal point of view, this discontinuity is not exactly interesting. A more careful evaluation is needed in order to assess the local implications of this scenario for animation purposes.



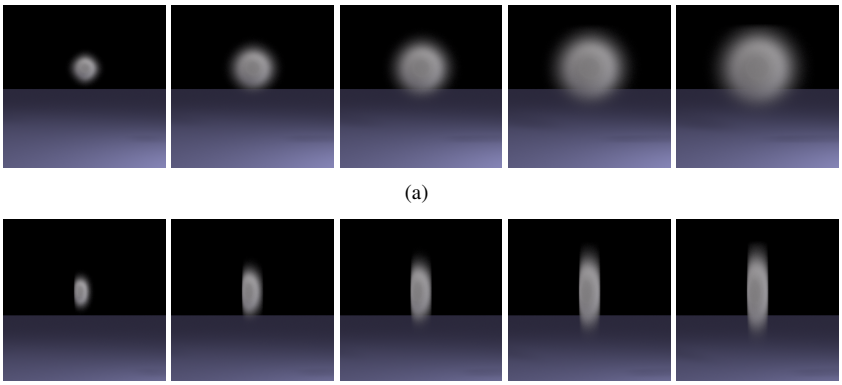

(a) (b)

Figure 9: Two tensor fields used for comparison with ordinary fluid simulations. They are shown here in reduced resolution to clarify their structures. Colors indicate the tensor norm. (a) 3-point field; (b) Vertical tube field.

## 3. Results

The experiments here described were conceived as a proof of concept of our work. First we compare our proposal to Stam's original method, and then we show how we can design tensor fields with the purpose of controlling simulations. Apart from our smoke renderings, which were produced via the Mitsuba [32] rendering software, all other figures use colors to indicate values of interest, such as fluid speed or tensor anisotropy. Colors range from blue (lowest value) to red (highest value), with intermediate values set on green.

In all simulations, we used a no-stick boundary condition, with stationary solid walls. Fluid is initially at rest and is put in motion by the application of some external force. Density insertion varies through different simulations, so we will describe it separately for each experiment. All simulations were run in a computer with an Intel Core i7-4700MQ processor, 8GB RAM and an NVIDIA GeForce GTX 765M video card. For most cases here presented, we were able to run at interactive times. This was more difficult for higher resolution fields, such as the $128 \times 128$ 2D disk field and the helical field, but this can be circumvented by taking advantage of parallelization. Also, the additional computations involving tensors do not affect significantly the performance during the steps of the simulation. For the disk field, for example, we obtained an average 4fps with or without the tensor advection. For the diffusion and projection parts, since our Laplacian masks are precomputed, the performance hit is unnoticeable. However, due to issues that will be discussed in Section 3.4, the customized projection was not used in the results that follow.



(a)

(b)

Figure 10: Diffusion sequence for the vertical tube in Fig. 9(b). Densities are inserted at grid center. (a) Simulation using Stam's diffusion. The smoke is diffused isotropically, completely unaware of the tensor field influence. (b) Simulation using the anisotropic masks for diffusion. The smoke is dispersed predominantly in the direction of the tube. Parameters: $\Delta t = 0.2$, $S = 10$.

### *3.1. Fluid dynamics comparisons*

The effects of our tensor-based advection become especially apparent when compared with the standard Stable Fluids method [19]. To illustrate this, we used two tensor fields: one defined by 3 points and a simple vertical field composed only of linear tensors in a tube surrounded by null tensors. Both fields are depicted in Fig. 9 with reduced resolution so that individual tensors are more easily distinguishable. These fields are colored based on their norm.

We separated the experiments in two parts. First, we leave the classic Stam advection in place and apply our anisotropic diffusion, comparing it to the original isotropic one. Then, we invert the setup: we use the tensor advection, without altering the diffusion step, to discuss the role of the tensor field in the advection process. In all cases, we leave the original isotropic projection step.

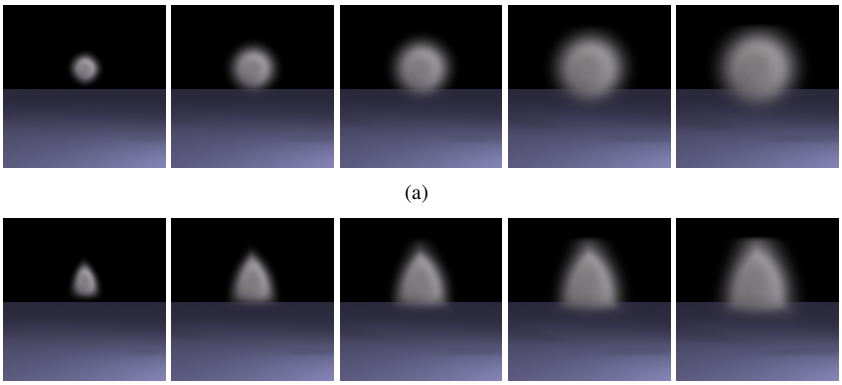
#### *3.1.1. Diffusion and viscosity*

First, let us consider the vertical tube field. Figure 10 shows the difference between the basic diffusion employed by Stam [19] and the anisotropic diffusion proposed in Section 2.3.

A similar comparison is performed for the 3-point field. This is a synthetic bidimensional tensor field, defined by three main points $\mathbf{q}_1 = (16, 16)$, $\mathbf{q}_2 = (48, 16)$ and $\mathbf{q}_3 = (32, 48)$ in a $64 \times 64$ grid. For each point $\mathbf{p}_{ij}$ in the grid, a tensor is computed as follows:

$$\mathbf{T}_{ij} = \sum_{k=1}^{3} \frac{(\mathbf{q}_k - \mathbf{p}_{ij}) \cdot (\mathbf{q}_k - \mathbf{p}_{ij})^T}{\|(\mathbf{q}_k - \mathbf{p}_{ij})\|}, \tag{21}$$

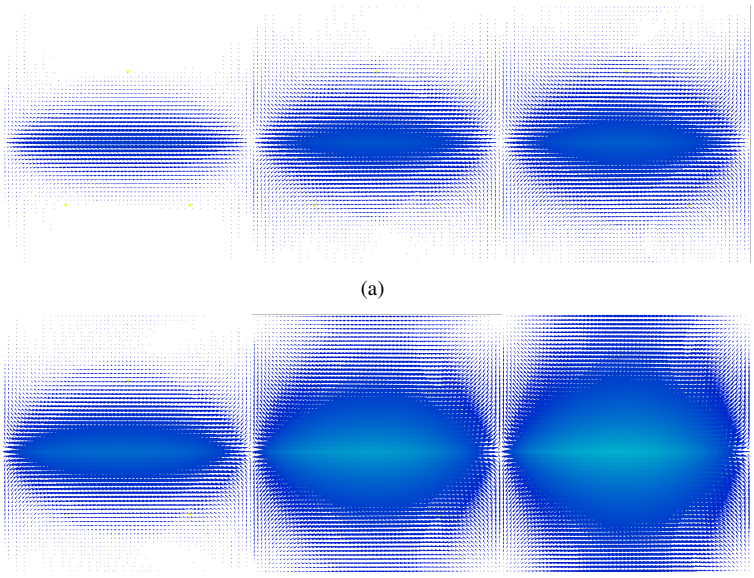
where $\mathbf{q}_k \neq \mathbf{p}_{ij}$. $\mathbf{T}_{ij}$ captures the uncertainty of the direction between grid point $\mathbf{p}_{ij}$ and every point $\mathbf{q}_k$. This uncertainty is weighed by the Euclidean distances between the points. Figure 11 shows the compared results.



(a)

(b)

Figure 11: Diffusion sequence with densities being constantly inserted in the grid center. (a) Simulation using Stam's diffusion, as in Fig. 10(a). (b) Simulation using the anisotropic diffusion for the 3-point field shown in 9(a). The fluid gradually assumes the shape of the central region of the field. Parameters: $\Delta t = 0.2$, $S = 10$.



(a)

(b)

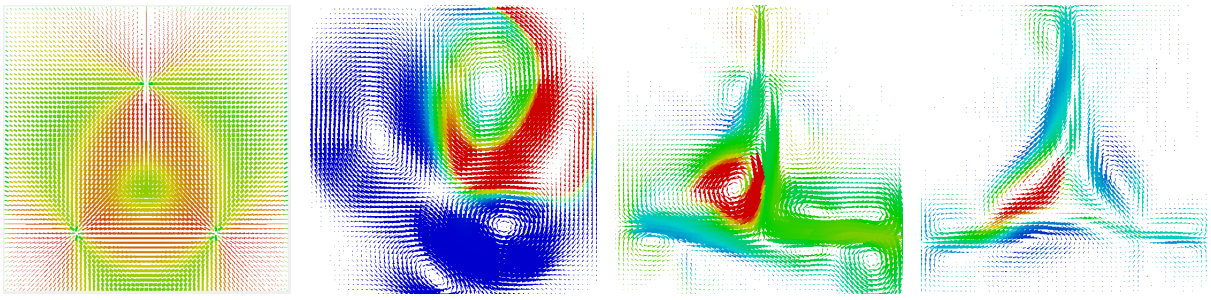
Figure 12: Comparison between isotropic (a) and anisotropic viscosity (b). While isotropic viscosity just reduces momentum equally in all directions, the anisotropic viscosity reduces it in specific directions. We show the evolution of both velocity fields at 3 iterations ($t = 5$, $t = 15$ and $t = 25$). The third frame is very close to steady state in both cases. Colors indicate the velocity norm.

.

In order to verify the effect of the anisotropic masks in the viscosity, we used the vertical tensor field with a constant force applied perpendicularly to the field. As we can see in Figure 12, our method diffuses the velocities in the direction of the tensors, while the original one just reduces momentum isotropically. The direction followed by the velocities in the default method is just the direction of the applied force. With anisotropic viscosity, the tensor is also taken into account in this process.

### *3.1.2. Advection*

Here we show the advantage of using the customized advection proposed in Section 2.2 over the possibility of using Eq. 3 as a force. Figure 13 shows the differences between these two approaches. Velocity vectors are displayed



(a) (b) (c) (d)

Figure 13: (a) $32 \times 32$ tensor field used in the simulation; (b) Classical fluid simulation, without using Eq. 3; (c) Simulation using Eq. 3 as an external force; (d) Simulation using Eq. 3 in advection. Notice how the simulations in Figures (c) and (d) generate a velocity field with a shape that resembles the collinear regions of the tensor field in (a). We can also see that this shape is much more accentuated when Eq. 3 is used in advection (d) than as an external force (c). Colors indicate level of anisotropy in (a) and density values in (b), (c) and (d).

for each grid cell in Figures 13(b), 13(c) and 13(d), with colors mapping the amount of density in the cell. In Fig. 13(a), colors indicate the degree of collinearity in a neighborhood of the tensor, measured by the lattice index defined in [33]. In Fig. 13(b), the advection acts freely, exactly as in the Stable Fluids method. In that case, the simulation is completely unaware of the underlying tensor field. In Figs. 13(c) and 13(d), instead, Eq. 3 is acting on the fluid, constraining fluid motion to the collinear regions marked in red in Fig. 13(a). The difference between the two tensor simulations is in the way tensor affects the fluid. Fig. 13(c) shows a simulation where Eq. 3 was applied as an external force (following the formulation in Eq. 4), while in Fig. 13(d) the advection is being molded by the proposed differential equation. While both approaches are capable of successfully controlling the fluid, affecting the advection enable a finer control. The velocities in the external force version tend to oscillate a bit more, forcing us to use very small time steps, so that the fluid can be adequately contained in the paths. Velocities in the tensor advection, however, are more directed, allows for larger time steps and dissipate quicker in planar regions.

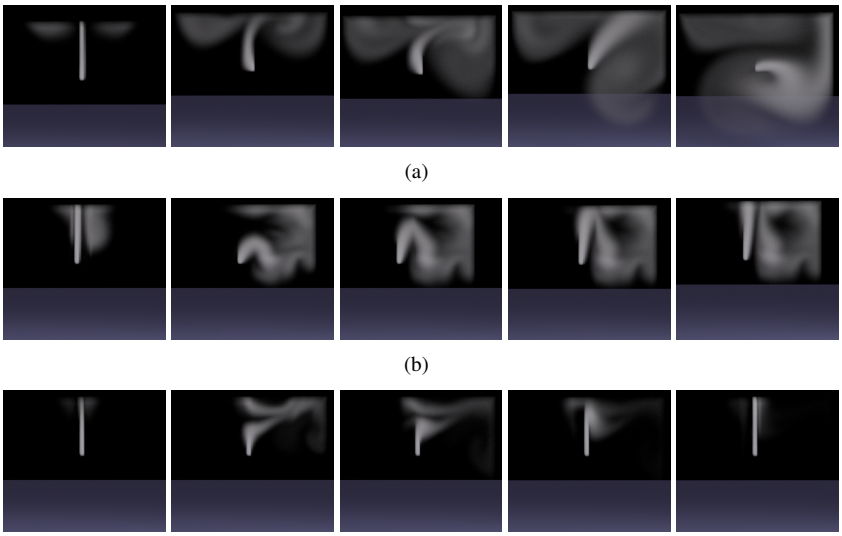
Another way to illustrate the effects of our method is by showing how densities are affected. In this case, we use the vertical tube field. Two separate simulations were conducted. Results are provided in Figures 14(a) and 14(b). The simulation in Fig. 14(a) shows the result of Stam's classical advection, while Fig. 14(b) shows our results. Fluid is inserted at the center of the grid and propelled upwards. At a certain moment, an external force is applied perpendicularly to the fluid path, in an attempt to change its trajectory. Notice how the fluid in Stam's simulation assumes a completely different direction when compelled by an external force. In our method, after the initial disturbance ceases, the fluid returns to its original path, responding to the tensor influence. There are some densities, though, that remain still outside the tube's path. As all tensors are null in those regions, advection does not influence them. A damping function can be used to smoothly dissipate these densities. Stam's method used such a function for dissipating densities in every cell in the grid, while we only apply it to null tensor regions.

### *3.2. Tensor field design for fluid control*

The addition of the tensor field complements the classic formulation, where we had basically three main variables: density, velocity and external forces. Now, we have four elements:

- the density field, with selected points for fluid injection;
- the velocity field, which represents initial fluid momentum;
- the external forces;
- and the tensor field, which limits fluid flow, i.e., it introduces new boundary conditions.

Thus, having these elements in mind, one could think of designing tensor fields to produce controlled fluid animations. Tensor information can be used to determine fluid injection or the external forces, with the orientational information of the tensor used to deflect the fluid, keeping it contained to the desired paths or regions. According



(a)

(b)

(c)

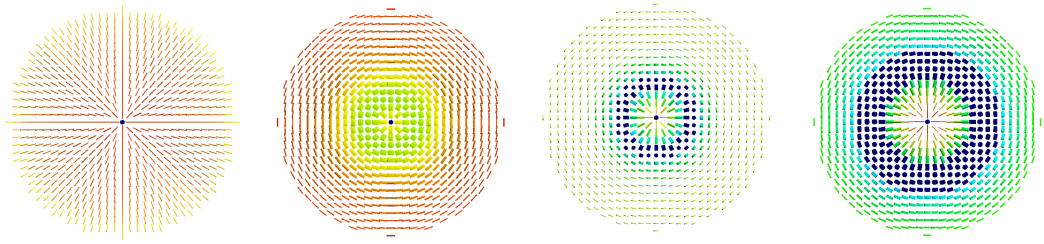
Figure 14: Animation sequence for the vertical tube in Fig. 9(b). (a) Simulation using Stam's advection. There is no tensor affecting the advection, thus the fluid loses its vertical path after being propelled by a force in a different direction. (b) Simulation using tensor advection. The method ensures that the fluid returns to its intended path, even after being deviated by the external force. No dissipation function was used, reason why densities accumulated outside the path after the force was applied. (c) Simulation using tensor advection, but with dissipation instead. The dissipation is applied everywhere outside the tube. Simulation parameters: $\Delta t = 0.05$, $S = 10$, $\beta = 1.3$ ($\beta$ does not influence simulation (a)).

to the motion pattern the user wants to apply to the fluid, different levels of anisotropy may be deemed necessary. In a general manner, linear tensors would be preferable for directed flow, while planar tensors are more useful for fluid containment. Isotropic tensors do not restrict fluid flow, so it is a good choice for filling regions with fluid. For example, if the user wants to simulate fluid flowing through pipes, linear tensors in the direction of intended flow should be considered. If, however, the user desires to create a fluidic shape by filling it with fluid, a certain dose of planar or isotropic tensors may be needed to adequately diffuse the fluid to all regions of the shape. Also, we could define the boundaries of a three-dimensional object by setting planar tensors around it to deflect the fluid along the borders, forming a kind of permeable wall. Naturally, most practical fields will be a combination of these kinds of tensors in order to obtain interesting simulations.

An example is the disk-shaped tensor field, composed by the sum of the 3 fields depicted in Figure 15. For visual clarity, we show these tensor fields in a smaller resolution. The field is composed of radial linear tensors near the center and perpendicular linear tensors following the circumference near the border, forming a transition section formed by planar tensors. The rationale here is that we want a tensor field that initially propels the fluid radially from the insertion point and then keeps it in a circling pattern, forming a disk shape. The mid region (depicted in dark blue) allows the fluid to flow more freely, with no specific associated direction. Further from the center, the field becomes more aligned with the disk tangent in order to trap the fluid and reveal its external boundary.

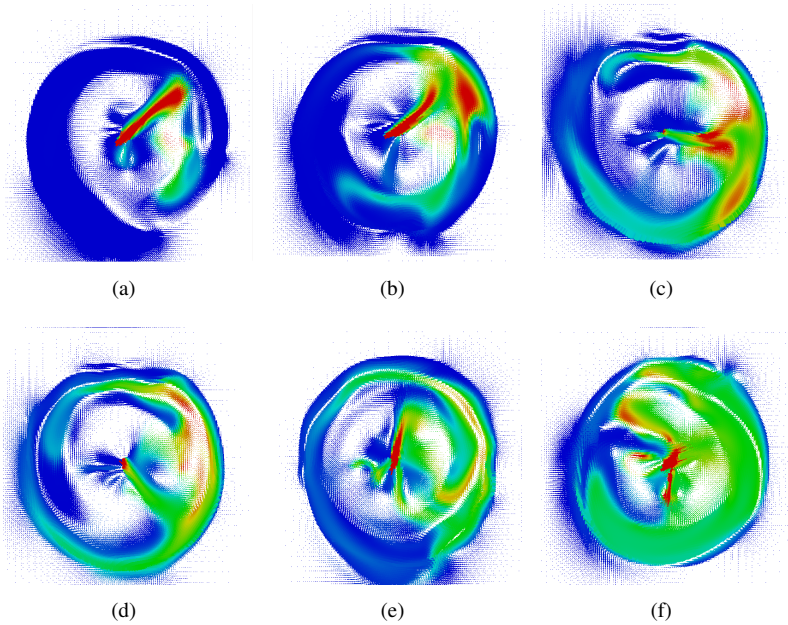
Let $d$ be the distance between grid cell $\mathbf{P}(i, j)$ and grid center $\mathbf{C}(N_x/2, N_y/2)$, where $N_x$ and $N_y$ are the grid dimensions. In order to obtain the disk, we define tensors for all cells within some radius $r$ from the center. All tensors outside the disk are set to isotropic tensors with negligible norm. The calculated distances are all normalized to the interval $[0, 1]$ and then applied to a Gaussian function to generate a factor $f$, which is used to set the eigenvalues in Eq. 1 in order to create the tensors. By varying the standard deviation $\sigma$, the mean $\mu$ and the eigenvalues to which we apply the Gaussian factor, we can obtain different distributions of tensors. If we look at Fig. 15, we see how changing these parameters influence the field design. In the case of Fig. 15(c), for example, as $\lambda_1 = f$ and $\lambda_2 = 1 - f$ are varying inversely, the orientation of the tensors gradually change from the center to the borders. Near the border, $\lambda_2$ is much bigger than $\lambda_1$, so the second eigenvector $\mathbf{e}_2$, which is perpendicular to the main eigenvector $\mathbf{e}_1$, will define a linear tensor oriented perpendicularly to the radial linear tensors near the disk center.

Naturally, this is just a way of mathematically generating a tensor field. Any other function could be used instead



(a) (b) (c) (d) Final tensor field

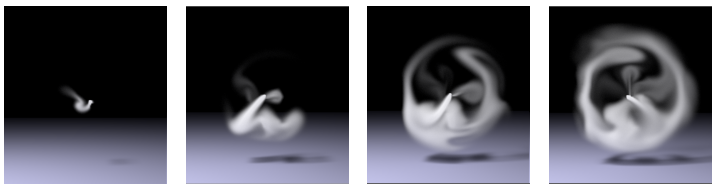
Figure 15: $48 \times 48$ tensor field (d) composed by the sum of fields (a), (b) and (c). For tensor field (a), $\mu = 0$, $\sigma = 0.4$, $\lambda_1 = f$ and $\lambda_2 = 0.1$; for (b), $\mu = 1$, $\sigma = 0.3$, $\lambda_1 = 0.1$ and $\lambda_2 = f$; for (c), $\mu = 0$, $\sigma = 0.1$, $\lambda_1 = f$ and $\lambda_2 = 1 - f$. As this is a bidimensional field, $\lambda_3$ was set to 0 for all of them.



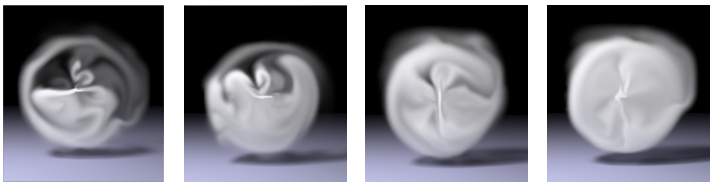
(a) (b) (c)

(d) (e) (f)

Figure 16: Velocity field in six different frames of a simulation with the disk tensor field. Color indicates densities, flowing from the center towards the borders, gradually filling the region. Simulation parameters: $\Delta t = 0.05$, $\beta = 3.0$, and $\alpha = 0.5$, $S = 100$.

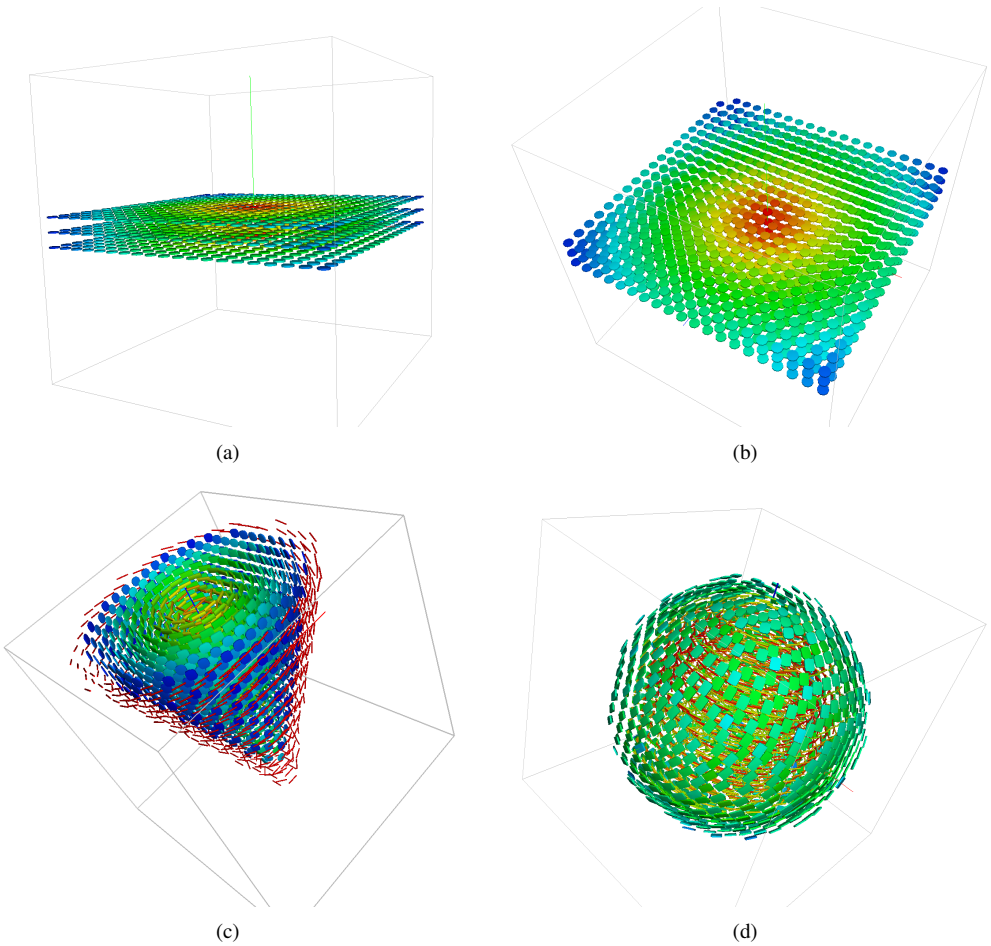


(a) $t = 50$ (b) $t = 200$ (c) $t = 300$ (d) $t = 450$



(e) $t = 550$ (f) $t = 650$ (g) $t = 800$ (h) $t = 1100$

Figure 17: Rendered densities for the disk simulation. The fluid gradually fills the region corresponding to the tensor field in Fig. 15(d). The fluid starts slowly, building momentum until it reaches a "final" state, where the shape is already formed but the fluid continues its circling motion in a stable manner. Frame (h) shows this state. Simulation parameters: $\Delta t = 0.05$, $\beta = 3.0$, $\alpha = 0.5$, $S = 100$.



(a) (b)

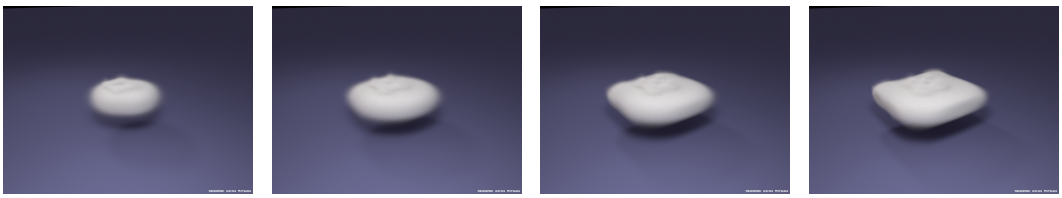
(c) (d)

Figure 18: $20 \times 20 \times 20$ 3D tensor fields: (a-b) wafer, (c) twister and (d) sphere.

to produce tensors, like in the 3-points tensor field (Fig. 13). The central idea behind the construction of all these fields is the definition of eigenvalues and eigenvectors for each tensor in a way that it serves the purposes of the simulation.

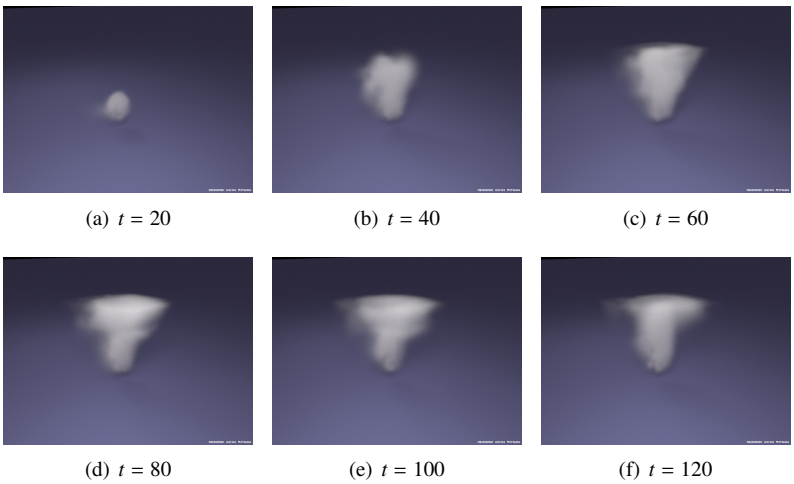
The velocity field for some frames of the disk simulation are displayed in Figure 16. We used a higher resolution tensor field ($128 \times 128$). Notice the circling pattern at its borders. Fluid density is inserted at the disk center and is propelled outwardly by the radial linear tensors at the core. Upon reaching the external layers, the fluid starts to move along the boundaries, circling the disk, as imposed by the tensors in that region. Figure 17 shows the rendered smoke for this example. In this case, the damping function was used to gradually dissipate densities outside the disk.

We have also designed some 3D fields to test our method, which are illustrated in Figure 18. The wafer field (Fig. 18(a) and 18(b)) is composed of parallel layers of planar tensors, with their norms reducing radially from the center. Planar tensors encourage diffusion in more than one direction, so we used them to stimulate fluid diffusion along the layer, entirely filling the shape. Fig. 19 shows some frames of our simulation for the wafer field.



(a) $t = 5$ (b) $t = 15$ (c) $t = 25$ (d) $t = 35$

Figure 19: Rendered densities for the wafer simulation. Parameters: $\Delta t = 0.1, \beta = 1.3, \alpha = 0.5, S = 100$.



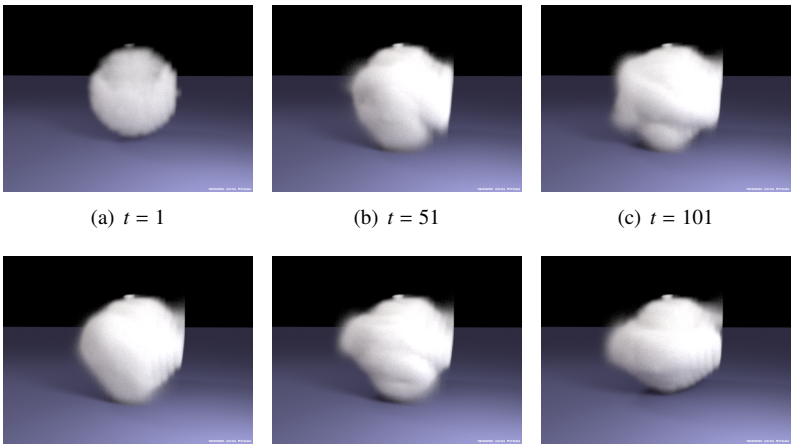
(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

(d) $t = 80$ (e) $t = 100$ (f) $t = 120$

Figure 20: Rendered densities for the twister simulation. Parameters: $\Delta t = 0.2, \beta = 5.0, \alpha = 0.1, S = 100$.

In Fig. 20 we can see the result of simulating the twister field (Fig. 18(c)). In this case, we applied a constant external force at the basis of the twister, which is also the density insertion point. The force is directed towards the top. Despite being vertically directed, the linear tensors that form the core of the twister produce a swirling motion, generating the expected vortex. The planar tensors at the boundaries of the tensor field are used to better contain the fluid. We also added a spiral of linear tensors around the twister so we could introduce some finer detail to the fluid motion at the boundaries.

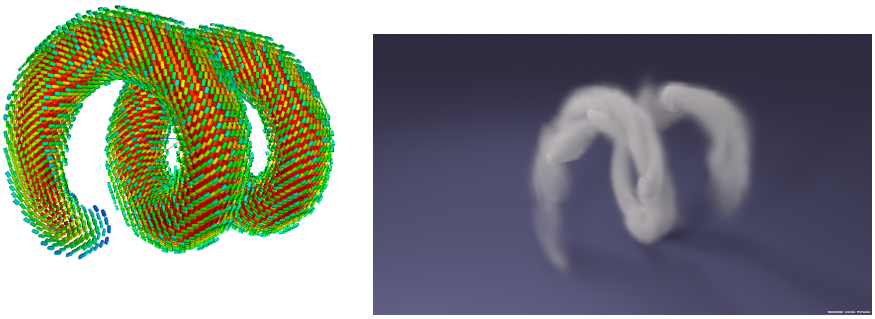
Finally, the spherical field (Fig. 18(d)) follows a similar rationale as used in the twister field. The interior of the sphere is composed of linear tensors, while the surface is made of planar tensors. The linear tensors here also have the objective of producing circular motion around the sphere. We applied radial forces in the center of the sphere, propelling the fluid outwards. The tensor advection generates velocities that encircle the sphere, forcing the fluid to assume a circular motion. Fluid is inserted throughout the whole volume of the sphere. We can notice some



(a) $t = 1$ (b) $t = 51$ (c) $t = 101$

(d) $t = 151$ (e) $t = 201$ (f) $t = 251$

Figure 21: Rendered densities for the sphere simulation. Parameters: $\Delta t = 0.1, \beta = 3.42, \alpha = 0.18, S = 1$.



(a) (b)

Figure 22: Helix tensor field visualization with direct application of our method. Parameters $\Delta t = 0.05$, $\beta = 6.0$ and $\alpha = 10.0$. (a) $38 \times 39 \times 40$ tensor field. (b) iteration 651.

fluid moving around the surface. Figures 18(d) and 21 shows, respectively, the tensor field and some frames of its simulation.

### *3.3. General 3D tensor fields*

A well known 3D dataset for testing DT-MRI based methods is the helix tensor field [34]. This synthetic field represent continuous trajectories and patches that are not intended for keeping the fluid inside the helix volume. However, it is possible to use our method to partially visualize the volume without any change or processing of the original field. To do this, we inserted 6 sources of density throughout the medial axis of the volume. This was needed to form a completely connected flow inside the helix since a long range current is not possible with only one source. The field trajectories tend to eject the density outside the volume. Notice that a large damping factor ($\alpha = 10.0$) is needed to visualize the field. The result, however, does not show the details of the field trajectories.

The simulation result shown in Fig. 22 is just a selected frame where the helix is visibly recognizable. In this case, the simulation is not able to keep the helix connected through all iterations, as we did in other experiments.

### *3.4. Known issues*

As we mentioned in the previous section, the results of our design method for 3D fields still needs improvements. Additional control mechanisms may be required in order to acquire the level of control seen in the 2D simulations,

since in 3D we have a higher number of degrees of freedom. When simulating fluid for the sphere field, for example, fluid can be seen circling the sphere's surface in thin sheets, but it fails to enclose the sphere as a whole unless we provide enough points of density insertion. By carefully adjusting simulation parameters, we can obtain results such as those demonstrated in Fig. 21, but in most cases we end up with a deformed sphere.

Further investigation must also be done regarding the application of the customized projection. Our proposed projection is mathematically consistent but we still have to analyze what conditions the tensor field needs for it to work smoothly. For example, discontinuities in the field may lead to problems, such as small localized instabilities, which may be aggravated depending on the simulation parameterization.

## 4. Conclusions

This work investigated the relationship between symmetric positive-definite tensors and fluids in the context of controlled fluid simulations. We presented a customized mathematical model based on tensor fields for describing fluid flow, which led to an adaptation of the classic Stable Fluids [19] method where tensors are treated as an inherent property of the medium, dictating fluid behavior in every step of the simulation. The goal was to obtain control without losing physical plausibility. While finding this balance is not an easy task, results demonstrate that it is possible to use tensors to achieve plausible simulations where fluids are directed through intended paths. However, even though fluid simulations are usually anchored on a proper choice of parameters, our method excessively relies on parameter configuration to obtain acceptable results. Our 2D simulations are more resilient to subpar parameterization, but that does not apply to 3D. We wish to explore ways of reducing this overdependence on parameters, enabling our method to work in less-controlled scenarios. We also intend to investigate the best ways of constructing tensor fields, since a well-thought field may help the proposed method on obtaining good simulations.

While the simple averaging scheme we used for the Laplacian mask calculation worked well in all experiments (including 3D), highly anisotropic tensors may present artifacts during the diffusion process. Moreover, there is a slightly perceptible drifting problem to the diffused values. It would be interesting to explore alternatives for the anisotropic diffusion, such as devising or testing more complete discrete Laplacian estimators, or different combinations of them. This is extremely relevant, since we conjecture that diffusion may be paramount to ensure that densities fill regions of the volume that are not reached by the advection alone.

It would be interesting to apply our method to different contexts in the future. Porous flow simulations are a possibility, considering the permeability tensor is exactly of the same form as the tensors we used in this work. Also, these path-oriented simulations can be very useful for enhancing DT-MRI tensor field visualizations, where highlighting collinear paths is usually the focus of interest. The poor resolution and information cluttering of these tensor fields often lead medical practitioners to resort to visualization techniques to better understand the data. However, these are also the reasons why it is difficult to provide good visualizations. Some preprocessing would probably be necessary in order to obtain satisfactory results.

While we aimed to achieve control solely by the use of tensors, one could also think of additional mechanisms to improve fluid adherence to the paths. A possibility would be the use of optimal control methods. In fact, some fluid-related works presented solutions based on this kind of method, such as the work by [5], which uses an optimal control approach known as the adjoint method. Another example is the application of optimal control to fluid flow in pipe networks [35]. Some recent works, such as [36, 37], contributed with adaptive fuzzy optimal control methods for strict-feedback nonlinear general systems. This can be a potential line of future investigation, since the aforementioned works can lead to reduced parameter dependence and can still guarantee stability.

## References


[1] A. Treuille, A. McNamara, Z. Popović, J. Stam, Keyframe control of smoke simulations, in: ACM Transactions on Graphics (TOG), vol. 22, ACM, 716–723, 2003.

[2] M. Browning, C. Barnes, S. Ritter, A. Finkelstein, Stylized keyframe animation of fluid simulations, in: Proceedings of the Workshop on Non-Photorealistic Animation and Rendering, ACM, 63–70, 2014.

[3] E. Shechtman, A. Rav-Acha, M. Irani, S. Seitz, Regenerative morphing, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 615–622, 2010.

[4] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, P. Sen, Image melding: Combining inconsistent images using patch-based synthesis., ACM Trans. Graph. 31 (4) (2012) 82–1.

[5] A. McNamara, A. Treuille, Z. Popović, J. Stam, Fluid control using the adjoint method, in: ACM Transactions On Graphics (TOG), vol. 23, ACM, 449–456, 2004.
[6] D. Bonilla, L. Velho, A. Nachbin, L. G. Nonato, Fluid warping, in: Proceedings of the IV Iberoamerican Symposium in Computer Graphics, 2009.
[7] D. Bonilla, L. Velho, Control methods for fluid-based image warping, in: Proceedings of WTD-Workshop of Theses and Dissertations-SIBGRAPI 2011, 2011.
[8] R. Fattal, D. Lischinski, Target-driven smoke animation, in: ACM Transactions on Graphics (TOG), vol. 23, ACM, 441–448, 2004.
[9] M. C. Renhe, A. Oliveira, C. Esperança, R. Marroquim, Enhanced Target Driven Smoke Morphing, in: Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on, IEEE, 213–220, 2012.
[10] Y. Kim, R. Machiraju, D. Thompson, Path-based control of smoke simulations, in: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, 33–42, 2006.
[11] Z. Yuan, F. Chen, Y. Zhao, Pattern-guided smoke animation with lagrangian coherent structure, in: ACM transactions on graphics (TOG), vol. 30, ACM, 136, 2011.
[12] B. Yang, Y. Liu, L. You, X. Jin, A unified smoke control method based on signed distance field, Computers & Graphics 37 (7) (2013) 775–786.
[13] S. Sato, Y. Dobashi, K. Iwasaki, T. Yamamoto, T. Nishita, Deformation of 2D flow fields using stream functions, in: SIGGRAPH Asia 2014 Technical Briefs, ACM, 4, 2014.
[14] S. Sato, Y. Dobashi, Y. Yue, K. Iwasaki, T. Nishita, Incompressibility-preserving deformation for fluid flows using vector potentials, The Visual Computer 31 (6-8) (2015) 959–965.
[15] S. Sato, Y. Dobashi, T. Nishita, A combining method of fluid animations by interpolating flow fields, in: SIGGRAPH ASIA 2016 Technical Briefs, ACM, 4, 2016.
[16] N. S. Hageman, A. W. Toga, K. L. Narr, D. W. Shattuck, A diffusion tensor imaging tractography algorithm based on Navier–Stokes fluid mechanics, IEEE transactions on medical imaging 28 (3) (2009) 348–360.
[17] M. C. Renhe, J. L. de Souza Filho, M. B. Vieira, A. Oliveira, Tensor field visualization using Eulerian fluid simulation, in: Computational Science and Its Applications–ICCSA 2013, Springer, 332–347, 2013.
[18] C. Westin, S. Peled, H. Gudbjartsson, R. Kikinis, F. Jolesz, Geometrical diffusion measures for MRI from tensor basis analysis, in: Proceedings of ISMRM, vol. 97, 1742, 1997.
[19] J. Stam, Stable fluids, in: Proceedings of the 26th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., 121–128, 1999.
[20] S. Günter, Q. Yu, J. Krüger, K. Lackner, Modelling of heat transport in magnetised plasmas using non-aligned coordinates, Journal of Computational Physics 209 (1) (2005) 354–370.
[21] J. Bonelle, Compatible Discrete Operator schemes on polyhedral meshes for elliptic and Stokes equations, Ph.D. thesis, Université Paris-Est, 2014.
[22] K. Lipnikov, G. Manzini, M. Shashkov, Mimetic finite difference method, Journal of Computational Physics 257 (2014) 1163–1227.
[23] W. Press, B. Flannery, S. Teukolsky, W. Vetterling, Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, ISBN 0521431085, 1992.
[24] A. Selle, R. Fedkiw, B. Kim, Y. Liu, J. Rossignac, An unconditionally stable MacCormack method, Journal of Scientific Computing 35 (2-3) (2008) 350–371.
[25] J. U. Brackbill, D. B. Kothe, H. M. Ruppel, FLIP: a low-dissipation, particle-in-cell method for fluid flow, Computer Physics Communications 48 (1) (1988) 25–38.
[26] C. Jiang, C. Schroeder, A. Selle, J. Teran, A. Stomakhin, The affine particle-in-cell method, ACM Transactions on Graphics (TOG) 34 (4) (2015) 51.
[27] J. Stam, Real-time fluid dynamics for games, in: Proceedings of the game developer conference, vol. 18, 25, 2003.
[28] B. van Es, B. Koren, H. J. de Blank, Finite-difference schemes for anisotropic diffusion, Journal of Computational Physics 272 (2014) 526–549.
[29] R. Bridson, Fluid Simulation for Computer Graphics, CRC Press, 2008.
[30] N. Foster, R. Fedkiw, Practical animation of liquids, in: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, 23–30, 2001.
[31] M. C. Renhe, A stable tensor-based deflection model for controlled fluid simulations, Universidade Federal do Rio de Janeiro, Rio de Janeiro, URL `http://www.cos.ufrj.br/uploadfile/publicacao/2795.pdf`, 2017.
[32] W. Jakob, Mitsuba renderer, http://www.mitsuba-renderer.org, 2010.
[33] C. Pierpaoli, P. J. Basser, Toward a quantitative assessment of diffusion anisotropy, Magnetic resonance in Medicine 36 (6) (1996) 893–906.
[34] G. Kindlmann, Diffusion tensor MRI datasets, `http://www.sci.utah.edu/~gk/DTI-data/`, 2017.
[35] F. M. Hante, G. Leugering, A. Martin, L. Schewe, M. Schmidt, Challenges in optimal control problems for gas and fluid flow in networks of pipes and canals: From modeling to industrial applications, in: Industrial Mathematics and Complex Systems, Springer, 77–122, 2017.
[36] K. Sun, Y. Li, S. Tong, Fuzzy Adaptive Output Feedback Optimal Control Design for Strict-Feedback Nonlinear Systems., IEEE Trans. Systems, Man, and Cybernetics: Systems 47 (1) (2017) 33–44.
[37] K. Sun, S. Sui, S. Tong, Fuzzy adaptive decentralized optimal control for strict feedback nonlinear large-scale systems, IEEE Trans. Cybern 48 (4) (2018) 1326–1339.