Realistic Smoke Simulation Using A Frustum Aligned Grid

by

Alan Wai Lun Woo

B.Sc., University of British Columbia, 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Computer Science)

The University of British Columbia

April 2006

Abstract

Realistic simulation of smoke is used in the special effects industry to produce smoke in both feature films and video games. Traditional simulations utilize uniformly spaced rectangular computational grids to perform the smoke simulation. Various changes had been proposed to improve different aspects of the simulation, including level of details, memory usage and simulation speed. In this thesis, I propose a novel computational grid that improves upon the level of details as well as memory usage. I propose a frustum aligned grid that takes advantage of the viewing camera because details are most important in the area close to the camera. A frustum aligned grid reduces the amount of grid points necessary to cover the whole domain by placing a high concentration of grid points near the camera while having sparse grid points away from the camera. By using a larger number of grid lines in the direction parallel to the camera and fewer grid lines in the direction perpendicular to the camera, high level of details using a smaller amount of memory can be achieved. The grid is logically rectangular and a perspective transformation can map the grid into a spatially rectangular one. These properties enable the use of existing simulation tools with some modifications, thus maintaining the level of speed. Experimental results and comparison with a standard uniform grid demonstrate the practicality and effectiveness of the proposed method.

Contents

List of Tables

List of Figures

Acknowledgements

I'd like to acknowledge my very understanding supervisor Robert Bridson, for all his help and advice.

Alan Wai Lun Woo

The University of British Columbia
April 2006

Chapter 1

Introduction

Smoke simulation techniques have made various advancements in recent years to a point where high-quality realistic smoke can be simulated in reasonable time. The major breakthroughs come from the introduction of the semi-lagrangian method by Stam [8] and vorticity confinement by Fedkiw et al[1]. Together, they form the basic backbone of simulation methods that are utilized today.

The memory requirement is high traditionally and various techniques have been developed to address this problem. Rasmussen [6] developed a technique to reduce the memory usage by using interpolated two dimensional flow fields instead of a full three dimensional one. The two dimensional flow fields are combined with a tiled Kolmogorov velocity field to produce large-scale phenomena such as nuclear explosions. This method is inherently limited by its interpolative nature, thus missing small scale details that reside in the physical space between the two dimensional flow fields. The octree data structure introduced by Losasso [5] uses large grid cells in area of low activities and smaller grid cells in areas with interesting flow, reducing the overall number of cells used to divide the computational domain. However, the definition of interesting flow and criteria for refinement are difficult to define

accurately to capture the required details.

Recently, Selle [7] combined the vortex particle method with an Eulerian grid to produce highly turbulent effects. The vorticity form of the Navier-Stokes equations is solved to obtain velocity for advection and vorticity particles are used to conserve the vorticity of the flow. A vorticity conserving force is used to drive the grid vorticity to match the particles. Feldman [2] introduces fluid simulation using unstructured tetrahedral grid and in combination with regular hexahedral grid. This method allows for simulation in irregular domains that are difficult with traditional grids.

We introduce a novel computational grid that addresses the memory issue as well as level of details. A frustum aligned grid is used to take advantage of the viewing camera. Unlike octree, our grid has a defined area of interest, the front of the frustum, where we place high concentration of small grid cells to capture the details. Progressively larger grid cells are placed towards the back of the frustum to reduce the number of cells necessary to divide the computational domain. However, the structure of the grid does impose additional overheads in the computation, but they can be compromised by cutting back the grid dimension perpendicular to the camera. Since the grid is aligned with the camera, the dimensions parallel to the camera play a more vital role in maintaining the level of details and cutting back the grid dimension perpendicular to the camera has minimal impact on the quality of the images. Divergence and gradient operators on the frustum aligned grid are developed based on the orthogonal decomposition theorems by Hyman and Shashkov [4].

Chapter 2 will define the traditional simulation method, chapter 3 will outline the details of the frustum aligned grid method, chapter 4 will present some

experimental results and chapter 5 will conclude and discuss some future work.

Chapter 2

Traditional Smoke Simulation

Smoke simulations have been used to produce realistic looking smoke for many years and they all generally follow the framework discussed in this chapter.

2.1 Governing Equations

Smoke is composed of tiny soot particles suspended in air, which behaves according to a set of governing equations known as the Navier-Stokes equations.

$$V_t + (V \cdot \nabla)V + \frac{\nabla P}{\rho} = \frac{(\nabla \cdot \tau)}{\rho} + f$$

where ρ denotes density and τ denotes viscous stress tensors.

Viscosity and compressibility are negligible in smoke and density can be assumed to be constant; therefore the equation can be simplified into the incompressible Euler equations that conserve both mass and momentum.

$$V_t + (V \cdot \nabla)V + \nabla P = f$$

$$\nabla \cdot V = 0$$

where V = (u, v, w) denotes the velocity, P denotes rescaled pressure and f denotes any external forces such as buoyancy.

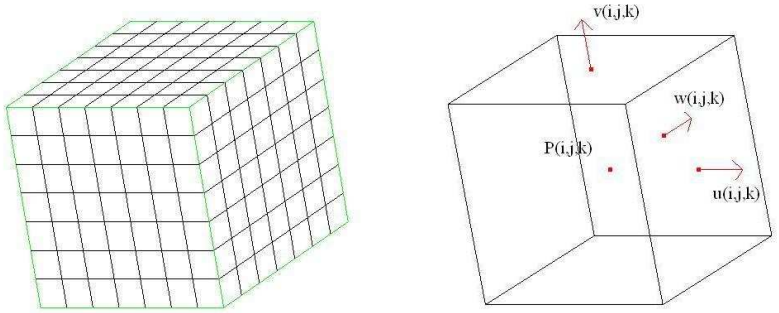

Figure 2.1: Computational domain and staggered grid alignment.

2.2 The Grid

Traditional simulations divide the computational domain into equal volume cells. Velocity and pressure are arranged in a staggered grid alignment or MAC grid [3] where velocities lie perpendicular to the faces of a cell and pressures lie at cell centers. See figure 2.1. This arrangement ensures the correct quantity will be at the correct location during the computation. Average velocity and divergence of velocity will be collocated with pressure, and gradient of pressure will be collocated with components of velocity.

2.3 Simulation Loop

Simulating smoke requires solving the incompressible Euler equations for each frame of the simulation. The typical frame rate of an animation is 24 frames per second. To solve the Euler equations, a projection method is used. The equations are broken

up into parts and solved in a few steps.

$$V^1 = V^n - (V \cdot \nabla)V\Delta t$$
$$V^* = V^1 + f\Delta t$$
$$\nabla^2 P^* = \nabla \cdot V^*$$
$$V^{n+1} = V^* - \nabla P^*$$

The first step is to advect the velocity using the semi-lagrangian method discussed in the next section. By ignoring the pressure term, we can reduce the Euler equations into

$$\frac{V^* - V^n}{\Delta t} + (V \cdot \nabla)V = f$$

and rearrange to get

$$V^* = V^n - (V \cdot \nabla)V\Delta t + f\Delta t$$

which can be broken up into

$$V^1 = V^n - (V \cdot \nabla)V\Delta t$$
$$V^* = V^1 + f\Delta t$$

An intermediate velocity field V^1 is found through advection and external forces are added to it. External forces usually include buoyancy and vorticity confinement, which will be discussed in section 2.5.

Boundary conditions are then applied to the velocity and pressure. Types of boundary conditions include Dirichlet, where values are specified, and Neumann, where the normal derivative is specified. In a smoke simulation, typical boundary condition used for pressure is $\frac{\partial P}{\partial N} = 0$ for all boundaries and typical boundary conditions used for velocities are $\frac{\partial V}{\partial N} = 0$ for open boundaries and velocity = 0 for the ground.

After boundary conditions are applied, the correct pressure to force a divergence free velocity is computed.

$$\frac{V^{n+1}-V^*}{\Delta t} + \nabla P = 0$$
$$V^{n+1} - V^* + \nabla P \Delta t = 0$$
$$V^{n+1} + \nabla P \Delta t = V^*$$

By taking the divergence of the equation

$$\nabla \cdot V^{n+1} + \nabla^2 P \Delta t = \nabla \cdot V^*$$

and using the fact that V^{n+1} should be divergence free,

$$\nabla \cdot V^{n+1} = 0$$

we can derive the Poisson equation required to solve for the pressure.

$$\nabla^2 P \Delta t = \nabla \cdot V^*$$

A time scaled pressure $P^* = P\Delta t$ is usually used to simplify the computation.

The divergence of a cell can be computed as

$$\begin{aligned} \nabla \cdot V^*_{ijk} &= \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} \cdot \begin{pmatrix} u \\ v \\ w \end{pmatrix} \\ &= u_{i+1jk} - u_{ijk} + v_{ij+1k} - v_{ijk} + w_{ijk+1} - w_{ijk} \end{aligned}$$

and the Laplacian of pressure can be computed as

$$\nabla^2 P^* = 6P_{ijk} - P_{i-1jk} - P_{i+1jk} - P_{ij-1k} - P_{ij+1k} - P_{ijk-1} - P_{ijk+1}$$

The pressure is then defined by

$$P_{ijk} = \frac{\nabla \cdot V^*_{ijk} + P_{i-1jk} + P_{i+1jk} + P_{ij-1k} + P_{ij+1k} + P_{ijk-1} + P_{ijk+1}}{6}$$

The Poisson equation can be solved using methods such as Gauss-Seidel relaxation, multigrid or conjugate gradient.

The final step in the simulation is to update the velocity by subtracting the pressure gradient.

$$V^{n+1} = V^* - \nabla P \Delta t$$

2.4 Advection

Advection of quantities such as velocity and smoke density are carried out using the semi-lagrangian method introduced by Stam. Semi-lagrangian method is a blend between Eulerian methods, which keep track of quantities at fixed grid points, and Lagrangian methods, which keep track of the location of the quantities over time. This method aims to eliminate the instability and time step restriction in the computation. In explicit Eulerian methods, the time step is restricted by the spacing of the grid nodes. Lagrangian methods have no time step restriction but the lack of a regularly sampled grid poses other challenges. By using fixed grid nodes as in Eulerian methods and tracing the quantity backward similar to Lagrangian methods, the semi-lagrangian method is able to combine the best of both classes of methods into a stable and efficient package. Some disadvantages of the method include numerical dissipation from interpolation and failure to conserve mass and energy.

To advect a quantity, the velocity field at the specific location has to be computed. Since smoke densities are located at the cell centers, averaged cell center velocities need to be derived.

$$V_{ijk} = \begin{pmatrix} u_{i+\frac{1}{2}jk} \\ v_{ij+\frac{1}{2}k} \\ w_{ijk+\frac{1}{2}} \end{pmatrix} = \begin{pmatrix} \frac{u_{ijk}+u_{i+1jk}}{2} \\ \frac{v_{ijk}+v_{ij+1k}}{2} \\ \frac{w_{ijk}+w_{ijk+1}}{2} \end{pmatrix}$$

This velocity is used to trace back to where the quantity is coming from.

$$X^* = X_{ijk} - V_{ijk}\Delta t$$

Interpolation of the eight neighboring values is used to update the current value of the quantity. Trilinear or cubic spline interpolations are normally used but any higher degree interpolation can be used to increase the accuracy of the simulation.

Advection of velocities is performed in a similar fashion with averaged velocity field defined at the appropriate cell faces. For example, the velocity field at the location of U_{i+1jk} is defined as

$$V_{i+\frac{1}{2}j+\frac{1}{2}k+\frac{1}{2}} = \begin{pmatrix} u_{i+1jk} \\ v_{i+\frac{1}{2}j+\frac{1}{2}k} \\ w_{i+\frac{1}{2}jk+\frac{1}{2}} \end{pmatrix} = \begin{pmatrix} u_{i+1jk} \\ \frac{v_{ijk}+v_{ij+1k}+v_{i+1jk}+v_{i+1j+1k}}{4} \\ \frac{w_{ijk}+w_{ijk+1}+w_{i+1jk}+w_{i+1jk+1}}{4} \end{pmatrix}$$

2.5 Vorticity Confinement

Vorticity confinement is introduced by Fedkiw et al. in 2000 to attack the problem of small scale details. Since the computation involves a fair amount of averaging or interpolating of quantities, small scale details are lost in the numerical dissipation. Vorticity confinement is an effort to reintroduce the lost details by applying a confinement force to areas of high vorticity. The force is proportional to the grid spacing and disappears as the spacing decreases due to increased grid size. This property is consistent with the Euler equations.

The first step is to compute the vorticity of each cell using cell center velocity averaged from the faces.

$$V^c_{ijk} = \begin{pmatrix} U_{ijk} \\ V_{ijk} \\ W_{ijk} \end{pmatrix}$$

$$\omega = \nabla \times V = \begin{pmatrix} \frac{W_{ij+1k} - W_{ij-1k} - V_{ijk+1} + V_{ijk-1}}{2} \\ \frac{U_{ijk+1} - U_{ijk-1} - W_{i+1jk} + W_{i-1jk}}{2} \\ \frac{V_{i+1jk} - V_{i-1jk} - U_{ij+1k} + U_{ij-1k}}{2} \end{pmatrix}$$

Then vorticity location vectors that point toward higher vorticity concentration are computed.

$$\Omega = \frac{\nabla|\omega|}{|\nabla|\omega||}$$

Finally, a confinement force is computed as

$$f = \epsilon h \left(\Omega \times \omega\right)$$

where h is the grid spacing. The force is then applied to the velocities.

$$\begin{pmatrix} u_{ijk} \\ v_{ijk} \\ w_{ijk} \end{pmatrix} += \Delta t \begin{pmatrix} \frac{f^x_{ijk} + f^x_{i-1jk}}{2} \\ \frac{f^y_{ijk} + f^y_{ij-1k}}{2} \\ \frac{f^z_{ijk} + f^z_{ijk-1}}{2} \end{pmatrix}$$

The parameter ϵ controls the amount of confinement force and it can have significant influence on the resulting smoke appearance. As ϵ increases the amount of activity in the smoke increases and the smoke appears to be more alive. But if ϵ is set too high, it will dominate the other quantities and causes the smoke to appear unrealistic or even uncharacteristic.

Chapter 3

Frustum Aligned Grid Method

Like vorticity confinement discussed in the last chapter, many other methods exist that try to improve the realism or efficiency of a smoke simulation. Examples of such methods include the vortex particle method, octree data structure and unstructured grids, among others. The Frustum Aligned Grid method is a method that takes advantage of the viewing camera and tries to improve on both realism and efficiency.

3.1 Frustum Aligned Grid

Since the viewing camera can only see so much at a particular instance, we can concentrate our simulation within the area of interest: the frustum. The frustum can be described using four parameters: field of view, aspect ratio, near clipping plane and far clipping plane. Field of view defines the angle to the left and right of vertical that the camera is capable of capturing. Typical cameras have a field of view of 30 degrees or less. Aspect ratio defines the ratio of width to height. Common aspect ratios are 4:3 for television and 16:9 for widescreen display. Near and far clipping planes defines the closest and farthest distance the camera could

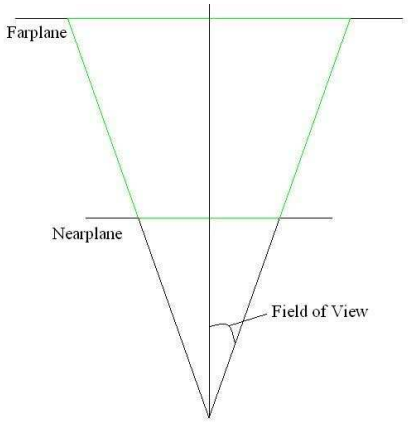

Figure 3.1: Frustum parameters

see. See figure 3.1.

In computer graphics, a perspective transformation is used to transform a frustum into a rectangular domain. Generally, the rectangular domain will have dimension of -1 to 1 in all directions. The perspective transformation can be defined as

$$M_{perspective} = \begin{pmatrix} \frac{-1}{\tan fov} & 0 & 0 & 0 \\ 0 & \frac{-aspect}{\tan fov} & 0 & 0 \\ 0 & 0 & \frac{-(farplane+nearplane)}{farplane-nearplane} & \frac{2(farplane)(nearplane)}{farplane-nearplane} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

and its inverse can be defined as

$$M^{-1}_{perspective} = \begin{pmatrix} -\tan fov & 0 & 0 & 0 \\ 0 & \frac{-\tan fov}{aspect} & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & \frac{farplane-nearplane}{2(farplane)(nearplane)} & \frac{-(farplane+nearplane)}{2(farplane)(nearplane)} \end{pmatrix}$$

These two matrices are used to convert position between grid space and real space.

As in traditional simulations, the computational domain is divided into cells.

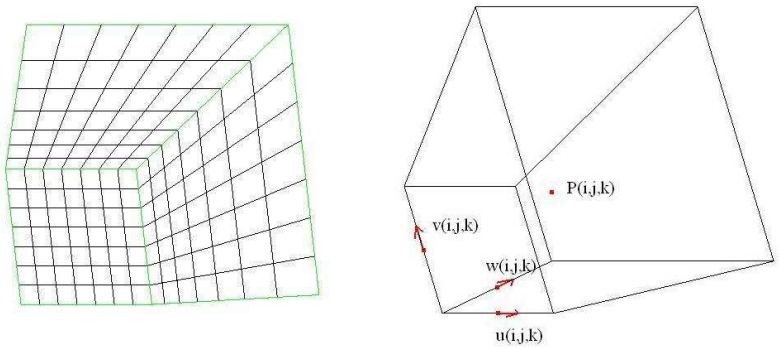

Figure 3.2: Frustum aligned grid

We divide the domain using grid lines that will transform to equally spaced grid lines in the rectangular grid space. The grid spacing in all directions will increase gradually from the front to the back of the domain. This produces cells with smaller volume near the camera and cells with larger volume away from the camera. This also produces denser grid nodes near the camera and sparser grid nodes away from the camera.

Velocities and pressure are still arranged in a staggered grid alignment but velocities lay along the grid lines and pressure resides at the grid nodes. See figure 3.2. This can be view as a shifted version of the standard MAC grid. If a cell is placed around the node, the pressure will lay in the center and velocities at the faces. In traditional method, this shift would not cause any change in the simulation process.

Since the grid lines are not orthogonal, velocities values are the projection of the real world values onto the directions of the grid lines. The axes for any grid nodes are $\xi = (1, 0, 0)$, $\eta = (0, 1, 0)$ and $\zeta = (x, y, z)$, where x, y and z varies depending on the location of the grid node. The relationship between velocities and

real world values can be described as

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & y & z \end{pmatrix} \begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix}$$

and more usefully

$$\begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{-x}{z} & \frac{-y}{z} & \frac{1}{z} \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix}$$

since we will need these conversions when performing later computations.

3.2 Simulation Loop

The structure of the simulation loop of the frustum aligned grid method is identical to traditional methods. Essentially, it is still solving the incompressible Euler equations using the projection method. But since the physical domain and the computational grid is different, computations such as advection, vorticity, divergence and pressure solve need to be redefined. The following sections describe each in detail.

3.3 Divergence

Divergence is the rate of change of a control volume. In a frustum aligned grid, the control volume is the volume around a grid node and the flux is related to the velocities on the grid lines. The divergence operator using the frustum align grid is

defined as

$$\nabla \cdot V = -N^{-1}\nabla^T LV = -N^{-1}\left(\frac{\partial}{\partial x}\frac{\partial}{\partial y}\frac{\partial}{\partial z}\right)\begin{pmatrix} L_{11} & L_{12} & L_{13} \\ L_{21} & L_{22} & L_{23} \\ L_{31} & L_{32} & L_{33} \end{pmatrix}\begin{pmatrix} u \\ v \\ w \end{pmatrix}$$

where N is the volume of the node, T is the transpose and L is a 3 by 3 matrix that relates the formal and natural inner product of velocities.

To compute L, we must first define the natural and formal inner product of the domain. The first step in deriving the natural inner product of two velocities X and Y is to convert them into real world values X^r and Y^r using the matrix defined in the section 3.1.

$$\begin{pmatrix} X_u^r \\ X_v^r \\ X_w^r \end{pmatrix} = \begin{pmatrix} X_u \\ X_v \\ \frac{-X_u(x)-X_v(y)+X_w}{z} \end{pmatrix} \text{ and } \begin{pmatrix} Y_u^r \\ Y_v^r \\ Y_w^r \end{pmatrix} = \begin{pmatrix} Y_u \\ Y_v \\ \frac{-Y_u(x)-Y_v(y)+Y_w}{z} \end{pmatrix}$$

where ζ = (x, y, z) is the unit vector of the direction of the grid line. We can then take the inner product of the two values to get

$$(X^r \cdot Y^r) = \begin{pmatrix} X_u & X_v & \frac{-X_u(x)-X_v(y)+X_w}{z} \end{pmatrix}\begin{pmatrix} Y_u \\ Y_v \\ \frac{-Y_u(x)-Y_v(y)+Y_w}{z} \end{pmatrix}$$

$$= \left(1+\frac{x^2}{z^2}\right)X_uY_u + \left(1+\frac{y^2}{z^2}\right)X_vY_v + \frac{1}{z^2}X_wY_w +$$

$$\frac{xy}{z^2}(X_uY_v + X_vY_u) + \frac{-x}{z^2}(X_uY_w + X_wY_u) + \frac{-y}{z^2}(X_vY_w + X_wY_v)$$

This defines the natural inner product at a specific grid node and axis alignment. To compute the natural inner product of a cell, the natural inner product of the eight corners of the cell are weighted by the volume it occupies and then summed. The natural inner product of the domain is obtained by summing the natural inner product of all the cells in the domain.

The formal inner product of the domain is defined as

$$[X^r \cdot Y^r] = \Sigma_u \Sigma_v \Sigma_w (X_u Y_u + X_v Y_v + X_w Y_w)$$

which sums the inner product over the domain. The formal and natural inner product of the domain is related by

$$(X^r \cdot Y^r) = [LX^r \cdot Y^r]$$

Applying the matrix L and expanding the equation, we get

$$[LX^r \cdot Y^r] = \left[\left(\begin{pmatrix} L_{11} & L_{12} & L_{13} \\ L_{21} & L_{22} & L_{23} \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} X_u \\ X_v \\ X_w \end{pmatrix} \right)^T \begin{pmatrix} Y_u \\ Y_v \\ Y_w \end{pmatrix} \right]$$

$$= \Sigma_u \Sigma_v \Sigma_w (L_{11} X_u Y_u + L_{12} X_v Y_u + L_{13} X_w Y_u + L_{21} X_u Y_v + L_{22} X_v Y_v +$$

$$L_{23} X_w Y_v + L_{31} X_u Y_w + L_{32} X_v Y_w + L_{33} X_w Y_w)$$

The explicit formulas for L can be derived by comparing the natural and formal inner products. To derive L_{11}, we need to calculate the equivalent of $\Sigma_u \Sigma_v \Sigma_w L_{11} X_u Y_u$ in terms of the natural inner products. We need to look at the eight axis configurations of natural inner products that contain the same $X_u Y_u$ term and weight them according to their volume within the cell. For $X^u_{ijk} Y^u_{ijk}$, the eight axis configurations are

ξ	ζ	η	$\frac{Vol_k^x}{Vol_k}(1+\frac{x^2}{z^2})$
$(1,0,0)$	$(0,1,0)$	(x_{ij}, y_{ij}, z_{ij})	$\frac{Vol_k^0}{Vol_k}(1+\frac{x_{ij}^2}{z_{ij}^2})$
$(1,0,0)$	$(0,1,0)$	$(-x_{ij}, -y_{ij}, -z_{ij})$	$\frac{Vol_{k-1}^1}{Vol_{k-1}}(1+\frac{x_{ij}^2}{z_{ij}^2})$
$(1,0,0)$	$(0,-1,0)$	(x_{ij}, y_{ij}, z_{ij})	$\frac{Vol_k^0}{Vol_k}(1+\frac{x_{ij}^2}{z_{ij}^2})$
$(1,0,0)$	$(0,-1,0)$	$(-x_{ij}, -y_{ij}, -z_{ij})$	$\frac{Vol_{k-1}^1}{Vol_{k-1}}(1+\frac{x_{ij}^2}{z_{ij}^2})$
$(-1,0,0)$	$(0,1,0)$	$(x_{i+1j}, y_{i+1j}, z_{i+1j})$	$\frac{Vol_k^0}{Vol_k}(1+\frac{x_{i+1j}^2}{z_{i+1j}^2})$
$(-1,0,0)$	$(0,1,0)$	$(-x_{i+1j}, -y_{i+1j}, -z_{i+1j})$	$\frac{Vol_{k-1}^1}{Vol_{k-1}}(1+\frac{x_{i+1j}^2}{z_{i+1j}^2})$
$(-1,0,0)$	$(0,-1,0)$	$(x_{i+1j}, y_{i+1j}, z_{i+1j})$	$\frac{Vol_k^0}{Vol_k}(1+\frac{x_{i+1j}^2}{z_{i+1j}^2})$
$(-1,0,0)$	$(0,-1,0)$	$(-x_{i+1j}, -y_{i+1j}, -z_{i+1j})$	$\frac{Vol_{k-1}^1}{Vol_{k-1}}(1+\frac{x_{i+1j}^2}{z_{i+1j}^2})$

where Vol_k^0 and Vol_k^1 are the volumes of the front and back half of cells with index k, respectively, and Vol_k is the volume of cells with index k. The rightmost column represents the volume weighted coefficient of the corresponding natural inner product. By combining the eight terms, we get the explicit formula for L_{11}.

$$L_{11}X_{ijk}^u = 2\left(\frac{Vol_{k-1}^1}{Vol_{k-1}} + \frac{Vol_k^0}{Vol_k}\right)\left(2 + \frac{x_{ij}^2}{z_{ij}^2} + \frac{x_{i+1j}^2}{z_{i+1j}^2}\right)X_{ijk}^u$$

We can derive the other terms in the matrix L in similar fashion.

$$L_{12}X_{ijk}^v = \left(\frac{Vol_{k-1}^1}{Vol_{k-1}} + \frac{Vol_k^0}{Vol_k}\right)\left(\frac{x_{ij}y_{ij}}{z_{ij}^2}\left(X_{ijk}^v + X_{ij-1k}^v\right) + \frac{x_{i+1j}y_{i+1j}}{z_{i+1j}^2}\left(X_{i+1jk}^v + X_{i+1j-1k}^v\right)\right)$$

$$L_{13}X_{ijk}^w = 2\left(\frac{-x_{ij}}{z_{ij}^2}\left(\frac{Vol_{k-1}^1}{Vol_{k-1}}X_{ijk-1}^w + \frac{Vol_k^0}{Vol_k}X_{ijk}^w\right) + \frac{-x_{i+1j}}{z_{i+1j}^2}\left(\frac{Vol_{k-1}^1}{Vol_{k-1}}X_{i+1jk-1}^w + \frac{Vol_k^0}{Vol_k}X_{i+1jk}^w\right)\right)$$

$$L_{21}X_{ijk}^u = \left(\frac{Vol_{k-1}^1}{Vol_{k-1}} + \frac{Vol_k^0}{Vol_k}\right)\left(\frac{x_{ij}y_{ij}}{z_{ij}^2}\left(X_{ijk}^u + X_{i-1jk}^u\right) + \frac{x_{ij+1}y_{ij+1}}{z_{ij+1}^2}\left(X_{ij+1k}^u + X_{i+1j+1k}^u\right)\right)$$

$$L_{22}X^v_{ijk} = 2\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(2 + \frac{y^2_{ij}}{z^2_{ij}} + \frac{y^2_{ij+1}}{z^2_{ij+1}}\right)X^v_{ijk}$$

$$L_{23}X^w_{ijk} = 2\left(\frac{-y_{ij}}{z^2_{ij}}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}}X^w_{ijk-1} + \frac{Vol^0_k}{Vol_k}X^w_{ijk}\right) + \frac{-y_{ij+1}}{z^2_{ij+1}}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}}X^w_{ij+1k-1} + \frac{Vol^0_k}{Vol_k}X^w_{ij+1k}\right)\right)$$

$$L_{31}X^u_{ijk} = 2\left(\frac{-x_{ij}}{z^2_{ij}}\right)\left(\frac{Vol^0_k}{Vol_k}\left(X^u_{ijk} + X^u_{i-1jk}\right) + \frac{Vol^1_k}{Vol_k}\left(X^u_{ijk+1} + X^u_{i-1jk+1}\right)\right)$$

$$L_{32}X^v_{ijk} = 2\left(\frac{-y_{ij}}{z^2_{ij}}\right)\left(\frac{Vol^0_k}{Vol_k}\left(X^v_{ijk} + X^v_{ij-1k}\right) + \frac{Vol^1_k}{Vol_k}\left(X^v_{ijk+1} + X^v_{ij-1k+1}\right)\right)$$

$$L_{33}X^w_{ijk} = \frac{1}{z^2_{ij}}X^w_{ijk}$$

To compute the divergence, the first step is to compute the temporary values of LV.

$$\begin{pmatrix} u^{temp} \\ v^{temp} \\ w^{temp} \end{pmatrix} = \begin{pmatrix} L_{11}u + L_{12}v + L_{13}w \\ L_{21}u + L_{22}v + L_{23}w \\ L_{31}u + L_{32}v + L_{33}w \end{pmatrix}$$

Then the transpose of gradient is applied and finally multiplied by the inverse of the nodes volume.

$$\nabla \cdot V = \frac{-1}{VNode_k}\left(\frac{u^{temp}_{i-1jk} - u^{temp}_{ijk}}{\Delta X_k} + \frac{v^{temp}_{ij-1k} - v^{temp}_{ijk}}{\Delta Y_k} + \frac{w^{temp}_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{w^{temp}_{ijk}}{\Delta Z_{ijk}}\right)$$

ΔX, ΔY and ΔZ are the grid spacing in X, Y and Z respectively. The grid spacing in the X and Y directions are constant within the same k index while the grid spacing in the Z direction varies.

3.4 Advection

Advection is performed using the semi-lagrangian method as before, but a few changes need to be made in order to accommodate the frustum aligned grid.

To advect the smoke density at the grid nodes, the first step is to convert the grid location into a physical location using the inverse of the perspective matrix. The next step is to define a velocity field at the grid nodes by averaging the velocities along the grid lines. Since the velocities are located in the midpoint between grid

nodes and the grid nodes are not equally spaced in the ζ direction, we need to do a weighted average instead.

$$\begin{pmatrix} U_{ijk} \\ V_{ijk} \\ W_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{u_{ijk}+u_{i-1jk}}{2} \\ \frac{v_{ijk}+v_{ij-1k}}{2} \\ \frac{w_{ijk}\Delta Z_{ijk-1}+w_{ijk-1}\Delta Z_{ijk}}{\Delta Z_{ijk}+\Delta Z_{ijk-1}} \end{pmatrix}$$

The node velocity is then converted to real world values and used to trace the smoke density backward. The calculated physical location is converted back into a grid location using the perspective matrix. Trilinear interpolation is then carried out using the grid location in grid space. Special treatment is needed for the interpolation in the ζ direction because the perspective transformation is not affine in that direction. The grid location is used to locate which cell the smoke is from. The physical location of the cell's corners and the calculated physical location of the smoke in the ζ direction are used to compute a weighting factor for the interpolation. The grid location is sufficient to compute the weighting factor for interpolation in the ξ and η directions. Higher degree interpolation schemes can be used to obtain more accurate results.

Unlike traditional methods where the velocities are advected individually, the averaged node velocity is advected instead. The process is identical to advecting smoke described above. The advected node velocity is then averaged back to the grid lines using the usual method.

$$\begin{pmatrix} u_{ijk} \\ v_{ijk} \\ w_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{U_{ijk}+U_{i+1jk}}{2} \\ \frac{V_{ijk}+V_{ij+1k}}{2} \\ \frac{W_{ijk}+W_{ijk+1}}{2} \end{pmatrix}$$

3.5 Pressure solve

To solve for the pressure, we need to solve the Poisson equation defined as

$$\nabla^2 P = \nabla \cdot V$$

The divergence is defined in section 3.3 and we still need to define the Laplacian of the pressure. We can rewrite the Laplacian as the divergence of the gradient and apply the definition of divergence from above. Since the divergence operator is not as compact as on a regular grid, the Laplacian operator involves 18 of the 26 neighboring grid nodes.

$$\nabla^2 P = \nabla \cdot \nabla P = -N^{-1} \nabla^T L \nabla P$$

Expanding the equation we get

$$-N^{-1}\nabla^T \begin{pmatrix} L_{11} & L_{12} & L_{13} \\ L_{21} & L_{22} & L_{23} \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} \nabla P_x \\ \nabla P_y \\ \nabla P_z \end{pmatrix}$$

$$\begin{aligned} & -N^{-1}\Big(\frac{(L_{11}\nabla P_x + L_{12}\nabla P_y + L_{13}\nabla P_z)_{i-1jk}}{\Delta X_k} - \frac{(L_{11}\nabla P_x + L_{12}\nabla P_y + L_{13}\nabla P_z)_{ijk}}{\Delta X_k} + \\ = \; & \frac{(L_{21}\nabla P_x + L_{22}\nabla P_y + L_{23}\nabla P_z)_{ij-1k}}{\Delta Y_k} - \frac{(L_{21}\nabla P_x + L_{22}\nabla P_y + L_{23}\nabla P_z)_{ijk}}{\Delta Y_k} + \\ & \frac{(L_{31}\nabla P_x + L_{32}\nabla P_y + L_{33}\nabla P_z)_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{(L_{31}\nabla P_x + L_{32}\nabla P_y + L_{33}\nabla P_z)_{ijk}}{\Delta Z_{ijk}}\Big) \end{aligned}$$

$$\begin{aligned} & -N^{-1}\Big(\frac{(L_{11}\nabla P_x)_{i-1jk} - (L_{11}\nabla P_x)_{ijk}}{\Delta X_k} + \frac{(L_{12}\nabla P_y)_{i-1jk} - (L_{12}\nabla P_y)_{ijk}}{\Delta X_k} + \frac{(L_{13}\nabla P_z)_{i-1jk} - (L_{13}\nabla P_z)_{ijk}}{\Delta X_k} + \\ = \; & \frac{(L_{21}\nabla P_x)_{ij-1k} - (L_{21}\nabla P_x)_{ijk}}{\Delta Y_k} + \frac{(L_{22}\nabla P_y)_{ij-1k} - (L_{22}\nabla P_y)_{ijk}}{\Delta Y_k} + \frac{(L_{23}\nabla P_z)_{ij-1k} - (L_{23}\nabla P_z)_{ijk}}{\Delta Y_k} + \\ & \frac{(L_{31}\nabla P_x)_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{(L_{31}\nabla P_x)_{ijk}}{\Delta Z_{ijk}} + \frac{(L_{32}\nabla P_y)_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{(L_{32}\nabla P_y)_{ijk}}{\Delta Z_{ijk}} + \frac{(L_{33}\nabla P_z)_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{(L_{33}\nabla P_z)_{ijk}}{\Delta Z_{ijk}}\Big) \end{aligned}$$

Further expanding the above equation by replacing the pressure gradient with

$$\nabla P_{ijk} = \begin{pmatrix} \nabla P^x_{ijk} \\ \nabla P^y_{ijk} \\ \nabla P^z_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{P_{i+1jk} - P_{ijk}}{\Delta X_k} \\ \frac{P_{ij+1k} - P_{ijk}}{\Delta Y_k} \\ \frac{P_{ijk+1} - P_{ijk}}{\Delta Z_{ijk}} \end{pmatrix}$$

which defines the gradient of velocities along the grid lines at the node, we can derive the coefficient of the grid node and its 18 neighbors. The derivation of the coefficients is included in the appendix.

The pressure can then be solved using the standard Gauss-Seidel relaxation or conjugate gradient.

3.6 Vorticity

To increase the realism of the simulation, a force similar to voriticity confinement is used. The idea is to maintain the level of curl within the system, thus maintaining the small scale details that would have otherwise be damped out. The first step is to calculate the curl for each face of the cells.

$$\nabla \times V_{ijk} = \begin{pmatrix} curl^u_{ijk} \\ curl^v_{ijk} \\ curl^w_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{v_{ijk+1}\Delta Y_{k+1} - w_{ij+1k}\Delta Z_{ij+1k} - v_{ijk}\Delta Y_k + w_{ijk}\Delta Z_{ijk}}{Area^x_{ik}} \\ \frac{w_{i+1jk}\Delta Z_{i+1jk} - u_{ijk+1}\Delta X_{k+1} - w_{ijk}\Delta Z_{ijk} + u_{ijk}\Delta X_k}{Area^y_{jk}} \\ \frac{v_{i+1jk}\Delta Y_k - u_{ij+1k}\Delta X_k - v_{ijk}\Delta Y_k + u_{ijk}\Delta X_K}{Area^z_k} \end{pmatrix}$$

Then if the magnitude of the curl of any face is less then a predefined minimum, a force is applied to increase the curl of the face. The curl is always perpendicular to the face and the faces along the same grid lines lay on the same plane. The two dimensional confinement force can be calculated for each plane using the normalized

gradient of curl as a weighting term.

$$\nabla \times V^u_{ijk}:$$

$$weight_{ijk} = \begin{pmatrix} weight^y_{ijk} \\ \\ weight^z_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{curl^u_{ij+1k} - curl^u_{ij-1k}}{\Delta Y_k + \Delta Y_{k+1}} \\ \frac{curl^u_{ijk+1} - curl^u_{ijk-1}}{\Delta Z_{ijk} + \Delta Z_{ij+1k}} \end{pmatrix}$$

$$weight_{ijk} = \frac{weight_{ijk}}{|weight_{ijk}|}$$

$$f^v_{ijk+1} = -\epsilon(curl^u_{ijk})(weight^z_{ijk})\Delta Y_{k+1}/2$$

$$f^w_{ij+1k} = \epsilon(curl^u_{ijk})(weight^y_{ijk})\Delta Z_{ij+1k}/2$$

$$f^v ijk = -\epsilon(curl^u_{ijk})(weight^z_{ijk})\Delta Y_k/2$$

$$f^w_{ijk} = \epsilon(curl^u_{ijk})(weight^y_{ijk})\Delta Z_{ijk}/2$$

$$\nabla \times V^v_{ijk}:$$

$$weight_{ijk} = \begin{pmatrix} weight^x_{ijk} \\ \\ weight^z_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{curl^v_{i+1jk} - curl^v_{i-1jk}}{\Delta X_k + \Delta X_{k+1}} \\ \frac{curl^v_{ijk+1} - curl^v_{ijk-1}}{\Delta Z_{ijk} + \Delta Z_{i+1jk}} \end{pmatrix}$$

$$weight_{ijk} = \frac{weight_{ijk}}{|weight_{ijk}|}$$

$$f^w_{i+1jk} = -\epsilon(curl^v_{ijk})(weight^x_{ijk})\Delta Z_{i+1jk}/2$$

$$f^u_{ijk+1} = \epsilon(curl^v_{ijk})(weight^z_{ijk})\Delta X_{k+1}/2$$

$$f^w ijk = -\epsilon(curl^v_{ijk})(weight^x_{ijk})\Delta Z_{ijk}/2$$

$$f^u_{ijk} = \epsilon(curl^v_{ijk})(weight^z_{ijk})\Delta X_k/2$$

$\nabla \times V^u_{ijk}$:

$$weight_{ijk} = \begin{pmatrix} weight^x_{ijk} \\ weight^y_{ijk} \end{pmatrix} = \begin{pmatrix} \frac{curl^w_{i+1jk} - curl^w_{i-1jk}}{\Delta X_k} \\ \frac{curl^w_{ij+1k} - curl^w_{ij-1k}}{\Delta Y_k} \end{pmatrix}$$

$$weight_{ijk} = \frac{weight_{ijk}}{|weight_{ijk}|}$$

$$f^v_{i+1jk} = \epsilon(curl^w_{ijk})(weight^x_{ijk})\Delta Y_k/2$$

$$f^u_{ij+1k} = -\epsilon(curl^w_{ijk})(weight^y_{ijk})\Delta X_k/2$$

$$f^v ijk = \epsilon(curl^w_{ijk})(weight^x_{ijk})\Delta Y_k/2$$

$$f^u_{ijk} = -\epsilon(curl^w_{ijk})(weight^y_{ijk})\Delta X_k/2$$

Both ϵ and the minimum curl can be used to control the amount of force applied and the activities within the smoke. They have similar effect as the ϵ used in vorticity confinement where values too large can have negative overall effects.

Chapter 4

Experimental results

The experimental simulations are implemented using Visual C++ and OpenGL. They are run on a laptop with 1.5Ghz Pentium M processor and 512MB of memory.

The simulation requires a few parameters including field of view, aspect ratio, near plane, far plane, number of grid lines in the x, y, and z direction, number of iterations for the Gauss-Seidel pressure solver, number of seconds to simulate, buoyancy, ϵ for vorticity, τ for opacity and minimum curl. Typical values used in experimental simulations are summarized in table 4.1. Pressure at boundaries are set to zero and Neumann conditions are used for velocities.

Experiments are conducted to investigate the effects of various parameters have on the simulation, and a comparison with traditional simulation is also carried out. A hypothetical mushroom cloud where the smoke originate from a small opening in the ground in the center of the domain is used to perform the testing, but in practice, there can be more than one source and they can be located anywhere within the domain.

The images are rendered using a ray marching technique which simply accumulates the luminance along the ζ direction. Two light sources are used to illuminate

Parameter	Value
fov	15^o
aspect	1.0
near plane	5.0
far plane	10.0
XMAX	128
YMAX	128
ZMAX	64
iterations	10
seconds	20
buoyancy	5.0
ϵ	2.0
τ	0.1
MINCURL	5.0

Table 4.1: Parameters for simulation.

the smoke, one from overhead and one from the left hand side. More advanced techniques such as photon mapping can be used to create higher quality images. Figure 4.1 shows several sample images obtained from different styles of smoke. Some artifacts are visible due to aliasing from the grid.

4.1 Source location

Since the configuration of the frustum aligned grid places emphasis more on the space at the front of the frustum than the back, the location of the source should be an important factor in the appearance of the resultant smoke. Figure 4.2 compares the simulations of three different source locations. These simulations use the typical values found in the beginning of the chapter except the value of ϵ is set to zero. This effectively turns vorticity confinement off and allows the smoke to rise under buoyancy alone. The sizes of the source in all simulation are the same and they are in the shape of a square. The first sequence is generated with a source located

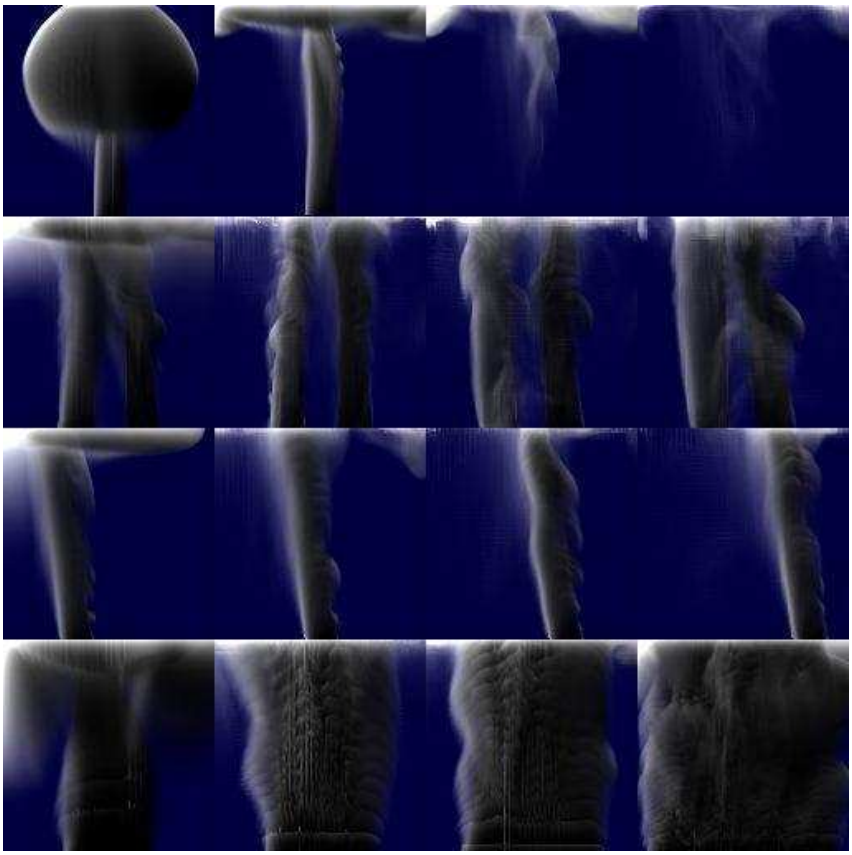

Figure 4.1: Images obtained from different smoke simulations. Top: a pulse, Second: two sources, Third: moving source, Bottom: approaching source.

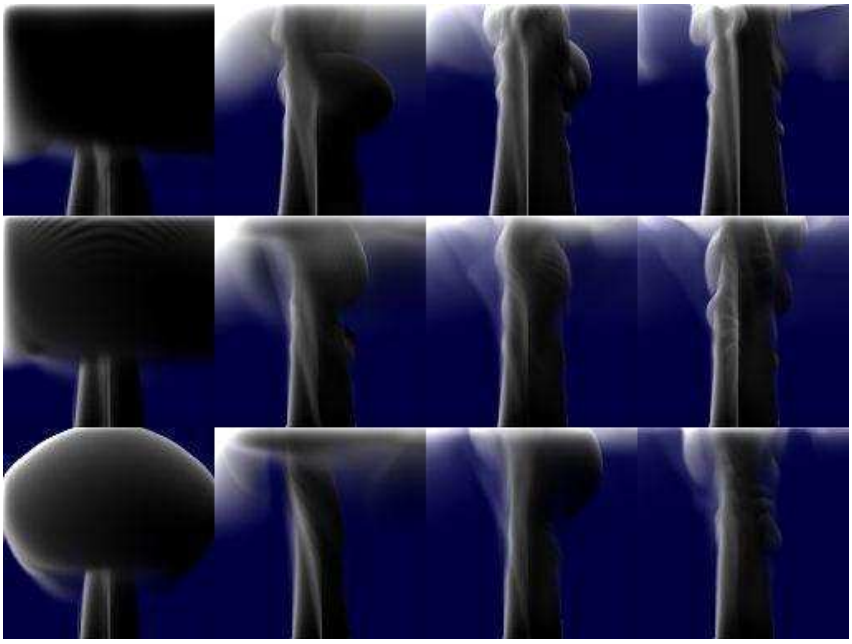

Figure 4.2: Simulations with different source locations. Top: back half of frustum, Middle: middle of frustum, Bottom: front half of frustum.

at the back half of the frustum, the second sequence is generated with a source located at the middle of the frustum and the last is generated with a source located at the front half of the frustum. Since the size of the source is identical, as it moves further away from the camera, it will appear smaller. The level of details will also be lowered as less grid nodes are located at the back of the frustum, but coupled with the reduction in the apparent size, the resulting image still maintains a comparable level of details.

4.2 Grid size

The grid size is a crucial factor in the smoke simulation. To increase the realism of a simulation, the trivial solution is to increase the resolution of the grid since the smaller the grid spacing, the better the ability to capture the details. The drawback

Grid Size	Run Time(s/frame)	File Size(kb)
32 by 32 by 32	0.461	517
64 by 64 by 32	1.342	2057
64 by 64 by 64	2.874	4113
128 by 128 by 32	5.448	8209
128 by 128 by 128	20.920	32833
256 by 256 by 32	41.309	32801

Table 4.2: Run time of sample simulations with different grid sizes in seconds per frame and file size of one frame of smoke data in kilobytes.

of such approach includes the increase in simulation time and memory requirement. Typically, the simulation time varies directly with the grid size; doubling the grid dimension in all directions transforms to eight times the original simulation time. Table 4.2 lists the simulation time for several sample grid sizes as well as the file size for one frame of smoke data.

The frustum aligned grid method is aimed to increase the realism while addressing the time and memory issue. Each grid lines in the direction perpendicular to the camera is projected into a single point at the front of the camera. The dimension parallel to the camera can be viewed as the resolution of the image where each pixel is a projected grid line. Using this arrangement, all grid nodes are located at a relevant position with respect to the camera and no grid nodes are wasted or overlapping. In traditional rectangular computational grids, when rendering the resulting image, usually many different grid nodes will be mapped to the same pixel, thus wasting the computational time and memory space for these redundant values. The grid spacing in a frustum aligned grid is not uniform. The spacing at the back of the frustum can be couple times larger than the ones at the front. Yet, the details are not lost from the sparse spacing because the frustum aligned grid accounted for what could and couldn't be seen by the camera. To cover the same

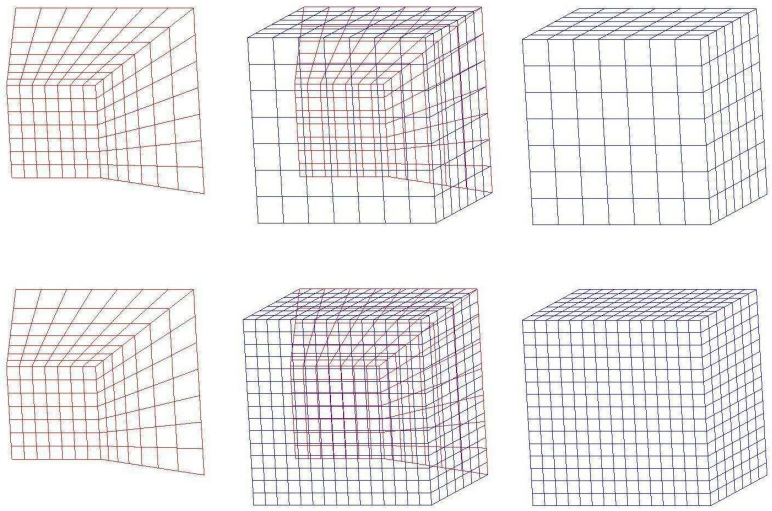

Figure 4.3: Comparison between frustum aligned grid and rectangular grid. Top: grid spacing match back of frustum, Bottom: grid spacing match front of frustum.

volume occupied by the frustum using a traditional grid, considerably more grid nodes are needed since in order to obtain the same level of details, the grid spacing of the whole domain should match the grid spacing at the front and extra grid nodes are needed outside the frustum to get a rectangular grid. See figure 4.3. To cover the 64 by 64 by 64 grid of the frustum described by the parameters at the beginning of the chapter, a 128 by 128 by 128 uniform grid will be needed. The amount of extra grid nodes required is mainly depended upon the near and far clipping planes of the frustum.

4.3 Vorticity

The realism of the smoke simulation is enhanced by the introduction of vorticity force. The force is controlled using two parameters: ϵ and MINCURL. The MINCURL controls the minimum curl desirable. Any face with a curl smaller than

ϵ	MINCURL
1	5
2	5
3	5
4	5
1	10
2	10

Table 4.3: ϵ and MINCURL used in figure 4.4

MINCURL will have a force applied to it and that force is proportional to the curl itself. ϵ controls the amount of force that will be applied. Figure 4.4 shows several simulations with different setting of ϵ and MINCURL and table 4.3 lists the values used in the simulations. The typical values at the beginning of the chapter are used for the other parameters.

The ideal value of ϵ and MINCURL depends upon the scene and varies greatly. Trial and error may be required to determine the ideal value but in most case, the ideal value is not required to produce a realistic and interesting simulation.

4.4 Comparison with traditional simulation

A similar implementation to the frustum aligned grid method of the traditional grid method is used for the comparison. Table 4.4 compares the run time and memory usage of the two methods for several different grid sizes and table 4.5 compares the simulation time used in the subroutines of the two methods.

As seen from the data that the frustum aligned grid method is on average half as fast as the traditional method. The extra computation time is used mainly in the divergence and pressure solver as expected. To bring the frustum aligned method on par with the traditional method in term of computation time, we can

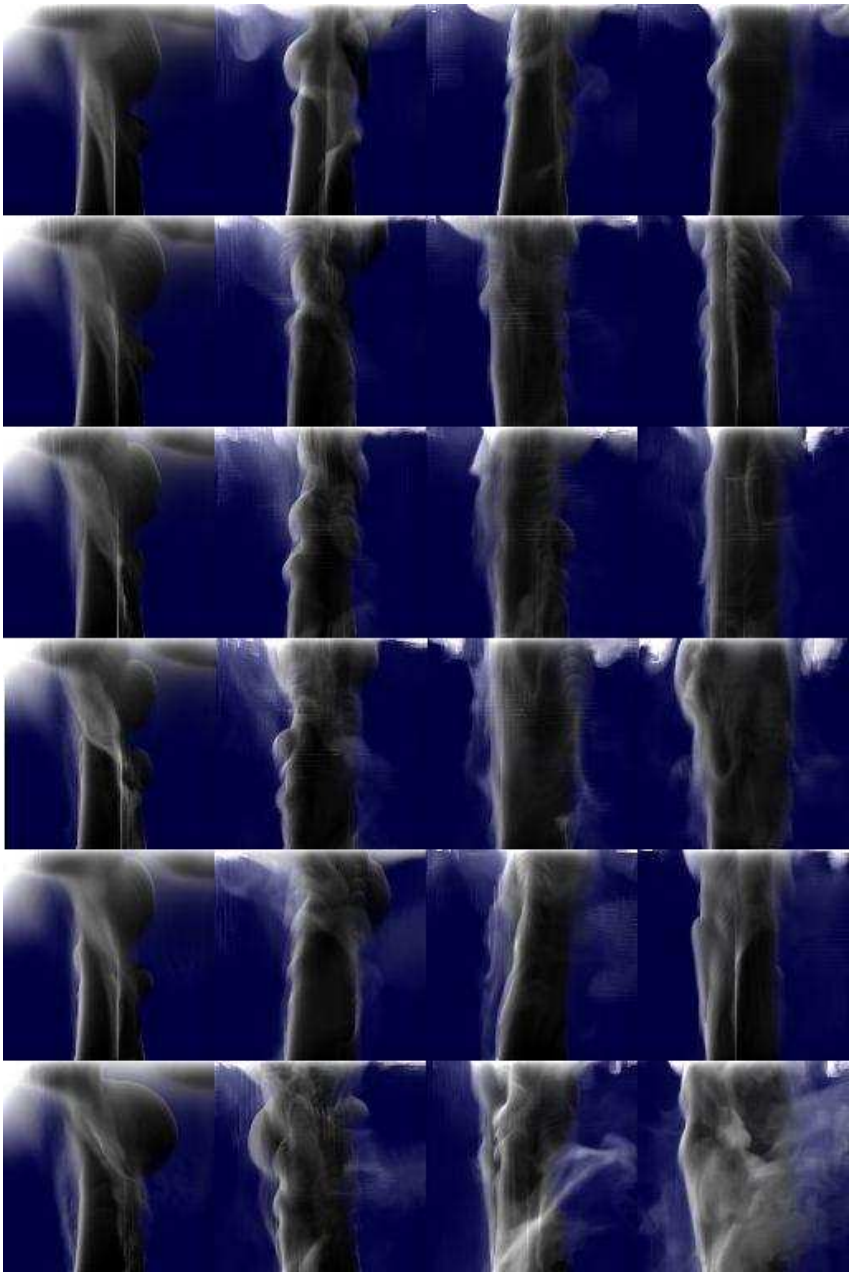

Figure 4.4: Simulations with different ϵ and MINCURL.

Grid Size	Frustum Run Time	Memory	Traditional Run Time	Memory
32 by 32 by 32	0.461	4124	0.260	3972
64 by 64 by 32	1.342	13812	0.982	12132
64 by 64 by 64	2.874	26500	1.903	23456
128 by 128 by 32	5.448	52104	3.876	46120

Table 4.4: Run time and memory usage comparison of frustum aligned grid method and traditional method.

Subroutine	Frustum	Traditional
Initalization	0.160	0.010
Advection	0.280	1.472
Vorticity	0.170	0.160
Divergence	0.160	0.010
Pressure Solve	1.502	0.100
Gradient	0.010	0.010

Table 4.5: Comparison of simulation time used in subroutines between frustum aligned grid method and traditional method using a 64 by 64 by 64 grid.

reduce the number of grid lines in the direction perpendicular to the camera by a factor of two. Since the grid lines are concentrated at the front of the frustum and they are not equally spaced, decreasing the number of grid lines will have a smaller effect on the sampling at the front of the frustum where details are most important. Table 4.6 shows the grid spacing of the first few grid lines of varies grid sizes and figure 4.5 compares the resulting images from the different simulations.

The result indicates that the reduction of grid lines produces comparable

Grid Size	δz_1	δz_2	δz_3	δz_4	δz_5	δz_{last}
128 by 128 by 32	0.0820	0.0847	0.0875	0.0906	0.0938	0.3125
128 by 128 by 64	0.0400	0.0406	0.0414	0.0419	0.0427	0.1563
128 by 128 by 128	0.0198	0.0199	0.0201	0.0202	0.0204	0.0781

Table 4.6: Grid spacing of varies grid sizes.

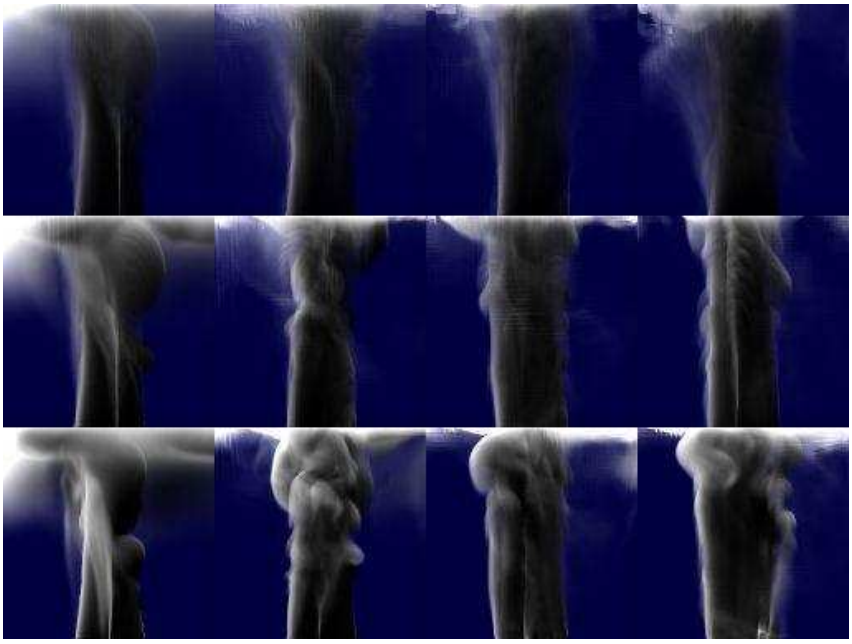

Figure 4.5: Comparison of simulations with different number of grid lines in the ζ direction.

Figure	Method	Grid Spacing	Grid Size	Simulation Time
a	Traditional	0.0419	128 by 128 by 128	14.221
b	Frustum	0.0425	64 by 64 by 64	2.874
c	Frustum	0.0211	128 by 128 by 64	9.684
d	Frustum	0.0211	128 by 128 by 128	20.920

Table 4.7: Statistics for figure 4.6.

images that suffer slight detail lost and it also leads to lower memory requirement.

Figure 4.6 compares the images obtained from traditional methods and frustum aligned grid method. Three different criteria are used for comparison including equal grid spacing, equal grid size and equal computation time. Table 4.7 lists the statistics for the simulations.

With equal grid spacing at the front of the frustum, the traditional simulation produces better result than frustum aligned method but the simulation time and

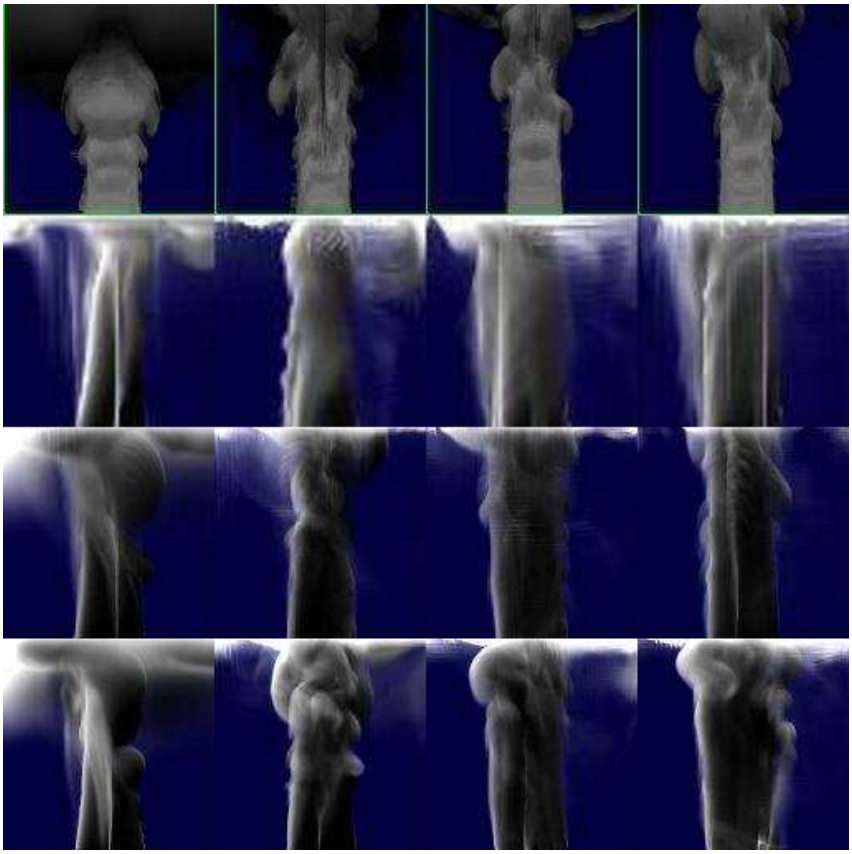

Figure 4.6: Comparison of traditional method and frustum aligned grid method.

memory usage are both five times more. Increasing the grid size by a factor of two in the x and y direction produces comparable result. The memory usage is half of the traditional's and the simulation time is about 30% less. With equal grid size, the frustum aligned method produces superior simulation than traditional method at a cost of increased simulation time. These results confirm the performance of the frustum aligned grid method in terms of memory usage and level of details.

Chapter 5

Conclusion

We presented a frustum aligned grid for realistic smoke simulation that benefits from knowledge of the viewing camera. The novel computation grid addressed the problem of memory usage and level of details by utilizing non uniform frustum aligned grid cells. Significantly less grid nodes are required to cover the computation domain, thus reducing the memory usage and computation time. The grid lines in the direction perpendicular to the camera can be reduced without suffering detail lost. The reduction further reduces the memory usage and they can translate to increased grid lines in other directions, thus increasing the level of details. Comparison with traditional simulation proved the validity and effectiveness of the method. The currently fixed frustum is a limitation and a topic for future work. The current configuration can realistically simulate a large scale phenomenon at a fixed viewpoint but a fly-by scenario with a moving camera will require additional modification, possibly in the form of particles.

Bibliography

[1] R. Fedkiw, J. Stam, and H. W. Jensen. Visual simulation of smoke. In *Proc. of SIGGRAPH 2001*, pages 15–22, 2001.

[2] B. E. Feldman, J. F. OBrien, and B. M. Klingner. Animating gases with hybrid meshes. In *Proc. of SIGGRAPH 2005*, 2005.

[3] F. Harlow and J. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface. In *The Physics of Fluids 8*, pages 2182–2189, 1965.

[4] J. M. Hyman and M. Shashkov. The orthogonal decomposition theorems for mimetic finite difference methods. *SIAM Journal on Numberical Analysis*, 36(3):788–818, 1999.

[5] F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. In *Proc. of SIGGRAPH 2004*, pages ??–??, 2004.

[6] N. Rasmussen, D. Nguyen, W. Geiger, and R. Fedkiw. Smoke simulation for large scale phenomena. In *Proc. of SIGGRAPH 2003*, pages 703–707, 2003.

[7] A. Selle, N. Rasmussen, and R. Fedkiw. A vortex particle method for smoke, water and explosions. In *Proc. of SIGGRAPH 2005*, pages 910–914, 2005.

[8] J. Stam. Stable fluid. In *Proc. of SIGGRAPH 1999*, pages 121–128, 1999.

Appendix A

Laplacian Coefficients

$$
\begin{aligned}
&\frac{L_{11}\nabla P^x_{i-1jk} - L_{11}\nabla P^x_{ijk}}{\Delta X_k} \\
&= (2(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(2 + \frac{x^2_{ij}}{z^2_{ij}} + \frac{x^2_{i-1j}}{z^2_{i-1j}})(\frac{P_{ijk} - P_{i-1jk}}{\Delta X_K}) - \\
&\quad 2(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(2 + \frac{x^2_{ij}}{z^2_{ij}} + \frac{x^2_{i+1j}}{z^2_{i+1j}})(\frac{P_{i+1jk} - P_{ijk}}{\Delta X_K}))/\Delta X_k \\
&= -\frac{2}{\Delta X^2_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(2 + \frac{x^2_{ij}}{z^2_{ij}} + \frac{x^2_{i-1j}}{z^2_{i-1j}})P_{i-1jk} \\
&\quad -\frac{2}{\Delta X^2_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(2 + \frac{x^2_{ij}}{z^2_{ij}} + \frac{x^2_{i+1j}}{z^2_{i+1j}})P_{i+1jk} \\
&\quad +\frac{2}{\Delta X^2_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(4 + \frac{2x^2_{ij}}{z^2_{ij}} + \frac{x^2_{i-1j}}{z^2_{i-1j}} + \frac{x^2_{i+1j}}{z^2_{i+1j}})P_{ijk} \\
&\frac{L_{12}\nabla P^y_{i-1jk} - L_{12}\nabla P^y_{ijk}}{\Delta X_k} \\
&= \frac{1}{\Delta X_k \Delta Y_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(\frac{x_{i-1j}y_{i-1j}}{z^2_{i-1j}})P_{i-1j+1k} \\
&\quad -\frac{1}{\Delta X_k \Delta Y_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(\frac{x_{i-1j}y_{i-1j}}{z^2_{i-1j}})P_{i-1j-1k} \\
&\quad -\frac{1}{\Delta X_k \Delta Y_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(\frac{x_{i+1j}y_{i+1j}}{z^2_{i+1j}})P_{i+1j+1k} \\
&\quad +\frac{1}{\Delta X_k \Delta Y_k}(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k})(\frac{x_{i+1j}y_{i+1j}}{z^2_{i+1j}})P_{i+1j-1k}
\end{aligned}
$$

$$\frac{L_{13}\nabla P^z_{i-1jk} - L_{13}\nabla P^z_{ijk}}{\Delta X_k}$$

$$\begin{aligned}
= \quad & \frac{2}{\Delta X_k}\left(\frac{-x_{i-1j}}{z^2_{i-1j}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k \Delta Z_{ijk}}\right)P_{i-1jk} \\
& - \frac{2}{\Delta X_k \Delta Z_{ijk-1}}\left(\frac{-x_{i-1j}}{z^2_{i-1j}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}}\right)P_{i-1jk-1} \\
& + \frac{2}{\Delta X_k \Delta Z_{ijk}}\left(\frac{-x_{i-1j}}{z^2_{i-1j}}\right)\left(\frac{Vol^0_k}{Vol_k}\right)P_{i-1jk+1} \\
& - \frac{2}{\Delta X_k}\left(\frac{-x_{i+1j}}{z^2_{i+1j}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k \Delta Z_{ijk}}\right)P_{i+1jk} \\
& + \frac{2}{\Delta X_k \Delta Z_{ijk-1}}\left(\frac{-x_{i+1j}}{z^2_{i+1j}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}}\right)P_{i+1jk-1} \\
& - \frac{2}{\Delta X_k \Delta Z_{ijk}}\left(\frac{-x_{i+1j}}{z^2_{i+1j}}\right)\left(\frac{Vol^0_k}{Vol_k}\right)P_{i+1jk+1}
\end{aligned}$$

$$\frac{L_{21}\nabla P^x_{ij-1k} - L_{21}\nabla P^x_{ijk}}{\Delta Y_k}$$

$$\begin{aligned}
= \quad & \frac{1}{\Delta X_k \Delta Y_k}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(\frac{x_{ij-1}y_{ij-1}}{z^2_{ij-1}}\right)P_{i+1j-1k} \\
& - \frac{1}{\Delta X_k \Delta Y_k}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(\frac{x_{ij-1}y_{ij-1}}{z^2_{ij-1}}\right)P_{i-1j-1k} \\
& - \frac{1}{\Delta X_k \Delta Y_k}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(\frac{x_{ij+1}y_{ij+1}}{z^2_{ij+1}}\right)P_{i+1j+1k} \\
& + \frac{1}{\Delta X_k \Delta Y_k}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(\frac{x_{ij+1}y_{ij+1}}{z^2_{ij+1}}\right)P_{i-1j+1k}
\end{aligned}$$

$$\frac{L_{22}\nabla P^y_{ij-1k} - L_{22}\nabla P^y_{ijk}}{\Delta Y_k}$$

$$\begin{aligned}
= \quad & -\frac{2}{\Delta Y_k^2}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(2 + \frac{y^2_{ij}}{z^2_{ij}} + \frac{y^2_{ij-1}}{z^2_{ij-1}}\right)P_{ij-1k} \\
& - \frac{2}{\Delta Y_k^2}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(2 + \frac{y^2_{ij}}{z^2_{ij}} + \frac{x^2_{ij+1}}{z^2_{ij+1}}\right)P_{ij+1k} \\
& + \frac{2}{\Delta Y_k^2}\left(\frac{Vol^1_{k-1}}{Vol_{k-1}} + \frac{Vol^0_k}{Vol_k}\right)\left(4 + \frac{2y^2_{ij}}{z^2_{ij}} + \frac{y^2_{ij-1}}{z^2_{ij-1}} + \frac{y^2_{ij+1}}{z^2_{ij+1}}\right)P_{ijk}
\end{aligned}$$

$$\frac{L_{23}\nabla P^z_{ij-1k} - L_{23}\nabla P^z_{ijk}}{\Delta Y_k}$$

$$\begin{aligned}
= \quad & \frac{2}{\Delta Y_k}\left(\frac{-y_{ij-1}}{z^2_{ij-1}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k \Delta Z_{ijk}}\right)P_{ij-1k} \\
& - \frac{2}{\Delta Y_k \Delta Z_{ijk-1}}\left(\frac{-y_{ij-1}}{z^2_{ij-1}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}}\right)P_{ij-1k-1} \\
& + \frac{2}{\Delta Y_k \Delta Z_{ijk}}\left(\frac{-y_{ij-1}}{z^2_{ij-1}}\right)\left(\frac{Vol^0_k}{Vol_k}\right)P_{ij-1k+1} \\
& - \frac{2}{\Delta Y_k}\left(\frac{-y_{ij+1}}{z^2_{ij+1}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k \Delta Z_{ijk}}\right)P_{ij+1k} \\
& + \frac{2}{\Delta Y_k \Delta Z_{ijk-1}}\left(\frac{-y_{ij+1}}{z^2_{ij+1}}\right)\left(\frac{Vol^1_{k-1}}{Vol_{k-1}}\right)P_{ij+1k-1} \\
& - \frac{2}{\Delta Y_k \Delta Z_{ijk}}\left(\frac{-y_{ij+1}}{z^2_{ij+1}}\right)\left(\frac{Vol^0_k}{Vol_k}\right)P_{ij+1k+1}
\end{aligned}$$

$$
\begin{aligned}
&\frac{L_{31}\nabla P^x_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{L_{31}\nabla P^x_{ijk}}{\Delta Z_{ijk}} \\
&= -\frac{2}{\Delta X_k}(\frac{-x_{ij}}{z^2_{ij}})(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k\Delta Z_{ijk}})P_{i-1jk} \\
&\quad -\frac{2}{\Delta X_{k-1}\Delta Z_{ijk-1}}(\frac{-x_{ij}}{z^2_{ij}})(\frac{Vol^0_{k-1}}{Vol_{k-1}})P_{i-1jk-1} \\
&\quad +\frac{2}{\Delta X_{k+1}\Delta Z_{ijk}}(\frac{-x_{ij}}{z^2_{ij}})(\frac{Vol^1_k}{Vol_k})P_{i-1jk+1} \\
&\quad +\frac{2}{\Delta X_k}(\frac{-x_{ij}}{z^2_{ij}})(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k\Delta Z_{ijk}})P_{i+1jk} \\
&\quad +\frac{2}{\Delta X_{k-1}\Delta Z_{ijk-1}}(\frac{-x_{ij}}{z^2_{ij}})(\frac{Vol^0_{k-1}}{Vol_{k-1}})P_{i+1jk-1} \\
&\quad -\frac{2}{\Delta X_{k+1}\Delta Z_{ijk}}(\frac{-x_{ij}}{z^2_{ij}})(\frac{Vol^1_k}{Vol_k})P_{i+1jk+1}
\end{aligned}
$$

$$
\begin{aligned}
&\frac{L_{32}\nabla P^y_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{L_{32}\nabla P^y_{ijk}}{\Delta Z_{ijk}} \\
&= -\frac{2}{\Delta Y_k}(\frac{-y_{ij}}{z^2_{ij}})(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k\Delta Z_{ijk}})P_{ij-1k} \\
&\quad -\frac{2}{\Delta Y_{k-1}\Delta Z_{ijk-1}}(\frac{-y_{ij}}{z^2_{ij}})(\frac{Vol^0_{k-1}}{Vol_{k-1}})P_{ij-1k-1} \\
&\quad +\frac{2}{\Delta Y_{k+1}\Delta Z_{ijk}}(\frac{-y_{ij}}{z^2_{ij}})(\frac{Vol^1_k}{Vol_k})P_{ij-1k+1} \\
&\quad +\frac{2}{\Delta Y_k}(\frac{-y_{ij}}{z^2_{ij}})(\frac{Vol^1_{k-1}}{Vol_{k-1}\Delta Z_{ijk-1}} - \frac{Vol^0_k}{Vol_k\Delta Z_{ijk}})P_{ij+1k} \\
&\quad +\frac{2}{\Delta Y_{k-1}\Delta Z_{ijk-1}}(\frac{-y_{ij}}{z^2_{ij}})(\frac{Vol^0_{k-1}}{Vol_{k-1}})P_{ij+1k-1} \\
&\quad -\frac{2}{\Delta Y_{k+1}\Delta Z_{ijk}}(\frac{-y_{ij}}{z^2_{ij}})(\frac{Vol^1_k}{Vol_k})P_{ij+1k+1}
\end{aligned}
$$

$$
\begin{aligned}
&\frac{L_{33}\nabla P^z_{ijk-1}}{\Delta Z_{ijk-1}} - \frac{L_{33}\nabla P^z_{ijk}}{\Delta Z_{ijk}} \\
&= -\frac{1}{\Delta Z^2_{ijk-1}z^2_{ij}}P_{ijk-1} \\
&\quad -\frac{1}{\Delta Z^2_{ijk}z^2_{ij}}P_{ijk+1} \\
&\quad +\frac{1}{z^2_{ij}}(\frac{1}{\Delta Z^2_{ijk-1}} + \frac{1}{\Delta Z^2_{ijk}})P_{ijk}
\end{aligned}
$$