Application of Two-Dimensional Cellular Automaton Lattice-Gas Models to the Simulation of Hydrodynamics

Brian J N Wylie

Doctor of Philosophy
1990

Abstract

HYDRODYNAMIC equations are notoriously difficult to solve, both analytically and computationally, therefore less complicated methods, drawing on the power of cellular automata and lattice-gases are introduced. Their combined ability to capture the fundamental properties of fluid dynamics in an inherently simple manner is discussed. The *FHP7* cellular automaton lattice-gas model of Frisch, Hasslacher and Pomeau, which will form the basis for the subsequent simulations, is described in detail, with a more general covering of the associated models.

THE SCALABLE and flexible computational power of the transputer-based ECS multicomputer, and how this may be applied to the lattice-gas simulations at hand is addressed. The distributed multiprocessor architecture provides unique challenges, such that the implementation might achieve its potential. It is found that a straightforward one-dimensional geometric decomposition of the lattice, in conjunction with the loosely-synchronous nature of the distributed update, provides a natural load-balancing, and highly scalable efficiency.

VISUALISATION of the development of the hydrodynamic features captured by the simulations, such that their content may be clearly extracted is also addressed. Many interesting transient and dynamic features, often occuring on time-scales which make their analysis by other methods difficult, are easily identified. Those occasionally found to be the result of artifacts, perhaps in the initialisation of the simulation are quickly identified, such that the simulations may be refined.

ELEMENTARY static systems and flows are designed, such that the ability of the *FHP7* lattice-gas to model incompressible hydrodynamics, and its multicomputer implementation, are verified against the theoretically and experimentally expected behaviour. Subsequently, more complex flow configurations involving obstructions and jets, generally beyond the limits of current analytic techniques are constructed, and found to qualitatively match experimental visualisations.

NO LATTICE-GASES are currently known to accurately model compressible fluid dynamics completely, and the ultimate cause of this limitation still requires clarification. The behaviour of the *FHP7* lattice-gas, in régimes where compressibility effects are expected to be important, is investigated with the aim of identifying those aspects of its microdynamics which cause breakdown of its macrodynamics.

Declaration

The material of this thesis (except where otherwise stated), and its composition, was done entirely by myself.

Brian J N Wylie

Acknowledgements

Use of any trademark is not intended in any way to infringe on the rights of the trademark holder.

Can it be there's some sort of error,
Hard to stop the surmounting terror,
Is it really the end, not some crazy dream ?

— [IM82]

For
wee folk, a'where
&
an t-Alba mo ghràdh

"Now's the day, and now's the hour . . . "
". . . Europe sans frontières"

Contents

List of Figures

List of Tables

0 Prologue

All of my dreams have fell through
  and their taste is so sour
Take second place in my mind
  for this one sacred hour
Still, I've been moved for so long
  by this strange fascination
Here, as I stand, all alone
  in complete concentration

— *[Magnum82]*

I NDUSTRIES as diverse as aerospace and automobile engineering, oil-extraction platform and racing yacht construction, all require detailed knowledge about how their product will fare in a range of conditions and environments, before they actually tool-up and put it into production. Whether it's stability and strength or an aerodynamically efficient profile that's desired, it's important to get it right the first time, since the expense involved in getting it wrong is prohibitive. [Settles89, Cogotti89, Frank89]

Two-dimensional simulations of hydrodynamics are known to be a viable means of providing the necessary analysis, in appropriate time-scales, of the complex three-dimensional processes which govern the design and construction of numerous industrial products, from cars to space shuttles. Many theoretical and experimental techniques are currently employed, but only computational methods provide the promise of supplying the desired flexibility and control to study extreme conditions. Computational fluid dynamics (CFD) for engineering, and additionally the weather forecasting industry, is currently one of the most insatiable uses of available computing resources.

Lattice-gases and cellular automata are two newly established disciplines, each investigated in isolation of the other for decades. Recently however, their combined power, with that of parallel digital computers, has made them formidable tools in probing the field of fluid dynamics. These extremely simple, fictitious models are described in **Chapter 1**. Particular emphasis is on the *FHP7* model, a variant of the original two-dimensional lattice-gas of Frisch, Hasslacher and Pomeau, which they demonstrated modelled the two-dimensional Navier-Stokes equation.

This model forms the basis of the simulations in **Chapter 3**, in which, first of all, a number of elementary static systems and flows are undertaken to validate the model and its implementation. A number of phenomenological flows follow, with configurations involving obstructions and jets, generally beyond current analytic techniques. The lattice-gas is shown to capture, in a fundamental way, a wide range of complex hydrodynamic features, previously confined to experimental methods of investigation.

Analysis of the two-dimensional lattice-gases, including the *FHP7* model, has shown that they only accurately represent valid hydrodynamics in the incompressible limit. However, a number of compressible features are evident in lattice-gas simulations. The reasons for this limitation are not currently understood and require clarification. To this end, **Chapter 4** describes simulations performed with the *FHP7* lattice-gas in régimes where compressibility effects are expected to be crucial, with a view to identifying those aspects of the model which may be anomalous.

In addition, the novel multicomputer architecture of the transputer-based Edinburgh Concurrent Supercomputer is considered in **Chapter 2**. This is shown to provide scalable and flexible computational power, which can be straightforwardly applied to the lattice-gas simulations, with extremely high efficiency. Real-time, high-resolution graphics facilities are also provided, and these are exploited to dramatically visualise the developing flows. Since hydrodynamic features exist on a variety of time-scales, and some do not equilibrate to steady-state solutions, flow visualisation allows the attributes of interest to be quickly and clearly identified and extracted.

1   Lattice-gas hydrodynamics

Contents

He [Arthur] looked downwards with intense curiosity.
Between him and the shivering ground
were now some thirty feet of empty air,
empty that is if you discounted the boulders
which didn't stay in it for long,
but bounded downwards in the iron grip of the law of gravity:
the same law which seemed, all of a sudden,
to have given Arthur a sabbatical.

Slowly, carefully, inch by inch, he began to bob downwards,
swinging gently from side to side like a nervous sheet of paper
feeling its way to the ground.

— [Adams82]

1.1 Origins

F LUIDS are differentiated from solids by their continuous deformation under tangential (*shear*) stresses, no matter how small. This *fluidity* provides the basis for fluid dynamics. Their reaction to normal (*pressure*) stresses divides fluids into two main classes: highly *compressible* gases, and only slightly compressible liquids. Liquids are usually treated as *incompressible* for practical purposes, since compressibility introduces thermodynamic considerations which complicate the description of the state of the fluid and its motion. An incompressible treatment of gases is also possible, but only if the change in pressure is small throughout the system.

1.1.1 The hydrodynamic equations

The century-old Navier-Stokes equation, in conjunction with the continuity equation, are known to accurately describe the behaviour of a fluid in a wide range of situations[1], from the simplest leisurely river in a straight channel to the complex turbulence in the extreme conditions of the Jovian atmosphere.

The continuity equation, expressing the conservation of mass, takes the form

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1.1}$$

where ρ is the fluid density and $\mathbf{u}$ the velocity, which simplifies in the case where the density is constant to its incompressible form

$$\nabla \cdot \mathbf{u} = 0 \ . \tag{1.2}$$

The Navier-Stokes equation, pertaining to Newton's Second Law of Motion — that the rate of change of momentum of a fluid particle equals the net force acting on it — can be expressed, for a continuum fluid in D dimensions,

$$\frac{\mathrm{D}(\rho \mathbf{u})}{\mathrm{D}t} + \rho(\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \eta \nabla^2 \mathbf{u} + \left(\zeta + \frac{1}{D}\eta \right) \nabla(\nabla \cdot \mathbf{u}) + \mathbf{F} \ , \tag{1.3}$$

where the substantive derivative — the derivative *following* the motion — is defined,

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \ , \tag{1.4}$$

p is the pressure, η and ζ are the shear and bulk viscosities respectively, and $\mathbf{F}$ is the body force (such as gravity) acting on the fluid.

In the case of a fluid of constant density, **Equation 1.3** reduces to the incompressible Navier-Stokes equation,

---

[1] A heat equation may also be necessary to model thermal phenomena.

$$\rho \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla p + \eta \nabla^2 \mathbf{u} + \mathbf{F} \ , \tag{1.5}$$

with, $\nu = \eta/\rho$, the kinematic (shear) viscosity.

In Cartesian coordinates, **Equation 1.2** becomes:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{1.6}$$

while **Equation 1.5** becomes the set of equations:

$$\left[\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z} \right] = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu \left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right] + \frac{F_x}{\rho} \tag{1.7a}$$

$$\left[\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z} \right] = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu \left[\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} \right] + \frac{F_y}{\rho} \tag{1.7b}$$

$$\left[\frac{\partial w}{\partial t} + u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z} \right] = -\frac{1}{\rho}\frac{\partial p}{\partial z} + \nu \left[\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} \right] + \frac{F_z}{\rho}. \tag{1.7c}$$

Unfortunately, these equations are highly nonlinear, so in all but the most trivial of cases, they are analytically intractable. The use of powerful computers and numerical techniques allow approximate solutions to be made, but great care must be taken with the assumptions made in simplifying the problem. Numerical instabilities and floating-point rounding errors can also make such solutions unreliable.

It is therefore typical, in cases where complete analytic solutions are impossible, to complement the numerical solutions with actual (or scale) experiments. However, this is generally expensive where it can be done, and may be impossible in the most-demanding of situations, such as those encountered by space shuttle orbiters on re-entry to the Earth's atmosphere (see, for example, [Turner89, BH90, Dujarric90]).

Since something close to one-third of all the supercomputer time in the world is used in such fluid flow simulations, a model which did not suffer from the same deficiencies, or which could more effectively utilise massively parallel computers to come up with a faster or more accurate solution, would be of great significance.

Cellular automaton (CA) lattice-gas hydrodynamics (LGH) models arose from the application of ideas of the generality and universality of CAs to dynamic systems such as the simplified models of molecular dynamics.

Richard Feynman's view of lattice-gases, as paraphrased by one of his co-workers, Daniel Hillis [Hillis89], was:

> We have noticed in nature that the behaviour of a fluid depends very little on the nature of the individual particles in that fluid. For example, the flow of sand is very similar to the flow of water or the flow of a pile of ball bearings. We have therefore taken advantage of this fact to invent a type of imaginary particle that is especially simple for us to simulate. This particle is a perfect ball bearing that can move at a single speed in one of six directions. The flow of these particles on a large enough scale is very similar to the flow of natural fluids.

1.1.2 Cellular automata

Cellular automata (CAs[2]) were originally investigated by John von Neumann and Stanislaw Ulam in the early 50's [vonNeumann66]. Space and time are considered to be discrete, with the simple automaton resident at each *time-step* updated according to a small set of rules and using information from only a small number of close neighbours. However, when a number of such automata are considered as an ensemble, their collective behaviour can be extremely complex.

One of the simplest cellular automaton update rules consists of a site's state being *alive* only if one of its neighbours was at the previous time-step, i.e.

$$a_i(t) = [a_{i-1}(t-1) + a_{i+1}(t-1)] \bmod 2 \,. \tag{1.8}$$

From a single seed start in a infinite world this would propagate indefinitely, producing complex structures. Even within a finite world (as shown in **Figure 1.1a**) this is almost always the case.

Greatly varied behaviour can be found with only a slightly more complex automaton rule, e.g. where a site is alive only if two or four of its immediate neighbours and itself were previously alive, i.e.

$$a_i(t) = \begin{cases} 1 & \text{if } \sum_{j \in \{i-2, i+2\}} a_j(t-1) = 2 \text{ or } 4 \\ 0 & \text{otherwise} \end{cases} \,. \tag{1.9}$$

A section of an infinite world evolving from a random initial state of such an automata is shown in **Figure 1.1b**. Most features die rather quickly, however a few can propagate indefinitely — some in space, though more often only within a limited locale as they cycle through a regular, and occasionally extremely complex, sequence.

John Conway popularised the field when he discovered two-dimensional automata capable of growing, reproducing and dying, known as the "Game of Life" [BCG84]. More recently, Stephen Wolfram has further investigated and popularised cellular automata as a tool for simulating hydrodynamic lattice-gases and other physical systems [Wolfram86b, SW86, Wolfram87, Wolfram84b]. He also was able to classify CAs into four basic types [PW85, Wolfram83, Wolfram84a] including classes which appeared random (and could be used as a source of random numbers [Wolfram85a, Wolfram85c, Wolfram85b]), or capable of producing regularly repeating structures on a variety of length scales, i.e. they possessed a *fractal* property [PS88, PT86]. He also conjectured that one class were capable of universal computation [Wolfram84c], i.e. any computation which can be done on a computer could be performed equally accurately, though not necessary efficiently, by a suitable cellular automata.

---

[2] The incorrect pluralisation of the abbreviation is often used when discussing cellular automata.
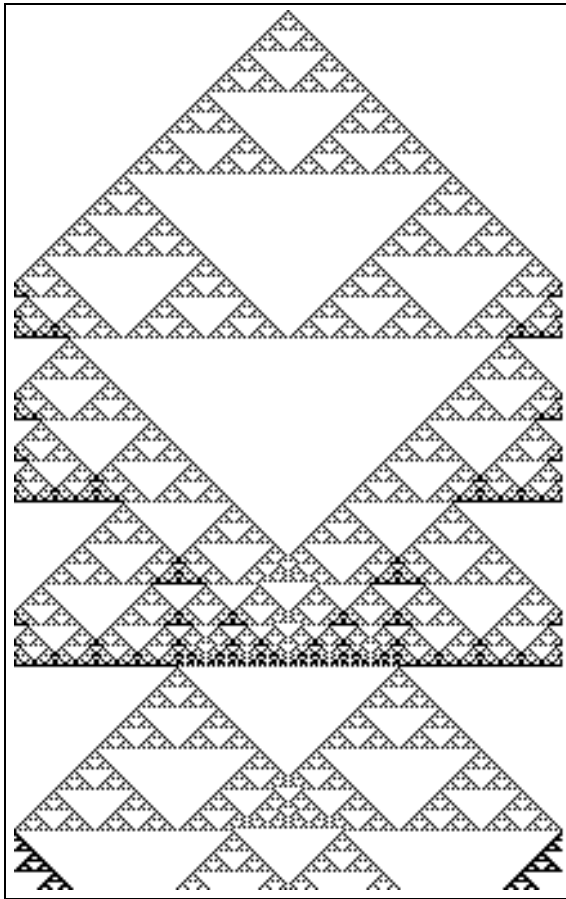
Figure 1.1: Example one-dimensional cellular automata

In each case the initial state is the top line and subsequent lines are updates later. The simple automata of **Equation 1.8** shown on the left generates fractal structures, within a world of 213 sites, from a single site seed. Below, a random initial state of the automata of **Equation 1.9** has produced complex structures propagating in space and time, however most of the initial activity dies within 100 iterations. Space-propagating automata tend to die eventually, by encountering space-static automata, which can propagate indefinitely.

(a) Simple automata, single point start (left)
(b) Complex automata, random start (below)
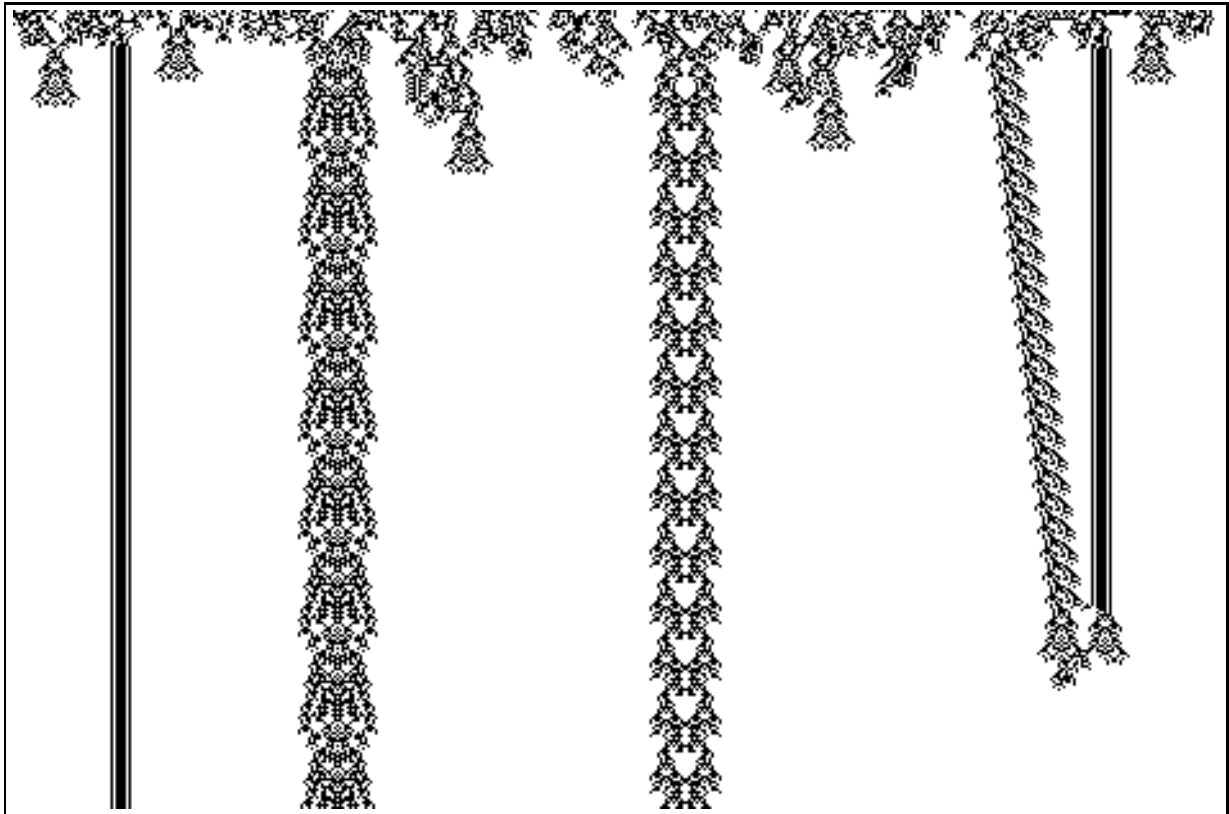
1.1.3 Elementary gases

The elementary Boltzmann gas, in which particles possess discrete velocities from a finite set, is one of the simplest models of a fluid which is true to kinetic theory. The interactions within a gas of hard elastic spheres capable of movement in only six directions (in three dimensions) at constant speeds were considered in [Broadwell64]. Such models lend themselves to analysis and were applied to a range of fluid systems.

The elementary gas models have a similar place in fluid dynamics research as the Ising model does for the theory of phase transitions in ferromagnets. However, for all their simplification, these models were not universally applicable to the wide range of hydrodynamics, and became obsolete.

1.2 Cellular automaton lattice-gas models

Several lattice-gas models have been investigated since the mid-70's when they were first considered. Mostly these have been constrained to two-dimensional lattices; partly for simplicity and partly to minimise computational expense. More recently some three-dimensional models have been proposed and a few simulations done.

1.2.1 HPP

Figure 1.2: HPP configurations
The 16 possible node particle configurations coded into 4 bits.

The original lattice-gas model was that proposed and analysed by John Hardy, Yves Pomeau and Otto de Pazzis in the mid 70's [HP72, HPdP73, HdPP76]. It consisted of a square lattice, boolean particles of unit mass and momentum residing on the links at each vertex. The possible configurations of particles at each node are shown in **Figure 1.2**, along with one possible coding of the configurations' particles into a four bit word.
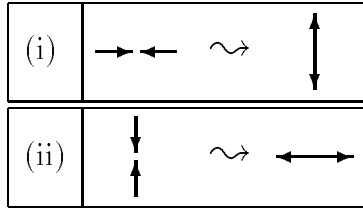
Figure 1.3: HPP collision rules
The configuration update rules for the *HPP* model, corresponding to interchange of configurations 0101 and 1010.
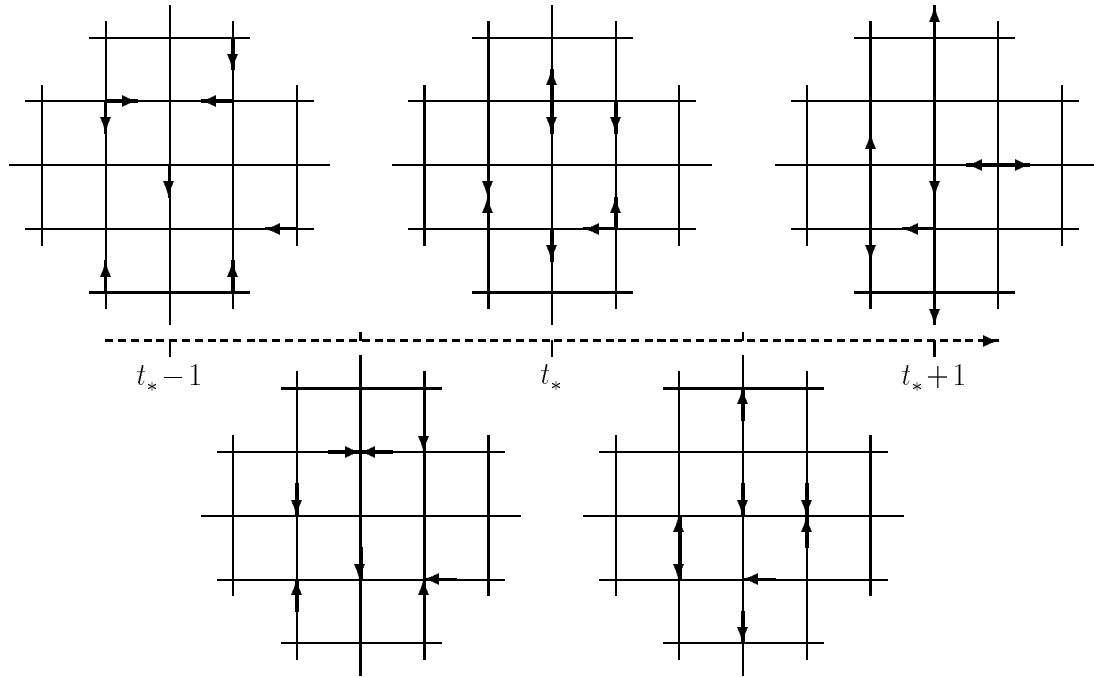
Figure 1.4: HPP lattice update example
A sample of a fragment of the *HPP* lattice is shown over two time-steps around the time t_*. Collision update is considered to happen only at the intermediate stages between propagation updates. Two examples of actual collisions can be identified: at the previous upper, central node and a subsequent middle, right node.
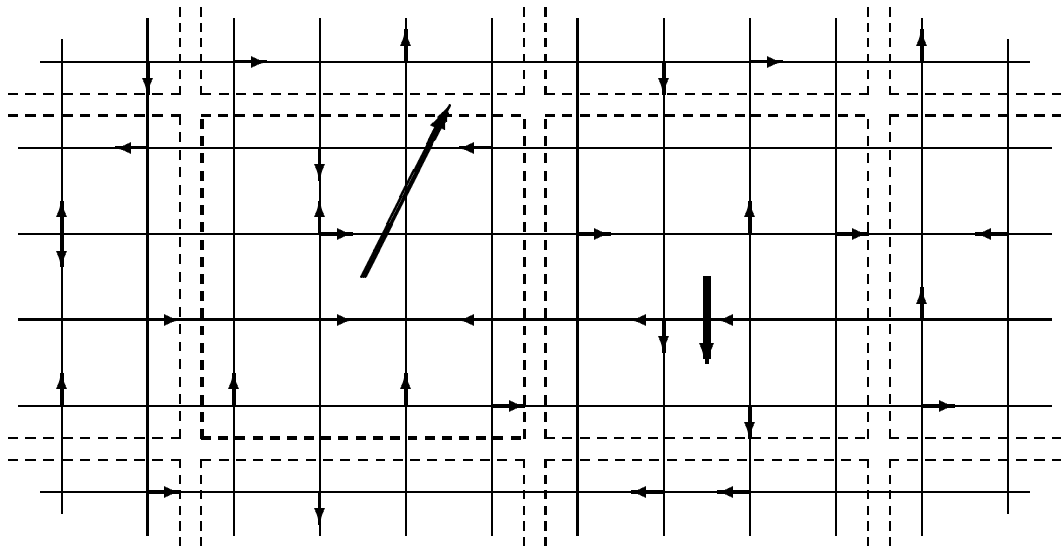
Figure 1.5: HPP lattice cells
A sample of a fragment of the *HPP* lattice is shown along with the outlines of the 4×4 averaging cells within which the macroscopic quantities are calculated. The highlighted cell (with the bold outline), has 9 particles, therefore the *site density* is $\frac{9}{16}$ and the *link density* is $\frac{9}{64}$. Summing the particles' momenta produces a cell momentum of 1 unit to the right, and 2 units to the top, i.e. magnitude $\sqrt{3}$ at $\sim 55°$, as shown by the large, bold vector.

Update of the lattice consisted of two stages: *propagation* of each particle in the direction of its momentum, from its current site to a neighbouring one, alternated with *collision* where the particles present at a node interact such that the number of particles (mass) and momentum are conserved. In this model, only two configurations can be updated consistently, as shown in **Figure 1.3**; all other updates would violate a conservation law, or be indistinguishable from the incoming state. **Figure 1.4** shows the various stages in the update of a fragment of a lattice.

Regular partitioning of the lattice into *cells*[3] consisting of a local collection of lattice sites, such as the 4 by 4 tiles of **Figure 1.5**, was used to extract local average values for measurables such as density and momentum flux.

It was these macroscopic averages which were considered to represent the continuous fluid, rather than the discrete values from the lattice sites. Microscopic details are removed by this averaging.

Reflection of particles from collisions with barriers, within the lattice or on its edges, through the reversal of their direction of propagation, or *wrap* boundaries which re-inject particles leaving on one edge at its opposite edge, enable elaborate flows to be designed and simulated easily, at little additional complexity to the basic model.

---

[3]not to be confused with the much simpler cells of the cellular automaton, which correspond to *single* sites of the lattice-gas
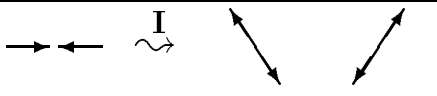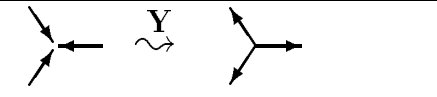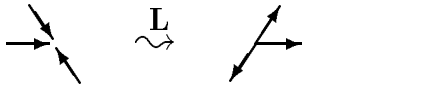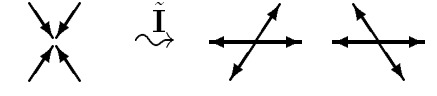
(i)	→ ←	**I** ⤳	↘	↗	3
(ii)	↘↗←	**Y** ⤳	⤚		2
(iii)	→↘	**L** ⤳	↗→		12
(iv)	↘↗↘↗	**Ĩ** ⤳	⤙↗	←↘→	3

Figure 1.6: FHP6 collision rules
Representatives of the collision channels for the FHP6 model.

Some simple investigations using this model quickly showed that the underlying lattice lacked sufficient symmetry to allow a rich collision set, and that as a consequence the prefered directions of the lattice remained after macroscopic averaging. Such a lattice-gas, although not able to model general hydrodynamics because of this lack of isotropy, was capable of accurately modelling sound wave propagation [MTV86, FdHHLPR87]. If a lattice-gas could be found which had higher symmetry (while retaining the elegance and integrity of the basic model), then this would allow a greater diversity of particle interaction and might allow full hydrodynamics to be tackled.

1.2.2 FHP6

10 years later, the first model to accurately simulate hydrodynamics was described by Uriel Frisch, Brosl Hasslacher and Yves Pomeau [FHP86]. The progression from a square lattice to a hexagonal lattice, allows the interaction of six particles at each vertex, while still retaining a simple space-filling regular grid. Of the 64 configurations possible for the particles at a vertex, 20 can be rearranged while preserving momentum, and in 6 of these there is a choice of two possibilities.

A complete classification of the configuration codes, $\mathbf{C}$, for the *FHP6* model (and its extension, *FHP7*, incorporating a possible rest particle, which will be discussed later) can be found in **Appendix C**. The *FHP6*-specific entries are limited to the first four tables corresponding to binary codes 000**** — 011****. $\mathbf{C}$ denotes an arbitrary configuration class, the presence of a rest particle is shown as the subscripted bullet, $\mathbf{C_\bullet}$, and a 'dual' class represented with a tilde overhead, i.e. $\tilde{\mathbf{C}}$ or $\tilde{\mathbf{C}}_\bullet$. (Duality is discussed in **Section 1.3**.) Only a subset of the listed collision channels may be actually used, and for the *FHP6* model, channels containing rest particles are unavailable.

These collision channels are shown graphically in **Figure 1.6**, and represented by the equations:

$$\text{linear} \qquad \mathbf{I} \rightsquigarrow \{\mathbf{I}^2\}, \qquad\qquad\qquad\qquad\qquad\qquad\qquad (1.10a)$$
$$\text{triple} \qquad \mathbf{Y} \rightsquigarrow \mathbf{Y}, \qquad\qquad\qquad\qquad\qquad\qquad\qquad (1.10b)$$
$$\text{lambda} \qquad \mathbf{L} \rightsquigarrow \mathbf{L}, \qquad\qquad\qquad\qquad\qquad\qquad\qquad (1.10c)$$
$$\text{dual-linear} \quad \tilde{\mathbf{I}} \rightsquigarrow \{\tilde{\mathbf{I}}^2\}. \qquad\qquad\qquad\qquad\qquad\qquad\qquad (1.10d)$$

The *linear* channel, **I** (**Equation 1.10a**), updates three equivalent configurations, rotationally symmetric, of which **Figure 1.6 (i)** is representative. It consists of a head-on collision between two particles, and two outcomes are possible[4], differing only in the sense of the rotation, such that the particles are transfered to another of the major axes of the lattice.

The *triple* channel, **Y** (**Equation 1.10b**), updates two rotationally symmetric configurations, of which **Figure 1.6 (ii)** is one. The three-particle interaction transfers particles between the major axes, and is an important mechanism in the breaking of the spurious invariant which plagues the *HPP* model with its lower lattice connectivity.

Together the linear and triple collision channels form the minimal set of collision update rules which let particles interact while preventing spurious conservation laws. It is this model that was initially investigated in works such as [dHPL85, dHL87], and analysed in [FHP86, dHL86b, dHL87, FdHHLPR87], where it was known as *FHP-I*.

The *lambda* channel, **L** (**Equation 1.10c**), consists of a linear pair of particles which are restricted in their choice of outgoing configuration by a *spectator* particle. Twelve such configurations are possible, rotational and symmetric variations of the example shown in **Figure 1.6 (iii)**.

The final collision channel is the dual of the linear channel, where the interaction can be considered in terms of a linear collision of holes or one of a couple of possible linear channels. An example of this *dual-linear* channel, $\tilde{\mathbf{I}}$ (**Equation 1.10d**), is shown in **Figure 1.6 (iv)**.

No other particle re-arrangements are possible without violating the desired conservation laws, resulting in a collision set of 20 channels. If duality-invariance[5] is not desired in the model then the latter channel can be omitted, though for the minimal decrease in complexity little benefit would be expected.

An example of the update of a fragment of the *FHP6* hexagonal lattice is shown in **Figure 1.7**. Actually, it is depicted as a *triangular* lattice with the particles residing at the vertices of the lattice, rather than the entirely equivalent (and much more representative) *hexagonal* lattice with particles at the centres of the hexagons. In both views, particles propagate from their initial site to one of their six neighbours.

---

[4]Where a number of outcomes are possible these are denoted within braces, and superscripted with the number of the same type.

[5]equivalence under substitution of particles and 'holes'
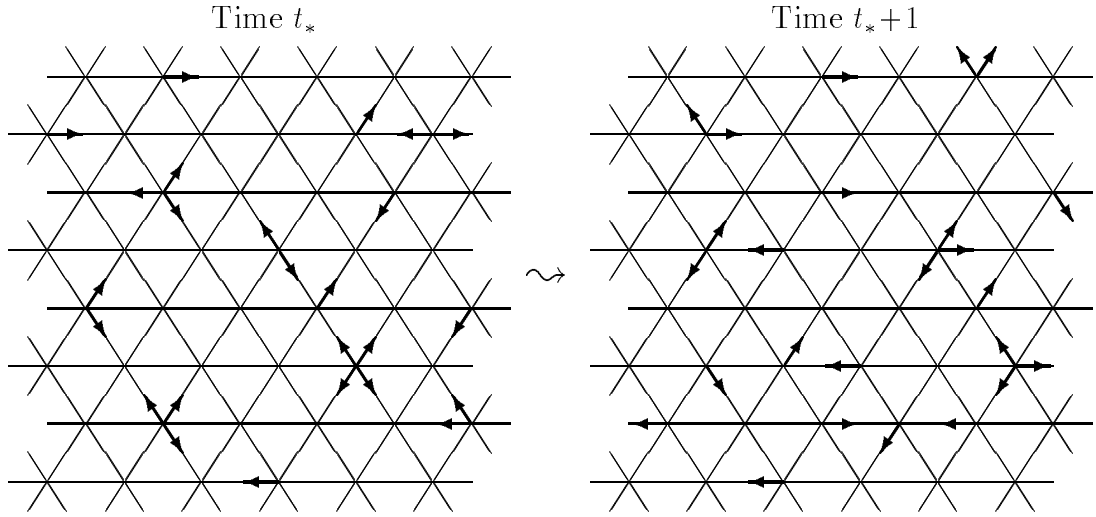
Figure 1.7: FHP6 lattice update example

A sample of a fragment of the *FHP6* lattice is shown at time t_*, along with its subsequent state after a *complete* update, consisting of collision *then* propagation. A total of 5 collision configurations were updated, representing all of the possible channels: $2 \times$**I**, **L**, **Y** and **Ĩ**.

As was done in the *HPP* model, cell averaging of the lattice provides macroscopic values for the local mass and momentum, but since the lattice is now hexagonal the cells are necessarily distorted, as shown in **Figure 1.8**.

1.2.3 FHP7

The addition of stationary particles (i.e. rest particles at the vertices) was also proposed by Frisch, Hasslacher and Pomeau in their introductory paper [FHP86]. In the *HPP* model, rest particles would have added nothing to the dynamics, since it is impossible to incorporate them into additional collision rules. However, several additional collision types are possible by introducing rest particles to the basic *FHP6* model, resulting in momentum transfer between the moving and stationary particles. These extra collision channels have a favourable effect in reducing the kinematic viscosity and thereby increasing the Reynolds number.

It has been shown both analytically, from the Rivet and Frisch calculation of the viscosity [RF86, FR86], and experimentally from measurements of the relaxation of shear waves [dHL86b, Rivet87a], that the richest collision sets produce the maximum possible Reynolds number. Additional rest particles are useful in this respect.

The full list of configuration codes and classifications shown in **Appendix C** are applicable to the *FHP7* model. All available channels are shown, though only a subset may be actually used.
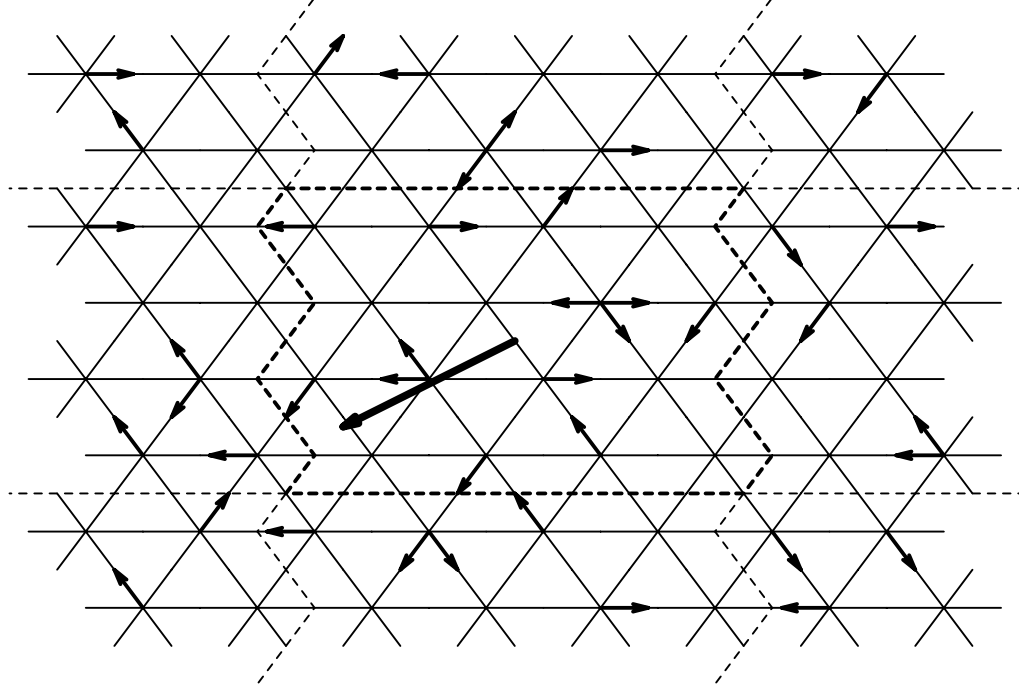
Figure 1.8: FHP lattice cells

A sample of a fragment of the *FHP6* lattice is shown along with the outlines of the 4×4 averaging cells within which the macroscopic quantities are calculated. The highlighted cell (with the bold outline), has 13 particles, therefore the *site density* is $\frac{13}{16}$ and the *link density* is $\frac{13}{96}$. Summing the particles' momenta produces a cell momentum of 0 units purely along the x-axis, 2 units to the lower-left and 1 unit to the upper-left, i.e. magnitude $\sqrt{3}$ at $\sim 215°$, as shown by the large, bold vector.

As can be seen from the complete *FHP7* collision set shown in **Figure 1.9**, the *FHP6* collision channels (**Equation 1.10a**) are still valid, however, they are supplemented by an additional set:

$$
\begin{array}{llll}
triple & \mathbf{Y} \rightsquigarrow \{\mathbf{Y},\mathbf{I_\bullet}^3\}\,, & & (1.11a) \\
linear{+}rest & \mathbf{I_\bullet} \rightsquigarrow \{\mathbf{I_\bullet}^2,\mathbf{Y}^2\}\,, & & (1.11b) \\
fundamental{+}rest & \mathbf{F_\bullet} \rightsquigarrow \quad \mathbf{J}\,, & & (1.11c) \\
jay & \mathbf{J} \rightsquigarrow \quad \mathbf{F_\bullet}\,, & & (1.11d) \\
lambda & \mathbf{L} \rightsquigarrow \{\mathbf{L},\mathbf{J_\bullet}\}\,, & & (1.11e) \\
jay{+}rest & \mathbf{J_\bullet} \rightsquigarrow \quad \{\mathbf{L}^2\}\,, & & (1.11f)
\end{array}
$$

and their duals:

$$
\begin{array}{llll}
dual\text{-}triple & \tilde{\mathbf{Y}}_\bullet \rightsquigarrow \{\tilde{\mathbf{Y}}_\bullet,\tilde{\mathbf{I}}_\bullet^3\}\,, & & (1.12a) \\
dual\text{-}linear{+}rest & \tilde{\mathbf{I}}_\bullet \rightsquigarrow \{\tilde{\mathbf{I}}_\bullet^2,\tilde{\mathbf{Y}}_\bullet^2\}\,, & & (1.12b) \\
dual\text{-}fundamental & \tilde{\mathbf{F}} \rightsquigarrow \quad \tilde{\mathbf{J}}_\bullet\,, & & (1.12c) \\
dual\text{-}jay{+}rest & \tilde{\mathbf{J}}_\bullet \rightsquigarrow \quad \tilde{\mathbf{F}}\,, & & (1.12d) \\
dual\text{-}lambda{+}rest & \tilde{\mathbf{L}}_\bullet \rightsquigarrow \{\tilde{\mathbf{L}}_\bullet,\tilde{\mathbf{J}}\}\,, & & (1.12e) \\
dual\text{-}jay & \tilde{\mathbf{J}} \rightsquigarrow \quad \{\tilde{\mathbf{L}}_\bullet^2\}\,. & & (1.12f)
\end{array}
$$

15

(i)		3
(ii)		2
(iii)		3
(iv)		6
(v)		6
(vi)		12
(vii)		6
(viii)		3
(ix)		2
(x)		3
(xi)		6
(xii)		6
(xiii)		12
(xiv)		6

Figure 1.9: FHP7 collision rules

Representatives of the collision channels for the FHP7 model. Channels (viii)—(xiv) are the duals of (i)—(vii). The duplicity under rotations and reflections, producing similar channels, is shown in the final column.

The most interesting new possibilities are the co-channels *fundamental with rest*, $\mathbf{F_\bullet}$ (**Equation 1.11c**) and *jay*, $\mathbf{J}$ (**Equation 1.11d**), which allow momentum to be exchanged between rest and moving particles.[6]

The *FHP6 triple* channel, $\mathbf{Y}$ (**Equation 1.10b**) has been supplemented by the provision of three further channels (**Equation 1.11a**) producing stationary particles. This can be considered as two particles of the triple undergoing a $\mathbf{J}$-type interaction. These new channels are much more efficient than the previous three-particle interaction at exchanging momentum in the major lattice axes.

Similarly, the other channels, and their duals, are derived from these simple two-particle interactions, even when many particles are involved.

It is this model, with its fully saturated collision set, which has been most favoured in lattice-gas simulation in recent years, and which forms the basis for the simulations described in the following chapters. In other works, such as [dHL87] and [FdHHLPR87] it is referred to as *FHP-III*.

Variants of this model using subsets of the full collision set, carefully chosen to ensure that the model is still valid, are also possible. An example would be to only augment the *FHP6* channels, with two-body collisions interchanging rest and moving particles, as described as *FHP-II*[7] in [dHL87]. These models have poorer characteristics, such as higher viscosities, therefore tend not to be generally used, except perhaps where hardware constraints limit model complexity.

An example update of a fragment of the *FHP7* lattice is shown in **Figure 1.10**. This fragment, rather similar to that used in the *FHP6* example (**Figure 1.7**), has far more possible collision possibilities. **Plate 1** shows two consequtive frames from the implemented lattice-gas simulator, while also includes examples of reflections from wall sites and absorption and generation of particles by sources.

A full classification of the *FHP6* and *FHP7* model collision configurations, by their particle number and net momentum which require to be conserved, are shown in **Table 1.1**.

1.2.4 FCHC

The only model to have been seriously used for three-dimensional lattice-gas simulations is that of Dominique d'Humières, Pierre Lallemand and Uriel Frisch [dHLF86], which is fully discussed in [FdHHLPR87]. A pseudo four-dimensional face-centred hypercubic lattice is projected into three dimensions, since none of the regular three-

---

[6]Since kinetic energy is not conserved in this type of collision, energy conservation requires the stationary particle to possess an extra unit of internal energy. However, this is insufficient to produce a properly thermal model.

[7]corresponding to the following active channels involving 22 configurations: $\mathbf{I} \rightsquigarrow \{\mathbf{I}\}$, $\mathbf{Y} \rightsquigarrow \mathbf{Y}$, $\mathbf{I_\bullet} \rightsquigarrow \{\mathbf{I_\bullet}\}$, $\tilde{\mathbf{Y}}_\bullet \rightsquigarrow \tilde{\mathbf{Y}}_\bullet$, $\mathbf{F_\bullet} \rightsquigarrow \mathbf{J}$ and $\mathbf{J} \rightsquigarrow \mathbf{F_\bullet}$. Note particularly the absence of the channels $\mathbf{Y} \rightsquigarrow \{\mathbf{I_\bullet}\}$ and $\mathbf{I_\bullet} \rightsquigarrow \{\mathbf{Y}\}$ and the inclusion of only $\tilde{\mathbf{Y}}_\bullet$ configurations in the collision set and not their $\tilde{\mathbf{I}}_\bullet$ equivalents.

Classification		Code	#	m	p	Channels
Empty		**E**	1	0	0	
Empty	+rest	**E.**	1	1	0	
Fundamental		**F**	6	1	1	
Fundamental+rest		**F.**	6	2	1	**J**
Vee		**V**	6	2	$\sqrt{3}$	
Vee	+rest	**V.**	6	3	$\sqrt{3}$	
Jay		**J**	6	2	1	**F.**
Jay	+rest	**J.**	6	3	1	$\{\mathbf{L}^2\}$
Lambda		**L**	12	3	1	$\{\mathbf{L},\mathbf{J.}\}$
Linear		**I**	3	2	0	$\{\mathbf{I}^2\}$
Linear	+rest	**I.**	3	3	0	$\{\mathbf{I.}^2,\mathbf{Y}^2\}$
Sigma		**W**	6	3	2	
Triple		**Y**	2	3	0	$\{\mathbf{Y},\mathbf{I.}^3\}$
dual-Triple	+rest	**Ỹ.**	2	4	0	$\{\tilde{\mathbf{Y}}\mathbf{.},\check{\tilde{\mathbf{I}}}^3\}$
dual-Sigma	+rest	**W̃.**	6	4	2	
dual-Linear		**Ĩ**	3	4	0	$\{\tilde{\mathbf{I}}^2,\check{\mathbf{Y}}\mathbf{.}^2\}$
dual-Linear	+rest	**Ĩ.**	3	5	0	$\{\check{\tilde{\mathbf{I}}}\mathbf{.}^2\}$
dual-Lambda	+rest	**L̂.**	12	4	1	$\{\tilde{\mathbf{L}}\mathbf{.},\tilde{\mathbf{J}}\}$
dual-Jay		**J̃**	6	4	1	$\{\tilde{\mathbf{L}}\mathbf{.}^2\}$
dual-Jay	+rest	**J̃.**	6	5	1	**F̃**
dual-Vee		**Ṽ**	6	4	$\sqrt{3}$	
dual-Vee	+rest	**Ṽ.**	6	5	$\sqrt{3}$	
dual-Fundamental		**F̃**	6	5	1	**J̃.**
dual-Fundamental+rest		**F̃.**	6	6	1	
dual-Empty		**Ẽ**	1	6	0	
dual-Empty	+rest	**Ẽ.**	1	7	0	

Table 1.1: Particle configuration classifications
Each particle configuration, **C**, (# is the number of such configurations differing only in rotations and reflections) is categorised by its mass, m, i.e. the number of (link) particles, and the magnitude of its net momentum, p. Dual configurations, categorised with the tilde, $\tilde{\mathbf{C}}$, result from the substitution of rest and link particles with holes, and vice versa. Configurations with stationary particles are coded by the addition of a bullet, **C.** (or $\tilde{\mathbf{C}}\mathbf{.}$).

This is a full classification for the FHP7 model; the reduced classification for the FHP6 model doesn't consist of any of the +rest (**C.**) configurations, with the corresponding removals from the collision channel options of the remainder. The existence of several collision channels is shown braced, i.e. {**C**}, with the number of alternatives of each type (if any) shown superscripted.
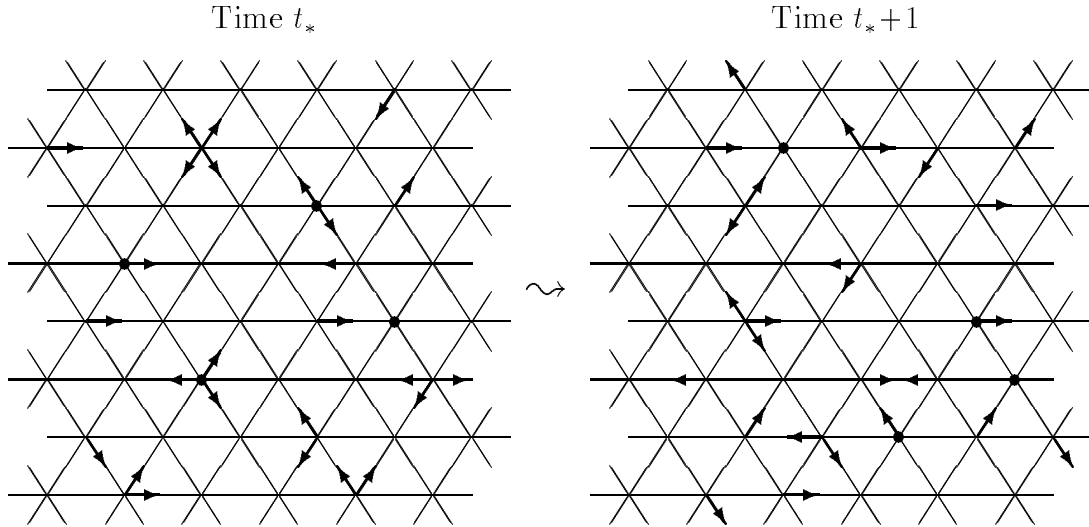
Figure 1.10: FHP7 lattice update example
A sample of a fragment of the *FHP7* lattice is shown at time t_*, along with its subsequent state after a *complete* update, consisting of collision *then* propagation. A total of 6 collision configurations were updated, representing a few of the possible channels: $\mathbf{F_\bullet} \rightsquigarrow \mathbf{J}$, $\mathbf{J} \rightsquigarrow \mathbf{F_\bullet}$, $\mathbf{I_\bullet} \rightsquigarrow \mathbf{Y}$, $\mathbf{L} \rightsquigarrow \mathbf{J_\bullet}$, $\mathbf{\tilde{Y}_\bullet} \rightsquigarrow \mathbf{\tilde{I}}$ and $\mathbf{\tilde{I}} \rightsquigarrow \mathbf{\tilde{Y}_\bullet}$.

dimensional space-filling lattices have sufficient symmetry to avoid anisotropies [Wolfram86a].

A node of the *FCHC* lattice showing the links to the neighbouring sites is shown in **Figure 1.11**. With 24 links, and the possibility of a number of rest particles, an extremely rich collision set is possible with millions of possible collisions. A configuration classification along the lines of that undertaken for the *FHP* models in 2D can be undertaken, where configurations are replaced wherever possible by a (pseudo-random) selection from the range of alternative configurations with identical particle numbers, momentum[8], etc. Considerable optimisation of the basic model is possible to improve the performance [Hénon87a].

1.2.5   Others

Various other models have been proposed at one time or other, and may be preferable in certain types of simulation. One of these consisted of a regular square lattice, and allowed particles of $\sqrt{2}$ velocity moving along diagonal links in addition to particles of unit and zero velocity. This was one of few models which had a minimal range of velocities and may be useful in incorporating temperature characteristics. Feynman investigated random lattices for lattice-gas models without apparent success ([Hillis89]). For three-dimensional simulations a combination of

---

[8]The momentum component in the fourth dimension must be separately conserved in the collision update, even though in the propagation dynamics it plays no part.
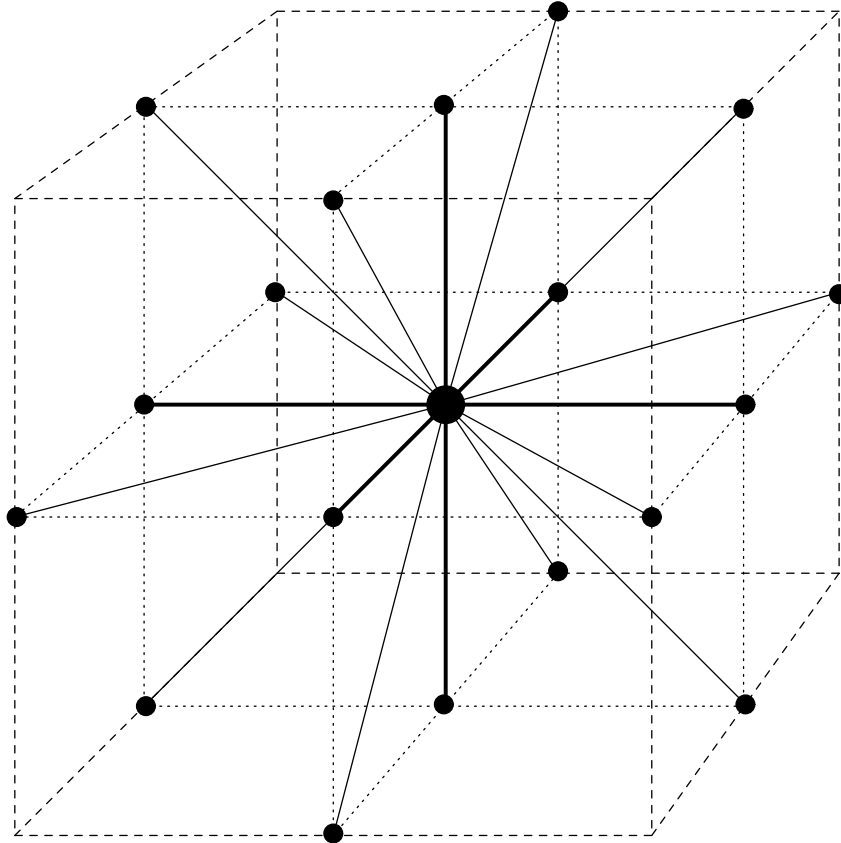
Figure 1.11: FCHC node neighbourhood
The neighbourhood of a single lattice node in the three-dimensional FCHC model comprises 6 nearest neighbours and 12 next-nearest neighbours. A maximum of 24 moving particles can reside at the central node (shown larger than the rest) at any instant, along with an arbitrary number of rest particles. On the nearest neighbour links (shown by bold solid lines) up to 2 particles can propagate differing only in their, otherwise unused, 4D 'quantum' number, while only a single particle can propagate on each next-nearest neighbour link (shown by thin solid lines).

two regular 3D lattices was proposed, with propagation alternately on one then the other.

In general, the provision of additional rest particles is an accepted method for reducing the kinematic viscosity. More than a single rest particle is possible, though typically only one would participate in any collision. Particles of multi-unit masses, stationary at lattice sites can also be beneficial in reducing viscosity, so quite a variety of modifications can be made to optimise the basic lattice-gas models for the desired flow characteristics [Hénon87a].

1.2.6 The models investigated

1.2.6.1 2D models

Initial investigation was of the *HPP* model, to familiarise with the required techniques. It was quickly found that this simple model was considerably deficient for general simulations and this manifested itself most dramatically in harsh shadowing (or incomplete mixing) in barrier wakes. However, simulations of pressure waves, as reported in [MTV86] were satisfactory and provided confidence and incentive to progress to the hexagonal lattice models.

The *FHP6* model was later implemented, followed soon afterwards by the *FHP7* model. The optimally saturated *FHP7* model was chosen to form the basis of the investigations in the following chapters.

1.2.6.2 Why not 3D ?

A 3D model implementation was seriously considered on several occasions. Apart from the lack of 3D visualisation tools, which would have hampered such fluid investigations, a more critical concern was the requirements of the model itself. For the *FCHC* model, a 100 Mbyte collision look-up table is almost feasible in a large supercomputer main memory, but utterly ridiculous in a distributed memory situation where each processor has a total of 4 Mbyte for look-up table, lattice and code. Although such a table could have been distributed, the performance penalty in accessing a remote table on a very frequent basis would have been crippling.

The cost of attempting all the various collision particles using bit manipulation was also estimated to be prohibitive, at least in the absence of special purpose hardware for the task.

Recently, J. A. Somers and P. C. Rem [SR89] have proposed and implemented a variant of the basic *FCHC* model which uses the symmetry about the 24 axes to reduce the size of the collision tables. Each configuration has its net momentum vector rotated such that it lies within the desired section, a suitable update is selected for the particles, and they are finally rotated back to preserve the momentum flux vector. It was found that the collision tables could be reduced to less than 64 kbytes, which could easily be incorporated onto each of a number of distributed processors.

Such techniques are likely to facilitate more extensive simulation of three-dimensional flows using lattice-gas methods in the future.

1.2.7 Model extensions

A number of extensions to the basic lattice-gas models, in two or three dimensions, to enable simulations of mixtures of fluids or different flow systems have also been proposed and implemented.

1.2.7.1 Multi-species

When considering the interaction between fluids of different types, these are simply represented by *tagging* particles with their requisite type, which must be tracked through their propagation and collision update phases. If a preferential update, such as *gravitational separation*, is required then this can additionally be performed on any or all types. Alternatively, some form of *chemical interaction* between types, where one will preferentially replace another, is also possible. The relative fractions of each type, or their concentrations are easily determined, by considering each type individually when performing the macroscopic averaging phase.

Examples of the implementation of such multiple species models, through tagging each of the bits corresponding to particles with an additional bit, generally known as its *colour*, which propagates with the particle, can be found in [dHLS87, BTR89].

Initial conditions with a heavier fluid above a lighter one will naturally produce the Rayleigh-Taylor instability [CdHLP86]; modelling two fluids of different viscosities in a porous medium allows study of the Saffman-Taylor instability [SRB89]; applying a shearing flow at the interface produces the Kelvin-Helmholtz instability [dHLS85]; and similarly the Rayleigh-Benard instability [BZ87] and the surface tension of immiscible fluids [RK88] have been simulated with these models.

1.2.7.2 Miscellaneous

Numerous models, some of which have been rather exotic, have also been proposed or actually simulated: for example, magnetohydrodynamics [CM87b, HM87, MD87], waves and unsteady flows [CCDL88, Lim88], buoyancy and seepage effects [BZ87], solid-fluid suspensions and Brownian motion [LCF88], reaction-diffusion models [dHLS87], interfaces and combustion phenomena [CdHLP86], Burger's model and diffusion [BL88], and two-dimensional turbulence [SSB88].

The later example is of special significance, since [OY86] conjectured that lattice-gases were not a computationally efficient alternative to standard numerical codes in the simulation of turbulent flows. Their arguments were based on the scaling of Reynolds number with the lattice dimension, for a gas of fixed (or only slowly varying) viscosity. However, with optimised collision rules providing the possibility of arbitrary viscosities and specially designed hardware for lattice-gas updates this is currently far from certain. Indeed, where complex geometries are concerned, it is conceivable that lattice-gas methods would be more efficient than existing methods.

1.3 Model fundamentals

1.3.1 Minimal basis

1.3.1.1 Basic properties

In constructing a lattice-gas model, there can be a number of properties which are desirable. These may be from the point of simplicity, so that unnecessarily complicated features are avoided, or alternatively, efficient implementation on a digital computer (or specially designed hardware) may be the goal.

Boolean particles   The simplest entity which can exist in a simulation is a single bit, which may be set or not. In associating a particle with such an entity, the only information maintained about the particle is whether it exists or not. It is possible to derive additional information from where the particle is found, such as its velocity if it resides on a certain link, but it cannot be distinguished from any other such particle[9].

Exclusion principle   A further simplification is to only allow one (indistinguishable) particle to occupy a particular state at a time, where a particle's *state* consists of information about its position and velocity[10].

Simplified microdynamics   Particles are constrained to reside on the sites[11] of a lattice, and to instantaneously *hop* between sites on the pulse of a regular timestep. Particles need only then be considered to interact at the sites of the lattice, again instantaneously.

Ficticious model   All these features result in a fictitious model in which particle movement is only between lattice sites, and particle interaction is only between the small number of particles. Both propagation and interaction are considered to happen simultaneously by all the existing particles, and are alternated to form a complete lattice update.

1.3.1.2 Consequences

There are a few consequences to the choice of simplifications made in constructing the basic model. These are not necessarily undesirable, but can have certain ramifications which need to be noted.

---

[9] at least without specially marking particles so they can be used as *tracers*.
[10] and also other attributes, such as colour, etc.
[11] actually, the *links* at the sites

Reversibility In general, the microdynamics are completely reversible, such that reflection of the direction of propagation of all the particles at a particular instant, has the effect of reversing the direction of time evolution. However, this requires complete knowledge of the state of every particle, which gets increasingly prohibitive with larger systems, and, if non-deterministic collision choices have been allowed, will require knowledge of their history as well. This microdynamic reversibility models that of molecular dynamics which we are mimicking, and is lost in the macrodynamic averaging — as also happens in real-world hydrodynamics and thermodynamics.

Galilean invariance Lattice-gases, such as the *FHP7* hydrodynamic model, have a prefered reference frame — that in which the lattice is at rest — therefore Galilean invariance does not hold at the microscopic lattice level, i.e. they are not invariant under arbitrary (non-relativistic) spatial transformations. This is due to the use of a finite set of directions for the velocity, and to the exclusion principle that leads to the boolean character of the particles.

However, Galilean invariance *is* restored in the incompressible limit [FdHHLPR87]. Recently, [dHLS87] has shown that it can be restored through tuning of the collision rules, and therefore models can be cured of the Galilean disease.

Duality The removal of every particle, and the replacement of every former *hole* with a particle, will result in an entirely equivalent model, since the dynamics of holes[12] in their propagation and interaction is identical to that of particles. Actually, duality-invariance is only the case when the collision set has been so chosen to be symmetric for particles and non-particles. The dual nature of the lattice-gas is manifest in a deviation from the expected form of the non-linear term of the Navier-Stokes equation by a factor, $g(\rho)$.

Since every additional collision channel contributes to lowering the viscosity of the lattice-gas, generally, it is preferable to include the dual collision channels. Alternatively, by biasing collisions which create and destroy rest particles, the duality factor can be normalised (i.e. $g(\rho) = 1$) [dHLS87]. The resulting model is, however, only valid over a small density range, while violating semi-detailed balance for a subset of the collisions.

1.3.1.3 Conservables

Details of the simplifications of the microdynamical model should not be evident in the *observables* of the macrodynamic world. As such, several points should be borne in mind:

---

[12]in the solid state sense

Conservation laws Since mass and momentum should be conserved in the observed macrodynamics, it is important that the microdynamics adhere to this. Propagation achieves this by solely moving particles from their initial site to their final site in accordance with their momentum. More choice is involved in the design of the collision rules, whereby momentum is redistributed between the particles resident at each site after propagation. Generally, only a small number of configurations of particles are possible, and to replace a particular configuration with another with the same number of particles (to conserve mass) and with the same net momentum may not be possible, so typically no interaction occurs.

It is desirable that whenever a possible replacement could occur that it should happen, since this will lead to more varied dynamics. This results in a reduction of the mean free path, λ, between collisions, and kinetic theory suggests that the viscosity,

$$\nu \cong \tfrac{1}{2}\lambda. \tag{1.13}$$

On length scales much larger than the mean free path, the system is likely to behave like a continuum fluid.

Isotropy The isotropy of the macroscopic gas dynamics is strongly dependent on the crystallographic group of the underlying lattice. In two-dimensions, the *HPP* model suffers from the prefered directions of the square lattice resulting in prefered axes for flow propagation. The hexagonal lattice of the *FHP* model, with its greater symmetry, can be freed of such problems (provided a balanced collision set is constructed), as was shown in [Wolfram86a], and experimentally verified. It is sometimes possible to counteract the problem of lattice anisotropy by careful *tuning* of the particle collision interactions, but it is preferable to avoid this complication where possible. The *FCHC* model is naturally isotropic and therefore makes a good choice for three-dimensional lattice-gas simulations.

Chirality Where several collision channels exist, it is important that no asymmetry or bias in the sense of rotation of the particles in the configuration be introduced. Typically, the redistribution of momentum in a net-zero momentum configuration, can be viewed as a rotation of the whole configuration about its central node. Considering the *FHP6* model, examples of these are the Linear and Triple channels, **I** and **Y**, of **Figure 1.6 (i)** and **(ii)**.

The Triple channel $\mathbf{Y} \rightsquigarrow \mathbf{Y}$ is symmetric in its update, with the outgoing configuration equivalent to the original under a rotation of $\pm\frac{\pi}{3}$.[13] However, the Linear channel $\mathbf{I} \rightsquigarrow \{\mathbf{I}\}$ has two possible outcomes, dependent on whether the $\frac{\pi}{3}$ rotation is positive or negative. In such a situation the decision to always rotate in one sense would result in a *chiral* model, while randomly choosing between the two would

---

[13]In the *FHP7* model, the Triple channel, $\mathbf{Y} \rightsquigarrow \{\mathbf{I_\bullet}\}$ is also available, and any preference in selecting one of the available outcomes could lead to chirality.

produce an *achiral* model in which there was no prefered sense of rotation, typical of most[14] physical fluids.

Actually, so long as the chirality is in some way destroyed in the microdynamics, for instance by equivalencing the sense of the rotation to the parity of the lattice site of the configuration, the macrodynamics will be achiral.

1.3.2 Achieving macrodynamics

1.3.2.1 Cell averaging

Having specified the microdynamics of the lattice-gas, macrodynamics is simply achieved by subdividing the micro-lattice into regular cells within which the average properties of the underlying sites and particles can be calculated. It is the collective behaviour of a number of such cells which shows how a flow is developing, rather than the much more intricate interactions of the individual particles, or any particular site.[15]

The natural units of time, length and mass in lattice-gas simulations are the time-step, link-length (between sites) and particle mass respectively. Velocities are therefore quoted in link-lengths per time-step.

Density Averaging the number of particles per site in a cell produces the *mean site density*,

$$\rho = \sum_i N_i \,, \tag{1.14}$$

where the N_i are the macroscopically averaged local populations of the boolean states (i.e. the link populations, so which there are seven for the *FHP7* model).

Alternatively, averaging the number of particles per link in a cell will produce the model-independent *mean link density*, d. This *reduced* density is more natural since the definition of density on a hexagonal lattice is slightly different from usual. Because the number of links per site is fixed in each model, the relation between the two densities allows one to be easily calculated from the other: for the *FHP7* model

$$\rho = 7d \,. \tag{1.15}$$

The density of particles is directly related to the density of holes, due to the dual nature of the boolean gas,

---

[14]probably even all

[15]Which isn't to say that a display of the microlattice isn't useful — far from it! Tracking an individual particle in its motion and interaction with other particles and barriers, say, can give a good 'feel' for the model and its validity. Properties like the particle density can be easily judged, and similarly the particle flux, though typically in only more extreme cases.

$$\bar{d} = 1 - d, \tag{1.16}$$

and a gas with particle density, d, is indistinguishable macroscopically[16] from one with particle density, $\bar{d}$.

Momentum flux A vector addition of the momenta of all the particles within a cell produces the local momentum flux density,

$$\rho\mathbf{u} = \sum_i N_i \mathbf{c}_i \tag{1.17}$$

This can be achieved by counting the number of particles heading in each of the link directions, and then vector summing those link flux vectors. If only the local flow speed is required, the magnitude u of the flux vector can be easily calculated.

Energy In a model where energy, i.e. c_i^2 (the square of the particle velocities), is conserved independently of mass and momentum, a thermal property will exist:

$$\tfrac{1}{2}\rho u^2 = \tfrac{1}{2}\sum_i N_i \mathbf{c}_i^2 \tag{1.18}$$

This is, however, not the case for the HPP or FHP lattice-gases in 2D, nor the FCHC in 3D.

In lattice-gases consisting solely of single velocity particles, kinetic energy and momentum are degenerate since no separate conservation law exists. Therefore no concept of temperature, affecting the movement ability of particles, is possible. It is not clear that gases with very few discrete velocities fair much better, since the distribution of velocities is necessarily very crude.

Duality As a consequence of the dual nature of the lattice-gas, certain macroscopic observables differ from the theoretical values by a density dependent scaling factor, $g(\rho)$. This has been theoretically calculated, and experimentally verified, for the *FHP7* models to be:

$$g(\rho) = \frac{7}{12}\frac{7 - 2\rho}{7 - \rho} = \frac{7}{12}\frac{1 - 2d}{1 - d}. \tag{1.19}$$

1.3.2.2   Properties

It is possible to derive a number of the physical properties of the lattice-gas from the microdynamic analysis, as was done in [FdHHLPR87] for a variety of lattice-gases. These are summarised for the *FHP7* model in **Table 1.3**, and in **Figure 1.12**.

---

[16] In the absence of exterior influences, like sources.
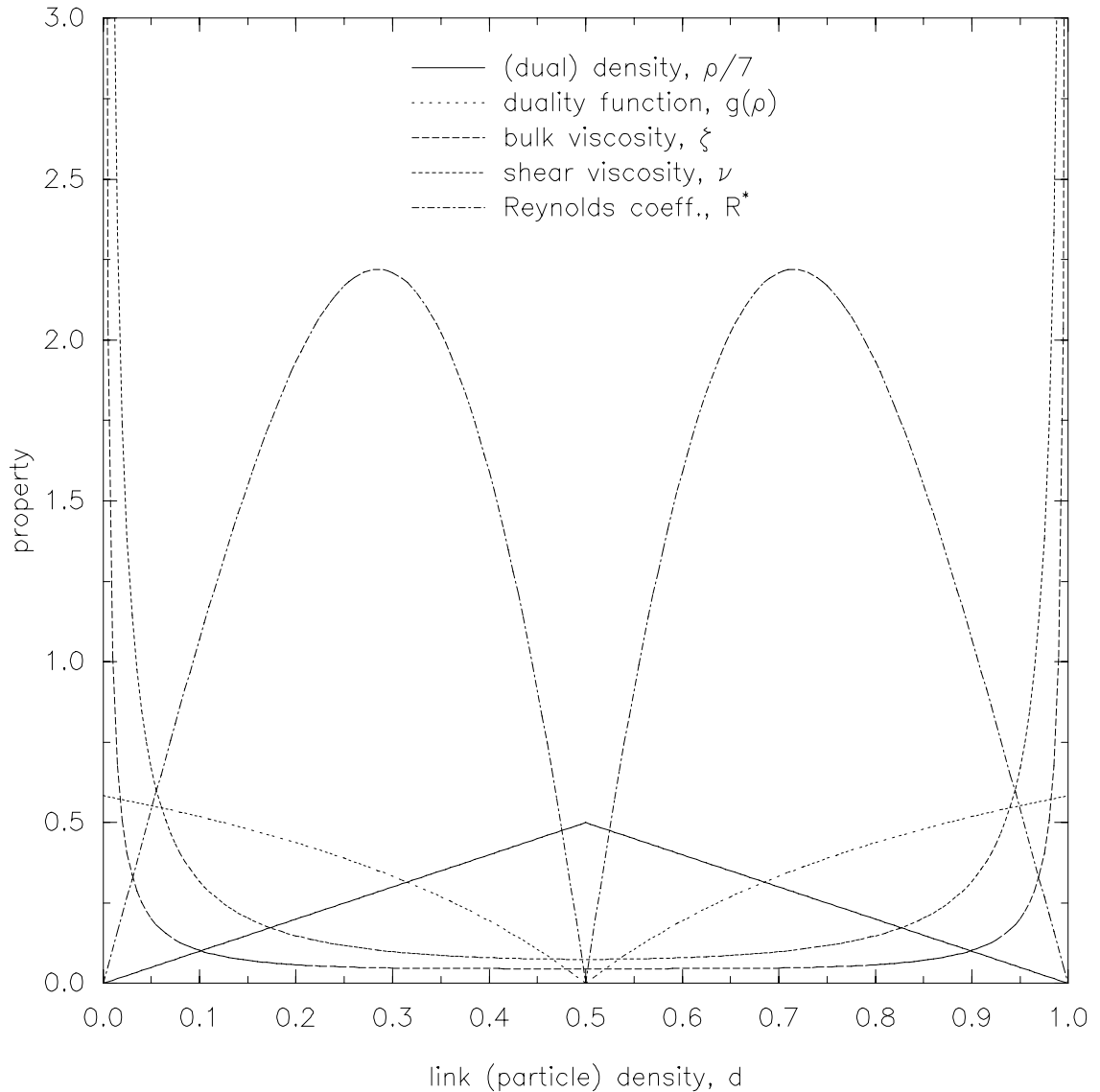
Figure 1.12: FHP7 model transport properties

The *FHP7* lattice-gas model transport properties are plotted to show their density dependence. The reduced (dual) density is that of the least-populated species (particles or holes), and therefore all graphs are symmetric about $d = 0.5$. The viscosities are found to be fairly constant for a wide range of (non-extreme) densities, while the Reynolds coefficient, R^*, is optimised at 2.22 when $d \sim 0.285$.

Viscosity   The kinematic shear viscosity, η, and bulk viscosity, ζ, (also calculated in [RF86]) are:

$$\eta \;=\; \frac{1}{28d\bar{d}(1 - \frac{8}{7}d\bar{d})} - \frac{1}{8}\,, \tag{1.20a}$$

$$\zeta \;=\; \frac{1}{98d\bar{d}(1 - 2d\bar{d})} - \frac{1}{28}\,. \tag{1.20b}$$

from the sums of the propagation and collision viscosity[17] components, ν_p and ν_c. The propagation viscosity is negative as a result of the lattice discreteness. These formulae have been verified in simulations [dHLS85, KMZ87].

Through a careful choice of collision rules it is possible to minimise the viscosity of a lattice-gas [Hénon87b], which is advantageous for increasing the range of flows which can be simulated. Although the dependence of the viscosity on the density is unphysical, in the incompressible limit it doesn't really matter.

Recently, lattice-gas models have been proposed which introduce a new irreversible collision rule which maximises the momentum flux in the direction of the local momentum gradient, yielding a negative shear viscosity [Rothman89]. Used in conjunction with the standard collision rules, arbitrarily low viscosities could be produced. It should be noted, however, that where the dissipation scale of the lattice-gas was less than a lattice link-length, a *grid viscosity* is expected, which limits how low the viscosity can be taken. This is likely, all the same, to provide a valuable technique for attaining high Reynolds numbers, e.g. for turbulence simulations.

Speed of sound   The analysis of the lattice-gas microdynamics (at least in the incompressible limit) shows that the speed of sound is a constant for the *FHP7* model,

$$c_s = \sqrt{\tfrac{3}{7}}\,, \tag{1.21}$$

however, the incorporation of additional rest particles (such that a number can reside at each node) can be used to reduce the speed of sound [FdHHLPR87].

Mach number   The dimensionless ratio of a velocity, u, to the speed of sound is known as the Mach number,

$$\mathrm{Ma} = \frac{u}{c_s}\,, \tag{1.22}$$

which determines the influence of compressibility effects. When $\mathrm{Ma} \ll 1$, these are negligible, and the flow is considered incompressible.

---

[17]The kinematic viscosities, previously denoted ν, are used instead of the true viscosities, e.g. η, for convenience and are often used interchangeably.

Pressure   A similar analysis of the pressure of the lattice-gas produces the formula

$$p = \tfrac{3}{7}\rho \left(1 - \tfrac{5}{6}g(\rho)u^2 \right) . \tag{1.23}$$

It can be seen that this follows the expected isothermal relation

$$p = c_s{}^2 \rho , \tag{1.24}$$

and results in the pressure generally being twice the density, unless the velocity gets quite high. When the velocity approaches the maximum possible velocity, the model breaks down, and the formula gives unrealistic negative pressures. Pressures are shown in **Figure 1.13** for a range of velocities.

Reynolds coefficient   The dimensionless Reynolds number, governing the relative importance of the viscous and inertial properties of the flow, can be stated

$$\mathrm{Re} = g(\rho)\frac{LU}{\nu(\rho)} = \frac{g(\rho)\,c_s}{\nu(\rho)}L\,\mathrm{Ma} \tag{1.25}$$

where L and U are, respectively, a characteristic length and a characteristic velocity, and a Reynolds coefficient, containing the model-specific terms, can be defined,

$$R^*(\rho) = \frac{c_s\,g(\rho)}{\nu(\rho)} . \tag{1.26}$$

It is found that this coefficient reaches a maximum value of 2.22 for the *FHP7* lattice-gas when the density is 0.285 .

The two dimensionless numbers, Re and Ma, characterise isothermal flows. When the Reynolds number is small ($\mathrm{Re} < 1$), the inertial component is dominant and a laminar, stationary flow is found, compared to the non-stationary and turbulent flows found when the viscous component is dominant and the Reynolds number is large.
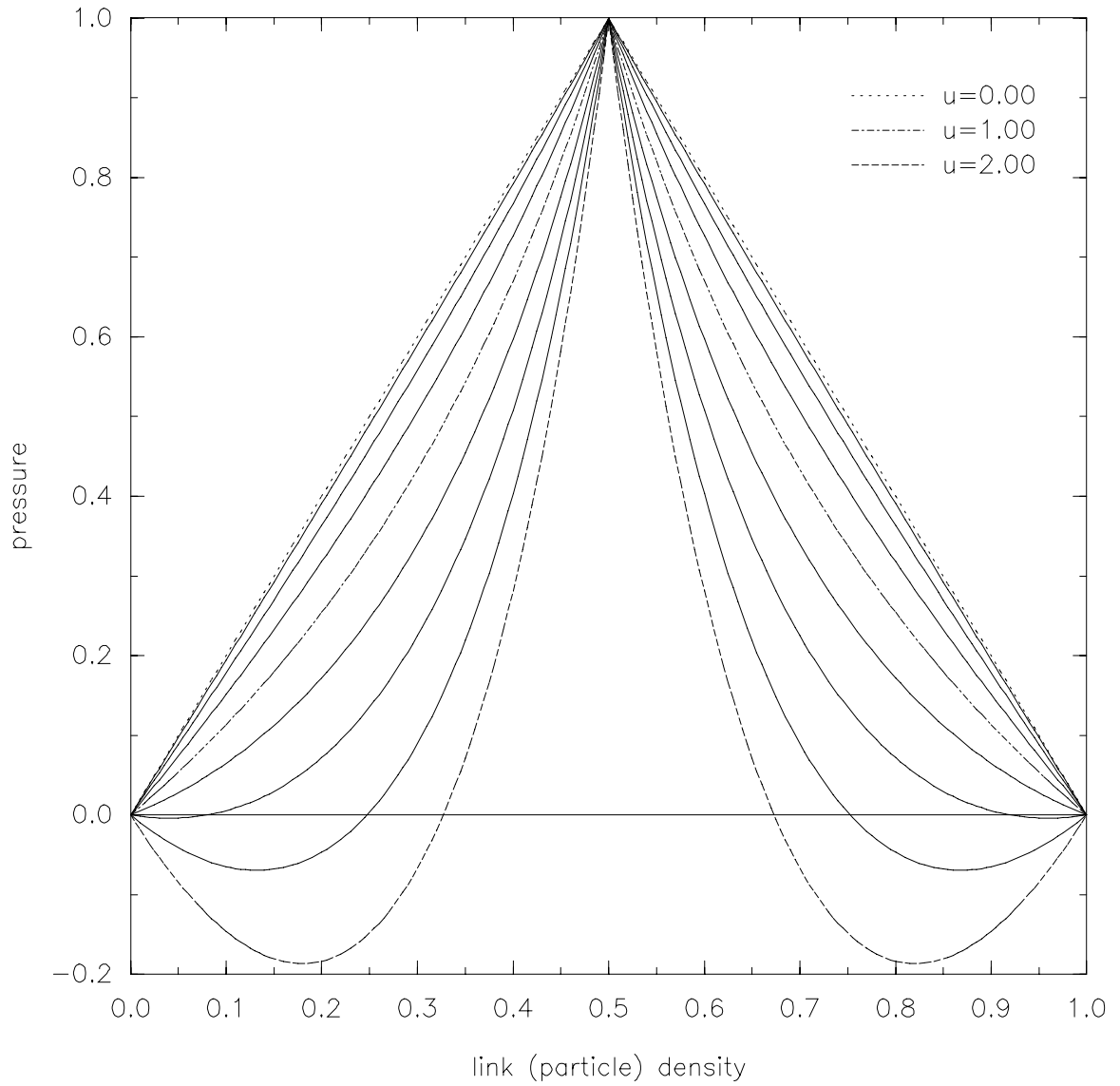
Figure 1.13: FHP7 model pressure variation

The pressure equation of the *FHP7* model is plotted to show the dependence on density and for the full range of velocities: $u = 0.00..2.00$ in increments of 0.25 . Negative pressures are seen to be possible when the velocity exceeds $u \simeq 1.50$.

Model	D	lattice	links	configurations	collisions	R^*_{max}
$HPP^\dagger$	2	square	4	16	2	?
$FHP6^\ddagger$	2	hexagonal	6	64	20	$O(0.5)$
$FHP7^\ddagger$	2	hexagonal	7	128	72	2.22
$FCHC^\S$	3	hypercubic	24	16,777,216	$O(10^7)$	$O(40)$

† original Hardy, de Pazzis & Pomeau model
‡ Frisch, Hasslacher & Pomeau model, with possible rest particle
§ Face-Centred HyperCubic model, requiring a single section of a 4-dimensional lattice

Table 1.2: Basic lattice-gas models' diversity
Comparisons of the number of possible configurations, and those considered collision configurations. D is the dimensionality of the resulting lattice-gas (rather than the lattice on which it resides), and R^*_{max} is the maximal Reynolds coefficient obtainable. Only the HPP model is necessarily deterministic, though the others can be made so. The addition of extra rest particles, to the FCHC model in particular, results in many more possible configurations and collisions, and also a higher R^*.

density	$\rho = 7d$
speed of sound	$c_s = \sqrt{\frac{3}{7}}$
duality function	$g(\rho) = \frac{7}{12}\frac{1-2d}{1-d}$
pressure	$p = 3d\left(1 - \frac{5}{6}g(\rho)u^2\right)$
shear viscosity	$\eta = \frac{1}{28d\bar{d}(1-\frac{8}{7}d\bar{d})} - \frac{1}{8}$
bulk viscosity	$\zeta = \frac{1}{98d\bar{d}(1-2d\bar{d})} - \frac{1}{28}$

Table 1.3: FHP7 transport properties
Summary of the properties of the hexagonal lattice-gas, with a single rest particle, and the fully saturated collision set.

1.4   Summary

A simple comparison of the diversity of the collision sets in a few of the basic lattice-gas models is shown in **Table 1.2**. The HPP model is generally accepted to be lacking in sufficient richness to fully produce all but the most basic hydrodynamics. The variety of the $FCHC$ model, and its ability to simulate three-dimensional hydrodynamics makes it a very promising model for the future, but also puts it beyond our current capabilities.

Of the remaining FHP models, it was the $FHP7$ variant, which was selected as optimal for the basis of the following two-dimensional research, due to its superior properties for only marginally increased complexity. The implementation of the model will be detailed in the following chapter, and investigated in later chapters. Its properties are summarised in **Table 1.3**.

2 Multicomputer implementation & visualisation

Contents

As a rule,
software systems do not work well
until they have been used,
and have failed repeatedly,
in real applications.

— [Parnas90]

Since parallel programming tools exist,
MIMD machines are easy to use.

— [vanSchalkwijk90]

2.1   Introduction

C ELLULAR AUTOMATON lattice-gas models are inherently parallel, by virtue of their large regular lattice, and the uniformity of its update. A multicomputer, capable of harnessing the computational power of a number of relatively inexpensive processors, can bring considerably more muscle to bear on the problem than would otherwise be possible.

The novel computer architecture, and the issues and problems involved in utilising it efficiently are addressed with respect to the intended lattice-gas simulations. These factors govern the high-level implementation of the model, along with more usual implementation strategies for low-level realisation.

Another vital component for the investigation of hydrodynamics is the (potentially real-time) visualisation of the resulting flows, such that their detailed developing behaviour can be scrutinised and qualitatively verified, prior to subsequent quantitative analysis. The simple visualisation techniques implemented for the simulations are described.

2.2   Distributed parallelism

2.2.1   Concepts of concurrency

2.2.1.1   Computer architectures

Various means can be employed to attempt to improve the performance, or throughput, of a computer. The simplest is obviously to design faster hardware, capable of executing more instructions per second. This avenue has been thoroughly exploited, however, and there are fundamental limits (such as the speed of light which governs the speed of propagation of information) which suggest that performance improvements are unlikely to satisfy computational demand by this method alone. This has lead to a growth of interest in alternative computer architectures and methodologies, which have been proven to offer greatly improved performance through exploitation of concurrency — albeit, not for every program.

The taxonomy of computer architectures by [Flynn72] is summarised in **Table 2.1**. A full survey of the state of the parallel computer industry as it exists at the start of the 90's can be found in [TW91], while [HJ88] provides a more detailed look at the primary architectures covered.

SISD   Traditional computers consist of a single processor executing instructions one after the other on its data, in the conventional von Neumann architecture. Such serial computers are termed SISD machines — at any time they are capable of executing a single instruction on a single data element.

	Single Instruction	Multiple Instruction
Single Data	SISD	MISD
	'ordinary' serial computers[†]	pipelined computers
Multiple Data	SIMD	MIMD
	array processors	multiprocessor computers[‡]

[†] Current machines generally incorporate a number of transparent optimisations which may include small-scale parallelism.

[‡] Covers a wide range of shared- and distributed-memory architectures and small- and large-scale parallelism.

Table 2.1: Computer architecture classifications

The common classification of parallel computers by their ability to execute single or multiple instructions simultaneously on single or multiple data items. Hybrid architectures utilising techniques from distinct groups have also been developed, however, the categorisation is still generally useful.

MISD To optimise the throughput of a sequence of instructions, a *pipeline* of processing units can be designed such that intermediate results need not be stored, but can be forwarded to the next unit to be processed further. Various forms of pipelining can produce considerable performance gains, but since programs branch regularly and pipelines have a startup overhead as instructions are loaded, their usefulness is restricted.

SIMD One approach to faster execution through parallelism is to execute each instruction on multiple data elements, and is termed SIMD. The Distributed Array Processor (DAP) [BBKPW87], with 4096 (i.e. 2^{12}) processing elements (PEs), and Connection Machine (CM) [Hillis85], with 65536 (i.e. 2^{16}) PEs, are examples of computers with thousands of simple processors capable of executing instructions simultaneously on large data arrays. These are massively parallel, fine-grained, array processors, and are updated in lock-step at the direction of a master control processor.

This works very well when the bulk of the computation requires to be done on a typically large amount of data; however, this is not always the case, and when a program branches only some of the data requires to be updated by the current instructions, and the overall performance drops.

MIMD When multiple processors each execute multiple (different) instructions on their own distinct data — as is the case with a MIMD computer — they can collectively be very powerful, provided that the problem can be suitably divided into parts which can be executed simultaneously. In general, medium-grained problems can be efficiently solved on MIMD computers, such that processors require relatively little synchronisation with each other.

True multiprocessors, physically complete and independent computers in their own right, or shared-resource multiprocessors, consisting of skeleton processing elements sharing system resources, have both been employed in MIMD architectures. The former case is known as a *multicomputer*.

2.2.1.2 MIMD architectures

Shared memory Enabling multiple processors to work cooperatively on a single calculation involves an overhead, since data integrity is only retained when coordination between processors ensures that they do not interfere with each others data. With shared memory each processor can access all the data, but data requests must be mediated through the use of semaphores and locks which result in *contention*.

Transparent use of a small number of very powerful vector processors, as employed by the dominant supercomputing vehicle, the Cray-YMP series, elegantly achieves world-leading performance in a variety of problems. However, such techniques are fundamentally limited by increasing contention problems (only partially alleviated by extra caching and pre-fetching) as additional processors are added.

Novel machines like the BBN Butterfly, have pushed this architecture to its limits (approximately 64 to 128 processors) at the expense of sophisticated switching hardware, with an extremely complex crossbar switch capable of routing data requests between any two processors. Alternatively, processors can compete for access to a bus connecting all the processors and their data, but again this is limited to a realistic maximum of around 32 processors, as is the case with the Sequent Symmetry.

Distributed memory To achieve *scalable* performance, such that each additional processor contributes fully to the computation rather than an extra burden to the switching or bus resource, distributed memory is required. In this way, a modular and expandable compute resource can be constructed, tailored to the needs of the problem. This has been successfully employed with computers such as the Intel iPSC series (with 128 nodes) and the NCUBE (with 8192) nodes. The fixed topology hypercube architecture of these machines allows simple and efficient hardware message through-routers to be designed.

The Meiko Computing Surface [Meiko87], on the other hand, was designed to allow multiple transputers[1] to be connected into the optimal processor configuration for the problem. Other manufacturers such as Parsys and Parsytec provide similar products. A communications *harness*, either custom designed or general purpose such as Tiny [Clarke90] is required to allow interchange of data between processors via *message-passing*.

---

[1]or other processors, such as i860s, SPARCs, etc

2.2.2 Multicomputer architecture

The computer selected for the lattice-gas investigations, was a multiprocessor computer, or *multicomputer*, consisting of Inmos transputers mounted on boards within a Meiko Computing Surface. Such a MIMD computer has a number of transputers each capable of working on part of the computation independently of the others, while cooperatively ensuring that system integrity is retained.

The use of many identical and relatively inexpensive boards consisting of cheap, mass-produced microprocessors, provides a cost-effective alternative to traditional serial computers for certain types of computation. For certain classes of application, which are efficiently parallelisable, a sufficient[2] number of transputers can provide near supercomputer performance for a fraction of the cost.

2.2.2.1 Why a multiprocessor computer ?

A multiprocessor computer offers the unique ability to scale the amount of compute resource to the problem at hand, ensuring that it is being utilised efficiently, and therefore cost-effectively. The provision of additional services, such as graphics or fast disk access, can also be done when required, enabling the best usage of the available resources.

The Edinburgh Concurrent Supercomputer (ECS) provides several domains, with a mix of compute and additional resources, so that an application can be targeted at an appropriate size of domain, or can progress through domains of increasing sizes as more compute resource is required — and, importantly, as the efficiency of utilisation is demonstrated. It should be possible in the near future to allocate exactly the resource desired by an application in a dynamic context, however, currently the allocation of resource into domains is done statically, with each domain allocated to applications dynamically.

The lattice-gas simulations were able to efficiently use a range of the available resources. For large, high-resolution, simulations, the considerable resource of a domain of 128 transputers were required for a day to take a system from its initial starting state to a fully developed flow. However, the same simulation could have similarly been done on a domain of 16 transputers, and would have taken a week to cover the same time-scale.

Smaller simulations, which were unable to utilise large domains effectively, could be efficiently carried out on the smaller domains. Likewise, the interactive analysis of the results of a large simulation were done on a smaller domain with graphics resource.

---

[2]The number will depend on several factors, including the nature of the scaling of the application with processor number, the compute power of the particular processors used, and, of course, the current metric for classification as a supercomputer at the time of the comparison.

2.2.2.2 The transputer

Transputers[3] are high performance multitasking 32-bit CMOS[4] microprocessors. Each chip has an integrated ALU[5], 4 bi-directional serial links and 2kbytes of static RAM memory, as well as a bus servicing all the components. In the case of the T800 series chip, a floating-point unit (FPU) has also been incorporated, capable of achieving 1Mflops — i.e. one million floating point operations per second. All components can potentially operate simultaneously, which importantly allows all four links to be handling communications events while the ALU (and FPU on the T800) are independently processing data [Inmos88b].

The provision of communications services within the chip facilitates the ease with which a number of transputers can be harnessed to work independently on parts of a particular problem, but cooperate jointly on solving a larger problem, since information can be efficiently passed to adjacent processors or through-routed to remote processors. This relies on the transputers ability to automatically deschedule and swap processes, e.g. when awaiting communication.

2.2.2.3 The *occam* model

The transputer, and its native language *occam*, support the communicating process model of concurrency (see e.g., [Hoare85]).

Communication A process consists of a complete computation, with its own unique data space, which interacts with other processes via channels. Channels are point-to-point communication paths, which allow a particular process to send data to another and receive data from it; although they are capable of being used for communication in both directions, for consistency, they should only be used in one sense. Communication is synchronised, such that when a communication event is requested on a channel, the process blocks and can only continue when that event occurs, i.e. the process on the other end of the channel is also ready for the transfer to proceed. An abstraction is attempted between communication of data on channels between processes within a processor, and on channels (over the physical links) to processes on adjacent processors.

Processes The process model is hierarchical, so that a process can be one of a number forming a larger more complex process, which may have internal (shared between local processes) as well as external (shared with other processes) channels and data structures. In reverse, a large process can be suitably decomposed into

---

[3]Taking their name from *trans*istor and com*puter*, designed to be replicated components, yet fully-capable computers in their own right.

[4]Complementary Metal-Oxide Semiconductor

[5]Arithmetic and Logic Unit

a number of smaller (and simpler) processes. The relationship between the sub-processes must be specified as either sequential or parallel. If *sequential*, that process must finish before the following will commence. If *parallel*, all will run simultaneously and they must all finish before following code will be executed.

Processes naturally run simultaneously on separate processors, yet could just as simply execute in parallel on a single processor — though generally at a fraction of the speed, since they would then be sharing the computational resource. The channel abstraction allows the distinction between a remote process executing on a separate processor, or on the same processor, to be unnecessary.

2.2.3 Problem decomposition

Decomposing a problem for a MIMD computer can be done in several ways, and indeed a problem may utilise different decompositions in different sections of the problem, though typically not all will be suitable for any given problem. (see, e.g., [FJLOSW88])

2.2.3.1 Algebraic decomposition

Perhaps the most natural decomposition is to split the problem into a number of distinct operations, and to execute these on different processors passing the partially processed data between them. Unfortunately, this method is rarely useful, since the different operations typically do not happen at the same time, or else make load-balancing extremely difficult since they take different and varying times to execute. There can also be excessively large amounts of data to pass from one processor to another, which is very difficult to overlap with useful computation, and results in a *bottleneck* where processors idle waiting for the data they require to be made available.

2.2.3.2 Data decomposition

Where the same sequence of operations must be applied to a range of data elements, independently of neighbouring elements, advantage can be made of *event parallelism*. Typically, a large amount of work can be decomposed into a number of smaller tasks, and these can be distributed over the available processors, with the resulting partial solutions combined when all have completed to produce the final solution. A *task farm* operates in such a manner.

An example of such a computation would be the calculation of the number of matches for a particular key within a large database, where each processor can be given a section of the database, can match the key against the entries within its section, and return the matches to a coordinating processor.

There will be an overhead involved in initially distributing the database, but this would typically only require to be done once, and then for each interrogation the key has to be broadcast, and the resulting matches collected. However, provided that each processor has a sufficiently large section of the database, the work involved in processing on that section will be in excess of the distribution overheads, and a suitable performance gain should be observable as more processors are added while this remains true.

2.2.3.3   Geometric decomposition

Where each of the decomposed sections of the problem require to interact with one another, it is important that the overheads involved in doing so are minimised, through provision of an efficient message routing mechanism and careful consideration of the partitioning so that inter-processor communication is minimised.

In a grid-based problem, each point within the grid will rarely (if ever) require information from remote regions of the grid, but will typically interact closely with adjacent points, or those within its immediate locale. A suitable decomposition will ensure that as much as possible of each points neighbourhood is located within the same processor, or failing that, on processors which are directly (physically) connected to it.

Each grid update will therefore typically involve the computation of the interactions within the local section of the grid, and additionally the communication of data necessary for the update of the border neighbourhood within each processor (and equivalently, the provision of that information for each neighbour).

In the simplest case, since transputers with four links have been used, each processor can be directly connected to four neighbours, and a two-dimensional decomposition of the lattice will be most optimal in minimising the section surface to area. This corresponds to minimising the communication of border information to the calculation of interior information.

On a multitasking microprocessor, like the transputer, advantage can also be made of the ability to communicate border information while simultaneously updating the interior, before finally merging the border information with that of the interior to complete the update. Therefore, if the time taken to communicate the border is less than that to update the interior it will be completely overlapped and hence transparent (and immaterial) in efficiency considerations.

2.2.4   Problems with parallelism

2.2.4.1   Communication

Although the transputer has fast links to enable it to transmit data quickly between processors, communication hardware technology is considerably lagging be-

hind computational hardware, and the transputer is no exception — though it is far better balanced than most. Furthermore, this is likely to be the case for (at least) a considerable time, and therefore it is important to recognise that communication can be just as important (and more so in some cases) than the computation involved in a program.

The transputer can overlap the operation of the links with that of calculations in its processor, and when done successfully, it is possible for the communication time to be completely transparent. However, as a problem is partitioned into smaller and smaller parts distributed over larger numbers of processors, the communication can tend to rise with the number of processors, and even where it doesn't, the communication overhead rises since the computation time for the smaller data segments will decrease.

2.2.4.2 Load imbalance

Inhomogeneity It is also important to note that in the event of an inhomogeneity in the update characteristics, for instance in the lattice-gas, where sites requiring special update are particularly (or uniquely) prevalent, the load distribution problem requires more critical examination. When a particular processor is *light* in work — such that it will finish processing before the other processors — the load-imbalance can be significant, but the overall processing efficiency will not be significantly degraded. The contrary case, where a particular processor has a greater workload than the rest, i.e. a *heavy* load, it will finish processing later than all the others, and this is extremely undesirable, and to be minimised as much as possible.

In the first case, one lightweight processor doesn't significantly degrade the overall processing efficiency, since the majority of the processors are themselves working at full efficiency. With a single heavyweight processor, overall processing efficiency is massively degraded since only a single processor is working at full efficiency.

These considerations become much more serious when additional processors are available, since the overall throughput expected by their provision will be catastrophically degraded by any initial processing inefficiency resulting from heavyweight processes.

Fragmentation Fine subdivision of a large problem, such that each processor has a number of smaller tasks (rather than just one), sometimes can increase efficiency by achieving better task overlapping — the processing of one task can be done which communicating the results of another. Generally, if the workload profile is unknown, a random decomposition and scatter is necessary to achieve a reasonable balance. Since the connectivity of the lattice-gas simulation is high (requiring a significant data transfer rate), and the likely imbalance small, fragmentation is unlikely to prove beneficial in this instance.

2.3 Implementation of model

A complete summary of the essential core of the *occam2* implementation is included in **Appendix F**. A number of books (e.g. [BKPR87, PM88, Wexler89, Inmos88a]) provide suitable references on details of the *occam* language, though none of the concurrency specific features are used in these fragments.

2.3.1 Data structures

2.3.1.1 Sites

The two operations at the heart of a lattice-gas simulation are propagation of particles between lattice sites and their collision update when there. The models investigated have been 6-bit and 7-bit models, so the information required for a particular lattice site can conveniently be stored in a single (8-bit) byte, with the spare bit used to signify special sites (such as barriers and sources) when set. Thus, a particle moving in the n-direction of the lattice would be represented by the presence of the n'th bit being set in that particular site byte. The special site and link codes are shown in **Code Fragment 1**, and the particle configuration codings in **Appendix C**.

2.3.1.2 Lattice

Since a two-dimensional lattice is required by the model, it is stored as a two-dimensional byte array, containing all the information about the particles and the special sites. A trigonal lattice is required such that each node has six (equidistant) neighbours, as shown in **Figure 2.1**.

This can be implemented straightforwardly, using a particular site's index parity in one direction (e.g. y, say) to suitably specify its neighbourhood. For even sites, this provides north-east and south-east neighbours in addition to the four prime compass directions; odd sites have additional north-west and south-west neighbours. Instead of a hexagonal cell update template (or stencil), two chevron-shaped templates are required — the fact that the distance to diagonal neighbours is longer than that to nearest neighbours and that they are not equiangular are obscured in the coding (see **Code Fragment 9**).

The update requires some temporary storage for particles as they propagate to neighbouring sites, and it was simplest to specify two complete copies of the lattice. This obviously reduces (by almost a factor of two) the largest lattice which can be simulated, but was not a problem since the transputers used had 4Mbytes of memory, and this was never all used. Indeed, the computational effort involved in updating even the largest lattice (segment) possible on a single processor, required tens of seconds, and such a lattice would require many such updates to equilibrate.
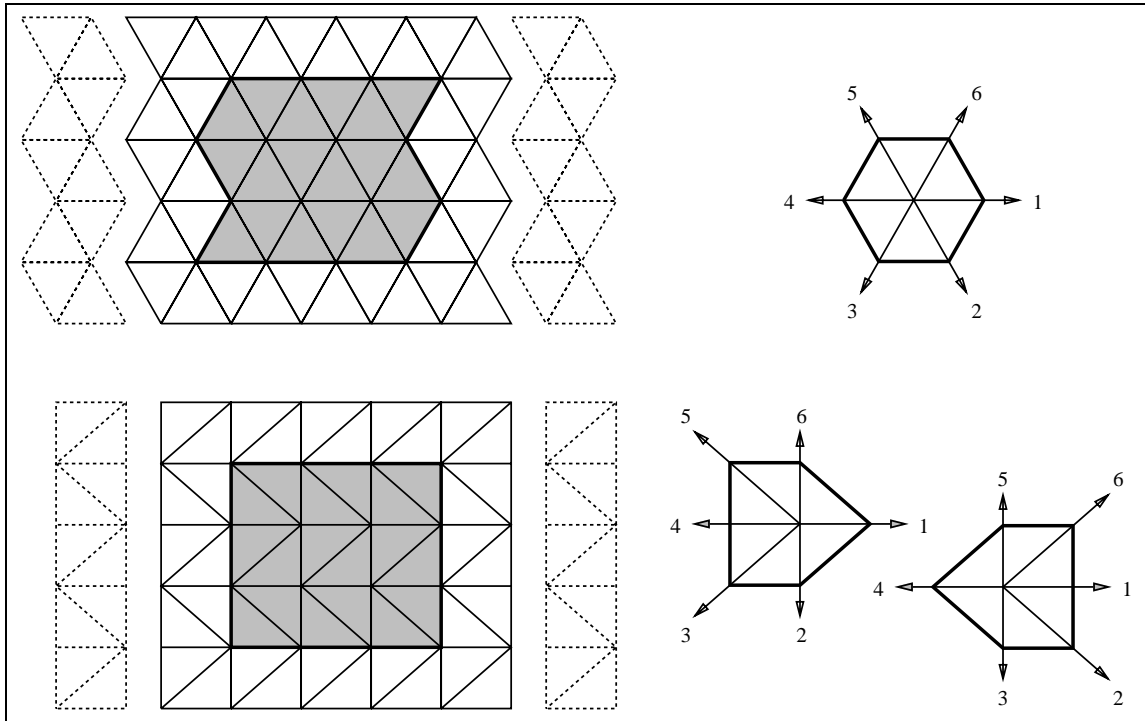
Figure 2.1: Hexagonal lattice storage

The regular hexagonal neighbourhood of the triangular lattice, with six equally separated connections from each node to its neighbours, can be viewed (and more easily stored) as the equivalent rectangular lattice with an additional diagonal connection. Alternating the diagonal connections facilitates the regular partitioning of the lattice, as well as providing a more compact averaging cell — nodes within the shaded area constitute a single (4×4) cell.

2.3.2   Site update

The update algorithm, as detailed in **Code Fragment 9**, consists of propagation of the particles in the configuration at each site after they have collided or been generated, should the site have been a source site. Barrier sites (and sinks, which are exceptional source sites) themselves require no update, but affect the propagation of particles from neighbouring sites.

2.3.2.1   Generation

Source sites absorb all incoming particles, and on update would produce particles, with a given probability dependent on their strength, for propagation. Configuration generation for sources is simply and quickly achieved by selecting a pseudo-random integer and accepting particles in each of the link directions with a probability according to the source density[6] (see **Code Fragment 8**). Each random number is used to generate several particles, by considering different parts of its bit

---

[6] which has been stored as a percentage in the source code

Type	FHP6	#	FHP7	#
Unchanged	$\mathbf{E,F,V,J,W},\tilde{\mathbf{J}},\tilde{\mathbf{V}},\tilde{\mathbf{F}},\tilde{\mathbf{E}}$	44	$\mathbf{E},\mathbf{E_{\bullet}},\mathbf{F,V},\mathbf{V_{\bullet}},\mathbf{W},\tilde{\mathbf{W}}_{\bullet},\tilde{\mathbf{V}},\tilde{\mathbf{V}}_{\bullet},\tilde{\mathbf{F}}_{\bullet},\tilde{\mathbf{E}},\tilde{\mathbf{E}}_{\bullet}$	52
Deterministic	$\mathbf{L,Y}$	14	$\mathbf{F_{\bullet}},\mathbf{J},\tilde{\mathbf{J}}_{\bullet},\tilde{\mathbf{F}}$	24
Non-deterministic	$\mathbf{I},\tilde{\mathbf{I}}$	6	$\mathbf{J_{\bullet}},\mathbf{L,I},\mathbf{I_{\bullet}},\mathbf{Y},\tilde{\mathbf{Y}}_{\bullet},\tilde{\mathbf{I}},\tilde{\mathbf{I}}_{\bullet},\tilde{\mathbf{L}}_{\bullet},\tilde{\mathbf{J}}$	52

Table 2.2: Particle configuration update categories
Configuration classifications for the *FHP6* and *FHP7* models.

pattern, to provide a fast, though obviously poor-quality generator.

2.3.2.2 Propagation

Within the storage scheme, particle propagation involves removing the relevant particle from its original site byte, and adding it to the appropriate destination site byte of its neighbours. Because the model incorporates an exclusion principle, the site receiving a moving particle is guaranteed to have that bit free for it.

More complex decisions have to be made if the destination was a barrier site, so that the particle is reflected and re-incorporated in its original site (though in a different particle orientation). Equally, those particles with source site destinations are simply removed.

Bitwise masking operations on the bytes containing the particles are unfortunately rather slow on a transputer, and it turns out to be faster to promote the byte configuration to a (word-length) integer while doing the movement (see **Code Fragment 7**). Logical tests determine the presence of particles, combined with arithmetic additions and subtractions[7] to remove particles from the input configuration and add them to the appropriate output configuration. Any decisions to be made when a neighbour cannot accept a particle, because it must be reflected or absorbed, are also easily undertaken in this stage.

2.3.2.3 Collision

The 128 (or 64) possible configurations of particles coded into the bits at a particular site can be quickly collided using a table look-up. The collision update categories for each of the particle configuration classifications are shown in **Table 2.2**, and the codes in **Code Fragment 2**. **Code Fragment 6** shows the collision update routine, where each site's configuration code is used as an index into the `collision` table (shown in **Code Fragment 3**) to get the configuration collision type code.

The majority of sites are not considered collision configurations and therefore are unchanged; they are assigned the `invariant` code[8]. The remainder are either coded

---

[7] in the faster, unchecked modulo arithmetic

[8] using `font` for program constant names

to have a `deterministic` collision channel, or a non-deterministic update from a number of possibilities, `ND?.particle.?`.

The `deterministic` configurations can read their output configuration directly by indexing the `collider` table in **Code Fragment 3**, but for the non-deterministic updates an entry has to be selected from the appropriate `ND?.collider.?` table (shown in **Code Fragment 4**) corresponding to the number of particles in the configuration and their net momentum, which must be conserved.

Details of the mechanism for selecting an entry from one of the non-deterministic update tables are unlikely to be important, at least insofar as anomalous asymmetries are avoided. A fully random selection, ensuring that the updated configuration is distinct from the original if possible, is implemented in **Code Fragment 5**.

2.3.2.4 Update cost

The update cost for a particular site is variable, but can generally be estimated according to the characteristics of the simulation. For a barrier site this is effectively nil (though it may have a complex effect on its neighbouring sites). Other sites will have propagation and collision update components. A summary of configuration update times is shown in **Figure 2.2**.

Propagation cost   Each particle in the lattice has to be moved, therefore the cost of propagation is proportional to the number of particles, which will generally depend on the mean density of particles. Although a dual-simulation with every particle replaced with an empty space and vice-versa is completely equivalent in its behaviour, its update will be slower because of the increased number of particles to propagate[9].

Collision cost   The cost of the collision lookup depends on the mix of configurations representative of the lattice at a particular time. The mean density of particles will govern the likelihood of a particular site having a specific number of particles and coupled with knowledge of the proportion of configurations with that number of particles and the cost for each, an estimate of the average cost could be made.

Since all but the non-deterministic type of collision can be updated for the cost of a table-lookup, and even those could be reduced to a single lookup, it is unlikely that any noticeable variation in update cost would be discernible over the propagation cost.

Generation cost   Since sources require to generate configurations, which can involve pseudo-random number generation, they have a generation cost in addition

---

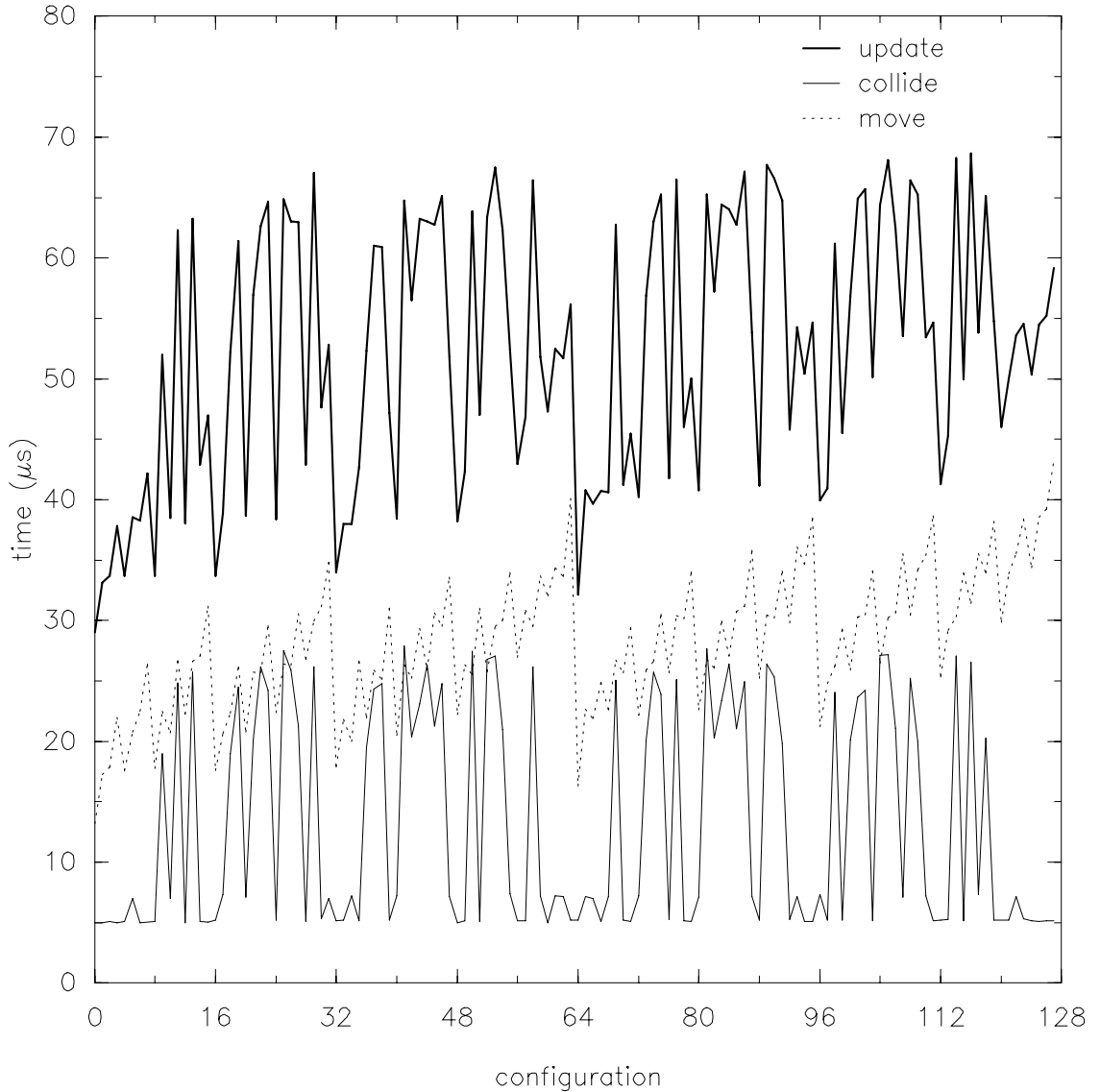[9]assuming the density was less than 0.5 initially

Figure 2.2: Configuration update times

For each possible particle configuration, the total update time, and the collision and propagation components are shown. Invariant collisions take $5\mu s$, and deterministic collisions $7\mu s$, while the non-deterministic collision times range from $18..27\mu s$. Movement is seen to take just over $4\mu s$ per particle on a base of roughly $14\mu s$, with timings based on non-special neighbouring sites — these would be faster, but less representative. Each update also has a constant overhead of $10\mu s$ to set up the site neighbourhood (which is the update time for wall and sink sites).

Mean update time, for an even mix of all configurations, is $52 \pm 11\mu s$. For source sites, a generation time of $45..52$ (dependent on strength) replaces the collision component, and the propagation time is variable (dependent on the resulting configuration). By comparison, the *achieved* update rate in the multiprocessor implementation (i.e. including communication) is $58\mu s$.
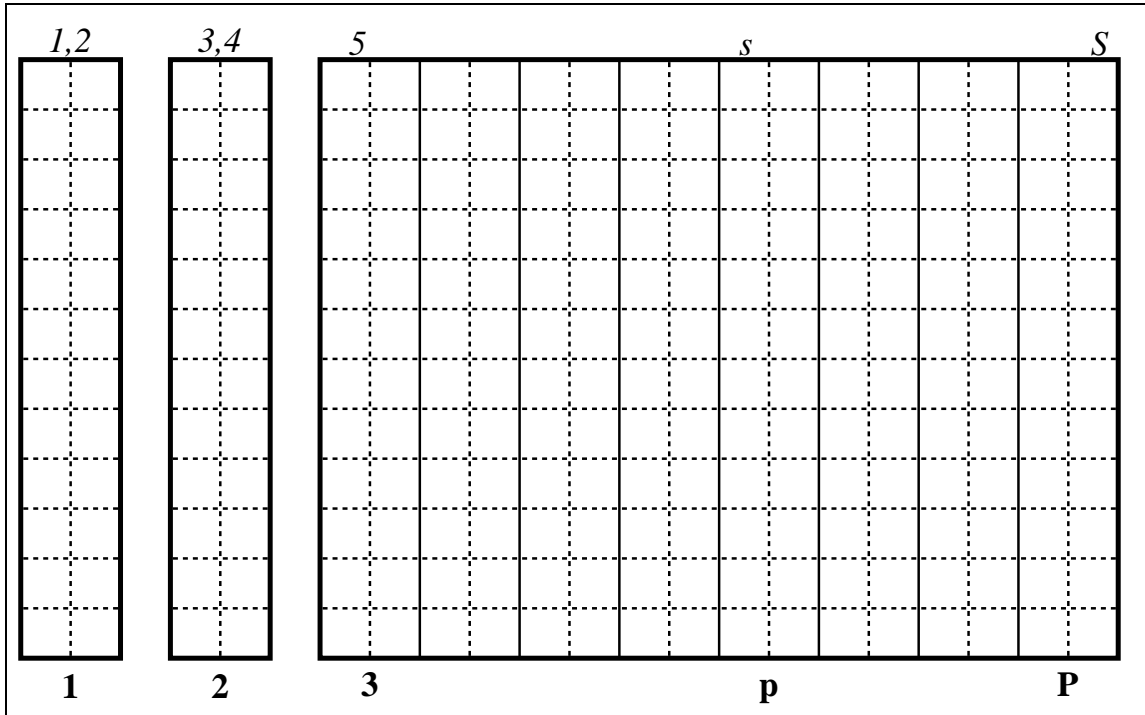
Figure 2.3: Lattice partition
An example lattice of 20×12 cells is partitioned in one dimension and distributed between 10 processors, each receiving two segments.

to a propagation cost (but avoid a collision cost). Large areas consisting of sources should be carefully investigated to determine whether they noticeably overload a particular processor, potentially undermining overall throughput. The random numbers can be arbitrarily poor, however, to minimise their cost.

2.3.3 Distribution of work

2.3.3.1 Partition

The lattice containing the particles, as well as the special sites, is typically very large, e.g. several million sites. This makes it rather straightforward to divide it equally amongst the available processors, to ensure that they all have equal amounts of work. Because of the restriction on complete averaging cells residing within a processor, for maximum efficiency it should be ensured that the number of cells is a multiple of the number of processors. In practice, this is not an important consideration, unless the number of processors is very large, i.e. comparable to the number of cells.

Since a one-dimensional decomposition was found to be satisfactory, on the basis of simplicity and efficiency, it is natural to configure the available processors into a chain, and divide the lattice into columns of cells, or *segments*. Each processor in the chain, p, is therefore responsible for updating its corresponding section of the
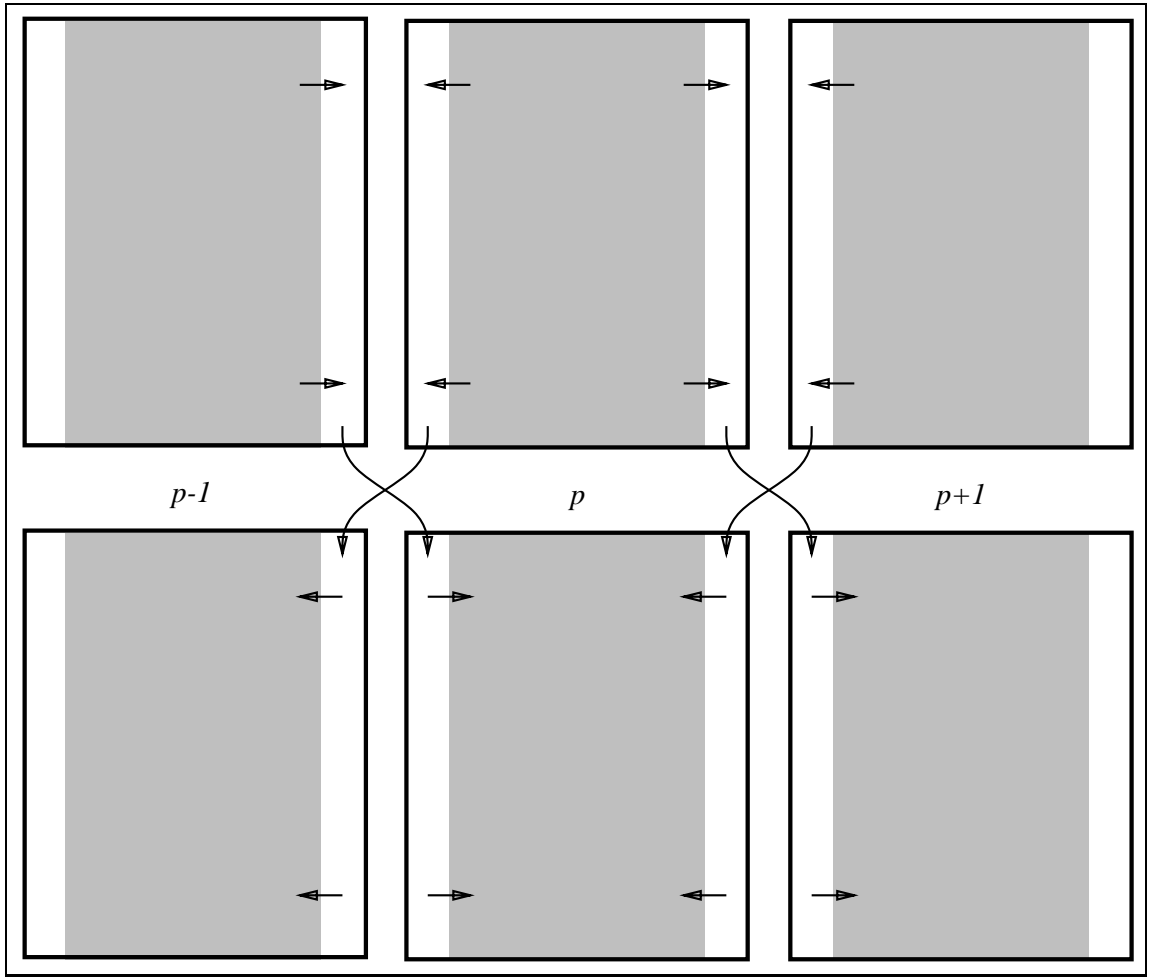
47

Figure 2.4: Boundary swapping
Each processor updates its local lattice (shaded) with particles in sites on the edge able to move into buffer regions (white), which are subsequently swapped with their neighbours, and the particles contained therein merged with the existing ones in the local lattice.

lattice, which may consist of several segments, s. (**Figure 2.3**)

2.3.3.2 Borders

Since the border sites of each segment require to be updated in conjunction with their neighbours, who happen to be on an adjacent processor, some transfer of data is also necessary. The up-chain border of processor p must be exchanged with the down-chain border of its predecessor in the chain, $p-1$, and similarly its down-chain lattice border swapped with the up-chain border of its successor, $p+1$ (see **Figure 2.4**) every update.

This simple view of the partitioning is complicated slightly when the fact that the underlying lattice has a hexagonal symmetry is considered. However, if it is assumed that the number of lattice sites within the edge of an averaging cell is even, then is it possible to ensure that the phases of the interlock between segments

match, and the border is reduced to a simple line of sites, with the phase consistent along its length (see **Figure 2.1**).

Figure 2.4 shows the required buffer-zone, into which particles in sites on the edge of the local segment will move on update. These edge buffers are then swapped with each neighbour, and the particles therein incorporated into the segment local to each processor.

2.3.3.3   Border swapping

In principle, the interior of each segment can be updated without knowledge of the state of the incoming borders, and could therefore be done while such border communications are in progress. It turns out, however, that the update of the interior typically takes 500 times longer than the border communication[10], so the gain by overlapping the border transfers with interior updating and thereby subtracting the transfer time from the total update time is generally negligible.

Synchronisation   Each processor is able to update its segment of the lattice independently of the others, but the necessity of the border transfer at every time-step enforces a *loose synchronisation* between processors. No processor, no matter how much more quickly (or slowly) it completes its update, can ever be more than a single update step ahead (or behind) its immediate neighbours, because it requires information from its neighbour (or vice versa) to continue to the next time-step.

It can be ascertained that no processor can ever be out of step with any other processor by a number of updates more than the number of intervening processors. The worst case would actually be when the processor on one end of the chain was $P-1$ updates (where P is the number of processors) out of step with the processor on the other end.[11]

Processor lag   It is observed that it is rarely possible to interrupt the free evolution of the lattice and have it immediately respond. This is due to a global consistency requirement whereby no processor can stop processing its segment of the lattice until it has reached the same update time-step state as the most advanced segment. In general, a few more updates are required (though only on the processors lagging behind) to bring the lattice into a consistent state for subsequent dumping, interrogation or analysis.

The synchronisation does *not* imply that the simulation *necessarily* goes at the rate of the slowest to complete each update, since any temporal fluctuation in workload

---

[10]The actual communication time is negligibly small, however, an arbitrarily long wait may be necessary for the other processor involved in the transfer to be ready. Minimising the imbalance between processors is far more crucial.

[11]When chain wrap-around boundary conditions are used, such that both ends are effectively joined into a ring, the worst case is half this.

causing a processor to lag slightly behind on a particular time-step, could easily be compensated by the inherent slack in the exchange with the neighbour, and potentially recovered in a slightly faster update on a subsequent time-step. However, any spatial imbalance in workload, where a particular processor was continually slower to update than its neighbours, can lead to a dramatic loss of performance.

2.3.4 Collection and analysis of data

2.3.4.1 Cell averaging

The averaging cells of microlattice sites, which produce macroscopic observables, are generally square and therefore have equal numbers of sites on each side, even though this leads to a physical cell shortened in one dimension by $\frac{\sqrt{3}}{2}$ when transformed from the hexagonal lattice.

In principle, the shape of the cells should be related to the shape of the computational box, such that the highest resolution in the region of interest is possible by locating the largest number of cells in that area. There is little point, however, in reducing the size of cells in only one particular area of interest, since there has been no gain in computation (because the underlying microlattice must be globally homogeneous) at the expense of smaller cells, which are more susceptible to fluctuations.

Averaging the quantities on the microlattice is not particularly computationally expensive, but since it does not require to be done every lattice update (and the averaged quantities vary only slowly with time anyway) it would be extravagant to do so. In general, the cell averaging can be done infrequently, and is probably rarely necessary to do so more often than time taken for a particle to traverse a cell.

2.3.4.2 Graphics update

To track the progress of a simulation, it can be useful, though not always necessary, to be able to sample the state of the lattice in each processor occasionally while it is being updated. It is possible for the averaging process to be carried out occasionally, and this information sent to a graphics board for display. There is essentially no overhead in doing so, above that of calculating the macroscopic averages, since the graphics packets are routed independently of the operation of the rest of the program, and can be displayed in real-time on a dedicated display.

Such a facility can provide an important insight into the way in which a flow is developing, or alternatively spot anything untoward at an early stage, and therefore avoid wasted computation. Flow visualisation of the lattice-gas is discussed more fully in **Section 2.4**.

2.3.4.3   Data analysis

The complete analysis of the state of a lattice could be executed sequentially on a single processor, after the complete byte-wise dump of the lattice data. Alternatively, the lattice can be analysed interactively, with each processor partially analysing the data in its segment and forwarding the results for final collation on the master processor.

If the simulation requires to be restarted, or subsequently analysed in the future the complete byte-dump of the lattice data is required. In general, the macroscopically averaged data (although much more compact) doesn't contain enough information to transparently continue the simulation, and certainly can not be used to reverse the simulation, since the information about every single particle is necessary to do so.

For the expense of initially calculating a suitable distribution of particles for each cell, it should be possible to generate a starting configuration from the macroscopic averages. Since it is known that the relaxation time is in general only a few time-steps, the microlattices will be essentially equivalent after all the particles have undergone several collisions. This then leads, however, to the further complication of the storage of arbitrarily designed barriers and source features which were previously transparently stored within the byte-dump.

2.3.5   Flow configuration

Initially when a lattice-gas system is being designed, it is necessary to suitably position and shape the barriers and sources which will form the flow. This is simply achieved by setting the individual sites in the lattice with the appropriate codes.

2.3.5.1   Barriers

Barrier sites are generally coded to be non-slip, i.e. they reflect incident particles back along the link on which they arrived, but they could be set with a different code to specify that particles should attempt to be specularly reflected, or some other update mechanism used. Since they themselves require no update, though they may complicate that of their neighbours, they are cheap, and large regions of barrier on a processor are likely to considerably lighten the workload for that processor. Although not essential for overall efficiency, it is usually beneficial if large barriers are split over as large a number of processors as possible.

2.3.5.2   Sources

The implemented strategy   Source sites generate new particles to be introduced to the lattice, and the rate at which particles are generated is the major con-

trol over the dynamics of the simulation. Source sites therefore have their strength coded into the lattice (on a site-by-site basis), whereby a source site with strength 20% would produce a particle of each available type[12], on average once every five time-steps. At any particular generation, more than one particle may have been generated, and indeed, with such a strength the average number of *site particles* will be $6 \times 0.20 = 1.2$. This case would be refered to as specifying a particle *link density* of 0.20, producing a *site density* of 1.2 .

Source sites also completely absorb incoming particles, and it is the difference between the net influx and outflux of particles which governs the sources' impact on the neighbouring lattice sites. A source with zero strength will continually remove particles from the lattice and has the effect of an absolute vacuum (or 'black hole'). Such a component within a system is unlikely to be very realistic, producing an apparent *streaming* of particles along the lattice axes (see, e.g. **Plate 9b**).

More usefully, when a source site produces fewer particles than exist in its neighbourhood, the net outflux models the behaviour of a *sink*.

Alternative strategies   Instead of solid blocks of source sites, producing particles at a probabilistic rate, an alternative strategy is to continually produce particles from a diffuse pattern of source sites [WM89b]. This method has the advantage that it is simpler (since it is deterministic) but may well be unbalancing for the resulting flow, since particles tend to stream from the source sites. To minimise this effect, periodic relocation of the source sites could be employed at relatively little expense. Either way, the region affected is likely to be restricted to the vicinity of the source site region.

From measurements of the average particle flux through the system, more complex boundary conditions can be designed, which may well improve boundary behaviour [CdH87, SR88, BN87].

2.3.5.3   Configuration

Using the various components available it is possible to design a system within the computational box within which the desired flow will form. Typically, if an obstacle, or channel is desired, appropriate sites would be set as barriers. Then the box would be filled with particles, such that the required density of gas was obtained. A flow will result when the upstream edge of the box, or whatever source feature, is set to produce a net influx of particles, and the downstream edge or sink, is set to produce a net outflux.

The flow will require time to equilibrate after it has been set up. Local equilibrium, to which the initial randomly generated particle configurations will relax to, occurs

---

[12]It makes no sense to generate stationary particles, since they have no possibility of leaving the source site at that update, and it would be an unnecessary complication to also have to store them.

after a few updates. However, global equilibration requires a time of the order of the length of the dimensions of the computational box, since information about remote sources or whatever will only propagate through the fluid at the speed of (unopposed) particle propagation, which is one site/timestep.[13]

2.3.6   Lattice design

With the inherent flexibility in the lattice-gas implementation, the only impediment to designing obstacles, or equally sources, of arbitrary shape, is the quality of the *lattice construction editor*. Since, in general, the lattice is considerably larger than can be held resident on a single processor, and the construction operations typically require to be done on a site by site basis anyway, a distributed construction is desirable.

In a distributed lattice, where consistency is important, and in the lattice construction phase, where performance is not crucial, the technique of applying each construction operation to the data resident on each processor is not inappropriate. Some performance gain is possible in calculating crude bounds on the data to which the operations will subsequently be attempted.

Only a few basic construction primitives are provided, which can be powerfully and flexibly combined in the construction of completely arbitrary features. These primitives are:

blocks for rectangular features, like sources along edges of the computational box,

circles and arcs which are useful in the design of smoothly curving features,

lines to specify a connected line of feature sites[14], and

wedges which fill an arbitrary triangular wedge with a feature.

Primitive location and specification can be done interactively through use of the mouse and/or keyboard input, or alternatively through a command file specification.

These are in addition to the *single-point editor*, which allows much greater control over the individual sites to be modified, and is therefore useful in tidying ragged-edges and ensuring that interfaces between sites specified through primitives are correct, i.e. there are no gaps in what is expected to be a solid object, or the join is suitably smooth between objects or at the boundaries between processors.

When editing, a primitive object-type is matched with a data-type, which specifies what type of feature should be set into the sites of the lattice. All of the desirable data-types can be selected: barriers, sources of the desired strength, a

---

[13]Actually, pressure waves from sources propagate with the speed of sound, which is of order one as well.

[14]A modification of the Bresenham line algorithm [FvD82] for a hexagonal (rather than the usual square) lattice is an interesting complication.

particle distribution of the desired density and momentum (so that each site within the selected region is given appropriate random particles), or null sites (devoid of particles).

2.3.7 Comments

2.3.7.1 Update speed

Current implementation The *FHP7* lattice-gas as implemented, which includes the full collision set and a pseudo-random choice between equivalent collision channels, is found to run at the rate of 17,000 site updates per second per processor (for a typical simulation). This corresponds to the update of just over one million sites per second when 64 processors are used, and indeed a display rate of one frame per second was achieved with a million site system. The maximum sustained throughput achieved was two million site updates per second, using the full ECS complement of 128 processors.

The lattice site update can be speeded up, by a factor of perhaps[15] 2, although at the expense of storage. Since the fastest suitable operations on a transputer are those on words, when each particle is coded into a word throughput is greatest. However, a single particle stored in a word, is 32 times as wasteful of memory, as a particle stored as a single bit in a byte.[16]

It is unfortunate that no benefit can be made of the real power of the transputers used, the fast floating-point unit, since all the update operations are inherently bitwise in nature. Indeed, while most ECS applications enjoyed a performance increase of a factor of 10 in upgrading from T414 to T800 processors, the lattice-gas code only went marginally faster (and some of that increase may have been due to the link speed being twice as fast)

Other implementations By way of comparison, a FPS-164 implementation [dHL86b, dHPL85] of the *FHP7* model also updated one million sites per second and a 16,384 processor Connection Machine CM implementation [Hiebeler90] of a deterministic "*FHP* model" was claimed to update at a rate of 3 million sites per second. However, [RHFdH88] claims an amazing 30 million 24-bit site updates per second for the far more complex *FCHC* model on a 4 processor Cray-2.

Using specially constructed hardware for the lattice-gas update is also a possibility, and indeed machines like CAM [TM87, MT87] and RAP [CdH87] can achieve 40 million site updates per second. However, these machines tend to be of very restricted use, since their design precludes diversification into investigations of new models which require a different basis — for instance, hardware limitations of 4 or

---

[15]dependent on the choice of collision rules

[16]This strategy was investigated, producing a contender for the greatest *occam* IF statement, but never incorporated into a full simulation since it would have only implemented a restricted update rule.

16-bits per node are rapidly seen to be unduly restrictive for general simulations, which require 8 or, more often now, in excess of 24 bits per site. It is likely that hardware implementations not designed with 24 bits for moving particles, a couple of bits for rest particles, and probably enough bits for another species of the same — in addition to some special bits for barriers and sources — (let's say roughly 64 bits per site in total) will be considered restrictive.

Although slower, software implementations allow the flexibility which is necessary to modify the basic model to research new fields. Cellular automaton environments, such as CAPE [MW90, WN90] or PUSSYCAT [Pauwels89] have become popular recently because of this. Indeed, it is a software approach on general, scalable transputer hardware, which was favoured by Shell for lattice-gas research into simulations of oil reservoirs [Sopsick89].

2.3.7.2 Load balance

Experience The problem of unequal load balancing is very real in lattice-gas simulations, and perhaps even more so when a two-dimensional partitioning is available (as in CAPE). When designing, say, a system where two channels meet, it is inevitable that large areas of the computational box will consist solely of barrier sites which require no update. Those processors to which they have been assigned are essentially wasted, and can lead to very poor overall performance.

From experience of a number of applications developed and ported to the ECS, it is generally trivial to achieve excellent load-balancing on less than 16 transputers, however, it is much more difficult to achieve even reasonable performance on 64 and more. Indeed, the achieved performance of 64 processors may well be only twice that of 16, and the culprit is usually poorer load-balancing as the work per processor decreases.

When extreme lattice-gas system designs are avoided, the load balance can be exceptionally good. This is perhaps not unexpected, since the lattice update is fairly uniform, and the amount of work involved in updating a large lattice is considerable, even for a number of processors.

Practice For a fairly undersize simulation, consisting of 500,000 sites or so, the $T_{\alpha}\chi$ transputer activity monitor[17] [Wylie90b] showed that each processor over which the lattice was distributed was equally busy, even when the number of processors used was non-trivial. On 16 processors, each transputer sustained an activity, α, of 99(1)%, while with 64 processors busyness peaked at $\alpha = 97.5(5)\%$ as shown in **Figure 2.5**. This performance is only achieved with 64 processors, however, once there are sufficient timesteps between display frames. Otherwise, the graphics of the display (and the packet routing to it) is a bottleneck which requires the processors to wait before they can send the information for another frame. The

---

[17]The *activity* or *busyness*, α, is determined from the recorded *idleness*, ι, which is how often the processor is found with an empty process queue.
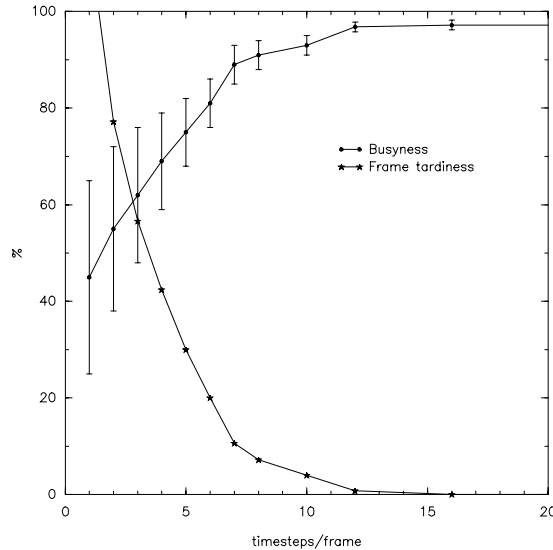
Figure 2.5: Update performance
The degradation of the update performance with frame rate corresponds to inefficient use of the work processors. The mean processor busynesses, as recorded by the Toχ monitors, is sub-optimal with fewer than a dozen or so timesteps per rendered frame due to graphics bottlenecks.

worst degradation, when every frame is being displayed, results in a doubling of the simulation time.

The Toχ busyness monitor includes a display (in conjunction with the graphics of the simulation) consisting of an array — though a chain is readily configured for the case at hand — of activity meters showing each processor's busyness. When the simulation is running smoothly, all the bars are the same height and uniform in colour, with an occasional ripple indicating a slight imbalance which is quickly compensated through the loose synchrony. The effect of the graphics bottleneck is to cause massive waves of inactivity which propagate up and down the chain of processors, dramatically decreasing overall performance.

2.4   Flow visualisation

2.4.1   Introduction

The visualisation of the rich diversity of processes of nature as observed in fluid experiments and engineering situations has been an active subject of research, as well as a tool widely used in industry, for many years [vanDyke82, JSME88]; indeed, Leonardo da Vinci applied flow visualisation in his experiments around 1500, sketching the formation of eddies in a wake, etc.

2.4.1.1   Conventional techniques

Many techniques have been developed to extract the features of interest as clearly as possible from wind-tunnel, water-channel and wave-tank (see, for example, [Yang89]). These can be classified into the four groups:

wall tracing where the flow over a suitable coating on the surface of a body shows on the surface [Řezníček89],

tufts where small strands of material, generally over the surface of bodies, deflect to follow the flow direction [Crowder89],

tracers where small visible particles, like smoke, bubbles or dye, are incorporated into, and carried along by, the flow (e.g. [Werlé89a, Mueller89, Nakayama89]), and

optical methods relying on the differing optical properties of the fluid as it shears or from density gradients (e.g. [Merzkirch89, PSV89a]).

The flow is typically either observed by the naked eye or recorded on film (either photographs or video-tape).

Such visualisation provides an extremely valuable qualitative insight into the processes operating, however, they have been typically regarded with scepticism due to the lack of quantitative information retrievable. Flow measurement, while capable of providing quantitative data, traditionally could only do so for limited sections of the flow field, due to contraints of the equipment available — modern 'whole-field' flow measurement techniques avoid this restriction. Capturing the physical phenomena of the entire flow field is generally to be prefered.

2.4.1.2   The computational rôle

With the advent of powerful computers capable of manipulating the extremely large data-sets which can be generated by such flow experiments, they have been able to provide a valuable contribution to the analysis of the measured data. This is in addition to computational simulation, which typically generate such data-sets from theoretical models, which are an approximation of reality, when they cannot be solved exactly. However, they have had an even more significant impact on the visualisation process, permitting the collected and analysed data to be interactively displayed in a variety of formats to the engineers specification, such that the distracting information is filtered out while the features of interest are highlighted to be as prominent and unambiguous as possible [Smith90].

These techniques are complementary, bringing together the expertise of the experimentalist in recording the measurements of the observed flow, the mathematical models and their solution of the theorist, and analysing the two simultaneously side by side such that their differences can be contrasted and compared. The desired result is a better understanding of the processes involved. Or indeed, it is possible to combine the techniques of simulation and visualisation to create something which looks like the real thing; like the atmosphere of the planet Jupiter for the film "2010", where the cloud motion was created by solving the Navier-Stokes equations on the planet's surface and visualising the paths of millions of tracer particles within it [YUM86].

2.4.1.3 The visualisation process

Large scale numerical simulations and detailed experiments are capable of generating data faster than its significance can be comprehended, while also requiring massive, fast-access data storage devices. Real-time visualisation is capable of providing a powerful data reduction, such that the inaccuracies or instabilities can be discovered early (allowing revised techniques to be incorporated), insufficiently resolved phenomena identified, or results summarised.

The simulation and analyses processes are cycles, requiring generation, filtering, transformation and condensing of data, such that the resulting description is more relevant. Even at this point, extracting the important features can be difficult, and a mapping of the processed data into a form where its characteristics can be immediately recognised is best done as an image — preferably one which can be interacted with.

Although it possible to display all the data produced from a simulation or experiment, this is rarely desirable. The goal in scientific visualisation is to reduce the image complexity to the bare minimum needed to comprehend the science.

Since the imageable area on a display, known as the *screen real estate*, is a valuable commodity, it is important that every component of the screen image is as clear and concise as possible, while not clashing with other features — hence the current progression to window-based displays, like that employed in CAPE [WM89a, WM89b], allowing images to be displayed inside windows, which can be overlapped, repositioned and resized on screen, or iconified when not required.

2.4.2 Visualisation techniques

A variety of techniques (e.g. [Upson90] provides a survey) can be employed to best present the details within a data-set as an image, but there is generally a trade-off between an informative and an economical representation. These considerations are more important when dealing with three-dimensional (or higher) data-sets than the (spatially) two-dimensional data to be considered in the lattice-gas simulations undertaken in the following chapters, but the necessity is greater as well.

To provide a basic (while pseudo-realtime) display of the two-dimensional lattice-gas only two[18] visualisation techniques were employed. Both produced a two-dimensional map: one of icons and the other a continuous tone map of tiles.

---

[18]An additional technique would have been to specially mark certain particles and follow their paths, using them as *tracers*. This was not investigated, but is inherently feasible in lattice-gas simulations.

2.4.2.1 Direction as icons

The local direction of motion of an element of the fluid can be specified as an angle, and can be visualised by the rendition of an informative icon at the point. When an icon is produced for each cell in a complete mapping of the flow, it is possible to continuously trace the flow-lines, and eddies are quite prominent.

Vectors   Although other techniques can be used to visualise angles, they are best represented by vectors at the sample points pointing in the requisite directions. The computational complexity of generating a properly scaled arrowhead in a suitable orientation at each point can be quite large, however, while also commanding a considerable amount of screen space. Also, when the arrowheads are small, they tend to be poorly resolved[19] being more akin to 'blobs' than vectors.

Lines   When only lines (i.e. without arrowheads) are used, the absolute directional information is missing, but it is still possible to objectively determine the sense of motion of each element. Such a process is reminiscent of that using tufts to show flow direction [Crowder89]. Although lines are obviously simpler (and faster) to render, and result in a less cluttered display, the ability to immediately determine the absolute direction of a flow, or the sense of rotation of an eddy, is lost. This can least to ambiguity in the display, at least initially, until the direction of each element is objectively determined (potentially, by locating the position of the source and following the line directions from there) in the form of streamlines.

Hybrid icons   One compromise technique is to combine a visible mark (generally a small circle, but a simple 'blob' is sufficient) from where the 'vector' originates with a line in the flow direction. In such a way, it is possible to determine the flow direction, albeit in a less intuitive manner. Alternatively, only one half of an arrowhead need be drawn.

While such techniques can produce clearer and less cluttered images for the trained eye, they are intrinsically less intuitively obvious to the casual observer. The optimal form of the lines in any situation therefore ultimately depends on the complexity of the flow and the 'desired' image.

2.4.2.2 Attribute mapping of magnitude

Where a property takes values within a fixed range, such as the density and speed of the lattice-gas (which are constrained within the ranges [0..1] and [0..2] respectively), it is possible to represent their values with appropriate colours from a con-

---

[19]Even on a high-resolution, bit-mapped display, arrows have to be fairly large to be properly resolved without resorting to even more computationally expensive *anti-aliasing* techniques, where pixels bordering the line receive an intensity contribution depending on their proximity to the 'true' line.

tinuous spectrum, or alternatively, gray-levels. A continuous tone map highlights the continuity over the image, instead of a multi-coloured map which can highlight contours when there is sufficient connectivity of the data.[20]

Intensity   Achromatic, gray-levels are a good start, since the eye is very good at correlating, for example, high intensity with high velocity, but it is less capable at determining levels in between accurately. There is also a problem in ensuring that other features of the system, such as barriers, have sufficient contrast against the intensity map surrounding them — in general this is not possible when only a gray-scale is available.

Colour   The use of colour provides a solution, since the eye[21] is very capable at distinguishing a wide range of colours. However, over-use of colour, or too much contrast between colours, ultimately leads to a confusing image in which it is difficult to extract any meaningful information, as well as being fairly unpleasant to look at.

A good compromise is therefore to use a basic intensity scale (though not necessarily a 'gray'-scale) for the continuous features of the map, while reserving a few contrast colours for the special features, like barriers, within the flow and for borders and annotations, etc [FvD82].

2.4.2.3   Composite mapping of the flow

With a careful choice of the design of components it is ultimately possible to combine the preceding techniques in what is known as *overloading* — where many properties are displayed simultaneously, even on the same primitive.

An example of such a case, which is especially informative in flow visualisation, is to code the flow speed into the length of a vector, with the flow direction into its angle. The direction of the flow can subsequently further be overloaded by also colour-coding the vector — the additional utility of having the direction specified in the angle of the vector and its colour is very effective and offsets the *redundancy*. In this way the flow has been very thoroughly visualised, and the combined impression of the longer arrows indicating the stronger flow-rates with the areas of strong colour being those where the flow is uniformly in one direction is very powerful.

2.4.3   Image optimisation

To achieve the greatest possible information content from the data to be visualised, it is important that it should be suitably scaled and rendered. Otherwise, the relevant information will be obscured or lost in the final image.

---

[20] 100×100 tiles would only yield a useful contour map when the surface was fairly smooth.

[21] At least for the 90% of the population which isn't (partially) colour-blind.
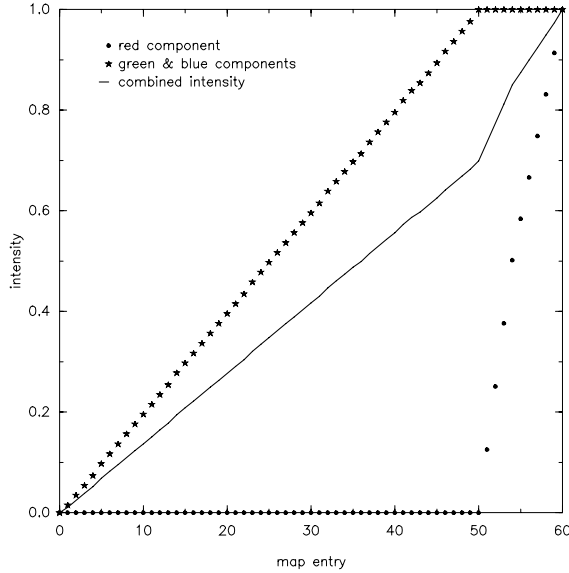
Figure 2.6: Map entry components and intensity
The continuous black-cyan-white scale of the colour maps is shown in terms of red-green-blue components, along with the perceived intensity:

$$Y = 0.301R + 0.586G + 0.113B \quad .$$

2.4.3.1 Scaling

If data is incorrectly scaled, lines or vectors will get too long or too short — if they overlap, information is lost and clutter is increased, though they should generally be scaled such as to (almost) reach their neighbour. With tiles, a suitable range of colour (or contrast) is generally desired to highlight the range of values — achieved by specifying *excess* and *incess* values, within which the mapping of values to colours is done, and setting tiles with values outside the selected range to the excessive or incessive[22] values.

It is also important to provide flexibility to rescale, since the display will be far less than informative with an unsuitable choice of scale, or downright misleading with an incompatible choice of rendering techniques.[23] Inevitably, this is best done interactively.

2.4.3.2 The palette

It is also helpful if an appropriate, and as natural as possible, choice of colour is made. In dealing with fluids, bluish-green seems to be a pleasing choice of 'water' colour, while the red and yellow fluid of CAPE distinctly gives the impression of molten lava, or some noxious chemical soup, and other users may prefer pastel shades. However, the eye is more sensitive in distinguishing reds and yellows than blues and blacks. Since no one palette is inherently better than another, flexibility to choose or design an appropriately informative or pleasing palette is desirable.

---

[22]Where excessive values are out of range by being too high, values are deemed to be *incessive* through being out of range and too low — the distinction is often crucial.

[23]Such as blobs at the vector hilts and poorly resolved arrowheads also appearing as blobs.
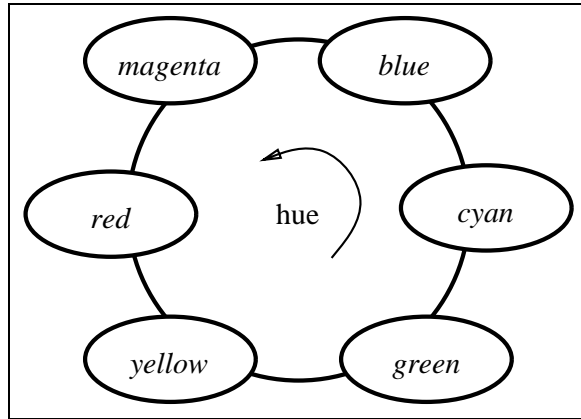
Figure 2.7: Direction colour cycle
Colours are assigned with the flow angle defining a hue. In this scheme, flow moving downstream (to the right) will be coloured cyan, with flow in this general direction, coloured green or blue. Flow in the opposite sense will be predominately red.

Monotonic range The default palette designed for the magnitude map in the lattice-gas visualisation consisted of a linear interpolation from black to 'pure' cyan[24] topped with red to produce whites over the remainder. A fairly linear[25] increase in intensity is found over the range (shown in **Figure 2.6**).

Only 60 palette entries were reserved for the intensity map, since it was found that the inherent fluctuations in the macroscopic quantities in the averaging cells, along with their relatively small number, made an increased number unnecessary for smoother continuity through reducing contouring from rounding. In fact, with the rescaling option a smaller number could have been used.

In the case of a velocity magnitude map, 'darkness' of the low section where the flow speed is lowest is distinguishable from 'blueness' of the mid-section and the 'brightness' of the high section where the flow speed is highest.

Light and dark shades of pink were selected as suitable contrast colours for possible excess and incess magnitudes of the map. A dark blue provided a neutral dark background, while yellow barriers and green sources were always easily visible against the map colours.

Cyclic range The previously described palette is fine for a *monotonic* range, however, it is completely unsuited to a *cyclic* range, as is required when directional information requires to be represented. In such a case it is important that there is continuity at the cross-over, but even then a palette going from dark to light and back to dark provides no way of discriminating between the shades of gray which multiply define the in-between states — flow in the two opposite senses is indistinguishable. Even if it is not necessary to distinguish the flow direction to any great accuracy, it is crucially important to be able to distinguish opposite senses.

Here the additional 'colour dimension' is essential, and it makes sense to use the full amount of colour information for maximum discrimination, i.e. use the primary colours red, green and blue for three primary directions (equally separated), and the secondary colours yellow, cyan and magenta for intermediate directions, as shown in

---

[24]one of the secondary colours, comprising equal amounts of blue and green
[25]Even a linear interpolation is not quite optimal, since perceived intensity is logarithmic.

Figure 2.7. This corresponds to a cycle in the *hue* of colour, with complementary colours 180° apart, so that it is possible to relate green and blue code to flow moving downstream, with reverse flow shown contrasted by being red.

2.4.4   Image construction

2.4.4.1   Display composition

Efficient use of the display is maximised by suitable scaling of the display window to be as large as possible. It is generally preferable if perspective and relative scale of image components is retained, therefore the image will not tend to fill the display, and should be centrally located within it, surrounded by a neutral background. In the case of the lattice-gas simulation, cells were required to be of equal size.

Without resorting to a fully-fledged windowing system, selected regions of the display can be enlarged by suitable recalculation of the display window and ignoring cells outwith those regions — alleviating the need to clip to a boundary. This concentrates the view on the area of interest.

Complex features, like arbitrarily drawn barriers, are rendered once and stored with the rest of the background in a spare frame. This is used as the basis within which the rendered data is overdrawn.

2.4.4.2   Data collection

Since cell-averaging is an integral part of the lattice-gas models, these averaged quantities could be calculated simply whenever desired, and displayed according to the selected format. Averaging was a fixed-penalty activity, which took a fixed amount of time from each of the processors when they could have been performing further lattice updates, but once done, the results were independently forwarded to the graphics processor for rendering. Data for a full image is collected and to ensure that image integrity is retained, data arriving for a subsequent image is stored before the current frame is rendered.

2.4.4.3   Rendering

Rendering the image takes a variable time, dependent on the complexity of the rendering technique selected.[26] When the rendering time exceeded the update time of the lattice it was possible for a back-log of unprocessed graphics packets to develop which would require the worker processors to idle until they could be cleared. Such a situation was obviously undesirable, and through careful tuning of

---

[26]It is also proportional to the number of cells, so that although extra processors could be added to speed up the lattice update (possibly for a higher resolution simulation), the graphics processor would be correspondingly overburdened.

the number of lattice-updates skipped between displays, it was possible to keep all the worker processors busy while the graphics processor was also busy producing updated displays.[27]

2.4.5   Image analysis

Once the image (or set of images) has been created, it's only then that it can be analysed and its meaning extracted — or it is found to be incorrect. Correlation can be done with experimental images or with the theoretically expected behaviour.

2.4.5.1   Image cycling

In fact, one image is itself generally only part of the story, and a whole sequence will be created tracking the flow from its inception, through to when it has equilibrated — with dynamic features, like eddy-shedding, no steady state will be reached. To get a feel for the dynamic properties of a developing feature in a fluid, it can be invaluable to be able to continuously cycle through the sequence of images. Different types of information are obtainable from playback at different speeds: slow playback allows small scale phenomena to be identified, while fast playback highlights the growth and motion of gross features.

2.4.5.2   Colour cycling

Another powerful technique in the analysis of images which contain cyclic information is to cycle the relevant entries of the (cyclic) palette — a process known as *noodling*. The resulting impression, with a suitable image, is to dramatically highlight the vorticity, while also increasing the feel for the flow.

2.4.5.3   Recall

When interactive investigation is completed, or not practical, video capture of the playback sequence provides an acceptable, albeit inferior, storage and replay mechanism. This retains the ability to analyse individual frames (through a pause facility), while also showing the whole sequence.

The alternative, a sequence of still pictures, or a single picture, while perhaps summarising the essence of the flow, is much less useful, unless steady state phenomena are being investigated.

Similarly, the production of an image with a fixed palette has removed the possibility for more complete subsequent analysis.

---

[27]The change between one frame and the next generally was more dominated by fluctuations in the cell average than any gross feature of the flow developing anyway.

2.4.6 Visualisation summary

Since, in general, the outcome of an experiment or a simulation is not known *a priori*, the best visualisation of it is unlikely to be successfully preprogrammed. Iterative re-mapping, selection of different scales to highlight phenomena on different scales and changing the palette to one which highlights a specific feature are all interactive processes, which rely on a operator knowledgeable of the fluid dynamics processes involved. The image at the end of this is, hopefully, more of an encapsulated summary of the exploration than the snap-shots of the voyage.

2.5 Summary

The transputer multicomputer has been shown to be capable of handling, and updating efficiently, large lattice-gas simulations. Although the transputer is not an ideal processor for the bit-wise manipulations of a cellular automaton update, it is indeed powerful, and capable of operation in conjunction with a number of others to provide enough compute power to undertake high-resolution simulations in reasonable time. Lengthy simulations would be expected to take up to a full day on 64 T800 processors, while most would be done overnight.

The partition and decomposition of the lattice was natural and straightforward, albeit only in one-dimension. Although, initially a message-passing harness required developing, which was quite a challenge and took a considerable time to become robust, the recent arrival of general message-passing harnesses, such as ᴛɪɴʏ [Clarke90], removes this overhead. ᴛɪɴʏ also facilitated the incorporation of additional functionality, such as end-of-chain boundary swapping, and the Tαχ performance monitoring.

Efficiency and load-balancing were sufficiently good that an almost perfect scaling was observed in performance with the number of processors — a speed-up of 3.85 is observed in quadrupling the number of processors from 16 to 64, a scaling efficiency of 96%.[28] Geoffrey Fox [Fox89] reckons that since the average Cray-XMP efficiency is around 12%, too high a standard is set for parallel computers. However, such a scaling is necessary for the use of a hundred or more processors to be viable — indeed, initial simulations were on 128 processors.

With the provision of a simple visualisation system, running independently of the lattice update, it was possible to monitor the progress of the simulation, and determine early on whether the expected behaviour was observed. In this way the equilibration of the system could be observed, and the eventual run time estimated, or the system re-designed and restarted.

The provision of a debugging and observing probe capable of displaying the state of part of the microlattice at any instance was also found to be extremely useful in

---

[28]No threat to Amdahl's law which limits scaling efficiency to 100%.

checking particle interactions and boundary swapping.[29]

Many interesting transient and dynamic features were observed during the validation of the code and in the investigation of the lattice-gas properties detailed in the following chapters. Some of these are captured in the colour plates of **Appendix P**, though a video would have been necessary to do them justice.[30] The lattice-gas simulator has, in any case proven, itself to be a flexible and useful tool for the investigation of complex hydrodynamic systems.

---

[29]In fact, the probe proved to be unexpectedly useful in determining that some memory was defective. A single bit — one in 400 Mbytes — tended to drift to the set state even after reset, which manifested as an irregular stream of particles, all along the one lattice axis. Obviously this was only observable when the lattice was empty, but any dubious memory would tend to result in the spurious generation or loss of particles.

[30]The creation of a suitable video presentation for inclusion with this thesis was not possible due to technical and logistical limitations.

3 Fundamental & phenomenological flow simulations

Contents

Arthur looked.
'Why,' he said, 'is there a sofa in that field?'
'I told you!' shouted Ford, leaping to his feet.
'Eddies in the space-time continuum!'
'And this is his sofa, is it?' asked Arthur,
struggling to his feet and, he hoped,
though not very optimistically, to his senses.

— [Adams82]

3.1   Introduction

ELEMENTARY simulations initially provide confidence in the basic lattice-gas model through comparison with the experimentally and theoretically predicted behaviour. Subsequently, more complex flow systems are designed which are also found to exhibit qualitatively similar behaviour to that experimentally observed.

By way of introduction, the static properties of the lattice-gas simulation were investigated. A couple of experiments were undertaken in situations where no net flow was involved, and special source sites were not necessary. Firstly, fluctuations inherent in the cell-averaging process were measured and the optimal cell size was thereby determined. This was followed by the generation of a pressure wave from the centre of the computational box and its propagation recorded to verify the isotropy of the lattice-gas.

The ability of the lattice-gas to correctly produce the Poiseuille velocity profile of a laminar flow in a channel, verifies the no-slip boundary conditions implemented on the lattice edges, as well as the use of the source sites in generating and maintaining a flow. Various barriers are then placed in the channel to obstruct it and eddies, or a von Kármán vortex street of eddies, are observed to naturally form. The eddy-shedding frequency of a prism, which was found to produce particularly well resolved eddies, was then measured.

Finally, through the specification of extra source sites, in addition to those specifying the channel, it was possible to design jets of faster moving fluid. The integrity of these jets and their interaction with the flow of the channel was subsequently investigated.

3.2   Fluids at rest

3.2.1   A contained isotropic fluid

A squarish rectangular box, consisting of 1024 cells was designed, and the number of lattice-sites contained within each cell varied, with the cell dimension, Z, increasing in multiples of four[1]. The exterior walls of the computational box were specified to be reflecting so that particles were confined and none were generated. No net velocity distribution was imparted in the random initialisation of the gas to the desired density, chosen to be 20%.

---

[1]The implementation of the simulator was optimised (and simplified) when cells were restricted to have dimensions in multiples of 4. Square, rather than rectangular, cells were, however, not necessary, but also only investigated for simplicity.
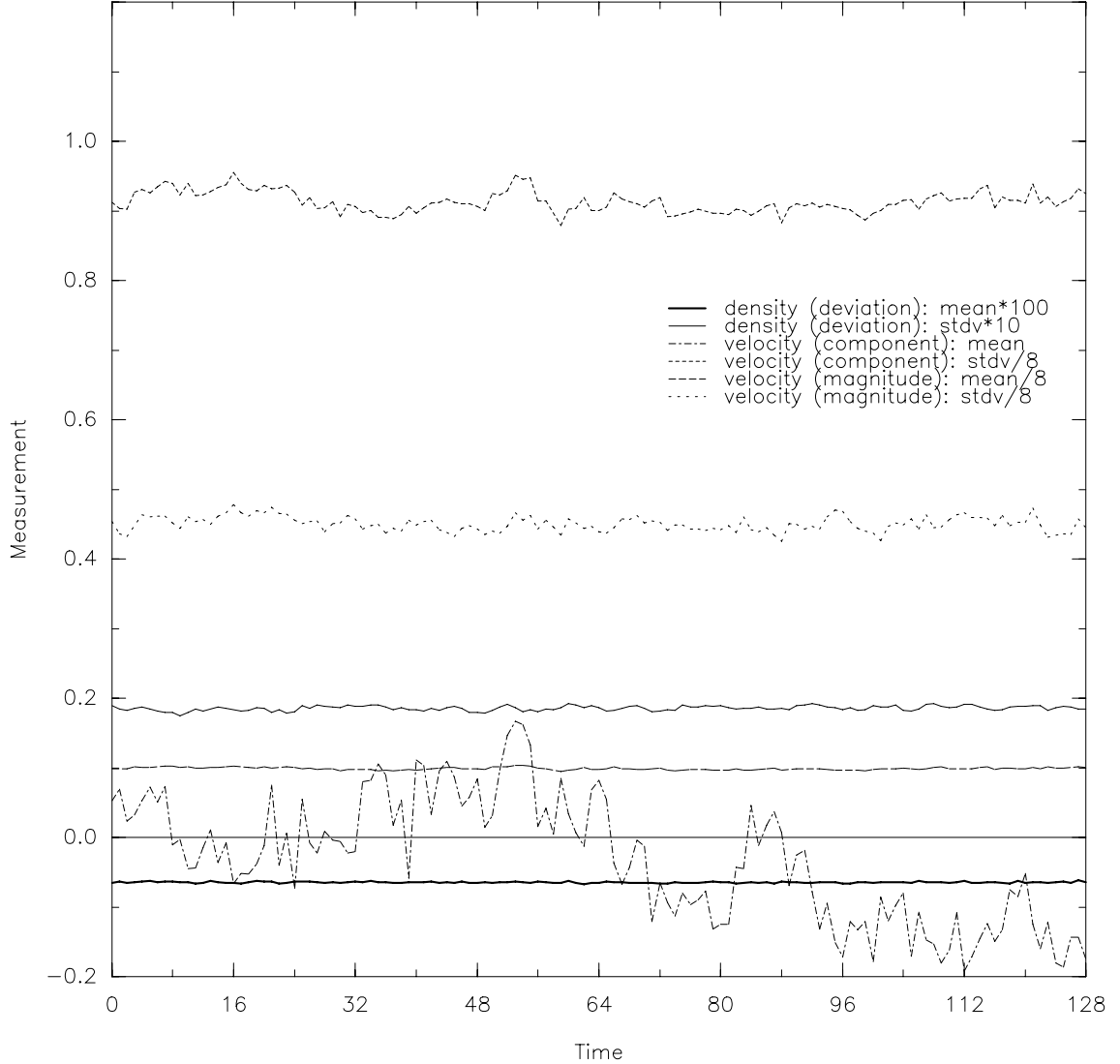
Figure 3.1: Fluctuations with time (example)
The density deviation (from the initialised mean density of 0.20), the component velocity and the velocity magnitudes, averaged over the 1024 cell system, are traced over 128 timesteps from system initialisation, for the example case of 8×8 site cells. No initial drift velocity was applied in initialisation, and no driving conditions were applied throughout its evolution. The velocity is seen to fluctuate far more than the density, with the component velocity varying most (about a zero mean).

3.2.1.1 Determination of the optimal averaging cell size

A range of lattice systems were designed, varying only in the cell dimension, i.e. $Z \in \{4, 8, 12, 16, 20, 32, 48, 64\}$. Considering the system as a whole, and averaging all of the cells in it, the mean and standard deviation of the density[2], the velocity magnitude (i.e. the speed) and the component velocities were measured. This was

---

[2]actually, the deviation from the initialised density

done for the initial random distribution, and over the first 128 timesteps of its evolution. An example of the traces of the fluctuations with time are shown in **Figure 3.1**.

From the summary of the spatial and temporal fluctuation dependence on cell size, shown in **Figure 3.2**, it can be seen that the fluctuations get less important with larger cells, as expected. The larger cells are seen to be fairly indistinguishable, and it can therefore be deduced that computation was wasted unnecessarily on them. Between the cell dimensions $Z = 12$ and 20, little difference is found, and it might be suggested that cells of 12×12 would be useful for quickly determining the characteristics of a flow, followed by simulations using 16×16 or 20×20 if less noisy measurements were desired. Obviously, larger cells impose a greater computational load per update (when the number of cells remain constant), as well as requiring a longer time to equilibrate.

3.2.2 A pressure wave in a contained fluid

To check the isotropy of the lattice-gas, a system was designed where a marked density imbalance would occur in a defined region of the lattice. As the system attempted to equilibrate the density, a pressure wave was expected to form which would propagate from the region. If it was found to propagate radially, the lattice-gas would be isotropic and no prefered directions would exist in the lattice.

A computational box of 32×32 cells, each of 16×16 sites, was uniformly initialised to a density of 10%. A circular region, 52 sites in diameter, at its centre was then initialised to 50% density — the high density region therefore corresponded to $\sim 1\%$ of the whole lattice. No net velocity was imparted to the random distributions of particles.

When the lattice-gas is updated, the density imbalance is equalised by an expansion wave of out-flowing fluid, which is found to propagate equally from the central circle in all directions. An almost instantaneous increase in the density, pressure and velocity is also found at the leading edge of the wave. A combined map showing the pressure map and the flow velocity vectors at an instant before the wave meets the edges of the computational box is shown in **Plate 2a**. As the wave meets the edges, it is reflected and results in similar circular wave arcs heading back into, and through, the centre.

3.2.2.1 Measurement of the isotropy of propagation

From a pressure map of the system before the high density area was incorporated in the centre, it was found that the pressure fluctuated from 0.16 to 0.25 . After the central high density area was introduced, it was manifest as a region of much higher pressure — indeed, the central four cells had pressures very close to 1.00 .
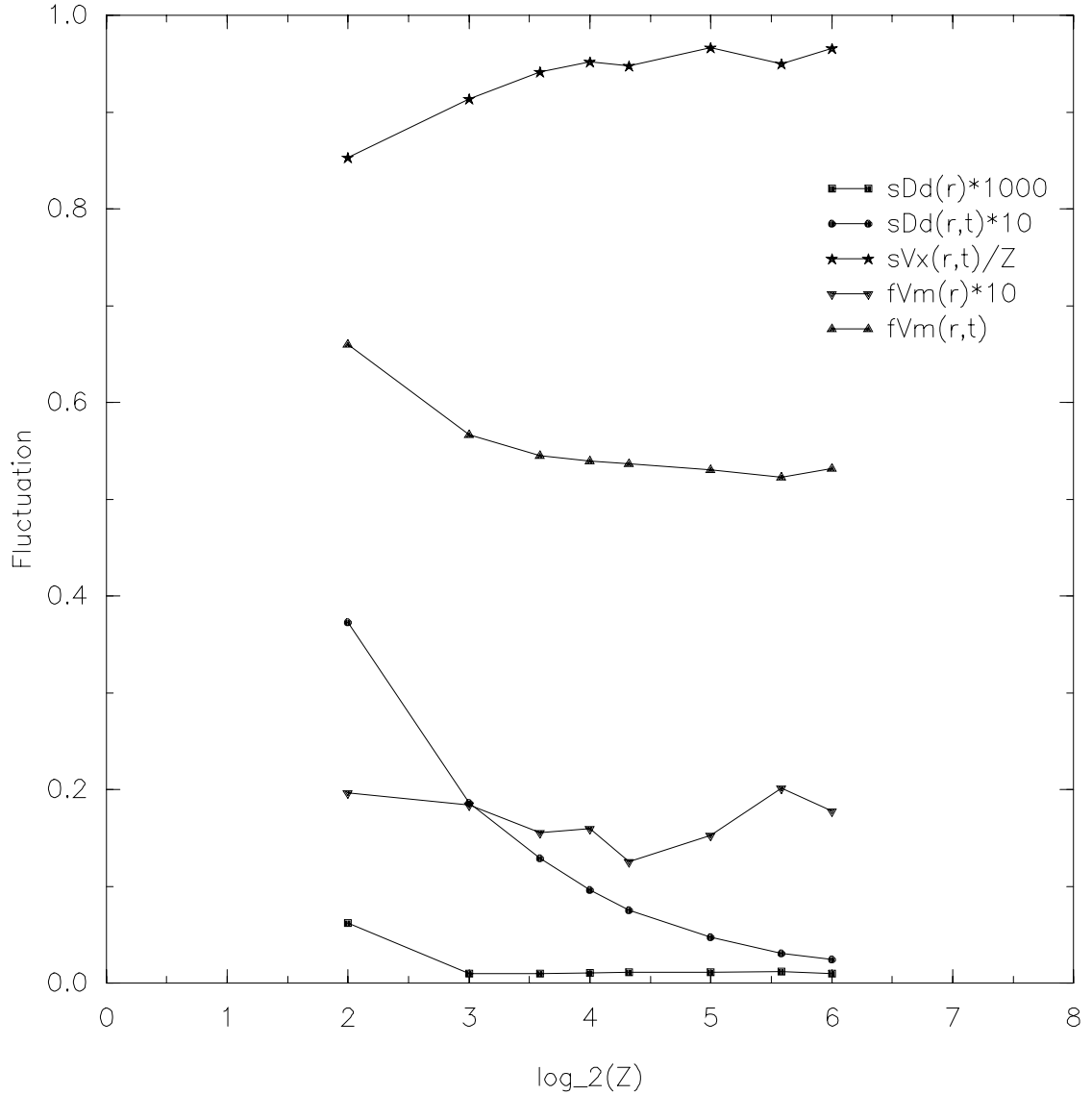
Figure 3.2: Fluctuations with cell size

For each cell size, Z, the fluctuations in the measured cell-averaged quantities are summarised. Spatial averages (r) are over 1024 cells, while temporal averages (t) are over 128 time-steps from initialisation, with no driving conditions.

Firstly, the standard deviation in the density deviation from the mean density (**sDd**) is seen to be constant for cell dimensions $Z > 8$ when the spatial average is taken, and decrease exponentially with increasing cell size when the temporal average is also considered.

The standard deviation in the velocity components (**sVx**) increases with cell size, however, when scaled by the cell dimension, it is found that an asymptotic limit of approximately 0.95 is reached.

Lastly, fluctuations in the velocity magnitude (**fVm**), i.e. the speed, are seen to be roughly constant when only a spatial average is performed, but to decrease asymptotically to approximately 0.5 when time averaged as well.

71

(a) Cartesian map   **(b)** Radial map
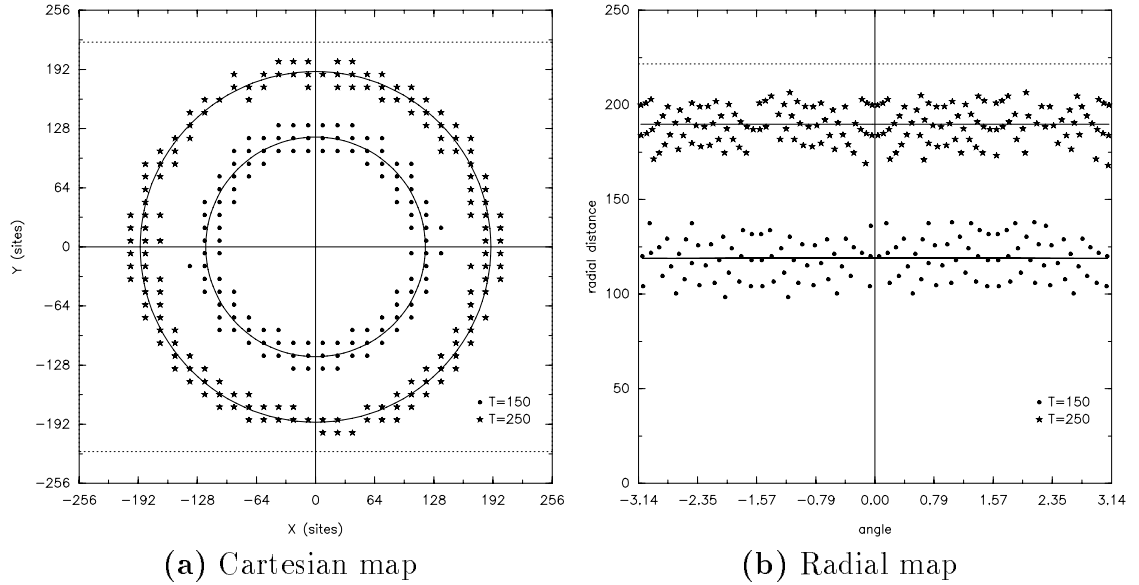
Figure 3.3: Pressure wave front location
The location of the pressure wave front at two instants in its evolution represented by those cells with a pressure in excess of 0.25 . The dotted lines are the limits of the computational box, and the solid lines in (**b**), and the circles in (**a**) are from the mean of the radial distances, namely:

Time	Points	Distance	Std.Dev.
150	121	119	9.2%
250	159	190	5.0%

Once the simulation started to be updated, an *expansion* wave was observed to propagate from the centre as a high pressure circle, followed some distance behind by the low pressure circle. These correspond to *compression* and *rarefaction* of the gas as a result of the wave, and can be clearly identified in **Plate 2a**.

At a couple of instants in the evolution of the wave before it reached the first bounding wall, those cells found to have pressures in excess of 0.25 were plotted, as shown in **Figure 3.3a**. Transforming the coordinates from Cartesian to polar, and performing the corresponding plot (**Figure 3.3b**) no radial dependence on angle is found. The wave has therefore propagated isotropically, and the lattice-gas is isotropic.

From the two mean positions of the expansion wave, it has travelled a distance of 71 lattice units in 100 timesteps, giving a crude speed measurement of 0.7, which is roughly the expected speed of sound for the *FHP7* lattice-gas. A more accurate measurement of the speed of sound follows in **Section 4.2.1**.

72

3.3   Channel flows

3.3.1   Introduction

A series of simulations based around a basic channel, with a uniform flow along the channel from left to right are the next subject of investigation.

Channel specification   A channel was formed by retaining the no-slip boundary of the upper and lower edges of the computational box, while positioning particle sources on *both* the perpendicular edges. Ensuring that the sources on the leftmost edge introduce particles at a higher rate than those on the rightmost edge leads to a net flux of particles through the channel from left to right.

Initialisation   The initial state of the region which is devoid of special features, and where the initial particle distribution resides, is not crucial to the equilibrium state. However, it will take longer to equilibrate from a grossly unnatural state, like a total vacuum corresponding to an empty box, than from one which is much closer to the eventual equilibrium state. To this end, it may be preferable to initialise the computational box to a particle distribution which one might expect to be eventually realised — for the specified channel, the particle density is likely to be intermediate to that pertaining at the boundaries, and the velocity distribution will tend to favour particles moving in the down-channel direction. Gross initial deviations from the equilibrium state tend to manifest in the production of expansion and compression waves, which can be detrimental to the speedy equilibration of the system, therefore a little thought and effort in initialisation is generally worthwhile.

Flow generation   The source on the downstream edge plays a very important rôle in maintaining the flow integrity. Like the upstream source it is responsible for maintaining the free boundary of the computational box, such that incident particles are removed, while new particles are freshly generated and introduced. Other conceivable states for the downstream edge would have had all particles reflected (as from a barrier) or removed (as from a sink, or vacuum). Even incorporating a mixture of the two would appear to, at best, lead to an imperfect solution.[3]

The presence of pure sinks (especially if they are static) tends to rapidly destabilise the simulation, resulting in *streaming* of particles along the lattice axes towards the sinks, since these are extremely deficient of particles traveling in the contrary direction. This is visually very obvious and rather unnerving, and must adversely effect the lattice-gas dynamics. It is likely, that these sinks have a similar perturbing influence as a pure vacuum (or 'black hole') has on its surrounding neighbourhood, leading to radically diverse behaviour, even considerable distances away.

---

[3]CAPE [WM89b] suggests a mixture of absorbent sources and barriers, which can lead to difficulties unless care is taken to ensure that they are regularly, randomly repositioned.

It is important to note that it is the *relative* strengths of the upstream and down-stream sources which will tend to govern the rate of flow through the channel; where they are identical no net flow will be observed. Obviously, other details of the lattice-gas, like its viscosity, and the channel, or how much it is obstructed, will also contribute to the resulting flow.

Flow characteristics   A diverse range of single flows[4] will be considered, and shown to exhibit a range of phenomena, primarily at moderate Reynolds numbers. Laminar flow, where each fluid element travels smoothly along a simple well-defined path, typically occurs where the Reynolds number is low (and also, generally, the flow speed). Fully turbulent behaviour, characterised by rapid, irregular, spatial and temporal velocity fluctuations, is not expected to be observed, unless the Reynolds number gets sufficiently high (certainly more than 1000) which is likely to be beyond the limitations of the *FHP7* model. However, surprisingly enough for a deterministic model, non-deterministic behaviour can be observed, along with static and transitory eddies, or regions of flow circulation.

3.3.2   Laminar flow in a channel

3.3.2.1   Introduction

Osborne Reynolds' famous experiments [Reynolds1883] were undertaken using a long pipe of uniform circular section, with flow between two reservoirs. The two-dimensional equivalent of such a fundamental flow in a pipe is flow in a channel between two parallel plates. The channel is considered infinite in its third dimension, with the walls so much further away from the narrow channel of interest to have a negligible contribution. Since the walls are no-slip, fluid immediately adjacent to a wall is stationary. However, in the centre of the channel (or pipe) the flow rate reaches a maximum, with the flow increasingly slowed down the closer it is to a wall (and slower moving fluid).

This *braking* effect of the channel walls has an increased effect the further distant from the entrance to the channel. A rather square profile at the entrance, resulting from the uniform speed across the inlet where the fluid was introduced, is progressively smoothed further down the channel until the limiting state parabolic profile is achieved some distance down-channel.

Poiseuille flow   Laminar flows in pipes and channels have similar parabolic sectional velocity profiles sufficiently far downstream from the entrance — upstream of this *entry length* the profiles are tending to the parabolic profile. In pipes, such a flow is known as *Poiseuille flow* and the limiting parabolic form is predicted theoretically [Schlichting79].

---

[4]i.e., flows consisting of a single source area and single sink area, as opposed to multiple, distinct source areas which will be considered in **Section 3.5.2**.

3.3.2.2 Experiment and analysis

The channel consisted of a lattice of 240×12 cells of 20×20 sites, i.e. 9600×240, of which one column of sites on each end were source sites, and the remainder free. The intended upstream source was 20% in strength, and the downstream one 10%. The lattice was initially filled with particles to a density of 0.20 (i.e. 1.4 particles/site) and with no net velocity.

After 6000 time-steps, the wave advancing from the left had almost reached the other end of the channel, as can be seen from **Figure 3.4** — the dramatic change in velocity (and to a lesser extent in density) around segment 202 (0.85 of the way through the channel) is clearly visible. The front is even more prominent in a colour-coded map of the velocity direction, uniformly heading down-channel in one part and randomly in all directions in the other.[5] Soon after, as the simulation continues and the wave passes, the channel reaches its equilibrium state; little down-channel variation is visible after 10,000 time-steps.

Analysis was done splitting the channel breadthwise into 12 equal sections of 20 segments (400×240 sites). Within each section, the down-channel component of the velocity of the particles within each row was accumulated. No averaging in time was performed to reduce the noise within such small averaging cells, though this could have been done if required.

Profiles   Velocity profiles for 400×1 averaging cells at a selection of points in the channel, before and after the shock front, are shown in **Figure 3.5**. After the channel has equilibrated, the mean profile is calculated, as the average of all the sectional profiles, and shown by the thin profile in **Figure 3.6**. From the regression analysis, only the quadratic and constant terms were found to be significant, shown by the bold curve.

It can be seen that the mean profile has the parabolic form of the Poiseuille profile, and as expected, the velocity is zero at the walls from the no-slip boundary condition. The section at the head of the channel (and also at the head of the compression wave) has a much squarer section, where the boundary layer hasn't expanded to fill the channel (**Figure 3.5**, sections 0 and 9). Comparison of the near-inlet profile with the standard solution of Blasius, and the eventual parabolic profile with the calculations of Schlichting, for a similar lattice-gas simulation produced a very good agreement [dHL87].

The peak velocity (taken as the mean velocity of the central sixth of the width of the channel) of each section is found to initially increase as the Poiseuille profile is formed, and then to decrease further downstream, as shown in **Figure 3.7**. Over the majority of the interior of the channel, however, the velocity is found to be close to 0.25 .

---

[5] From this map, which relies on colour to resolve direction and is therefore miserable in gray-shades, the front is clearly in segment 204.
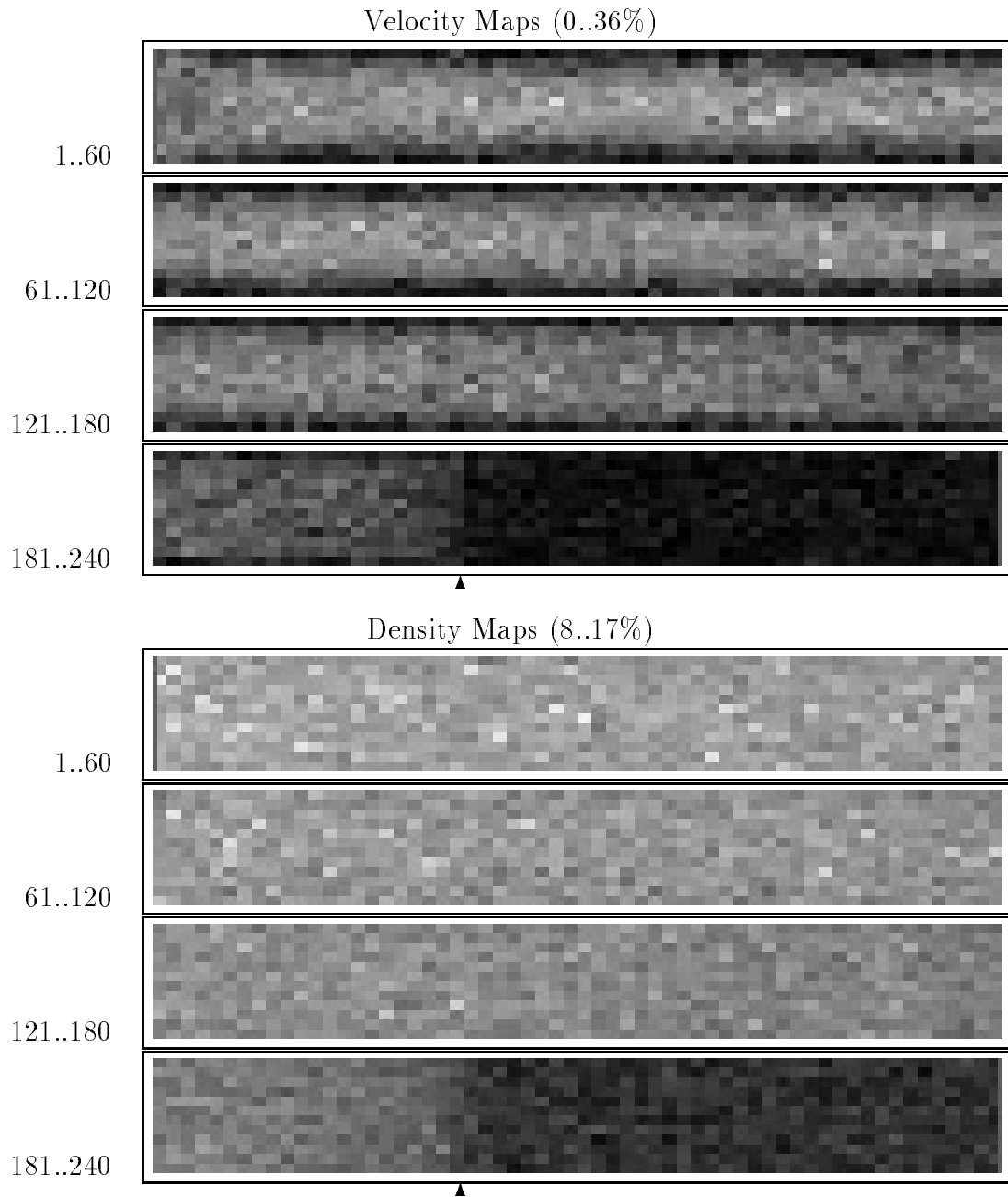
Figure 3.4: Channel velocity and density maps
Display of the 20×20 site cell channel system shown at $T = 6000$, with the channel split into 4 equal parts for ease of display. The values corresponding to the range of each map are shown above it. The advancing shock wave is clearly distinguishable in segment 202 in both cases (marked with the arrowhead). In fact, the shock front is a couple of segments ahead of where it can be identified in these maps, around segment 204.
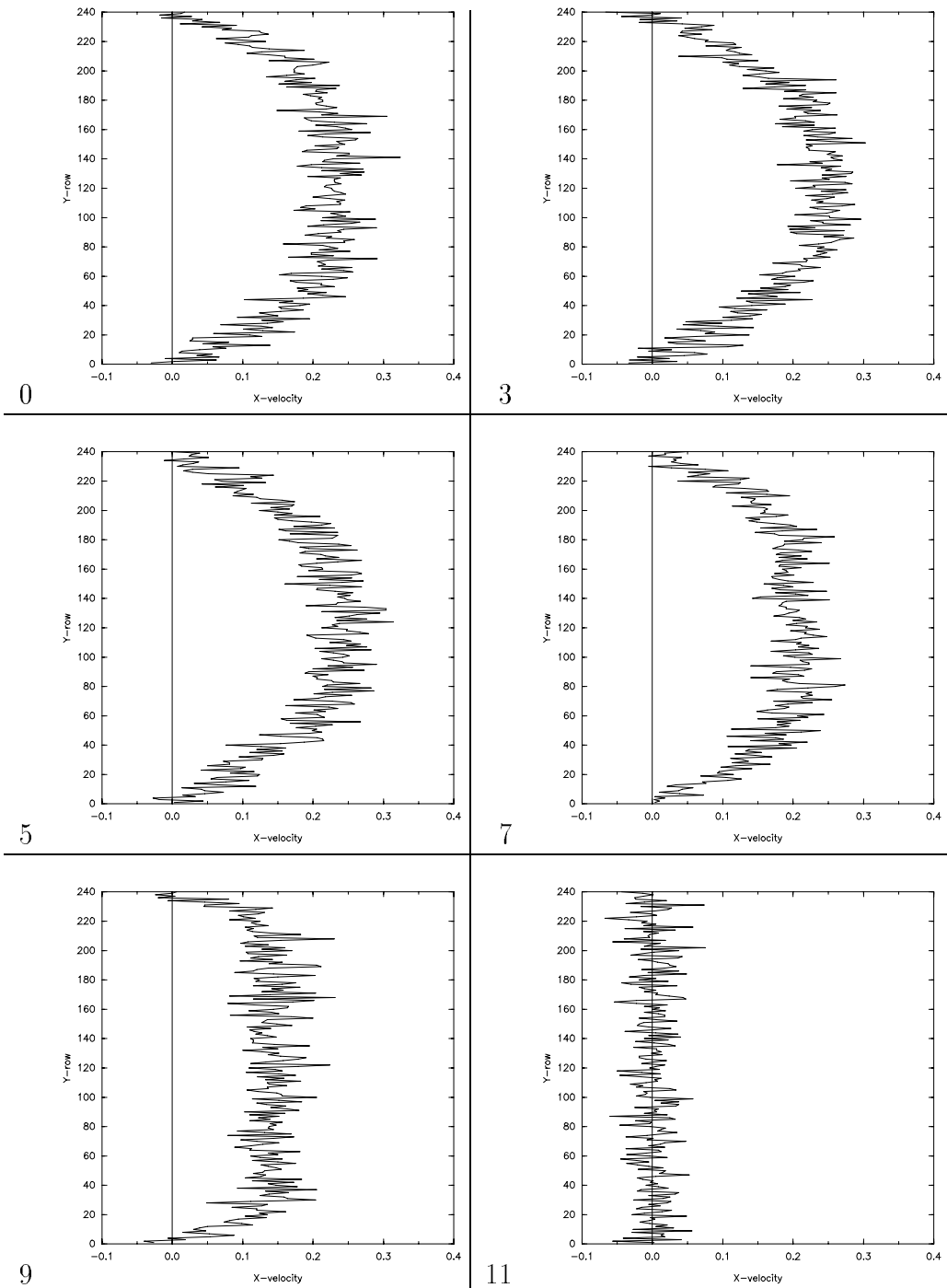
Figure 3.5: Channel velocity profiles

Sample velocity profiles at $T = 6000$: the entrance to the channel, in section 0, builds up the flow, which develops through sections 1 to 6, before being diminished at the front in sections 7 to 9, leading into the unencountered fluid static in sections 10 and 11.
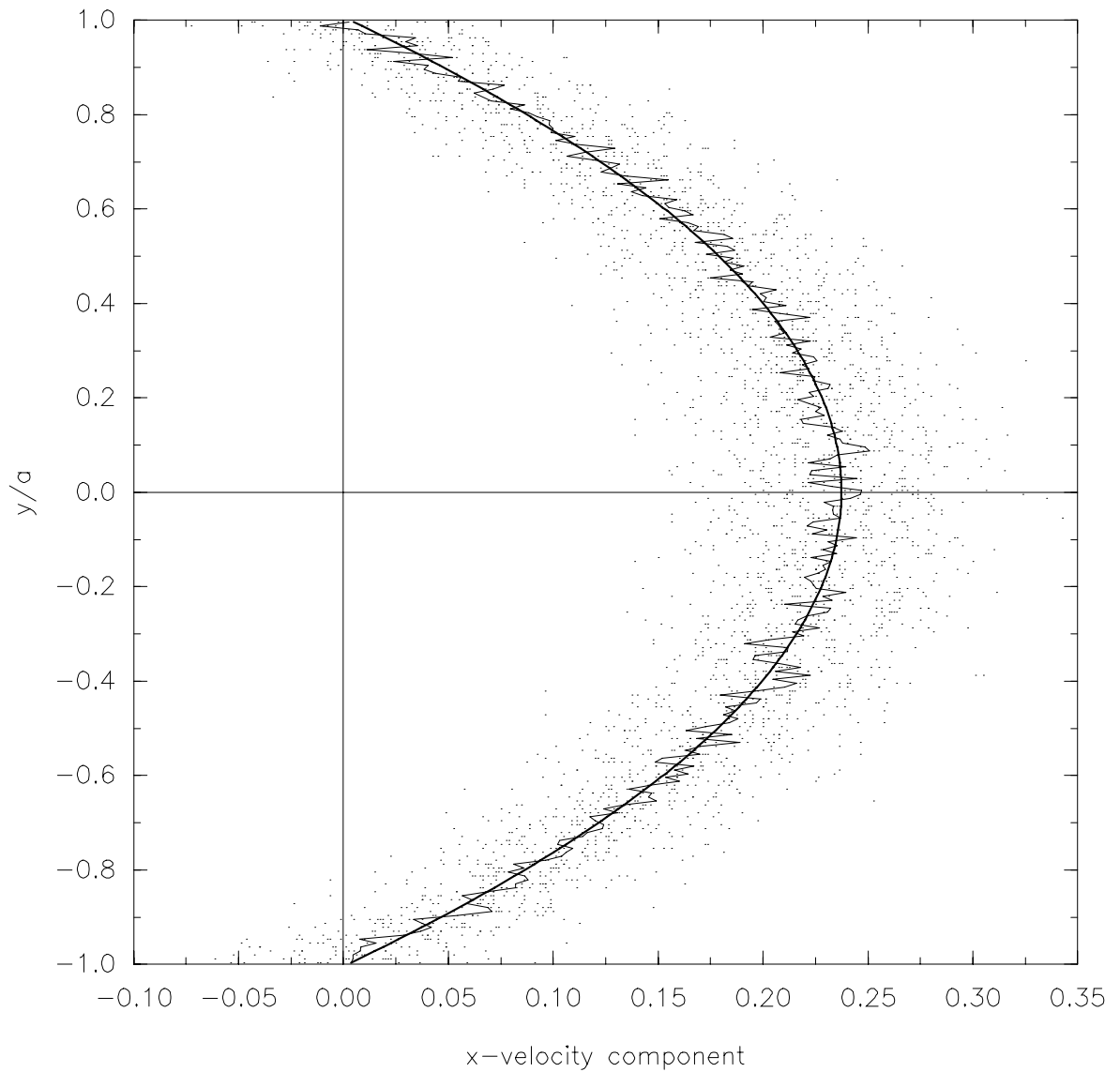
Figure 3.6: Channel mean velocity profile

At $T = 10000$, the mean down-channel velocity components for each of the 12 sections of 20 segments of 12 cells of 20×20 sites are shown as points, with the whole-channel row-averaged velocity shown as the (thin) connected line profile. The best-fit curve, found from linear regression analysis to be a quadratic (with a correlation coefficient of 0.9863), is also shown in bold.
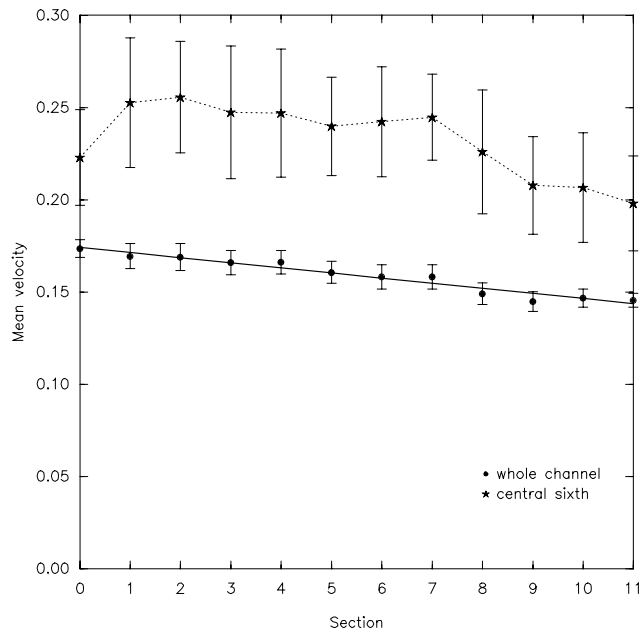
Figure 3.7: Channel and mid-channel mean velocities
At $T = 10000$, the mean channel velocity is seen to decrease linearly for each sectional profile down the channel. The peak down-stream velocity was determined as the mean velocity of the central two cells (where the highest velocities had previously been found). 12 equal sections consist of 20 segments, each containing 240 rows of sites, of which the central 40 were averaged.
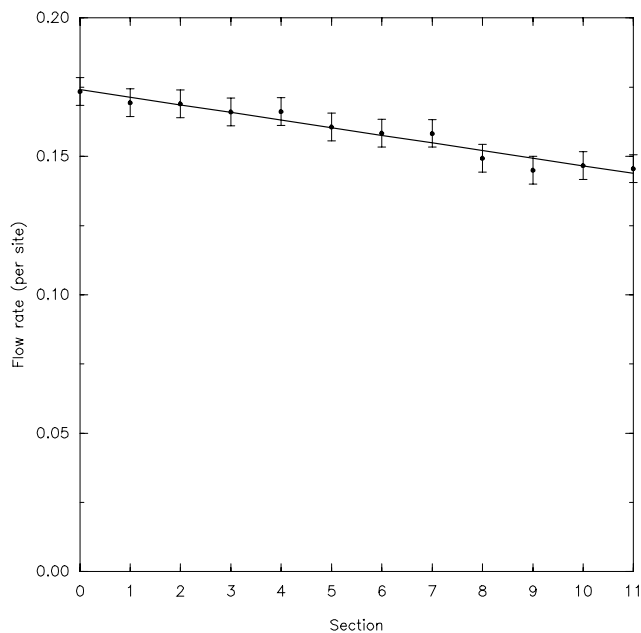
Figure 3.8: Sectional flow rates through channel
At $T = 10000$, the integrated downstream flow in each section decreases linearly with distance through the channel. The best-fit line has a correlation coefficient of 0.975 . Twelve equal sections consist of 20 segments, each containing 240 rows of sites.
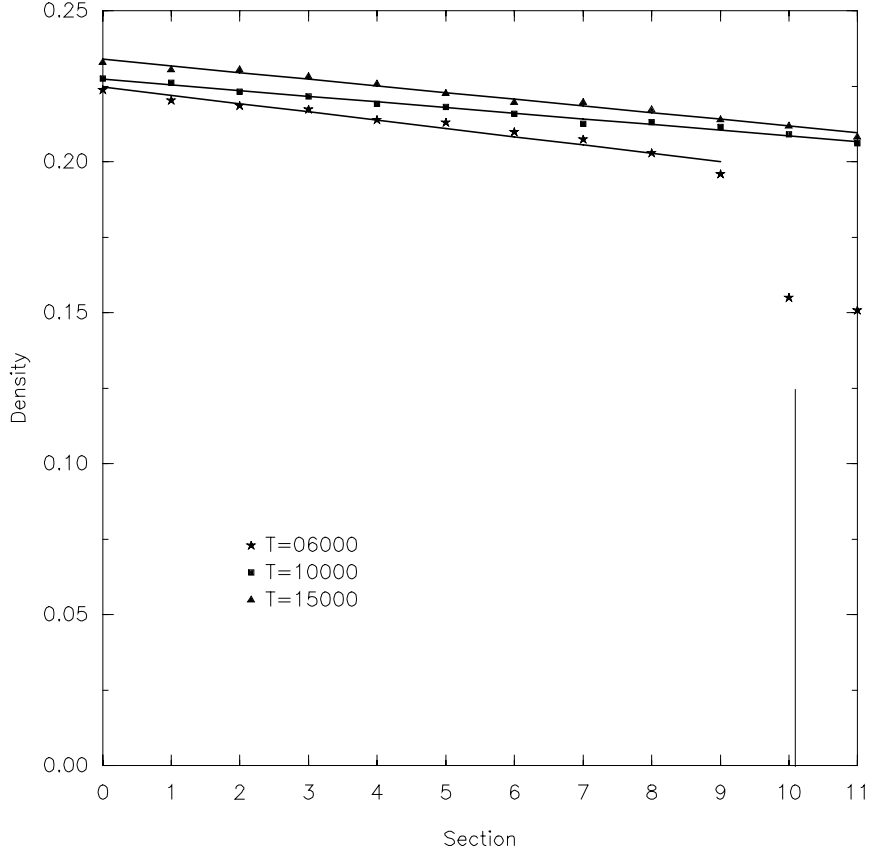
Figure 3.9: Density profiles through channel

The variation of the mean density per section through the channel is shown at three instances. The first instance, $T = 6000$ catches the density discontinuity of the advancing expansion wave (shown by the vertical line). The linear regression on the data can be summarised:

Time	points	gradient	correlation
6000	10	0.0027(2)	-0.975
10000	12	0.0019(1)	-0.994
15000	12	0.0022(1)	-0.993

Braking   Integrating each of the sectional profiles allows the subsidiary effect of the braking of the whole flow by the walls of the channel to be determined. From the linear regression analysis of the data shown in **Figure 3.8** it is found that the cumulative sectional flow rate decreases by 7×10^{-6} per site downstream, from 0.175 per site in the inlet section to 0.146 per site in the outlet section.

Density variation   The density profile of the channel during the course of the simulation is shown in **Figure 3.9**. Apart from the prominent discontinuity of the expansion wave front while it is part of the system, the profiles show the expected behaviour.

The density is found to gradually drop along the channel at a rate of roughly 5×10^{-6} per site downstream, from 0.23 at the inlet to 0.21 at the output. It is also found

to rise slowly with time, as the system equilibrates asymptotically from its rarefied initial state. With the density varying by $\approx 10\%$ over the length of the channel, the incompressibility condition has been stretched.

Reynolds number   The dimensionless quantity which characterises the flow in the channel, the Reynolds number, can be stated as,

$$\mathrm{Re} = \frac{u_{\mathrm{av}}L}{\nu}, \tag{3.1}$$

where u_{av} is the constant downstream average velocity, L is the width of the channel and ν is the kinematic viscosity.

Because of the dual nature of the lattice-gas, the duality factor, $g(\rho)$, must also be included, therefore

$$\mathrm{Re} = g(\rho)\frac{u_{\mathrm{av}}L}{\nu}. \tag{3.2}$$

Considering the measurements of the channel at $T = 10000$: the mean (whole) channel velocity was found to be 0.16(1) and the mean (whole) channel density was 0.217(7). Therefore from **Equation 1.19** the duality factor can be calculated to be 0.42(1) and from **Equation 1.20a** the kinematic shear viscosity is 0.135(4). (The Reynolds coefficient, R^{*}, is 2.04(7).) Since the width of the channel is 240 sites, corresponding to a length of 208 due to the geometry of the lattice, **Equation 3.2** specifies that the Reynolds number in the channel is roughly 100.

Entry length   Where X is the downstream distance from the entry (fluid entering with a uniform speed across the whole cross-section) at which the peak velocity is within 5% of its Poiseuille value, it is experimentally found [Smith60] that

$$\frac{X}{L} \approx \frac{\mathrm{Re}}{30}. \tag{3.3}$$

Therefore, for the channel here, the entry length, $X \approx 700$, i.e. $x = 0.15$, one-seventh of the way down the channel, or within the second section — only section 0 of **Figure 3.5** would show any of signs a square profile, which it does.

3.3.2.3   Summary

As the simulation continues, the flow is always laminar, with no evidence of transitory behaviour, which might have signified the onset of turbulence. Even though the incompressibility condition has been violated, and the peak flow speed of Mach 0.4 has become a significant fraction of the speed of sound, no instability is observed. This is hardly surprising, however, since in Reynold's experiments the critical parameters for turbulent instability corresponded to a Reynolds number of several thousands, while here the Reynolds number is $\mathrm{O}(100)$.

3.4 Obstructed flows

With the simple channel validated, it was possible to incorporate some obstructions by specification of barriers within regions of the lattice. The barriers will restrict the flow by their presence, as well as braking the fluid moving near to their surface through the no-slip particle reflection rule.

As the laminar flow in the channel is deformed in the region of the obstruction, a zone of recirculating fluid, known as a *vortex* or *eddy*, is expected to form — provided the Reynolds number is high enough. Since the Reynolds number measured in the channel was of order 100, and eddies form even for smooth obstacles like cylinders at $Re = 40$ or so, this should be the case. The extent of the eddy is dependent on the strength of the in-channel flow and the size of the obstruction. In fact, a sequence of progressively smaller eddies should be produced, but rarely is it possible to resolve all but the largest two.

For the following simulations, a wider channel was specified, such that it was 128×100 cells in size, with each cell 20 sites in dimension. The increased width allowed the eddies of interest to be resolved better, while minimising the effect of the channel walls. Although, the barriers obstruct a sizable proportional of the width of the channel, this was not generally found to present a problem. In general, the proximity of the downstream sources at the edge of the channel was the limiting factor in the eddy development.

With 20% sources on the upstream edge and 10% sources on the downstream one, the flow velocity is measured to be $0.16(3)$ over the bulk of the channel. The relative proximity of the upstream sources ensures that the flow velocity is uniform across the channel (apart from a small boundary layer of two or three cells, in general). The density is also found to be $0.157(1)$, which specifies that the duality is $0.475(3)$ and the kinematic shear viscosity is $0.193(1)$ — R^* is therefore $1.61(1)$.

3.4.1 Flow past steps

The first obstacle of interest is a trailing-edge step within the channel, leading to its rapid widening. To this end, a large barrier was constructed in the upper part of the channel, one third of the width and height of the channel. A sizable eddy is observed to form downstream of it (see **Plate 2b**). The flow in the channel itself, although diverging to fill the wider region, is not markedly diminished in its speed. The recirculating flow in the eddy itself, however, is much slower.

This behaviour is typical of land-based obstructions, such as buildings, where an area of relative calm, or possible up-draft, will be found in the wind shadow. Generally less pronounced are the calm areas up-wind, naturally foreshortened by the incident draft, and more susceptible to the angle of incidence (in three-dimensional cases).

These two-dimensional steps are representative of rather long three-dimensional

obstructions, lengthwise-on to the incident flow, such that the flow is forced over, rather than around, them. Examples would be high mountain chains, or tree-lined avenues (or 'wind-breaks').

3.4.2   Flow past in-channel obstacles

An obstacle within a channel forces the flow to bifurcate upstream and merge downstream, possibly with eddies. The nature of such flows is generally complex, especially when many obstacles are involved, each responsible for a bifurcation and subsequent merging. When the flow speed is low[6], the fluid has sufficient time to divert at the last moment in front of the obstruction and close right in behind it afterwards. This type of flow is known as *creeping motion* when the symmetry of the obstruction is retained in the flow around it. However, as the flow speed increases, asymmetry will be introduced as the downstream flow is elongated and a downstream eddy (or set of eddies) will grow, and then possibly smaller upstream eddies. Yet higher speed flows produce eddy shedding at the verge of degenerating into fully developed turbulence in the *wake*.

3.4.2.1   Orthogonal barrier

In the case of an orthogonal barrier in mid-stream, the flow is observed to initially split and divert equally above and below the barrier, wrapping round the edges before rejoining into a single stream again. A pair of eddies, symmetric but with rotations in opposite senses, are seen to develop on the trailing edges of the barrier. Initially circular, they become elongated as they grow until they fill the region behind the barrier, and stretch a distance downstream. As the eddies develop further, they will begin to shed in an alternating pattern. With a symmetric barrier the exact phase of the alternating pattern is non-deterministic, dependent on microscopic details such as the position or velocity of a single particle. However, once eddies have started to be shed, the shedding frequency is deterministic. Small stable eddies are also seen to form upstream of the barrier.

3.4.2.2   Inclined barrier

Varying the angle of incidence of the barrier results in differing degrees of asymmetry in resulting eddies. Where an orthogonal (and hence symmetric) barrier produces initially symmetric eddies, before the symmetry is spontaneously broken in the formation of an alternating eddy-shedding régime, the prefered geometry inherent in the oblique barrier ensures that the shedding pattern is deterministic throughout.

For a barrier obstructing one third of the channel width, inclined at an angle of 60° to the flow, some well resolved eddies can be seen in **Plate 3a**.

---

[6]at least for fluids of moderate viscosity, such that the Reynolds number is low.

3.4.2.3   Circular cylinder

This is probably the simplest obstacle to consider analytically. The absence of sharp edges, however, makes it rather less likely that eddies will be produced. A potential problem in attempting to use such a cylinder in eddy-shedding experiments, is the difficulty in sufficiently resolving the eddies, although a pair of fairly well resolved symmetric eddies were found for the flow about a circle in **Plate 3b**. At this time, good correlation is found with [vanDyke82, Plate 42] of the steady state eddies in the wake of a circular cylinder at Re~26. As the flow continues to develop, a von Kármán vortex street of shed eddies is found.

From the circle diameter of 554 sites (or alternatively the cross-sectional width of the inclined barrier previous), the channel flow velocity and the lattice-gas viscosity and duality, the Reynolds number can be calculated to be roughly 200 for these simulations.

3.4.2.4   Triangular prism

Very pronounced downstream eddies are naturally produced by a blunt wedge or (equilateral) triangular prism, with a point upstream, and the base orthogonal. No upstream eddies are produced, since the flow is naturally split by the prism, with the downstream eddies forming at the vertices of the base of the prism. The lack of upstream turbulence, in evidence on flows incident on bluff objects, like the orthogonal barrier discussed earlier, appears to intensify the downstream eddies, making the prism preferable for eddy tracing.

Indeed, [JSME88] claims that:

> Because the frequency of a Kármán vortex street is proportional to the velocity of the flow, a type of vortex street flow meter which measures flow rate by detecting vortices has been put to practical use. Triangular prisms are often used to create the vortices for such flow measurements.
> The vortices ... are clearer than those formed behind a circular cylinder.

It was therefore decided to investigate in detail the eddy-shedding process of a triangular prism and attempt to determine the eddy-shedding frequency. This is done in **Section 3.4.3**.

3.4.2.5   Miscellaneous obstacles

Using the lattice construction primitives of **Section 2.3.6** it is possible to create a diverse variety of obstacles. These objects, when designed and positioned within the computational wind-tunnel of the channel, are part of the process of design and refinement inherent in the prototyping of real-world objects, such as cars and buildings.

Many exotic flows were designed, including that about Joukowski and Kármán-Trefftz aerofoils, and through complex barrier systems, though they are outwith the scope of this work.

3.4.3   Eddy-shedding from a triangular prism

The purpose of this simulation was to track eddies as they grow and are shed for the purpose of determining the eddy shedding characteristics. A system was set up with a channel flow obstructed by a centrally located blunt wedge, towards the upstream end of the channel, allowing plenty of downstream room for the eddies to propagate. It was also interesting to attempt to perform the simulation in conditions which would produce as high a Reynolds number as possible, therefore the channel density was increased to nearer the optimal density of 29%.

A channel consisting of 128×100 cells of 20×20 sites was created, and a flow initialised using upstream sources of 50% — the resulting flow velocity was measured as 0.72(5). Subsequently, a roughly equilateral prism[7], was specified with a base-dimension of 0.18 of the channel width (i.e. 369 sites), and was located with its base one-third of the way down the centre of the channel. The measured density in the channel was 0.304(8), corresponding to a duality factor of 0.329(9) and a kinematic shear viscosity of 0.098(3) — R^* is therefore 2.20(6) which is close to the maximum for the model.

A couple of views from the eddy shedding sequence are shown in **Plate 4**, where the von Kármán vortex street has formed and several eddies can be recognised in the wake. The Reynolds number was calculated as roughly 900 in this simulation, from the combination of the optimal density and a very high flow velocity. It might be expected that compressibility effects would influence the simulation, and indeed a density variation of a factor of two was found in the wake, though no other effects were observed.

3.4.3.1   Eddy shedding mechanism

As the flow developed, the eddy creation and shedding mechanism could be traced. First of all, a pair of small eddies grow symmetrically from just behind the corners of the base of the prism, rotating in opposite senses. This continues until they are of such a size as to be competing for space in the wake. Then, one of the eddies is observed to grow at the expense of the other, such that it grows larger and expands downstream, while the sacrificed eddy shrinks and moves back a short distance towards the prism. The large eddy now eventually reaches a large enough size that it is shed through a combination of the downstream flow about it and the eddy growing again upstream of it. While the second eddy grows there is sufficient space for a smaller contra-rotating eddy to form at the opposite corner to it, which

---

[7]Actually, the measured upstream vertex angle of the prism was 66° while the base vertex angles were 57°.

eventually grows large enough to force the former downstream. This process is observed to then form a repetitive sequence, with several eddies in view in the simulation at a time.

3.4.3.2 Eddy tracking

The position of the centre[8] of the observed eddies, and their sense of rotation, was plotted at regular intervals as the simulation progressed. Eddy centre location was done visually, and measured through alignment of a mouse-based pointer. Since, at some instances, some eddies would be much less well-defined than others, or would be resolved in one coordinate only, occasionally an estimated position would be required. The eddy positions in **Figure 3.10** have been connected by suitable paths.

The times when each particular eddy passed a specific location as it drifted downstream were measured from this graph — the transition point was arbitrarily chosen to conveniently provide the most detailed and accurate information possible from the available data, and corresponded to the point $x = 0.58$.

3.4.3.3 The eddy shedding frequency

Figure 3.11 shows that the eddies pass an arbitrary point downstream at a regular rate — at least after the first couple of eddies have passed and the von Kármán street has formed. This rate is found to correspond to one eddy every 3.35(6) thousand timesteps, i.e. the eddy transition frequency, f, is $2.99(5) \times 10^{-4}$.

This is normally specified by the non-dimensional Strouhal number,

$$\mathrm{St} = \frac{fL}{U} \,. \tag{3.4}$$

Although found to be a weak function of the Reynolds number [AT89], it is approximately 0.2 in the wide range $\mathrm{Re} = 10^3 .. 10^5$, and slightly less at lower Reynolds numbers. For the flow here, it is found to be 0.15(1), in good agreement with experiment.

3.4.4 Miscellaneous

3.4.4.1 Transient behaviour

The initial behaviour observed as the fluid flows around the in-channel obstacles, provides a powerful insight into the mechanisms by which eddies and vortex streets

---

[8]or the *eye* or *core* where the velocity is zero and the vorticity highest
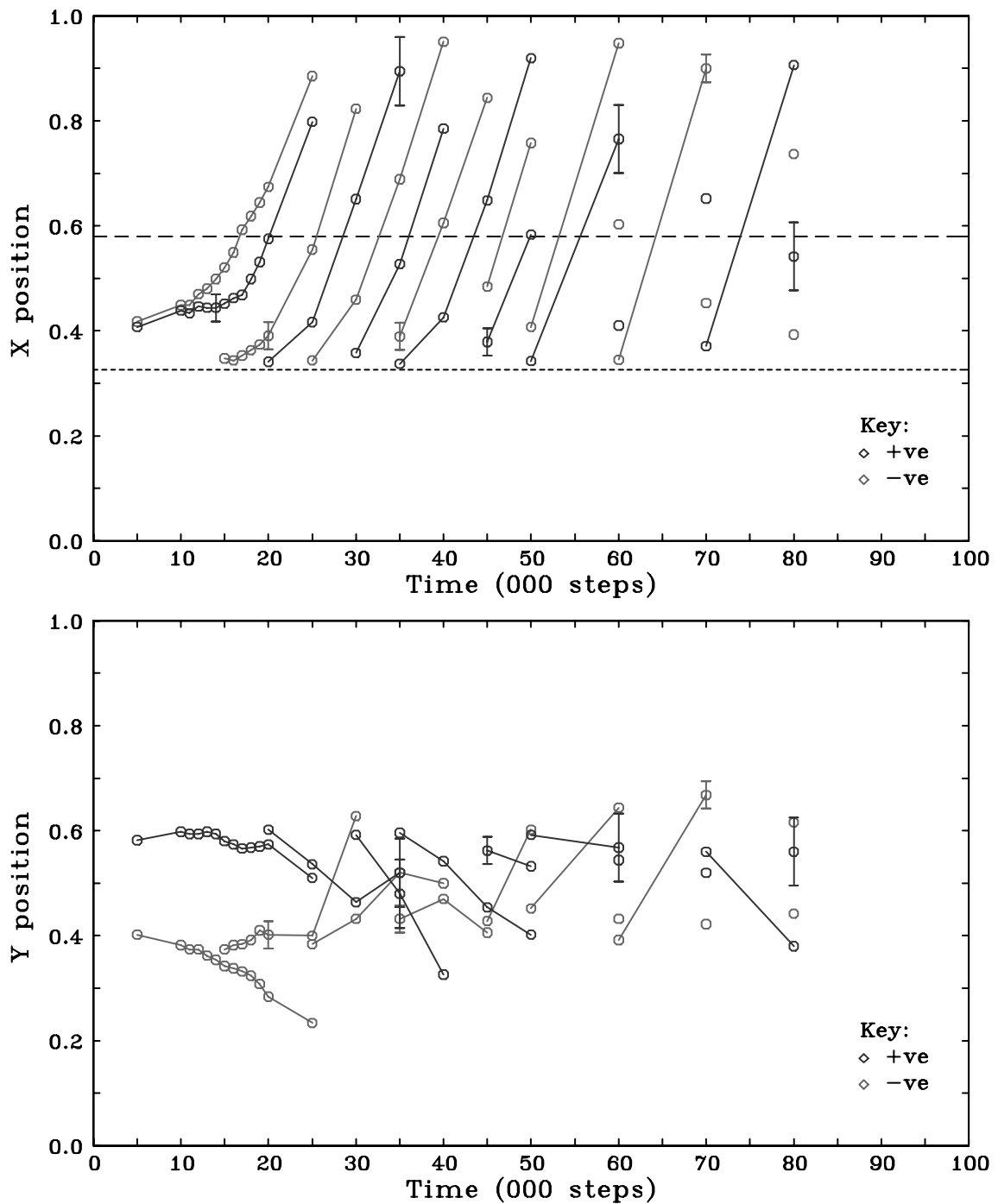
Figure 3.10: Eddy centre positions in wake of triangular prism
The horizontal and vertical positions of centres of the eddies identified in the wake of the triangular prism are plotted, along with an estimate in the error due to poor resolution. Each eddy has its sense of rotation represented by the two shades of gray, and timelines are connected. The position of the base of the prism is shown with short dashes and that of the transition measurements of **Figure 3.11** with long dashes.
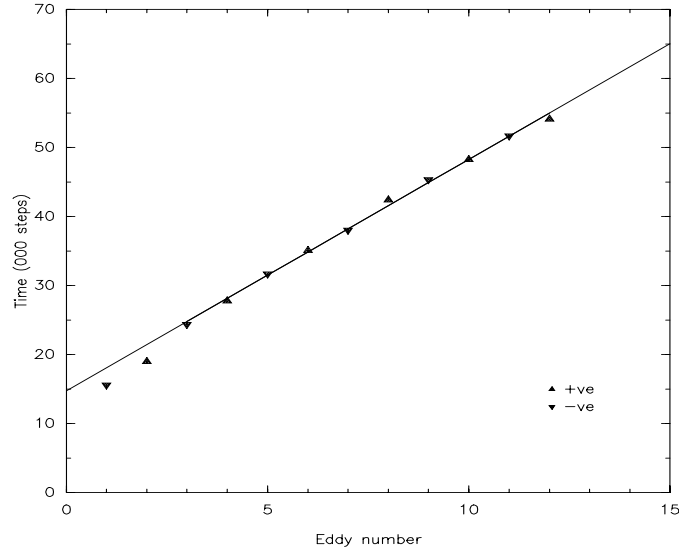
Figure 3.11: Eddy transition frequency
The times when the centres of each of the eddies of **Figure 3.10** pass the point $x = 0.58$ along the channel is shown, along with a marker indicating whether they were positive or negative in their sense of rotation. Ignoring the first two eddies, which were formed before the von Kármán vortex street had developed, the others are seen to lie closely on a line of gradient 3.35(6) thousand timesteps per eddy, with a correlation coefficient of 0.999 .

are formed from laminar flows. This control over the simulation is envied by experimenters who typically have to restrict themselves to steady-state recordings. The transient behaviour of impulsively started objects in a channel is difficult to capture and analyse, therefore most material in the field concerns the steady-state properties, which are shown in numerous plates in [vanDyke82]. Despite the obvious advantages of the triangular wedge for eddy creation and shedding measurements, the majority of the research appears to be primarily limited to circular cylinders and spheres.

3.4.4.2 Obstructions

The obstruction of the flow by the barriers incorporated within the channel is likely to have been slightly affected by the proximity of the bounding channel walls. There was some evidence of this happening when eddies grew in size to be comparable with the width of the channel. In wind-tunnel experiments, it is uncommon for the obstruction to exceed 10%, therefore smaller barriers could have been used, though at the expense of less well resolved eddies.

If a wrap-around boundary had been used, such that particles reaching the upper edge of the lattice were incorporated with no change in their direction at the appropriate site on the lower edge, this would potentially have been less of a problem. The boundary layer next to the walls, where the fluid was moving much slower than in the channel (and the eddy), would certainly have been eliminated.

88

With a few modifications to the collision update algorithm, particularly by tracking the way particles interact with barrier sites, it should be possible to calculate the momentum transfer, and hence the *drag* or *lift* on in-channel barriers. Indeed, the microscopic interpretation of particles colliding with the sites of a barrier and having their momentum reversed and imparted to the barrier is very natural.

Moving barriers are also possible[9], again with only a few modifications. One way of achieving this would be to have a one-site buffer-zone completely surrounding the barrier, within which particles would be cleared prior to repositioning of the barrier. It would be preferable if, for instance, stationary particles were imparted with a unit of momentum in a suitable direction (such as the normal to the adjacent surface), while others had their momentum reversed or halted. Rotation could be handled in a similar way, but general rotations of arbitrary objects in a discrete lattice are rather complicated.

The additional complications of re-distribution of the computation, and its efficient load-balancing, make such "extra features" hazardous in general, and even more so in the case of a multicomputer implementation. However, the lattice-gas itself is no impediment to them, and it may well be the case that they are more easily implemented within a lattice-gas simulation than other types of simulation.

3.5   Jets

One of the strengths of the lattice-gas models is the simplicity of the specification of additional sources, enabling a further variety of flow geometries to be designed. Source sites of any strength can be arbitrarily positioned within the computational box, and each contributes to the overall complex flow.

Even with all this flexibility, the model is guaranteed to be computationally robust, since it is impossible for any particular component (or any combination of components) to reach undefined states. However, in designing the fluid system, the considerations outlined previously, regarding the instabilities likely to be generated though the use of pure sinks, etc, should be borne in mind. Additionally, the use of extra sources can lead to variations in the system parameters (particularly density) which may not have been otherwise expected.

Jet design   A jet of faster moving fluid is naturally produced when a localised area of lattice creates particles in a particular direction only, at a rate faster than that within the fluid within which it is introduced. This can be achieved through specification of a small area of source sites (generally at an edge of the computational box) with a source strength higher than that of the neighbouring sources. Alternatively, the source density may be higher than that of the neighbouring lattice. Either case results in an excess of particles produced moving away from the

---

[9]they could have moved as a result of the incident momentum transfer

source. Increased definition of the direction of the resulting jet can be achieved though additional specification of a small outlet or 'pipe'.

A few simulations were undertaken with the previously defined channel system consisting of 128×100 cells of 20×20 sites, and a flow of speed 0.16(3) as the starting point.

Usually jets are considered entering into a stationary fluid, rather than into one already in motion. Indeed, the case of a fast moving fluid from a pipe exit decelerating to match fluid in a reservoir is covered in [JSME88, figures 106&107]. Since the lattice-gas is not Galilean invariant due to the prefered reference frame of the lattice, a transformation of the velocities is not appropriate in this case to equate the two. The braking effect of the slower moving fluid of the channel on the faster moving fluid of the jet is, however, expected to be similar.

3.5.1 A jet into a channel

Into the centre of the wall of source sites forming the upstream edge of the computational box were specified some additional sources. These were positioned in the central 8 cells, and were of strength 50% compared to the upstream sources of the channel which were 20%. A small pipe around the area extending 5 cells into the channel was also specified to achieve additional definition.

When the system is subsequently set evolving, the flow already resident in the channel experiences a pressure wave from the introduced high pressure fluid, and a well defined jet of faster moving fluid is produced (see **Plate 5**). This jet is partially arrested by the slower fluid around it, and small areas of recirculation observed at the front of the jet as the slower fluid makes way for it to pass.

This jet is observed to continue to progress downstream, still well-defined, but starts to kink slightly and then break up and disperse. Several oscillations are easily identified at the latter time in **Plate 5**. For this time, $T = 7000$, longitudinal and lateral velocity sections through the jet are shown in **Figure 3.12**. The fluid in the jet is seen to be gradually braked from its exit velocity around 0.95(5), though in a slightly erratic manner.

Defining the jet as those cells with flow speeds in excess of the background fluid, i.e. above ~ 0.21, the diffuse head of the jet at this later time has expanded to fill one-third of the width of the channel at 0.7 of the distance downstream. The more well-defined region of the base of the jet has only expanded to twice its original width at a distance 0.4 into the channel, i.e. ~ 0.35 downstream from the mouth of the pipe. It has therefore diverged by $5°$ over this distance.

The behaviour of the jet may be comparable to that observed in physical experiments (e.g. [vanDyke82, plates 166&167], and [Werlé89a, figures 1&2]). More likely, however, is the possibility that the jet is experiencing a two-dimensional anomaly, and the instability results from the jet partitioning the channel into two separate reservoirs, which is not the case in three-dimensional jet situations.

(a) Longitudinal section   (b) Lateral sections
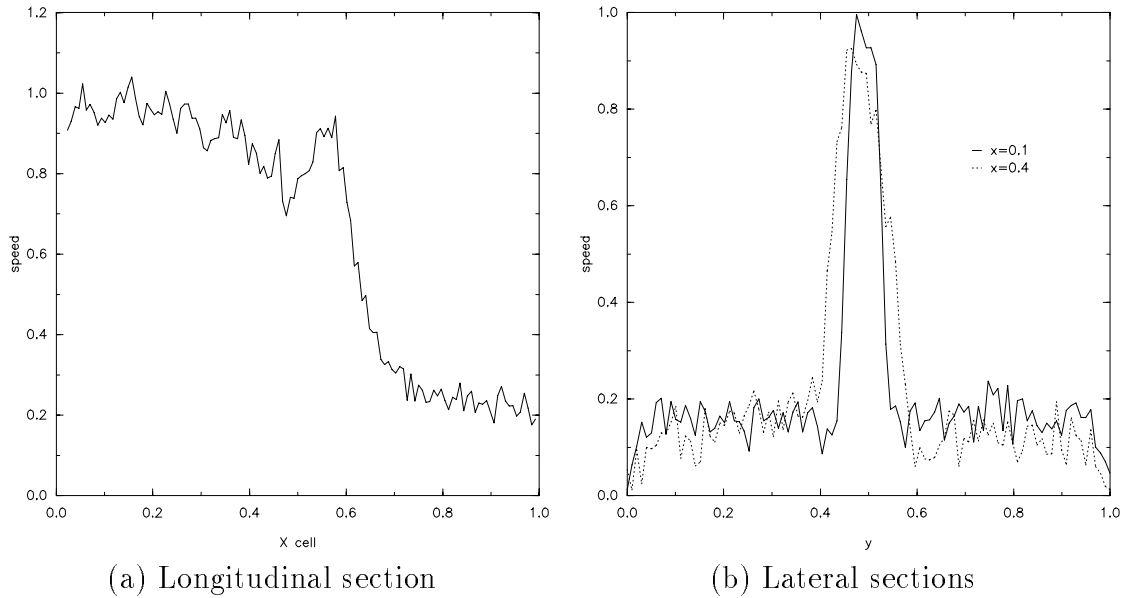
Figure 3.12: Jet velocity sections
A mid-channel longitudinal section (**a**) through the centre of the jet shows the gradual
and somewhat erratic decline in its speed as it extends into the channel, and the more
dramatic drop at the front, while the lateral sections (**b**) show its divergence.

3.5.2   Jets into a cross-flow

As a variation of the jets in the previous section, which were *in-channel*, some
simulations were undertaken with the jet orthogonal to the channel, i.e. *cross-
channel*.

Since the fluid in the jet is being injected orthogonally to the main body of the
channel, it attempts to stream directly across it. However, the effect of the channel
flow will push the jet downstream. The jet being more localised in its composition
will therefore tend to remain together and curve in a downstream arc. If the jet is
strong enough, it will also push fluid across the channel, the downstream region of
the base of the jet also tends to get rarefied, and an eddy can form in this region.

Implementation   The construction of a pipe projecting into the channel, with
sources at its base, as was done in the previous section would *not* form a suitable
starting point for the orthogonal jet. In a three-dimensional simulation the effect
of such a small pipe would have been minimal, since the fluid in the channel would
have been able to quickly divert around it. However, the suggested two-dimensional
equivalent actually corresponds more closely to an infinite barrier partly across the
channel, and would result itself in the spontaneous creation of eddies in its wake,
even in the absence of any flow from the pipe.

A better solution is therefore to set the pipe into the wall at the bottom of the
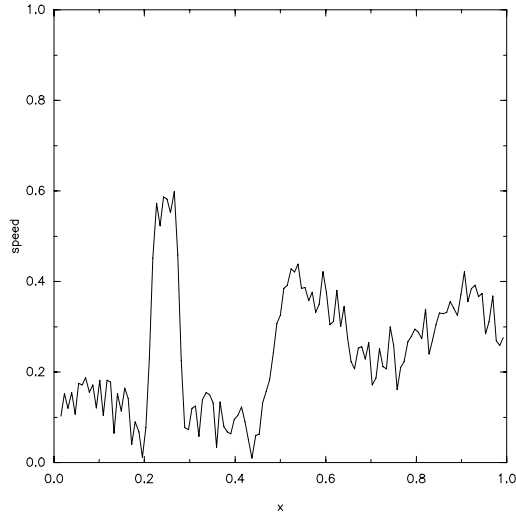
Figure 3.13: Cross-jet velocity section
A longitudinal section through the channel, at $y = 0.10$, cuts the lateral jet from the pipe mouth at $x = 0.25$. The transition from the channel, where the speed is $0.16(3)$, through the jet of speed $0.58(5)$ and the eye of the secondary eddy to the recirculation in the primary eddy is clearly identifiable.

channel, requiring that the bottom of the channel has several layers of wall sites. These are then partly excavated to form the pipe, and a small strip of source sites added to line its base. This arrangement minimised the impact of the pipe, and indeed if the sources at its base were removed it should only be responsible for a small region of stagnation.

Actually, an alternative solution, which was used in the initialisation of the flow in the now-narrower channel, was to *cap* the pipe (or otherwise block it). When properly equilibrated, the cap was removed from the pipe and the flow within the pipe left unimpeded to join the channel. A close-up view of the specification of a similar (though lower resolution) pipe in the microlattice is shown in **Plate 1**.

Observed behaviour   Several experiments were carried out at various flow rates. When the pipe injected fluid at a rate similar to that in the channel (i.e. $u \sim 0.2$) it was found that it very quickly diverted after leaving the pipe and joined the main body of the channel. A small region of stagnation in the shadow of the 'jet' quickly merged into the general calm of the boundary layer along the edge of the channel.

However, increasing the strength of the source in the base of the pipe, correspondingly resulted in increased flow rate from the pipe. This extra flow manifested in a more defined jet which was able to travel a considerable distance into the channel before merging completely with it. For the case of a jet of velocity $u \sim 0.6$ (formed from 35% sources in the pipe), a small, but reasonably well defined eddy is seen to grow in the shadow of the jet, and indeed partially drags the jet around into it (see **Plate 6**). A longitudinal section through the channel, which laterally intersects the jet from the pipe (**Figure 3.13**) shows the jet and the recirculating eddies.

A stronger jet still is able to retain its form and travel further across the channel, and then displays instability within itself before it breaks up and merges into the channel. By this stage, however, the jet has filled the channel and is no doubt affected by the boundary opposite. A much larger eddy is found in the jet's shadow, which grows and expands downstream, allowing a smaller, contra-rotating, eddy to

grow upstream of it. As they travel close to the downstream edge of the computational box, with its downstream sources, they are then also unduly perturbed by the finite size of the box.

3.6   Summary

A number of simulations have been performed using the *FHP7* lattice-gas where its ability to model fundamental and complex flow configurations has been verified; quantitatively against theory where possible, and qualitatively against experimental results otherwise. Universal agreement has been found, which provides considerable confidence in the model and its implementation.

The lattice-gas has also manifested inherently compressible features which appear to have been accurately modelled and not to undermined the integrity of the simulations. Although occasionally the density is observed to fluctuate by more than a few percent (generally, in the vicinity of strong jets, or where the velocity was extremely high) qualitative agreement was good.

In the following chapter, flow experiments are designed which endeavour to quantitatively assess the validity of the *FHP7* lattice-gas in modelling explicitly compressible hydrodynamics phenomena.

4   A study of compressible flow

Contents

Got to do something about where we're going
Step on a steam train
Step out of the driving rain
Maybe run from the darkness in the night

— [U2–88]

When I was a child
I caught a fleeting glimpse
Out of the corner of my eye
I turned to look but it was gone
I cannot put my finger on it now
The child has grown, the dream has gone

And I have become comfortably numb

— [PF79]

4.1 Introduction

QUALITATIVELY compressible behaviour was identified in some of the simulations of the previous chapter, namely the compressions and rarefactions of **Section 3.2.2.1**. In other cases, the flow was indeed compressible, and significant density variations were measured, but qualitative agreement with incompressible flow dynamics, would suggest that the effect of the compressibility was not important.

It is, however, of interest to investigate the suitability of the *FHP7* lattice-gas for modelling two-dimensional compressible flows. To this end, a couple of experiments were designed which would examine quantitatively whether true compressibility effects were accurately captured.

The first experiment corresponds to a shock tube, where the piercing of a partition between two reservoirs produces shock waves which are found to travel isotropically with the speed of sound. The other examines a similar scenario, but instead of the sharp transition from one reservoir to the next, the fluid is directed through a smooth nozzle. In this latter case, it is expected that the fluid will choke in the throat of the nozzle.

[FdHHLPR87] and [Wolfram86a] use a Chapman-Enskog analysis of the microdynamics of the lattice-gas to derive the compressible flow equations, followed by a Mach number expansion in the incompressible limit to get the incompressible equations. From these derivations, the equations of motion have additional terms which suggest that lattice-gases will exhibit anomalous effects in the compressible régime.

To date, none of the proposed lattice-gases have been shown to correctly model a compressible fluid. Why this should be the case has also not been explained, although the lack of Galilean invariance has been suggested [Boghosian89b].

4.2 A pressure wave at a pierced wall

When the dividing partition between two reservoirs, at disparate pressures, is punctured, the system attempts to rapidly equilibrate with a high-speed flow from the reservoir at the higher pressure to the lower one. The initial eruption causes shock waves to propagate from the origin of the puncture, in much the same manner as the circular wave in **Section 3.2.2.1**. A jet may be observed to form later in the process. [John84, Liepmann61]

The computational box was divided in two by the specification of a line of barrier sites across its width. To enable as much detail to be tracked for as long as possible, the partitioning was not done centrally, but slightly to one side, allowing the jet formed in one reservoir to expand without hindrance for a longer time than the other. The two reservoirs were then initialised to different densities, and maintained at such by the inclusion of sources of the appropriate strength at the edges.

When a small part of the central partition was removed, leaving empty lattice, there was a rapid influx of fluid on both sides to fill the gap. However, the pressure difference then further takes effect and an attempt to redress the imbalance results in additional flux of fluid from the high density reservoir to the low, and vice versa. A pair of expansion and compression waves are produced, one on each side and one the reflection of the other. Additionally, a 'jet' is formed extending into the low pressure reservoir from the source of the puncture.

4.2.1 Measurement of the speed of sound

When the partition between the two reservoirs is pierced, two pressure waves are created as they attempt to equilibrate the pressure difference between them. The fronts are initially semi-circular (from the centre of the puncture) until they reach the nearby walls, from whence reflected shock fronts are subsequently found (see, e.g. the early state in **Plate 8a**).

Although the position of the wave front is visually obvious as the limit of the out-flowing velocity, this is rather more difficult to measure since the velocity decreases over a fairly wide range at the front. Its position was more easily measured from the position of the more abrupt (though still quite subtle) discontinuity of the density and pressure at the front, or the dramatic transition of the mean flow direction from random to down-channel. Examples of the velocity, density and pressure maps in one typical case are shown in **Plate 7**.

An example of a sequence of longitudinal sections through the centre of the system is shown in **Figure 4.1**. From the positions of the front measured in this way, for the compression wave in the high-pressure reservoir, as well as the expansion wave in the longer low-pressure reservoir, it was possible to calculate the speed of propagation of the pressure wave front.

Figure 4.2 shows the positions of the wave fronts with time, and from the gradient of the best-fit lines (from linear regression analysis) their speed of propagation was found. From the mean speed of all eight waves, the front velocity is found to be 0.656, with a standard deviation of 3%, in close agreement with the theoretical value for the speed of sound in an *FHP7* lattice-gas, $c_s = \sqrt{\frac{3}{7}} = 0.655$.

Since pressure waves propagate with the speed of sound in the lattice-gas, such simulations provide a direct means to its measurement. It would have been preferable, however, to remove the whole partition and have larger, and better resolved planar waves — as is the situation in a shock tube [Dewey89].

Interesting features are also observed in the jets produced, where e.g. in **Plate 8b**, instead of a single jet forming, two smaller jets are formed outside the main one. In fact, this is completely anomalous, and the fact that the three jets are aligned with the lattice axes, along with the very high velocities recorded in them, suggests that *streaming* is responsible, and the microdynamics are no longer capable of producing correct macrodynamics.
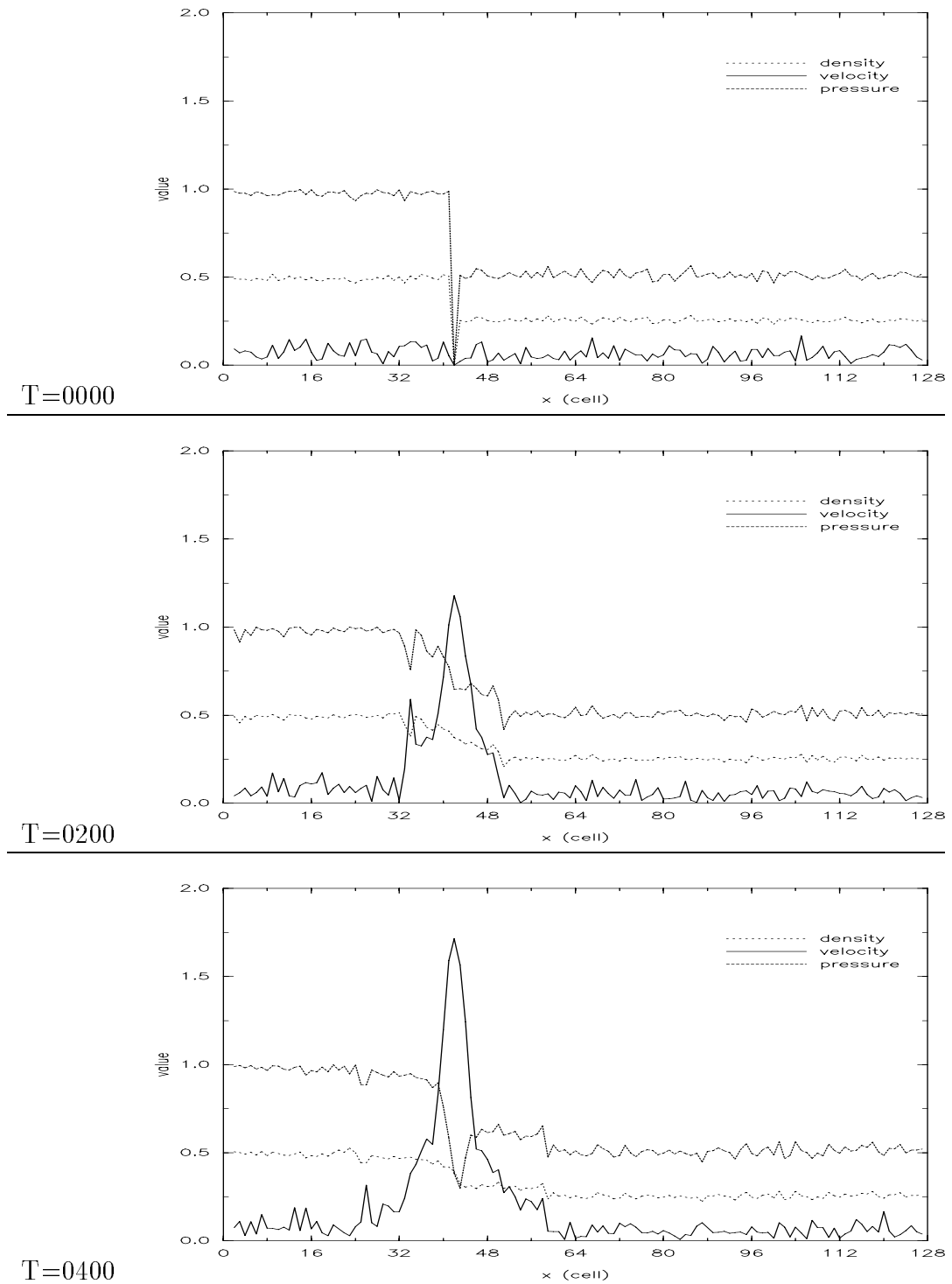
Figure 4.1a: Pierced wall; T=0000,0200,0400

Initially, a sharp discontinuity marks at the position of the partition, where the density and pressure differ by a factor of 2. Velocity fluctuations are noticable, but small. A little later, a growing flux through the gap attempts to equalise the pressure. (**continued**)
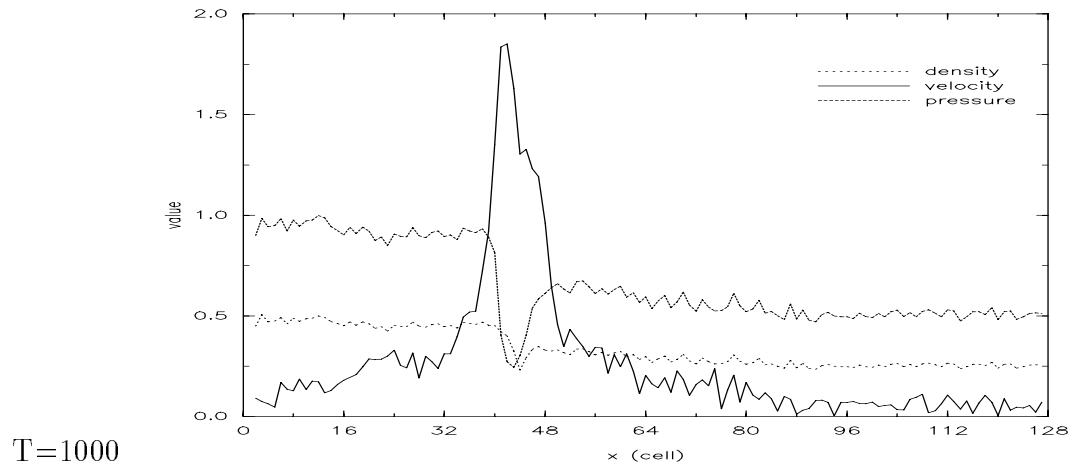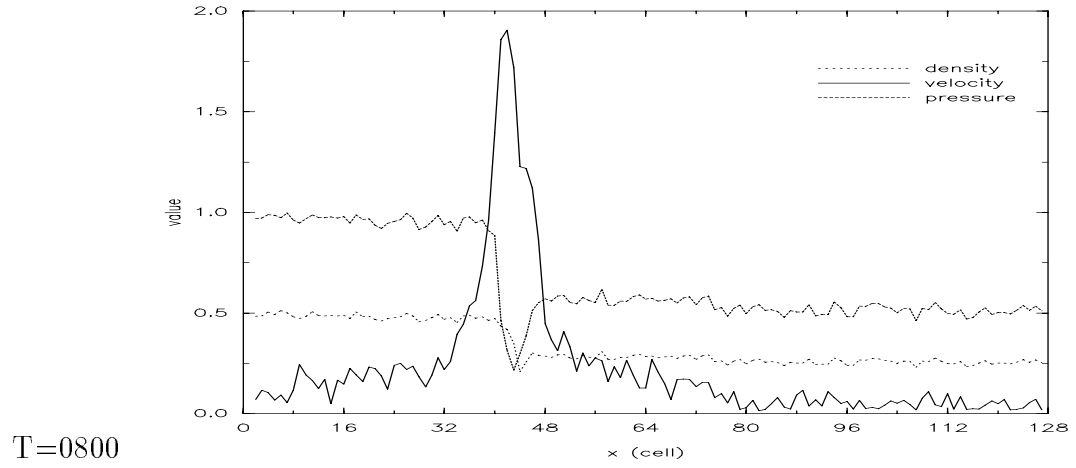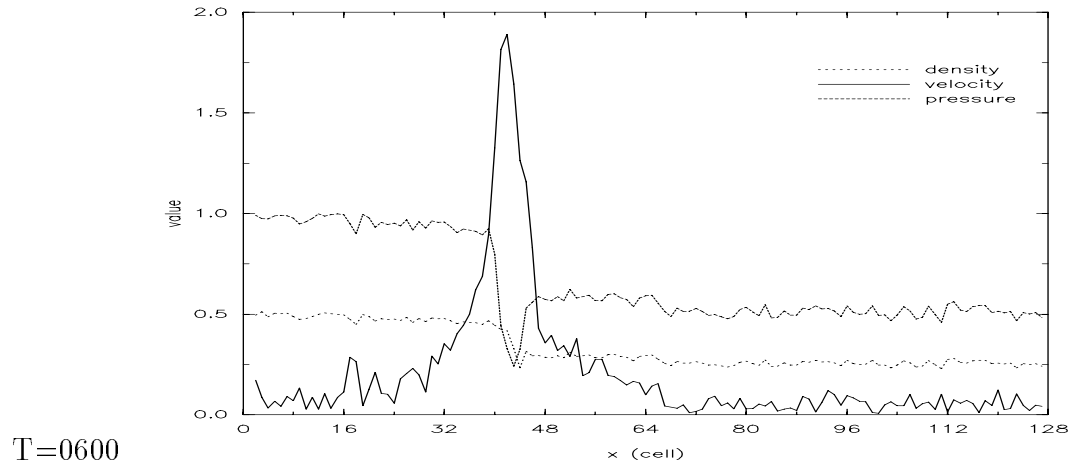
Figure 4.1b: Pierced wall; T=0600,0800,1000
The velocity has peaked in the centre of the gap, and can be seen to be just starting to split into fore and aft jets. (**continued**)
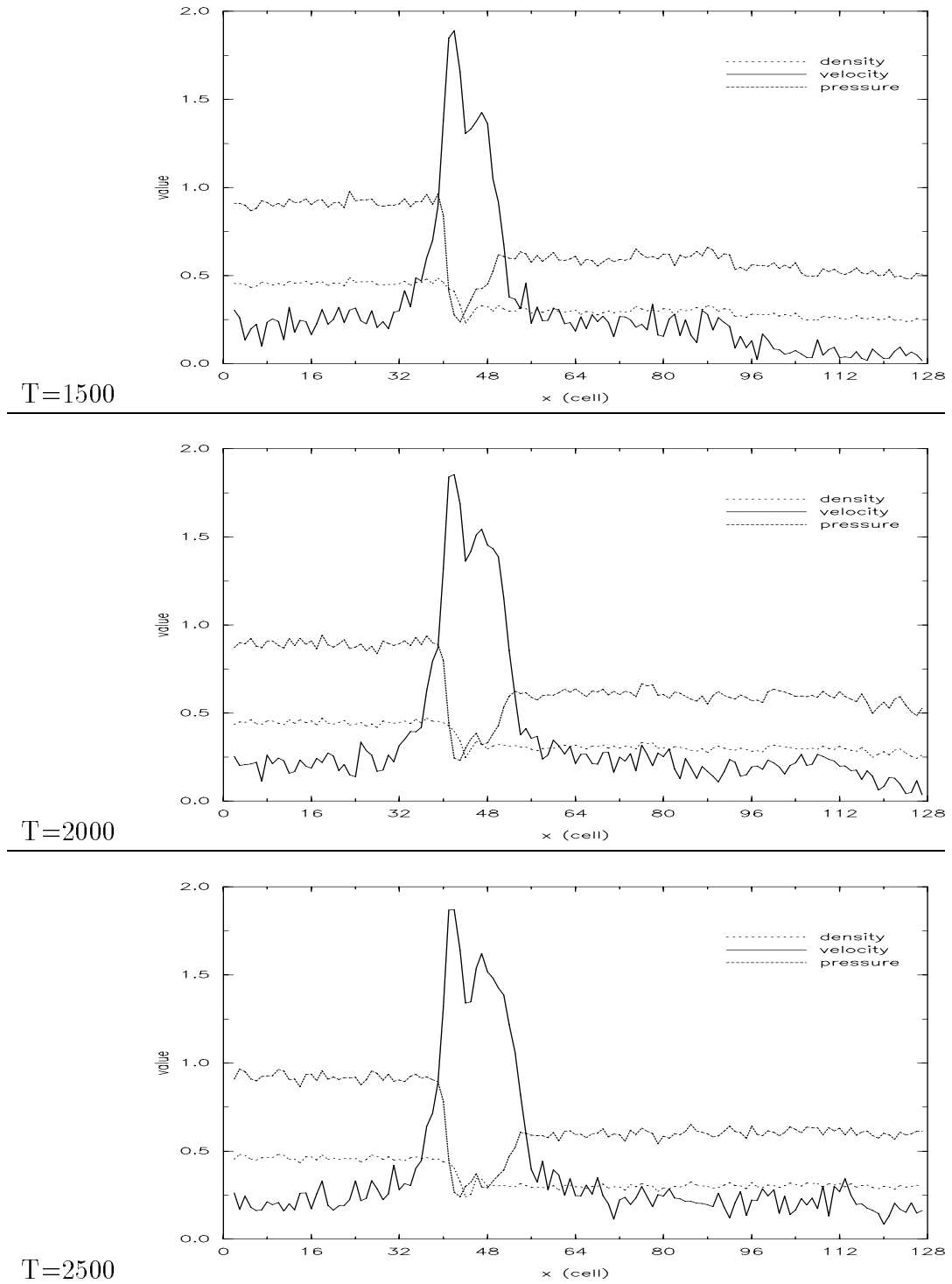
Figure 4.1c: Pierced wall; T=1500,2000,2500
The braking in the middle of the gap is seen as a pronounced dip in the velocity profiles, with a sharper peaked jet upstream and a wider one downstream.

50%–10%

50%–40%

50%–25%

50%–75%
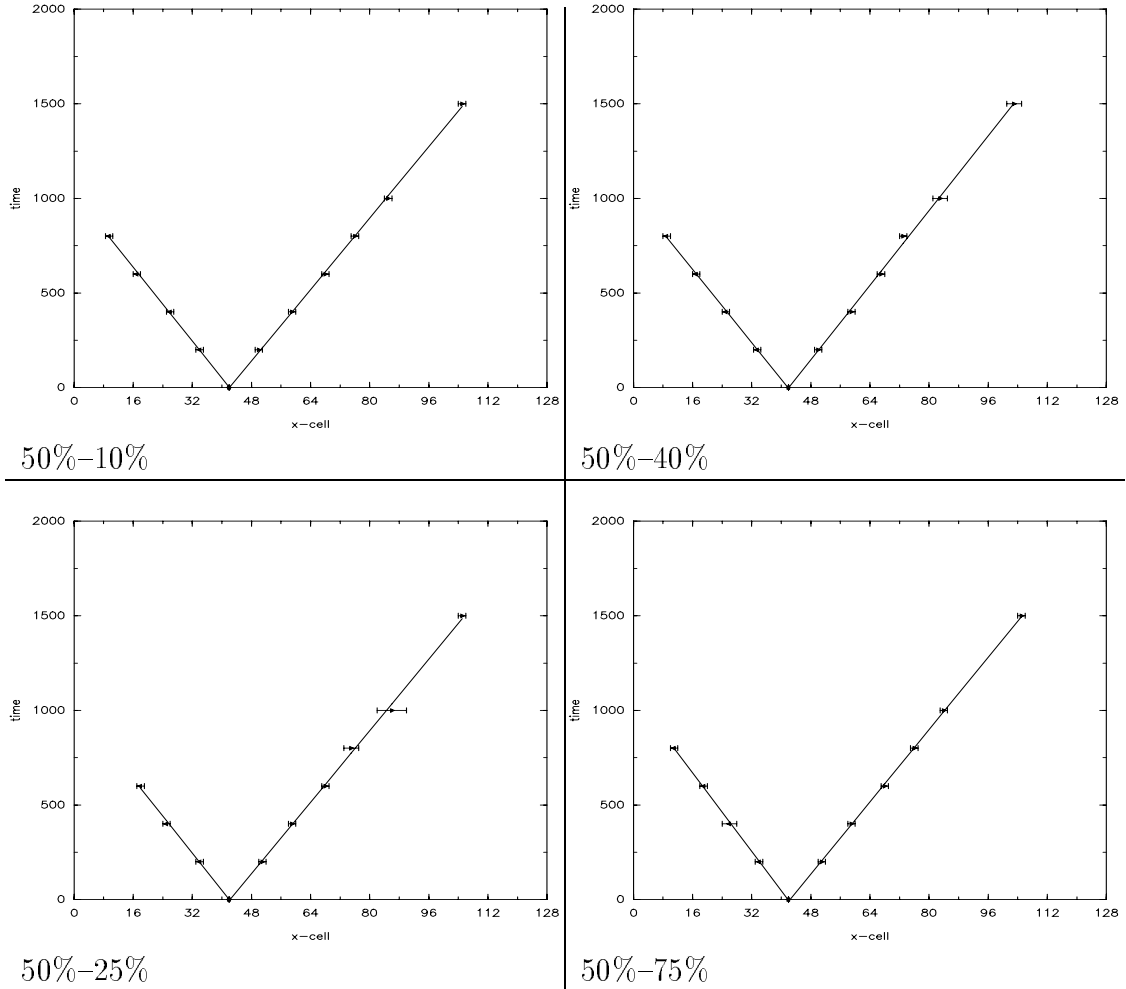
Figure 4.2: Pierced wall: shock front positions

The graphs show the position of the compression and expansion waves from the pierced wall (in cell 42) in four simulations, where the smaller reservoir was kept at a constant density (50%) and the density of the larger reservoir varied. The parameters of the best-fit lines by linear regression analysis are:

Density Ratio	◁			▷		
	gradient	correl.	velocity	gradient	correl.	velocity
50–10	-24.4(3)	-0.9997	0.66(1)	23.6(2)	0.9998	0.68(1)
50–25	-24.6(8)	-0.9989	0.65(2)	23.7(4)	0.9993	0.68(1)
50–40	-24.2(2)	-0.9999	0.66(1)	24.7(4)	0.9993	0.65(1)
50–75	-26.0(3)	-0.9997	0.62(1)	23.9(2)	0.9999	0.67(1)

4.3 Choked flow

This investigation consists of a flow of fluid through the restricted passage of a nozzle, from a reservoir maintained at constant base pressure, to another at a series of lesser pressures. Without the pressure drop, naturally, no net flow occurs.

As the pressure of the lower reservoir falls, the flux through the throat of the nozzle increases, until a point at which the flux cannot increase further — the nozzle is said to be *choked*. When choked, the speed of the fluid through the nozzle can be shown to be identically that of the propagation of sound in the fluid, i.e. Mach 1. If the lower pressure is subsequently reduced, a state of instability will be produced, and a series of shocks in the low pressure reservoir, would be expected as the fluid attempted to equilibrate. [Chorlton67, Pao66]

4.3.1 Analysis of flow in a duct of varying cross-section

The duct   Consider flow from a reservoir at high (and fixed) pressure through a duct, or *nozzle*, of varying cross-sectional area to another reservoir of equal, or lower, pressure. The cross-sectional area of the duct, A, is a function of axial distance, S, as shown in **Figure 4.3**. At the *throat* of the duct the area is a minimum, and $\frac{\partial A}{\partial S} = 0$ at this point. Therefore, $dA = 0$ for a change in S at the throat.
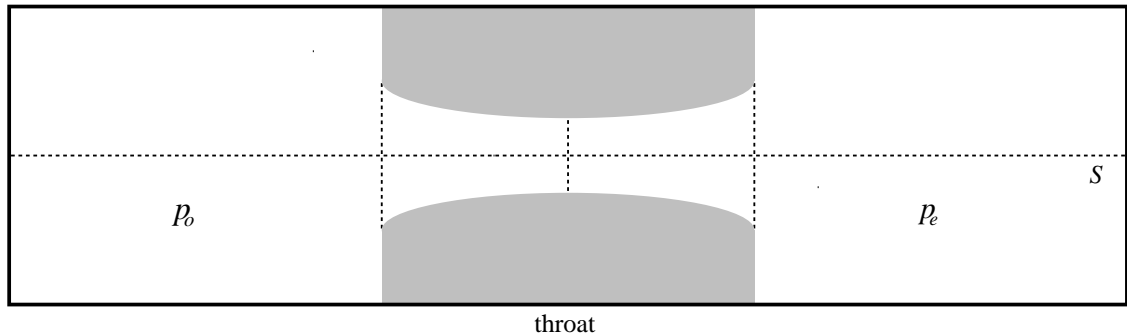
throat

Figure 4.3: Section through smooth nozzle
Two reservoirs, one at the reference pressure p_o and the other at the variable pressure p_e, separated by a constriction of cross-sectional area A in the throat.

Analysis of duct flow   Applying continuity, $\rho A v$ is constant along the duct, therefore,

$$\frac{d\rho}{\rho} + \frac{dA}{A} + \frac{dv}{v} = 0. \tag{4.1}$$

Applying Euler's equation (neglecting gravity),

$$\rho \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p, \tag{4.2}$$

for steady flow,

$$vdv = -\frac{dp}{\rho} = -\frac{dp}{d\rho}\frac{d\rho}{\rho}. \tag{4.3}$$

Since, for an isentropic flow, $\frac{dp}{d\rho} = c_s^2$, where c_s is the speed of sound, then

$$vdv = -c_s^2\frac{d\rho}{\rho}$$

and

$$\frac{d\rho}{\rho} = -\frac{1}{c_s^2}vdv = -\frac{v^2}{c_s^2}\frac{dv}{v} = -M^2\frac{dv}{v}, \tag{4.4}$$

where $M = \frac{v}{c_s}$, the Mach number.

Substituting for $\frac{d\rho}{\rho}$ back into the continuity equation (**Equation 4.1**),

$$\frac{d\rho}{\rho} + \frac{dA}{A} + \frac{dv}{v} = -M^2\frac{dv}{v} + \frac{dv}{v} + \frac{dA}{A} = 0. \tag{4.5}$$

So,

$$\frac{dA}{A} = (M^2 - 1)\frac{dv}{v}, \tag{4.6}$$

and, since at the throat $dA/A = 0$, either $dv = 0$ or $M^2 - 1 = 0 \Rightarrow M = 1$.

Thus, the flow reaches a maximum speed in the throat of the nozzle, and, for compressible fluids, sonic conditions may occur.

Critical parameters It is also possible to show that there is a critical discharge pressure, $p = p_c$, where,

$$p_c = p_o\left(\frac{2}{\gamma + 1}\right)^{\frac{\gamma}{\gamma - 1}}, \tag{4.7}$$

such that the nozzle has a maximum rate of mass flow (throughput). For air, $\gamma = 1.4$ and $p_c = 0.53p_o$.

When p is decreased below p_o the flow starts, increases in speed, and when $M = 1$ at the throat, so also $p = p_c$ (at the throat). If the discharge pressure is further reduced the flow rate does not increase, and the nozzle is said to be *choked*.

4.3.2 System specification

A channel consisting of 128×42 cells of 16×16 sites was specified, allowing the computational box (totalling roughly 1.4 million sites) to be split into three equal (square) sections, with sources on either end. Each segment of the distributed lattice, consisting of a column of 42 cells (and 11 thousand sites) could be allocated to a single processor for updating.[1]

Design of the computational box Within the middle section of the computational box, a nozzle was designed from block and arc lattice-filling primitives. The throat of the nozzle narrowed to eight cells, from entry and exit widths of 16 cells — a constriction of 50%. The left section of the system became the reference pressure reservoir, and the right section became the lower pressure reservoir, where the pressure was dropped throughout the series of experiments.

The series of simulations Three series of simulations were performed with the reference reservoir maintained at a range of different pressures. Pressure itself was not directly controlled, but only indirectly through the particle density, which was maintained by the boundaries on the left and right edges.[2]

In the first series, the base reservoir was maintained at (roughly) 50% populated, being reduced to (roughly) 30% and 10% populated in the other two series. Within each series, the exit reservoir population was decremented from the base reservoir population at a regular rate.

The details are summarised in **Table 4.1**.

Nozzle series	Base reservoir density	Exit reservoir decrement	Actual measured base reservoir ranges	
			density	pressure
50-*	0.50	0.05	0.51–0.43	0.98–0.85
30-*	0.30	0.03	0.30–0.26	0.61–0.52
10-*	0.10	0.01	0.10–0.09	0.20–0.18

Table 4.1: Nozzle series summary
The applied conditions in the base and exit reservoirs for each series are summarised. It can be seen that the density drifts by some 10% through each series.

---

[1] At least, when the full ECS complement of 128 processors were utilised.

[2] Due to the comparable sizes of the reservoirs and the nozzle regions, absolute maintenance of the fixed reservoir pressure was not attempted, and did, in fact, slightly fall throughout the series — however, it was the pressure ratios which were of concern for investigating criticality.

4.3.3   Results

In the first series of experiments, by the time the exit reservoir had been reduced in density from 50% to 20% it was obvious (from the visualisation) that the exit region was increasingly being perturbed by a jet from the nozzle, with no sign of equilibrating. This can be clearly observed in **Plate 9a**. The throat velocity was also observed to have peaked. At this point, a final simulation was done for this series with a free-boundary in the exit reservoir (i.e. 50%–0%) such that the extreme conditions could be investigated, and a similar jet and peak velocity were found.

Similarly, the other series of simulations used exit reservoir pressures decremented in stages from base reservoir densities of 30% and 10% until a similar jet was formed.

4.3.3.1   Profiles

Cell-averaged pressure and velocity were plotted for a section along the flow axis though the centre of the nozzle, and local fluctuations smoothed (by a running average filter). An example of such a section is shown in **Figure 4.4a**.

From these sections, the entry and exit regions were observed to have equilibrated to constant densities and pressures. To get representative measures of the density and pressure of the reservoirs, with as little noise as possible, a cross-section of cells one-eighth of the way distant from each end of the computational box was sampled and averaged, as shown in **Figure 4.4b**.

Longitudinal section profiles of density, velocity and pressure through the centre of the nozzle are shown in **Figure 4.5** for the three sequences of simulations under-taken.

Initial state   It can be seen that initially, when no density (or pressure) gradient is present, the velocity is uniform throughout the section — albeit non-zero. This non-zero velocity corresponds to the 'thermal' velocity of the gas particles, rather than a 'drift' velocity from an imposed flow. Fluctuations in this velocity arising from the fairly small averaging cells, and the fact that no spatial or temporal averaging has been performed are also visible.

From these first cases, the 'thermal' velocities from the Brownian motion of the particles were calculated by averaging throughout the whole section. These were found to be 0.07(4), 0.06(3) and 0.04(2) for the series' `50-*`, `30-*` and `10-*` respectively — the density dependence is to be expected (see **Section 3.2.1**).

Developed state   As the density applied in the exit reservoir is then reduced, it is found that a flow starts through the nozzle, as expected. The velocity is also seen to peak in the throat of the nozzle, with the density smoothly decreasing through the nozzle while remaining fairly constant in the reservoirs as desired.
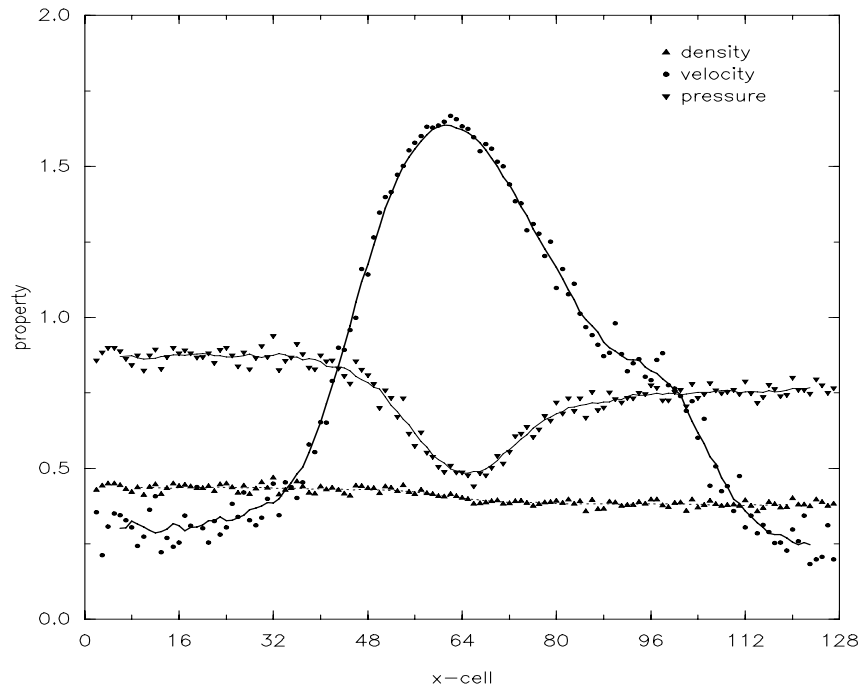
Figure 4.4a: Sample longitudinal section through nozzle: 50%–30%
For a sample simulation, where the applied inlet reservoir density was 50% and the exit reservoir 30%, the density and velocity (and from them the calculated pressure) measurements along the longitudinal section through the centre of the throat of the nozzle are shown. The smoothed curves were produced by a 9-point running average.
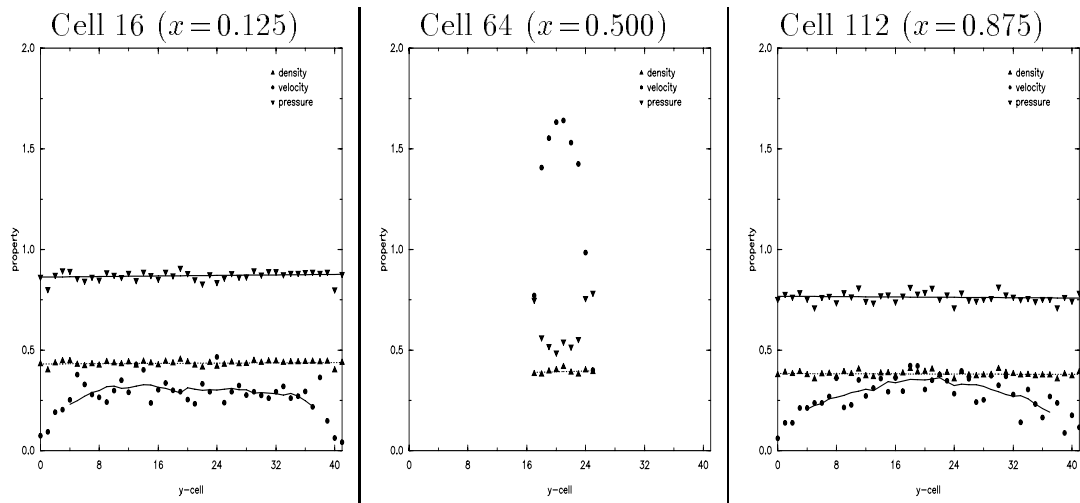
Figure 4.4b: Sample cross-sections of nozzle system: 50%–30%
For the sample simulation, some sample cross-sections show the density and velocity (and from them the calculated pressure) profiles. The central graph corresponds to the throat of the nozzle, with the others well within the reservoirs, where the mean density and pressure were measured.
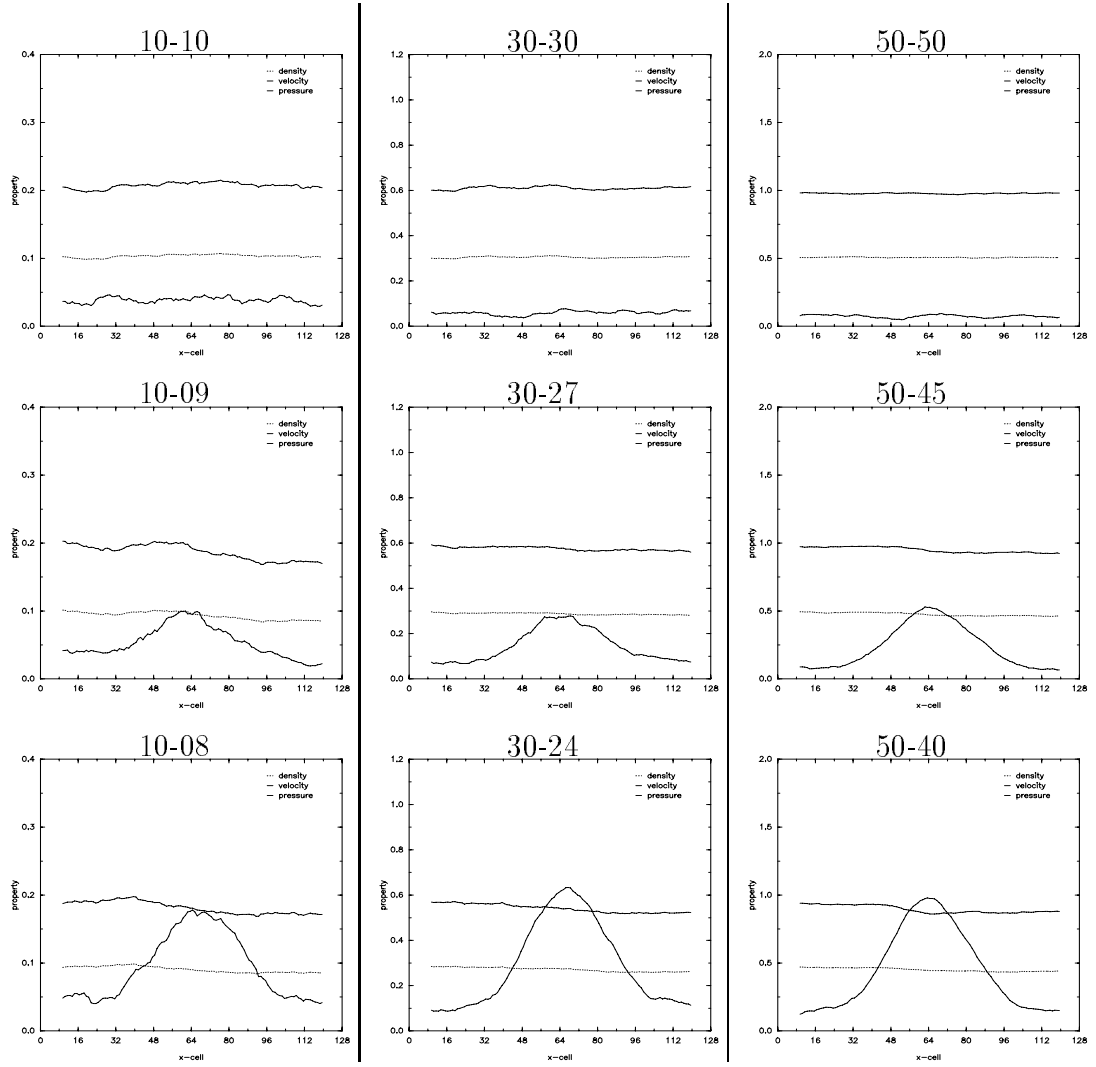
Figure 4.5a: Nozzle sectional profiles (initial)
For each sequence of simulations, at a range of inlet and outlet densities, the density and velocity profiles (and from them the pressure profile also) are shown for the longitudinal section through the nozzle. The symmetry when the reservoirs are balanced is broken when the density is dropped in the right reservoir and a flow starts through the nozzle. **(continued)**
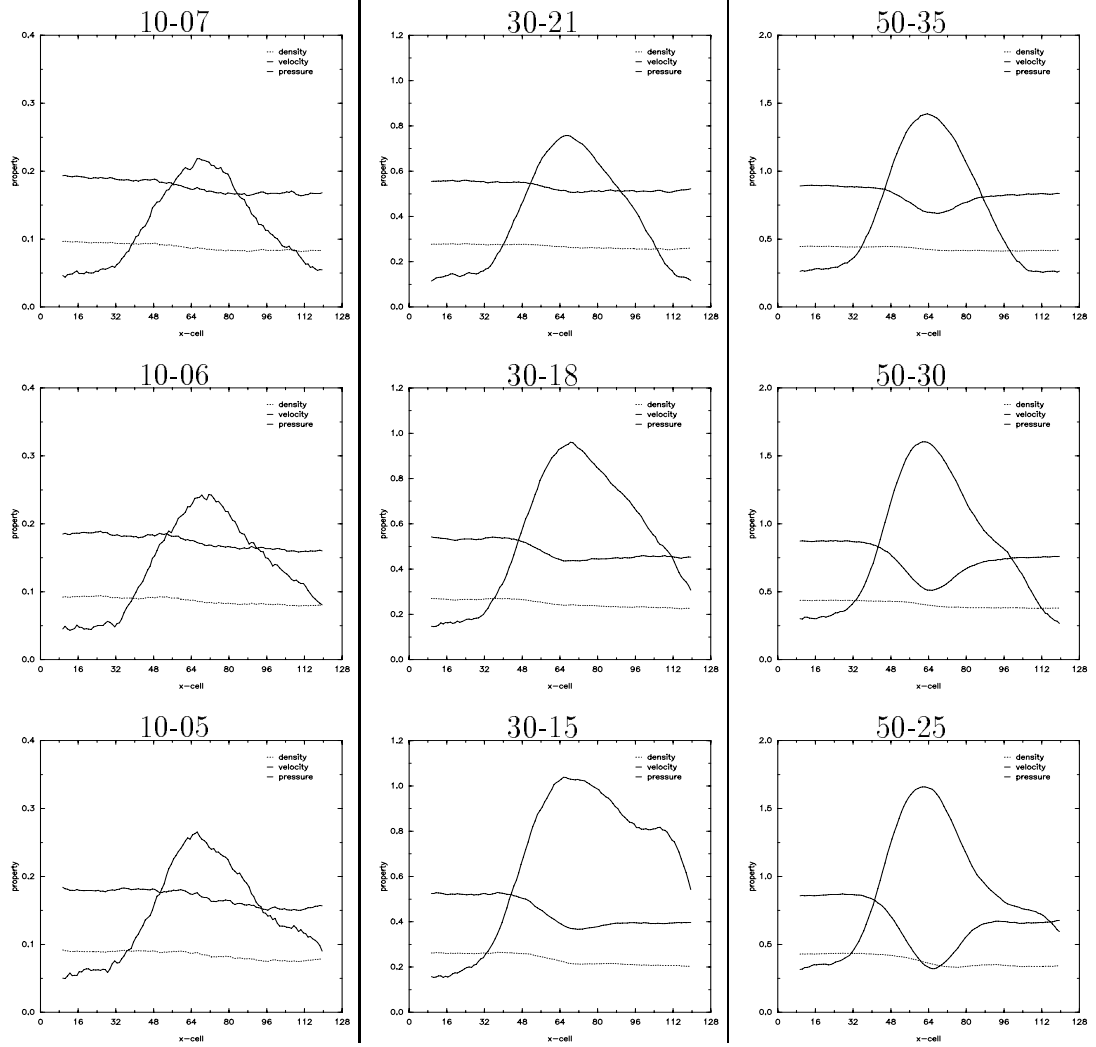
Figure 4.5b: Nozzle sectional profiles (choking)
As the density is dropped further, the velocity peaks in the throat of the nozzle, and a high-velocity jet is formed in the exit reservoir. (**continued**)
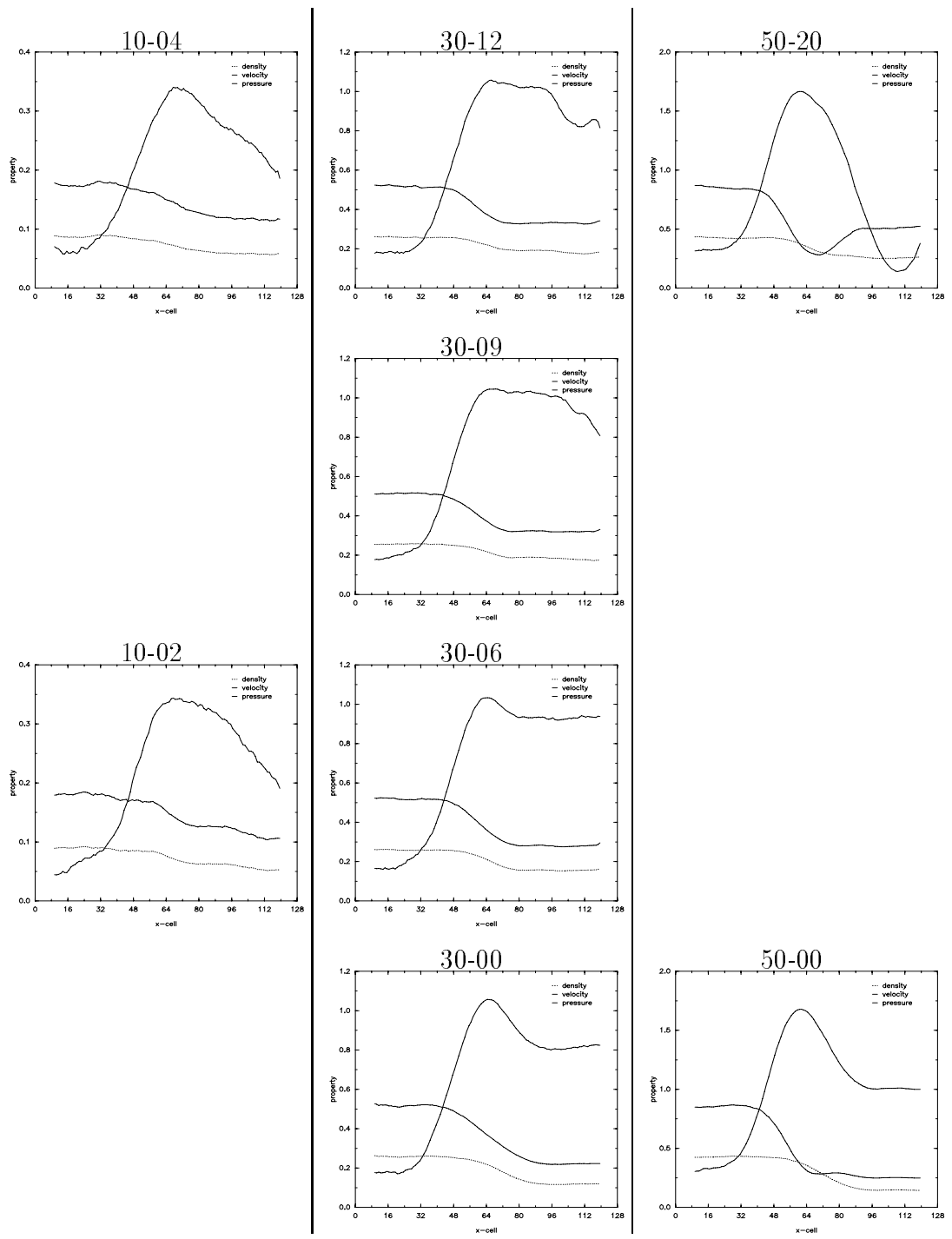
Figure 4.5c: Nozzle sectional profiles (choked)
With the exit reservoir pressure tending to zero, the flow through the nozzle cannot
increase and it is choked. A high-velocity jet thrashes about in the exit reservoir.

Nozzle series	Peak velocity	Mach number	Critical ratio
50-*	1.71(3)	2.61(5)	0.85(2)
30-*	1.08(3)	1.65(5)	0.80(3)
10-*	0.34(2)	0.53(3)	0.73(5)

Table 4.2: Series peak velocities summary
In each series, the measured maximal velocity is compared to the calculated speed of sound for the *FHP7* lattice-gas, $c_s = 0.655$. The critical ratio of reservoir densities (or equivalently pressures) at which the velocity is within 5% of its peak are also provided.

When the flow velocity gets very high[3], as it does in the throat of the nozzle when the pressure drop is sufficient, the calculated pressure is found to dip markedly. In such a case, where particles in the nozzle can be seen to be streaming along the lattice axes, such unphysical behaviour is perhaps to be expected. A close-up of the microlattice in the region of the throat of the nozzle where streaming has occured is shown in **Plate 9b**.

Choked state When the flow velocity cannot increase to compensate for the deficient pressure of the exit reservoir, the nozzle has choked. This occurs when the applied density at the exit wall of the computational box is dropped to be much lower than that in the base reservoir, or indeed reduced to zero, so that no particles are introduced (only removed). An unstable jet is formed, which either streams straight from the exit of the nozzle out of the simulation, or alternatively, it can thrash about wildly.

Therefore, although the density within the exit reservoir is representative of that shown in the profiles, the velocity profile in such cases is generally not. However, the absolute profile is not of primary interest, but rather the peak velocity which occurs reliably in the central core of the nozzle.

4.3.3.2 Maximal velocity

Measurement of the maximal velocity attained from each series was taken from those simulations where the exit reservoir was least dense. The peak nozzle velocities for each series, from the mean of the 9-point (linear) neighbourhood of the peak, are summarised in **Table 4.2**.

It is immediately obvious that the peak velocity is different in each case, and that it is also many times that of the expected speed of sound.

---

[3]approaching the limiting velocity of the lattice-gas, 2

4.3.3.3 Criticality

To determine the critical behaviour of the lattice-gas in the nozzle, the ratios of exit pressure to entry pressure were plotted against the (smoothed) peak velocity from the throat of the nozzle[4] (**Figure 4.6a**). Using the ratios of exit density to entry density, which could be measured directly (and therefore are more confident[5]) a similar plot was performed (**Figure 4.7a**) — the difference in all cases is slight.

The estimated pressure ratios at which criticality is reached are also collated in **Table 4.2**, and are seen to vary with the base pressure.

Comparisons   It is evident that choking has occured in the nozzle in each series, but the critical curves of **Figure 4.6a** are far from co-incident — and a uniform limiting velocity, supposedly that of the speed of sound, is not present. This is not the expected behaviour of physical fluids, even though a vaguely parabolic form is evident.

Even when the velocities of each simulation are scaled by the representative peak velocity from it (see **Figure 4.6b** and **Figure 4.7b**), each simulation has produced a characteristic curve which has a slightly different form, and which doesn't quite fall within the error range of the others.

4.3.3.4 Observations

Choking   These simulations show the phenomenon of *choking* with the throat velocity peaking as the pressure difference is increased. There are, however, several questions raised by certain aspects of the results. Clearly the speed of sound provided by the FHP analysis in the incompressible limit cannot be relied upon in these simulations. However, more disconcerting is not the fact that the high pressure case has a speed of sound much larger than FHP suggest, but that the low pressure case, which is much closer to the incompressible régime is also radically different.

Unusual behaviour   Apart from the velocity magnitude of the peak velocity, the behaviour of the velocity as choking is approached is quite as expected, though doubts about the low and mid pressure-ratio behaviour remain, since the form is quite pronounced and different in the two cases.[6] The high pressure case shows a rather sharp rise leaving equilibrium, and has reached 95% of maximum by a ratio of 0.85, while the low pressure case takes much larger ratios to produce the higher flow velocities. The whole form of the low pressure case is somewhat unsatisfactory, and

---

[4] Error bars on points are from the standard deviation from the mean value arising from the 9-point (linear) neighbourhood of the selected reference segments.

[5] since the derivation of the pressure was not done in the compressible régime

[6] Even allowing for the substantial uncertainties due to noise/fluctuations in the low pressure case.
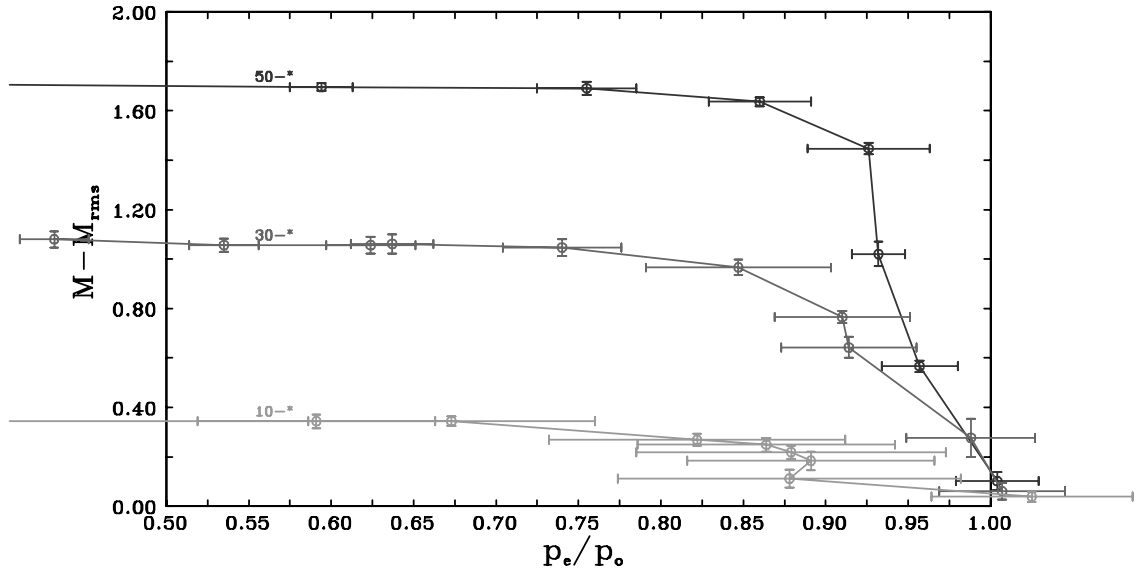
Figure 4.6a: Nozzle peak velocity with pressure ratio (unscaled)
The relative nozzle peak velocity is shown against exit-to-base reservoir pressure ratio, with the velocities unscaled.
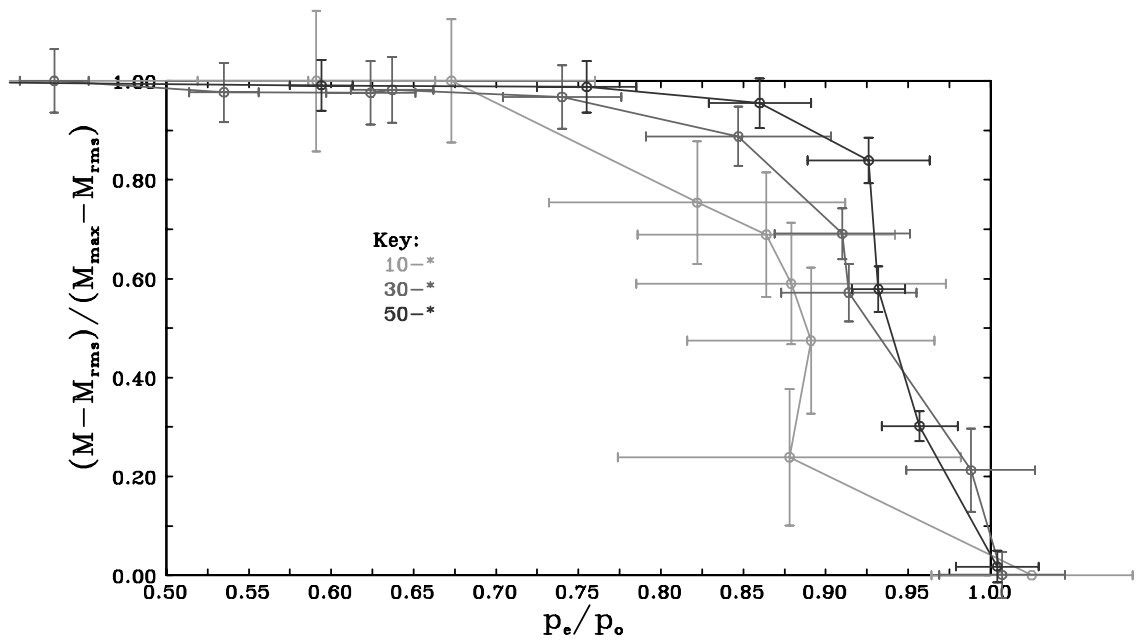
Figure 4.6b: Nozzle peak velocity with pressure ratio (rescaled)
The relative nozzle peak velocity is shown against exit-to-base reservoir pressure ratio, with the velocities rescaled by the maximal velocity.
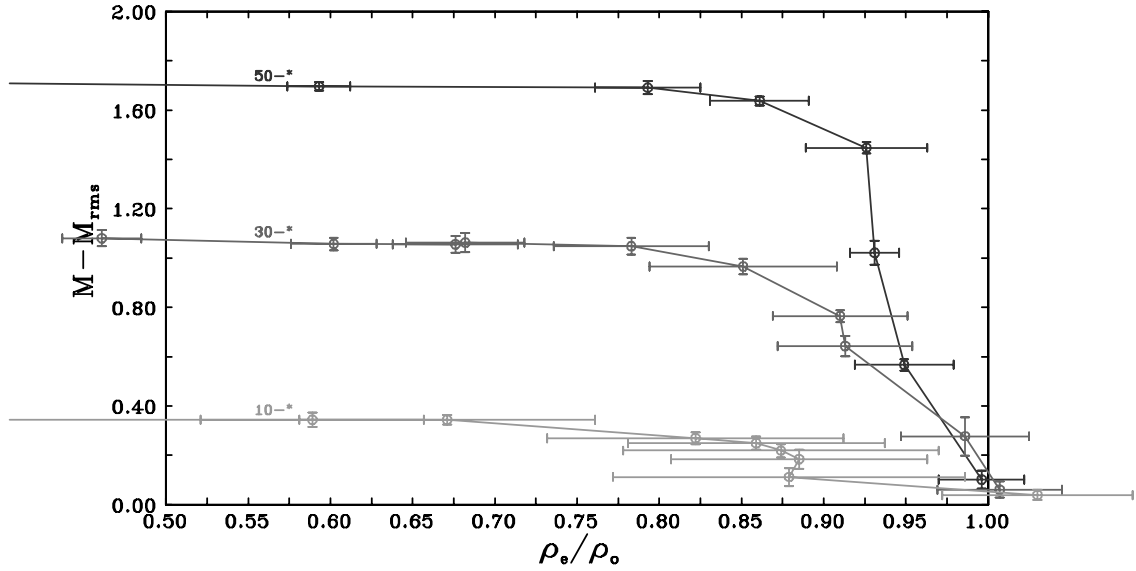
Figure 4.7a: Nozzle peak velocity with density ratio (unscaled)
The relative nozzle peak velocity is shown against exit-to-base reservoir density ratio, with the velocities unscaled.
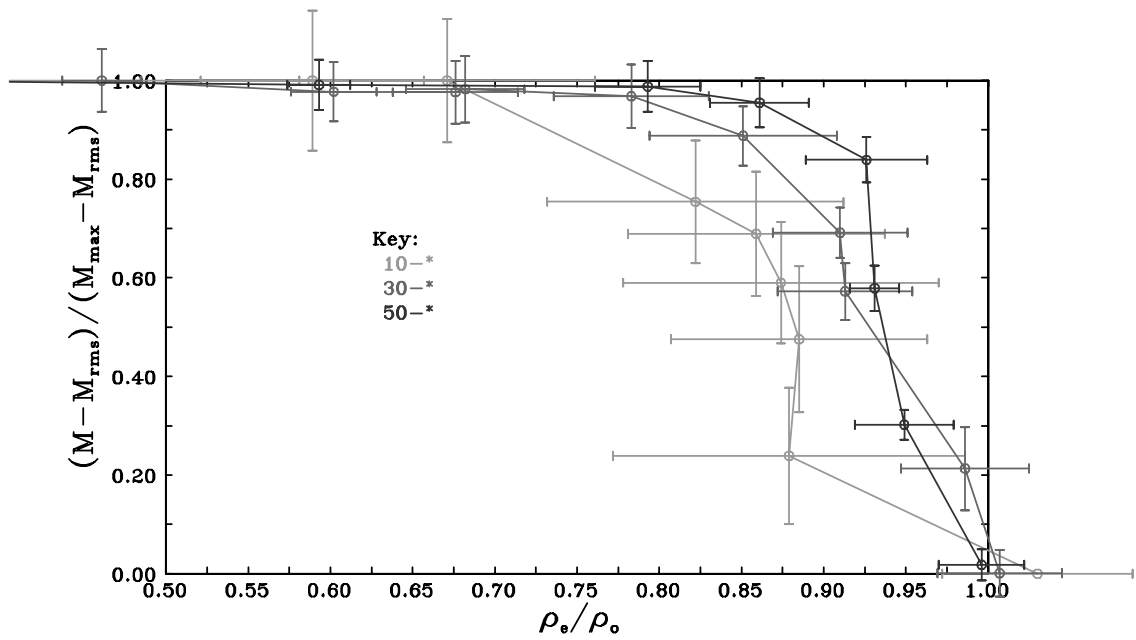
Figure 4.7b: Nozzle peak velocity with density ratio (rescaled)
The relative nozzle peak velocity is shown against exit-to-base reservoir density ratio, with the velocities rescaled by the maximal velocity.

might possibly benefit from a repeat of the simulation, but from the intermediate simulation of the 30% base reservoir, much the same behaviour is found.

The fact that different critical pressure ratios are found would also suggest that some breakdown in the dynamics of the gas has occurred, and that it is failing to model real fluids.

4.4   Comments

Several points can be raised with respect to the compressible simulations undertaken and the phenomena observed.

Incompressible formulae   The use of the formulae derived from the incompressible limit for pressure (and indirectly other things) is rather suspect, especially since the experiments have been designed to extend this régime, and in which the in-built assumptions are going to be violated. It might have been reasonable to expect an alternate derivation of even a pressure-like quantity. This, of course, would require validation against the provided formulae in some *reliable* test flows.

Extreme velocities   The velocity seems to get completely out of control. Examining the model there would appear to be no reason why the maximal velocity would not be precisely 2 lattice-links per time-step.[7] Yet the value measured for the speed of sound in **Section 4.2.1** matches the theoretical value of 0.655 which is wildly different. Experimentally, flow velocities as high as 1.92 have been detected for the throat velocities, which in terms of the speed of sound, corresponds to flows of nearly Mach 3 — it is extremely unlikely that, under these conditions, the lattice-gas is behaving as a real fluid though.

Density dependence   Explanation of the unphysical results of **Section 4.3.3.3** could be through a dependence of the speed of sound on the density. This seems unlikely, however, since the experiments where the speed of sound was measured showed no sign of density dependence. Any expected velocity dependence would be expected to be rather weak (as the square root, probably) and is unlikely to explain the range seen here.

---

[7] From a particle heading along the 1-link along with two particles on the 2&6-links contributing half-a-unit each.

4.5 Conclusion

The shock tube experiments of **Section 4.2** showed that the speed of sound of the lattice-gas was $\sqrt{\frac{3}{7}}$ as predicted by the incompressible analysis of [FdHHLPR87]. Indeed, no density dependence on the speed of sound was identified, which provides quantitative evidence that the model and its assumptions are acceptable.

The same simulations also pointed to a worrying breakdown in the dynamics of the gas, however, when the lattice density variation was extreme, resulting in the observation of streaming. This appeared not to jeoparise the experiment, since the effect was localised, and did not seem to manifest in any deviations from the expected large-scale behaviour.

In the choking experiments of **Section 4.3**, the expected behaviour was qualitatively found, in that the nozzle indeed reached a maximal outlet velocity. However, this peak velocity for choking was not the expected speed of sound, but in fact appeared to be dependent on the density of the base reservoir — or, perhaps, on the applied pressure ratio. Neither of these possibilities is a characteristic of physical fluids.

It is therefore necessary to approach the simulation of compressible fluids through the use of lattice-gases with some caution. Qualitatively compressible features, like compressions and rarefactions are observed, and quantitatively correct behaviour such as the isotropic propagation of shock waves with the speed of sound is also found in special cases. For general compressible situations, however, no quantitative agreement has been found.

From the observations of streaming, it might be concluded that the lack of Galilean invariance resulting from the prefered reference frame of the (static) lattice would be a major factor in the breakdown of the model. However, recent design of Galilean invariant models [dHLS87] has also failed to achieve success with compressible flow simulations.

Although no lattice-gas model is currently known which correctly models compressible flow, it is likely that they will remain an active topic of investigation.

5 Epilogue

And now you're trembling on a rocky ledge
  Staring down into a heartless sea
Can't face life on a razor's edge
  Nothing's what you thought it would be

All of us get lost in the darkness
  Dreamers learn to steer by the stars
All of us do time in the gutter
  Dreamers turn to look at the cars

Turn around and walk the razor's edge

— [Rush89]

L ATTICE-GASES are shown to be a more direct and simple formulation of fluid flow than the traditional approach with partial differential equations. Their strength in the low cost of their highly parallel update, unconditional numerical stability and flexibility in modelling complex flow situations was demonstrated. From the selection of lattice-gases currently actively being researched, the model most suitable for investigation with the available facilities was chosen. This simple model, the *FHP7* variant of the original lattice-gas model of Frisch, Hasslacher and Pomeau shown to exhibit valid hydrodynamics, has indeed "directly spanned the gap between the microscopic and macroscopic worlds." [MTV86]

The basic *FHP7* model was shown in **Chapter 3** to be adequate for a wide range of fundamental and more complex flows. Qualitative agreement with theory or experiment was found to be good, and similarly satisfactory for the cases where only qualitative comparison was possible. Despite exhibiting compressible phenomena, such as compression and rarefaction waves, and modelling shock wave propagation accurately, anomalous behaviour is observed in the simulations of choking in a smooth nozzle of **Chapter 4** — even though the nozzle indeed chokes, quantitatively no agreement with theory is found.

It seems possible that the breakdown in régimes where the flow speed is very high (several multiples of the sound speed) is due to unphysical streaming of particles along lattice axes, though this would be less likely to be important at velocities near the sound speed. Blaming lack of Galilean invariance of the basic *FHP7* model is not the complete story, however, since even the recent Galilean invariant lattice-gas models have not been able to accurately model compressible flows.

The basic lattice-gas models have demonstrated the capability to expand into three-dimensional hydrodynamics and a range of multi-species and other simulations with a great deal of simplicity, however, there are a number of additional avenues of investigation, such as compressible flows and thermal models which remain. While the lack of a useful thermodynamics, and the current restrictions on compressible flows, are indeed limiting, they are unlikely to be more so than typically restrict other simulation techniques, and provide a challenge for innovative model design.

Implementation on the ECS multicomputer, discussed in **Chapter 2**, was found to be straightforward, but a number of important considerations were required to avoid pitfalls which would have resulted in severe load-balancing problems, with the associated disastrous loss of performance. In fact, the implementation was extremely efficient and very well load-balanced.

Although, a transputer-based implementation is unlikely to be optimal from solely a performance evaluation, it does compare favourably when cost-effectiveness is

also important. Considerable flexibility in the construction of the basic model and also in the design of the computational box is possible.[1]

Visualisation of the complex flow systems investigated is found to provide a powerful insight into the underlying processes — even when these are anomalous they are quickly and clearly identified. A variety of techniques applicable to flow visualisation were employed, including vector and magnitude mapping, from which an extensive gallery of images were collated (**Appendix P** is but a small sample).

This thesis has attempted to present a comprehensive investigation of one of the simpler lattice-gas models, identifying the strengths and current limitations common to a growing range of such models, which are being applied to diverse complex dynamics problems. Work with lattice-gases has progressed at considerable pace, since they have clearly shown potential as a cost-effective alternative to traditional CFD techniques, and as such will undoubtedly continue. Lattice-gas models should be viewed as being a complementary simulation technique, rather than a successor, and as such they provide additional choice, and opportunity for comparison with other methods.

---

[1] The use of a Meiko Computing Surface for the lattice-gas simulations described complements nicely the use of fluid dynamical analysis to determine the complex heat transfer and transport within a Computing Surface cabinet performed in [Reynell90]. Although transputers are fairly robust components, supercomputers like the Crays have much more rigorous constraints on thermal build-up and transfer. It is claimed that a recent USAF report identified thermal overstressing as representing a total of 74% of avionics system failures, with vibration accounting for most of the rest. With excessive vibration unlikely to be as much of a problem in earth-bound electronics systems, temperature related failures may be even higher for electronics systems in general. Heat transfer by convection to the cooling fluid (whether air, or something more exotic, like liquid nitrogen) tends to be rather crudely estimated and little or no account is taken of how the (heated) coolant circulates and is extracted.

Appendices

[Corridor, Aft End, Heart of Gold]
> `go aft`
That entrance leads to the Infinite Improbability Drive chamber. It's supposed to be a terribly dangerous area of the ship. Are you sure you want to go in there ?
> `yes`
Absolutely sure ?
> `yes`
I can tell that you don't want to really. You stride away with a spring in your step, wisely leaving the Drive chamber safely behind you. Telegrams arrive from well-wishers in all corners of the galaxy congratulating you on your prudence and wisdom, cheering you up immensely.

— *[AM84]*

B   Bibliography & references

*[Granny Weatherwax] was opposed to books
on strict moral grounds,
since she had heard that many of them
were written by dead people,
and therefore, it stood to reason,
reading them would be as bad as necromancy.*

— [Pratchett87]

[AAE88]   Tim Bricheno, Andy Cousin, Julianne Regan, and Mark Price (All About Eve). 'In The Meadow.' *All About Eve*, Phonogram, 1988.

[Adams79] Douglas Adams. *The Hitchhikers Guide to the Galaxy*. Pan, 1979.

[Adams82] Douglas Adams. *Life, the Universe and Everything*. Pan, 1982.

[Adams87] Douglas Adams. 'How To Leave The Planet.' In *Don't Panic! The Official Hitch-Hiker's Guide to the Galaxy Companion*, by Neil Gaiman, Titan Books, 1987.

[Adobe90] Adobe Systems Inc. *PostScript Language Reference Manual*. Addison-Wesley, 1990.

[AGARD87] Advisory Group for Aerospace Research and Development (NATO). *Special Course on Modern Theoretical and Experimental Approaches to Turbulent Flow Structure and its Modelling*, number AGARD-R-755. AGARD, 92200 Neuilly-sur-Seine, France, August 1987. Material presented as an AGARD Special Course at the von Kármán Institute, Rhode-St-Genèse, Belgium, 16–20 March 1987.

[AJT87]   Hassan Aref, Scott W. Jones, and Grétar Tryggvason. 'On Lagrangian Aspects of Flow Simulation.' *Journal of Complex Systems*, 1(4):544–556, 1987.

[AM84]   Douglas Adams and Steve Meretzsky. *The Hitchhikers Guide to the Galaxy*. Infocom (Interactive Science Fiction), 1984.

[AT89]   Tsuyoshi Asanuma and Yoshimichi Tanida. 'Fluid Dynamics.' In [Yang89], chapter 2, pp 7–28. Hemisphere, 1989.

[BBKPW87] Ken C. Bowler, Alastair D. Bruce, Richard D. Kenway, G. Stuart Pawley, and David J. Wallace. 'Exploiting Highly Concurrent Computers for Physics.' *Physics Today*, pp 40–48, October 1987.

[BCG84]   E. R. Berlekamp, John H. Conway, and R. K. Guy. *Winning Ways for Your Mathematical Plays*, volume 2. Academic Press, 1984.

[BG85]   C. Bennett and G. Grinstein. 'Role of Reversibility in Stabilizing Complex and Nonergodic Behaviour in Locally Interacting Discrete Systems.' *Physical Review Letters*, 55:657, 1985.

[BH90]   W. Berry and J. Haeuser. 'Space Vehicle Aerodynamics.' *European Space Agency Bulletin*, 64:6–15, November 1990.

[Binder87] Phillippe M. Binder. 'Lattice Models of the Lorentz Gas: Physical and Dynamical Properties.' *Journal of Complex Systems*, 1(4):558–573, 1987.

[BKPR87]   Ken C. Bowler, Richard D. Kenway, G. Stuart Pawley, and Duncan Roweth. *An Introduction to occam2 Programming.* Chartwell-Bratt, Sweden, 1987.

[BL88]   Bruce M. Boghosian and C. David Levermore. 'A Deterministic Cellular Automaton with Diffusive Behaviour.' *Journal of Complex Systems*, October 1988.

[BN87]   Jean-Pierre Boon and Alain Noullez. 'Lattice-Gas Hydrodynamics.' In *Special Course on Modern Theoretical and Experimental Approaches to Turbulent Flow Structure and its Modelling*, pages 8.1–8.27. AGARD, 92200 Neuilly-sur-Seine, France, August 1987. Presented as an AGARD Special Course at the von Kármán Institute, Rhode-St-Genèse, Belgium on 20th March 1987.

[Boghosian88] Bruce M. Boghosian. 'Hexagonal (Triangular) Grids.' UseNet, `comp.theory.cell-automata`, February 1988.

[Boghosian89a] Bruce M. Boghosian. 'Navier-Stokes Equations & CA & Auto-CAD.' UseNet, `comp.theory.cell-automata`, October 1989.

[Boghosian89b] Bruce M. Boghosian. 'Problems with lattice-gases in compressible regimes,' March 1989. Personal communication by electronic mail.

[Broadwell64] James E. Broadwell. 'Shock Structure in a Simple Discrete Velocity Gas.' *Physics of Fluids*, 7(8):1243–1247, August 1964.

[BTR89]   Bruce M. Boghosian, Washington Taylor, IV, and Daniel H. Rothman. 'A Cellular Automata Simulation of Two-Phase Flow on the CM-2 Connection Machine Computer.' In J. L. Martin and S. F. Lundstrom, editors, *Science and Applications*, volume VII, pp 34–57. Supercomputing '88, IEEE Computer Society Press, 1989.

[BZ87]   Christopher Burges and Stéphane Zaleski. 'Bouyant Mixtures of Cellular Automaton Gases.' *Journal of Complex Systems*, 1(1):31–50, February 1987.

[CA84]   Interdisciplinary Workshop, Los Alamos. *Cellular Automata* volume 10D of *Physica*. North-Holland, Amsterdam, March 1984. Workshop 7–11 March, 1983.

[CCDL88]   Hudong Chen, Shiyi Chen, Gary Doolen, and Y. C. Lee. 'Simple Lattice-Gas Models for Waves.' *Journal of Complex Systems*, 2(2):259–267, 1988.

[CdH87]   Alain Clouquer and Dominique d'Humières. 'RAP1, a Cellular Automaton Machine for Fluid Dynamics.' *Journal of Complex Systems*, 1(4):585–597, 1987.

[CdHLP86]   Paul Clavin, Dominique d'Humières, Pierre Lallemand, and Yves Pomeau. 'Automates Cellulaires pour les Problèmes à Frontières Libres en Hydrodynamique à Deux et Trois Dimensions.' *Comptes Rendus de l'Académie des Sciences Paris*, 303(13):1169–1174, October 1986.

[Chorlton67] Frank Chorlton. *Textbook of Fluid Dynamics*. Van Rostrand, 1967.

[Chow79]   Chuen-Yen Chow. *An Introduction to Computational Fluid Mechanics*. John Wiley & Sons, 1979.

[Clarke90] Lyndon J. Clarke. *Tiny Version 2 (Computing Surface) Release 0 Overview*. Edinburgh Parallel Computing Centre, 1990. EPCC-UG-25.

[CM87a]   Hudong Chen and William H. Matthaeus. 'Cellular Automaton Formulation of Passive Scalar Dynamics.' *Physics of Fluids*, 30(5):1235–1237, May 1987.

[CM87b]   Hudong Chen and William H. Matthaeus. 'New Cellular Automaton Model for Magnetohydrodynamics.' *Physical Review Letters*, 58(18):1845–1848, May 1987.

[Codd68]   E. F. Codd. *Cellular Automata*. Academic Press, 1968.

[Cogotti89] A. Cogotti. 'Land Vehicles.' In [Yang89], chapter 40, pp 643–658. Hemisphere, 1989.

[Crowder89] J. P. Crowder. 'Tufts.' In [Yang89], chapter 9, pp 125–176. Hemisphere, 1989.

[CT87]   Henri Cabannes and Dang Hong Tiem. 'Exact Solutions for some Discrete Models of the Boltzmann Equation.' *Journal of Complex Systems*, 1(4):574–583, 1987.

[DC70]   J. R. Dorfman and E. G. D. Cohen. 'Velocity Correlation Functions in Two and Three Dimensions.' *Physical Review Letters*, 25(18):1257–1260, November 1970.

[Dettmer86] Roger Dettmer. 'The Artful Transputer.' *Electronics and Power (J. IEE)*, pp 578–582, August 1986.

[Dewey89]   John M. Dewey. 'Explosive Flows: Shock Tubes and Blast Waves.' In [Yang89], chapter 29, pp 481–498. Hemisphere, 1989.

[DH66]   James W. Daily and Donald R. F. Harleman. *Fluid Dynamics*. Addison-Wesley, 1966.

[dHL86a] Dominique d'Humières and Pierre Lallemand. 'Écoulement d'un Gaz sur Réseau dans un Canal Bidimensionnel: Développement du Profil de Poiseuille.' *Comptes Rendus de l'Académie des Sciences Paris*, 302(16):983–988, 1986.

[dHL86b] Dominique d'Humières and Pierre Lallemand. 'Lattice-Gas Automata for Fluid Mechanics.' *Physica*, 140A:326–335, 1986.

[dHL87] Dominique d'Humières and Pierre Lallemand. 'Numerical Simulations of Hydrodynamics with Lattice-Gas Automata in Two Dimensions.' *Journal of Complex Systems*, 1(4):599–632, 1987.

[dHLF86] Dominique d'Humières, Pierre Lallemand, and Uriel Frisch. 'Lattice-Gas Models for 3D Hydrodynamics.' *Europhysics Letters*, 2(4):291–297, August 1986.

[dHLQ89] Dominique d'Humières, Pierre Lallemand, and Yuehong Qian. 'Modèles Monodimensionnels de Gaz sur Réseau, Divergence de la Viscosité.' *Comptes Rendus de l'Académie des Sciences Paris*, 308(7):585–590, February 1989.

[dHLS85] Dominique d'Humières, Pierre Lallemand, and T. Shimomura. 'Lattice-Gas Cellular Automata: A New Experimental Tool for Hydrodynamics.' (Publication status unknown), October 1985.

[dHLS87] Dominique d'Humières, Pierre Lallemand, and Geoffrey Searby. 'Numerical Experiments on Lattice-Gases: Mixtures and Galilean Invariance.' *Journal of Complex Systems*, 1(4):633–647, 1987.

[dHPL88] Dominique d'Humières, Yves Pomeau, and Pierre Lallemand. 'Two Dimensional Hydrodynamics Calculations with a Lattice-Gas.' pp 241–248, 1988. (Unknown origin, probably a proceedings.)

[dHPL85] Dominique d'Humières, Yves Pomeau, and Pierre Lallemand. 'Simulation d'Allées de von Kármán Bidimensionnelles á l'Aide d'un Gaz sur Réseau.' *Comptes Rendus de l'Académie des Sciences Paris*, 301(20):1391–1394, 1985.

[DK84] E. Domany and W. Kinzel. 'Equivalence of Cellular Automata to Ising Models and Directed Percolation.' *Physical Review Letters*, 53:311, 1984.

[dMFL85] A. de Masi, P. A. Ferrari, and J. L. Lebowitz. 'Rigorous Derivation of Reaction-Diffusion Equations with Fluctuations.' *Physical Review Letters*, 55(19):1947–1949, November 1985.

[Doi89] Junta Doi. 'Agriculture.' In [Yang89], chapter 38, pp 627–636. Hemisphere, 1989.

[Dujarric90] C. Dujarric. 'Wind Tunnels for Hermes.' *Reaching for the Skies*, (quarterly of the European Space Agency Directorate of Space Transportation Systems), 1(4):6–7, July 1990.

[Dunning89] Ted Dunning. UseNet, `sci.research`, April 1989.

[Durham88] Tony Durham. 'How a Transputer got Physical in Scotland.' *Computing*, pp 22–23, March 1988.

[EH86] Denis J. Evans and William G. Hoover. 'Flows Far From Equilibrium Via Molecular Dynamics.' *Annual Review of Fluid Mechanics*, 18:243–264, 1986.

[EM83] Denis J. Evans and G. P. Morriss. 'Nonequilibrium Molecular-Dynamics Simulation of Couette Flow in Two-Dimensional Fluids.' *Physical Review Letters*, 51(19):1776–1779, November 1983.

[EM84] Denis J. Evans and G. P. Morriss. 'Nonlinear-Response Theory for Steady Planar Couette Flow.' *Physical Review A*, 30(3):1528–1530, September 1984.

[Enya88] Eithne Ní Bhraonáin (Enya). 'Storms in Africa.' *Watermark*, WEA, 1988.

[FdHHLPR87] Uriel Frisch, Dominique d'Humières, Brosl Hasslacher, Pierre Lallemand, Yves Pomeau, and Jean-Pierre Rivet. 'Lattice-Gas Hydrodynamics in Two and Three Dimensions.' *Journal of Complex Systems*, 1(4):649–707, 1987.

[Feynman88] Richard P. Feynman. 'An Outsider's Inside View of the Challenger Inquiry.' *Physics Today*, pp 26–37, February 1988.

[FHP86] Uriel Frisch, Brosl Hasslacher, and Yves Pomeau. 'Lattice-Gas Automata for the Navier-Stokes Equation.' *Physical Review Letters*, 56(14):1505–1508, April 1986.

[FJLOSW88] G. C. Fox, M. A. Johnson, G. A. Lyzenga, S. W. Otto, J. K. Salmon, and D. W. Walker. 'Solving Problems on Concurrent Processors: Volume 1. General Techniques and Regular Problems.' Prentice-Hall, 1988.

[Flynn72] Michael J. Flynn. 'Some Computer Organisations and Their Effectiveness.' *IEEE Transactions on Computers*, C-21(9):948–960, September 1972.

[Fox89] Geoffrey C. Fox. 'Parallel Computing Comes of Age: Supercomputer Level Parallel Computations at CalTech.' *Concurrency: Practice and Experience*, 1(1):63–103, September 1989.

[FR86] Uriel Frisch and Jean-Pierre Rivet. 'Gaz sur Réseau pour l'Hydrodynamique: Formule de Green-Kubo.' *Comptes Rendus de l'Académie des Sciences Paris*, 303(12):1065–1068, 1986.

[Frank89]   W. Frank. 'Building Aerodynamics.' In [Yang89], chapter 39, pp 637–642. Hemisphere, 1989.

[Freymuth89] Peter Freymuth. 'Vortices.' In [Yang89], chapter 28, pp 459–480. Hemisphere, 1989.

[FvD82]   James Foley and Andries van Dam. *Fundamentals of Interactive Computer Graphics*. Addison-Wesley, 1982.

[Gatignol87] Reneé Gatignol. 'The Hydrodynamical Description for a Discrete Velocity Model of Gas.' *Journal of Complex Systems*, 1(4):704–720, 1987.

[GD87]   Serge Gauthier and Gary D. Doolen. 'Compressible Rayleigh-Benard Spectral Simulations: A Useful Reference Solution.' *Journal of Complex Systems*, 1(4):722–729, 1987.

[GG86]   Martin Grant and J. D. Gunton. 'Cellular Automata, Langevin Equations and Unstable States.' *Physical Review Letters*, 57(16):1970–1973, October 1986.

[GJH85]   G. Grinstein, C. Jayaprakash, and Y. He. 'Statistical Mechanics of Probabilistic Cellular Automata.' *Physical Review Letters*, 55:2527, 1985.

[Grassberger84] P. Grassberger. 'Chaos and Diffusion in Deterministic Cellular Automata.' *Physica 10D*, 10(1/2):52–58, 1984.

[Harkins90] Jim Harkins. UseNet, `comp.lang.c`, 1990.

[Harris66] S. Harris. 'Approach to Equilibrium in a Moderately Dense Discrete Velocity Gas.' *Physics of Fluids*, 9(7):1328–1332, July 1966.

[Hasslacher86] Brosl Hasslacher. 'Modeling Collective Motion.' *LANL Research Highlights, Los Alamos*, pp 74–75, 1986.

[Hayot87]   F. Hayot. 'The Effect of Non-Invariance in Lattice-Gas Automaton One-Dimensional Flow.' *Journal of Complex Systems*, 1(4):748–756, 1987.

[HdPP76]   J. Hardy, O. de Pazzis, and Yves Pomeau. 'Molecular Dynamics of a Classical Lattice-Gas: Transport Properties and Time Correlation Functions.' *Physical Review A*, 13(5):1949–1961, May 1976.

[Hénon87a] Michel Hénon. 'Isometric Collision Rules for the Four-Dimensional FCHC Lattice-Gas.' *Journal of Complex Systems*, 1(3):475–494, 1987.

[Hénon87b] Michel Hénon. 'Viscosity of a Lattice-Gas.' *Journal of Complex Systems*, 1(4):763–789, 1987.

[Hiebeler88] David E. Hiebeler. 'Intro to FHP Hexagonal Lattice-Gas Rule.' UseNet, `comp.theory.cell-automata`, April 1988.

[Hiebeler90] Dave Hiebeler. 'Re: lattice gas on CM ref wanted.' UseNet, `comp.theory.cell-automata`, September 1990.

[Hillis85] W. Daniel Hillis. *The Connection Machine*. MIT Press, 1985.

[Hillis87] W. Daniel Hillis. 'Turbulence in Fluid Dynamics.' *Nature*, 325:299–300, 1987.

[Hillis89] W. Daniel Hillis. 'Richard Feynman and the Connection Machine.' *Physics Today*, pp 78–83, February 1989.

[HJ88] R. W. Hockney and C. R. Jesshope. *Parallel Computers 2 — Architecture, Programming and Algorithms*. Adam Hilger, Bristol, 1988.

[HM87] Tudatsuga Hatori and David Montgomery. 'Transport Coefficients for Magnetohydrodynamic Cellular Automata.' *Journal of Complex Systems*, 1(4):730–747, 1987.

[Hoare85] C. A. R. Hoare *Communicating Sequential Processes*. Prentice Hall, 1985.

[HP72] J. Hardy and Yves Pomeau. 'Thermodynamics and Hydrodynamics for a Modeled Fluid.' *Journal of Mathematical Physics*, 13(7):1042–1051, July 1972.

[HPdP73] J. Hardy, Yves Pomeau, and O. de Pazzis. 'Time Evolution of a Two-Dimensional Model System. I. Invariant States and Time Correlation Functions.' *Journal of Mathematical Physics*, 14(12):1746–1759, December 1973.

[IM82] Bruce Dickinson, Steve Harris, Dave Murray, Adrian Smith, and Clive Burr (Iron Maiden). 'Hallowed Be Thy Name.' *The Number of the Beast*, EMI, 1982.

[Inmos88a] Inmos Ltd. *occam2 Reference Manual*. Prentice Hall, London, 1988.

[Inmos88b] Inmos Ltd. *Transputer Reference Manual*. Prentice Hall, London, 1988.

[John84] James E. A. John. *Gas Dynamics*. Allyn and Bacon Books, 1984.

[Jones90] Geraint Jones. 'Measuring the Busyness of a Transputer.' In *occam User Group Newsletter*, number 12, pp 57–64. Occam User Group Newletter, Inmos, UK, 1990. Contributions from Andy Rabagliati (Inmos), Klaus Zeppenfeld, Michael Goldsmith and others.

[JSME88] Yusaki Nakayama, editor. *Visualized Flow: Fluid Motion in Basic and Engineering Situations Revealed by Flow Visualization*. Pergamon Press, 1988. Compiled by the Japan Society of Mechanical Engineers.

[Kadanoff86] Leo P. Kadanoff. 'On Two Levels.' *Physics Today*, pp 7&9, September 1986.

[KB88]   K. W. Kehr and K. Binder. 'Simulation of Diffusion in Lattice-Gases and Related Kinetic Phenomena.'

[Kerridge88] Jon Kerridge, editor. *8th Technical Meeting.* occam User Group, IOS, Amsterdam, March 1988. Held at Sheffield City Polytechnic, UK.

[KMZ87]   Leo P. Kadanoff, Guy R. McNamara, and Gianluigi Zanetti. 'A Poiseuille Viscometer for Lattice-Gas Automata.' *Journal of Complex Systems*, 1(4):784–794, 1987.

[Kraichnan76] Robert H. Kraichnan. 'Eddy Viscosity in Two and Three Dimensions.' *Journal of the Atmospheric Sciences*, 33:1521–1536, August 1976.

[Kraichnan87] Robert H. Kraichnan. 'Eddy Viscosity and Diffusivity: Exact Formulas and Approximations.' *Journal of Complex Systems*, 1(4):796–810, 1987.

[Lamport86] Leslie Lamport. LaTeX, *A Document Preparation System: User's Guide & Reference Manual.* Addison-Wesley, September 1986.

[Landau79] D. P. Landau. 'Two-Dimensional Lattice-Gas Ising Models.' In Kurt Binder, editor, *Applications of the Monte-Carlo Method in Statistical Physics.* Springer-Verlag, 1979.

[LCF88]   Anthony J. C. Ladd, Michael E. Colvin, and Daan Frenkel. 'Application of Lattice-Gas Cellular Automata to the Brownian Motion of Solids in Suspension.' *Physical Review Letters*, 60(11):975–978, March 1988.

[Liepmann61] Hans W. Liepmann. 'Gas Kinetics and Gas Dynamics of Orifice Flow.' *Journal of Fluid Mechanics*, 10:65–79, 1961.

[Lim88]   Hwa A. Lim. 'Lattice-Gas Automata of Fluid Dynamics for Unsteady Flow.' *Journal of Complex Systems*, 2(1):45–58, 1988.

[LMC86]   M. T. Landahl and E. Mollo-Christensen. *Turbulence and Random Processes in Fluid Mechanics.* 1986.

[Longair84] Malcolm S. Longair. *Theoretical Concepts in Physics.* Cambridge University Press, 1984.

[Magnum82] Bob Catley, Tony Clarkin, Wally Lowe, Kex Gorin, and Mark Stanway (Magnum). 'Sacred Hour.' *Chase The Dragon*, FM/Revolver, 1982.

[Margolus84] Norman Margolus. 'Physics-Like Models of Computation.' *Physica 10D*, 10(1/2):81–95, 1984.

[Marillion87] Fish, Steve Rothery, Mark Kelly, Pete Trewavas, and Ian Mosley (Marillion). 'Sugar Mice.' *Clutching at Straws*, EMI, 1987.

[MD87]   David Montgomery and Gary D. Doolen. 'Two Cellular Automata for Plasma Computations.' *Journal of Complex Systems*, 1(4):850–857, 1987.

[Meiko87] Meiko Ltd. *Computing Surface Reference Manual.* 1987

[Merzkirch89] W. Merzkirch. 'Steaming Birefringence.' In [Yang89], chapter 10, pp 177–180. Hemisphere, 1989.

[MKK87]   Gottfried Mayer-Kress and T. Kurz. 'Dimension Densities for Turbulent Systems with Spatially Decaying Correlation Functions.' *Journal of Complex Systems*, 1(4):840–848, 1987.

[Monti89] R. Monti. 'Thermography.' In [Yang89], chapter 21, pp 331–354. Hemisphere, 1989.

[MT87]   Norman Margolus and Tommaso Toffoli. 'Cellular Automata Machines.' Submitted to *Journal of Complex Systems*, 1987.

[MTV86]   Norman Margolus, Tommaso Toffoli, and Gérard Vichniac. 'Cellular Automata Supercomputers for Fluid Dynamics Modeling.' *Physical Review Letters*, 56(16):1694–1696, April 1986.

[Mueller89] Thomas J. Mueller. 'Smokes.' In [Yang89], chapter 5, pp 45–64. Hemisphere, 1989.

[MW90]   Eric Mackay and Matthew White. 'CAPE — A Cellular Automaton Programming Environment.' Appearing in *Parallel Processing For Fluid Flow*, British Computer Society Parallel Processing Specialist Group, 1990.

[Nakayama89] Yasuki Nakayama. 'Electric Sparks and Electric Discharge.' In [Yang89], chapter 6, pp 65–90. Hemisphere, 1989.

[Neff90]   Tom Neff. UseNet press release 89-7654, `sci.space.shuttle`, 1990.

[Norman89] Michael G. Norman. 'CAPE – A Cellular Automaton Programming Environment.' In *Edinburgh Concurrent Supercomputer Newsletter*, 9, 1989.

[NW]   Bruce Nemnich and Stephen Wolfram. 'Cellular Automaton Fluids 2: Two-Dimensional Hydrodynamics.' Proposed follow-up article to [Wolfram86a].

[OY86]   Steven A. Orszag and Victor Yakhot. 'Reynolds Number Scaling of Cellular Automaton Hydrodynamics.' *Physical Review Letters*, 56(16):1691–1693, April 1986.

[Packard83] Norman H. Packard. 'Complexity of Growing Patterns in Cellular Automata.' In J. Demongeot, E. Coles, and M. Tchuente, editors, *Workshop on Dynamical Behaviour of Automata: Theory and Applications*. Academic Press, September 1983.

[Pao66] Richard H.-F. Pao. *Fluid Dynamics*. C. F. Merill, Ohio, 1966.

[Parnas90] Dave Parnas. *Communications of the ACM*, 33, p636, June 1990.

[Pauwels89] Eddy Pauwels. 'PUSSYCAT: A Parallel Simulation System for Cellular Automata on Transputer.' In Mike Reeve and Steven E. Zenith, editors, *Parallel Processing & Artificial Intelligence*, number 13, pp 233–247. John Wiley & Sons, 1989. Communicating Process Architecture Series, ed. David May.

[PD84] K. Preston and M. Duff. *Modern Cellular Automata: Theory and Applications*. 1984.

[PF79] Roger Waters, Dave Gilmour, Richard Wright, and Nick Mason (Pink Floyd). 'Comfortably Numb.' *The Wall*, Harvest (EMI), 1979.

[PM88] Dick Pountain and David May. *A Tutorial Introduction to occam2*. BSP Professional Books, 1988.

[Pountain84] Dick Pountain. 'Microprocessor Design.' *Byte*, pp 361–366, August 1984.

[Pountain86] Dick Pountain. 'Personal Supercomputers.' *Byte*, pp 363–368, July 1986.

[Pountain88] Dick Pountain. 'T800 & Counting.' *Byte*, November 1988.

[Pountain89] Dick Pountain. 'Occam II.' *Byte*, pp 279–284, October 1989.

[Pountain90] Dick Pountain. 'Virtual Channels: The Next Generation of Transputers.' *Byte*, pp EW3–12, April 1990. Appears in Europe and World section only.

[Pratchett87] Terry Pratchett. *Equal Rites*. Corgi, 1987.

[PS88] Heinz-Otto Peitgen and Dietmar Saute. *The Science of Fractal Images*. Springer-Verlag, 1988.

[PSV89a] Michael Philbert, Jean Surget, and Claude Véret. 'Shadowgraph and Schlieren.' In [Yang89], chapter 12, pp 189–202. Hemisphere, 1989.

[PSV89b] Michel Philbert, Jean Surget, and Claude Véret. 'Interferometry.' In [Yang89], chapter 13, pp 203–210. Hemisphere, 1989.

[PSV89c] Michel Philbert, Jean Surget, and Claude Véret. 'Light Sheet Technique.' In [Yang89], chapter 14, pp 211–218. Hemisphere, 1989.

[PT86]   L. Pietronero and E. Tosati. *Fractals in Physics*. 1986.

[PW85]   Norman H. Packard and Stephen Wolfram. 'Two-Dimensional Cellular Automata.' *Journal of Statistical Physics*, 8(5/6):901–946, 1985.

[Queensrÿche88] Geoff Tate, Chris de Garmo, Michael Wilton, Eddie Jackson, and Scott Rockenfield (Queensrÿche). 'Suite Sister Mary.' *Operation: mindcrime*, EMI-Manhattan, 1988.

[RC86]   D. C. Rapaport and E. Clementi. 'Eddy Formation in Obstructed Fluid Flow: A Molecular Dynamics Study.' *Physical Review Letters*, 57(6):695–698, August 1986.

[Reynolds1883] Osborne Reynolds. 'An experimental investigation of the circumstances which determine whether the motion of water shall be direct or sinuous and of the law of resistance in parallel channels.' *Philos. Trans. R. Soc. London*, 174A:935–982, 1883.

[Reynell90] Michael Reynell. 'Fluid Dynamics Algorithms for Advanced Thermal Analysis.' *Electronic Product Design*, pp 53–56, March 1990.

[Řezníček89] R. Řezníček. 'Surface Tracing Methods.' In [Yang89], chapter 7, pp 91–104. Hemisphere, 1989.

[RF86]   Jean-Pierre Rivet and Uriel Frisch. 'Automates sur Gaz de Réseau dans l'Approximation de Boltzmann.' *Comptes Rendus de l'Académie des Sciences Paris*, 302(6):267–272, 1986.

[RHFdH88] Jean-Pierre Rivet, Michel Hénon, Uriel Frisch, and Dominique d'Humières. 'Simulating Fully Three-Dimensional External Flow by Lattice-Gas Methods.' *Europhysics Letters*, 7(3):231–236, October 1988.

[Rivet87a] Jean-Pierre Rivet. 'Green-Kubo Formalism for Lattice-Gas Hydrodynamics and Monte-Carlo Evaluation of Shear Viscosities.' *Journal of Complex Systems*, 1(4):839–851, 1987.

[Rivet87b] Jean-Pierre Rivet. 'Simulation d'Écoulements Tridimensionnels par la Méthode des Gaz sur Réseaux: Premiers Résultats.' *Comptes Rendus de l'Académie des Sciences Paris*, 305(?):751–756, 1987.

[RK88]   Daniel H. Rothman and Jeffrey M. Keller. 'Immiscible Cellular Automaton Fluids.' *Journal of Statistical Physics*, 52(3/4):1119–1127, 1988.

[Rodrigue82] Garry Rodrigue, editor. *Parallel Computations*. Academic Press, 1982.

[Rothman89] Daniel H. Rothman. 'Negative-Viscosity Lattice-Gases.' Publication status unknown, preprint February 1989.

[Rush89]   Geddy Lee, Alex Lifeson, and Neil Peart (Rush). 'The Pass.' *Presto*, Mercury, 1989.

[Schlichting79] Hermann Schlichting. *Boundary Layer Theory*. McGraw-Hill, New York, 1979.

[Settles89] G. S. Settles. 'Aerospace and Wind Tunnel Testing.' In [Yang89], chapter 25, pp 395–408. Hemisphere, 1989.

[Showcase89] Showcase. 'Computational Fluid Dynamics at Work: A Pictorial Study.' *Supercomputing Review*, 2(7):37–42, July 1989.

[SK89] Robert E. Smith and Robert A. Kudlinski. 'Flow Solutions.' In [Yang89], chapter 22, pp 355–362. Hemisphere, 1989.

[Smith60] A. M. O. Smith. 'Remarks on Transition in a Round Tube.' *Journal of Fluid Mechanics*, 7(4):565–570, 1960.

[Smith90] C. R. Smith. 'Computer-Aided Flow Visualization.' In [Yang89], chapter 24, pp 375–393. Hemisphere, 1989.

[Sopsick89] Frances Sopsick. 'Cellular Automata.' *Parallelogram*, (21):13–16, December 1989.

[SR88] John A. Somers and P. C. Rem. 'A Parallel Cellular Automata Implementation on a Transputer Network for the Simulation of Small Scale Fluid Flow Experiments.' In Gerrit A. van Zee and Johannes G. G. van de Vorst, editors, *Lecture Notes in Computer Science*, number 384. Springer, 1988.

[SR89] John A. Somers and P. C. Rem. 'The Construction of Efficient Collision Tables for Fluid Flow Computations with Cellular Automata (Draft).' Preprint, March 1989.

[SRB89] R. Santos, Daniel Rothman, and Bruce Boghosian. 'Lattice-Gas Studies of Immiscible Two-Phase Flow in Inhomogeneously Wet 2-D Porous Media.' In *NATO Advanced Research Workshop on Lattice-Gas Methods for P.D.E.'s: Theory, Application and Hardware*.

[SSB88] S. Succi, P. Santangelo, and R. Benzi. 'High-Resolution Lattice-Gas Simulation of Two-Dimensional Turbulence.' *Physical Review Letters*, 60(26):2738–2740, June 1988.

[SUSSP87] Richard D. Kenway and G. Stuart Pawley, editors. *Computational Physics*. 32nd Scottish Universities Summer School in Physics, SUSSP Publications, August 1987.

[SW86] James B. Salem and Stephen Wolfram. 'Thermodynamics and Hydrodynamics with Cellular Automata.' In [Wolfram86b], 1986.

[TD86] Christoph Franke, Edgar Froese, and Paul Haslinger (Tangerine Dream). *Underwater Sunlight*, Jive Electro, 1986.

[TM87]   Tommasso Toffoli and Norman Margolus. *Cellular Automaton Machines: A New Environment for Modeling*. MIT Press, Cambridge, MA, 1987.

[Toffoli84] Tommaso Toffoli. 'Cellular Automata as an Alternative to (Rather Than an Approximation of) Differential Equations in Modeling Physics.' *Physica 10D*, 10(1/2):117–127, 1984.

[Tritton85] David J. Tritton. *Physical Fluid Dynamics*. Van Nostrand Reinhold (UK), 1985.

[Turner89]   Judith Axler Turner. 'NASA's Ground-Hugging Right Stuff.' *Supercomputing Review*, 2(7):24–29, July 1989.

[TW91]   Arthur Trew and Greg Wilson, editors. *Past, Present, Parallel: A Survey of Available Parallel Computing Systems*. Springer-Verlag, 1991.

[U2–88]   Bono, The Edge, Adam Clayton, and Larry Mullen, Jnr. (U2). 'Running To Stand Still.' *Rattle and Hum*, film by Phil Joanau, Paramount, 1988, and *The Joshua Tree*, Island, 1987.

[Ultravox82] Chris Cross, Warren Cann, Billy Currie, and Midge Ure (Ultravox). 'Hymn.' *Quartet*, Chrysalis, 1982.

[Upson90]   Craig Upson. 'Volumetric Visualization Techniques.' Presented at *The State of the Art in Computer Graphics*, an ACM International Summer Institute, July 1990.

[USSS90]   United States Secret Service. Report on *Operation Sun Devil*, May 1990.

[vanDyke82] Milton van Dyke. *An Album of Fluid Motion*. Parabolic Press, Stanford, 1982.

[vanSchalkwijk90] Ruben van Schalkwijk (Meiko Benelux). 'Supercomputing on Massively Parallel Systems.' Presented at the *BIRA Conference on Software for Vector and Parallel Computers*, Antwerp, May 1990.

[Vichniac84] Gérard Y. Vichniac. 'Simulating Physics with Cellular Automata.' *Physica D*, 10(1/2):96–116, 1984.

[vonKármán11] Th. von Kármán. 'Über den Mechanismus des Wiederstandes, den ein bewegter Körper in einer Flüssigkeit erfarht.' *Gött. Nachricht.*, 12:509, 1911.

[vonKármán54] Th. von Kármán. *Aerodynamics: Selected Topics in the Light of Their Historical Development*. Cornell Univ. Press, Ithaca, NY, 1954.

[vonNeumann66] John von Neumann. *Theory of Self-Replicating Automata*. University of Illinois Press, 1966.

[vZvdV88] Gerrit A. van Zee and Johannes G. G. van de Vorst, editors. 'Parallel Computing 1988, Shell Conference, Amsterdam, June 1988.' *Lecture Notes in Computer Science*, number 384. Springer, 1988.

[Wayner88] Peter Wayner. 'Modeling Chaos.' *Byte*, pp 253–258, May 1988.

[Werlé89a] H. Werlé. 'Liquids.' In [Yang89], chapter 4, pp 41–44. Hemisphere, 1989.

[Werlé89b] H. Werlé. 'Water Tunnel Testing.' In [Yang89], chapter 26, pp 409–414. Hemisphere, 1989.

[Wexler89] John Wexler. *Concurrent Programming in occam2*. Ellis Horwood, 1989.

[Wilson88a] Greg Wilson. 'Computing in Parallel.' *New Scientist*, pp 54–58, February 1988.

[Wilson88b] Greg Wilson. 'The Life and Times of Cellular Automata.' *New Scientist*, pp 44–47, October 1988.

[Williams89] Roy D. Williams. 'Supersonic Fluid Flow in Parallel with an Unstructured Mesh.' *Concurrency: Practice and Experience*, 1(1):51–62, September 1989.

[WM89a] Matthew White and Eric Mackay. 'CAPE User Guide.' Technical Report EPCC-UG-16, Edinburgh Parallel Computing Centre, 1989.

[WM89b] Matthew White and Eric Mackay. 'CAPE Programmer's Guide.' Technical Report EPCC-UG-17, Edinburgh Parallel Computing Centre, 1989.

[WN90] Matthew White and Michael G. Norman. 'CAPE: Cellular Automata and Beyond ...' Technical Report EPCC-TN-39, Edinburgh Parallel Computing Centre, 1990.

[Wolfe89] Alexander Wolfe. 'Turbulent Times with Computational Fluid Dynamics.' *Supercomputing Review*, 2(7):24–29, July 1989.

[Wolfram83] Stephen Wolfram. 'Some Recent Results and Questions About Cellular Automata.' In J. Demongeot, E. Coles, and M. Tchuente, editors, *Workshop on Dynamical Behaviour of Automata: Theory and Applications*, pp 1–12. Academic Press, September 1983.

[Wolfram84a] Stephen Wolfram. 'Cellular Automata as Models of Complexity.' *Nature*, 311(5985):419–424, October 1984.

[Wolfram84b] Stephen Wolfram. Preface to [CA84], pages vii–xii. From proceedings of Interdisciplinary Workshop on Cellular Automata in Los Alamos, 7–11 March.

[Wolfram84c] Stephen Wolfram. 'Universality and Complexity of Cellular Automata.' *Physica*, 10D:1–35, 1984.

[Wolfram85a] Stephen Wolfram. 'Origins of Randomness in Physical Systems.' *Physical Review Letters*, 55:449, 1985.

[Wolfram85b] Stephen Wolfram. 'Random Sequence Generation by Cellular Automata.' *Advanced Applied Mathematics*, 1985.

[Wolfram85c] Stephen Wolfram. 'Undecidability and Intractability in Theoretical Physics.' *Physical Review Letters*, 54:735, 1985.

[Wolfram86a] Stephen Wolfram. 'Cellular Automaton Fluids 1: Basic Theory.' In *Journal of Statistical Physics*, 45(3/4):471–526, 1986.

[Wolfram86b] Stephen Wolfram. *Theory and Applications of Cellular Automata.* World Scientific Publishing, Singapore, 1986.

[Wolfram87] Stephen Wolfram. 'Cellular Automata and Computing, the Present and the Future.' In *Computational Physics*, pp 429–464. SUSSP Publications, August 1987. (Notes made by R.D. Kenway and G.S. Pawley)

[Wylie88] Brian J. N. Wylie. 'Cellular Automaton Lattice-Gas Hydrodynamics on a Parallel Supercomputer.' In [Kerridge88], pp 205–214. Held at Sheffield City Polytechnic, UK.

[Wylie90a] Brian J. N. Wylie. 'Pulsars: Dynamic Process Contention Modelling & Monitoring.' Technical Report EPCC-TN-36, Edinburgh Parallel Computing Centre, 1990.

[Wylie90b] Brian J. N. Wylie. 'T$\alpha\chi$ – Transputer Activity Tachometer.' Technical Report EPCC-TN-47, Edinburgh Parallel Computing Centre, June 1990.

[Yang89] Wen-Jei Yang. *Handbook of Flow Visualization.* Hemisphere Publishing Corp., 1989.

[YBO86] Victor Yakhot, Bruce J. Bayly, and Steven A. Orszag. 'Analogy Between Hyperscale Transport and Cellular Automaton Fluid Dynamics.' *Physics of Fluids*, 29(7):2025–2027, July 1986.

[YUM86] L. Yaeger, Craig Upson, and R. Myers. 'Combining Physical and Visual Simulation — Creation of the Planet Jupiter for the film "2010".' *ACM SIGGRAPH Computer Graphics*, 20(4):85–93, August 1986.

[ZZTop83] Billy Gibbons, Dusty Hill, and Frank Beard (ZZ Top). 'I Need You Tonight.' *Eliminator*, Warner Bros., 1983.

G Glossary of symbols & terms

. . . the higher-ups at NASA said that the probability
of failure [of a space shuttle] was 1 in 100,000.
. . . if you flew the shuttle every day,
the average time before your first accident would be 300 years . . .

— [Feynman88]

NASA Announcement:
New Deckchair Arrangement
For Space-Station Titanic

— [Neff90]

Symbols

α activity, or busyness, of processor (or network): $\alpha = 1 - \iota$

d mean link (particle) density — the reduced density

$\bar{d}$ mean link 'hole' density: $\bar{d} = (1 - d)$

D dimensionality

c lattice particle speed

$\mathbf{c}_i$ particle velocities

c_s speed of sound

f frequency (e.g., eddy-shedding frequency)

$\mathbf{F}$ body force

$g(\rho)$ duality factor

G   giga-prefix, 'billions', 10^9 or 2^{30}

ι idleness, or inactivity, of processor

i Boolean state index, $i = 0, 1..6$ for *FHP7*

λ mean free path between collisions

k   kilo-prefix, thousands, 10^3 or 2^{10} (1,024)

L characteristic length, e.g. channel or obstacle dimension

m mass

M   mega-prefix, millions, or 10^6 or 2^{20} (1,048,576)

Ma   Mach number; fraction of speed of sound: $\mathrm{Ma} = u/c_s$

N_i local average state population

$O(n)$ of order n

p pressure, or processor index

$\mathbf{p}$ momentum flux: $\mathbf{p} = \rho \mathbf{u}$

P processors

ρ mean site (particle) density

Re   flow-characteristic Reynolds number: $\mathrm{Re} = L\,U/\nu$

R^* model-characteristic Reynolds coefficient: $R^* = c_s g(\rho)/\nu(\rho)$

s lattice segment index (may be local or global)

S lattice segments

St   eddy-shedding characteristic Strouhal number: $\mathrm{St} = f\,L/U$

t time (continuous or stepped)

u speed

u, v, w velocity components

$\mathbf{u}$ velocity

U characteristic speed, e.g. mean channel speed

ν kinematic (shear) viscosity: $\nu = \eta/\rho$

η shear viscosity (may sometimes be kinematic)

ζ bulk viscosity (may sometimes be kinematic)

x, y, z spatial dimensions, or lattice coordinates

Z (square) averaging cell dimension

Terms & jargon

! The *occam* channel output statement, specifying that the following data should be sent over the aforenamed channel, when the process on the other end of the channel is ready to receive (and not before).

? The *occam* channel input statement, specifying that the following data should be read from the aforenamed channel, when the process on the other end of the channel is ready to send (and to wait until it is ready).

2dHLGHCA The two-dimensional hexagonal lattice-gas hydrodynamic cellular automaton code developed in *occam* on the ECS transputer multicomputer for the simulations.

activity The busyness of a processor, or network of processors.

alien language Any non-native language, i.e. with respect to the transputer, *occam* is the native language, while C and Fortran are alien.

ALT The *occam* alternation directive, for selecting input from the channel first ready.

ALU Arithmetic and Logic Unit; that part of a processor which executes the program instructions, issuing directives to other units, such as the FPU or DMA link controllers.

Amdahl's Law The statement that no MIMD computation can achieve a speed-up in excess of the number of processors, since the naturally parallel component of a code can run (at best) N times faster on N processors, but there always exists a sequential component, which can not be parallelised and which means that an overall speed-up of N is never achievable.

asynchrony The state in which all processors can update independently of one another, since they (no longer) require further data, or are being individually serviced by a master processor.

bit Contraction of binary digit; the smallest unit of information in a computer, representing the on or off state of a piece of memory.

bit democracy The endowment of equal importance to each bit in a simulation (such as a lattice-gas), as opposed to

bit hierarchy which rigidly imposes an order of importance to bit, such as in the storage of a floating-point number.

bit-map Term for any pixel-based (as opposed to vector-based) display, or part thereof, where pixels are generally bytes or byte triplets (RGB), rather than strictly single 'bits'.

bit-twiddling Modification of individual bits, usually on a one-by-one basis, or through swapping.

boundary layer The thin layer in which the velocity of the fluid rises from rest at the wall (from the no-slip condition) to the value that corresponds to external frictionless flow.

busyness The fraction of the time which a transputer is found to be active with computation (or engaged in, rather than awaiting, communication), i.e. with processes in its queue.

byte 8 bits; the smallest unit of addressable storage (generally) in a computer.

cell

- Traditionally, the cell in the name cellular automata refers to the lowest level computational entities, i.e. the lattice site.

- Unfortunately, the cell in the lattice-gas sense tends to refer to the macroscopic averaging cell containing a number of sites.

cellular automata Simple processes, uniformed spaced on a regular lattice, and regularly updated simultaneously depending on their current state and that of (a selection of) their immediate neighbours. Collectively capable of extremely complex behaviour.

CFD Computational Fluid Dynamics: collectively the computational techniques applied to approximate and solve the equations of fluid motion. Currently utilises one-third of all supercomputer time in the world, for vehicle and building aerodynamics and stability studies amongst others.

channel

occam The mechanism by which information is communicated between a pair of processes (and by further abstraction processors) in the CSP model, introducing synchronisation between them.

collision A path by which momentum is transfered between particles in a lattice site update, producing an outgoing configuration from an incoming one.

debug A Meiko-specific, low-bandwidth, independent route for (usually) debugging messages from each transputer in the network. Generally used when remote processors in a network can not directly access the screen and keyboard services.

chip Silicon wafer microprocessor.

chirality A chiral model has a preference for rotations in a certain sense. Lattice-gas models where each of all possible collision outcome is randomly selected with an equal probability are not chiral, though chiral models could be designed. Also known as *dissymmetry*.

CLUT Colour look-up table; the relation between the pixel values in an image and the colours (specifies by their RGB components) they correspond to.

CMOS Complementary metal-oxide on silicon; a currently favoured, semiconductor production process, allowing close integration of components and high performance.

collision That phase of the lattice-gas update which involves the replacement of collision configurations of particles at every site by another such that net momentum and mass are conserved.

Computing Surface A flexible topology, modular and expandable distributed memory multicomputer, consisting of a number of transputer boards (or other processors) containing compute elements (with dedicated memory), display elements, and others, which can be interconnected dynamically using a proprietary backplane, under the control of a global supervisor bus, which also provides each processor with an orthogonal debug channel. Machine partitioning into domains with their own resource (M^2VCS) allows flexible multi-user capability, with each domain provided with the MeikOS operating system and the OPS or CStools programming environments.

concurrency The ability to time-share dynamically between all currently executable processes, such that all may proceed without undue awareness of the influence of the others, modelling the nature of independent processors.

configuration

1. The arrangement of particles present at a site in the lattice at any particular time, typically coded into the bits of a byte.
2. The global arrangement of lattice features, such as sources and barriers, so as to specify the intended flow characteristics.

contention The blocking of access to data of one process due to its being locked (or reserved temporarily) by another, required to preserve integrity in shared-memory.

continuity equation The expression of the conservation of mass in a fluid system.

CPU The central processing unit in a computer.

CSP The Communicating Sequential Processes model (and others with the same acronym) of programming, where processes are considered to be sequential lists of instructions, and can extract data with other processes via (blocking) channel communication.

deadlock Inability for a process (or system of processes) to continue, generally because it requires a communication which will never be satisfied.

decomposition The partitioning of a problem, or its data, such that a number of processors can simultaneously cooperatively participate in its processing.

algorithmic Simultaneous execution of different parts of an algorithm on different processors. Generally an obvious, but extremely difficult to optimise, form of decomposition.

data Distribution of subsets of the data to different processors which are executing the same program. Requires no interprocessor communication. This model closely matches the operational characteristics of a SIMD computer.

geometric Distribution of subsets of the data to adjacent (or neighbouring) processors to be processed by the same program on each, yet requiring local information from the neighbours. Typically each update would require the exchange of border information with each neighbour.

density

site density the average number of particles per site of the lattice.

link density the average link occupancy (including rest particles), i.e., typically the site density divided by the number of links.

DMA Direct memory access; applies to the controllers responsible for the operation of the links on the transputer, such that data is transfered straight from memory when required.

domain A collection of transputer allocatable to a single user for an application, consisting of a host seat, a variable size of compute resource and possibly additional special purpose boards, such as graphics or dedicated fast disk boards.

DRAM Dynamic RAM which requires regular refresh to retain its contents (slower, but cheaper, than static RAM).

duality From the Boolean nature of the lattice-gas particles, the models with particles and 'holes' interchanged are equivalent.

duality factor Relates the properties of a lattice-gas to those of a real fluid through rescaling of the nonlinear term in the Navier-Stokes equation. It reflects the lack of Galilean invariance at lattice level and the Boolean nature of the particles.

DVI Device independent output from the LaTeX typesetter.

dvips The LaTeX DVI→PS filter used to produce the single PostScript output file of this document.

ECS The Edinburgh Concurrent Supercomputer, a transputer-based multicomputer built by Meiko Ltd, and jointly funded by Meiko Ltd, the Computer Board and the Science and Engineering Research Council (SERC). Originally planned to be capable of true supercomputer performance, with 1000 T800 transputers (a prospective cumulative performance of 1Gflops), currently consists of roughly 400 transputers, divided into a number of variously sized domains.

eddy A recirculation of fluid.

EPCC The Edinburgh Parallel Computing Centre; an interdisciplinary focus for the parallel processing research and service activities within Edinburgh, superseding and merging the ECS and DAP activities.

epsf The macros allowing the incorporation of conforming EPSF diagrams within LaTeX documents, for the `dvips` filter.

EPSF Encapsulated PostScript format specification for diagrams containing the necessary information for them to be suitable scaled and incorporated within other (PostScript) documents.

FCHC The face-centred hypercubic lattice-gas model, proposed by d'Humières, Lallemand and Frisch as suitable for three-dimensional lattice-gas simulations. Requires (at least) 24 bits for each lattice site, and a typically very large collision table.

flops The number of floating point operations executed by a processor in a second. More often quoted as millions (Mflops) or thousands of millions (Gflops).

FHP The lattice-gas models proposed by Frisch, Hasslacher and Pomeau, consisting of a hexagonal (6-connected) lattice on which boolean particles exist. Classified as two models:

>   **FHP6** with only moving (unit-velocity) particles,
>
>   **FHP7** with additional rest (zero-velocity) particles.

floating-point The representation of real (non-integer) numbers in a computer as a mantissa and exponent.

folding editor A powerful editor, provided as part of the TDS and OPS, allowing code fragments to be grouped into *folds* (and replaced by a fold marker) hierarchically, facilitating programming in *occam*.

FPU Floating-point unit; a numeric processing unit specialising in floating-point operations.

Galilean invariance The condition where dynamics are unaffected by arbitrary constant relative motions. This is broken by lattice-gases due to the finite set of directions for the velocity and to the exclusion principle which leads to the Boolean character of the particles.

gfx The graphics package provided on Meiko systems for bit-mapped monitors, used as the low-level primitives for the visualisation.

gfxtoppm A filter for interfacing **gfx**-format images to the PBM package.

Goth The guild of *occam* and transputer hackers, or one of its members, many of whom provided invaluable assistance throughout the duration of this work.

granularity The grain size is representative of the average subtask size in a computation, which should be carefully matched to the available processing element power.

  coarse-grained a small number of complex tasks, suited to MIMD systems with powerful processors.

  fine-grained a large number of relatively simple tasks, suited to SIMD systems with much less powerful processing elements.

  medium-grained an intermediate of the previous two, with a sizable number of fairly complex tasks.

grtool The extensive, WIMP-based graph drafting and data reduction tool for Sun workstations, capable of producing (reasonable) PostScript output, used for the majority of the graphs in this document.

H1 Planned upgrade for the transputer around 1991, providing virtual channels (i.e. point-to-point routing in hardware) as well as increased performance.

HPP The earliest lattice-gas model, investigated by Hardy, de Pazzis and Pomeau, consisting of a square (4-connected) lattice on which boolean particles exist.

HSV The user-orientated colour specification of hue, saturation and value based on tint, shade and tone.

idleness The fraction of time which is 'wasted' when a processor has an empty process queue, while awaiting communication.

idraw The object-oriented figure and diagram drafting tool for workstations running X windows, utilising the InterViews toolkit, used for the majority of the diagrams in this document.

IF The *occam* selection directive, which evaluates a sequence of guard statements and the corresponding process of the first found true.

ImMaX A WIMP-based image manipulation tool for Meiko graphics systems, allowing the capture, composition and manipulation of images and palettes in gfx-format. The image-sequencing and palette-cycling (noodling) features facilitate interactive visualisation.

incess Definition of values which are out or range by virtue of being too low, to match excess values which are out of range by virtue of being too high.

Inmos The company, now part of SGS-Thomson, responsible for the design and manufacture of the transputer, and the Transputer Development System (TDS).

Ising model The simple model of ferromagnetism, represented as a lattice of two-state spin systems, which has the same macroscopic properties, such as critical behaviour.

isotropy Similarity with respect to arbitrary rotations. Lattice-gases are isotropic at the macroscopic level, provided that the crystallographic group of the underlying lattice has sufficient symmetry.

jet Fluid ejected from an orifice into another, either at a different density or velocity.

laminar flow Characterised by smooth, simply defined fluid trajectories and corresponding to low Reynolds numbers.

LATEX The document typesetting system employed in the formatting of this work.

lattice-gas An example of an elementary gas, consisting of a grossly simplified (and fictitious) model of a gas, yet which does not violate kinetic theory.

link

1. Transputers have four hardware links on chip which are channels allowing communication with other transputers.

2. Microlattice sites have six (four) links in the FHP (HPP) models, on which particles may reside, before propagating to the neighbouring site which shares the link. The rest particles may also be considered to reside on link 0 which is not connected to another site.

livelock The trapping of a process (or system of processes) in a cycle of operations which will continue indefinitely since no valid termination condition exists. A form of deadlock.

Mach number, Ma The speed of a fluid, expressed as a fraction of the speed of sound in that fluid.

master/slave Typical applications are divided into a master and several slaves. Only the master is directly connected to the host, and has access to the file system and other I/O services, such as the keyboard, screen and mouse devices. The slaves rely on the master process to distribute work to them, and results are send back to the master for collation.

Meiko The company, based in Bristol, UK, originally comprising members of the transputer design team, who left to design a range of computer systems based on boards of transputers.

memory That part of a computer where the instructions and data are stored while the computation is in progress. Where multiple processors are involved, two memory arrangements are possible:

distributed memory such that each processor has its own memory space, and when it requires to access that of another processor a switching or message-routing overhead is incurred.

shared-memory memory is globally accessible from a common area, requiring the use of semaphores to avoid multiple simultaneous overwrites.

message-passing A paradigm by which processors in a distributed network, or multicomputer, can access remote data which was not, or could not be, retained on each processor, through packet-requesting and -routing.

MISD Multiple Instruction, Single Data; vector or pipeline computers, executing simultaneously a number of instructions on a single stream of data.

MIMD Multiple Instruction, Multiple Data; the ultimate in parallel computers, where multiple processors are capable of executing different instructions on different data elements simultaneously. Examples are the Meiko Computing Surface, expandable to $O(100)$ processors, and the Cray XMP series computers, typically using $O(8)$ processors.

multicomputer A collection of individual computers (or more generally compute elements) capable of cooperative work on a specified problem, which must typically be distributed so that each processor has a part of the problem.

Navier-Stokes equation The expression of Newton's Second Law of Motion applied to a fluid: that the rate of change of momentum of a fluid particle is equal to the net force acting on it. The equation is highly nonlinear making its solution generally difficult.

noodling CLUT entry cycling, generally with a smoothly varying palette (even over its ends) such that pixels of each entry cycle through the available values. Useful for highlighting vorticity or discontinuities.

occam The language of the transputer, designed in collaboration between Tony Hoare and David May, based on the CSP model of concurrent computation, and having specific constructs to make parallelism fully accessible.

OPS The Occam Programming System; Meiko's repackaged version of the Inmos TDS, the programming environment for the transputer.

palette The range of available colours for an image — generally the contents of a CLUT.

PAR The *occam* directive for executing a list of processes concurrently.

parallel computer The use of more than one processor to achieve additional performance through cooperation on a computation (as opposed to distinct computations).

parallelise Exploit the inherent parallelism in an algorithm (or implementation) specified in a sequential manner.

parallelism

    coarse-grained A small number of very powerful processors working jointly on a problem.

    fine-grained A large number of much less powerful processors, typically working on smaller subtasks of a problem.

particle One of the (single bit) entries in a configuration at a lattice site, distinguishable only in its momentum, which may be only such that it lies on one of the links to a neighbouring site, or alternatively, in the FHP7 model, at rest.

PBM The Portable Bitmap Manipulation package which allows interchange of numerous bitmap image formats, and is capable of producing (both monochrome and colour) PostScript output. The `gfxtoppm` filter interfaces to this package for hardcopy image output from the Meiko graphics boards.

pixel picture element; addressable dot on screen, constituting part of a bit-mapped display.

PostScript A simple interpreted programming language with powerful graphics capabilities. Its primary application is to describe the appearance of text, graphical shapes and sampled images on printed pages in a way which is independent of the resolution of output devices. This thesis is a single PostScript file[1], produced by the LaTeX document typesetting system, and includes a number of PostScript figures, graphs and diagrams from a variety of packages and tools.

priority The scheme by which certain specified processes can be preferentially executed over others.

high Those (typically lightweight) processes which should not be interrupted in their execution until they have completed, or are otherwise unable to proceed. Generally reserved for message passing subsystems, to quickly route messages while computation is continuing, or interrupt handlers.

low Normal computational tasks which can be interrupted without penalty while large-scale processing continues.

process A complete computational task, typically one of several capable of being simultaneously executed. Evolves from initiation, through sequential execution to termination.

heavyweight A task typically requiring a large amount of computation and capable of keeping a processor busy for a while.

lightweight A task which does not keep a processor busy, but rather idling, for significant amounts of time, when further computation could conceivably have been done.

propagation That phase of the lattice-gas update step which involves the movement of every (non-rest) particle to a neighbouring site, or possible reflection by barriers or absorption by sinks.

---

[1] Actually, the photographs in **Appendix P** obviously were done separately, but they could have equally have been rendered and included in colour PostScript, in the manner of **Figure 3.4** — this was unfortunately not possible in the window of the creation of this thesis.

PULSARS A graphical demonstration of dynamic interacting processes as pixel-trains which models contention and deadlocking in shared-memory computers.

RAM Random Access Memory; can be read from or written to in any order.

Reynolds number, Re The dimensionless parameter, relating the inertial and viscous properties of the fluid, used to characterise the complexity of a flow and compare flows in different media, arising from the Principle of Similarity. Low Reynolds number flows are laminar, while high Reynolds number flows are turbulent.

RGB The hardware-orientated specification of a colour from its red, green and blue components (c.f. HSV).

RISC Reduced Instruction Set Computer; provides a small, highly optimised, set of instructions.

segment That section of the complete lattice located on a particular processor, with a number of cells in each dimension. May be also referred to as a chunk.

semi-detailed balance The condition where all states having equal probabilities before collision, they will remain so afterwards.

SEQ The *occam* directive for executing a list of processes sequentially.

sequential The execution of instructions or processes in strict order one after the other.

sequentialisation The management of multiple, parallel operations, such that they appear to be sequential — like the multiplexing of file system requests from a number of processors.

SIMD Single Instruction, Multiple Data; one category of parallel computers, also known as *data parallel* since multiple processors execute each instruction (the same instruction) on (a typically fixed number of) many data elements under the direction of a master controller. Examples of such computers are the AMT DAP, with O(4096) processing elements and the Connection Machine, with O(65536) processing elements.

site An entry in the microlattice, connected to its neighbours by links, capable of containing particles in a configuration, or alternatively a code specifying a site at which a special update rule will be used.

    source A source site in the lattice is one where particles will be randomly generated at a rate corresponding to the strength of the source. A 20% source will produce particles on each link with a 20% probability each time step.

    sink A sink completely absorbs particles incident on it. All sources are actually also sinks, since they absorb in-coming particles, but also produce out-going particles.

wall/barrier Wall, or barrier, sites reflect incident particles. Typically reflection is specular, and particles will return along the link on which they arrived.

free lattice That part of the computational box absent of special sites, and therefore free to be occupied by particles.

SISD Single Instruction, Single Data; a terminology for traditional serial computers, when comparing with parallel computers. Only one instruction can be executed on one data element at a time, in the classical von Neumann sequential manner.

SRAM Static RAM; fast memory, not requiring refresh, which transputers have on-chip.

supercomputer A device capable of computation at a rate comparable to the fastest computers of the day. Originally coined for the Cray-1, rated at 10 Mflops, nowadays reserved for computers achieving Gflops performance.

synchrony The state of being synchronised with another or all processors, such that each processor is dependent on data from the others before it can proceed.

T414 One of the first transputers, with four serial links.

T800 The first transputer to have a floating-point unit on-chip, rated at 1Mflops.

Taχ Transputer activity tachometers; provide a continuous or polled processor activity reading from transputers based on the recorded idleness (availability for processing when none is present).

task farm A master/slave model of parallelism, where one processor (the master) distributes task packets to whichever of a number of workers are first able to process it and send the results back.

TDS The transputer development system, incorporating the *occam* toolset of compiler, configurer, debugger, etc, within a folding editor, to facilitate *occam* programming.

through-routing The transfer of data packets through a network of processors by routing and forwarding through intermediate processors, required in distributed-memory computers without hardware switching.

Tiny An optimised, general message-passing harness for transputers, allowing fast, efficient communication of data between processes or processors.

transputer The transputer is a 32-bit RISC microcomputer designed and manufactured by Inmos, supporting the CSP of concurrency via time-slicing. Integrated on-chip are a CPU, four bidirectional, 10 Mbit/s (or faster) serial links and 2 kbytes of local memory, complete with hardware task scheduler. Later transputers also have a floating point unit. All components are capable of concurrent execution.

turbulent flow Flow characterised by rapid, irregular, spatial and temporal velocity fluctuations, and corresponding to high Reynolds number.

UseNet The global computer network, allowing exchange of electronic mail (E-mail) and providing an electronic 'news' bulletin board.

vector computer One capable of executing instructions simultaneously on the elements of a stream (or vector) of data using multiple functional units.

videoRAM Memory dual-ported to address monitor screen pixels, providing a bit-mapped display.

viscosity That process by which energy is removed from the inertial flow of a fluid by friction:

> **kinematic, ν** An intrinsic property of a fluid.

> **shear, η** The viscosity due to shearing action within the fluid.

> **bulk, ζ** The viscosity due to bulk rotational energy diffusion.

> **propagation, ν_p** That component of the viscosity of a lattice-gas due to the propagation of particles between sites. Generally negative, but smaller in magnitude than the collision viscosity.

> **collision, ν_c** That component of the viscosity of a lattice-gas resulting from the collision mechanism. Richer collision sets, with more possible collision channels, have lower viscosities.

visualisation The rendering of multi-dimensional data, such that it is presented as an image in a reduced form which enables the interesting features to be highlighted. Some visualisation techniques can be as computationally intensive as the simulations which provide the data.

VLSI Very Large Scale Integration; the technology allowing many thousands of transistors to be put on a single integrated circuit.

von Kármán vortex street The stream of vortices in alternating directions, and equally spaced, typical of the wake of in-channel obstructions at moderate Reynolds numbers.

vortex A recirculation of fluid.

wake The region of decelerated flow downstream of an obstruction, as the flow coalesces at its rear. It is naturally unstable, and the von Kármán vortex street is observed even in the wake of a flat (orthogonal) plate.

word 16 or 32 bits; corresponding to the naturally addressable or transferable data item length.

WIMP Windows, Icons, Menus and Pointers; the natural metaphor for interaction with a computer, and specification of commands.

C   Classification of FHP particle configurations

Table C.1: FHP configuration classifications

The following tables contain a classification of the 2^7 particle configurations possible at a node for the *FHP7* model. Each configuration can be refered to by its 7-bit binary code, a decimal number in the range 0..127, or by a configuration picture: e.g.

$$\text{\scriptsize(arrow picture)} \equiv 17 \equiv \texttt{0010001} \tag{C.1}$$

The particles in a configuration have been coded so that those moving in the positive x-direction are shown by an arrow directed such[a] from the central node of the configuration picture, appearing as the rightmost bit of the binary code and least-significant bit of the decimal code. Particles moving in the other directions are similarly coded in a clockwise manner, with a rest-particle, should it exist, represented by a disc at the central node.

Each configuration has been classified by its number of particles (mass) and net momentum (**p**) into one of the classes in **Table 1.1**.

In the *FHP6* model, only the first four tables of configurations (i.e. those without the rest particles) are relevant: the leading `0` can be dropped from the configuration binary code and collision channels to codes outwith the range 0..63 are invalid.

---

[a] Convention is to generally represent particles by arrows based at the central node, whether the particle is considered to be entering or leaving the node at the instant concerned.

No data yet exists indicating that
dangerous levels of radiation are produced
in these ["fusion-in-a-bucket"] experiments,
but there is no sense in being a famous dead person.
Still less in being a kinda famous, near dead, bald person.

— [Dunning89]

	Code	Configuration	Class	Mass	Net $\mathbf{p}$	Collision Channels
	0	0000000	E	0	0	
	1	0000001	F	1	1	
	2	0000010	F	1	1	
	3	0000011	V	2	$\sqrt{3}$	
	4	0000100	F	1	1	
	5	0000101	J	2	1	66
	6	0000110	V	2	$\sqrt{3}$	
	7	0000111	W	3	2	
	8	0001000	F	1	1	
	9	0001001	I	2	0	18, 36
	10	0001010	J	2	1	68
	11	0001011	L	3	1	38, 69
	12	0001100	V	2	$\sqrt{3}$	
	13	0001101	L	3	1	22, 74
	14	0001110	W	3	2	
	15	0001111	$\tilde{V}$	4	$\sqrt{3}$	

**Table C.1a: Configuration classifications (000****)**

	Code	Configuration	Class	Mass	Net $\mathbf{p}$	Collision Channels
	16	0010000	F	1	1	
	17	0010001	J	2	1	96
	18	0010010	I	2	0	9, 36
	19	0010011	L	3	1	37, 98
	20	0010100	J	2	1	72
	21	0010101	Y	3	0	42, 73, 82, 100
	22	0010110	L	3	1	13, 74
	23	0010111	$\tilde{\text{J}}$	4	1	75, 102
	24	0011000	V	2	$\sqrt{3}$	
	25	0011001	L	3	1	52, 104
	26	0011010	L	3	1	44, 84
	27	0011011	$\tilde{\text{I}}$	4	0	45, 54, 85, 106
	28	0011100	W	3	2	
	29	0011101	$\tilde{\text{J}}$	4	1	90, 108
	30	0011110	$\tilde{\text{V}}$	4	$\sqrt{3}$	
	31	0011111	$\tilde{\text{F}}$	5	1	110

Table C.1b: Configuration classifications (001**)**

	Code	Configuration	Class	Mass	Net **p**	Collision Channels
	32	0100000	F	1	1	
	33	0100001	V	2	$\sqrt{3}$	
	34	0100010	J	2	1	65
	35	0100011	W	3	2	
	36	0100100	I	2	0	9, 18
	37	0100101	L	3	1	19, 98
	38	0100110	L	3	1	11, 69
	39	0100111	$\tilde{V}$	4	$\sqrt{3}$	
	40	0101000	J	2	1	80
	41	0101001	L	3	1	50, 81
	42	0101010	Y	3	0	21, 73, 82, 100
	43	0101011	$\tilde{J}$	4	1	83, 101
	44	0101100	L	3	1	26, 84
	45	0101101	$\tilde{I}$	4	0	27, 54, 85, 106
	46	0101110	$\tilde{J}$	4	1	77, 86
	47	0101111	$\tilde{F}$	5	1	87

Table C.1c: Configuration classifications (010**)**

	Code	Configuration	Class	Mass	Net **p**	Collision Channels
	48	0110000	V	2	$\sqrt{3}$	
	49	0110001	W	3	2	
	50	0110010	L	3	1	41, 81
	51	0110011	$\tilde{V}$	4	$\sqrt{3}$	
	52	0110100	L	3	1	25, 104
	53	0110101	$\tilde{J}$	4	1	105, 114
	54	0110110	$\tilde{I}$	4	0	27, 45, 85, 106
	55	0110111	$\tilde{F}$	5	1	107
	56	0111000	W	3	2	
	57	0111001	$\tilde{V}$	4	$\sqrt{3}$	
	58	0111010	$\tilde{J}$	4	1	89, 116
	59	0111011	$\tilde{F}$	5	1	117
	60	0111100	$\tilde{V}$	4	$\sqrt{3}$	
	61	0111101	$\tilde{F}$	5	1	122
	62	0111110	$\tilde{F}$	5	1	93
	63	0111111	S	6	0	

Table C.1d: Configuration classifications (011**)**

|	Code	Configuration	Class	Mass	Net $\mathbf{p}$	Collision Channels
	64	1000000	$E_\bullet$	1	0	
	65	1000001	$F_\bullet$	2	1	34
	66	1000010	$F_\bullet$	2	1	5
	67	1000011	$V_\bullet$	3	$\sqrt{3}$	
	68	1000100	$F_\bullet$	2	1	10
	99	1000101	$J_\bullet$	3	1	11, 38
	70	1000110	$V_\bullet$	3	$\sqrt{3}$	
	71	1000111	$W_\bullet$	4	2	
	72	1001000	$F_\bullet$	2	1	20
	73	1001001	$I_\bullet$	3	0	21, 42, 82, 100
	74	1001010	$J_\bullet$	3	1	13, 22
	75	1001011	$L_\bullet$	4	1	23, 102
	76	1001100	$V_\bullet$	3	$\sqrt{3}$	
	77	1001101	$L_\bullet$	4	1	46, 86
	78	1001110	$W_\bullet$	4	2	
	79	1001111	$\tilde{V}_\bullet$	5	$\sqrt{3}$	

Table C.1e: Configuration classifications (100**)**

	Code	Configuration	Class	Mass	Net **p**	Collision Channels
	80	1010000	F.	2	1	40
	81	1010001	J.	3	1	41, 50
	82	1010010	I.	3	0	21, 42, 73, 100
	83	1010011	L.	4	1	43, 101
	84	1010100	J.	3	1	26, 44
	85	1010101	Ỹ.	4	0	27, 45, 54, 106
	86	1010110	L.	4	1	46, 77
	87	1010111	J̃.	5	1	47
	88	1011000	V.	3	$\sqrt{3}$	
	89	1011001	L.	4	1	58, 116
	90	1011010	L.	4	1	29, 108
	91	1011011	Ĩ.	5	0	109, 118
	92	1011100	W.	4	2	
	93	1011101	J̃.	4	1	62
	94	1011110	Ṽ.	5	$\sqrt{3}$	
	95	1011111	F̃.	6	1	

Table C.1f: Configuration classifications (101**)**

	Code	Configuration	Class	Mass	Net **p**	Collision Channels
	96	1100000	F.	2	1	17
	97	1100001	V.	3	$\sqrt{3}$	
	98	1100010	J.	3	1	19, 37
	99	1100011	W.	4	2	
	100	1100100	I.	3	0	21, 42, 73, 82
	101	1100101	L.	4	1	43, 83
	102	1100110	L.	4	1	23, 75
	103	1100111	$\tilde{V}$.	5	$\sqrt{3}$	
	104	1101000	J.	3	1	25, 52
	105	1101001	L.	4	1	53, 114
	106	1101010	$\tilde{Y}$.	4	0	27, 45, 54, 85
	107	1101011	$\tilde{J}$.	5	1	55
	108	1101100	L.	4	1	29, 90
	109	1101101	$\tilde{I}$.	5	0	91, 118
	110	1101110	$\tilde{J}$.	5	1	31
	111	1101111	$\tilde{F}$.	6	1	

Table C.1g: Configuration classifications (110**)**

	Code	Configuration	Class	Mass	Net **p**	Collision Channels
	112	1110000	V$_\bullet$	3	$\sqrt{3}$	
	113	1110001	W$_\bullet$	4	2	
	114	1110010	L$_\bullet$	4	1	53, 105
	115	1110011	$\tilde{\text{V}}_\bullet$	5	$\sqrt{3}$	
	116	1110100	L$_\bullet$	4	1	58, 89
	117	1110101	$\check{\text{J}}_\bullet$	5	1	59
	118	1110110	$\tilde{\text{I}}_\bullet$	5	0	91, 109
	119	1110111	$\tilde{\text{F}}_\bullet$	6	1	
	120	1111000	W$_\bullet$	4	2	
	121	1111001	$\tilde{\text{V}}_\bullet$	5	$\sqrt{3}$	
	122	1111010	$\check{\text{J}}_\bullet$	5	1	61
	123	1111011	$\tilde{\text{F}}_\bullet$	6	1	
	124	1111100	$\tilde{\text{V}}_\bullet$	5	$\sqrt{3}$	
	125	1111101	$\tilde{\text{F}}_\bullet$	6	1	
	126	1111110	$\tilde{\text{F}}_\bullet$	6	1	
	127	1111111	S$_\bullet$	7	0	

Table C.1h: Configuration classifications (111**)**

F Fragments of occam implementation

List of Code Fragments

*"... the conceivable criminal violations of this operation
have serious implications for the health and welfare
of all individuals, corporations, and ... agencies
relying on computers and telephones to communicate."*

— *[USSS90]*

*"Pascal gives you a water pistol
filled with distilled water.
C not only gives you a loaded .357,
it points it at your head as a default.
Why do you think Pascal is taught in school?
And which would you rather have
when there was a hungry bear in the area?"*

— *[Harkins90]*

```
    {{{ particle link codes
    VAL power2       IS [0, 1, 2, 4, 8, 16, 32, 64, 128, 256] :
    VAL link.code    IS              power2 :

    VAL n.links      IS                    7 :
  5 VAL n.configs    IS   power2[n.links+1] :   --- 128       10000000
    VAL max.config   IS          n.configs-1 :   --- 127       01111111

    {{{ link picture
    --                          5      6
    --                           \    /
 10 --                            \  /
    --                      4--- 0 ---1
    --                            /  \
    --                           /    \
    --                          3      2
 15 }}}

    VAL L1mover      IS        link.code[1] :   ---    1       00000001
    VAL L2mover      IS        link.code[2] :   ---    2       00000010
    VAL L3mover      IS        link.code[3] :   ---    4       00000100
    VAL L4mover      IS        link.code[4] :   ---    8       00001000
 20 VAL L5mover      IS        link.code[5] :   ---   16       00010000
    VAL L6mover      IS        link.code[6] :   ---   32       00100000
    VAL L0mover      IS        link.code[7] :   ---   64       01000000

    {{{ other site codes
    VAL null         IS BYTE (link.code[0]) :   ---    0       00000000
 25 VAL wall         IS BYTE 240             :   --- 240       11110000
    VAL sink         IS BYTE (link.code[8]) :   --- 128       10000000
    VAL source       IS BYTE (link.code[8]) :   --- 128..228  1*******
    VAL terminator   IS BYTE (power2[9] -1) :   --- 255       11111111
    VAL halt         IS BYTE (power2[9] -2) :   --- 254       11111110
 30 }}}

    VAL n.configs.b   IS BYTE n.configs :
    VAL null.velocity IS power2 [8] :
    }}}
```
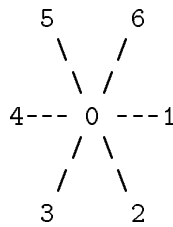
Code Fragment 1: Site codes

The particle codes which will constitute components of the configuration codes are coded into the lowest 7 bits of a single byte, leaving the remaining bit to signify special codes for barriers, sources, etc.

```
   {{{ collision codes
   VAL invariant      IS 0  :

   VAL deterministic  IS 1  :

   VAL ND2.particle   IS 2  :    --   non-deterministic 2 particle

 5 {{{ ND3.particle.* cases     --   non-deterministic 3 particle
   VAL ND3.particle.0 IS 30 :   --   net zero momentum case (link 0)
   VAL ND3.particle.1 IS 31 :   --   net unit momentum along link 1,
   VAL ND3.particle.2 IS 32 :   --   etc.
   VAL ND3.particle.3 IS 33 :
10 VAL ND3.particle.4 IS 34 :
   VAL ND3.particle.5 IS 35 :
   VAL ND3.particle.6 IS 36 :
   }}}

   {{{ ND4.particle.* cases     --   non-deterministic 4 particle
15 VAL ND4.particle.0 IS 40 :   --   net zero momentum case (link 0)
   VAL ND4.particle.1 IS 41 :   --   net unit momentum along link 1,
   VAL ND4.particle.2 IS 42 :   --   etc.
   VAL ND4.particle.3 IS 43 :
   VAL ND4.particle.4 IS 44 :
20 VAL ND4.particle.5 IS 45 :
   VAL ND4.particle.6 IS 46 :
   }}}

   VAL ND5.particle   IS 5  :    --  non-deterministic 5 particle
   }}}
```

Code Fragment 2: Collision update type codes
These codes are the entries in the collision table (**Code Fragment 3**), and determine the form of the collision update. The first digit, in the non-deterministic cases ND?, splits on the number of particles, and where configurations have a net momentum, this is shown by the last digit .?, corresponding to the link direction of the net momentum.

```
     {{{ collision table
     VAL collision IS
     --        0:    1:    2:    3:    4:    5:    6:    7:    8:    9:
             [  0,    0,    0,    0,    0,    1,    0,    0,    0,    2,        --  00
 5              1,   32,    0,   33,    0,    0,    0,    1,    2,   31,        --  10
                1,   30,   33,   42,    0,   35,   34,   40,    0,   44,        --  20
                0,    1,    0,    0,    1,    0,    2,   31,   32,    0,        --  30
                1,   36,   30,   41,   34,   40,   43,    1,    0,    0,        --  40
               36,    0,   35,   46,   40,    1,    0,    0,   45,    1,        --  50
10              0,    1,    1,    0,                                           --  60
                                  0,    1,    1,    0,    1,   32,             --  60
                0,    0,    1,   30,   33,   42,    0,   43,    0,    0,        --  70
                1,   36,   30,   41,   34,   40,   43,    1,    0,   45,        --  80
               44,    5,    0,    1,    0,    0,    1,    0,   31,    0,        --  90
15             30,   41,   42,    0,   35,   46,   40,    1,   44,    5,        -- 100
                1,    0,    0,    0,   46,    0,   45,    1,    5,    0,        -- 110
                0,    0,    1,    0,    0,    0,    0,    0               ] : -- 120
     }}}

     {{{ collider table
20   VAL collider  IS
     --        0:    1:    2:    3:    4:    5:    6:    7:    8:    9:
             [  0,    1,    2,    3,    4,   66,    6,    7,    8,  888,        --  00
               68,  888,   12,  888,   14,   15,   16,   96,  888,  888,       --  10
               72,  888,  888,  888,   24,  888,  888,  888,   28,  888,       --  20
25             30,  110,   32,   33,   65,   35,  888,  888,  888,   39,       --  30
               80,  888,  888,  888,  888,  888,  888,   87,   48,   49,       --  40
              888,   51,  888,  888,  888,  107,   56,   57,  888,  117,       --  50
               60,  122,   93,   63,                                          --  60
                                 64,   34,    5,   67,   10,  888,             --  60
30             70,   71,   20,  888,  888,  888,   76,  888,   78,   79,       --  70
               40,  888,  888,  888,  888,  888,  888,   47,   88,  888,       --  80
              888,  888,   92,   62,   94,   95,   17,   97,  888,   99,       --  90
              888,  888,  888,  103,  888,  888,  888,   55,  888,  888,       -- 100
               31,  111,  112,  113,  888,  115,  888,   59,  888,  119,      -- 110
35            120,  121,   61,  123,  124,  125,  126,  127            ] : -- 120
     }}}
```

Code Fragment 3: Collision update tables

To update a configuration, c, it is used as an index into collision to determine the collision type. If found to be invariant no change is necessary; if deterministic the new configuration is extracted from the collider table; otherwise the update is one of the non-deterministic (ND?.particle*) ones, and requires a selection from one of the appropriate ND?.collider* tables. 888 is an error code in the collider table.

```
     {{{ non-deterministic collider tables
     VAL ND2.collider.0 IS [  9,  18,  36              ] :

     VAL ND3.collider.0 IS [ 21,  42,  73,  82, 100 ] :
     VAL ND3.collider.1 IS [ 19,  37,  98              ] :
  5  VAL ND3.collider.2 IS [ 11,  38,  69              ] :
     VAL ND3.collider.3 IS [ 13,  22,  74              ] :
     VAL ND3.collider.4 IS [ 26,  44,  84              ] :
     VAL ND3.collider.5 IS [ 25,  52, 104              ] :
     VAL ND3.collider.6 IS [ 41,  50,  81              ] :

 10  VAL ND4.collider.0 IS [ 27,  45,  54,  85, 106 ] :
     VAL ND4.collider.1 IS [ 43,  83, 101              ] :
     VAL ND4.collider.2 IS [ 23,  75, 102              ] :
     VAL ND4.collider.3 IS [ 46,  77,  86              ] :
     VAL ND4.collider.4 IS [ 29,  90, 108              ] :
 15  VAL ND4.collider.5 IS [ 58,  89, 116              ] :
     VAL ND4.collider.6 IS [ 53, 105, 114              ] :

     VAL ND5.collider.0 IS [ 91, 109, 118              ] :
     }}}
```

Code Fragment 4: Auxiliary non-deterministic collision update tables
When a collision has been identified as non-deterministic, an entry is selected from the
appropriate table, dependent on the number of particles (`ND?`) and the link direction of
the net momentum (`.?`) which must be conserved by the update. Care can be taken to
ensure that the out-going configuration selection from the table is different from in-going
one (see **Code Fragment 5**).

```
     PROC ND.select (VAL []INT collider.table, BYTE config)

        VAL table.size IS SIZE collider.table :
        VAL in.config  IS INT config :
        INT out.config, index :

  5     SEQ
          prig (table.size, index) -- pseudo-random integer generator
          VAL chosen.entry IS collider.table [index] :
          IF
            chosen.entry = in.config
 10           out.config := collider.table [(index+1)\table.size]
            TRUE
              out.config := chosen.entry
          config := BYTE out.config
        :
```

Code Fragment 5: Non-deterministic selection routine
This routine chooses an entry from the provided non-deterministic collision table
(`ND?.collider.*`), and ensures that if the selection matches the incoming entry that
the following entry is returned. The `prig` routine randomly selects an integer from the
range `0..table.size` .

```
    PROC collide (BYTE config) -- full FHP7 collision set

       VAL int.config IS INT config :
       VAL collision.type IS collision [int.config] :

       IF
 5       collision.type = invariant
           SKIP
         collision.type = deterministic
           config := BYTE (collider [int.config])
         {{{ non-deterministic net zero momentum
10       collision.type = ND2.particle
           ND.select (ND2.collider,   config) -- ND 2 particle 0 momentum
         collision.type = ND3.particle.0
           ND.select (ND3.collider.0, config) -- ND 3 particle 0 momentum
         collision.type = ND4.particle.0
15         ND.select (ND4.collider.0, config) -- ND 4 particle 0 momentum
         collision.type = ND5.particle
           ND.select (ND5.collider,   config) -- ND 5 particle 0 momentum
         }}}
         {{{ non-deterministic net unit momentum
20       collision.type > ND3.particle.0 -- 3 particles with unit momentum
           IF
             collision.type = ND3.particle.1  -- unit momentum on link 1
               ND.select (ND3.collider.1, config)
             collision.type = ND3.particle.2  -- unit momentum on link 2
25             ND.select (ND3.collider.2, config)
             collision.type = ND3.particle.3  -- unit momentum on link 3
               ND.select (ND3.collider.3, config)
             collision.type = ND3.particle.4  -- unit momentum on link 4
               ND.select (ND3.collider.4, config)
30           collision.type = ND3.particle.5  -- unit momentum on link 5
               ND.select (ND3.collider.5, config)
             collision.type = ND3.particle.6  -- unit momentum on link 6
               ND.select (ND3.collider.6, config)
             TRUE
35             STOP -- Invalid collision type for config
         collision.type > ND4.particle.0 -- 4 particles with unit momentum
           ... non-deterministic update as for 3 particles
         }}}
         TRUE
40         STOP -- Invalid collision type for config
     :
```

Code Fragment 6: Configuration collision routine

Using the incoming configuration as an index into the `collision` table, provides the
`collision.type` which determines whether the returned value is unchanged (`invariant`),
indexed from the `collider` table (`deterministic`) or selected from the requisite
`ND?.collider.*` table.

```
     PROC move (VAL BYTE old.config, BYTE site, L1site, L2site, L3site,
                                      L4site, L5site, L6site)
       INT config :

       SEQ
5        config := INT old.config
         {{{ rest particle
         IF
           config >= L0mover
             {{{ move particle from config
10           SEQ
               config := config - L0mover
               IF
                 site < n.configs.b
                   site := BYTE ((INT site) + L0mover)
15               TRUE
                   SKIP
             }}}
           TRUE
             SKIP
20       }}}
         {{{ L6mover
         IF
           config >= L6mover
             {{{ move particle from config
25           SEQ
               config := config - L6mover
               IF
                 L6site = wall
                   site := BYTE ((INT site) + L3mover)
30               L6site >= source
                   SKIP
                 TRUE
                   L6site := BYTE ((INT L6site) + L6mover)
             }}}
35         TRUE
             SKIP
         }}}
         ... L5mover
         ... L4mover
40       ... L3mover
         ... L2mover
         ... L1mover
       :
```

Code Fragment 7: Configuration propagation routine

Each particle located in the incoming configuration (`old.config`) is stripped off in turn
and added to the configuration at the appropriate neighbouring site unless that site has a
special code. If the neighbour is a source site, the particle is just deleted, while a barrier
site requires to reflect the particle direction and reincorporate it into the new configuration
at the incoming site (`site`).

```
      PROC generate (VAL INT source.density, INT config)

        INT test.pc :

        SEQ
          config := 0
    5     prig (100000000, test.pc) -- pseudo random integer generator
          {{{ first 4 links from first random number
          {{{ link 1
          IF
            test.pc > (source.density*1000000)
   10         SKIP
            TRUE
              config := config + link.code [1]
          test.pc := test.pc \ 1000000
          }}}
   15     {{{ link 2
          IF
            test.pc > (source.density*10000)
              SKIP
            TRUE
   20         config := config + link.code [2]
          test.pc := test.pc \ 10000
          }}}
          {{{ link 3
          IF
   25       test.pc > (source.density*100)
              SKIP
            TRUE
              config := config + link.code [3]
          test.pc := test.pc \ 100
   30     }}}
          {{{ link 4
          IF
            test.pc > source.density
              SKIP
   35       TRUE
              config := config + link.code [4]
          }}}
          }}}
          prig (1000000, test.pc)
   40     {{{ remaining 3 links from second
          ... link 5
          ... link 6
          ... link 7
          }}}
   45   :
```

Code Fragment 8: Configuration generation routine

When a new particle configuration is required, it is generated by including a particle in each possible link direction with the desired probability. Up to four particles are (fairly randomly) generated from the bits of one random number.

```
    SEQ k=0 FOR (y.sites/2)
      VAL J IS (k TIMES 2)+1 :
      SEQ
        {{{ update row J   (odd)  <<
 5      VAL j IS J :
        SEQ i=1 FOR local.cols
          {{{ site abbreviations  <<
                                                        --        5 6
          VAL curr.centre IS (i TIMES maxYrows)+j :     --         |/
10        VAL last.centre IS curr.centre-maxYrows :     --        4-0-1
          VAL next.centre IS curr.centre+maxYrows :     --         |\
                                                        --        3 2
          old.site.code IS old.lattice [curr.centre  ] :
          new.site.code IS new.lattice [curr.centre  ] :
15        link1site      IS new.lattice [next.centre  ] :
          link2site      IS new.lattice [next.centre-1] :
          link3site      IS new.lattice [curr.centre-1] :
          link4site      IS new.lattice [last.centre  ] :
          link5site      IS new.lattice [curr.centre+1] :
20        link6site      IS new.lattice [next.centre+1] :
          }}}
          IF
            (old.site.code = wall) OR (old.site.code = sink)
              new.site.code := old.site.code
25          old.site.code > source
              {{{ generate a new configuration and move those particles
              VAL source.density IS (INT old.site.code)-(INT source) :
              INT new.config :
              SEQ
30              generate (source.density, new.config)
                new.site.code := null
                move (BYTE new.config, new.site.code,
                                  link1site, link2site, link3site,
                                  link4site, link5site, link6site)
35              new.site.code := old.site.code
              }}}
            TRUE
              {{{ collide & move particles in old configuration
              SEQ
40              collide (old.site.code)
                move (old.site.code, new.site.code,
                                  link1site, link2site, link3site,
                                  link4site, link5site, link6site)
              }}}
45      }}}
        ... update row J+1 (even) >>
```

Code Fragment 9: Lattice update
Update of the (local segment of the) lattice, involved selecting each site from a row in
turn and moving the collided particles from the incoming configuration, or those gener-
ated when the current site is a source, into the neighbouring sites. Odd and even rows
are updated similarly, but with different neighbour locations. Remapping of the two-
dimensional lattice array into an equivalent vector, allows one level of indexing to be
avoided for increased performance.

P   Plate gallery

List of Colour Plates

Faithless in faith,
We must behold the things we see.

— [Ultravox82]

If a picture is worth a thousand words,
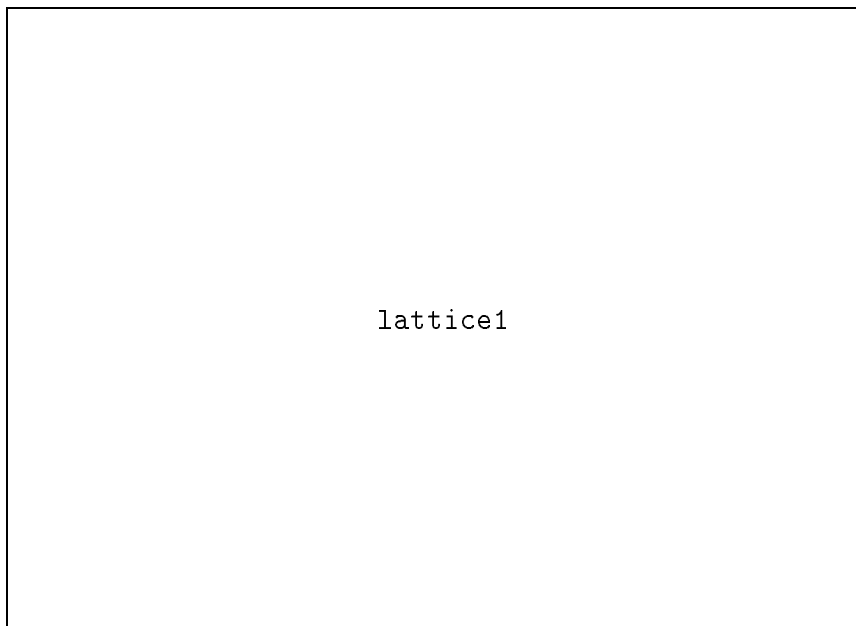*it must surely be worth a million numbers.*

— [YUM86]

Plate 1a: Hexagonal microlattice, $T = t_*$

A view of part of the microlattice (design similar to the mouth of the pipe in **Plate 6**), with particles shown as small vectors (or 'blobs' at the nodes for the stationary ones), and hexagonal wall (yellow) and source (green) sites. 16x16 averaging cell outlines are also visible in blue. Particle configurations which will undergo a collision update are highlighted in cyan — there are 12 of these, all of which represent two particle collisions at this density.
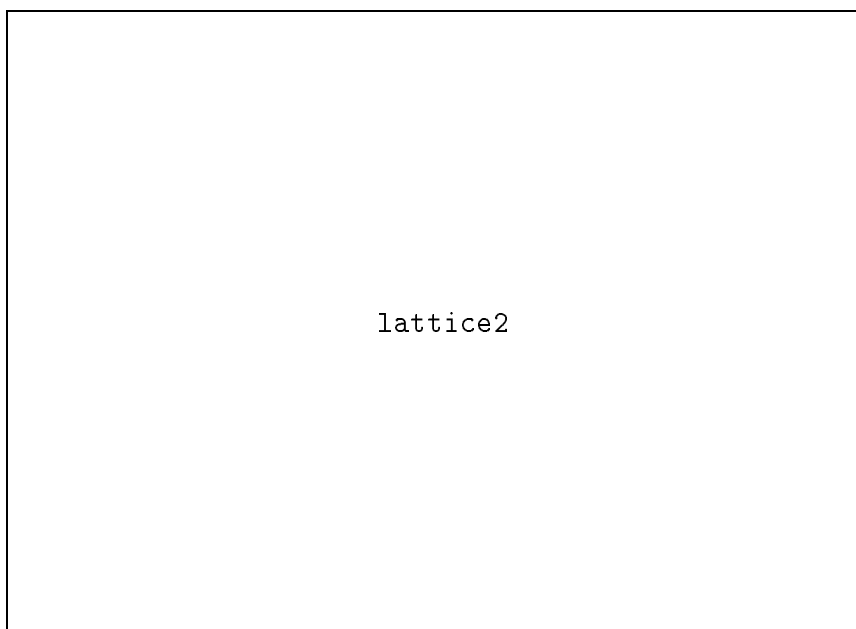
Plate 1b: Hexagonal microlattice, $T = t_* + 1$

An update later, particles have collided (or been generated) at their previous locations and propagated to their new locations, where they may have been reflected or absorbed. An excess of particles from the source strip into the channel can be distinguished (3 particles were absorbed to the 7 created).

pwaveseq

Plate 2a: Pressure wave

A map of the cell pressure and the flow vector field of the pressure wave of **Section 3.2.2** is charted through the time evolution from initiation, circular growth and reflection from the walls. Both the compression and rarefaction waves can be distinguished above the fluctuating background.
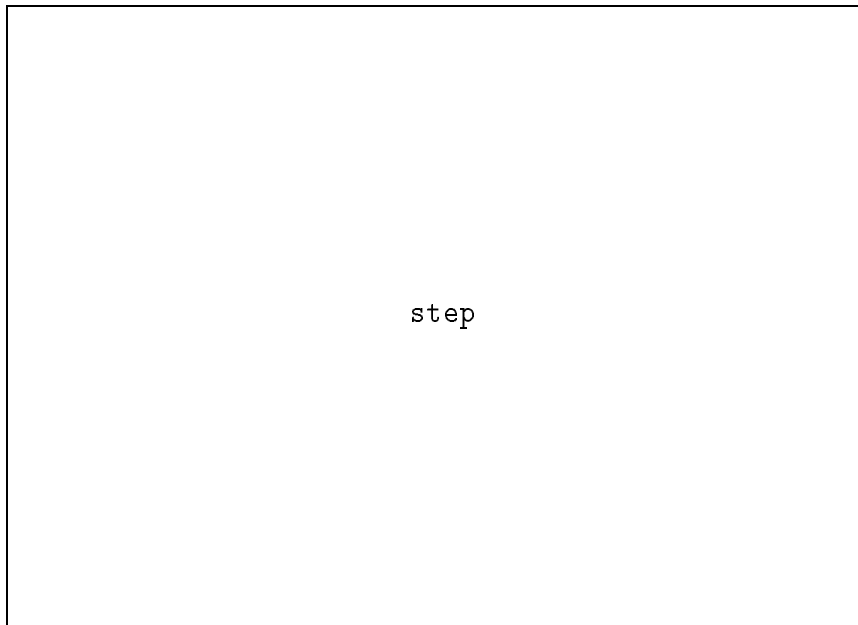
step

Plate 2b: Flow past a backward step

A large and slowly recirculating eddy has formed behind the step, but has merged with the downstream sink, preventing a measurement of the point of reattachment (**Section 3.4.1**).

angle
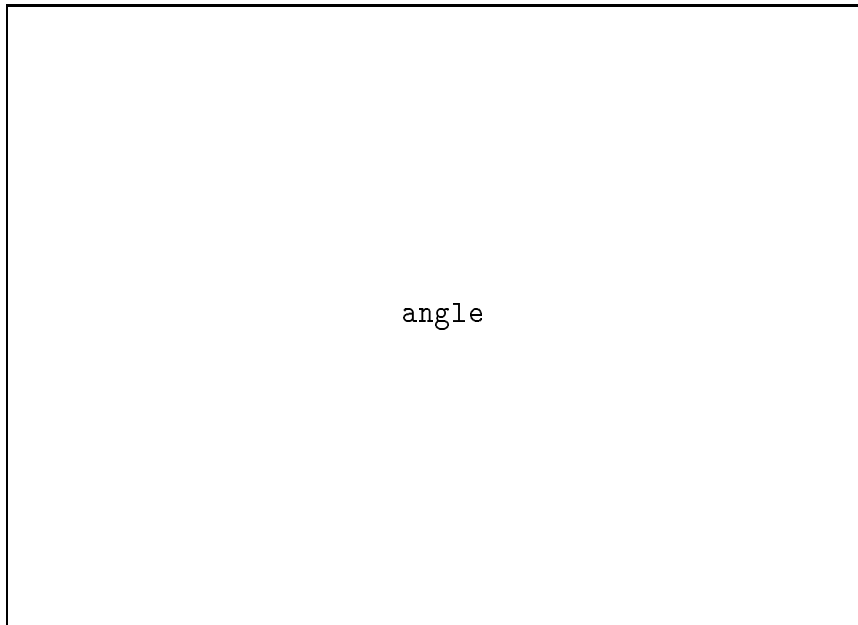
Plate 3a: Flow past an inclined barrier
A vortex street has been created in the wake of the barrier. (**Section 3.4.2.2**: Re ~ 200)

circle
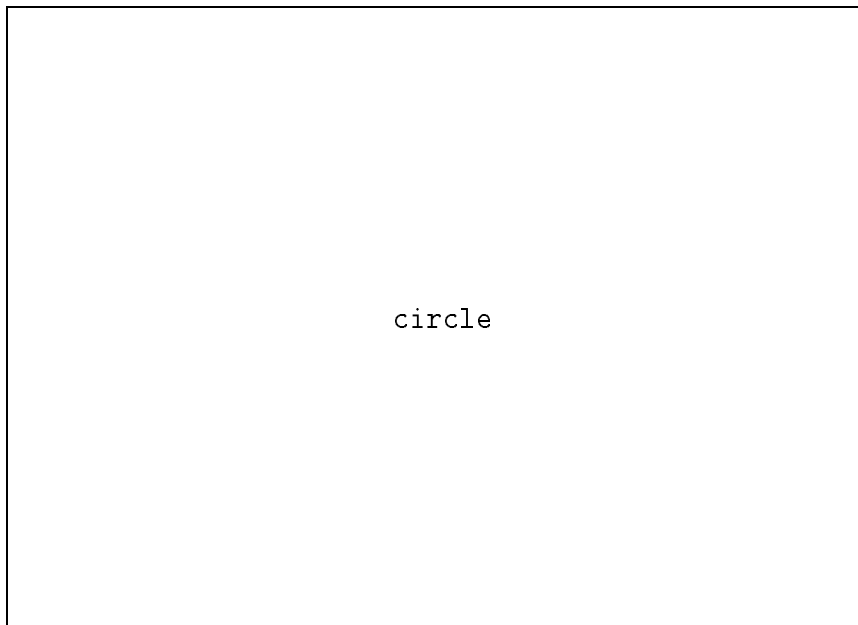
Plate 3b: Flow past a circular cylinder
In this transient phase of the development, the symmetry of the pair of eddies is just about to break as one grows at the (temporary) expense of the other, and a vortex street is formed. (**Section 3.4.2.3**: Re $\sim 200; Ut/L \sim 4.6$)

```
prismflo
```

```
prismmap
```
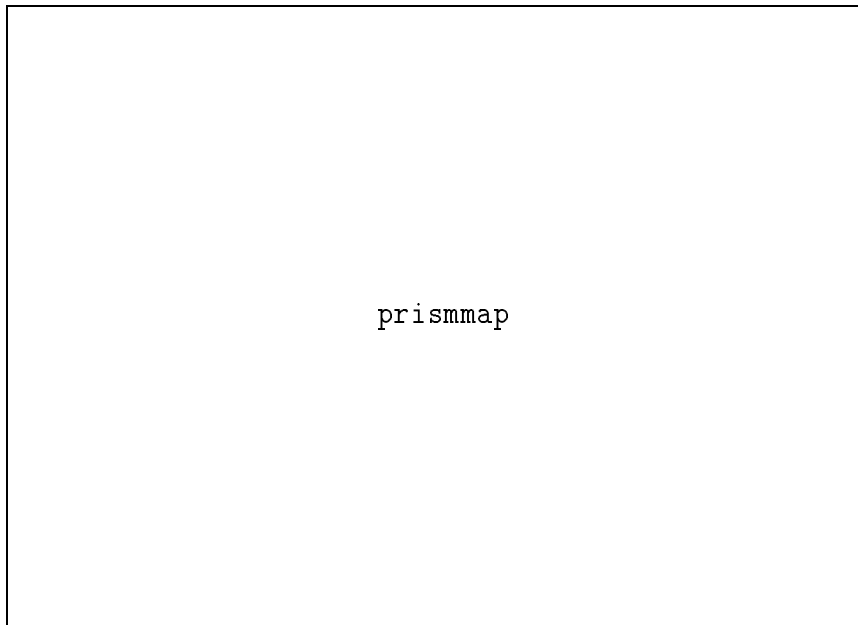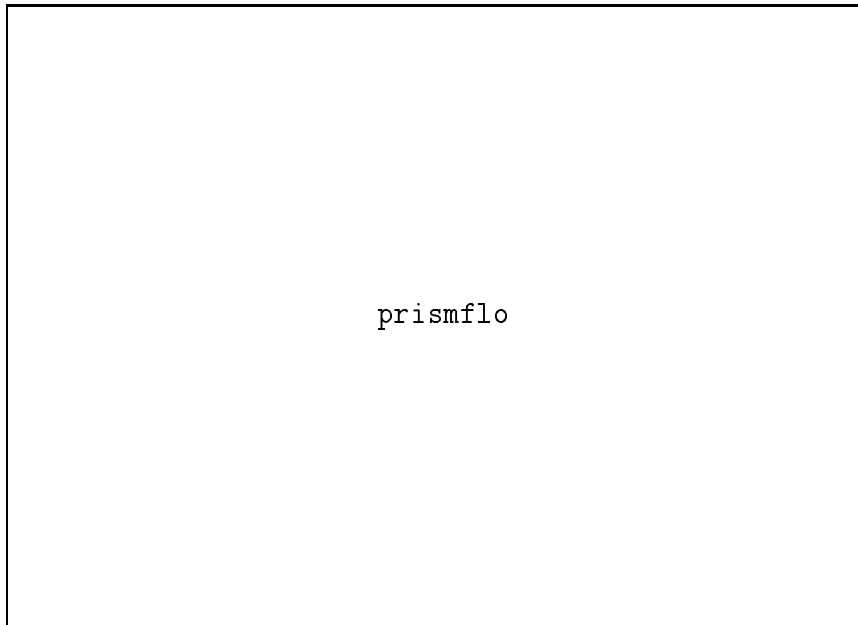
Plate 4: Flow past a triangular prism
Pronounced eddies are produced by the prism in this high velocity simulation which
were used in the determination of the eddy-shedding properties. Two snapshots of the
wake which highlight the vortex street have been selected, and the mean channel velocity
subtracted from the upper vector map to facilitate easier identification of the vortex
centres. (**Section 3.4.3**: Re ~ 900; St ~ 0.15)

jet-3000

jet-7000
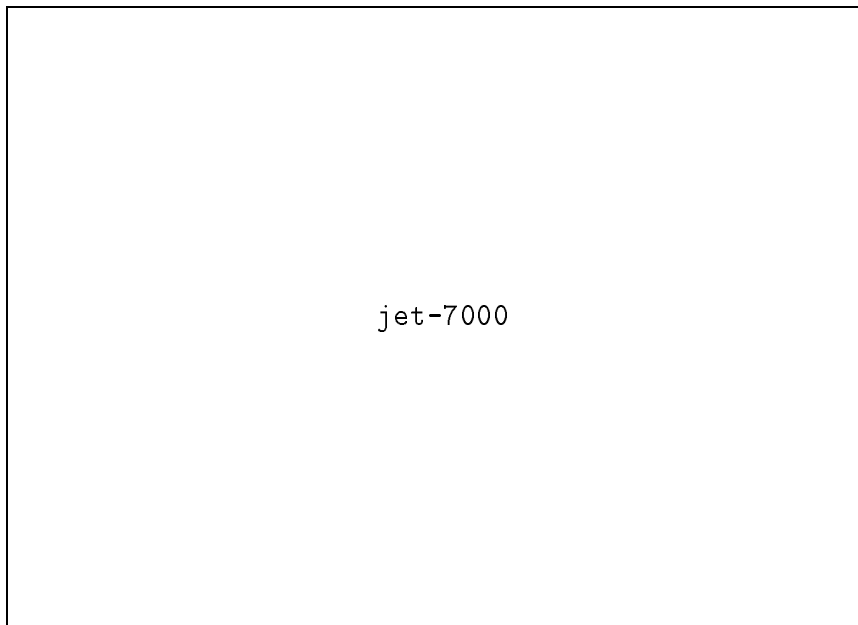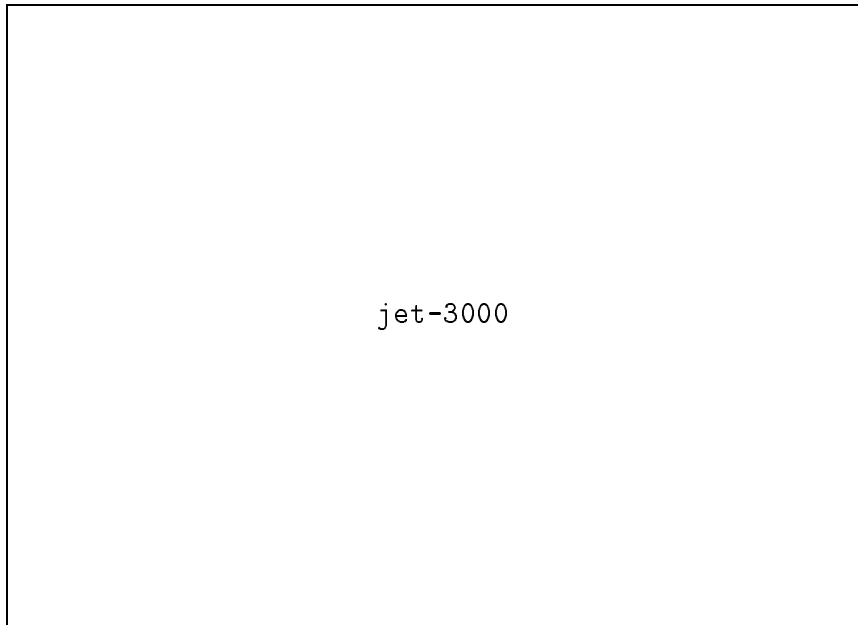
Plate 5: Jet into a channel
Two instances in the evolution of a jet introduced from the in-channel nozzle are recorded
(**Section 3.5**). Since the fluid in the jet is moving faster than that which it meets in
the channel, it oscillates as it attempts to match velocities. The advancing pressure wave
can also be seen in the first instance. For each case, the mean channel velocity has been
subtracted.

crossjet5

crossjet20

Plate 6: Jet into a cross-flow
A strong jet (with velocity almost twice that of the channel) orthogonal to the flow in the
channel is initially deflected downstream, before being dragged round into the sheltered
area and forming an eddy which grows and become increasingly unstable. The mean
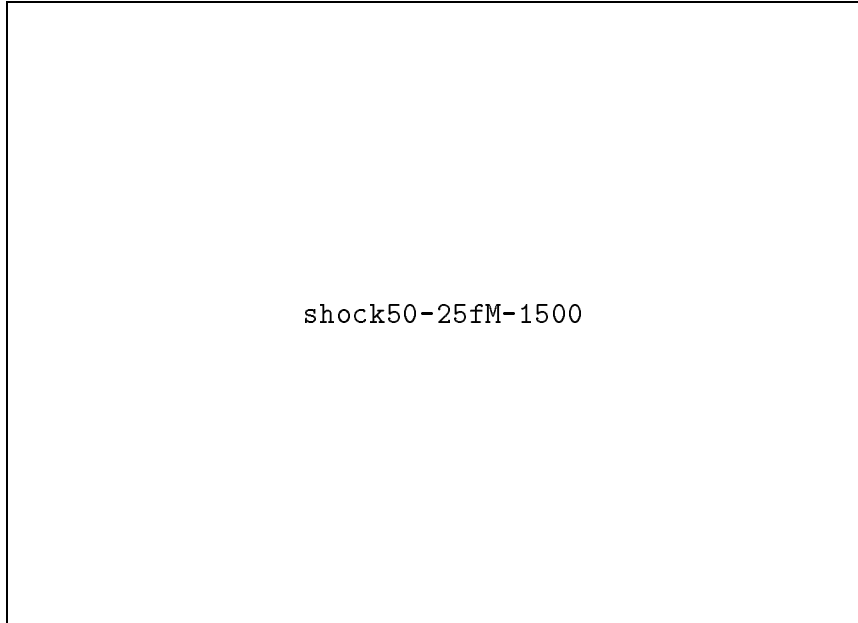channel velocity has been subtracted in each case (**Section 3.5.2**).

shock50-25fM-1500

shock50-25dP-1500
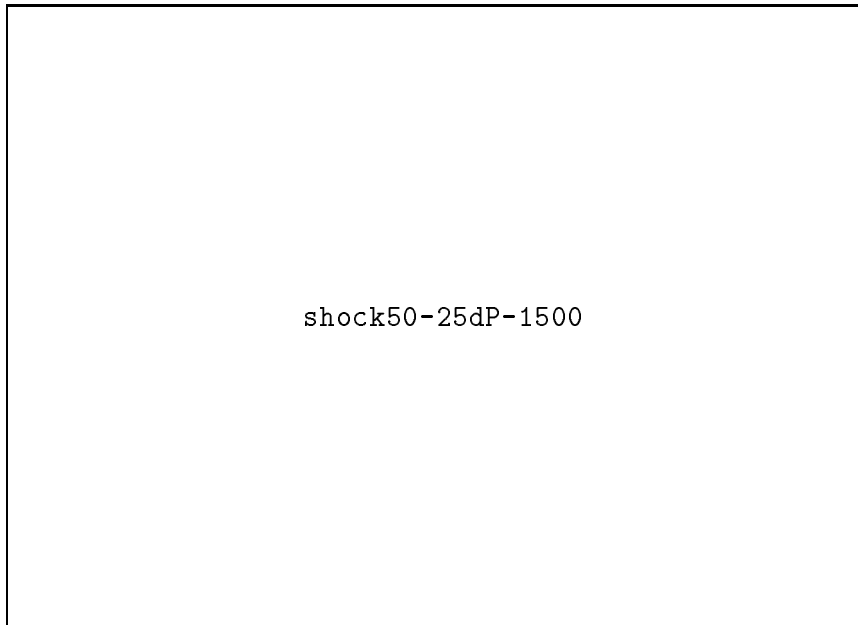
Plate 7: Shock tube; attribute maps
The flow field and its density and pressure maps are shown for the 50%–25% reservoir case of the shock tube experiment (**Section 4.2**).

```
shock50-25M
```
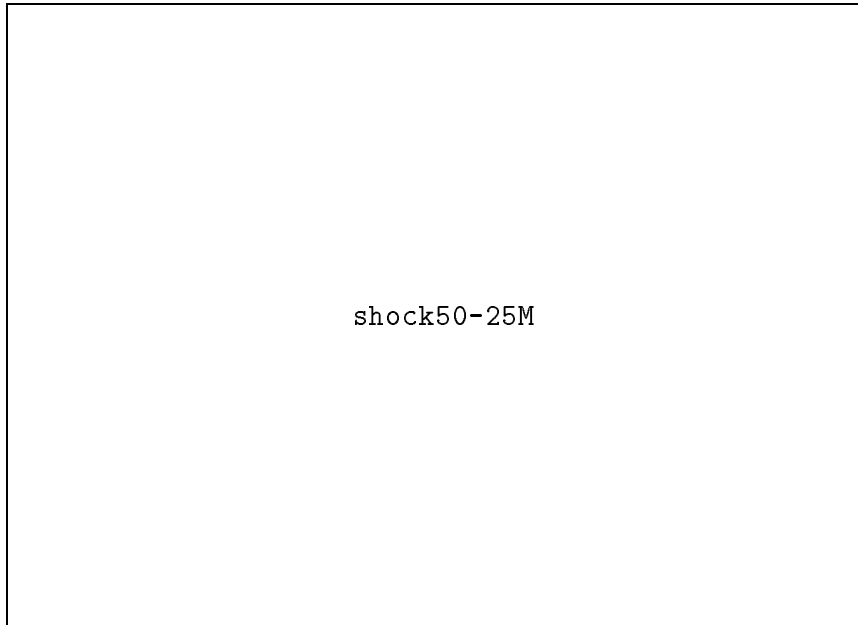
Plate 8a: Shock tube, 50%–25%

Early in the wave evolution, the front is identifiable as the circular arc heading down-stream, while the jet lags much further behind. Later, the jet has broken down in the channel and complex recirculating eddies have filled the tube.

```
shock50-10M
```
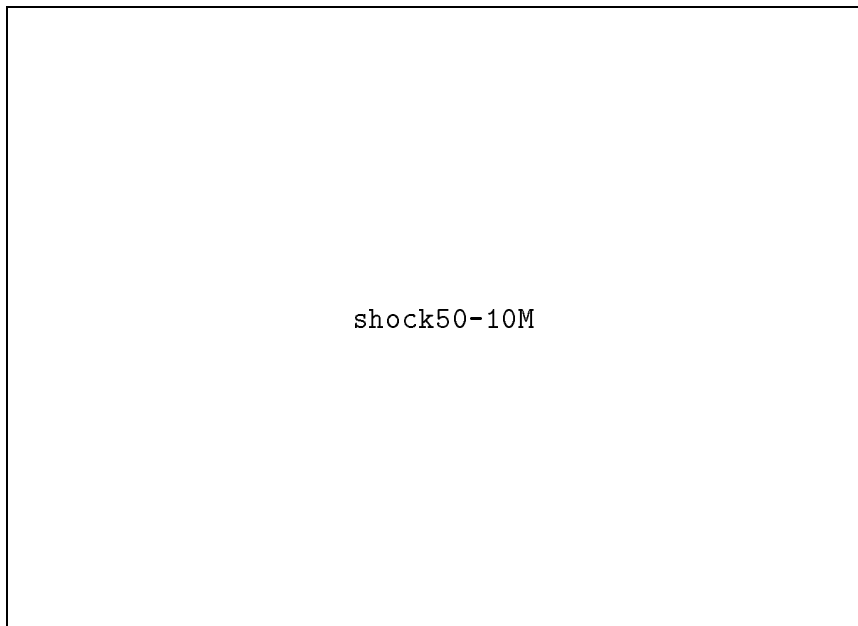
Plate 8b: Shock tube, 50%–10%

Where the pressure ratios are large, as in this case, long multi-stranded jets are formed where preferential streaming along the lattice axes occurs. The front propagation rate is found to be independent of the pressure difference and the density (**Section 4.2.1**).
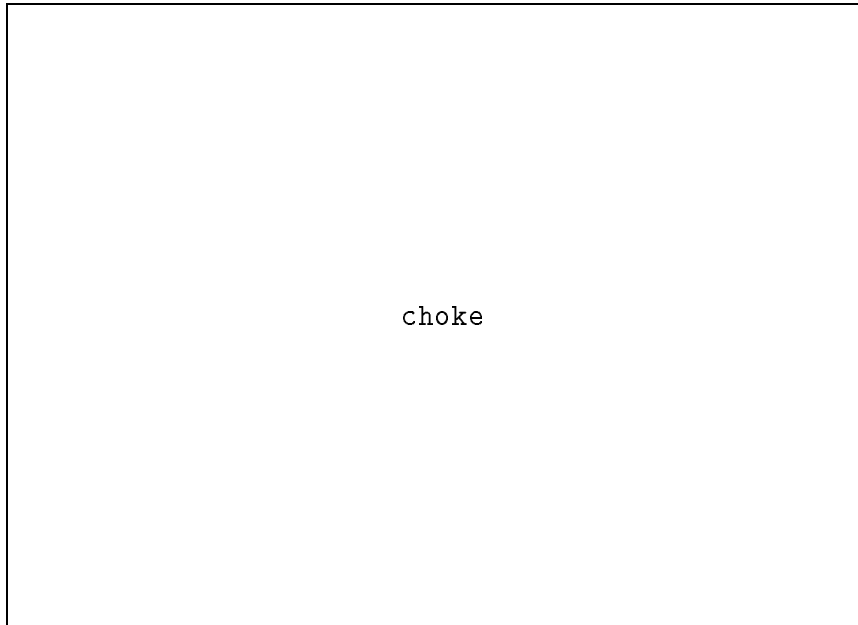
Plate 9a: Choked flow in a nozzle

Flow between two reservoirs through a smooth nozzle has *choked* in its throat (see **Section 4.3**). Instability of the exit jet is also starting in the exit reservoir. The position of the microlattice probe to the right of the throat can be seen as a small rectangle.
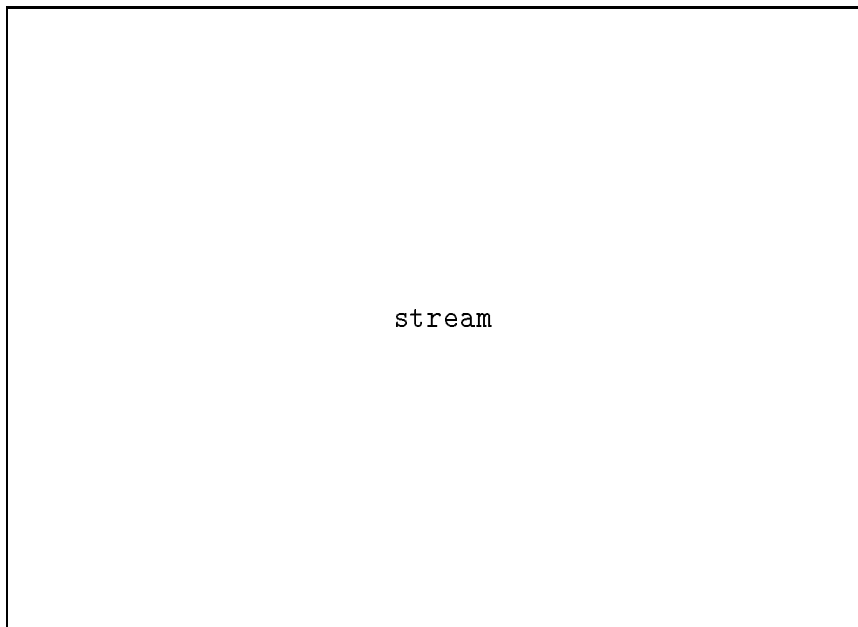

**Plate 9b: Microlattice of nozzle showing streaming**

Where there is a very marked deficiency of opposing particles, unphysical *streaming* along the lattice axes occurs. This sample was taken from the throat of the nozzle in **Plate 9a**.

It's 3 o'clock in the morning
  And the rain begins to fall
I know what I'm needing
  But I can't have it all

— [ZZTop83]

10 p.m.
  I feel the rain coming down
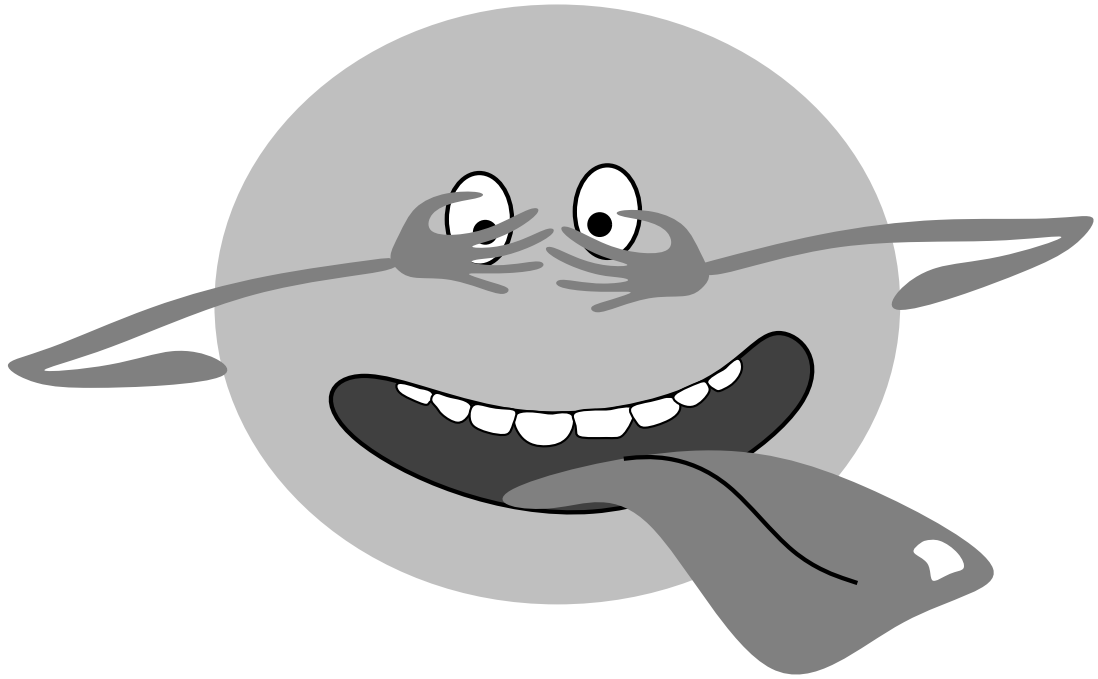*My face feels the wet*
  My mind the storm

— [Queensrÿche88]
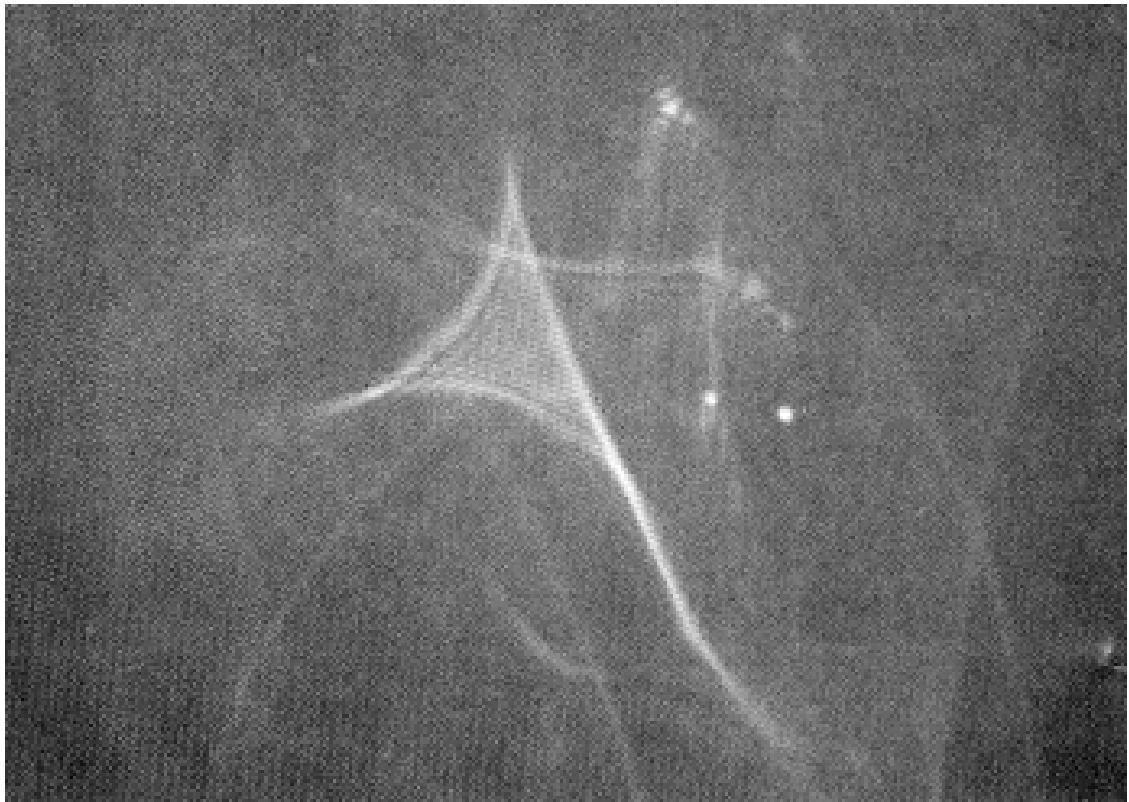
— [TD86]

Súil tríd na stoirmeacha
Dul tríd na stoirmeacha

Cá fhad é ó
  An tús don stoirm
Cá fhad é ó
  An tús go deireadh

— [Enya88]

But I must not forget
  That I am what I am
And you must not forget
  That you are what you are

It is dark ... in the meadow

— [AAE88]

Reassurance Panel

In case of doubt, confusion or alarm, please touch this panel:

> # Hi there

At times of stress it is often reassuring to make physical contact
with friendly objects.   This panel is your friend.

— [*Adams87*]