Using this formalism we now write the equation of motion for the ith particle in the form:

$$d\mathbf{v}_i/dt = -(\nabla P/\rho)_i - \nabla_i \Phi + (\mathbf{F}_i)_{\text{visc}} \tag{2.4}$$

with $(\nabla P/\rho)_i = \sum_j m_j P_j/\rho_j \rho_i \nabla_i w(r_{ij}, h_j)$.

Here $r_{ij} = |\mathbf{r} - \mathbf{r}_i|$, Φ is the external potential (gravitational, for example) and $(\mathbf{F}_i)_{\text{visc}}$ is the term due to natural and artificial viscosity. To ensure better conservation of the momentum and energy, Gingold and Monaghan [6] have rewritten the pressure terms,

$$-\sum_j m_j (P_j/\rho_j^2 + P_i/\rho_i^2) \nabla_i w(r_{ij}, h_j) \tag{2.5}$$

which can be further symmetrised by taking $h_{ij} = (h_i + h_j)/2$ (see [4, 9]), to satisfy the action reciprocity principle.

However, in the problems that we have treated, the difference with the direct terms (more economic in CPU) was not significant, and we did not use this complex form. Indeed, we have tested the *local* conservation of momentum by checking that:

$$d\mathbf{v}/dt = -\nabla P/\rho - \nabla \Phi + \mathbf{F}_{\text{visc}}, \tag{2.6}$$

i.e., by checking the accuracy of the Eulerian equation of motion, when all terms are estimated by the SPH smoothing procedure, in other words by averaging all physical quantities over all neighbours. If momentum was not conserved because of asymmetries between h_i and h_j, Eq. (2.6) would not be satisfied, even if all particles were advanced according to the Lagrangian equation (2.4).

The tests have been performed at 1D on the isothermal shock problem and in 2D with the Gaussian density distribution (cf. discussion in Section 3). Equation (2.6) was verified within 1%. This can be explained by our determination of h_i (Section 4), which leads a slowly varying h over the spatial resolution of the simulations.

We will go back to the smoothing length and artificial viscosity in Section 4.

Algorithms of the Vectorisation

The equations of Section 2 show that the pressure and force calculations are completed by summations over neighbours, with weighting W. Since the average number of neighbours (around 21 with our kernel W_3, see Section 4) is small, it is not efficient to vectorise these summations directly. As the number of particles N is of the order of 10^4, the summations using vectors of size N will be much more gratifying. The direct method would consist of, for each given particle, a search for its neighbours and then calculation of the corresponding physical quantities. The method presented here proceeds in the reverse way: it is necessary to define in advance the neighbours of each particle and to construct arrays storing all physical quantities corresponding to these neighbours.

To optimise the search of neighbours, we proceed in several steps. In the first step we superpose a grid on the ensemble of particles, to locate each particle, and then