INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2006; **50**:1381–1404
Published online 9 November 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1100

A mesh patching method for finite volume modelling of shallow water flow

Keming Hu[1,*,†], C. G. Mingham[2,‡] and D. M. Causon[2,§]

[1]*Black & Veatch, Grosvenor House, 69 London Road, Redhill RH1 1LQ, U.K.*
[2]*Centre for Mathematical Modelling and Flow Analysis, Manchester Metropolitan University, Chester Street, Manchester M1 5GD, U.K.*

SUMMARY

A new mesh-patching model is presented for shallow water flow described by the 2D non-linear shallow water (NLSW) equations. The mesh-patching model is based on AMAZON, a high-resolution NLSW engine with an improved HLLC approximate Riemann solver. A new patching algorithm has been developed, which not only provides improved spatial resolution of flow features in particular parts of the mesh, but also simplifies and speeds up the (structured) grid generation process for an area with complicated geometry. The new patching technique is also compatible with increasingly popular parallel computing and adaptive grid techniques. The patching algorithm has been tested with moving bores, and results of test problems are presented and compared to previous work. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: nonlinear shallow water equations; finite volume method; mesh patching; multi-block; approximate Riemann solver

1. INTRODUCTION

The patching technique, also known as the *zonal approach* [1] in computational aerodynamics, was originally developed to tackle difficulties associated with grid generation over complex geometries. Although a considerable level of flexibility has been obtained with single-block structured grids, there are many applications especially in aerodynamics involving multiply connected domains, which may not be suitably discretized by a single structured grid [2]. Recently, increased interest in parallel computing has led to a need for more accurate, efficient and reliable patching algorithms, also called multi-block algorithms in the parallel computing community. Although unstructured grids have great flexibility in terms of grid

*Correspondence to: K. Hu, Black & Veatch, Grosvenor House, 69 London Road, Redhill RH1 1LQ, U.K.
†E-mail: huk@bv.com
‡E-mail: c.mingham@mmu.ac.uk
§E-mail: d.m.causon@mmu.ac.uk

Received 31 January 2004
Revised 4 June 2005
Accepted 18 August 2005

adaptation, research on patching techniques with their simpler data structures has focused on the structured grids, particularly the quadrilateral grid system. Patching algorithms for structured grids can be classified into overlaid, e.g. References [3–5], and non-overlaid systems, e.g. References [1, 2].

Probably the development of patching techniques in computational hydraulics was initially associated with the problems with mesh nesting [6]. In a nested model, a fine grid is embedded in a coarse gird to improve the grid resolution in areas of interest, but the components are not hydrodynamically linked. Nevertheless, the nesting technique has been widely used in tidal and wave modelling in coastal engineering in spite of this potential defect. The ideal solution to this problem is the patching technique where the individual mesh components are hydrodynamically linked. However, in many existing popular implicit finite difference solvers, the technique is difficult to implement, and such research in computational hydraulics is rarely reported.

Compared to mesh nesting, mesh patching has many other uses apart from the advantage of a hydrodynamically-linked connection between the main flow model and sub-models. Unlike the nesting technique, a patched model does not have to be embedded in the main model. It can simply be an extension of the main model or another sub-model. This attribute is most useful for modelling an estuarine area with complicated mud flats and water causeways. A rectilinear Cartesian grid with its saw-tooth representation of irregular boundaries is insufficiently accurate, and a curvilinear model based on the equation transformation method may have difficulty in generating a suitable structured grid. Using the patching technique, a computational grid is generated separately for each region, and sub-grids that may vary in resolution from one area to another are then linked together. This approach simplifies the generation of a suitable curvilinear structured grid.

Furthermore, the patching technique can be used as a basis for a so-called adaptive grid system. An adaptive grid is useful for tracking a moving flow phenomenon. This requires a fine grid generated locally to the moving phenomenon so that the hydraulic details can be revealed. An efficient grid generation method is essential as the grid may need to be re-generated at each time step. Using the patching technique, a front tracking sub-grid model is embedded in the main model, which may occupy a block of a few cells in the main model. The tracking sub-grid is based on the splitting of relevant cells of the main grid. This approach requires little extra computational effort since only the sub-grid model requires grid re-generation. This resembles a quad-tree grid in an adaptive grid system [7] but without the small time step problem for the whole grid.

In summary, the benefits of the patching technique are:

(1) simplifies the generation of a boundary-fitted curvilinear grid;
(2) enhances the ability to adopt a resolution-varying grid for higher computing efficiency;
(3) improves computing speed by using local time steps and the one-dimensional CFL condition, an advantage of the structured grid (see Section 3.3);
(4) compatible with parallel computing;
(5) extendable to an adaptive grid system.

Although successfully patching techniques to the Reynolds-averaged Navier–Stokes equations were report by Furukawa *et al.* [1] and Lien *et al.* [2], the authors are not aware of any similar studies on the non-linear shallow water equations and MUSCL re-construction [8] with Hancock time integration method [9] associated with modern upwind Godunov schemes.

Difference in time steps between two models prevents straightforward implementation of the MUSCL re-construction of values at boundaries at the corrector step.

A new patching algorithm is presented in this paper, which is based on a non-linear shallow water (NLSW) numerical engine AMAZON [10], which uses a cell-centred finite volume method with a structured quadrilateral mesh. The flux-balance form of the finite volume method provides a reliable platform to implement a patching algorithm. The conservation of mass and momentum is assured by proper treatment of the numerical fluxes at the patching interface. The new patching algorithm has been tested, and the potential problem in time matching for second-order accuracy is carefully addressed.

The new model is written in an object-orientated form. The model provides two basic patching operators, i.e. *attach* and *embed*. Each model is treated as an 'object', which can be *attached* or *embedded*. This provides effortless multiple attach and embed connections. The test results demonstrate that the patching model can dramatically reduce the computation time while the same resolution used by the non-patching model is maintained in the area of interest.

2. GOVERNING EQUATIONS

The shallow water equations can be written in integral form as an expression of the fundamental laws of conservation of mass, momentum and energy:

$$\frac{\partial}{\partial t}\int_{\Omega}\mathbf{U}\,\mathrm{d}\Omega+\int_{\Gamma}\mathbf{H}\bullet\mathrm{d}\mathbf{S}=\int_{\Gamma}\mathbf{Q_S}\bullet\mathrm{d}\mathbf{S}+\int_{\Omega}\mathbf{Q_V}\,\mathrm{d}\Omega \tag{1}$$

$$\mathbf{U}=\begin{pmatrix}\phi\\ \phi u\\ \phi v\end{pmatrix},\quad H=\begin{pmatrix}\phi\mathbf{q}\\ \phi u\mathbf{q}+\frac{1}{2}\phi^2\vec{i}\\ \phi v\mathbf{q}+\frac{1}{2}\phi^2\vec{j}\end{pmatrix},\quad \phi=gD \tag{2}$$

$$\mathbf{Q_S}=\begin{pmatrix}0\\ \phi gh+\dfrac{1}{\rho}\phi\tau_{xx}\,\vec{i}+\dfrac{1}{\rho}\phi\tau_{xy}\,\vec{j}\\ \phi gh+\dfrac{1}{\rho}\phi\tau_{xy}\,\vec{i}+\dfrac{1}{\rho}\phi\tau_{yy}\,\vec{j}\end{pmatrix} \tag{3}$$

$$\mathbf{Q_V}=\begin{pmatrix}0\\ fv\phi+\dfrac{g}{\rho}(\tau_{wx}-\tau_{bx})\\ -fu\phi+\dfrac{g}{\rho}(\tau_{wy}-\tau_{by})\end{pmatrix} \tag{4}$$

where u, v are the depth-averaged velocity components in the horizontal x, y directions and $\mathbf{q}$ is the vector of the depth-averaged velocity; D is the depth of water; f is the Coriolis force; τ_{bx} and τ_{by} are the bed friction shear stresses in x and y directions, respectively; τ_{wx} and τ_{wy} are the surface wind shear stresses in x and y directions, respectively; Ω is an arbitrary

2D region with boundary Γ; $\mathbf{S}$ is an outward-pointing surface normal vector; $\vec{i}$ and $\vec{j}$ are unit vector in x and y directions respectively; and $\mathbf{H}$ is the flux tensor for inviscid flows. The source terms are contained within the vector $\mathbf{Q}_\mathrm{S}$ representing bed slope and viscous terms, and the vector $\mathbf{Q}_\mathrm{V}$ containing bed and surface stresses; h is the bed level below an arbitrary datum.

3. NUMERICAL SCHEMES FOR NLSW

The NLSW model described by Equations (1)–(4) includes the convective, bed slope, viscous, Coriolis force, bed and surface stress terms. From the numerical viewpoint, most of the problems arise from the non-linear convective terms. Using term-by-term operator splitting [11], the shallow water equations can be solved by decomposing them into the following two split equations:

$$\frac{\partial}{\partial t}\int_\Omega \mathbf{U}\,\mathrm{d}\Omega + \int_\Gamma \mathbf{H}\bullet \mathrm{d}\mathbf{S} = 0 \tag{5}$$

$$\frac{\partial}{\partial t}\int_\Omega \mathbf{U}\,\mathrm{d}\Omega = \int_\Gamma \mathbf{Q}_\mathrm{S}\bullet \mathrm{d}\mathbf{S} + \int_\Omega \mathbf{Q}_\mathrm{V}\,\mathrm{d}\Omega \tag{6}$$

3.1. *Inviscid terms*

Equation (5) can be discretized as follows:

$$\frac{\partial}{\partial t}\mathbf{U}_{i,j}\Omega_{i,j} + \sum_{m=1}^{M}\mathbf{H}_m\bullet\mathbf{S}_m = 0 \tag{7}$$

where Ω_{ij} denotes the area of cell (i,j), $\mathbf{S}_m$ is the outward pointing normal vector to side m of cell (i,j) whose length is the length of the side, $\mathbf{H}_m$ represents the numerical flux tensor at cell interface m with the sum taken over the number of sides M of cell (i,j). $\mathbf{U}_{ij}$ is the integral average value of the flow solution vector $\mathbf{U}$ over the cell (i,j) located at its centre.

The numerical method we use is a Godunov-type upwind scheme and a detailed exposition can be found in Reference [12]. Godunov [13] showed how to make use of characteristic information within the framework of a conservative method. He proposed that the numerical flux could be obtained by solving a local Riemann problem at each cell interface. In one dimension, the Godunov scheme can be expressed as

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n - \frac{\Delta t}{\Delta x}[\mathbf{F}(\mathbf{U}_{i+1/2}^\mathrm{L},\mathbf{U}_{i+1/2}^\mathrm{R})_{x/t=0} - \mathbf{F}(\mathbf{U}_{i-1/2}^\mathrm{L},\mathbf{U}_{i-1/2}^\mathrm{R})_{x/t=0}] \tag{8}$$

where $\mathbf{F}$ represents the numerical flux at the cell interface obtained by solving a local Riemann problem using the data $\mathbf{U}^\mathrm{R}$ and $\mathbf{U}^\mathrm{L}$ on each side of the cell interface.

Exact Riemann solvers are computationally expensive, particularly for nonlinear systems. Fortunately, a number of computationally inexpensive approximate Riemann solvers have been developed. The HLL approximate Riemann solver [14] has been found to be robust in applications and simple to implement [15]. However, the HLL approximate Riemann solver ignores the existence of the contact wave (Figure 1). Without consideration of the contact wave, any difference in the flux components parallel to a cell interface is assumed to take place across

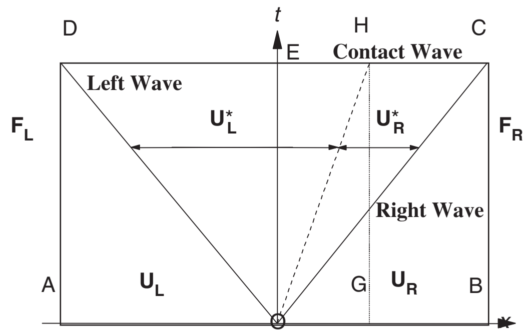

Figure 1. Riemann problem and contact wave.

the interface. This assumption is not always physically correct. In some applications, for example in the test case of *Oblique Bore Reflection* (Section 5.3), this defect can lead to poor numerical resolution around the contact surface [10].

To overcome this problem, Toro *et al.* [16] introduced the HLLC approximate Riemann solver. Subsequently, Hu [10] suggested an improved HLLC approximate Riemann solver (referred to HLLCM in this paper).

The first-order Godunov method uses a piecewise constant approximation to reconstruct the data at cell interfaces from the stored cell centre data. In a MUSCL (stands for monotone upstream-centred schemes for conservation laws) scheme [8], the piecewise constant approximation is replaced by piecewise linear approximation, which gives second-order spatial accuracy:

$$\hat{\mathbf{U}}^{\mathrm{R}}_{i+1/2,j} = \mathbf{U}_{i+1,j} - \tfrac{1}{2}\Psi(\mathrm{r}_{i+1,j}) \bullet \delta\mathbf{U}_{i+1/2,j} \tag{9}$$

$$\hat{\mathbf{U}}^{\mathrm{L}}_{i+1/2,j} = \mathbf{U}_{i,j} + \tfrac{1}{2}\psi(\mathrm{r}_{i,j}) \bullet \delta\mathbf{U}_{i-1/2,j} \tag{10}$$

where $\delta\mathbf{U}_{i+1/2,j} = \mathbf{U}_{i+1,j} - \mathbf{U}_{i,j}$, $\delta\mathbf{U}_{i-1/2,j} = \mathbf{U}_{i,j} - \mathbf{U}_{i-1,j}$, $\mathbf{r}_{i,j} = \delta\mathbf{U}_{i+1/2,j}/\delta\mathbf{U}_{i-1/2,j}$, $\mathbf{r}_{i+1,j} = \delta\mathbf{U}_{i+3/2,j}/\delta\mathbf{U}_{i+1/2,j}$. ψ is a slope limiter function and $\mathbf{r}$ is the ratio of successive gradients. The slope limiter function ψ limits the gradient in order to avoid non-physical overshoots or undershoots in the reconstructed data at each cell interface. We use the well-known van Leer limiter for the present test problems:

$$\Psi = \frac{\mathbf{r} + |\mathbf{r}|}{1 + \mathbf{r}} \tag{11}$$

To maintain stability and obtain second-order accuracy in time, a Hancock two-stage time integration method is applied [10].

3.2. Source terms

The split Equation (6) for the viscous terms, incorporating the remaining source terms, can be solved by the implicit Euler method after evaluating the fluxes around cell interfaces using Green's theorem. To construct the viscous numerical fluxes, it is necessary to evaluate first-order derivatives of the velocities at each cell interface. This is not a trivial task particularly

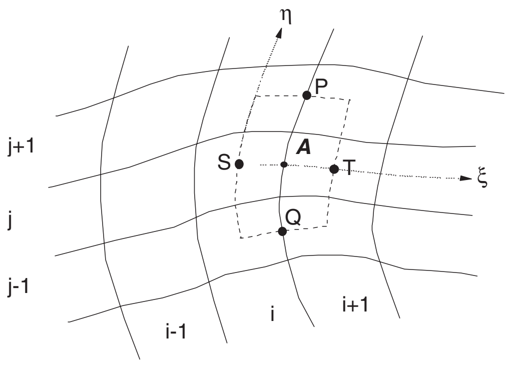

Figure 2. Evaluation of first-order derivatives.

if the mesh is distorted. By the method of Jacobian transformation, Deiwert [17] provided the following approximations for the first-order derivatives:

$$\frac{\partial(\bullet)}{\partial x} = \frac{\Delta(\bullet)_\xi \Delta y_\eta - \Delta(\bullet)_\eta \Delta y_\xi}{\Delta x_\xi \Delta y_\eta - \Delta x_\eta \Delta y_\xi} \tag{12}$$

$$\frac{\partial(\bullet)}{\partial y} = \frac{\Delta(\bullet)_\xi \Delta x_\eta - \Delta(\bullet)_\eta \Delta x_\xi}{\Delta y_\xi \Delta x_\eta - \Delta y_\eta \Delta x_\xi} \tag{13}$$

where $(\cdot)$ denotes u or v; ξ and η are two-dimensional directions in curvilinear grid. For the interface A illustrated in Figure 2, the differences of u, v, x and y can be obtained using an auxiliary control volume Ω_A, i.e. $\Delta(\cdot)\xi = (\cdot)_{i,j+1} - (\cdot)_{i,j}$ and $\Delta(\cdot)\eta = (\cdot)_\mathrm{P} - (\cdot)_\mathrm{Q}$, where $(\cdot)_\mathrm{P}$ is interpolated from cell values of $(\cdot)_{i+1,j}$ and $(\cdot)_{i+1,j+1}$, and $(\cdot)_\mathrm{Q}$ is interpolated from cell values of $(\cdot)_{i-1,j}$ and $(\cdot)_{i-1,j+1}$. To achieve second-order spatial accuracy, the values of the six cells encompassed by Ω_A are used to provide the numerical flux at cell interface A. The Jacobian approximation relies on some degree of smoothness of the mesh and the mesh should not be too distorted.

3.3. *Stability conditions and operator splitting*

The explicit scheme for the inviscid terms gives rise to the CFL time step constraint. One of the advantages of structured grid is the use of operator splitting to increase the time step [11]. The time step for 2D inviscid flow with dimensional splitting is given by [10]

$$\Delta t^i = \alpha \,\mathrm{Min}\left\{\frac{\Omega_{i,j}}{(\mathbf{q}_i \cdot \mathbf{S}_i + c|\mathbf{S}_i|)}\right\} \tag{14a}$$

$$\Delta t^j = \alpha \,\mathrm{Min}\left\{\frac{\Omega_{i,j}}{(\mathbf{q}_j \cdot \mathbf{S}_j + c|\mathbf{S}_j|)}\right\} \tag{14b}$$

$$\Delta t = \mathrm{Min}(\Delta t^i, \Delta t^j) \tag{14c}$$

where Ω denotes the area of a cell; $c=\phi^{1/2}$; α is the Courant number; and i, j denote the two coordinate directions of a structured grid. It is the one-dimensional CFL condition. If cell sizes and velocities are equal in the i and j directions, the time step can be as twice long as the one by the two-dimensional CFL condition using unstructured grid.

3.4. Boundary conditions

For the problems studied in this paper, two types of boundary conditions are required, namely fully-slip solid and transmissive boundary conditions. These are imposed as follows [18]:

(i) Fully-slip solid boundary conditions

$$\phi_{M+1}=\phi_M, \quad u'_{M+1}=-u'_M, \quad v'_{M+1}=v'_M \tag{15}$$
$$\phi_{M+2}=\phi_{M-1}, \quad u'_{M+2}=-u'_{M-1}, \quad v'_{M+2}=v'_{M-1}$$

(ii) Transmissive boundary conditions

$$\phi_{M+1}=\phi_M, \quad u'_{M+1}=u'_M, \quad v'_{M+1}=v'_M$$
$$\phi_{M+2}=\phi_{M-1}, \quad u'_{M+2}=u'_{M-1}, \quad v'_{M+2}=v'_{M-1} \tag{16}$$

where $M-1$ and M denote the last two cells adjacent to the boundary inside the computational domain (Figure 3). $M+1$ and $M+2$ are two fictitious cells ('ghost cells') outside the computational domain and u' and v' are the components of velocity $\mathbf{q}$ normal and tangential to the boundary, respectively. Transmissive boundaries, also called 'transparent boundaries', allow waves to pass through without reflection.

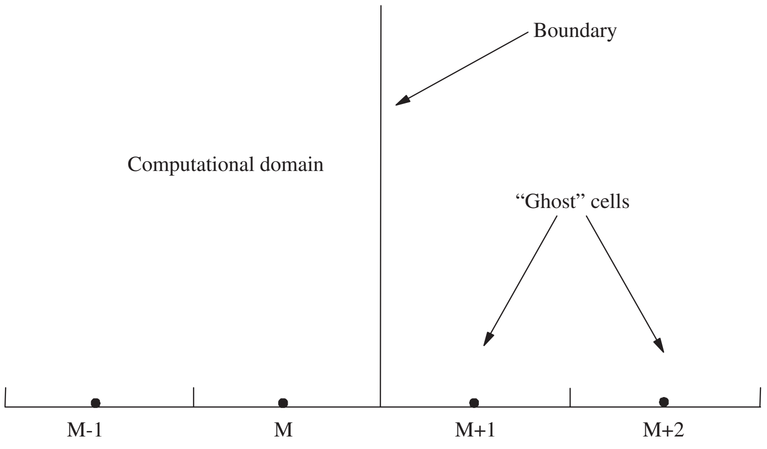

Figure 3. Sketch of computational cells inside and outside boundary.

4. PATCHING ALGORITHM

4.1. Patching operators and interface zone

The new patching algorithm is an non-overlaid system. However, overlaid grids can be converted to non-overlaid grids in a straightforward manner. Consider an overlaid patched grid system illustrated in Figure 4. The overlaid patched grid can be converted to a non-overlaid grid by introducing an interface line (Figure 4). Therefore, the new patching algorithm is also applicable to overlaid patching with a pre-processor to establish the interface line.

Any patching system can be expressed using two basic patching connections, i.e. *embed* and *attach*, which are illustrated in Figures 5 and 6. It is interesting to note that a model can be attached to itself, i.e. one side of a model can be attached to other side of itself (Figure 7).This can be directly applied to an O-type branch-out grid configuration in curvilinear grid generation [19]. The 'attach-to-itself' configuration is used in the *circular dam break* test problem (see Section 5.2).

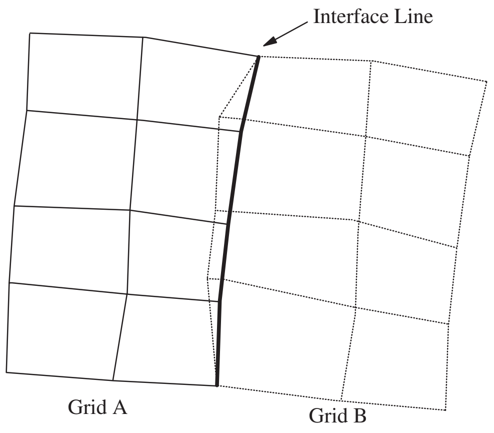

Figure 4. Sketch showing an interface line for overlaid grids.

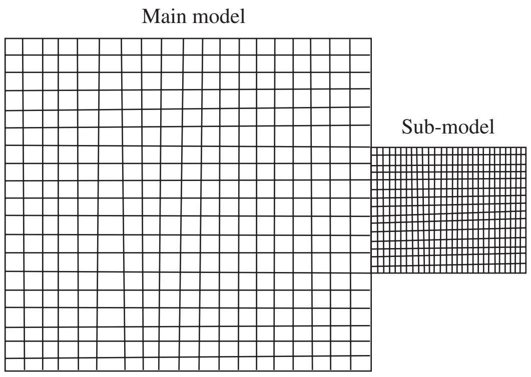

Figure 5. Sketch of the *attach* connection.

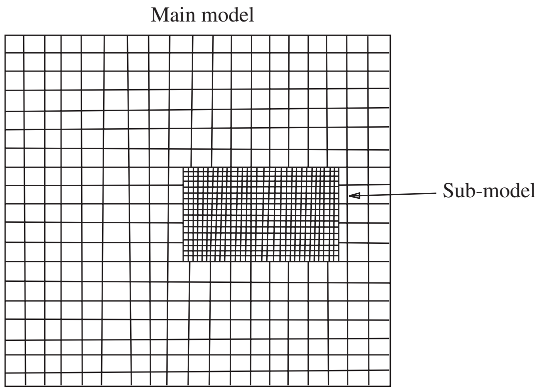

Figure 6. Sketch of the *embed* connection.

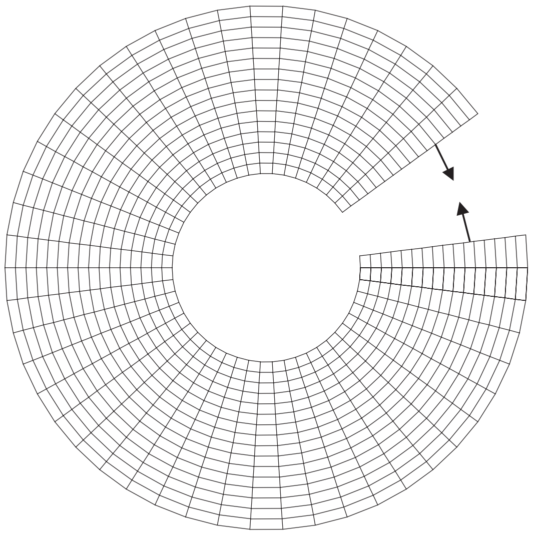

Figure 7. One side of a grid attaches to the other side of itself.

In MUSCL reconstruction, to maintain second-order spatial accuracy, an interface zone between two grids is defined (Figure 8). The original cells in the interface zone may be divided into small cells (referred to as sub-cells) to match the opposite grid. The use of a sub-cell is for MUSCL reconstruction only, and the details are discussed in the next section.

4.2. Numerical inviscid flux

To ensure the conservation of mass and momentum during the process of patching, it is important that the models are linked by the numerical flux, i.e. fluxes are conserved. In first-order patch modelling, cell division into sub-cells is not required because of the assumption

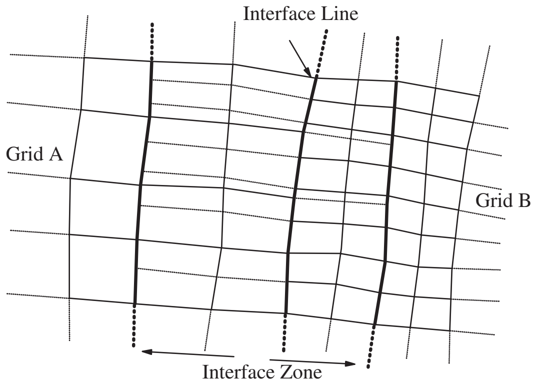

Figure 8. Sketch of the patching interface zone.

of piecewise constant data at each cell interface. However, MUSCL reconstruction to second order is not straightforward at the patch interface. The piecewise linear approximation requires values of **U** from neighbouring cells. At the patch interface, one neighbouring cell sits in the opposite model.

For the purposes of MUSCL reconstruction, the original cell in the interface zone is divided into sub-cells to match the grid in the other model. The values of **U** in the sub-cell are set to the same values as the original cell of which the sub-cell is a part. It may be argued that the values of **U** in the sub-cell could be spatially interpolated with respect to the location of the sub-cell centre. Nevertheless, our numerical experiments show that the former treatment provides better results. This may indicate that of more importance is the need to conserve mass and momentum in the solution algorithm.

The process of MUSCL reconstruction and consequent numerical flux computation is carried out on sub-cells at the patch interface. The numerical flux at the original cell at the patch interface is the sum of numerical fluxes in its all sub-cells:

$$\mathbf{F}_{i,j} = \sum_{k}^{m} \mathbf{F}_{i,j}^{(k)} \tag{17}$$

where $\mathbf{F}_{i,j}$ is the numerical inviscid flux for the original cell at the patch interface which consists of m sub-cell boundaries.

4.3. Numerical flux for source terms

For the source terms including the viscous terms, the patching process is less complicated. The only remaining problem may be the evaluation of first-order derivatives for the velocities at the cell boundary. The Deiwert method described in Section 3.2 for the viscous terms can be applied in a straightforward manner. Assuming the interface A is located at the patch interface (Figure 2), the only difference from the normal calculation for first-order derivatives is that the values at point T are obtained by interpolation within the right-side model and the values at points P and Q are obtained by interpolation between the two models. It should be noted that interpolation is used only for the values of the first-order derivatives of the velocities. Conservation of the numerical flux for the source terms, including the viscous terms, is satisfied by directly evaluating the flux around the cell interfaces using the finite volume formulation.

4.4. Time marching

Since the local time steps may vary from one sub-model to another, the patching algorithm allows the use of different time steps in each sub-model for improved computational efficiency. For each step of the whole patching model, there may be several solution steps in a sub-model.

The time marching of a sub-model can be expressed in a form of:

$$L^{\mathrm{p}}(\Delta t)L^{\mathrm{c}}(\Delta t)=\{L^{\mathrm{p}}(\Delta t_1')L^{\mathrm{c}}(\Delta t_1')\}\{L^{\mathrm{p}}(\Delta t_2')L^{\mathrm{c}}(\Delta t_2')\}\cdots\{L^{\mathrm{p}}(\Delta t_k')L^{\mathrm{c}}(\Delta t_k')\} \tag{18}$$

where L^{p} and L^{c} represent the predictor and corrector steps of the two-stage Hancock time integration scheme [10]; Δt is the time step for the whole patched model, and Δt_i is the maximum local time step for a sub-model *i;* $\Delta t_i'$ is the actual local time step, $\Delta t_i'=\Delta t/k$ and $k=\mathrm{int}((\Delta t-\delta)/\Delta t_i)+1$; δ is a small number ($\leqslant 10^{-6}$). When $\Delta t_i'=\Delta t$ ($k=1$), it is single step time marching.

In first-order patch modelling, the operator sequence (18) is applied in each sub-model independently. As MUSCL reconstruction is required at the second stage of the Hancock time integration method, implementation of (18) at each sub-model is no longer independent for the second-order patch modelling, and the following time marching procedure is introduced:

Time marching procedure for second-order patching

1. Hancock first-stage MUSCL reconstruction for all sub-models.
2. Predictor for the first time step of all sub-models.
3. Hancock second-stage MUSCL reconstruction for all sub-models.
4. Corrector for the first time step of all sub-models.
5. Complete the following loop for sub models which have more than one time step.
    - 5.1. First-stage MUSCL reconstruction.
    - 5.2. Predictor.
    - 5.3. Second-stage MUSCL reconstruction.
    - 5.4. Corrector.
    - 5.5. Repeat 5.1–5.4 until all time steps have been done for this sub-model.
6. Repeat 5.1–5.5 until time integration has been done for all sub-models.

The above time marching procedure is illustrated in Figure 9. The first step of this procedure is that all sub-models are advanced by one local time step of each patch. The sub-model with largest grid size will be updated by a full time step by a single iteration whilst other sub-models may require a few more local time steps to achieve the same time level. The potential problem of this procedure is that, at second-stage MUSCL reconstruction, variables at two sides of a patch interface may be held at different time levels. This may be argued that repeat of the second-stage MUSCL reconstruction is needed in the sub-models with larger local time steps. It can be a time consuming process if there are many different local time steps across the whole model. Our numerical experiments show there is little benefit in correcting second-stage MUSCL reconstruction in coarse-grid sub-models. This may be explained by the

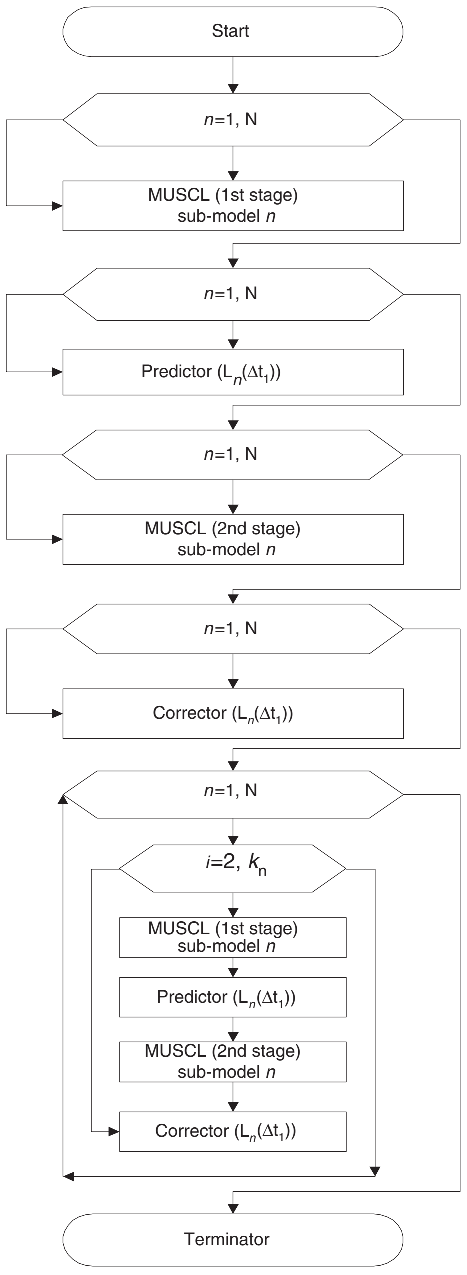
Start
n=1, N
MUSCL (1st stage) sub-model n
n=1, N
Predictor ($L_n(\Delta t_1)$)
n=1, N
MUSCL (2nd stage) sub-model n
n=1, N
Corrector ($L_n(\Delta t_1)$)
n=1, N
i=2, k_n
MUSCL (1st stage) sub-model n
Predictor ($L_n(\Delta t_1)$)
MUSCL (2nd stage) sub-model n
Corrector ($L_n(\Delta t_1)$)
Terminator

Figure 9. Time marching procedure for second-order patch modelling.

fact that the MUSCL reconstruction is used to suppress non-physical over-shoots or under-shoots problem due to arrival of bores, and an arriving shock from a fine-grid sub-model will be tackled in the following local time step if it is missed in the previous local time step.

It is important to note that much of this algorithm consists of independent computations over a number of patches and thus the code is easily parallelizable for increased computing speed.

5. NUMERICAL EXPERIMENTS

In the following test problems, the van Leer slope limiter is adopted for all tests and the Courant number is set to unity.

5.1. Bore wave travelling through an embedded fine grid

The problem domain was a rectangular basin of 450 m wide and 600 m long. The computation grid for the main model had 30×40 cells, i.e. $\Delta x = \Delta y = 15$ m. A fine-grid sub-model 150 m wide by 300 m long was embedded in the middle of the main model (Figure 10). The edges of the embedded model were 150 m away from the four edges of the main model. Only inviscid flow was considered here. Initially, the water depth in the computational domain was set to 1.0 m and the water was at rest.

A right travelling bore was assumed to enter the domain from the left boundary. A transmissive boundary (defined in Section 3.4) was used at the other three boundaries. For the incident bore, three different Froude numbers (Fs) were tested, namely 2, 3 and 4. The computation was stopped when the bore reached the middle of the domain.

The left boundary conditions were obtained using the following equations [20]:

$$\phi_{\mathrm{L}} = d\phi_{\mathrm{R}}, \quad u_{\mathrm{L}} = \left(1 - \frac{1}{d}\right)s, \quad v_L = 0 \tag{19a}$$

$$d = 0.5\left(\sqrt{1 + 8F_S^2} - 1\right) \tag{19b}$$

where $\phi_{\mathrm{R}} = \sqrt{gh_{\mathrm{R}}}$, $h_{\mathrm{R}} = 1$; s is the propagation speed of the incident bore and $s = F_S\sqrt{\phi_{\mathrm{R}}}$.

Three different grid resolutions were tested for the embedded sub-model. They were $7.5\,\mathrm{m} \times 7.5\,\mathrm{m}$, $3.75\,\mathrm{m} \times 3.75\,\mathrm{m}$ and $1.875\,\mathrm{m} \times 1.875\,\mathrm{m}$, i.e. resolution ratios of the main model and the sub-model were 1:2, 1:4 and 1:8. The results are presented in the form of water depth contours (Figures 11–13). The interval of the water depth contours is 0.1 m. Table I summarizes the mean errors calculated within the patched area. The errors are based on the comparison with a fine-grid non-patched model. The left boundary of this non-patched model is at the same location of the left boundary of the patched model, and boundary conditions are provided by the coarse-grid model. Therefore, Table I presents the errors originated along the top and bottom of patch interfaces.

A bore wave travelling along the patch interface provides an extreme condition for testing the patch model, which is the case along the top and bottom of the embedded model. It is inevitable that there are differences in the conservative variables between two models due to a difference in the grid resolution. The magnitude of the error introduced along the patching interface depends on the Froude number and the resolution ratio between the two models.

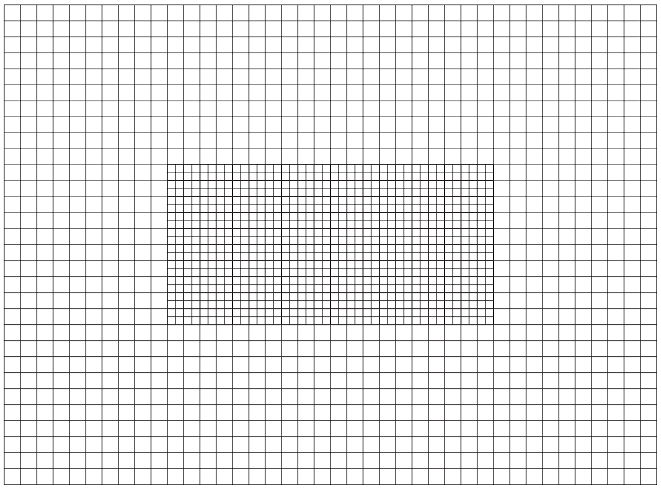

Figure 10. Test 5.1—sketch of the embedded grid.

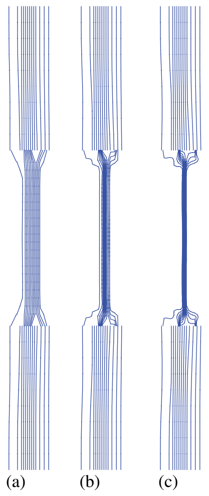

Figure 11. Test 5.1—water depth contours: (a) $Fs = 2$ and resolution ratio = 1:2; (b) $Fs = 2$ and resolution ratio = 1:4; and (c) $Fs = 2$ and resolution ratio = 1:8.

Table I. Average errors of the patched model (water depth within patched area).

Froude no.	2	2	2	3	3	3	4	4	4
Resolution ratio	1:2	1:4	1:8	1:2	1:4	1:8	1:2	1:4	1:8
Average error (%)	0.16	0.26	0.36	0.25	0.42	0.57	0.28	0.46	0.62

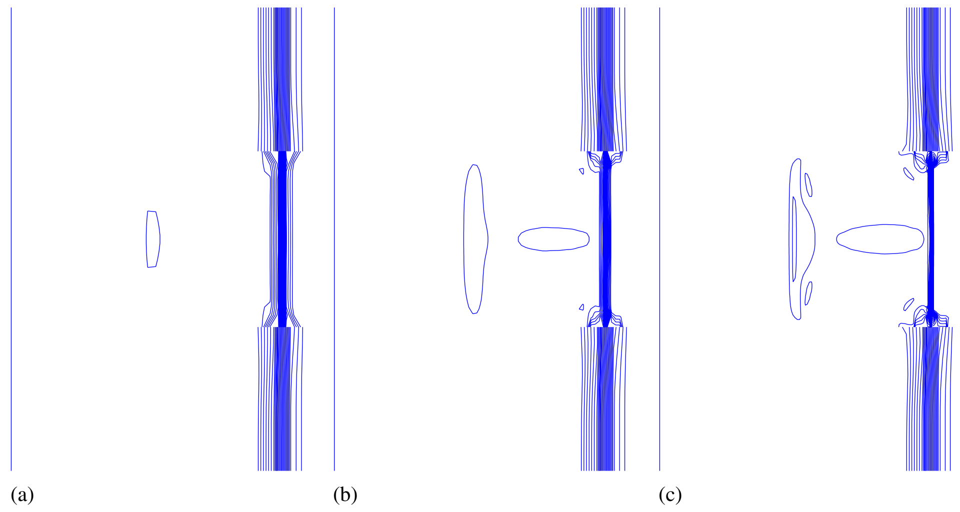

Figure 12. Test 5.1—water depth contours: (a) $Fs=3$ and resolution ratio $=1{:}2$; (b) $Fs=3$ and resolution ratio $=1{:}4$; and (c) $Fs=3$ and resolution ratio $=1{:}8$.

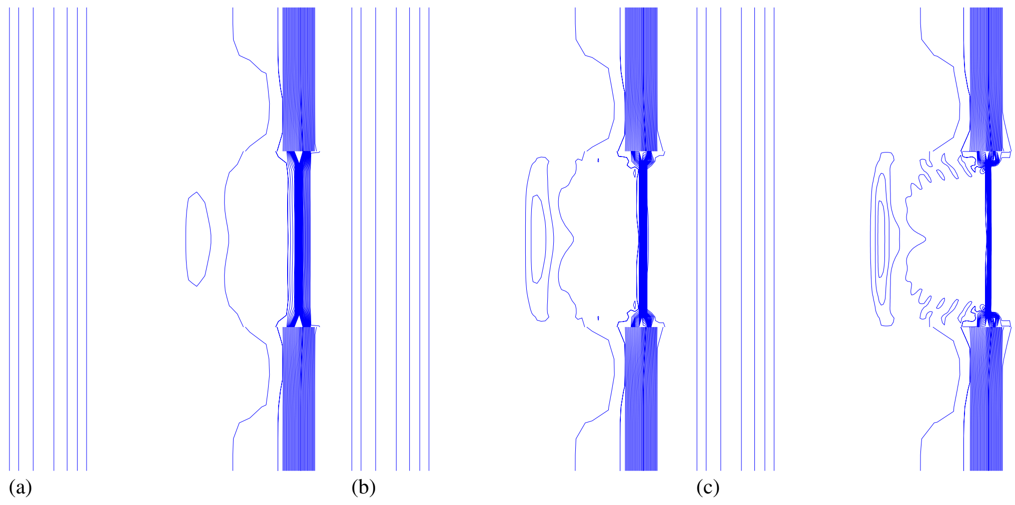

Figure 13. Test 5.1—water depth contours: (a) $Fs=4$ and resolution ratio $=1{:}2$; (b) $Fs=4$ and resolution ratio $=1{:}4$; and (c) $Fs=4$ and resolution ratio $=1{:}8$.

When the Froude number of the bore and resolution ratio exceed 3 and 4, respectively, relatively large errors were shown. However, excellent results were obtained for all resolution ratios, i.e. 1:2, 1:4 and 1:8 when the Froude number was equal and less or equal to 2. Therefore, it may be concluded that the error introduced along the patch interface is insignificant for most practical shallow water flow simulations (e.g. $Fs \leqslant 2$). However, further attention may be needed for flows where the Froude number exceeds 2.

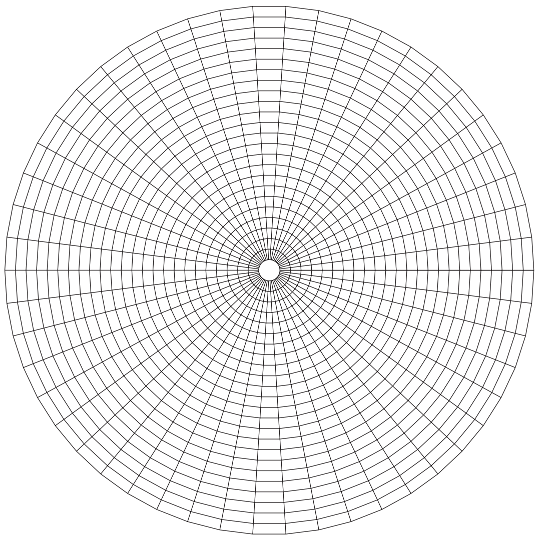

Figure 14. Test 5.2—computation grids for circular dam break.

This test also provided a good opportunity to test the time marching algorithm since the errors associated with a bore travelling along a patch interface could be amplified by an improper time marching procedure.

5.2. *Circular dam break*

A circular dam with a radius of 11 m was collapsed suddenly and completely, with water flushed out in all directions. The initial water depth within the dam was 10 and 1 m outside. The bed of the basin was assumed frictionless and flat, and the fluid viscosity was ignored.

In this case, an O-type branch-out grid (25×50 cells) with 'patch-to-itself configuration' was used (Figure 14). Apart from testing the patch modelling, it also provides a test for the finite volume modelling because of the non-uniformity of grid cells both in size and orientation and the flow singularity at the origin.

Figure 15 shows a 3-D view of the water surface elevation obtained $t = 0.69$ s after breaking. Water surface elevation contours and velocity vector plots are presented in Figures 16 and 17, respectively. For the cells around the centre point, extrapolation instead of interpolation is used within MUSCL reconstruction. The results compare favourably with the work of Alcrudo [21] and Mingham and Causon [22].

5.3. *Oblique bore reflection*

This is equivalent to shock wave reflection at a wedge in gas dynamics, which has been extensively used in testing the capability of numerical schemes for shock waves particularly the interaction of shock waves [23]. Four different types of self-similar oblique bore reflections have been observed in shallow water and discussed by Mingham and Causon [22], namely regular reflection (RR), single Mach reflection (SMR), transitional Mach reflection (TMR) and

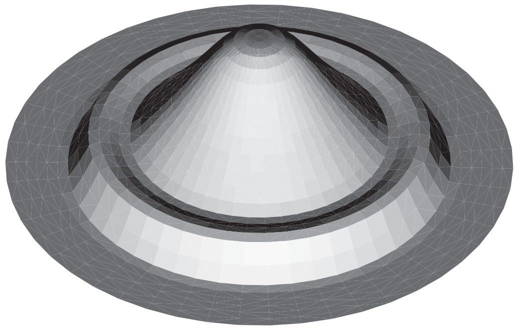

Figure 15. Test 5.2—water surface at $t = 0.69$ s after breaking.

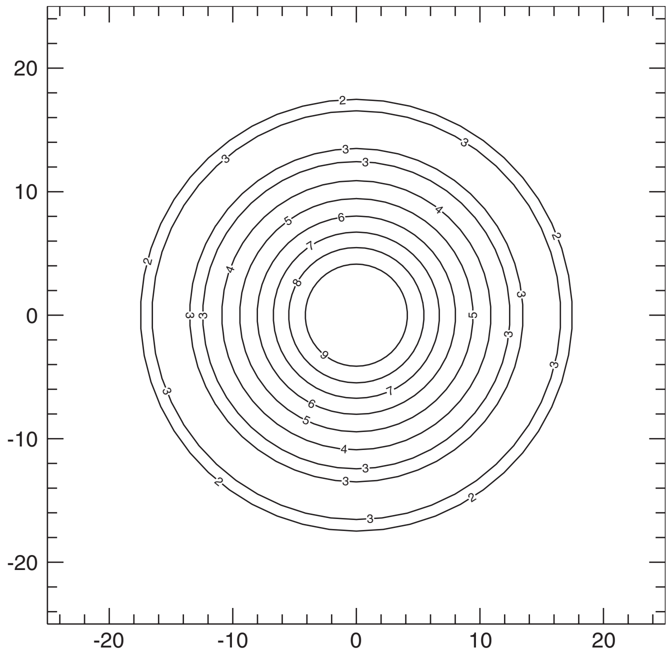

Figure 16. Test 5.2—contour of water depth at $t = 0.69$ s after breaking.

double Mach reflection (DMR). All four types of bore reflection have been tested previously using the non-patching AMAZON code [10]. In this paper, the most complicated DMR case is used to test the new patching method.

A right travelling bore wave travels down a channel and interacts with one bank inclined at an angle of into the direction of flow. The channel is flat-bottomed, and both bed and side wall friction are ignored. Flow turbulence and fluid viscosity are also not considered. Initially, the water depth is 1.0 m and the water is at rest.

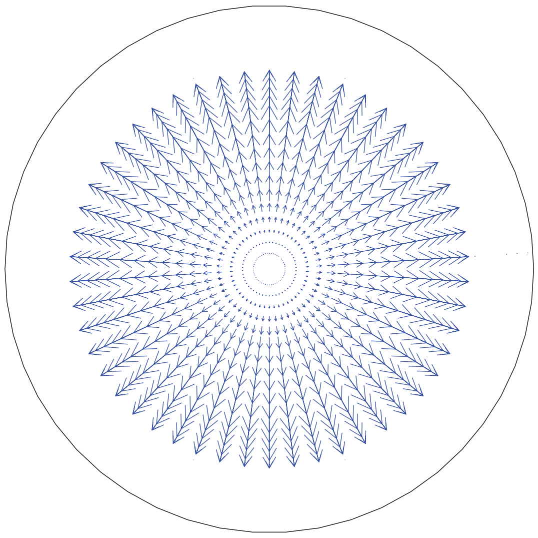
Figure 17. Test 5.2—velocity vectors $t = 0.69$ s after breaking.

The wall inclination angles θ is set to 50°. The Froude number (Fs) of the incident bore was set to $Fs = 3.81$. Time $t = 0$ corresponds to the arrival of the incident bore at the corner point on the bank. The left boundary conditions were obtained using Equations (19a) and (19b). The fully-slip boundary (described in Section 3.4) was used along the lower and the upper banks, and the transmissive boundary condition was applied at the right boundary.

A boundary fitted skewed grid with 600 cells in the flow direction and 300 cells in transverse direction is used in a single-grid model. Instead of a single fine grid with 600×300 cells ($1\,\text{m} \times 3\,\text{m}$) over the whole area, a patched model includes a main model with 120×75 cells ($5\,\text{m} \times 12\,\text{m}$) and an embedded sub-model with 110×80 cells ($1\,\text{m} \times 4\,\text{m}$) as illustrated in Figure 18. The resolution ratios were 1:5 and 1:3 in the longitudinal and transverse directions, respectively.

The results are presented in Figure 20. Compared to the results with the single grid model (Figure 19), the features around the Mach stem are almost identical, though there are some minor spurious contours along the patch interface. It can be seen that the errors introduced along the patch interface do not spread even at a Froude number $Fs \approx 3.81$ and resolution ratios of 1:5 and 1:3. In Section 5.1, the results suggest that particular attention may be needed for Froude numbers exceeding 2 and resolution ratios exceeding 1:2. However, the results of this test may suggest that patched modelling is still valid even for flows with high Froude numbers ($Fs \geqslant 2$) if the patch interface is kept sufficiently far from the area of interest.

The above patching model required 30 min on a PC powered by a 650 MHz Pentium III processor. The non-patching model with a single fine grid took 2 h 8 min to complete on the same PC. Thus, the patching model was 4.3 times more efficient than the non-patching model in this case.

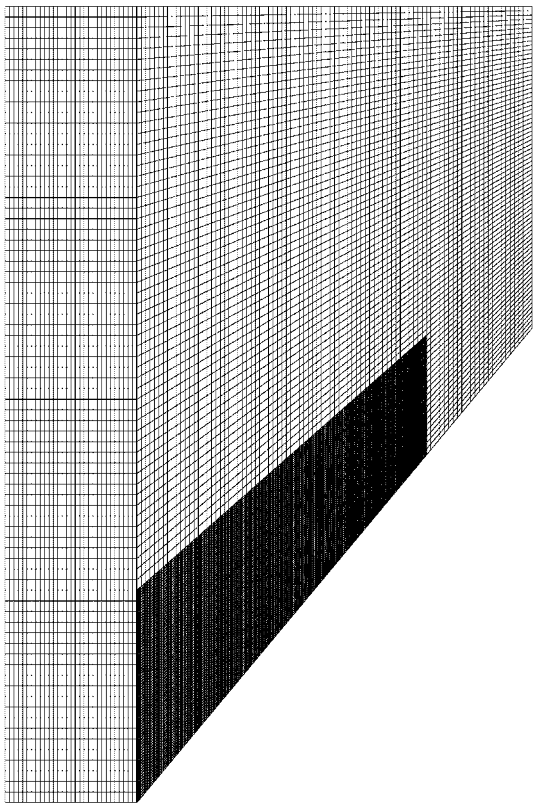

Figure 18. Test 5.3—patch grid for oblique bore reflection.

5.4. Bore wave travelling over a submerged hump

This test case is given by LeVeque [24], which shows bore refraction by a bed feature. The case provides a good opportunity to test the patch algorithm as the error introduced by discontinuity of grid resolution may be intensified by topographic change.

An isolated elliptical shaped hump defined by Equation (20) is in a rectangular basin of 2 m long and 1 m wide, as illustrated in Figure 21.

$$B(x,y) = 0.8\exp(-5(x-0.9)^2 - 50(y-0.5)^2) \tag{20}$$

The water surface is initially flat with water depth $h = 1 - B$ except for $0.05 < x < 0.15$, where h is perturbed upward by 0.01 m. The perturbation induces two bores travelling in an opposite direction. Transmissive boundaries (described in Section 3.4) are applied to the four edges of the basin so bores eventually leave the basin without any reflection back to the domain.

The above described case was simulated on both a single fine grid (600×300) and a patched grid. In the patched grid, a fine grid of 525×90 embedded in a coarse grid of 200×100 in a rectangular of $[(0.05, 0.35) - (1.8, 0.65)]$ (see Figure 21). Like other cases, bed friction, turbulence and viscosity are not considered. The gravitational constant g was set to 1 in this case so that the results were comparable to those produced by LeVeque [24].

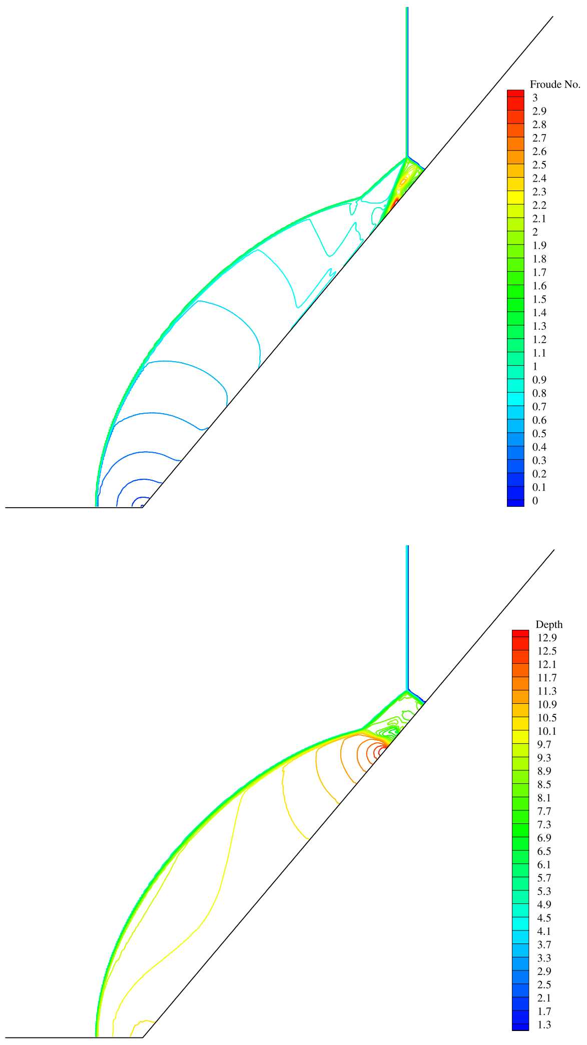

Figure 19. Test 5.3—oblique bore reflection $t = 24$ s (double Mach reflection)—single mesh (above—Froude number; below—water depth).

The results of single-grid and patched-grid models are presented in Figures 22 and 23, respectively. Figure 23 shows that the disturbance complex in the centre of the basin was well captured by the patched-grid model, compared to the single-grid model. It is noted the difference of the interaction of the bores in front of the disturbance complex. The bore interaction is caused by faster moving bores along two edges due to deeper water and subsequent refraction once they pass the slow moving bore in the centre. The poorer resolution at the bore interaction was a result of coarse grids along two edges. In general, the results compare closely with the work of LeVeque [24], the errors generated along the discontinuities between two grid resolution do not spread, and the performance of the patched-grid model is satisfactory.

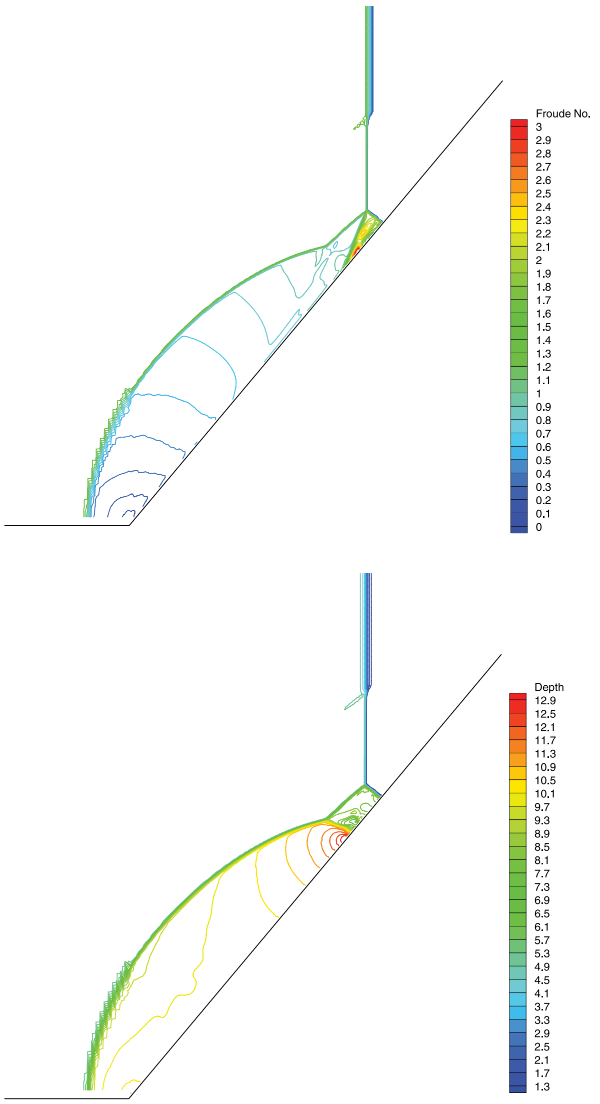

Figure 20. Test 5.3—oblique bore reflection $t = 24$ s (double Mach reflection)—patched mesh (above—Froude number; below—water depth).

6. CONCLUSIONS

A new patching algorithm is presented for shallow water flows together with brief description of the numerical schemes of finite volume model AMAZON. The model has been tested using dam break and moving bores with high Froude numbers. Excellent results were obtained for resolution ratio up to 1:8 between coarse and fine grids for Froude numbers not exceeding 2, which is the situation for most practical shallow water flows. The numerical experiments

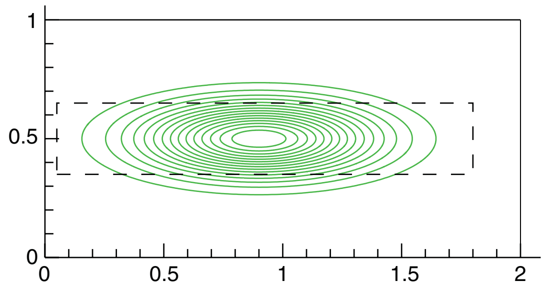

Figure 21. Test 5.4: elliptical hump (dashline indicates an embedded fine grid in the patch model).

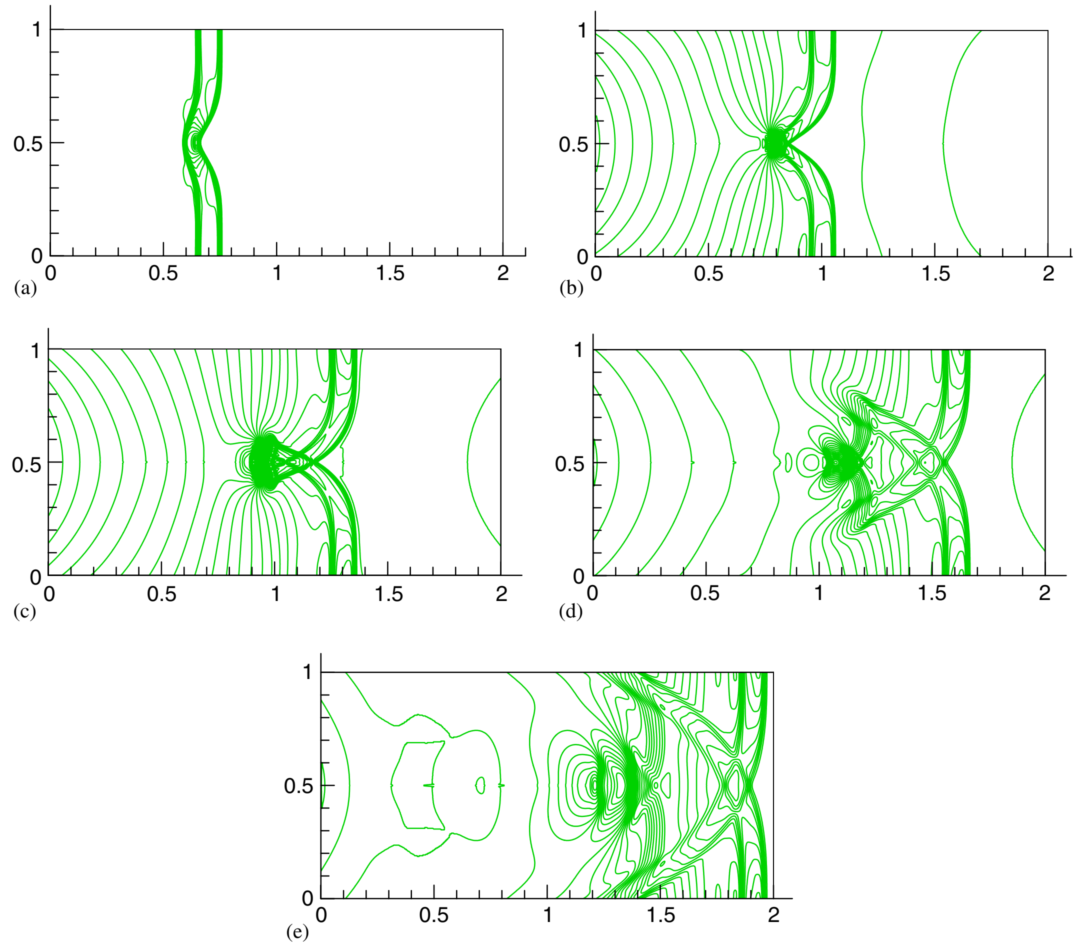

Figure 22. Test 5.4: water depth contours at: (a) $t = 0.6$ s; (b) $t = 0.9$ s; (c) $t = 1.2$ s; (d) $t = 1.5$ s; and (e) $t = 1.8$ s (single grid model).

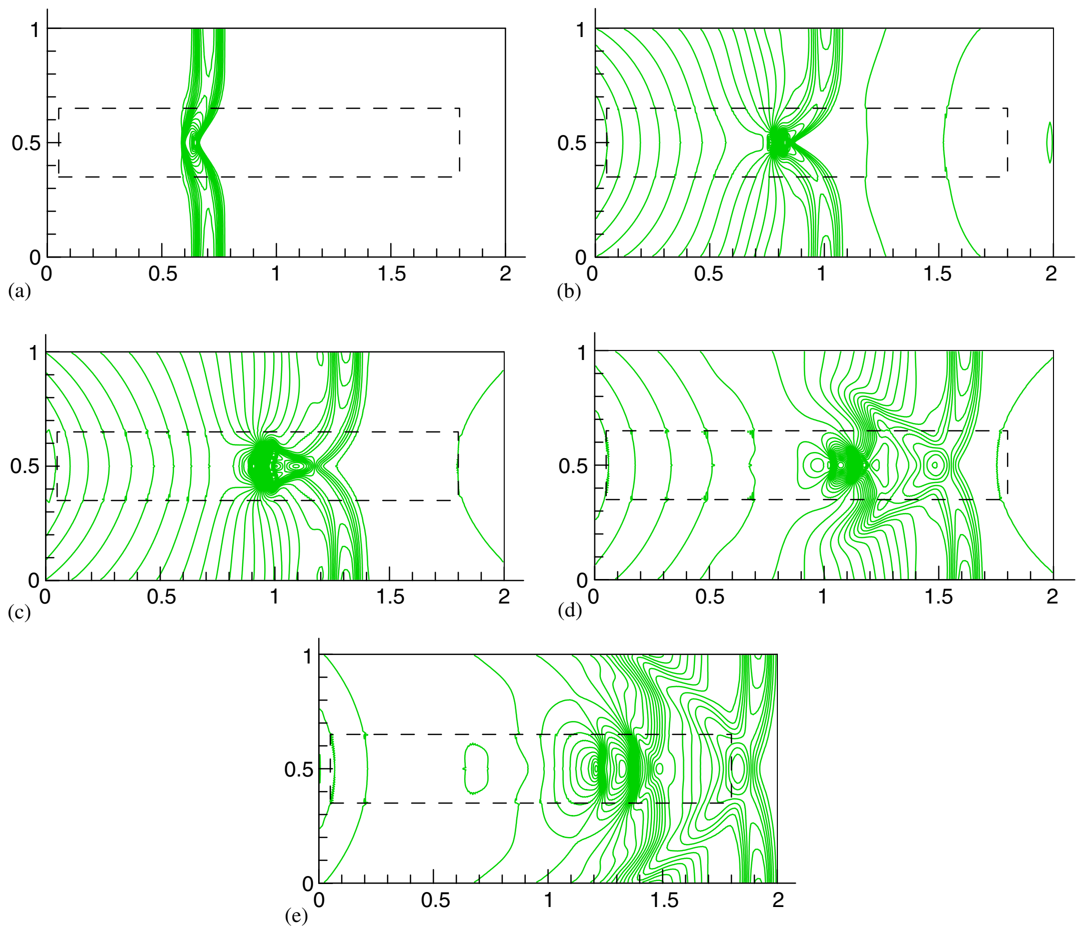

Figure 23. Test 5.4: water depth contours at: (a) $t = 0.6$ s; (b) $t = 0.9$ s; (c) $t = 1.2$ s; (d) $t = 1.5$ s; and (e) $t = 1.8$ s (patched model; dash line indicates the embedded fine-grid).

show that attention is needed for flow where the Froude number exceeds 2 and that the interface between coarse and fine grids should be kept sufficiently far from the area of interest. The algorithm is parallelizable, thus computation speed can be further increased. The authors believe that the new mesh patchable model is particularly useful for practical real-time simulation with complicated geometry and the occurrence of surges and supercritical flow, for example real-time flood inundation modelling in a complicated estuary or urban area.

REFERENCES

1. Furukawa M, Yamasaki M, Inoue M. A zonal approach for solving the compressible Navier–Stokes equations using a TVD finite volume method. *International Journal*, *Series II* (JSME) 1990; **33**(4):665–673.
2. Lien FS, Chen WL, Leschziner MA. A multiblock implementation of a non-orthogonal collocated finite volume algorithm for complex turbulent flows. *International Journal for Numerical Methods in Fluids* 1996; **23**:567–588.

3. Rai MM. A conservative treatment of zonal boundaries for Euler equation calculations. *AIAA Paper No. 84-0164*, 1984.
4. Benek JA, Buning PG, Steger JL. A 3-D chimera grid embedding technique. *AIAA Paper No. 85-1529*, 1985.
5. Bush RH. External compression inlet predications using and implicit, upwind, multiple zone approach. *AIAA Paper No. 85-1521*, 1985.
6. Falconer RA. Flow and water quality modelling in coastal and inland waters. *Journal of Hydraulic Research* 1992; **30**(4):437–452.
7. Pearson RV, Causon DM, Mingham CG. Simulation of shallow water hydrodynamics using a high-resolution finite-volume technique on hierarchically-structured Cartesian grids. *Water Pollution 97*, Computational Mechanics Publications: Southampton, Boston, 1997.
8. van Leer. 1979.
9. van Albada GD, van Leer B, Roberts Jr WW. A comparative study of computational methods in cosmic gas dynamics. *Astronomy and Astrophysics* 1982; **108**:76–84.
10. Hu K. High-resolution finite volume methods for hydraulic flow modelling. *Ph.D. Thesis*, The Manchester Metropolitan University, England, 2000.
11. Strang G. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis* 1968; **5**(3):506–517.
12. Hu K, Mingham CG, Causon DM. A bore-capturing finite volume method for open-channel flows. *International Journal for Numerical Methods in Fluids* 1998; **28**:1241–1261.
13. Godunov SK. A difference method for the numerical calculation of discontinuous solutions of hydrodynamic equations. *Matemsticheskly Sboraik* 1996; **47**(3):271–306 (Translated U.S. Joint Publications Research Service, JPRS 7226).
14. Harten A, Lax PD, van Leer B. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review* 1983; **25**(1):35–61.
15. Toro EF. Riemann problems and the WAF method for solving the two-dimensional shallow water equations. *Philosophical Transactions of the Royal Society of London* 1992; **338**(A):43–68.
16. Toro EF, Spruce M, Spears W. Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves* 1994; **4**:25–34.
17. Deiwert GS. Numerical simulation of high Reynolds number transonic flows. *AIAA Journal* 1975; **13**(10):1354–1359.
18. Toro EF. *Shock-Capturing Methods for Free-Surface Shallow Flows*. Wiley: Chichester, 2001.
19. Thompson JF, Warsi ZUA, Mastin CW. *Numerical Grid Generation—Foundation and Applications*. Elsevier Science: New York, 1985.
20. Henderson FM. *Open Channel Flow*. Macmillan Publishing: New York, NY, 1966.
21. Alcrudo F, Garcia-Navarro P. A high-resolution Godunov-type scheme in finite volumes for the 2D shallow-water equations. *International Journal for Numerical Methods in Fluids* 1993; **16**:489–505.
22. Mingham CG, Causon DM. A high resolution finite volume method for shallow water flows. *Journal of Hydraulic Engineering* 1998; **124**(6):606–614.
23. Glass II, Sislian JP. *Nonstationary Flows and Shock Waves*, Oxford Engineering Science, No. 39. Clarendon Press: Oxford, 1994.
24. LeVeque BJ. Balancing source terms and flux gradients in high-resolution Godunov methods: the quasi-steady wave-propagation algorithm. *Journal of Computational Physics* 1998; **146**:346–365.