A High-Performance Integrated Hydrodynamic Modelling System for Urban Flood Simulations

Qiuhua Liang,
State Key Laboratorty of Hydrology, Water
Resources and Hydraulic Engineering,
Hohai University
qiuhua.liang@ncl.ac.uk *

Luke S. Smith,
School of Civil Engineering
and Geosciences,
Newcastle University
l.s.smith@ncl.ac.uk

Abstract

A new High-Performance Integrated hydrodynamic Modelling System (Hi-PIMS) is tested for urban flood simulation. The software solves the two-dimensional shallow water equations (SWEs) using a first-order accurate Godunov-type shock-capturing scheme incorporated with the Harten, Lax and van Leer approximate Riemann solver with the contact wave restored (HLLC) for flux evaluation. The benefits of modern graphics processing units (GPUs) are explored to accelerate large-scale high-resolution simulations. In order to test its performance, the tool is applied to predict flood inundation due to rainfall and a point source surface flow in Glasgow, Scotland, and a hypothetical inundation event at different spatial resolutions in Thamesmead, England caused by embankment failure. Numerical experiments demonstrate potential benefits for high-resolution modelling of urban flood inundation, and a much-improved level of performance without compromising result quality.

*Telephone: +44 (0) 191 208 6413

1   Introduction

The past year of 2012 has seen the UK and numerous regions around the world subjected to an unusually wet summer, causing severe flash flooding. In July, August and September, many places in the UK, including Wales, Cornwall, Devon, North Somerset, North and West York-shire, Newcastle and the Scottish Borders, suffered flash flooding that caused substantial damage to property and major disruption to transport networks. Outside of the UK, a devastating flood killed 144 people in the Krasnodar region of Russia in July; Beijing in China, Uttarakhard in India and Manilla in the Phillipines have also been struck since. Most commonly associated with torrential rainfall, these flash floods are characterised by a sudden rise in river and subsequent floodplain inundation, and high-velocity overland flow following a rapid catchment response to the intense rainfall. Numerous processes related to the catchment response including bore for-mation (so-called 'walls of water') remain poorly understood but numerical modelling provides a means through which these events may be reproduced. Reliable simulation of these violent and unpredictable natural events demands a shock-capturing hydrodynamic model, generally beyond the capabilities of hydrological and simplified hydraulic modelling. However, the high compu-tational burden associated with full hydrodynamic models has restricted their wider application to small spatial extents and short duration events. Most of the existing shock-capturing hydro-dynamic models are not able to provide efficient and high-resolution simulations for large-scale flash flood events. Particularly, most of the aforementioned flash floods occur in urban areas, where high-resolution simulation is essential in order to resolve the complex urban topographic features consisting of buildings, streets and embankments in order to provide reliable numeri-cal predictions. This poses a great challenge to existing two-dimensional hydrodynamic flood modelling tools, which are generally computationally demanding. This work therefore presents a new High-Performance Integrated hydrodynamic Modelling System (Hi-PIMS) for efficient high-resolution urban flood modelling.

Considerable research effort has been directed towards acceleration of flood modelling in order to achieve higher spatial resolutions and greater extents. Much of this literature focusses on simplification of the underlying equations to create kinematic- or diffusive-wave approxi-mations. However, most of these simplified models are not appropriate to depict the complex catchment responses hypothesised whilst providing accurate depths and velocities, and their reduced physical complexity may cause increased sensitivity to and dependence on parame-terisation (Costabile et al., 2009, 2012; Fewtrell et al., 2011; Yeh et al., 2011). Furthermore, there is evidence to suggest these simplified approaches will not achieve significant and consis-tent reductions in computation time, albeit case and resolution dependent (Hunter et al., 2008; Néelz and Pender, 2010; Wang et al., 2011). In order to improve computational efficiency of the diffusion-wave models for high-resolution simulations, Bates et al. (2010) presented a new for-mula for estimating inter-cell discharges for this type of models by partially restoring the inertial terms from the fully dynamic momentum equation. The numerical stability of the resulting par-tial inertial model is now controlled by the much less restricted Courant–Freidrichs–Lewy (CFL) condition, identical to the explicit hydrodynamic models. Due to the use of simplified governing equations and a simpler numerical scheme, the partial inertial model is shown to save 20% - 30% of the computational cost by comparison with a shock-capturing hydrodynamic model that

solves the full two-dimensional shallow water equations in the similar code base (Zhang et al., 2014). However, for a city-scale high-resolution flood simulation that covers millions of computational nodes, the partial inertial models are still computationally too demanding. Furthermore, Neal et al. (2012), after comparing the performance of their diffusion-wave, partial inertial and shock-capturing dynamic wave models against a set of benchmark test cases, pointed out that while the simplified models may reproduce numerical results comparable to a full model they are unable to simulate supercritical flows accurately. However, urban flash flood events as a result of intense rainfall or failure of flood defences are generally characterised as rapidly varying trans-critical and supercritical flows. Therefore, the shock-capturing fully dynamic models, representing the most recent developments in simulating complex shallow flow hydrodynamics, appear to be a natural option for urban flash flood modelling as considered in this work, although they are still inadequate in representing certain small-scale three-dimensional flow features (Soares-Frazão et al., 2008; Guinot, 2012).

By creating a refined mesh only in those areas of interests, dynamic grid adaption provides an effective means to relax the high computational burden inherent in the full dynamic inundation models. While retaining the robustness and complexity of the numerical approach, dynamic grid adaptation however presents new problems if mass conservation is to be achieved. It is unlikely to increase the timestep for cases where the highest level of refinement is concentrated on the most complex flow dynamics and highest velocities or free-surface gradients (see descriptions of refinement criteria in Liang et al., 2008; Kubatko et al., 2009).

Rather than creating high-resolution mesh to directly capture small-scale topographic or flow features as used in the adaptive mesh methods, different sub-grid parameterization techniques have also been proposed to integrate high-resolution topographic features into flood models to enable more accurate and efficient coarse-resolution simulations (e.g. Soares-Frazão et al., 2008; Guinot, 2012; Schubert and Sanders, 2012; Chen et al., 2012). Most of these models are essentially based on rigorous reformulation of full dynamic shallow water equations to effectively reproduce complex urban topography (e.g. Soares-Frazão et al., 2008; Guinot, 2012; Schubert and Sanders, 2012). Soares-Frazão et al. (2008) introduced a new shallow flow model with porosity to account for the reduction in storage due to sub-grid topographic features. The performance of the porosity model was compared with that of a refined mesh model explicitly reflecting sub-grid scale urban structures and a more classical approach of raising local bed roughness. While able to reproduce the mean characteristics of the urban flood waves at a much lower computational cost than the refined mesh simulations, the porosity model was unable to accurately predict the formulation and propagation of certain localised wave features, e.g. reflected bores. In another study, Schubert and Sanders (2012) investigated various approaches of representing sub-grid topographic features in simulating a dam-break flood in an urban area and concluded that only those methods taking into account of building geometries can capture building-scale variability in the velocity field. They also indicated the benefit of using high-resolution simulation to explicitly represent buildings and road structures if run-time execution costs were not a major concern (see also Gallegos et al., 2009). The issue of high-computational cost may be resolved by exploring recent developments in computational hardware.

If software developers are to keep pace with the increasing power of hardware, they must accept and respond to the stagnation of Central Processing Unit (CPU) clock speeds and look to

parallel processing to fully harness computing power. Despite this the majority of commercial hydraulic modelling software can only utilise a single CPU core. Parallel programming approaches are shown to exhibit good weak and strong scaling when software is structured appropriately (e.g. Neal et al., 2010; Saetra and Brodtkorb, 2012). But there is a far more interesting potential for looking beyond the CPU and examining the role heterogeneous computing might play. Graphics Processing Units (GPUs) are designed to process large volumes of data by performing the same calculation numerous times, typically on vectors and matrices. Such hardware architectures are well-suited to the field of computational fluid dynamics. New programming languages including CUDA and OpenCL have exposed this hardware for use in general-purpose applications (GPGPU). A number of attempts have been made to explore the benefits of GPU computing for highly efficient large-scale flood simulations. Early pioneers of such methods include Crossley et al. (2009) who harnessed graphics APIs directly to implement a diffusion wave model (JFlow) for GPUs, Kalyanapu et al. (2011) with a finite-difference implementation of the full shallow water equations, and later Brodtkorb et al. (2012) with a finite-volume scheme. Such software is becoming increasingly mainstream; Néelz and Pender (2013) report results from several commercial GPU hydraulics implementations while Smith and Liang (2013) demonstrate the potential for generalised approaches applicable to both CPU and GPU co-processors. The most recent research also explores how domain decomposition across multiple GPUs can provide further performance benefits (Saetra and Brodtkorb, 2012).

This work aims to present a high-performance hydrodynamic model for simulating different types of urban inundation processes including those induced flash floods, which generally involves rapidly-varying transcritical flows that should be reliably resolved by a robust shock-capturing shallow water model. The new computational power provided by recent developments in GPU computing is explored by adopting the model framework as introduced in Smith and Liang (2013), which solves the 2D shallow water equations using a 2nd-order accurate shock-capturing finite-volume Godunov-type scheme. The model can take advantage of either CPUs or GPUs with a single codebase by leveraging OpenCL. To further improve the computational efficiency of the modelling framework, a 1st-order accurate scheme is implemented, due to the fact that a 2nd-order accurate numerical scheme may not always be superior in real-world applications (Zhang et al., 2013). Extra source and sink terms are also included in the new implementation to better describe the urban rainfall-runoff processes. With Hi-PIMS, the focus of this work is to demonstrate that high-resolution large-scale urban inundation modelling can be realised at an affordable computational cost. Application to a standard benchmark test in Glasgow and a hypothetical case at Thamesmead reaffirms its efficiency and suitability. Results are presented for different vendors' devices.

2 Finite-volume Godunov-type shallow flow solver

In the general case for a flood event, the water depth is much smaller than the horizontal dimensions of the water body. Therefore the hydrodynamics of the flood wave can be suitably described by the shallow water equations (SWEs) that are derived to take full account of mass and momentum conservation in a two-dimensional manner. In matrix form, the SWEs may be

written as

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = \mathbf{s}, \tag{1}$$

where t is the time, x and y the Cartesian coordinates, q the vector representing the flow variables, f and g the fluxes in the two Cartesian directions, and s is the source term vector. The vector terms are given by Liang and Borthwick (2009),

$$
\mathbf{q} = \begin{bmatrix} \eta \\ q_x \\ q_y \end{bmatrix}, \qquad
\mathbf{f} = \begin{bmatrix} q_x \\ uq_x + g(\eta^2 - 2\eta z_b)/2 \\ uq_y \end{bmatrix},
$$

$$
\mathbf{g} = \begin{bmatrix} q_y \\ vq_x \\ vq_y + g(\eta^2 - 2\eta z_b)/2 \end{bmatrix}, \qquad
\mathbf{s} = \begin{bmatrix} q_s \\ -\frac{\tau_{bx}}{\rho} - g\eta\frac{\partial z_b}{\partial x} \\ -\frac{\tau_{by}}{\rho} - g\eta\frac{\partial z_b}{\partial y} \end{bmatrix}. \tag{2}
$$

Herein, η and z_b denote water level and bed elevation above datum and therefore $h = \eta - z_b$ calculates the water depth; u and v are the two depth-averaged velocity components; $q_x(= uh)$ and $q_y(= vh)$ are the $x-$ and $y-$directional unit-width discharges; g is the gravitational acceleration; q_s includes source and sink terms as a result of rainfall and loss through drainage systems, etc.; ρ is the water density; $-\delta z_b/\delta x$ and $-\delta z_b/\delta y$ define the two bed slopes; τ_{bx} and τ_{by} are bed friction stresses calculated by

$$\tau_{bx} = \rho C_f u \sqrt{u^2 + v^2} \qquad \text{and} \qquad \tau_{by} = \rho C_f v \sqrt{u^2 + v^2}, \tag{3}$$

where $C_f = gn^2/h^{1/3}$ is the bed roughness coefficient with n known to be the Manning coefficient.

In order to better capture complex flow hydrodynamics including hydraulic jump-like flow discontinuities, the above SWEs are numerically solved using a finite-volume Godunov-type scheme, which updates flow variables to the next time step using the following time-marching formula

$$\mathbf{q}_{i,j}^{k+1} = \mathbf{q}_{i,j}^k - \Delta t \left(\frac{\mathbf{f}_{i+1/2,j} - \mathbf{f}_{i-1/2,j}}{\Delta x} + \frac{\mathbf{g}_{i,j+1/2} - \mathbf{g}_{i,j-1/2}}{\Delta y} - \mathbf{s}_{i,j} \right), \tag{4}$$

where k represents the time level; i and j indicate the cell indices; Δt, Δx and Δy are the time step, cell size in the $x-$ and $y-$directions, respectively. In order to update the flow variables to a new time step, the four flux vectors ($f_{i+1/2,j}, f_{i-1/2,j}, g_{i,j+1/2}, g_{i,j-1/2}$) and the source term vector ($s_{i,j}$) must be properly calculated.

In the context of a Godunov-type scheme, the interface fluxes are evaluated by solving local Riemann problems, e.g. $\mathbf{f}_E = \mathbf{F}(\mathbf{q}_E^L, \mathbf{q}_E^R)$, that are defined by the left and right Riemann states $\mathbf{q}_E^L$ and $\mathbf{q}_E^R$. In order to obtain the Riemann states, the values of flow variables must first be

reconstructed on both sides of the cell interface under consideration. A first-order accurate scheme assumes piecewise distribution of the flow information and therefore the face values are essentially the same as those at the cell centres. Taking the eastern cell interface of cell (i, j) as an example, the left face values of the flow variables and bed elevation are simply

$$\mathbf{q}_E^L = \mathbf{q}_{ic} \quad \text{and} \quad \tilde{z}_{bE}^L = z_{bic}. \tag{5}$$

Subsequently, the associated face values of water depth and velocity components are given by

$$\tilde{h}_E^L = \tilde{\eta}_E^L - \tilde{z}_{bE}^L, \qquad \tilde{u}_E^L = (\tilde{q}x)_E^L / \tilde{h}_E^L, \quad \text{and} \quad \tilde{v}_E^L = (\tilde{q}y)_E^L / \tilde{h}_E^L. \tag{6}$$

The face values on the right-hand-side of the cell interface can be obtained in a similar way, which is actually equal to those at the centre of cell $(i + 1, j)$. Based on these face values, the Riemann states are derived after defining a single value of bed elevation across the cell interface, i.e.

$$z_{bE} = max\left(\tilde{z}_{bE}^L, \tilde{z}_{bE}^R\right). \tag{7}$$

The corresponding Riemann states are therefore reconstructed as

$$h_E^L = max\left(0, \tilde{\eta}_E^L - \tilde{z}_{bE}^L\right), \quad \eta_E^L = h_E^L + z_{bE}, \quad (q_x)_E^L = (\tilde{u}x)_E^L \tilde{h}_E^L, \text{ and } (q_y)_E^L = (\tilde{v}y)_E^L \tilde{h}_E^L. \tag{8}$$

Similarly, the Riemann states can be obtained at the right hand side of the cell interface.

In the current formulation of SWEs as given in 1 and 2, the water level instead of water depth is used as a flow variable. In a dry cell, the reconstructed water level is actually the ground level. If the ground level (i.e. the reconstructed 'water level') is higher than the actual water level, spurious fluxes will be calculated which in turn breaks the so-called well-balanced property of the governing equations. Therefore, the difference between the actual and fake water levels must be identified and subtracted from the reconstructed bed elevation and water level (Liang, 2010). Again taking the eastern interface of cell (i, j) as an example, the level difference can be easily calculated by

$$\Delta z = max\left(0, z_{bE} - \tilde{\eta}_E^L\right), \tag{9}$$

which is then used to modified the reconstructed bed elevation and water level as follows

$$z_{bE} \leftarrow z_{bE} - \Delta z, \qquad \eta_E^L \leftarrow \eta_E^L - \Delta z, \qquad \text{and} \qquad \eta_E^R \leftarrow \eta_E^R - \Delta z. \tag{10}$$

These reconstructed Riemann states are then employed by the Harten, Lax and van Leer approximate Riemann solver (HLLC) with the contact wave restored (Toro et al., 1994) to compute the interface fluxes.

The bed slope source terms are directly approximated by a central differencing approach. For example, in the x-direction,

$$-g\eta\frac{\delta z_b}{\delta x} = -g\tilde{\eta}\left(\frac{z_{bE} - z_{bW}}{\Delta x}\right), \tag{11}$$
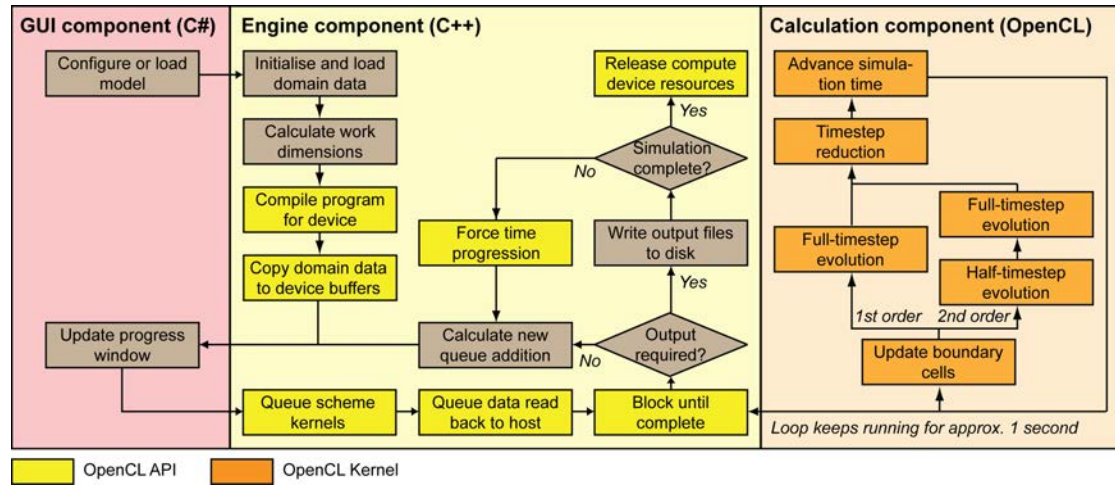
Figure 1 Flowchart of the software processes indicating which components are OpenCL kernels or API calls.

where $\tilde{\eta} = (\eta_E^L + \eta_E^R)^2$. For the friction source terms, a splitting-limited-implicit scheme adopted by Liang (2010) is implemented to ensure better numerical stability. This gives a robust finite-volume Godunov-type solver for simulating shallow flows over complex domain topography with wetting and drying. As demonstrated by Liang (2010), the numerical scheme preserves the well-balanced solution of a lake at rest and ensures non-negative water depth.

The present first-order accurate finite-volume Godunov-type scheme is overall explicit and its numerical stability is restricted by the Courant–Friedrichs–Lewy (CFL) criterion. Simple transmissive and reflective (slip) boundary conditions are used in the test cases considered in this work (Liang, 2010).

3   GPU-accelerated framework

The framework used herein adopts a perhaps unusual approach and more details can be found in Smith and Liang (2013). Code responsible for applying the finite-volume scheme is generated at least in part dynamically, then compiled by the underlying operating system drivers using an appropriate instruction set and optimisations for the hardware available. This is accomplished through the Application Programming Interface (API) functionality made available through the OpenCL standard. Relevant constants such as the cell dimensions of the domain are hence embedded in the assembly instructions themselves, and functionality which is not required or disabled (e.g. friction, atmospheric boundary conditions) can be removed altogether. Dynamic type definitions also allow the same codebase to be used with single- (32-bit) or double-precision (64-bit) floating-point computation.

Moving data from the main system memory (RAM) to a GPU device is an expensive operation (see recommendations in Advanced Micro Devices Inc, 2011; NVIDIA Corporation, 2009). Cell data is only moved periodically for output files, and the numerical scheme is otherwise permitted to run in a loop for approximately one second before a small amount of data is transferred
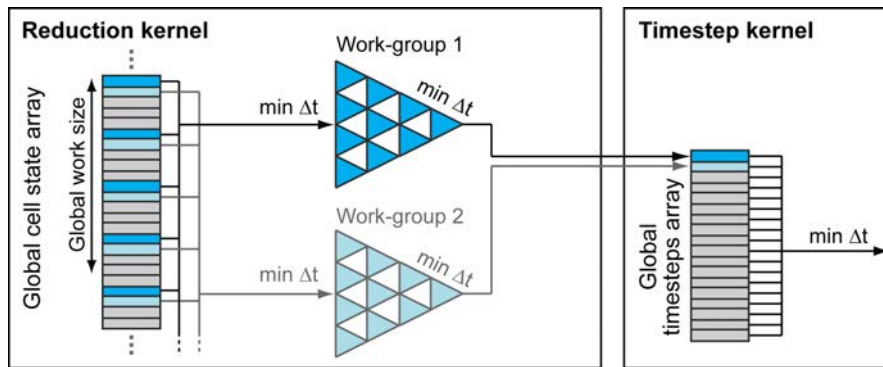
Figure 2 Representation of a two-stage reduction process with a global stride of six (stride in reality would be much larger) performing a series of binary comparisons to create a smaller array of potential values.

to update the user on the progress of the simulation. This process is represented in 1. No vendor-specific optimisations were made, as the authors' intentions were to create a system appropriate for use with any of the mainstream vendors' hardware, most notably NVIDIA, AMD and Intel. The compiler is instructed to adhere to all of the appropriate IEEE standards for floating-point arithmetic (i.e. accurate square root operations, treatment of denormals, etc.).

A regular Cartesian grid is used to represent the domain, whereby transient state variables are stored in a four-element vector and constant values (bed elevation and friction coefficients) require two further elements. This means 48 bytes or less are required per cell. Considering further constraints imposed on the size of a single memory allocation, the software is presently limited to 50 million cells for current hardware offering up to 6GB of memory.

A two-stage reduction process (2) is used to identify the largest permissible timestep within the domain: cells are sampled with a regular stride and recursive binary comparisons used to provide a single value which is carried forward to a much smaller array. This is examined in the second stage when incrementing the overall simulation time. The reduction process ensures processors perform sufficient work to mask the considerable latency introduced by transferring data from a GPU's globally-accessible memory to compute unit-specific registers. A variety of raster formats are supported for importing and exporting domain data, initial conditions, and periodic output.

4   Results and discussion

The aforementioned software has been applied to inundation simulations in Glasgow and Thames-mead, both in the United Kingdom. The first represents a standard test-case to allow comparison with commercially-available hydraulic modelling packages. The latter represents a much larger-scale test that would be burdensomely slow without GPU acceleration. The Courant number is 0.5 for both cases. Both cases are simulated using an Intel Xeon E5-2609 CPU device (Intel Corporation, 2012), AMD FirePro V7800 GPU (Advanced Micro Devices Inc, 2010), and NVIDIA Tesla M2075 GPU (NVIDIA Corporation, 2011).
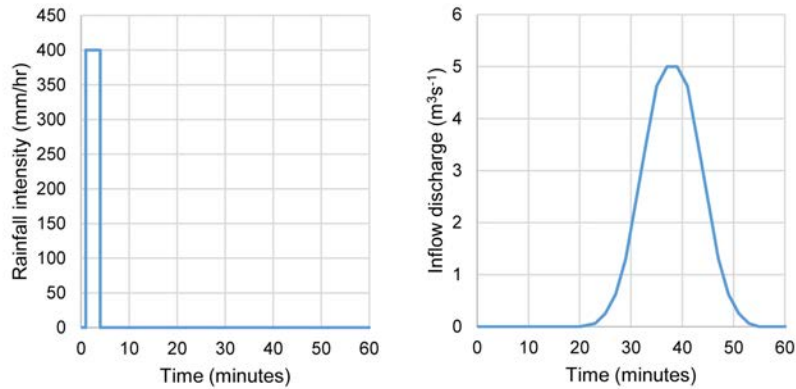
Figure 3 (a) Uniformly distributed rainfall hyetograph; (b) volumetric discharge at the inflow point.
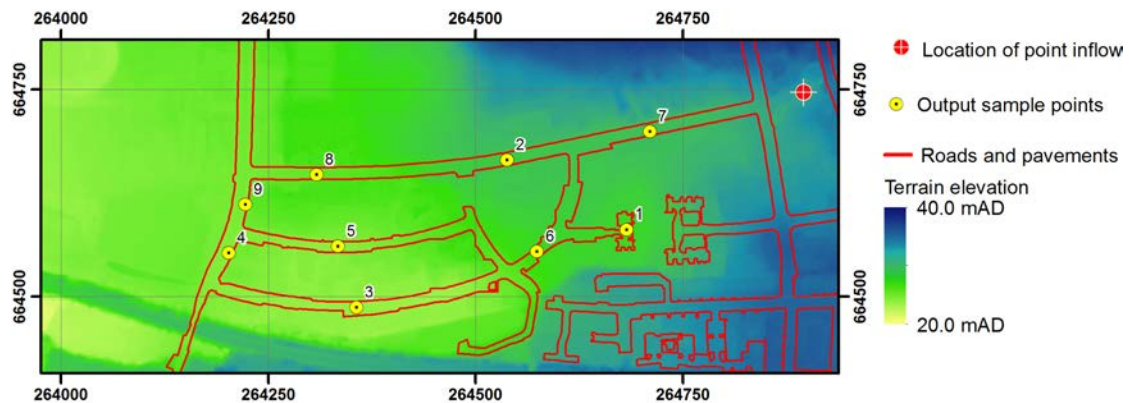

Figure 4 The surface elevation for a 0.39 km^2 area of Glasgow with the inflow location indicated and the position of 9 output sample points.

4.1 Glasgow

The UK Environment Agency commissions a report periodically examining the differences in results, suitability and performance of different 2D hydraulic modelling packages. One of the more complex test-cases therein is a short hypothetical flood event occurring as a combination of both a point inflow and uniform precipitation in the area surrounding Cockenzie Street, Glasgow. The test was performed in accordance with Néelz and Pender (2010) and Néelz and Pender (2013), simulating a 5-hour period.

The inflow hydrograph and hyetograph are given in 3. The digital terrain model (DTM) is shown in 4 alongside the location of the point inflow at $(264896, 664747)$. Data was supplied at 0.5m resolution but has been resampled to 2m to allow comparison with published results for other software. The computational domain contains 97,083 cells. A uniform Manning coefficient of 0.05 is used everywhere except for roads and pavements where a value of 0.02 is assigned. Closed boundary conditions are applied around the extremity of the domain.
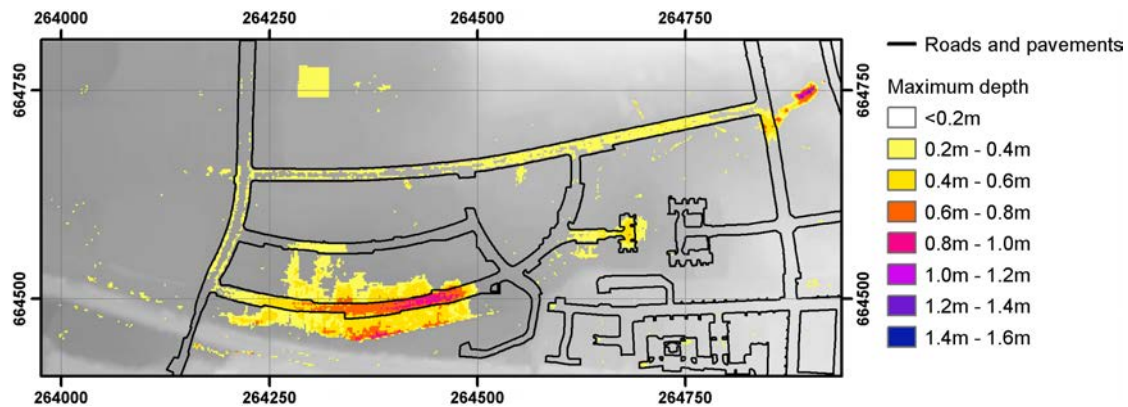
Figure 5 Maximum water levels observed per cell after 5 hours, shown at 0.2m intervals.

Simulations were carried out using 32- and 64-bit floating-point arithmetic (i.e. single- and double-precision). 32-bit arithmetic introduced significant errors in mass conservation for the given numerical scheme and resulted in timesteps of approximately 0.1s. This is believed to be caused by the lack of numerical resolution for the small depths by which unit-width discharge is divided to give velocities. The typical timestep with 64-bit arithmetic is approximately 0.3s. This difference largely negates the performance benefits that are normally achievable with reduced precision arithmetic for both GPUs and CPUs, and furthermore has implications for any other simulations in which extremely shallow flows might be expected. Results presented hereafter are for 64-bit simulations except where indicated.

The maximum depths recorded per cell at the end of the simulation are displayed in 5 at 0.2m intervals. In addition water levels and velocities were output at 9 different points; levels are shown in 6 while velocities are omitted for brevity. The maximum depths and timeseries data are consistent with results produced by other software, and close to those of other finite-volume software packages in particular. Small differences are discernible but all fall within the ranges of results presented in Néelz and Pender (2010).

The simulation correctly represents the double-peaked nature of the event from intense rainfall and subsequent surcharging of a sewer, with the second peak at point 7 shortly after the surcharging peaked at 38 minutes. The results are generally free of oscillations, except for some small oscillations at point 9 which may be unphysical. At point 2 there is some discrepancy among models as to when the water levels settle after both peaks have passed; the results presented herein suggest this occurs after approximately 120 minutes, which is consistent with the results for the comparable numerical scheme employed in TUFLOW FV. At point 3 the second peak begins at approximately 50 minutes, which is consistent with almost all of the software in Néelz and Pender (2013). At point 6 the second peak is predicted to occur at around 27.05 mAD, which is higher than many of the other software but there is significant variation across software at this point, ranging from 26.95 to 27.08 mAD. The final flood depths correspond to the areas which were predicted by the majority of software tested in Néelz and Pender (2013); clearly a shock-capturing scheme is not necessary to accurately predict the final extent, but can have a marked effect on the progression of the flood wave, localised flow dynamics and arrival tim-
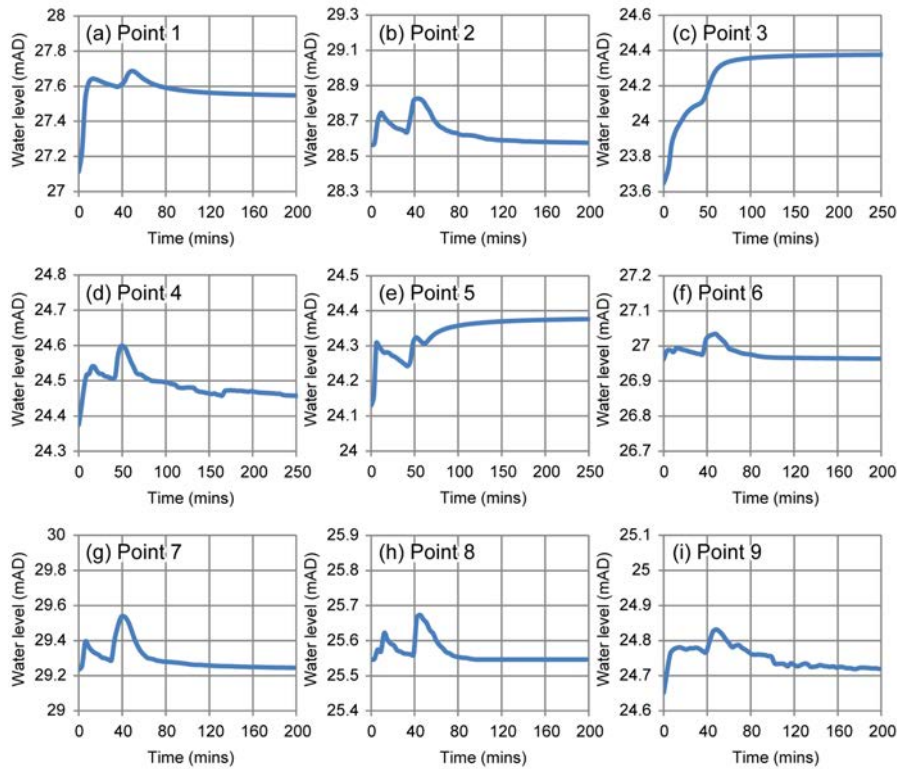
Figure 6 Changes in recorded water levels above datum recorded at 9 different sample points in the domain. The location of the sample points is given in 4.

ing, all of which could be significant factors in assessing flood risk and crucial in issuing flood warnings. It cannot be asserted as to which software is most accurate as the event and inflow data is hypothetical, however the software herein captures the same behaviour as comparable finite-volume codes solving the full shallow water equations. Small differences in the methods for solving the equations and implementation of boundary conditions can result in significant differences: treatment of wet-dry fronts, frequency of mass addition for precipitation, rounding or smoothing in the consideration of topography, and mechanism for considering the Manning coefficient are believed to be most significant in this instance.

The run-times for the simulation using three different processing devices are presented in

Table 1 Simulation run-times for Glasgow in minutes using three different processing devices.

Floating-point arithmetic resolution	CPU Intel Xeon E5-2609	GPU AMD FirePro V7800	GPU NVIDIA Tesla M2075
32-bit	9.05	2.47	1.98
64-bit	9.45	2.67	2.88

11

1. It is important to note however that the domain for this test-case is too small to fully exploit the weak scaling in GPUs. Nonetheless the times represent a significant reduction on those in Néelz and Pender (2010) and Hunter et al. (2008), with both CPU and GPU computation, despite the use of an explicit numerical scheme and Godunov-type scheme. Compared to figures reported for the same test in Néelz and Pender (2013), the software presented herein is slightly slower than some comparable GPU software (e.g. 1.40 minutes for TUFLOW GPU), which may in part be a result of using OpenCL parallelisation, where there is some evidence to suggest memory transfers and dispatch overheads could be slightly higher than CUDA (e.g. Karimi et al., 2010). The performance results cannot be directly compared however as different hardware was used for each software. Moreover, the CUDA and OpenCL APIs have subtle but important differences between implementations, such as the non-standard blocking behaviour of `clEnqueueNDRangeKernel` in NVIDIA's implementation of OpenCL. Comparison is also difficult as diffusion approximation codes can be expected to exhibit poor computational efficiency at 2m resolution because of a more severe timestep constraint.
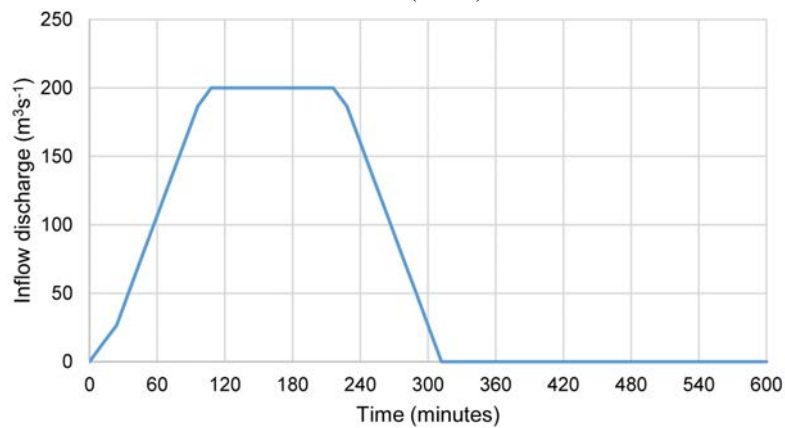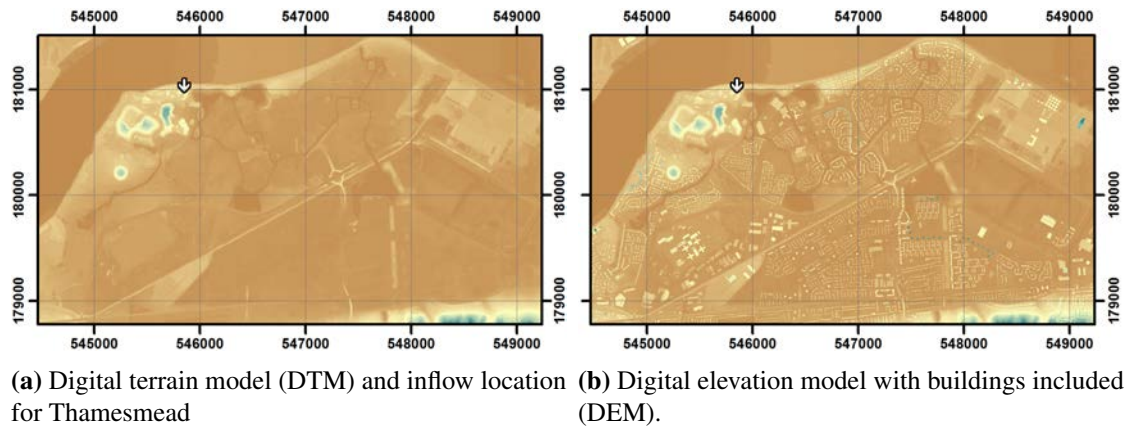
4.2 Thamesmead

The Thamesmead district of South London is located downstream of London's main flood defence, the Thames Barrier. The area is low lying but was heavily developed in the 1960s. Flood risk is posed by storm surges or a failure in the defence wall, with the area previously inundated in the North Sea Surge of 1953. A hypothetical breach in the defences is considered here, through which over 2,500,000m^3 of water enters the area with the hydrograph and entry point indicated in 7. A 10-hour period is simulated, allowing the water to continue spreading through the computational domain.

The same hypothetical event is considered by Liang et al. (2008), Liang (2010), and Vacondio et al. (2012). These studies used a 10m resolution grid of 360,000 cells, whereby buildings and vegetation were removed from LiDAR altimetry data to create a digital terrain model (DTM). The software presented herein allows us to go further. Updated datasets were created using a 2007 survey, and the resulting DTM and elevation model with buildings included (DEM) are shown in 7 at 2m resolution with 9,013,004 cells. As previously a uniform Manning coefficient of 0.035 is applied across the domain. Cells within and north of the River Thames are excluded from computation. Transmissive boundary conditions are imposed at the domain edges. The final volume error is <1% of the inflow volume for all Thamesmead simulations discussed.

We start by attempting to reproduce the simulation previously explored in literature with a 10m DTM, in order to further validate the model. The inundation results are presented in 8, where only minimal disagreement can be identified against the results in Liang (2010). This stems from differences in treatment of inflow boundary conditions for the new software. It can clearly be seen that a lesser flood extent is found with the new DTM in 9 for the same resolution. The updated terrain model depicts a deeper network of canals and lakes, offering increased storage.

Simulations were undertaken using both 32-bit and 64-bit floating-point arithmetic. Results presented are for 64-bit unless otherwise indicated. Whilst the average error in depth introduced by 32-bit arithmetic is small, the localised errors are in some cases unacceptably large. Errors

(a) Digital terrain model (DTM) and inflow location for Thamesmead

(b) Digital elevation model with buildings included (DEM).

(c) Volumetric discharge at the breach location.

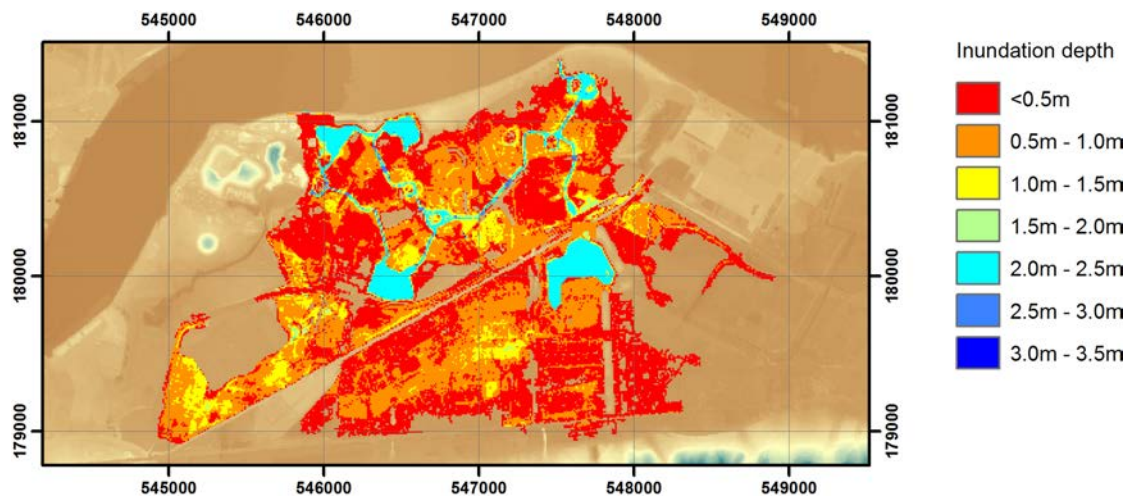**Figure 7** Boundary conditions and topography used in Thamesmead simulations.

Figure 8 Inundation results after a 10-hour period for the 10m resolution DTM used in Liang (2010).

were more pronounced at 10m resolution with the DEM; the average error exceeded 0.1m and the largest error was 0.98m. By contrast the average errors for all simulations at 5m and 2m resolution were <0.01m. The magnitude of these errors is in part a function of the numerical scheme employed, although it would not be entirely surprising if coarse resolutions are more sensitive to numerical resolution given the larger volume affected by the same change in depth. Further research is required.

The final extent and inundation depth for the three resolutions and two elevation models employed are shown in 9. The building layout is unusual in Thamesmead, with numerous connected networks of buildings. Small tunnels and alleyways allow for pedestrian access, and the elevation models have been adjusted to ensure these are present as a flow pathway. These passages are small however, hence the buildings can be expected to reflect a large volume of the flow impacting them; this is confirmed by the results obtained. Inclusion of buildings results in an increased inundation in the west of the domain. Spatial resolution also has a substantial impact on the flood extent. 10 shows the magnitude of the velocities after six hours, with the highest velocities present at the highest resolutions where narrow gaps are clearly resolved, concentrating flow to alleyways and streets. As a consequence flood progression is more rapid at higher resolutions. In this case, coarse resolutions may result in underestimation of flood extent.

Additional simulations were run for each grid resolution with different Manning's n values from 0.01 to 0.09 at 0.02 increments, giving 30 simulations in total. This allows the sensitivity to parameterisation to be assessed. The minimum and maximum extent of flooding after 10 hours across the calibration range is shown in 9. As expected, a lower Manning's n results in a larger area of inundation by the end of the simulation. Consistent with the findings of Yu and Lane (2006), coarser grid resolutions exhibit lower sensitivity to Manning's n, while there is a substantial range in final extent for both the DEM and DTM at 2m resolution. This is unsurprising given the higher localised velocities at fine grid resolutions. Low sensitivity to parameterisation at coarse grid resolutions is not a justification for using these resolutions however, as the difference in inundation from coarse to fine resolution grids far exceeds the scope of influence any parameterisation might have. Consequently there is a clear need for sub-grid scale representation of topographic features, dynamically adaptive grids, or use of high-resolution grids throughout.

The high-velocity nature of this hypothetical event contributes to the higher sensitivity, in contrast to studies exploring slow fluvial floodplain inundation by overtopping of defences rather than breach. Fewtrell et al. (2011) demonstrate that floodplain sensitivity is significant in both a LISFLOOD and ESTRY-TUFLOW simulations of the Carlisle 2005 flood, using a 25m grid. However a more detailed representation of the in-channel flow dynamics using a high resolution grid and shock-capturing scheme throughout is shown to decrease floodplain sensitivity to have very little effect with a 2m grid for the same event by Smith et al. (2015). Evidently high-resolution simulations alone are insufficient for flood risk analyses in defence breach situations, requiring comprehensive exploration of inundation with different parameterisations to fully assess the potential consequences for hypothetical and statistically-derived flood events. At its most extreme, the use of coarse grids and a single Manning's n in broad-scale flood risk analysis could significantly underestimate the threat, as demonstrated by the marked differences in results obtained herein.
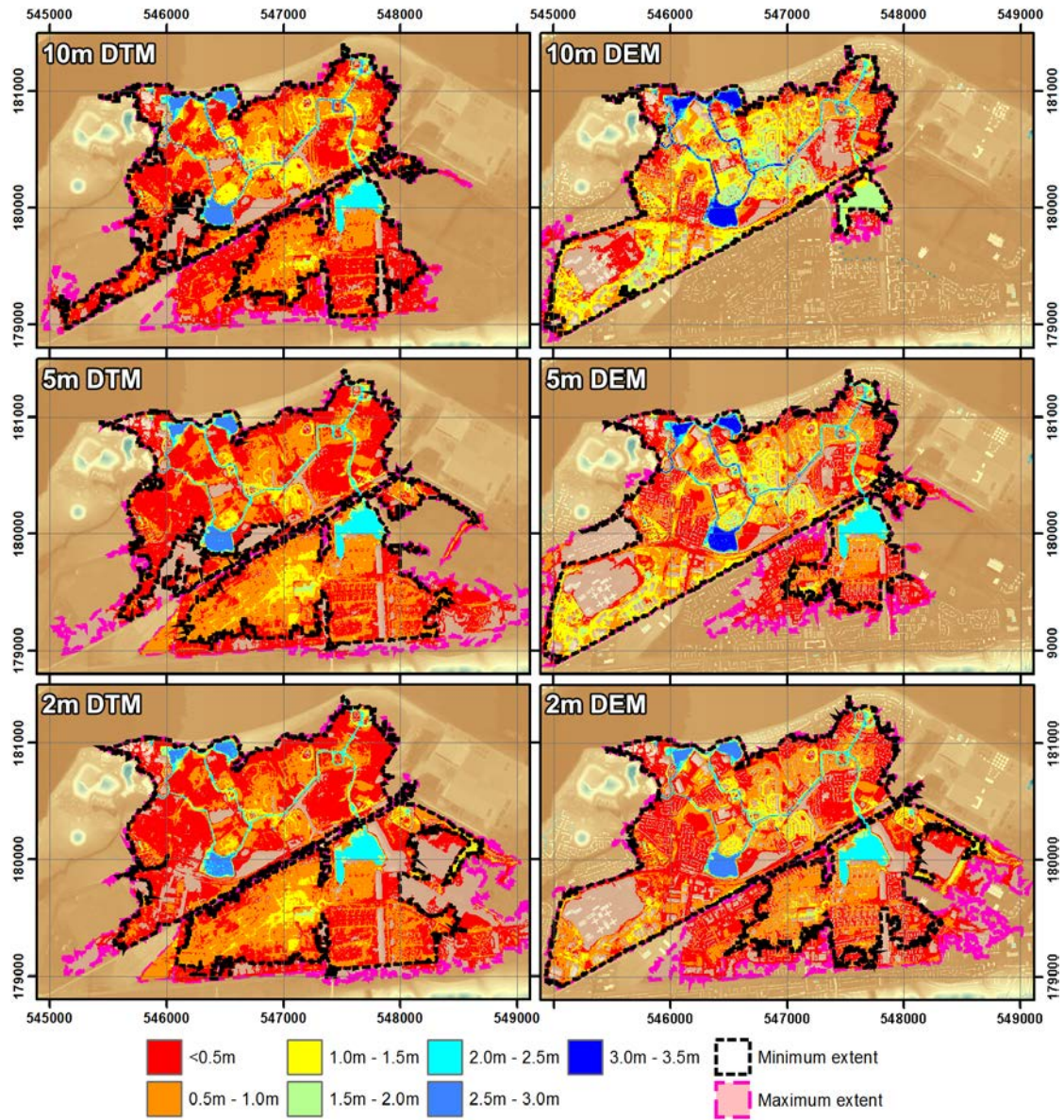
Figure 9 Inundation results after a 10-hour period using DEM and DTM at different spatial resolutions, and the minimum and maximum inundation extents with Manning values from 0.01 to 0.09.
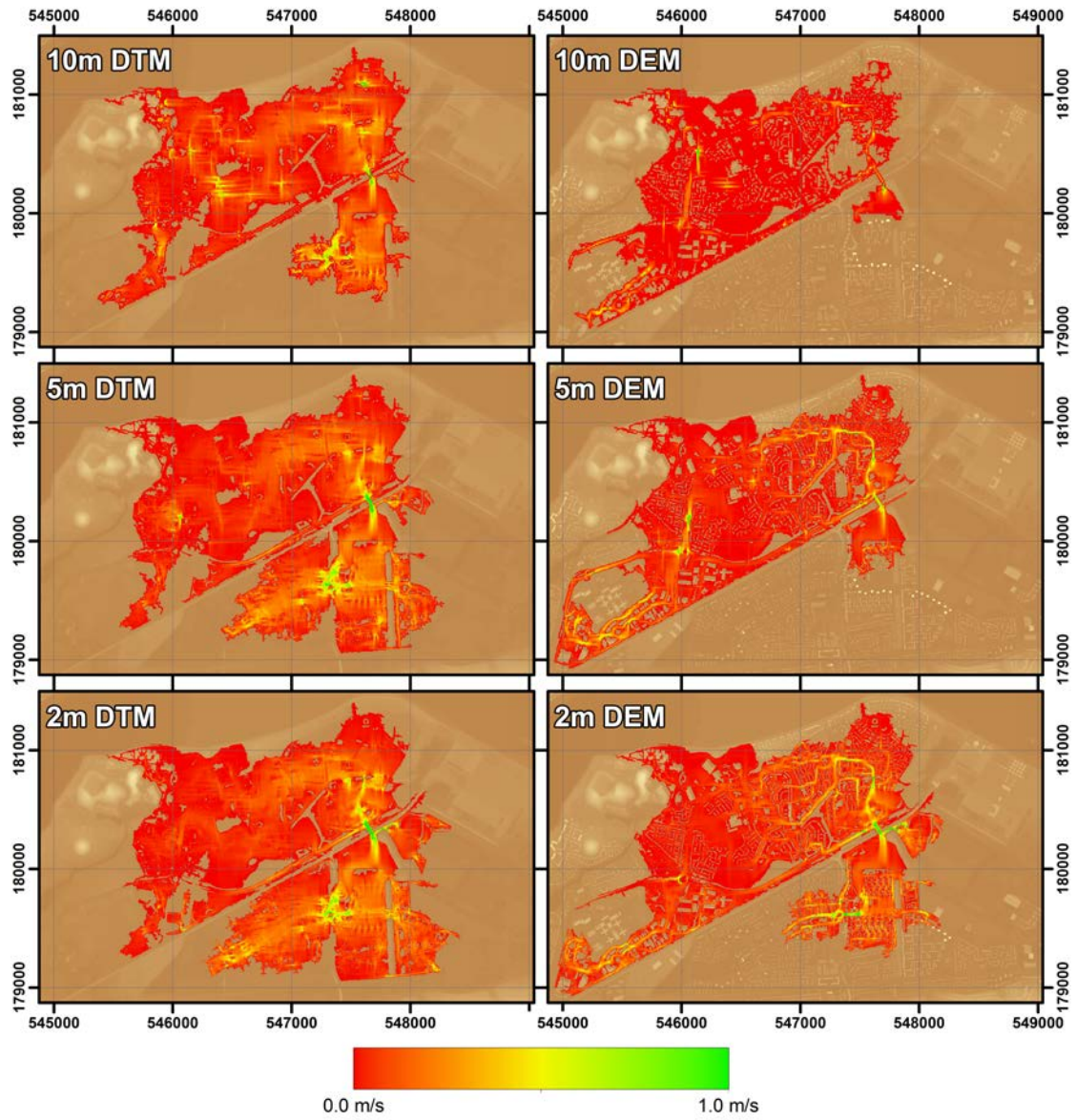
Figure 10 Magnitude of velocities at 6 hours using DEM and DTM at different spatial resolutions.

Table 2 Simulation run-times for Thamesmead in minutes using three different processing devices at different resolutions and DEM or DTM.

Cell elevations	Spatial resolution (m)	Floating-point arithmetic resolution	CPU Intel Xeon E5-2609	GPU AMD FirePro V7800	GPU NVIDIA Tesla M2075
DTM	10	32-bit	10.53	0.73	0.83
		64-bit	23.50	1.88	2.40
	5	32-bit	47.20	2.78	3.25
		64-bit	109.12	10.50	10.33
	2	32-bit	636.72	40.43	40.20
		64-bit	>1,800.00	154.25	137.88
DEM	10	32-bit	10.57	0.77	0.85
		64-bit	10.87	1.92	2.35
	5	32-bit	50.55	3.03	3.45
		64-bit	118.20	10.68	10.97
	2	32-bit	675.18	40.73	40.20
		64-bit	>1,800.00	147.62	137.70

Total simulation times for a 10 hour period in Thamesmead with different processing devices are given in 2. The software presented herein takes advantage of all four CPU cores made available to it, giving a much more realistic comparison between the achievable performance of GPU and CPU devices than comparisons against single-threaded code which may not have been optimised. Multiple orders of magnitude speed-up should not be expected when comparing against optimised code, based on vendor-quoted performance figures (Brodtkorb et al., 2012; Smith and Liang, 2013); examination of device peak compute power in terms of floating point operations per second (FLOPS) reaffirms that such dramatic levels of performance boost are highly unlikely. The AMD device performs well in all of the simulations, only slightly slower than the NVIDIA, but retailing at a much lower price. For large domains with millions of cells it is possible to reduce simulation time to a fifteenth or less of the multi-core CPU equivalent for the software presented herein.

5 Conclusions

We have presented a GPU-accelerated shallow flow model for urban flood modelling. Benefiting from a shock-capturing capability as a result of implementing a first-order accurate finite-volume Godunov-type scheme, the model is able to reproduce a wide range of complex flow hydrodynamics including hydraulic jump-like transcritical flow with discontinuities. It is there-

fore well-suited to urban flood modelling where local complex flow hydrodynamics may occur due to flood waves interacting with complex structures and topographic features. The model also ensures non-negative water depth through a reconstruction technique, thereby allowing robust simulation of realistic flood events with wetting and drying without causing numerical instability.

In the past, due to high computational expense, it has been challenging to apply such a sophisticated fully-2D model for high-resolution simulations across large spatial extents. With the assistance of OpenCL and increased support for the standard by mainstream vendors, the presented modelling package enables simulations on a range of devices including multi-core CPUs and GPUs, which allows us to leverage the benefits of heterogeneous computing. It has been observed from the simulation results that significant speed-up is achievable even with inexpensive desktop GPUs not designed for scientific use.

With this new GPU-accelerated modelling package, high-resolution flood modelling with shock-capturing finite-volume schemes becomes feasible. The results for Thamesmead reveal that high-resolution flood modelling predicts markedly different results to coarse resolutions, justifying the need to comprehensively depict complex urban topographies. Preliminary work has also been conducted to compare the use of 32- and 64-bit floating-point arithmetic. Using 32-bit arithmetic, significant errors in mass conservation occur and computational efficiency is hindered by small time steps in the Glasgow test, occurring due to the extremely small water depths following rainfall and domain wetting and drying. Therefore further research is required to conclude whether 32-bit arithmetic is sufficiently accurate for flood simulations.

Acknowledgements

References

Advanced Micro Devices Inc (2010). ATI FirePro V7800.

Advanced Micro Devices Inc (2011). AMD Accelerated Parallel Processing OpenCL: Programming Guide. Technical report, Advanced Micro Devices, Inc., Sunnyvale, CA.

Bates, P. D., Horritt, M. S., and Fewtrell, T. J. (2010). A simple inertial formulation of the shallow water equations for efficient two-dimensional flood inundation modelling. *Journal of Hydrology*, 387(1-2):33–45.

Brodtkorb, A. R., Saetra, M. L., and Altinakar, M. (2012). Efficient shallow water simulations on GPUs: Implementation, visualization, verification, and validation. *Computers and Fluids*, 55:1–12.

Chen, A. S., Evans, B., Djordjević, S., and Savić, D. A. (2012). Multi-layered coarse grid modelling in 2D urban flood simulations. *Journal of Hydrology*, 470-471:1–11.

Costabile, P., Costanzo, C., and Macchione, F. (2009). Two-dimensional numerical models for overland flow simulations. *River Basin Manage. V*, pages 137–148.

Costabile, P., Costanzo, C., and Macchione, F. (2012). Comparative analysis of overland flow models using finite volume schemes.

Crossley, M., Lamb, R., and Waller, S. (2009). A fast two-dimensional floodplain inundation model.

Fewtrell, T. J., Neal, J. C., Bates, P. D., and Harrison, P. J. (2011). Geometric and structural river channel complexity and the prediction of urban inundation. *Hydrological Processes*, 25:3173–3186.

Gallegos, H. A., Schubert, J. E., and Sanders, B. F. (2009). Two-dimensional, high-resolution modeling of urban dam-break flooding: A case study of Baldwin Hills, California. *Advances in Water Resources*, 32(8):1323–1335.

Guinot, V. (2012). Multiple porosity shallow water models for macroscopic modelling of urban floods. *Advances in Water Resources*, 37:40–72.

Hunter, N. M., Bates, P. D., Neelz, S., Pender, G., Villanueva, I., Wright, N. G., Liang, D., Falconer, R. A., Lin, B., Waller, S., Crossley, A. J., and Mason, D. C. (2008). Benchmarking 2D hydraulic models for urban flooding. *Proceedings of the Institution of Civil Engineers Water Management*, 161(1):13–30.

Intel Corporation (2012). Intel {Xeon} {Processor} {E5-2600} {Series}.

Kalyanapu, A. J., Shankar, S., Pardyjak, E. R., Judi, D. R., and Burian, S. J. (2011). Assessment of GPU computational enhancement to a 2D flood model. *Environmental Modelling and Software*, 26:1009–1016.

Karimi, K., Dickson, N. G., and Hamze, F. (2010). A Performance Comparison of {CUDA} and OpenCL. *CoRR*, abs/1005.2.

Kubatko, E. J., Bunya, S., Dawson, C., and Westerink, J. J. (2009). Dynamic p-adaptive Runge-Kutta discontinuous Galerkin methods for the shallow water equations. *Computer Methods in Applied Mechanics and Engineering*, 198:1766–1774.

Liang, Q. (2010). Flood Simulation Using a Well-Balanced Shallow Flow Model. *Journal of Hydraulic Engineering*, 136:669–675.

Liang, Q. and Borthwick, A. G. L. (2009). Adaptive quadtree simulation of shallow flows with wet-dry fronts over complex topography. *Computers and Fluids*, 38:221–234.

Liang, Q., Du, G., Hall, J. W., and Borthwick, A. G. L. (2008). Flood inundation modeling with an adaptive quadtree grid shallow water equation solver. *Journal of Hydraulic Engineering*, 134:1603–1610.

Neal, J., Villanueva, I., Wright, N., Willis, T., Fewtrell, T., and Bates, P. (2012). How much physical complexity is needed to model flood inundation? *Hydrological Processes*, 26(15):2264–2282.

Neal, J. C., Fewtrell, T. J., Bates, P. D., and Wright, N. G. (2010). A comparison of three parallelisation methods for 2D flood inundation models. *Environmental Modelling and Software*, 25(4):398–411.

Néelz, S. and Pender, G. (2010). Benchmarking of 2D Hydraulic Modelling Packages. Technical report, Environment Agency, Bristol, UK.

Néelz, S. and Pender, G. (2013). Benchmarking the latest generation of 2D Hydraulic Modelling Packages. Technical report, Environment Agency, Bristol, UK.

NVIDIA Corporation (2009). NVIDIA OpenCL JumpStart Guide v0.9.

NVIDIA Corporation (2011). Tesla¢ M-Class GPU Computing Modules Accelerating Science. Technical report, NVIDIA Corporation, Santa Clara, CA.

Saetra, M. L. and Brodtkorb, A. R. (2012). Shallow Water Simulations on Multiple GPUs. *Proceedings of the Para 2010 Conference Part II, Lecture Notes in Computer Science*, 7134(2012):56–66.

Schubert, J. E. and Sanders, B. F. (2012). Building treatments for urban flood inundation models and implications for predictive skill and modeling efficiency. *Advances in Water Resources*, 41:49–64.

Smith, L. S. and Liang, Q. (2013). Towards a generalised GPU/CPU shallow-flow modelling tool. *Computers and Fluids*, 88:334–343.

Smith, L. S., Liang, Q., and Quinn, P. F. (2015). Towards a hydrodynamic modelling framework appropriate for applications in urban flood assessment and mitigation using heterogeneous computing. *Urban Water Journal*, 12(1):67–78.

Soares-Frazão, S., Lhomme, J., Guinot, V., and Zech, Y. (2008). Two-dimensional shallow-water model with porosity for urban flood modelling.

Toro, E. F., Spruce, M., and Speares, W. (1994). Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4:25–34.

Vacondio, R., Rogers, B. D., Stansby, P. K., and Mignosa, P. (2012). SPH Modeling of Shallow Flow with Open Boundaries for Practical Flood Simulation.

Wang, Y., Liang, Q., Kesserwani, G., and Hall, J. W. (2011). A positivity-preserving zero-inertia model for flood simulation. *Computers and Fluids*, 46(1):505–511.

Yeh, G. T., Shih, D. S., and Cheng, J. R. C. (2011). An integrated media, integrated processes watershed model. *Computers and Fluids*, 45(1):2–13.

Yu, D. and Lane, S. N. (2006). Urban fluvial flood modelling using a two-dimensional diffusion-wave treatment, part 1: Mesh resolution effects. *Hydrological Processes*, 20:1541–1565.

Zhang, C., Liang, Q., and Yin, J. (2013). A first-order adaptive solution to rapidly spreading flood waves. *Progress in Computational Fluid Dynamics*, 13(1):1–10.

Zhang, H., Wang, Y., Liang, Q., Smith, L. S., and Kilsby, C. G. (2014). Non-negative depth reconstruction for a two-dimensional partial inertial inundation model. *Journal of Hydroinformatics*, 16(5):1158–1177.