A fully hydrodynamic urban flood modelling system representing buildings, green space and interventions

V. Glenis[*], V. Kutija and C.G. Kilsby

Affiliation address

School of Engineering, Newcastle University, NE1 7RU, UK.

Authors' email addresses

V. Glenis, vassilis.glenis@ncl.ac.uk

V. Kutija, vedrana.kutija@gmail.com

C.G. Kilsby, chris.kilsby@ncl.ac.uk

Corresponding author

Vassilis Glenis

School of Engineering

Cassie Building

Newcastle University

NE1 7RU

UK

Email: vassilis.glenis@ncl.ac.uk

Tel: +44 (0)191 208 5221

Fax: +44 (0)191 208 6502

Abstract

City Catchment Analysis Tool – CityCAT- is a novel software system for rapid assessment of combined pluvial and fluvial flood risk using a unique combination of: efficient software architecture throughout and especially in the numerical part; use of standard, readily available data sets; efficient algorithms for grid generation; and robust and accurate solutions of the flow equations. It is based on advanced software architecture and accurate solutions for complex free-surface flow over the terrain distinguishing between permeable and impermeable surfaces and taking into account effects of man-made features such as buildings as obstacles to flow. The software is firstly rigorously validated with demanding test cases based on analytical solutions and laboratory studies. Then the unique capability for assessment of the effectiveness of specific flood alleviation interventions across large urban domains, such as roof storage on buildings or introduction of permeable surfaces, is demonstrated.

Preliminary Information

Keywords

Urban flood model, Object-oriented numerics, shock-capturing, finite-volume, green urban infrastructure

Highlights

- An object-oriented 2D hydrodynamic model is presented for use in urban flood analysis and design.
- The model retains accuracy in representing complex flows while allowing rapid modelling of large city domains at 1m resolution.
- Buildings and green urban infrastructure are flexibly and accurately represented.

Software availability section

Name of software: CityCAT

Developer: Newcastle University

Contact: vassilis.glenis@ncl.ac.uk

Year first available: 2010

Hardware required: 32bit or 64bit CPU

Software required: Windows or Linux operating system

Programming language: Delphi

Programme size: ~5mb, Memory: depends on application

Availability: Contact author and web download will be available from: http://research.ncl.ac.uk/citycat (pending publication)

Cost: Free (to researchers and for demo version)

1 Background

Assessment of pluvial flood risk in urban environments is complicated because it is sensitive to the space-time characteristics of rainfall, topography, performance of urban drainage systems and local runoff and surface flow processes influenced by buildings and other man-made features. There are three modelling approaches used in current engineering practice for assessment of pluvial flood risk: the topographic index analysis, the 2D overland flow routing and the so called dual drainage modelling, see Hankin et al. (2008).

The topographic index analysis, also called raster screening approach, uses Digital Elevation Models (DEMs) with no rainfall input. Hence, it is not really a flow modelling tool but an assessment tool based on topography only. It combines areas defined as flat, areas close to drainage pathways and areas identified as local depressions into areas of high risk. Tools for this analysis are readily available in GIS systems and their ease of use makes the method attractive. However, there is little evidence of validation that areas identified as high risk correlate to areas that have been flooded in the past (Pitt, 2008).

The 2D overland flow routing method usually applies uniform rainfall over the whole domain and models overland flow using some form of the depth averaged shallow water equations which are solved by one of the standard numerical methods. Depending on the level of approximation (e.g. fully hydrodynamic, diffusive or kinematic wave) and the numerical method (e.g. finite differences, finite elements, finite volumes with shock-capturing schemes) there is a number of different sub-types of models in this category. If no adjustments are made for lost volume of water due to infiltration and inflow into the drainage network, the models of this type usually overestimate the run-off volumes. The magnitude of this overestimation becomes less significant as the severity (or return period) of the event being modelled becomes greater. Due to the complexity of urban situations, it has been reported that models based on “shock-capturing” schemes are best suited to the task where raster based models, which do not take into account the inertial forces, are not able to simulate the same flood extent as the other models (Hunter et al., 2008; Mignot et al., 2006). There are several different approaches to capture complex flow paths taking into account the effect of buildings as obstacles to flow (Schubert et al., 2008; Syme, 2008). The first approach uses additional surface roughness and it is the most widely used approach, however, it has difficulties with modelling of predominantly flat areas while parameterisation and calibration of larger urban areas can be extremely difficult and time consuming, see Alcrudo (2004). The second approach amends the standard free surface equations with the equations of flow through porous media (Sanders et al., 2008; Soares-Frazao et al., 2008) and is able to produce realistic flow patterns without the need of extensive calibration. The third approach manipulates the DEMs so that buildings are represented by upward “extrusion” of the DEM surface. This approach can be time consuming for large areas and extrusion of the buildings' height introduces inclined walls with very steep slopes which can lead to numerical instabilities (Alcrudo, 2004). A compromise variation of this approach limits the height of the buildings to typically 0.3 m, which avoids instabilities but introduces major ambiguity as flow over the buildings is allowed. A fourth approach, often called the "building hole model", takes buildings into account explicitly by treating their outer walls as solid boundaries with flows through these boundaries set to zero (Costanzo and Macchione, 2006; Schubert et al., 2008). This approach is accurate in describing the flow patterns but if the cell sizes are large compared to the gaps between the buildings it can erroneously predict blockages which do not exist in reality, see Neal et al. (2009).

Dual drainage models integrate sewer drainage network models with overland flow routing models with diverse levels of coupling and complexity. They all consist of a one-dimensional hydraulic drainage network and a representation of the surface flow either as a one-dimensional network based on the road network or a two-dimensional domain derived from the DEM (Mark et al., 2004). The connection between the two components is usually only partially coupled, meaning the drainage network model can pass the volume of flooding to the surface model but there is no flow from the surface to the drainage network. In a fully coupled system, the volume of flooding is passed from the drainage network model to the surface and *vice versa* (Allitt et al., 2009; Bertsch et al., 2017; Liu et al., 2015; Noh et al., 2018). Teng et al. (2017) presented recently a review of methods and advances in flood modelling. See also the review paper by Bach et al. (2014).

2 Introduction

In this paper, a new software for modelling, analysis and visualisation of surface water flooding, City Catchment Analysis Tool – CityCAT, is presented and validated. It includes a 2D overland flow routing model that enables rapid assessment of combined pluvial and fluvial urban flood risk and effects of different flood alleviation measures.

The architecture of CityCAT is based on the object-oriented approach. This offers great flexibility in development and allows rapid extension of functionality (Kutija and Murray, 2007). Also, the efficiency of the computational algorithms is improved considerably by removing the conditional statements ("If-then-else" type statements) during run time. This is achieved by making all the decisions during the initial set up which is a main feature of the object-oriented design.

CityCAT uses standard datasets: the Digital Elevation Model (DEM) for the topography and the UK Ordnance Survey MasterMap™ data to delineate the urban features such as buildings, roads and permeable surfaces. For other countries, GIS datasets at varying levels of detail may be available to be used to delineate the urban features. Simulations are usually driven by rainfall events over the whole or part of the domain and/or time dependent boundary conditions of flow and/or water depth time series at the boundaries of the domain. The computational grid is generated automatically using the DEM and the OS-MasterMap™ data or GIS datasets. The buildings layer from OS-MasterMap™ or GIS datasets is used to exclude the buildings footprint from the grid. This improves the ability of the model to capture realistically the flow paths in urban areas and reduces the simulation time due to the reduction in the number of computational cells. The removed cells are used to generate the model's buildings layer which is used in the roof drainage algorithms. Also, during the grid generation the layers from OS-MasterMap™ or GIS datasets which describe the permeable areas are used to locate the permeable cells and assign appropriate properties to them. Additionally, polygons can be used when the grid is generated to delineate areas to assign different friction coefficients, soil properties, spatially distributed rainfall and initial conditions for reservoirs, lakes and ponds

The simulation of the free surface flow is based on the full 2D shallow water equations (Tan, 1992) and the solution is obtained using high-resolution finite volume methods with shock-capturing schemes (Toro, 2013) which are able to accurately capture propagation of flood waves as well as wetting and drying of the domain. The Green-Ampt method is used to estimate the infiltration over the pervious areas as a function of the soil hydraulic conductivity, porosity and suction head (Kutílek and Nielsen, 1994; Warrick, 2003). The solution of the Green-Ampt equation for infiltration is obtained using an iterative method. Also, a roof storage algorithm simulating retention of rainwater on the roofs can be applied to the buildings layer of the grid.

The model provides two types of graphical outputs: time series of water depths and flow velocities at selected locations for the whole duration of the simulation and snapshot maps of water depths and velocities at different times during the simulations. These maps can be combined into animations of the simulations.

3 Software architecture

3.1 Object-oriented approach

The model is written in Delphi (Embarcadero) and, uniquely amongst hydrodynamic models for flood risk assessment, is completely object-oriented. Both the Graphical User Interface (GUI) and the numerical engine of the model are designed and implemented following the object-oriented approach. This enables the connection and direct interaction between corresponding objects of the GUI and the numerical engine e.g. cells, cell lines and interfaces. As a result, the properties of each numerical cell can be accessed and if required easily edited from the GUI during the setup of the model. Also, during the simulation values of the properties of the numerical cells can be displayed and continuously updated in the GUI enabling real-time graphing of water depths and velocities.

The main features and the advantages of the object-oriented design of the numerical algorithm for the solution of the 2D flow equations are illustrated here by means of an example showing the structure of some of the main objects involved, their main properties and inter-connectivity while the complete details of the numerical algorithms are given in the following section.

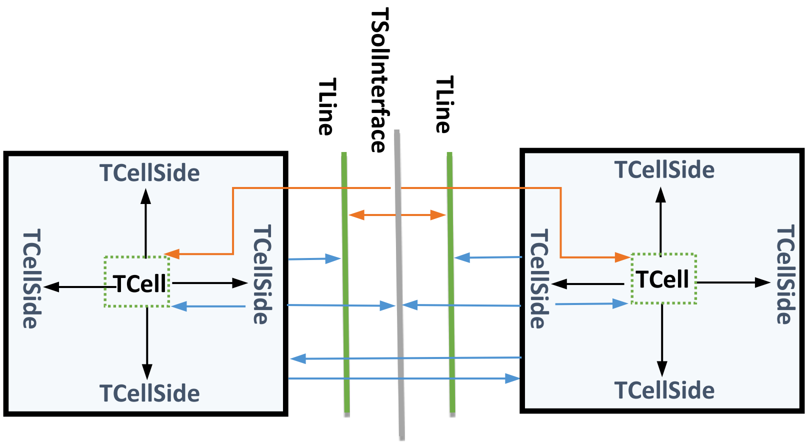


Fig. 1. Interconnection of computational objects

In the structure presented in Fig. 1 four different object types are used: *TCell*, *TCellSide*, *TLine* and *TSolInterface*. The complete solution algorithm is split into methods encapsulated within appropriate objects. In Fig. 1 pointers are presented by arrows with their origins at the host object and the arrowhead at the object they point to. They provide connections between objects and access to fields/data needed for the execution of the methods.

Each cell object *TCell* has properties (cell id, area, elevation, etc.), fields (water depth, *Vx* - velocity in x direction, *Vy* - velocity in y direction, CellSidesList, etc.) and methods (set initial conditions, rotations, fluxes, integration, etc.). The *CellSidesList* is a list of pointers and is used to hold the connections with the cell side objects *TCellSide*.

Each cell side object *TCellSide* has pointers to its cell *TCell*, the cell line *TLine*, the solution interface *TSolInterface* and the neighbouring cell *TCell*. This object has only pointers and does not have any methods. Its purpose is to hold the connections between the objects.

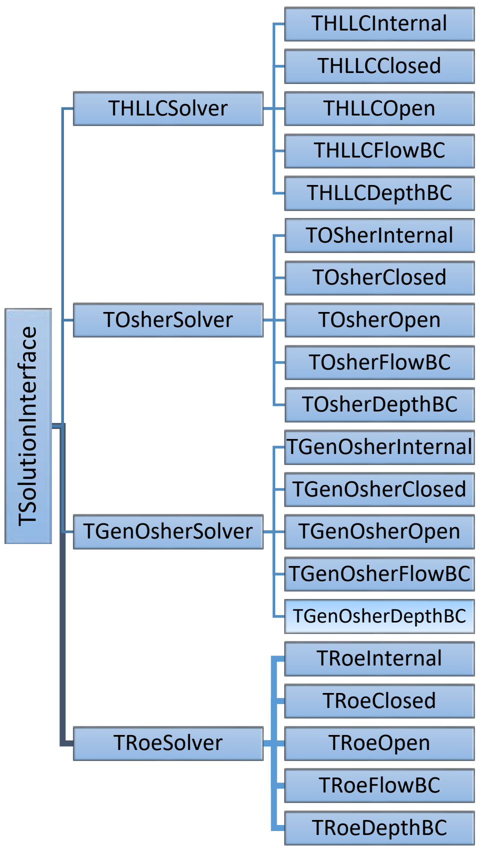


Fig. 2. TSolInterface object family tree

The solution interface *TSolInterface* has pointers to the left cell line, the left cell, the right cell line and the right cell. The *TSolInterface* is the parent object and during the construction of the solution the appropriate instances of descendant objects are generated depending on the type of the Riemann solver and if it is an internal or external solution interface. Furthermore, for the external interfaces, there are different types of objects for different boundary conditions. The TSolInterface object family tree is presented in Fig. 2. Five Riemann solvers are implemented in the model: the HLL *THLLSolver*, the HLLC solver *THLLCSolver*, The Roe solver *TRoeSolver*, the Osher-Solomon solver *TOsherSolver* and the Generalised Osher-Solomon solver *TGenOsherSolver* which are derived from the parent object *TSolInterface*. Further extension of the family tree takes into account if the interface is internal or external and which boundary condition is specified. The advantage of this approach is that all descendant objects inherit the pointers structure from the parent object and only the methods for the calculation of the fluxes are overridden. Other Riemann solvers can be implemented by creating a descendant object from the TSolInterface object.

During the setup of the model appropriate instances of TSolInterface object family are created taking into account the Riemann solver and if it is an internal or an external interface. This eliminates the need for “if-then-else” statements during the simulations and aids code execution efficiency.

In Algorithm 1, an example of a standard procedural code for the calculation of the fluxes at the interfaces is shown and it is clear that within a simulation time loop, there is an extensive need to check which solution needs to be used. In simulations with millions of time steps and millions of cell interfaces, this presents a heavy computational burden.

Algorithm 1. *Example of a procedural code for calculation of fluxes at cell interfaces at one time step*

for each Solution Interface **in** SolutionInterfaceList **do:**
  if solver = HLLC **then**:
    if internal interface **then** compute HLLC internal interface
    else if closed external interface **then** compute HLLC closed interface
    else if open external interface **then** compute HLLC open interface
    else if water level interface **then** compute HLLC water level interface
    else if discharge interface **then** compute HLLC discharge interface
  else if solver = Osher **then**:
    …………………………………
    …………………………………
    …………………………………
    …………………………………
  else if .... **then:**
    …………………………………
    …………………………………
    …………………………………
    …………………………………
  endif
  ……………………………………………
  ……………………………………………
enddo

When the object-oriented aproach is used all the decisions are performed at the beginning of the simulation as it is shown in Algorithm 2.

Algorithm 2. Example of an object-oriented code for creation of appropriate objects at cell interfaces.

if solver = HLLC **then**:
  if internal interface **then**:
    Create HLLC internal interface
    Add HLLC solution interface to SolutionInterfaceList

```
    else:
        if closed external interface then: Create HLLC closed external interface
        else if open external interface then: Create HLLC open external interface
        else if water level external interface then: Create HLLC water level external interface
        else if discharge external interface then: Create HLLC discharge external interface
        endif
        Add HLLC solution interface to SolutionInterfaceList
    endif
else if solver = Osher then:
    …………………………………….
    …………………………………….
    …………………………………….
    …………………………………….
endif
```

After all the solution interface objects are created and added to the list *SolutionInterfaceList* the fluxes at every time step are calculated by calling just one method as seen in Algorithm 3. Although only one method is called, different implementations are triggered in different objects due to the polymorphism of the object-oriented code.

Algorithm 3. Example of an object oriented code for calculation of fluxes at cell interfaces at one time step.

```
for each Solution Interface in SolutionInterfaceList do:
    compute flux
enddo
```

3.2 Software versions and deployment in the Cloud

The available versions of the model are: a) 32bit or 64bit for Windows without GUI; b) 32bit or 64bit for Linux without GUI; and c) a 32bit for Windows with a GUI (see Fig. 3). The versions without the GUI are multithreaded and take advantage of multiple cores CPUs to reduce the execution time.

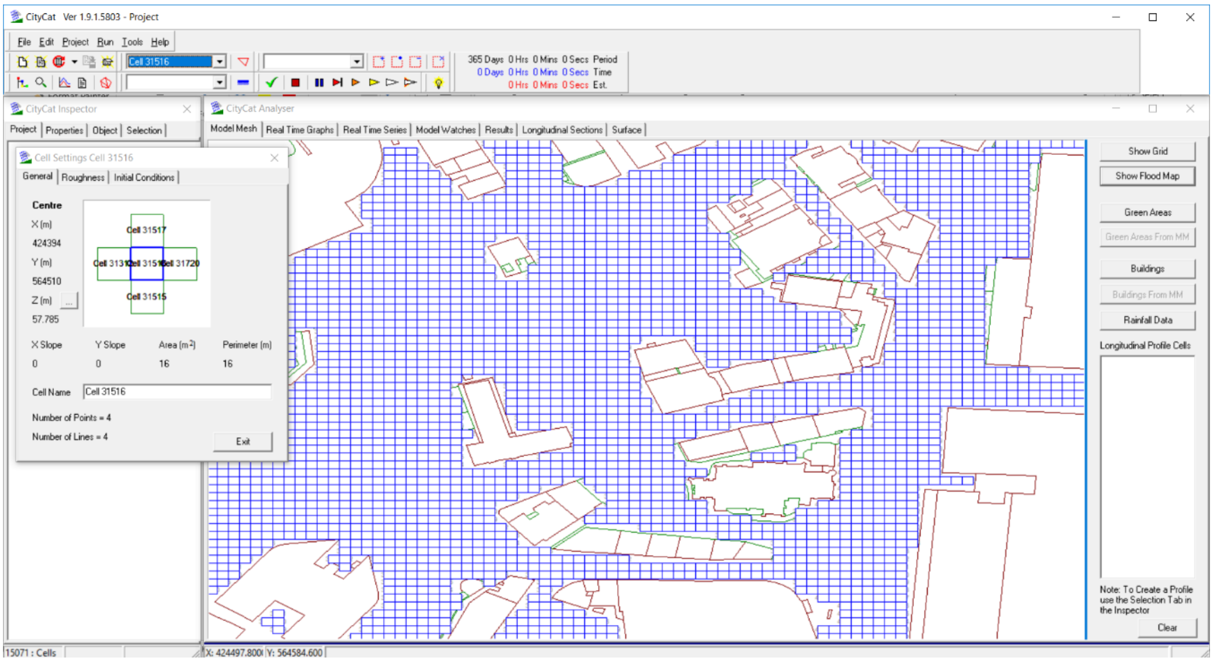


Fig. 3. Screen shot of the GUI of CityCAT

CityCat has been deployed and used to run parameter sweep jobs in both the Amazon's Elastic Compute Cloud (EC2) and Microsoft's Azure Cloud.

In Amazon's EC2 a high throughput model was used to deploy a Condor cluster of Linux virtual machines. Each job was instantiated by passing a single integer number as part of the command line arguments to select the configuration files and a script was used to wrap each job: a) decompress the files needed for each simulation, b) run the main program, and c) compress and upload the results to the master Condor computer on the Cloud. For details, see Glenis et al. (2013).

In Microsoft's Azure Cloud the Azure Batch service was used to run parameter sweep jobs where CityCat was used to model 571 European cities using a range of different storm events (Guerreiro et al., 2017). The results can be found at: http://ceg-research.ncl.ac.uk/ramses/ .

4 Grid generation algorithms

The numerical methods used for the solution of overland flow in CityCAT can use regular or irregular grids but the model only generates regular grids based on the resolution of the DEM. However, irregular grids generated by other models or grid generators can be read in and used by the model.

The buildings are taken into account as solid (no flow) boundaries by default and as such their footprint needs to be removed from the computational domain for the overland flow. Boundaries of buildings can be selectively opened to represent inflow and outflow of flood water. As buildings are defined with irregular polygon outlines they have to be "cut into" the original grid generated on the basis of the DEM. Exclusion of the covered cells from the original grid can be done according to three different algorithms. In algorithm A, a cell is excluded from the computational domain if any part of it is covered by a building. In algorithm B, a cell is excluded from the computational domain only if the whole cell is covered by a building and in algorithm C, a cell is excluded from the computational domain if the centroid of the cell lies inside a building. See Fig. 4 for a graphical illustration of the algorithms and Fig. 5 for a CityCAT computational domain with excluded buildings' footprint using algorithm A.

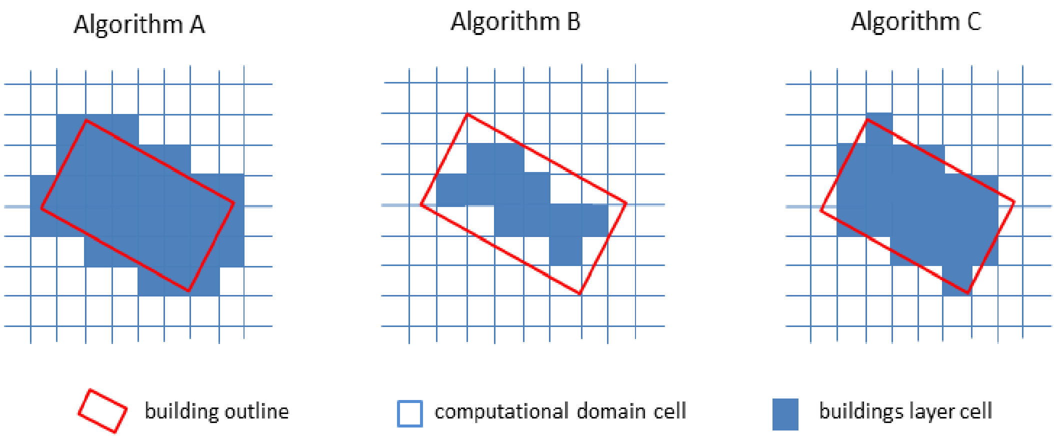


Fig. 4. Algorithms for exclusion of cells from the computational domain and inclusion into the buildings layer

Which of the three algorithms is the best depends on the size of the grid and the size of the gaps between the buildings.

Note that this is different to the standard approach used in other models which retain the buildings as areas of (arbitrarily) higher elevation or allow water to flow through them by assigning them specific roughness or porosity parameters.

For built-up areas, our procedure substantially reduces the number of the cells in the computational domain (see Fig. 5), allowing major reduction of the run time in comparison to the conventional models. The cells which are removed from the computational domain are not lost from the system. They form the "buildings" layer which is used in the roof drainage part of the solution algorithm.

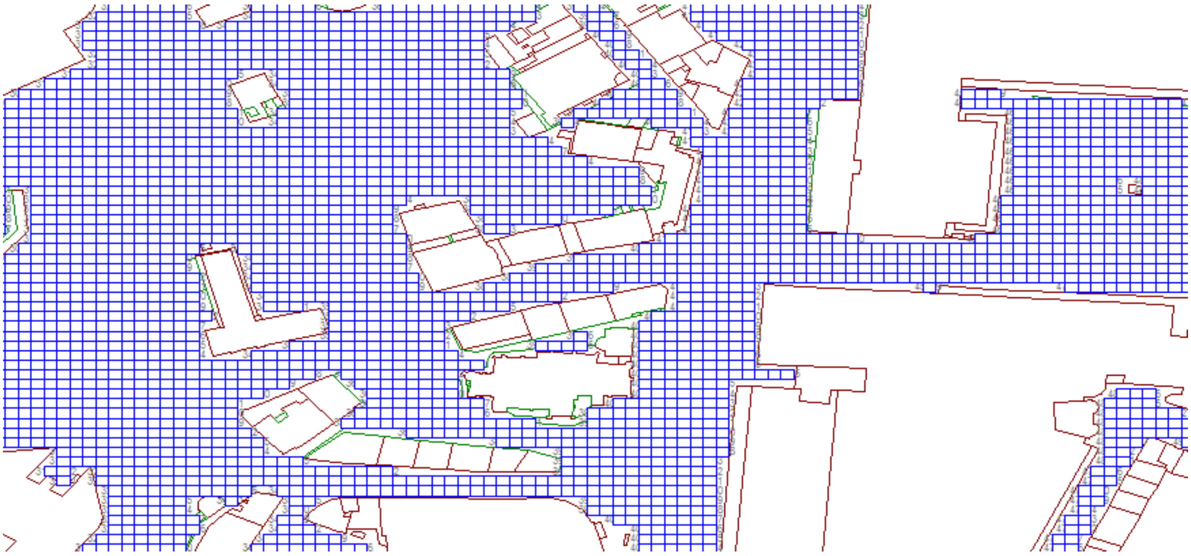


Figure 5. An example of the CityCat computational domain with exclusion of buildings using algorithm A

MasterMap data are used to delineate the urban features such as: buildings, roads and permeable surfaces. A parser based on the Simple API for XML (SAX) which is an event-based sequential access parser has been developed in order to parse the raw “gml” Mastermap data. The parser is very efficient and requires much less memory than Document Object Model (DOM)-style parsers.

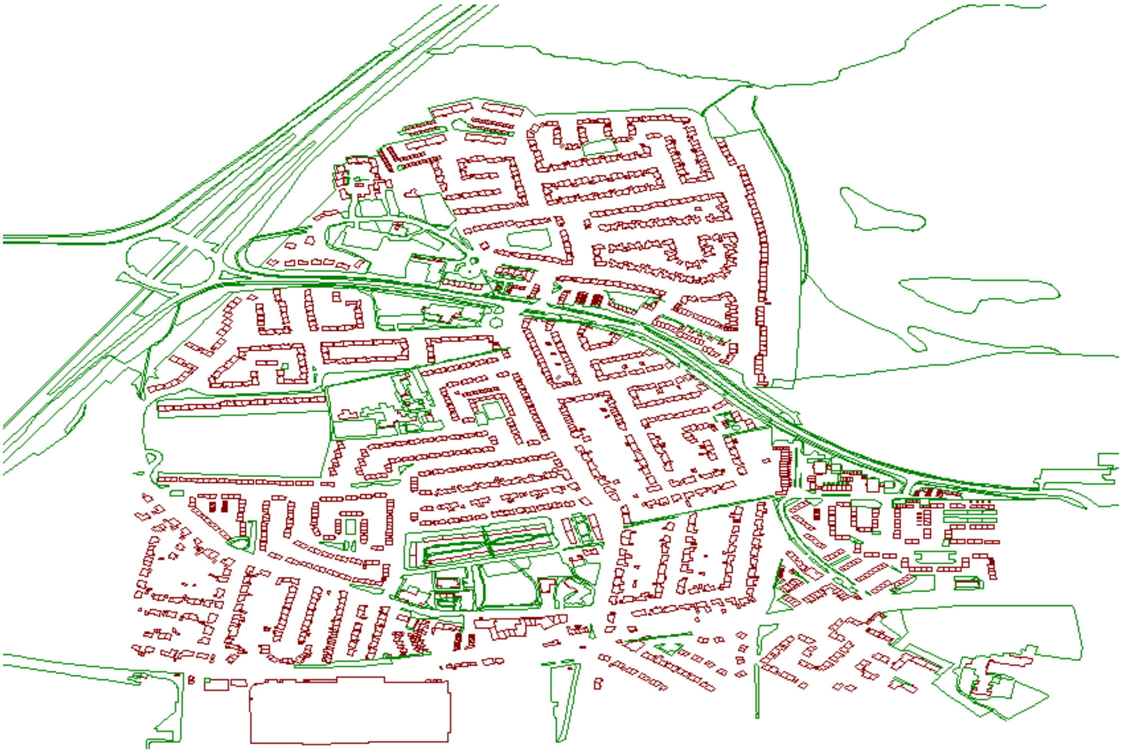


Fig. 6. An example of the buildings and green surfaces polygons extracted from the Master Map

In addition to the algorithms for extracting the buildings and green surfaces polygons from the MasterMap™ layers (Fig. 6), CityCat can also read polygons prepared by other software packages. This option is mainly used to define areas of different roughness, different soil properties and proposed new developments, new green areas, etc.

The object-oriented architecture of the model, and the polygon representation of buildings supports direct and interactive editing of attributes (elevations, flow permeability and building properties). This unique feature ensures realistic and efficient simulation of flow around and into buildings, as well as allowing the use of roof drainage algorithms for each building

5 Numerical solutions

The overland flow component of CityCAT is based on the full shallow water equations (Tan, 1992) and the solution is obtained using the method of finite volumes with shock-capturing schemes. This method has been successfully applied in the field of free surface flows, see (Alcrudo and Garcia-Navarro, 1993; Brufau et al., 2004; Castro Díaz et al., 2013; Fraccarollo and Toro, 1995; Michel-Dansac et al., 2016; Mingham and Causon, 1998). In CityCAT we have implemented and evaluated a range of different Riemann solvers: the HLL (Harten et al., 1983), the HLLC (Toro et al., 1994), the Roe (Roe, 1981) with the Harten-Hyman entropy fix (Harten and Hyman, 1983), the Osher-Solomon (Osher and Solomon, 1982), and the Generalised Osher-Solomon (Dumbser and Toro, 2011b).

The Osher-Solomon Riemann solver is one of the most accurate solvers (Erduran et al., 2002) and has the following properties: it is a complete solver as it contains all the waves; it is differentiable with respect to its arguments and therefore suitable for implicit schemes; it is entropy satisfying which means that it does not require an entropy fix at sonic points; and it captures slow moving shocks. The Osher-Solomon Riemann solver is a very robust solver, however, very rarely has been applied to complex systems of equations due to its complexity as it requires the evaluation of a path-integral in phase-space, see Toro (2013) for details for the Euler equations. The idea proposed by Dumbser and Toro (2011b) to evaluate the path integral numerically using Gaussian quadrature simplifies the Osher-Solomon Riemann solver and makes it an attractive solver for complex systems of hyperbolic conservation laws (Dumbser and Toro, 2011a). In this solver only the eigenstructure of the hyperbolic system needs to be known in order to evaluate the viscosity matrix of the numerical flux. In case the eigenstructure is not known then it can be approximated numerically or an alternative Osher-Solomon Riemann solver proposed by Castro et al. (2016) can be used where the eigenstructure of the system is not needed and the viscosity matrix of the numerical flux is approximated using functional evaluations of the Jacobian based on Chebyshev polynomials or rational functions.

Here we present in detail the shallow water equations and details of the Generarised Osher-Solomon solver. A second-order Total Variation Diminish (TVD) scheme (Harten, 1983) based on the Weighted Average Flux (Toro, 1989) and the Generarised Osher-Solomon solver is also presented.

5.1 Formulation of the Swallow Water Equations

The shallow water equations can be written as follows:

$$\partial_t \mathbf{Q} + \partial_x \mathbf{F}(\mathbf{Q}) + \partial_y \mathbf{G}(\mathbf{Q}) = \mathbf{S}(\mathbf{Q}), \qquad \mathbf{Q} = \mathbf{Q}(\mathbf{x}, t) \in \mathcal{D},\ \mathbf{x} = (x, y) \in \Omega \subset \mathbb{R}^2,\ \ t > 0 \qquad (1.1)$$

Where $\mathcal{D}$ is an open convex subset of $\mathbb{R}^p$; p is the number of conservation laws; $\mathbf{Q}$ is the conserved quantities vector; $\mathbf{F}, \mathbf{G}: \mathcal{D} \to \mathbb{R}^p$ are the flux vectors; and $\mathbf{S}: \mathcal{D} \to \mathbb{R}^p$ is the source terms vector. With initial conditions: $\mathbf{Q}(\mathbf{x}, 0) = \mathbf{Q}_0(\mathbf{x}),\ \mathbf{x} \in \Omega$; and boundary conditions: $\mathbf{Q}(\mathbf{x}, t) = \mathbf{Q}_{BC}(\mathbf{x}, t),\ \mathbf{x} \in \partial\Omega$, $\mathrm{t} > 0$.

The vectors are given as follows:

$$\mathbf{Q} \equiv [q_1, q_2, q_3]^T = \left[h, hv_x, hv_y\right]^T; \mathbf{F}(\mathbf{Q}) \equiv [f_1, f_2, f_3]^T = \left[hv_x, hv_x^2 + gh^2/2, hv_x v_y\right]^T$$

$$\mathbf{G}(\mathbf{Q}) \equiv [g_1, g_2, g_3]^T = \left[hv_y, hv_x v_y, hv_y^2 + gh^2/2\right]^T;$$

$$\mathbf{S}(\mathbf{Q}) = \boldsymbol{R} - \boldsymbol{L} + \boldsymbol{S_o} - \boldsymbol{S_f}$$

Where v_x and v_y represent the depth-averaged velocity components in the x and y directions respectively; h is the water depth; g is the gravity acceleration; $\boldsymbol{R} = [R, 0,0]^T$ is the rainfall intensity; $\boldsymbol{L} = [L, 0,0]^T$ is the infiltration rate; $\boldsymbol{S_o} = \left[0, gh\partial_x z_b, gh\partial_y z_b\right]^T$ is the bed slope source term and z_b denotes the bed elevation; $\boldsymbol{S_f} = \left[0, ghSf_x, ghSf_y\right]^T$ is the friction term with: $Sf_x = n^2 v_x (v_x^2 + v_y^2)^{1/2} h^{-4/3}$, $Sf_y = n^2 v_y (v_x^2 + v_y^2)^{1/2} h^{-4/3}$ and n denotes the Manning's roughness coefficient.

Integration of (1.1) over a control volume and application of the Gauss's theorem gives:

$$\int_V \partial_t \mathbf{Q}\, \mathrm{dV} + \oint_{\partial V} \mathbf{H} \cdot \mathbf{n}\, \mathrm{ds} = \int_V \mathbf{S}\, \mathrm{dV}$$

Where $\mathbf{H} = (\mathbf{F}, \mathbf{G})$ is the flux tensor; V is the control volume over which the integration is performed; ∂V is the boundary of the control volume V; and $\mathbf{n} \equiv (n_x, n_y) \equiv (\cos\theta, \sin\theta)$ is the outward normal vector to ∂V and θ is the angle with the x-axis measured anticlockwise.

The domain is divided into cells $(V_i)_{i\in\mathbb{Z}}$ and the total normal flux though the edges of each cell using the rotational invariance property can be written as:

$$\oint_{\partial V_i} \mathbf{H}\cdot\mathbf{n}\,\mathrm{ds} = \sum_{k=1}^{NE} \int_{m_k}^{m_{k+1}} \mathbf{H}\cdot\mathbf{n}_k\,\mathrm{ds} = \sum_{k=1}^{NE} \int_{m_k}^{m_{k+1}} \boldsymbol{T}_k^{-1}\mathbf{F}(\mathbf{T}_k\mathbf{Q})\,\mathrm{ds} \quad (1.2)$$

Where $\mathbf{T}_k \equiv \mathbf{T}(\theta_k)$ is the rotation matrix; $\mathbf{T}_k^{-1} \equiv \mathbf{T}^{-1}(\theta_k)$ is the inverse rotation matrix; NE is the number of edges of the V_i cell; and m_k denotes the cell vertices. The vector of the transformed conservative variables and the normal fluxes at the edges of each cell in the local rotated $(\hat{x}, \hat{y})$ Cartesian frame can be written as: $\widehat{\mathbf{Q}}_k \equiv \mathbf{T}_k\mathbf{Q} = [h, hu, hv]^T$ and $\widehat{\mathbf{F}}_k \equiv \mathbf{F}(\mathbf{T}_k\mathbf{Q}) \equiv \mathbf{F}(\widehat{\mathbf{Q}}_k) = [hu, hu^2 + gh^2/2, huv]^T$

Where $u = v_x\cos\theta + v_y\sin\theta$, $v = -v_x\sin\theta + v_y\cos\theta$

$$\mathbf{T}(\theta_k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}, \mathbf{T}^{-1}(\theta_k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

The integral (1.2) can be approximated as:

$$\sum_{k=1}^{NE} \int_{m_k}^{m_{k+1}} \boldsymbol{T}_k^{-1}\mathbf{F}(\mathbf{T}_k\mathbf{Q})\,\mathrm{ds} \approx \sum_{k=1}^{NE} L_k\,\boldsymbol{T}_k^{-1}\widehat{\mathbf{F}}_k$$

Where L_k denotes the length of the k^{th} edge of the cell. The numerical flux through the cell edges can be obtained by solving the Riemann problem for the rotated conservative equations:

$$\partial_t\widehat{\mathbf{Q}} + \partial_{\hat{x}}\mathbf{F}(\widehat{\mathbf{Q}}) = 0 \qquad (1.3)$$

with initial data:

$$\widehat{\mathbf{Q}}_k(\hat{x}, 0) = \begin{cases} \widehat{\mathbf{Q}}_{k,L} & if\ \hat{x} \le 0 \\ \widehat{\mathbf{Q}}_{k,R} & if\ \hat{x} > 0 \end{cases}$$

Where L and R denote the cells on the left and right hand sides of the interface.

A fully discretised first-order finite-volume conservative scheme can be obtained by:

$$\mathbf{Q}_i^{n+1} = \mathbf{Q}_i^n - \frac{\Delta t}{A_i}\sum_{k=1}^{NE} L_k\,\boldsymbol{T}_k^{-1}\widehat{\mathbf{F}}_k\left(\widehat{\mathbf{Q}}_{k,L}, \widehat{\mathbf{Q}}_{k,R}\right) + \Delta t\mathbf{R}_i - \Delta t\mathbf{L}_i + \frac{\Delta t}{A_i}\sum_{k=1}^{NE} L_k\,\widehat{\mathbf{S}}_{\boldsymbol{o},k} - \Delta t\boldsymbol{S}_{f_i}$$

(1.4)

Where A_i is the area of the cell V_i; Δt is the time step and $t^{n+1} = t^n + \Delta t$; NE is the number of edges of each cell; $\mathbf{Q}_i^n$ is the averaged integral of the solution at time t^n; $\hat{\mathbf{F}}_k\left(\hat{\mathbf{Q}}_{k,L}, \hat{\mathbf{Q}}_{k,R}\right)$ is the numerical flux through the cell edge and for simplicity we denote $\mathbf{f}_k \coloneqq \hat{\mathbf{F}}_k\left(\hat{\mathbf{Q}}_{k,L}, \hat{\mathbf{Q}}_{k,R}\right)$; $\mathbf{R}_i$ is the rainfall Intensity; $\mathbf{L}_i$ is the infiltration rate; $\hat{\mathbf{S}}_{o,k}$ is the bed slope source term at each cell interface; and $\boldsymbol{S_f}_i$ is the friction source term.

Full details on how each term is computed are presented in the following sections.

5.2 Generalised Osher-Solomon Riemann solver

The system of equations (1.1) is hyperbolic and strictly hyperbolic when $h > 0$. Every linear combination of the Jacobian matrices $\mathbf{A}(\mathbf{Q}) = \partial\mathbf{F}(\mathbf{Q})/\partial\mathbf{Q}$ and $\mathbf{B}(\mathbf{Q}) = \partial\mathbf{G}(\mathbf{Q})/\partial\mathbf{Q}$ has real eigenvalues and linearly independent eigenvectors and can be diagonalized. The Jacobian matrix $\mathbf{A}(\mathbf{Q})$ can be expressed as: $\mathbf{A}(\mathbf{Q}) = \mathbf{K}(\mathbf{Q})\,\Lambda(\mathbf{Q})\,\mathbf{K}^{-1}(\mathbf{Q})$.

Where $\mathbf{K}(\mathbf{Q})$ is the right eigenvectors matrix; $\mathbf{K}(\mathbf{Q})^{-1}$ is its inverse; and $\Lambda(\mathbf{Q})$ is the diagonal matrix with the eigenvalues λ_i.

$$\mathbf{A}(\mathbf{Q}) = \begin{bmatrix} 0 & 1 & 0 \\ -u^2 + c^2 & 2u & 0 \\ -uv & v & u \end{bmatrix}, \mathbf{K}(\mathbf{Q}) = \begin{bmatrix} 1 & 0 & 1 \\ u - c & 0 & u + c \\ v & 1 & v \end{bmatrix}, \mathbf{K}^{-1}(\mathbf{Q}) = \frac{1}{2c}\begin{bmatrix} u + c & -1 & 0 \\ -2vc & 0 & 2c \\ -u + c & 1 & 0 \end{bmatrix}$$

$$\Lambda(\mathbf{Q}) = diag(\lambda_1, \lambda_2, \lambda_3) = diag(u - c, u, u + c)$$

Where $c = \sqrt{gh}$ is the celerity.

We introduce the notation:

$\lambda_i^+ = \max(\lambda_i, 0)$; $\lambda_i^- = \min(\lambda_i, 0)$; $|\lambda_i| = \lambda_i^+ - \lambda_i^-$; for $i = 1,2,3$

$\Lambda^+(\mathbf{Q}) = diag(\lambda_1^+, \lambda_2^+, \lambda_3^+)$; $\Lambda^-(\mathbf{Q}) = diag(\lambda_1^-, \lambda_2^-, \lambda_3^-)$;

$|\Lambda(\mathbf{Q})| = diag(|\lambda_1|, |\lambda_2|, |\lambda_3|) = \Lambda^+(\mathbf{Q}) - \Lambda^-(\mathbf{Q})$

$|\mathbf{A}(\mathbf{Q})| = \mathbf{K}(\mathbf{Q})|\Lambda(\mathbf{Q})|\mathbf{K}^{-1}(\mathbf{Q})$

The Osher-Solomon flux is given by:

$$\mathbf{f}_k = \frac{1}{2}\left(\mathbf{F}\left(\hat{\mathbf{Q}}_{k,L}\right) + \mathbf{F}\left(\hat{\mathbf{Q}}_{k,R}\right)\right) - \frac{1}{2}\int_{\hat{\mathbf{Q}}_{k,L}}^{\hat{\mathbf{Q}}_{k,R}} |\mathbf{A}(\mathbf{Q})|\,d\mathbf{Q}$$

In the original Osher-Solomon solver (Osher and Solomon, 1982) the integral is evaluated by using tractable paths which follow the integral curves of the eigenvectors to connect the left and right states: $\widehat{\mathbf{Q}}_{k,L}$and $\widehat{\mathbf{Q}}_{k,R}$.

In the Generalised Osher-Solomon solver the left and the right states are connected via a path in the phase-space:

$\Phi(\xi) = \widehat{\mathbf{Q}}_{k,L} + \xi(\widehat{\mathbf{Q}}_{k,R} - \widehat{\mathbf{Q}}_{k,L}),\ \xi \in [0,1]$

Where $\Phi(\xi)$ is a Lipschitz continuous function with $\Phi(0) = \widehat{\mathbf{Q}}_{k,L}$ and $\Phi(1) = \widehat{\mathbf{Q}}_{k,R}$

The flux can be written as:

$$\mathbf{f}_k = \frac{1}{2}\left(\mathbf{F}\left(\widehat{\mathbf{Q}}_{k,L}\right) + \mathbf{F}(\widehat{\mathbf{Q}}_{k,R})\right) - \frac{1}{2}\int_0^1 \left|\mathbf{A}\left(\Phi(\xi)\right)\right| \partial_\xi \Phi \, d\xi$$

$$= \frac{1}{2}\left(\mathbf{F}\left(\widehat{\mathbf{Q}}_{k,L}\right) + \mathbf{F}(\widehat{\mathbf{Q}}_{k,R})\right) - \frac{1}{2}\left(\int_0^1 \left|\mathbf{A}\left(\Phi(\xi)\right)\right| d\xi\right)(\widehat{\mathbf{Q}}_{k,R} - \widehat{\mathbf{Q}}_{k,L})$$

Where $\int_0^1 \left|\mathbf{A}\left(\Phi(\xi)\right)\right| d\xi$ is the viscosity matrix of the numerical flux and represents the numerical diffusion.

Transformation of the integral to $[-1,1]$ gives:

$$\mathbf{f}_k = \frac{1}{2}\left(\mathbf{F}\left(\widehat{\mathbf{Q}}_{k,L}\right) + \mathbf{F}(\widehat{\mathbf{Q}}_{k,R})\right) - \frac{1}{4}\left(\int_{-1}^1 \left|\mathbf{A}\left(\Phi(0.5\cdot\xi + 0.5)\right)\right| d\xi\right)\left(\widehat{\mathbf{Q}}_{k,R} - \widehat{\mathbf{Q}}_{k,L}\right) \tag{1.5}$$

The integral in (1.5) is approximated using a three-point Gaussian quadrature and the Generalised Osher-Solomon flux is given by:

$$\mathbf{f}_k^{osh} = \frac{1}{2}\left(\mathbf{F}\left(\widehat{\mathbf{Q}}_{k,L}\right) + \mathbf{F}(\widehat{\mathbf{Q}}_{k,R})\right) - \frac{1}{4}\left(\sum_{j=1}^{3} w_j \left|\mathbf{A}\left(\Phi\left(0.5\cdot\xi_j + 0.5\right)\right)\right|\right)\left(\widehat{\mathbf{Q}}_{k,R} - \widehat{\mathbf{Q}}_{k,L}\right)$$

$$= \frac{1}{2}\left(\mathbf{F}\left(\widehat{\mathbf{Q}}_{k,L}\right) + \mathbf{F}(\widehat{\mathbf{Q}}_{k,R})\right) - \frac{1}{4}\left(\sum_{j=1}^{3} w_j \left|\mathbf{A}_j\right|\right)\left(\widehat{\mathbf{Q}}_{k,R} - \widehat{\mathbf{Q}}_{k,L}\right) \tag{1.6}$$

Where $\left|\mathbf{A}_j\right| := \left|\mathbf{A}\left(\Phi\left(0.5\cdot\xi_j + 0.5\right)\right)\right|$; w_j are the weights ; and ξ_j are the points of evaluation.

$w_1 = w_3 = \frac{5}{9}$, $w_2 = \frac{8}{9}$, $\xi_1 = -\sqrt{\frac{3}{5}}$, $\xi_2 = 0$, $\xi_3 = \sqrt{\frac{3}{5}}$

The steps required for the calculation of the flux are given below:

1. Let $p_j = 0.5 * \xi_j + 0.5$, for $j = 1,2,3$

Calculate $\Phi(p_j)$, for $j = 1,2,3$ and define three new states:

$\mathbf{Q}_j \equiv \Phi(p_j) = [h_j, h_j u_j, h_j v_j]^T$ for $j = 1,2,3$

2. For each of the states $j = 1,2,3$ calculate: $c_j, |\lambda_{j,1}|, |\lambda_{j,2}|, |\lambda_{j,3}|$
3. For each of the states $j = 1,2,3$ calculate the absolute matrix:

$$|\mathbf{A}_j| \equiv \left|\mathbf{A}\left(\Phi(p_j)\right)\right| = \mathbf{K}(\mathbf{Q}_j)|\Lambda(\mathbf{Q}_j)|\mathbf{K}^{-1}(\mathbf{Q}_j)$$

$$= \frac{1}{2c_j}\begin{bmatrix} |\lambda_{j,1}|(u_j + c_j) + |\lambda_{j,3}|(-u_j + c_j) & -|\lambda_{j,1}| + |\lambda_{j,3}| & 0 \\ |\lambda_{j,1}|(u_j^2 - c_j^2) + |\lambda_{j,3}|(c_j^2 - u_j^2) & -|\lambda_{j,1}|(u_j - c_j) + |\lambda_{j,3}|(u_j + c_j) & 0 \\ |\lambda_{j,1}|v_j(u_j + c_j) - |\lambda_{j,2}|2v_j c_j + |\lambda_{j,3}|v_j(-u_j + c_j) & v_j(|\lambda_{j,3}| - |\lambda_{j,1}|) & 2c|\lambda_{j,2}| \end{bmatrix}$$

4. Use equation (1.6) to calculate the flux at the cell interface

5.3 Second-order TVD WAF numerical flux

The TVD WAF numerical flux is an extension of the first order Godunov upwind scheme. The TVD WAF is second order accurate in time and space in the smooth regions and it was first presented for the solution of the Euler equations (Toro, 1989). Application of the TVD WAF numerical flux to the shallow water equations can be found in (Ata et al., 2013; Fernández-Nieto and Narbona-Reina, 2008; Guan et al., 2013; Kim et al., 2009; Loukili and Soulaimani, 2007; Toro, 1992). All these applications of the WAF are based on the HLLC Riemann solver. Here we present a TVD WAF numerical flux which is based on the Generalised Ohser-Solmon Riemann solver.

The additional steps for the computation of the TVD WAF flux are: a) approximation of the speed of the waves; b) computation of the Courant number for each wave; c) computation of the flux limiter function; and d) computation of the weights of the WAF flux, see (Toro, 2013).

For the approximation of the wave speeds for the non-linear waves S_L, S_R and for the linear contact wave S_* we use an adaptive approximate-state Riemann solver similar to the one presented by Loukili and Soulaimani (2007). An initial approximation of the water depth in the star region (wedge between the two non-linear waves) is obtained using a two-rarefaction approximate-state Riemann solver. If the approximated water depth in the star region is less or equal to the water depth in the left and right cell h_L, h_R then the two-rarefaction approximate-state Riemann solver is used for the estimation of the speed of each wave. Otherwise the two-shock approximate-state Riemann solver is used for the estimation of the speed of each wave. Details about approximate-state Riemann solvers can be found in (Toro, 2013).

Also, special treatment is required in the presence of a wet-dry front. Algorithm 4 below provides details for the calculation of the speed of the waves.

***Algorithm 4**. Calculation of wave speeds.*

if $h_L and\ h_R > 0$ **then**:

 First approximation using a two-rarefaction approximate-state Riemann solver

 $$h_0 \coloneqq \frac{1}{g}\left(0.5 \cdot (c_L + c_R) + 0.25 \cdot (u_L - u_R)\right)^2$$

 if $h_0 \leq Min(h_L, h_R)$ **then:**

  use two-rarefaction approximate-state Riemann solver

  $h_* = h_0$

  $u_* = 0.5 \cdot (u_L + u_R) + c_L - c_R$

 else if $h_0 > Min(h_L, h_R)$ **then:**

  use two-shock approximate-state Riemann solver

  $$p_L = \sqrt{\frac{g(h_0 + h_L)}{2h_0 h_L}}\ , \qquad p_R = \sqrt{\frac{g(h_0 + h_R)}{2h_0 h_R}}$$

  $$h_* = \frac{p_L h_L + p_R h_R + u_L - u_R}{p_L + p_R}$$

  $$u_* = 0.5 \cdot (u_L + u_R) + 0.5 \cdot \left(p_R(h_* - h_R) - p_L(h_* - h_L)\right)$$

 endif

 $$\alpha_L = \begin{cases} \dfrac{\sqrt{0.5 \cdot (h_* + h_L) h_*}}{h_L} & if\ h^* > h_L \\ 1 & if\ h^* \leq h_L \end{cases}, \qquad \alpha_R = \begin{cases} \dfrac{\sqrt{0.5 \cdot (h_* + h_R) h_*}}{h_R} & if\ h^* > h_R \\ 1 & if\ h^* \leq h_R \end{cases}$$

 $S_L = u_L - \alpha_L c_L$

 $S_R = u_R + \alpha_R c_R$

 $S_* = u_*$

else if $h_L = 0\ and\ h_R > 0$ **then:**

 $S_L = u_R - 2c_R$

 $S_R = u_R + c_R$

 $S_* = u_* = S_L$

else if $h_L > 0\ and\ h_R = 0$ **then:**

 $S_L = u_L - 2c_L$

 $S_R = u_L + 2c_L$

 $S_* = u_* = S_R$

endif

The calculation of the courant number (CN) for each wave is given by:

$$CN_L = \frac{S_L \Delta t}{\Delta x}, CN_R = \frac{S_R \Delta t}{\Delta x}, CN_* = \frac{S_* \Delta t}{\Delta x} \tag{1.7}$$

Godunov (1959) has shown that second or higher order schemes are not monotone and produce spurious oscillations at discontinuities. Harten (1983) proposed the Total Variation Diminishing (TVD) schemes to avoid spurious oscillations. The drawback of the TVD constraint is that the schemes reduce to first order at extrema. Here we apply the WAF flux limiter function to obtain a TVD WAF flux. For details about flux limiters, see (Sweby, 1984; Toro, 2013).

The WAF flux limiter function is given by:

$$\Psi(r, CN) = 1 - (1 - |CN|)B(r) \tag{1.8}$$

And the Flux limiters are given by:

Superbee limiter: $\boldsymbol{if}\ r > 0\ \boldsymbol{then}\text{: } B_{sb}(r) = Max[Min(1,2r), Min(2,r)]\ \boldsymbol{else}\text{: } B_{sb}(r) = 0$

van Leer limiter: $\boldsymbol{if}\ r > 0\ \boldsymbol{then}\text{: } B_{vl}(r) = 2r/(1+r)\ \boldsymbol{else}\text{: } B_{vl}(r) = 0$

van Albada limiter: $\boldsymbol{if}\ r > 0\ \boldsymbol{then}\text{: } B_{va}(r) = r(1+r)/(1+r^2)\ \boldsymbol{else}\text{: } B_{va}(r) = 0$

Minbee limiter: $\boldsymbol{if}\ r > 0\ \boldsymbol{then}\text{: } B_{mb}(r) = Min(1,r)\ \boldsymbol{else}\text{: } B_{mb}(r) = 0$

(1.9)

Where r is the ratio of upwind change to local change and is given by:

$$r_K = \frac{\Delta q_K^{upw}}{\Delta q_K^{loc}}, K = L, R, * \tag{1.10}$$

$$\Delta q_K^{loc} = q_{K,i+1} - q_{K,i}$$

$$\Delta q_K^{upw} = \begin{cases} q_{K,i} - q_{K,i-1}, & if\ S_K \leq 0 \\ q_{K,i+2} - q_{K,i+1}, & if\ S_K > 0 \end{cases}$$

For the left and the right non-linear waves the q_K is chosen as the water depth and for the contact linear wave the q_* is chosen as the tangential velocity.

$$\boldsymbol{if}\ \ K = L, R\ \ \boldsymbol{then}\text{: } q_K = h\ \ \boldsymbol{else\ if}\ \ K = *\ \ \boldsymbol{then}\text{: } q_* = v$$

The weights for the TVD WAF flux are given by:

$$w_L = 0.5 \cdot \left(1 + sign(CN_L)\Psi(r_L, CN_L)\right)$$
$$w_{LR} = 0.5 \cdot \left(sign(CN_R)\Psi(r_R, CN_R) - sign(CN_L)\Psi(r_L, CN_L)\right)$$
$$w_R = 0.5 \cdot \left(1 - sign(CN_R)\Psi(r_R, CN_R)\right)$$
$$w_{L*} = 0.5 \cdot \left(1 + sign(CN_*)\Psi(r_*, CN_*)\right)$$
$$w_{R*} = 0.5 \cdot \left(1 - sign(CN_*)\Psi(r_*, CN_*)\right)$$
$$(1.11)$$

The three components of the TVD WAF numerical flux $\mathbf{f}_k^{waf} = \left[f_{k,1}^{waf}, f_{k,2}^{waf}, f_{k,3}^{waf}\right]^T$ are given as follows:

$$f_{k,1}^{waf} = w_L f_1(q_{1,L}) + w_{LR} f_{k,1}^{osh} + w_R f_1(q_{1,R})$$
$$f_{k,2}^{waf} = w_L f_2(q_{2,L}) + w_{LR} f_{k,2}^{osh} + w_R f_2(q_{2,R})$$
$$\textbf{\textit{if}}\ w_{L*} > w_{R*}\ \textbf{\textit{then}}\text{:}\ f_{k,3}^{waf} = w_{L*} f_{k,3}^{osh} + w_{R*} v_R f_{k,1}^{osh}$$
$$\textbf{\textit{else if}}\ w_{L*} < w_{R*}\ \textbf{\textit{then:}}\ f_{k,3}^{waf} = w_{L*} v_L f_{k,1}^{osh} + w_{R*} f_{k,3}^{osh}$$
$$\textbf{\textit{else if}}\ w_{L*} = w_{R*}\ \textbf{\textit{then:}}\ f_{k,3}^{waf} = f_{k,3}^{osh}$$
$$(1.12)$$

The steps required for the calculation of the TVD-WAF Generalised Osher-Solomon flux are given below:

1. Use equation (1.6) to calculate the first order Generalised Osher-Solomon flux
2. Use Algorithm 1.1 to calculate the wave speeds
3. Use equations (1.7) to calculate the courant number (CN) for each wave
4. Use equation (1.10) to calculate the ratio of upwind change to local change
5. Use equations (1.8) and (1.9) to calculate the WAF flux limiter function
6. Use equation (1.11) to calculate the weights
7. Use equation (1.12) to calculate the TVD-WAF Generalised Osher-Solomon flux

5.4 Bed slope source term approximation and well-balanced schemes

An essential feature of a robust finite volume shock-capturing scheme is to be well-balanced (Greenberg and Leroux, 1996) or to satisfy the C-property (Bermúdez and Vázquez-Cendón, 1994). The upwind method (Bermúdez and Vázquez-Cendón, 1994; Garcia-Navarro and Vazquez-Cendon, 2000;

Vazquez-Cendon, 1999) and the hydrostatic reconstruction method (Audusse et al., 2004; Audusse and Bristeau, 2005) have been used for the construction of well-balanced, non-negative water depth schemes.

In the hydrostatic reconstruction the left and the right water depth values at an interface between two cells are reconstructed as:

$$h_L^{HR} = \max(0, h_L + z_{b,L} - z_{b,LR})$$
$$h_R^{HR} = \max(0, h_R + z_{b,R} - z_{b,LR})$$

Where $z_{b,L}$ and $z_{b,R}$ are the bed elevations of the cells on the left and right hand side of the interface; and $z_{b,LR}$ is the bed elevation at the interface and is given by: $z_{b,LR} = \max(z_{b,L}, z_{b,R})$.

The bed slope is approximated as:

$$\hat{\mathbf{S}}_{\boldsymbol{o},k} = \begin{bmatrix} 0 \\ g/2\left[(h_{i,k}^{HR})^2 - (h_i)^2\right]\mathbf{n}_k \end{bmatrix}$$

Where $h_{i,k}^{HR}$ is the hydrostatic reconstructed water depth at the k^{th} interface of the V_i cell; h_i is the water depth of the V_i cell; $\mathbf{n}_k$ is the outward normal vector to the k^{th} edge of the cell.

Details about the upwind method can be found in (Bermúdez and Vázquez-Cendón, 1994; Garcia-Navarro and Vazquez-Cendon, 2000; Vazquez-Cendon, 1999).

5.5 Infiltration source term

The evaluation of the infiltration rate $\mathbf{L}_i$ is based on the Green-Ampt method and estimates are needed for the hydraulic conductivity, the wetting front suction head and the porosity, for details see (Chow et al., 1988; Kutílek and Nielsen, 1994; Warrick, 2003). Some typical values of the infiltration parameters of the Green-Ampt model for different soils are presentenced in Table 1. For details, see Chow et al. (1988). The Green-Ampt infiltration equation is solved by the Newton–Raphson's method.

Table 1 – Typical values for the Green-Ampt model parameters for different soils (from Chow et al. (1988))

Soil	Porosity n	Effective porosity θe	Soil suction head ψ (cm)	Hydraulic conductivity K *(cm/h)*
Sandy loam	0.453	0.412	11.01	1.09
Loam	0.463	0.434	8.89	0.34
Silt loam	0.501	0.486	16.68	0.65

5.6 Stability condition

The numerical scheme presented above is explicit and the time step is given by:

$$\Delta t = CFL \cdot \min_{i \in \mathbb{Z}} \left(\frac{\min(d\chi_i)}{\left(u_{x,i}^2 + u_{y,i}^2\right)^{1/2} + (gh_i)^{1/2}} \right)$$

Where $d\chi_i$ denotes the distance between the i^{th} cell and its neighbouring cells; and CFL is the Courant-Friedrichs-Lewy condition and is set to: $CFL \leq 0.5$.

5.7 Roof drainage algorithm

The cells which are excluded from the overland flow domain are included in the 'buildings' layer of the model. The rain falling onto this layer is redistributed to the cells of the overland flow domain along the boundaries of the buildings. If a roof storage is specified then the rain falling onto the buildings layer is accumulated until the water depth on the roof reaches the specified storage depth. Any further rainfall is redistributed to the neighbouring cells of the overland flow domain.

The purpose of the roof storage algorithm is to enable assessment of the effect of potential rainwater harvesting policies could have on pluvial flooding. The algorithm used for the roof storage is very simple, however, more sophisticated algorithms for green and blue roofs are currently being developed and tested. Additionally, the object-oriented structure facilitates an easy and efficient way to extend the algorithms and the functionality of the model.

6 Case studies and validations

Three case studies have been chosen to firstly validate the model and then illustrate the capabilities of CityCAT. The first case is a validation using an analytic solution of moving boundary shallow water flow in a parabolic bowl. The second case is a validation of the model using data from a physical model study of a dam-break. The third, by contrast, is a real world case on a much larger domain with complex urban features and processes.

6.1 Case 1 – Moving boundary shallow water flow in a parabolic bowl

The moving boundary shallow water flow in a parabolic bowl with friction (Sampson et al., 2006) is used to assess the performance of the numerical solutions in tracking wet/dry interfaces. The analytical solutions for water depth and velocity are given by Thacker (1981) and Sampson et al. (2006). The fluid motion decays with time due to friction and finally converges to motionless steady state. The dimensions of the computational domain are: $[-5000,5000] \times [-5000,5000]$, which is divided into 200×200 cells and the size of each cell is 50m. The topography of the parabolic bowl is given by:

$$z(x,y) = \frac{h_0}{\alpha^2}(x^2 + y^2) \quad (1.13)$$

Where: $h_0 = 10m$ and $\alpha = 3000m$ are constants

The peak amplitude parameter is defined as:

$$p = \sqrt{\frac{8gh_0}{\alpha^2}}$$

If the friction parameter τ is smaller than the peak amplitude parameter then the analytical solution for the water free surface and the velocities $V_x\ and\ V_y$ are given by:

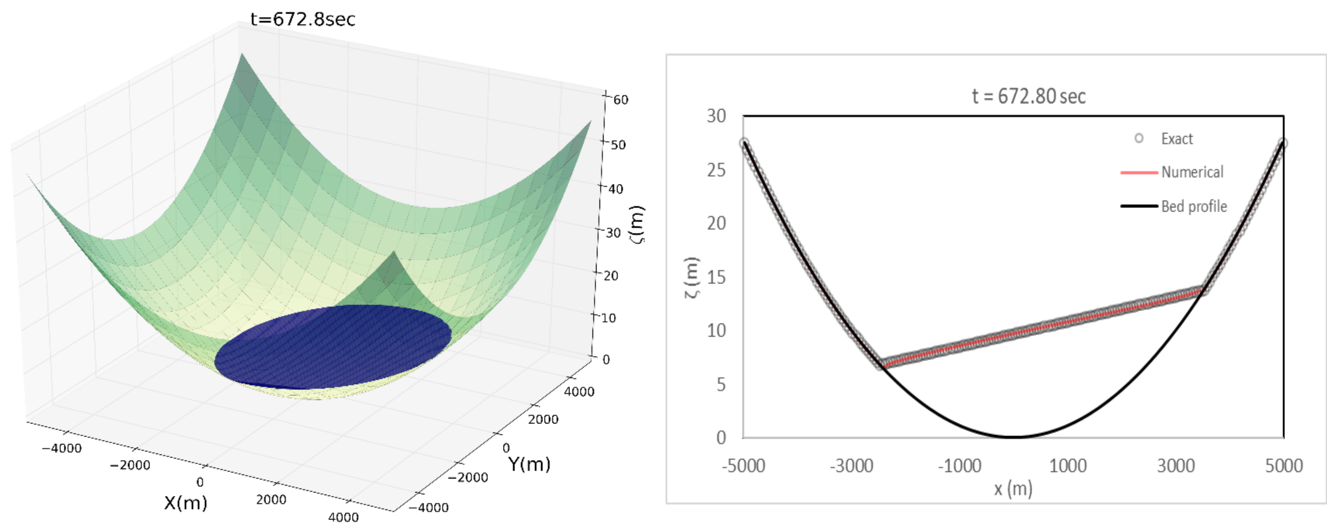
$$\zeta(x,y,t) = h_0 - \frac{B^2 e^{-t\tau}}{2g} - \frac{Be^{-0.5t\tau}}{g}[(0.5\tau\sin st + s\cos st)x + (0.5\tau\cos st - s\sin st)y] \quad (1.14)$$

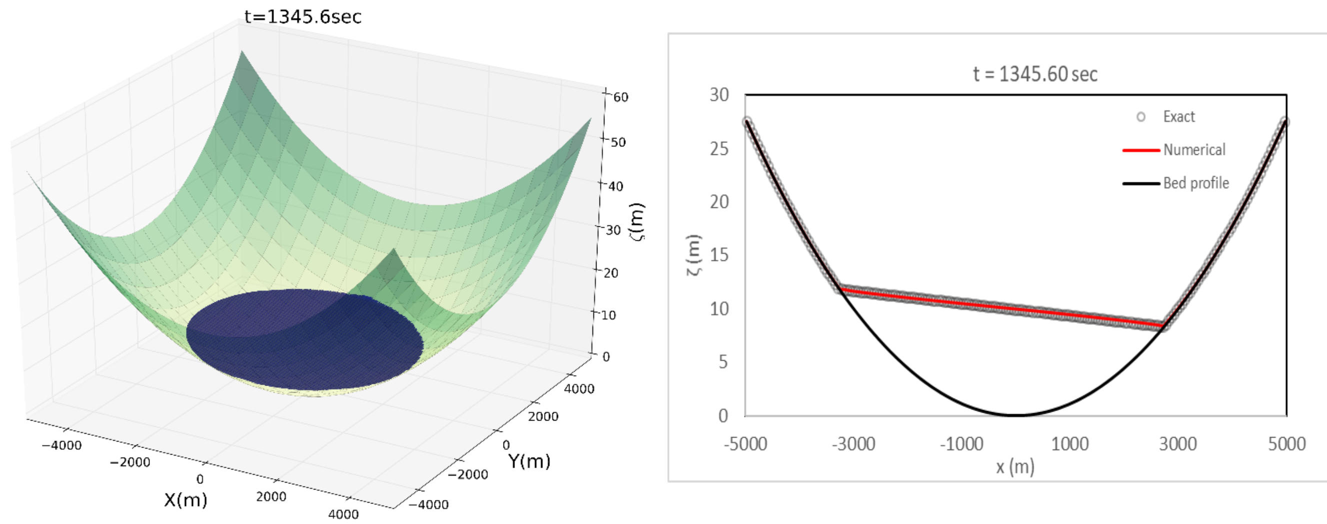
$$V_x(t) = Be^{-0.5t\tau}\sin st \quad (1.15)$$

$$V_y(t) = -Be^{-0.5t\tau}\cos st \quad (1.16)$$

Where $s = 0.5\sqrt{p^2 - \tau^2}$ and the chosen values for the constants $B\ and\ \tau$ are: $B = 5\ ms^{-1}$ and $\tau = 0.002\ s^{-1}$.

The initial conditions for the water depths ($t = 0$) are computed using equation (1.14) and the initial conditions for the velocity components are $V_x(0) = 0\,ms^{-1}$ and $V_y(0) = -5\,ms^{-1}$. The surface profiles at three times ($t_1 = 672.8s, t_2 = 1345.6s\,, t_3 = 5384.2$) along the $x - axis$ at $y = 25m$ are presented in Fig. 7. The computed solution is in very close agreement with the analytical solution and after almost four periods, it converges to a steady state motionless condition. The velocity time series for both components $V_x\ and\ V_y$ at $(x, y) = (1200,25)$ are presented in Fig. 8 and there is good agreement between the analytical and the numerical values.





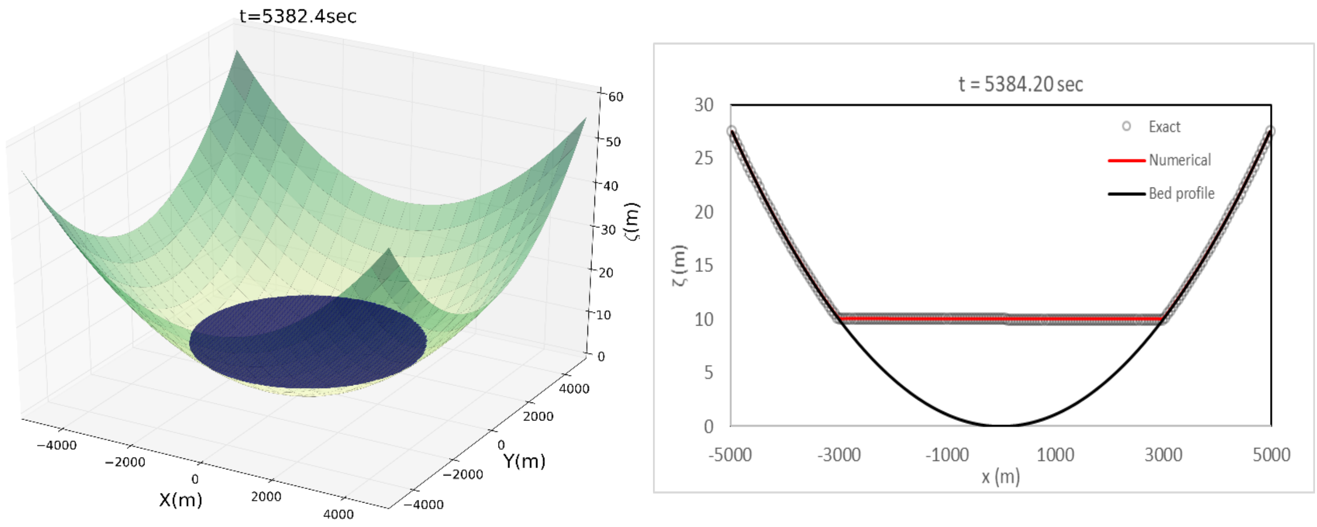
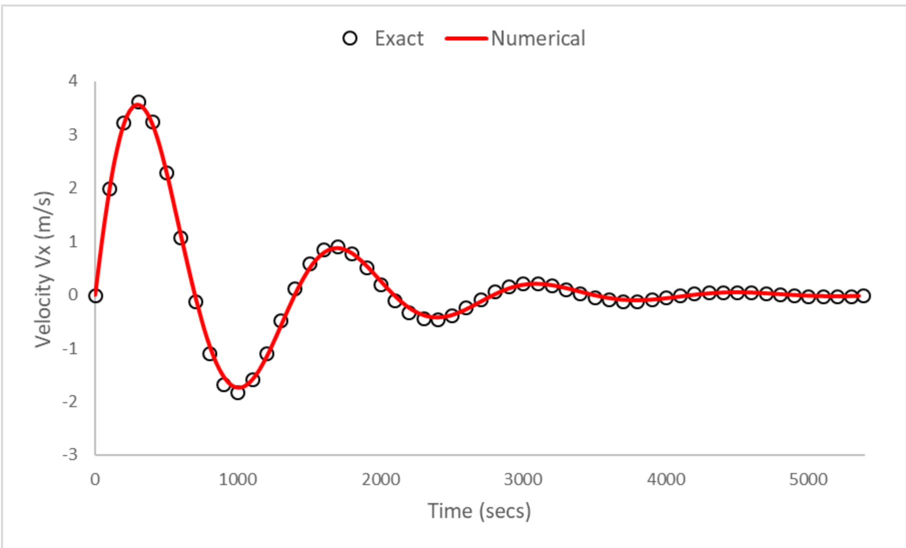


Figure 7. Water surface profiles along the x-axis at y=25m



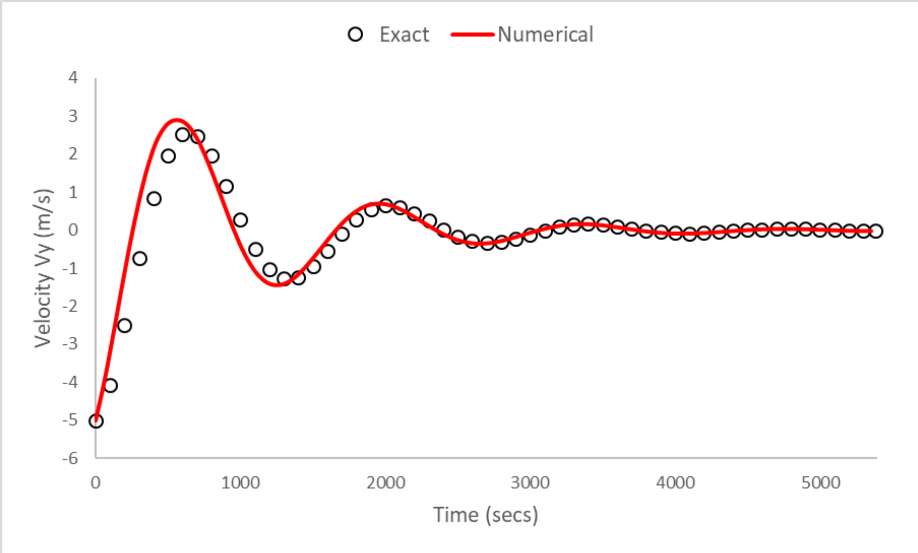


Figure 8. Velocity time series for both components $\boldsymbol{V_x}$ ***and*** $\boldsymbol{V_y}$ at $(\boldsymbol{x}, \boldsymbol{y}) = (\mathbf{1200}, \mathbf{25})$

6.2 Case 2 - Shock propagation and flow around obstacles

This validation case, originates from the physical model developed at the Civil Engineering Laboratory of the Université Catholique de Louvain (Soares-Frazao and Zech, 2007). Measurements from the laboratory experiment supplied with the paper are used for validation of the modelling results here. The study involves a simple topography, a dam with a 1m wide opening, and an idealised representation of a single building downstream of the dam, see Fig. 9. Upstream from the dam the initial water depth is 0.4m and downstream is dry. The flow is contained by vertical walls at the boundaries of the domain. This case has previously been used in a benchmarking study carried out on behalf of the Environment Agency for England and Wales (Néelz and Pender, 2010; Néelz and Pender, 2013) where it is referred to as Test 6A. This is the only case in these studies which is based on real data, thus supporting validation, rather than hypothetical cases where only inter-model comparisons (i.e. benchmarking) can be achieved. This demanding case is increasingly used for testing new numerical schemes and has been selected to test the performance of CityCAT in modelling of dam-break flow conditions (i.e. shock-capturing) and reproduction of trans-critical flow patterns around buildings. . This capability is not only crucial for flood modelling in cities, but is also increasingly important as statutory obligations now exist in many countries for dam operators to publish reservoir flood-risk maps.

The initial conditions and input data of the model are:

- Initial depth: to the left of the gate 0.4m and to the right of the gate 0.00m

- All boundaries closed
- Manning coefficient n=0.01 (uniform)
- Model grid resolution 0.05m (144000 cells)

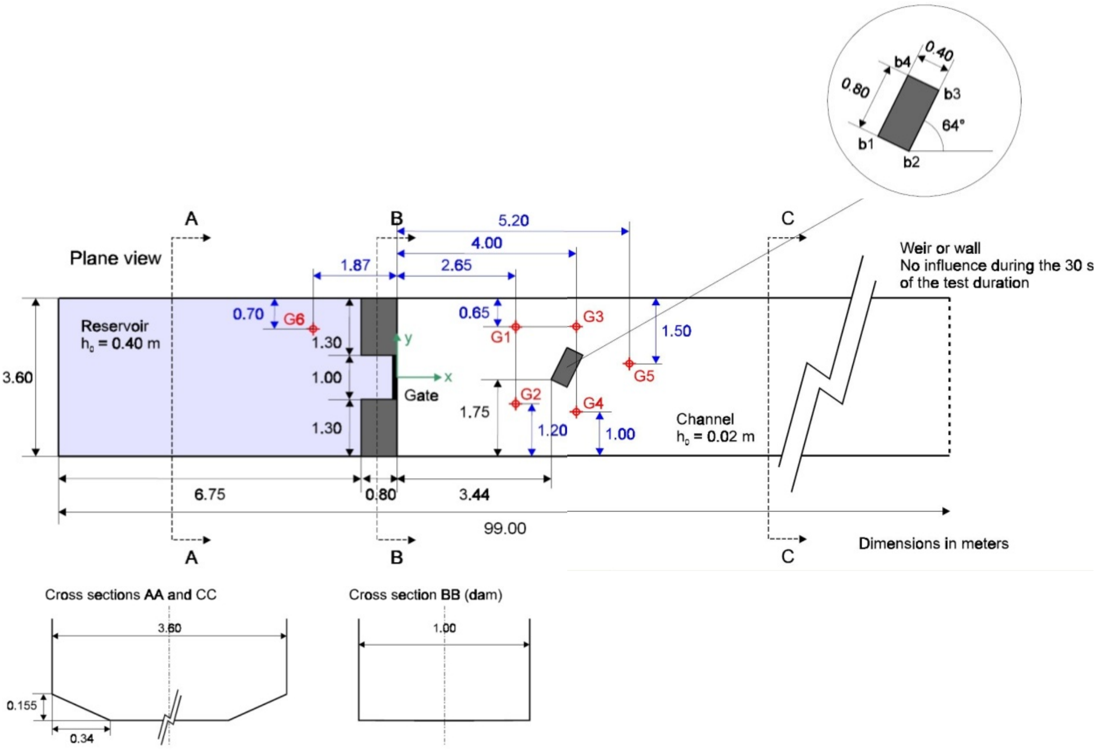
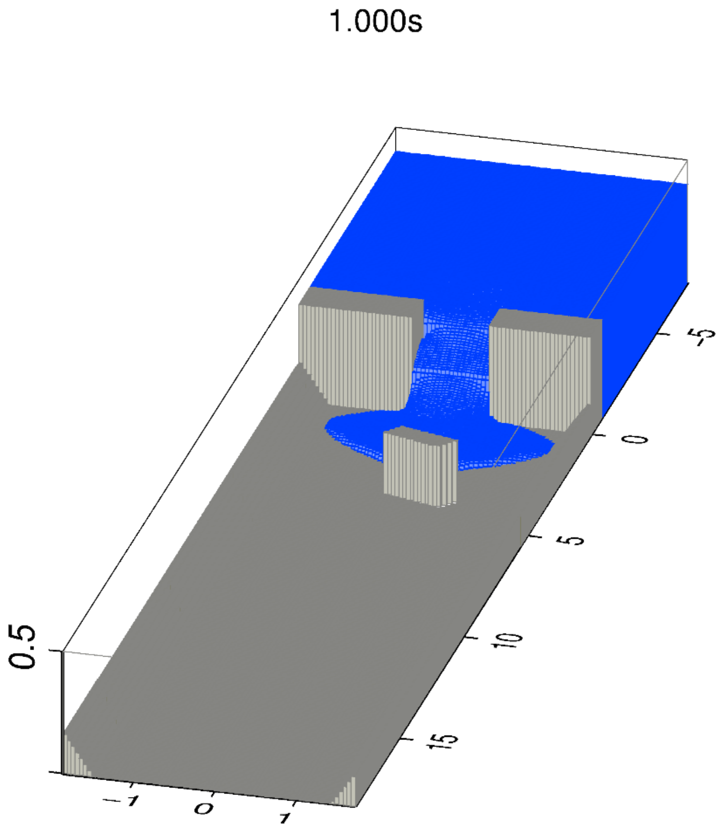
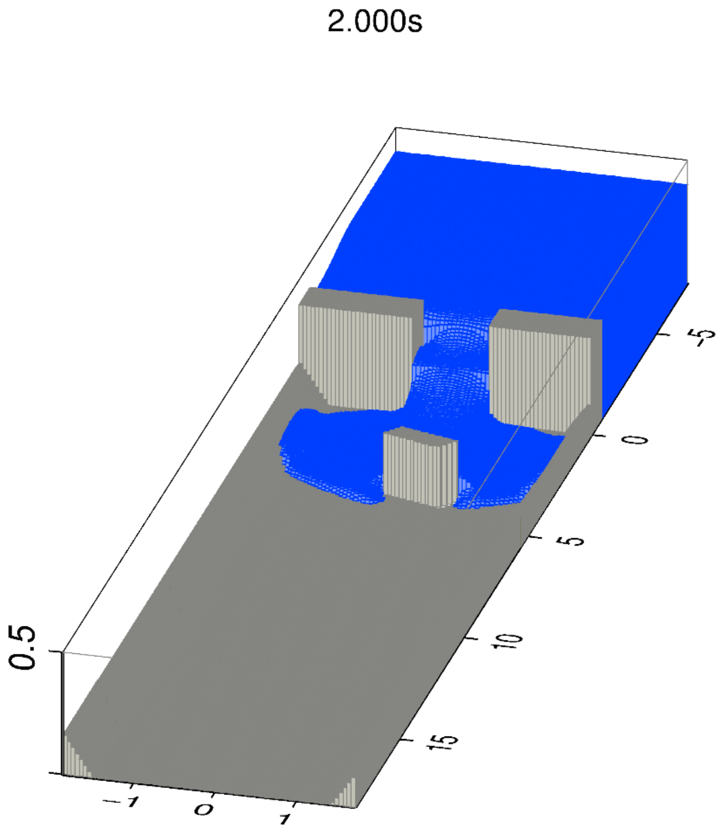


Fig. 9 - Set-up for Example 2. (From Néelz and Pender (2013)).

In Fig. 10, a sequence of 3D plots of water depths obtained by CityCAT is presented. The plots cover a duration of the first three seconds and they clearly show the expected pattern of dam-break wave propagation and flow around an obstacle.


1.000s
0.5
-5
0
5
10
15
-1
0
1


2.000s
-5
0
5
10
15
0.5
-1
0
1

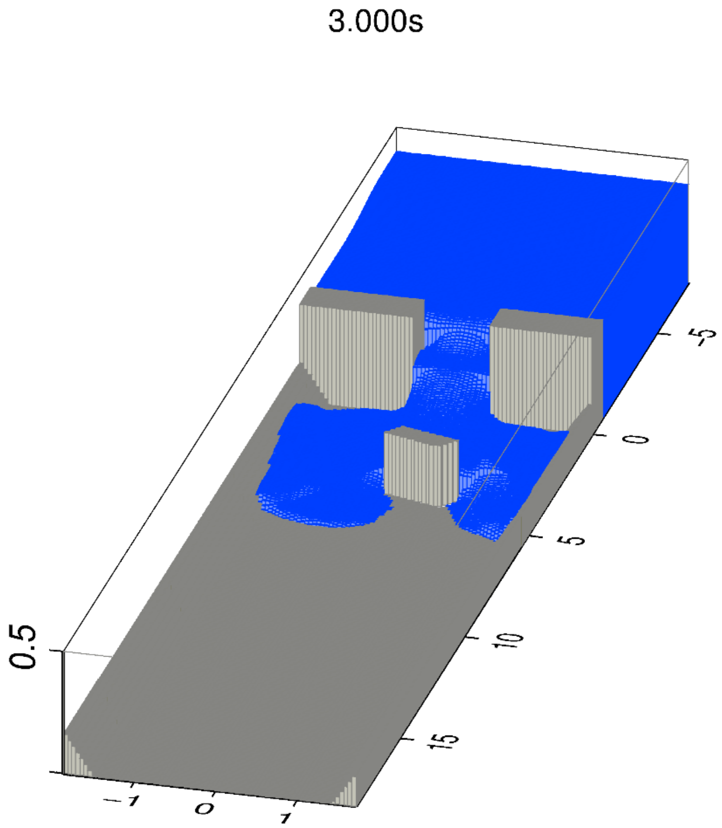
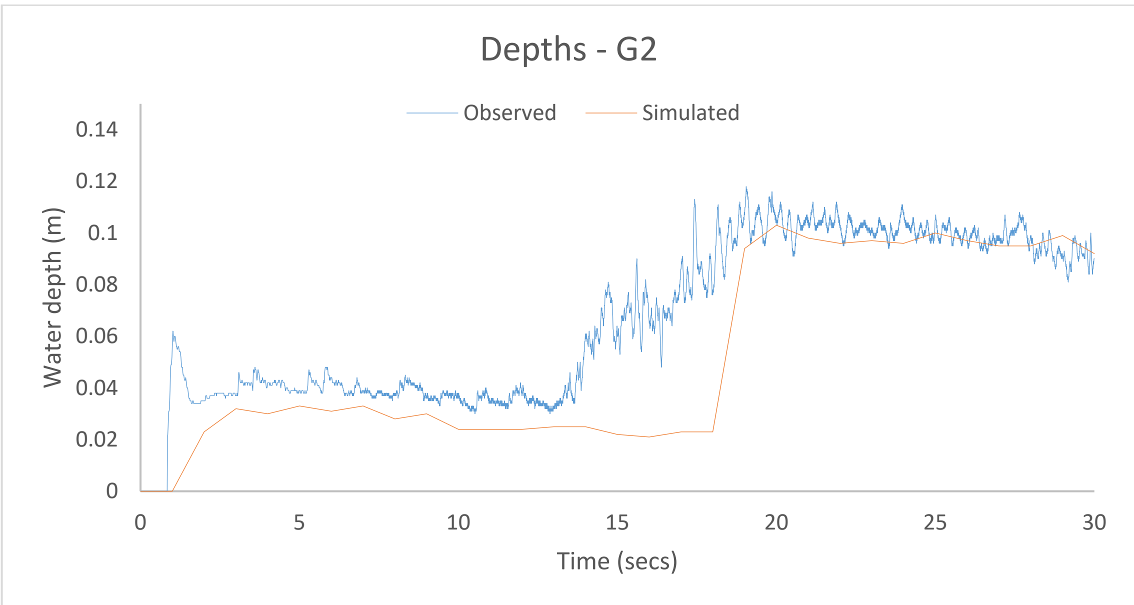
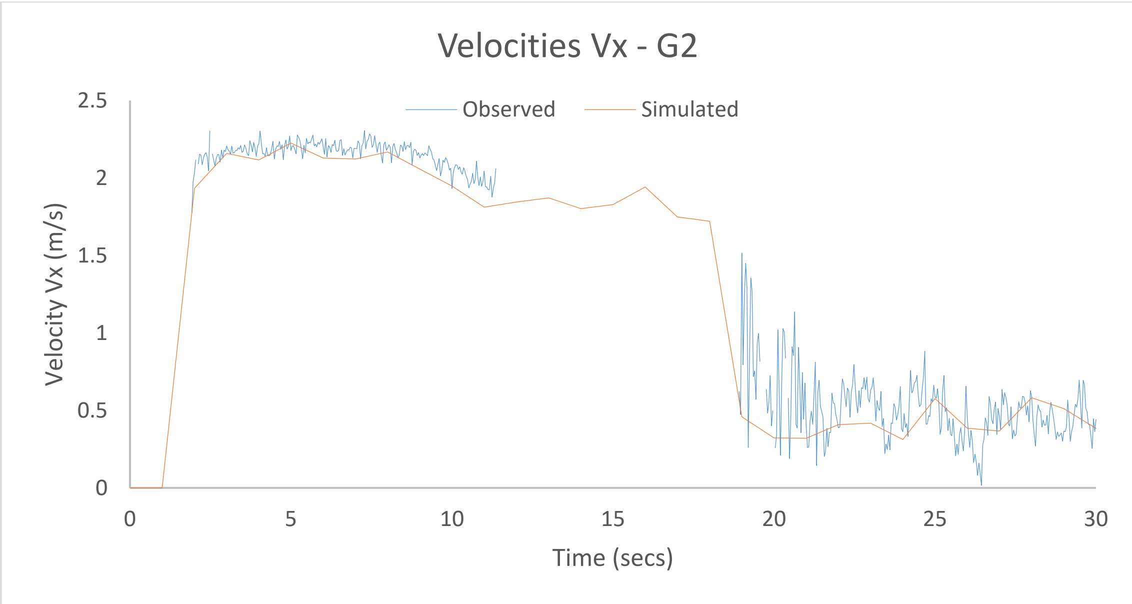
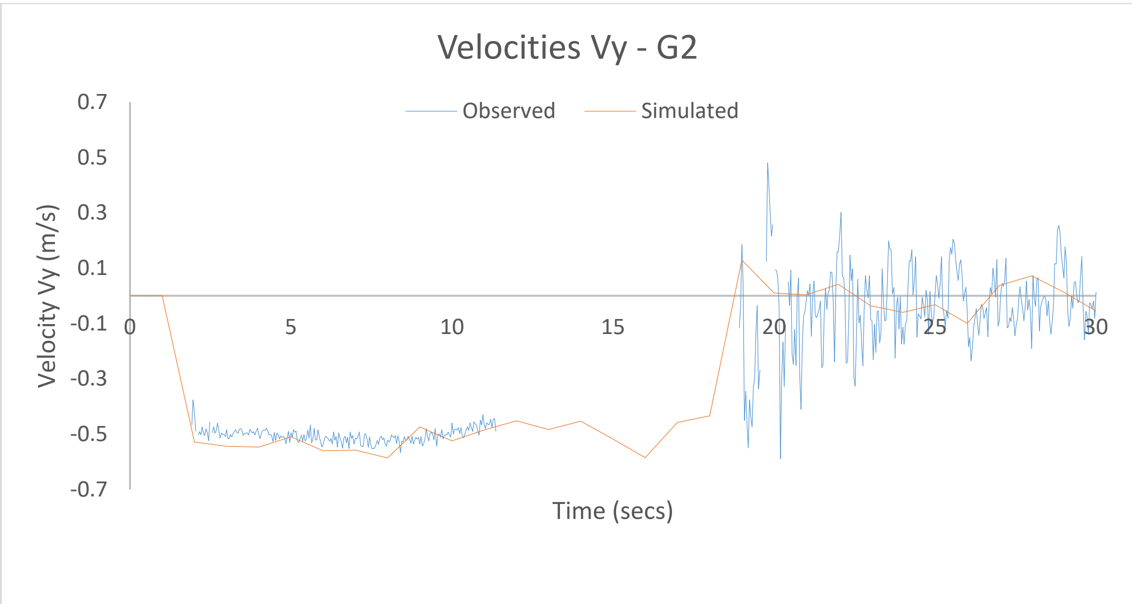
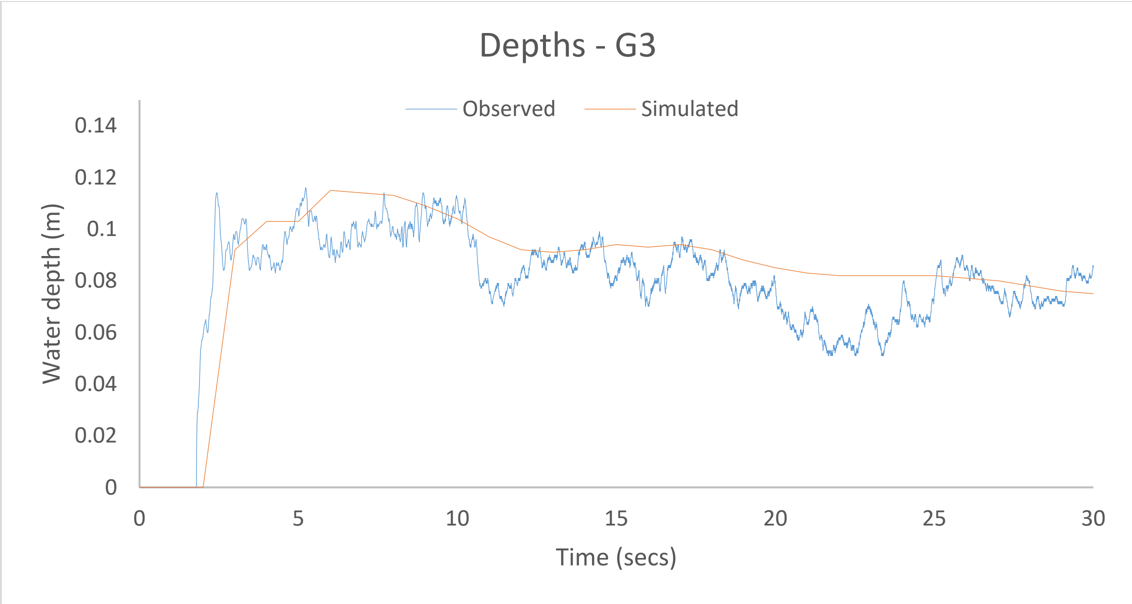


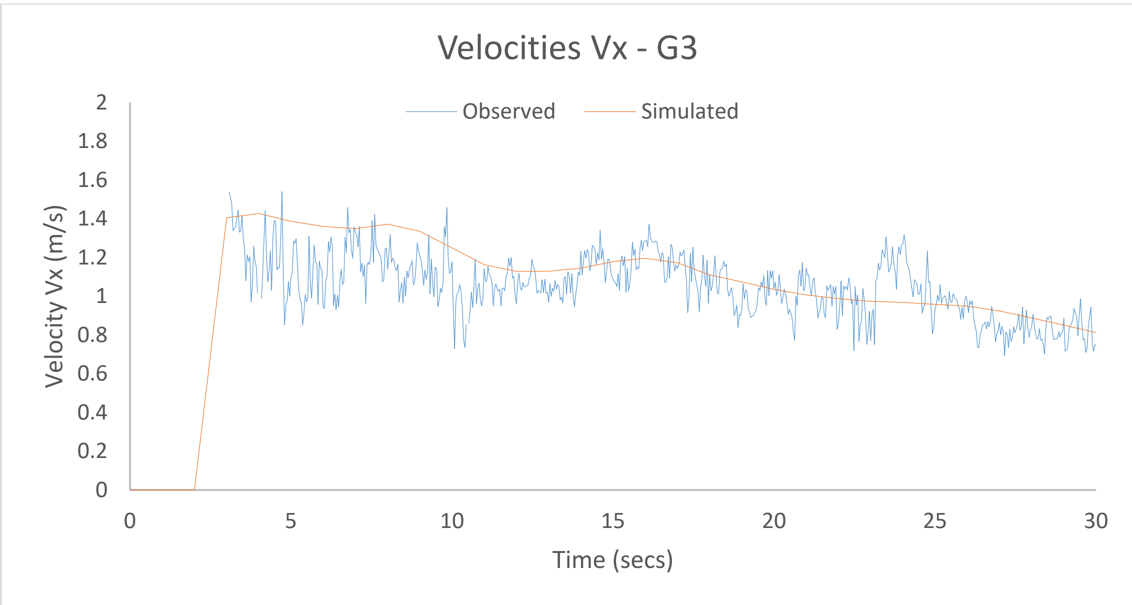
Fig. 10 3D plots showing water depths following the dam break




Velocities Vx - G2
Observed
Simulated
Velocity Vx (m/s)
2.5
2
1.5
1
0.5
0
0
5
10
15
20
25
30
Time (secs)


Velocities Vy - G2
Observed
Simulated
Velocity Vy (m/s)
0.7
0.5
0.3
0.1
-0.1
-0.3
-0.5
-0.7
0
5
10
15
20
25
30
Time (secs)


Depths - G3
Observed
Simulated
Water depth (m)
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
5
10
15
20
25
30
Time (secs)


Velocities Vx - G3
Observed
Simulated
Velocity Vx (m/s)
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
25
30
Time (secs)

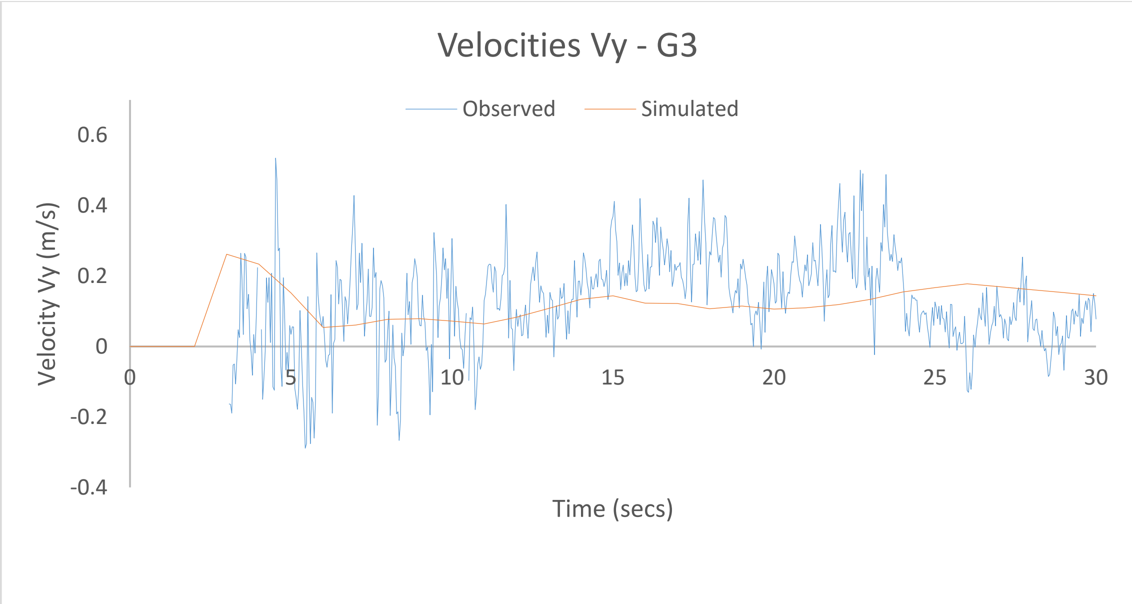


Fig. 11. Comparison of measured and simulated water depths and velocities at points G2 and G3

Comparison of the simulated and measured water depths and velocities at points G2 and G3 are presented in Fig. 11. These two points were selected as they are the most challenging to model (Néelz and Pender, 2013). The initial supercritical flow and the hydraulic jump at point G2 are captured well by the model. However, the timing of the hydraulic jump was predicted a little later than measured. This is probably due to the resolution and the algorithm used to cut out the building from the numerical grid. The predicted velocities at point G2 in the x and y direction (V_x, V_y) are in good agreement with the measured ones. In addition, the model replicates well the water depths and velocities (V_x, V_y) at point G3.

This example shows that CityCAT can accurately simulate dambreak wave propagation and complex flows around obstacles. This feature is very important in modelling urban environments using the "building hole" approach. The results presented above are clearly superior to the results from other models reported in Néelz and Pender (2013), (Figures 4.25 and 4.26).

6.3 Case 3 - Pluvial Flooding in an Urban Environment

In order to test the performance of the CityCAT in a real urban environment, a model was set up for the city centre of Newcastle upon Tyne, UK. The area of the domain is $4 km^2$, the DEM resolution is 1m and the number of cells is 4,000,000. The buildings and the permeable/impermeable surfaces were extracted from MasterMap, see Fig. 10. A 30-minute duration rainfall event of 31.1 mm depth corresponding to the 100 year event (or 1% Annual Exceedance Probability) with a summer rainfall profile following the FEH procedure (Hydrology, 1999) was applied as a uniform input over the whole domain (see Fig. 13). The Manning's coefficient was set to 0.02 for the impermeable surfaces and 0.035 for the permeable surfaces.

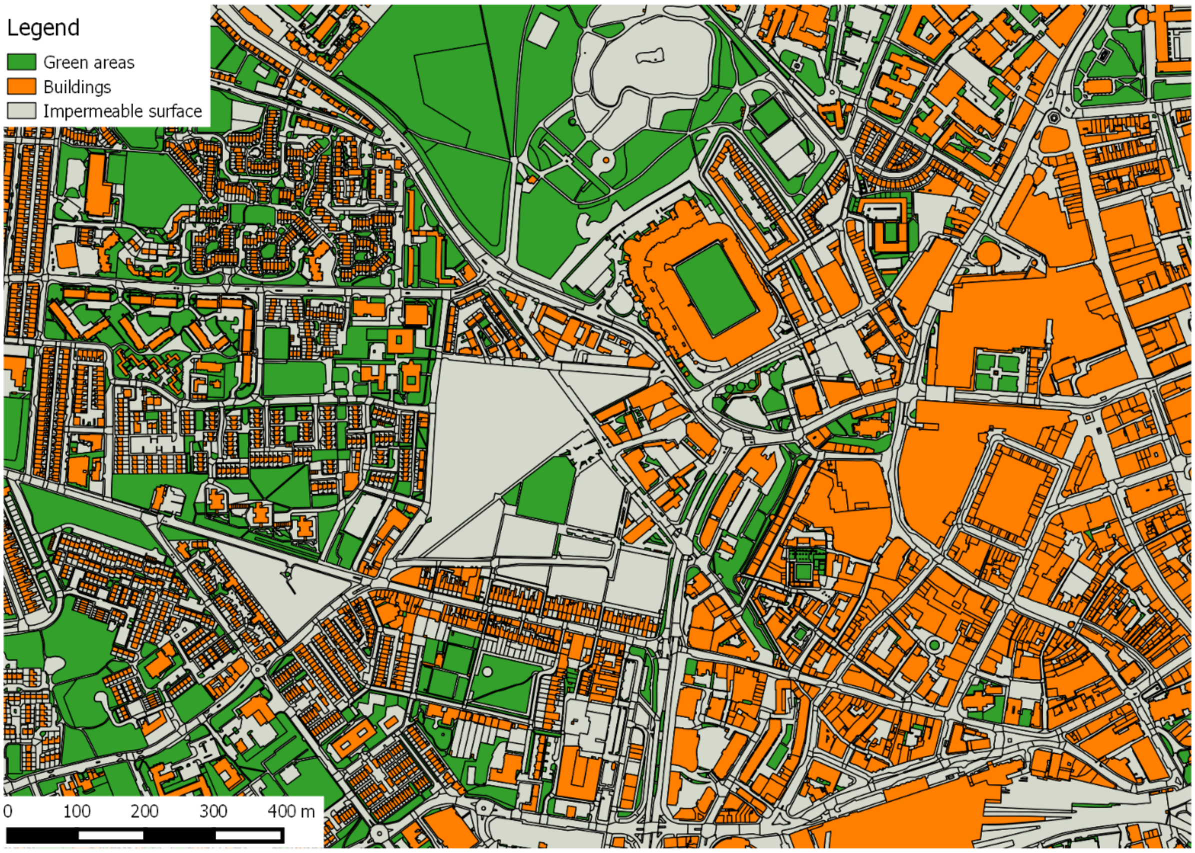


Fig. 12 Mastermap® data for a part of Newcastle upon Tyne city centre

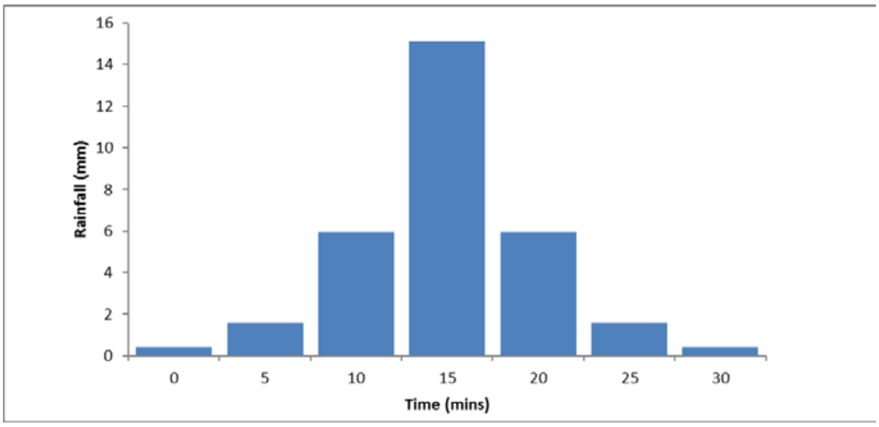


Fig. 13. Storm profile corresponding to a storm event of 30 minutes duration and 100 year return period

A water depth map at the end of the 30-minute simulation is shown in Fig. 14. The dark grey areas represent the buildings' footprint and the light grey areas are the dry areas. The use of $1m^2$ cells enabled realistic representation of the buildings' footprint and other features that influence the flow paths. The use of larger cells would have reduced the number of cells and the size of the model but this may cause blockages between buildings when they are separated by narrow alleyways. It should be noted that when larger cells are used then algorithms B or C might be more suitable for the generation of the numerical grid.

Fig. 14. Water depths over the whole modelled domain of $4km^2$ at the end of a 30 minutes rainfall event with 100 years return period - current situation

The snapshot of water depths presented in Fig. 14 clearly identifies the flow paths which are very much influenced by the topography and the buildings. It is possible to identify dual carriageway roads and this shows that CityCAT is capable of modelling the influence of raised kerbs or other flow diverting measures provided a sufficiently detailed DEM is used. Another feature that can be observed at various locations in Fig. 14 is that water is trapped behind buildings where local topography directs the runoff towards a building. This is captured very well using the building hole approach.

A more detailed water depth map at a particular area of the domain (Newgate Street and the surrounding area) is shown in Fig. 14 where it can be clearly identified how a building placed across a major natural flow path, creates a flooding hotspot. The photograph shown in Fig. 15 was taken at that location during the extreme rainfall event in Newcastle on 28.06.2012.

Fig. 15 Photograph from the Newgate Street, Newcastle during the flood on 28.6.2012 (courtesy of Newcastle City Council)

Apart from the current configuration, three additional hypothetical scenarios have been modelled: 1) current configuration (Fig. 16); 2) all the surfaces are impermeable (Fig. 17); 3) all the surfaces are permeable (Fig. 18); and 4) current configuration with roof storage of 3 cm on all buildings (Fig. 19). While neither of these three hypothetical cases is realistic, they serve to show the model's capabilities and illustrate how such changes would influence the extent of flooding, the water depths and the velocities in a pluvial event.

In Fig 16, representing the current situation, it can be observed that at the end of the 30 minutes rainfall event of 100 years return period, the water depth at one particularly low spot reaches a depth of around 2.0 metres. In the hypothetical scenario where all the surfaces are impermeable the water depths and the velocities are higher, see Fig. 17. The differences are more significant in the hypothetical scenario where all the surfaces are permeable, see Fig. 18. The maximum depth is around 1m and the velocities are considerably smaller. In the last hypothetical scenario where roof storage of 3cm is added to every building in the domain (Fig. 19) the reduction of water depths is significant and the velocities are also smaller.

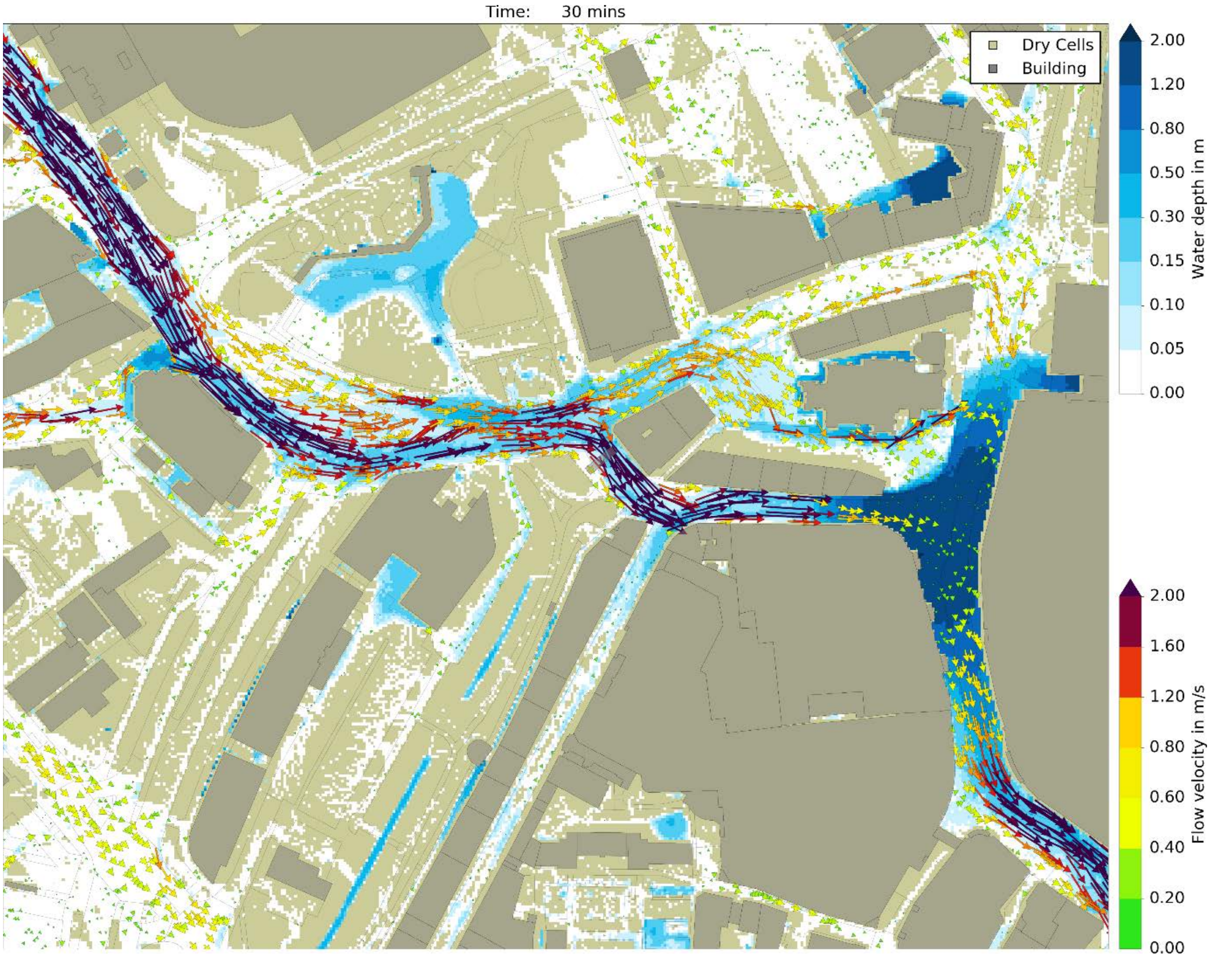


Fig. 16. Water depths and velocities in central Newcastle upon Tyne at the end of the 30 minutes rain event with 100 years return period - current configuration.

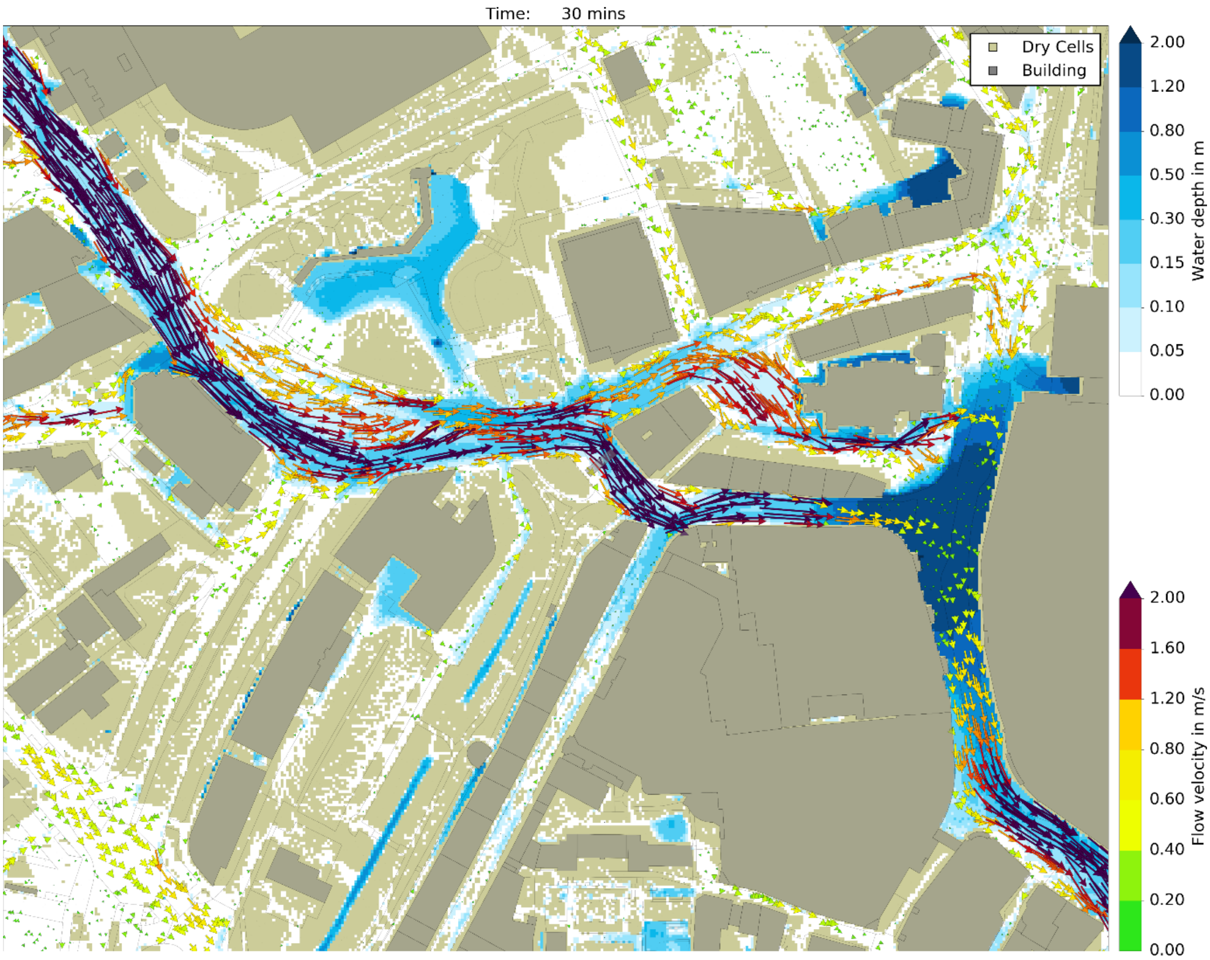
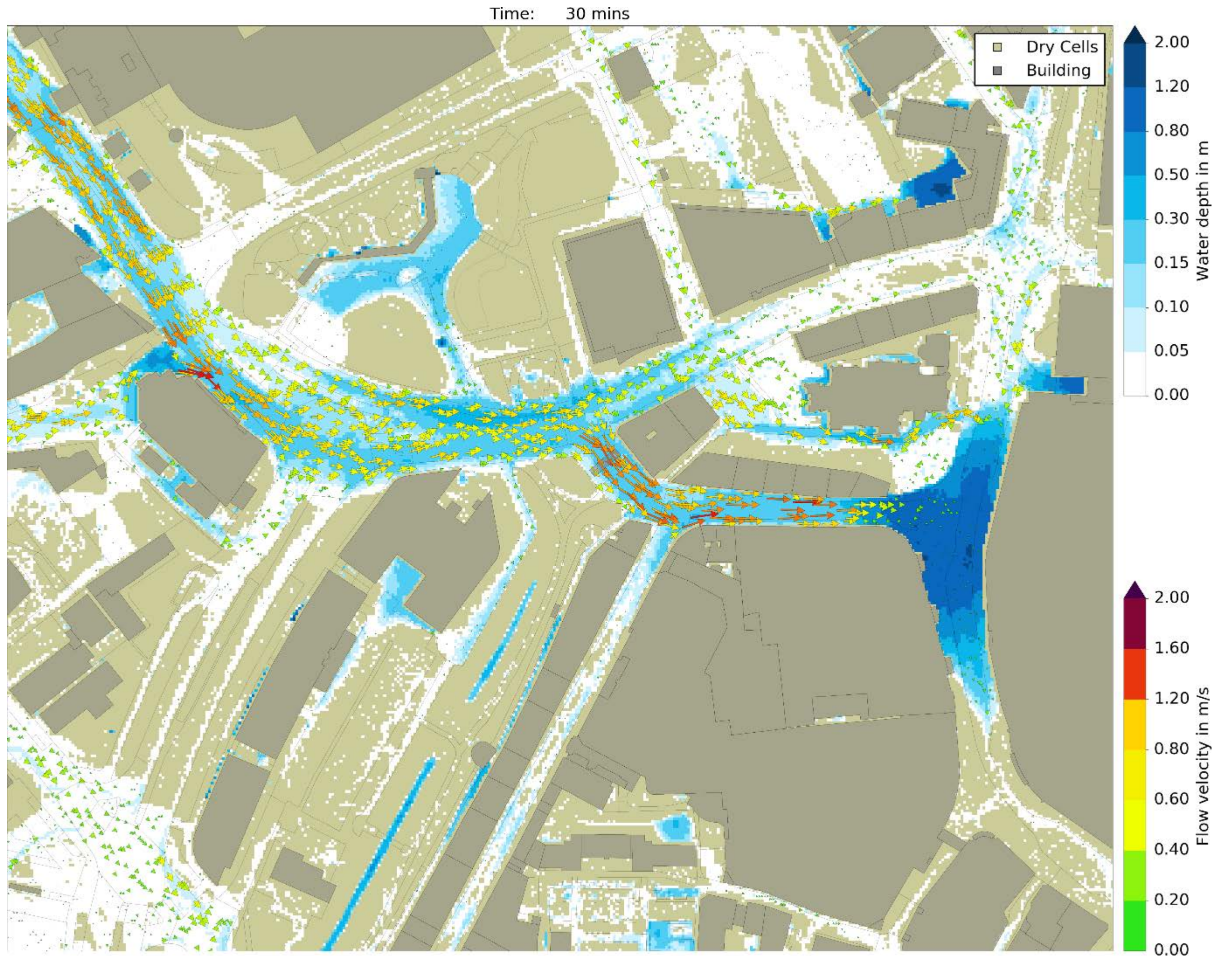


Fig. 17. Water depths and velocities in central Newcastle upon Tyne at the end of the 30 minutes rain event with 100 years return period – hypothetical scenario: all surfaces impermeable.

947 

948 **Fig. 18.** Water depths and velocities in central Newcastle upon Tyne at the end of the 30 minutes rain
949 event with 100 years return period – hypothetical scenario: all surfaces permeable.

950

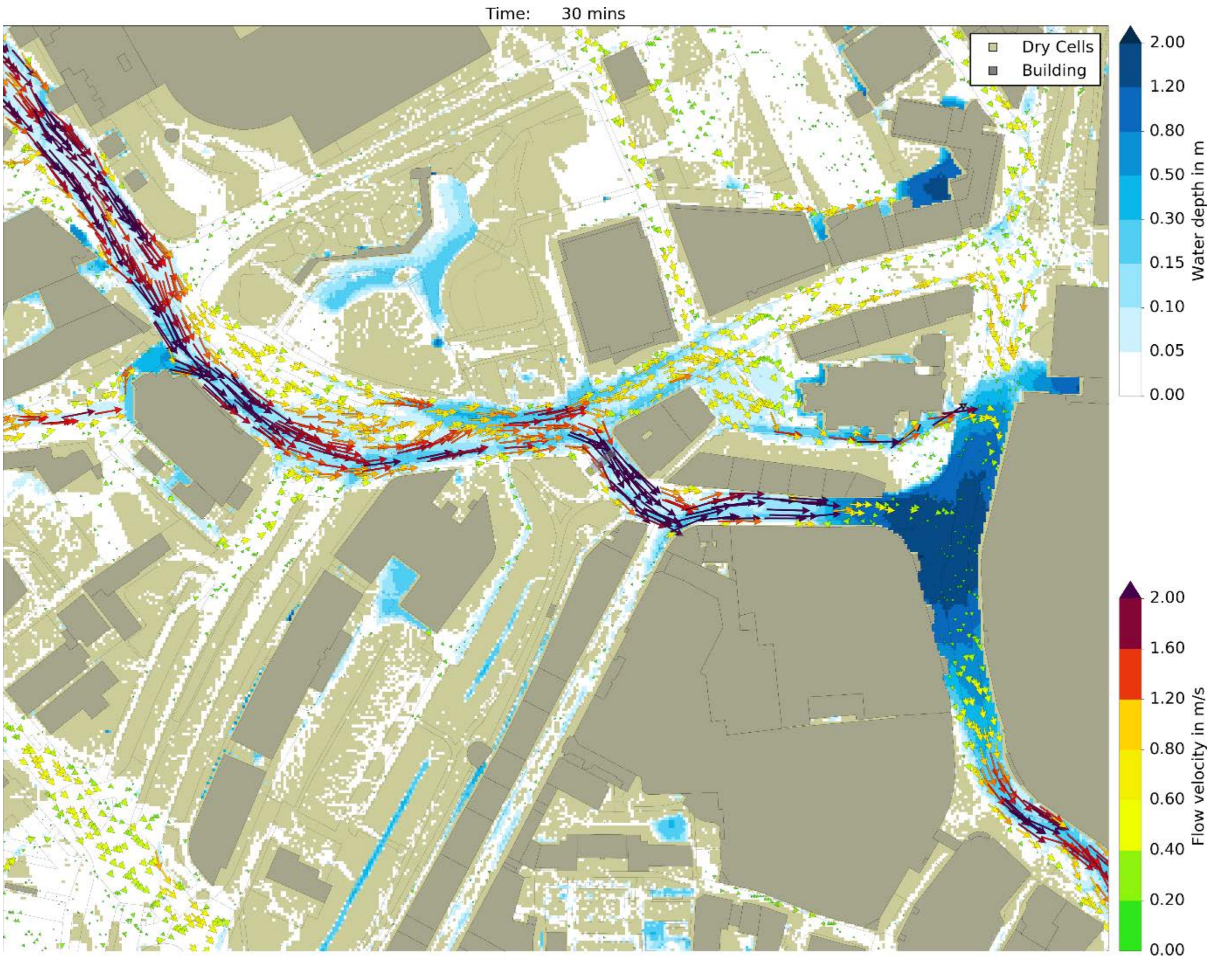


Fig. 19. Water depths and velocities in central Newcastle upon Tyne at the end of the 30 minutes rain event with 100 years return period – hypothetical scenario: current configuration with roof storage of 3cm on all the buildings in the domain.

This example shows the ability of CityCAT to model pluvial flood events over high resolution urban domains. Furthermore, it demonstrates the first use of a hydrodynamic model, resolving individual features and buildings, to assess the effect of specific interventions across a whole city domain.

7 Conclusions

CityCAT is a novel and unique software package in the field of flood modelling as it combines accurate numerical methods with advanced software architecture providing rapid and flexible set up without compromising accuracy. Combination of those two main properties results in a versatile package able to model complex flow situations such as propagation of shocks and flows over initially dry areas as well as to efficiently simulate flash floods over large urban domains generated using standard data sets, additionally allowing alternative scenarios of urban fabric and green urban infrastructure to be efficiently trialled.

The examples presented in this paper rigorously validate and illustrate CityCAT's capabilities. . Comparison with analytical solutions for moving-boundary shallow water flow in a parabolic bowl with friction assesses the performance of the numerical solutions in tracking wet/dry interfaces. Comparison with results from a laboratory experiment validates its ability to model dam-break situations with propagation of shocks around obstacles. The final example demonstrates its ability to model pluvial flood over extended urban areas and assess the influence of potential design interventions on local and large area urban flood risk.

The efficiency at overall code and algorithm level also provides significant speed up enabling very large domains to be simulated at unprecedented resolution. The object oriented approach to numerics offers great advantages in the development of numerical code as the fully modular approach allows rapid extension of functionality, through implementation of changes to appropriate computational objects and avoidance of "if-then-else" statements improves computational efficiency.

Furthermore, the separation of buildings from the flow domain, and their treatment as computational objects, allows for the first time the possibility of varying their permeability and storage attributes. This then leads to a new era of urban drainage design with the exciting prospect of using a fully specified and accurate hydrodynamic code in "design" mode, where multiple options for flood adaptation features such as roof storage, surface flow routeing and permeable surfaces can be assessed.

8 Authors' contribution

V.G, V.K., C.G.K, designed the research. V.G. coded and developed the model and performed the research. V.G and C.G.K. wrote the paper.

9 Acknowledgements

The CityCAT software has been developed with support by a number of funders: the Environment Agency's Local Levy (raised by the Northumbria Regional Flood Defence Committee) , the JISC/EPSRC project "Flood Modelling for Cities using Cloud Computing" (EP/I034351/1) which was one of the Pilot projects in cloud computing for research, and the Blue Green Cities project (EPSRC Grant EP/K013661/1). We gratefully acknowledge Greg O'Donnell and Robert Bertsch of Newcastle University who helped with providing figures 10,12,14,16,17,18,19.

10 References

Alcrudo, F., 2004. Mathematical modelling techniques for flood propagation in urban areas. Project report: IMPACT Project.

Alcrudo, F., Garcia-Navarro, P., 1993. A High-resolution Godunov-type Scheme in Finite Volumes for the 2D Shallow-water Equations. International Journal for Numerical Methods in Fluids 16(6) 489-505.

Allitt, R., Blanksby, J., Djordjevic, S., Maksimovic, C., Stewart, D., 2009. Investigations into 1D-1D and 1D-2D urban flood modelling, WaPUG Autumn Conference.

Ata, R., Pavan, S., Khelladi, S., Toro, E.F., 2013. A Weighted Average Flux (WAF) scheme applied to shallow water equations for real-life applications. Advances In Water Resources 62 155-172.

Audusse, E., Bouchut, F., Bristeau, M.-O., Klein, R., Perthame, B., 2004. A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows. Siam Journal On Scientific Computing 25(6) 2050-2065.

Audusse, E., Bristeau, M.-O., 2005. A well-balanced positivity preserving “second-order” scheme for shallow water flows on unstructured meshes. Journal of Computational Physics 206(1) 311-333.

Bach, P.M., Rauch, W., Mikkelsen, P.S., Mccarthy, D.T., Deletic, A., 2014. A critical review of integrated urban water modelling–Urban drainage and beyond. Environmental Modelling & Software 54 88-107.

Bermúdez, A., Vázquez-Cendón, M., 1994. Upwind Methods for Hyperbolic Conservation Laws with Source Terms. Computers Fluids 23-28.

Bertsch, R., Glenis, V., Kilsby, C., 2017. Urban Flood Simulation Using Synthetic Storm Drain Networks. Water 9(12) 925.

Brufau, P., Garcia-Navarro, P., Vazquez-Cendon, M.E., 2004. Zero mass error using unsteady wetting-drying conditions in shallow flows over dry irregular topography. International Journal for Numerical Methods in Fluids 45 1047-1082.

Castro Díaz, M., López-García, J.A., Parés, C., 2013. High order exactly well-balanced numerical methods for shallow water systems. Journal of Computational Physics 246 242-264.

Castro, M.J., Gallardo, J.M., Marquina, A., 2016. Approximate Osher-Solomon schemes for hyperbolic systems. Applied Mathematics and Computation 272 347-368.

Chow, V.T., Maidment, D.R., Mays, L.W., 1988. Applied hydrology. McGraw-Hill, N. Y.

Costanzo, C., Macchione, F., 2006. Two-dimensional numerical simulation of flood propagation in presence of buildings, International conference on fluvial hydraulics; River flow 2006. London: Lisbon, pp. 291-302.

DOM, Available from: https://www.w3.org/TR/dom/.

Dumbser, M., Toro, E.F., 2011a. On universal Osher-type schemes for general nonlinear hyperbolic conservation laws. Communications in Computational Physics 10(03) 635-671.

Dumbser, M., Toro, E.F., 2011b. A Simple Extension of the Osher Riemann Solver to Non-conservative Hyperbolic Systems. Journal of Scientific Computing 48, Numb 1-3 70-88.

Embarcadero, Delphi. Available from: https://www.embarcadero.com/products/delphi.

Erduran, K.S., Kutija, V., Hewett, C.J.M., 2002. Performance of finite volume solutions to the shallow water equations with shock-capturing schemes. International Journal for Numerical Methods in Fluids 40(10) 1237-1274.

Fernández-Nieto, E.D., Narbona-Reina, G., 2008. Extension of WAF type methods to non-homogeneous shallow water equations with pollutant. Journal of Scientific Computing 36(2) 193-217.

Fraccarollo, L., Toro, E.F., 1995. Experimental and numerical assessment of the shallow water model for two-dimensional dam-break type problems. Journal of Hydraulic Research 33(6) 843-864.

Garcia-Navarro, P., Vazquez-Cendon, M.E., 2000. On numerical treatment of the source terms in the shallow water equations. Computers and Fluids 29(8) 951-979.

Glenis, V., McGough, A.S., Kutija, V., Kilsby, C., Woodman, S., 2013. Flood modelling for cities using Cloud computing. Journal of Cloud Computing: Advances, Systems and Applications 2(1) 1.

Godunov, S.K., 1959. Finite Difference Method for Numerical Computation of Discontinuous Solutions of the Equations of Fluid Dynamics. Matematicheski Sbornik 47 271-306.

Greenberg, J.M., Leroux, A.-Y., 1996. A well-balanced scheme for the numerical processing of source terms in hyperbolic equations. Siam Journal on Numerical Analysis 33(1) 1-16.

Guan, M., Wright, N.G., Sleigh, P.A., 2013. A robust 2D shallow water model for solving flow over complex topography using homogenous flux method. International Journal for Numerical Methods in Fluids 73, Numb 3 225-249.

Guerreiro, S.B., Glenis, V., Dawson, R.J., Kilsby, C., 2017. Pluvial Flooding in European Cities—A Continental Approach to Urban Flood Modelling. Water 9(4) 296.

Hankin, B., Waller, S., Astle, G., Kellagher, R., 2008. Mapping space for water : screening for urban flash flooding. Journal of Flood Risk Management 1, Numb 1 13-22.

Harten, A., 1983. High Resolutions Schemes for Hyperbolic Conservation Laws. Journal of Computational Physics 49 357-393.

Harten, A., Hyman, J.M., 1983. Self adjusting grid methods for one-dimensional hyperbolic conservation laws. Journal of Computational Physics 50(2) 235-269.

Harten, A., Lax, P.D., van Leer, B., 1983. On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws. Siam Review 25 35-61.

Hunter, N.M., Bates, P.D., Neelz, S., Pender, G., Villanueva, I., Wright, N.G., Liang, D., Falconer, R.A., Lin, B., Waller, S., 2008. Benchmarking 2D hydraulic models for urban flooding. Proceedings-Institution of Civil Engineers Water Management 161, Issu 1 13-30.

Hydrology, I.o., 1999. Flood Estimation Handbook, vol 3: Statistical procedures for flood frequency estimation. Institute of Hydrology, Wallingford, UK.

Kim, S.D., Lee, B.J., Lee, H.J., Jeung, I.S., 2009. Robust HLLC Riemann solver with weighted average flux scheme for strong shock. Journal of Computational Physics 228, Numb 20 7634-7642.

Kutija, V., Murray, M.G., 2007. An object-oriented approach to the modelling of free-surface flows. Journal Of Hydroinformatics 9 81-94.

Kutílek, M., Nielsen, D.R., 1994. Soil hydrology. Catena Verlag.

Liu, Q., Qin, Y., Zhang, Y., Li, Z., 2015. A coupled 1D–2D hydrodynamic model for flood simulation in flood detention basin. Natural hazards 75(2) 1303-1325.

Loukili, Y., Soulaimani, A., 2007. Numerical Tracking of Shallow Water Waves by the Unstructured Finite Volume WAF Approximation. International Journal for Computational Methods in Engineering Science and Mechanics 8, Numb 2 75-88.

Mark, O., Weesakul, S., Apirumanekul, C., Aroonnet, S.B., Djordjevic, S., 2004. Potential and limitations of 1D modelling of urban flooding. Journal Of Hydrology 299, Numb 3-4 284-299.

Michel-Dansac, V., Berthon, C., Clain, S., Foucher, F., 2016. A well-balanced scheme for the shallow-water equations with topography. Computers & Mathematics with Applications 72(3) 568-593.

Mignot, E., Paquier, A., Haider, S., 2006. Modeling floods in a dense urban area using 2D shallow water equations. Journal Of Hydrology 327, Numb 1-2 186-199.

Mingham, C.G., Causon, D.M., 1998. High-Resolution Finite Volume Method for Shallow Water Flows. Journal Of Hydraulic Engineering 124 605–614.

Neal, J.C., Bates, P.D., Fewtrell, T.J., Hunter, N.M., Wilson, M.D., Horritt, M.S., 2009. Distributed whole city water level measurements from the Carlisle 2005 urban flood event and comparison with hydraulic model simulations. Journal Of Hydrology 368, Numb 1-4 42-55.

Néelz, S., Pender, G., 2010. Benchmarking of 2D hydraulic modelling packages.

Néelz, S., Pender, G., 2013. Delivering benefits thorough evidences: Benchmarking the Latest Generation of 2D Hydraulic Modelling Packages. Report—SC120002.

Noh, S.J., Lee, J.-H., Lee, S., Kawaike, K., Seo, D.-J., 2018. Hyper-resolution 1D-2D urban flood modelling using LiDAR data and hybrid parallelization. Environmental Modelling & Software 103 131-145.

Osher, S., Solomon, F., 1982. Upwind Difference Schemes for Hyperbolic Conservation Laws. Mathematics Of Computation 38(158) 339-374.

Pitt, M., 2008. Learning lessons from the 2007 floods.

Roe, P.L., 1981. Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes. Journal of Computational Physics 43 357-372.

Sampson, J., Easton, A., Singh, M., 2006. Moving boundary shallow water flow above parabolic bottom topography. Anziam Journal 47 373-387.

Sanders, B.F., Schubert, J.E., Gallegos, H.A., 2008. Integral formulation of shallow-water equations with anisotropic porosity for urban flood modeling. Journal Of Hydrology 362, Numb 1-2 19-38.

SAX, Available from: http://www.saxproject.org/about.html.

Schubert, J.E., Sanders, B.F., Smith, M.J., Wright, N.G., 2008. Unstructured mesh generation and landcover-based resistance for hydrodynamic modeling of urban flooding. Advances In Water Resources 31, Numb 12 1603-1621.

Soares-Frazao, S., Lhomme, J., Guinot, V., Zech, Y., 2008. Two-dimensional shallow-water model with porosity for urban flood modelling. Journal of Hydraulic Research 46, Numb 1 45-64.

Soares-Frazao, S., Zech, Y., 2007. Experimental study of dam-break flow against an isolated obstacle. JOURNAL OF HYDRAULIC RESEARCH : Special Issue: Dam-Break Flow Experiments and Real-Case Data. A Database from the European IMPACT Research Program 45, Supp 1 27-36.

Sweby, P.K., 1984. High resolution schemes using flux limiters for hyperbolic conservation laws. Siam Journal on Numerical Analysis 21(5) 995-1011.

Syme, W., 2008. Flooding in urban areas-2D modelling approaches for buildings and fences, 9th National Conference on Hydraulics in Water Engineering: Hydraulics 2008. Engineers Australia, p. 25.

Tan, W.Y., 1992. Shallow Water Hydrodynamics: Mathematical Theory and Numerical Solution for a Two-dimensional System of Shallow-water Equations. Elsevier Science.

Teng, J., Jakeman, A., Vaze, J., Croke, B.F., Dutta, D., Kim, S., 2017. Flood inundation modelling: A review of methods, recent advances and uncertainty analysis. Environmental Modelling & Software 90 201-216.

Thacker, W.C., 1981. Some exact solutions to the nonlinear shallow-water wave equations. Journal Of Fluid Mechanics 107 499-508.

Toro, E.F., 1989. A Weighted Average Flux Method for Hyperbolic Conservation Laws. Proceedings-Royal Society of London A 423 401-418.

Toro, E.F., 1992. Riemann problems and the WAF method for solving the two-dimensional shallow water equations. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 338(1649) 43-68.

Toro, E.F., 2013. Riemann solvers and numerical methods for fluid dynamics: a practical introduction. Springer Science & Business Media.

Toro, E.F., Spruce, M., Speares, W., 1994. Restoration of the Contact Surface in the HLL-Riemann Solver. Shock Waves 4 25-34.

Vazquez-Cendon, M.E., 1999. Improved Treatment of Source Terms in Upwind Schemes for the Shallow Water Equations in Channels with Irregular Geometry. Journal of Computational Physics 148(2) 497-526.

Warrick, A.W., 2003. Soil water dynamics. Oxford University Press.