Songklanakarin J. Sci. Technol.
44 (1), 217-222, Jan. - Feb. 2022

http://www.sjst.psu.ac.th

*Original Article*

# Simulation and visualization of rainfall overland flow based on diffusion equation

Anurak Busaman[1, 2], Somporn Chuai-Aree[1, 2], and Rhysa McNeil[1, 2*]

[1] *Department of Mathematics and Computer Science, Faculty of Science and Technology, Prince of Songkla University, Pattani Campus, Muang, Pattani, 94000 Thailand*

[2] *Centre of Excellence in Mathematics, Commission on Higher Education (CHE), Ratchathewi, Bangkok, 10400 Thailand*

Received: 25 April 2020; Revised: 9 June 2021; Accepted: 1 July 2021

## Abstract

Numerous applications in virtual reality and computer animation need tools including efficient algorithms, for modeling and simulation of water overland flow. This paper proposes an algorithm and a model for simulation and visualization of rainwater overland flow. We use the finite difference method associated with a dynamic domain to solve the diffusion equations in order to reduce the computation time. The simulation results show the water propagation in rugged terrains. Moreover, the results indicate that our algorithm is highly efficient because it reduces the computation time. The diffusion model and a numerical method were applied in this work for the simulation of water flooding caused by continuous heavy rain in Nakhon Si Thammarat province, southern Thailand. Results show that the flood plain obtained by the diffusion model is similar to the actual flooding area from the real satellite image. This indicates that the diffusion model has high potential, and can be adopted for predictive use in flood risk assessments, water resources management, and disaster prevention from water flooding.

**Keywords**: simulation, visualization, finite difference method, rainfall overland flow, diffusion equation

## 1. Introduction

Modelling, simulation and visualization of water overland flow are important tools for numerous applications in virtual reality and computer animation, such as flood risk assessments, water resources management, and disaster prevention from water flooding. Most water overland flow simulations have involved numerical methods for solving the shallow water equations (Fiedler & Ramirez, 2000). However, several water overland flow models (Alsdorf, Dunne, Melack, Smith & Hess, 2005; Benes & Forsbach, 2001; Dottori &Todini, 2011; Santillana, 2008; Wang, 2011) have been developed and successfully applied using simpler equations with reduced complexity. The motivation of the flood models is that solving the simpler equations should reduce the computational burden and the simulation run times (Dottori & Todini, 2011). Diffusion and wave equations have been used for description of water and wave propagation (Alsdorf *et al.*, 2005; Benes, 2007; Chuai-Aree & Kanbua, 2007; Santillan, 2008). However, both equations were not used to describe the water and wave propagation in rugged terrains.

Nowadays, the developed geographic information systems (GIS) are useful for flood related modeling. A weather radar image is one form of GIS that can show rainfall by area and intensity, and can be used as input data of rainfall rate for modeling water flooding caused by continuous heavy rains.

This paper proposes a novel method for modeling, simulation, and visualization of water overland flow based on the diffusion equations, which require reduced computation time. The method integrates the segmentation of weather radar with the model to simulate the water flooding caused by continuous heavy rain.

## 2. Model Equations

The initial value problem for the diffusion equation for modeling water overland flow caused by

*Corresponding author
Email address: rhysa.m@psu.ac.th

continuous heavy rain is defined as follows:

$$\frac{\partial H}{\partial t} = C\left(\frac{\partial^2 H}{\partial x^2} + \frac{\partial^2 H}{\partial y^2}\right),\ x, y \in \Omega \qquad (1)$$

where $\Omega$ is the areal domain of the simulation. $H(x,y,t)$ is the water level (the sum of the topographic elevation and the depth of water); $x$ and $y$ are grid indices in the longitudinal and latitudinal directions, respectively; $C$ is water propagation speed; and $t$ is time.

In order to allow the model to simulate the water flood caused by continuous heavy rain, we added a term for the rainfall rate to obtain the model of rainwater overland flow, as follows:

$$\frac{\partial H}{\partial t} = C\left(\frac{\partial^2 H}{\partial x^2} + \frac{\partial^2 H}{\partial y^2}\right) + q,\ x, y \in \Omega \qquad (2)$$

where $q$ is the rainfall rate (in m/s). For the water flooding simulation, the initial conditions of the model can be given by

$$H(x, y, 0) = B(x, y) + W^*(x, y),\ x, y \in \Omega \qquad (3)$$

when $W^*$ is initial water over regions, while the boundary conditions can be defined as

$$\frac{\partial H(x,y,t)}{\partial x} = 0,\ \frac{\partial H(x,y,t)}{\partial y} = 0,\ x, y \in \partial\Omega \qquad (4)$$

where $\partial\Omega$ is the boundary of the area domain.

## 3. Numerical Method

The numerical method for modeling of the rainwater overland flow has two major components. The first component is finding the depth of flow, which depth can be transferred to neighbors on the natural topography, and the second component is applying the depth of flow to the simulation of the water overland flow using difference approximations to the diffusion model in Equation (2).

### 3.1 The depth of flow

For each location of height field, water depth which exceeds the height of its neighbors can be detected and transferred to the neighbors. The depth of flow of the model is shown in Figure 1(a).

In Figure 1(a), the water in cell $L$ that is transferred to cell $C$ can be calculated by deduction of the water level on cell $L$ with the maximum value of the heights of topography on cells $L$ and $C$. Therefore, the depth of flow from cell $L$ to cell $C$ is given by:

$$l_{L->C} = H_L - \max(B_L, B_C) \qquad (5)$$

where $l_{L->C}$ is the depth of flow from cell $L$ to cell $C$, $B_L$ and $B_C$ are the heights of topography in cell $L$ and cell $C$, respectively, $H_L = W_L + B_L$ is the water level in cell $L$, and $W_L$ is the water depth in cell $L$.

The term $l_{L->C}$ in Equation (5) may be negative when the water level is lower than the height of topography in the neighboring cell as shown in Figure 1(b). In this situation, the depth of flow from cell $L$ to cell $C$ should be zero because the water at cell $L$ cannot be transferred to cell $C$. Therefore, Equation (5) is modified following Audusse, Bouchut, Bristeau, Klein and Perthame (2004) in the context of hydrostatic reconstruction:

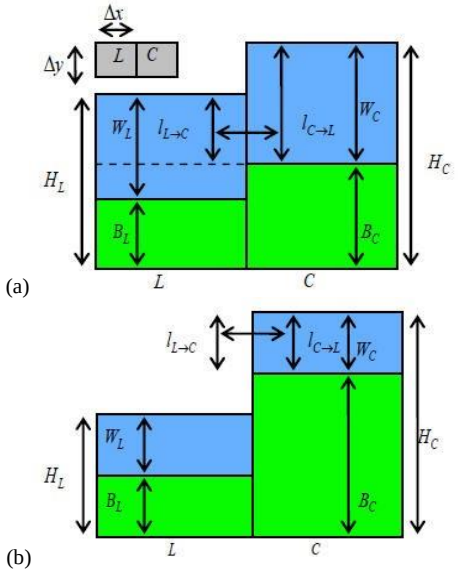


Figure 1. Schematic representation for (a) the water that can be transferred to the neighboring cells, and (b) the water that cannot be transferred to the neighboring cells.

$$l_{L->C} = \max(0, H_L - \max(B_L, B_C)). \qquad (6)$$

In this paper, the simulation of the rainwater overland flow uses the data grid as shown in Figure 2.

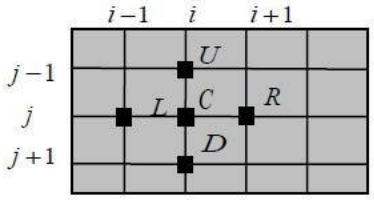


Figure 2. Two dimensional data grid

where $i$ and $j$ are the data grid indices in the $x$ and $y$ directions, respectively, and $C$, $L$, $R$, $U$ and $D$ represent the data cells $(i, j)$, $(i-1, j)$, $(i+1, j)$,$(i, j-1)$ and $(i, j+1)$, respectively.

The transfer of water from cell $C$ to its neighboring cells consists of outflows from cell $C$ to its neighbors and inflows from cell C's neighbors to cell $C$. These can be defined by applying Equation (6) as follows:

$$l_{L\to C} = \max(0, H^t_{i-1,j} - \max(B_{i-1,j}, B_{i,j})) \qquad (7)$$

$$l_{C\to L} = \max(0, H^t_{i,j} - \max(B_{i-1,j}, B_{i,j})) \qquad (8)$$

$$l_{R\to C} = \max(0, H^t_{i+1,j} - \max(B_{i+1,j}, B_{i,j})) \qquad (9)$$

$$l_{C\to R} = \max(0, H^t_{i,j} - \max(B_{i+1,j}, B_{i,j})) \quad (10)$$

$$l_{U\to C} = \max(0, H^t_{i,j-1} - \max(B_{i,j-1}, B_{i,j})) \quad (11)$$

$$l_{C\to U} = \max(0, H^t_{i,j} - \max(B_{i,j-1}, B_{i,j})) \quad (12)$$

$$l_{D\to C} = \max(0, H^t_{i,j+1} - \max(B_{i,j+1}, B_{i,j})) \quad (13)$$

$$l_{C\to D} = \max(0, H^t_{i,j} - \max(B_{i,j+1}, B_{i,j})) \quad (14)$$

### 3.2 Difference discrete scheme of the diffusion equation

The diffusion model in Equation (2) can be solved by the following difference discretization scheme:

$$\frac{H^{t+\Delta t}_{i,j} - H^t_{i,j}}{\Delta t} = C\cdot\left(\frac{H^t_{i-1,j} - H^t_{i,j} + H^t_{i+1,j} - H^t_{i,j}}{\Delta x^2} + \frac{H^t_{i,j-1} - H^t_{i,j} + H^t_{i,j+1} - H^t_{i,j}}{\Delta y^2}\right) + q^t_{i,j}, \quad (15)$$

where $\Delta x$ and $\Delta y$ are the data grid step sizes for index increment in the longitudinal and latitudinal directions, respectively, and $\Delta t$ is the size of the time step. The superscript $t$ denotes the index for time.

In order to allow the model to simulate water propagation in rugged terrain, we substituted the water level in Equation (15) with the depth of flow using Equations (7) to (14) as follows:

$$H^{t+\Delta t}_{i,j} = H^t_{i,j} + \Delta t\cdot C\cdot\left(\frac{l_{L\to C} - l_{C\to L} + l_{R\to C} - l_{C\to R}}{\Delta x^2} + \frac{l_{U\to C} - l_{C\to U} + l_{D\to C} - l_{C\to D}}{\Delta x^2}\right) + \Delta t\cdot q^t_{i,j} \quad (16)$$

And in order to have a stable scheme, the time step is limited by the following condition:

$$\Delta t \le 0.25\cdot\left(\frac{\min(\Delta x^2, \Delta y^2)}{C}\right). \quad (17)$$

### 3.3 Faster computing and visualization techniques

In order to improve the computational efficiency, the dynamic domain defining method or dynamic DDM (Busaman, Mekchay, Siripant & Chuai-Aree, 2015) is adapted in this work. The grid nodes are inspected to identify whether or not the cell is inside the computational domain. The cell is excluded from the computational domain if it and all its neighboring cells are dry. Only cells within the computational domain are adapted and computed in order to minimize the total number of computational cells. Thus, dynamic DDM keeps the same accuracy as other methods using all cells.

For visualization, the call list technique in OpenGL is adapted. The technique creates a list of topography images and only updates the water area from the computational domain. This technique is applied because when the cell and its neighbor are all dry with no rainfall yielding zero cell result, the computation and visualization is unnecessary for the cell.

### 3.4 Simulation algorithm

In this section, the simulation algorithm is presented. The algorithm overview of the models is shown in Algorithm 1. The inputs include the 2D grid array of size $w \times h$ and of resolution $\Delta x$, $\Delta y$, the values of the diffusion coefficient $C$, the elevation function $B$, the water over regions function $W^*$, the rainfall area $R$, and the rainfall rate $q$. The elevation function for each grid point is defined using a linear interpolation.

**Algorithm 1** Simulation and visualization algorithm
1: Inputs:
2: 2D grid array size $w \times h$ and resolution $\Delta x$, $\Delta y$.
3: Values of Diffusion coefficient $C$.
4: Elevation function $B$
5: Water over regions function $W^*$.
6: Rainfall area $R$, and Rainfall rate $q(t)$.
7: Simulation:
8: Initialization();
9: Draw Etopo and create a list $E$;
10: while the simulation is not finished do
11: Set Boundaries();
12: Calculate Water();
13: Tracking();
14: Call the list $E$ and draw water results;
15: $t \leftarrow t + \Delta t$;

The algorithm consists of steps in such a way that each cell can be computed within a step in the algorithm. The first step is the initialization. After that, a list for the elevation images is created by the call list technique. The while loop consists of the steps for the model computation, which includes setting of the boundaries, computation of water results, and illustration of the results. The loop is repeated until the simulation is finished according to $t$ that is increased at each iteration. In detail, there are four subsection algorithms consisting of initialization, setting boundary conditions, water calculation algorithm, and tracking.

#### 3.4.1 Initialization and system setup

For the initialization step, the parameters are set to be zero. The water depth for each grid node is defined by the water over regions function $W*$, whereas the array values $q^t_{i,j}$ are set to 0 if cell $(i, j)$ is not in the rainfall area $R$. In this step, each grid node is checked to identify whether the node is dry, wet, or experiencing rainfall. Only the nodes that are wet or experiencing rainfall are sent to the tracking procedure in order to define the initial domain for computing.

#### 3.4.2 Boundary conditions

The open boundary conditions are used for the simulation in this study. The boundary nodes can be defined by the water depth value with the value of its neighbors.

#### 3.4.3 Water results calculation

In this step, the model formulas in Sections 2 are used for calculating the water results. The algorithm starts by calculating $\Delta t$ in Equation (17). In the first loop, only the water results are computed with Equations (7) - (14) and (16). After

obtaining the solution, the wet nodes are checked and sent to the tracking procedure in order to define the new domain.

### 3.4.4 Tracking procedure

The tracking procedure is an important part for dynamic DDM. When a node is sent to this procedure, it and its neighbors which are not the boundary nodes must be checked to determine whether or not they are in the computation domain.

## 4. Model test

In this section, the models from the previous sections are tested for simulating and visualizing the water overland flow. The models can be applied to develop a computer program for simulating and visualizing water overland flow for any region in terms of 2D and 3D images. The earth topology (ETOPO) data by Shuttle Radar Topography data source, with a grid size of resolution 90 meters, can be used for the water overland flow simulation. The results from simulation can show the water propagation from the water sources to the risk regions. Figure 3 shows the simulation for the case of continuous heavy rain using the model based on the diffusion equation.

To show the performance of our algorithm of the simulation in each case, we compare the computational and visualization times of the simulation with our techniques and without the techniques (calculating all cells and not using the call list technique in OpenGL). Since the computation and visualization costs are mostly determined by the grid size, the simulation is done on different grid sizes. Table 1 shows the performance of the simulation for the case of rainfall. The simulation obtained by our algorithm takes less computational and visualization times than a simulation without our algorithm, as shown in Table 1.

## 5. Application

In this section, we applied the diffusion model with a numerical scheme to simulate the water flooding in 2011 in Nakhon Si Thammarat province of southern Thailand, which covers an area of approximately 3474 km$^2$. The latitude is from 8.02416666665457 N to 8.7666666666516 N, and the longitude is from 99.7799999999809 E to 100.184999999979 E.

The simulation was performed on the domain of digital terrain data 43,470 m × 79,920 m, generated from a resolution of 90 m using the Shuttle Radar Topography data source. The data grid size is 483×888 cells. For each cell, the rainfall rate $q_{i,j}$ is obtained from weather radar images. In this work, we used 165 weather radar images from a station in Surat Thani, a neighboring province, recorded during March 22 - 29, 2011. Figure 4 shows an example of a weather radar image and its components. All images were segmented by radar reflectivity values (dBZ) for each cell using the following algorithm.

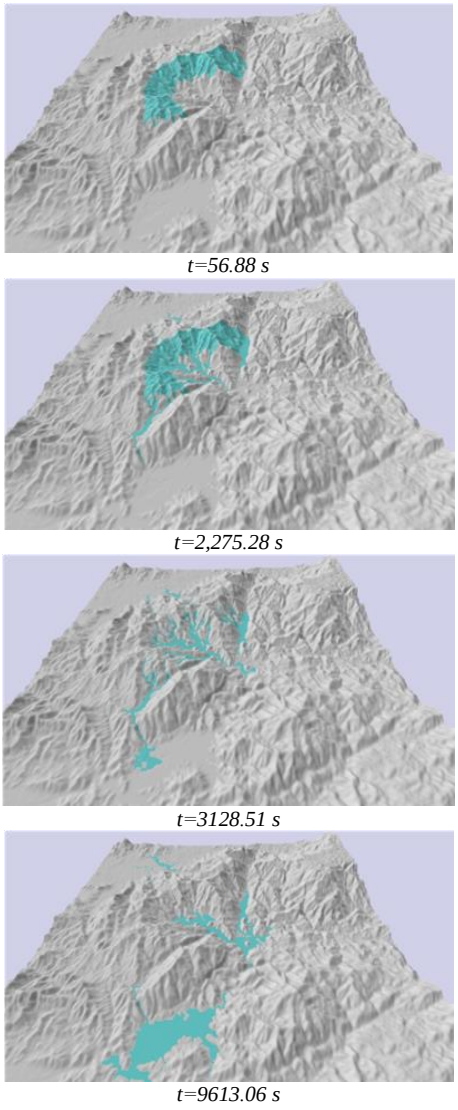


Figure 3. The simulation of water flooding from continuous heavy rain for each time step using $q = 0.0001m/s$, $C = 35.6m/s$, and $\Delta x = \Delta y = 90m$.

Table 1. Simulation times for computation ($T_c$) and visualization ($T_v$) using various grid sizes

| Grid sizes | With our technique | | | Without our technique | | |
|---|---|---|---|---|---|---|
| | $T_c$(s) | $T_v$(s) | Total time (s) | $T_c$(s) | $T_v$(s) | Total time (s) |
| 200×200 | 19.371 | 168.664 | 188.035 | 118.166 | 501.531 | 619.697 |
| 400×400 | 51.568 | 537.098 | 588.666 | 358.136 | 1767.933 | 2126.069 |
| 800×800 | 132.243 | 1826.586 | 1958.829 | 1493.914 | 7513.377 | 9007.291 |

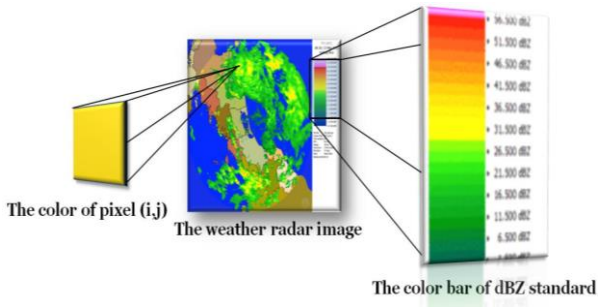


Figure 4. An example of an input radar image and its components

**Algorithm 2** Rainfall rate segmentation algorithm
1: Load data of the color value ($CdBZ_k$), and the dBZ standard value ($dBZ_k$);
2: For each cell of the image ($C_{i,j}$) Do
3: For each color of dBZ standard Do
4: Compute $norm_k := \left\| C_{i,j} - CdBZ_k \right\|$
5: Find $k_{\min}$, the index of the minimum value of $norm_k$
6: If $norm_{k_{\min}} \le \varepsilon$ then $dBZ_{i,j} := dBZ_{k_{\min}}$
7: Else $dBZ_{i,j} := -9999$

By the algorithm, each cell $(i, j)$ is checked with the dBZ standard bar. The dBZ value of each cell is defined by a dBZ standard value that has its color closest to the cell color. Thus $dBZ_{i,j} := dBZ_{k_{\min}}$, where $k_{\min} = \underset{k}{arc\min}\left\{\left\| C_{i,j} - CdBZ_k \right\|\right\}$. However, the cell cannot represent a rainfall area if the closest color has a difference more than a constant, thus $\left\| C_{i,j} - CdBZ_{k\min} \right\| > \varepsilon$. In this simulation, we used $\varepsilon = 20$. We set $dBZ_{i,j} := -9999$ for a non-rainfall area. In the simulation, the rainfall rate for each cell can be calculated from the dBZ values using the equation in Seliga (1997) as follows:

$$q_{i,j}^{t} = \begin{cases} \frac{1}{a}\left(10^{\frac{dBZ_{i,j}^{t}}{10}}\right)^{\frac{1}{b}} \cdot \left(\frac{10^{-3}}{3600}\right), & dBZ_{i,j}^{t} \neq -9999 \\ 0, & \text{otherwise} \end{cases} \quad (18)$$

where $a$ and $b$ are arbitrary parameters, taken based on Seliga (1997) to be 0.3 and 1.6, respectively**.**

The numerical experiment was simulated for 164 hours of flooding from March 22 to March 29, 2011 with the parameters $C = 135m/s$, $\Delta x = \Delta y = 90m$, and $\Delta t = 15$s.

The simulation for the numerical model took about 2 hours 12 minutes to complete. Figure 5 shows the flood simulation results at various times. The diffusion model illustrates the propagation from the water sources to the regions at high risk of flooding.

To determine the accuracy, the simulation results are compared with an actual satellite image as shown in Figure 6. It is seen that the flood plain obtained by the diffusion model is similar to the flooding area from the actual satellite image. The accuracy, $A$, can be calculated using the following equation:

$$A = \frac{1}{N}\left(\sum_{i}^{N} P_i^{M_1R_1} + \sum_{i}^{N} P_i^{M_0R_0}\right) \times 100 \quad (19)$$

Here, the values of $P_i^{M_1R_1}$ and $P_i^{M_0R_0}$ are checked at each cell $i$, where $P_i^{M_1R_1}$ is 1 when the model result and the real satellite images give the same wet area, while $P_i^{M_0R_0}$ is 1 when the model result and real satellite images show the same dry area; otherwise, the values of $P_i^{M_1R_1}$ and $P_i^{M_0R_0}$ are 0. The term $N$ is the total number of cells. By using the formula (19), the percentage of accuracy of the model result was 76.34%.

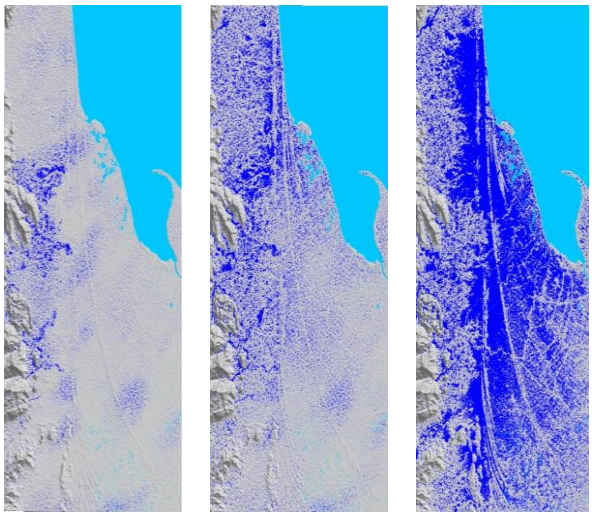


Figure 5. Simulation of water flooding at Nakhon Si Thammarat province at various times using C = 135*m/s*, Δx = Δy = 90*m*.

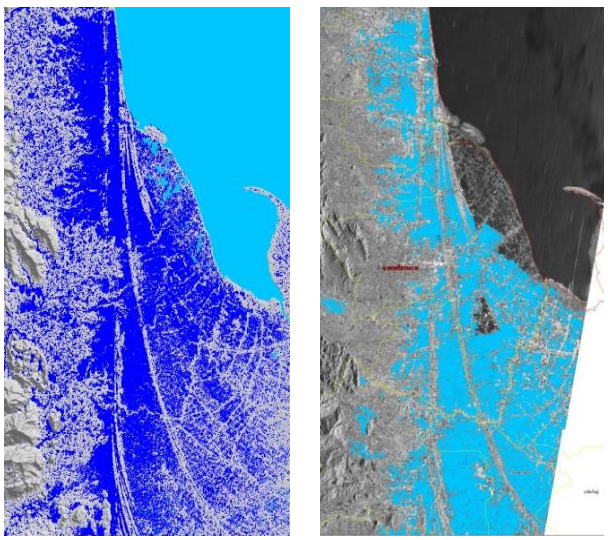

Figure 6. Comparison of flood simulation results (left) and real flooding (right) at Nakhon Si Thammarat province on 29 March 2011 at 10.30 AM.

## 6. Conclusions

There are numerous applications in virtual reality and computer animation that make use of the simulation and visualization of water overland flow. For water simulations, faster algorithms are preferred as long as their accuracy is not compromised. This paper provided a method for the modeling, simulation and visualization of water overland flow caused by continuous heavy rain. The results show that the model can describe the water propagation in rugged terrain. Moreover, the algorithm is efficient because it reduces the computational and visualization times for the simulations. The diffusion model and numerical method can be integrated with the algorithm for segmentation of weather radar to simulate the water flooding caused by continuous heavy rain in Nakhon Si Thammarat province, Thailand. The flood plain obtained by the diffusion model was similar to the flooding area shown by the satellite image. The investigated methods can be applied in virtual reality and computer animation such as flood risk assessments, water resources management, and disaster prevention from water flooding. However this method is not appropriate for the rogue waves that will be subject in further investigations.

## Acknowledgements

This research is supported by the Centre of Excellence in Mathematics, the Commission on Higher Education, Thailand.

## References

Alsdorf, D., Dunne, T., Melack, J., Smith, L., & Hess, L. (2005). Diffusion modeling of recessional flow oncentral Amazonian floodplains. *Geophysical Research Letters*, *32*.

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.

Benes, B. (2007). Real-time erosion using shallow water simulation. *Proceeding of the 4th Workshop in Virtual Reality Interactions and Physical Simulation.* The Netherlands*:* The Eurographics Association.

Benes, B., & Forsbach, R. (2001). Visual simulation of hydraulic erosion. *The Journal of WSCG*, *1*, 79–86.

Busaman, A., Mekchay, K., Siripant, S., & Chuai-Aree, S. *(*2015*).* Dynamically adaptive tree grids modeling for simulation and visualization of rain-water overland flow. *International Journal for Numerical Methods in Fluids*, *79*(11), 559–579.

Chuai-Aree, S., & Kanbua, W. *(*2007*).* Fast and real-time simulation of tsunami propagation. *Asia Modeling Symposium International Conference*, 490-495. doi: 10.1109/AMS.2007.95.

Dottori, F., & Todini, E. *(*2011*).* Developments of a flood inundation model based on the cellular automata approach: Testing different method to improve model performanceh. *Physics and Chemistry of the Earth*, *36*, 266–280.

Fiedler, F. R., & Ramirez, J. A. (2000). A numerical method for simulating discontinuous shallow flow over an infiltrating surface. *International Journal for Numerical Methods in Fluids*, *32*, 219–240.

Santillana, M. (2008). *Analysis and numerical simulation of the diffusive wave approximation of the shallow water equations* (Doctoral thesis, University of Texas at Austin, Austin, TX). Retrieved from https://pdfs.semanticscholar.org/da44/b64b7a09f1ff90e30a0116eac2ab59d63a13.pdf.

Seliga, T. A. (2011). *The NEXRAD radar system as a tool in highway traffic management* (Final Technical Report WA-RD 416.1). Olympia, WA*:* Washington State Department of Transportation.

Wang, Y. (2011). *Numerical improvements for large-scale flood simulation* (Doctoral thesis, Agriculture and Engineering Newcastle University, Newcastle, England*).* Retrieved from https*:*//core.ac.uk/download/pdf/40013498.pdf.