# A REAL-TIME 3D ANIMATION ENVIRONMENT FOR STORM SURGE

*Shu-Ching Chen[1], Keqi Zhang[2], Min Chen[1]*

[1]Distributed Multimedia Information System Laboratory
School of Computer Science, Florida International University, Miami, FL 33199, USA
[2]International Hurricane Center
Florida International University, Miami, FL 33199, USA

## ABSTRACT

This paper describes an approach to develop a high performance animation environment for storm surge. The system provides the capability to simulate the storm surge effects in the physical world by (1) modeling a region using the airborne light Detection And Ranging (LIDAR) data, USGS orthophotos, RLG road data and photos; (2) animating the storm impact by using the features of this model; and (3) providing the capability for users to explore the animation environment. We present our system by modeling the dataset collected from Ft. Lauderdale, a region in South Florida, USA.

## 1. INTRODUCTION

With the availability of digital data archive, the exponential growth of the affordable computational power and maturation of computer graphics technology, real time animations of the locations and events in the physical world become possible. Real time modeling of the physical world has many uses, such as disaster impact prediction, disaster recovery planning and training, urban planning and virtual tourism.

However, the current state of technology lacks the capability to translate storm damage predictions into a meaningful form by depicting the actual damage estimate at a location efficiently to be understandable by the general public who has little knowledge of computations, meteorology and mechanics. In order to address this issue, we use the progressive morphology filter, developed by our group, to process the LIDAR data to acquire a high-resolution Digital Terrain Model (DTM) automatically [1], utilize the OpenGL technology [6], 3D Studio tools [7] and the Virtual Terrain Project (VTP) [5] to create the 3D interactive environments, and extend the capability to animate buildings, vegetation and flooding as it pertains to storm surge effects.

This paper is organized as follows. Section 2 outlines the system architecture and introduces three modules in this system. Section 3 concludes this paper.

## 2. SYSTEM PARADIGM

The high-level system architecture of our proposed approach is outlined in Figure 1. As can be seen from this figure, there are three modules in this system: dataset processing module, model construction module and animation module.

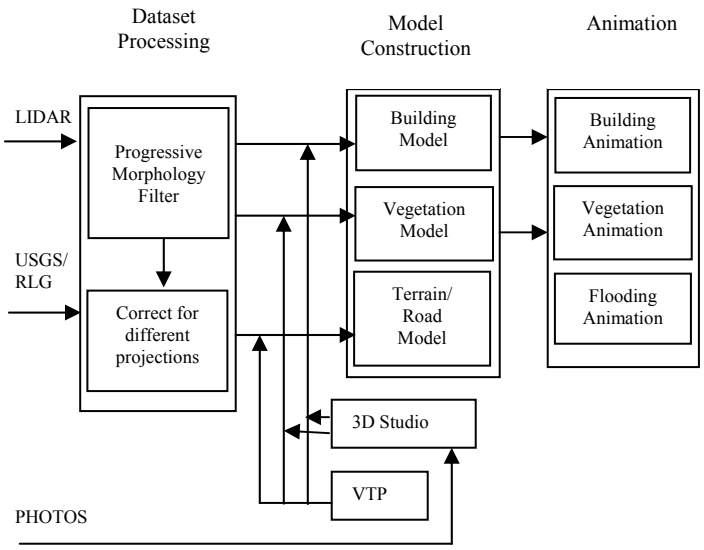


Figure 1: System architecture.

### 2.1. Dataset Processing Module

We have collected datasets to model the terrain, buildings, roads and vegetation of Ft Lauderdale, Florida, USA, which includes the collection of the LIDAR elevation data, USGS orthophotos and RLG of road location, and photos of real building and vegetation models. In order to create a modeled area rapidly and accurately, an important step

is to generate a high-resolution DTM for that area by processing the datasets automatically.

The methodology adopted in this system is the progressive morphological filter [1] for removing the non-ground measurements from the LIDAR data. The LIDAR data were derived using Optech 1210 LIDAR mapping system. Each flight surveyed a 600 m wide swath with a 0.3 m diameter laser footprint spaced approximately every 2.5 m. The raw points are sampled every 2*2 $m^2$ cell. If more than one measurement falls within a cell, the point with the minimum elevation is selected. If there is no measurement for a cell, nearest neighborhood interpolation is used to derive an elevation.

In the progressive morphological filter, two fundamental operations based on set theory, dilation and erosion, are extended to remove non-ground measurements in the LIDAR data. Considering a LIDAR measurement $p(x, y, z)$, the dilation of elevation $z$ at $x$ and $y$ is defined as follows.

$$d_p = \max_{(x_p, y_p) \in w} (z_p) \tag{1}$$

where points $(x_p, y_p, z_p)$ represent $p$'s neighbors (coordinates) within a filtering window $w$. Erosion is a counterpart of dilation and is defined as follows.

$$e_p = \min_{(x_p, y_p) \in w} (z_p) \tag{2}$$

The combination of erosion and dilation generates the opening operation that is employed to filter the LIDAR data. The opening operation is achieved by performing an erosion of the data set followed by a dilation. The erosion operation removes the tree objects of sizes smaller than the window size, while the dilation restores the features larger than the window size.

There is an abrupt change in elevation between a non-ground object (such as a building and a tree) and the adjacent ground, while the elevation changes of the terrain are gradual. This difference in elevation change makes the progressive morphological filter to be able to extract the DTM information by selecting the filtering window size and elevation difference threshold in the following manner. That is, instead of using a fixed filtering window size as [2], we perform the opening operation to a laser-scanned data with a line window that increases in sizes gradually.

$$w_k = 2b^k + 1 \tag{3}$$

where $w_k$ is the full window size, $b$ is the base of an exponential function, $k = 0,1,2, \ldots M$, and $2b^M+1$ is equal to the maximum window size.

Let $dh_0$ be the initial elevation difference threshold, $s$ be the slope, $c$ be the cell size, and $dh_{max}$ be the maximum elevation difference threshold. The elevation threshold $dh_{,k}$ can be given by

$$dh_k = \begin{cases} dh_0 & if \ w_k \le 3 \\ s(w_k - w_{k-1})c + dh_0 & if \ w_k > 3 \\ dh_{\max} & if \ dh_k > dh_{\max} \end{cases} \tag{4}$$

The DTM generating steps using the progressive morphological filter is shown in Table 1.

1. Given a set of LIDAR measurements P = $\{p_1, p_2, \ldots, p_n\}$, where $n$ is the total number of the points in the measurements and $p_i = (x_i, y_i, z_i)$, sample them in every 2*2 $m^2$ cell.
2. If the number of sampled points $m > 1$ in one cell, select the point $p$ with minimum elevation $(z_p)$.
3. If $m = 0$, use nearest neighborhood interpolation to derive an elevation.
4. Upon the initial filtering window size $w_0$ and elevation threshold $dh_0$ according to Equations (3) and (4), using the morphological filter whose major component is an opening operation to the measurements.
5. Obtain the non-ground point $p_{i,k}$ with $z_{i,k} > dh_k$ and the approximate surface model.
6. Continue to calculate the next values for $w_k$ and $dh_k$ by applying the morphological filter to the surface model obtained from the previous iteration, until $w_k$ is greater than a predefined maximum value.
7. Generate the DTM based on the data set after the non-ground measurements have been removed.

Table 1: the DTM generating steps using the progressive morphological filter

The advantages of using the progressive morphological filter is that various sizes of non-ground objects such as buildings and trees can be filtered out from the LIDAR data without prior knowledge of detailed size and elevation information of these objects. Therefore, it enables the automation in processing the laser scanning data. The resulting DTM, combined with the USGS orthophoto and RLG road data following the same map projection, is then imported to the model construction module to create the terrain model.

**2.2. Model Construction Module**

To the aim of creating the 3D interactive environments rapidly, we use OpenGL and VTP [5] (a current project using Open Scene Graph (OSG) API [8]) in our system.

OpenGL is an API that provides the interface to the underlying platform 3D graphics hardware. Based on this technology, our system provides the ability to support lighting/shading models, textures mapping and polygon manipulation, and delivers real-time 3D graphics display performance. What is lacking from OpenGL is the support for scene organization and collision detection between the objects and occlusion detection.

VTP [5] solves this problem by providing the capability to manage the creation and display of objects in a scene. VTP also provides a user-friendly interface which enables the users to easily walk or fly around the environment by using mouse navigation, such as location selection, compass control and zoom in/zoom out. In addition, the libraries are written in C++ and allow for the realization of maximum system performance. However, VTP lacks the capability to model complicated 3d objects. Most of the VTP objects consist of a single large polygon for each surface, and thus the programmers have to define the size, shape, location and texture for each polygon, which is quite tedious and inefficient to create more realistic objects.

In our model construction module, we provide the capability to model a region automatically and realistically by importing the DTM data and realistic 3D models into our system using the VTP interface. The DTM data is generated in the previous module, while the 3D models are constructed in 3D Studio [7], based on the photos of the typical building models and vegetation models in South Florida area.

Model construction can be further grouped into three categories: building models, vegetation models and terrain/road models.

- **Building models:** As mentioned earlier, VTP lacks the capability to model complicated buildings. To solve this problem, we use 3D Studio that is a prominent modeling tool used for animation, game and architectural development. A 3D building model can be created in 3D Studio by two major steps: object construction and object rendering. First, we create a building shape by using some primitive object such as a box and a cylinder, or by making the rotation or loft of some 2D object, with some modifications to get more complicated shapes. Second, to make a realistic building model, the rendering operation is quite important. The material properties, such as color, reflection and opacity, need to be set. Images from close range photos are used to map on the object to create textures. The result is then exported as a 3ds file that can be read by the VTP. As a result, we can import 3D Studio defined objects into our system by using the VTP interface to create a more realistic environment.
- **Vegetation models:** In addition to the believable buildings, we use two approaches to generate our vegetation models. One is planar billboard, and the other one is to utilize the models created with 3D Studio (as mentioned earlier). We combine these two methods to produce more realistic vegetation. Along each road, bush trees are densely distributed and palm trees are sparsely distributed. Because of the large number of bush trees, they are created by planar billboard, which is composed of two perpendicular meshes, to maintain the real-time performance of the system. The palm trees are created by importing the 3ds files, which are much more realistic and normally each palm tree has more than one hundred meshes.
- **Terrain/Road models:** The DTM created in the dataset processing module can be read by the system to create the terrain and road model directly.

Figure 2 presents the result of the modeled area based on our proposed model construction module.

Figure 2: Modeled area including terrain, building, and vegetation models.

### 2.3. Animation Module

The goal of the animation module is to generate believable storm effects to the modeled area. To this aim, we provide the capability to animate buildings, vegetation and flooding. A user-friendly GUI has been developed to allow users to navigate the animation environment.

- **Flooding Animation:** In our system, the approach of parametric modeling has been implemented. This technique provides a solution to the traveling wave equation based upon the work of Gerstner and models the wave as sinusoid modulated by decaying exponentials [3][4]. It describes the disturbance (X, Y and Z translation) of a particle on the water surface as a wave passes the particle, which can then capture the effect of the surface of the water flowing over itself. In VTP, water surfaces are represented as meshes of triangles, where the sizes of these triangles constrain

the levels of details achievable. A higher level of detail is directly related to the computational cost. In our system, we have applied this model to a portion of the terrain in favor of real time rendering and have approximated some of the fine scale detail by textures. Figure 3 shows the simulated environment in flooding.

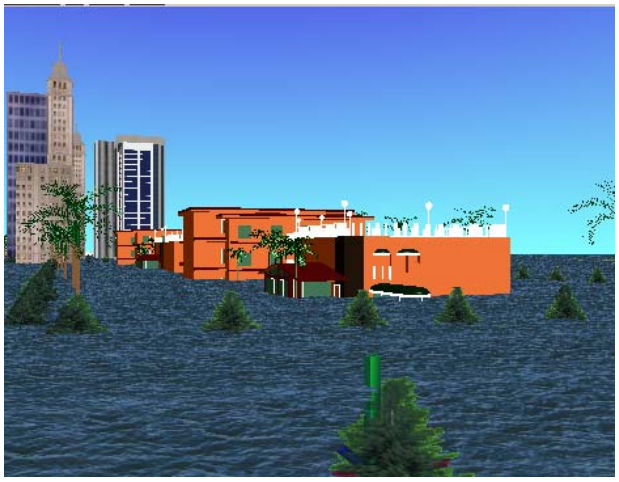

Figure 3: Simulated environment in flooding.

- **Vegetation Animation:** Variation of the planar billboard angles is used for vegetation animation in our system. This approach involves changing the angle of the planes with respect to the terrain surface. The tree could be portrayed as slanting in the direction of the wind. Figure 4 shows the animation result of a tree in the wind.

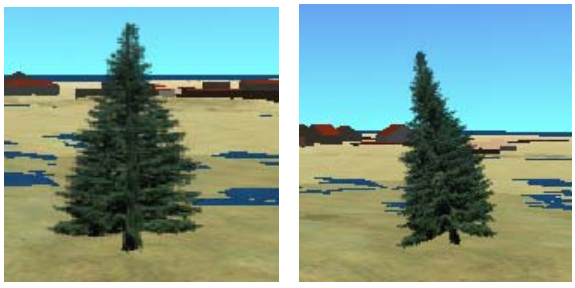

Figure 4: A tree and its corresponding animation result.

- **Building Animation:** In order to animate the effects of the shingle and sections of the roofs being blown off, we made a mesh, which is divided into many triangles, to model the surface. Then we calculate the track of each triangle being blown off by wind and change its location accordingly. Since these operations are computationally expensive, we limited them to the user's immediate viewing area.

In summary, to create a believable animation environment, we achieve fine scale modeling of dynamic surfaces, which are computationally expensive, around the user's immediate viewing area; and employ simpler texture mapping techniques at larger scales and distances to maintain the real-time performance of the system.

## 3. CONCLUSION

In this paper, we presented the high performance 3D animation environment for storm surge. Our system is based on the progressive morphology filter methodology, and utilizes the OpenGL technology, 3D Studio tools and VTP. The inputs for this system are the LIDAR data, USGS orthophotos, RLG road data and photos. This system has a variety of applications. For example, it can facilitate the planning and assessment of storm damage predictions by public and researches.

## 4. ACKNOWLEDGEMENT

This research was partly supported by a grant (FEMA-DR-1249-FL) from the Federal Emergency Management Agency. We would also like to thank Jianhua Yan and Jeff Strickrott for their contributions to this system.

## 5. REFERENCES

[1] Keqi Zhang, Shu-Ching Chen, Dean Whitman, Mei-Ling Shyu, Jianhua Yan, and Chengcui Zhang, "A Progressive Morphological Filter for Removing Non-ground Measurements from Airborne LIDAR Data," *accepted for publication, IEEE Transactions on Geoscience and Remote Sensing*, 2002.

[2] P. Lohmann, A. Koch, and M. Schaeffer, "Approaches to the filtering of laser scanner data," *International Archives of Photogrammetry and Remote Sensing,* vol. XXXIII, Part B3, pp. 540-547, 2000.

[3] A. Fournier and W. Reeves, "A Simple Model of Ocean Waves," *Computer Graphics and ACM Proceedings of SIGGRAPH,* vol. 20, no. 4, pp. 75-84, 1986.

[4] D. Peachey, "Modeling Waves and Surface," *Computer Graphics and ACM Proceedings of SIGGRAPH,* vol. 20, no. 4, pp. 65-74, 1986.

[5] http://www.vterrain.org

[6] http://www.opengl.org

[7] http://www.discreet.com

[8] http://www.openscenegraph.org