Modeling and visualizing avalanche flow using genetic algorithms and OpenGL

Donna Delparte*
Department of Geosciences, Idaho State University
Michael Peterson, Jahrain Jackson, John Perkins
Department of Computer Science and Engineering, University of Hawaii at Hilo

ABSTRACT: Simulating snow avalanche flow with numerical models and digital elevation model (DEM) data has allowed snow scientists to better understand flow movement in complex terrain. Commonly used parameters in these numerical models include knowledge of surface friction and snowpack characteristics, parameters which may be unknown for remote areas. Our model draws from gaming technology to simulate avalanche flow with an OpenGL particle physics based emulation using six gaming parameters. The data input requirements for the model are a DEM and an approximation of starting zone.

Using a genetic algorithm (GA) approach, a training dataset of 30 known avalanche paths with maximum runout and associated starting zones was used to optimize the gaming parameters to provide a best fit to the avalanche path outlines. Each iteration of the GA generated a path that is saved to an image file that can be reviewed. The optimal combination of parameters were used in an interactive tool to view the results as an animated flow. The purpose of the model and visualization component is to provide an easy to use interface to quickly and interactively view avalanche flow and runout for snow avalanche practitioners who do not require impact measurements or velocity values.

1. INTRODUCTION

This paper explores a novel approach to model avalanche flow using a historical avalanche dataset from Rogers Pass, Canada. Digital Elevation Model (DEM) data, along with the training database of known avalanche paths, is incorporated in an Open Graphics Library (OpenGL) environment. Genetic Algorithms (GAs) are then used to determine optimal physical based flow parameters.

Digital elevation model (DEM) data are used for mapping avalanche hazard and visualizing flow in mountainous regions around the world. Many hazard mapping techniques use solely terrain-based parameters to identify avalanche starting zones or potential release areas and for snow avalanche runout calculations (Haeberli et al. 2004; Maggioni and Gruber 2003). For estimating maximum avalanche runout conditions, topographic-statistical models (Lied and Bakkehoi 1980; McClung and Mears 1991) and one to three dimensional dynamic flowing avalanche models (Bartelt et al. 1999; Gruber and Margreth 2001; Gruber and Bartelt 2007) have been utilized in mountain ranges worldwide.

The computer model RApid Mass MovementS (RAMMS) developed by the Swiss Institute for Snow and Avalanche Research is a three dimensional numerical model designed to model and visualize snow avalanche risk assessment (Christen et al. 2010; Fischer et al. 2012). The RAMMS model requires inputs of a DEM, release zone with fracture height and friction parameters. The RAMMS model is similar in purpose to our approach but utilizes numerical models specific to avalanche flow (Voellmy-Salm or random kinetic energy) and requires the fracture height and friction inputs.

Open Graphics Library (OpenGL) is a cross-platform, multi-language, low-level API of over 250

**Corresponding author address:* Donna M. Delparte, Idaho State University, 921 S 8th Ave., Pocatello, ID, USA 83209-8072; tel: 208-282-4419; fax: 208-282-4414; email: delparte@isu.edu

functions that provide a uniform programming interface for many different graphics acceleration hardware platforms. Every implementation includes the entire feature set, even if the feature must be emulated in software. OpenGL has the primitive objects a programmer requires to render scenes, such as points and polygons, as well a rendering pipeline called the OpenGL state machine to convert these primitives into pixels on a display device. The OpenGL API includes a robust particle physics simulation capacity. Despite the steep learning curve of its low-level design, OpenGL is widely used in scientific, mathematic and data visualization, video games, Computer Aided Design (CAD) and flight simulation (www.opengl.org).

Genetic Algorithms (GAs) are stochastic optimization algorithms that mimic principles of natural selection in order to 'evolve' a solution to a problem. The GA evolves a population of initially random solutions to a problem over a preset number of generations. The quality of a solution is referred to as 'fitness'; the GA seeks to optimize the fitness of evolved functions given an objective function. In successive generations, the GA applies fitness pressure in preference to improved solutions through use of selection, recombination, and mutation operators. Over a number of generations, poor solutions are removed while solutions demonstrating improved fitness reproduce and propagate. Due to evolutionary pressure, the average fitness of the population increases until highly fit solutions are obtained or until the GA reaches a predetermined limit on the number of generations (Holland 1975). In the present study, the GA evolves a set of parameters controlling a model of avalanche flow. The GA seeks to select a set of parameter values such that the behavior of the avalanche model closely matches previously observed real world avalanche data (Delparte 2008).

2. METHODS

2.1 Study Area

The digital data sets used for this analysis include an enhanced DEM that was created with digital stereo photogrammetry for Rogers Pass, Glacier National Park, Canada. A 5 m resolution gridded DEM was generated by interpolation from stereo digitized breaklines indicating breaks in slope and digitized point data (Delparte 2008). Training datasets consist of 30 avalanche paths from the highway corridor that were digitized in stereo with maximum runout extent based on a 40 year history (Fig 1).

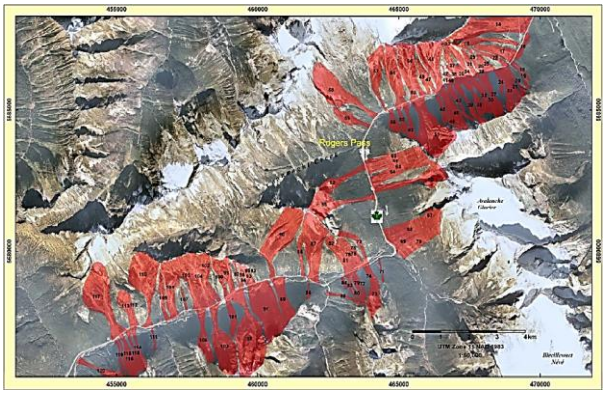

Figure 1: Avalanche paths along the Rogers Pass highway corridor

2.2 OPENGL Programming

To simulate avalanche flow an OpenGL particle physics based emulation incorporated eight parameters: initial height, bounce friction, stickiness, damping force, turbulence force, clumping factor, viscosity and cell grid size. The data input requirements for the model are a DEM and an approximation of starting zone. The parameters are defined below:

Initial height is the starting position of the particles.

Bounce friction affects the velocity of the particles. A larger bounce friction value will result in a smaller post-collision velocity.

Stickiness is a minimum velocity; particles that do not exceed this minimum velocity in a time slice immediately stop moving.

Damping force reduces the velocity of particles when there is no collision, much like air friction.

Turbulence force has a negative influence on a particle's velocity during collision. Higher turbulence values result in a smaller average velocity.

Clumping factor affects how particles move as a group and interact with both individual and average particle velocities.

Viscosity is defined as the "thickness" of the particle "flow". Thus a higher viscosity results in slower overall flow.

Cell grid size refers to a 'force map' that is created as particles flow in the study area, and this force map can then be compared to an observed avalanche. The cell grid size changes the size of the grid used to create this map.

The following formula computes the velocity of the particle, and incorporates some randomness.

Formula: Pv = **R** * **L** * (1 - **bf**) + **L** * **rv** * **t** * **d**
Where **R** is the particle radius, **L** is the length of the velocity vector, **bf** is the bounce friction, **rv** is a random vector**, t** is turbulence and **d** is density.

In the OPENGL environment the formula used for flow was:

$$\tau = \mu \frac{du}{dy}$$

Where:

τ is the shear stress exerted by the fluid

μ is the fluid viscosity

$\frac{du}{dy}$ is defined as the velocity gradient perpendicular to the direction of shear, or correspondingly the strain rate

The flow equation is estimated using the Finite Difference Method (FDM). The study area is discretized by dividing it into an overlapping grid and solutions are interpolated across the grid in time steps. Ultimately, this allows the OpenGL pipeline to approximate values for particles moving on a complex geometric shape very quickly. FDM is defined as:

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} + \frac{\partial H}{\partial z} = 0$$

where Q is the vector of conserved variables, and F, G, and H are the fluxes in the x, y, and z directions respectively.

2.3 Genetic Algorithm

The GA seeks to optimize the values of the previously described flow parameters. Each parameter is represented by the GA as a real-valued number. The initial values for each parameter are randomly chosen from a uniform distribution among the legal parameter values. Parameter values are permitted to fluctuate within the following ranges:

Parameter	Minimum	Maximum
initialHeight	0.0	100.0
bounceFriction	0.0	0.25
stickiness	0.1	0.75
dampingForce	0.0	0.15
turbulenceForce	0.25	5.0
clumpingFactor	0.0	1.0
viscosity	0.0	1.0
gridSize	64.0	512.0

The pseudocode for the GA appears below:

```
initialize Population
-get initial fitness values
-sort population by fitness
for i = 1 to #generations do
      -save best individual for
            next gen
      -generate next gen from
            current gen
      -calculate fitness of new
            gen
      -sort new population by
            fitness
end
```

In each generation, a new generation is calculated from the previous generation as follows. The current generation forms a population of parents. The most highly fit parent is saved for the next generation. The remainder of the new generation is populated by creating children from parents within the parent pool. For each child, two parents are chosen using tournament selection (Goldberg & Deb, 1991). Two randomly chosen individuals participate in each tournament; the higher fit of the two is chosen as the first parent. Two more randomly chosen individuals are selected as the second parent. Selection occurs with replacement.

Once two parents are chosen, a child is created from them via uniform recombination (Spears & De Jong, 1991). Here, we employ an adaptation of standard uniform crossover for real-valued representations. For each parameter, one of the two parents is randomly chosen. The new value for the parameter in the child is chosen from a Gaussian distribution centered around the value of the parameter within the parent. By drawing from a random distribution rather than using the original value from the parent, the GA may consider new parameter values rather than being constrained to the values that were present in the original population.

After creating new children via recombination, 10% of the parameters in the new generation are randomly mutated in order to promote diversity in the population. The new generation is sorted by fitness, and the GA then proceeds to the next generation. By applying fitness pressure via the selection operator, the GA evolves an initially random population into a highly fit population over a number of generations. The solution reported by the GA will be the most highly fit solution present in the final population.

3. RESULTS

The GA fitness function calculates a mean squared error (MSE) by comparing an output force map with its corresponding training set. Errors are weighted so that the farther the pixel is from the observed flow path, the larger the error weight value returned. Errors are biased in that if the pixel falls within the observed path, the error is scaled by 1.5, meaning that under-fitting the path is worse than over-fitting. “Smaller” fitness values trump “larger” fitness values, with 0.0 being a perfect fit. For our preliminary results, fitness values have been observed ranging from 0.01 to 0.25. The GA was run on a population size of 20 for 100 generations on training path 7. The best fit individual parameters from that run were tested on avalanche path 11. The fitness for the best individual on path 7 is 0.02; when run on 11 the fitness is 0.12.

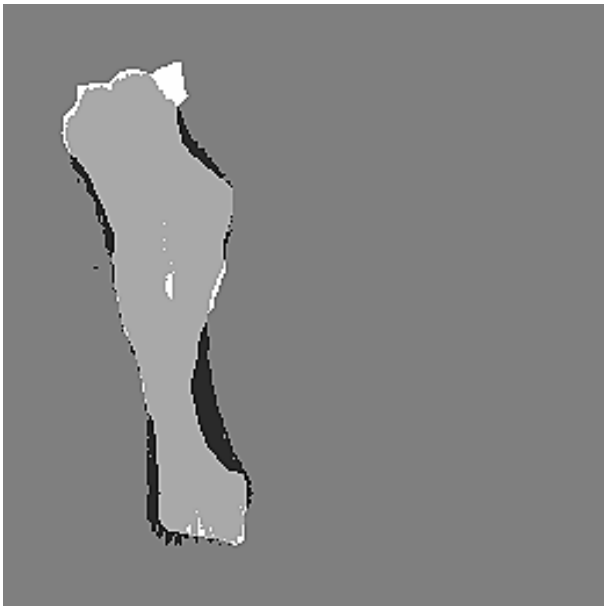

Figure 2: Avalanche Path 7. White areas represent the observed avalanche path, black areas represent predicted coverage and grey shading represents overlap between predicted and observed.

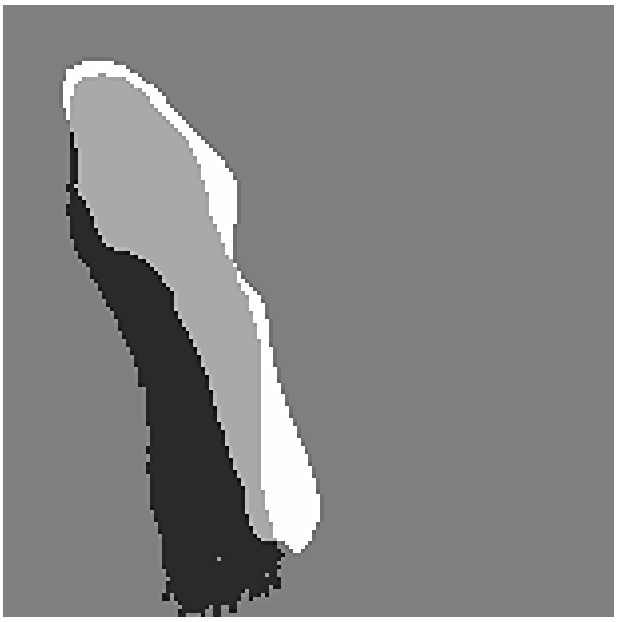

Figure 3: Avalanche Path 11. White areas represent the observed avalanche path, black areas represent predicted coverage and grey shading represents overlap between predicted and observed.

The OpenGL environment has been customized to allow a user to visualize the results of the optimized GA parameters (Fig 4).

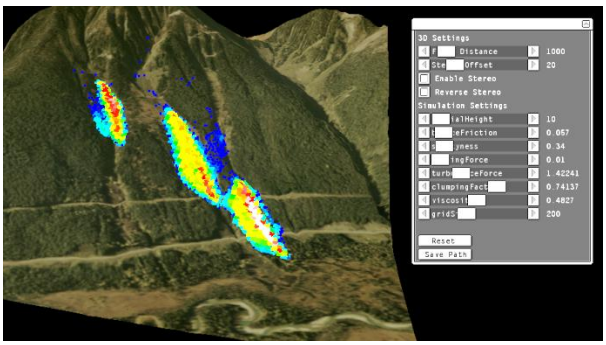

Figure 4: OpenGL Avalanche Flow Simulation

4. DISCUSSION

The animated flow interface allows a user to easily adjust all parameters on the fly and visualize the animation in a 3D view or in stereo if desired. Colored shading indicates the "intensity" of the flow with lighter blue colors around the edge and reddish tones in the mass of the flow (Fig 4). The application provides an easy to use interface to quickly and interactively view avalanche flow and estimated maximum runout for snow avalanche practitioners who do not require impact measurements or velocity values, but who want to view a customized avalanche flow approximation and maximum runout suitable for their mountain region.

5. REFERENCES

Bartelt P, Salm B, Gruber U (1999) Calculating dense-snow avalanche runout using a Voellmy-fluid model with active/passive longitudinal straining. Journal of Glaciology 45(150):242-254

Christen M, Kowalski J, Bartelt P (2010) RAMMS: Numerical simulation of dense snow avalanches in three-dimensional terrain. Cold Regions Science and Technology 63:1-14

Delparte D (2008) Avalanche Terrain Modeling in Glacier National Park, Canada. In: Department of Geography. University of Calgary, Calgary, AB, Canada, p 179

Fischer J, Kowalski J, Pudasini S (2012) Topographic curvature effects in applied avalanche modeling. Cold Regions Science and Technology 74-75:21-30

Goldberg D, Deb K (1991) A comparative analysis of selection schemes used in genetic algorithms. In: Foundations of Genetic Algorithms. Morgan Kaufmann 69-93

Gruber U, Margreth S (2001) Winter 1999: a valuable test of the avalanche-hazard mapping procedure in Switzerland. Annals of Glaciology, 32: 328-332

Gruber U, Bartelt P (2007) Snow avalanche hazard modelling of large areas using shallow water numerical methods and GIS. Enivronmental Modelling & Software 22:1472-1481

Haeberli W, Benz C, Gruber U, Hoelzle M, Kääb A, Schaper J (2004) GIS applications for snow and ice in high-mountain areas: Examples from the Swiss Alps. In: Bishop MP, Shroder JFJ (eds) Geographic Information Science and Mountain Geomorphology. Springer-Verlag, Berlin, pp 381-402

Holland J (1975) Adaptation in Natural and Artifical Systems. University of Michigan Press, Ann Arbor, MI

Lied K, Bakkehoi S (1980) Empirical calculation of snow-avalanche run-out distance based on topographic parameters. Journal of Glaciology, 26(94): 165-177

Maggioni M, Gruber U (2003) The influence of topographic parameters on avalanche release dimension and frequency. Cold Regions Science and Technology 37:407-419

McClung D, Mears A (1991) Extreme value prediction of snow avalanche runout. Cold Regions Science and Technology, 19:163-175

Spears W, De Jong, K (1991) On the virtues of parameterized uniform crossover. In: Proc. 4th Int. Conf. on Genetic Algorithms. Morgan Kaufmann 230-236