Virtual Sandbox

Koichi Onoue Tomoyuki Nishita
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{onoue,nis}@is.s.u-tokyo.ac.jp

Abstract

Interactive applications such as virtual reality systems have become popular in recent years. A ground surface composed of a granular material can be deformed when it comes into contact with an object, and, in this paper, we propose a deformation algorithm for the ground surface which is useful for such applications. The deformation algorithm is divided into three steps: (1) detection of the collision between an object and the ground surface, (2) displacement of the granular material, and (3) erosion of the material at steep slopes. The proposed algorithm can handle objects of various shapes, including a concave polyhedron, and a texture sliding technique is proposed to represent the motion of the granular materials. In addition, the proposed algorithm can be used at an interactive frame rate.

1 Introduction

In recent years, computer performance has increased rapidly, and computer graphics has been applied to interactive applications. Such applications include virtual sculpting systems [4, 8, 13] and virtual clay modeling systems [11, 6]. For interactive applications such as virtual reality systems, modelers, and games, it is very useful to deform the ground surface, where this consists of a granular material, such as sand. For example, users can interactively drag a bucket on the sand, scoop it up in the bucket, or drop it.

The aim of this paper is to study the deformation of a ground surface consisting of a granular material when it comes into contact with rigid objects. In this paper, the ground surface is modeled as a height field. Granular material on objects is represented by height spans and that in the air is modeled using a particle system.

The main contributions of this paper are:

- A ground surface can be deformed by objects of various shapes including concave polyhedra using a *Height Span Map* (HS Map). Collapse of granular material on objects is also considered.
- An efficient and intuitive deformation algorithm is proposed to handle various types of object motion such as dragging/falling to the ground and scooping/dropping granular material.
- A rendering method is proposed to represent the motion of granular material by texture sliding.
- An anti-aliasing technique is proposed. This gives a reasonable rendering of the results at interactive frame rates.

The rest of this paper is organized as follows. First, previous work related to the deformation of ground surfaces is introduced in Section 2. The algorithm proposed in this paper for deformation of the ground surface is described in detail in Section 3. Next, the method of displaying the ground surface is described in Section 4. The results of deformation of the ground surface by collisions with various objects are shown in Section 5. The advantages of the proposed algorithm are summarized in Section 6. Future work is also described in this section.

2 Related Work

Several researchers have proposed methods for the deformation of ground surfaces on coming into contact with objects [5, 2, 15]. Li et al. proposed a model [5] based on physical laws, where the ground surface is represented as a height field and which simulates slippage of the soil and other operations (*e.g.* digging, piling, or carrying) in real time. Their goal is similar to ours. However, their method can only deal with convex objects for carrying the soil, and also does not permit dragged objects other than a bulldozer. Moreover, the mark made where the bulldozer has passed cannot be shown. These limitations are overcome using the method proposed in this paper.

Chanclou et al. also proposed a model based on physical laws [2]. In their model, the ground surface was modeled as an elastic sheet. Their model can generate contact marks on the ground caused by objects, but the calculation time is long, and their model is unable to represent granular material on objects.

Sumner et al. proposed a simple algorithm in which the contact marks made by objects on ground surfaces are generated by a simulation based on the appearance [15]. Their basic idea is employed in this paper. However, their model also cannot represent granular material on objects.

None of these three methods can give a representation of granular material on a concave object. One of the contributions in the work presented here is a method for doing this.

Benes et al. proposed a layered representation for visual simulation of terrain erosion [1]. Their method can be used to calculate the motion of granular material on static rigid objects, but is unable to deform the ground surface when an object comes into contact with it.

Among the various possible types of ground surfaces, several researchers have proposed models and rendering techniques for snow. Nishita et al. modelled snow using metaballs, and represented the snow piled onto various objects [10]. Fearing proposed a model to depict the fall and accumulation of snow on objects and the ground surface, and a stability model for the snow [3].

For larger-scale modelling of the ground surface, Musgrave et al. proposed a terrain generation method [9]. They generated the terrain by using fractal and erosion models. Onoue et al. generated and rendered desert terrain, which included wind-ripples and dunes [12].

3 Ground Surface Deformation

In this section, the algorithm used to calculate the deformation of the ground surface is explained. The model of the ground surface used in this algorithm is explained first, then an outline of the deformation algorithm is described, and each step of the algorithm is explained in detail thereafter.

3.1 Model of the ground surface

In the proposed algorithm, the volume of granular material is divided into vertical columns along a square lattice that defines a height field. Each lattice point in the height field represents a vertical column of granular material and the top of the column is centered at the lattice point. The height of the ground surface is assumed to be the distance from a base plane (0 in height), and is assumed to be positive. The initial height of each column can be set in various ways, such as a constant across the height field or with random heights. The technique can be extended to real terrain data or data made by other modelers.

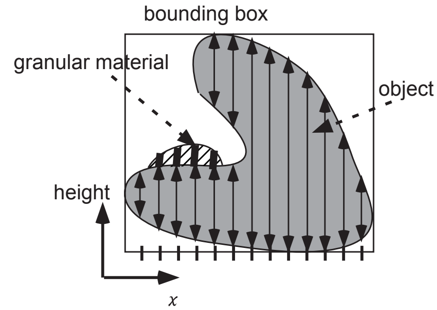


Figure 1. HS Map in cross section.

3.2 Model of granular material on objects

Any object that can be represented by polygons including a concave polyhedron can be dealt with in this paper.

As described in Section 3.1, the ground surface is defined as a height field, but, because a height field is a single-valued function, it cannot itself represent granular material on an object.

Though particle systems and voxels are used to calculate the motion of granular material, these models are time-consuming, and not suitable for interactive applications. It is necessary for particle systems to calculate the interactions between a lot of particles with volume, and it is necessary for voxels to calculate the quantity of granular material for each voxel at every time step, and the number of voxels needs to be large to achieve a realistic rendering of the result. In the proposed algorithm, objects and granular material on them are represented by an HS Map, which is a two-dimensional array of Height Spans (arrows(↕) in Fig. 1). Each height span represents the span of an object at each grid point. The height of the granular material at each grid point is also represented in the HS Map (bold lines on arrows in Fig. 1). The motion of the granular material on the object can be calculated efficiently using the HS Map, because only the change in the height of the granular material for each height span needs to be taken into account, and the number of height span can be less than the number of particles or voxels.

Related data representations have been proposed for image based rendering [14] and for simulation of the erosion of a layered terrain [1]. An HS Map is a modified structure of a layered terrain [1] allocated to each object. An example of an HS Map is shown in Fig. 8. Each red/green rectangle represents the upper/lower side of the height span, and each blue rectangle is for the upper limit of granular material on each height span.

An object formed from polygons is translated to this data structure as follows.

1. The size and resolution of the HS Map is set to the size of the bounding box of the object and to the value used

for the height field, respectively.

2. Using the bottom plane of the bounding box as a screen, each polygon of the object is rasterized using the Z-buffer method (implemented by software). The resolution of the screen corresponds to the resolution of the HS Map. Let d be the depth at each pixel and ori be the orientation (upward/downward) of the normal to the polygon. A pair (d, ori) is stored in a list structure allocated for each pixel. d corresponds to the height in the bounding box.
3. After rasterizing all polygons, each list is sorted by the value of d.
4. Height spans are generated by coupling elements with $ori = upward$ and $ori = downward$.
5. If granular material is on the object, the height of the granular material is set by rasterizing the polygons representing it.

3.3 Deformation of the ground surface

Because this paper targets deformation of the ground surface, objects are assumed to be unaffected by collision with this surface. Although the motion of objects is controlled by the user in this paper, the motion of the objects could be computed using dynamic simulations, or determined by motion capture data. In addition, all objects are assumed to be above the base plane.

The deformation of the ground surface for one frame of animation is calculated in the following order.

Collision detection The detection of a collision between an object and the ground is achieved using the HS Map.

Displacement of granular material The granular material of columns in contact with the object is forced outward to the surrounding columns.

Erosion Steep slopes are detected, and the surface is made smooth by moving granular material from high to low columns.

Each step of the algorithm is explained in detail in the following subsections.

3.3.1 Collision detection with objects and the ground surface

In order to deform the ground surface in contact with an object, it is necessary to find both where and to what depth the object penetrates. This information is found by testing the collision between each object and the ground surface as follows.

1. An intersection test is carried out between the bounding box of the object and that of the ground.
2. If these bounding boxes do not intersect, the object is assumed not to be in contact with the ground. Even if the object is not in contact with the ground, the HS Map of the object is updated if granular material is on the object and the object is rotated.
3. If these bounding boxes do intersect, the HS Map of the object is updated by the algorithm described in Section 3.2.
4. The bounding box of the object is projected onto the ground to define a region for collision testing.
5. Detection of the collision between an object and the ground is done using the HS Map. The height of each column in the region is compared with the height of the bottom of the object found in the HS Map.

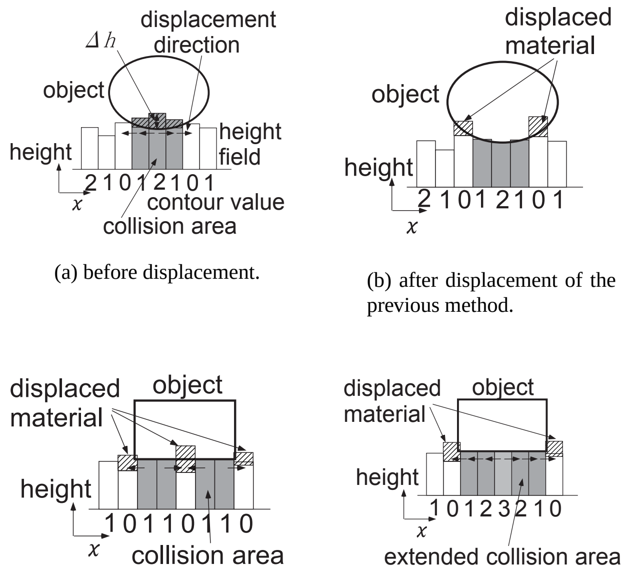
3.3.2 Displacement of granular material

The granular material of a column that collides with the object is displaced to its neighboring columns. The amount of material $\Delta h(x, y)$ displaced from the column is the height of the intersection of the object and the column.

In this paper, we simplify the motion of objects into two types: fall and drag. An object is defined as falling if the vertical component of the object's velocity is less than zero; otherwise the object is defined as being dragged.

In the case of an object falling, Sumner's displacement algorithm [15] is used. In this algorithm, a contour value is defined as the discrete distance from the boundary columns of the collision area to each column of the height field, and a contour map is defined as a two-dimensional array of contour values. Displacement is processed sequentially from those columns with large contour values, and $\Delta h(x, y)$ is distributed equally to those adjacent columns with lower contour values (refer to Fig.2(a)). Eight adjacent columns are examined here. However, this algorithm allows the columns at the boundary of the collision area to be intersected by objects (Fig.2(b)), and this will give incorrect results in the erosion step described later. We have modified the algorithm as follows: if any column is intersected by an object after Sumner's displacement step, displacement is continued and the contour map is extended outside of the collision area.

In the case of an object being dragged, the displacement is processed sequentially from the columns behind the object to columns in front of the object along the direction of the object's movement. In this case, $\Delta h(x, y)$ is distributed equally from each column $C(x, y)$ in the collision area to it's adjacent columns $C_n(x', y')$ which satisfy the following condition: $\mathbf{d}_{obj} \cdot \mathbf{d}_{col} \geq 0$, where $\mathbf{d}_{obj}$



(a) before displacement.

(b) after displacement of the previous method.

(c) after displacement of the previous method.

(d) Displacement again after (c).

Figure 2. Displacement of granular material.

is the object's direction vector in the horizontal plane, and $\mathbf{d}_{col} = (x', y') - (x, y)$.

In the case such that the ground surface has dents as shown in Fig.2(c), the above method allows an object to intersect the ground where the collision area surrounds a small non-collision area. In this case, we extend the collision area to the projected area of the object (Fig.2(d)). Then we calculate the displacement step again.

3.3.3 Erosion

After the displacement step, granular material accumulates in the boundary area of the height field in contact with the object, and so the height differentials in the area are larger than the surroundings. In the erosion step, steep slopes are detected (points with a large difference in height between adjacent columns), and these slopes are decreased by collapsing the columns, that is, by moving the granular material from the higher to lower columns.

The erosion algorithm we used is based on Sumner's method [15]: (1) examine the slope between each pair of adjacent columns in the height field, where eight adjacent columns are examined for each column, (2) if the slope is larger than θ (an unique *angle of repose* for the granular material), move the granular material from the higher to lower column. We have optimized the sequence of columns examined to calculate the erosion step more efficiently. We reduce the iteration times of the erosion algorithm by examining columns from the boundary to the surrounding area of the collision area.

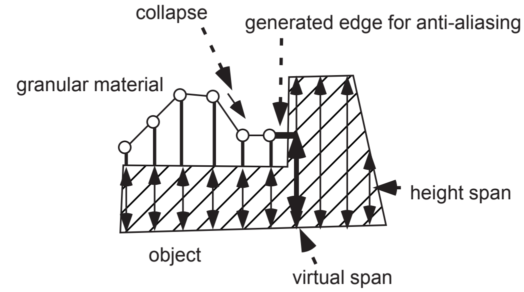


Figure 3. Erosion on object.

In order to represent the time-dependent progress of an erosion process, such as sand settling back down after an object has been removed, columns for which collapse is calculated are stored for a constant time. The erosion step is calculated for these stored columns at each time step. Collapse from the ground onto objects is also calculated for this time.

Erosion on objects is calculated as follows.

1. For each height span in the HS Map, the neighboring height spans onto which granular material can collapse is found.

2. The collapse between neighboring height spans is calculated: compare tops of granular materials on height spans, and move the granular material from the height spans with higher granular materials to the height spans with lower granular materials using the same method used for collapse on the ground (see Fig. 3). The tops of granular materials are shown as white circles in Fig. 3.

3. In the case where granular material on an object collapses to the ground, the material is moved directly to the ground if the height of the granular material on the height span is near to the height of the neighboring column on the ground. Otherwise, particles are generated and dropped vertically assuming that wind does not blow (see Fig. 4). The volume of particles is the same as the quantity of granular material moved by the collapse.

4 Rendering of Ground Surface

The rendering method of the ground surface is as follows. The height field is rendered as a polygonal mesh. Granular material on objects is also rendered as a polygonal mesh which is generated by connecting the granular material on each height span. Particles are rendered as small points.

4.1 Texture sliding

The appearance of granular material is represented as textures in our method. We use a 256x256 texture image repeatedly mapped onto the polygonal mesh. A part of the image (32x32) is texture mapped onto each polygonal. However, when erosion of granular material occurs, the generated animation looks unnatural if the texture image mapped onto a polygon is fixed. Therefore we should represent the granular material as moving on the ground surface.

Though a flow visualization method using moving textures [7] has been proposed, the texture of granular material should be constant everywhere on the ground surface. We propose the following texture sliding technique exploiting the fact that the texture image has a very fine pattern and continuity of the pattern at the edges of polygons is unimportant.

Each vertex $\mathbf{v}(x, y, z)$ of the polygonal mesh is given a texture coordinate $\mathbf{T}(u, v) = (\omega_x x, \omega_y y)$, where ω_x and ω_y are parameters which determine the size of a texture image mapped onto a polygon. We used $\omega_x = \omega_y = 0.125(= 32/256)$ in our experiment. Each polygon is given an offset $\mathbf{T}_{offset}$ of texture coordinate (refer to Fig. 4). For each column where collapse is calculated, we update $\mathbf{T}_{offset}$ by adding $\Delta\mathbf{T}_{offset}$ at each time step. $\Delta\mathbf{T}_{offset}$ is calculated by the following equation.

$$\Delta\mathbf{T}_{offset} = -\gamma Q_{collapse}\mathbf{E}, \qquad (1)$$

where $\mathbf{E}(e_x, e_y)$ is the collapse direction which is one of the eight directions from a column to it's adjacent columns in the horizontal plane, $Q_{collapse}$ is the quantity of granular material moved by the collapse, and γ is a constant which determines the apparent speed of granular material. We used $\gamma = 0.05$ in our experiment. When granular material can collapse to several adjacent columns from a column, equation (1) is calculated for each collapse direction $\mathbf{E}$, and $\Delta\mathbf{T}_{offset}$ for each corresponding polygon is added. Finally, each polygon is rendered. A texture coordinate for each vertex of the polygon is shifted by $\mathbf{T}_{offset}$.

4.2 Anti-aliasing

There is a trade-off between accuracy and computing time of the proposed algorithm. We propose an anti-aliasing technique to achieve a reasonable rendering of the results at the interactive frame rate.

We use *virtual spans* to reduce aliasing at the boundaries between granular material and objects' polygons which are approximately vertical (the region (1) of Fig. 6(a)). Virtual spans are generated inside height spans with which granular material is in contact as shown in Fig. 3. The height of a virtual span is set to that of the adjacent height spans. The aliasing is reduced by rendering polygons of the granular material connected to the virtual spans.

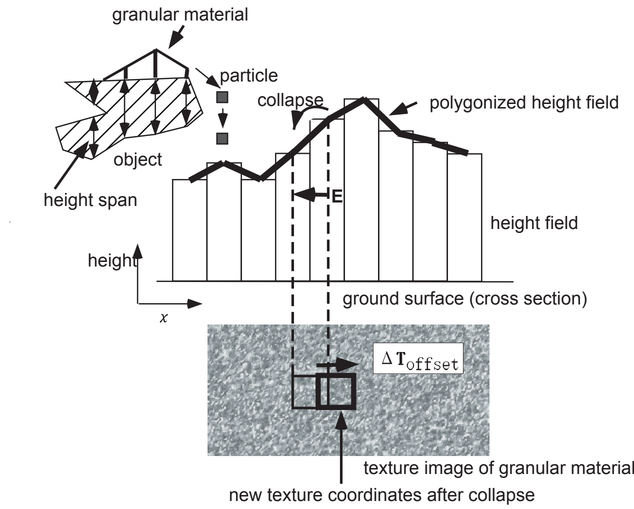


Figure 4. Texture sliding.

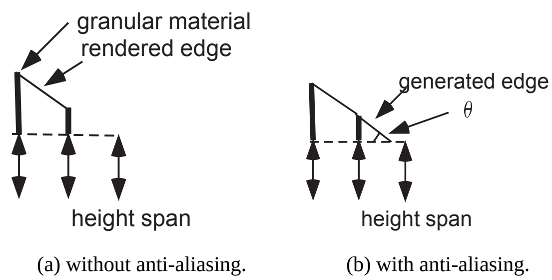


Figure 5. Anti-aliasing by linear interepolation.

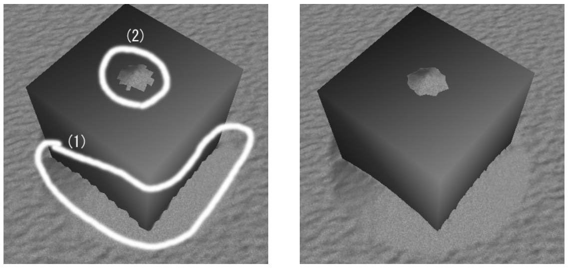
We use linear interpolation at the boundaries between granular material and an object where the granular material is on the object (the region (2) of Fig. 6(a)). Vertices are inserted between spans with granular material and spans without them. The positions of the inserted vertices are determined from θ as shown in Fig. 5.

The effect of anti-aliasing is shown in Fig. 6. A reasonable image quality of rendered result is achieved by using the anti-aliasing method.

5 Results

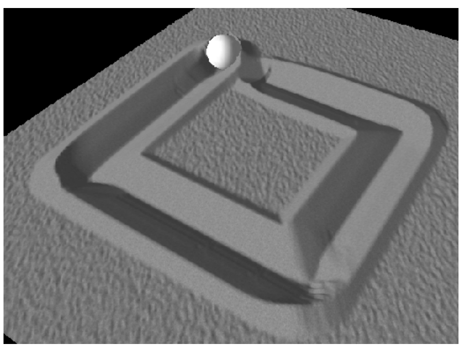
Fig. 7 compares the result of moving the sphere on the sandy surface with a photograph. This example shows that our algorithm can generate realistic marks on the ground surface.

Fig. 9 shows sandy surfaces deformed by a torus with various motions. Fig. 9(a) is a result of burying the torus

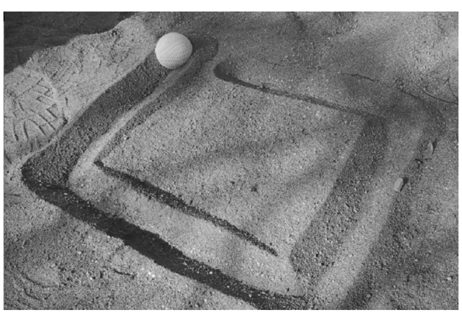


(a) without anti-aliasing. (b) with anti-aliasing.

Figure 6. Examples showing the anti-aliasing effect.



(a) result of the proposed method.



(b) photograph

Figure 7. Marks created by the moving sphere compared with the photograph.

in sand. Then the torus is taken up as shown in Fig. 9(b). Particles falling down from the torus are displayed in this image. Fig. 9(c) shows a result of dragging the torus. Sand passing through the hole of the torus leave such a mark displayed in this image. The proposed algorithm can handle vaious types of object motions as shown in these results.

Ground surfaces of sand and soil deformed by a foot are displayed in Fig. 10. The values of the angle of repose parameter θ are $\pi/6$ for sand and $5\pi/18$ for soil, respectively. Therefore, merely changing these parameters can represent different types of ground surfaces.

In order to show that our algorithm is useful for virtual reality systems, results of manipulating a bucket on sand are shown in Fig. 11. The bucket is dragged (Fig. 11(a)), burried in sand (Fig. 11(b)), and dropping sand (Fig. 11(c)). These results have been made interactively. The height field of 256×256 resolution and HS Map of 30×30 resolution are used, and the bucket consists of about 4,000 triangles. The frame rate varies with the amount of moving granular material. In this case, the frame rate varies from 7 to 14. The calculation times were measured by using a Dell Dimension8250 computer with an Intel Pentium4 3GHz CPU, and a Radeon 9700 Pro GPU.

More complex examples are shown in Figs. 12 and 13. Fig. 12 shows a result of lifting the letters “HENRY” burried in sand. There is sand on each letter and particles are falling from each letters. Fig. 13 shows the mark created by placing “HENRY” on sand.

6 Conclusion and Future Work

This paper has presented a deformation algorithm for a ground surface in contact with objects. We have proposed an efficient and intuitive deformation algorithm that can handle various types of object motions such as dragging/falling on the ground and scooping/dropping the granular materials. The deformation has been done by calculating the following three steps: (1) the collision detection with objects and the ground surface, (2) displacing granular material, and (3) calculating the erosion. The collapse of granular materials on the objects of various shapes including concave polyhedron have been represented by using an HS Map. We have represented the granular material as moving on the ground surface realistically by the texture sliding technique. Moreover we have proposed an anti-aliasing technique using virtual spans. This gives a reasonable rendering of the results at the interactive frame rate.

Future work may include the followings. The proposed method works well if objects move relatively slowly, but if objects move fast, dynamic effects such as spattering sand will be necessary. The motion of the objects is not affected by the ground in our method. If we consider the affection, we will be able to automatically make scenes such that a bucket free-falls and rolls on sandy surface or a ball begin to roll on sandy surface if collapse occured around the ball.

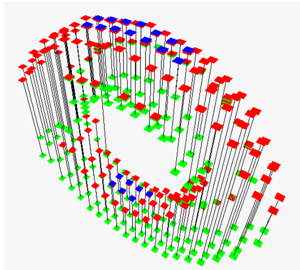
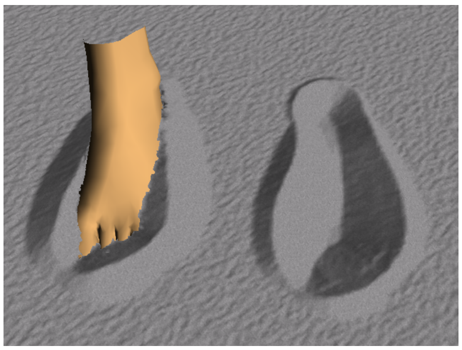


Figure 8. HS Map of torus.



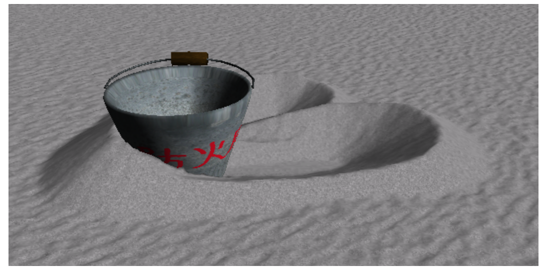
(a) footprints on sand.

(b) footprints on soil.

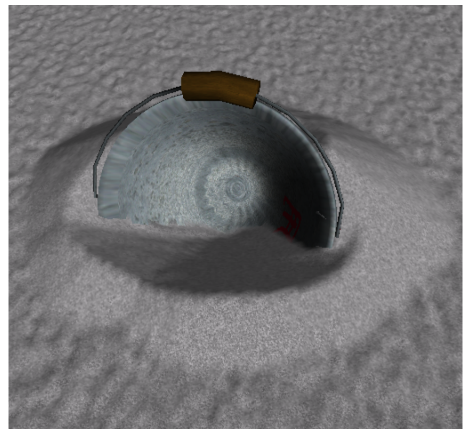
Figure 10. Marks created by a foot.

Acknowledgements

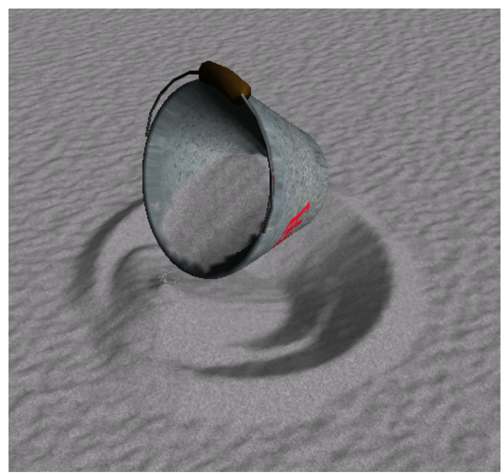
We would like to thank Professor Nelson Max (Lawrence Livermore National Laboratory) for helpful comments and useful discussions.



(a) dragging a bucket.

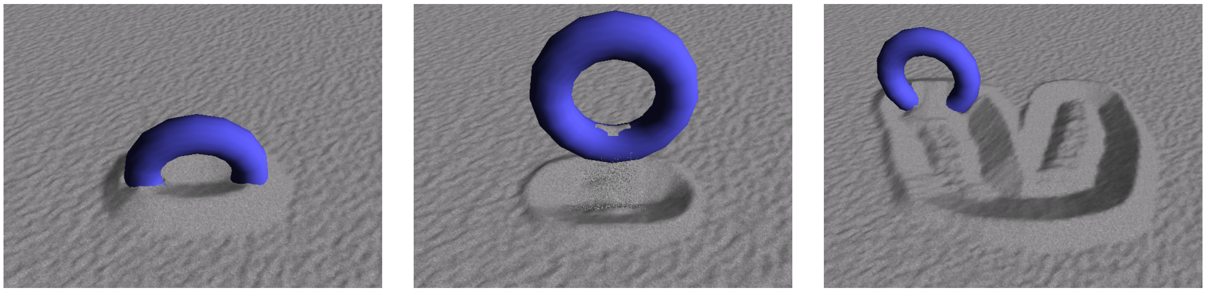


(b) a bucket burried in sand.



(c) a bucket is dropping sand.

Figure 11. Results of manipulating a bucket on sand.



(a) torus burried in sand. (b) torus lifted. (c) torus dragged.

Figure 9. Sandy surface deformed by a torus with various motions.

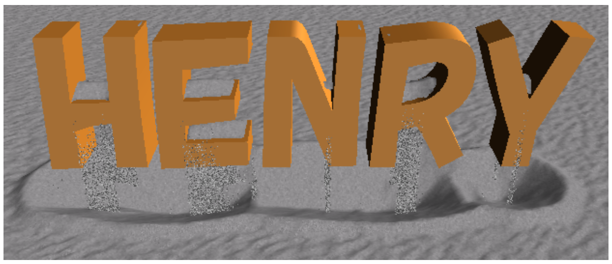


Figure 12. "HENRY" lifted from sand.

Figure 13. A mark created by placing "HENRY" on sand.

References

[1] B. Benes and R. Forsbach. Layered data representation for visual simulation of terrain erosion. *SCCG '01*, pages 80–85, 2001.

[2] B. Chanclou, A. Luciani, and A. Habibi. Physical Models of Loose Soils Dynamically Marked by a Moving Object. *Computer Animation '96*, pages 27–35, 1996.

[3] P. Fearing. Computer modelling of fallen snow. *ACM Computer Graphics (SIGGRAPH '00 Proceedings)*, pages 37–46, 2000.

[4] T. A. Galyean and J. F. Hughes. Sculpting: An interactive volumetric modeling technique. *ACM Computer Graphics (SIGGRAPH '91 Proceedings)*, pages 267–274, July 1991.

[5] X. Li and J. M. Moshell. Modeling soil: Realtime dynamic models for soil slippage and manipulation. *ACM Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 361–368, Aug. 1993.

[6] M. Matsumiya, H. Takemura, and N. Yokoya. A virtual clay modeling system for 3d free-form design using implicit surfaces. *IPSJ Journal*, 42(5):1151–1160, 2001.

[7] N. Max and B. Becker. Flow visualization using moving textures. In *Proceedings of the ICASW/LaRC Symposium on Visualizing Time-Varying Data*, Sept. 1995.

[8] S. Mizuno, M. Okada, and J. Toriwaki. An interactive designing system with virtual sculpting and virtual woodcut printing. *Computer Graphics Forum*, 18(3):183–193, Sept. 1999.

[9] F. K. Musgrave, C. E. Kolb, and R. S. Mice. The synthesis and rendering of fractal terrains. *ACM Computer Graphics (SIGGRAPH '89 Proceedings)*, pages 41–50, July 1989.

[10] T. Nishita, H. Iwasaki, Y. Dobashi, and E. Nakamae. A modeling and rendering method for snow by using metaballs. *Computer Graphics Forum*, 16(3):357–364, Aug. 1997. Proceedings of Eurographics '97. ISSN 1067-7055.

[11] Y. Oda, K. Muraoka, and N. Chiba. Particle-based visual simulation of virtual clay. *IPSJ Journal*, 42(5):1142–1150, 2001.

[12] K. Onoue and T. Nishita. A method for modeling and rendering dunes with wind-ripples. *Pacific Graphics*, pages 427–428, Oct. 2000.

[13] K.-L. Perng, W.-T. Wang, M. Flanagan, and M. Ouhyoung. A real-tme 3d virtual sculpting tool based on modified marching cubes. *ICAT '01*, pages 64–72, 2001.

[14] J. W. Shade, S. J. Gortler, L. He, and R. Szeliski. Layered depth images. *ACM Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 231–242, July 1998.

[15] R. W. Sumner, J. F. O'Brien, and J. K. Hodgins. Animating sand, mud, and snow. *Computer Graphics Forum*, 18(1):3–15, 1999.