A Low-Cost Solution For Excavator Simulation With Realistic Visual Effect

Ni Tao
Virtual System Laboratory
Gifu University
Gifu, Japan
ture_nitao@163.com

Zhao DingXuan
College of Mechanical Science and Engineering
Jilin University
Changchun, China
zdx@mail.jlu.edu.cn

Hironao Yamada
Department of Mechanical & Systems Engineering
Gifu University
Gifu, Japan
yamada@gifu-u.ac.jp

Ni Shui
Translation and Promotion Center
Dongying Science & Technology Bureau
Dongying, China
nishui_2000@163.com

***Abstract*—A low-cost excavator simulator has been developed in this paper for training human operators and evaluating control strategies for heavy-duty hydraulic machines. In such a system, the operator controls a virtual excavator by means of a joystick while experiencing realistic operating feelings through force feedback, graphical displays, and sound effects in virtual operating environments. A GPU (graphics processing unit)-based, terrain-rendering methodology is adopted to create a realistic visual appearance of different operation regions on terrain such as excavated areas and dumped soil during the excavator simulation. The kinematic relationships of the excavator's rotary joints are described, and the 3D coordinate of the bucket tip are calculated as the basis of a dynamic interaction between the excavator and the virtual terrain. A simplified mathematical model of the excavator's digging volume is proposed, and a surface of excavated soil is constructed using the Bezier patch and is deformed by manipulating its control points along a pre-defined path so as to provide visually meaningful soil bucket interaction information to the simulator.**

***Keywords*— GPU-based, terrain deformation, construction operation, height map, bezier patch**

I. INTRODUCTION

Virtual reality (VR) equipment simulators have been gaining acceptance since they were introduced in the military, aerospace, automobile and ship industries as tools, not only for reducing the high cost of product design and training, but also for real-time decision-making and process evaluation of equipment operation [1-3]. At the same time, VR assisted with tele-operation has been another research hotspot in recent years. One of the most distinctive reasons for the popularity of these simulators is their capability of transferring 3D high quality, graphical, machine-operating information to users in real-time and in an intuitive way [4].

Most construction vehicle manufacturers in the advanced countries have designed and utilized their own heavy construction machinery simulators for the virtual testing of newer models. These companies have developed virtual operation environments and specific topographical models for terrain tests using high performance parallel-processed computers or a group of server computers for real-time graphics and dynamics. More than three image generators are used for high-resolution graphic projection [5].

The general objective of this project is to develop a real-time simulator in order to train workers in the safe use of and work with construction machinery, such as an excavator, because any failure in their operation could cause major damage to people, the work and equipment. The project deals with the lack of training tools and methodologies for the wide range of tasks that are performed by construction workers. Besides its training purpose, the simulator has become essential equipment in the development of new excavator models and the evaluation of new design variables.

Previous efforts in VR excavating machine simulator development, however, showed a lack of balance between the fidelity of the model to the physics and the visual representation of the simulated equipment [4]. In this paper, a GPU-based terrain-rendering technique is used to provide convincing graphic operating results for users. Also, a profile of virtual excavated clay in a bucket changes dynamically in real time based on a simplified mathematical model of the excavator digging and provides physically meaningful soil bucket interaction information to the simulator.

II. SYSTEM OF EXCAVATOR SIMULATION

In such an excavator simulation system, the user is surrounded by stereoscopic computer images rendered on three individual displays at an angle of 120° to each other. Two force-feedback joysticks ("Side Winder Force Feedback 2," Microsoft Co., Ltd.) are used in this system as the control devices. The first joystick is used to drive the boom and swing

978-1-4244-1676-9/08 /$25.00 ©2008 IEEE

joints while the second is used to move the stick and bucket joints. A DC motor is connected under the joystick, and the reaction between the virtual excavator's bucket and the terrain is fed back to the operator through a speed change gear connected to the motor. The open development tool "DirectInput" (which is included in "Microsoft DirectX") is used for programming APIs for the joystick and enables the displacement of the joystick and the force to be displayed to the operator at a high speed (maximum sampling frequency: 500 Hz) [6].

Each joystick operates on an X and Y axis. The four angular displacements of the two joysticks are mapped onto the linear displacement of the virtual excavator's hydraulic cylinder for the swing, boom, stick and bucket. By the forward kinematics calculation of every joint, the input angular position of the joystick corresponds accordingly to single poses of the excavator. Thus, the user can manipulate the virtual excavator with the interface of the joystick under the guidance of 3D graphics, sounds and the force feedback, as Fig. 1 shows. The surrounding display system provides the operator with a wide field of view. Currently, a common primary graphic card can at most drive two displays simultaneously, and therefore two graphic cards are plugged in the computer for the excavator simulation system.

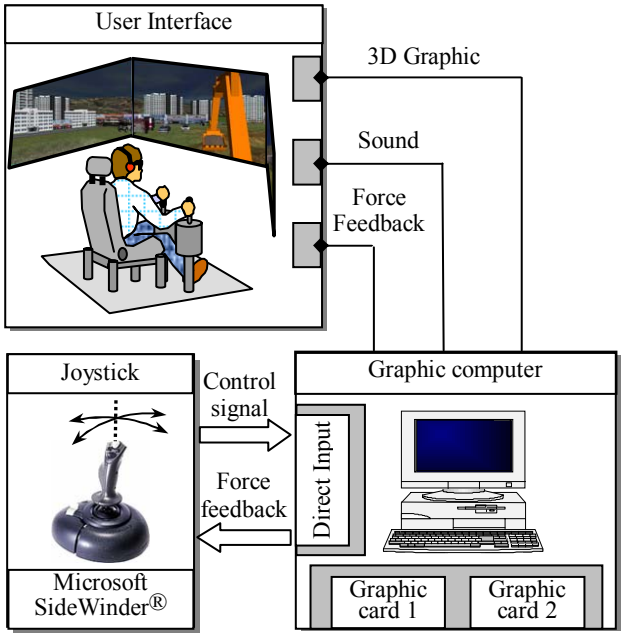


Figure 1. System composition

The graphic module for the excavator simulator has been developed utilizing the open source OpenGL SDK in a Visual C++ programming environment. The Open Producer library is used as the toolkit to allow multiple cameras to run in parallel supporting the hardware configurations with multiple display subsystems [7]. Considering the complexity of building and maintaining the relationships between all the elements of the virtual excavator and terrain, a scene-graph representation is adopted, in which each graphic entity (such as the excavator, dynamic deformable terrain, static terrain and light, etc.) is attached to a node in a tree structure. All the nodes in the scene are traversed and updated in every frame.

III. RENDERING THE GPU-BASED TERRAIN

To render a ground surface, it is often practical to only store samples of the ground height function at some intervals along the X and Z axes. A set of such samples for an area is called an elevation or height map [8]. The height map is generally encoded in colors and stored as a grayscale image, in which low intensities represent low-lying land and bright intensities represent high land. Using an elevation grid makes it easy to put the excavator on the virtual terrain. As the excavator drives over the terrain, the appropriate position and orientation of the excavator is computed depending on the elevation of the grid at the center and the four corners of the excavator tracks.

A number of techniques have been proposed to render terrains, such as Real-time Optimally Adapting Meshes (ROAM) [9], Quad Trees [10], and Triangular Irregular Networks (TINs) [11]. However, the most straightforward and accurate approach for rendering terrain in small portions is to use the "brute force" method [12], in which every height value in the height map is used to form a terrain mesh. This involves a one-to-one mapping between the height map values and terrain vertices and allows the terrain to be rendered in maximum detail, which is more suitable for exact operation of the excavator simulation. An elevation grid with a smaller individual span will yield a better visual appearance of the terrain deformation for digging and dumping dirt by the excavator.

Although a "brute force" algorithm provides a satisfying visual effect for terrain rendering, it is implemented via a large computation. In order to achieve a considerable speedup and ensure the rendering of terrain in real-time, GPU–based techniques are applied in this paper.

All of today's commodity GPUs structure their graphic computation in a similar organization called the graphic pipeline. This pipeline is designed to allow hardware implementations to maintain high computation rates through parallel execution. Over the past several years, graphics vendors have moved away from the fixed-function pipeline into a more flexible programmable pipeline. This recent evolution of GPUs affords us the opportunity to directly control powerful graphic hardware [13,14]. In the programmable pipeline, fixed-function operations, including transformations and lighting calculations, are replaced with a user-defined vertex and fragment program. Cg (C for Graphic) and GLSL (OpenGL Shading Language) both abstract the capabilities of the underlying GPU and allow the programmer to write GPU programs in a more familiar C-like programming language[15]. In this paper, the terrain contains 256 × 256 vertices for the excavation field. All of these vertices are stored in the video memory of the graphics card as Vertex Buffer Objects [16].

A 3D polygon mesh constructed from the height map takes only the shape of a patch of terrain. In order for this mesh to "look" like a real terrain and to enhance its realistic appearance , a texture composition technique is used in a terrain-shading model. Texture composition requires separate textures to represent different earth materials such as grass and dirt. In the processing step of the programmable pipeline, a blended texture is generated in the video memory, and the output image

appears as a banded texture with a smooth cross-fade between neighbouring textures.

$$DirtGrass = T_{grass} * w + T_{Dirt} * (1 - w) \quad (1)$$

$$\begin{aligned} Out &= \lambda_m \lambda_a \lambda_b * DirtGrass \\ &+ (1-\lambda_a) * [(1-\lambda_m) * DirtGrass + \lambda_m * T_{Dig}] \\ &+ (1-\lambda_b) * [(1-\lambda_m) * DirtGrass + \lambda_m * T_{Dump}] \end{aligned} \quad (2)$$

Here, *DirtGrass* represent output texture of normal terrain, i.e., without digging or dumping operation. *Out* represent the output terrain texture of construction areas. T_{dirt}, T_{grass}, T_{dig} and T_{dump} represent the texture of dirt, the grass, the excavated hole and the dumped soil respectively, and w is a parameter related to the height of the terrain used to produce a smooth transition effect between different textures. And $\lambda(\lambda_m, \lambda_a, \lambda_b)$ is the user-defined vector used for differentiating different districts of the terrain, such as the digging and dumping regions as well as normal areas.

TABLE I. GLSL PARAMETERS UNDER DIFFERENT CONDITION

Condit -ion	GLSL Parameters			
	λ_m	λ_a	λ_b	*Output*
normal	1	1	1	*DirtGrass*
dig	0~1	0	1	$(1-\lambda_m)*DirtGrass+\lambda_m*T_{Dig}$
dump	0~1	1	0	$(1-\lambda_m)*DirtGrass+\lambda_m*T_{Dump}$

At runtime, not only the position, the normal and the texture color of each vertex in the virtual terrain, but also an additional user-defined vector λ (λ_m, λ_a, λ_b) are passed from the Windows application to the GPU's programmable pipeline. In the vector λ, λ_m is a parameter that relates changes in the elevation of the terrain vertex by values of a float type ranging from 0 to 1. It is used for producing a smooth transition between the normal terrain texture *DirtGrass* and the texture of operation fields, such as digging and dumping regions. λ_a and λ_b are two boolean variables.

Each vertex in the terrain model is evaluated, and the texture coordinates are queried. Equation (1) is used to calculate a combination of the dirt and grass textures of the terrain region. Equation (2) is used to calculate the final output color of operation fields including digging and dumping regions for every vertex in the GPU's fragment program. As shown in Table I, under normal conditions, vector λ is set to (1,1,1), which implies the appearance of virtual terrain and which takes on the color of the variable *DirtGrass*. In the region of the excavated terrain, the vector λ is set to (λ_m,0,1), and therefore the final colour of this region is affected by the texture T_{dig} . While in the fields of dumped soil, the vector λ is set to (λ_m,1,0), and texture T_{dig} becomes the primary output color.

IV. THE KINEMATIC MODEL OF EXCAVATOR

The purpose of establishing the kinematic model of excavator is to transfer the bucket tip's reference trajectories to a corresponding, required reference angle sequence for each joint and to get motion sequences of hydraulic cylinders[17] (the basis for the graphic simulation) and to perform collision detection for the virtual excavator.

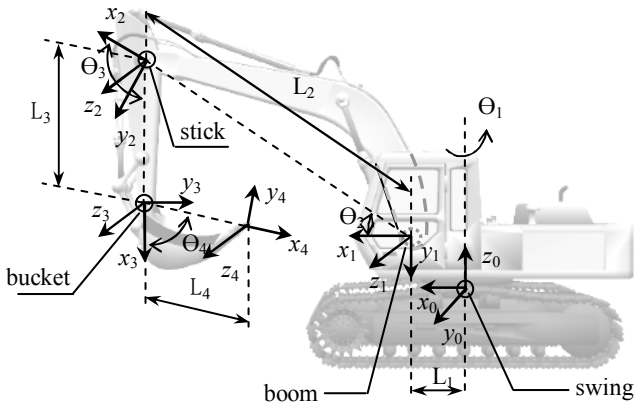


Figure 2. Definition of excavator coordinates

Generally, as Fig.2 shows, pieces of construction excavator equipment can be thought of as series-wound hierarchies, which are made up of a slewing cabin, boom, stick and a backhoe consisting of a bucket, each being actuated by a hydraulic cylinder. During excavation, the swing joint is held fixed, and the other three links move in a plane. The position of the bucket tip is derived from the orientation of boom, stick and bucket with respect to each other. That is, if one wishes to drive the bucket in a straight horizontal line, keeping it in a constant orientation, one must operate all three actuators in a rather complex way. This is determined by means of their respective joint angles. Because the excavator parameters of each component are defined in local space, the digging implement's position and orientation in world coordinates has to be determined so as to enable a dynamic interaction between the bucket on the backhoe and the virtual terrain.

In order to do this effectively while describing the position and orientation of the bucket, the Denavit-Hartenberg (D-H) approach based on homogenous coordinates is utilized for the forward kinematic equations calculation[18], which is useful for rigid robots consisting of an open chain.

To perform the kinematic analysis, we rigidly attach a coordinate frame to each link. In particular, we attach $o_i x_i y_i z_i$ to link i. This means that whatever motion the excavator executes, the coordinates of each point on link i are constant when expressed in the i^{th} coordinate frame. Furthermore, when joint i is actuated, link i and its attached frame, $o_i x_i y_i z_i$, experience a resulting motion. The base coordinate frame {0} is defined by the position of the cabin's swing joint; the endmost coordinate frame {4} is defined at the tip of the backhoe's bucket; and coordinate frame {1},{2},{3} is attached to the rotary joint of the excavator's boom, stick and bucket, respectively. The axes x of every coordinate is along the direction of the current coordinate's origin and previous coordinate's origin. And the axes z is defined as being perpendicular to the rotary plane of every joint. The axes y is the cross product result of axes x and z satisfying right hand rules.

Based on the D-H homogeneous transformation matrix, we would like to find the pose of the excavator bucket at given joint angles. The 3D coordinate of the bucket tip relative to the position and orientation of the excavator's slewing cabin in world space can be described by 4×4 matrices ${}^0T_4$.

$$ {}^0T_4 = \begin{bmatrix} C_1C_{234} & -C_1S_{234} & S_1 & C_1(L_1+L_2C_2+L_3C_{23}+L_4C_{234}) \\ S_1C_{234} & -S_1S_{234} & -C_1 & S_1(L_1+L_2C_2+L_3C_{23}+L_4C_{234}) \\ S_{234} & C_{234} & 0 & L_2S_2+L_3S_{23}+L_4S_{234} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3) $$

Here, $S_i = \sin\theta_i$, $C_i = \cos\theta_i$, $S_{ij} = \sin(\theta_i + \theta_j)$, $C_{ij} = \cos(\theta_i + \theta_j)$, and $i = 1,2,3,4$.

L_i is the distance between the axes z_{i-1} and z_i measured along the axes x_i. θ_i represents the joint angle between the axes x_{i-1} and x_i measured in a plane normal to the axes z_{i-1}.

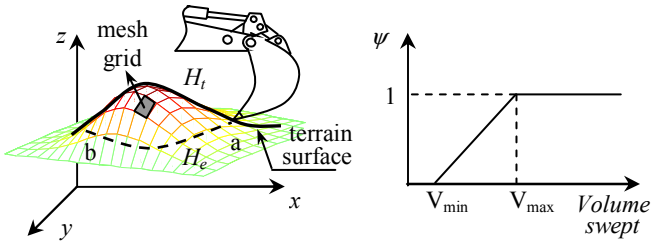
Key points are chosen on the digging implement and their world coordinates are calculated by multiplication their local coordinates with the matrix of ${}^0T_4$. By comparing the positions of those key points with elevations of the terrain at corresponding projected locations, the state of collision between the virtual excavator and terrain could be determined.

V. DYNAMATIC INTERACTION BETWEEN VIRTUAL EXCAVATOR AND TERRAIN

The usual task of a backhoe excavator is to loosen and remove material from its original location and transfer it to another location by lowering the bucket, digging by dragging the bucket through the soil, and then lifting, slewing, and dumping the bucket load [19]. Several common construction tasks such as excavating, filling and dumping are performed with the deliberate intention of altering a jobsite topography. Accurate visualization of the evolving jobsite terrain is, thus, of significant importance to the accurate portrayal and credibility of several animated construction processes.

During the animation of the digging stroke, the position of the virtual excavator's bucket edge is compared with the elevation in the virtual terrain model at the projected locations and monitored continuously to detect whether or not the virtual terrain has been penetrated. If the elevation of terrain at those points is greater than the digging implement's current height, then it is concluded that the implement has penetrated the terrain surface. As a result, the elevation value of the terrain model at the penetrated places is then set equal to the digging implement's current height. The outcome of continuously performing this computation and updating the terrain database during visualization is a terrain deformation whose shape conforms to the digging implement's trajectory during the particular digging stoke [20]. Similarly, deposition of dirt is computed and depicted during visualization of the dumping stokes. At each instant, the position of the excavator bucket is monitored during dumping. The elevation of the projected terrain points along the trajectory followed by the involved bucket and is then increased as it dumps soil. The visual outcome in this case is the depiction of a heap whose size is proportional to the virtual excavator bucket's dumping soil.

One of the requirements for VR excavating machine simulators in order for them to be a convincing engineering tool is that the interaction between the excavating bucket and soil should be understood clearly and represented as a mathematical model so that a mesh animation of excavated clay can be performed. In this paper, we are not aiming to develop a complex bucket-ground interaction-force model [4,21], but only a real-time simplified mathematical model of an excavator's digging volume from the visual aspect of view.



(a) Digging operation (b) Swept volume model

Figure 3. Principle for excavator swept terrain volume calculation

The principle for the calculation of the excavator bucket swept terrain volume is shown in Fig. 3. In Fig. 3(a), x, y, and z are axes of 3D coordinates in a virtual world. Imagine that at a certain location on the x-y plane, the original elevation of the virtual terrain mesh grid is H_t, and the corresponding trajectory height of the excavator bucket during its digging stoke at that position is H_e. Vol is the terrain volume swept by the excavator implement's bucket. Thus, Vol can be computed as following:

$$ Vol = \iint_{D_{ab}} \kappa(x,y) * (H_t - H_e) d_x d_y \quad (4) $$

where D_{ab} is the double integral region of the excavating area projected in an x-y plane, and κ is a mark with a boolean value of 0 or 1, representing whether the mesh grid of terrain at the excavating region has been penetrated or not. κ is determined by the relationship between the heights of virtual excavator bucket edge and the elevations of virtual terrain at the corresponding (x,y) location, as (5) shows.

$$ \kappa(x,y) = \begin{cases} 1 & (H_t > H_e) \\ 0 & (H_t \le H_e) \end{cases} \quad (5) $$

$$ \psi = \begin{cases} 0 & (Vol < V_{\min}) \\ Vol/V_{\max} & (V_{\min} \le Vol \le V_{\max}) \\ 1 & (Vol > V_{\max}) \end{cases} \quad (6) $$

In Fig. 3(b), V_{min} and V_{max} are the minimum and maximum values of the virtual excavator bucket's volume threshold. As (6) shows, ψ is the normalized value of the volume swept, which is simply the ratio of the volume of material in the bucket to the maximum bucket capacity. When the bucket sweeps less than V_{min}, ψ is set to 0, i.e., all the actions that sweep less than V_{min} are vetoed. Above V_{max}, ψ is constant because V_{max} represents the bucket capacity, and no additional value is attached to sweeping more material.

VI. DEFORMATION OF SWEPT CLAY

During the simulation of the digging stoke, it is not sufficient just to handle static models since the shape of the clay changes in accompanying the swept volume. The method of Smoothed Particle Hydrodynamics (SPH) [22] can be used to simulate deformable granular materials such as sand and soil. However, because in physics it is built on the basis of a large amount of particles, it occupies too much GPUs resources and is not very fit for such a real-time excavator simulation. In this paper, the emphasis is not on the precise physical simulation of the deformations, but in producing visually appealing and realistic looking results at a low performance cost.

The Bezier patch is adopted for surface representation of dug clay during the simulation of the excavator backhoe's excavating operation. The patch is constructed from an n×m (currently 4×4) array of control points and the resulting surface, which is now parameterized by two variables. It is given by the following equation:

$$P(u,v)=\sum_{j=0}^{3}\sum_{i=0}^{3}P_{i,j}B_{i,3}(u)B_{j,3}(v) \tag{7}$$

Here, $B_{i,n}(t)=\dfrac{t^i(1-t)^{n-i}\cdot n!}{i!(n-i)!}$, $P_{i,j}$ are the control points defining the control polygon, and i,j=0,1,2,3.

The matrix format of $P(u,v)$ can be written as follows:

$$P(u,v)=\begin{bmatrix}1 & u & u^2 & u^3\end{bmatrix}M\begin{bmatrix}P_{0,0} & P_{0,1} & P_{0,2} & P_{0,3}\\ P_{1,0} & P_{1,1} & P_{1,2} & P_{1,3}\\ P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3}\\ P_{3,0} & P_{3,1} & P_{3,2} & P_{3,3}\end{bmatrix}M^T\begin{bmatrix}1\\ v\\ v^2\\ v^3\end{bmatrix} \tag{8}$$

Here $M=\begin{bmatrix}1 & 0 & 0 & 0\\ -3 & 3 & 0 & 0\\ 3 & -6 & 3 & 0\\ -1 & 3 & -3 & 1\end{bmatrix}$, and $0\le u\le 1$, and $0\le v\le 1$

As seen from (8), the surface of the Bezier patch is determined by the position of the control points $P_{i,j}$. Therefore, manipulating individual control points can deform the surface of the clay according to the normalized swept volume value ψ of the excavator bucket. This process is shown in Fig. 4.

First, several key positions of the control points that determine the profile of the excavated clay surface are selected beforehand. These key positions include the start positions (S_{ij}), the middle inflexion positions (M_{ij}), and the end destination positions (E_{ij}). Second, we put the control points $P_{i,j}$ at the position of S_{ij} when a digging operation starts. Third, according to the excavator bucket's swept volume ψ, we divide the motion path for the control points into two segments by a constant pre-defined value of ψ_m. If $0<\psi\leqslant\psi_m$, the control points $P_{i,j}$ move from S_{ij} to M_{ij}; if $\psi_m<\psi\leqslant 1$, $P_{i,j}$ move from M_{ij} to E_{jj}. That is, ψ with a value of 0 corresponds to control points with a position of S_{ij}; ψ with a value of ψ_m corresponds to the position of M_{ij}, and ψ with a value of 1 corresponds to the position of E_{ij}. Linear interpolation is used for calculating the middle position of the control points between S_{ij} and M_{ij} and M_{ij} and E_{ij}. As a result, by manipulating the control points to move along their corresponding path, the surface of excavated clay could produce a deformation effect in real time. Fig. 5 shows the surface of the excavated clay with a different bucket swept volume ψ_m and 1.

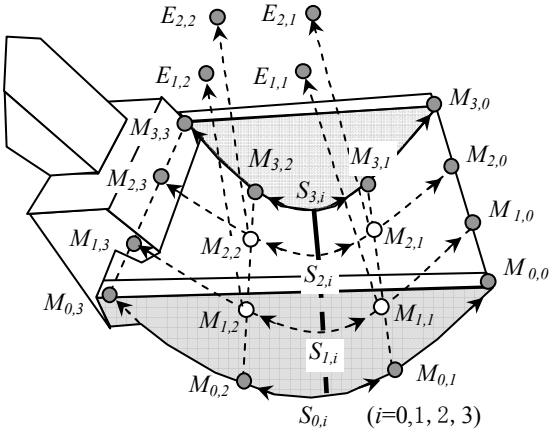
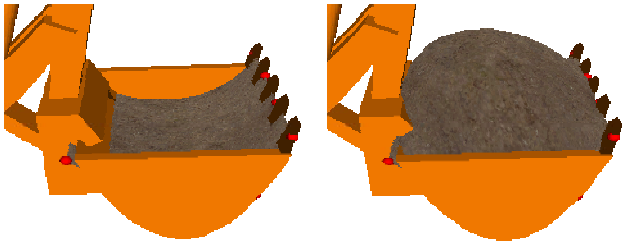


Figure 4. Motion path of control points for dug clay surface



(a) $\Psi=\Psi_m$ (b) $\Psi=1$

Figure 5. Surface of dug clay with different values of Ψ

VII. SIMULATION RESULTS AND CONCLUSIONS

In this paper, a low-cost, PC-based virtual excavator simulator has been developed for the purpose of operator training and evaluation. The specifications of the computer used in the excavator simulation can be found in Table Ⅱ.

TABLE II. CONFIGURATION OF COMPUTER

Processor	Intel Core™ 2 CPU, 6400, 2.13GHz
RAM	2GB
Video card	2×NVIDIA GeForce 7900GS, PCI-Express
System	Windows XP SP2

The final rendered image (Fig. 6) displays a dynamic deformation of the terrain model in an operation jobsite during construction processes. The blending effects among different textures (digging and dumping dirt and grass) enrich the scene with visual cues and detail that improve the visual quality of the image. The results are very promising since frame rates for

the solution were maintained well above the 60 per second required of most real-time graphic applications.

(a) Digging operation

(b) Dumping operation

Figure 6. Screen for excavator simulation

At the same time, the key techniques involved, such as real-time mesh deformation and GPUs-based realistic rendering, not only provide satisfying solutions for excavator simulation, but also fit for some other 3D virtual reality fields, including vehicle driving simulation and a virtual battlefield and navigational simulation. In addition to this, we have included a kinematic model for a series-wound, joint-type construction manipulator with 4 degrees of freedom, which is the basis of controller designing and excavation trajectory planning for the excavator.

REFERENCES

[1] Millheim K. K., "The role of simulator in drilling operations," SPE Drilling Engineering 11170, pp. 347-357, 1986, in press.

[2] Yamada H., and Muto T., "Using virtual reality to assess factors affecting shipboard accessibility for wheelchair users," International Journal of Control and Intelligent Systems, vol. 32, No. 1, pp. 52-57, 2004, in press.

[3] Yamada H., and Muto T., "Development of a hydraulic tele-operated construction robot using virtual reality—new master-slave control method and an evaluation of a visual feedback system," International Journal of Fluid Power, vol. 4, No. 2, pp. 35-42, 2003, in press.

[4] Borinara Park, "Development of virtual reality excavator simulator: a mathematical model of excavator digging and calculation methodology," Doctor Dissertation, Virginia Polytechnic Institute and Sated University, 2002.

[5] Kwon Son, Sang-Hwa Goo, Kyung-Hyun Choi, Wan-Suk Yoo, Min-Cheol Lee, and Jang-Myung Lee, "A driving simulator of construction vehicles," International Journal of the Korean Society of Precision Engineering, vol. 2, No. 4, pp. 12-13, 2001, in press.

[6] Yamada H., Mingde G., and Dingxuan Z., "Master-slave control for constructiong robot teleoperation," Journal of Robotics and Mechatronics, vol. 19, No. 1, pp. 60-67, 2007, in press.

[7] http://www.andesengineering.com/Producer

[8] Hans Häggström, "Real-time generation and rendering of realistic landscapes," Master Dissertation, University of Helsinki, 2006.

[9] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, and M. B. Mineev-Weinstein, "ROAMing terrain: real-time optimally adapting meshes," Proceeding of the 8th Conference on Visualization. IEEE Computer Society Press, United States: Phoenix, Arizona, pp. 81-88, 1997, in press.

[10] Renato Pajarola, "Overview of quadtree-based terrain triangulation and visualization," UCI-ICS Technical Report, No. 02-01, University of California Irvine, 2002.

[11] Hugues Hoppe, "Smooth view-dependent level-of-detail control and its application to terrain rendering," IEEE Visualization 1998, pp. 35-42, 1998.

[12] Trent Polack. "Focus on 3D terrain programming," Course Technology, 2002.

[13] Dave Baldwin, Randi J. Rost, and John Kessenich, "The openGL shading language," Version 1.05, 3Dlabs Inc, 2003.

[14] Marc Olana, "A programmable pipeline for graphics hardware," PhD Dissertation, University of North Carolina at Chapel Hill, Department of Computer Science, 1998.

[15] Serpil Tokdemir, "Digital compression on GPU," Master Dissertation, Georgia State University, College of Arts and Sciences, 2006.

[16] NVIDIA Corporation, "Using vertex buffer objects," White Paper, 2003.

[17] Daqing Zhang, Qinghua He, Peng Hao, and HaiTao Zhang, "Modelling and controlling for hydraulic excavator's arm," 22nd International Symposium on Automation and Robotcs in Construction, ISARC 2005, pp. 2-3, 2005, in press.

[18] http://www4.cs.umanitoba.ca/~jacky/Teaching/Courses/74.795-Humanoid-Robotics/ReadingList/chap3-forward-kinematics.pdf

[19] Quang Ha, Miguel Santos, Quang Nguyen, David Rye, and Hugh Durrant-Whyte, "Robotic excavation in construction automation," IEEE Robotics & Automation Magazine, pp. 20-28, 2002, in press.

[20] Vineet R. Kamat, M.ASCE, and Julio C. Martinez, M.ASCE, "Large-scale dynamic terrain in three-dimensional construction process visualizations," Journal of Computing in Civil Engineering, pp. 160-171, 2005, in press.

[21] DiMaio S.P., Salcudean S.E., Reboulet C., Tafazoli S., and Hashtrudi-Zaad K., "A virtual excavator for controller development and evaluation," Robotics and Automation, vol. 1, pp. 52-58, 1998, in press.

[22] Müller M., Charypar D., and Gross M., "Particle-based fluid simulation for interactive applications," Proceeding of the 2003 ACM SIGGRAPH Eurographics Symposium on Computer Animation, Eurographics Association, pp. 154-159, 2003, in press.