國立交通大學

多媒體工程研究所

碩 士 論 文

以 粒 子 為 基 礎 之 沙 雕 與 水 的 模 擬

Particle-Based fluid Simulation for Sand Sculpture and Water

研 究 生：滕薇鈞

指導教授：施仁忠　教授

中 華 民 國　九 十 九　年 六 月

以 粒 子 為 基 礎 之 沙 雕 與 水 的 模 擬

Particle-Based fluid Simulation for Sand Sculpture and Water

研　究　生：滕薇鈞	Student：Wei-Chun Teng
指導教授：施仁忠	Advisor：Zen-Chung Shih

國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年六月

以粒子為基礎之沙雕與水的模擬

研究生: 滕薇鈞　　　　　　　　　　指導教授: 施仁忠 教授

國立交通大學多媒體工程研究所

摘　　要

在這篇論文裡，我們提出了一個有效率的演算法模擬沙雕和水之間的互動。近年來，流體模擬在電腦動畫和遊戲裡有舉足輕重的地位，然而沙雕和水的互動始終是個複雜的問題。

我們提供了一個以流體力學爲基礎的方法，以粒子的觀點同時模擬沙子和水。由於沙子和水是由龐大數量的粒子所組成，本篇論文使用了適應取樣(Adaptive Sampling)演算法作爲基礎做更有效率的模擬。除此之外，爲了要模擬出水被沙子吸收的情況，在我們的系統裡面也考慮滲流(Porous Flow)的因素。如此一來，本篇論文可以完全結合沙子跟水的粒子。最後，本系統也會展示水和沙雕互動的效果。

Particle-Based Fluid Simulation for Sand Sculpture and Water

Student: Wei-Chun Teng Advisor: Dr. Zen-Chung Shih

Institute of Multimedia Engineering

National Chiao-Tung University

ABSTRACT

In this thesis, we present an effective algorithm to simulate the interaction between sand sculptures and water. Recently, fluids simulation is an important topic in computer animation and games. However, the interaction between sand sculptures and water is still a complex problem.

We propose a unified Smoothed Particle Hydrodynamics framework for simulating both fluids and sand particles. Since the sand and water are sampled by a large number of particles, we use an adaptive sampling algorithm to simulate particles more effectively. Moreover, we also consider the water absorption of sand. We incorporate the porous flow simulation to our framework. As a result, we can combine sand and water completely. Our system will show how the water collapse sand sculptures during the simulation.

Acknowledgements

First of all, I would like to express my sincere gratitude to my advisor, Pro. Zen-Chung Shih for his guidance and patience. Without his encouragement, I would not complete this thesis. Thank also to all the members in Computer Graphics and Virtual Reality Laboratory for their reinforcement and suggestion. I want to thank to all the people who ever supported me during these days. Finally, I will sincerely dedicate this thesis to my parents.

Contents

ABSTRACT (in Chinese)……....……………………………………………….............……….I

ABSTRACT (in English)……......……………………………………….............…………………II

ACKNOWLEDGEMENTS………………………………………………............………III

CONTENTS..……………………………………………………………….................IV

LIST OF FIGURES..V

CHAPTER 1 INTRODUCTION……………………………………………….................1

1.1 Motivation………………………………………………………….............…..1

1.2 System Overview………………………………………………..…......................2

CHAPTER 2 RELATED WORKS……………………………………………..............…3

2.1 Fluid Simulation…………………………………………………………….............3

2.2 Porous Flow Simulation…………………………………………………….................3

2.3 Granular Simulation..4

CHAPTER 3 WATER AND SAND INTERACTIVE MODEL..………………………..............5

3.1 Smoothed Particle Hydrodynamics (SPH) Model...……………………….................5

3.2 Fluid Simulation..………………………………...............6

3.3 Granular Simulation............……………………………………………..................7

3.4 Porous Flow Simulation……......……………………………………………...............11

3.5 Combine Difference Particles..15

3.6 Adaptive Sampling Algorithm..17

3.7 Summary of the Algorithm………………………………………………….............18

CHAPTER 4 IMPLEMENTATION AND RESULTS………………………………..….........20

CHAPTER 5 CONCLUSION AND FUTURE WORKS..……………..........……………......72

REFERENCE………………………………………………………………........………........73

Figure List

Figure 3.1: Particle p_i and particle p_j are neighbors of each other……………….................7

Figure 3.2: Sand particles interactive with water particles...……………………….............9

Figure 3.3: Algorithm for a step in SandSimulator..…....11

Figure 3.4: Porous materials.………..........…………………………............………....12

Figure 3.5: Actual fluid velocity through the pore space of sand materials........……..............14

Figure 3.6: Algorithm for a step in MixSimulator 16

Figure 3.7: Three different cases of how the mass transport from one particle to another…...16

Figure 3.8: Particles split and merge...……………………18

Figure 3.9: Simulation proceeds in our system.......................................……………………....19

Figure 4.1: Slicing the object to grids...………………………………21

Figure 4.2(a)-(s): Simulation scene 1......................................…………………………………22

Figure 4.3(a)-(z): Simulation scene 2……..……………………........32

Figure 4.4(a)-(z): Simulation scene 3...……………………………....45

Figure 4.5(a)-(z): Simulation scene 4………………….......................…………………......59

Chapter 1
Introduction

1.1 Motivation

Fluid simulation is now a popular topic in computer graphics animations and games. Physically-based simulation methods are usually used to model the behavior of fluids since fluids interacting with other objects or material is quite complex to handle. Recently, fluids animations have made great progress. However, the interaction between sand sculpture and water has still few discussions. Considering the interacting between sand and water, the sand can be regard as porous materials, which means that the sand will absorb water when water contacts to sand. Also, the water will affect the physical behavior of the sand. For example, imagine that you are creating the sand sculpture at the beach. Suddenly a wave hits the beach, causing the sand sculpture destroyed. It is a common phenomenon at the beach.

The goal of this thesis is to provide a system that can handle the mixed particles interactive simulation with efficient performance. The difficulty is how to combine difference kinds of materials together and then speed up.

1.2 System Overview

This thesis presents a way to combine various kinds of objects using adaptive sampling for particle-based fluid simulation. Our method bases on the Smoothed Particle Hydrodynamics (SPH) framework for the simulation of particles, such as water and sand. We implement the granular materials (e.g. sand) and fluids (e.g. water) separately, then mixing them together. The sand is based on the Zhu and Bridson's sand model, we will show that how to transfer this grid-based model to a particle-based model. The sand then extend by the porous flow simulation which can model the behavior of the mixed materials. Since the system is based on the adaptive sampling for particle-based fluid simulation, it can handle huge particles with higher speed than before.

Chapter 2
Related Works

2.1 Fluid Simulation

Up to now, fluid simulation is still a challenging topic in computer animation. The simulation of complex fluid is often based on Navier-Stokes equation. Foster and Metaxas [12, 13] were the first solving the full 3D Navier-Stokes equations by using a Marker-and-Cell (MAC) method on Eulerian grids. Stam [24] provided a stable model on grid for fluid simulation and allowed much larger time-step. Foster and Fedkiw [11] improved the MAC method by using level set methods to track the free surface of fluids. Génevaux et al. [14] combined the force between solids and fluids. Carlson et al. [5] provided a technique for two-way coupling between rigid bodies and fluids. Since they treat rigid objects as fluids, the method is called rigid fluid method.

The basic Smoothed Particle Hydrodynamics (SPH) model we used is based on the work of Müller et al. [19], which proposed an interactive system for water simulation.

2.2 Porous Flow Simulation

Lenaerts et al. [16] presented the simulation of a fluid flowing through rigid and elastic materials. They used physical principles of Darcy's Law and combined with SPH framework to simulate fluids. Their algorithm modeled the changing behavior of wet materials which is similar to our simulation.

2.3 Granular Simulation

Miller and Pearce [18] introduced sand animations based on particles. Carlson et al.[6] provided simulations of wet sand dripping by increased viscosity in fluids. Bell et al.[4] simulated granular materials using particles. They regarded granular material as a large collection of non-spherical particles. By using a particle-particle collision model, they can handle particles efficiently. Zhu and Bridson [26] provided a physically-based simulation method called Particle-In-Cell for animating sand. Our model of sand sculptures is based on the works of Bell et al. [4] and Zhu and Bridson [26].

In this thesis, we construct a system for simulating the interaction between sand sculptures and water. Our system is similar to Lenaerts and Dutré [17]. However, we use an adaptive particle sampling algorithm. So our system can handle huge amount of particles effectively.

Chapter 3
The Interaction Model

Our simulation framework is based on Smoothed Particle Hydrodynamics (SPH). In the following sections, we will describe how to simulate the interaction between water and sand sculptures. In Section 3.1, we will show the basic smoothed particle hydrodynamics (SPH) model. Then we demonstrate the fluid simulation in Section 3.2. In Section 3.3, we discuss how to compute the forces acting on the sand particles and how the grid-based particle sand simulation framework in Zhu and Bridson [26] can be transferred to a particle framework. Sections 3.4 and 3.5 explain how to combine the porous flow with the sand materials. We will describe the adaptive sampling algorithm in Section 3.6. Finally, Section 3.7 is the summary of our algorithm.

3.1 Smoothed Particle Hydrodynamics (SPH) Model

In order to simulate fluids, we represent fluids as a set of particles. Smoothed particle hydrodynamics (SPH) is a method to compute approximate numerical solution of the fluid dynamics equations.

The classical SPH equation from [21], [22] is:

$$A(\boldsymbol{x}) = \sum_j m_j \frac{A_j}{\rho_j} W\left(x - x_j, h\right), \tag{1}$$

where ρ_j is the density of particle i and $W\left(x - x_j, h\right)$ is a smoothing kernel function, with support radius h. A particle is at position $\boldsymbol{x}_i$, with mass m_i and additional attributes A_i.

3.2 Fluid Simulation

We use Smoothed Particle Hydrodynamics (SPH) based on previous works of Müller et al. [19] and Müller et al. [20] to simulation fluids. Since we use adaptive sampling to simulate particles, the definitions of the force we use are different.

To simulate fluids, we need to use particles to solve the Navier-Stokes equations. The Navier-Stokes momentum equation is:

$$\rho \frac{Dv}{Dt} = -\nabla P + \rho \boldsymbol{g} + \mu \nabla^2 \tag{2}$$

where $\boldsymbol{v}$ is velocity, P is pressure μ is viscosity and $\boldsymbol{g}$ is the gravity. Then, particle forces can be obtained by using the work as Müller et al. [19]. Equation (2) modeling three forces: pressure force $(-\nabla p)$, external forces $(\rho \boldsymbol{g})$ and viscosity force $(\mu \nabla^2 \boldsymbol{v})$. Thus, equation (2) can be rewritten as:

$$\rho \boldsymbol{a} = \boldsymbol{f}^{pressure} + \boldsymbol{f}^{external} + \boldsymbol{f}^{viscosity}. \tag{3}$$

We define that the $\boldsymbol{f}^{external}$ is external body forces as gravity force or surface tension force.

Since our system is based on adaptive sampling algorithm [1]. By the definition of [1], a particle p_i has a neighbor p_j if $\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| \leq max\,(r_i, r_j)$. In Figure 3.1, particle p_i and p_j are neighbors of each other. We use k-d tree to compute particles' neighborhood.

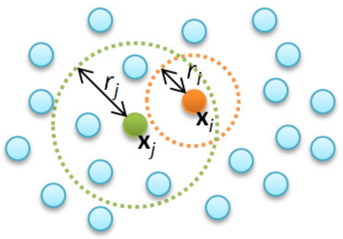


Figure 3.1: Particle p_i and particle p_j are neighbors of each other

In our system, particles may have difference radius. So we use the shooting-gathering approach from the work of Desbrun and Cani [10] and Adams et al. [1]. Combing the force defined in [19], we obtain the forces:

$$f_{ij}^{pressure} = -V_i V_j (P_i + P_j)(\nabla W(\boldsymbol{x}_{ij}, r_i) + \nabla W(\boldsymbol{x}_{ij}, r_j))/2, \tag{4}$$

$$f_{ij}^{viscosity} = \mu V_i V_j (\boldsymbol{v}_j - \boldsymbol{v}_i)(\nabla^2 W(\boldsymbol{x}_{ij}, r_i) + \nabla^2 W(\boldsymbol{x}_{ij}, r_j))/2, \tag{5}$$

where $V_i = {}^{m_i}/_{\rho_i}$ is the particle volume, ρ_i is the particle density, $\boldsymbol{x}_{ij} = \boldsymbol{x}_j - \boldsymbol{x}_i$, P_i is the pressure, μ is the viscosity, and $\boldsymbol{v}_i$ is the particle velocity. The ρ_i can be computed:

$$\rho_i = \rho(\boldsymbol{x}_i) = \sum_j m_j W(\boldsymbol{x}_{ij}, r_i) \tag{6}$$

The pressures $P_i = k({}^{\rho_i}/_{\rho} - 1)$, ρ is the physical fluid density and k is a user defined constant. The kernel functions of $W(\boldsymbol{x}, r)$ we use is as defined in [19]. Since the kernel function is radially symmetric, our system obeys Newton's Third Law.

The surface tension model we use is similar to the work of Becker and Teschner [3].

3.3 Sand Simulation

To simulate sand materials, we use the method based on Zhu and Bridson [26]. Zhu and Bridson [26] provided a physically-based simulation method to animate sand as fluids. At first, we use the SPH model to solve the pressure gradients, and then we use Tait's pressure equation in the work of Becker and Teschner [3] to make the intermediate velocity field nearly incompressible. Tait's equation is

$$P = B((\frac{\rho}{\rho_0})^{\gamma} - 1) \tag{7}$$

with the pressure constant B and a user define constant γ.

In Zhu and Bridson [26], they decompose the sand materials into two parts: one is the region moving rigidly and the other is the region of flow. When the particle of sand is flowing, the frictional stress is given as:

$$\sigma_f = -\mu_f \frac{D}{\sqrt{1/3|D|_F}} \tag{8}$$

where μ_f is the friction coefficient. The strain rate $D = (\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}})/2$ can be evaluated by using the SPH method. In the Solenthaler et al. [25], $\nabla\boldsymbol{u}$ is defined as the displacement gradient:

$$\nabla\boldsymbol{u}_i = \sum_j V_j \nabla W(x_j - x_i, h_j)(\boldsymbol{v}_j \Delta t)^T \tag{9}$$

Note that the result of $\nabla\boldsymbol{u}$ is a 3×3 matrix, since $\nabla W(x_j - x_i, h_j)$ is a 3×1 matrix and $(\boldsymbol{v}_j \Delta t)^T$ is a 1×3 matrix. To find out which particle is moving rigidly, we need to use the Mohr-Coulomb condition as the work in Zhu and Bridson [26]. The Mohr-Coulomb condition determines the martial will not yield as long as:

$$\sqrt{3}\bar{\sigma} < \mu_f \sigma_m + c \tag{10}$$

where $\bar{\sigma}$ is the shear stress, $\sigma_m = tr(\sigma)/3$ is the mean stress and $c > 0$ is the cohesion coefficient. $\bar{\sigma}$ can be compute as:

$$\bar{\sigma} = |\sigma - \sigma_m \delta|_F / \sqrt{2} \tag{11}$$

where $|\cdot|_F$ is the Frobenus norm. The definition of Frobenus norm is

$|A|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|a_{ij}\right|^2}$, A is an $\mathrm{m} \times \mathrm{n}$ matrix. We use the rigid stress to check if the particles satisfy the Mohr-Coulomb condition. The rigid stress is:

$$\sigma_{rigid} = -\frac{\rho D h^2}{\Delta t} \tag{12}$$

Equation (10) can help us to recognize which particle is rigid or not. Then, we search for clusters of particles which are moving rigidly. If two particles are neighboring particles and within a support range, we mark these particles as in the same clusters. Using the breadth-first search, we can easily find the clusters of particles.

Figure 3.2: Sand particles interactive with water particles.

In Figure 3.2, there are two kinds of particles. The blue particles are water particles, and the brown and orange particles are sand particles. The orange particles are moving rigidly. h is the support range of a particle and h' is the distance between the neighboring particles. If $h' \leq h$, the neighboring particles are in the same cluster. After we define the clusters, we need to compute the force acting on the clusters. Since the particle clusters are moving rigidly, we consider the rigid body motion. The basic rigid body simulation described in Baraff [2]. According to Solenthaler et al. [25] and Baraff [2], the torque vector τ can be computed as:

$$\tau_i = (\boldsymbol{r}_i - \boldsymbol{r}^{cm}) \times \boldsymbol{F}_i \tag{12}$$

where $\boldsymbol{r}^{cm}$ is the center of mass of a cluster and $\boldsymbol{F}_i$ is the total force on the *i*th particle. The total force acting on a cluster is the sum of the $\boldsymbol{F}_i$:

$$\boldsymbol{F}_{cluster} = \sum\nolimits_{i \in cluster} \boldsymbol{F}_i \tag{13}$$

and the total torque can be computed similar to (13):

$$\tau_{cluster} = \sum\nolimits_{i \in cluster} \tau_i \tag{14}$$

After the total force and torque of a cluster are computed, we can use this information to compute the angular accelerations and then update the angular velocity and position of particles. By the definition of the Newton's second law, the relation between torque and angular acceleration is:

$$\tau = \boldsymbol{I}\alpha \tag{15}$$

where τ is the total torque acting on a cluster (a rigid body) and $\boldsymbol{I}$ is the inertia tensor. To compute the angular acceleration action on the *i*th cluster, equation (15) can be rewritten as:

$$\alpha_i = \boldsymbol{I}_i^{-1}\tau_i \tag{16}$$

In Baraff [2], the inertia tensor $\boldsymbol{I}_i$ in the *i*th cluster:

$$\boldsymbol{I}_i = \sum \begin{pmatrix} m_j(r_{jy}'^2 + r_{jz}'^2) & -m_j r_{jx}' r_{jy}' & -m_j r_{jx}' r_{jz}' \\ -m_j r_{jy}' r_{jx}' & m_j(r_{jy}'^2 + r_{jy}'^2) & -m_j r_{jy}' r_{jz}' \\ -m_j r_{jz}' r_{jx}' & -m_j r_{jz}' r_{jy}' & m_j(r_{jx}'^2 + r_{jy}'^2) \end{pmatrix} \tag{17}$$

with $r_j' = \boldsymbol{r}_j - \boldsymbol{r}^{cm}$, for all the particles $j \in i$th cluster. Then the velocity of the particles can be updated:

$$v_i(t) = v_i(t-1) + \left(\frac{\boldsymbol{F}^{total}}{m_i}\right)\Delta t + (\alpha_i \Delta t) \times r_i' \tag{18}$$

The brown particles in Figure 3.2 are in a state of shearing flow. Since in equation (8) we have compute the frictional stress, the force acting on these particles can be computed by using the work in the Solenthaler et al. [25]. The force is similar to the elastic force:

$$\boldsymbol{F}_{ji}^{elastic} = -\nabla \boldsymbol{u}_j U_i = -2V_i(I + \nabla \boldsymbol{u}_i^T)\sigma_f \boldsymbol{d}_{ij} \tag{19}$$

where *I* is the identity matrix, V_i is the body volume of particle i, and $\boldsymbol{d}_{ij}$ is defined as:

$$\boldsymbol{d}_{ij} = V_j \nabla W(x_j - x_i, h_j) \tag{20}$$

Finally, Figure 3.3 summarizes the algorithm for simulating the sand material.

```
Algorithm: Step of SandSimulator.
1.  foreach particle i
2.      apply gravity force and pressure force
3.      compute $\nabla u_i$ and strain rate D
4.      compute rigid stress $\sigma_r$
5.      get mean stress from rigid stress: $\sigma_m = tr(\sigma_r)/3$
6.      get shear stress $\bar{\sigma}$
7.      use Mohr-Coulomb condition in equation (10) to test the particle i is rigid
        or not.
8.      If a particle i is moving rigidly, then search for clusters of
        particles marked as rigid.
9.          // compute the force and torque acting on the clusters of particles
10.     If a particle i is in the region of shearing flow:
11.         // compute the force acting on this particle
```

Figure 3.3: Algorithm for a step in SandSimulator

3.4 Porous Flow Simulation

To simulate the water absorption of sand, we use porous flow simulation as the work of Lenaerts et al. [16]. A porous particle is defined by its porosity and permeability. We can think that a porous particle has a lot of hole inside its body. Thus, a porous particle is capable of holding a amount of water. Figure 3.4 shows the microscopic view and macroscopic view of the porous material model.

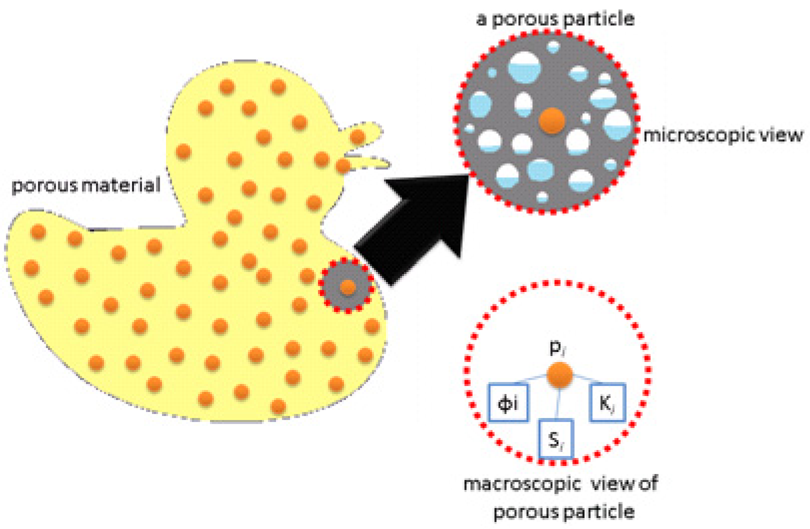


Figure 3.4 Porous materials. Top right region is microscopic view of a porous particle. A porous particle can hold water inside; the water inside a porous particle is in blue color. Bottom right region is macroscopic view of a porous particle. A porous particle p_i with porosity ϕ_i, saturation S_i and permeability K_i.

By the definition of porous particles in Lenaerts et al. [16], the porosity ϕ_i represents the unit volume of void space among particles. For example, $\phi_i V_i$ is the void volume in a particle p_i. Therefore, a particle p_i with porosity ϕ_i can hold the absorbed masses $m_{pi} \leq \rho^{fluid} \phi_i V_i$. The particle's saturation S_i can be computed as:

$$S_i = \frac{m_{pi}}{\rho^{fluid} \phi_i V_i}, 0 \leq S_i \leq 1 \tag{21}$$

The permeability K_i is a constant. For different materials, the K_i is different.

In microscopic view, the capillary pressure is the reason why the water can enter into a porous particle and diffuse to neighboring porous particles. The capillary pressure force also keeps the water inside the porous materials. Since our model does not have real pores, we use the method of Lenaerts et al. [16] to model the capillary forces.

$$\nabla P_i^c = \sum_j V_j P_j^c \nabla W(\boldsymbol{x}_j - \boldsymbol{x}_i, h_j), \tag{22}$$

where P^c is the capillary potential, it can be defined as:

$$P_i^c = k^c (1 - S_i)^{\alpha}, \tag{23}$$

where k^c is a constant and $0 < \alpha < 1$. When a porous particle's saturation increases, the capillary force decreases.

During the simulation, particles in our system will split and merge due to adaptive sampling algorithm. The volume of the pore space will be different. The porosity ϕ_i is depending on the local density of the material:

$$\phi_i = \phi_0 \frac{\rho_0^s}{\rho_i^s} \tag{24}$$

ϕ_0 is the porosity in the beginning. To compute the pore pressure P_i^p, we use the equation similar to the Tait's equation [3]:

$$P_i^p = k^p S_i \left(\left(\frac{\rho_0^s}{\rho_i^s} \right)^{\gamma} - 1 \right) \tag{25}$$

Darcy's law [9] is an equation to describe incompressible fluid flow inside the porous materials. The pore velocity is $\boldsymbol{v_{pi}} = {}^{q}/_{\phi}$, where q is the Darcy flux. In three dimensions, we need to consider the gravity, so the Darcy's law can be:

$$q = -\frac{K_i}{\mu} (\nabla P - \rho \boldsymbol{g}) \tag{26}$$

By using equation (26), the pore velocity $\boldsymbol{v_{pi}}$ can be computed as:

$$\boldsymbol{v_{pi}} = -\frac{K_i}{\phi_i \mu} (\nabla P_i^p - \nabla P_i^c - \rho \boldsymbol{g}) \tag{27}$$

The pore velocity inside the materials can't be easily predicted because it's an anisotropic diffusion. Figure 3.5 shows the pore velocity of water through the sand materials in macroscopic view.

Figure 3.5: Actual fluid velocity through the pore space of sand materials.

In our thesis, we just consider the fluid flow inside the materials, like the diffusion process in Müller et al. [20]. During the simulation, fluid mass is diffusing from one porous particle to its neighbors. This way is easy than tracking the fluid particle inside the materials. The diffusion equation for compute absorbed fluid mass m_p is:

$$\frac{\partial m_{pi}}{\partial t} = \sum_j d_{ij} V_j m_{pj} \nabla^2 W(\boldsymbol{x}_j - \boldsymbol{x}_i, h_j) \tag{28}$$

with the diffusion coefficients d_{ij}:

$$d_{ij} = \boldsymbol{v}_{\boldsymbol{pi}} \cdot \frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|} S_j^{\beta}, \ \ \beta > 0 \tag{29}$$

Using an explicit Euler integration step, the fluid mass can be updated:

$$m_{pi} \leftarrow m_{pi} + \Delta t \frac{\partial m_{pi}}{\partial t} \tag{30}$$

Before updating the fluid mass, we need to check the free and occupied volumes of the porous particles. The free volume of a porous particle is $(1 - S_i)\phi_i V_i$, and occupied volumes of a porous particle are $S_i \phi_i V_i$.

When porous particles absorb fluid, the absorbed fluid mass m_{pi} should be taken into account to all computations. The density of a porous particle will be changed if the absorbed fluid mass m_{pi} increases. Thus, the density in rest becomes:

$$\rho_{0i} = \rho_0 + S_i \phi_i \rho_0^{fluid} \tag{31}$$

Considering the simulation of elastic bodies, the fluid pressure also influences the stress σ_i in the elastic bodies:

$$\sigma_i^{eff} = \sigma_i - \eta P_i^{p*} \boldsymbol{I} = \sigma_i - \eta (k^p S_i) \boldsymbol{I} \tag{32}$$

The $P_i^{p*} = k^p S_i$ is the pore pressure of the present fluid. η is a scaling factor.

3.5 Combing Different Particles

This section presents how we combine different kinds of particles together. During the simulation, if there is no collision or any intersection between the sand particles and water particles, we just simulate them separately. But if the water particles are too close to the sand particles, or there are collisions occur, the interaction between difference kinds of particles must take into account. The collision handling we use in our system is based on the approach of the Molecular Dynamics in Bell et al. [4] and Clavet et al. [7]. Figure 3.6 is the algorithm of the mix simulation step.

```
Algorithm: MixSimulator step.
1.  foreach particle i
2.      if particle i is a water particle
3.          // do water simulation
4.      if particle i is a sand particle, and water is interactive with sand
5.          // do sand simulation
6.  move particle
7.  compute the Absorption and Emission
8.  solve collisions
```

Figure 3.6 Algorithm for a step in MixSimulator

In the beginning, we do not need to consider the sand particle until the water particles are too close to the sand particles. Line 1 to line 5 are the step of computing the total forces on the sand and water particles. Line 6 is to update the sand and water particles position and velocity. Line 7 is to compute the absorption and emission of the sand particles. Since we simulate the sand particles as porous materials, there are three different cases to describe the phenomena of absorption and emission. Figure 3.7 can help us easy to know how the porous particles absorb and emit water.

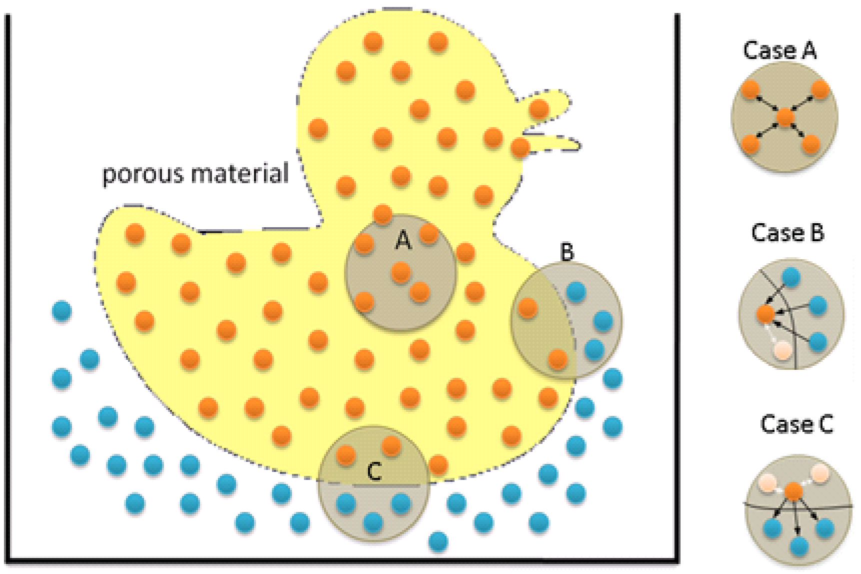


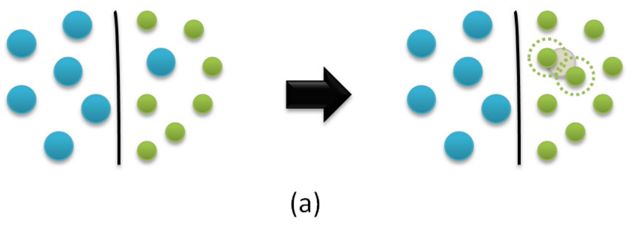
Figure 3.7: Three different cases of how the mass transport from one particle to another.

In Figure 3.7, **case A** is porous particles within the porous material. Fluid mass is diffused among the porous particles at the macroscopic view. **Case B** is the water particles near the porous particles, the water particles may be absorbed due to capillary forces. We can use the same diffusion model as in **case A**. At this time, the water particles at the surface can be treated as saturated porous particles with saturation $S_i = 1$ and porosity $\phi_i = 100\%$. Then we can use equations (22) to (30) to compute the absorption of water. When the mass of water particles is diffused in the absorbing porous particles, these particles will be deleted. **Case C** is porous particles emit water. In the view of microscopic, the absorption always accompany with emission. When the porous particles emit water, mass is transform from porous particles to the neighboring water particles. We can treat the water particles at the interface as unsaturated porous particles with $S_i = 0$. The absorbed fluid mass will diffuse to the water particles by equations (22) to (30) as described in the **case B**.

Finally, in Line 8 of Figure 3.6, we need to consider the collision between sand and water particles. With collision handling, the sand sculpture will collapse more realistic.

3.6 Adaptive Sampling Algorithm

Our system is based on the adaptive sampling algorithm of Adams et al. [1]. They first compute the local feature size $lfs(\boldsymbol{y})$ and extend it. The extended local feature size can be used to decide which particle needs to be split or merge. If $elfs(\boldsymbol{x}_i) < \alpha r_i$, a particles p_i is split. If $elfs(\boldsymbol{x}_i) > \beta r_i$ a particle p_i is merge.



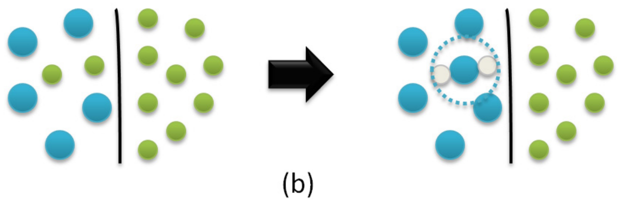


Figure 3.8: Particles split and merge. (a) In the left region, a blue particle is a level *l* particle. In the right region, a blue particle splits into two level *l-1* green particles. (b) Show two level *l-1* green particles merge to a level *l* blue particle.

We simplify the splitting and merging model in Figure 3.8. For more detail about the adaptive sampling algorithm is described in Adams et al. [1].

Since our system is based on the adaptive sampling algorithm, the surface construction is the same as the work in the Adams et al. [1].

3.7 Summary of the Algorithm

Figure 3.9 is the simulation proceeds in our system. First, we mix the sand and water particles together and then compute the particle neighbors. We use a *k-d* tree for the neighborhood queries. Then we consider the porous flow simulation of both sand and water particles. The absorption and emission are describe in Section 3.4 and Section 3.5. Using the SPH, the forces acting on the particles can be computed. The collisions handling is describe in

Section 3.5. After handling collision for sand and water particles, we use the adaptive sampling algorithm for splitting and merging particles. Finally, we update the position of sand and water particles. The time step in our system is a fixed integration, we use 0.001s in all examples.

```
Algorithm: Summary.
1.  mix particles and compute neighbors
2.  compute porous flow of particles
    absorption and emission (section 3.4 and section 3.5)
3.  handle collision
4.  compute density
5.  compute forces for sand and water particles
6.  compute fast marching algorithm for construct medial axis
7.      // particles split and merge (section 3.6)
8.  construct surface and output (section 3.6)
9.  update the position of sand and water particles
```

Figure 3.9: Simulation proceeds in our system.

Chapter 4
Implementation and Results

We implement our system by using C++ framework and OpenMP for parallelized. The simulations are executed on a PC with Intel(R) Core i7 975 3.33GHz CPU and 8GB of memory. The animations are rendered using POV-Ray [23].

Since the scene objects we load are wireframe mesh, we need to transform these wireframe objects into particles. First, we slice a wireframe object to grids as shows in Figure 4.1. Next, we compute the signed distance fields in each grid. In order to compute the signed distance field, we have to sample points on the triangles of the object. After sampling the points, it is easy to compute signed distance fields. For each grid, we search for the nearest point on the triangles of the object. Then we can use this point and it's normal to decide the distance field of the grid.

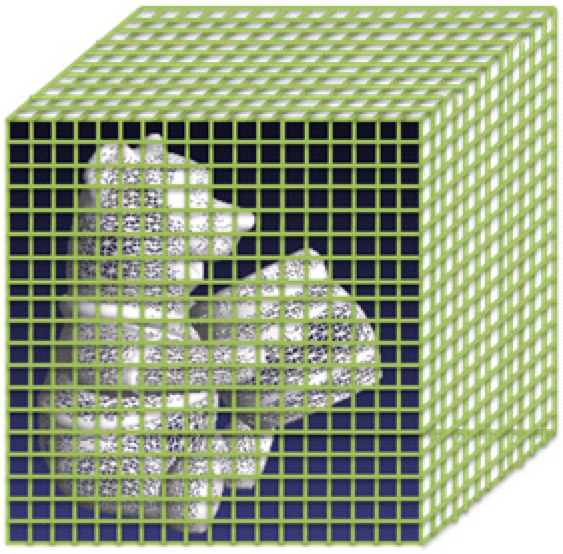


Figure 4.1: Slicing the object to grids.

In the following figures, we show the results of the interaction between sand and water. Figures 4.2(a) to (s) are the same scene with a sand duck and a column of water. In the beginning, the simulation consists of 11,234 sand and 23,050 water particles. The average computation time is 3.8s per step.

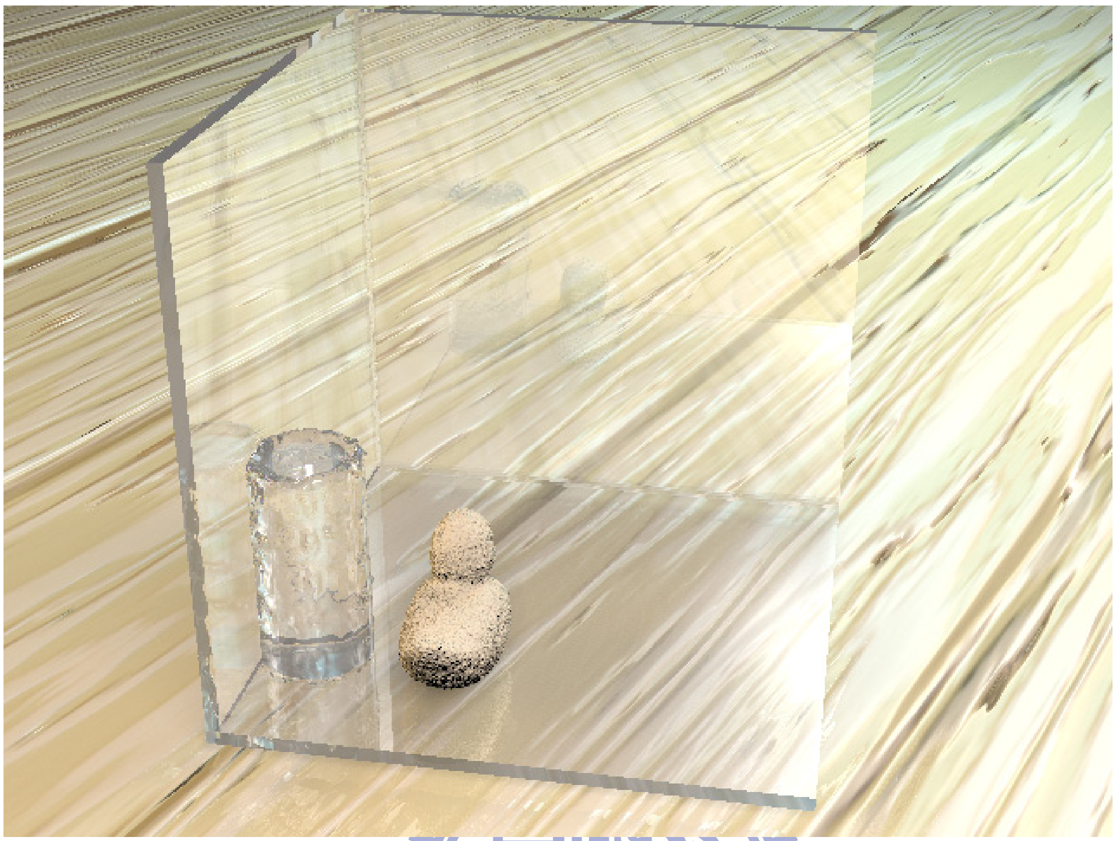


Figure 4.2(a): Step No. 000.

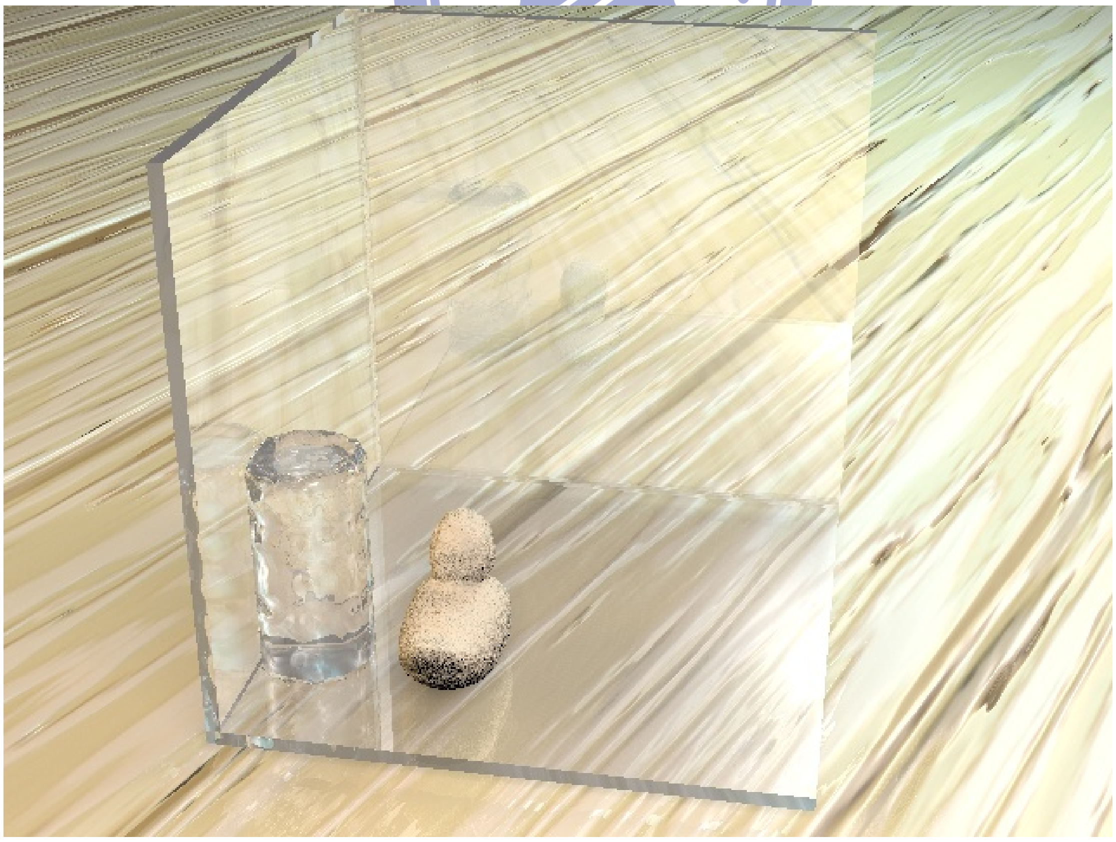


Figure 4.2(b): Step No. 030.

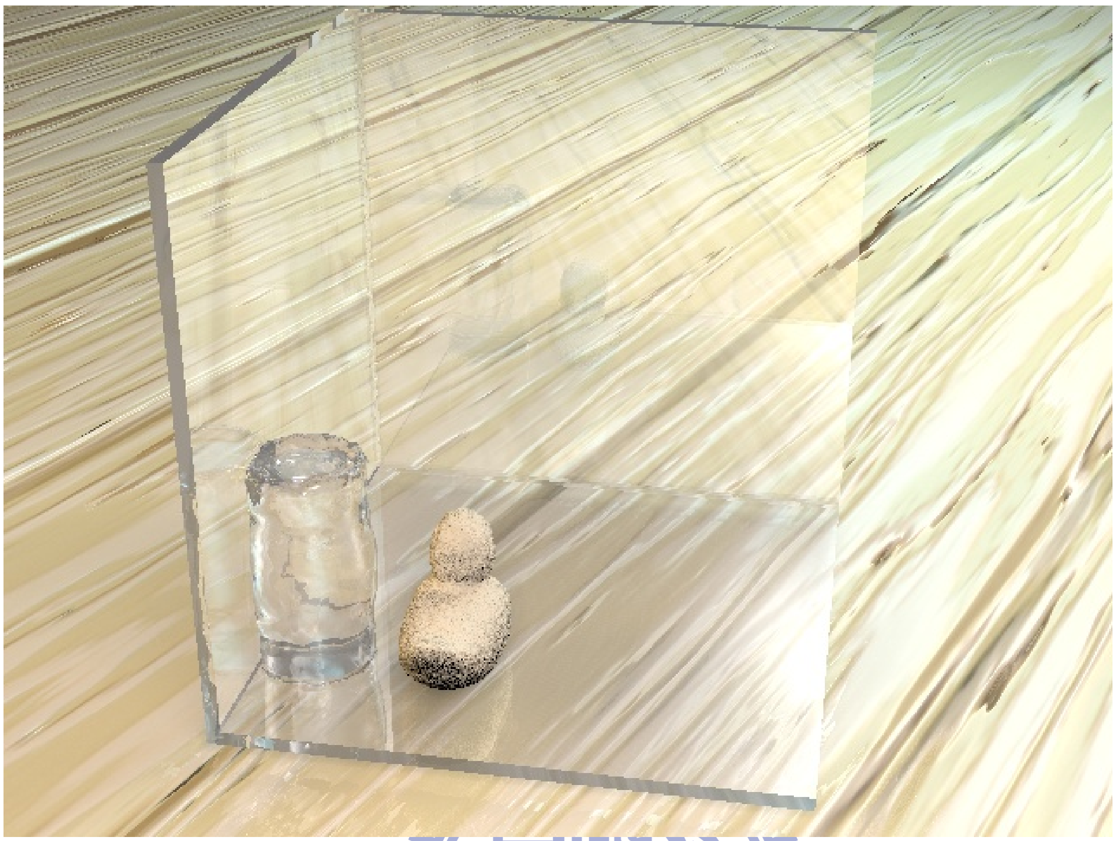


Figure 4.2(c): Step No. 060.

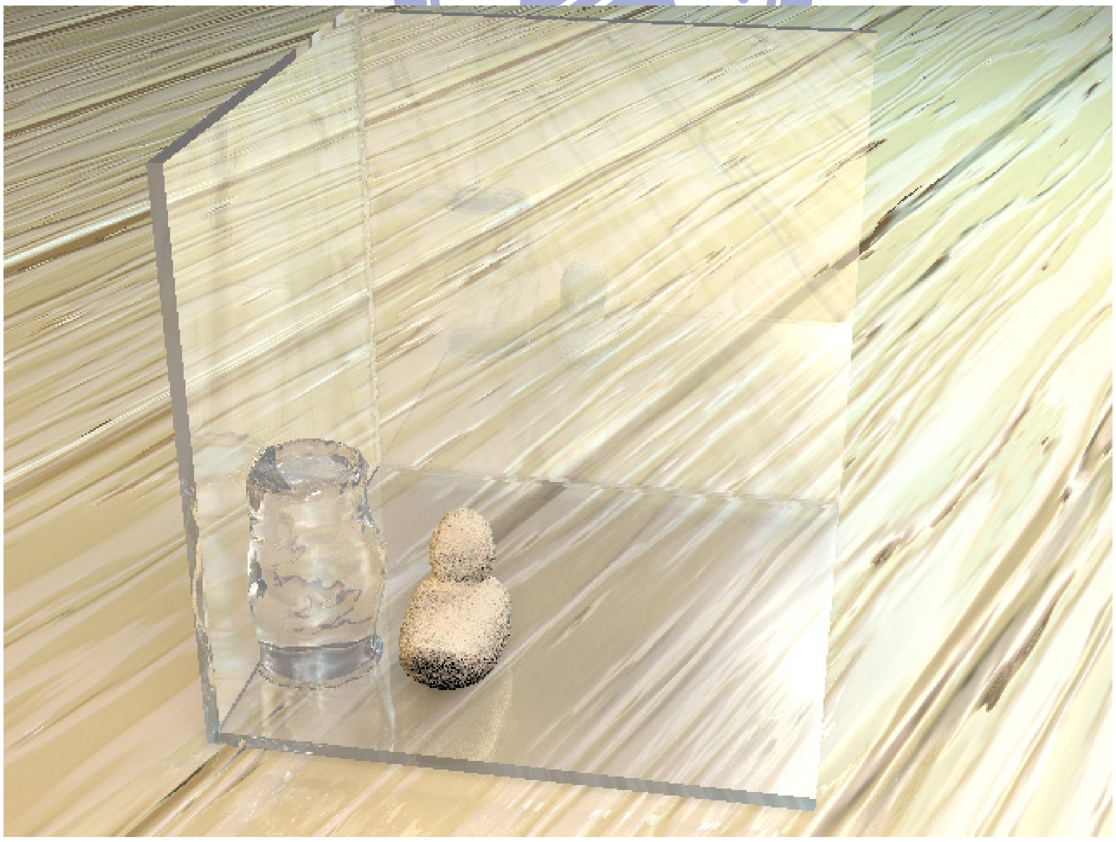


Figure 4.2(d): Step No. 090.

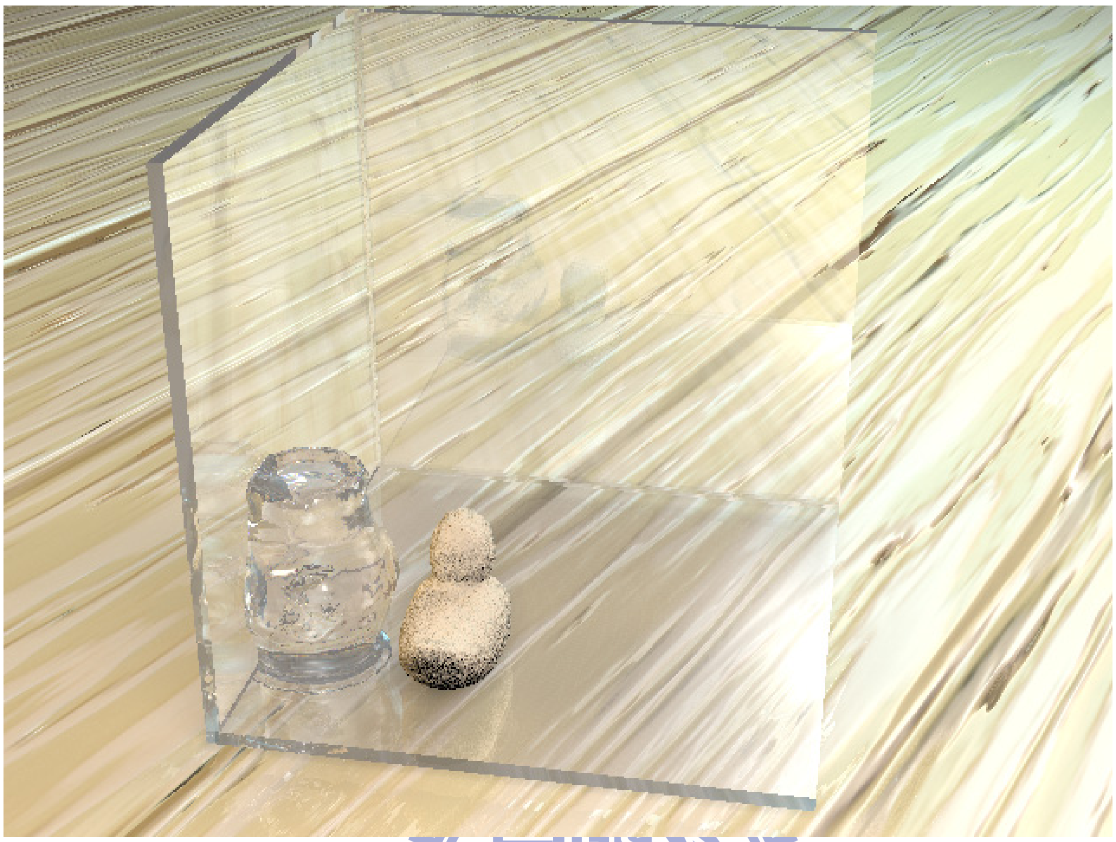


Figure 4.2(e): Step No. 120.

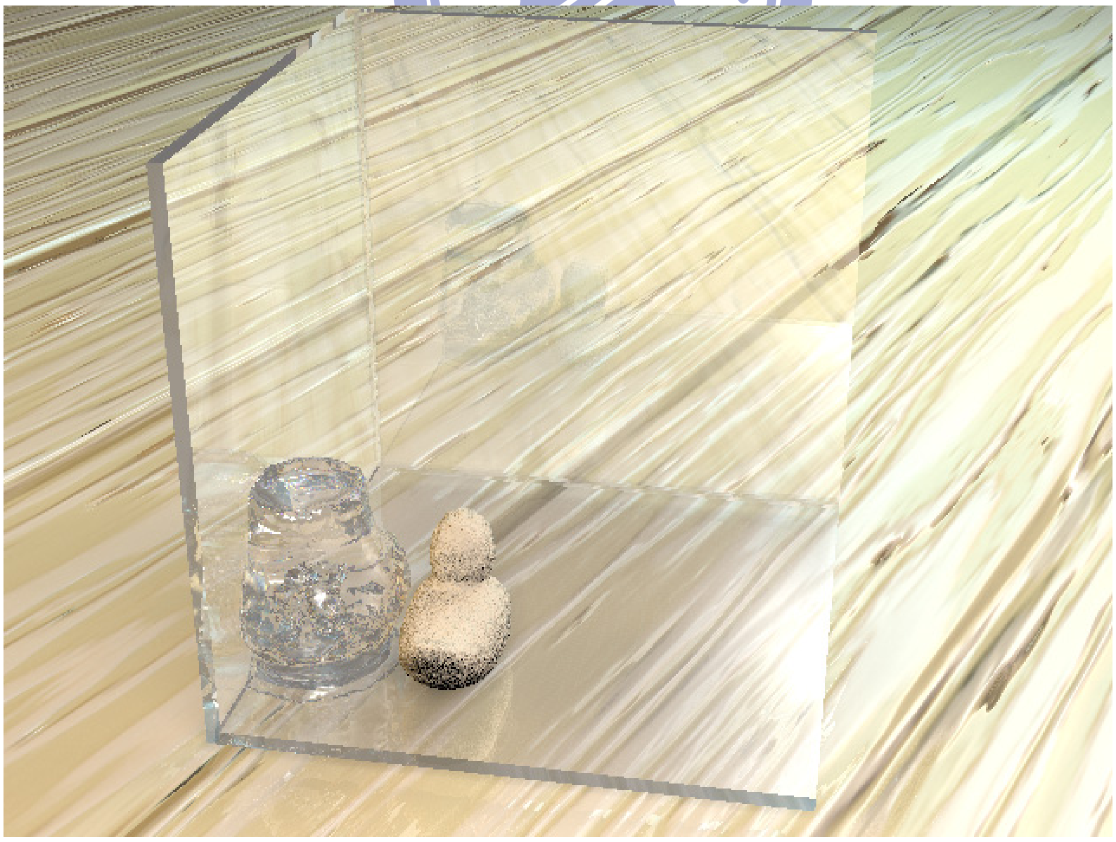


Figure 4.2(f): Step No. 150.

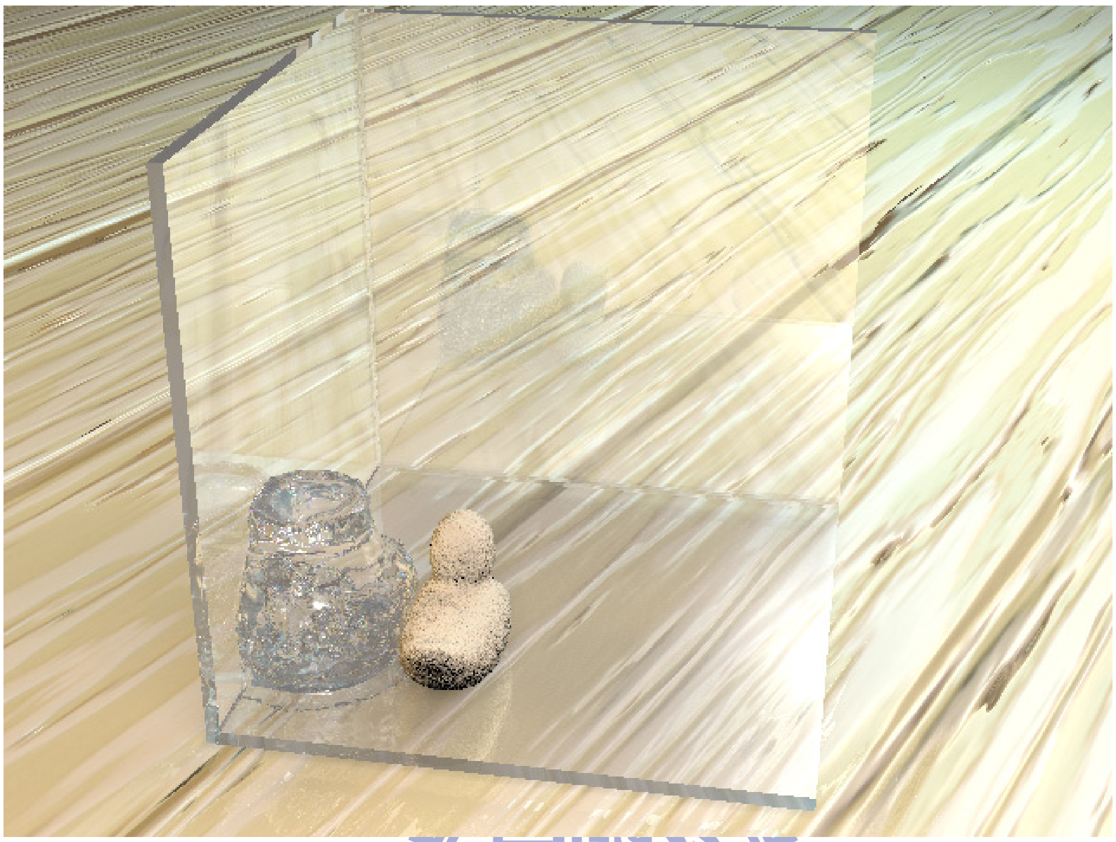


Figure 4.2(g): Step No. 180.

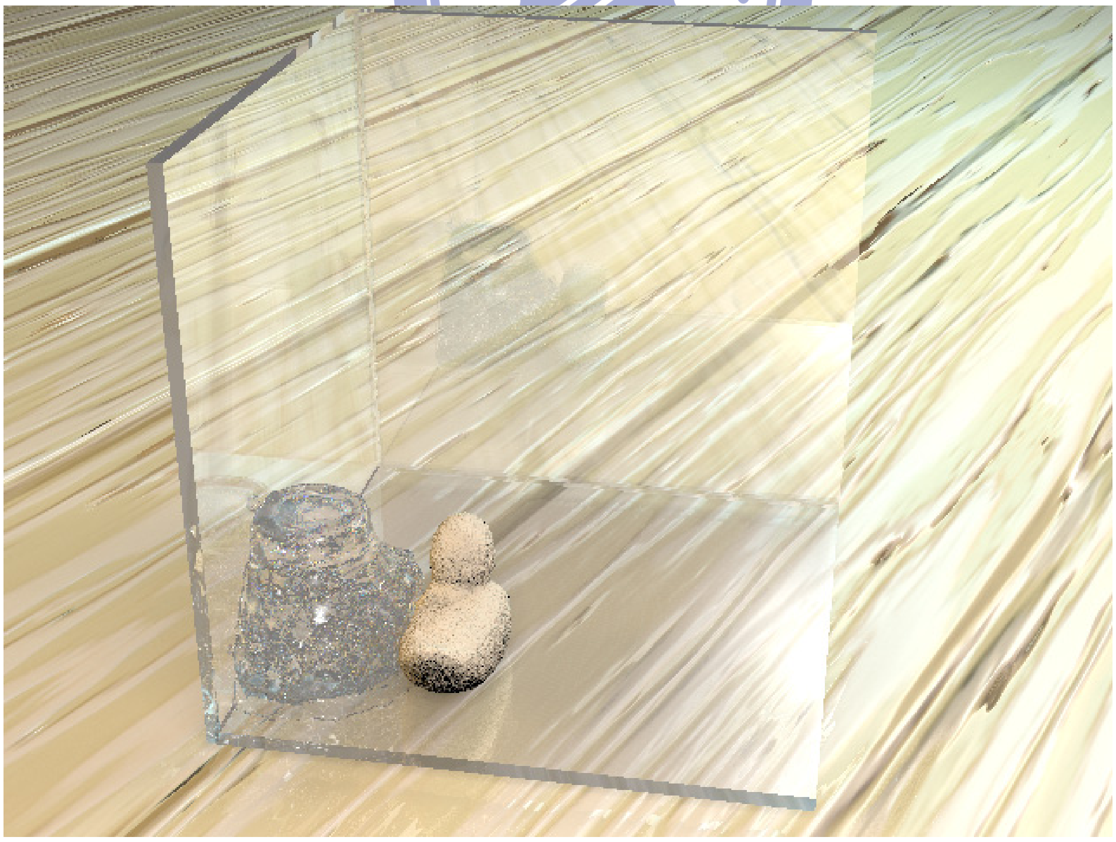


Figure 4.2(h): Step No. 210.

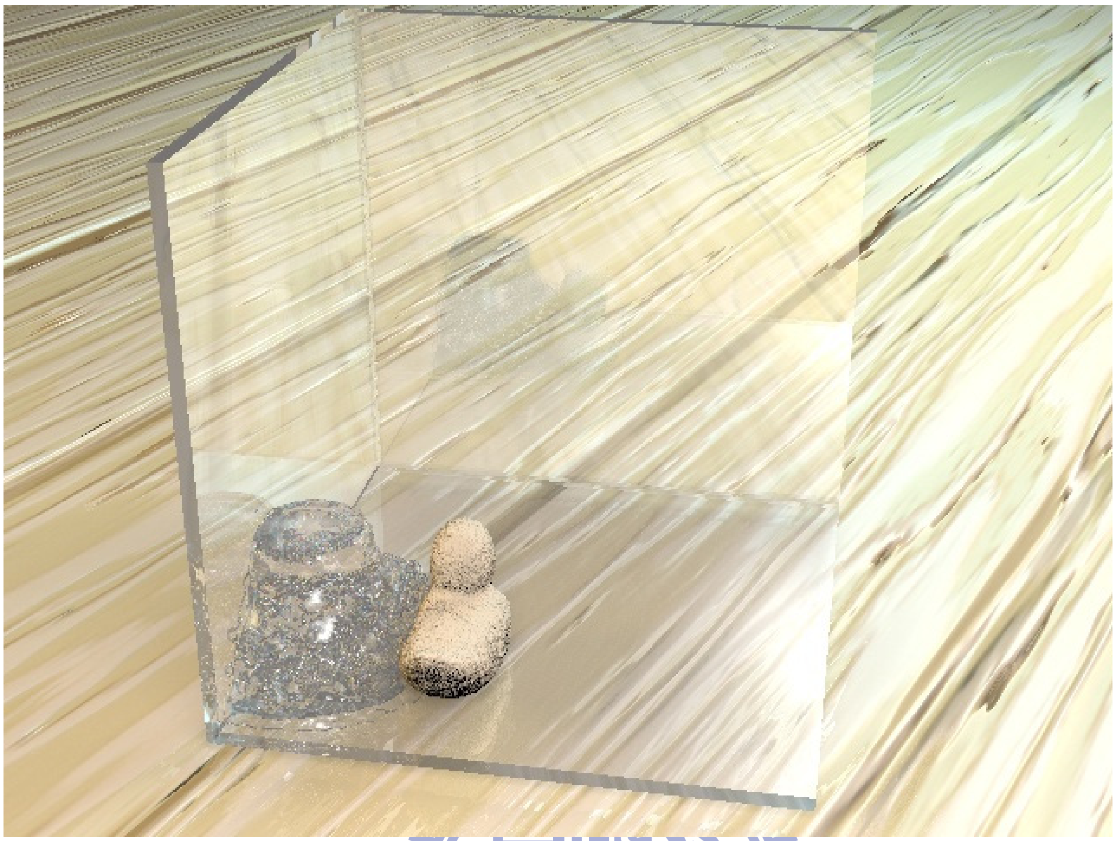


Figure 4.2(i): Step No. 240.

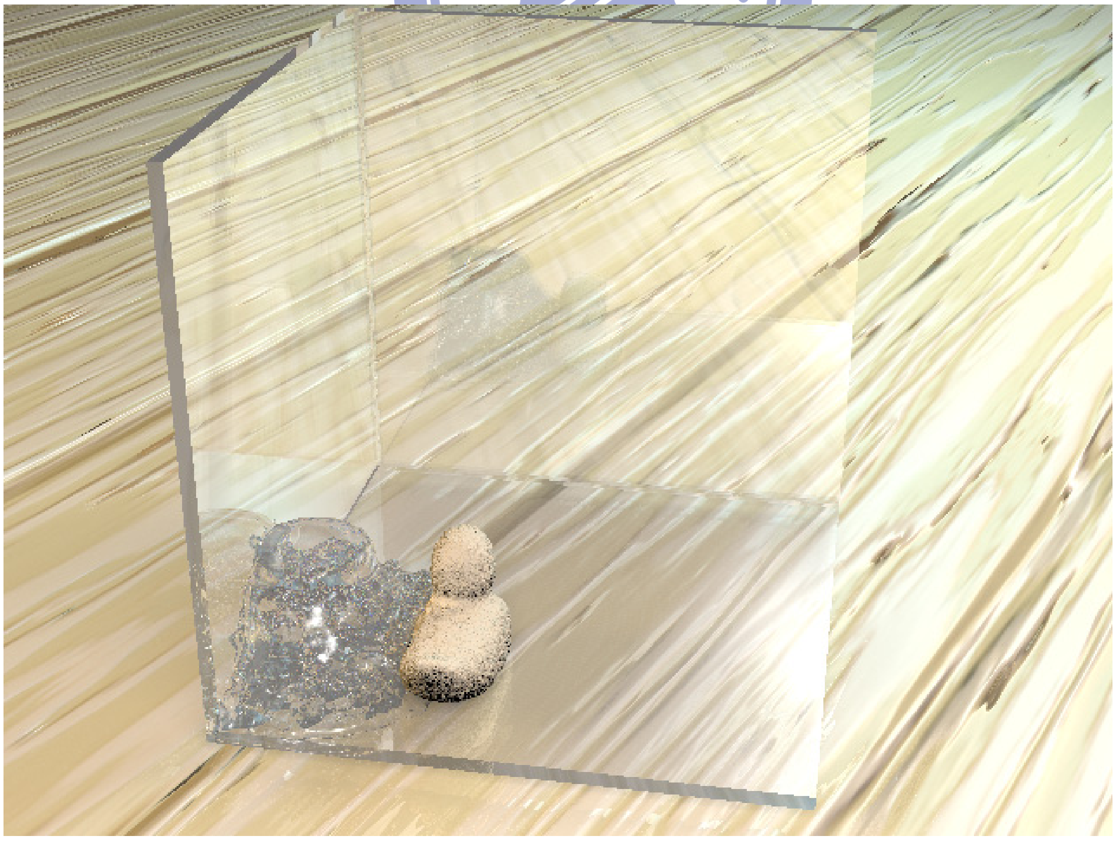


Figure 4.2(j): Step No. 270.

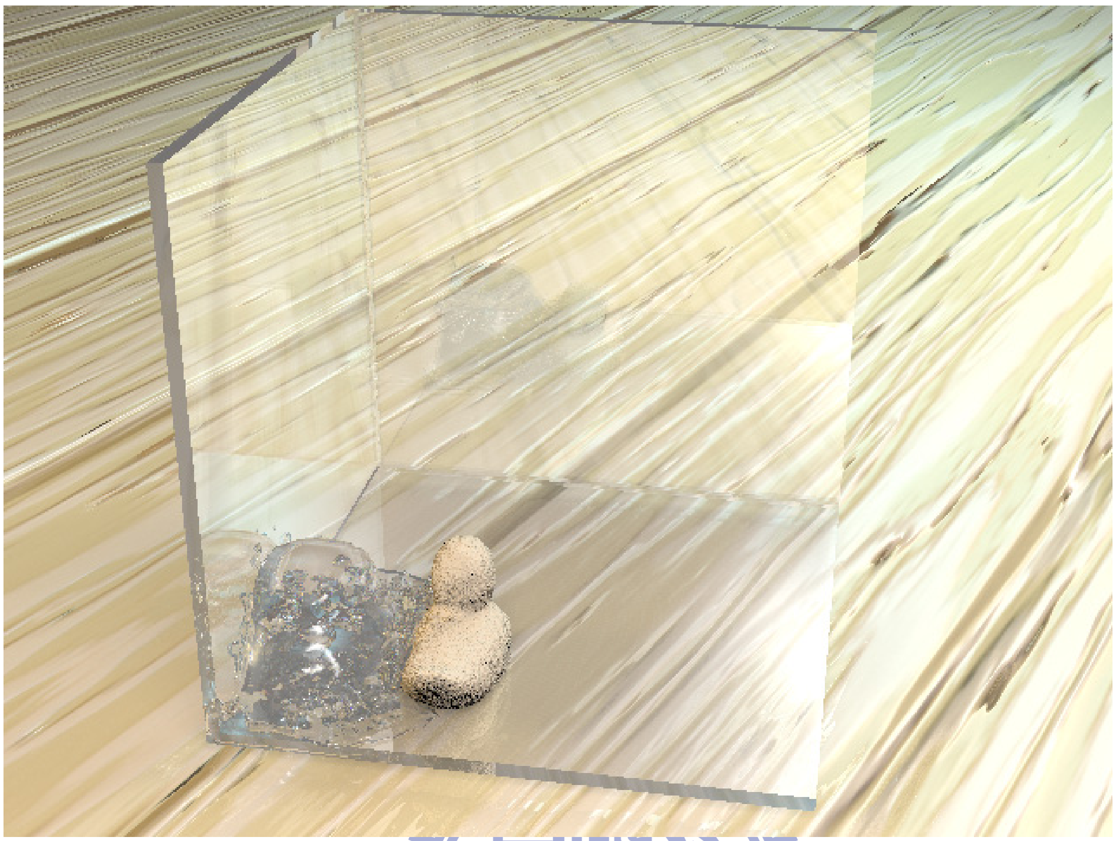


Figure 4.2(k): Step No. 300.

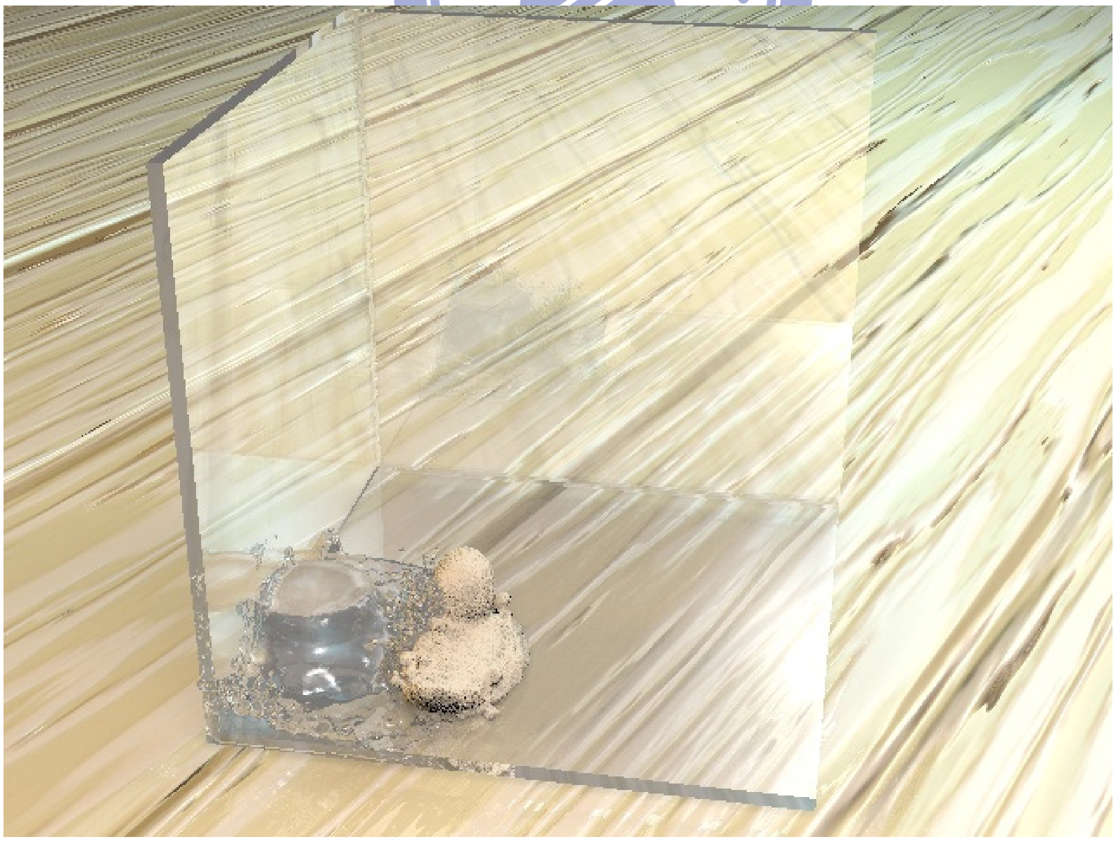


Figure 4.2(l): Step No. 330.

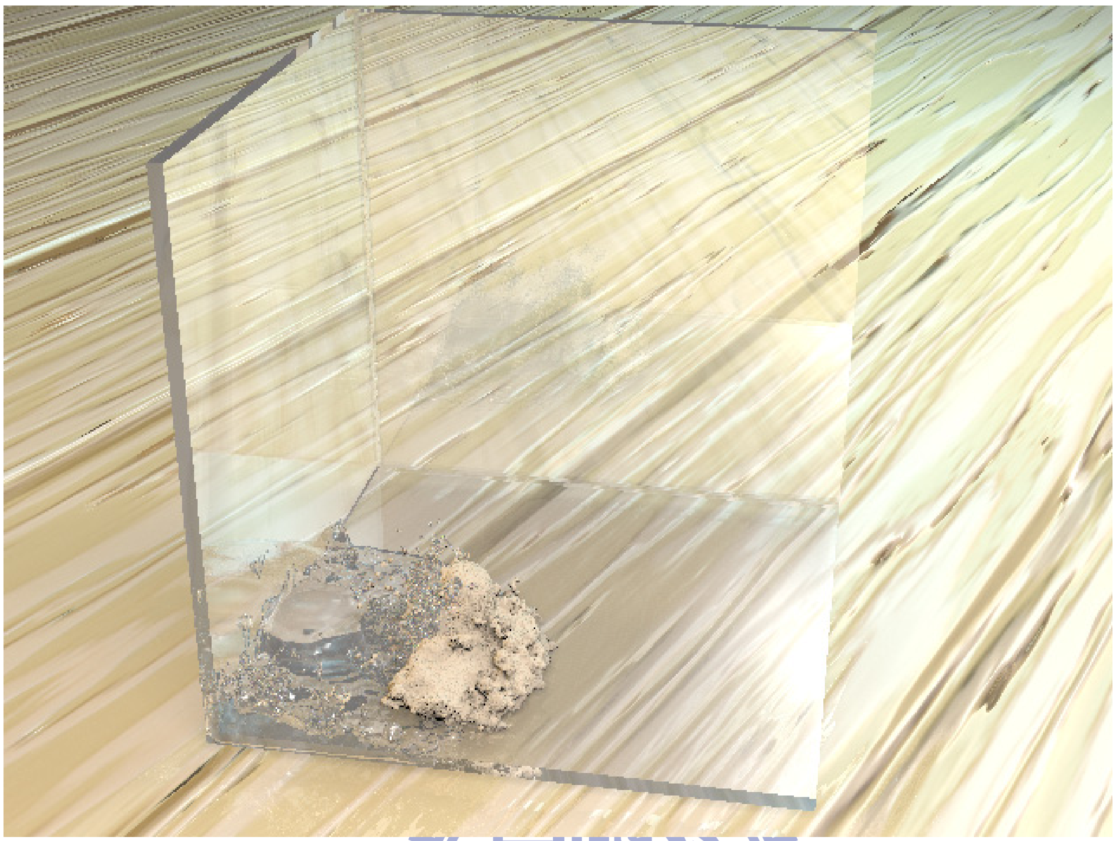


Figure 4.2(m): Step No. 360.

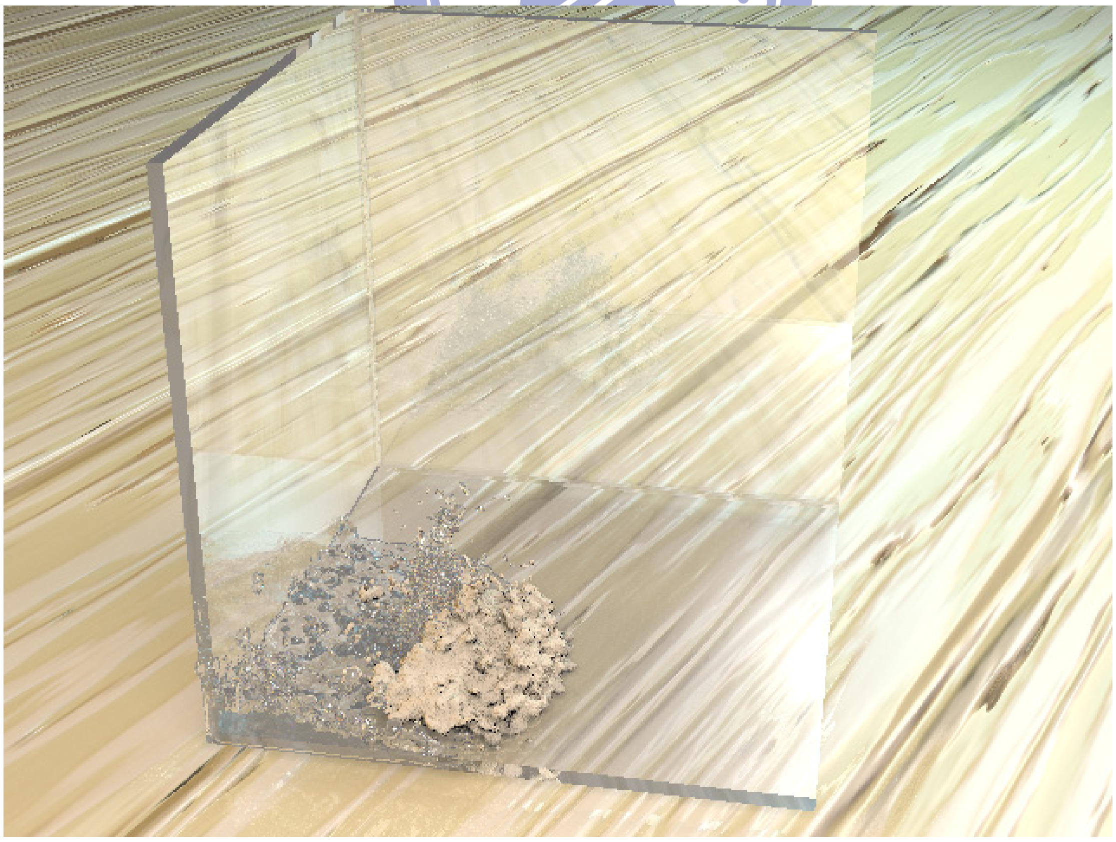


Figure 4.2(n): Step No. 390.

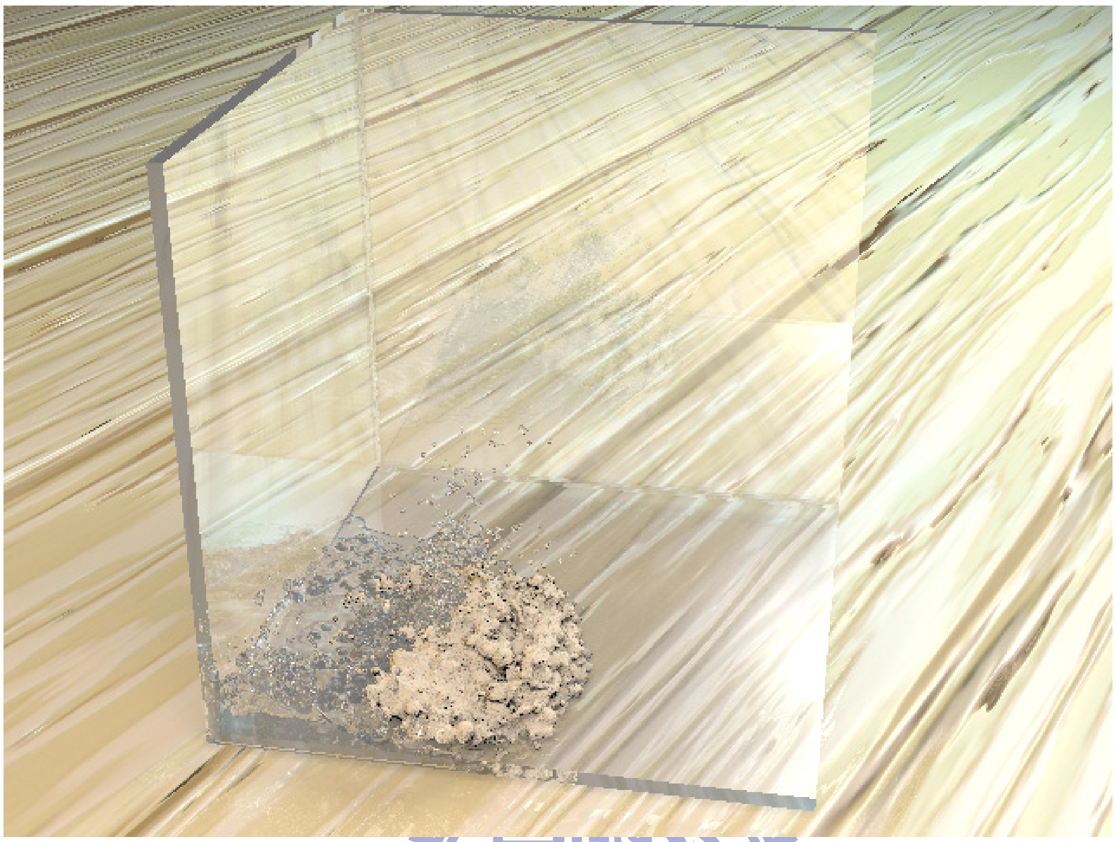


Figure 4.2(o): Step No. 420.

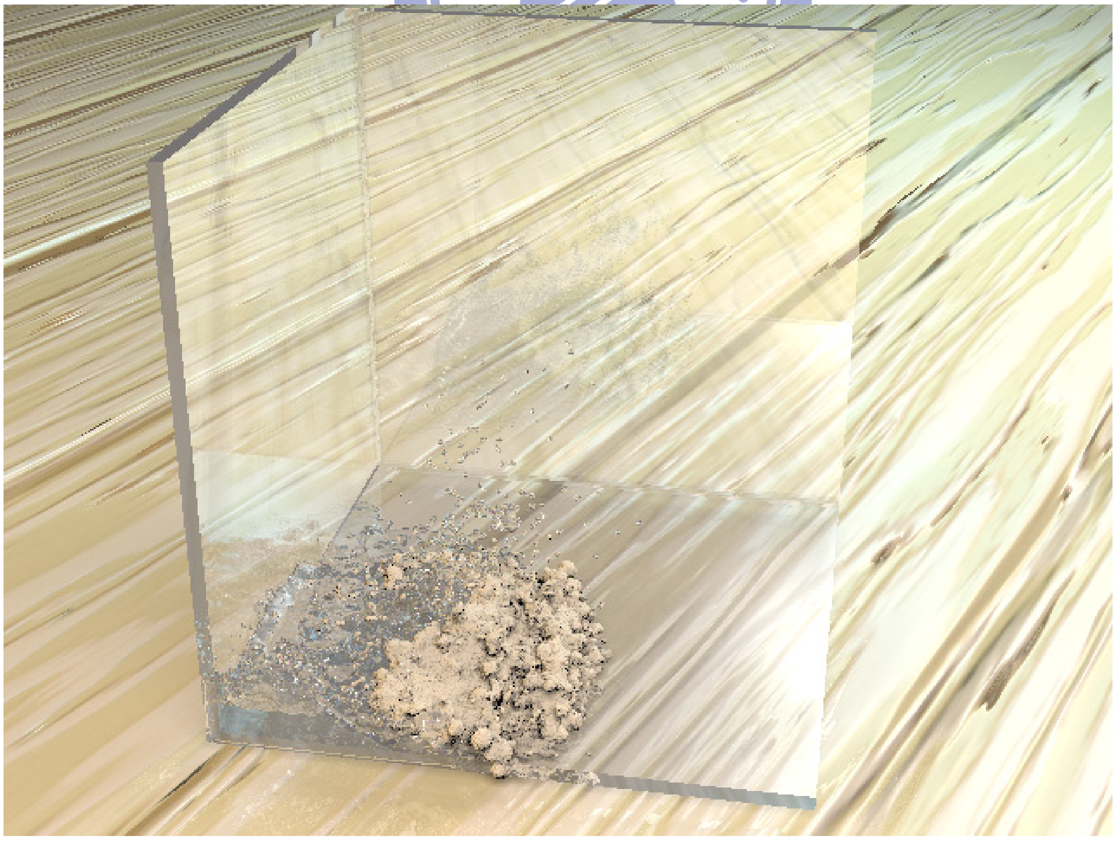


Figure 4.2(p): Step No. 450.

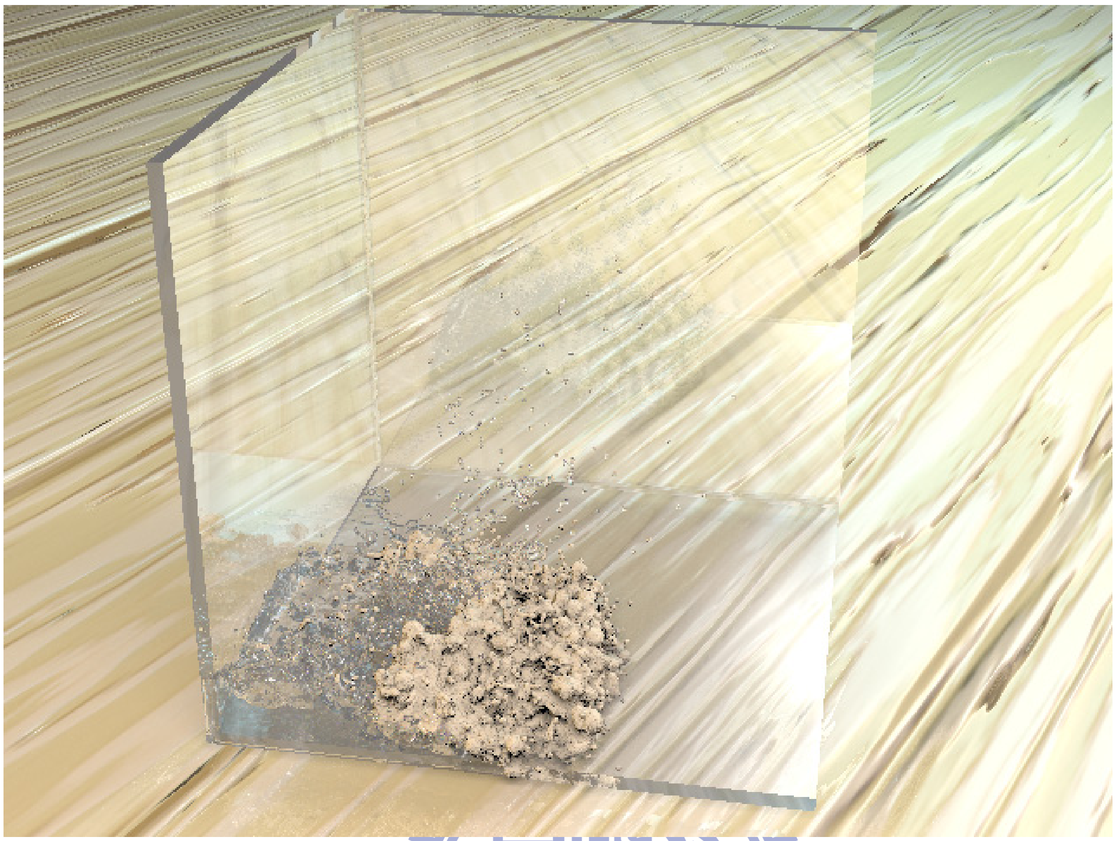


Figure 4.2(q): Step No. 480.

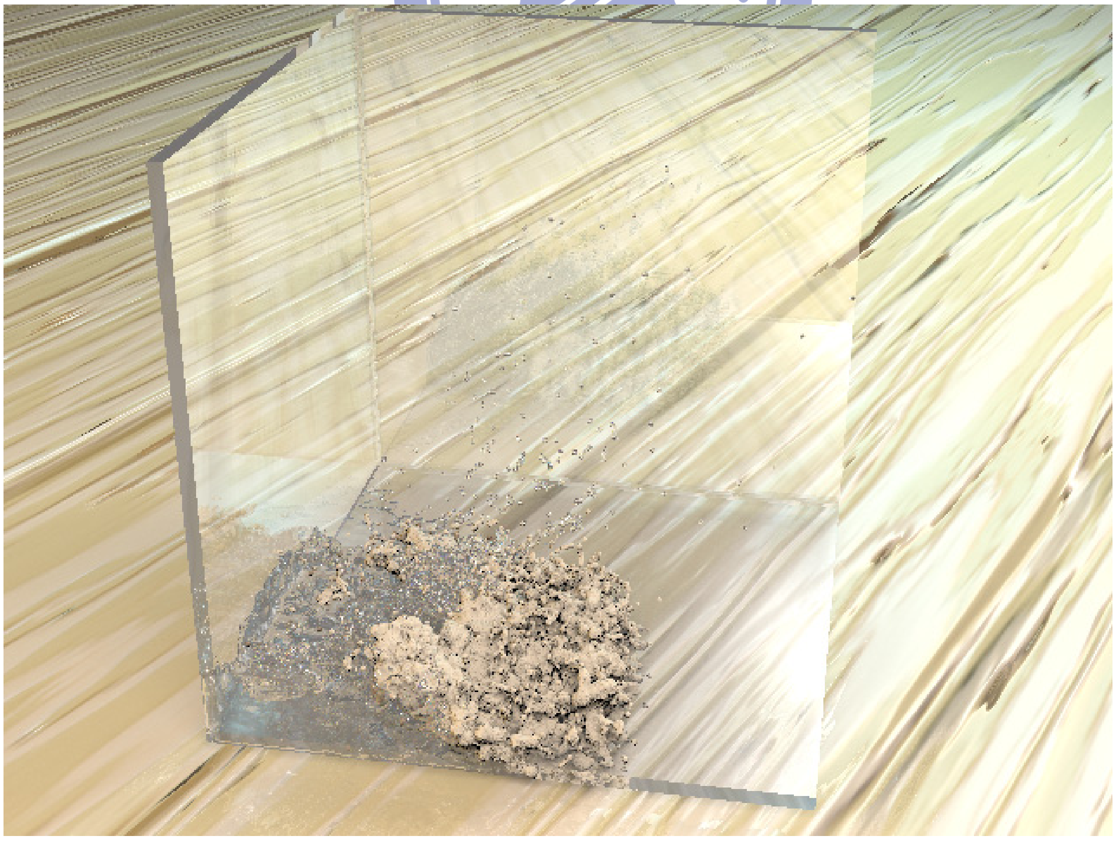


Figure 4.2(r): Step No. 510.

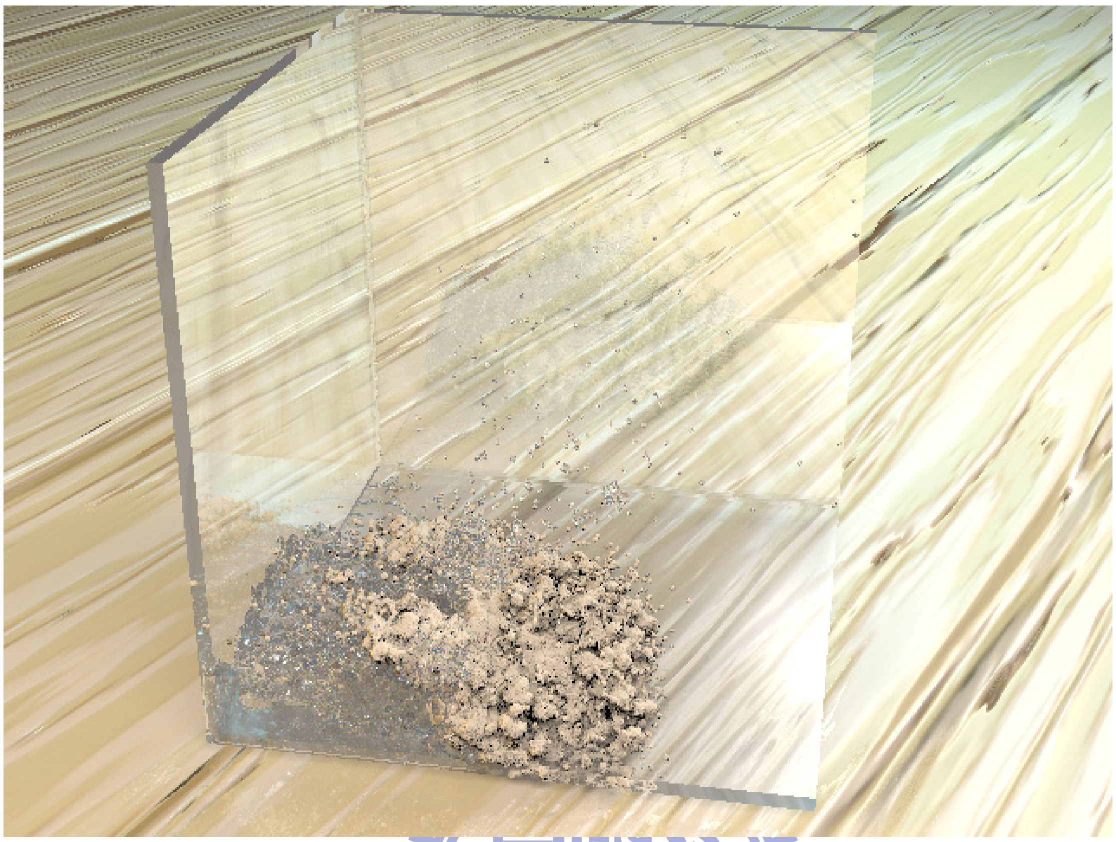


Figure 4.2(s): Step No. 540.

In Figure 4.3(a) to (z) are the same scenes with sand bunny and water armadillos. At the beginning, the simulation consists of 21,456 sand and 15,525 water particles. The average computation time is 3.67s per step.

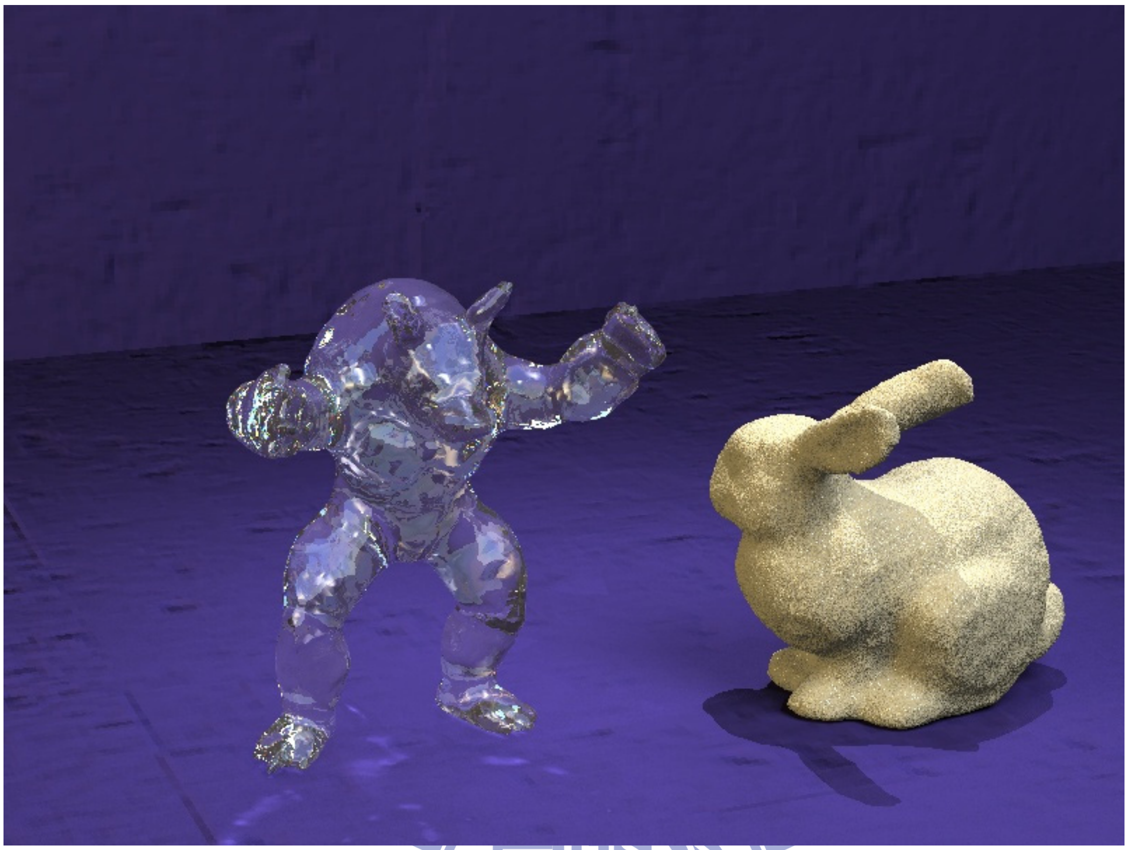


Figure 4.3(a): Step No. 000.

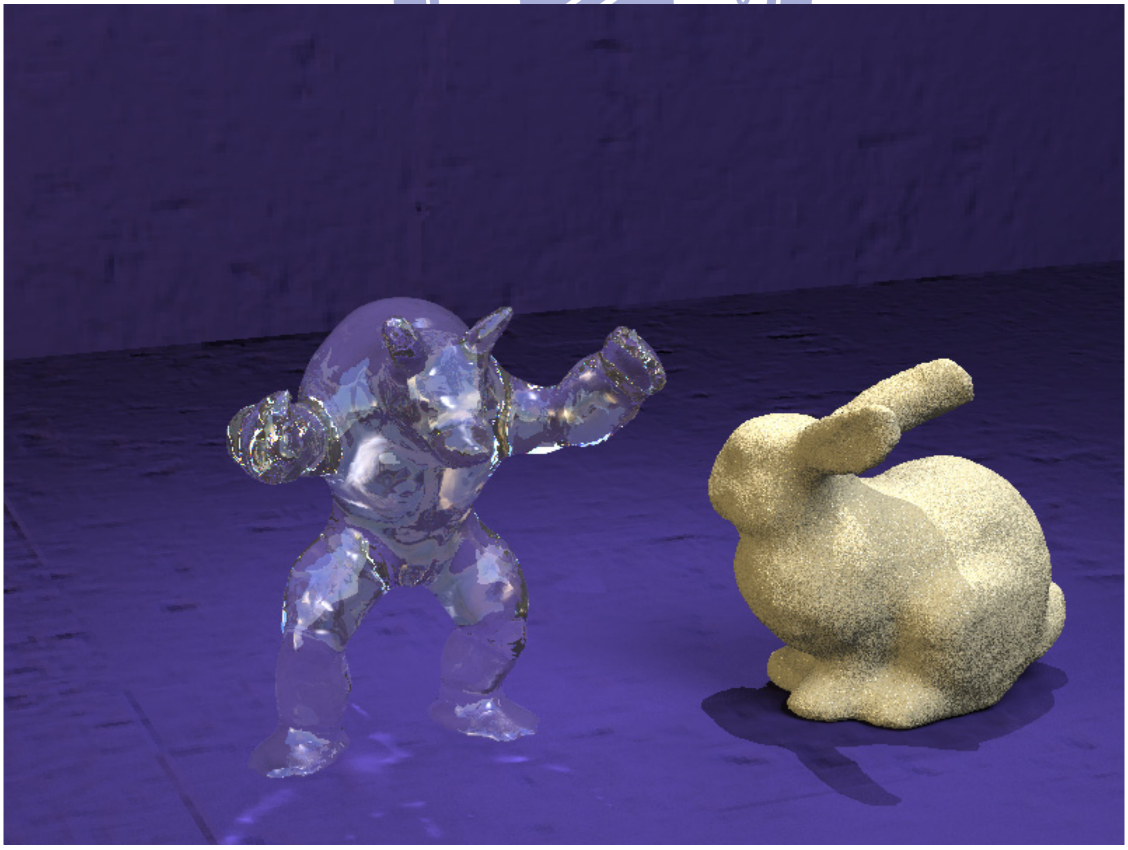


Figure 4.3(b): Step No. 100.

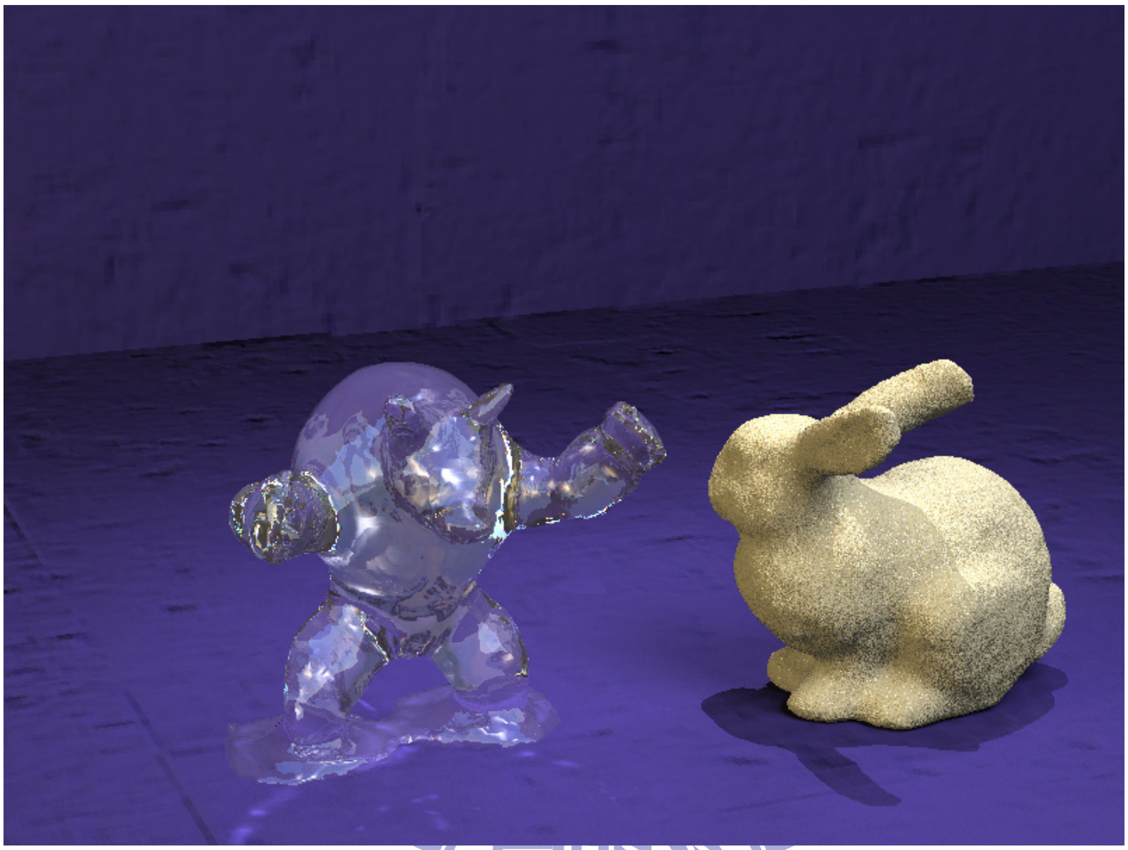


Figure 4.3(c): Step No. 200.

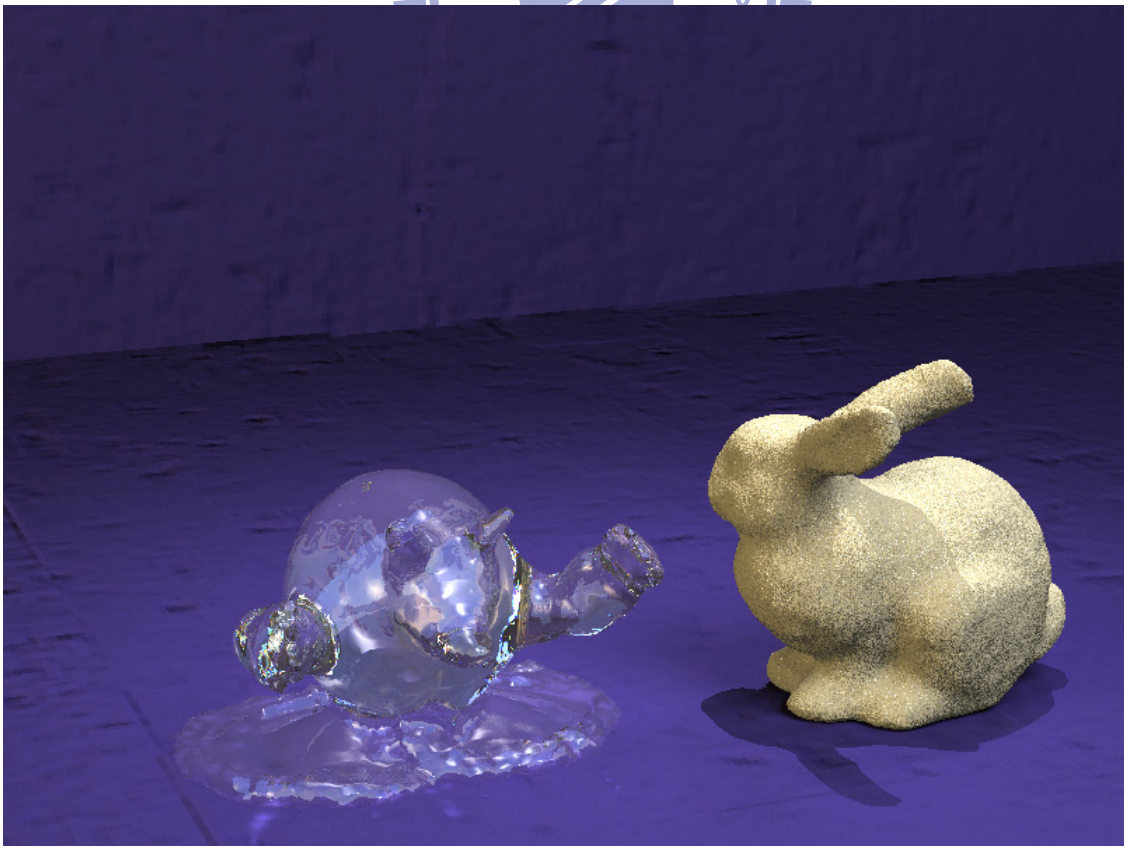


Figure 4.3(d): Step No. 300.

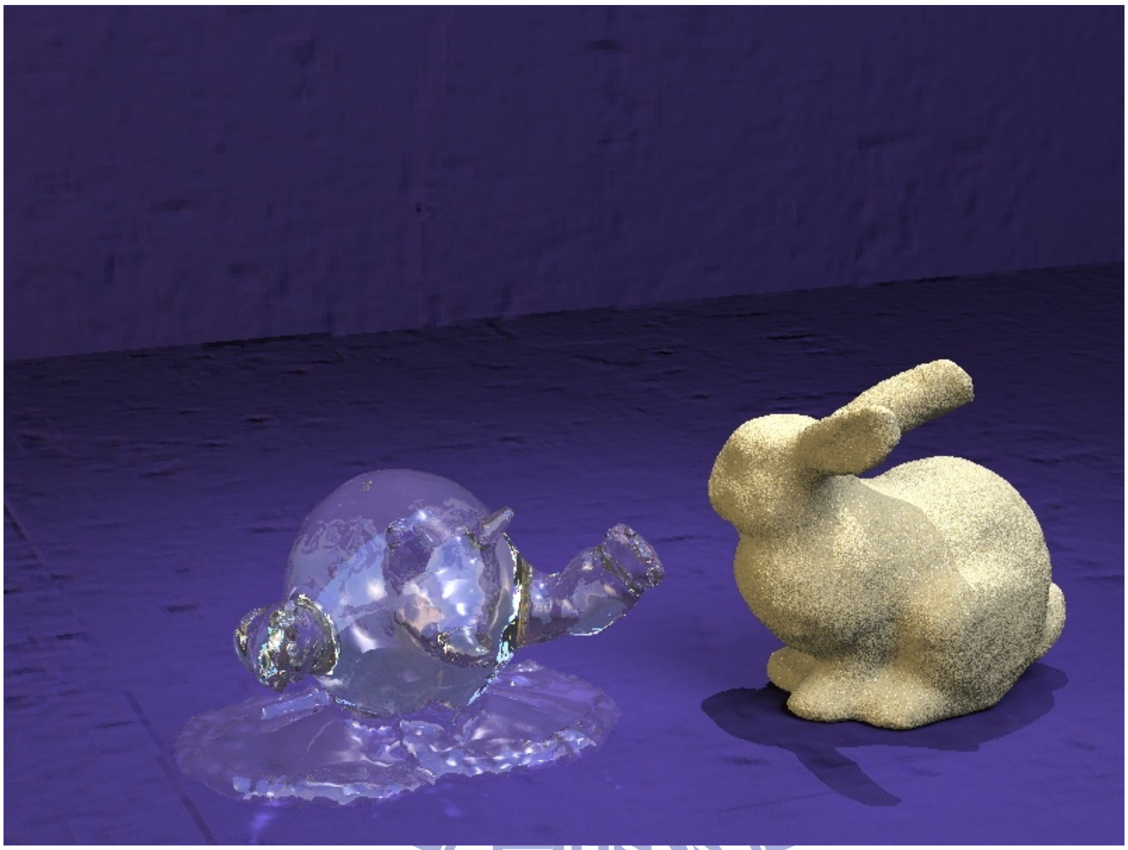


Figure 4.3(e): Step No. 300.

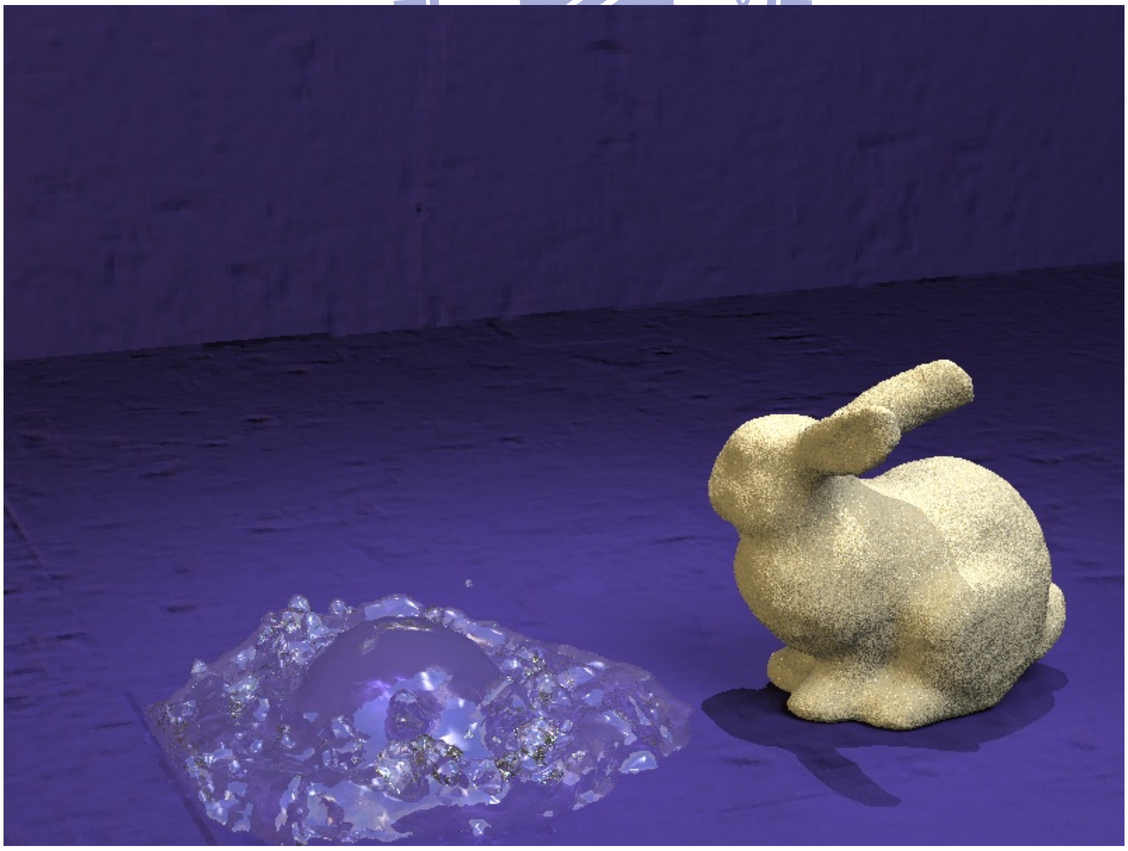


Figure 4.3(f): Step No. 400.

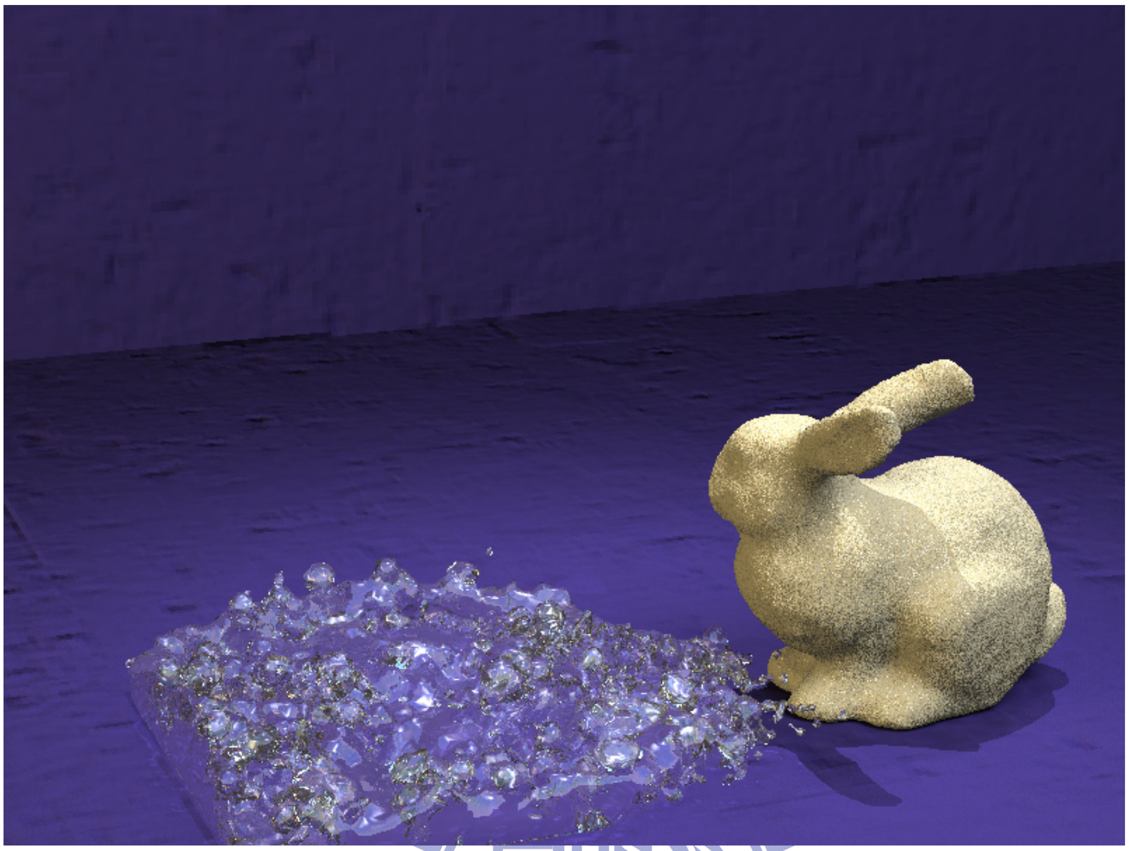


Figure 4.3(g): Step No. 450.

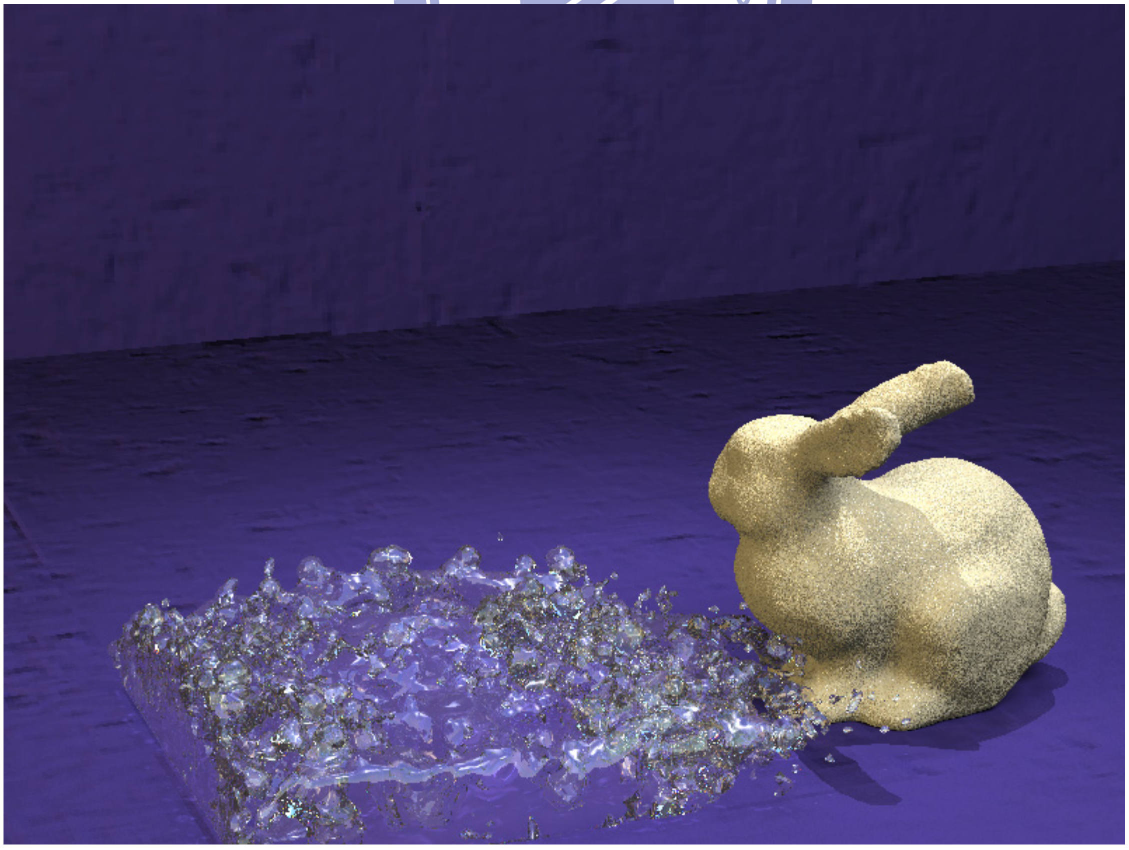


Figure 4.3(h): Step No. 480.

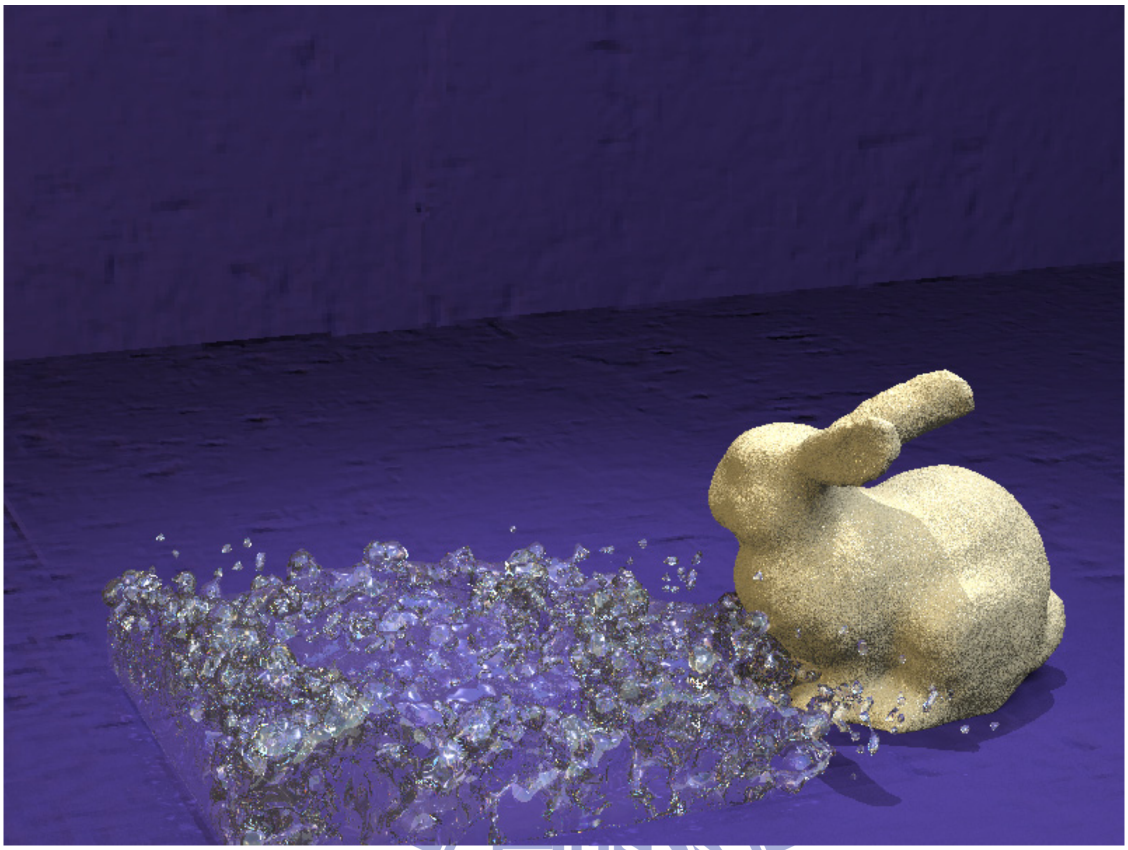


Figure 4.3(i): Step No. 510.

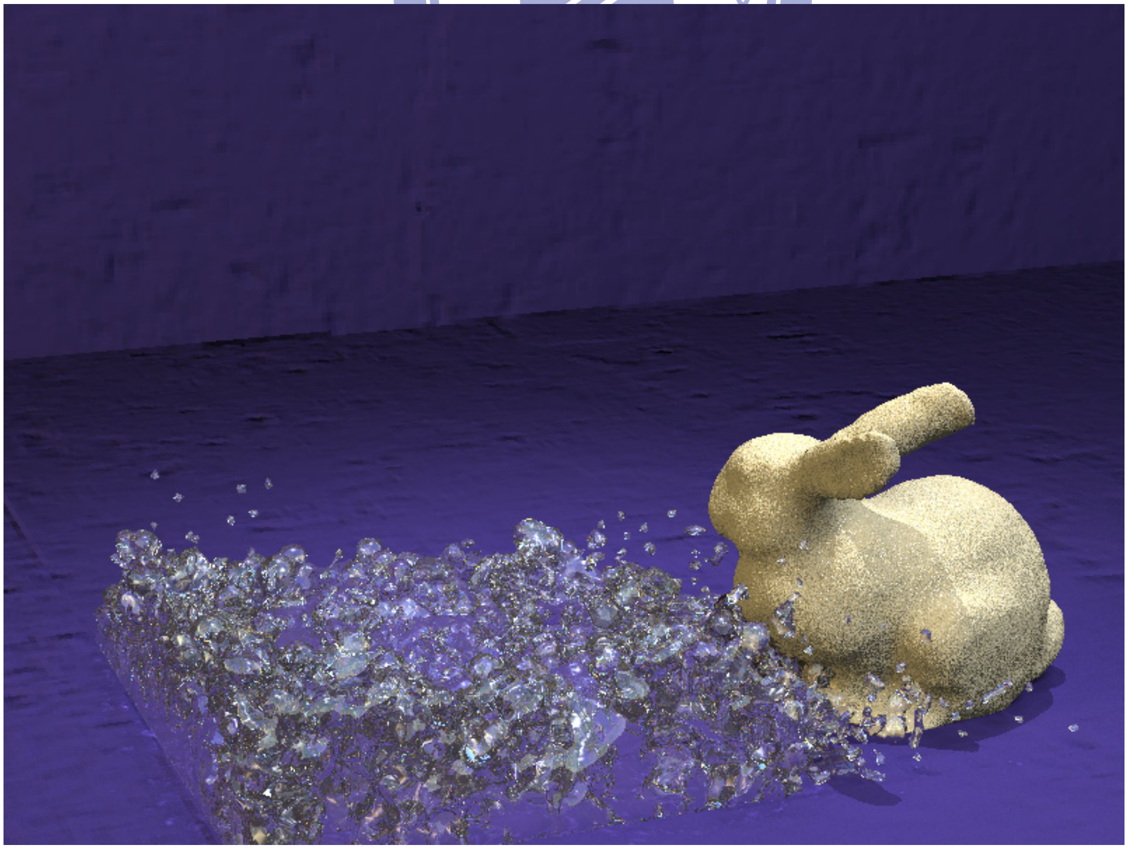


Figure 4.3(j): Step No. 540.

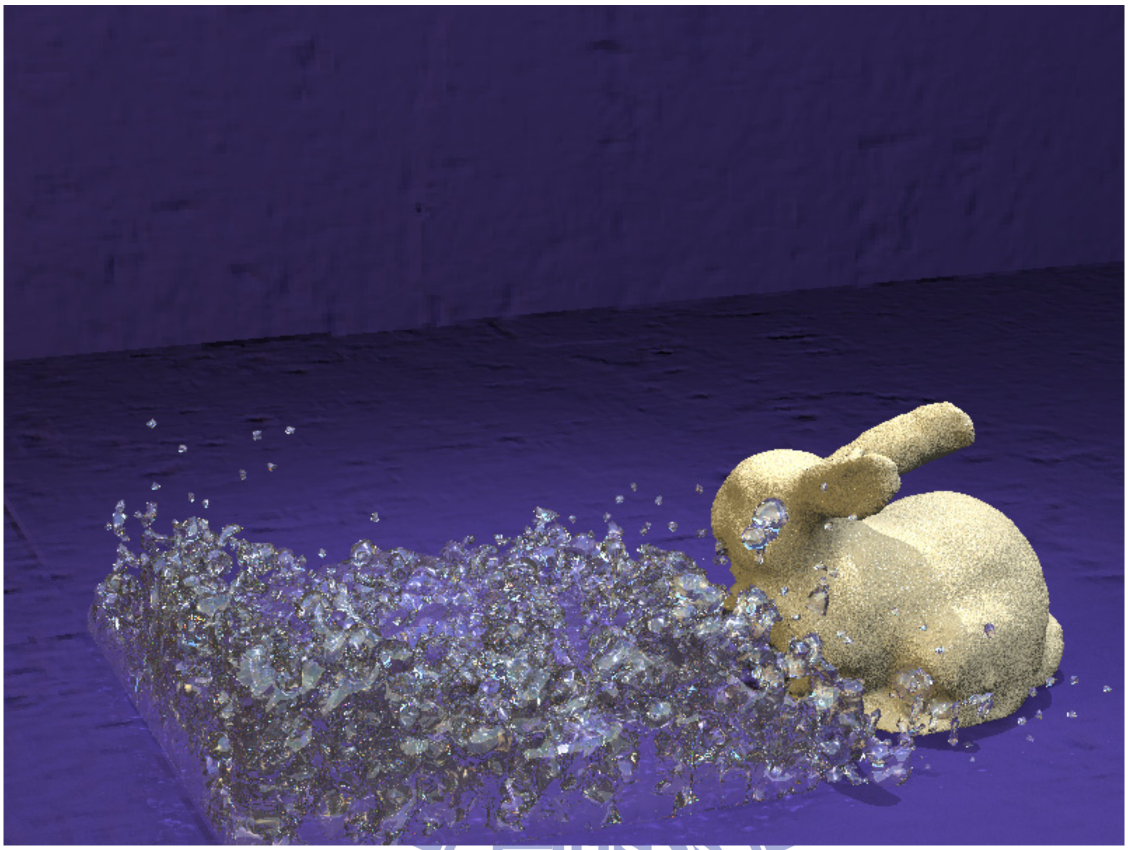


Figure 4.3(k): Step No. 570.

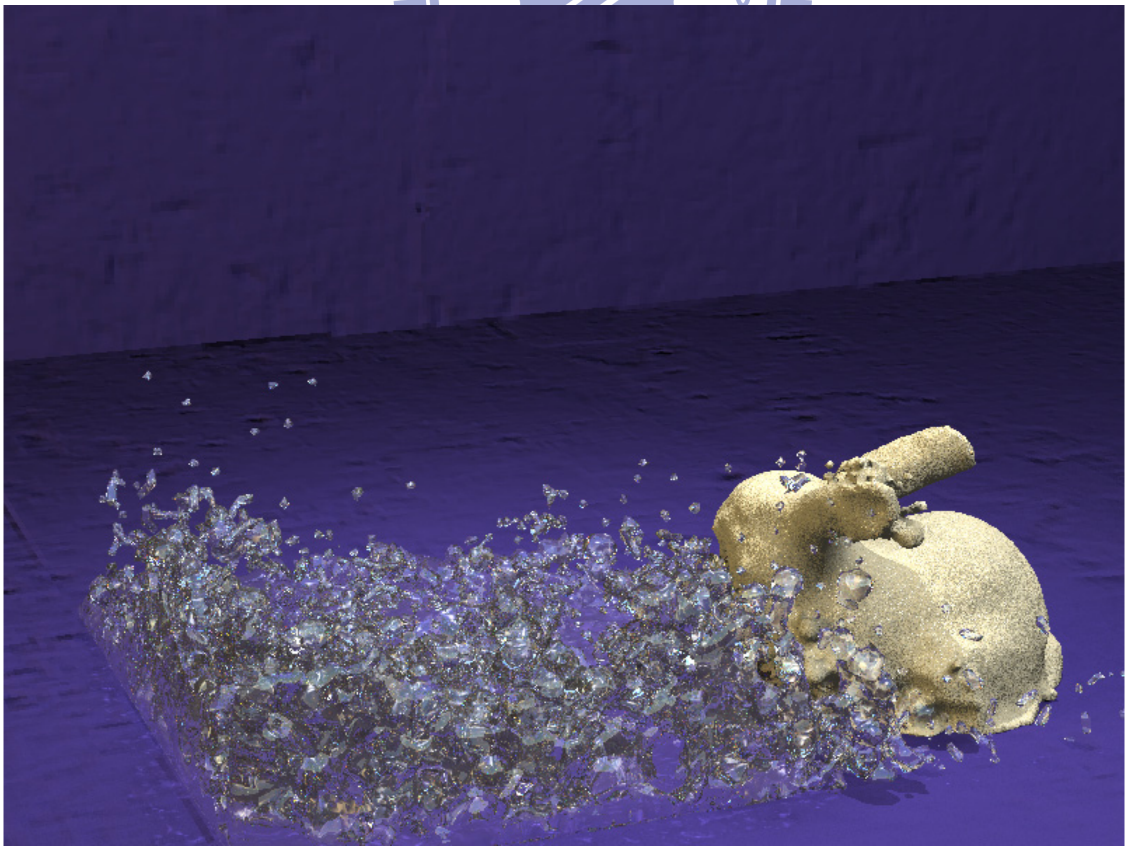


Figure 4.3(l): Step No. 600.

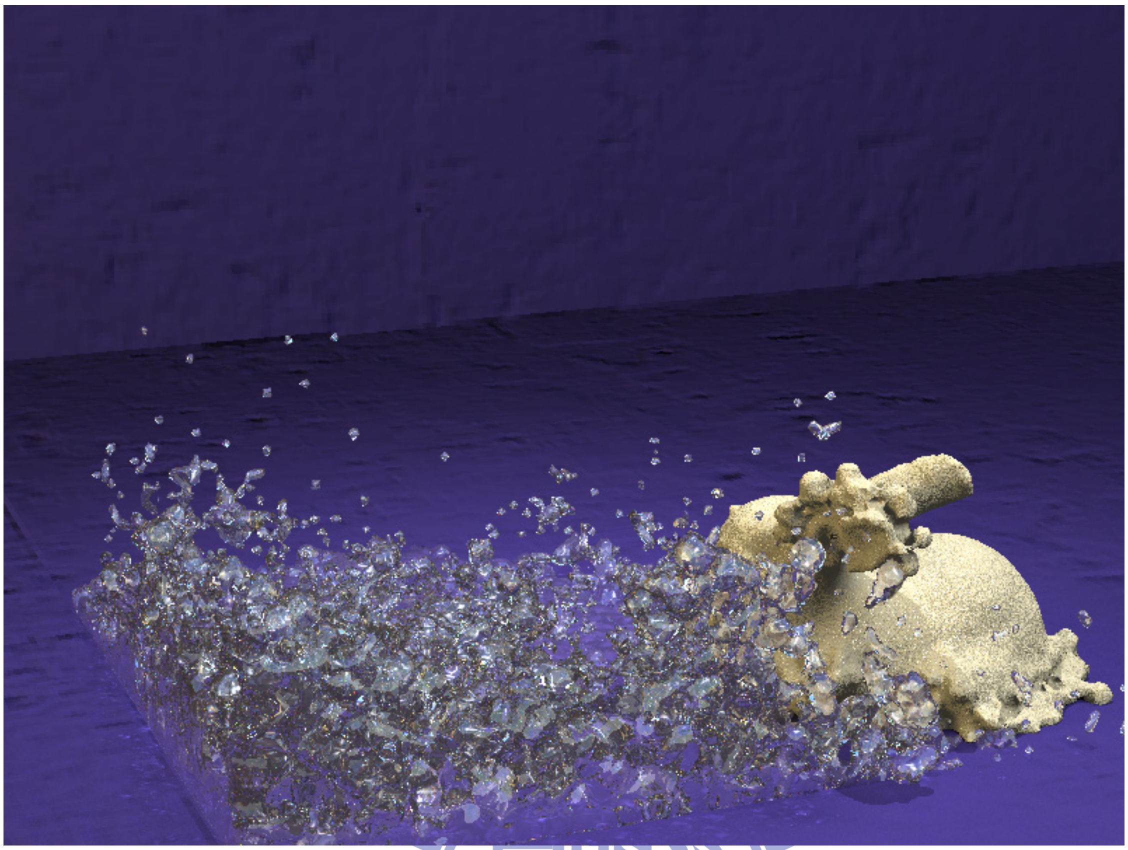


Figure 4.3(m): Step No. 630.

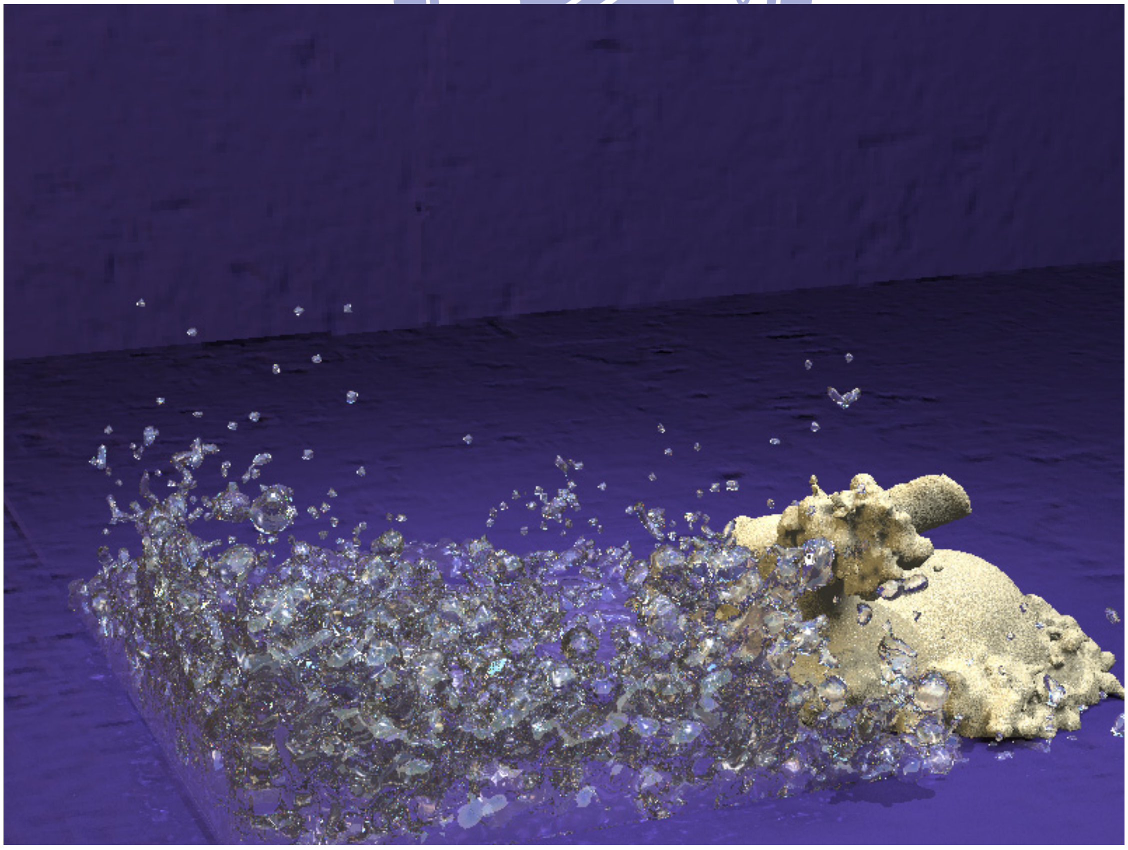


Figure 4.3(n): Step No. 660.

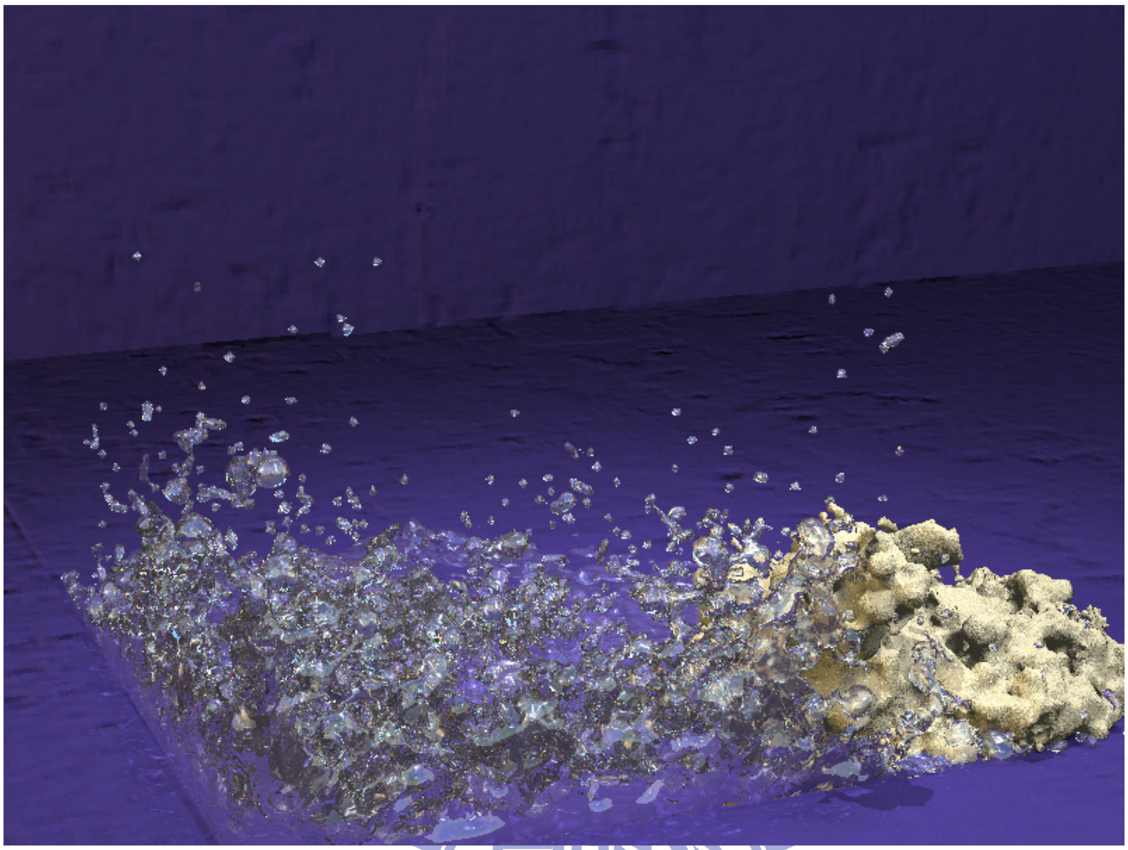


Figure 4.3(o): Step No. 690.

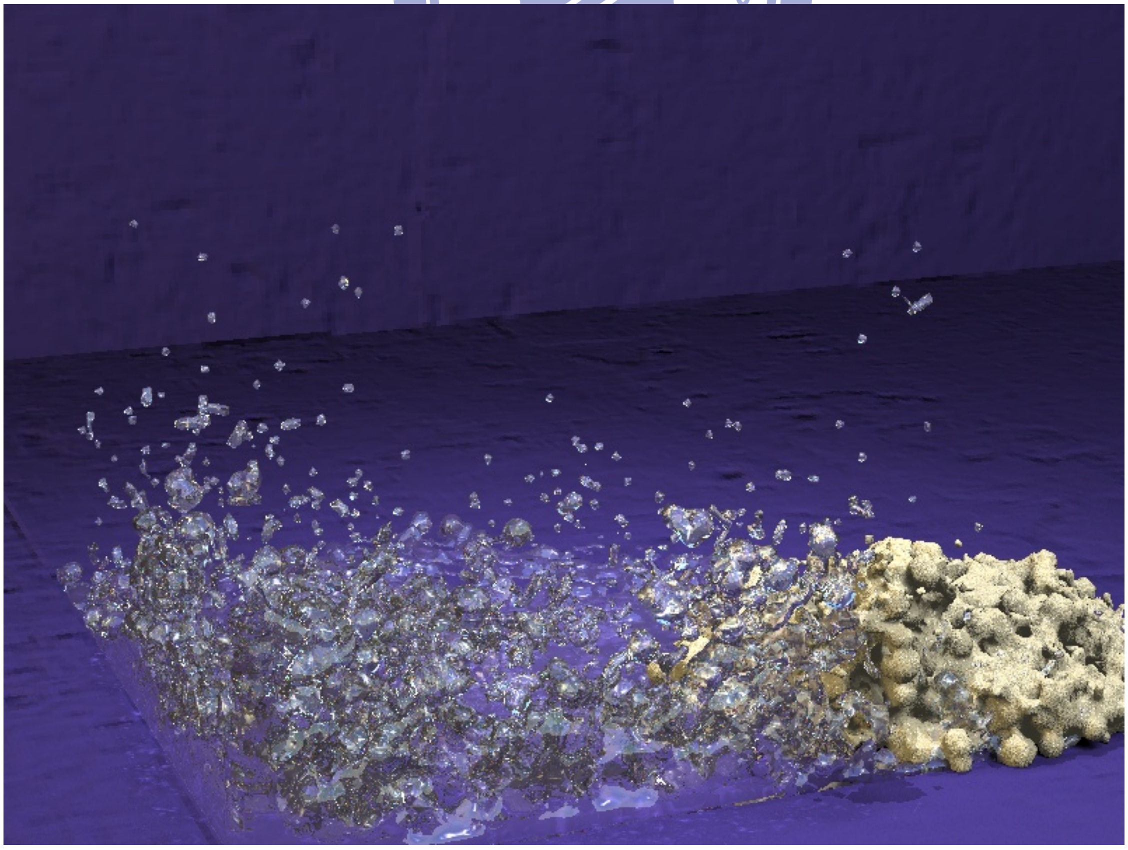


Figure 4.3(p): Step No. 720.

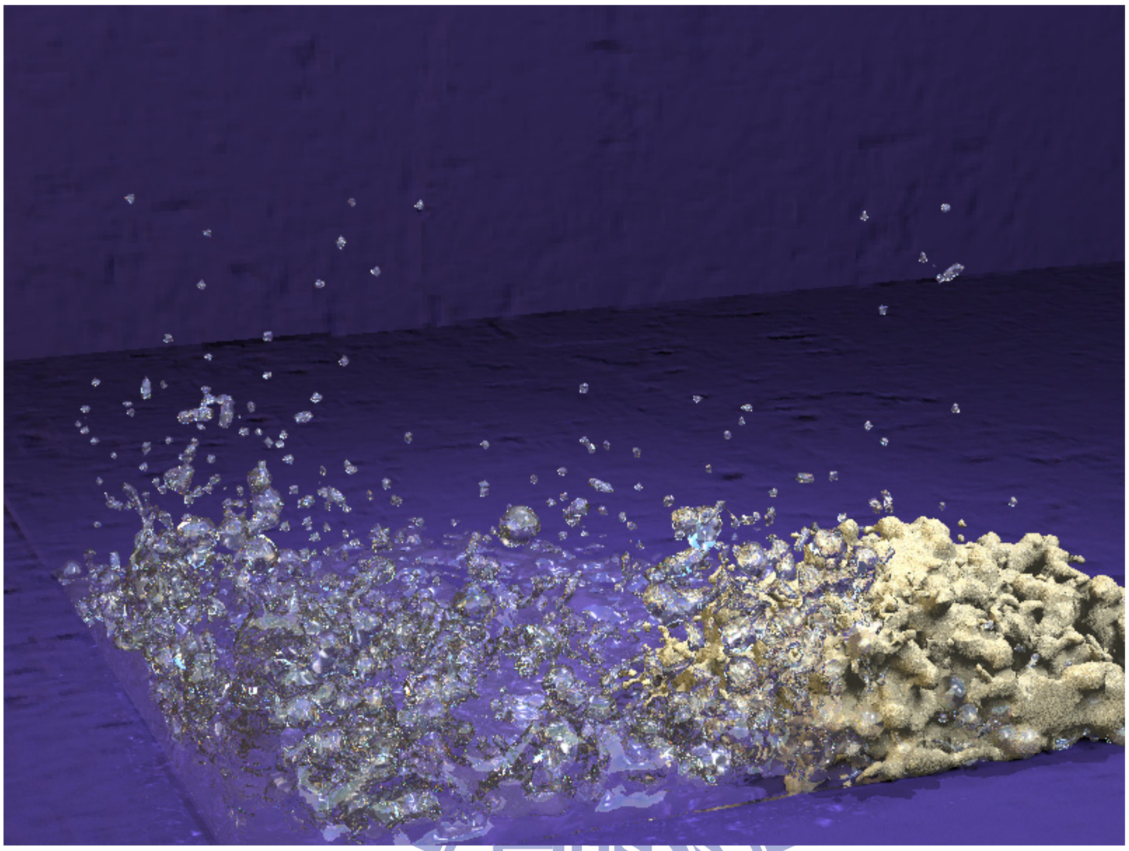


Figure 4.3(q): Step No. 750.

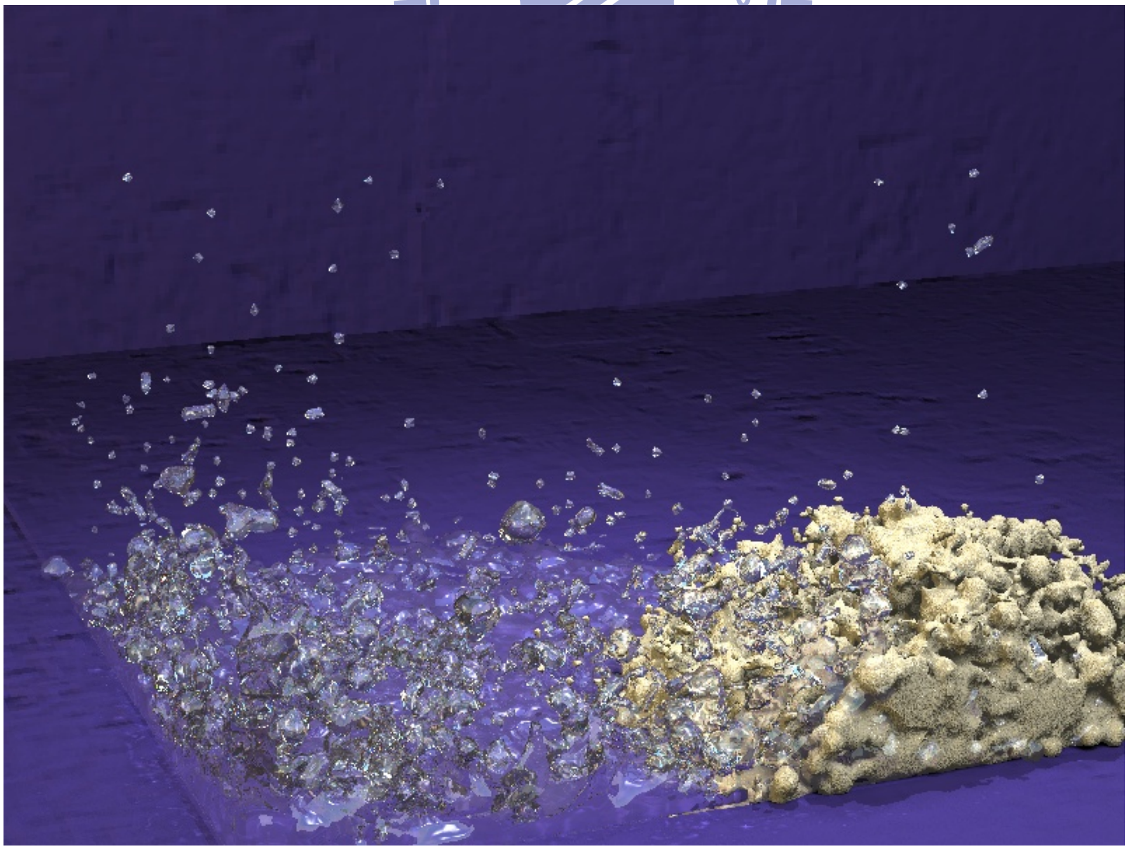


Figure 4.3(r): Step No. 780.

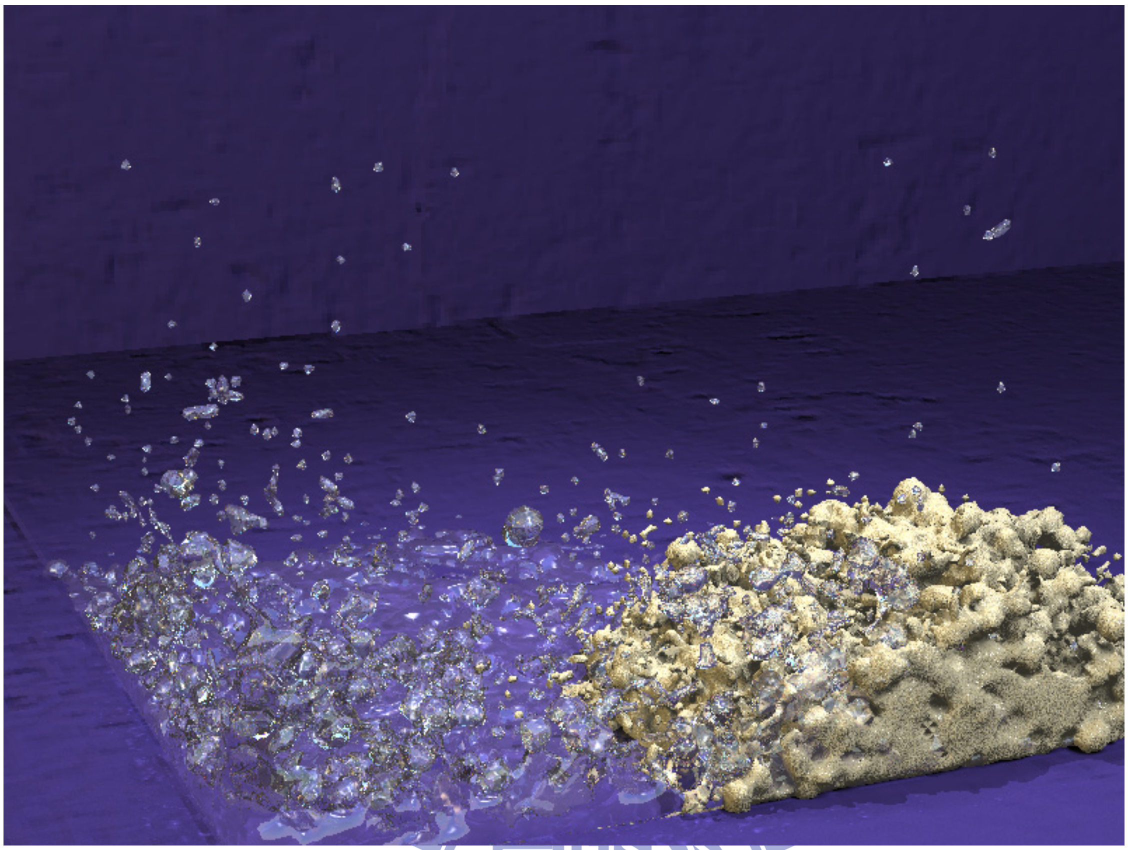


Figure 4.3(s): Step No. 800.

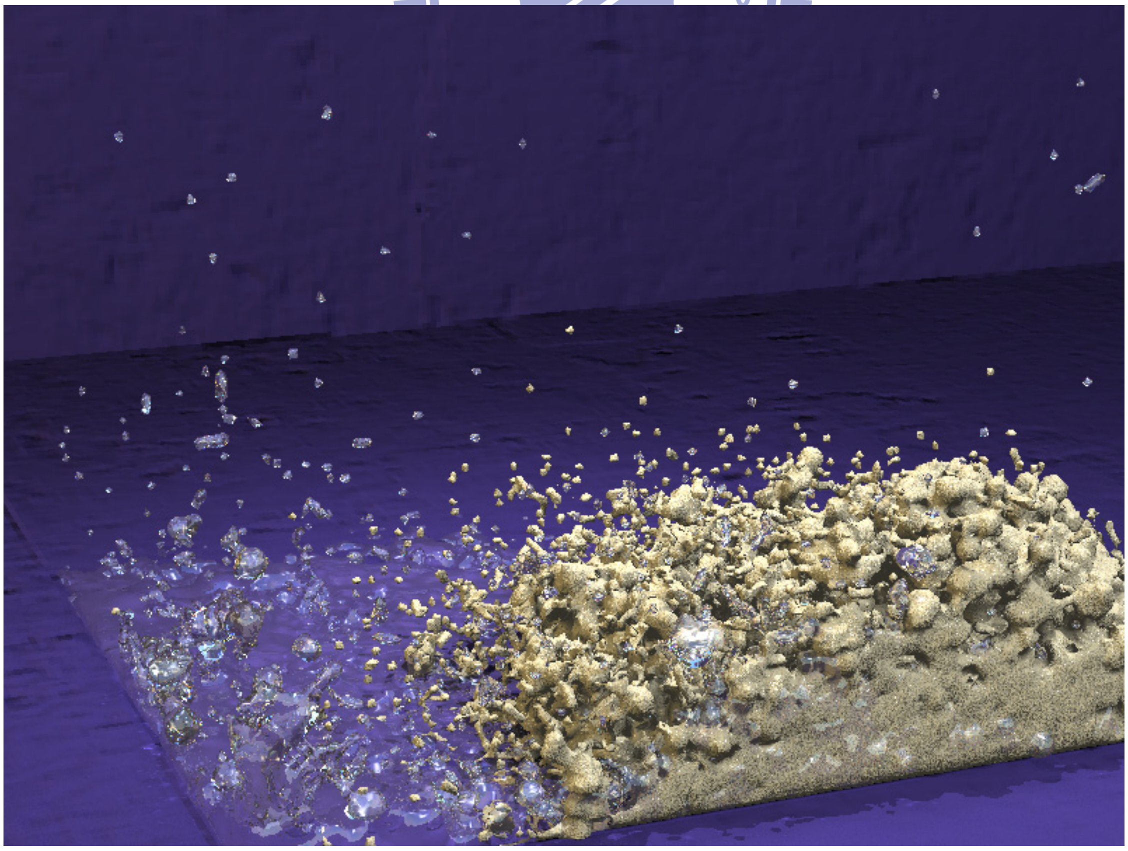


Figure 4.3(t): Step No. 900.

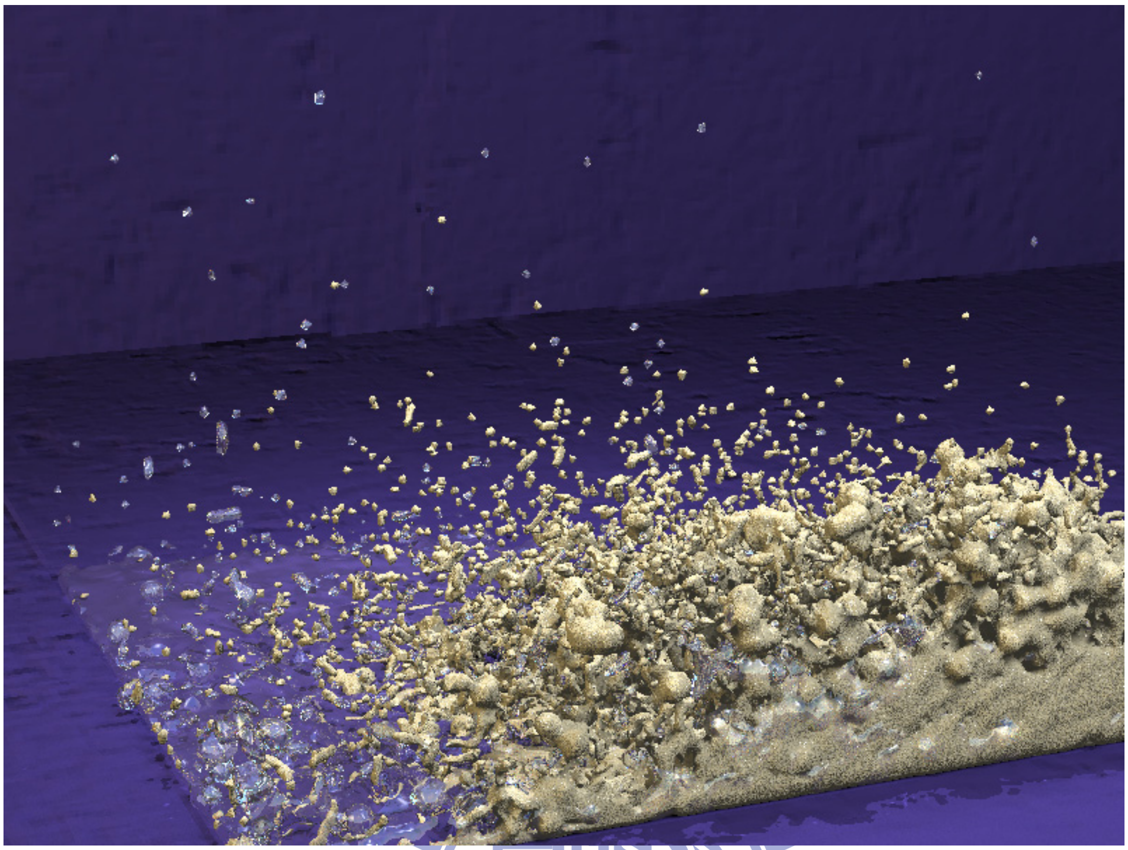


Figure 4.3(u): Step No. 1000.

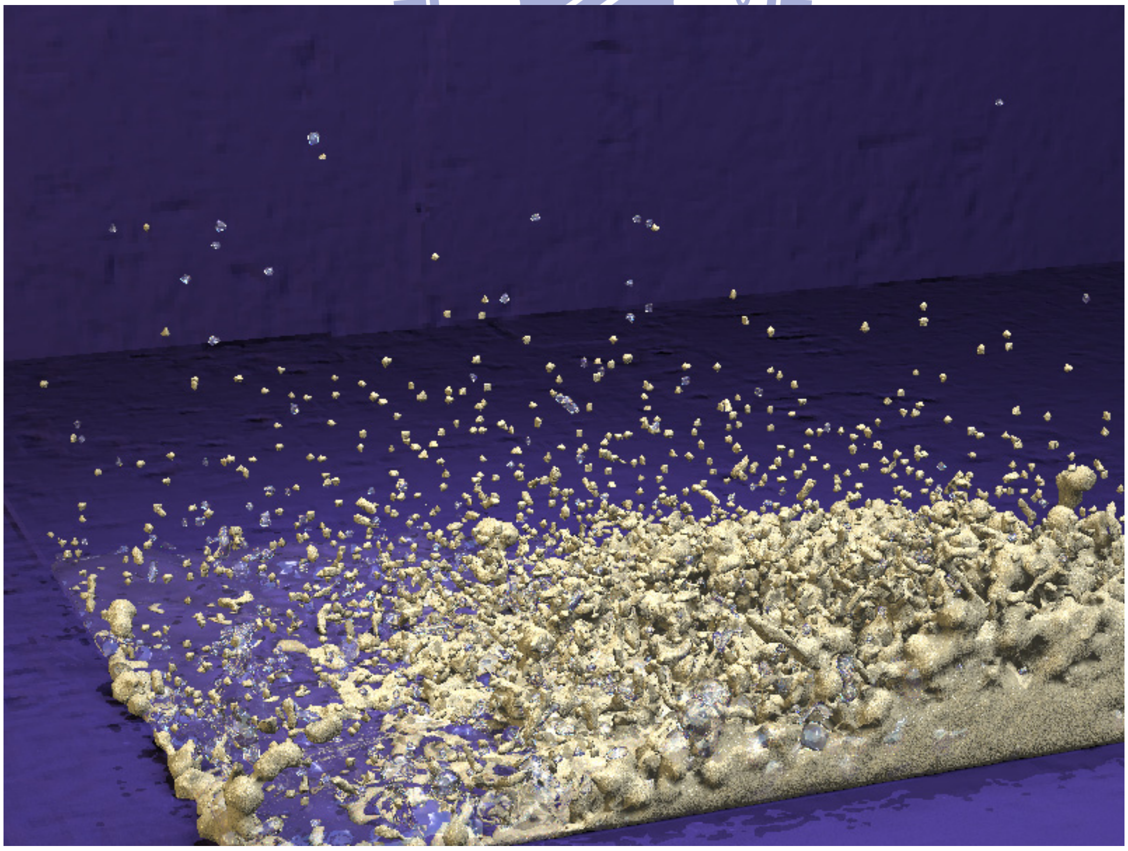


Figure 4.3(v): Step No. 1100.

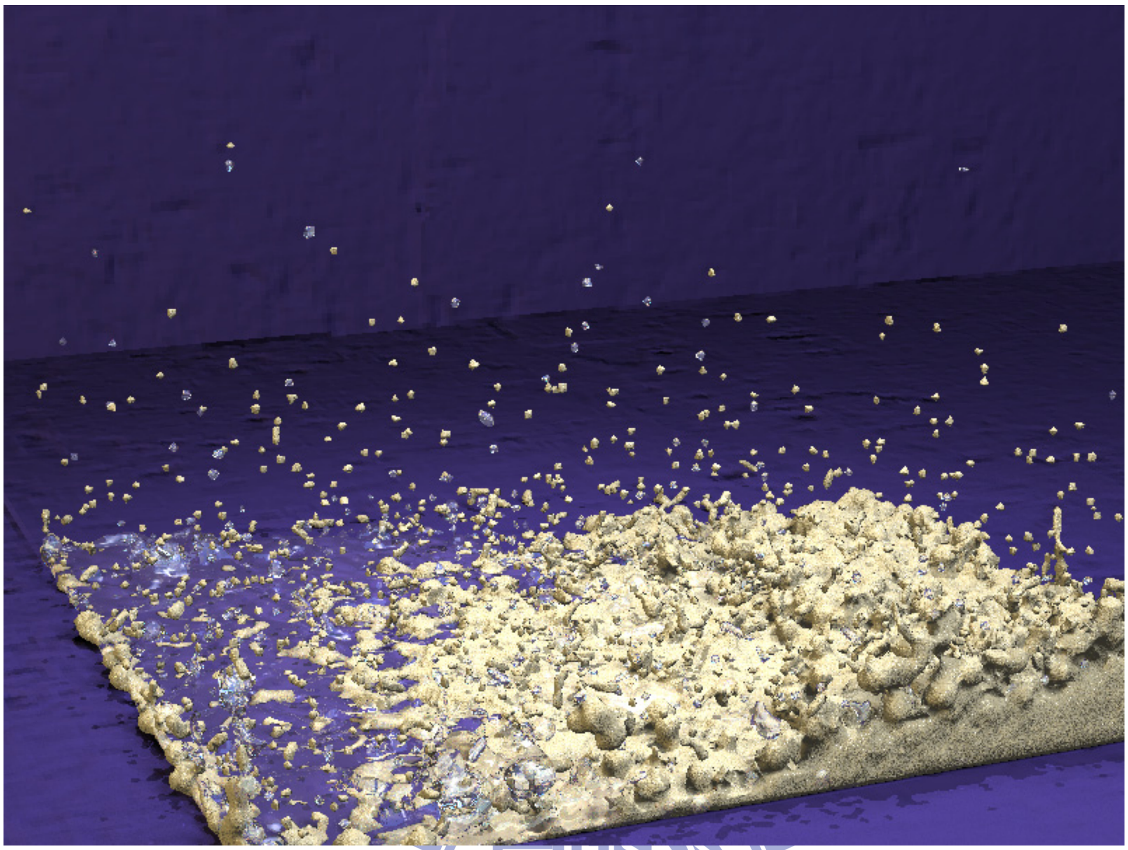


Figure 4.3(w): Step No. 1200.

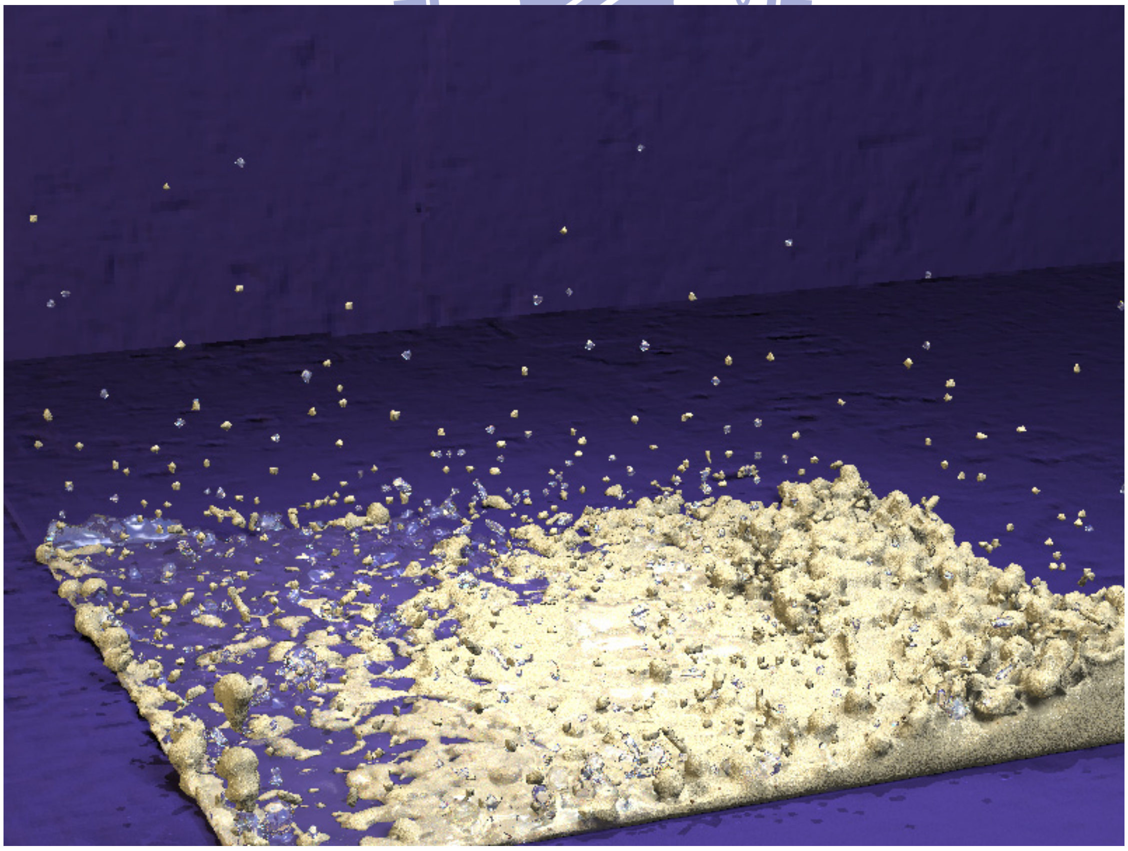


Figure 4.3(x): Step No. 1300.

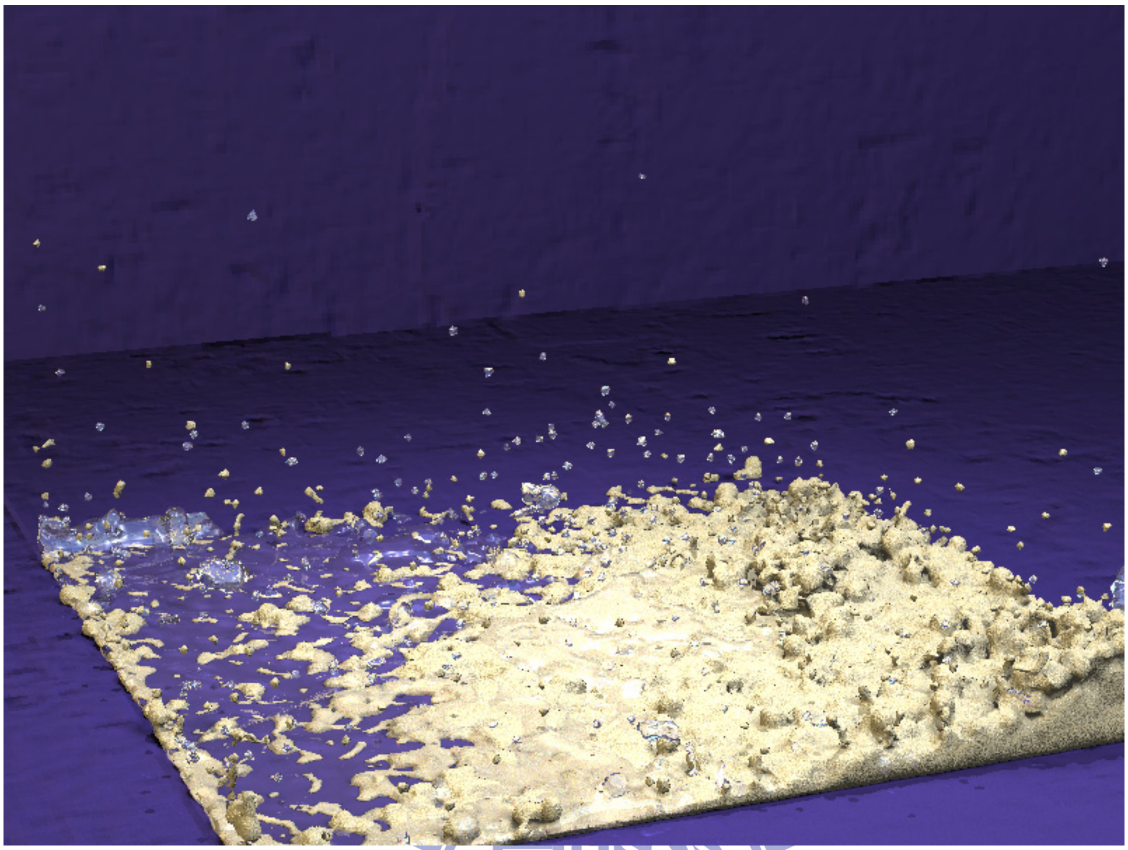


Figure 4.3(y): Step No. 1400.

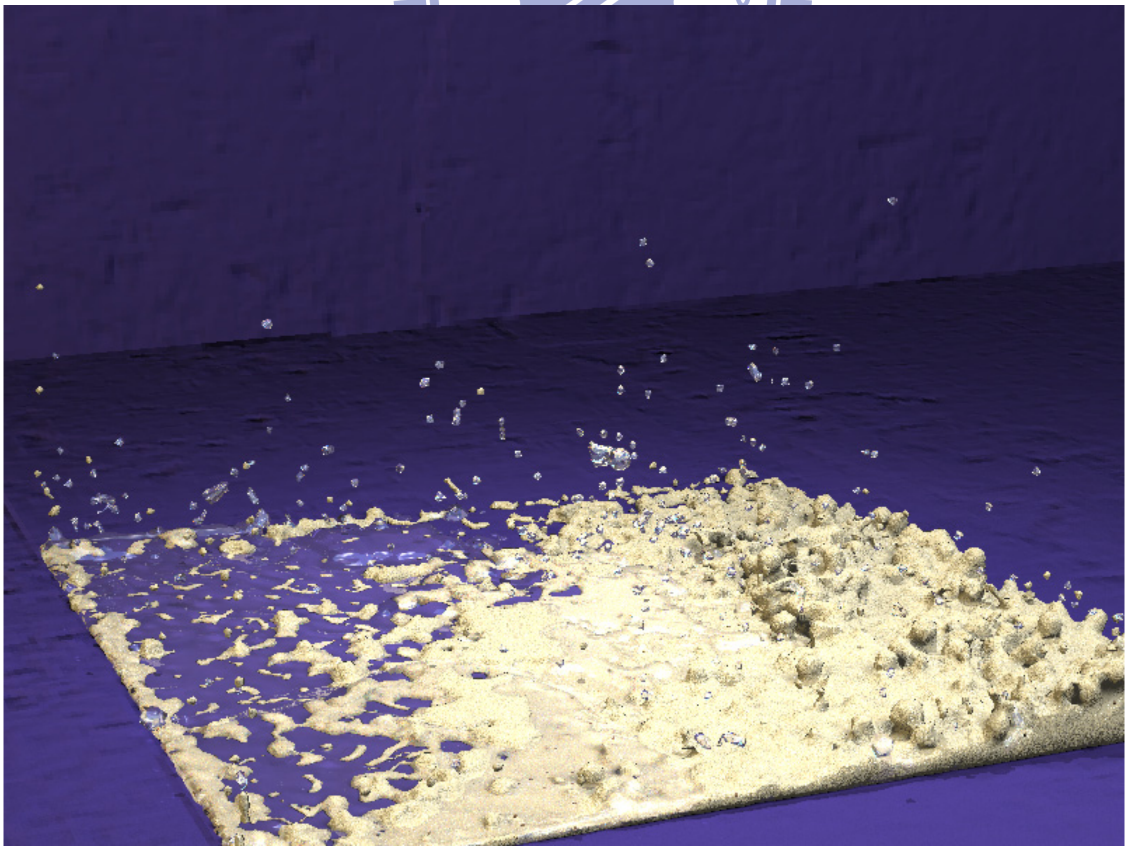


Figure 4.3(z): Step No. 1500.

Figures 4.4(a) to (z) are the same scene with sand Bunny and water Winnie. At the beginning, the simulation consists of 21,456 sand and 28,116 water particles. The average computation time was 9.87s per step.

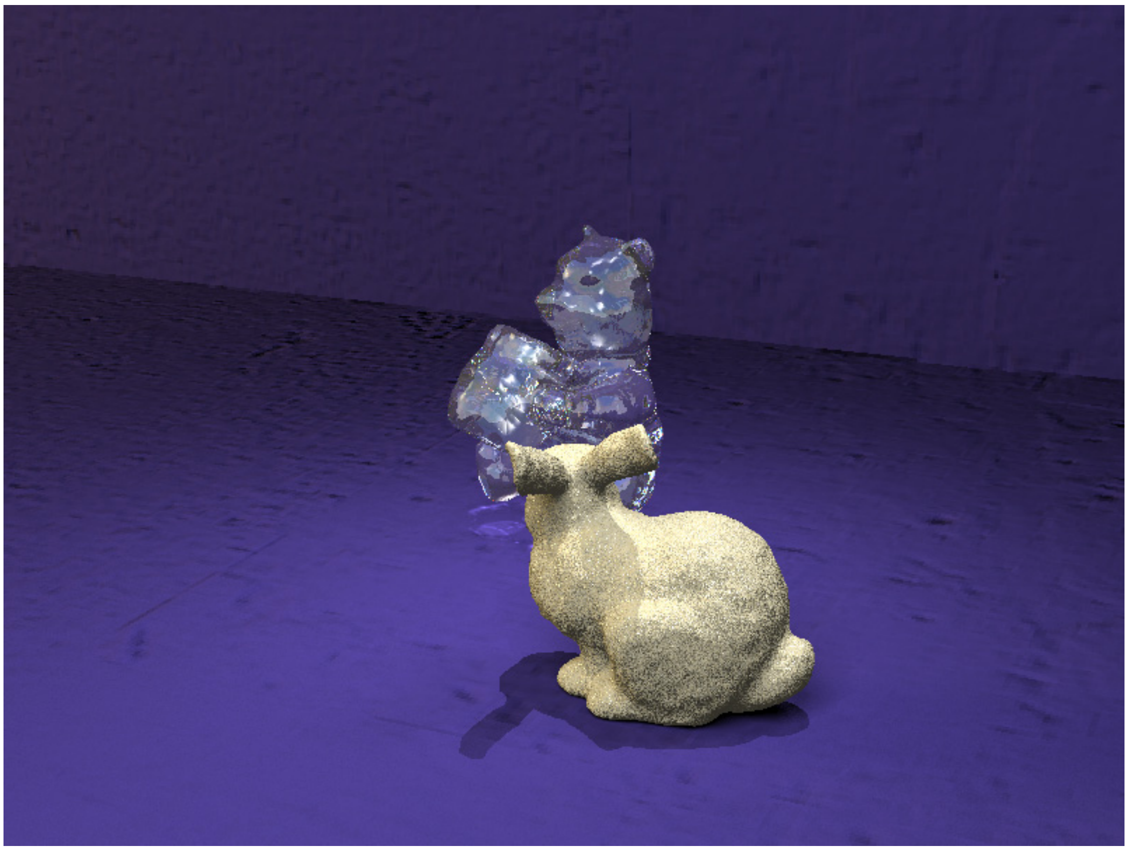


Figure 4.4(a): Step No. 000.

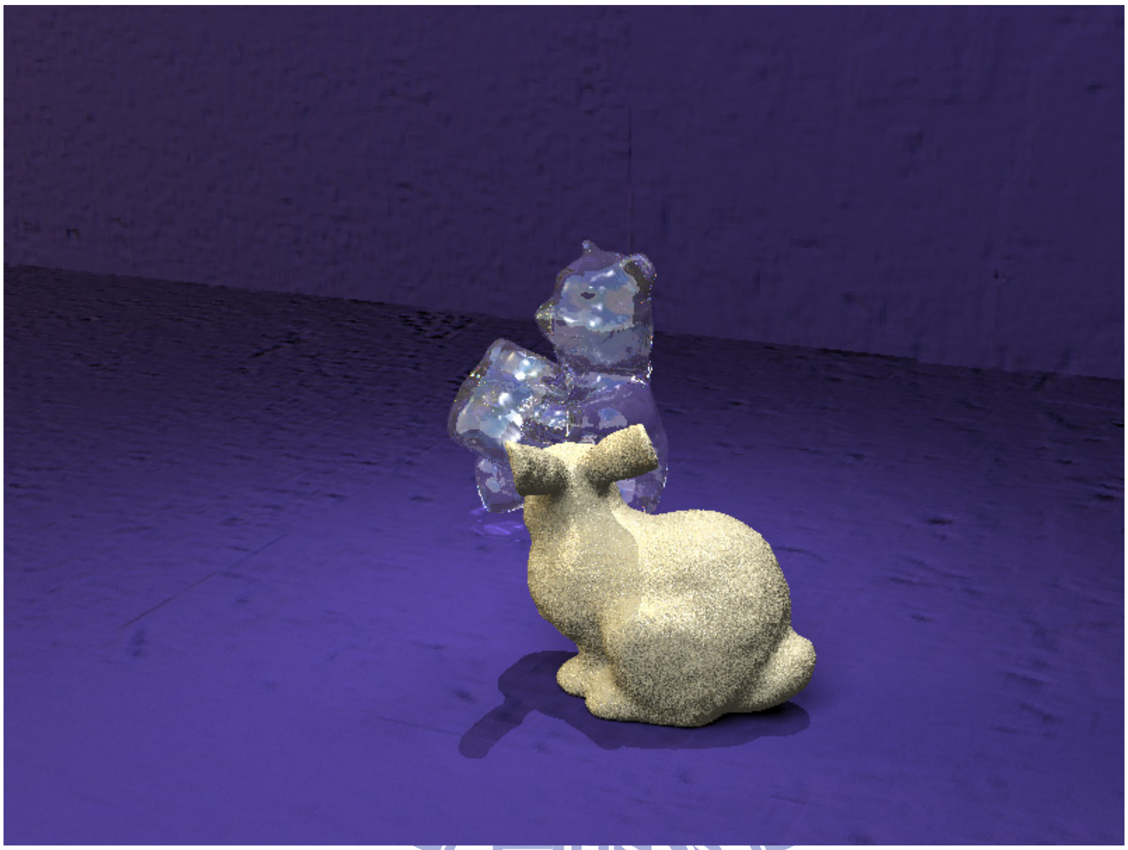


Figure 4.4(b): Step No. 100.

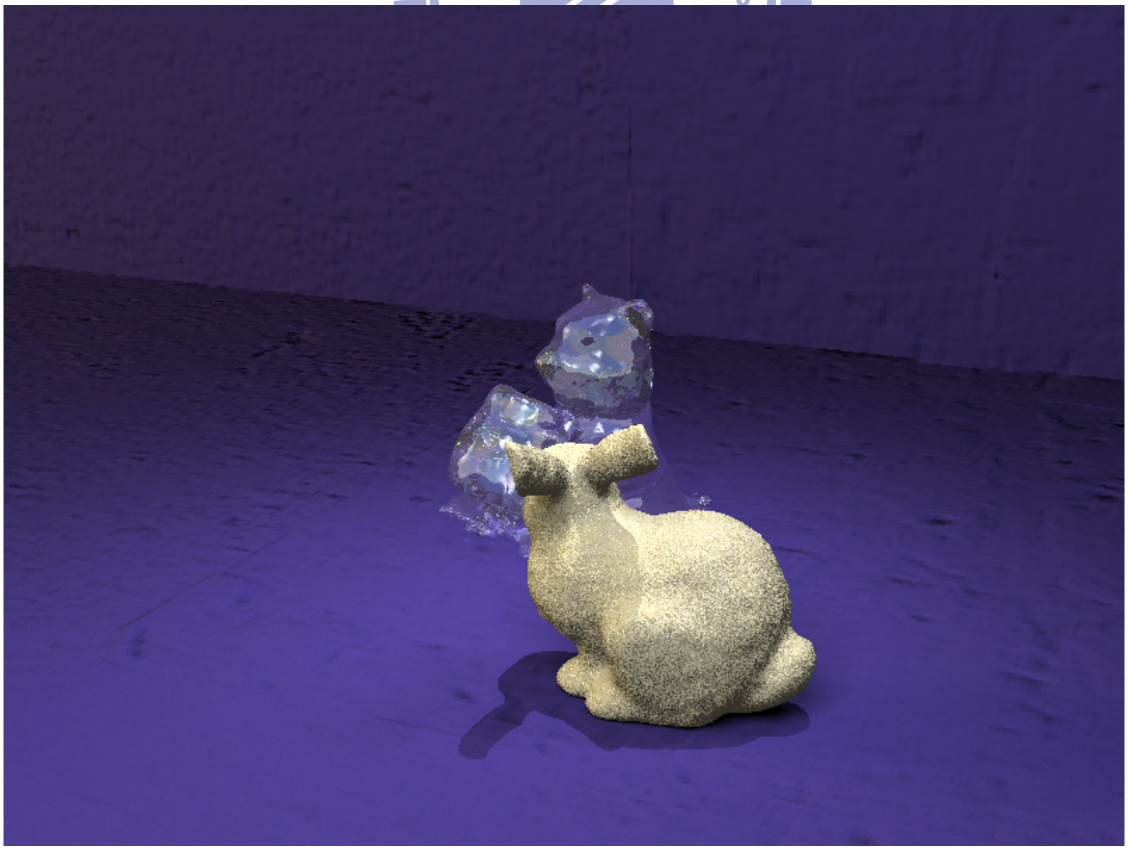


Figure 4.4(c): Step No. 200.

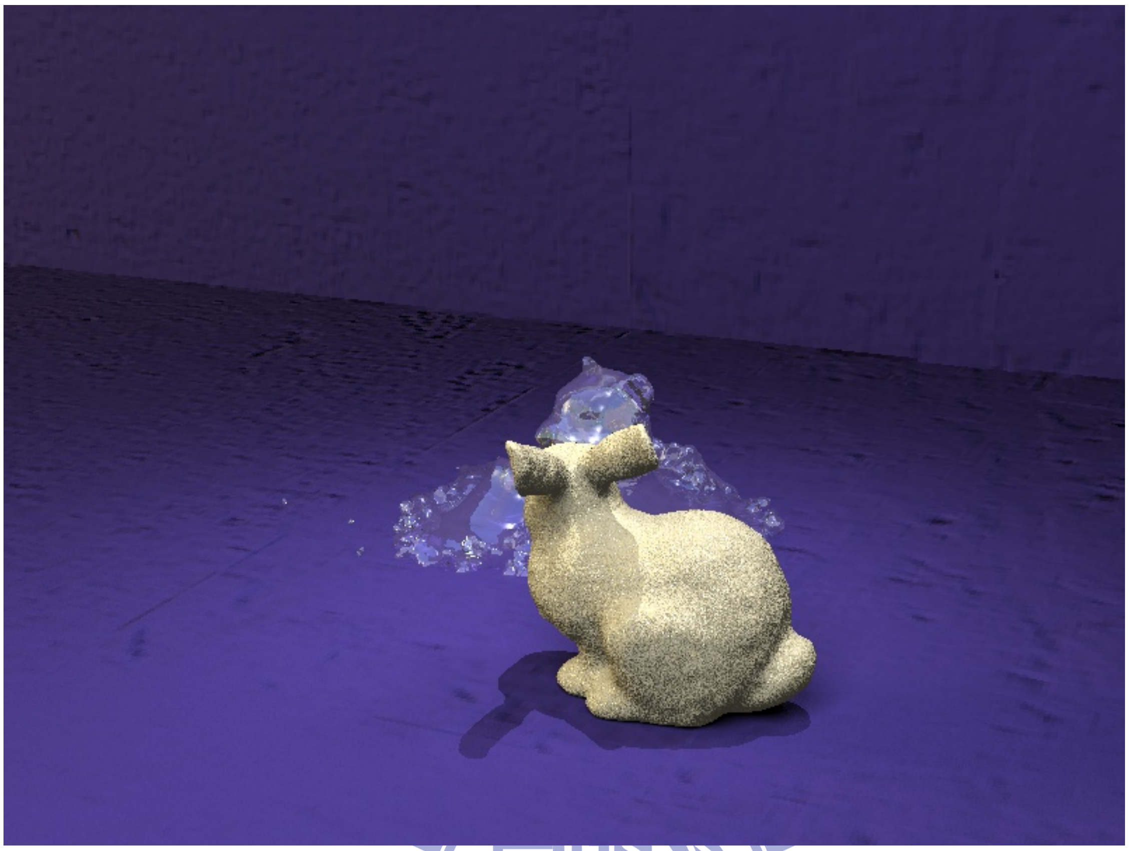


Figure 4.4(d): Step No. 300.

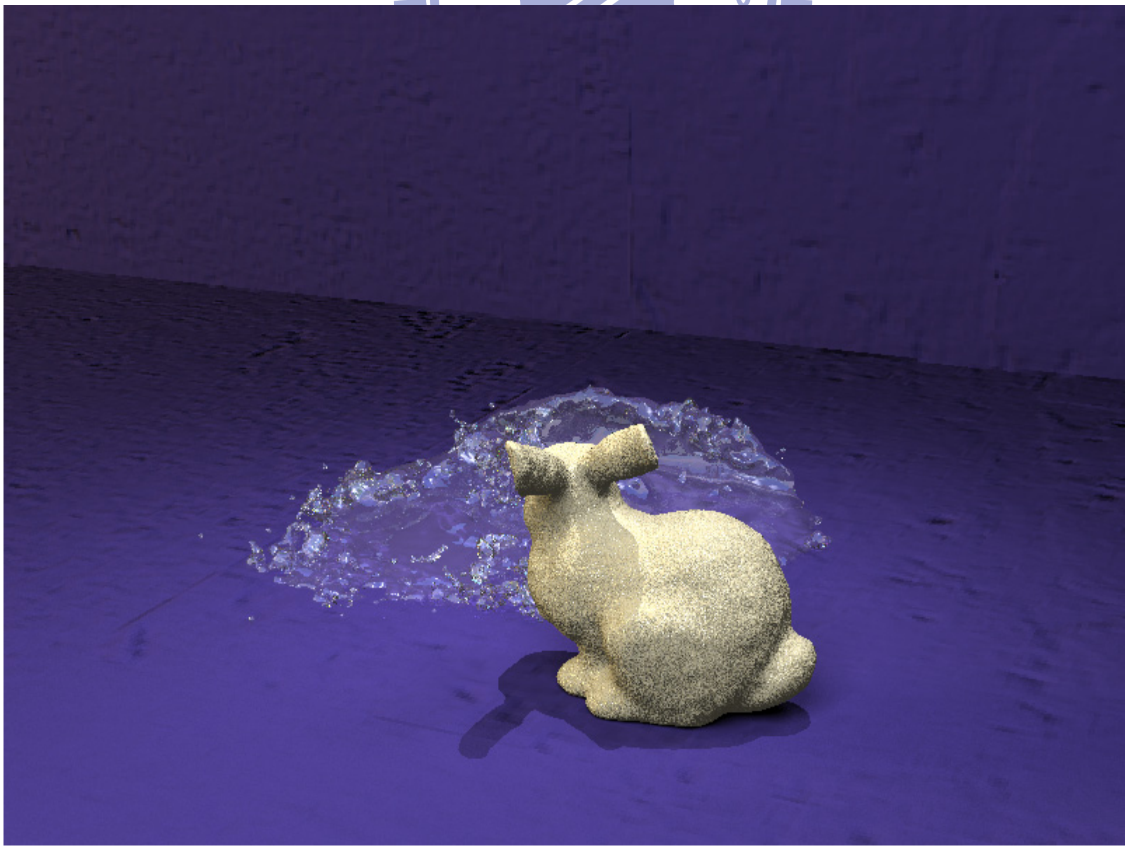


Figure 4.4(e): Step No. 400.

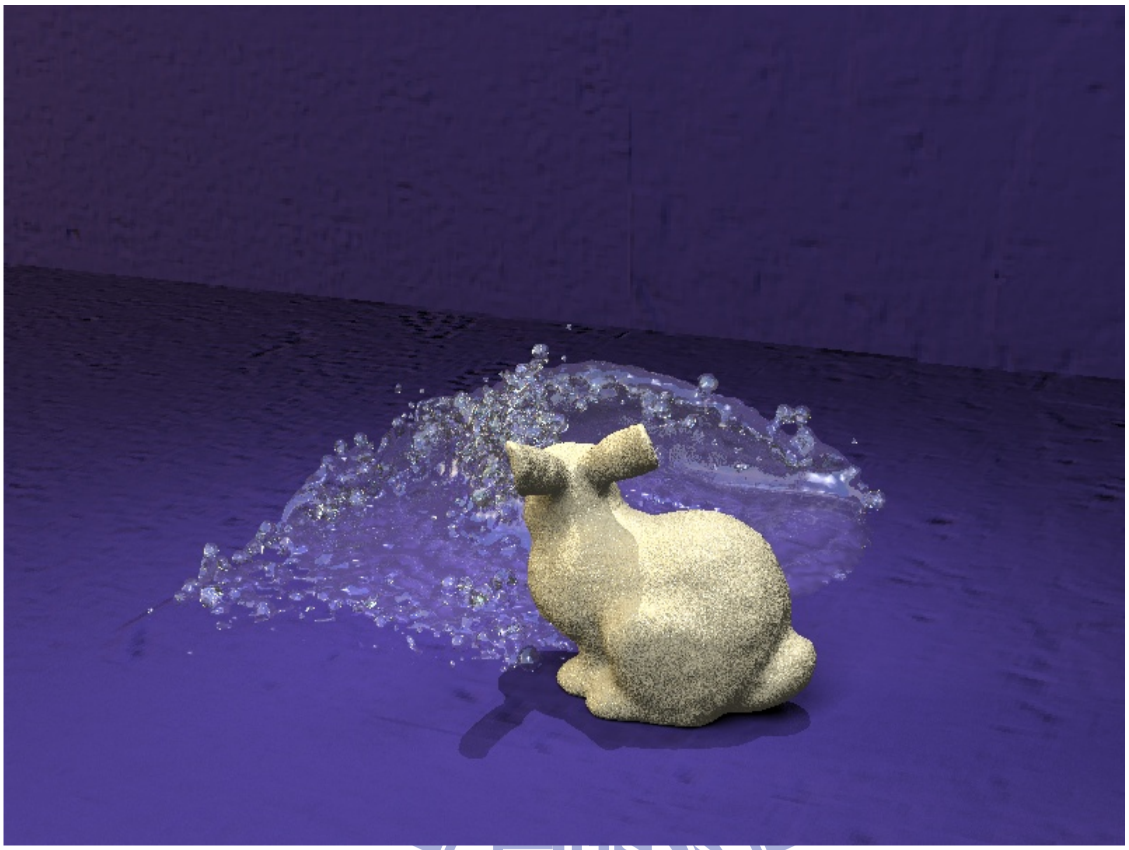


Figure 4.4(f): Step No. 500.

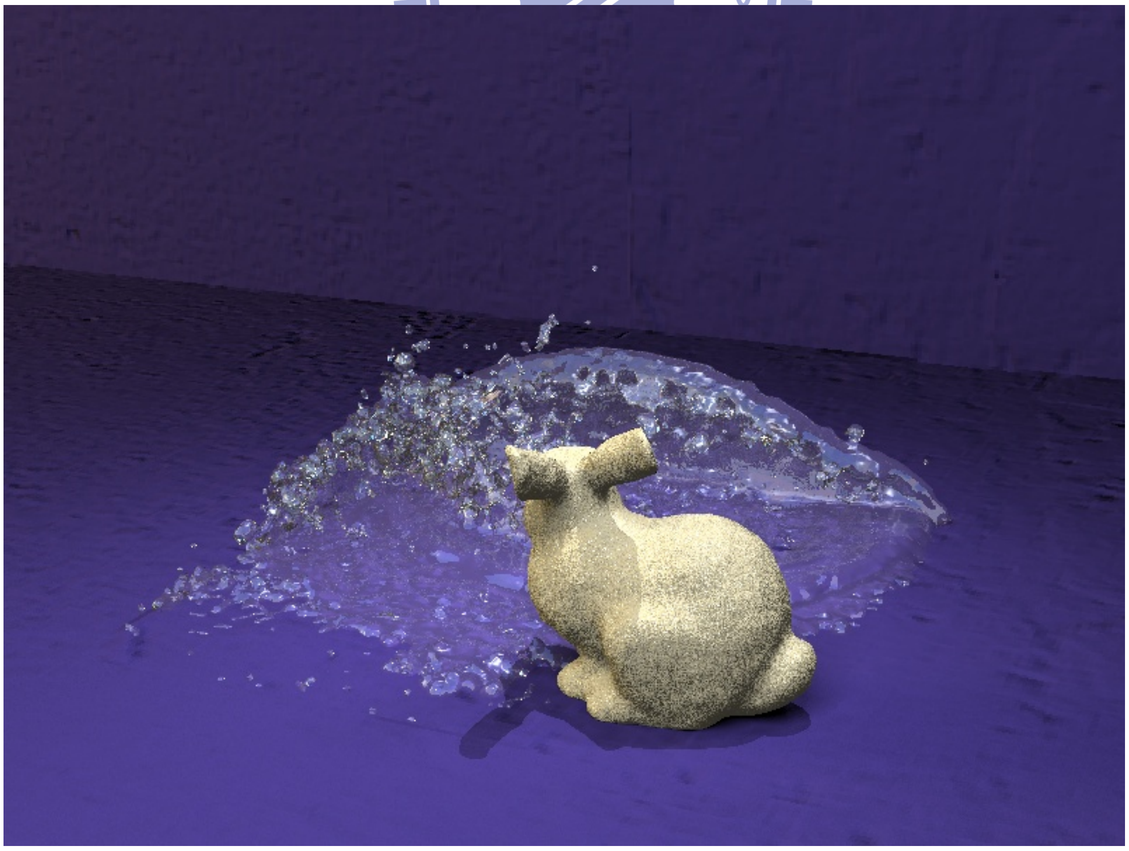


Figure 4.4(g): Step No. 600.

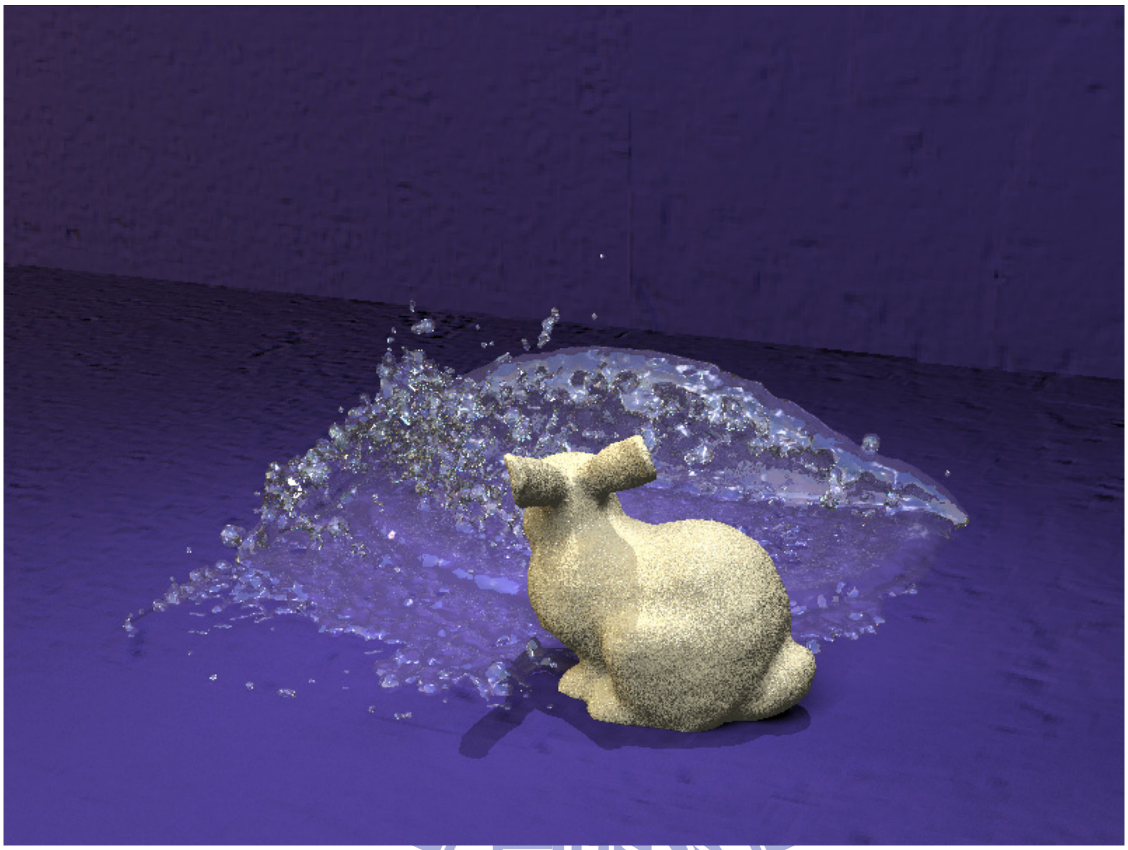


Figure 4.4(h): Step No. 630.

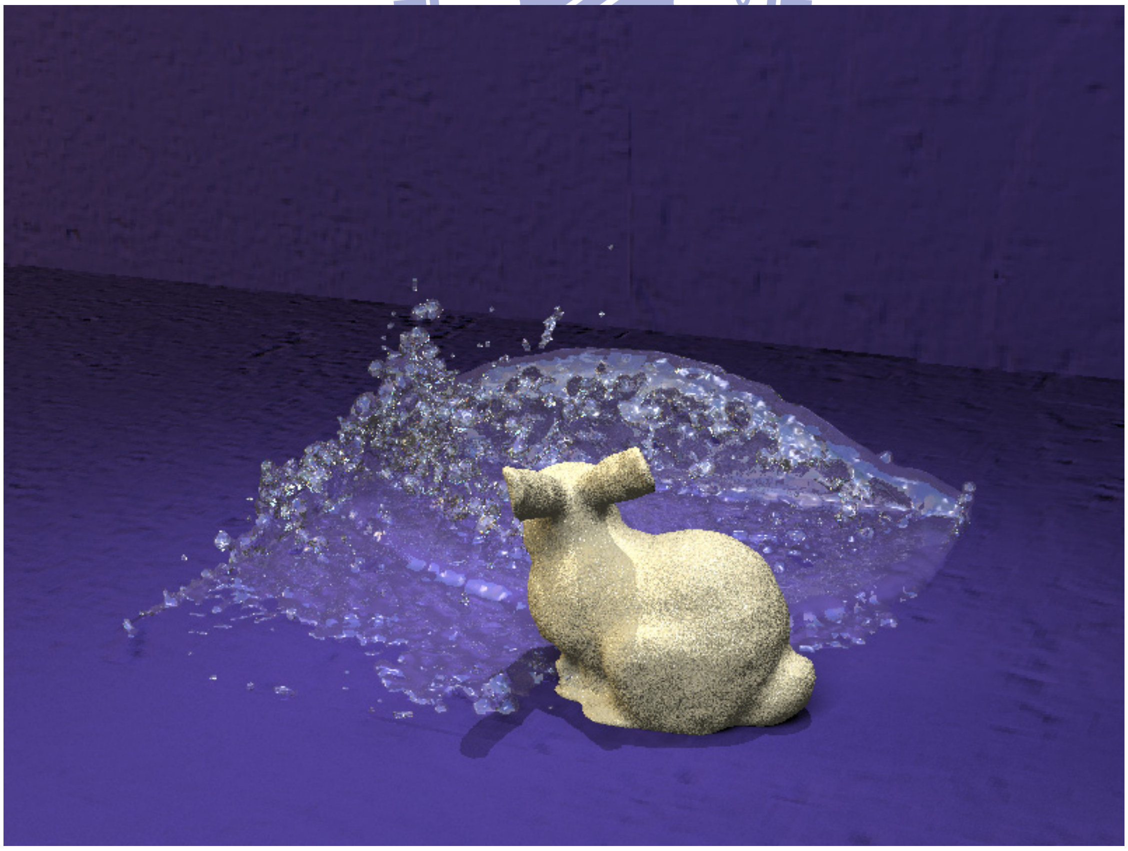


Figure 4.4(i): Step No. 660.

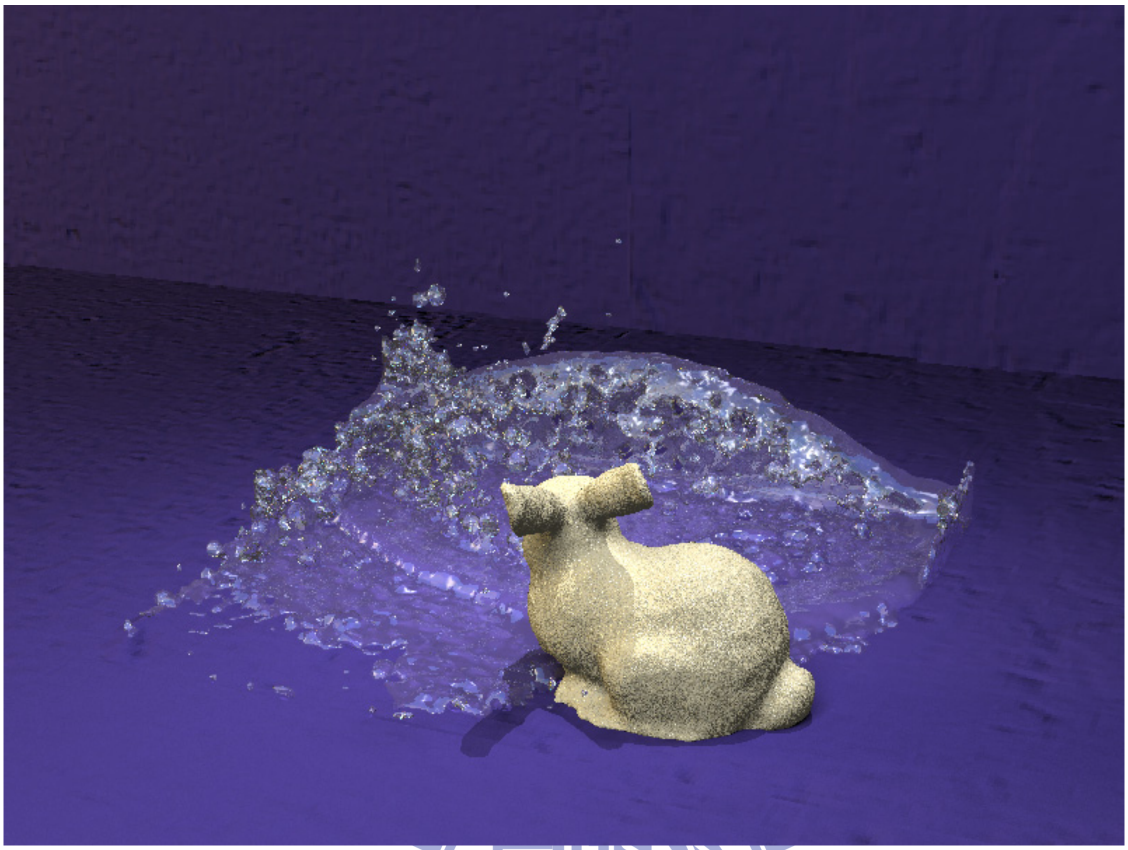


Figure 4.4(j): Step No. 690.

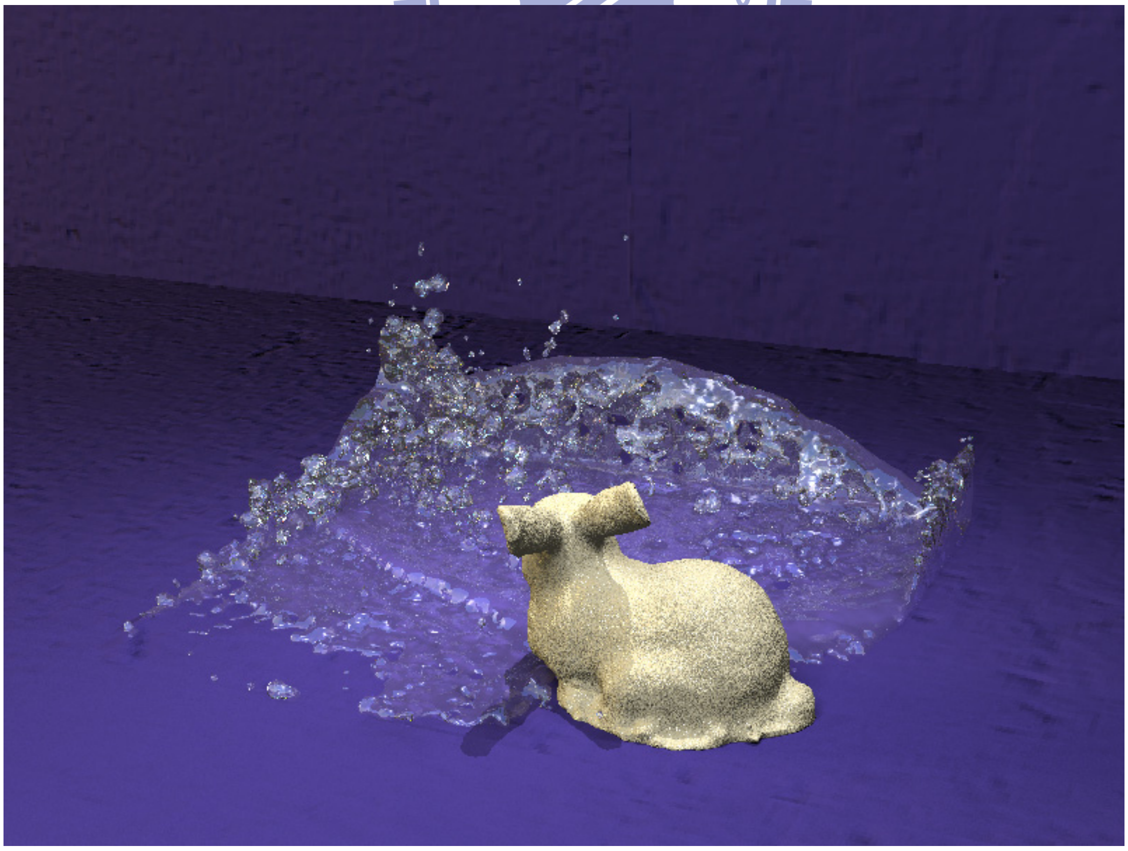


Figure 4.4(k): Step No. 720.

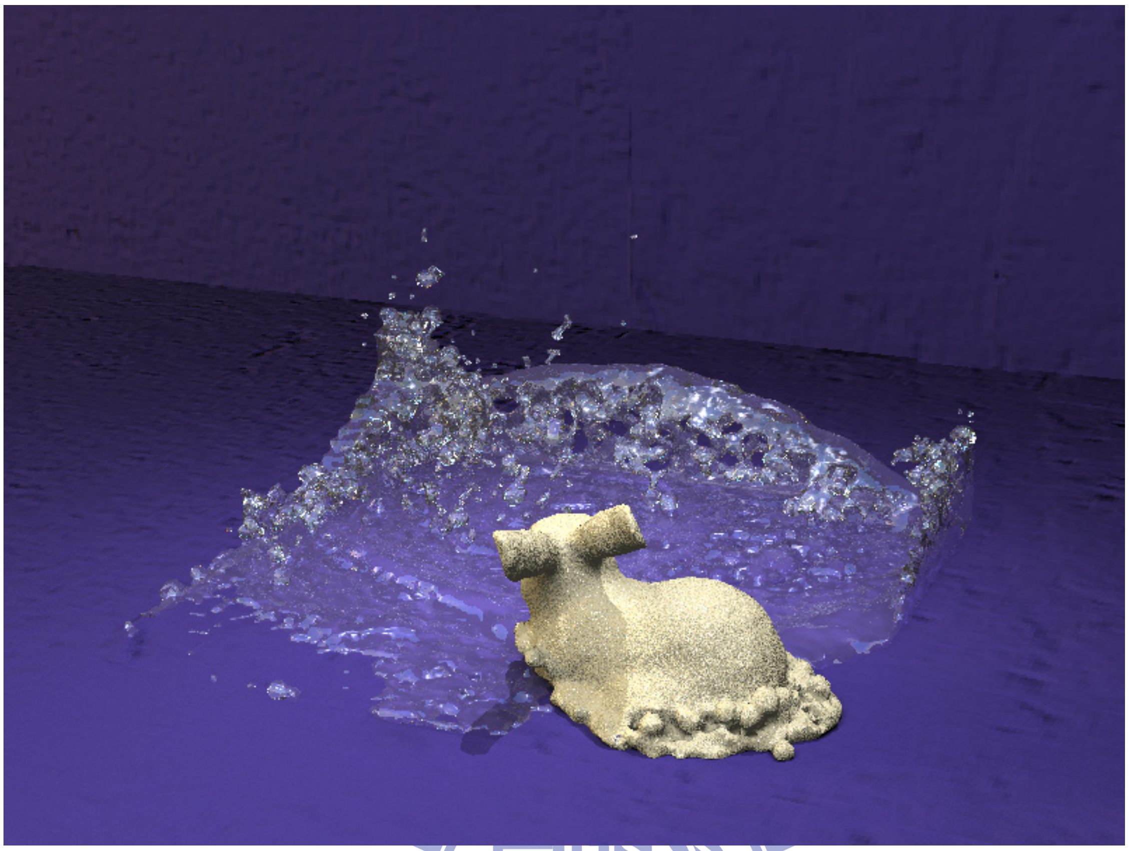


Figure 4.4(l): Step No. 750.

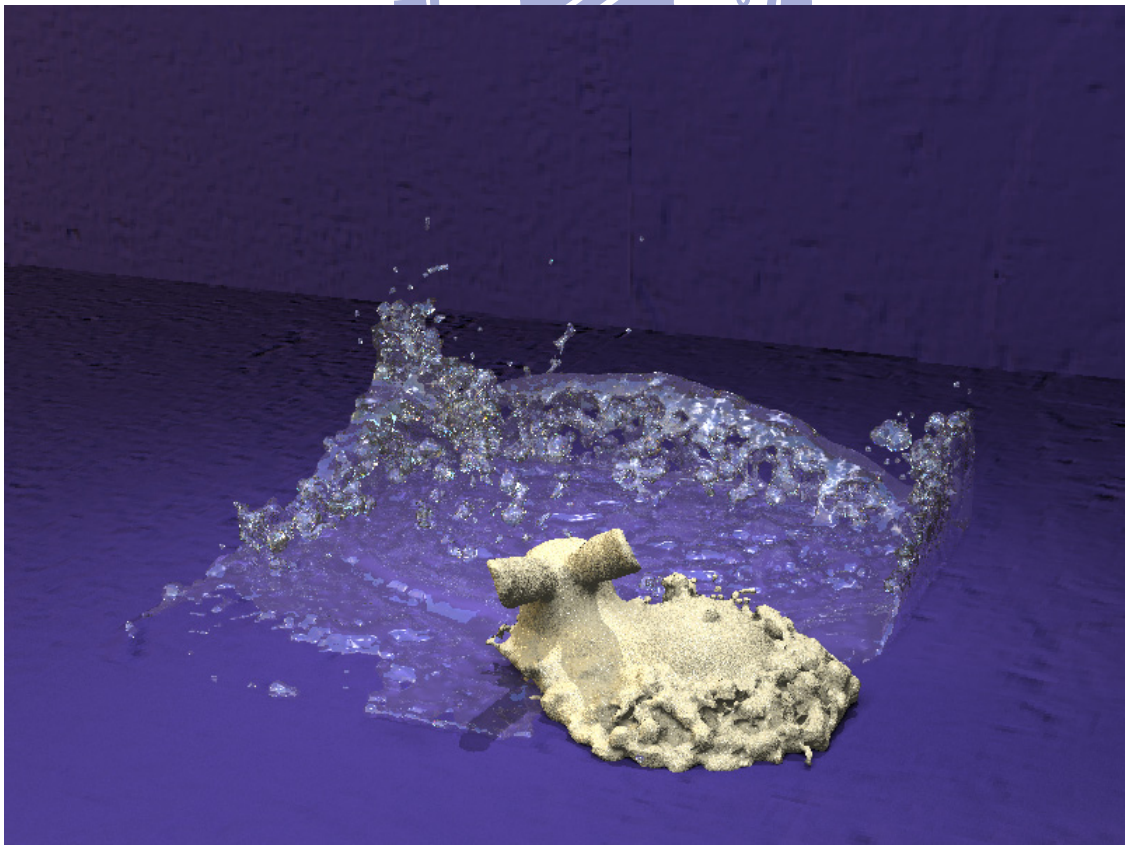


Figure 4.4(m): Step No. 780.

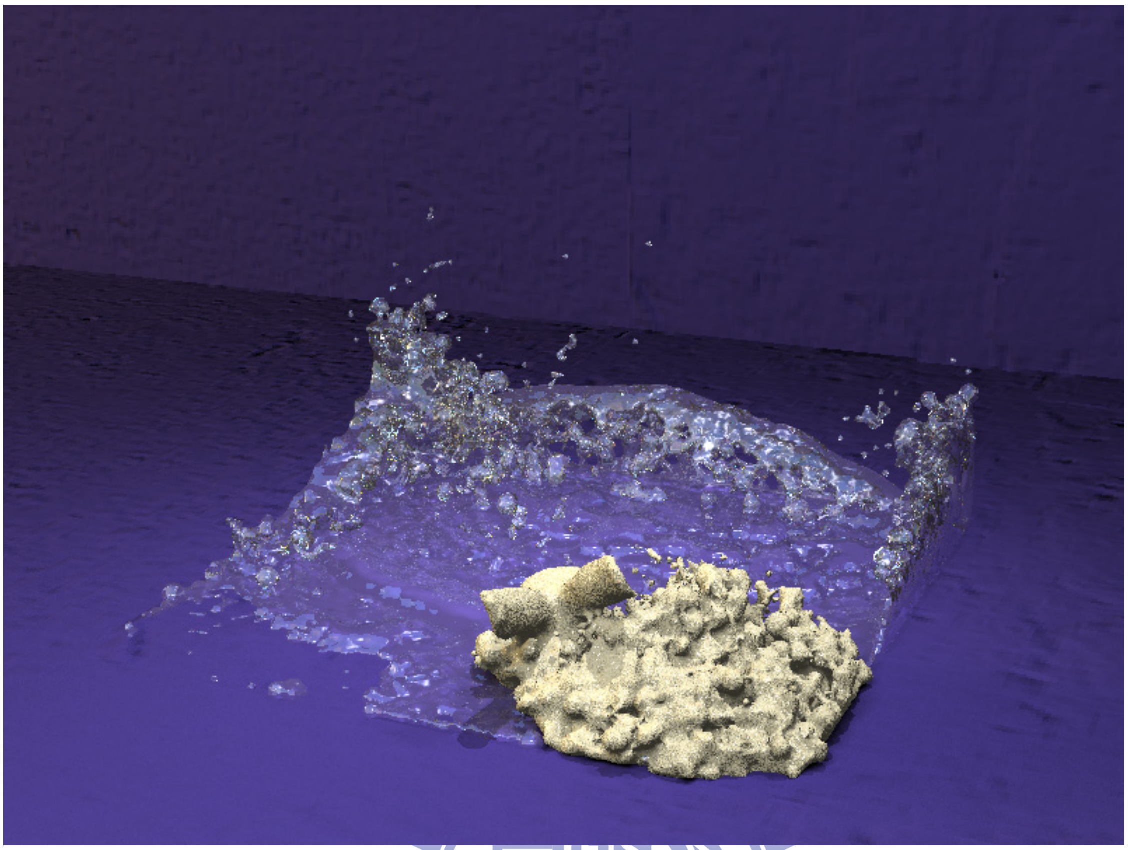


Figure 4.4(n): Step No. 810.

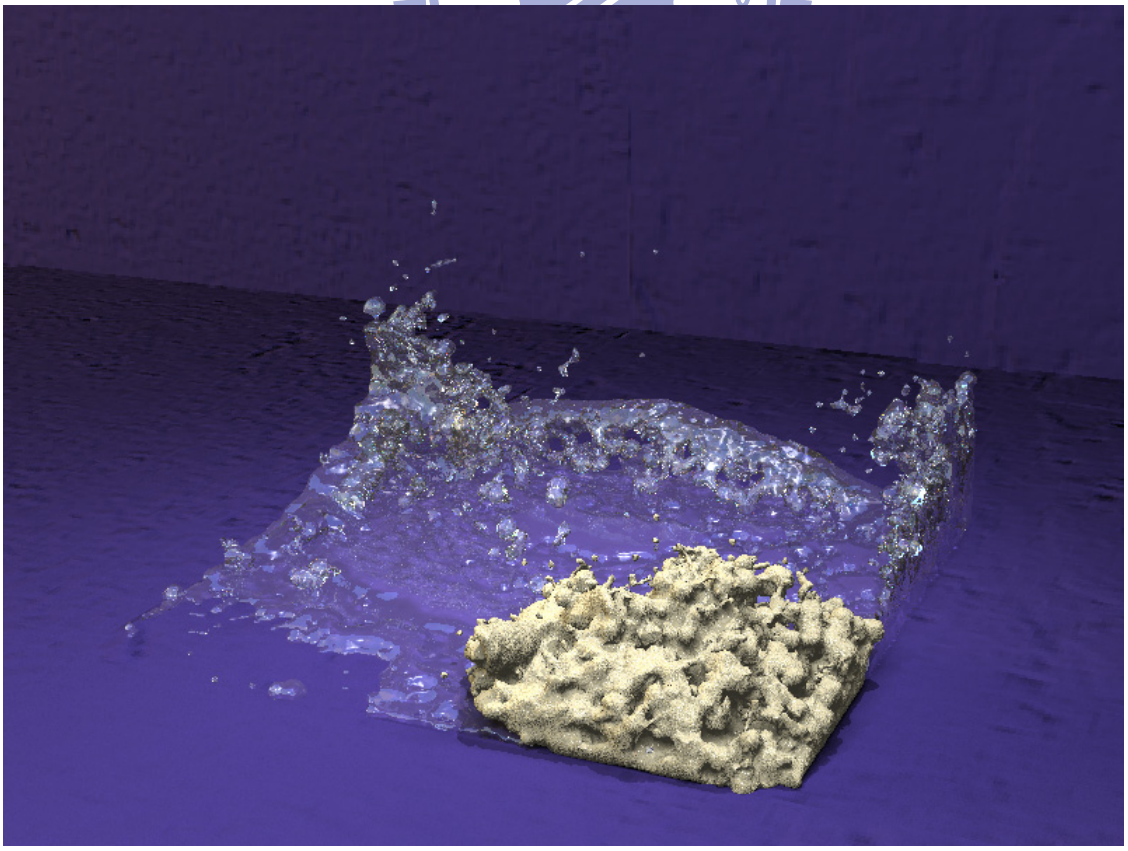


Figure 4.4(o): Step No. 840.

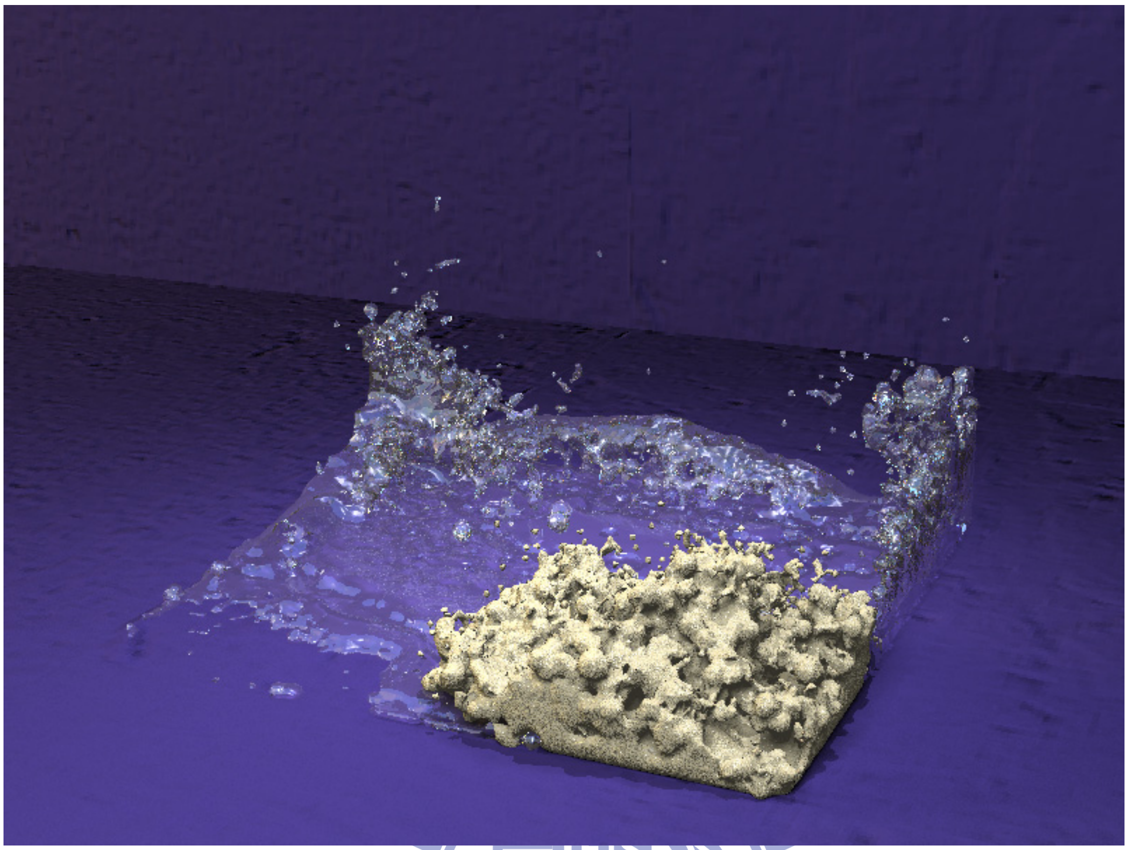


Figure 4.4(p): Step No. 870.

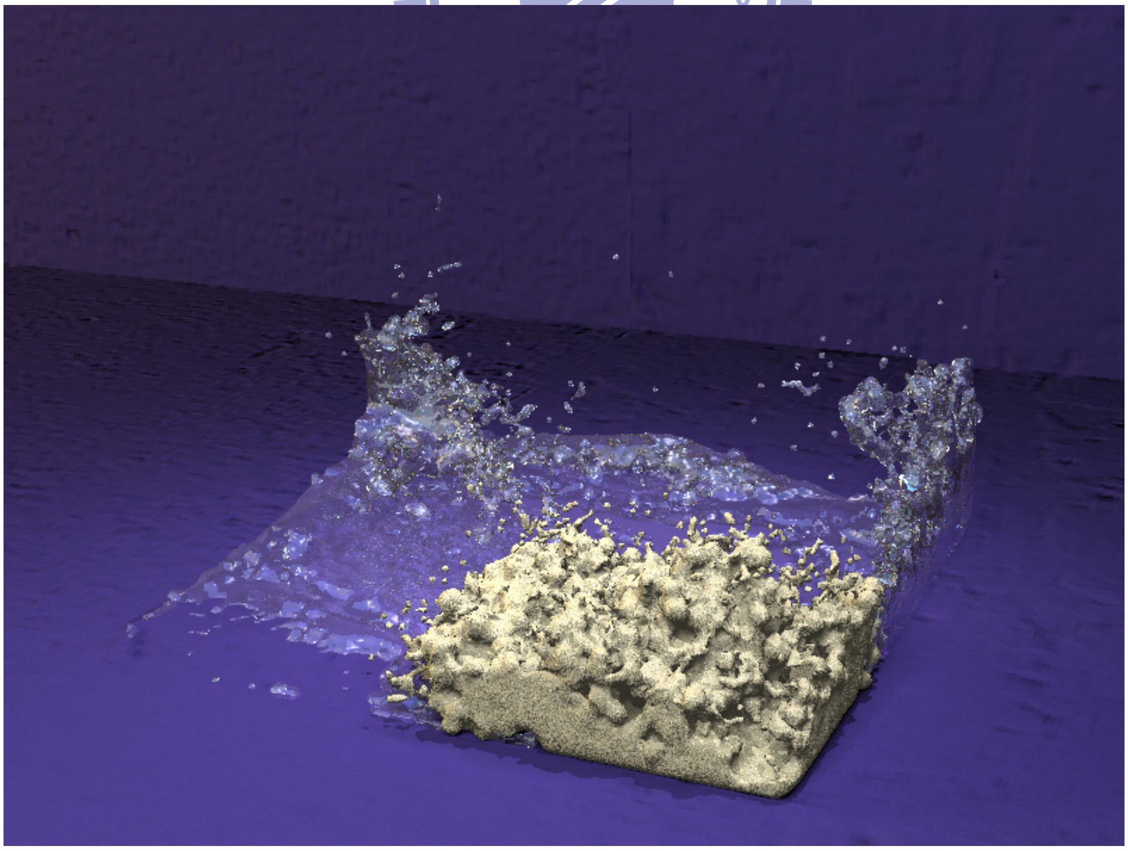


Figure 4.4(q): Step No. 900.

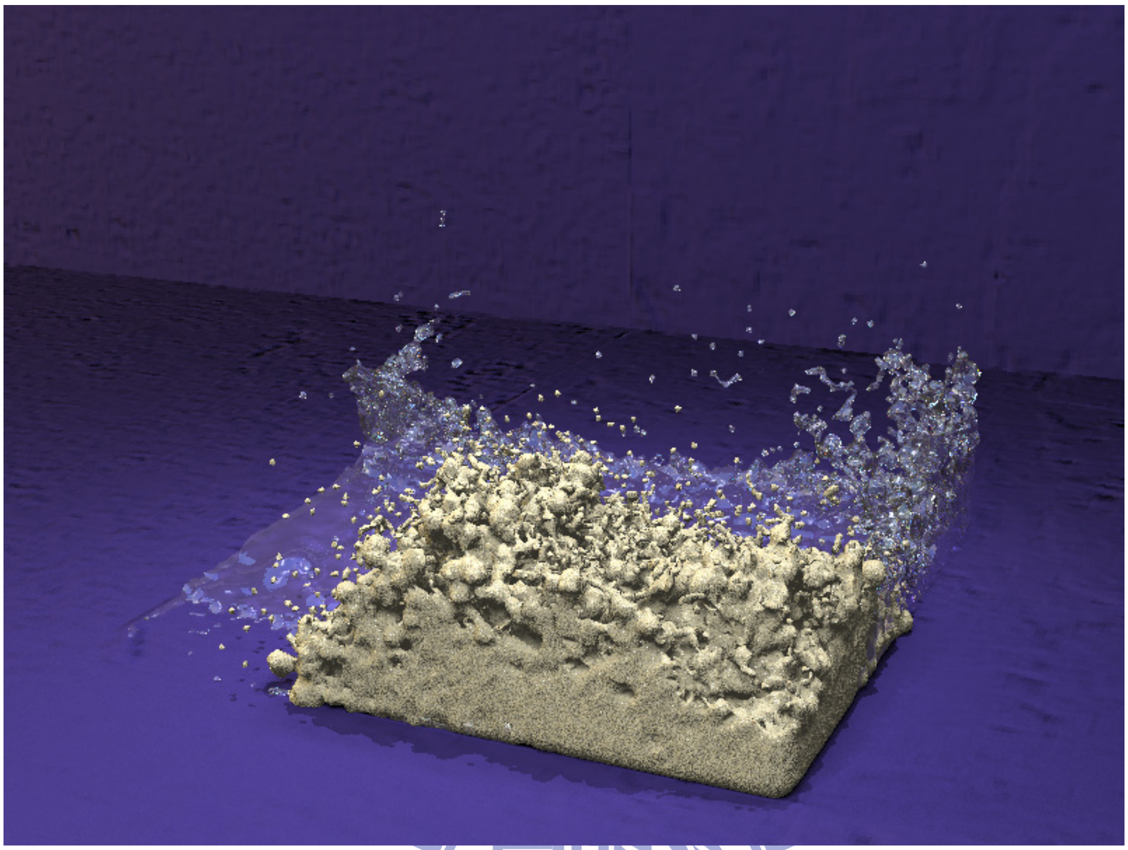


Figure 4.4(r): Step No. 1000.

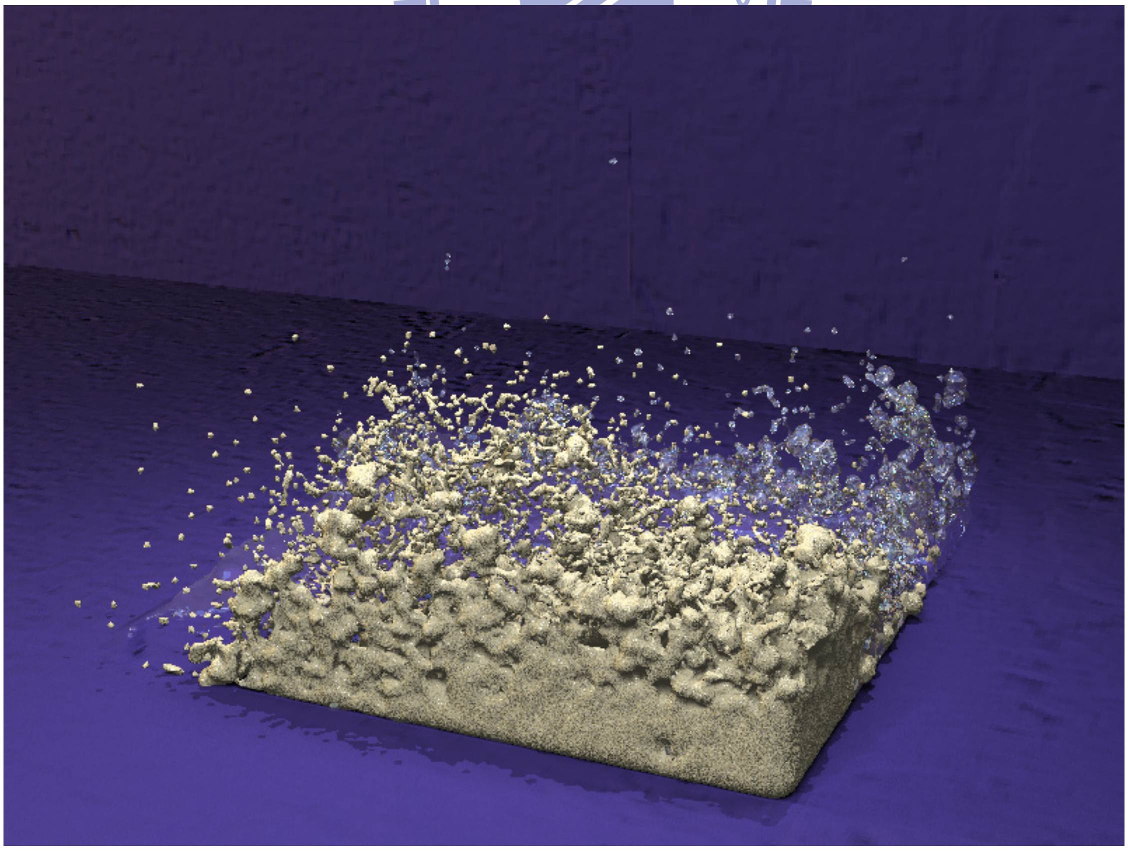


Figure 4.4(s): Step No. 1100.

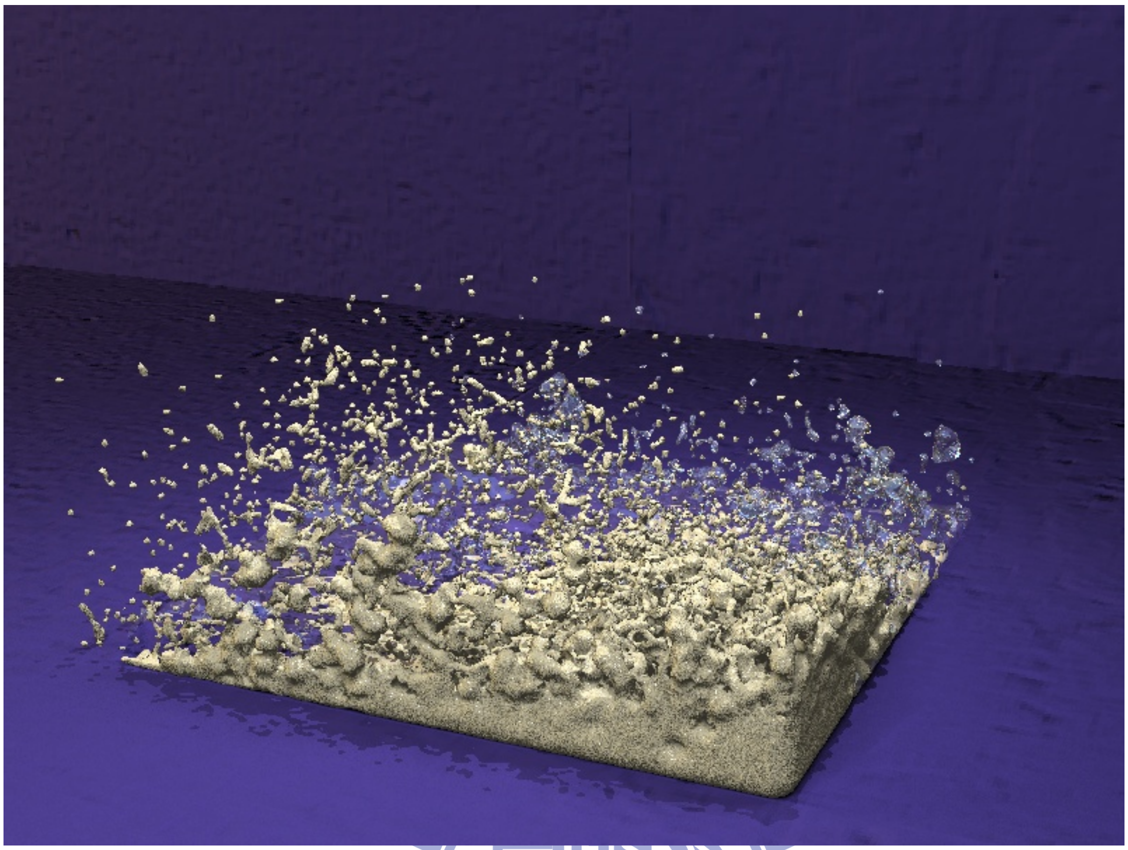


Figure 4.4(t): Step No. 1200.

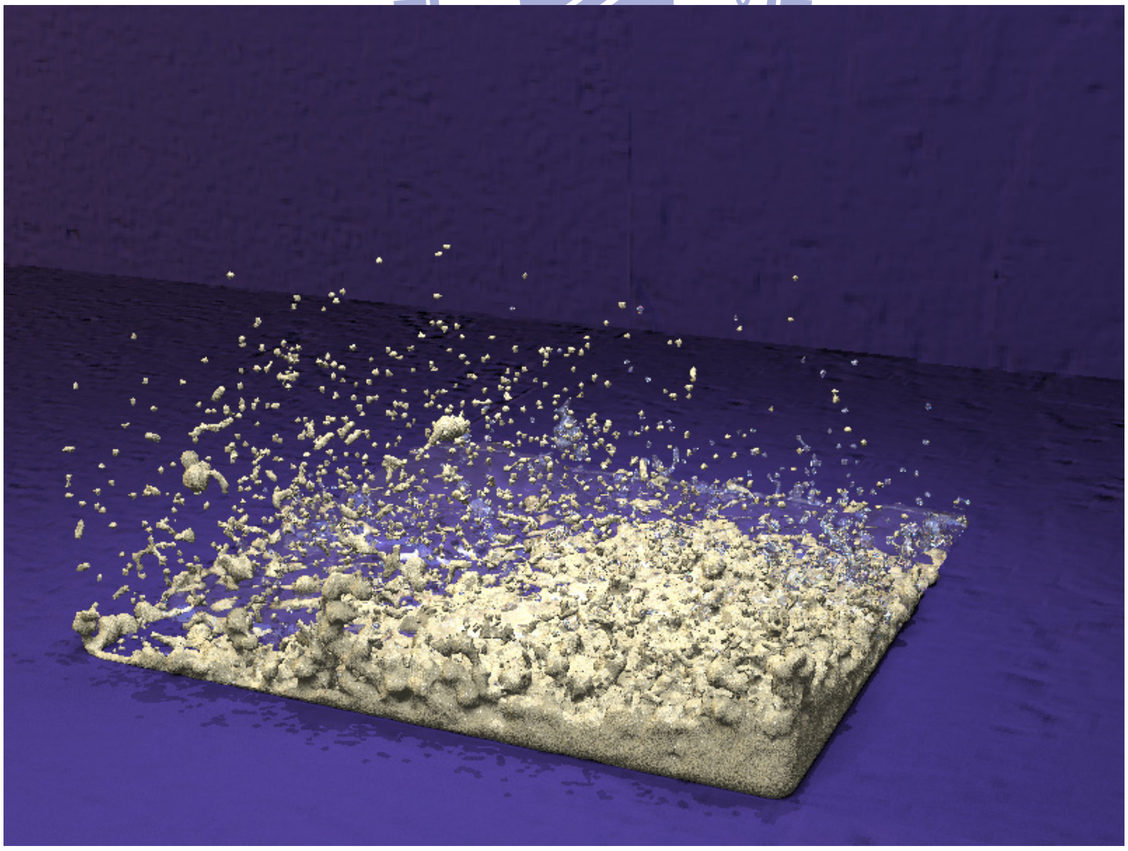


Figure 4.4(u): Step No. 1300.

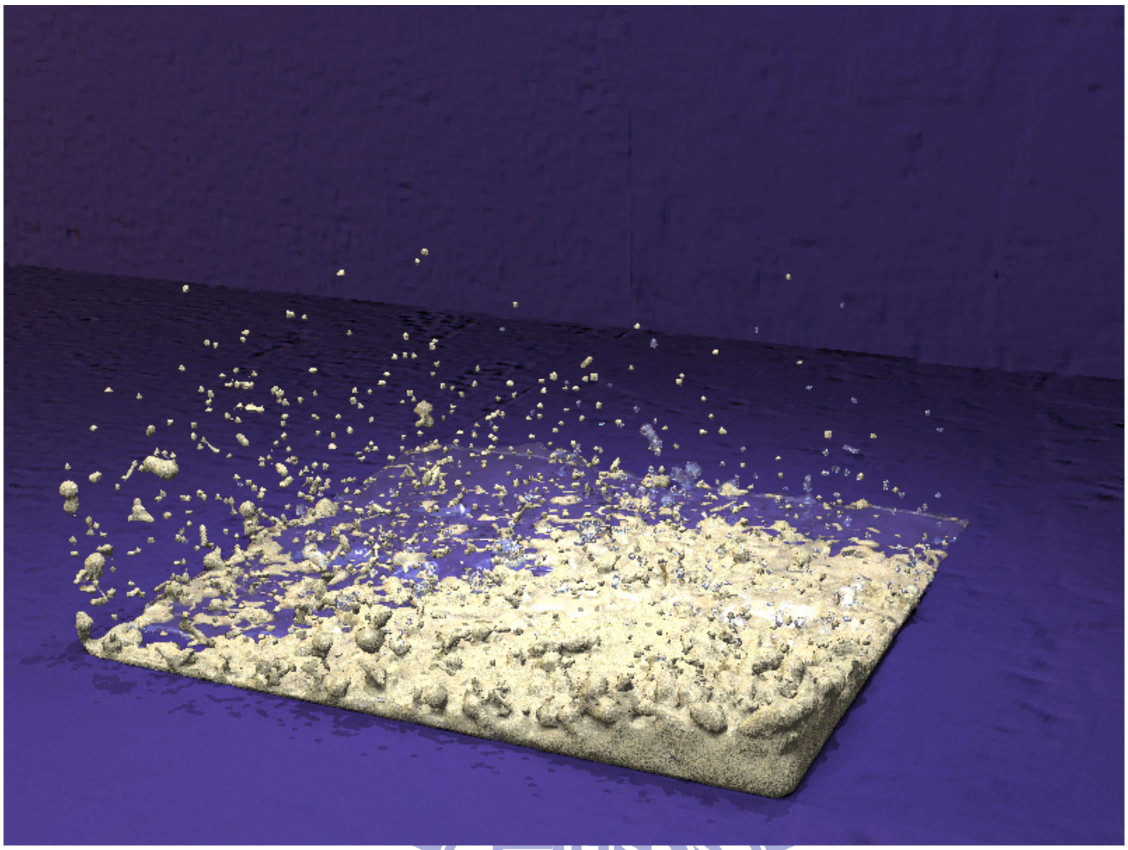


Figure 4.4(v): Step No. 1400.

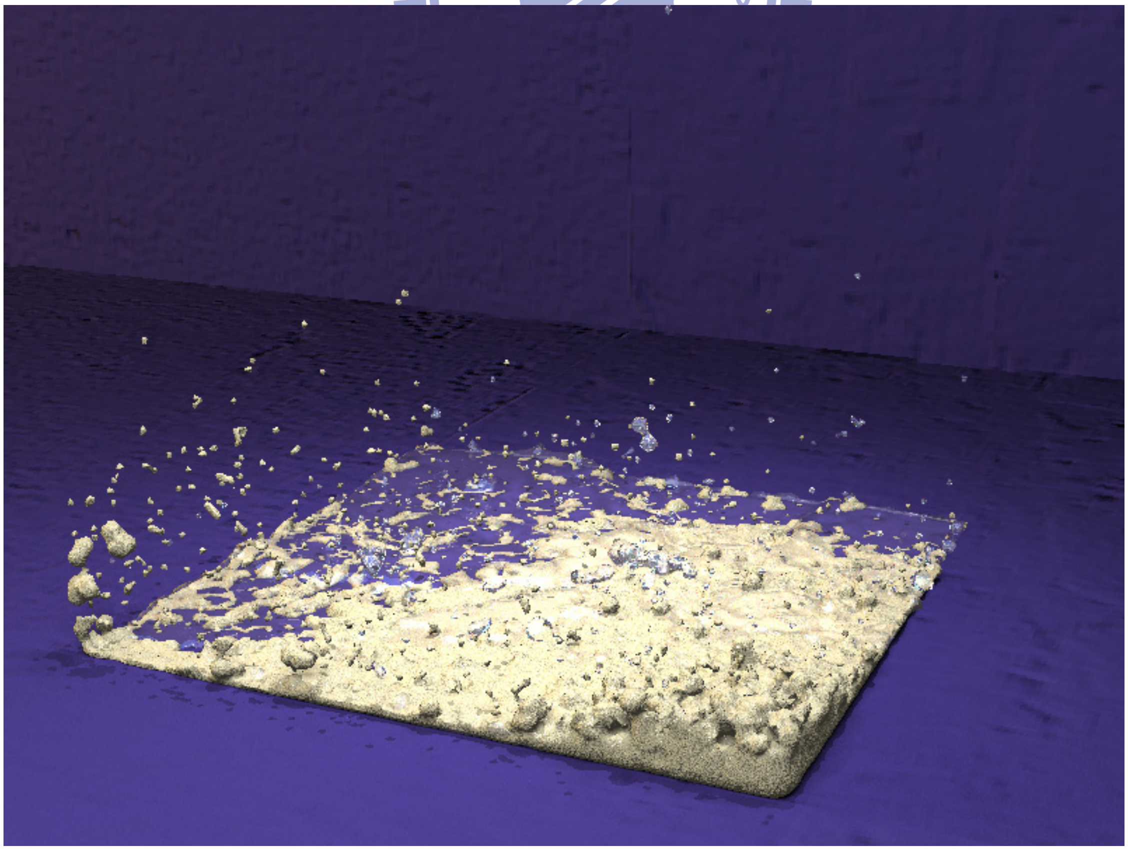


Figure 4.4(w): Step No. 1500.

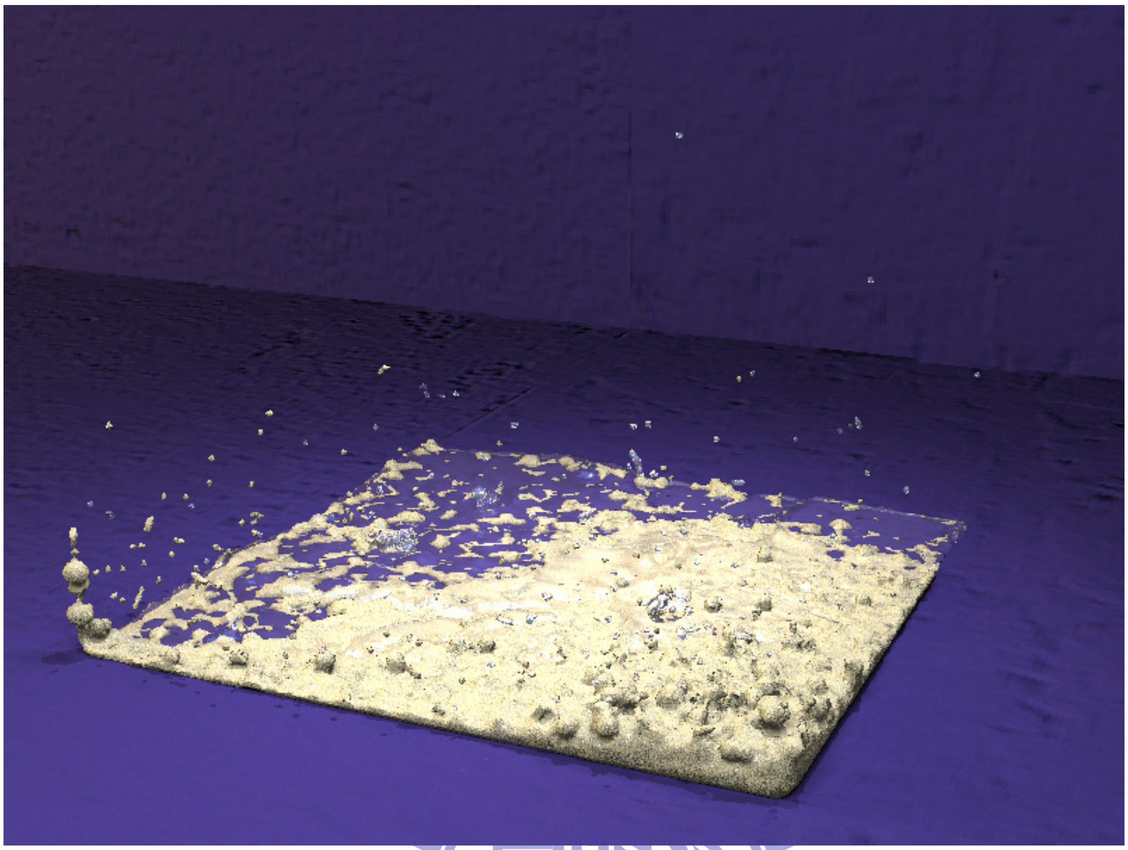


Figure 4.4(x): Step No. 1600.

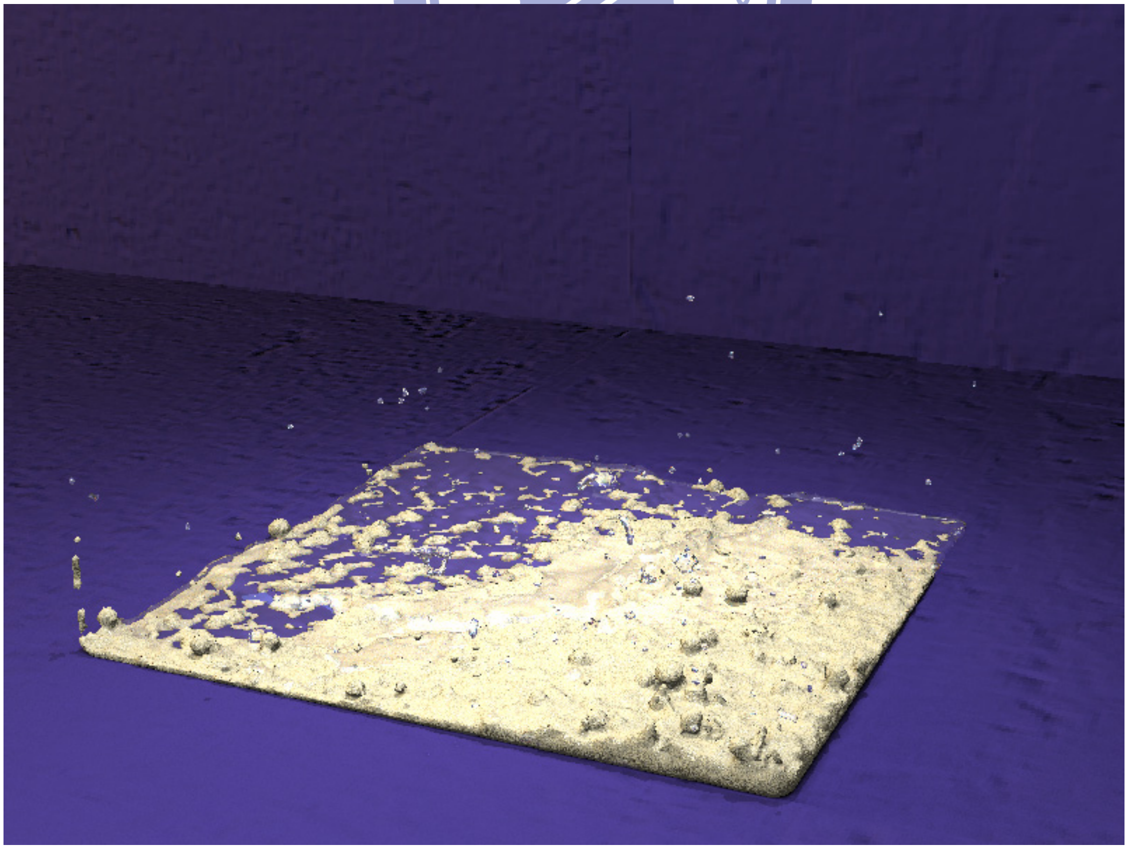


Figure 4.4(y): Step No. 1700.

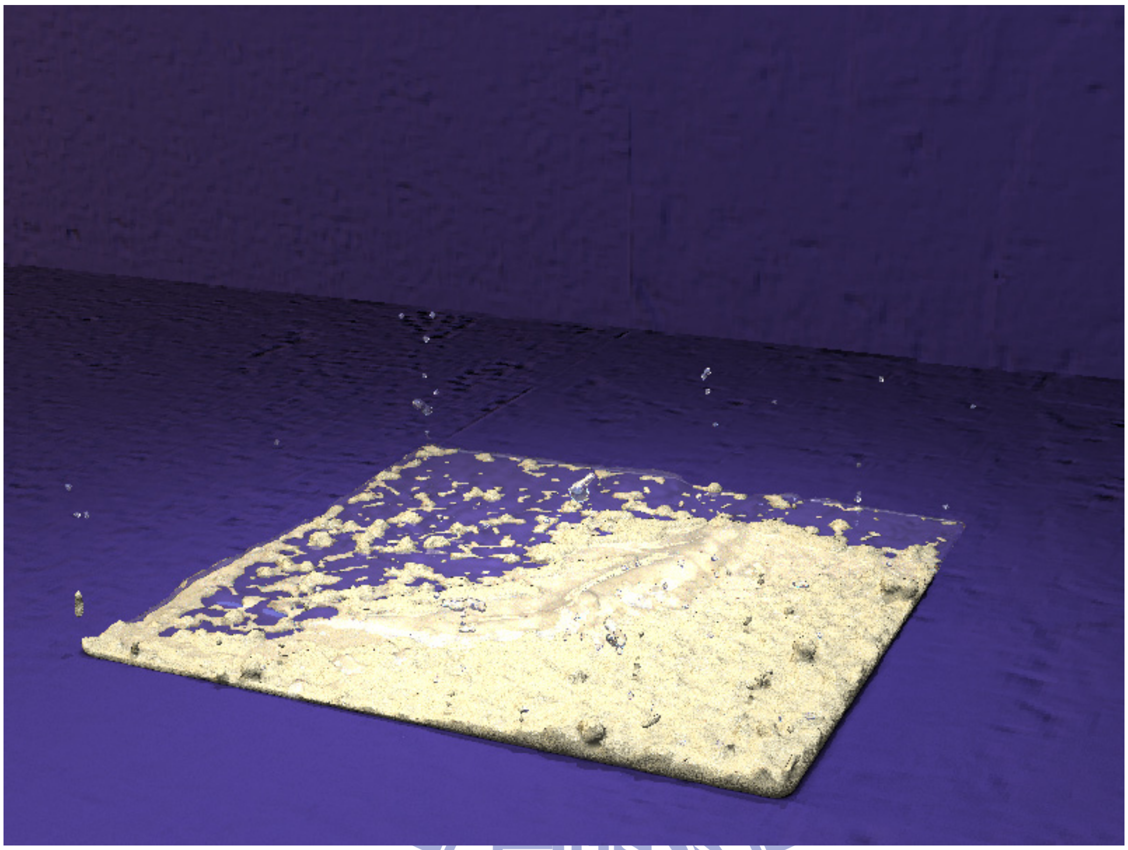


Figure 4.4(z): Step No. 1800.

Figures 4.5(a) to (z) shows an animation of sand Winnie and a box of water. The simulation consists of 31,107 sand and 40,858 water particles. The average computation time is 15.74s per step.

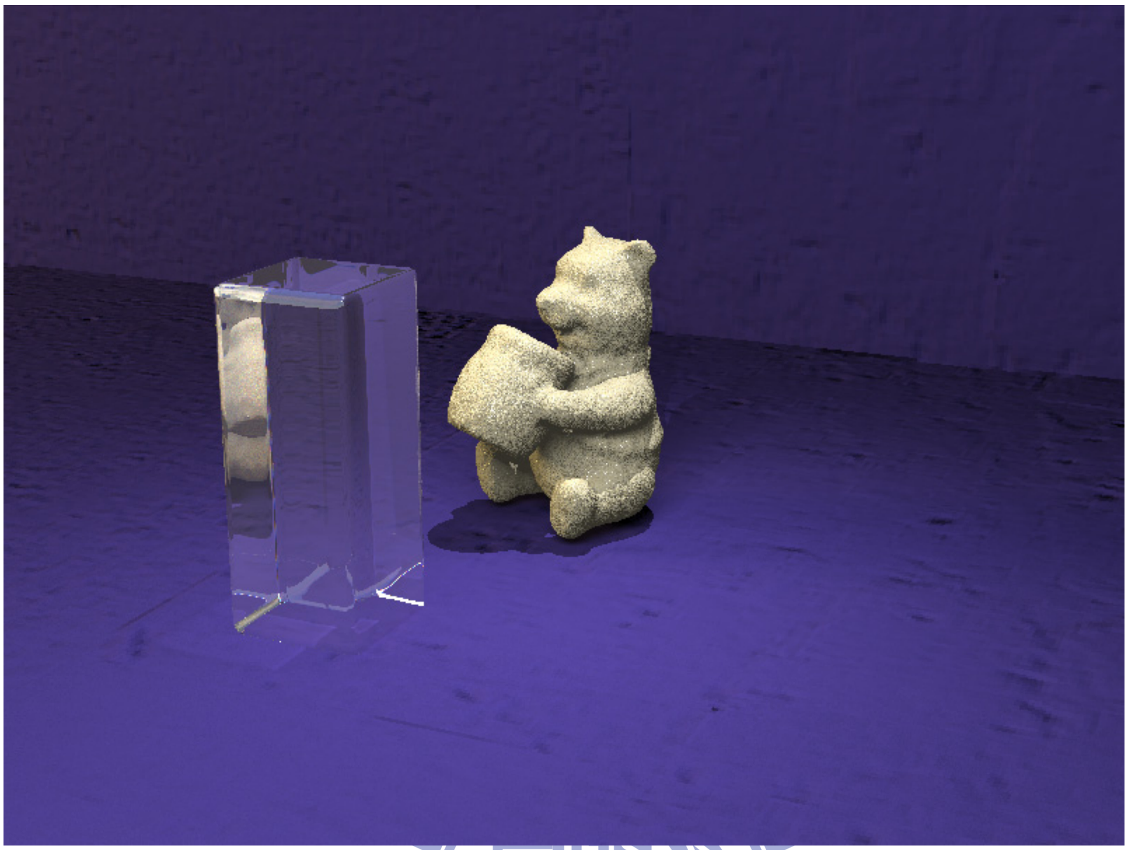


Figure 4.5(a): Step No. 000.

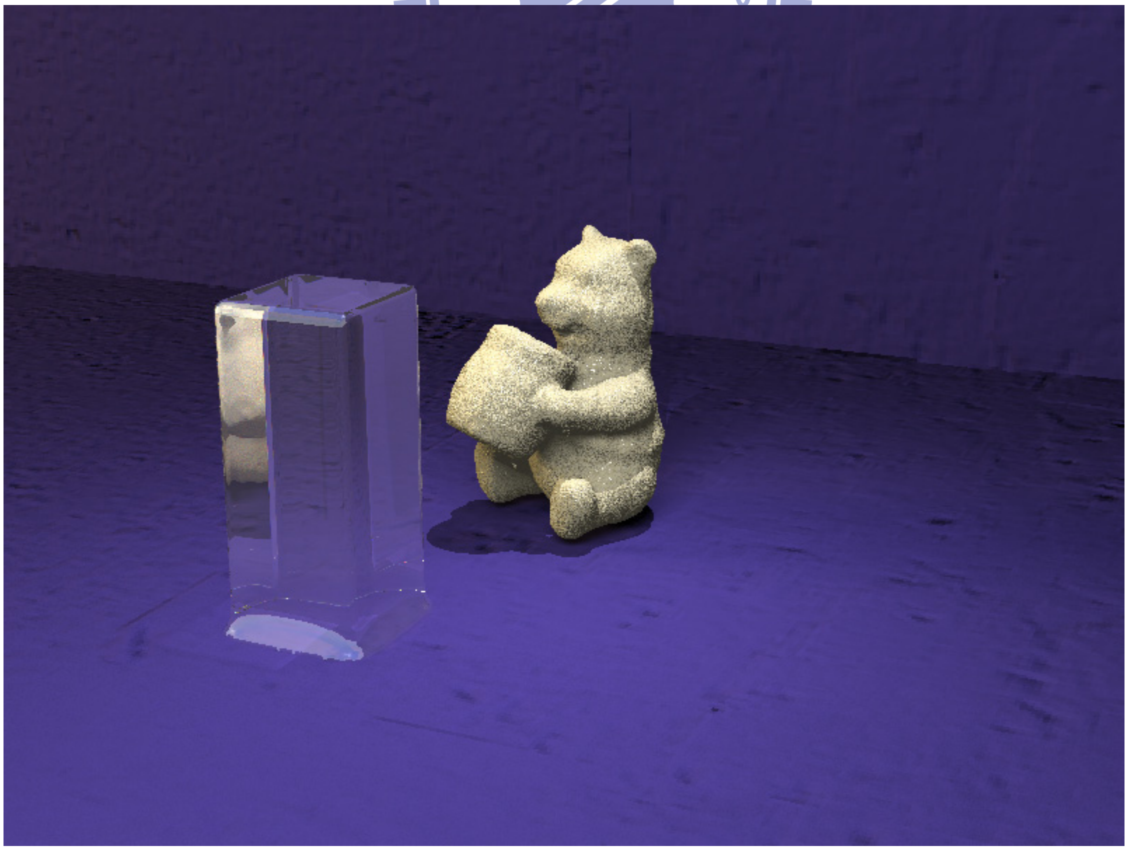


Figure 4.5(b): Step No. 100.

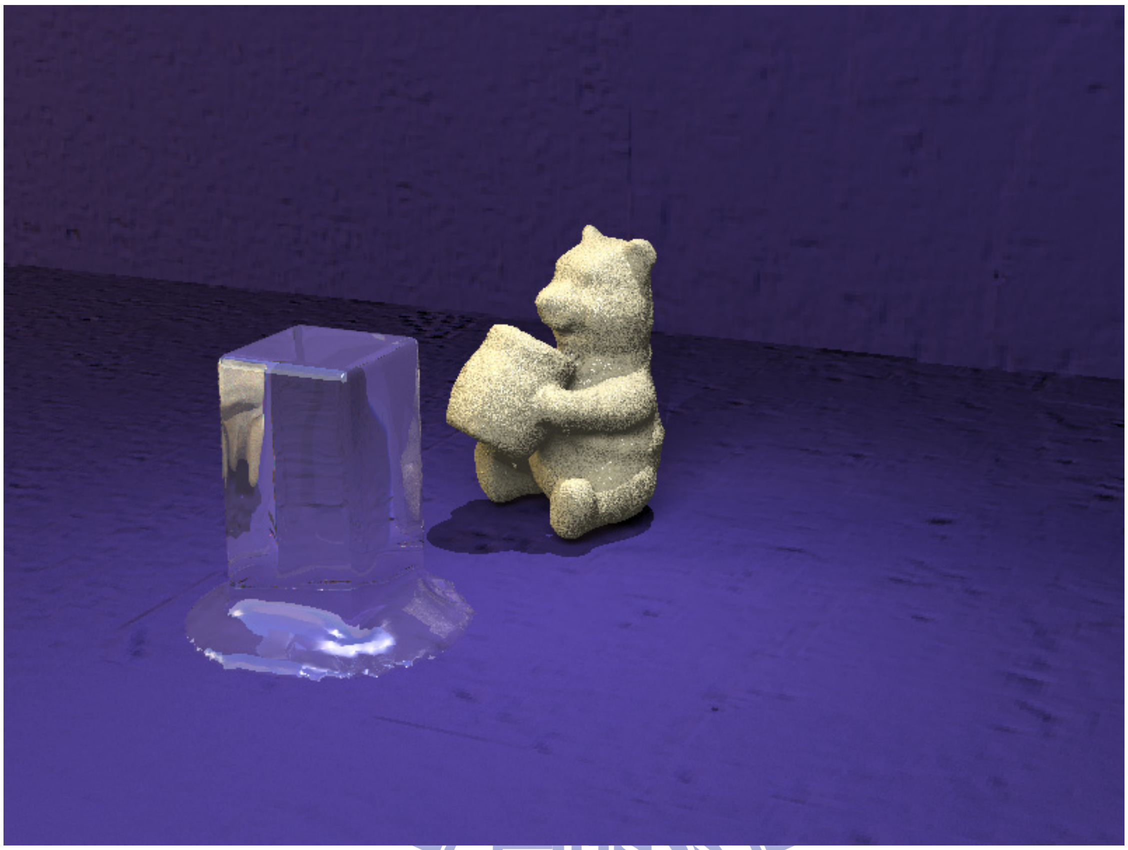


Figure 4.5(c): Step No. 200.

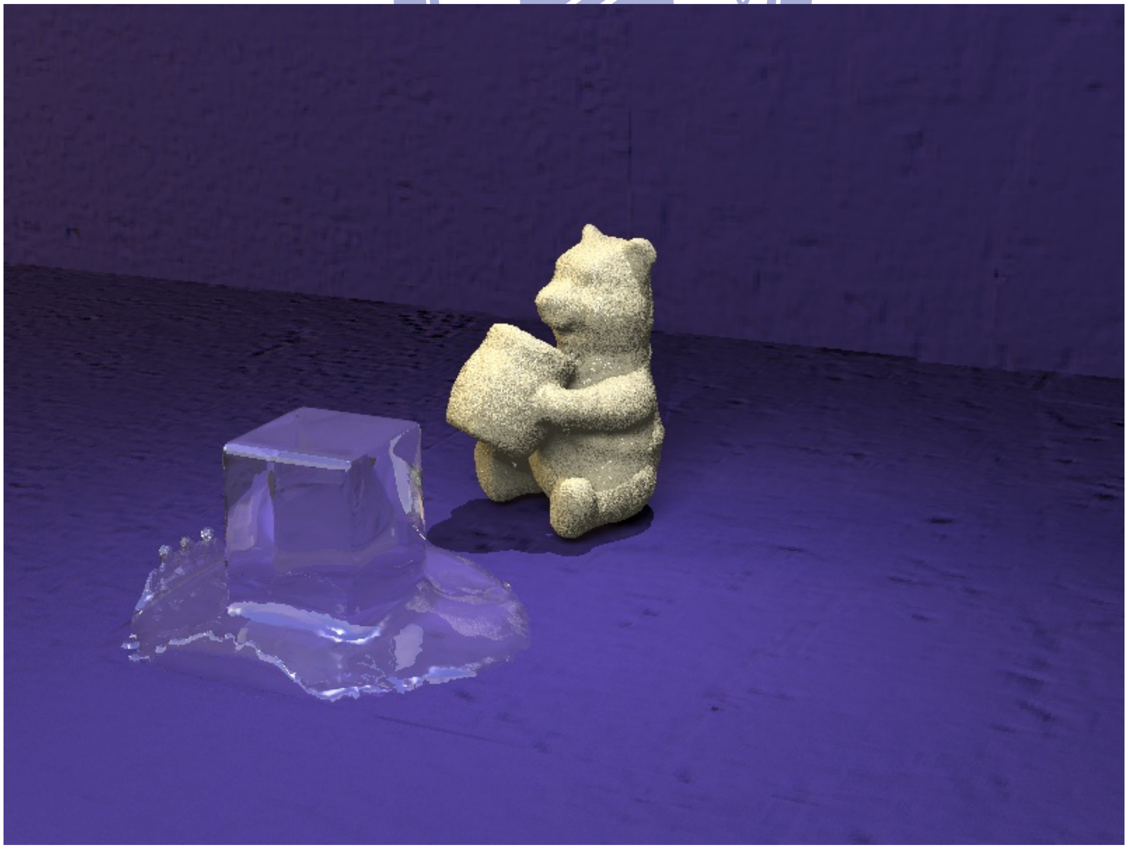


Figure 4.5(d): Step No. 300.

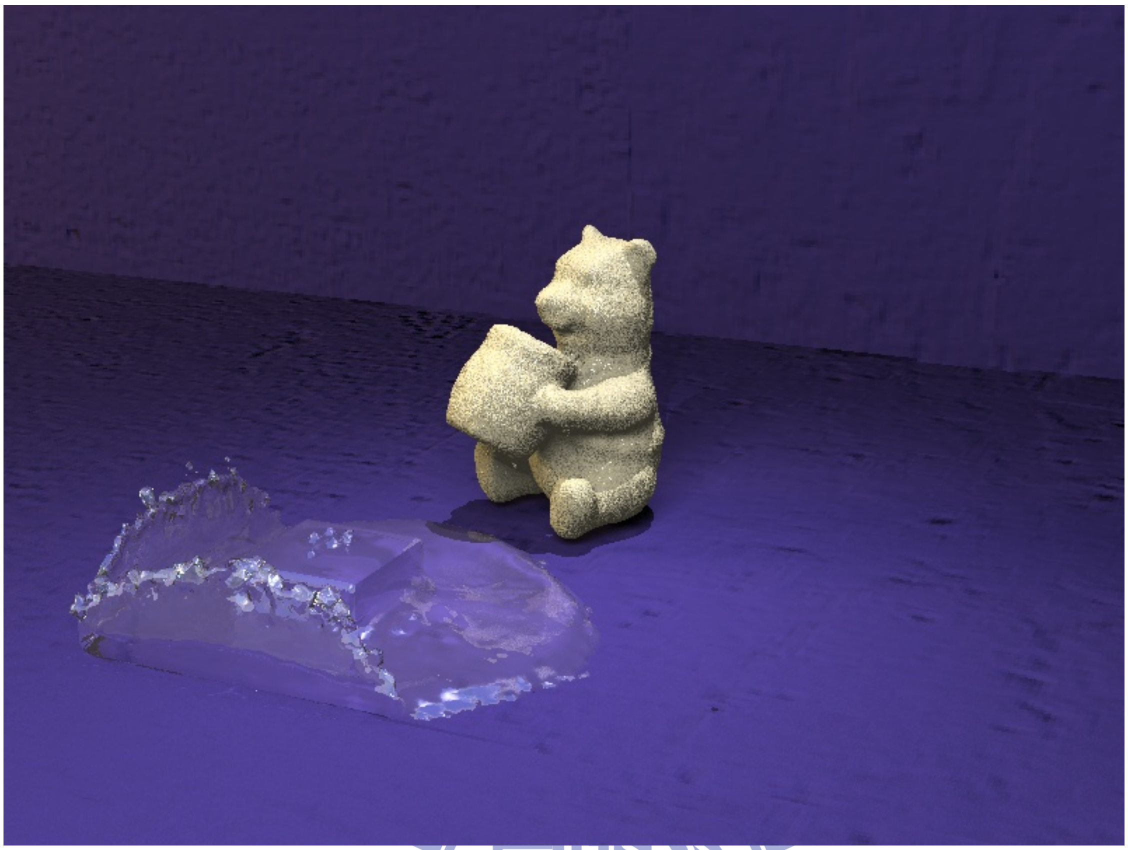


Figure 4.5(e): Step No. 400.

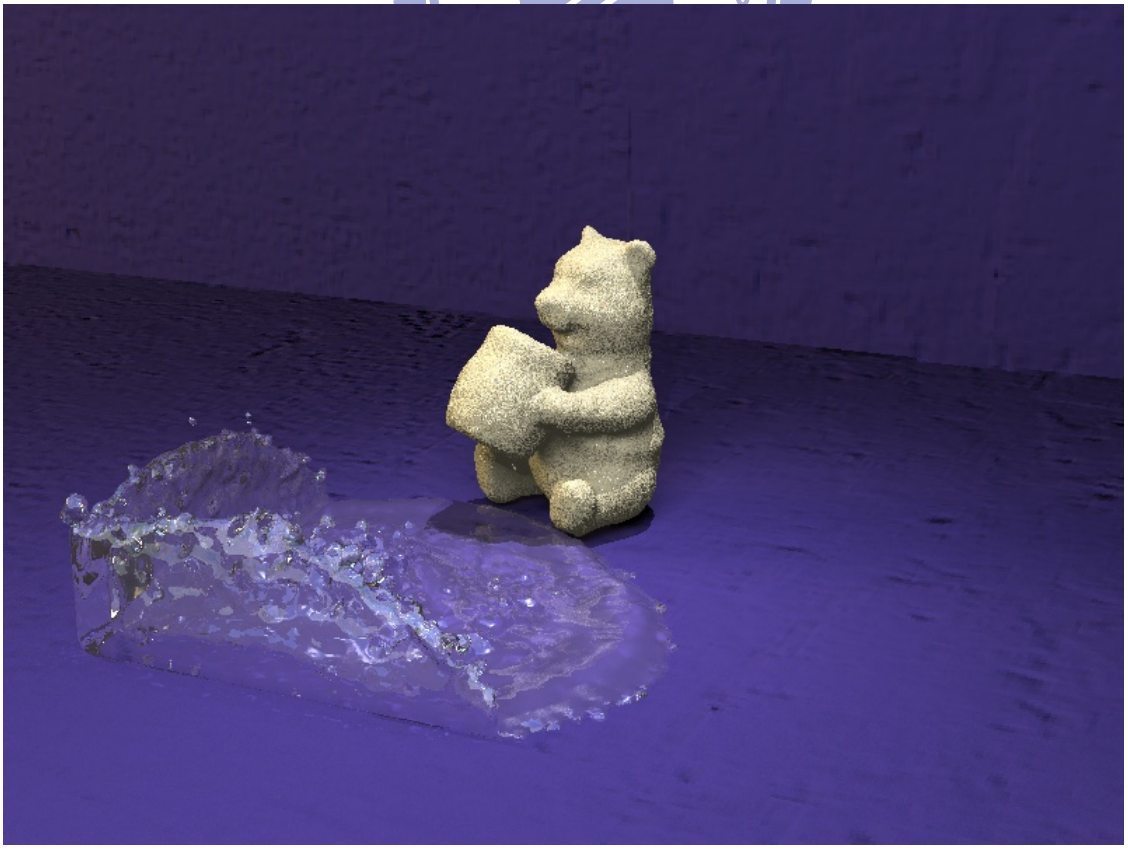


Figure 4.5(f): Step No. 500.

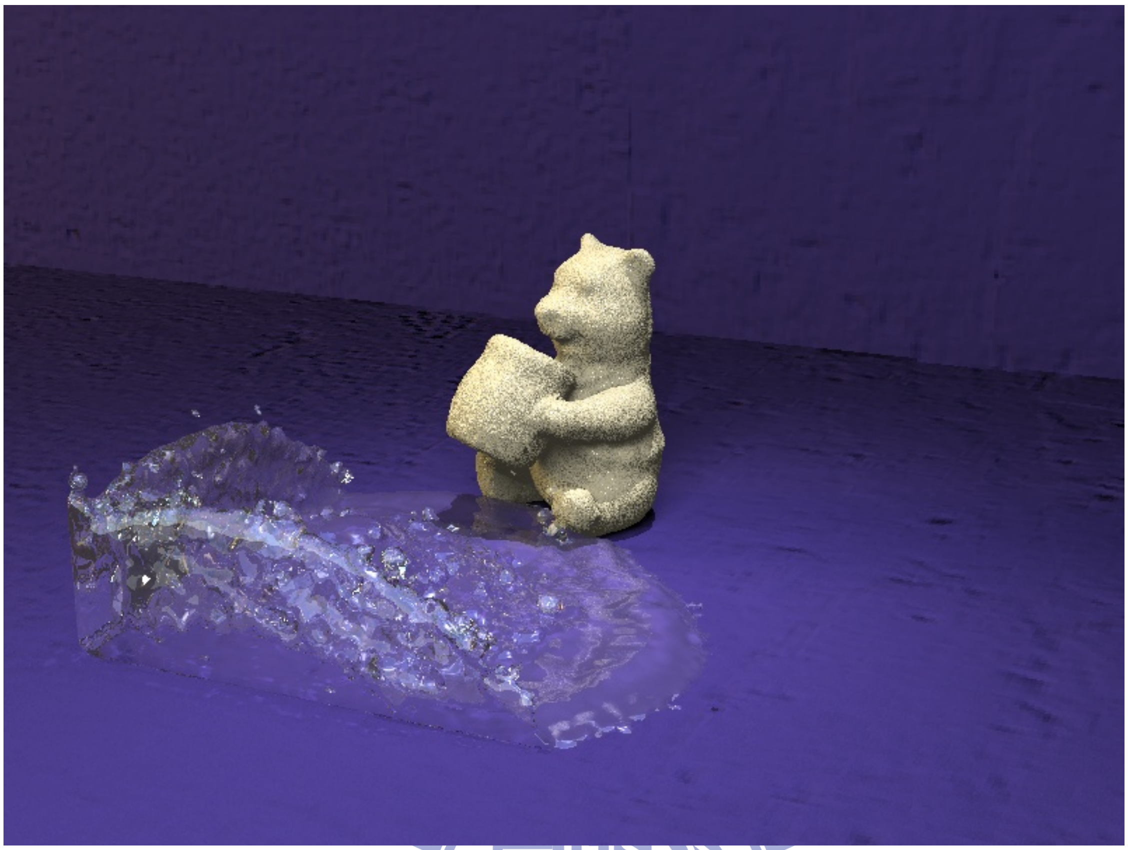


Figure 4.5(g): Step No. 540.

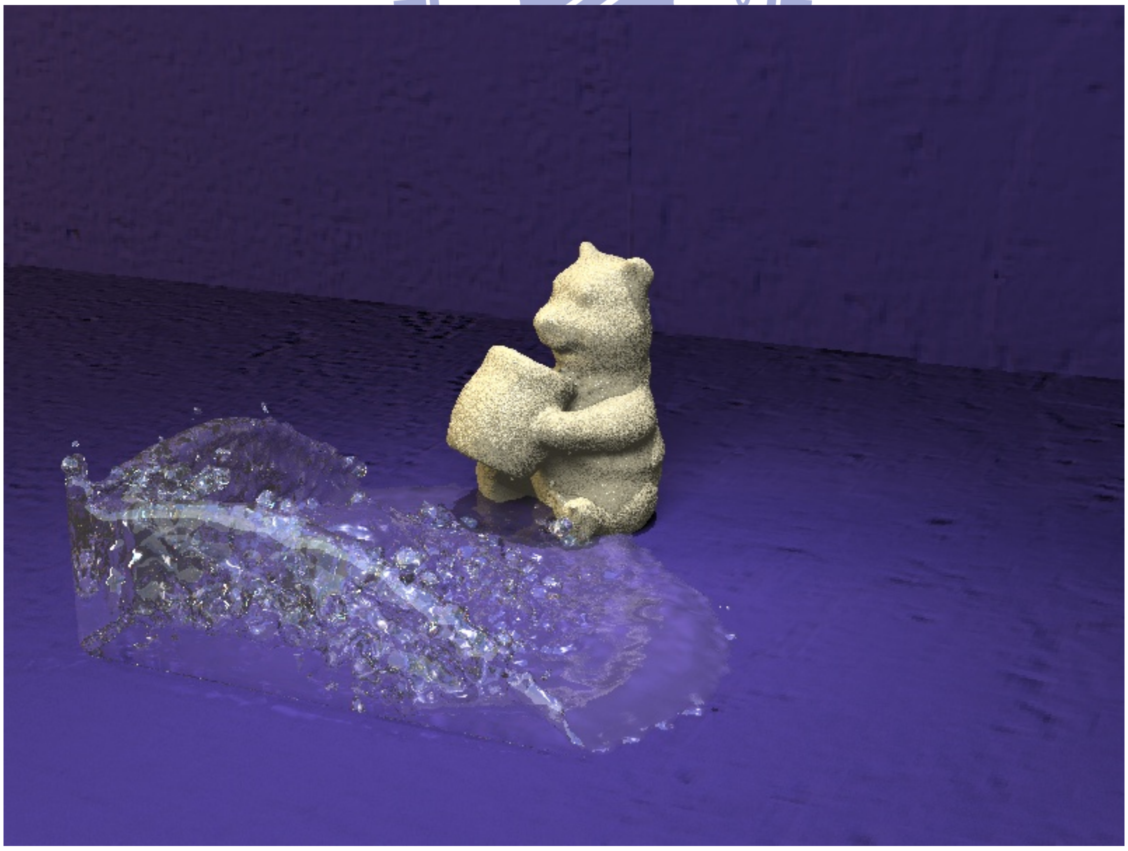


Figure 4.5(h): Step No. 570.

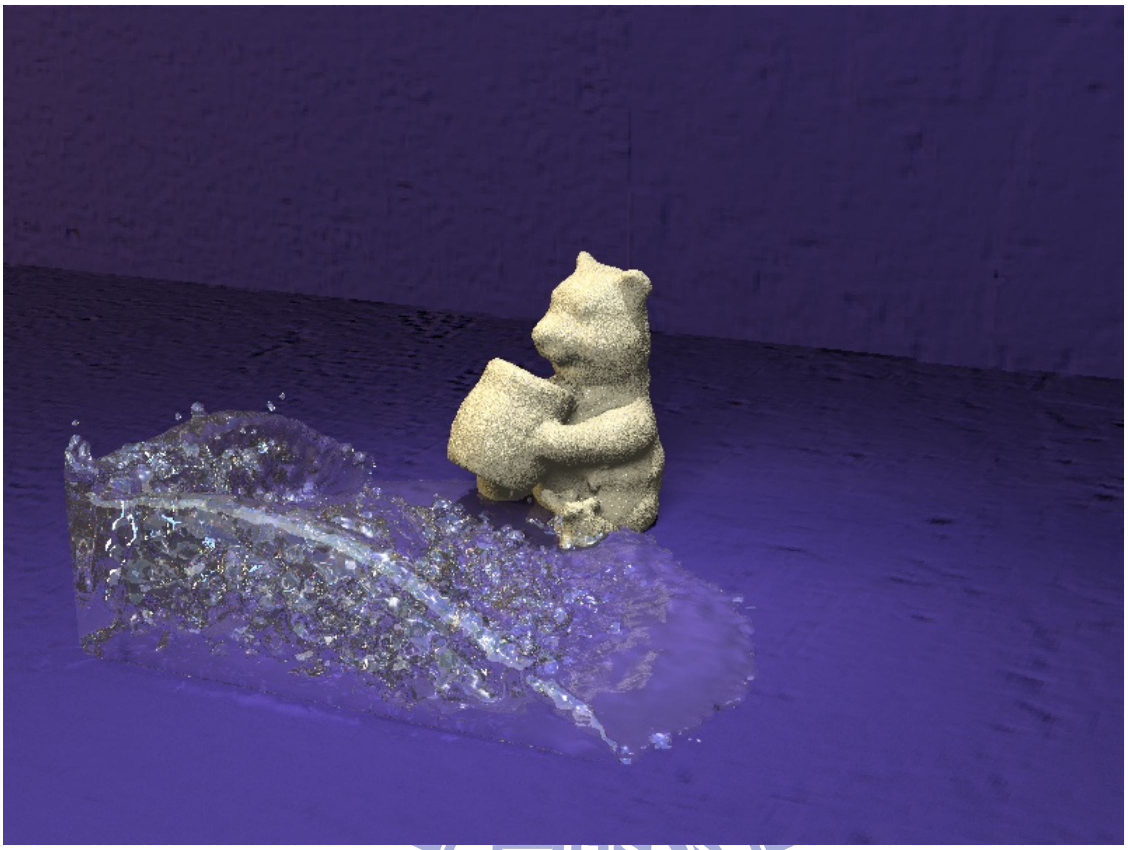


Figure 4.5(i): Step No. 600.

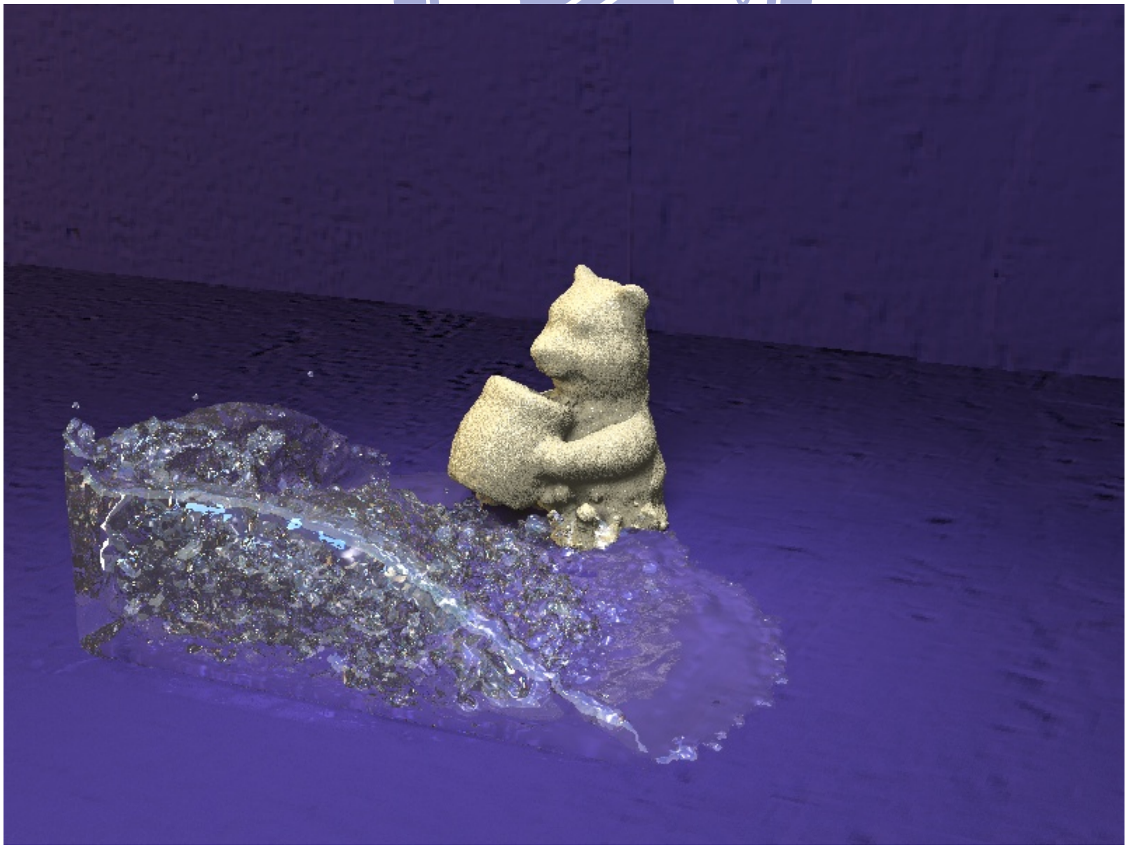


Figure 4.5(j): Step No. 630.

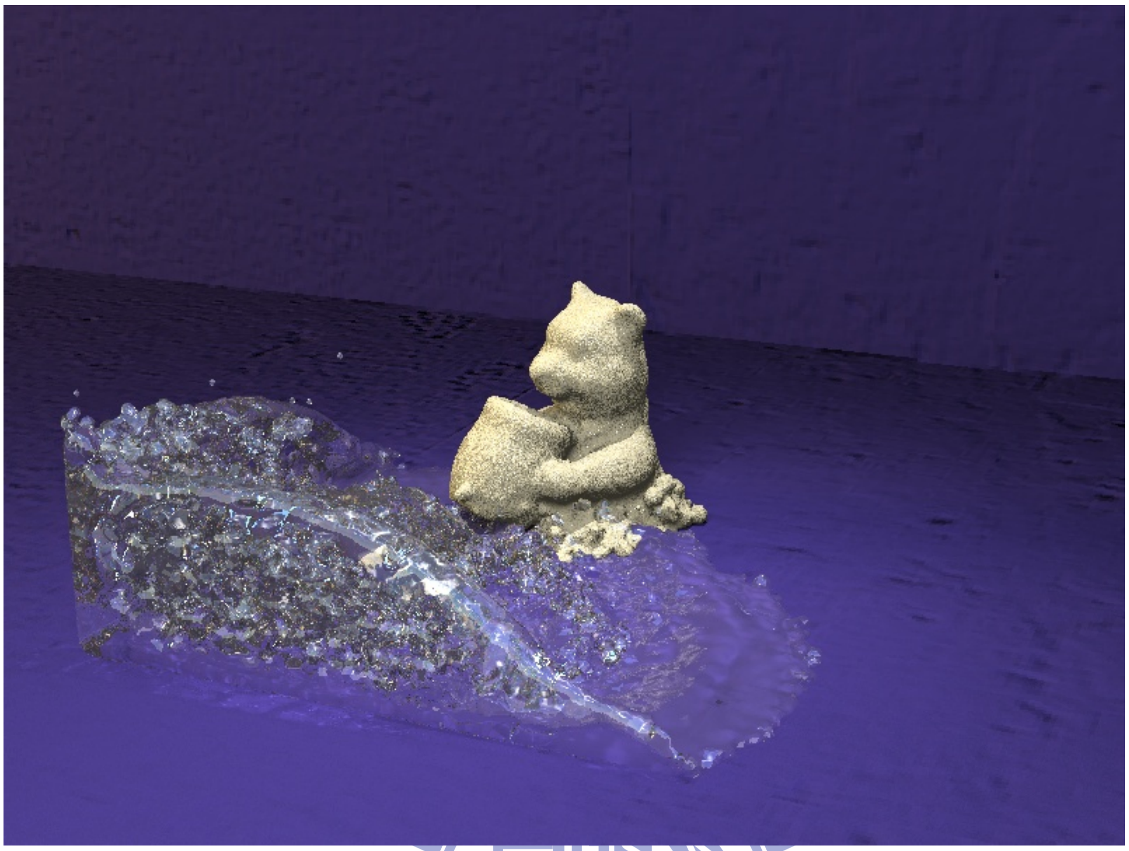


Figure 4.5(k): Step No. 660.

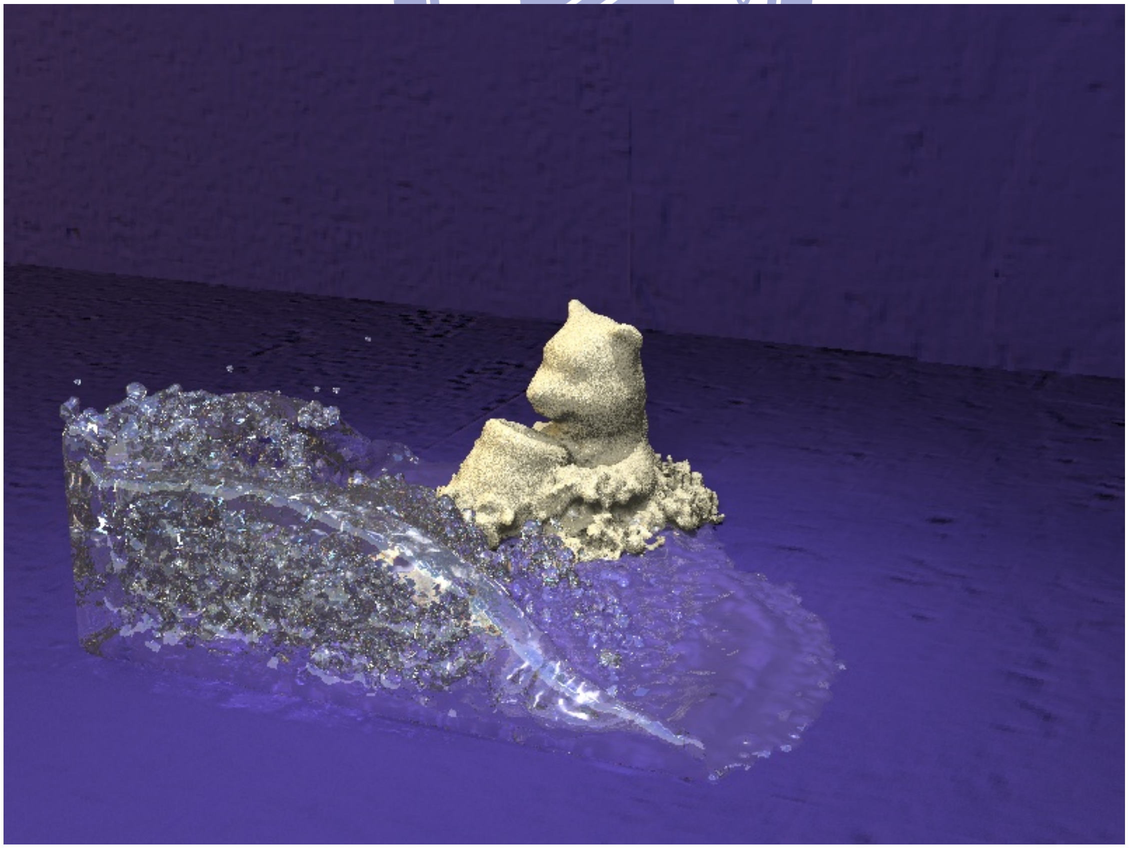


Figure 4.5(l): Step No. 690.

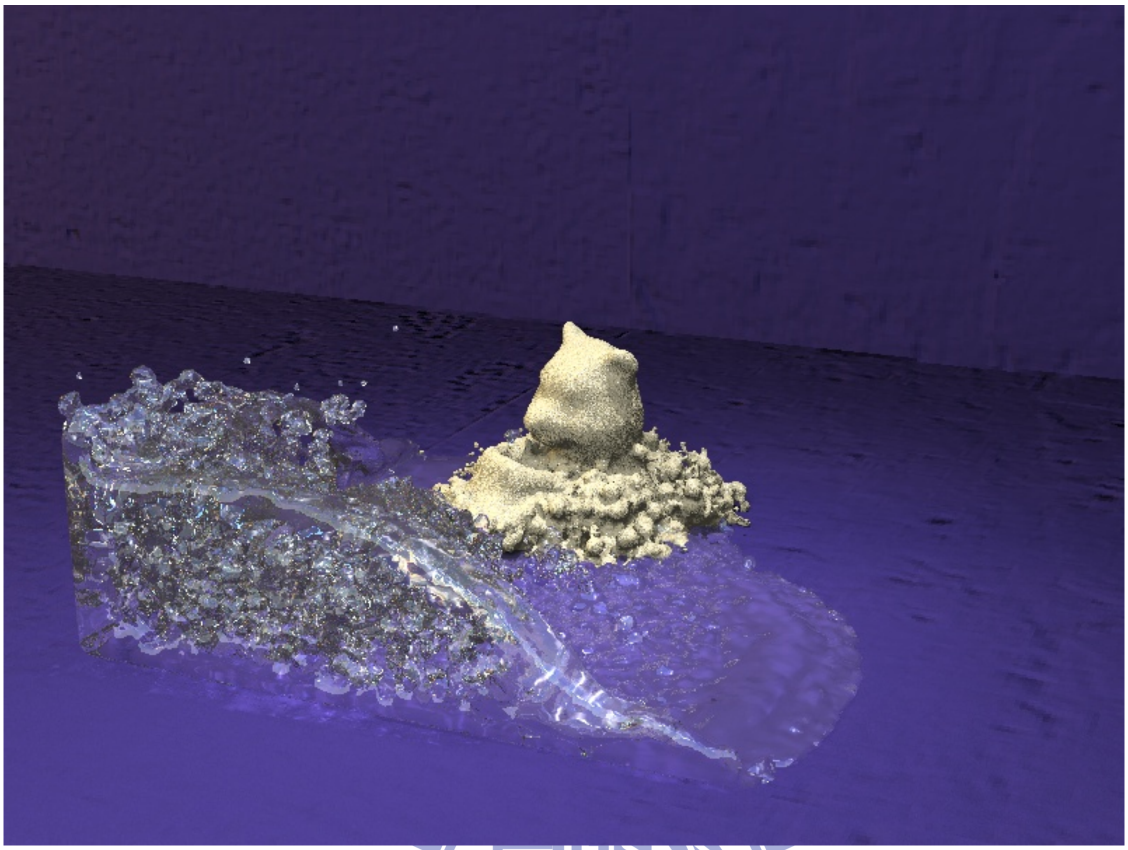


Figure 4.5(m): Step No. 720.

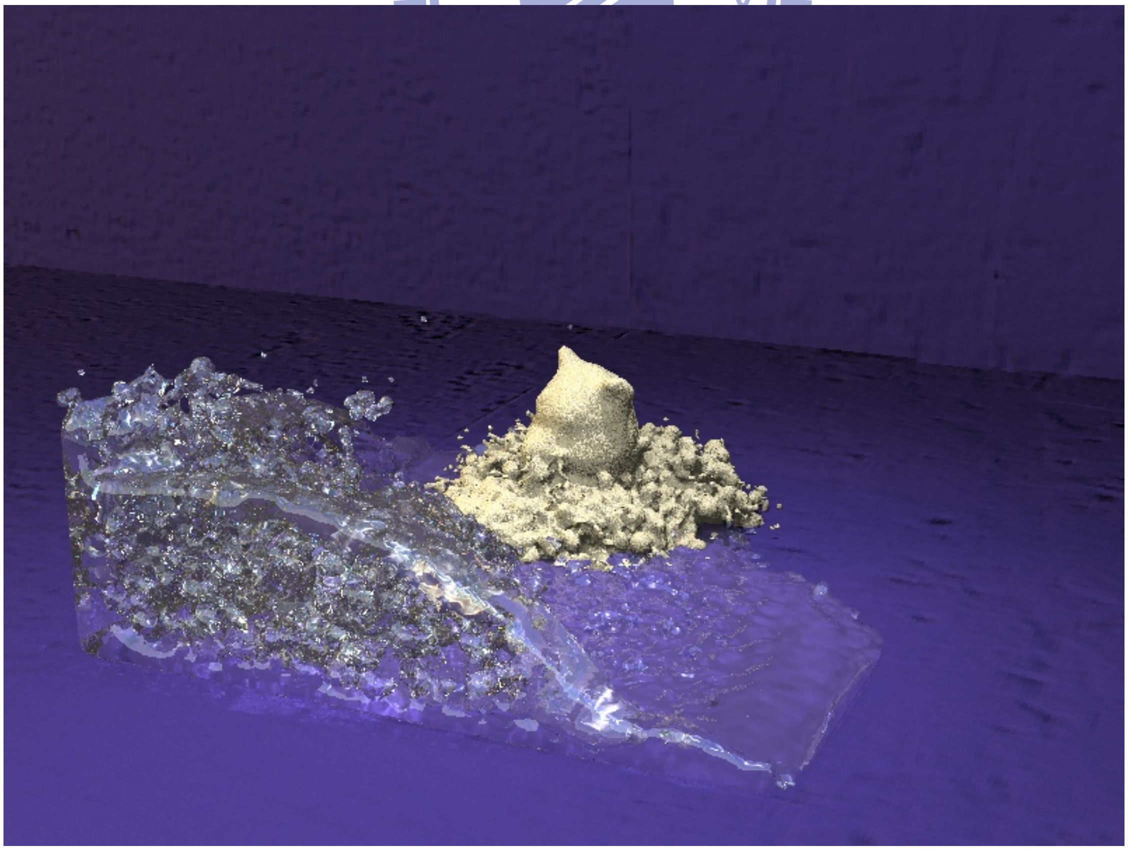


Figure 4.5(n): Step No. 750.

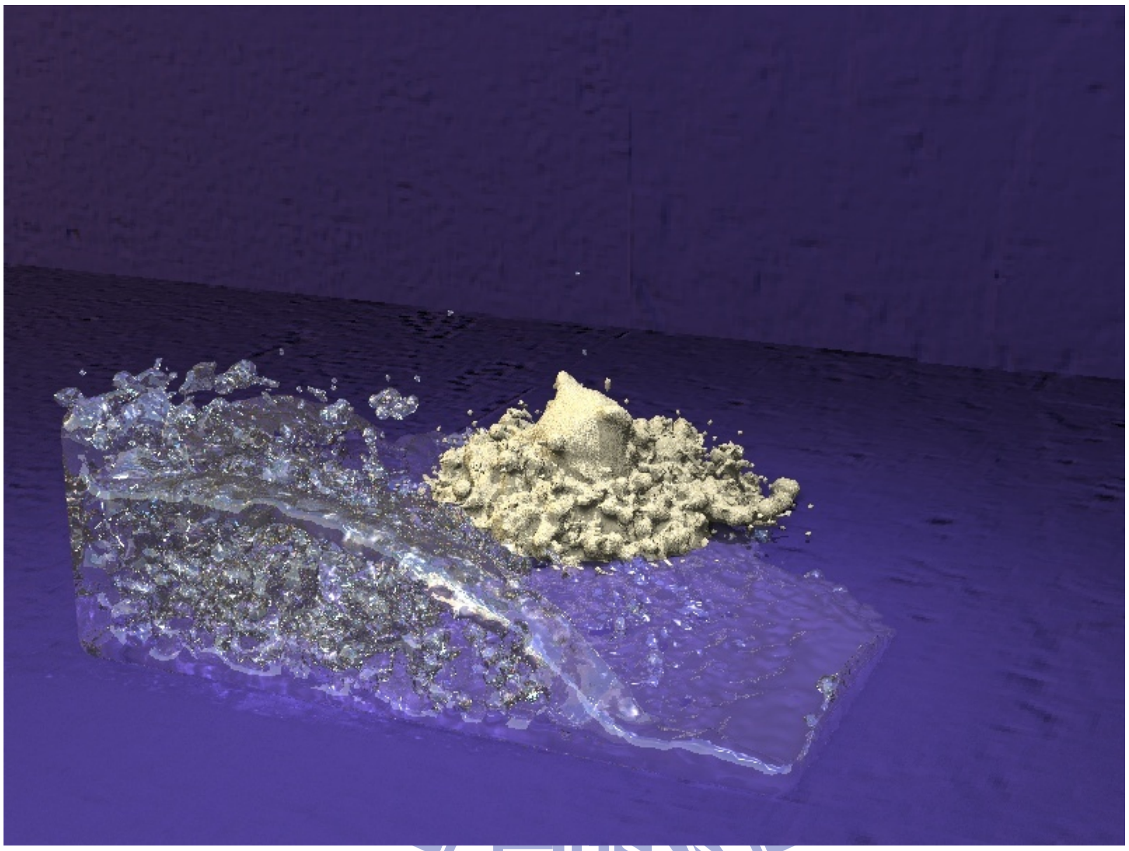


Figure 4.5(o): Step No. 780.

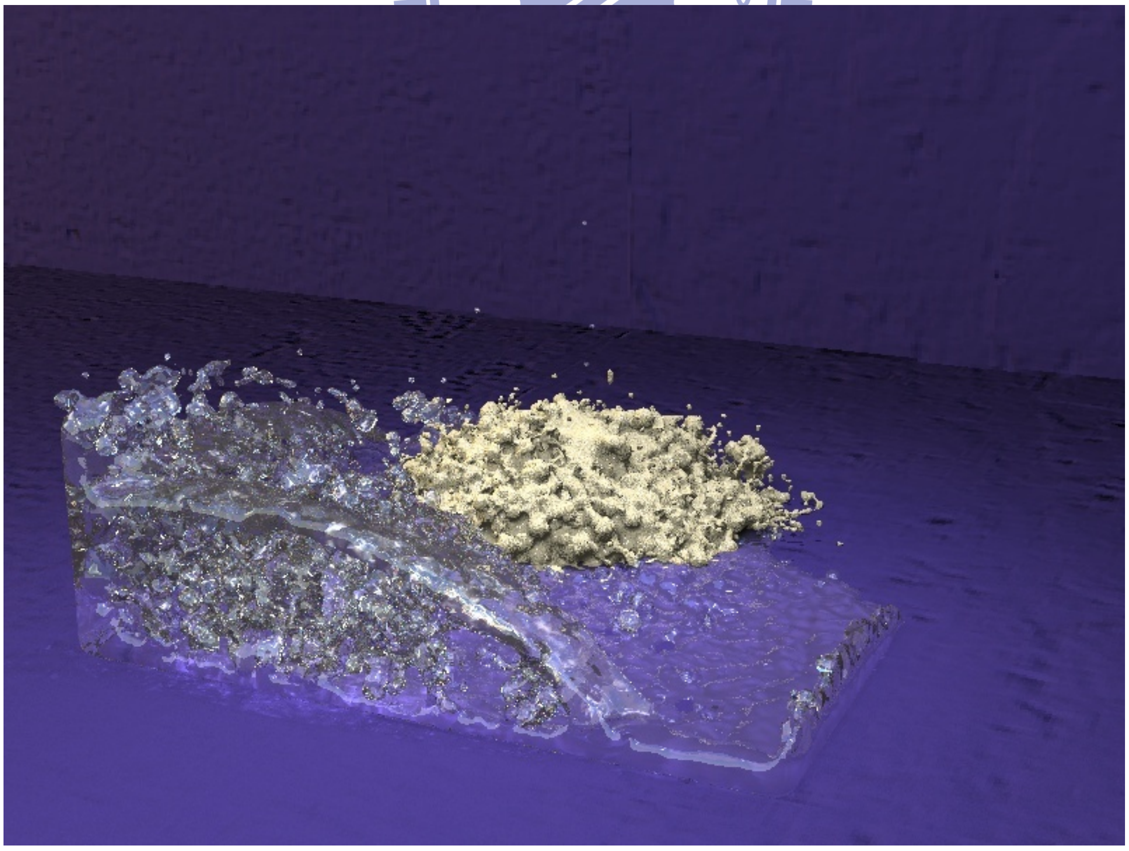


Figure 4.5(p): Step No. 810.

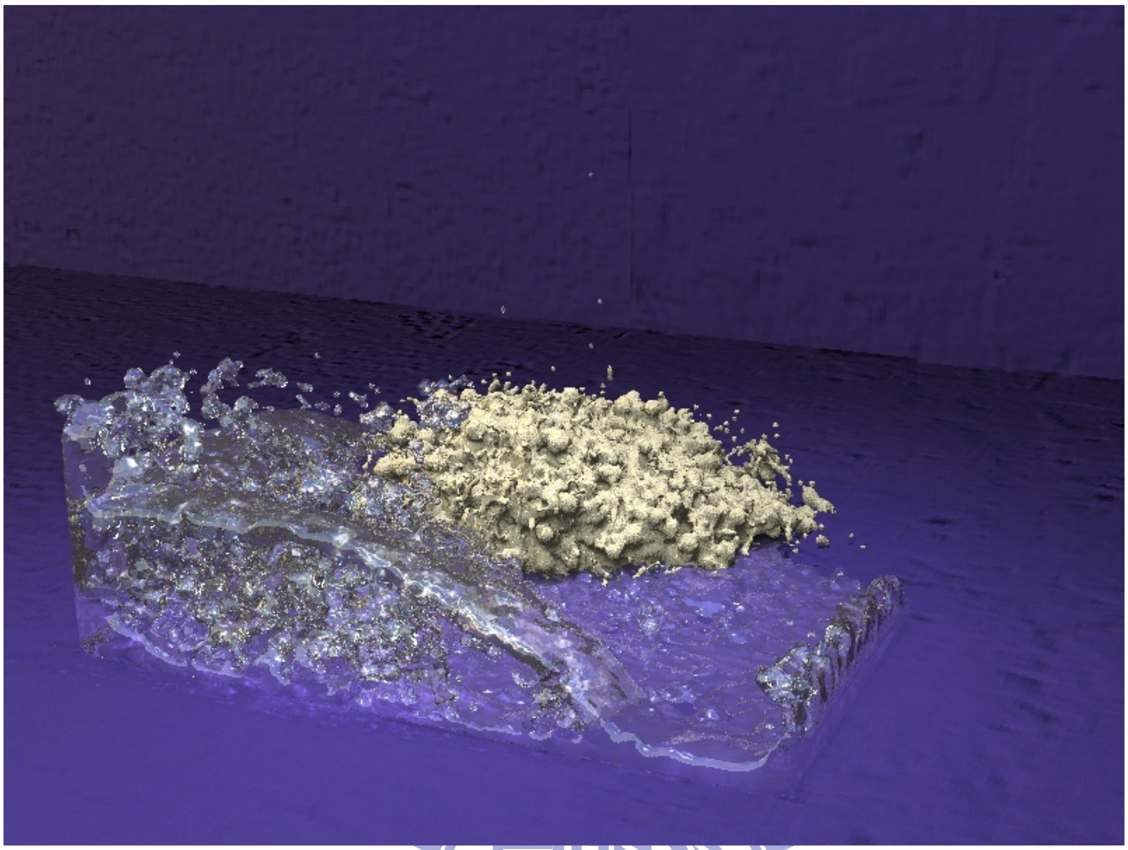


Figure 4.5(q): Step No. 840.

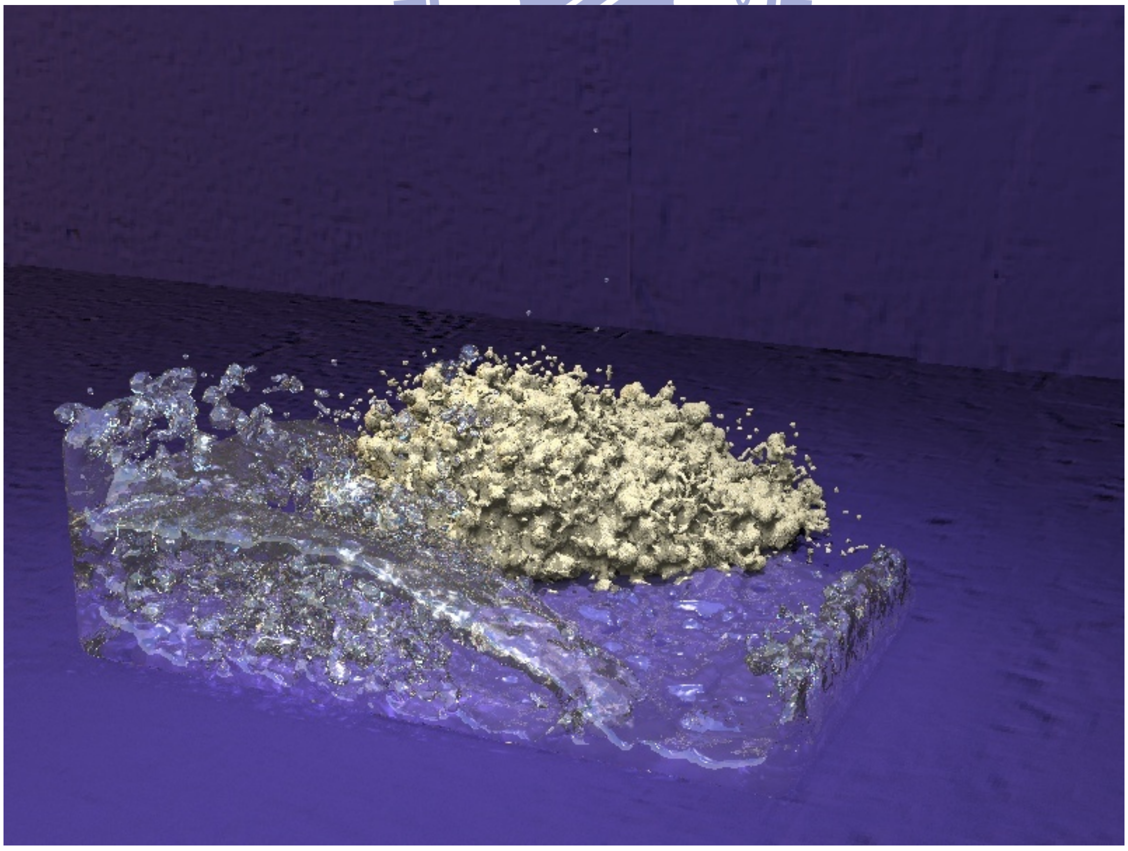


Figure 4.5(r): Step No. 870.

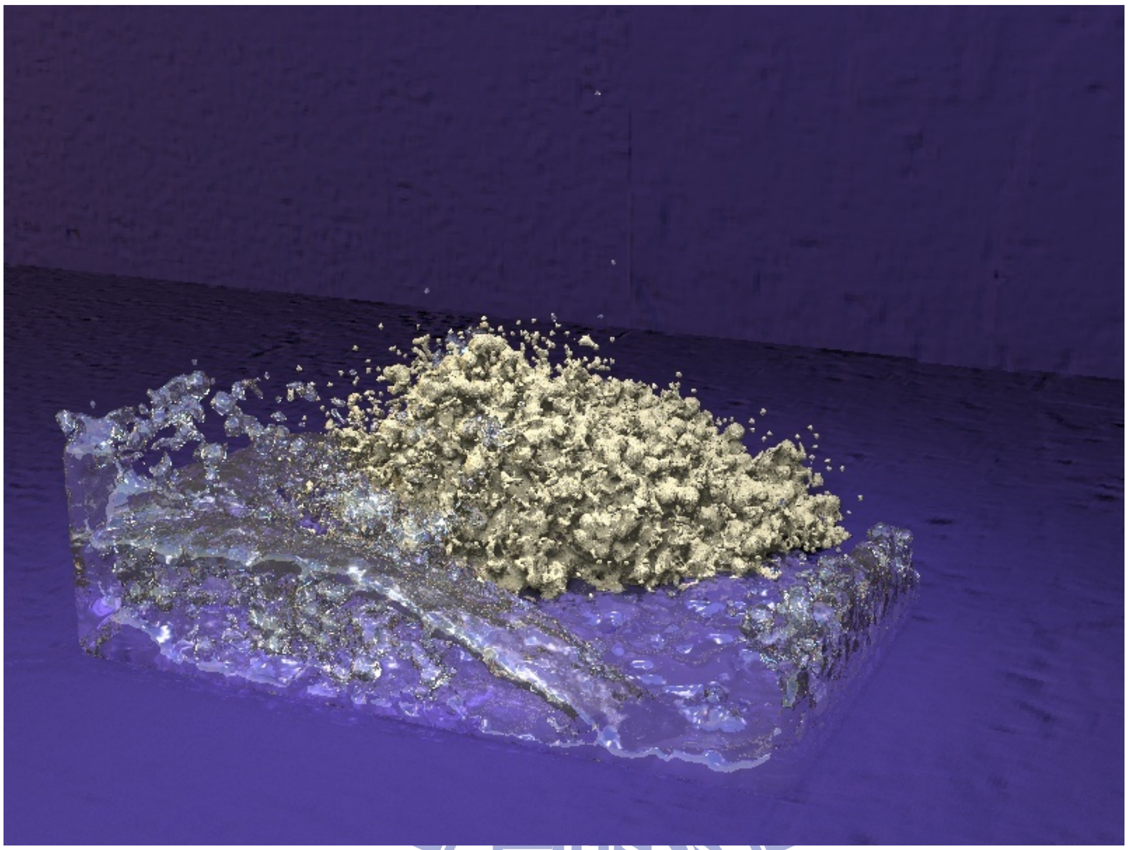


Figure 4.5(s): Step No. 900.

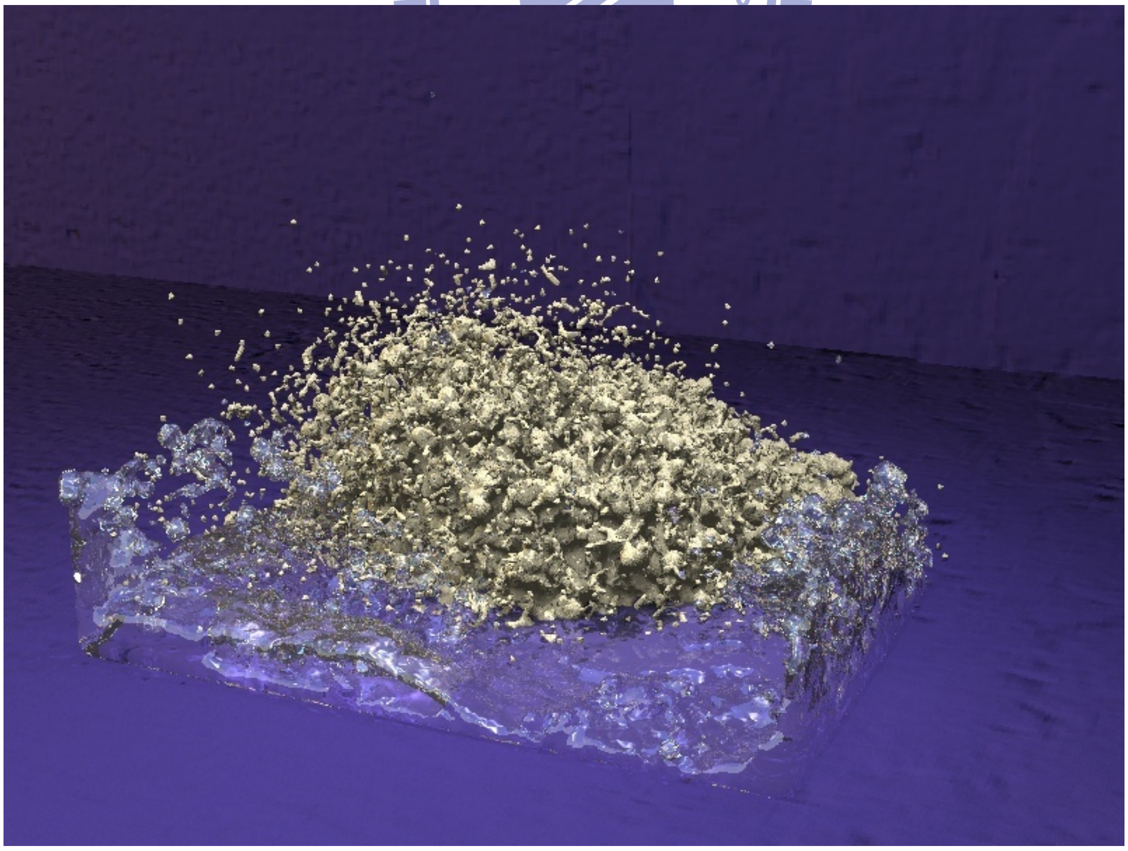


Figure 4.5(t): Step No. 1000.

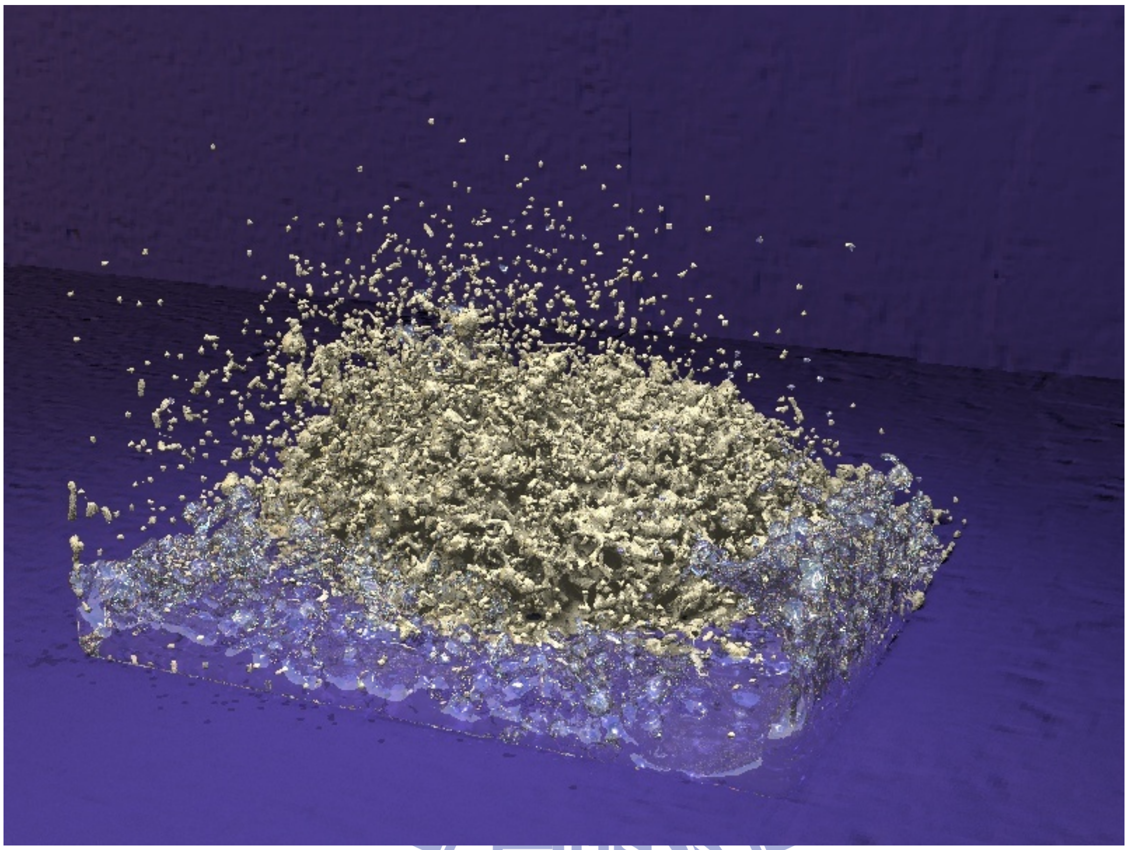


Figure 4.5(u): Step No. 1100.

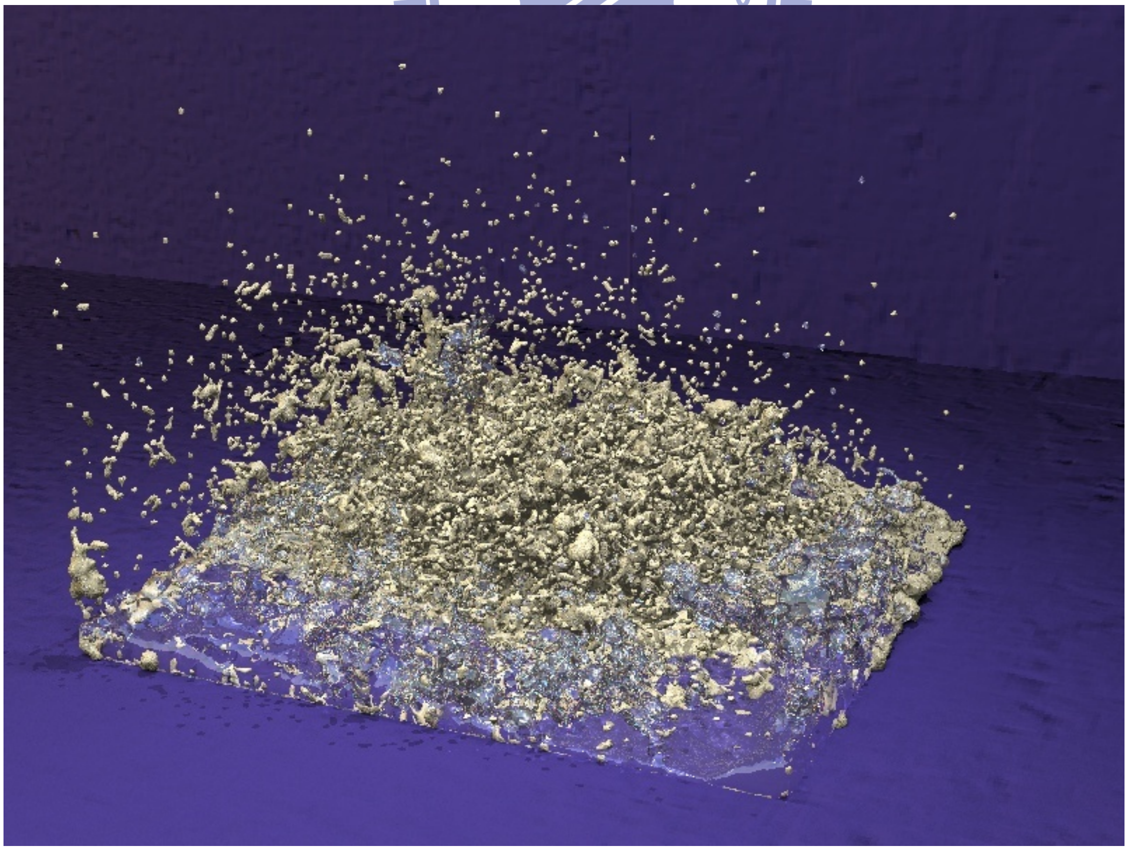


Figure 4.5(v): Step No. 1200.

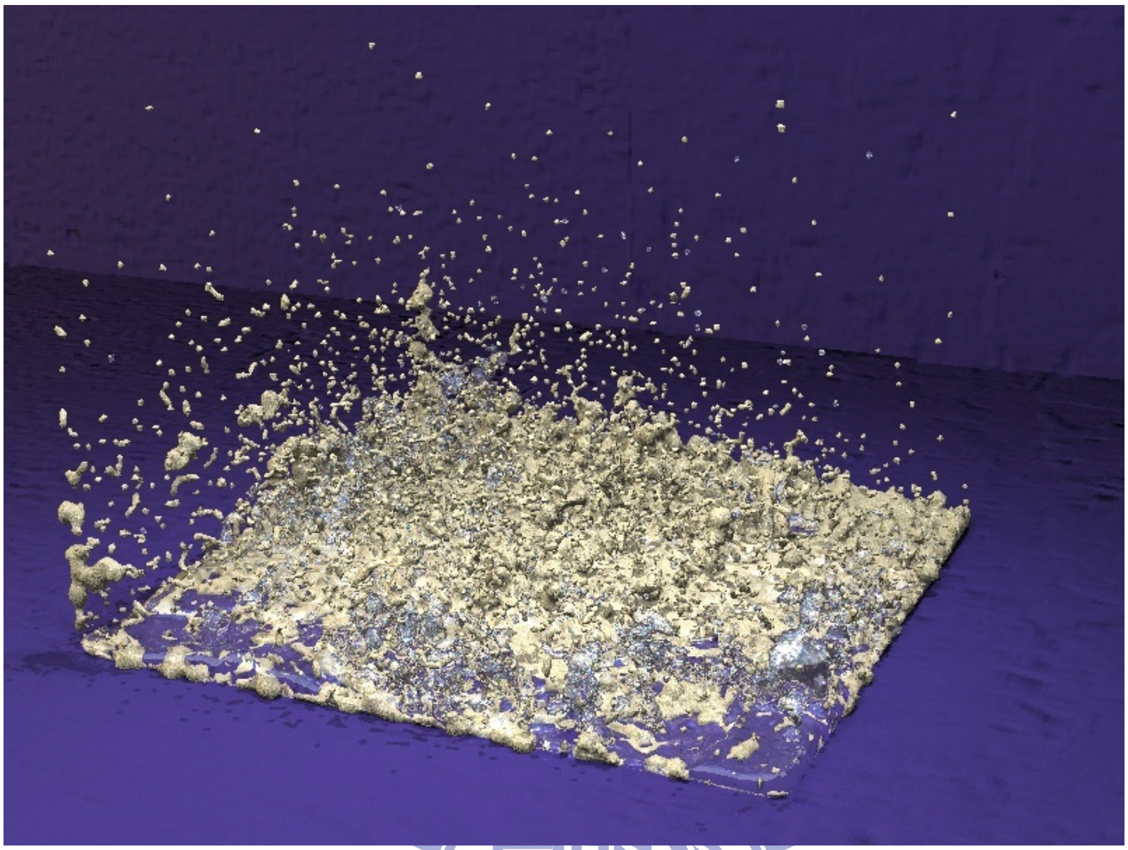


Figure 4.5(w): Step No. 1300.

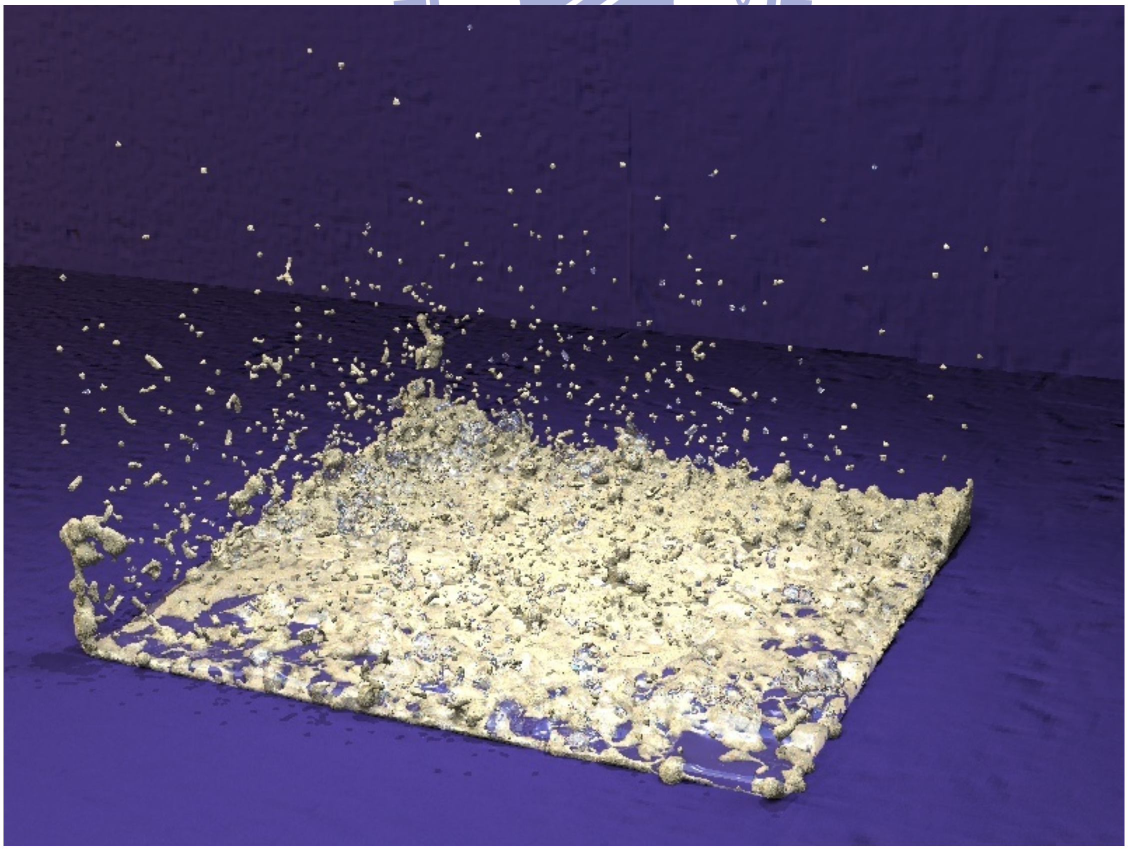


Figure 4.5(x): Step No. 1400.

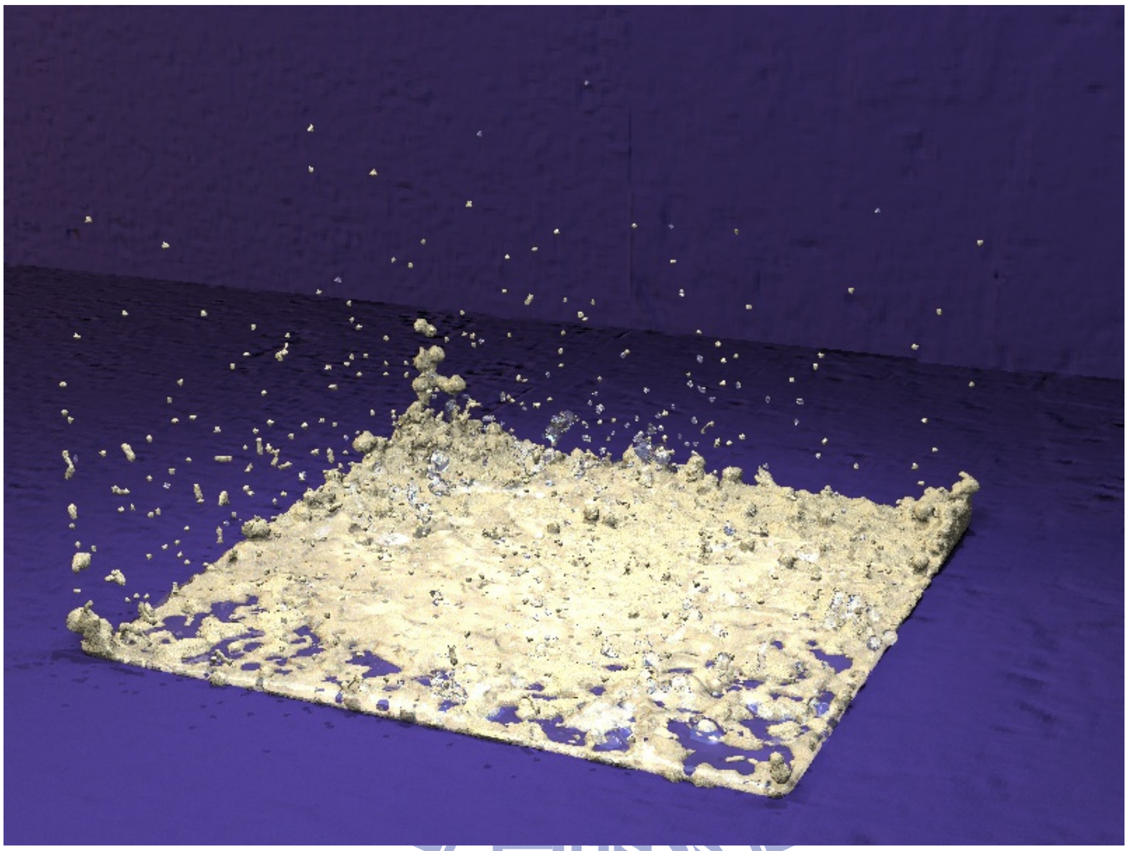


Figure 4.5(y): Step No. 1500.

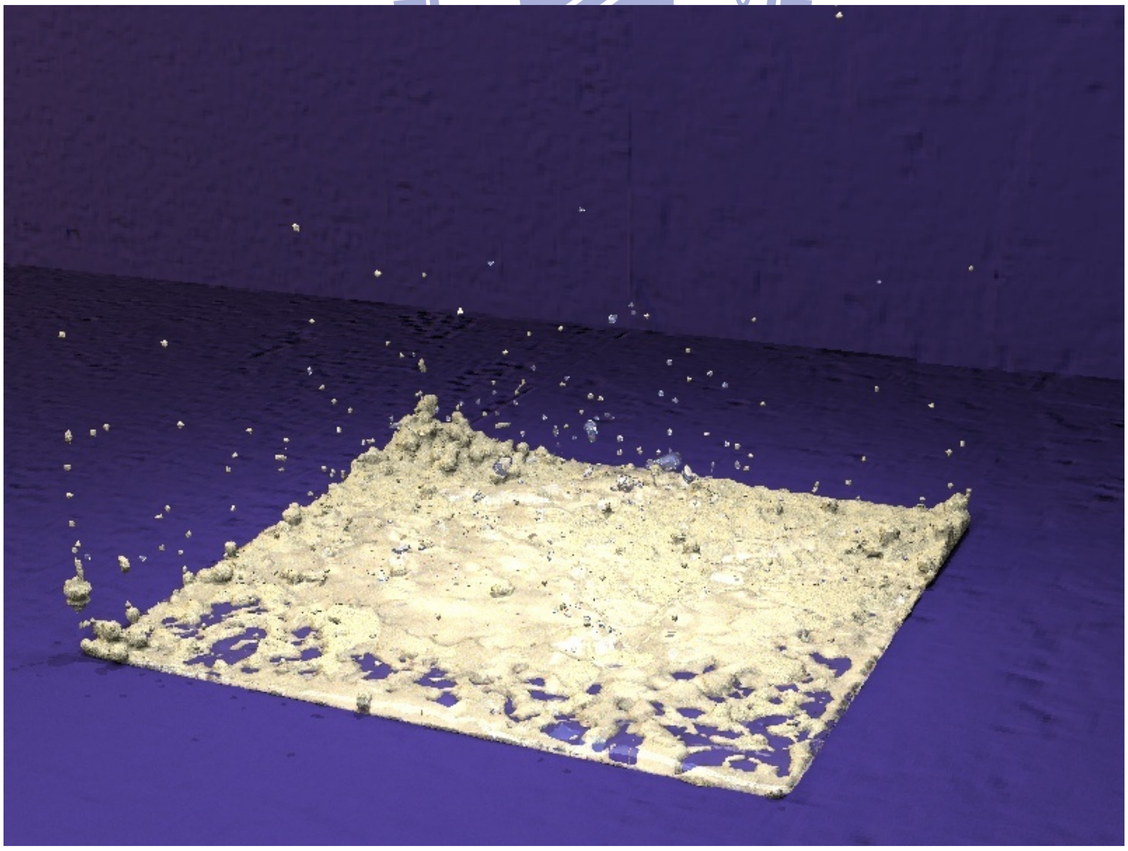


Figure 4.5(z): Step No. 1600.

Chapter 5
Conclusion and Future Works

In this thesis, we present a method to combine various kinds of objects using adaptive sampling for particle-based sand and water interaction. Our model can combine the sand and water particles completely. The water particles are absorbed during the interaction with sand particles. By using the adaptive sampling algorithm, we can handle huge amount of particles with higher speed than before. Our system enables a realistic effect of animating the interaction between sand sculptures and water. During the simulation, the water collapses sand sculptures and sand particles turn into mud.

However, we do not consider the air particles in our system. For example, the bubbles generated when the water hit the sand sculptures. In the future, we plan to extend our scheme to produce the bubble effect as in the work of Cleary et al. [8] and Hong et al. [15].

References

[1] Adams B., Pauly M., Keiser R., Guibas L. J.: Adaptively sampled particle fluids. In SIGGRAPH '07: ACM SIGGRAPH 2007 papers (New York, NY, USA, 2007), ACM, p. 48.

[2] Baraff D.: An introduction to physically based modeling: Rigid body simulation 1 - unconstrained rigid body dynamics. SIGGRAPH Course Notes, 1997.

[3] Becker M., Teschner M.: Weakly compressible SPH for free surface flows. In SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation (2007), pp. 209–217.

[4] Bell N., Yu Y., Mucha P. J.: Particle-based simulation of granular materials. In SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation (New York, NY, USA, 2005), ACM Press, pp. 77–86.

[5] Carlson, M., Mucha, P. J., Turk, G. 2004. Rigid fluid: animating the interplay between rigid bodies and fluid. *ACM Trans. Graph. (Proc. SIGGRAPH)* 23, 377–384.

[6] Carlson, M., Mucha, P., Van Horn III, R., and Turk, G. 2002. Melting and flowing. In *Proc. ACM SIGGRAPH/Eurographics Symp. Comp. Anim.*, 167-174.

[7] Clavet S., Beaudoin P., Poulin P.: Particle-based viscoelastic fluid simulation. In SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation (New York, NY, USA, 2005), ACM Press, pp. 219–228.

[8] Cleary, P. W., Pyo, S. H., Prakash, M., and Koo, B. K. 2007. Bubbling and frothing liquids. ACM Trans. Graph. (SIGGRAPH Proc.) 26, 3, 971–976.

[9] Darcy, H. 1856. Les Fontaines Publiques de la Ville de Dijon, Dalmont, Paris.

[10] Desbrun, M., and Cani, M.-P. 1999. Space-time adaptive simulation of highly deformable substances. Tech. rep., INRIA Nr. 3829

[11] Foster N., Fedkiw R.: Practical animation of liquids. In SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 2001), ACM, pp. 23–30.

[12] Foster N., Metaxas D.: Controlling fluid animation. In CGI ’97: Proceedings of the 1997 Conference on Computer Graphics International (Washington, DC, USA, 1997), IEEE Computer Society, p. 178.

[13] Foster N., Metaxas D.: Realistic animation of liquids. Graph. Models Image Process. 58, 5 (1996), 471–483.

[14] Génevaux, O., Habibi, A., Dischler, J.-M. 2003. Simulating fluid-solid interaction. In *Graphics Interface*, 31–38.

[15] Hong J.-M., Lee H.-Y., Yoon J.-C., Kim C.-H.: Bubbles alive. In ACM SIGGRAPH (2008), pp. 1–4.

[16] Lenaerts T., Adams B., Dutre P.: Porous flow in particle-based fluid simulations. In *SIGGRAPH 08: ACM SIGGRAPH 2008 Papers* (New York, NY, USA, 2008), ACM, pp. 49:1–8.

[17] Lenaerts, T., and Dutré, P. 2009. Mixing fluids and granular materials. *Computer Graphics Forum* 28, 2, 213–218.

[18] Miller G., Pearce A.: Globular dynamics: A connected particle system for animating viscous fluids. *Computers and Graphics 13*, 3 (1989), 305-309.

[19] Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In SCA ’03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 154–159.

[20] Müller M., Solenthaler B., Keiser R., Gross M.: Particle-based fluid-fluid interaction. In SCA ’05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation (New York, NY, USA, 2005), ACM Press, pp. 237–244.

[21] Monaghan J.: Smoothed particle hydrodynamics. Annual Revision on Astronomy and Astrophysics 30 (1992), 543–574.

[22] Monaghan J. J.: Smoothed particle hydrodynamics. Reports on Progress in Physics 68, 8 (August 2005), 1703–1759.

[23] POV-Ray, http://www.povray.org

[24] Stam J.: Stable fluids. In SIGGRAPH ’99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 121–128.

[25] Solenthaler B., Schlafli J., Pajarola R.: A unified particle model for fluid-solid interactions. Computer Animation and Virtual Worlds 18, 1 (2007), 69–82

[26] Zhu Y., Bridson R.: Animating sand as a fluid. In SIGGRAPH '05: ACM SIGGRAPH 2005 Papers (New York, NY, USA, 2005), ACM Press, pp. 965–972.