Animation of Dust Behaviors in a Networked Virtual Environment

Jim X. Chen, Edward J. Wegman and Jingfang Wang
Department of Computer Science and Center for Computational Statistics
George Mason University
jchen@cs.gmu.edu

ABSTRACT

Real-time simulation of physically realistic complex dust behaviors is very useful in computer simulation, training, education, art, advertising, and entertainment. There is no successful model for realistic dust behaviors generated by a traveling vehicle. In this paper, we use particle systems and behavioral simulation techniques to simulate these dust behaviors in real-time. First we analyze the forces and factors which affect the dust generation and the behaviors after dust particles are generated. Then we construct physically-based empirical models to generate dust particles and control the behaviors accordingly. After that, we further simplify the numerical calculations by dividing the dust behaviors into three stages, and establishing simplified particle system models for each stages. In addition, we discuss the methods to simulate the behaviors in a networked virtual environment. Our major contribution includes analyzing dust behaviors in detail, constructing physically-based empirical models that correlate the behaviors to the dust generating forces and other factors, and that achieve simulations in a networked virtual environment.

Keywords: *Particle Systems, Dust, Real-time Simulation, Physically Inspired Modeling, DIS*

1 Introduction

In many virtual environments and distributed interactive simulations, it is desirable to simulate trucks, tanks, armored vehicles, bulldozers, and other ground moving objects. However, typically dust behaviors are not generated when these objects travel on an unpaved road. Dust behaviors caused by different factors (such as natural wind and a fast traveling vehicle) appear everywhere. Simulating physically realistic, complex dust behaviors is very useful in interactive graphics applications, such as computer art, advertising, education, entertainment, and training. However, due to the lack of modeling and simulation techniques and methodologies, there is currently no successful real-time simulation for realistic dust behaviors. As computers and their graphics systems become much faster and more powerful, many natural phenomena (such as the behaviors of fluids, terrains, trees, fireworks, volcanos, clouds, etc.) are simulated in real-time [Chen97, Li93, Oppenheimer86, Reeves83, Reynolds87, Guritz95, Sims90, Fiume93]. We believe it is appropriate now to include dust behaviors into real-time simulation. Here we use *real-time* to mean that the simulation is interactive, and the frame rate of generating the dust behaviors is at an interactive rate of human perception.

Hsu and Wong [Hsu95] introduced a dust accumulation model. The model presents static appearance of dust accumulation without animation. Cowherd, et. al. [Cowherd89, Williams88] studied dust and the mechanisms of dust generation. Their purpose was to study and measure the density of the dust in the real battlefield instead of simulating the dust behaviors in graphics. Today, military training in graphics and distributed interactive simulation is one of the major research topics [IST94], and generating dust behaviors in real-time significantly increases the realism of the simulated training environment.

In this paper, we introduce a method for simulating the dust behaviors caused by a fast traveling vehicle in real-time. The method is a combination of particle systems and behavioral simulation techniques. The *Particle systems* technique was first introduced to computer graphics by Reeves [Reeves83], and is now widely used to simulate fuzzy or dynamic objects, such as fire, grass, explosions, clouds, water, trees, etc. These objects have no fixed shape and sometimes change their shapes or behaviors stochastically. They have ill-defined boundaries that make surface-based modeling impractical. It is apparent that dust behaviors behind a moving vehicle belong to this category. The *Behavioral simulation* technique uses a physically inspired modeling method to calculate and update the object's state, and draw the object repetitively after each calculations to achieve the behavior animation in real-time. We also employ motion blur for small and fast moving particles, particle blending instead of hidden-surface removal, texture mapping, and other graphics techniques to achieve better performance and appearance of the final results.

In order to build up a physically inspired realistic simulation, we first analyze the forces and factors which affect the dust generation and the behaviors after dust particles are generated, and then construct physically inspired empirical models to generate dust particles and control the behaviors accordingly. However, the models are time-consuming and inefficient. Therefore, based on the models and analysis of the forces, we further simplify the numerical calculations by dividing the dust behaviors into three stages, and establishing simplified particle system models for each stages. The resulting models are satisfactory for real-time simulation as well as achieving realistic dust behaviors.

Distributed Interactive Simulation (DIS) requires a time and space coherent representation of a virtual environment. Simulating physically realistic behaviors in DIS presents an extremely challenging problem. The synchronization of physical activities such as dust behaviors in DIS is important to guarantee fast and accurate simulation. In this paper, we also discuss a mechanism which uses a uniform time scale proportional to the clock-time and variable time-slicing to synchronize physical models such as dust particle system in a networked environment for DIS. The advantage of this system is the mechanism of variable time slice corresponding to each simulator's own clock, which reduces the network communication significantly. We have also developed a time differential strategy to maintain numerical stability of the simulation. With this strategy, the physical numerical calculation will be stable no matter how the time slice changes.

Our major contribution includes analyzing dust behaviors in detail, constructing physically inspired empirical models that correlate the behaviors to the dust generating forces and other factors, and that achieve simulations in a networked virtual environment. Our work is a useful addition to many applications in simulated virtual environments, including military simulation and training.

The rest of the paper is organized as follows. In Section 2 we discuss the dust generation, the factors affecting the behaviors after the dust particles are generated, and dynamics of a dust particle. In Section 3 we simplify the models, and divide the dust particle systems into three stages (*turbulent vortex*, *inertial momentum*, and *airborne drift*.) In Section 4 we discuss some rendering issues and present some simulation results with different parameters. In Section 5 we discuss synchronization in a networked virtual environment. Finally, in Section 6, we summarize our work, and describe several avenues of future research.

2 Dust Behaviors

In this section, we first discuss and analyze how dust particles are generated; then we introduce the factors which affect the dust behaviors, after that we analyze the forces acting on a dust particle and establish corresponding physically inspired empirical models to calculate and update the dust behaviors.

2.1 Dust Generation

As a vehicle passes over an unpaved surface, three basic forces are developed: vertical pressure, horizontal stress, and friction. Vertical pressure, which is due to the weight of the vehicle (*WT_{car}*), will produce ground surface vibration and/or deformation, crushing large particles into smaller ones. Horizontal stress and friction, which are largely due to the driving power which sustains the velocity (V_{car}) and acceleration of the vehicle, will further comminute the particles and carry them on the surface of the tire. The slippage between the tire and the ground will lift particles of different sizes due to the adhesive and shear forces, and eject them at different places on the tire surface due to the centrifugal forces as shown in Fig. 1.

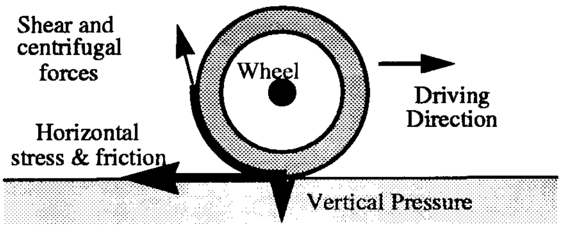

Fig. 1: Forces between the wheels and the ground

The ground vibration and/or deformation will also eject fine particles. The dust particles are then either entrained in the turbulent air behind the vehicle or return to the ground depending on their properties and conditions. Bigger particles will fall back to the ground surface more rapidly while the fine ones will remain suspended in the air drifting with the current. Small stones and blocks of muds will fall back to the ground immediately after ejection from the tires, and will bounce up and down, also generating dust into the air.

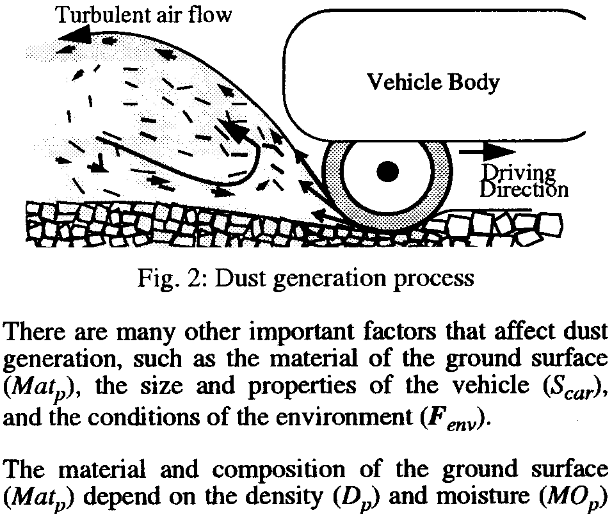

Fig. 2: Dust generation process

There are many other important factors that affect dust generation, such as the material of the ground surface (Mat_p), the size and properties of the vehicle (S_{car}), and the conditions of the environment (F_{env}).

The material and composition of the ground surface (Mat_p) depend on the density (D_p) and moisture (MO_p) of the ground surface, the average size (S_p), mass, and adhesion of each individual dust particle. If the ground is wet and the average size of the particles is large, then there will be fewer particles. If the dust density of the ground surface is high, then there will be more particles. We use the following equation to measure this parameter:

$$Mat_p = \frac{D_p}{\alpha_1 \cdot S_p + \alpha_2 \cdot MO_p} \qquad (1)$$

where $\sum_{i=1}^{2} \alpha_i = 1$, and α_i is the weight coefficient of the corresponding parameter, for $i = 1, 2$.

The size and properties of the vehicle (S_{car}) depend on the weight ($\boldsymbol{WT}_{car}$), height (H_{car}), and width (W_{car}) of the vehicle. Heavier and bigger vehicle will generate more dust particles. We use the following equation to measure this parameter:

$$S_{car} = \beta_1 \cdot |WT|_{car} + \beta_2 \cdot H_{car} + \beta_3 \cdot W_{car} \qquad (2)$$

where $\sum_{i=1}^{3} \beta_i = 1$, and β_i is the weight coefficient of the corresponding parameter, for $i = 1, 2, 3$.

The conditions of the environment ($\boldsymbol{F}_{env}$) including humidity, air pressure, and many other environment damping forces will also affect the dust generation.

In summary, most dust particles are generated right behind the wheels. Some fine dust particles are lifted from the ground surface because of the turbulent wake behind the vehicle. Each particle is generated with its own initial mass, size, and velocity. The number of dust particles generated by the vehicle for each simulating frame is calculated as follows:

$$N_p = \frac{|\boldsymbol{V}_{car}|(\gamma_1 \cdot Mat_p + \gamma_2 \cdot S_{car})}{|\boldsymbol{F}_{env}|} \qquad (3)$$

where $\sum_{i=1}^{2} \gamma_i = 1$, and γ_i is the weight coefficient of the corresponding parameter, for $i = 1, 2$.

Once the particles are entrained in the turbulent wakes behind the vehicle, their behaviors are affected by similar factors which will be discussed in the next section.

2.2 Factors Affecting Dust Behaviors

There are numerous factors that affect the dust behaviors caused by a fast traveling vehicle. Here we summarize the important factors that have more serious effects on the dust behaviors.

Table 1: Parameters affecting the dust behaviors

Items	Parameters	Description
Vehicle	Velocity — V_{car} Height — H_{car} Width — W_{car}	Decide the size of the 3D volume behind a car where the turbulent wake is generated.
Dust particle	Size — S_p Mass — m_p	Affect how an individual particle will react to the external forces
Environment	Wind — V_{air} Moisture — MO_{air} Damping — particle movement	Influence the dust behaviors in general.

When a vehicle travels quickly, it produces a 3D volume behind where the atmosphere pressure is lower than that of the other areas. The turbulent wake is mostly inside this volume and generates vigorous dust behaviors. The shape and size of the 3D volume are mainly decided by the velocity, height, and width of the vehicle. At the same time, the properties of each individual dust particle will affect its own behaviors. These properties include the shape, size, mass, and initial conditions. Intuitively, a particle will fall back to the ground faster if it is heavier and smaller. In addition, the humidities, wind, and environmental damping forces have an effect on the dust behaviors also. We summarize the parameters which have the greatest influence on the dust behaviors in table 1.

2.3 Dynamics of a Dust Particle

Computational fluid dynamics could be used to calculate the exact turbulent wake behind a vehicle. However, this approach is computationally complex and prohibits achieving simulation in real-time. In order to describe the dust behaviors caused by a fast traveling vehicle, here we simplify the dynamics of a dust particle, analyze all the important forces, correlate these forces with the parameters affecting the dust behaviors, and establish analytical models which can be used to simulate the dust behaviors in real-time. Here we first analyze the forces behind a traveling vehicle, then we study the effects of these forces on an individual particle.

As the vehicle travels forward quickly, it produces a 3D volume behind the vehicle where the atmosphere pressure is smaller than that of the other areas. To simplify the situation for our analysis, let's assume that the vehicle does not turn, and the area affected by the vehicle is a box. That is, our particle systems' range is a box moving at the speed of the vehicle. The box's height and width are the same as those of the vehicle's (H_{car}, W_{car}), and its length (L) depends on the vehicle's velocity. Because of the fast movement of the vehicle, different places within the box have different atmosphere pressures. The differences among the pressures will generate turbulent wakes, and the dust behaviors accordingly (Fig. 3).

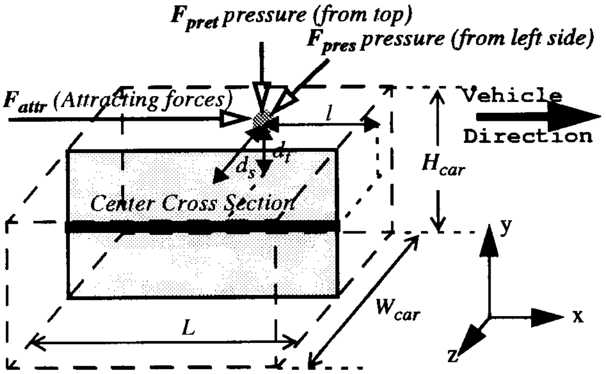

Fig. 3: The pressures (forces) generated

There are five kinds of forces acting on any particle inside the box area:

- The pressure or attraction force F_{attr} towards the rear side of the vehicle
- The *atmosphere pressure force* F_{pres} from the two sides of the box area
- The *atmosphere pressure force* F_{pret} from the top and bottom of the box area
- The atmosphere damping force F_{air} against the particles relative movement
- The dust particle's gravity F_{grv}

F_{attr} is a function of the vehicle's velocity (V_{car}), the size of the vehicle (S_{car}), and the distance between the dust particle and the rear side of the vehicle (l). As the vehicle moves ahead, l becomes larger, and therefore F_{attr} is reduced rapidly. Its direction is approximated by the direction which the vehicle travels. We have the following equation:

$$F_{attr} = \zeta\left(\frac{V_{car} \cdot S_{car}}{l}\right) \tag{4}$$

where ζ is a constant scale parameter.

F_{pres} is a function of the vehicle's velocity (V_{car}), the distance between the dust particle and the rear side of the vehicle (l), and the distance between the particle and the horizontal cross section in the center of the box (d_s). As the vehicle moves ahead, l becomes larger, and therefore F_{pres} is reduced rapidly. Its direction is towards the center cross section and parallel to the ground, which is approximated by the direction of $\mathbf{d}_s$. Here we use bold to represent the vector direction as well as magnitude. We have the following equation:

$$F_{pres} = \vartheta_1\left(\frac{|V_{car}| \cdot \mathbf{d}_s}{l}\right) \tag{5}$$

where ϑ_1 is a constant scale parameter.

F_{pret} is similar to F_{pres}. It is a function of the vehicle's velocity (V_{car}), the distance between the dust particle and the rear side of the vehicle (l), and the distance between the particle and the horizontal cross section in the center of the box (d_t). Its direction is towards the center cross section and perpendicular to the ground, which is approximated by the direction of $\mathbf{d}_t$. We have the following equation:

$$F_{pret} = \vartheta_2\left(\frac{|V_{car}| \cdot \mathbf{d}_t}{l}\right) \tag{6}$$

where ϑ_2 is a constant scale parameter.

F_{air} is a function of the particle's velocity (V_p), the particle's size (S_p), the environment wind (V_{air}), and the air moisture (MO_{air}). Its direction is against the particle's movement in opposite to V_p. *We have the following equation:*

$$F_{air} = (\delta_1 \cdot V_{air} - \delta_2 \cdot V_p)(\eta_1 \cdot S_p + \eta_2 \cdot MO_{air}) \tag{7}$$

where δ_1, δ_2, η_1, and η_2 are constant parameters.

In summary, all the forces acting on a particle are shown in Fig. 4. Here we ignore the collisions among the dust particles.

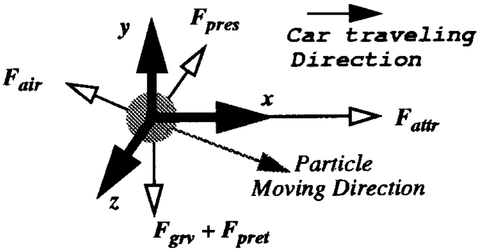

Fig. 4: *Forces acting on a dust particle*

Let F_p be the force acting on a dust particle, P the position, V_p the velocity, A_p the acceleration, and m_p the mass of the particle. Then a dust particle's behavior is described by the following equations:

$$F_p = F_{attr} + F_{pres} + F_{pret} + F_{grv} + F_{air} \tag{8}$$

$$A_p = \frac{F_p}{m_p} \tag{9}$$

$$V_p = V_0 + \int_{t_0}^{t_1} A_p dt \tag{10}$$

$$P = P_0 + \int_{t_0}^{t_1} V_p dt \tag{11}$$

To simplify the calculation, we use Euler's method to approximate the particle's next state:

$$V_i = V_{i-1} + A_p \cdot \Delta t \tag{12}$$

$$P_i = P_{i-1} + V_i \cdot \Delta t \tag{13}$$

The algorithm to compute the solution to the dust behaviors then is as follows. For the known current state of a particle $\{V_{i-1}, P_{i-1}\}$, the next state $\{V_{i-1}, P_{i-1}\}$, after Δt time, is calculated by Equation (12) and (13). These equations use functions (4) to (9). Equation (1) to (3) are used to generate a number of dust particles. We have frames of the dust behaviors as in Fig. 5. Changing the parameters and conditions, we can achieve different behaviors and appearances to suit the needs of the applications. The simulation is at about 8 frames per second.

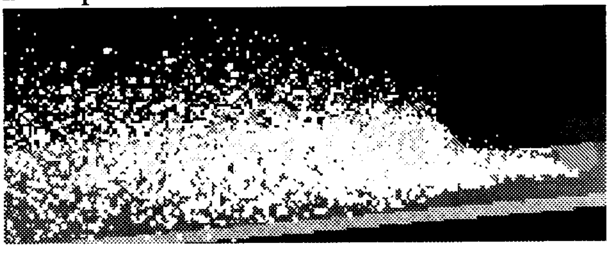

Fig. 5: A frame of dust simulation

3 Simplified Dust Particle Systems

The above physically inspired dust particle model is time-consuming for a large number of particles. There are too many factors in the equations, and the forces on each individual particle have to be calculated during the whole simulation period. There are redundant calculations because when the particles are further away from the vehicle, the forces $\boldsymbol{F}_{attr}$, $\boldsymbol{F}_{pres}$, and $\boldsymbol{F}_{pret}$ are all reduced to near zero. Based on the above analysis (equation 4-7), we can divide a dust particle's behaviors into three stages to further simplify the simulation. We assume three different particle systems (models) working together to simulate the dust behaviors. The three stages are called *turbulent vortex, inertial momentum*, and *airborne drift*, respectively.

3.1 Turbulent Vortex (the First Stage)

Once a particle is generated, the initial forces $\boldsymbol{F}_{attr}$, $\boldsymbol{F}_{pres}$, and $\boldsymbol{F}_{pret}$ acting on it are very large (see Fig. 3, Fig. 4, and equation 4-7). $\boldsymbol{F}_{attr}$ causes the particle to move in the forward direction, $\boldsymbol{F}_{pret}$ causes the particle to move up and down, and $\boldsymbol{F}_{pres}$ causes the particle to move left and right. All other forces are relatively small at this time. Suppose the particle is located in side 1 of the box area, because the particle has a side pressure force $\boldsymbol{F}_{pres}$ pointing towards side 2, the particle will move from side 1 to side 2. Once the dust particle goes across the center section into side 2, $\boldsymbol{F}_{pres}$ will change its direction, and the particle will accelerate and move back from side 2 to side 1. It is similar for $\boldsymbol{F}_{pret}$ but in a perpendicular direction. At this stage, the forces acting on the dust particles are relatively strong. Overall, depending on the initial velocity, the particle will behave as in a turbulent vortex shown in Fig. 6.

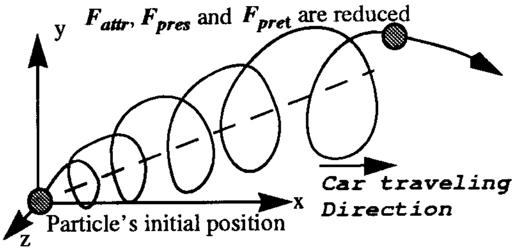

Fig. 6: Dust particle traveling trace (the first step)

Instead of calculating all the forces, we can simplify the model by just simulating the turbulent vortex with some random behaviors at this stage. Particles are rotating around the center of the vortex. Here we assume the center of the vortex is the center of the cross section in the box area (Fig. 3). Its height is $\frac{H_{car}}{2}$. The angle of rotation (*RotAngle*) for each frame of simulation can be calculated by the following simplified equation:

$$RotAngle = \frac{RotD}{|\boldsymbol{R}|^{Tightness}} \tag{14}$$

where $\boldsymbol{R} = \boldsymbol{d}_s + \boldsymbol{d}_t$, which is the distance from the particle to the center of rotation (we use bold to represent vectors). Its initial value is a function of N_p (equation 3). We use the following equation to adjust $\boldsymbol{d}$ for each simulation frame:

$$|\boldsymbol{R}|_i = |\boldsymbol{R}|_{i-1} \cdot (1 + \Delta r) \tag{15}$$

$$\Delta r = dR + k_1 \cdot \frac{m_p}{S_p} \cdot myRandom(- \ , 0, \ ) \tag{16}$$

where *dR* is a constant related to the initial velocity towards the center of the rotation, k_1 is a constant coefficient, m_p is the mass of the particle, S_p is the size of the particle, function *MyRandom*(a, b, c) returns a random pick of the numbers enclosed, and   is a comprehensive weight parameter reflecting the vehicle's properties:

$$= \frac{\delta_1 \cdot S_{car} + \delta_2 \cdot |\boldsymbol{V}_{car}|}{S_{car} + |\boldsymbol{V}_{car}|} \tag{17}$$

where $\sum_{i=1}^{2} \delta_i = 1$, δ_i is the weight coefficient of the corresponding parameter, for $i = 1, 2$; $0 \leq \ \ \leq 1$.

The value *Tightness* is usually between 1.0 and 2.0. Higher *Tightness* causes *RotAngle* to fade away more quickly when the distance becomes larger.

RotD is the distance the particle traveled at each time frame around the vortex circle, perpendicular to the center of vortex. We use the follow equation to calculate this value:

$$RotD_i = RotD_{i-1} \cdot (1 + \Delta D) \tag{18}$$

$$\Delta D = dD + k_2 \cdot \frac{m_p}{S_p} \cdot myRandom(- \ , 0, \ ) \tag{19}$$

where *dD* is a constant related to the initial velocity perpendicular to the center of the rotation, and k_2 is small constant.

A dust particle may be below ground as its rotating radius increases. In our simulation, we just remove those particles which hit the ground.

The particle's translation distance (*TransD*) along the vortex axis is calculated as follows:

$$TransD_i = TransD_{i-1} \cdot (1 + \Delta TD) \tag{20}$$

$$\Delta TD = dTD + k_3 \cdot \frac{m_p}{S_p} \cdot myRandom(0, \ ) \tag{21}$$

where *dTD* is a constant related to the initial velocity parallel to the center of the rotation, and k_3 is a small constant.

3.2 Inertial Momentum (the Second Stage)

As the vehicle travels and time passes by, the forces F_{attr}, F_{pres} and F_{pret} reduce rapidly, and finally disappear. At this moment, the particle will continue its movement at its current momentum. The forces F_{grv} and F_{air} are the primary forces governing its behavior (acceleration.) We call this stage the *Inertial Momentum* stage. This stage will continue until dust particle's velocity is reduced such that the particle's velocity becomes a small constant.

Every dust particle has a lifetime parameter, which is used to decide when its motion enters from the first stage into the second stage, and from the second to the third. We use the following equation to simulate the dust particle movement at this stage. The initial velocity is calculated from the first stage's values.

$$A_i = \Lambda(m_p, S_p, V_{i-1}) \tag{22}$$

$$V_i = V_{i-1} + A_i \cdot \Delta t \tag{23}$$

$$P_i = P_{i-1} + V_i \cdot \Delta t \tag{24}$$

where Λ is a function to calculate the acceleration from the particle's mass, size, gravity, and velocity. Here we use $\Lambda(m, S, V) = \frac{VS}{m}$.

When A_i becomes very small, we get into the drifting stage, as discussed in the next section.

3.3 Airborne Drift (the Third Stage)

When the total force on a particle becomes very small, the dust particle begins to drift with constant velocity. The forces acting on the particle are balanced. Most of the dust particles will drift with the wind and eventually fall back to the ground. In this stage, the particles stay in the air are those with very small mass quantities. We simply keep the constant velocity with some random disturbances. If a dust particle touches the ground, it is dead. If a dust particle drifts from the range of the particle systems to the outside area, it is considered dead. Dead particles are faded away after a few frames of simulations.

4 Rendering Techniques and Results

4.1 Motion Blur

We use motion blur to achieve better animation. We record every dust particle's several continuous positions. Each dust particle has a head pointer which is the current position, and a tail pointer which is the fading position. A particle is drawn a number of times into the buffer with bigger and bigger fading coefficients. The head is drawn at its current position with the particle's original color, and the tail is drawn at the earliest position with a much dimmer color. We let these smaller particles to have more blurring effect. That is, the smaller the dust particle is, the longer blur process will be. This simple technique seems to make the simulation more realistic. We also use the comprehensive parameter (equation 17) to control the blurring process. Larger value causes particles having more blurring effects. We can modify different parameter for blur according to the simulation applicable for certain application.

4.2 Blending

Dust particles can obscure other particles that are behind them, or they can be transparent and can cast shadows on other dust particles. We uses Reeves's method to deal with this situation. Every particle is treated as a point light source when it is displayed. Each particle adds a bit of light to the pixels that it covers. A particle behind another particle is not obscured but rather adds more light to the pixels covered.

In order to speed up the rendering process, we restrict our rendering area to be the box area shown in Fig. 3. As the vehicle travels ahead, the box area moves ahead the same distance. Any dust particle outside the box area is treated as dead. This method allows us to have a background texture which is not updated. So we only need to calculate and render the dust particles within this box area. This approach reduces the number of dust particles needed and also reduces the memory volume needed.

4.3 Particles

For each calculated particle, we generate a number of particles that have movement similar to this particle, with some random behaviors. This way we only calculate one particle, but a system of particles will behave accordingly. It saves time for calculating all the particles and enhances the richness of the picture at the same time.

4.4 Results

Fig. 10 and Fig. 11 are the simulation results under the same circumstances with different vehicle velocities (30 miles/hour and 60 miles/hour.) The number of particles and the density of the dust are greatly affected by the vehicle's velocities. Fig. 12 and Fig. 13 are the simulation results with different dust densities. There are many other parameter which affect the simulation. In most cases, the simulation looks better with more dust particles, but the simulation is much slower because all the particles must be calculated in the particle systems.

5 Synchronization in DIS

An entity is an object such as a particle system, a boat, a piece of changing terrain, or even a piece of fluid surface simulated in a DIS. Each entity has a local variable (say lastTime) used to record the last time this entity updated its state. So when an entity begins to update its state, it can read the simulator's clock to get the current time (say currentTime) and subtract the currentTime by lastTime to determine the period between the current time and the last time when the state was updated. This is the time slice passed and its value together with the old state uniquely determines the new state. At the same time, lastTime is updated to currentTime. So each entity proceeds at its own pace synchronized by a uniform time scale of our clock. The whole system is synchronized in the sense that, at a certain instance, all entities have advanced approximately the same amount of time. For a simulator with a fast simulation cycle, the time slice tends to be small and the fidelity high; for a simulator with a slow simulation cycle, the time slice tends to be large and the fidelity low. This synchronization ensures that all entities use the same time scale.

When an entity updates its corresponding entities on different hosts, it sends its current state to them and they update their states by the received state and the network delay. So when an entity receives a message, it must be able to compute the network delay between the time when the message was sent and when it is received. When the sender sends a message including the current time information of the sender's clock, the receiver can decide the network delay of the received message by looking at its own clock and the time information sent with the received message. For example, if we know that the time difference between simulator 1 and simulator 2 is exactly 3 seconds (simulator 1 is 3 seconds ahead of simulator 2), then if simulator 2 sends a message to simulator 1 at 12:00:04 pm (of simulator 2's clock) and simulator 1 receives the message at 12:00:03 pm (of simulator 1's clock), we know the network delay is exactly two seconds.

There are two major advantages of this synchronization mechanism. One is that it needs very little network information. The synchronization is not achieved by sending messages across the network. The other is that activities can be predicted and time elapses of events can be recorded. For example, if we know the speed of a vehicle, then we can predict its position ahead of the simulation. Thus, the simulated environments provide a better perception of real world time.

Existing efforts in physically inspired modeling have used different approaches and employed different physical laws (e.g. Newton's second laws, energy equations, etc.) to achieve the modeling and simulation. All these physical laws are functions of time. The numerical simulations of these physical laws include calculating current state from last state and the time passed between last and current states. In a stand-alone simulation, Δt_n is often a chosen constant allowing the same numerical stability for all n's. In a real-time DIS, there are three major activities between states of simulation as shown in Fig. 7: processing (network) messages and other information, calculating next states of the (physical) entities, and rendering the calculated entities graphically. To achieve real-time simulation, all the activities cannot be slower than certain thresholds. Within the thresholds there are problems which need to be solved.

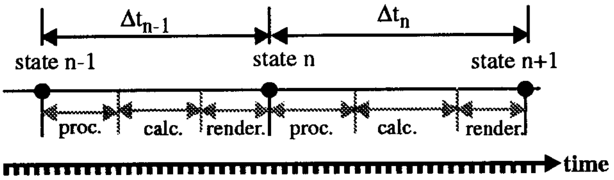

Fig. 7: Activities between states of simulation

The network synchronization method introduced earlier is aimed at synchronizing the network activities as well as minimizing the network trafficking in DIS. However, it could happen that the varying time slices between states are so big that the numerical computation for physically inspired modeling diverges. As we know, in physically inspired modeling and simulation, the time slice must be smaller than a certain criterion to retain numerical stability and to limit the numerical offset error. For example, the numerical calculations of the dust particle system need very small time slices to retain numerical stability. Here we provide a time differential solution for this problem. The idea is that, instead of using a big time interval Δt_n, we differentiate it into smaller time slices satisfying the numerical calculation requirements of the physical equations. For example, if we know that Δt is satisfying these requirements, we still use it to calculate the physical equations regardless of the varying time slice between simulation states, yet we can also synchronize the activities of all physical models in the network. When we get Δt_n which is larger than Δt, we can divide Δt_n into a number of Δt's and calculate the activities of the physical phenomena $\lfloor \Delta t_n/\Delta t \rfloor$ times. The residue of the time division can be added to the next simulation period. If Δt_n is smaller than Δt, we can consider it as a residue time and add it to the next simulation period. This is illustrated in Fig. 8. Notice the difference between the time slice used to calculate the state changes and the time slice between state changes.

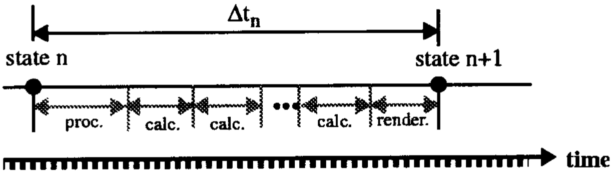

Fig. 8: Physical models need small time slice

In the case where the time slice is small but the physical phenomena evolve so fast that the animation appears jumpy (temporally aliased) after a series of calculations, we can render every frame after each numerical calculation (Fig. 9). This situation occurs when we simulate fast particle movements such as dust behaviors.

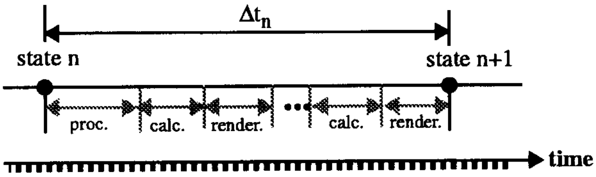

Fig. 9: models need small time slice but evolve fast

6 Conclusion and Future Work

We have introduced our approach to simulating dust behaviors behind a traveling vehicle. We have two primary goals: one is realism of the simulation, the other is real-time computation. In order to achieve realism, we analyze the forces and factors and construct physically inspired empirical models to generate particles and control the dust behaviors accordingly. In order to achieve real-time, we further simplify the numerical calculations by dividing the dust behaviors into three stages, and establishing simplified particle system models for each stages. We employ motion blur, particle blending, texture mapping, and other techniques in computer graphics to achieve better results. Our work is a useful addition to many applications in simulated virtual environments.

Our model is a physically inspired empirical model. Using CFD to calculate the turbulent vortex behind a vehicle would yield an accurate physical model, which could be integrated into our work. The problem is that CFD models are too computationally complex. We have succeeded simulating fluid flows in real-time [Chen97]. We hope to find a solution for dust behaviors also. We plan to further consider the interaction between the dust particles and the environment.

Fig. 10: Vehicle Speed=30, Dust Density=40, BlurN=2

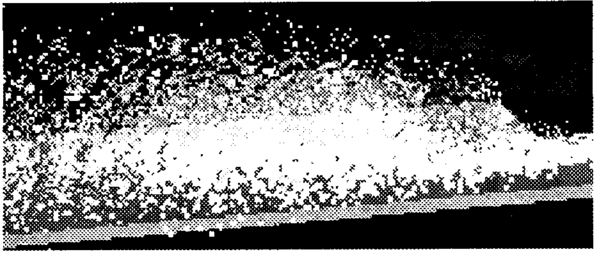

Fig. 11: Vehicle Speed=60, Dust Density=40, BlurN=2

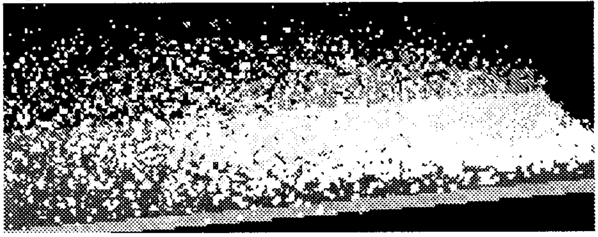

Fig. 12: Vehicle Speed=30, Dust Density=10, BlurN=2

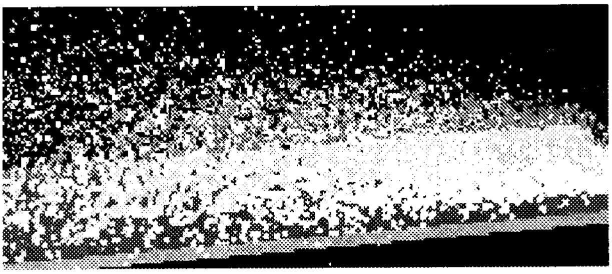

Fig. 13: Vehicle Speed=30, Dust Density=60, BlurN=2

References

[Chen97] J. X. Chen, N. V. Lobo, C. E. Hughes and J. M. Moshell, "Real-time Fluid Simulation in a Networked Virtual Environment," *IEEE Computer Graphics and Applications*, Vol. 17, No. 3, 1997, pp. 52-61.

[Cowherd89] C. Cowherd, M. Grelinger, and P. J. Englehart, "An Apparatus and Methodology for Predicting the Dustiness of Materials," *American Industrial Hygiene Association Journal*, v. 50 (Mar. '89) pp. 123-30.

[Hsu95] S. Hsu and T. Wong, "Simulating dust accumulation," *IEEE Computer Graphics and Applications* v. 15 (Jan. '95) pp. 18-22.

[IST94] IST (Institute for Simulation and Training), *The DIS Vision: A Map to the Future of Distributed Simulation*, version 1, Univ. of Central Florida, Orlando, Florida, May 1994.

[Li93] X. Li and J. M. Moshell, "Modeling Soil: Real-time Dynamic Models for Soil Slippage and Manipulation," *Computer Graphics* Proceedings, Aug. 1993, pp. 361-368.

[Like92] T. Loke, D. Tan, H. Seah, "Rendering Fireworks Displays," *IEEE Computer Graphics and Applications*, May, 1992, pp. 33-43.

[Oppenheimer86] P. E. Oppenheimer, "Real Time Design and Animation of Fractal Plants and Trees," *Computer Graphics*, Vol. 20, No.4, 1986, pp. 55-64.

[Reaves83] W. T. Reeves, "Particle Systems -- a Technique for Modeling a Class of Fuzzy Objects," *ACM Trans. Graphics*, Vol. 2, April, 1983, pp. 91-108.

[Reffye88] P. Reffye, C. Edelin, J. Francon, M. Jaeger, C. Puech, "Plant Models Faithful to Botanical Structure and Development," *Computer Graphics*, Vol.22, No.4, 1988, pp. 151-158.

[Reynolds87] C. W. Reynolds, "Flocks, Herds, and Schools: a Distributed Behavioral Model," *Computer Graphics*, Vol. 21, No. 4, July 1987, pp. 25-35.

[Roth95] M. Roth and R. Guritz, "Visualization of Volcanic Ash Clouds, *IEEE Computer Graphics and Applications*, Vol. 15 (July '95) pp. 34-39.

[Sims90] K. Sims, "Particle Animation and Rending Using Data Parallel Computation," *Computer Graphics*, Vol24, No.4, Aug 1990, pp. 405-413.

[Fiume93] J. Stam and E. Fiume, "Turbulent Wind Fields for Gaseous Phenomena," *Computer Graphics* proceedings, August 1993, pp. 369-376.

[Williams88] R. R. Williams and R. E. Davis, "Battlefield Dust Environment Symposium III," *U.S. Army Engineer Waterways Experiment Station*, Vicksburg, Miss., 1988. (Call No.: UG683 B39s 1988)