International Journal of Control and Automation
Vol. 7, No. 12 (2014), pp. 23-36
http://dx.doi.org/10.14257/ijca.2014.7.12.03

Research on the Flame Simulation Method Based on the Particle System

Xumin Liu, Zilong Duan, Dawei Qian and Xiaojun Wang

College of Information Engineering
Capital Normal University, Beijing 100048, P. R. China
liuxumin@126.com

Abstract

In reality, flame is a common fluid scene. In this paper, we simulated flame by using improved particle system and detail the properties and changes of particles in the model. We realized a realistic color change and dynamic flickering flame characteristics. We used texture mapping method to render the flame and used the hierarchical programming structure to provide a simple and practical interactive flame simulation. We also designed and implemented demonstration system of flame simulation. Experiments show that using the flame image generated in this paper can not only satisfy the visual effect but also meet the real-time requirements.

Keywords: *realistic, particle system, texture mapping, flame simulation*

1. Introduction

In the narrow sense, computer graphics is one kind of science based on the physical laws, empirical methods and cognitive theory. It uses various mathematical algorithms to process two-dimensional or three-dimensional graphics data and generate performance of visualization data. In a broad sense, computer graphics includes not only the processes of three-dimensional graphical modeling, drawing and animation, but also the researches of two-dimensional vector graphics and processing of video image fusion. In 1982, International Organization for Standardization gave the definition: Computer graphics is the methods and technologies which research the mutual transformations between the data and the graphics by using computer. Computer graphics after 40 years of development has entered a more mature stage of development. At present, the applications of computer graphics include computer-aided design and processing [1], film and television animation [2], military simulation [3], medical image processing [4], realistic simulation [5], and so on.

With the rapid development of computer graphics technology, more and more flame burning scene appear in film and television, computer animation, advertising, game development and other areas, which shows the simulated flame essential. Flame the same as other irregular objects (such as smoke, clouds, fog, waves, *etc.*,) has the characteristics of no rules and real-time variability. They look fuzzy and their surface is extremely rough, which can not use the classical Euclidean geometry to model and render [6]. In addition, various flame forms, different chemical composition of combustible and different effect internal and external environmental conditions on the combustion process of burning will have very different flame scenarios, which increase more difficult for research. Thus, how to use the computer to generate the same as real flame burning scenarios is always a difficult problem in computer graphics.

ISSN: 2005-4297 IJCA

The methods of computer flame simulation are generally divided into three types: flame simulation based on particle systems, flame simulation based on mathematics physical model and flame simulation based on texture [7].

This paper is intended to improve on the original particle system. We combine with texture mapping technology to simulate the flame and use circular texture images instead of point particles, which can greatly reduce the number of particles and meet the requirement of real-time. Through the experimental analysis, the method of this paper can satisfy the requirement of real-time and the results of simulation are also more real.

2. Basic Principle of Particle System

Particle system adopts a quite different from the past method of modeling system to simulate flame, cloud and rain, smog and so on. Particle system simulates appearance and randomness of the irregular object mainly through the overall form, features and the dynamic change of a large number of particles. Particle system is the theoretical basis of this paper.

2.1. Basic idea of Particle System

In 1983, W.T. Reeves and others first systematically proposed a method for irregular fuzzy objects (fire, clouds, water, *etc*.,) modeling [8]. Particle system is a common method for computer simulation of fuzzy objects. Any object, no matter it is solid, liquid or gaseous, consists of the simplest particles. This also accords with the laws of physics: all objects are composed by the most basic particles. Therefore, the particle system is to solve the problems: the presence of particles, the law of motion of particles and the force on particles. The basic idea of particle system is that the fuzzy object is regarded as a particle mass which is consisted of particles. And each particle has its own properties such as position, color, shape, size, lifetime, speed, *etc.*, [9]. The physical properties of objects are mainly displayed by changes of particle properties. Particle is a geometric unit, which has its own properties such as shape, size, color, transparency, position and speed. In general particle system, various properties of particles are changing over time, thus the dynamic and randomness of irregular objects are well represented. These changes are divided into the formative stage of particles, the movement stage of particles and the waning stage of particles [10].

We use particle system to generate images of irregular objects in the computer. Each frame needs to finish the following five steps:

1) Remove the dead particles in the system;
2) Generate new particles according to the conditions;
3) Initialize various properties of new particles;
4) Move all particles in the system and change the properties of particles;
5) Render the existing particles in the system.

Generally, various properties of particles can be determined by the different empirical function. It is also given certain random distribution characteristics. Figure 1 is the principle diagram of particle system.

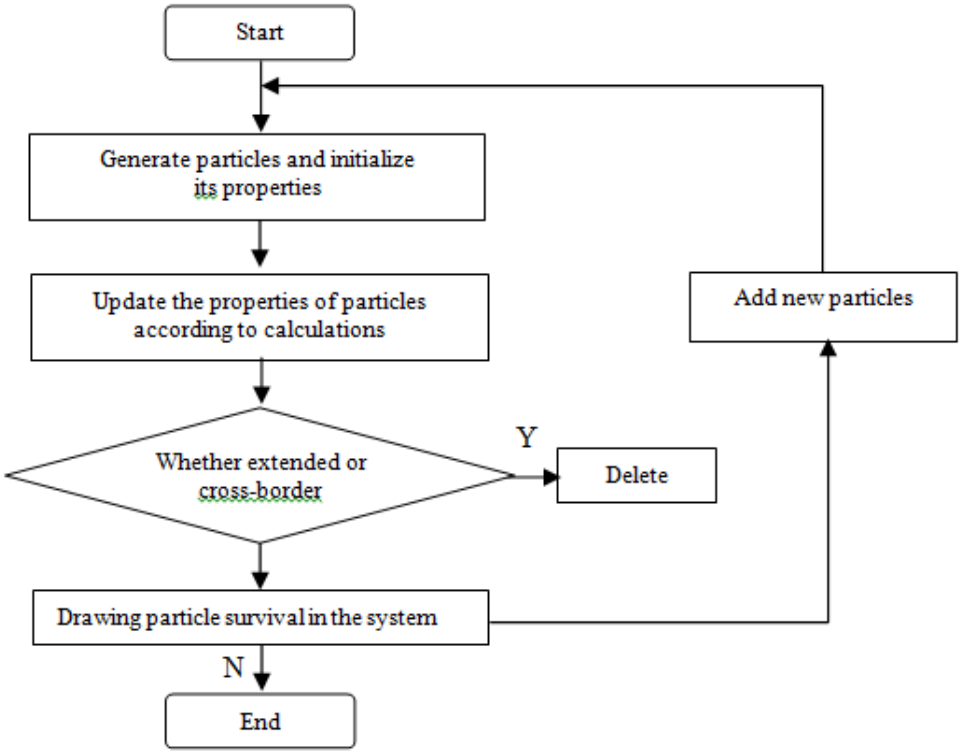


Figure 1. Principle Diagram of Particle System

2.2. Basic Model of Particle System

2.2.1. The Formal Description of Particle System

The definition of particle system model has a variety of different methods and the most two typical definitions are as follows:

Definition 1: The particle is defined as an n-vector $R_n\ (n \in I)$ in the real number field. It can be represented as follow:

$$R_n = \{A_1, A_2, L, A_i, L, ..., A_n \mid n \geq 3, n \in I\} \quad (1)$$

A_i Represents the state or attribute of particles. It usually includes position, velocity, acceleration, color, transparency, life cycle, etc.

Definition 2: Particle system is a set of particles in the definition 1. Each particle of the set has an index which represents the map from I to R_n :

$$P = \{I \to R_n \mid n \geq 3, n \in I\} \quad (2)$$

$P(i) = R_n$ Represents the state or attribute of particles whose index is i. The set of particles represents the states of the particle system of certain moment as a whole.

2.2.2. The Generation of Particles

The generated areas of particles are different based on the different simulation object of particle system. In the fireworks particle system, the detonation point of fireworks in space is the generation of particles; in the waterfall particle system, the source of the waterfall is the generation of particles; in the rain and snow particle system, the

particles are produced in the sky or the screen; in the flame particle system, particles are produced on the fuel surface. Particles can be generated in any region of the system space as needed.

The number of particles in the particle system is a critical value. The number of particles which are generated in each frame will directly affect the density of simulation object. Generally, the determination of generation number of particles in the particle system mainly has two kinds of methods [8]:

1) Measure the average quantity of particles

At first, give each animation a certain average value and random variation range of the generated new particles. After that, calculate the actual number to generated new particles at this moment:

$$NewCount = MeanCount + Rand() \times VarCount \quad (3)$$

Rand () is a uniform distribution random function in [-1,+1].

2) Measure the object area

The method of measuring the object area is that the number of generated particles depends on the size of the display screen. The average number of particles and random variation range are the mean and variance of particles which is generated in the unit area of the display screen:

$$NewCount = \left(MeanCount + Rand() \times VarCount \right) \times ScreenArea \quad (4)$$

Screen Area represents the projection of the current simulation objects on the screen.

In the flame simulation, if you want to generate some particles in certain screen, the initial value is critical. It directly determines the density and size of the flame. The number of particles is too large to increase system processing time and it can not satisfy the requirement of real-time. The number of particles is too small to meet the requirements of realism. In addition, the number of particles is inversely proportional to the size of the particles. If individual particle is large, the number of particles is fewer. If individual particle is small, the number of particles is more.

2.2.3. The Properties and Initialization of Particles

Each particle in the system will be given a certain property. The particle's properties directly lead to the state of the particle. The particle's properties are different based on the different of simulation object. Usually, the simple model, which mainly includes spatial properties, appearance attributes, sports attribute and life attribute, is established on the basis of meeting the visual effect.

1) Color of particles

Color is one of the most important features in flame display. In reality, the color of flame depends on the burning of chemical elements. If the proportion of various elements is different, the effect of flame simulation will be different. When the computer simulates flame object, we use red, green and blue the three variables to describe the basic color of the particles. The color of the particles will changes over time, until it turns black. And it will not work for the scene. Then, we delete it from the list and release the memory space occupied by particles.

2) Shape of particles

The shapes of the flame are mainly controlled by the generated areas of particles. They mainly include sphere, cylindricality, *etc.*, To the particles of different shapes, the results of simulation are also different. In order to make the simulation effect irregular, we introduce some randomness when we assign the initial position and velocity of specified particles.

3) Size of particles

The size and the shape of particles are determined at the same time. The bigger individual particle, the worse the generated flame realism is. But, the calculation is small and the real-time of the system is high. On the contrary, if the size of the particles is too small, the reality of flame will be strong. And the calculation will be particularly large, thus it can't meet the real-time requirements. In the process of specific implementation, the size of the particles needs compromise between reality and real time of the image.

4) Position of particles

The initial position of particles is determined by the area of particles' generation. The specific location of particles in space is determined by the three-dimensional space coordinate of particles. In every frame we need to calculate the particles' new position. The particle's new position is determined by the position and velocity of particles in the last frame.

5) Velocity and acceleration of particles

Particles are moving in the air. Properties of particles mainly include velocity and acceleration. The velocity represents the quantity of particles' movement speed in the system. The faster particles' velocity is, the more drastically flame burn. The changes of particles' velocity are determined by the acceleration of particles. The acceleration of a particle depends on external force of particles.

6) Lifecycle of particles

The new particles are given lifecycle by system when they are generated. The particle's lifecycle is used to represent the particles' survival time in the system. As a function of time, the lifecycle of particles is reducing until the lifecycle is zero and the particle is disappear from the system. Generally lifecycle is represented by the number of frames in the system.

7) Transparency of particles

In the actual flame simulation, due to particles is burning continually; they will eventually fade and disappear gradually. This kind of phenomenon is realized by managing the transparency of the particle. When a particle is generated, it has an initial transparency. In its lifecycle, the transparency will be changing continually until the particle disappears. Therefore we can get realistic simulation results.

3. Flame Simulation

Based on the simulation of natural phenomenon of flame burning, this paper raises the method of flame simulation based on the theory of particle system.

3.1. The Generation of Flame

We regard that flame is composed of a large number of small particles. Set the particle as F. When F is mainly affected by gravity, wind and air buoyancy, there will be corresponding changes of speed, acceleration, location, quality, *etc.*,

Set properties of the known F as follows:

Initial mass: m

Initial spatial location: $P_0 = \{x_0, y_0, z_0\}$

Initial velocity vector: $V_0 = \{v_x, v_y, v_z\}$

Initial acceleration vector: $a = \{a_x, a_y, a_z\}$

Lifetime: lifespan

Age: age

Color: color

Transparency: alpha

Particles' generation is controlled by random function. Referring to the properties of every particle, its parameters determine the range. And in this range, its properties value is

determined randomly. The range depends on the given average expectations and maximum variance. The basic expression for the amount of particles: $n_p(f_i) = m_p(f_i) + rand(f_i) \times v_p(f_i)$. In this formula, rand () is random function uniformly distributed in [-1, 1], $m_p(f_i)$ and $v_p(f_i)$ are the No. f_i Frame's mean and variance of the new generation amount of particles. Generally, variance is defined as pre-given constant. Mean $m_p(f_i)$ can use relatively simple linear function L: $m_p(f_i) = m_p(f_0) + \Delta m \times (f_i - f_0)$. In this formula, f0 is the frame number activated by the particle system. $m_p(f_0)$ is the No. f_0-frame's average number of new particles, Δm is the particle rate of change (variable or constant).

Settings of particles' initial properties:

Initial velocity $v(f_0)$ = Average velocity $v_0 \times rand() \times$ Velocity variance v_v.

Initial color $c(f_0)$ =Average color $c \times rand() \times$ Color variance c_v.

Initial transparency $t(f_0)$ =Average transparency $t \times rand() \times$ Transparency variance t_v.

Lifetime $l(f_0)$ = Average lifetime $l \times rand() \times$ Lifetime variance l_v.

Age and lifetime is a pair of complementary value. When the lifetime reduces, the age will increase.

The location of the flame particles generally generated in the vicinity of the combustion point. Usually the initial position of the flame particles is jointly determined by the flame ignition and the location offset. Generally, the burning regional center of the flame is brighter and the edge is darker. Sparse level of particles expresses this feature when the particle system simulates flame. Generally, a normal distribution determines the initial position of the flame [11]. We use coordinate (x_0, y_0, z_0) to represent the center position of the burning flame. We use R_i to represent the radius of the flame burning. The initial position of the flame particles is $(X_i(t_0), Y_i(t_0), Z_i(t_0))$. The spatial distribution of the particles is:

$$\begin{cases} X_i(t_0) = \frac{1}{\sqrt{2\pi}R} \exp\left\{-\frac{(x_i - x_0)}{2R^2}\right\} \\ Y_i(t_0) = \frac{1}{\sqrt{2\pi}R} \exp\left\{-\frac{(y_i - y_0)}{2R^2}\right\} \\ Z_i(t_0) = \frac{1}{\sqrt{2\pi}R} \exp\left\{-\frac{(z_i - z_0)}{2R^2}\right\} \end{cases} \quad (5)$$

In this formula, $R = 1/3R_i, R_i$ represents the radius of the flame burning.

Generally, the initial location and size of particles and the initial direction of motion are determined by the shape of the particles' emitter. Common basic shapes are spherical, flat, rectangular and fractal attractors, etc. In this paper, the spherical shape is adopted as a particle emitter [12].

3.2. Changes of Particles' Properties

With the production of each frame image, the number and the properties of particles must be updated. The following will discuss the update mode of particle attributes.

3.2.1. Changes of Position

Once particles have initial speed, they will comply with the principles of dynamics and move in three-dimensional space. At time t, the particles' position in the space is determined by (6)-(8).

$$x = x_0 + v_x \times t \quad (6)$$

$$y = y_0 + v_y \times t \quad (7)$$

$$z = z_0 + v_z \times t \quad (8)$$

In these formulas, (x_0, y_0, z_0) is the position of the particle at time t, $v = (v_x, v_y, v_z)$ is particle velocity at time t.

3.2.2. Changes of Velocity and Acceleration

Changes of the particles' velocity also meet Newton's laws of motion: $v = v + a \times t$. In order to increase the flexibility of the particle system, we introduce the field of air velocity $v(x,t)$. At time t particles can simulate the phenomenon of flame's dance according to the movement of air velocity. Function $v(x,t)$ should reflect the various factors affecting flame's dance. It should change with space and time and reflect the intermittent gusts, strong hurricanes and other factors.

Particles' acceleration is mainly determined by forces on the particles. These forces typically include gravity, buoyancy, wind, etc. To reduce the computational complexity and increase the image frame rate, particles' acceleration is obtained by a random function which conforms to hypo dispersion and its range is [-1, 1]. At the beginning stage of flame burning, its acceleration is 0 and reaches the maximum in the mid-burning. At the later stage of burning, the acceleration is less than 0.

3.2.3. Changes of Color and Transparency

The color and burning intensity of flame relate to the properties of the combustible material. Flame's color is usually yellow, red and green. The color of different flame's parts is slightly different. In the particle system, because the particles are generated more in the center of the flame, the flame is bright white by the mixed color of multiple particles in the flame core. The particles of outer flame parts gradually reduce. The flame color gradually turns into bright yellow, orange, and finally edge becomes red, black. This mixing process also conforms to the feature of flame changes in color.

Transparency decreases with the decrease of lifetime. When lifetime is zero, transparency also reduces to zero, that is completely transparent and achieves integration with the background.

3.2.4. Changes of Lifetime

Flame particles' lifetime represents its survival time on the screen, which is represented by number of frames. When the velocity is certain, it determines the maximum height of particle. If the average height of the flame is h_0 and the average survival period of the particle is $m_l = h / v_y$, and v_l is the particle's lifetime variance, the lifetime of the particle is determined by the following formula:

$$l(i) = m_l + rand() \times v_l \quad (9)$$

As we all know, the flame's height in the zone of flame center is generally higher than others in nature. According to this, we propose that we extend the life cycle of the flame particles in flame center by disturbing the location factor determined by the initial position of lifecycle. For the circular particle emitter which is parallel to the plane in the world coordinate system XOZ, the location factor p is defined as follows:

$$p = \begin{cases} a, \sqrt{x^2 + y^2 + z^2} < r \\ 0, \sqrt{x^2 + y^2 + z^2} \geq r \end{cases} \quad (10)$$

In this formula, a>0， r is pre-control value, and 0<r<R. After the disturbance of the location factor p, the particles' new lifecycle is: L= (1+k) l.

With the increase of time, the particle's life span will be less and less until to 0 and particles disappear.

3.3. The Hierarchy Structure of the Particle System

The accomplished structure of flame particle system is shown in Figure 2. The core of this system is the flame rendering engine, which receives the environmental control information. The particle's control mechanism controls changes of particle's motion. The particle's survival mechanism controls the particle's newborn and dying in the particle field [13].

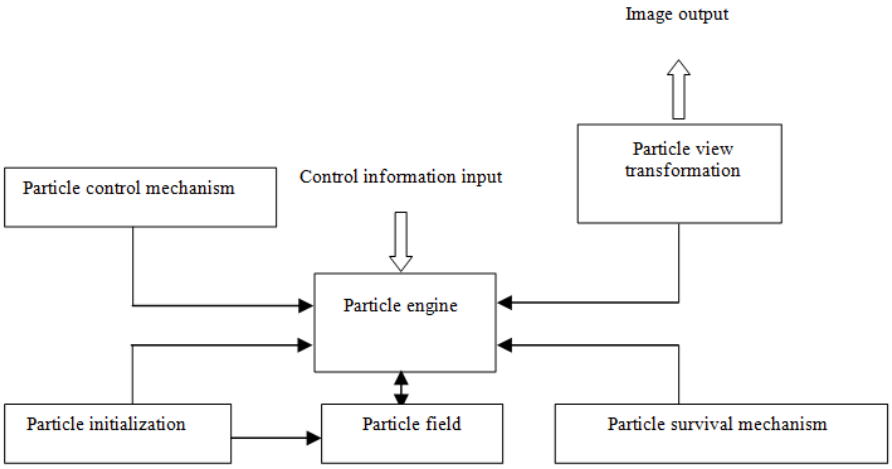


Figure 2. Structure of the Flame Particle System

The implementation steps of particle system are as follows:
(1) Generate the initial flame particle set in the ignition
(2) Give the properties of the particle flame
(3) Draw the initial flame
(4) For each frame
(5) Re-select the set of flame particles
(6) Particles' motion, transformation in the particle field
(7) The particle's lifetime minus 1
(8) Some of the flame particles die out
(9) Generate new flame particles
(10) Draw the flame particle set

The fidelity of the flame generated by the flame particle system depends on the settings of the particle properties.

4. Draw the Flame

Once the position and attributes of flame particles are determined, we can draw the flame particles. The drawing of flame particle is a process that we make the simulation results into graphic information and export to the screen finally.

This paper adopts the method of surface particles' drawing. Namely the particles are constructed to very small patches which can reflect the light source of polygon. Thus, each

patch can replace the original large particles, and it can greatly reduce the number of flame particles required for each frame and increase the frame frequency of system simulation.

To enhance the realism of flame, we use two techniques in the rendering process: texture mapping and eye tracking.

Texture mapping is a simple and effective means to enhance the realism [14]. Texture is mainly divided into 2d and 3d texture. This paper uses 2d texture to simulate the flame. Its essence is a mapping form the plane of 2d texture to the surface of 3d objects. In order to put the pattern of 2d texture to map to the surface of 3d object, the corresponding relations between the object space coordinates and texture space coordinates must be established. It is the equivalent to parameterization of object's surfaces. Then map from object space coordinates to screen space projection. Generally the two mapping transformations can be combined to one mapping transformation.

We suppose that texture pattern is defined on the coordinate of texture space: (u,v). Object surface is defined on the coordinate of scene space: (x,y,z). The corresponding parameter equation can be indicated as follows:

$$\begin{cases} x = x(\theta, \phi) \\ y = y(\theta, \phi) \\ z = z(\theta, \phi) \end{cases} \tag{11}$$

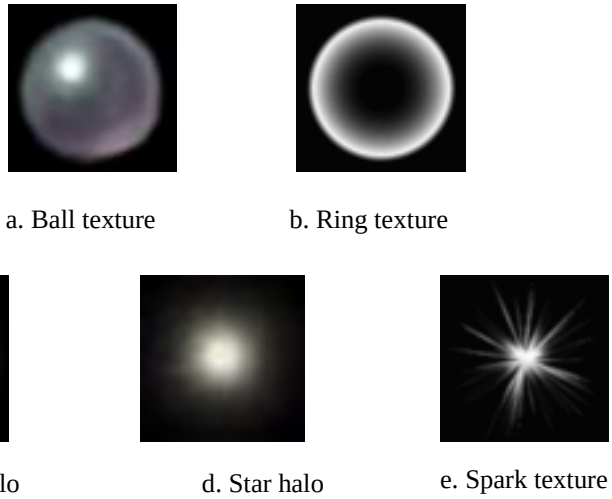
Adding texture on the hook face will refer to a mapping function between them. The mapping function from the texture space to the parameter space is as follow:

$$\begin{cases} \theta = f(u, v) \\ \phi = g(u, v) \end{cases} \tag{12}$$

The formula (12) shows a mapping change from the texture space to the parameter space. Generally speaking, f and g are reversible, so the inverse mapping from any cell on the hook face to the texture space is as follow:

$$\begin{cases} u = h(\theta, \phi) \\ v = i(\theta, \phi) \end{cases} \tag{13}$$

Usually flame particles are represented by geometric primitives, which own the size, shape and color, such as circular elements. Radius of the circle can be set on demands. In order to make each particle like one small Mars, we can use texture mapping techniques. We paste the texture map for each particle. With the process of color blending, it can generate a more realistic model of flame burning. Different texture patterns can produce different flame effects. Figure 3 is some kinds of textures. Different textures can generate different effects of simulation.



a. Ball texture b. Ring texture

c. Round halo d. Star halo e. Spark texture

Figure 3. Texture Image

In order to achieve three-dimensional flame simulation, this paper uses eye tracking strategy. The particles move in three-dimensional space and there locations are also three-dimensional distribution. However the texture mapping techniques are usually two-dimensional. So when flame or point of view move in a scene, there may be a distortion phenomenon due to the mutual angle. When the line of sight is parallel to the texture mapping plane, we even can't see the presence of flame. Though, some OpenGL implementations have provided support for 3-d texture. And they can solve this problem. But only a few high-end graphics cards provide hardware support for 3-d texture mapping. Due to a large amount of computation, the method that using software rendering mode to conduct 3-d texture mapping in the usual graphics cards is very time-consuming and impractical. Eye tracking technology [15] uses 2-d texture mapping to make 3-d flame simulation. This method is simple and the calculation is small, and works well. The technology's principle is that getting the sight vector by calculating the relative position of point of view and flame particles. Then make the particle texture mapping a certain rotation in order to make it perpendicular to the sight vector. As a result, no matter how flame or point of view move, the particle's rendering plane is always facing the sight. It also achieves a 3-d simulation by eliminating the flame's distortion phenomenon caused by the changes in relative perspective. Let p_e and p_f separately represent the point of view and the location of flame particles. b represents the unit normal vector that flame particle rendering plane at present. The angle which should be turned by the flame particle rendering plane is as follows:

$$\alpha_y = \perp_{XZ}(e) - \perp_{XZ}(b) \quad (14)$$

$$\alpha_x = \perp_{YZ}(e) - \perp_{YZ}(b) \quad (15)$$

In this formula, $e = {pe-pf}/{\|pe-pf\|}$ represents the unit sight vector, that is the unit normal vector of flame particle rendering plane after rotation; α_y and α_x respectively represent the angle that rendering plane around the X axis and Y axis rotation; $\perp$ is the projection function, $\perp_{YZ}(e)$ represents the projected vector of the vector e on the plane YOZ[16-17].

The basic steps of algorithm generated every frame image of flame is as follows:

1) Particle system produces new flame particles and motion field.

2) Assign each new flame particles and motion field certain initial values. The initial spatial location of new flame particles meets Gaussian distribution function.

3) Check the flame particles which are beyond the lifecycle or the scope of screen display in the system and the death motion field and delete them.

4) Update the position of the motion field, velocity and other attributes according to the laws of setting.

5) Update the size, color and transparency of flame according to the laws of setting. Update the particles' position and velocity according to the laws of setting.

6) Draw images of the living flame particles in the system. The motion field has no need to drawn because it is invisible for observers.

5. Experimental Results

This system is built on VC + + platform. We used OpenGL graphics library to design and develop. The CPU is Pentium Dual Core 1.86 GHz and the memory is 2 GB DDR2 667, and graphics card is ATi3470 with 256 MB memory. From the generated image, we can see that the generated dynamic flame is realistic with real visual effects, and at least 20 frame pictures are generated per second, which meets the requirements of real-time. From the results of flame simulation, we can see under a certain number of particles the key to affect the sense of reality is the color and texture. Therefore, at the time of switching between different states of the flame, we can achieve the effect by changing the required state of particles for the color and texture timely. Figure 4 shows the effects of flame simulation of different texture image simulation. From the simulation results, the texture c has the best effect and is the most realistic.

This paper's algorithm increases the speed of the system from several aspects:

1) On the basis of graphics rendering realistic, we use polygon patch instead of dot particles to reduce the calculation and memory footprint and improve the efficiency.

2) Allocated memory reasonably. Due to the system need add and remove particles continually, memory allocation structure and efficiency will greatly affect the speed of system running. This paper uses the chain table structure to manage the allocation and collection of memory to ensure the efficiency.

3) Simplify the movement model of the flame and introduce the idea of movement velocity field. It not only simulates the movement of the flame but also ensures the running speed of system.

Experiments show that the algorithm of flame simulation in this paper can produce at least 30 frames per second in the ordinary PC and reaches the requirement of real-time.

6. Conclusion and Outlook

This paper makes improvements on the traditional method of the particle system. The particle is defined as the round face represented by a triangle inscribed circle. We describe and design the initial properties of the particle setting, changes of lifetime, and changes in particle motion to enhance the effect of overall realism; by combining the particle system with the texture mapping to simulate the flame, we implement the programming on the computer. Experiments show that simulated flame has greater real-time, adaptability, flexibility, and can achieve a better realistic effect. Moreover, this method can not only be used to simulate the flame but also be applied in the rain, fountains, smoke, explosions and other special effects simulation. We will go on further study and hope to gain more real-time and more realistic simulation effects.

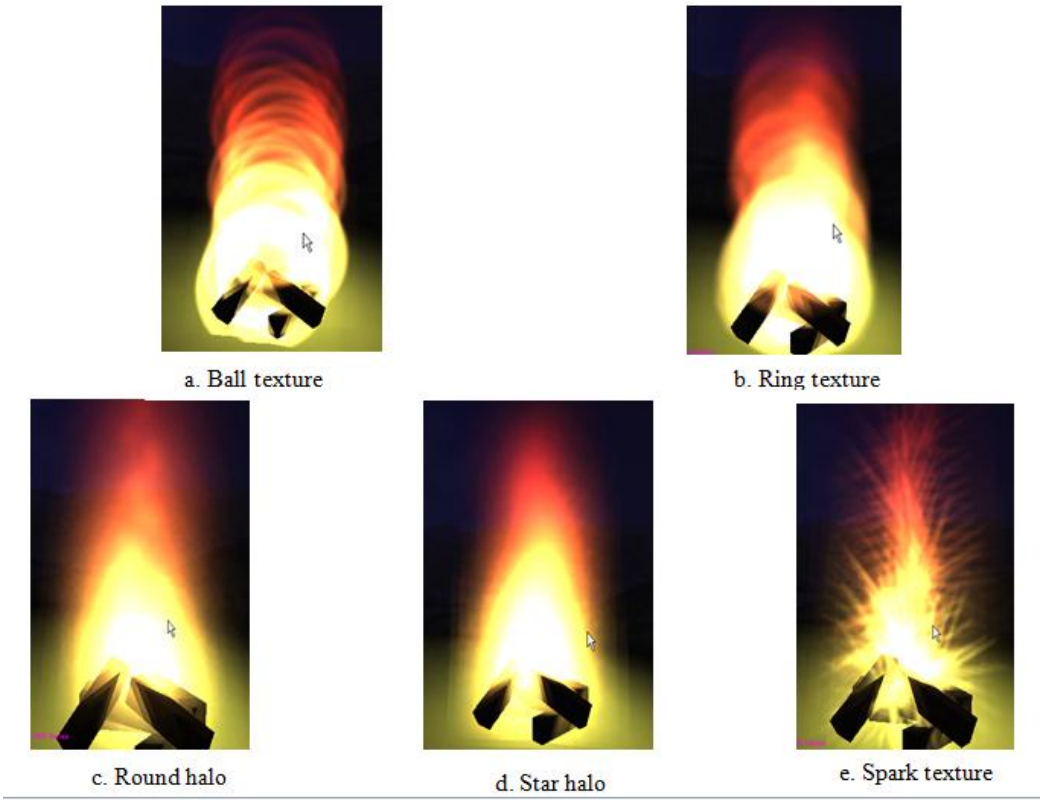


Figure 4. Flame Effect of Different Texture Image Simulation

Acknowledgments

This research was supported by the National Natural Science Foundation of China General Projects Grant No. 61272029.

References

[1] G. Xu, "Consistent approximations of several geometric differential operators and their convergence", Applied Numerical Mathematics, vol. 1, no. 69, **(2013),** pp. 1-12.
[2] A. Bousseau, J. P. O'shea, F. Durand, *et al.*, "Gloss perception in painterly and cartoon rendering", ACM Transactions on Graphics (TOG), vol. 2, no. 32, **(2013),** pp. 18.
[3] H. N. Wu, Y. Liu, G. Yang, *et al.*, "Research of Battlefield Environment Virtual Simulation", Applied Mechanics and Materials, vol. 1, no. 347, **(2013),** pp. 3204-3207.
[4] K. K. L. Wong, Z. Sun, J. Tu, *et al.*, "Medical image diagnostics based on computer-aided flow analysis using magnetic resonance images", Computerized Medical Imaging and Graphics, vol. 7, no. 36, **(2012),** pp. 527-541.
[5] S. Chan, F. Conti, K. Salisbury, *et al.*, "Virtual reality simulation in neurosurgery: technologies and evolution", Neurosurgery, vol. 1, no. 72, **(2013),** pp. 154-164.
[6] Z. Huang, G. Gong and L. Han, "Physically-based modeling, simulation and rendering of fire for computer animation", Multimedia Tools and Applications, **(2012),** pp. 1-27.
[7] W. Zhigang, C. Heping and L. Xinxiong, "Flame simulation based on particle system and texture mapping", Journal of engineering graphics, vol. 4, no. 23, **(2002),** pp. 49-53.
[8] T. William and L. Reeves, "Particle System-a technique for modeling a class of fuzzy object", ACM Transactions on Graphics, vol. 1, no. 3, **(1983),** pp. 91-108.
[9] Z. Fu, "Rummukainen-Gottlieb formula on a two-particle system with different masses", Physical Review D, vol. 1, no. 85, **(2012),** pp. 014506.

[10] T. Ilmonen and J. Kontkanen, "The second order particle system", Journal of WSCG, vol. 1, no. 11, **(2003)**, pp. 1213-1218, 6972.
[11] Z. Qin, W. Huizhong and X. Juanyi, "Study of particle system based flame modeling and realization", Journal of computer-aided design& computer graphics, vol. 1, no. 13, **(2001)**, pp. 78-82.
[12] D. Q. Nguyen, R. Fedkiw and H. W. Jensen, "Physically based modeling and animation of fire", ACM Transactions on Graphics, vol. 3, no. 21, **(2002),** pp. 721-728.
[13] S. Somasekaran, "Using Particle Systems to Simulate Real-time Fire [DB/OL]", **(2005),** pp. 09-25, http://undergraduate.csse.uwa.edu.au/year4/Current/Students/Files/2005/SanandananSomasekaran/Corrected Dissertation.pdf.
[14] X. Wei, W. Li, K. Mueller, *et al.*, "Simulating fire with texture splats", IEEE Visualization, **(2002),** pp. 227-234, Boston, Massachusetts, USA.
[15] F. Y. Kuo, C. W. Hsu and R. F. Day, "An exploratory study of cognitive effort involved in decision under Framing—an application of the eye-tracking technology", Decision Support Systems, vol. 1, no. 48, **(2009)**, pp. 81-91.
[16] K. J. H. Giesbertz, O. V. Gritsenko and E. J. Baerends, "Response calculations based on an independent particle system with the exact one-particle density matrix: Excitation energies", The Journal of chemical physics, vol. 9, no. 136, **(2012),** 094104.
[17] Z. Chunxia, Z. Yan and Z. Shouyi, "3-dimensional fire simulation based on particle system", Computer engineering and applications, vol. 28, no. 40, **(2004),** pp. 73-75.

