Applied Soft Computing 9 (2009) 1170–1176

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Controlling the movement of crowds in computer graphics by using the mechanism of particle swarm optimization

Ying-ping Chen *, Ying-yin Lin

Department of Computer Science, National Chiao Tung University, 1001 Ta Hsueh Road, HsinChu City 300, Taiwan

ARTICLE INFO

Article history:
Received 21 September 2007
Received in revised form 14 September 2008
Accepted 15 March 2009
Available online 25 March 2009

Keywords:
Crowd control
Moving path generation
Computer graphics
Particle swarm optimization
Adaptive behavior

ABSTRACT

This paper presents a uniform conceptual model to co-operate with particle swarm optimization (PSO) for controlling the movement of crowds in computer graphics. According to the PSO mechanism, each particle in the swarm adopts the information to automatically find a path from the initial position to the optimum. However, PSO aims to obtain the optimal solution instead of the searching path, while the purpose of this work concentrates on the control of the crowd movement, which is composed of the generated searching paths of particles. Hence, in order to generate seemingly natural, appropriate paths of people in a crowd, we propose a model to work with the computational facilities provided in PSO. Compared to related approaches previously presented in the literature, the proposed model is simple, uniform, and easy to implement. The results of the conducted simulations demonstrate that the coupling of PSO and the proposed technique can generate appropriate non-deterministic, non-colliding paths for the use in computer graphics for several different scenarios, including static and dynamic obstacles, moving targets, and multiple crowds.

1. Introduction

Nowadays virtual crowds created by using the technologies of computer graphics can be frequently seen in games, movies, commercials, and the like. Creating computer animations for crowds is largely applied to movies and games but in fact, it is not an easy task to create virtual human beings behaving like real human beings. Many research fields and sophisticated techniques are involved to achieve acceptable results. In order to create high quality virtual human beings or animals, at least three facets must be taken into consideration [1]: The first one is appearance modeling. Lots of computer graphic techniques have been developed to create a vivid human including the shapes of face and body, skin textures, hairstyle, and clothes. Appearance largely affects people to judge how much the computer creation is similar to a real person. Then, the second facet is to produce realistic, smooth, and flexible motions in possible situations. Most existing methods for creating motions are parameter-based models with numerous parameters for controlling the motions. It is difficult to have a single flexible, versatile model which can fit in all scenarios. Finally, realistic high-level behavioral actions have to be generated for the virtual human being. It is undoubtedly an extremely difficult problem because defining what kinds of behaviors are human itself is worth a philosophical debate. To resolve the issue technically, many artificial intelligence and agent-based techniques are adopted to achieve the goal, while the techniques are still being developed and improved.

Particle swarm optimization (PSO) [2] is an optimization paradigm proposed in the field of evolutionary computation for searching the problem domain and reaching the global optimum. The concept of PSO is easy to comprehend, and the mechanism is easy to implement. The ability of PSO to reach the position of the optimum creates the possibility to automatically generate non-deterministic paths of virtual human beings from one specified position to another. In this study, we focus on creating a realistic smooth and flexible moving pattern for virtual human beings by utilizing the computational facilities offered by PSO. Particularly, we present a uniform conceptual model to co-operate with PSO to simulate the movement of all the persons in a crowd based on the analogy between a particle swarm and a human crowd. By "uniform," we mean that functions of a single family is used to describe all kinds of objects in the simulation system, including the target, static obstacles, dynamic obstacles, as well as persons. A person can be considered as a particle, which would like to find a way to reach the best solution. The proposed model can be used in several different scenarios, including static obstacles, moving targets, and multiple crowds. The figures as well as the video clips for the simulations in those scenarios are presented.

* Corresponding author. Tel.: +886 3 5712121x31446; fax: +886 3 6126520.
E-mail addresses: ypchen@nclab.tw (Y.-p. Chen), yylin@nclab.tw (Y.-y. Lin).

1568-4946/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2009.03.004

The remainder of this paper is organized as follows. Section 2 describes the related studies on the control of crowds in computer graphics to give the readers some background information. Section 3 briefly introduces swarm intelligence and the methodology of particle swarm optimization. Section 4 proposes the key idea as well as the framework to utilize PSO for controlling the movement of crowds. Section 5 demonstrates the simulation results in several common scenarios. Finally, Section 6 concludes this paper.

2. Related work

Collective behavior had been studied for a long time in many research domains but was applied to computer graphics and computer simulation only recently. In the field of computer graphics, Reynolds [3,4] created a distributed behavior model to simulate the aggregate motion of the flock. Bouvier and Guilloteau [5] presented an application specifically oriented to the visualization of urban space dedicated to transportation. Brogan and Hodgins [6] described an algorithm for controlling the movements of creatures that travel as a group. Still [7] developed a model to simulate the crowd as an emergent phenomenon using simulated annealing and mobile cellular automata. Helbing et al. [8] used a model of pedestrian behavior to investigate the mechanisms of panic and jamming by uncoordinated motion in crowds.

Moreover, there are many studies on the realistic and real-time performance for crowd control. Aubel and Thalmann [9] used a multi-layered approach to handle muscles of varying shape, size, and characteristics and does not break in extreme skeleton poses. Tecchia and Chrysanthou [10] showed a real-time visualization system based on image-based rendering techniques for densely populated urban environments. Aubel et al. [11] presented a hardware-independent technique that improves the display rate of animated characters by acting on the sole geometric and rendering information. Ulicny and Thalmann [12] defined a modular behavioral architecture of a multi-agent system allowing autonomous and scripted behavior of agents supporting variety. Treuille et al. [13] presented a real-time crowd model based on continuum dynamics. Stylianou and Chrysanthou [14] used a flow grid to measure flow over an area and navigate the crowd.

Although there are many approaches for controlling the movement of crowds in computer graphics, only a few researchers try to use evolutionary algorithms for this purpose. Kwong and Jacob [15] presented breeding experiments of dynamic swarm behavior patterns using an interactive evolutionary algorithm. Kim and Shin [16] incorporated several specifically designed mechanisms into the conventional particle swarm optimization methodology for simulating decentralized swarm agents. Instead of modifying the conventional PSO and designing different mechanisms for different issues, in this paper, we propose a conceptual model to work with PSO for creating a stochastic, non-deterministic, non-colliding path for each agent with a uniform approach. With the proposed framework, there is no need for sophisticated mathematical models and complicated algorithmic implementations. Generating paths for each person in the crowd can be flexible and easy to control.

3. Particle swarm optimization

Swarm intelligence is a concept and a methodology used in artificial intelligence, possibly first proposed by Beni and Wang [17] in 1989. It studies the collective behaviors of agents interacting in the environment. There is no centralized control to manage the agents, but all agents or some of the agents, depending on the adopted neighborhood structure or other equivalent algorithmic design, exchange their information to cooperate and emerge group behaviors. Many swarm intelligence systems are inspired by nature, including ant colonies, bird flocking, and fish schooling. They have been adopted in a lot of research fields, such as ant colony optimization (ACO) [18], stochastic diffusion search (SDS) [19], PSO [2], and the like. For many hard, real-world problems, methods developed based on the idea of swarm intelligence can deliver acceptable or good results. Among these swarm intelligence systems, PSO models a solution as a flying particle on a hyperplane and conducts continual movements in the search space.

PSO was proposed by Kennedy and Eberhart [2] in 1995, inspired by the social behavior of bird flocking or fish schooling. PSO is a population-based optimization method and considers each potential solution as a particle. In a D-dimensional problem, a particle is represented as

$$x = [x_1, x_2, \ldots, x_D]^T.$$

Each particle has a position, a velocity, and an objective value determined by the objective function. It uses the experiential and social metaphor to move toward the expected and currently known best solution. The velocity is varied according to Eq. (1):

$$v_i(t+1) = \omega * v_i(t) + c_1 * r_1 * (p_{BLS} - p_i(t)) + c_2 * r_2 * (p_{BGS} - p_i(t)), \quad (1)$$

where v_i and p_i are the velocity and position of the ith particle. ω is the weight for the previous velocity. p_{BLS} is the best position where this particle had been, and p_{BGS} is the overall global best position ever achieved by the swarm. c_1 and c_2 are the cognitive and social parameters, controlling the level of influence of p_{BLS} and p_{BGS} to make different movements. r_1 and r_2 are random numbers uniformly distributed in $[0, 1]$. The stochastic scheme makes the velocity more diverse.

After computing the velocity, the position can then be updated according to Eq. (2):

$$p_i(t+1) = p_i(t) + v_i(t+1). \quad (2)$$

For every iteration, each particle updates its velocity and position. The new position is evaluated by the given objective function, and an objective value is assigned to the particle accordingly. Based on the objective value, p_{BLS} and p_{BGS} might be updated and have influence in the next iteration. Because the fundamental concept is quite similar to that of the movement of pedestrians, in this study, we would like to simulate a crowd with the optimization process in which particles move toward the expected and currently known best solutions.

Although PSO does possess certain characteristics of the crowd behavior, it is still incompatible with the use for controlling the crowd movement in computer graphics. Firstly, the particle in PSO is absolutely free to fly everywhere in the given multidimensional space, i.e., the search space. However, the given environment for a crowd may have obstacles, and the pedestrians in the crowd must avoid collisions, including the collision with the given obstacles and the collision with their fellow pedestrians, where other pedestrians can be considered as dynamic obstacles. These dynamic obstacles are not predictable and may appear and disappear in the environment at any moment. Moreover, it is important to make a virtual pedestrian to walk smoothly and naturally, instead of just oscillate uncertainly and strangely. The walking path must be reasonable and appropriate. For these essential reasons, we propose a uniform model to work with the original PSO for path creation in the next section.

4. The proposed framework

In the light of the analogy between swarms and crowds, we may consider a person as a particle and a group as a swarm. To resolve the aforementioned incompatibility issues, we propose a framework to

co-operate with PSO such that the movement of particles can be made similar to that of real people.

4.1. PSO essence

We control the movement on the x–z plane,[1] just like to have a bird's eye view on the top of people's heads. The position and the velocity can be represented by 2D vectors, such as $p_i = [p_{ix}, p_{iz}]^T$ and $v_i = [v_{ix}, v_{iz}]^T$. Although it is unnecessary, simply for convenience, we separate the velocity into a direction part, $D_i = [D_{ix}, D_{iz}]^T$ and a speed part, S. The speed part S is used to model and match the various paces of different people and has a maximum limit. Each particle holds the information about itself, including a position, a direction, a speed, and an objective value. Based on PSO, the direction is computed as

$$D_i(t+1) = \omega * D_i(t) + c_1 * r_1 * (p_{iBLS} - p_i(t)) + c_2 * r_2 * (p_{BGS} - p_i(t)),$$

where $(p_{iBLS} - p_i(t))$ and $(p_{BGS} - p_i(t))$ are both unit vectors for indicating the direction only. Other parameters are defined as the same parameters of the velocity in the conventional PSO.

After deciding the direction of the particle, the position can be computed as

$$p_i(t+1) = p_i(t) + S_i(t+1) * D_i(t+1),$$

where $S_i(t+1)$ is the speed part. The range of speed is $[0, V_{max}]$, and V_{max} is set by a step size with a random number or some extra parameters adopted by the user. The speed is updated proportional to the reverse of the particle's objective value. Such a design can fit the different pace of each person and make the environment more dynamic. Because the whole environment is established as a landscape for minimization, if a particle approaches an obstacle, the speed will be slower, due to greater objective values, for the particle to avoid the obstacle. The mechanism can eliminate the collision issues and reduce the oscillatory situation that occurs in PSO. As a consequence, the generated paths may look more natural and realistic.

At each step, each particle gets an objective value to measure the current position. At the same time, the local best position and the global best position are updated according to the newly obtained objective value. Such a PSO mechanism, with appropriate parameter settings, will make the particles in the group converge to the target in a high probability. Moreover, we can adopt different objective functions to arrange the final status of a group, such as a line, a circle, or other possible, desired shapes.

4.2. Objective function

As a matter of fact, PSO is only interested in the final positions of particles and cares nothing about the particle paths at each step. It is very different from crowds in the real world. In the real world, not everywhere can be stepped on or gone through. There are obstacles, such as holes and walls. Moreover, a person normally also does not step on another person. These situations contribute to the incompatibility of PSO for controlling the movement of crowds. In order to resolve these incompatibility issues with a uniform approach, we firstly design an objective function to represent the specified target, the static obstacle, as well as all the particles.

4.2.1. Cost function for a unit obstacle

The objective value of a particle is affected by two factors: the target and the obstacles. For the purpose to use the optimization ability of PSO, we make the target as the minimum on the x–z plane, considered as the mathematical search space. If a particle approaches the target, it should get a better objective value. All the particles in the crowd move toward the region with lower objective values just like water flows downward. On the other hand, the objective value rises as penalty if the particle comes close to or even possibly touches obstacles. Since PSO used in this study is for solving a minimization problem, we will view the objective value as "cost" in the remainder of this paper.

We use an exponential function to represent everything in the search space, including the target and the obstacles. Exponential functions are adopted because they are easy to calculate and to manipulate. Functions of other families can also be adopted in a similar way as long as they can fulfill the need of users. The function adopted in this study for calculating the cost of an object p relative to another object q is defined as

$$\text{Cost}(p,q) = \exp\left(-\left(\frac{(p_x - q_x)^2}{(\sigma_{p_x} + \sigma_{q_x})^2} + \frac{(p_z - q_z)^2}{(\sigma_{p_z} + \sigma_{q_z})^2}\right)\right), \quad (3)$$

where (q_x, q_z) is the the position the object q, and $(\sigma_{q_x}, \sigma_{q_z})$ is the *scope* of q. q can be a target, an obstacle, or a particle. For example, if an obstacle's area is 30 by 50, we can set $(\sigma_{q_x}, \sigma_{q_z}) = (15, 25)$. If it is the target, $(\sigma_{q_x}, \sigma_{q_z})$ is set to the whole search area. (p_x, p_z) is the position of a particle for which the cost is calculated. $(\sigma_{p_x}, \sigma_{p_z})$ is the scope in the search space occupied by p. Fig. 1 shows the exponential model for an obstacle. The box indicates the obstacle to be represented by the function. The scope for an object can be easily adjusted and controlled with the designed parameter.

The proposed exponential model is similar to the 2D normal distribution with a mean and a standard deviation. The difference is that every exponential function representing an object in the system has its own volume size, while the volume size is always one for the normal distribution. The different sizes in volume make it relatively easy to model the landscape for pedestrians to go through as well as for the space occupied by particles and objects. A complex landscape with a minimum position as the target can be created by overlaying these exponential functions with identical or different parameters. Therefore, the overall objective function proposed in this study can be described as

$$F_{obj}(p) = c_{obs} * \max_{o \in O}(\text{Cost}(p,o)) + \frac{1}{\text{Cost}(p,g)},$$

where O is the set of all obstacles, g is the target for particles to reach, and c_{obs} is a constant for the user to adjust the relative importance between the target and obstacles.

It has to be noticed that the set O contains not only all the specified obstacles, such as holes and walls, on the landscape but

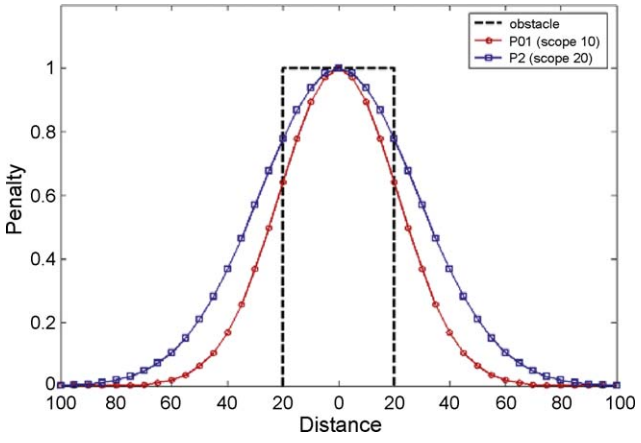


Fig. 1. The exponential model for an obstacle.

[1] Because the y-axis usually represents the height in the 3D space in computer graphics, the ground surface is the x–z plane.

also other people in the crowd. We adopt the identical model for everything in the scenario instead of using different models for objects of different kinds. By doing so, no extra model has to be introduced into the system when new objects are included, such as moving cars or running animals. Furthermore, the proposed model induces an interesting situation for PSO. Every particle actually "sees" a different landscape due to the relative relations among particles. Such a situation does not exist in common optimization applications. Under this condition, according to the results of this study, with appropriate parameter settings, PSO can accomplish its assigned task, and the particles converge to the desired goal via reasonable paths in a high probability.

4.2.2. Local search for collision avoidance

Even with the specifically designed objective function, the possibility for a particle to pass through an obstacle still need to be eliminated. In this study, we implement this functionality as a form of local search, which is a common mechanism used with PSO. When we get the cost for a new position of a particle, whether or not the new position is accepted should be checked. We use a stochastic mechanism to decide whether the new position should be accepted according to the cost. The probability to accept a newly generated position is computed by

$$\text{Prob}(f) = 1 - \frac{f}{e^{-k}}, \tag{4}$$

where f is the cost for the newly generated position, and k is a constant to control the behavior when a particle come close to the obstacle. The shape of the exponential function is quite appropriate to estimate whether or not a particle is too close to an obstacle. It helps the particle to avoid collisions and makes the path smoother. Moreover, there exists a hard boundary when $\sigma = 1$ according to Eq. (3), because the probability will be 0 if $f = e^{-1}$. Therefore, we can theoretically verify that collisions under this checking mechanism can never happen. Similar to the use of exponential functions, Eq. (4) can also be possibly replaced by other kinds of functions as long as the user deems fit. Fig. 2 shows the probability to accept a new position around an obstacle according to Eq. (4). Different values of k can be used to conveniently and appropriately specify the accepting probability around the obstacle.

If the cost is not accepted, another direction must be taken to create a smooth path. According to the original direction, a random angle less than 20° is added to or subtracted from the direction vector of the particle to find an acceptable position. The position of a lower cost is chosen to be the new position.

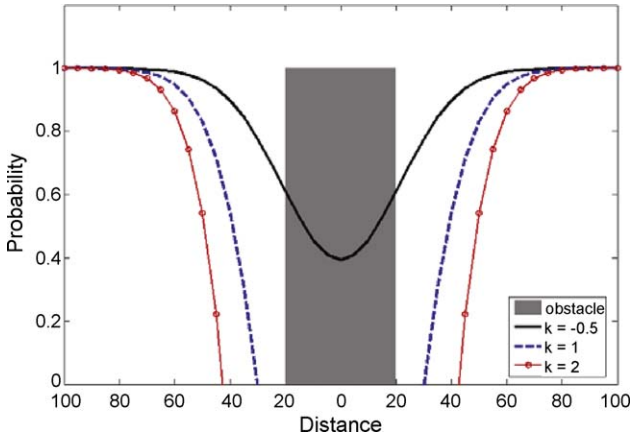


Fig. 2. The probability to accept a new position around an obstacle.

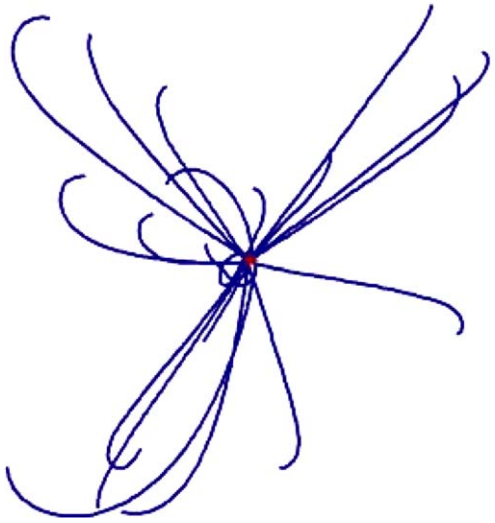


Fig. 3. The paths of an influx simulated by the original PSO.

5. Moving path generation in different scenarios

With the proposed framework, we generate the moving paths for several common scenarios on the x–z plane and show the movement of crowds. In addition to the figures displayed in this paper, for animated visualization, readers can watch the video clips at http://www.nclab.tw/SM/2007/02/video/ and the `.avi` filenames will be indicated in the following sections. Readers can also directly access the file with its URL, such as http://www.nclab.tw/SM/2007/02/video/01.avi.

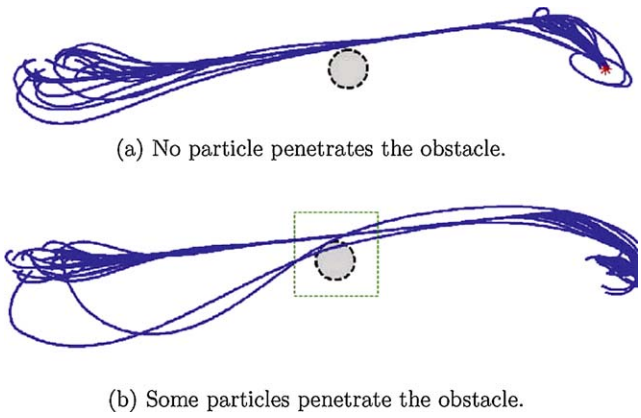
5.1. Crowd simulation by using the original PSO

We first attempt to simulate a crowd by using the original PSO. The parameter settings of this experiment for PSO are $\omega = 1.0$, $c_1 = 0.5$, $c_2 = 0.5$, and the population size is 20. Because the purpose in this study is to generate moving paths, the initial positions of the particles are assigned according to the need of users. The velocities are initialized at random. Fig. 3 shows an influx. Each solid line indicates a path of a person. People initially are scattered and finally converge to the target. Fig. 4 shows a stream, in which people on the left side move toward the right side. We can observe that they converge first and move toward the target together. For an influx or a stream, the appropriate paths can be created under the mechanism of the original PSO. The video clips `01.avi` and `02.avi` display the dynamics for the two cases.

5.2. Crowd simulation with static obstacles

The original PSO may make all the persons to reach the goal, and sometimes an obstacle can be avoided with certain probability as shown in Fig. 5(a). For this experiment, the parameter settings are $\omega = 1.0$, $c_1 = 0.5$, $c_2 = 0.5$, and the population size is 20. The static obstacle is modeled at (500, 15) with $\sigma = 30$ and $k = 1$. As we can clearly observe in another typical run, the original PSO does not have the ability to make the particles to avoid obstacles. The particles may fly through obstacles, which should not be stepped on or penetrated, as shown in Fig. 5(b) as well as in `03.avi`.

Fig. 4. The paths of a stream simulated by the original PSO.



(a) No particle penetrates the obstacle.

(b) Some particles penetrate the obstacle.

Fig. 5. The original PSO does not guarantee to avoid obstacles. (a) No particle penetrates the obstacle. (b) Some particles penetrate the obstacle.

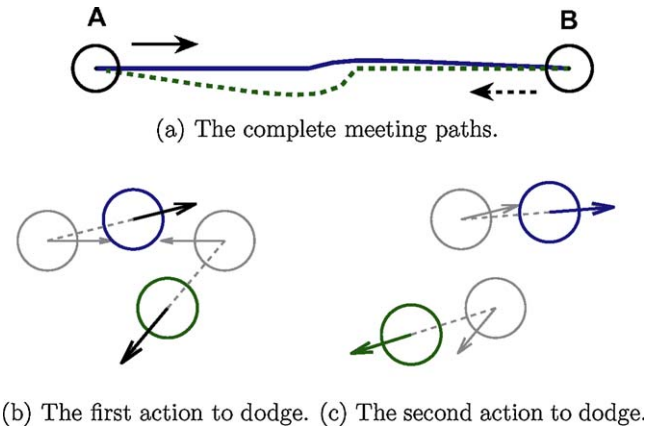
Therefore, the proposed collision avoidance mechanism has to be employed. The simulation results will be given in the next section with dynamic obstacles considered simultaneously.

5.3. Crowd simulation with dynamic obstacles

Avoiding collisions between persons in a crowd is a necessity for generating reasonable moving paths. In this study, we do not resort to any extra method or mechanism to handle this issue. In the proposed framework, each person is considered as an dynamic obstacle with $\sigma = 5$ and $k = 1$. If two persons come close, the cost of each person will be checked, and an acceptable position for each person will be determined. The parameter settings are $\omega = 1.0$, $c_1 = 0.5$, $c_2 = 0.5$, and the population size is 20. Fig. 6(a) presents the paths when two persons almost collide. The dotted line is the path from A to B, and the solid line is the one from B to A. Fig. 6(b) and (c) show the jinking situation. More people are simulated in Fig. 7. Each circle represents a person, and the dotted lines are their waking paths. `04.avi` shows the movement of a crowd with collision avoidance. The simulation results demonstrate the generated paths without collisions. Furthermore, `05.avi`, in which the static obstacle is modeled at (500, 15) with $\sigma = 30$ and $k = 1$, demonstrates that static and dynamic objects can be avoided simultaneously as mentioned in Section 5.2.

5.4. Crowd simulation with a dynamic target

In addition to the fundamental path generation presented in the previous sections, we also conducted experiments on the crowd



(a) The complete meeting paths.

(b) The first action to dodge. (c) The second action to dodge.

Fig. 6. The dodge of two people with opposing directions is simulated. (a) The complete meeting paths. (b) The first action to dodge. (c) The second action to dodge.

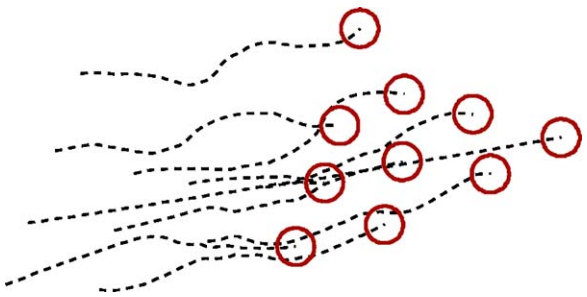
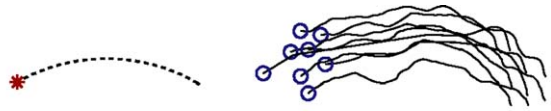


Fig. 7. A walking crowd of which each particle avoids others.

simulation with a dynamic target in our proposed framework. The parameter settings are $\omega = 1.0$, $c_1 = 0.5$, $c_2 = 0.5$, and the population size is 20. As the target moves, the crowd is capable of following the moving target as shown in Fig. 8. The path of the goal is designed as a circle with radius = 250. Fig. 9 depicts the generated circular paths according to the moving goal. `06.avi` shows the process of the particles chasing the moving goal. When a particle comes close to the target, the target goes one further step. We can simulate a continuous chasing by setting up very small steps for particles and the goal.

5.5. Simulation for multiple crowds

For the proposed framework, simulating multiple crowds going toward different directions and/or targets is a simple extension. Multiple groups can be simply overlaid on the same area such that more complex scenes are made possible. Fig. 10 demonstrates the scenario in which four groups at the four corners move toward their respective opposite corners. The parameter settings for each group are $\omega = 1.0$, $c_1 = 0.5$, $c_2 = 0.5$, and the population size is 10. For observation, we use four different kinds of symbols to represent the persons in different groups. Each group consists of



(a) A moving goal. (b) The paths of the crowd.

Fig. 8. The moving goal can control a group to behave as expected. (a) A moving goal. (b) The paths of the crowd.

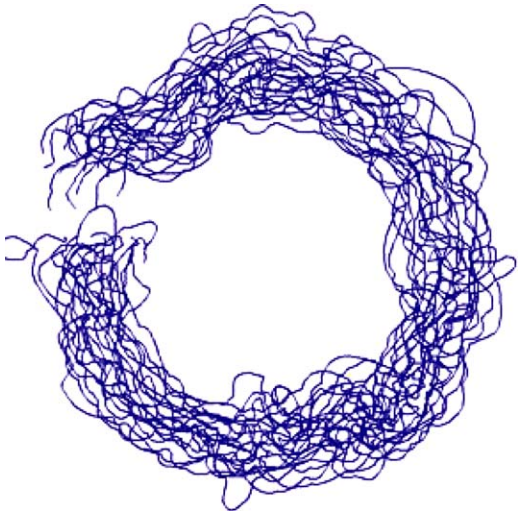


Fig. 9. The paths of a crowd following a dynamic goal moving circularly.

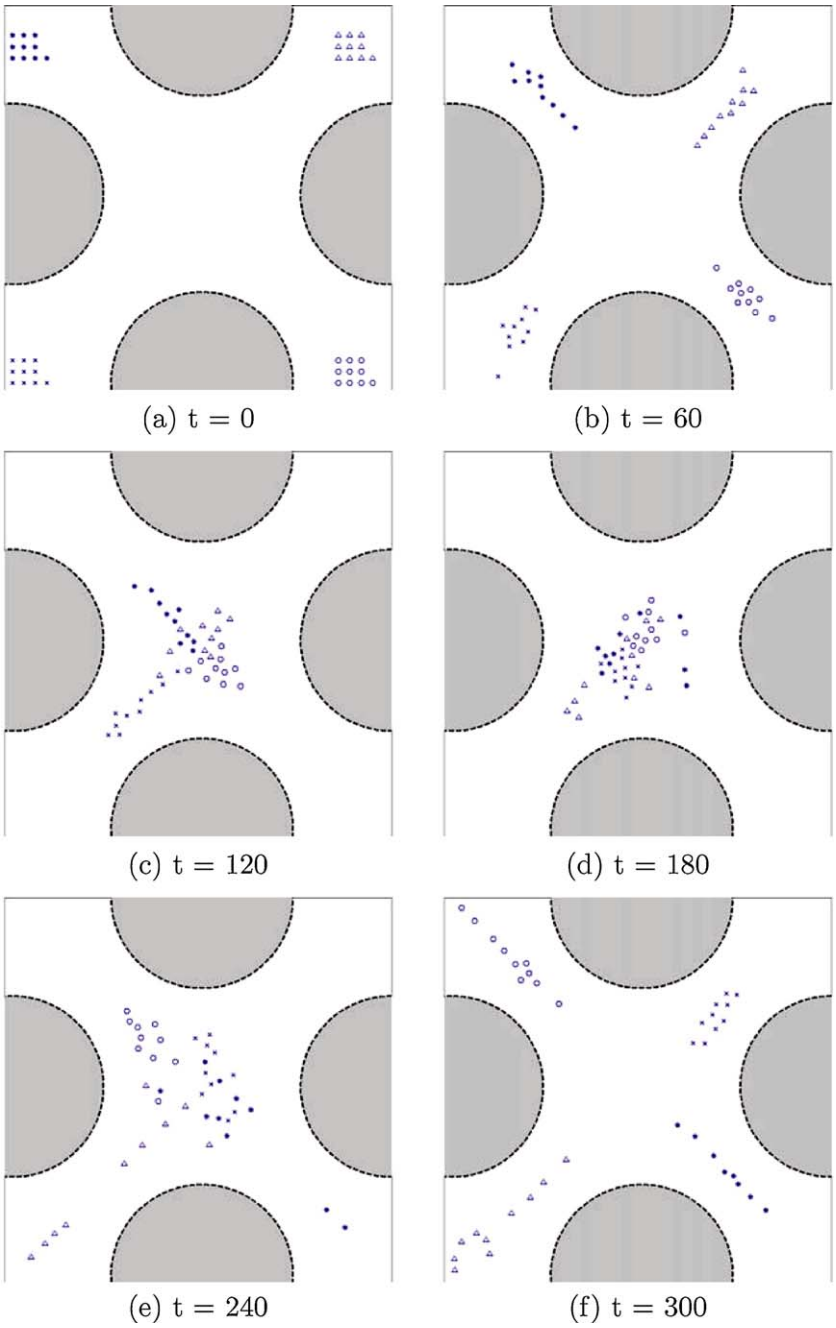


Fig. 10. Four groups move toward the opposite corners without any collisions. (a) $t = 0$, (b) $t = 60$, (c) $t = 120$, (d) $t = 180$, (e) $t = 240$, (f) $t = 300$.

ten members. As time goes by, persons in each group can pass by the obstacles and persons in other groups and reach the designated goals. `07.avi` provides the animation for the scenario. The dynamics of the four groups can be easily observed via the video clip.

5.6. Control of the 3D landscape

The proposed model can also be used to control and model the 3D landscape of the given virtual environment and therefore to obtain the desired crowd movement. `08.avi`, `09.avi`, and `10.avi` demonstrate the whole process of a crowd passing over mountains of three different heights. The parameter settings in this experiment are $\omega = 1.0$, $c_1 = 0.5$, $c_2 = 0.5$, and the population size is 300. As we can see in `08.avi`, the mountain is highest, and no one goes to the top of the mountain. As the height of mountain becomes lower, more individuals are able to step on the top of the mountain and go straightforward toward the destination.

6. Conclusions

This paper proposed a uniform model to co-operate with particle swarm optimization to simulate crowd movements. We considered that people finding a walkable path to their target as the process to find the optimal solution by PSO. The advantages of PSO are simple, fast, and easy to implement. By the PSO mechanism, each person can search for a path automatically. However, particles controlled by the original PSO may penetrate obstacles. Hence, we developed the collision avoidance mechanism in the form of local search to work with PSO. Static obstacles, dynamic obstacles, and the target were all modeled with exponential functions. Combining these exponential functions,

the scenario environments were constructed, and the particle paths were generated by the proposed framework.

The proposed method in this study is compact, coherent and controls the movement of crowds easily. Based on the uniform model, we can construct a complex "crowd-scape" by stacking up several crowds. The created paths in such an environment can therefore be more dynamic and stochastic. Although this study is not the first to apply the concept of swarm intelligence on crowd control, to the best of our limited knowledge, it retains the most design of the original PSO and almost leaves PSO unmodified. The proposed model is flexible, versatile and can be used to represent a variety of objects.

The future work includes understanding how the parameters affect the paths, determining whether functions of other classes can be employed for creating better paths, and integrating the framework into the existing computer graphics systems. Theoretical insights into the crowd behavior might also be obtained through the development of the proposed framework.

Acknowledgments

The work was supported in part by the National Science Council of Taiwan under Grant NSC-96-2221-E-009-196 and Grant NSC-96-2627-B-009-001. The authors are grateful to the National Center for High-performance Computing for computer time and facilities.

References

[1] N. Magnenat-Thalmann, D. Thalmann, Virtual humans: thirty years of research, what next? The Visual Computer 21 (12) (2005) 997–1015.
[2] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the 1995 IEEE International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948.
[3] C.W. Reynolds, Flocks, herds, and schools: a distributed behavioral model, Computer Graphics 21 (4) (1987) 25–34.
[4] C.W. Reynolds, Steering behaviors for autonomous characters, in: Proceedings of Game Developers Conference, 1999, pp. 763–782.
[5] E. Bouvier, P. Guilloteau, Crowd simulation in immersive space management, in: Proceedings of the Eurographics Workshop on Virtual Environments and Scientific Visualization'96, 1996, pp. 104–110.
[6] D.C. Brogan, J.K. Hodgins, Group behaviors for systems with significant dynamics, Autonomous Robots 4 (1) (1997) 137–153.
[7] G.K. Still, Crowd dynamics, Ph.D. thesis, Warwick University, 2000.
[8] D. Helbing, I. Farkas, T. Vicsek, Simulating dynamical features of escape panic, Nature 407 (2000) 487–490.
[9] A. Aubel, D. Thalmann, MuscleBuilder: a modeling tool for human anatomy, Journal of Computer Science and Technology 19 (5) (2004) 585–595.
[10] F. Tecchia, Y. Chrysanthou, Real-time rendering of densely populated urban environments, in: Proceedings of the Eurographics Workshop on Rendering Techniques, 2000, pp. 83–88.
[11] A. Aubel, R. Boulic, D. Thalmann, Real-time display of virtual humans: levels of details and impostors, IEEE Transactions on Circuits and Systems for Video Technology 10 (2) (2000) 207–227.
[12] B. Ulicny, D. Thalmann, Towards interactive real-time crowd behavior simulation, Computer Graphics Forum 21 (4) (2003) 767–775.
[13] A. Treuille, S. Cooper, Z. Popović, Continuum crowds, ACM Transactions on Graphics 25 (3) (2006) 1160–1168.
[14] S. Stylianou, Y. Chrysanthou, Crowd self organization, streaming and short path smoothing, Journal of WSCG 14 (2006) 33–40.
[15] H. Kwong, C. Jacob, Evolutionary exploration of dynamic swarm behaviour, in: Proceedings of IEEE Congress on Evolutionary Computation 2003 (CEC 2003), 2003, pp. 367–374.
[16] D.H. Kim, S. Shin, Self-organization of decentralized swarm agents based on modified particle swarm algorithm, Journal of Intelligent and Robotic Systems 46 (2) (2006) 129–149.
[17] G. Beni, J. Wang, Swarm intelligence in cellular robotics systems, in: Proceedings of the NATO Advanced Workshop on Robots and Biological Systems, 1989, pp. 703–712.
[18] M. Dorigo, Optimization, learning and natural algorithms, Ph.D. thesis, Politecnico di Milano, Italy, 1992.
[19] J.M. Bishop, Stochastic searching networks, in: Proceedings of the First IEE International Conference on Artificial Neural Networks, 1989, pp. 329–331.