

## Research Article

# Research of Simulation in Character Animation Based on Physics Engine

Yang Yu,<sup>1</sup> Jucheng Yang,<sup>1</sup> Xiaofei Zan,<sup>2</sup> Jiangang Huang,<sup>1</sup> and Xiangbo Zhang<sup>1</sup>

<sup>1</sup>College of Computer Science and Information Engineering, Tianjin University of Science and Technology, Tianjin, China

<sup>2</sup>School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China

Correspondence should be addressed to Jiangang Huang; [huangjg@tust.edu.cn](mailto:huangjg@tust.edu.cn)

Received 28 February 2017; Accepted 17 May 2017; Published 12 July 2017

Academic Editor: Jenq-Neng Hwang

Copyright © 2017 Yang Yu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Computer 3D character animation essentially is a product, which is combined with computer graphics and robotics, physics, mathematics, and the arts. It is based on computer hardware and graphics algorithms and related sciences rapidly developed new technologies. At present, the mainstream character animation technology is based on the artificial production of key technologies and capture frames based on the motion capture device technology. 3D character animation is widely used not only in the production of film, animation, and other commercial areas but also in virtual reality, computer-aided education, flight simulation, engineering simulation, military simulation, and other fields. In this paper, we try to study physics based character animation to solve these problems such as poor real-time interaction that appears in the character, low utilization rate, and complex production. The paper deeply studied the kinematics, dynamics technology, and production technology based on the motion data. At the same time, it analyzed ODE, PhysX, Bullet, and other variety of mainstream physics engines and studied OBB hierarchy bounding box tree, AABB hierarchical tree, and other collision detection algorithms. Finally, character animation based on ODE is implemented, which is simulation of the motion and collision process of a tricycle.

## 1. Research Background and Significance

The mainstream animation production technologies are key frame technology based on artificial production and motion capture technology based on capture equipment. Key frame technology based on artificial production is the original animation combination technology. It depends on the key frame production and the interpolation between key frames to drive the continuous animation playing. The technology allows designers to create animation by their own will with the greatest degree of freedom, but the products of the technology are not very good in terms of natural performance; moreover, the process of production is quite complicated [1]. Since the motion capture equipment appeared, it can be used to capture the motion data as the information source of animation production, so that the efficiency of producing character animation and the reality sense of character animation can be greatly improved. However, since the technology needs the genuine characters to take part in the process of capturing

the motion information, its application scope becomes quite limited. When using this technology, it is not allowed for the genuine actors to take some dangerous actions. Furthermore, the high cost of this kind of equipment will also restrict the wide use of the technology.

In the background of growing industrial demand, increased user experience requirement, and fast developing of computer-related technologies, the above-mentioned two kinds of common traditional animation production methods are becoming less and less possible to satisfy the requirements of design and use. The root cause is that the two kinds of technologies use the offline way during the process of animation combination and storage and then for the post-production use; thus, the animation will be charged with the corresponding huge data storage space considering its high quality and long duration. Moreover, the biggest issue of this method is the poor rate in terms of real-time and self-adaptation performance, so that the animation, which has cost plenty of time and money, can be just used for the specific

circumstance for once. All of these issues do not fit into the current trend of fast increase of computer processing and user experience [2].

This article, in order to deal with the issues of poor real-time performance, low utilization rate, and complicated procedure, existing during the process of producing character animation, does research on the character animation simulation method based on physics engine and uses an example to implement the method.

## 2. Foundation of Related Researches

Nowadays, character animation is always the focus and difficulty in the computer graphics research. In the research, there are various animation production methods, which can be categorized as follows [3]:

- (i) The animation production method based on kinematics
- (ii) The animation production method based on dynamics
- (iii) The animation production method based on motion data

A brief introduction of the above three kinds of character animation is next.

*2.1. Principle of Character Animation Production Based on Kinematics.* Kinematics is a very important branch in physics mechanics. During the research, the object is abstracted as a rigid body model and a particle model [4]. These models are used to study the motion of objects. The key point of the rigid body model is to research the rigid body's angular velocity, rotary motion, linear velocity, and so forth. The focus of the research on the particle model is on the velocity, acceleration, deceleration, and motion equation of the particle in the specified reference system. In addition, it is not necessary to consider the influence factors of dynamics, such as quality and force.

The kinematics method is divided into forward kinematics and inverse kinematics.

*2.1.1. Forward Kinematics Method [5].* In a simple kinematic system, there are often two ends. They are free end and fixed end. Forward kinematics describes the motion of the free end with the fixed end of the system as the starting point. Forward kinematics method: the first step is to establish the hierarchical structure of the object and then to establish the rotation and position of each object in the hierarchy.

*2.1.2. Inverse Kinematics Method.* Another simulation technique based on kinematics is the inverse kinematics method [5]. Inverse kinematics is just opposite to forward kinematics. The inverse kinematics is the motion of which the fixed end drove by the free end.

Use the following formula to express the basic principles of the inverse kinematics [6]:

$$\Delta X = J\Delta\theta. \quad (1)$$

Formula (1) is the basic formula of the inverse kinematics, in which  $\Delta X$  is the displacement of the end of the joint chain,  $\Delta\theta$  is the rotate angle of the end of the joint chain, and  $J$  is the Jacobian matrix. By the formula above, the displacement of joint is equal to a product, the rotation angle of joint chain ends, and the Jacobian matrix. But formula (2) is frequently used in inverse kinematics [6]:

$$\Delta\theta = J^+ \Delta X + \alpha (I - J^+ J) \Delta z. \quad (2)$$

In the formula,  $\Delta X$  and  $\Delta\theta$  are similarly found in formula (1).  $J^+$  is pseudo-inverse matrix of the Jacobian matrix in linear algebra.  $I$  is a unit matrix.  $\alpha$  is optimization constant.  $\Delta z$  is the energy consumption of joints in minimizing motion. The inverse kinematics is calculated from the end of the entire joint chain. Compared with the forward kinematics, inverse kinematics is more suitable for the creation of more complex motion and easier to enable the flip of the object in the animation. However, the inverse kinematics is not universal. Some problems which are suitable for forward kinematics are not suitable for inverse kinematics.

*2.2. Principle of Animation Production Based on Dynamics.* Dynamics is also a very important branch of physics, which researches the relationship between force and motion. The basic theory of kinetic research is Newton's law of motion. The object of the dynamics study is a macroscopic object that is far lower than the speed of light. The animation produced by this method has a strong physical reality, because it is based on the laws of physics.

In the simulation of character animation, kinematics method is to calculate the position and rotate angle of each joint for the characters, and the key frames are composed of two kinds of data.

In the simulation of character animation, the dynamic method first gives the object to the physical properties and exerts various forces and then uses Newton's law of motion to calculate the state of the character. The essence of this method is to calculate the acceleration of an object by force. In dynamics, the type of force can be divided into two types: point force and field force [7].

In addition to the force, the collision is also an important factor affecting the movement of the character. In the real world, when two objects collide, they collide, but they do not penetrate each other. When the collision occurs, the interaction force will change the original state of the two objects, such as position, trajectory, and direction. Therefore, in the simulation of the character animation, if the collision is between objects, the original motion state of the character will be changed because of the interaction force.

### 2.3. Principle of Character Animation Production Based on Motion Data

**2.3.1. Motion Capture.** The animation methods mentioned above are realized by the use of mechanics, biomechanics, and daily life experience. In motion capture technology, data is originated from the real actor performance, through motion capture equipment to obtain action data. Because the processed data is produced by human beings, the motion capture technology can make the animation more real and natural. There was no artificial data from capture and processing. But, motion capture equipment is very expensive and complicated to operate. During the data collection, the motion model's limb, body, and head commonly set sensors point, and those collective data are a very important part in whole motion capture technology [8].

The dynamic and kinematics techniques introduced in this paper are used to animate the physical laws of motion. However, the complexity of human motion with physical laws, especially in terms of human motion coordination mechanism, makes simulation work very difficult, thus leading to the produced animation being not realistic and the lack of rich detailed information of actual human body movement. In recent years, the motion capture technology has entered the line of sight; it overcomes the shortcomings of the above two methods of animation and has become one of the most promising technologies in human animation.

**2.3.2. Motion Blend.** The motion blending technology is a technology based on special motion database and algorithm. To use this technique to deal with animation, we generally need to use interpolation algorithm and interpolation parameters.

The main advantage of motion blend technology is its low computational complexity. And it is based on motion capture data, to maintain the original dynamic motion. Its main drawback is that, for example, tension, worry, and other acts are too dependent on the existing operating data, and the amount of data increases. If a character is walking in a gait that is not too tired or tired, it is necessary to have a normal walking gait data and also to have a very tired walking data. In the interpolation process, the producers usually deal with the transition and transformation between the collection of parameters by the persistent and linear default views [9].

**2.3.3. Motion Deformation.** Motion deformation is the technology to modify and use the existing motion information. Popoivc and Witkin use interactive methods to shift and scale the selected key frames to modify the existing data. Compared with the trajectory in the time domain, Williams and Bruderlin are used to adjust, modify, and reuse the existing animation data in the frequency domain. Compared with moving animation technology and process animation technology, the advantage of motion mixing technology is that it can use the existing real capture data for animation generation. Moreover, this technology gives the animation creator and key frame animation compatible with a tool that will be very convenient to deal with.

Compared with kinematics animation technology and process animation technology, the advantage of the motion blending technology, which belongs to the process animation technology, is that it can make use of the existing real capture data to generate animation [10]. Moreover, this technology is compatible with key frame animation technology.

## 3. The Physics Engine Related Research

Generally, there are two types of physics engines: open source physics engine and commercial physics engine.

Commonly used professional physics engines include ODE, PhysX, Bullet, Havork, Newton, and Vortex. These physics engines are widely used in the domains as graphics modeling tools, game development, virtual reality systems, and so on.

Collision detection is the technical core of physics engine, so most of the research on physics engine is the research on the collision detection algorithm. In real life, some reactions occur naturally at and after the collision of objects [11]. However, in the virtual world, the collision issue needs some algorithms to be solved. Collision detection is the important part of the domains like Robot, VR, and so on. Its main purpose is to determine whether two or more objects will penetrate or contact. The research of collision detection is, through a certain algorithm, to enable two or more objects not to occupy the same spatial area at the same time, without damaging the objects.

At present, there are two major categories of research on collision detection worldwide: spatial decomposition method and bounding box method. The bounding box method simulates a complexly shaped object into an approximate simple three-dimensional space. The hierarchical bounding box method of the bounding box method is suitable for the collision detection in complex environment.

The ODE (Open Dynamics Engine) is an open source library of rigid body dynamics that is often used in the simulation of joint connections. ODE can be applied to simulate the moving objects in the VR environment, legged creatures, and vehicles on the ground. It is robust, flexible, and fast. Flexibility is reflected in the portability; Unix/Linux, Windows, and MacOS can be used. And it has a built-in collision detection system.

## 4. Research and Design of Character Animation Simulation Based on ODE

**4.1. Dynamics Modeling of Simulated Vehicle Based on ODE.** In ODE, there is a special type of joint as shown in Figure 1.

Hinge-2 joint is made up of two noncoaxial hinges. The utility model comprises two rotating shafts and a supporting point. Among them, Axis 2 can only rotate around the axis; however, Axis 1 not only can rotate around the axis but also can limit the scope of its rotation.

The car model consists of body and wheels. According to the joint type of Hinge-2, Body 1 can represent the body and Body 2 can represent the wheel; similarly, Axis 1 can represent the steering wheel of the wheel and Axis 2 can represent the

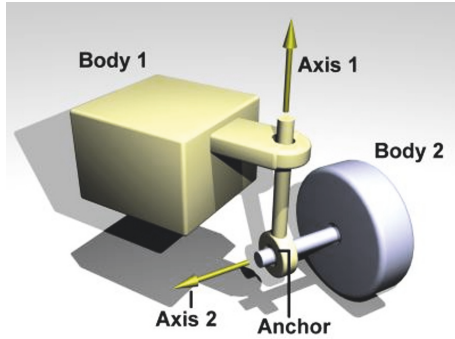


FIGURE 1: Hinge-2 joints [20].

wheel rolling shaft. In the dynamics design of the ODE system, ODE rigid object with the quality and position is generated, and ODE geometry corresponding to the rigid body is created at the same time. Finally, the body and the wheels are connected by Hinge-2 joint. A simple car model has been built.

**4.2. Simulation of Character Collision Based on ODE.** The simulation process of character collision based on ODE is as follows.

**4.2.1. Spatial Initialization in Collision Simulation.** At this stage, it is necessary to set the type of collision space and the relevant properties of the contact surface, such as flexibility, elasticity, and friction.

**4.2.2. Data Initialization in Collision Simulation.** At this stage, it is necessary to do some work on the initialization and cleanup of the data before the collision simulation [12].

**Collision Data Cleaning.** Clear all kinds of collision information, including clearing collision data, clearing contact group, and destroying the current collision space. Then, reset all the properties of the contact surface to comply with the needs of the user.

According to user needs, create four-fork tree space, hash space, and simple space. These three new types of collision space use different collision detection algorithms and different data structures. Different geometries are stored using different data structures.

**Collision Data Detection.** Collision data is a custom variable that stores all the data needed for a collision geometry. The character object is the source of the data. Prior to the crash simulation, you need to set up a collision data for each collision object and add the data to a list of collisions. Finally, the collision information is transferred to the collision function when the collision simulation is performed.

**4.2.3. Collision Simulation Start.** The simulation process takes the time step as a unit. With the ODE simulator, simulation to complete a time step for the unit collision effect

needs to use the current state before the impact and role of the data and then the collision effect exerted on the role of the body [13, 14]. This link needs to call the existing space collision function in ODE. Finally, according to the data of the contact nodes generated by the two contact rigid bodies in the collision, the ODE completes the collision [15]. The space collision function is a very important function in this link. Its function prototype is

```
Void dSpaceCollide (dSpaceID space,
void *data, dNearCallback *callback).
```

This function determines which geometries are likely to occur in space and then calls the callback function to handle the pair of geometries and finally completes the collision.

Callback's function type is dNearCallback, defined as follows:

```
Typedef void dNearCallback (void
*data, dGeomID o1, dGeomID o2)
```

Among them, data is user-defined parameters, directly from the dSpaceCollide pass. A pair of geometries that detect a possible collision are represented by parameters o1 and o2 [16]. This callback function then uses dCollide to generate the contact point between the geometries. Then, these contact points will be added to the simulation process as a contact joint.

The following operation will occur inside the callback [17]:

- (i) Use the dCollide function to generate the selected contact points between the geometries, while returning the number of contact points.
- (ii) Based on all the collision data in the previous link, for each contact point, two eligible collision data points are searched.
- (iii) ODE completes the collision effect according to the contact joint information contained in these two collisions and shows the picture again.

**4.2.4. Collision Simulation End.** With the end of the ODE simulation, the collision simulation also acts as a driving force to stop the character of force. This link will do some related data cleanup work, which is conducive to the next simulation.

## 5. Implementation and Testing of Character Animation Simulation

**5.1. Creating an Example in the Character Animation.** Open the physics engine software and edit and run the C language program to get the scenario shown in Figure 2.

In Figure 2, you can see the basic elements of the scene, including the sky, the ground, the car, and the obstacles.

The simulation example consists of simple elements: the power of vehicle comes from the motor; the power output and the steering are set in the front wheel; the wheel and the vehicle body joint through the hinge; the vehicle may have interaction forces when touching any object of the scenario; there is a collision detection device in the vehicle.

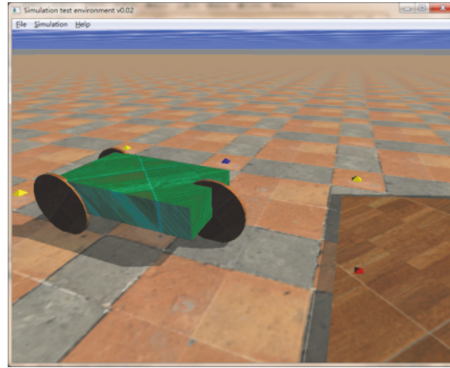


FIGURE 2: Scenario picture.

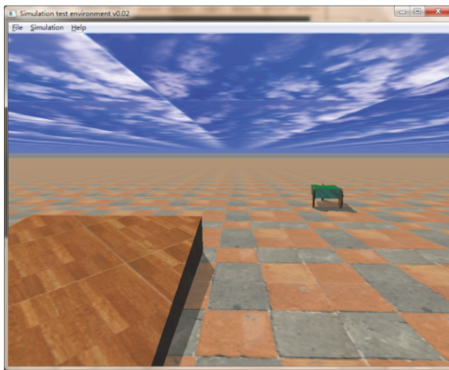


FIGURE 3: Forward, backward, acceleration, and deceleration of the vehicle model.

**5.2. Testing.** By setting the power of the vehicle, the interaction force between the vehicle and the objects in the scenario, as well as the ground slip rate, the simulation of the vehicle motion, and collision process can be implemented.

The performance testing is shown in Figures 3, 4, and 5.

In the course of the testing, change the vehicle or the various parameters of the surrounding environment and observe whether the character change in the pictures conforms to the kinetic principle.

**5.2.1. Friction Testing.** When the vehicle has contact with the ground, change the Coulomb friction coefficient  $\mu$  from the original  $\infty$  (there will not be sliding contact) to 0 (no friction contact). This coefficient must be set. And the testing result is that the wheel is idling. At this time, the front wheel as the power output can only be idling, since there is no friction, so the vehicle cannot move [18].

**5.2.2. Penetration Testing.** When the vehicle is in contact with the ground,  $\text{soft\_cfm}$  is the contact normal flexibility parameter, which is used to adjust the flexibility of the object. Enlarging the  $\text{soft\_cfm}$  value, from the original 0.3 to 1, indicates that there will be very obvious object penetration

phenomenon; the testing result is shown in Figure 6; there appears very obvious penetration phenomenon between the wheel and the ground [18].

**5.2.3. Slide Coefficient Testing.** Set the FDS (Force Dependent Slip factor) to zero. After the vehicle model rolls over, continue to give power to the vehicle, and the vehicle has no side slide but has been static. (FDS indicates the lateral slide phenomenon due to the lateral external force when the object is moving, and the sliding distance is proportional to the moving speed of the object and the external force, so FDS is also called lateral force slide coefficient [18, 19].)

If the FDS slide coefficient of friction is set to 0.1, continue to give power to the vehicle after its rollover, and the vehicle can still move laterally. The testing result is shown in Figure 7.

When the world is created, the  $z$ -axis direction of gravity acceleration that is set to 0 can be obtained as in Figure 8.

From the test results, due to the acceleration of gravity of 0, there is simulation of the suspension in the vehicle.

Through the simulation of the car and the surrounding environment of various parameters set, you can carry out a variety of physics engine-based animation simulation.

## 6. Summary

Regarding the research on the simulation of character animation, there have been many achievements at home and abroad. On the basis of studying and researching the existing scientific research achievements, this article has done the following works: in-depth research on the mainstream computer character animation production technologies; in-depth analysis and research on the mainstream physics engines; in-depth research on the related collision detection algorithms; designing and building the simulation architecture, to make the character animation simulation feasible by using ODE engine; building the simulation experiment platform and successfully operating examples

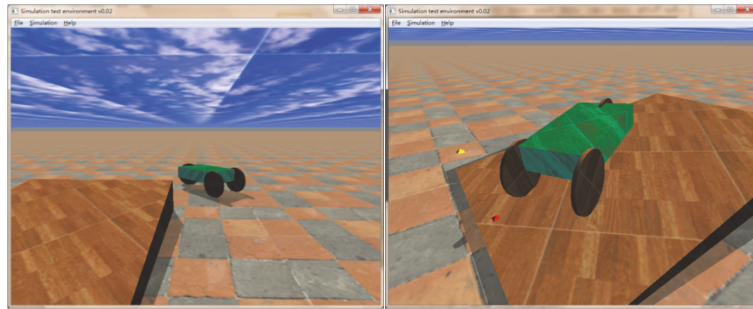


FIGURE 4: Turning and climbing of the vehicle model.

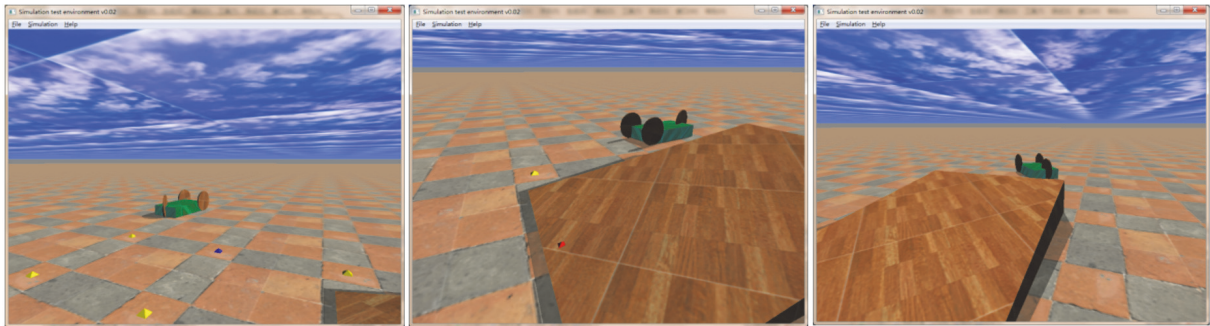


FIGURE 5: The vehicle model has high-speed collision with obstacle and rollover, due to high-speed turning and rushing out of the ramp.

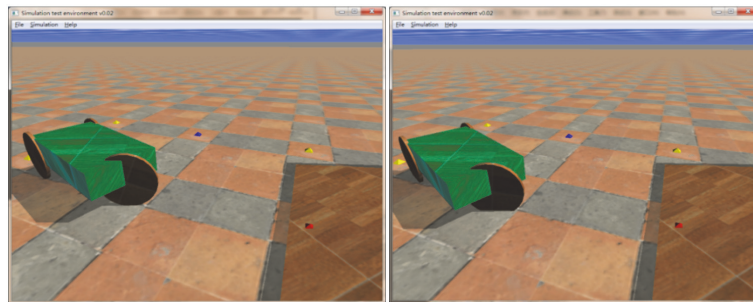


FIGURE 6: Comparison between preadjusted and adjusted penetration effects.

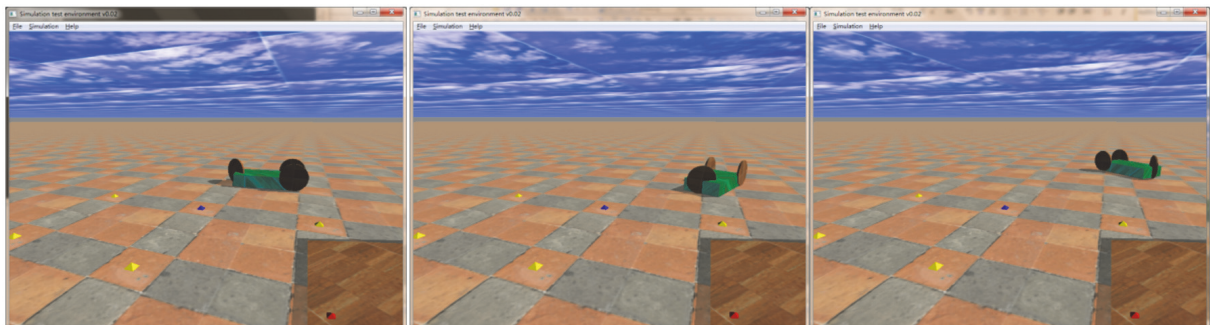


FIGURE 7: Horizontal movement.

to implement the ODE-based character animation simulation; and carrying out the test and analysis with different parameters.

The focus for next research includes the following: from the vehicle motion model upgrading to the more complex human body motion model, simulating the more complex

and precise movements of human body; establishing different controllers for different character models and motion states and implementing more efficient and more realistic simulation through more intelligent controllers; enabling the natural switch-off states of character through animation blending technology.

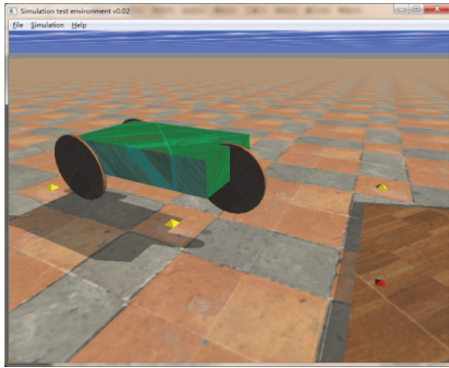


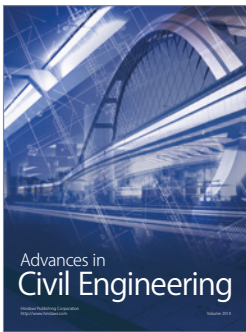
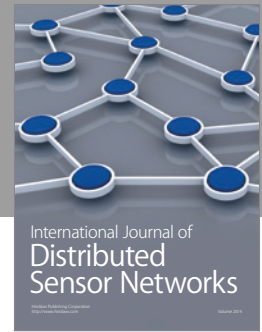
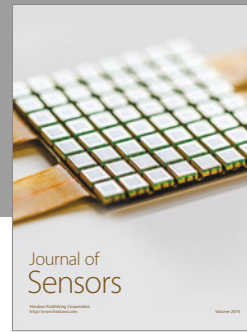
FIGURE 8: Vehicle suspension.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## References

- [1] B. Dai, *Research and Implementation of Role Based Animation Simulation*, University of Electronic Science and Technology of China, 2012.
- [2] C. K. Liu, A. Hertzmann, and Z. Popović, "Composition of complex optimal multi-character motions," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '06)*, pp. 215–222, ACM, Vienna, Austria, September 2006.
- [3] H. Duan, *Research and Application of Key Technologies of Physics Engine in Virtual Reality*, Shandong University of Science and Technology, 2010.
- [4] <http://baike.baidu.com/view/85895.htm>.
- [5] S. Gao, *Design and Implementation of Driving Simulation System Based on OGRE and ODE*, Wuhan University of Technology, 2006.
- [6] V. B. Zordan and J. K. Hodgins, "Motion capture-driven simulations that hit and react," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '02)*, pp. 89–96, ACM, July 2002.
- [7] Z. Wang, H. Jiazhen, and H. Yang, "Summary of research on the collision detection problem," *Journal of Software*, no. 5, 1999.
- [8] T. Larsson and T. A. Moiler, "Collision detection for continuously deforming bodies," in *Proceedings of the Eurographics*, 2001.
- [9] Z. Popović, "Controlling physics in realistic character animation," *Communications of the ACM*, vol. 43, no. 7, pp. 51–58, 2000.
- [10] M. Zhang, *Research on Dynamic Fault Diagnosis Method for Vehicle Mounted Track*, North Central University, 2014.
- [11] T. Zhang, *Research on Automobile Virtual Driving Training System Based on Quest 3D*, Henan Polytechnic University, 2012.
- [12] X. Yang, *Improvement and Optimization of Collision Detection Module for Space Desktop System*, Capital Normal University, 2011.
- [13] T. Wang, "Adaptive stochastic collision detection between deformable objects using particle swarm optimization," in *Proceedings of the EvoWorkshops (EvoIASP '06)*, 2006.
- [14] F. A. Madera, A. M. Day, and S. D. Laycock, "A hybrid bounding volume algorithm to detect collisions between deformable objects," in *Proceedings of the 2nd International Conferences on Advances in Computer-Human Interactions (ACHI '09)*, pp. 136–141, IEEE, Cancún, Mexico, February 2009.
- [15] Y. Zou, G. Ding, M. Xu, and Y. He, "Overview of real time collision detection algorithm," *Computer Application Research*, no. 1, 2008.
- [16] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-DOPs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 4, no. 1, pp. 21–36, 1998.
- [17] R. Smith, "Open Dynamics Engine v0.5 User Guide," 2006, <http://www.ode.org/ode-latest-userguide.html>.
- [18] T. Geijtenbeek and N. Pronost, "Interactive Character Animation using Simulated Physics," *Games and Virtual Worlds*, 2011.
- [19] M. Wang, *Research and Practice of Virtual Reality Art in Engineering*, Harbin Engineering University, 2010.
- [20] P. Jiménez, F. Thomas, and C. Torras, "3D collision detection: A survey," *Computers & Graphics*, vol. 25, no. 2, pp. 269–285, 2001.



**Hindawi**

Submit your manuscripts at  
<https://www.hindawi.com>

