Dynamics Simulation Method of 3D Virtual Motion for Vehicle Based on GTENG Engine

Weijun Chu, Xiaohui He and Qing Liu
Army Engineering University of PLA, Nanjing, Jiangsu, 210007, China
Email: chuwj2008@qq.com

Abstract. The self-developed GTENG engine technology is a complete set of real-time 3D engine based on Microsoft Windows, and the underlying interface is for above directx9.0 version. GTENG engine adopts real-time computer graphics algorithm, including computer graphics and highly optimized physical simulation algorithm set. This paper introduced two ways of using CVehicle of GTENG Engine to simulate vehicle motion, and illustrated the method of defining the amount and number of wheels and setting the scene node pointer associated with the wheel in initialization of vehicle. Further, we discussed setting their coordinates and determining the relation of nodes combined with 3D modeling, and gave the methods of setting other parameters including mass, traction, steering torque, wheel radius, shaft distance, drag, frictional factor and wheel number. At last, we analysed relevant dynamics simulation functions and their calculation in CVehicle to control the motion state of vehicle. The application shows that the simulation method of using CVehicle in GTENG Engine can be spread to write the other instances of dynamics simulation.

1. Introduction
Rigid body dynamics is the basis of simulation. Rigid body mechanics can be used to simulate almost anything in the real world, but there is a balance between computational efficiency and simulation accuracy. Kinematics class of GTENG engines is rigid body kinematics class, namely CKinematics. CKinematics calculates the motion of an object by using linear acceleration and angular velocity. In other words, to use CKinematics, the acceleration of an object needs to be set first, and then the direction and position of an object can be calculated by an integral function.

We can write various examples that use dynamics for simulation. To simplify, we can only use kinematics when there is no collision with other objects, and add dynamics when the kinematics cannot achieve the real effect. GTENG engine provides CVehicle for simulating vehicle objects such as cars, trucks and engineering equipment.

2. Initialization of Vehicle
2.1. CVehicle (Class of Vehicle)
CVehicle simulates vehicle by the mixed use of kinematics and dynamics. To import CVehicle to the program, we can either inherit from CVehicle or use CVehicle directly in the program. Although the using ways are different, the steps are basically similar. The second way is introduced here.

To use the CVehicle, the program needs include the header file:

```
#include "Vehicle."
```

In the corresponding position of the program, such as establishing its own CCar, the instance of CVehicle is defined:

```
CVehicle m_car;
```
2.2. The Wheels in CVehicle and Setting Their Coordinates

To add the initialization code of CVehicle instance in the initialization function of the program, the first step is to set the scene node pointer associated with the wheel:

```cpp
m_car.m_pWheel[0] = this->FindSceneNodeByUnicode("tyre 01");
```

CVehicle supports up to 8 wheels, and you can redefine the maximum number of wheels supported in the header file of class:

```cpp
#define NUM_WHELL 8
```

These wheels do not participate in calculation unless they are associated with the actual node. Top view of the number of 8 wheels is shown in figure 1.

![Figure 1. The number of 8 wheels](image)

The role of each wheel is different, of which four wheels (wheel 0 to 3) are related to the vehicle chassis, called power wheel, the other wheels (wheel 4 to 7) do passive motion, called passive wheel. If the vehicle with more than 4 wheels is to be simulated, the driving simulation can be carried out only with power wheel, while the other wheels move passively and do not participate in dynamics calculation. To simulate 3 wheels car, CVehicle needs to be modified to reduce one power wheel.

For simplicity, the wheels of CVehicle actually do three things:

- Simulation wheel. The wheel rotates around Y-axis of the local coordinate.
- Simulation guide device. When the wheel turns, it rotates around Z axis of the local coordinate.
- Simulated suspension. The local coordinate system is synchronized with the chassis.

The reason for these is that wheels should rotate around local Y-axis, which causes X-axis and Z-axis to change. When the wheel is turning, it needs a firm Z-axis of the local coordinate which pointing to upward direction. If Z-axis changes, the rotation calculation result using this coordinate axis is indeterminate. Physically, wheel, steering and suspension are three kinds of objects, so we use three coordinate systems to simulate three systems, namely suspension, steering and wheel.

2.3. Determining the Relation of Nodes

When modeling, a group is set up including all parts of a wheel, and the group becomes the father node of the wheel, which is used as a steering node. Then this node is set up a group which becomes the father node of the steering node, and it is used as a suspension node. All suspension nodes are set as child nodes of the chassis (figure 2).
Figure 2. The relation of nodes

In CVehicle, steering calculation is directly used by the pointer of the father node of a wheel, and suspension calculation is used by the pointer of the grandfather node of a wheel.

\( m\_\text{car}.m\_pBody = \text{this->FindSceneNodeByUnicode(" Car body ");} \)

2.4. Setting Other Parameters

Other parameters include mass, traction, steering torque, wheel radius, shaft distance, drag, frictional factor and wheel number.

Table 1. Examples of setting other parameters

| Parameters       | Format of sentence                                      |
|------------------|--------------------------------------------------------|
| Mass             | \( m\_\text{car}.\text{SetBodyMass}(1.0\ f); \)       |
| Traction         | \( m\_\text{car}m\_vFEngineConstant.Set(6000.0f, 0.0f, 0.0f); \) |
| Steering torque  | \( m\_\text{car}m\_vTorqueConstant.Set(0.0\ f, 0.0f, 6000.0f); \) |
| Wheel radius     | \( m\_\text{car}m\_fWheelRadius = 488.58f/2.0f; \)     |
| Shaft distance   | \( m\_\text{car}m\_fShaftDistance = 1916.0f; \)       |
| Drag             | \( m\_\text{car}m\_fDrag = 0.1f; \)                   |
| Frictional factor| \( m\_\text{car}m\_fRr = 2.0f; \)                     |
| Wheel number     | \( m\_\text{car}.m\_uRealWheelNum = 6; \)             |

Note:
- Mass is the total mass of a vehicle.
- The next step is to set the traction force of the vehicle. This force can be used as a constant, as long as the wheel is being pulled. This vector can be used as the traction force, and it can be changed at any time during operation according to conditions.
- Torque is used to turn wheels. It is not the torque that pulls the vehicle. The vehicle uses the traction force as the power when it is moving, instead of the torque that drives wheels to roll.
- The shaft distance of front and rear wheels is used to calculate the rotation radius of car body.
- The default is eight wheels in CVehicle, which is the maximum number of wheels.

To set the associated scene pointer is mainly used for collision detection, because vehicles need to interact with objects in the scene, such as terrain.

\( m\_\text{car}m\_pSceneManager = m\_\text{manager}; \)

The initialization function is called to set the mass and rotation tensors for the objects contained in
the vehicle to calculate the dynamics and kinematics. 

\( m_{\text{car}}. \text{Initialize}(); \)

3. 3D Modeling

The first step of using CVehicle is not to write a program, but to build 3D models. CVehicle has a few requirements for 3D model, and only by meeting these requirements can 3D model be easily connected with code.

A vehicle is composed of chassis, wheel system, suspension system and so on, and wheel system and suspension system can be simplified further as a system, because they are closely linked.

3.1. Chassis

Chassis is the root node of vehicle system, and wheel system is its child node. Other parts of the vehicle, such as cab, tank on truck, toolbox, etc., are rigidly connected to the chassis and are as child nodes of the chassis.

Chassis coordinate axis is shown in figure 3. X-axis points to the forward direction of the vehicle. Y axis is the rotation axis of the vehicle's pitching. Z-axis points to the upward direction of the vehicle.

![Figure 3. Chassis and its coordinate](image)

The relation between the right-handed coordinate system in 3DS MAX and the left-handed coordinate system in the GTENG engine is that Y-axis and Z-axis interchange. The origin of the coordinates of the chassis is also very important. If the origin is in the front or rear of the vehicle, the vehicle steering will drift in the rear or front of the vehicle. So it is best to set the origin in the middle of the foremost wheels and the last wheels.

3.2. Wheel System

The wheel system consists of up to eight wheels, suspension nodes, and steering nodes. The axes of each wheel are aligned with the chassis, as shown in figure 3.

4. Simulation of Vehicle Motion

Dynamics simulation is necessary to operate integration every frame for the object. For example, there is such a function in simulation Class: \( \text{UpdatePerFrame}() \)

This function is called by the engine on each frame update, and an instance of CVehicle can be integrated within this function:

\( m_{\text{car}}. \text{Step} (m_{\text{manager}} -> \text{GetElapsedTime}()); \)

Above parameter is the time that the system elapse on the frame, calculated in milliseconds.

The vehicle moves forward using the following function. When the parameter go is true for driving forward and false for reverse. This function applies traction to the vehicle.

\( m_{\text{car}}. \text{Move} (\text{go}); \)

The following function is used for vehicle turning left:

\( m_{\text{car}}. \text{TurnLeft}(); \)

The following function is used for vehicle turning right:
These two functions are responsible for calculating the turning radius of the vehicle and changing the driving direction of the vehicle.

Note: vehicles sometimes have more than two steering wheels. For example, some engineering equipment have four steering wheels. Usually, we use the two wheels in front of the vehicle as steering wheels. If more steering wheels are needed, these wheels can be used as auxiliary steering wheels, as shown below:

\[
m_{\text{car}}.\ m_{\text{uTurnWheel}}[3] = \text{ASSIST}_\text{TURNS}_\text{WHEEL};
\]

Figure 4 shows a simulation example of vehicle motion.

5. Conclusion
CVehicle is introduced to our program for simulating vehicle objects such as cars, trucks and engineering equipment by the mixed use of kinematics and dynamics. We can either inherit from CVehicle or using CVehicle directly in the program. According to above ways of using CVehicle, we have achieved satisfactory simulation results. Also, as underlying technology, GTENG engine has scene management, 3D model rendering and dynamics calculation simulation functions, so it is used on developing overlying tools, custom 3D simulation product and large simulation project.

6. References
[1] Chu W J, He X H, Zhu Z C 2019 Research on the Semi-physical Simulation Maintenance Training System of One Engineering Equipment Advances in Intelligent Systems Research and Engineering vol 168, pp 34–35
[2] Jia L X, Zhu Y H, Zhang J 2017 3D Visual Simulation (Beijing: National Defense Industry Press) pp 91–112
[3] Zhao W, Mao P, Shen F Y 2019 Development Trend of Next Generation 3D Graphics Engines Journal of System Simulation vol 29, pp 2935–2942
[4] Hou X, Li C H, Cao L 2014 Individual Dynamics and Kinematics Simulation in Micro-construction Simulation Journal of System Simulation vol 26, pp 2589–2591
[5] Hong L, Ji B J, Wang Y G 2015 Virtual Human Motion Analysis Software Based on VC Platform Computer Engineering vol 41, pp 71-75
[6] Guo L W, Wang Q, Gao K B, Gu W 2019 Research on Armored Vehicle Motion Simulation Based on Virtual Reality Computer Systems & Applications pp 221–227