Master Thesis

Jian Wu – xcb479@alumni.ku.dk

Deep Contact
Accelerating Rigid Simulation With Convolutional Networks

Supervisor: Kenny Erleben

August 6th 2018

Abstract

This is a master theis from

Contents

References 13

List of Figures

List of Tables

Chapter 1

Introdustion

1.1   Motivation

1.2   Thesis Overview

Chapter 2

Rigid Body Dynamics Simulation

This chapter mainly introduces rigid body simulation to help you understand how computer simulate rigid dynamics based on traditional newton-euler equations. For more details, some contact forces solvers are decribed in this chapter. Afterwards, we will use one of solver to run some simulation and get the image data for the next step, grids-transfer. All the discussion about rigid simulation and contacts solver are based on 2-D view.

2.1 Rigid dynamics Simulation

2.1.1 Simulation Basics

Simulating the motion of a rigid body is almost the same as simulating the motion of a particle, so I will start with partcle simulation. For particle simulation, we let function $x(t)$ describe the particle's location in world space at time t. Then we use $v(t) = \frac{\mathrm{d}}{\mathrm{d}(t)} x(t)$ to denote the velocity of the particle at time t. So, the state of a particle at a time t is the particle's position and velocity. We generalize this concept by defining a state vector $\mathbf{Y}(t)$ for a system: for a single particle,

$$\mathbf{Y}(t) = \begin{pmatrix} x_1(t) \\ v_1(t) \end{pmatrix} \tag{2.1}$$

For a system with n particles, we enlarge $\mathbf{Y}(t)$ to be

$$\mathbf{Y}(t) = \begin{pmatrix} x_1(t) \\ v_1(t) \\ ... \\ x_n(t) \\ v_n(t) \end{pmatrix} \tag{2.2}$$

However, to simulate the motion of particles actually, we need to know one more thing – the forces. $F(t)$ is defined as the force acting on the particle. If the mass of the particle is m, then the changes of $\mathbf{Y}(t)$ will be given by

$$\frac{\mathrm{d}}{\mathrm{d}(t)}\mathbf{Y}(t) = \frac{\mathrm{d}}{\mathrm{d}(t)}\begin{pmatrix} x(t) \\ v(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ F(t)/m \end{pmatrix} \tag{2.3}$$

2.1.2   Rigid Body Concepts

Unlike a particle, a rigid body occupies a volume of space and has a particular shape. Rigid bodies are more complicated, beside translating them, we can rotate them as well. To locate a rigid body, we use $x(t)$ to denote their translation and a rotation matrix $R(t)$ to describe their rotation.

2.1.3   Rigid Body Equations of Motions

Finally, we can covert all concepts we need to define the state $\mathbf{Y}(t)$ for a rigid body.

$$\mathbf{Y}(t) = \begin{pmatrix} x(t) \\ R(t) \\ P(t) \\ L(t) \end{pmatrix} \tag{2.4}$$

Like what is epressed in $\mathbf{Y}(t)$, the state of a rigid body is mainly consist by its position and orientation (describing spatial information), and its linear and angualr momentum(describe velocity information). Since mass M and bodyspace inertia tensor I_{body} are constants, we can the auxiliary quantities $I(t)$, $\omega(t)$ at any given time.

$$v(t) = \frac{P(t)}{M} \qquad I(t) = R(t)I_{body}R(t)^T \qquad \omega(t) = I(t)^{-1}L(t)$$

The derivative $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{Y}(t)$ is

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{Y}(t) = \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} x(t) \\ R(t) \\ Mv(t) \\ L(t) \end{pmatrix} = \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} v(t) \\ \omega(t) * R(t) \\ F(t) \\ \tau(t) \end{pmatrix} \qquad (2.5)$$

Then, we can conclude the simulation algorithm

Data: this text
Result: how to write algorithm with LaTeX2e
initialization;
while *running the simulation world* **do**
> read current;
> **if** *understand* **then**
>> go to next section;
>> current section becomes this one;
>
> **else**
>> go back to the beginning of current section;
>
> **end**

end

Algorithm 1: How to write algorithms

2.2 Contact Forces Solver

2.3 Simulation Results

Chapter 3

Partcle-grid-particle

The basic method for generating training data which is more accessible to learning is that we will map a discrete element method(DEM) into a continuum setting use techniques from smooth particle hydrodynamics. Given a set of bodies δ and a set of contacts between these bodies C.

3.1   Grid-Based method

Traditional rigid motion simulation mainly use particle-based method. However, if we want to replace traditional contact solver with deep learning model, it is hard for cnn model to recognize the original image and do learning. Grid-based methos is a good to transfer original image to a grid-cells and then use

3.2   Smoothed Particle Hydrodynamics

Smoothed particle hydrodynamics (SPH) was invented to simulate nonaxisymmetric phenoma in astrophysis initially. The principal
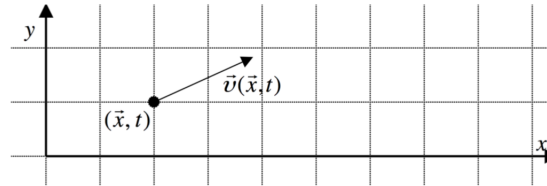

Figure 3.1: Grid description, $retrieved from MIT$ (2011)

idea of SPH is to treat hydrodynamics in a completely mesh-free fashion, in terms of a set of sampling particles. It turns out that the particle presentation of SPH has excellent conservation properties. Energy, linear momentum, angular momentum, mass and velocity.

3.2.1 Fundamentals

At the heart of SPH is a kernel interpolation method which allows any function to be expressed in terms of its values at a set of disordered points - the particles[1]. For ant field $A(\mathbf{r})$, a smoothed interpolated version $A_I(\mathbf{r})$ can be defined by a kernel $W(\mathbf{r}, h)$,

$$A_I(\mathbf{r}) = \int A(\mathbf{r}')W(\|\mathbf{r} - \mathbf{r}'\|, h) \, \mathrm{d}\mathbf{r}' \tag{3.1}$$
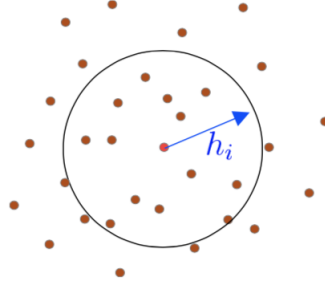

Figure 3.2: Visilaztion of SPH

where the integration is over the entire space, and W is an interpolating kernel with

$$\int W(\|\mathbf{r} - \mathbf{r}'\|, h) \, \mathrm{d}\mathbf{r}' = 1 \tag{3.2}$$

and

$$\lim_{h \to 0} W(\|\mathbf{r} - \mathbf{r}'\|, h) \, \mathrm{d}\mathbf{r}' = \delta(\|\mathbf{r} - \mathbf{r}'\|) \tag{3.3}$$

Normally, we want the kenel to be Non-negative and rotational invariant.

$$W(\|\mathbf{x}_i - \mathbf{x}_j\|, h) = W(\|\mathbf{x}_j - \mathbf{x}_i\|, h) \tag{3.4}$$

$$W(\|\mathbf{r} - \mathbf{r}'\|, h) \geq 0 \tag{3.5}$$

For numerical work, we can use midpoint rule,

$$A_I(\mathbf{x}) \approx A_S(\mathbf{x}) = \sum_i A(\mathbf{x}_i) W(\|\mathbf{x}_i - \mathbf{x}\|, h) \Delta V_i \qquad (3.6)$$

Since $V_i = m_i/\rho_i$

$$A_S(\mathbf{x}) = \sum_i \frac{m_i}{\rho_i} A(\mathbf{x}_i) W(\|\mathbf{x}_i - \mathbf{x}\|, h) \qquad (3.7)$$

The default, gradient and Laplacian of A are:

$$\nabla A_S(\mathbf{x}) = \sum_i \frac{m_i}{\rho_i} A(\mathbf{x}_i) \nabla W(\|\mathbf{x}_i - \mathbf{x}\|, h)$$

$$\nabla^2 A_S(\mathbf{x}) = \sum_i \frac{m_i}{\rho_i} A(\mathbf{x}_i) \nabla^2 W(\|\mathbf{x}_i - \mathbf{x}\|, h) \qquad (3.8)$$

3.2.2 Kernels

Smoothing kernels functions are one of the most important points in SPH. Stability, accurancy and speed of the whole method depends on these fuctions. Different kernels are being used for different purposes. One possibilyty for W is a Gaussian. However, most current SPH implementations are based on kernels with finite support. We mainly introduce gaussian, poly6 and spicky kernel here. And compare the different kernels and their property.

Poly6

The kernel is also known as the 6th degree polynomial kernel.

$$W_{poly6}(\mathbf{r}, h) = \frac{315}{64\pi h^9} \begin{cases} (h^2 - \|\mathbf{r}\|^2)^3 & 0 \leq \|\mathbf{r}\| \leq h \\ 0 & \text{Otherwise} \end{cases} \qquad (3.9)$$

Then, the gradient of this kernel function can be

$$\nabla W_{poly6}(\mathbf{r}, h) = -\frac{945}{32\pi h^9} \begin{cases} \mathbf{r}(h^2 - \|\mathbf{r}\|^2)^2 & 0 \leq \|\mathbf{r}\| \leq h \\ 0 & \text{Otherwise} \end{cases} \qquad (3.10)$$

The laplacian of this kenel can be expressed by,

$$\nabla^2 W_{poly6}(\mathbf{r}, h) = -\frac{945}{16\pi h^9} \begin{cases} (h^2 - \|\mathbf{r}\|^2)(3h^2 - 7\|\mathbf{r}\|^2) & 0 \leq \|\mathbf{r}\| \leq h \\ 0 & \text{Otherwise} \end{cases}$$
$$(3.11)$$

Spicky

The kernel proposed by Desbrum[2]

$$W_{spiky}(\mathbf{r}, h) = \frac{15}{\pi h^6} \begin{cases} (h - \|\mathbf{r}\|)^3 & 0 \le \|\mathbf{r}\| \le h \\ 0 & \text{Otherwise} \end{cases} \tag{3.12}$$

Then, the gradient of spiky kernel can be described by,

$$\nabla W_{spiky}(\mathbf{r}, h) = -\frac{45\mathbf{r}}{\pi h^6 \|\mathbf{r}\|} \begin{cases} (h - \|\mathbf{r}\|)^2 & 0 \le \|\mathbf{r}\| \le h \\ 0 & \text{Otherwise} \end{cases} \tag{3.13}$$

The laplacian of spiky can be expressed by,

$$\nabla^2 W_{spiky}(\mathbf{r}, h) = \frac{90}{\pi h^6} \begin{cases} h - \|\mathbf{r}\| & 0 \le \|\mathbf{r}\| \le h \\ 0 & \text{Otherwise} \end{cases} \tag{3.14}$$
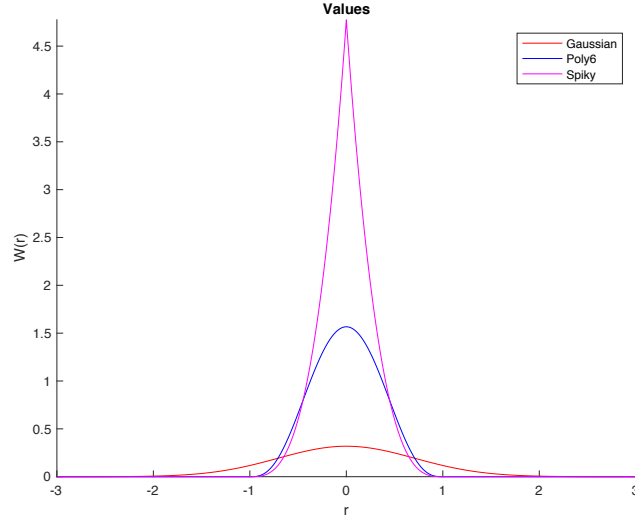

Figure 3.3: Comparation of different kernels, we set smoothing length $h = 1$ here.
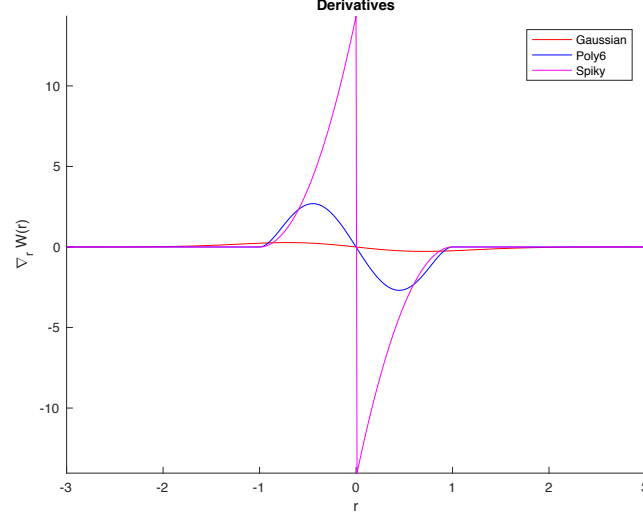
Figure 3.4: Comparation of gradient of different kernels, we set $h = 1$ here.

3.2.3 Grid size and smoothing length

The grid should be also fine enough to capture the variation in our simulation. In our case, it is reasonable to have a grid fine enough such that no two contact points are mapped into the same cell.

Smoothing length, h, is one of the most important parameters that affects the whole SPH method by changing the kernel value results abd neighbor searching results. Too small or too big values might cause lose essencail information in the simulation.

3.2.4 Neignbor Search

Neighbor search is one of the most crucial procedures in SPH method considersing all interpolation equations, $A(\mathbf{r})$, needs the neighbor list for every particle (refer to equation 3.8). A naive neighbor searching approach would end up with a complexity of $\Omega(n^2)$. The complexity is not good enough since it is impossoble to reach any interactive speed when the particle count increses. With an efficient nearest neighbor searching(NNS) algorithm, it is possible to have a signifi-

9

cant performance increase since it is the most time consuming procedure in SPH computation,

k − d Tree

3.3   Grid to particle

3.3.1   interpolation

3.4   Conclution

Chapter 4

Deep Learning For Simulation

4.1 Convolutional Neural Networks

4.2 CNN Constructure

4.3 Traing Results

4.4 Simulation based on Trained model

Chapter 5

Impletement

References

[1] J. J. Monaghan, "Smoothed particle hydrodynamics", *Annual review of astronomy and astrophysics*, vol. 30, no. 1, pp. 543–574, 1992.

[2] M. Desbrun and M.-P. Gascuel, "Smoothed particles: A new paradigm for animating highly deformable bodies", in *Computer Animation and Simulation'96*, Springer, 1996, pp. 61–76.