Technical Report OUCS-2004-01

Spherical Springs

Author:
Alexis Angelidis
Department of Computer Science

Status: unpublished

Spherical Springs

Category: Research

Abstract

The unit sphere is a space where geodesic distances can be defined easily. Dynamic elements can therefore be efficiently animated in it, provided that they are represented adequately. We present basic elements for dynamic simulation in the unit sphere. The presented formulas which apply to quaternions parallel euclidean physics.

1 Introduction

One of the many ways of using spherical coordinates is for texturing shapes in computer graphics. Traditionally, the object is first build, and the texture is then added artificially by projecting the points on the surface of a sphere, regardless of issues like homogeneous texture density or surface folds in texture space. In this paper, we focus on performing relaxation of the texture coordinates in texture space, considering the object is homeomorphic to a sphere. The first step to do this is to introduce spring in the spherical texture space. Because of its bend nature, it is undesirable to use linear distance spring between to points in texture space: the points would have to be reprojected on the sphere very often, and the behavior of long springs would be unpredictable. In this paper, we formulate spherical springs, which we believe is the best way to define springs on the unit sphere.

2 Preliminaries

An excellent reference on quaternion is [Dam et al. 1998], which we paraphrase briefly for the understanding of this paper.

Quaternion space H is a 4 dimensional space. A quaternion is noted $q = (s, v)$, where $s \in \mathbb{R}$ and $v \in \mathbb{R}^3$. The *addition* of two quaternions is obtained by adding the corresponding components. The *product* of two quaternions is:

$$(s, v) * (s', v') = (ss' - vv', sv' + s'v + v \times v')$$

The most remarkable elements are the unit quaternions H_1, which can be written $q = (\cos(\theta), \sin(\theta)n)$, and whose norm $s^2 + v^2$ is equal to 1. The set of unit quaternions constitute a unit sphere in four-dimensional space. The *logarithm* of quaternion q is $p = (0, \theta n)$, and the *exponential* of p is q. From the above definitions arise the *exponentiation* of $q \in H_1$:

$$q^t = \exp^{t \log q}$$

A 3D point can be represented in quaternion notation: $\mathrm{x} = (0, (x, y, z))$. Note the for clarity, we note $\cdot$ the dot product in opposition with $*$ the quaternion product. Unit quaternion q doesn't only represent a point on the unit 4D sphere, but also a rotation of angle $\theta/2$ around axis v. The rotation of x around q is:

$$q * \mathrm{x} * \bar{q}$$

where $\bar{q} = (s, -v)$ is the *conjugate* of q.

In this paper, the operation we perform on quaternion is inspired from [Alexa 2002]. We use the logarithmic space to perform linear combination of unit quaternions. The advantage of using unit quaternions instead of matrices is the availability of closed forms formulas for the log and exp. In the following sections, we use greedily the fact that $\exp^{\log p + \log q}$ is a commutative combination of quaternion p and q.

3 Spherical Forces

Given a scalar condition $C(x)$ which we want to be zero, C gives rise to a force quaternion f, which expression parallels the one for euclidean forces [Witkin and Baraff 2001], and is obtained from expression

$$f = (\dot{\nabla} C)^{-kC} \tag{1}$$

where $\dot{\nabla}$ denotes the spherical gradient quaternion(see appendix A), and where k is a stiffness constant of our choice. The logarithm has a more familiar expression:

$$\log f = -kC \log(\dot{\nabla} C) \tag{2}$$

From now on, all forces will be expressed through their logarithm: this yields to more efficient formulas since the log of a quaternion can be handled as a mere three dimensional vector. Moreover it simplifies the writing and makes more obvious the parallelism with euclidean physics.

3.1 Angle Spring

The angle spring aims at keeping a constant angle between two points. Consider two mass points x_i and x_j, connected with a spherical spring with rest angle θ_{rest} and stiffness coefficient $k_s \in \mathbb{R}^+$ (see figure 4). The condition we use is:

$$C(\mathrm{x}_i, \mathrm{x}_j) = \theta_{ij} - \theta_{rest}$$

Applying equation 2, the log of the force applying on p_i, $f_{j \to i} \in H_1$ is simply the quaternion:

$$\log f_{j \to i} = \theta r \tag{3}$$

$$\text{where} \quad \begin{array}{rcl} \theta & = & \frac{k_s}{2}(\arccos(\mathrm{x}_i \cdot \mathrm{x}_j) - \theta_{rest}) \\ r & = & \frac{\mathrm{x}_i \times \mathrm{x}_j}{||\mathrm{x}_i \times \mathrm{x}_j||} \end{array}$$

Note that by using the log instead of quaternion directly, the stiffness k_s won't suffer from the periodicity of cos and sin, thus large forces will be handled better. However, keep in mind that a large stiffness may make explode the physical integration (though the particles will always stay on the sphere).

3.2 Solid angle spring

The solid angle spring aims at keeping a constant solid angle for a spherical triangle. Since three arcs define two spherical triangles on either side of the sphere, this force is used to keep constant the area of the spherical triangle whose vertices are clockwise oriented when observed from outside the sphere. The effect is to flip smoothly the triangle if the vertices are wrongly oriented. Consider three mass points x_i,

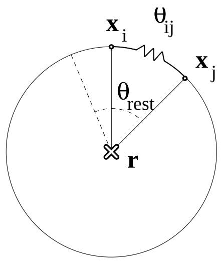

Figure 1: Spherical angle spring.

x_j and x_k. Let Θ_{ijk} be the solid angle of the triangle ijk, and let Θ_{rest} be the rest solid angle. The condition we use is:

$$C(\mathrm{x}_i, \mathrm{x}_j, \mathrm{x}_k) = \Theta_{ijk} - \Theta_{rest}$$

An efficient expression of Θ_{ijk} is given by [Oosterom and Strackee 1983], which we can further simplify for points on the unit sphere:

$$\Theta_{ijk} = 2\arctan\left(\frac{\mathrm{x}_0 \cdot (\mathrm{x}_1 \times \mathrm{x}_2)}{1 + \mathrm{x}_0 \cdot \mathrm{x}_1 + \mathrm{x}_1 \cdot \mathrm{x}_2 + \mathrm{x}_2 \cdot \mathrm{x}_0}\right)$$

Let us call α and β the numerator and denominator of the fraction inside the arctan. As in [Oosterom and Strackee 1983], we use the C function atan2 with α and β to compute the arctan. The euclidian gradient of the solid angle is:

$$\frac{1}{2}\nabla\Theta_{ijk} = \frac{\beta(\mathrm{x}_1 \times \mathrm{x}_2) - \alpha(\mathrm{x}_1 + \mathrm{x}_2)}{\alpha^2 + \beta^2}$$

The logarithm of the spherical gradient is (see the definition of the spherical gradient in appendix A):

$$\log \dot{\nabla}\Theta_{ijk} = (\frac{1}{2}\nabla\Theta_{ijk} \cdot x)y - (\frac{1}{2}\nabla\Theta_{ijk} \cdot y)x$$

where x and y are two arbitrary orthogonal tangent vectors. The logarithm of the force is:

$$\log f_{jk\to i} = -k_s(\Theta_{ijk} - \Theta_{rest}) \log(\dot{\nabla}\Theta_{ijk}) \qquad (4)$$

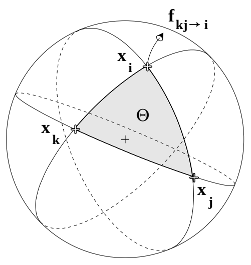

Figure 2: Solid angle spring.

3.3 Oriented-Angle Spring

3.4 Particle damping

The damping of particle p_i is defined using its speed $v_i \in H_1$, which is defined in section 4. Its effect is to keep in place the particle. Let $k_d \in \mathbb{R}^+$ be the damping coefficient. The damping is:

$$\log d_i = -k_d \log v_i$$

where $k_d \in [0, \frac{m}{d_t}]$ (mass m and time step d_t are defined later on). However, as remarked [Baraff and Witkin 1998], this damping is a simple viscous function that dissipates kinetic energy independently from the type of motion.

3.5 Spring damping

To define the damping function in terms of the condition C, we propose:

$$\log d_i = -k_d(\log \dot{\nabla}C \cdot \log v_i) \log \dot{\nabla}C \qquad (5)$$

3.6 Combining forces

To compute the overall force applied on p_i, we combine all the forces in logarithmic space:

$$F_i = \exp^{\sum_j log(f_{j\to i}) - k_d log(v_i)}$$

4 Physical integration

For computing efficiency purposes, all the computations are done in logarithmic space, thus the exponential is computed only once for each particle at each time step. The combination of quaternions has to be done carefully, depending when they represent vectors or positions.

Acceleration: To use an explicit euler integration, we propose an adaptation of the second law of motion for quaternions:

$$\log a_i = \frac{1}{m}\sum_j log(f_{j\to i})$$

Speed: At the begining of the simulation, the speed's log is initialized to the quaternion $(0, (0, 0, 0))$. Given a previous speed $\log v_i$, the new speed's log at next time step d_t adds up a portion d_t of the acceleration. We choose the following formula for computing the new speed:

$$\log v_i' = \log v_i + d_t \log a_i$$

Another possibility for computing v_i' would be to use $v_i' = a_i^{d_t} v_i$, but this would yield to a more time consuming formula.

Position The new position is obtained by rotating the point using the speed:

$$p_i' = v_i^{d_t} p_i v_i^{-d_t}$$

where
$$\begin{aligned} v_i^{d_t} &= \exp^{d_t \log v_i} \\ v_i^{-d_t} &= v_i^{d_t *} \text{, the conjugate.} \end{aligned}$$

5 Summary

All the computations are done in logarithmic space, thus are quite efficient. For each particle:

- compute the current acceleration (see section **??** for computing forces):

$$\log a_i = \frac{1}{m}\sum_j \log f_{j\to i}$$

- compute the new speed:

$$\log v_i' = \log v_i + d_t \log a_i$$

- update the position:

$$d_v = \exp^{d_t \log v_i'}$$

$$p(t + d_t) = d_v * p(t) * \bar{d}_v$$

6 Results

The result of implementing the springs is shown in figure 4. The motion integration is done with a simple explicit euler [Witkin and Baraff 2001]. One of the advantage of using spherical springs on the unit sphere instead of using euclidian springs is when the system is unstable; instead of sending the particles to infinity, they stay on the sphere.

A temporary problem occurs when the triangle flips with the solid angle spring: at the transition state, the vertices can gain a extremely large speed because of the arc-spring. Using the solid angle in the arc-spring definition would correct that.

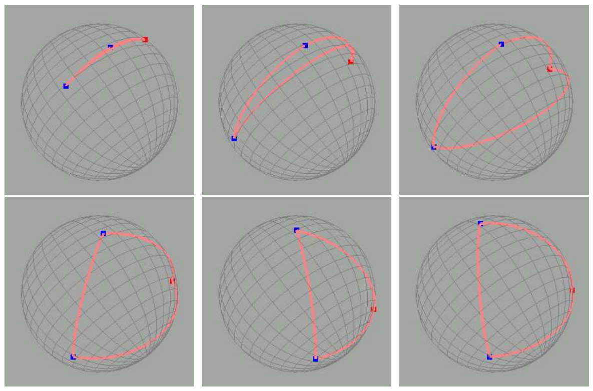
Figure 3: Animation of three spherical springs with rest angles equal to $\pi/2$. At rest (sixth picture), the triangle is an octant of the sphere.

A Spherical Gradient

The spherical gradient is the equivalent of the straight vector euclidian gradient, but constrained to the sphere; thus it is quaternion.

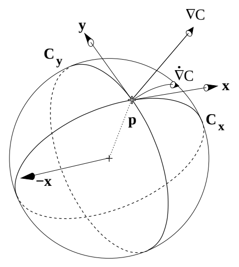

Figure 4: Spherical gradient.

Let us consider two great circles C_x and C_y that are orthogonal, defined by the tangents x and y, and a point p at which the great circles intersect, such that $p = x \times y$. The quaternion $(0, y)$ defines a rotation of angle π of p in direction x along great circle C_x, and the quaternion $(0, -x)$ defines a rotation of angle π of p in direction y along great circle C_y. In those terms, the spherical gradient of a function C is the quaternion:

$$\dot{\nabla} C = \exp^{(0, \frac{1}{2}((\nabla C \cdot x)y - (\nabla C \cdot y)x))} \tag{6}$$

In logarithm space, it has a more familiar look:

$$2 \log \dot{\nabla} C = (\nabla C \cdot x)y - (\nabla C \cdot y)x \tag{7}$$

The disapearance of $\nabla C \cdot z$ can be observed, which is aligned with p. In spherical coordinates, the reader can verify that it disappears too for the parametric gradient.

References

ALEXA, M. 2002. Linear combination of transformations. In *Proceedings of SIGGRAPH'02*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 380–387.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proceedings of SIGGRAPH'98*, Computer Graphics Proceedings, Annual Conference Series, ACM, 46–54.

DAM, E. B., KOCH, M., AND LILLHOLM., M. 1998. Quaternions, interpolation and animation. Tech. rep., Institute of Computer Science (DIKU) University of Copenhagen. http://www.diku.dk/forskning/image/teaching/Studentprojects/Quaternion/.

OOSTEROM, A. V., AND STRACKEE, J. 1983. The solid angle of a plane triangle. In *IEEE Transactions on Biomedical Engineering*, vol. 30, 125–126.

WITKIN, A., AND BARAFF, D., 2001. Physically based modeling, online siggraph 2001 course notes. http://www.pixar.com/companyinfo/research/pbm2001/.