Maria Skublewska-Paszkowska[1]

COMPARISON OF ARM ANIMATION

3D animation is a complex task which involves many methods used to create a moving object. The article presents two techniques used to represent angular positions in 3D space: Euler Angles and Quaternions. Two arm animations were created using these two methods and the obtained results were evaluated for an application in real animation process. The animations were written in objective C++ computer program and then written to C3D files.

Keywords: *Euler Angles, Quaternion, object animation.*

Марія Скублевска-Пашковська

ПОРІВНЯННЯ МЕТОДІВ АНІМАЦІЇ РУХУ РУКИ

У статті показано, що 3D анімація є складним завданням, яке включає в себе безліч методів, використовуваних для створення рухомого об'єкту. Представлено дві технології, які використовуються для представлення дугових координат у 3D-просторі: кути Ейлера і кватерніони. Дві анімації руки було створено з використанням цих методів і отримані результати було застосовано в реальному процесі анімації. Анімації було написано на С++ в цільовій комп'ютерній програмі і потім збережено як файли C3D.

Ключові слова: *кути Ейлера, кватерніон, анімація об'єктів.*

Рис. 2. Форм. 9. Літ. 10.

Мария Скублевска-Пашковска

СРАВНЕНИЕ МЕТОДОВ АНИМАЦИИ ДВИЖЕНИЯ РУКИ

В статье показано, что 3D анимация является сложной задачей, которая включает в себя множество методов, используемых для создания движущегося объекта. Представлены две технологии, используемые для представления дуговых координат в 3D-пространстве: углы Эйлера и кватернионы. Две анимации руки были созданы с использованием этих методов и полученные результаты были применены в реальном процессе анимации. Анимации были написаны на С++ в целевой компьютерной программе и затем сохранены как файлы C3D.

Ключевые слова: *углы Эйлера, кватернион, анимация объектов.*

Introduction

3D animation is a complex process that makes objects moving. There are many methods and programs dedicated to create various kinds of animations. However, there are 4 main methods that are used to represent angular position in 3D space: fixed axis, Euler Angles, rotation and angle and Quaternions (Parent, 2008). They all can be used to rotate 3D objects and thus create arm animation. The article presents two chosen algorithms: Euler Angle and Quaternions. The aim of the article is to present the selected arm moves and compare them due to further usage in human animation. The economic aspect of using these two algorithms is also analysed. The program creates arm animation based on 3 joints: wrist, elbow and shoulder. Each joint is represented as one dot placed in the 3D space. A spot is represented by 3 axis coordinates: X, Y and Z. The right-handed system is used to animate the objects. The animation parameters, such as frame number, point number in a frame and the frame frequency are set in the program. The created animations are written to C3D files in order to enable the visualization of animations in many specific programs such as

[1] PhD, Assistant Professor, Lublin University of Technology, Poland.

Mokka. The visualization is represented with the segments concerning the proper pair of moving points that are created with a computer program.

Economic aspect of animation

Animation is a process of creation of moving objects. In computer graphics and animation the matrixes 4x4 are often used because operations using these matrices are simple and fast, so very economical. There are many ways of representing the angular positions and creating rotations. There are a few aspects that should be taken into consideration: the choice of a suitable method of moving objects and the possibility of creating animation in real time which reflects real moves. The first step is very important and it should be well analysed. The incorrect choice of a method for animation can result in huge loss of time, effort and money. The first step is also connected with other issues. The animation has to be in real time, which means that all computations have to be quick. That is why a program should be written in an effective way. The final aspect concerns the creation of animation which reflect the reality, e.g. human moves. Not all the available methods are proper. Sometimes, during the animation creation the problem can arise which effects the whole program or system. The removal of such a problem can be very expensive and can cause changes of basic algorithms.

There are many intuitive and simple ways of rotating objects in animation, e.g.: around fixed axis and Euler angles. However not always the simplest way is the most effective and economical. Their major inconvenience is the possibility of obtaining a phenomenon called "gimbal lock" (Eckel, 2002; Parent, 2008). It causes the natural move be disturbed, and not all object positions can be obtained after object rotation. That is why it is wise to consider more complex and sophisticated methods which are much more reliable. One of this methods is the quaternion representation. Although it is more complex than Euler angles, it allows fulfiling all the described economic animation aspects and is much more economical.

Methods for representing angular position

The choice of the best method for representing angular and spatial object position is one of the initiative steps before animation. The interpolating methods between the frames is also a very important aspect that affects the created animation (Smolka, Skublewska-Paszkowska, 2013). This article focuses on two methods: Euler angles and quaternions.

Euler angles

In this representation axes of rotation are local of coordinate system. They rotate along with the object. There are 3 angles: roll, pitch and yaw (Parent, 2008). The final rotation consists of the rotation around the 3 axis according the established order (e.g. x-y-z). Rotations are represented by 3 matrices: $R_x(\alpha)$, $R_y(\beta)$ and $R_z(\gamma)$. Firstly, the object is rotated around x axis, then the object is rotated around y axis (rotated local coordinate) and finally around the z axis. The points are represented as a vertical vector. They are multiplied by transformation matrices (Parent, 2008). The object rotation can be also represented in the global coordinate system which is shown by equation 1 (Parent, 2008). The transformation consists of two rotation: firstly around axis x by an angle α and secondly around the changed local coordinate system axis y by an angle β.

$$R'_y(\beta)R_x(\alpha)=R_x(\alpha)R_y(\beta)R_x(-\alpha)R_x(\alpha)=R_x(\alpha)R_y(\beta) \quad (1)$$

The third rotation around the axis z ($R''_z(\gamma)$) can be done after the above described two rotations. It can be performed as it is presented in equation 2 (Parent, 2008).

$$R''_z(\gamma)R'_y(\beta)R_x(\alpha)=R_x(\alpha)R_y(\beta)R_z(\gamma)R_y(-\beta)R_x(-\alpha)R_x(\alpha)R_y(\beta)=R_x(\alpha)R_y(\beta)R_z(\gamma) \quad (2)$$

It seems that the transformation with Euler angles around axis z-y-x are equal to the rotation around the fixed axis x-y-z (Parent, 2008).

The Euler angles are used because transformations are mathematically easy and quick. In animation there is a serious problem named "gimbal lock" which causes the loss of the one degree of freedom in 3D space (Parent, 2008). It occurs during the interpolation of an angular position. That is the reason why this method has to be replaced by others. One of the possible methods are quaternions.

Quaternions

Quaternion is represented by 4 values $[s,x,y,z]$. It can be expressed as a pair $[s,v]$, where s stands for a scalar and v for a vector consists of 3 coordinates (Hanson, 2006; Parent, 2008). It can be used both to create rotations and to interpolation. Quaternions can be adopted in the interpolation of angular positions. They are also often used to assemble various rotations to one transformation.

There are basic mathematical operations done with quaternions. They are adding and multiplication. There are also such operations like the dot product of vectors or vector product. Adding two quaternions is done as summing 4 numbers which is presented in equation 3 (Parent, 2008).

$$[s_1,v_1]+[s_2,v_2]=[s_1+s_2,v_1+v_2] \quad (3)$$

The operation of multiplying 2 quaternions is presented in equation 4 (Parent, 2008). It is cumulative but not convertible.

$$[s_1,v_1][s_2,v_2]=[s_1s_2-v_1\times v_2,s_1v_2+s_2v_1+v_1\times v_2] \quad (4)$$

Inverse of the quaternion (q^{-1}) is presented in equation 5 (Parent, 2008). It changes the return vector on the opposite and then divides all 4 quaternion elements by square it length.

$$q^{-1}=\left(1/\|q\|^2\right)[s,-v], \quad (5)$$

where $\|q\|$ is computed as $\sqrt{s^2+x^2+y^2+z^2}$ (Parent, 2008). The multiplication quaternion by its inversion gives the single quaternion ([1,0,0,0])(Parent, 2008).

All rotations can be represented with the use of quaternions. The rotation by an θ angle around the axis designated by the single vector $v=[x,y,z]$ can be represented by a quaternion given by the equation 6 (Parent, 2008).

$$q=[\cos(\theta/2),\sin(\theta/2)[x,y,z]] \quad (6)$$

The rotation by an θ angle is the same rotation by an -θ angle around axis opposite oriented. The q quaternion given as $q=[s,v]$ and the opposite quaternion given as $-q=[-s,-v]$ represents the same rotation (Parent, 2008).

In order to rotate a vector w with the use of quaternion, the new quaternion has to be created $[0,w]$ which represents vector and the second quaternion q representing the rotation. The new vector (w') after the rotation is represented by the equation 7 (Parent, 2008).

$$[0,w'] = q[0,w]q^{-1} \tag{7}$$

Several rotations are represented as multiplication of the quaternions. For example, for two vector rotations which are given by two quaternions p and q the final vector can be described as in the equation 8 (Parent, 2008). The w vector is represented by p quaternion.

$$[0,w''] = q\left(p[0,w]p^{-1}\right)q^{-1} = (qp)[0,w](qp)^{-1} \tag{8}$$

The inversion of the quaternion represents the rotation around the same axis by the same angle but in the opposite direction. After assembling two rotations represented by q and q^{-1} the synonymous transformation is obtained, as presented in equation 9.

$$q^{-1}\left(q[0,w]q^{-1}q\right) = [0,w] \tag{9}$$

Arm animation

Animation can be created in computer programs or can also be acquired from various acquisition systems (Kopniak, 2012). This article presents two computer animations representing various arm moves. Each move was created with the use of both Euler angle and quaternion transformations. The animations were created in computer program. It was written in objective C++ language in "QtDevelopment" IDE (Eigen documentation). Two main libraries were included into the program: biomechanical toolkit (b-tk) (Definition of the gimbal lock) and Eigen (Hanson, 2006). They both are open source so can be used for free. B-tk is a library that is used for biomechanical analysis. It allows to read, write and modify acquisition files (definition of the gimbal lock). The library was used to write arm animation into c3d file (The C3D File Format. User Guide). This file can be then opened in various compatible programs which show the movements written in the file. The animation created in my program was opened in Mokka (Mokka Documentation) to visualise the arm moves. The chosen frames from the created animations are presented in this article (Fig. 1 and 2). The second library was used to create animations using Euler angles and quaternions. The animation parameters, such as frame number, point number in a frame and the frame frequency is set in the program. There are 200 frames in each animation, and the frequency was automatically set to 50 Hz. During the animation process the interpolation is used [K] to calculate arm orientations between two key frames (the first and the last one). The animation with Euler angles uses linear interpolation while with quaternions spherical linear interpolation (slerp).

The created animation presents two various arm moves. The program creates arm animation based on 3 joints: wrist, elbow and shoulder joint. Each joint is represented as one dot placed in the 3D space. Their spot is represented by 3 axis coordinates: X, Y and Z. The right-handed system is used to animate the objects. The visualization is represented with the segments created with the proper pair of moving points.

Two various arm moves are animated. First move is very easy and shows the arm movements as bending and extension. In order to animate these moves only one joint changes its positions – wrist joint. These moves are assembled in one smooth animation. The move starts from the position of upright hand. Two joints representing shoulder and elbow lay in straight line. The wrist joint lays a bit lower, under the angle of 10 degrees. This joint changes the angle from minus 10 to 150 degrees. In the beginning all points lay in the same position (0,0,0). The wrist point is moved to a proper location during the animation process. The selected frames of animation are presented in Figure 1.They are obtained from Mokka program. The frames were zoomed to improve the visualization. The animations created using Euler angles and quaternions are smooth and identical. There are no differences. It seems two analysed methods are proper to animate easy arm moves.

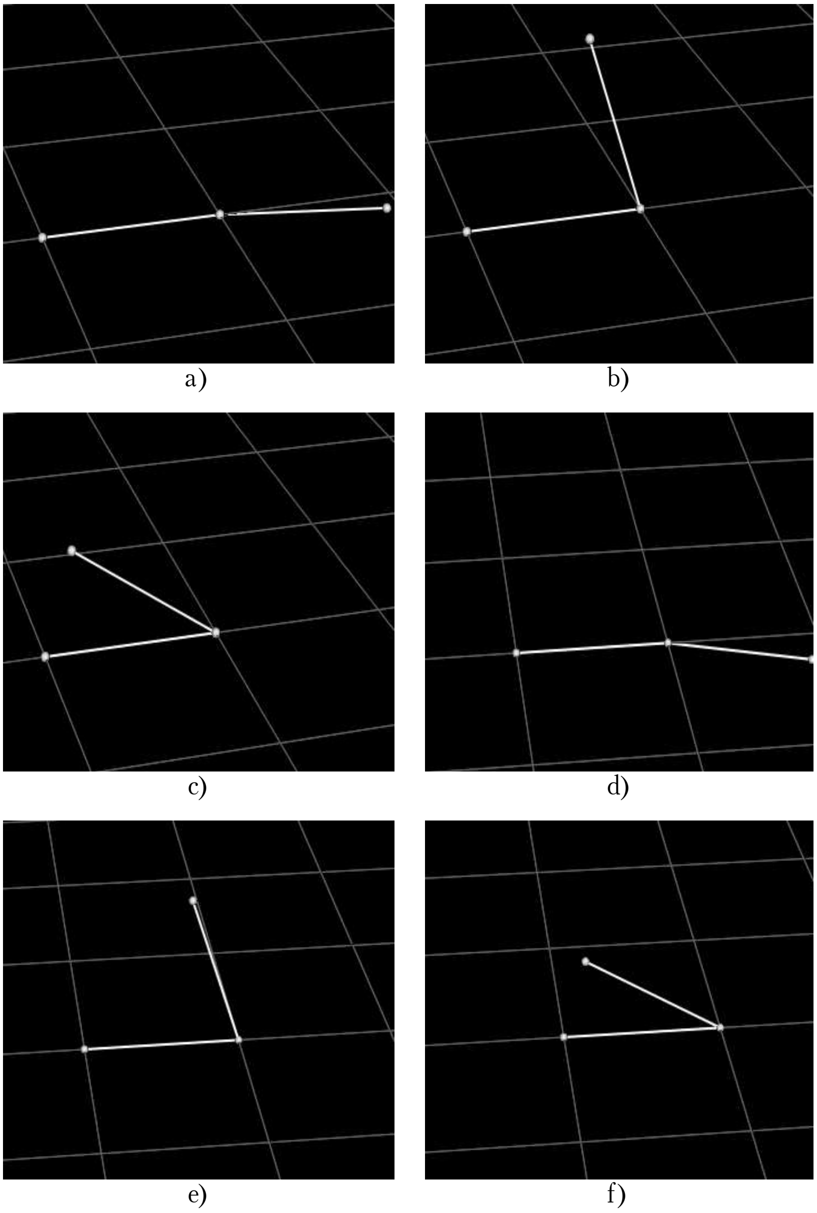

Figure 1. **Chosen frames of arm animation for bending and extension using a) Euler angles – frame 1b) Euler angles – frame 75 c) Euler angles – frame 100 d) quaternions – frame 1 e) quaternions – frame 75 f) quaternions – frame 100**

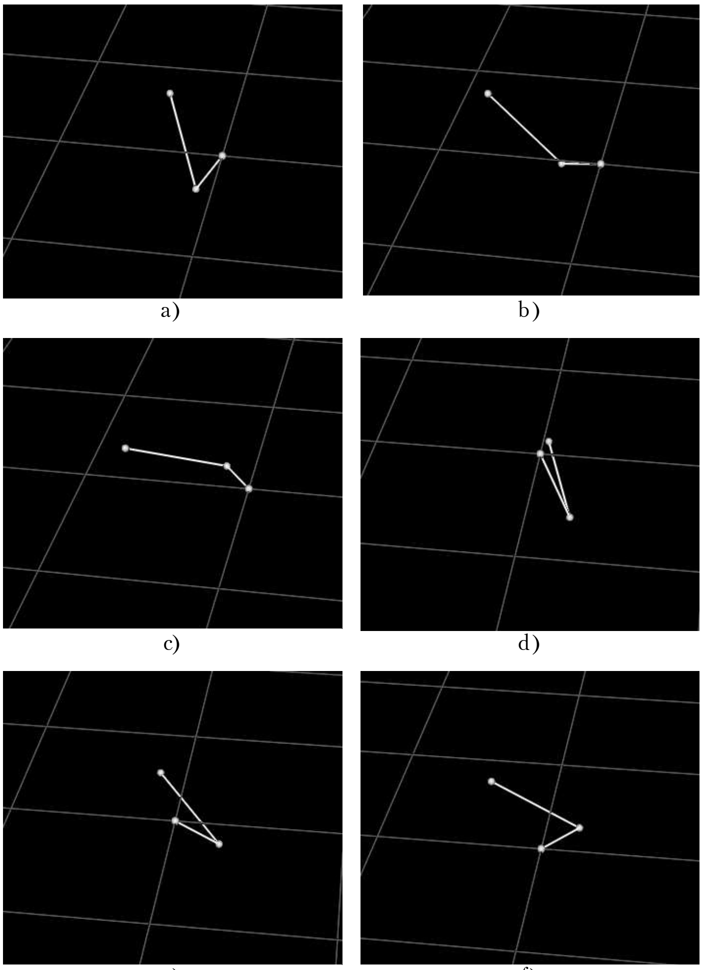

Figure 2. **Chosen frames of complex arm animation using a) Euler angles – frame 25 b) Euler angles – frame 50 c) Euler angles – frame 75 d) quaternions – frame 25 e) quaternions – frame 50 f) quaternions – frame 75**

The second arm move animated is more complex — two joints (wrist and elbow) change their positions. In the beginning the shoulder is lowered and the forearm is located at 90 degrees. The animation presents arm moving up and left so that the angle between shoulder and forearm is still 90 degrees. Then the arm is moving back to the initial position. The chosen frames from the moves are shown in Figure 2.

These two animations are not the same almost from the beginning. The most visible difference is from 40 to 60 frames. The elbow does an additional move. It goes left and at the middle of the move it is placed to the right position. In Euler representation two angle rotations are given: the initial one (-90, 0, 0) and the final one (90, -90, 90). The move is animated based on the interpolation between these two settings. The problems appears during the interpolation when the unexpected angles are computed in the intermediate frames disturbing the move. The solution to get rid of the unexpected moves is to choose different sets of Euler angles in the key frames. Additionally, another problem is the "gimbal lock" which occurs when the object is rotated around 3 axes. It causes that in some specific situations (when rotation axes align with each other) it is impossible to reach all object positions in the 3D space (Eckel, 2002). The process of adjusting the right angular position may be time consuming and expensive. The problems mentioned above cause that the Euler angle method may not be reliable in animation process reflecting natural arm moves.

The animation based on quaternions also uses interpolation. In the beginning the two quaternions are defined: the initial one and the final one. Each angular position is represented by quaternion defining rotation around an axis. The animated move is smooth, natural and not disturbed.

Conclusion

The article presents and compares two various arm animations. They are created with the use of two methods: Euler Angles and Quaternions. The simple arm moves can be animated with the use of two analysed methods. They are both adequate for this purpose. However, animating more complex moves with these methods demonstrates the difference in the created moves. The obtained results indicate that more complex moves should be animated using quaternions. Although this is a more complex method, it is more effective and reliable. It provides smooth and natural arm movements in comparison to using Euler angles. That is why quaternion is the best choice for arm animation from to compared methods.

References:

Biomechanical Toolkit (b-tk) documentation, http://code.google.com/p/b-tk/

Definition of the gimbal lock, The Free Dictionary, http://encyclopedia2.thefreedictionary.com/gimbal+lock

Eckel, B. (2002). Thinking in C++, Helion.

Eigen documentation, http://eigen.tuxfamily.org/dox/

Hanson, A. J. (2006). Visualizing Quaternions, Elsevier/Morgan Kaufmann Publishers.

Kopniak, P. (2012). Rejestracja ruchu za pomoca urzadzenia Microsoft Kinect, Pomiary Automatyka Kontrola VOL. 58, Wydawnictwo PAK, Warszawa.

Mokka Documentation, http://b-tk.googlecode.com/svn/doc/Mokka/0.6/index.html

Parent, R. (2008). Computer Animation: Algorithms & Techniques, Elsevier.

Smolka, J., Skublewska-Paszkowska, M. (2013). Analysis of selected interpolation methods with artificial 3D data. Actual Problems of Computer Science, 1(3): 3—18.

The C3D File Format. User Guide. Motion Lab System, http://www.c3d.org/pdf/c3dformat_ug.pdf

Стаття надійшла до редакції 11.07.2013.