Department of Mathematics
Faculty of Mechanical Engineering
Slovak University of Technology in Bratislava

6th International Conference
APLIMAT 2007

LIBRARY FOR COMPUTATION IN THE PROJECTIVE SPACE

SKALA Vaclav (CZ), KAISER Jan (CZ), ONDRACKA Vit (CZ)

Abstract. The paper describes a library for computation in the projective space developed for use within C# and .NET environment. This experimental library is used to prove that computation in projective space can lead to elimination of the division operation in many cases and therefore to more robust algorithms. The taken approach unfortunately requires a change of the architecture of the current CPUs, nevertheless there is a hope that the proposed approach is reasonable.

Keywords: computer graphics, numerical algorithms, projective space, numerical library

Mathematics Subject Classification: Primary 51N15, 68N99

1 Introduction

Many problems solved in computer graphics, computer vision, visualization etc. require fast and robust computation, usually made in Euclidean representation. The most dangerous operations are the division operation and control structures that use comparison of numbers in floating point representation, as the numerical precision is very limited. This paper presents a new approach that enables elimination of division operation using homogeneous coordinates. The homogeneous coordinates are very often used especially in computer graphics and computer vision. This approach enables to postpone the division operation to the very last computational step if needed. Recent experiments proved that in some cases computations using homogeneous coordinates is faster and increase robustness of algorithms as well. In order to simplify development and verification of new algorithms a specific library has been developed.

2 Mathematical background

Homogeneous coordinates are frequently used for geometric transformations as they enable us to represent all fundamental transformations as matrix-vector multiplication. For simplicity, let us consider two-dimensional case. Every point **P** is represented as **P** = (X, Y), where X and Y are X and Y coordinates of the given point **P**. In homogeneous coordinates the same point **P** is represented as **P** = $[x, y, w]^T$, where x, y resp. w are homogeneous coordinates in the projective space. A transformation between Euclidean and projective coordinates is given as X = x/w, Y = y/w, where w <> 0, for details see [1], [9]. It can be seen that the point **P** is actually a line **p** in the projective space with x, y, w axes excluding the origin.

On the other hand a line in two-dimensional Euclidean space is defined as ax + by + c = 0, that is actually a plane ax+by+cw=0 in the projective space with coordinates $[x, y, w]^T$ excluding the origin again. This plane is represented by a vector $[a, b, c]^T$ in the dual space representation. It means that a line in the Euclidean space is dual to a point $[a, b, c]^T$ in the dual space representation. It is known (see [3], [6]), that a line is dual to a point and a point is dual to a line for two-dimensional case. In three-dimensional case, a point is dual to a plane and vice versa.

From [7] can be seen that if the cross product operation is applied on points $\mathbf{P_1}$ and $\mathbf{P_2}$ in homogeneous coordinates, the coefficients $[a, b, c]^T$ are obtained as follows:

$$[a,b,c]^T = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ x_1 & y_1 & w_1 \\ x_2 & y_2 & w_2 \end{vmatrix} \tag{1}$$

Due to the principle of duality an intersection point of two lines $\mathbf{p_1} = [a_1, b_1, c_1]^T$ and $\mathbf{p_2} = [a_2, b_2, c_2]^T$ can be computed as the cross product of two vectors representing those two lines:

$$[x,y,z]^T = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{vmatrix} \tag{2}$$

Another positive effect of using homogeneous coordinates in calculations is that applying division to vector $[x, y, w]^T$ only requires multiplication of its w component. Similarly, for the three dimensional case, point is dual to a plane and vice versa.

When dealing with iterative algorithms, we often run into problems with numerical overflow of vector components. Using homogeneous coordinates, we can easily prevent this problem. The following formula is valid for vector with homogeneous coordinate:

$$x = [x, y, w]^T = [kx, ky, kw]^T \tag{3}$$

Therefore for any homogeneous vector **v** we can write:

$$v = [a.2^i, b.2^j, c.2^k]^T = \left[a.2^{(i-k)}, b.2^{(j-k)}, c\right]^T \tag{4}$$

6th Int.Conf. Aplimat, Bratislava, ISBN 978-80-969562-00-8, pp. 125-130, 2007

We will call this operation the “exponent normalization” further in this paper.

3 Implementation

A library for computation in the projective space was developed in C# using .NET technology. The library structure was also inspired by the Cg/HLSL programming language used for GPU programming, see [2]. It provides user with basic data structures representing floating point vectors and matrices up to four dimensions in both Euclidean and projective coordinates, fundamental vector and matrix operations with these structures. Common operations as addition, subtraction, multiplication and homogeneous division are supported, as well as the dot product, the cross product and linear interpolation of vectors. Comparisons and all required conversions are also supported (see Tables 1-4). The operands of each operation are checked automatically to prevent numerical overflow and/or not-a-number results. Data types are implemented as structures for speedup.

Supported data types for vectors are Double*X* in the Euclidean space, resp. Double*X*P in the projective space. Matrices are represented by types Double*X*M. In all cases, $X \in \{1, \ldots, 4\}$. By notation N.**x** we mean every Euclidean component of given vector N; one-dimensional vector is noted only as N.

Operations for Double*X*P C = Double1 A ***op*** Double*X*P B

±	C.**x** := A * B.w ± B.**x** C.w := B.w
*	C.**x** := A * B.**x** C.w := B.w
For B: 1D only	
/	C.**x** := A * B.w C.w := B.**x**
≠	A * B.w ≠ B.**x**
=	A * B.w = B.**x**
≤	A * B.w ≤ B.**x**

Operation for Double*X*P C = Double1 A ***op*** Double1P B

/	C.**x** := B.**x** C.w := A * B.w

Operation for Double*X*P C = Double*X*P A ***op*** Double1P B

/	C.**x** := B.**x** * A.w C.w := A.**x** * B.w

Operations for Double*X*P C = Double*X*P A ***op*** Double*X*P B

±	C.x := A.x * B.w ± A.w * B.x C.w := A.w * B.w
*	C.**x** := A.**x** * B.**x** C.w := A.w * B.w
≠	A.**x** * B.w ≠ B.**x** * A.w
=	A.**x** * B.w = B.**x** * A.w
≤	A.**x** * B.w ≤ B.**x** * A.w
For B: 1D only	
/	C.**x** := A.**x** * B.w C.w := A.w * B.**x**

Vector and matrix operations

(**x** denotes vector, **X** denotes matrix)

Euclidean operations

±	**c** := **a** ± **b** – Double*X*, Double*X*
*	**c** := **a** * **b** – dot product

Matrix operations

±	**C** := **A** ± **B** – Double*X*M, Double*X*M
*	**C** := **A** * **B** – Double*X*M, Double*X*M

Vector-Matrix operations

*	**C** := a * **B** – Double1, Double*X*M
*	**c** := **A** * **b** – Double*X*M, Double*X*

Projective vector operations

±	**c** := **a** ± **b** – Double*X*P, Double*X*P
*	**c** := **a** * **b** – dot product

Mixed Euclidean-homogeneous operations

*	**c** := **A** * **b** – DoublexM, DoublexP

Sample code implementing the homogeneous 2D dot product operation follows:

```
public static Double1P operator *(Double2P a, Double2P b) {
    Double1P c;

    a = PLib.NormalizeExp(a); // exponent normalization
    b = PLib.NormalizeExp(b);

    c.X = a.X * b.X + a.Y * b.Y; // computation
    c.W = a.W * b.W;

    if (!PLib.Check(c)) // validation
        throw new PLibException(String.Format(CultureInfo.CurrentCulture,
            "\nargument 1: {0}\nargument 2: {1}\nresult: {2}", a, b, c));

    if (c.W < 0) c = PLib.DeSign(c); // make sure that w >= 0

    return c;
}
```

The proposed library allows user to develop and verify new algorithms faster, by enabling straightforward transcription from commonly used mathematical notation directly into the source code. The downside of chosen approach is that final implementation for speed testing purposes still has to be done manually. Obtaining exact results would be otherwise impossible due to exponent normalization calls made by the library automatically.

4 Experimental results

The presented approach has been used in implementation of several algorithms, for example line and line segment clipping [5], Cyrus-Beck line clipping algorithm, solver for non-linear equations including conjugate gradient method and solver for linear system of equations. In all cases the stability and robustness was examined and proved. It should be noted that even for intersection of two parallel lines in two-dimensional case stable solution can be found and the resulting intersection point $[x, y, w]^T$ has the homogeneous coordinate equal to zero, e.g. it is in infinity in the direction given by the homogeneous coordinates $[x, y, 0]^T$.

The computational slowdown is especially caused by the need to normalize exponents of non-homogeneous and homogeneous part of the given vector.

5 Conclusions

The above proposed approach has been used for a library implementation supporting computation in projective space using C# and .NET technology environment. Current library supports only computation in 2D, 3D and partially in 4D space. All operations are implemented as overloaded operators (if possible), using object oriented technology. Due to the fact that in iterative algorithms the exponents of all parts of the given vector grow, additional operation for exponent normalization was introduced. Unfortunately this operation is not supported by the current processor architecture. This operation was implemented as a function, which partially influenced the speed of computation. Nevertheless many non-iterative algorithms can use the presented approach directly without use of the presented library that is aimed for algorithm design and verification. Such algorithms can use advantages of matrix-vector multiplication and significant speedup can be expected (see [4]).

In future work we expect that the library will be extended for computation in n-dimensional projective space. In an ideal case, it could influence also a structure of CPUs in the future. It is necessary to note that this approach enables to avoid or to postpone division operations that cause instability of algorithms especially in cases close to singular.

6 Acknowledgements

The authors would like to express their thanks to students at the University of West Bohemia for recommendations and constructive discussions that helped to finish this work. Many thanks belong to our colleagues for their valuable comments and suggestions that improved this work significantly, especially to Martin Janda for verification of Cyrus-Beck implementation and Slavomir Petrik for implementation of a solver for linear system of equations.

This work was supported by the project 6FP NoE 3DTV No.511568 and by the project Virtual No.2C06002 of the Ministry of Education of the Czech Republic.

References

[1] Bloomenthal,J., Rokne,J.: Homogeneous Coordinates. The Visual Computer, 11, pp. 15-26, Springer Verlag, 1994

[2] Fernando,R., Kilgard,M.J.: Cg Tutorial: The Definitive Guide to Programmable Real- Time Graphics, Addison Wesley, 2003

[3] Johnson,M.: Proof by Duality: or the Discovery of "New" Theorems, Mathematics Today, December 1996

[4] Ondracka,V.: Clipping algorithms (in Czech), BSc. Thesis (Supervisor: Skala,V.), University of West Bohemia, 2006

[5] Skala,V.: A New Line Clipping Algorithm with Hardware Acceleration, CGI'2004 conference proceedings, IEEE, Greece, 2004

[6] Skala,V.: New Approach to Line and Line Segment Clipping in Homogeneous Coordinates, The Visual Computer, Vol.21, No.11,pp.905-914, ISSN 0178-2789, 2005

[7] Skala,V.: GPU Computation in Projective Space Using Homogeneous Coordinates, Game Programming GEMS 6 (Ed.Dickheiser,M.), pp.137-147, ISBN 1-58450-450-1, Charles River Media, 2006

[8] Skala,V.: Length, Area and Volume Computation in Homogeneous Coordinates, International Journal of Image and Graphics, Vol.6., No.4, pp.625-639, ISSN 0219-4678, 2006

[9] Stolfi,J.: Oriented Projective Geometry, Academic Press, 2001

[10] Yamaguchi,F., Niizeki,M., Fukunaga,H.: Two robust point-in-polygon test based on 4x4 determinant method, Advanced on design automation (ASME), Vol. 23, No.1,pp.89-95, 1990

[11] Yamaguchi,F.: Computer-Aided Geometric Design: A Totally Four-Dimensional Approach, Springer Verlag, ISBN 4-431-70340-3, 2002

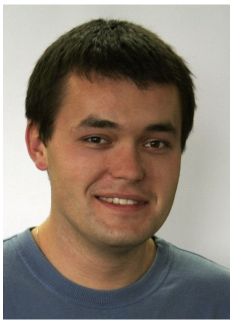
6th Int.Conf. Aplimat, Bratislava, ISBN 978-80-969562-00-8, pp. 125-130, 2007

Current address

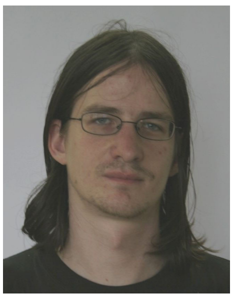
Department of Computer Science and Engineering
University of West Bohemia in Pilsen
Univerzitni 8
306 14 Plzen
Czech Republic

Authors

Vaclav Skala is a full professor of Computer Science at the Faculty of Applied Sciences at the University of West Bohemia in Plzen, Czech Republic. He is a member of The Visual Computer and Computers & Graphics editorial boards. He has been a research fellow or lecturing at the Brunel University (London, U.K.), Moscow Technical University (Russia), Gavle University (Sweden) and others institutions in Europe. He organizes the WSCG International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision (http://wscg.zcu.cz) held annually since 1992 and .NET Technologies conferences (http://dotnet.zcu.cz). He is interested in algorithms, data structures, mathematics, computer graphics, computer vision and visualization. He has been responsible for several research projects as well.
Currently he is a director of the Center of Computer Graphics and Visualization (http://herakles.zcu.cz).



Jan Kaiser is a research assistant at the Centre of Computer Graphics and Data Visualization at the University of West Bohemia, Czech Republic. He received MSc. degree in Computer Graphics at the University of West Bohemia in 2006. His interests are in the field of real-time data visualization.



Vit Ondracka finished his study of Computer Science at the Faculty of Applied Sciences at the University of West Bohemia, Plzen, Czech Republic, in 2006. He is interested in computer graphics, object oriented programming, C# and .NET. Currently he is responsible for the projective library development.