レイトレーシングによるコンピュータグラフィクス入門
- ベクトル・マトリクスライブラリの使用方法 -

金森 由博*

2010 年 12 月 1 日

1　ベクトル・マトリクスライブラリの使用方法

雛形ソースコードの中には my_algebra.h というヘッダファイルが含まれており、この中にベクトル・マトリクス演算についてのコードが記述されている。定義されているクラス[*1]は次のとおりである。

vec2　2 次元ベクトル
vec3　3 次元ベクトル
vec4　4 次元ベクトル
mat2　2×2 行列
mat3　3×3 行列
mat4　4×4 行列
quat　クォータニオン (quaternion; 四元数)…3 次元の回転を表す複素数の拡張。

この実験で実際に使うのは 3 次元ベクトルクラス vec3 のみなので、以下 vec3 についてのみ、主な使用例を示す。ここに示す以外にも関数が定義されているので、詳しくは my_algebra.h を見てほしい。

初期化

```
vec3 a;
vec3 b(1,1,1); // スカラ値を代入して初期化
vec3 c(b); // 他のベクトルの値を代入して初期化 … c = b;
vec3 d(a, b); // ベクトル a から b に向かうベクトル … d = b - a;
```

ベクトルの各要素へのアクセス

```
Real x = a.x; // x, y, z がメンバ変数
Real y = a.vec_array[1]; // x, y, z は配列 vec_array[3] との共用体
Real z = a[2]; // a.vec_array[2] と同じ
```

ベクトルへの値の設定

* kanamori@cs.tsukuba.ac.jp
*1 struct として宣言されているが、C++ では struct は変数および関数がすべて public であるような class と考えてよい。

```
a.set( 1.f ); // x, y, z すべてに同じ値を代入
a.set( 1.f, 2.f, 3.f ); // x, y, z 別々に値を代入
a.set( b ); // a にベクトル b を代入
```

ベクトルの加減算

```
a = b - c;
a = b + c;
a += b;
a -= c;
a.add( b ); // a += b; と同じ
a.sub( b ); // a -= b; と同じ
```

スカラ値との乗除算

```
Real f; // my_algebra.h で float を Real として typedef してある
a = f * b; // スカラ値とベクトルの乗算結果を代入
a *= f; // 現在の値にスカラ値を乗算
a = b / f; // スカラ値で除算
a /= f; // 現在の値をスカラ値で除算
a.scale( f ); // a *= f; と同じ
```

ベクトルの内積 (dot product)

```
Real d;
d = dot(a, b);
d = a.dot(b);
d = a * b;
```

ベクトルの外積 (cross product)

```
c = a ^ b;
c.cross(a, b); // 計算結果が c に代入される
```

ベクトルの大きさ

```
Real l = a.norm(); // ||a||
Real l2 = a.sq_norm(); // ||a||^2 … 2 乗ノルム
```

2 つのベクトルの距離

```
Real dist = a.distance( b ); // || a - b ||
Real dist2 = a.squareDistance( b ); // || a - b ||^2 … 2 乗距離
```

ベクトルの正規化

```
a.normalize(); // a を単位ベクトルに
```

```
b = normalize( a ); // b に正規化した a を代入
```

反射ベクトル

```
vec3 i; // 光の入射ベクトル
vec3 r = reflect(i, n); // n は単位法線ベクトル
```

屈折ベクトル

```
Real eta = 1.5f; // 相対屈折率
vec3 r = refract(i, n, eta); // n は単位法線ベクトル、全反射の場合 r はゼロベクトル
```

ベクトルの表示

```
cout << a << endl; // (1,2,3) のように表示される
```

2　使用上の注意

ヘッダ my_algebra.h の内容は、MyAlgebra という名前空間 (namespace) の中に入っている:

my_algebra.h

```
#pragma once // ヘッダが一度だけインクルードされるようにするためのおまじない

namespace MyAlgebra {

// 中略

}
```

なので、my_algebra.h の内容を使うときには

```
MyAlgebra::vec3 a;
```

のように MyAlgebra::という修飾子をつけてやるか、あるいはソースコードの最初の方で

```
using namespace MyAlgebra;
```

と書いて、MyAlgebra 名前空間を修飾子なしで参照できるようにする必要がある。