cs805 Assignment 1

Ray Shulang Lei
200253624
Department of Computer Science
University of Regina

September 27, 2012

Abstract

This assignment is written in literate programming style, generated by noweb, rendered by LaTex, and compiled by clang++ with c++ 11 standard.

1 Question 1

Let n be a 3 tuple vector, and given that it is along $V1$. It is trivial that we can imply:

$$n = \frac{V1}{[|V1|, |V1|, |V1|]}$$

where $|V1| = \sqrt{V1_x^2 + V1_y^2 + V1_z^2}$

Thus n is now known.

By the definition of cross product, denoted as $\times$ here, knowning that $V1$ and $V2$ is non-collinear, we can also derive:

$$u = \frac{V2 \times V1}{[|V2 \times V1|, |V2 \times V1|, |V2 \times V1|]}$$

Finally, it is also trivial that:

$$v = n \times u$$

2 Question 2

According to the requirement, we need a function that gets the new coordination U, V, N from our two vectors.

First, assuming we have the function already. Thus giving it two points, our function will get the U, V, N from them.

```
<<src/main.cpp>>=
#include <iostream>
#include <typeinfo>//debugging only
#include "util.h"

int main () {
  Point V1;
  decltype(V1) V2;// V2 is of same type of V1

  V1 = {0,0,1000};
```

```
  V2 = {0,1,1};

  auto uvn = get_uvn(V1, V2);// compiler will replace 'auto' with the right type

  for (auto point : uvn) {//for each point in uvn
    for (auto num : point) {//for each number in point
      std::cout<<num<<',';
    }
    std::cout<<std::endl;
  }

  return 0;
}
@
```

I use a header file for typedefs and function declarations for more readable code.

```
<<src/util.h>>=
#ifndef POINTS_HPP
#define POINTS_HPP
#include <tr1/array>
typedef std::tr1::array<float, 3> Point;
typedef std::tr1::array<Point, 3> UVN;
UVN get_uvn(Point V1, Point V2);
float get_length(Point);
Point cross_product(Point, Point);
Point normalize(Point);
#endif
@
```

Finally, here is the function.

```
<<src/util.cpp>>=
#include "util.h"
#include <math.h>

//get u,v,n from two non-collinear vectors
UVN get_uvn(Point V1, Point V2) {
  //get n, which is just normalized V1
```

```
  Point n = normalize(V1);

  //get u, which is normalized V2 x V1
  Point u = normalize(cross_product(V2, V1));

  //get v, which is normalized n x u
  Point v = normalize(cross_product(n, u));

  return {u,v,n};
}

//normalize a point
Point normalize(Point x) {
  return { x[0]/get_length(x),
           x[1]/get_length(x),
           x[2]/get_length(x) };
}

//calculates cross product of two points
Point cross_product(Point x, Point y) {
  return { x[1]*y[2] - x[2]*y[1],
           x[2]*y[0] - x[0]*y[2],
           x[0]*y[1] - x[1]*y[0]};
}

//calculates length of a point
float get_length(Point x) {
  return sqrt(pow(x[0],2)+pow(x[1],2)+pow(x[2],2));
}
@
```

Furthermore, this is the command to link these files. Notice that I am using -std=c++11 flag to enable c++ 11 features. The output binary executable is bin/get_uvn_test

```
<<compile.sh>>=
clang++ -std=c++11 -o bin/get_uvn_test src/main.cpp src/util.cpp
@
```

3 Question 3

3.1 part a

By definition of matrix multiplication,

$$T \times T^{-1} =$$

$$\begin{bmatrix} 1+0+0+0 & 0+0+0+0 & 0+0+0+0 & VRP_x+0+0+-VRP_x \\ 0+0+0+0 & 0+1+0+0 & 0+0+0+0 & 0+VRP_y+0+-VRP_y \\ 0+0+0+0 & 0+0+0+0 & 0+0+1+0 & 0+0+VRP_z+-VRP_z \\ 0+0+0+0 & 0+0+0+0 & 0+0+0+0 & 0+0+0+1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = I$$

It is also trivial that any n-tuple vector VRP in n-demensional space will fall into this pattern.

3.2 part b

Similarly, by definition of matrix multiplication,

$$R \times R^{-1} =$$

$$\begin{bmatrix} u_x^2+u_y^2+u_z^2 & u_x \times v_x + u_y \times v_y + u_z \times v_y & u_x \times n_x + u_y \times n_y + u_z \times n_y & 0 \\ v_x \times u_x + v_y \times u_y + v_z \times u_z & v_x^2+v_y^2+v_z^2 & v_x \times n_x + v_y \times n_y + v_z \times n_z & 0 \\ u_x \times n_x + u_y \times n_y + u_z \times n_z & n_x \times v_x + n_y \times v_y + n_z \times v_z & n_x^2+n_y^2+n_z^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} u \times u & u \times v & u \times n & 0 \\ v \times u & v \times v & v \times n & 0 \\ n \times u & n \times v & n \times n & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

With the fact that u, v, n are all unit vectors,

$$\implies u \times u = 1, v \times v = 1, n \times n = 1$$

$$\implies \begin{bmatrix} u\times u & u\times v & u\times n & 0 \\ v\times u & v\times v & v\times n & 0 \\ n\times u & n\times v & n\times n & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & u\times v & u\times n & 0 \\ v\times u & 1 & v\times n & 0 \\ n\times u & n\times v & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

With the fact that u, v, n are orthogonal to each other,

$$\implies u\times v = 0, v\times n = 0, u\times n = 0$$

$$\implies \begin{bmatrix} 1 & u\times v & u\times n & 0 \\ v\times u & 1 & v\times n & 0 \\ n\times u & n\times v & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = I$$

3.3 part c