Projection to geometry
Theory　Manual

Toshihiro ookubo
toshihiro.ookubo@gmail.com

1. Projection to a geometry

CG(Computer Graphics) and CAE(Computer Aided Engineering) is be treated the geomery model. The geometry model are expressed by parametric curve and parametric surface. In the geometry operation, the calculation of a point near the curve or surface are important.

This manual explains the outline of projection to geometry(curve and surface).

2. Projection to a curve

2.1 outline

Curve are generally expressed parametric curve.(Spline curve,NURBS curve, Bézier curve and so on)

$$R(u) = \{x(u), y(u), z(u)\}^T \quad u \in [a, b]$$

R(a) is start point on curve. R(b) is end point on curve.

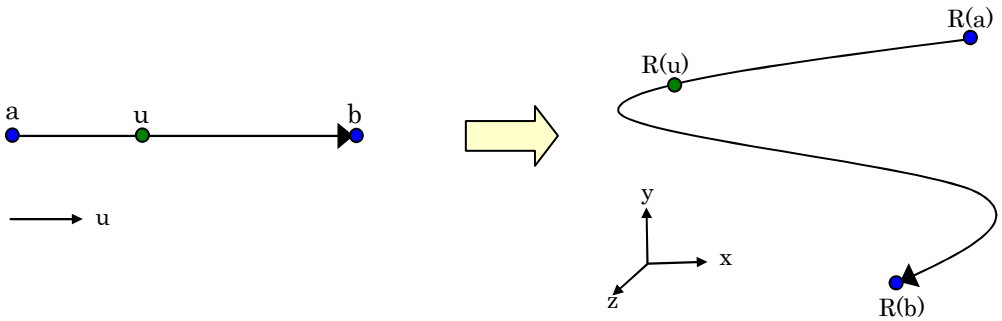

Figure Parameter space

Figure Real space (3D)

The process of projecting a point ***r*** to a curve must find minimum distance between ***R(u)*** and ***r.*** This projection process are generally calculated by Newton-Raphson method.

The distance function f(u) is expressed as follows.

$$f(u) = |R(u) - r|^2$$

The Derivative function of the distance function f(u) is expressed as follows.

$$d(u) = \frac{\partial f(u)}{\partial u} = \frac{\partial R(u)}{\partial u} \cdot \{R(u) - r\} = 0$$

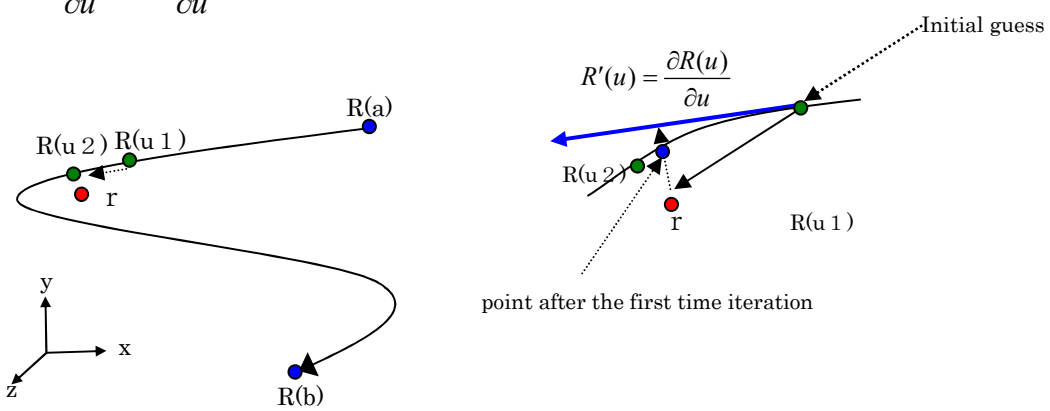

Figure Projection process

2.2 sample (source code /c language)

```c
#include<stdio.h>
#include<math.h>
#include "TNR_VECTOR.hxx"

#define EPS 1.0e-5
#define LIMIT 10000

TNR_POINT r(0.8,0.8,0.8);

/* points on line(start and end)*/
TNR_POINT b(0.0, 0.0, 0.0);
TNR_POINT a(1.0, 1.0, 1.0);

/* get position on line */
TNR_POINT curve_get_position(double u)
{
  TNR_POINT p = u * ( a - b ) + b;
  return p;
}

/* get derivative vector on line */
TNR_VECTOR curve_get_tangentVector(double u)
{
  return a - b;
}

//get  distance between target point and point on curve(param).
// f(u) =| R(u) - r |^2
double f(double u)
{
  TNR_POINT a = curve_get_position(u);
  TNR_POINT t = r - curve_get_position(u);
  return dotProduct(t,t) ;
}

// f(u)' =d(u)

double d(double u)
{
  TNR_VECTOR du = curve_get_tangentVector(u);
  TNR_VECTOR dist = r - curve_get_position(u);
  return dotProduct(du,dist);
}


int main(void)
{
   double  uOld = 0.1;
   double  uNew = 0.1;

   TNR_POINT A = curve_get_position(0.8);

   for(int i = 0;i <=LIMIT;i++){
            double fOld = f(uOld); /* distance function f(u) */
            double dOld = d(uOld); /* derivative function of f(u) */
            uNew = uOld + fOld / dOld; /* calculate next parameter u.*/

            double distanceOfPoints = f(uNew);

            double diffOfu = fabs(uNew - uOld);

            /*if u do'nt move or distance function less than tolerance(EPS),
             the calculation is ended. */

            if(diffOfu < EPS ||distanceOfPoints < EPS ) {
              printf("%d) end calculate.¥n",i);
             printf("dist=%f  parameter is  =%f¥n",sqrt(distanceOfPoints),uNew);
              return (0);
            }

     uOld = uNew;
   }
   printf("out of limit(%f)!!¥n",LIMIT);
}
```