The Answer for the $Q_1(t)$:
[t u v]= [3.0472e-01 1.0972e-02 5.3370e-01]
The point on the polynomial surface is (0.05486, 2.6685, 3.1221)
The point on the query curve is (1.5236, 1.5236, 4.1172)
Distance=2.1115

The Answer for the $Q_2(t)$
[t u v] =[2.0109e-01 6.4025e-03 5.1381e-01]
The point on the polynomial surface is (0.032012, 2.5690, 3.0295)
The point on the query curve is (1.0055, 1.0055, 4.3306)
Distance=2.2399

The following is some discussion.

M-file for Q1(t)

```
%t,u,v are variables
syms u v t
syms h real
digits(8)
%h is step size
%u,v is from 0 to 1
x=5*u;
y=5*v;
z=(-0.38)+25/3*x+4/3*y-25/3*x*x-10/3*x*y;
%Q1t, t is from 0 to 1
qx=5*t;
qy=5*t;
qz=25/6*t*t-25/6*t+5;
distancesquare=(x-qx)^2+(y-qy)^2+(z-qz)^2;
%diff(distance,'t'),diff(distance,'u'),diff(distance,'v');
Ft=diff(distancesquare,'t');
Fu=diff(distancesquare,'u');
Fv=diff(distancesquare,'v');
%t0,u0,v0 are initial point;
t0=0.5;
u0=0.5;
v0=0.5;
for i=1:200
%deltat,deltau,deltav are numbers of partial derivation
deltat=eval(subs(Ft,[t,u,v],[t0,u0,v0]));
deltau=eval(subs(Fu,[t,u,v],[t0,u0,v0]));
deltav=eval(subs(Fv,[t,u,v],[t0,u0,v0]));
```

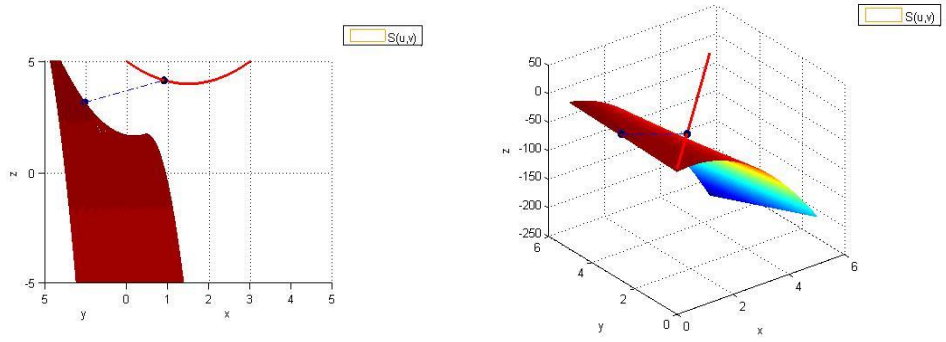
```
%t1,u1,v1 are variables
t1=t0+deltat*h;
u1=u0+deltau*h;
v1=v0+deltav*h;
newdistancesquare=subs(distancesquare,[t,u,v],[t1,u1,v1]);
%new t0,u0,v0
g=diff(newdistancesquare,'h');
h0=eval(solve(g==0,h));
[r,c]=size(h0);
if r~= 1

h0=h0.*((t0+deltat*h0>=0).*(t0+deltat*h0<=1).*(u0+deltau*h0>=0).*(u0+deltau*h0<=1).*(v
0+deltav*h0>=0).*(v0+deltav*h0<=1));
      h0(find(h0==0))=[];
end
t0=t0+deltat*h0;
u0=u0+deltau*h0;
v0=v0+deltav*h0;
end
point_tuv=[t0,u0,v0]
distance=sqrt(eval(subs(distancesquare,[t,u,v],[t0,u0,v0])))
```

Result

Point [t u v]= [3.0472e-01 1.0972e-02 5.3370e-01]

Distance=2.1115

Figure1



Note-The choice of initial point：

The choice of the initial point can somehow make the programme not workable.

Although in most case, the t,u,v will converge to the point [3.0472e-01 1.0972e-02

5.3370e-01], if the initial point [t0 u0 v0] is very close to [1 1 1] or [0 0 0] the programme may not converge or it may converge to some point that t,u,v are out of the range [0 1].
For example:

1. If the initial point is [0 0 0] then the result converges to [-7.9366e-03 1.7293e-01 -4.0682e-01]. I tried to increase the number of iterations, but it still converge to some point that the t,u,v go out of the range [0 1].
2. If the initial point is [0.9 0.9 0.9] then the program cannot run because in one of the iteration there are two h meet the two requirements, which are g'(h)=0 and the new [t u v] in the range of [0 1].
3. If the initial point is some point that is not too close to the [0 0 0] and [1 1 1],such as [0.2 0.2 0.2],[0.3 0.3 0.3],[0.5 0.5 0.5],[0.7 0.7 0.7] the result all converges to [3.0472e-01 1.0972e-02 5.3370e-01].
4. However if the initial point is the exact point [3.0472e-01 1.0972e-02 5.3370e-01] then it still converge to itself [3.0472e-01 1.0972e-02 5.3370e-01].

The previous discussion shows that my program is sensitive to the initial point.
I haven't figured out how big my program's converge interval is, for the initial point.

M-file for $Q_2(t)$

```
%t,u,v h are variables
syms u v t
syms h real
digits(8)
%h is step size
%u,v is from 0 to 1
x=5*u;
y=5*v;
z=(-0.38)+25/3*x+4/3*y-25/3*x*x-10/3*x*y;
%Q1t, t is from 0 to 1
qx=5*t;
qy=5*t;
qz=53.976*t^4-104.64*t^3+64.256*t^2-12.589*t+5;
distancesquare=(x-qx)^2+(y-qy)^2+(z-qz)^2;
%partial derivative,expression;
Ft=diff(distancesquare,'t');
Fu=diff(distancesquare,'u');
Fv=diff(distancesquare,'v');
%t0,u0,v0 are initial point , numbers;
t0=0.5;
u0=0.5;
v0=0.5;
for i=1:100
%deltat,deltau,deltav are numbers of partial derivation
deltat=eval(subs(Ft,[t,u,v],[t0,u0,v0]));
```

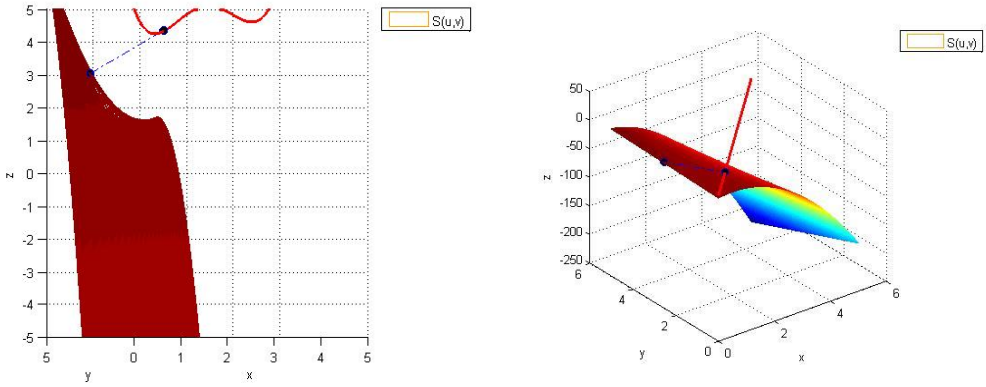
```
deltau=eval(subs(Fu,[t,u,v],[t0,u0,v0]));
deltav=eval(subs(Fv,[t,u,v],[t0,u0,v0]));
%t1,u1,v1 are variables
t1=t0+deltat*h;
u1=u0+deltau*h;
v1=v0+deltav*h;
newdistancesquare=subs(distancesquare,[t,u,v],[t1,u1,v1]);
%new t0,u0,v0
g=diff(newdistancesquare,'h');
h0=eval(solve(g==0,h));
[r,c]=size(h0);
if r~= 1

h0=h0.*((t0+deltat*h0>=0).*(t0+deltat*h0<=1).*(u0+deltau*h0>=0).*(u0+deltau*h0<=1).*(v
0+deltav*h0>=0).*(v0+deltav*h0<=1));
      h0(find(h0==0))=[];
end
%h0  多解，考虑 t0,u0，v0  大小约束
t0=t0+deltat*h0;
u0=u0+deltau*h0;
v0=v0+deltav*h0;
end
point_tuv=[t0,u0,v0]
distance=sqrt(eval(subs(distancesquare,[t,u,v],[t0,u0,v0])))
```

Result

Point [t u v] =[2.0109e-01 6.4025e-03 5.1381e-01]

Distance=2.2399

Figure 2



Note-The choice of initial point：

The same question appears. The program still is sensitive to the initial point for some extreme case. But in most case it can converge to [2.0109e-01 6.4025e-03 5.1381e-01].