Curves and Surfaces

Computer Graphics

CONTENTs

- Đường cong

- Mặt cong

CONTENTs

- Đường cong
- Mặt cong

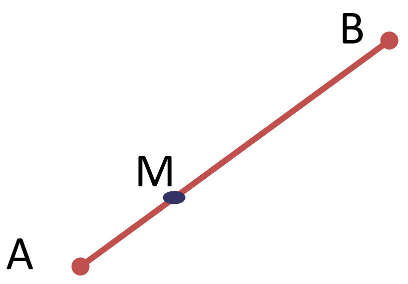
- **Cách biểu diễn**
  - Đường cong bất kỳ có thể biểu diễn bới ma trận điểm
    - Cần số lượng điểm vô cùng lớn để biểu diễn chính xác hình dạng
  - Sử dụng hàm đa thức để thể hiện hình dạng đường cong
    - Dạng tổng quát của hàm đa thức

$$p(x) = a_n x_n^n + a_{n-1} x_{n-1}^{n-1} + \ldots + a_1 x + a_0 = \sum_{i=0}^{n} a_i x^i$$

n – nguyên dương, a_0, a_1, ..., a_n là số thực

  - Đa thức thuận tiện cho tính toán bằng máy tính
  - Trong đồ họa đòi hỏi xác định tiếp tuyến, pháp tuyến cho đường cong. Đa thức dễ dàng tính vi phân.

Phöông trình tham soá cuûa ñöôøng thaúng



Goïi M(x,y) laø moät ñieåm thuoäc ñöôøng thaúng AB,

Ta coù:

$$\overrightarrow{AM} = t.\overrightarrow{AB}$$

$$\Leftrightarrow \begin{cases} x - x_A = t.(x_B - x_A) \\ y - y_A = t.(y_B - y_A) \end{cases} \Leftrightarrow \begin{cases} x = (1-t).x_A + t.x_B = X(t) \\ y = (1-t).y_A + t.y_B = Y(t) \end{cases}$$

Hay M=(1-t).A+t.B

Khi t=0 thì M≡A

t=1 thì M≡B

Neáu 0≤t≤1 thì M thuoäc ñoaïn AB, Phöông trình tham soá ñoaïn AB laø:

P(t)=(1-t).A+t.B vôùi 0≤t≤1

Hay P(t) = (X(t), Y(t))

Đường cong Bezier

Baøi toaùn: Cho n+1 ñieåm p_0, p_1, p_2,….,p_n ñöôïc goïi laø caùc ñieåm kieåm soaùt (ñieåm ñieàu khieån). Xaây döïng ñöôøng cong trôn ñi qua 2 ñieåm p vaø pn ñöôïc giôùi haïn trong bao loài do n+1 ñieåm treân taïo ra.

Thuật toán Casteljau

Để xây dựng đường cong P(t), ta dựa trên một dãy các điểm cho trước rồi tạo ra giá trị P(t) ứng với mỗi giá trị t nào đó.

Phương pháp này tạo ra đường cong dựa trên một dãy các bước nội suy tuyến tính hay ***nội suy khoảng giữa*** (In-Betweening).

Đường cong Bezier

Thuật toán Casteljau

Với 3 điểm P_0 , P_1 , P_2 có thể xây dựng một Parabol nội suy từ 3 điểm này bằng cách chọn một giá trị $t \in [0, 1]$ rồi chia đoạn P_0P_1 theo tỉ lệ t, ta được điểm P_0^1 trên P_0P_1. Tương tự, chia tiếp P_1P_2 cũng theo tỉ lệ t, ta được P_1^1. Nối P_0^1 và P_1^1, lại lấy điểm trên $P_0^1P_1^1$ chia theo tỉ lệ t, được P_0^2. Tập điểm P_0^2 chính là đường cong p(t).

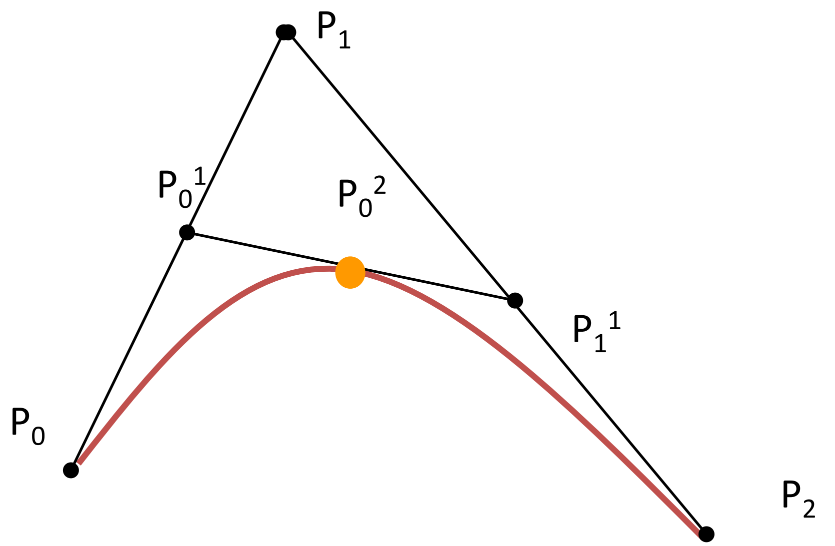
Biểu diễn bằng phương trình:

$$P_0^1(t) = (1-t).P_0 + t.P_1 \qquad (1)$$

$$P_1^1(t) = (1-t).P_1 + t.P_2 \qquad (2)$$

$$P_0^2(t) = (1-t).P_0^1 + t.P_1^1 \qquad (3)$$

Trong đó $t \in [0, 1]$



Thay (1), (2) vào (3) ta được:

$$\mathbf{P(t) = P_0^2(t) = (1-t)^2.P_0 + 2t.(1-t).P_1 + t^2.P_2}$$

Đây là một đường cong bậc 2 theo t nên nó là một Parabol.

Đường cong Bezier

Thuật toán Casteljau cho (n+1) điểm kiểm soát:

Giả sử ta có tập điểm: P_0, P_1, P_2, ..., P_n

Với mỗi giá trị t cho trước, tạo điểm $P_i^r(t)$ ở thế hệ thứ r, từ thế hệ thứ (r - 1) trước đó:

$$P_i^r(t) = (1-t).P_i^{r-1}(t) + t.P_{i+1}^{r-1}(t) \quad (\ r=0,1,...,n \ \text{ và } \ i=0,...,n-r)$$

Thế hệ cuối cùng:

$$\boxed{P_0^n(t) = \sum_{i=0}^{n} P_i.(1-t)^{n-i}.t^i.C_n^i}$$

là **đường cong Bezier** của các điểm P_0,P_1 ,P_2,...,P_n

Các điểm P_i, i=0,1,...,n gọi là các **điểm kiểm soát** (điểm Bezier).

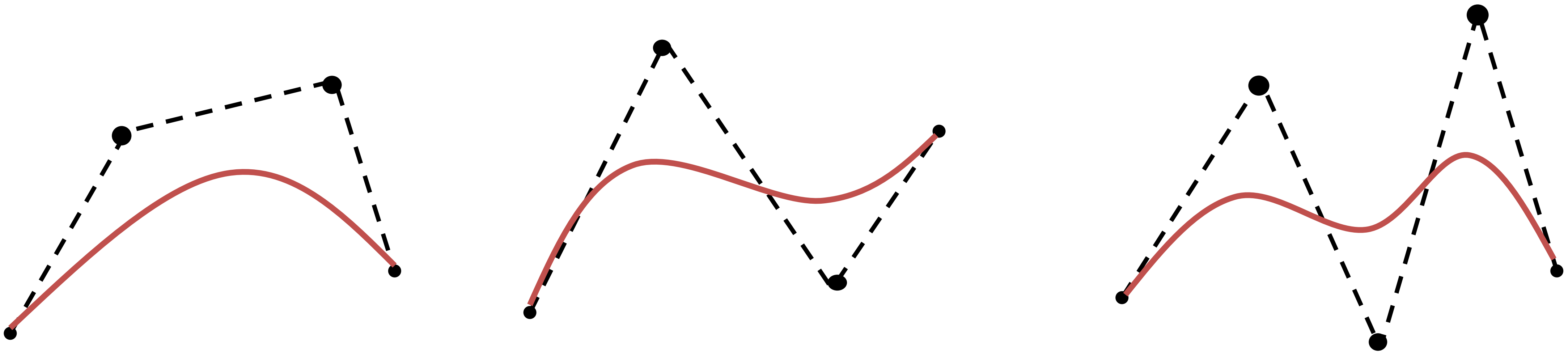
Đa giác tạo bởi các điểm kiểm soát gọi là **đa giác kiểm soát (**đa giác Bezier).

Đường cong Bezier

Thuật toán Casteljau cho (n+1) điểm kiểm soát:

Thế hệ cuối cùng:

$$P_0^n(t) = \sum_{i=0}^{n} P_i.(1-t)^{n-i}.t^i.C_n^i$$



Đường cong Bezier

Đường cong Bezier dựa trên (n+1) điểm kiểm soát P_0 ,P_1 , ...,P_n được cho bởi công thức:

$$P_0^n(t) = P(t) = \sum_{k=0}^{n} P_k . B_k^n(t)$$

Trong đó: P(t) là một điểm trong mặt phẳng hoặc trong không gian.

$B_k^n(t)$ gọi là đa thức Bernstein, được cho bởi công thức:

$$B_k^n(t) = C_n^k.(1-t)^{n-k}.t^k = \frac{n!}{k!.(n-k)!}.(1-t)^{n-k}.t^k$$ với $n \geq k$

Mỗi đa thức Bernstein có bậc là n. Thông thường ta còn gọi các $B_k^n(t)$ là các **hàm trộn** (blending function).

Đường cong Bezier

Đường cong Bezier dựa trên (n+1) điểm kiểm soát P_0 ,P_1 , ...,P_n được cho bởi công thức:

$$P_0^n(t) = P(t) = \sum_{k=0}^{n} P_k . B_k^n(t)$$

Tương tự, đối với **mặt Bezier** ta có phương trình sau:

$$P(u,v) = \sum_{i=0}^{m} \sum_{k=0}^{n} P_{i,k} . B_i^m(u) B_k^n(v)$$

Trong trường hợp này, khối đa diện kiểm soát sẽ có (m+1).(n+1) đỉnh.

Đường cong Bezier

Để tạo ra một đường cong Bezier từ một dãy các điểm kiểm soát ta áp dụng phương pháp lấy mẫu hàm p(t) ở các giá trị cách đều nhau của tham số t, ví dụ có thể lấy $t_i = i/m$, $i=0,1,...,m$.

Khi đó ta sẽ được các điểm $P(t_i)$ từ công thức Bezier. Nối các điểm này bằng các đoạn thẳng ta sẽ được đường cong Bezier gần đúng.

Đường cong Bezier

vẽ đường cong Bezier trong mặt phẳng:

```
typedef  struct {
            int x;     int y ;
}CPoint;
int      fact int n){
            if (n == 0)  return 1;
            else       return n*fact(n - 1);
}
float   power(float a, int n){
            if (n==0)  return 1;
            else       return   a*power(a,n-1);
}
```

Đường cong Bezier

```
float BernStein(float t,int n, int k){
        float ckn,kq;
        ckn = fact(n)/(fac(k)*fac(n - k));
        kq = ckn*power(1 - t,n - k)*power(t,k);
        return kq;
}
CPoint TPt(CPoint P[ ],float t, int n){
        CPoint Pt;      float B;        int k;
        Pt.x=0;         Pt.y=0;
        for (k = 0; k<=n; k++){
                B = BernStein(t,n,k);
                Pt.x = Pt.x + P[k].x*B;   Pt.y = Pt.y + P[k].y*B;
        }
        return Pt;
}
```

Đường cong Bezier

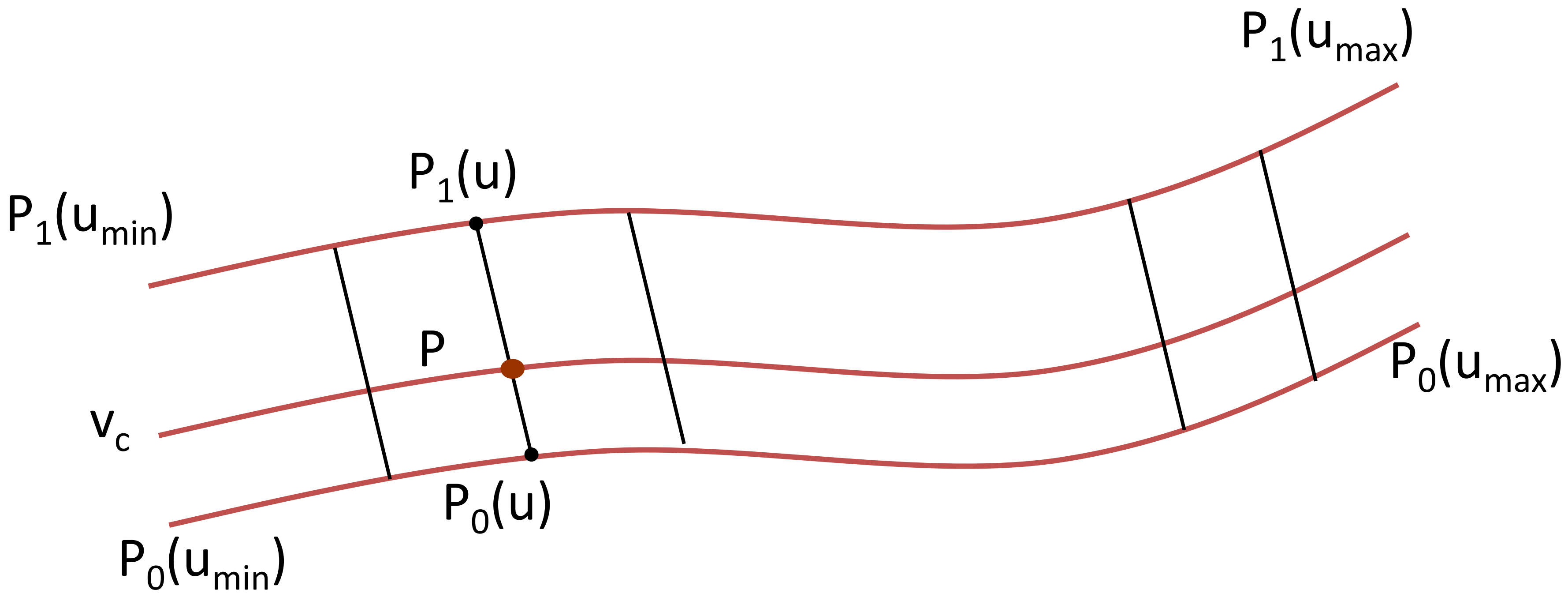
```
void DrawBezier( int n, CPoint P[ ]) {
        CPoint Pt;      float dt,t,m;     int i;
        t=0;    m=100;          dt=1/m;
        moveto(P[0].x,P[0].y);
        for(i = 1; i<= m ; i++){
                Pt=TPt(P,t,n);
                lineto(Pt.x,Pt.y);
                t = t + dt;
        }
        lineto(P[n].x,P[n].y);
}
```

CONTENTs

- Đường cong
- **Mặt cong**

Mặt có quy tắc

Mặt được tạo bằng cách quét một đường thẳng trong không gian theo cách nào đó.



Mặt có quy tắc

Phöông trình toång quaùt:

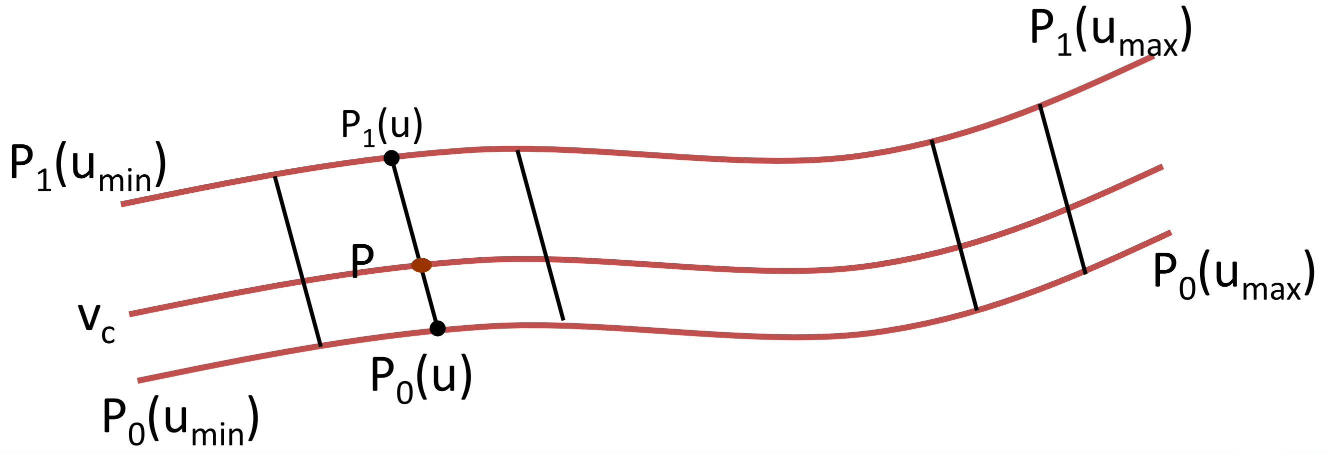
Goïi P_0P_1 laø ñoaïn thaúng sinh ra maët coù quy taéc, vaø $P \in P_0P_1$

Ta coù: $P(v)=(1-v)P_0+v.P_1$ vôùi $v\in[0,1]$

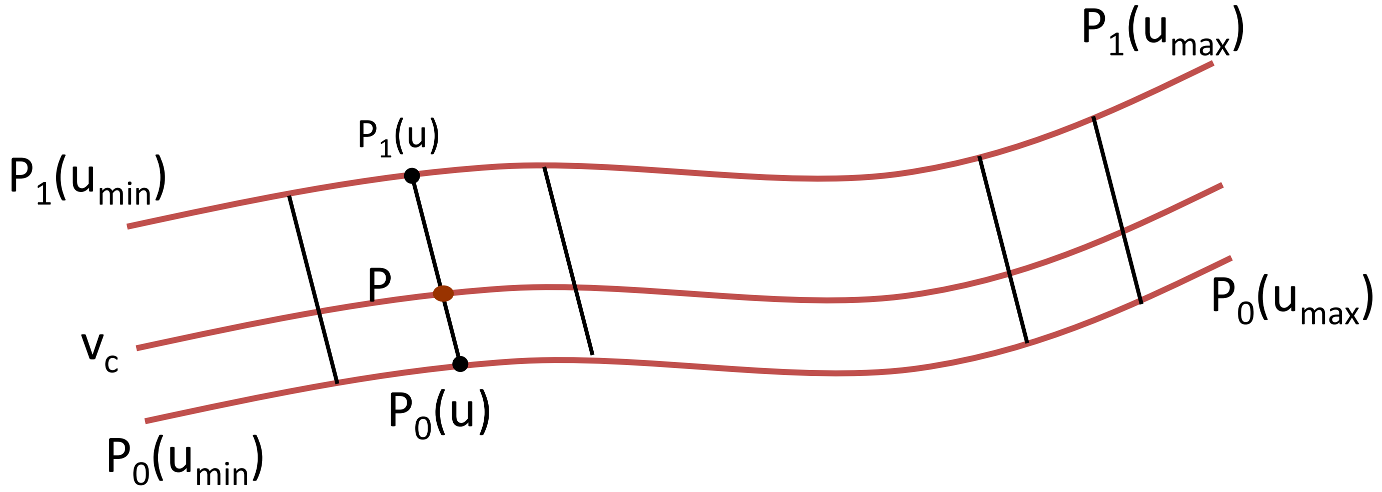
P_0 di chuyeån treân ñöôøng $p_0(u)$, P_1 di chuyeån treân ñöôøng $p_1(u)$

Neân P thuoäc maët coù quy taéc: $\mathbf{P(u,v)=(1-v).p_0(u)+v.P_1(u)}$

Vôùi $u_{min}\le u\le u_{max}$ vaø $0\le v\le 1$



Mặt có quy tắc



<u>Ñaëc bieät</u>:

Neáu v=0 vaø u thay ñoåi thì P di chuyeån treân ñöôøng $p_0(u)$

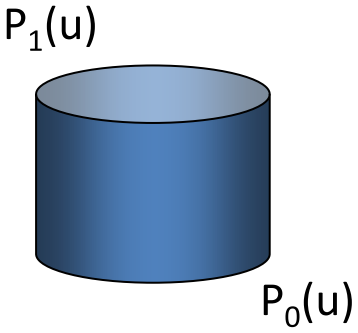
Neáu v=1 vaø u thay ñoåi thì P di chuyeån treân ñöôøng $p_1(u)$

Neáu $v=v_c$ vaø u thay ñoåi thì P di chuyeån treân cung song song vôùi $p_0(u_{min})$ $p_0(u_{max})$ goïi laø ñöôøng ñoàng möùc theo v.

Neáu $u=u_c$ vaø v thay ñoåi thì P di chuyeån treân ñoaïn thaúng $p_0(u_c)$ $p_1(u_c)$ goïi laø ñöôøng ñoàng möùc theo u.

Mặt có quy tắc

$$P(u,v)=(1-v).p_0(u)+v.P_1(u)$$

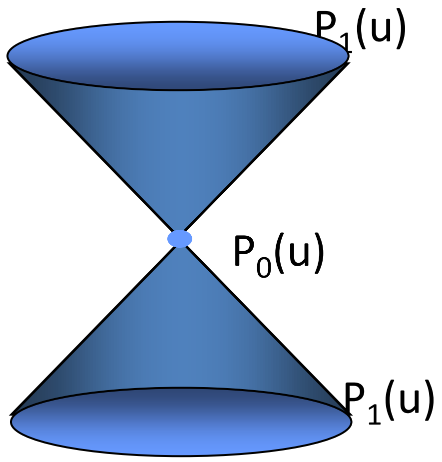


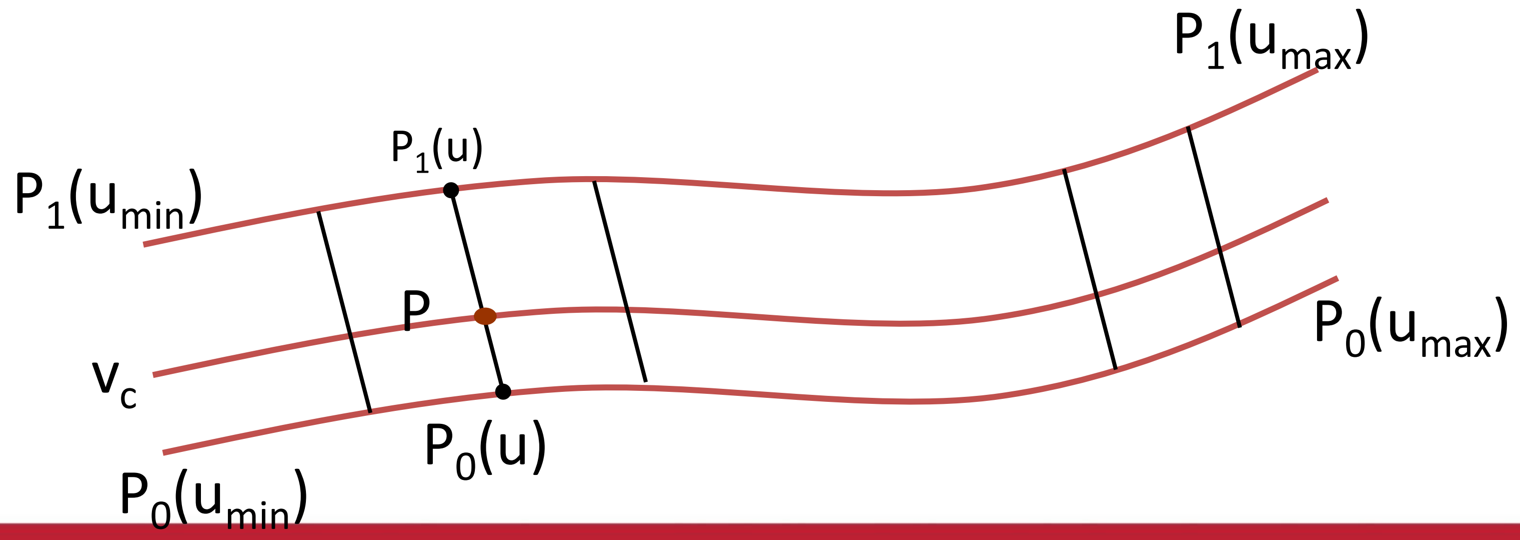
Ví duï: Maët truï: $p_0(u)$ laø ñaùy döôùi

$P_1(u)$ laø ñaùy treân

Maët noùn: $p_0(u)$ laø 1 ñieåm

$P_1(u)$ laø ñöôøng troøn





Mặt có quy tắc

Maët truï (Cylinder)

Maët truï laø maët ñöôïc taïo ra khi moät ñöôøng thaúng (ñöôøng sinh) ñöôïc queùt doïc theo moät ñöôøg cong $p_0(u)$ (ñöôøng chuaån). Ñöôøng cong $p_0(u)$ naèm treân moät maët phaúng naøo ñoù.

Goïi d laø ñöôøng sinh, d=const.

$p_0(u)$ laø ñaùy döôùi

$P_1(u)$ laø ñaùy treân

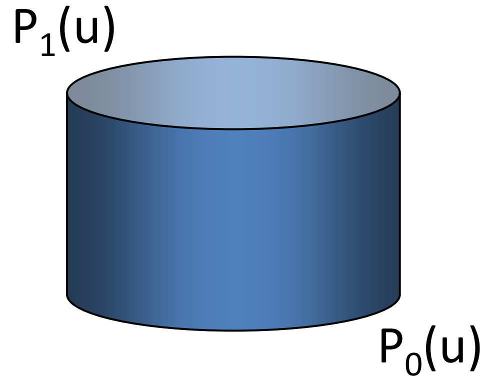
Suy ra: $d = P_1(u) - P_0(u)$

Ta coù: $p(u,v) = (1-v).P_0(u)+v.\,P_1(u)$

$= P_0(u)+(P_1(u)-P_0(u)).v$

$= P_0(u)+d.v$

Vaäy $p(u,v) = P_0(u)+d.v$



Mặt có quy tắc

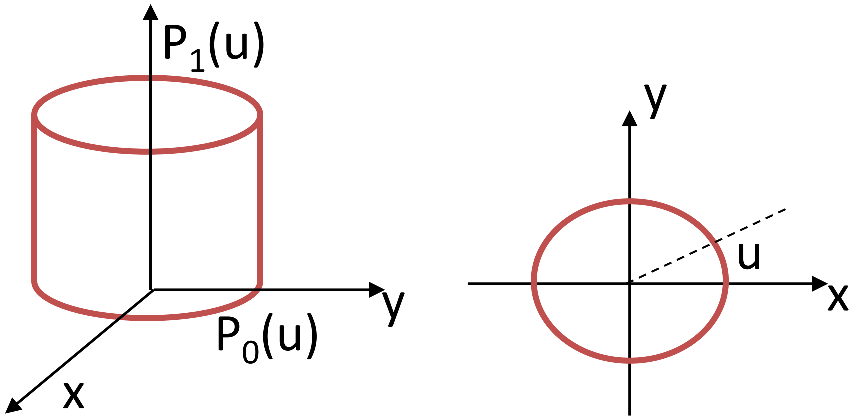
Maët truï (Cylinder)

Pt maët truï: $p(u,v) = P_0(u)+d.v$

Daïng quen thuoäc cuûa maët truï laø maët truï troøn: Trong maët phaúng xOy, laáy $P_0(u)$ laø ñöôøng troøn taâm O baùn kính r .

Ta coù: $d=(0,0,h)$

$P_0(u)=(r.\cos(u), r.\sin(u), 0)$



vaäy: $$\begin{cases} X(u,v) = r.\cos(u) \\ Y(u,v) = r.\sin(u) \\ Z(u,v) = h.v \end{cases}$$ vôùi $$\begin{cases} 0 \le v \le 1 \\ 0 \le u \le 2\pi \end{cases}$$

Mặt có quy tắc

```
void DrawCylinder(float R, float h){
    Point3D P;      Point2D P1;
    double          Delta_U,Delta_V,u,v;
    Delta_U = 0.06;         Delta_V = 0.03;
    for (u=0; u<2*M_PI; u+=Delta_U){
        for (v=0; v<1; v+=Delta_V){
            P.x = R*cos(u) ;    P.y = R*sin(u) ;
            P.z = v*h;          P1 = Chieu(KieuChieu,P);
            putpixel(xc+P1.x,yc+P1.y,WHITE);
        }
    }
}
```

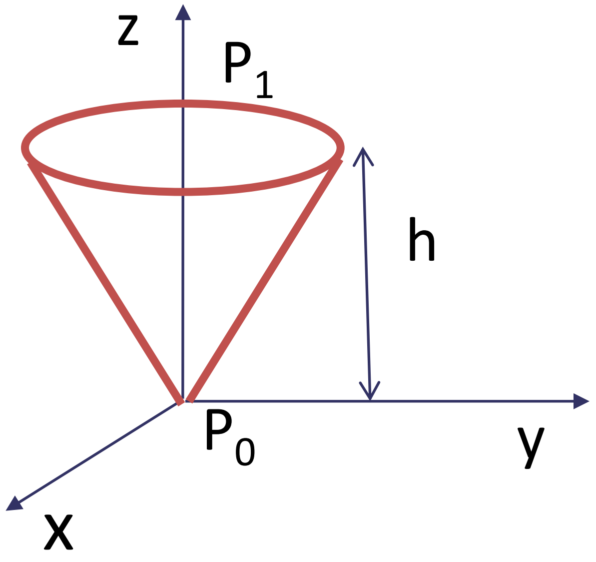
Maët noùn(Cone)

Maët noùn laø maët ñöôïc taïo ra khi moät ñöôøng thaúng di chuyeån doïc theo moät ñöôøng cong phaúng cho tröôùc. Caùc ñöôøng thaúng luoân ñi qua moät ñieåm coá ñònh goïi laø ñænh cuûa maët noùn

$p_0(u)$ truøng vôùi goác toaï ñoä O

$P_1(u)$ laø ñöôøng troøn taâm (0,0,h) baùn kinh r.

$P_1(u)=(r.\cos(u), r.\sin(u), h)$



Ta coù: $\mathbf{p(u,v)= (1-v).P_0(u)+v.\ P_1(u)= v.P_1(u)}$

Vaäy $\mathbf{p(u,v) = v.P_1(u)}$

Hay: $\begin{cases} X(u,v) = v.r.\cos(u) \\ Y(u,v) = v.r.\sin(u) \\ Z(u,v) = v.h \end{cases}$ vôùi $\begin{cases} 0 \le v \le 1 \\ 0 \le u \le 2\pi \end{cases}$

Maët noùn(Cone)

```
void DrawCone(float R, float h){
        Point3D P;      Point2D P1;   double Delta_U,Delta_V,u,v;
        Delta_U = 0.03;         Delta_V = 0.1;
        for (u=0; u<2*M_PI ;u+=Delta_U){
            for (v=0; v<1; v+=Delta_V){
                P.x = v*R*cos(u);   P.y = v*R*sin(u);
                P.z = v*h;                P1 = Chieu(KieuChieu,P);
                putpixel(xc+P1.x,yc+P1.y,WHITE);
            }
        }
}
```

Mặt tròn xoay

Taïo baèng caùch quay ñöôøng cong C quanh moät truïc naøo ñoù.

Giaû söû: Ñöôøng cong C thuoäc maët phaúng xOz, truïc quay laø truïc z

Goïi M(x, y, z) laø moät ñieåm thuoäc C.

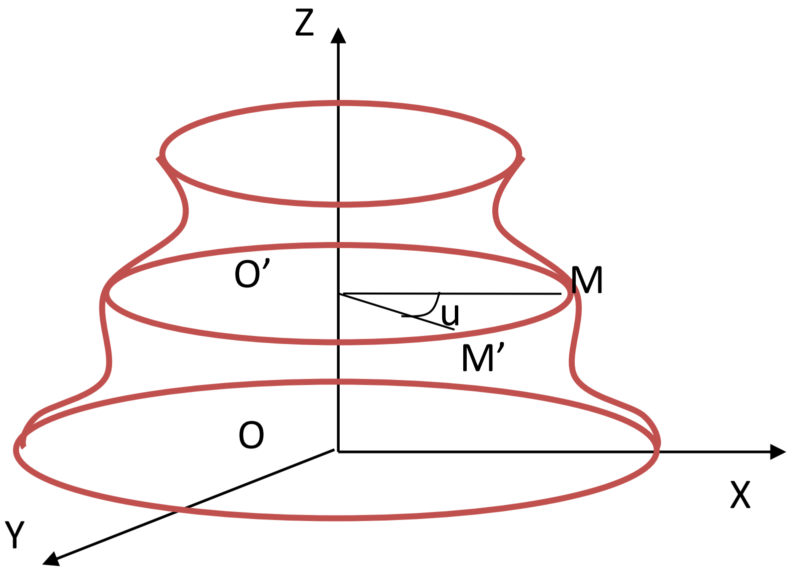
Ta coù   M(X(v),0,Z(v))

Cho M quay quanh truïc z moät goùc u ⇨M thuoäc maët phaúng z=Z(v)

Goïi aûnh cuûa M qua pheùp quay laø M’(x, y, z)

⇨ Tính M‘(x,y,z) qua X(v), Z(v) vaø u.

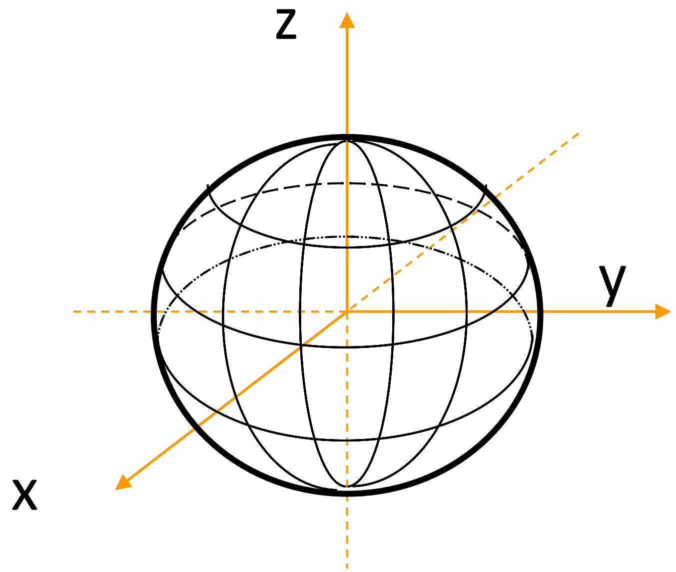
Ta coù: O’M=O’M’=X(v)



Hay: $$\begin{cases} x = X(v).\cos(u) \\ y = X(v).\sin(u) \\ z = Z(v) \end{cases}$$ vôùi $$\begin{cases} v_{min} \le v \le v_{max} \\ 0 \le u \le 2\pi \end{cases}$$

Mặt tròn xoay

Maët caàu: Vôùi maët caàu taâm O baùn kính r thì C laø ½ ñöôøng troøn trong maët phaúng xOz.

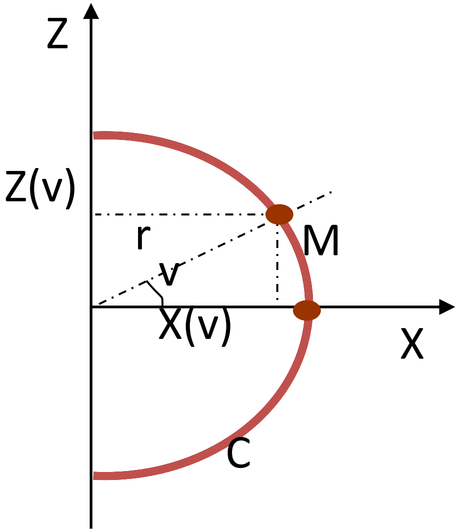


Ta coù: X(v)=r.cos(v);

Z(v)=r.sin(v)

$$\begin{cases} X(u,v) = r.\cos(u).\cos(v) \\ Y(u,v) = r.\sin(u).\cos(v) \\ Z(u,v) = r.\sin(v) \end{cases}$$

vôùi $\begin{cases} -\pi/2 \le v \le \pi/2 \\ 0 \le u \le 2\pi \end{cases}$



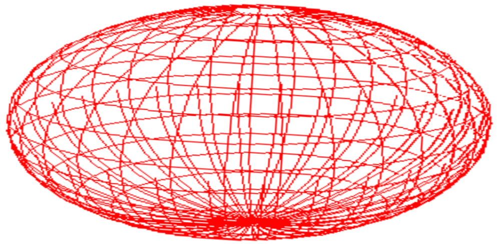
Mặt tròn xoay

Maët caàu:

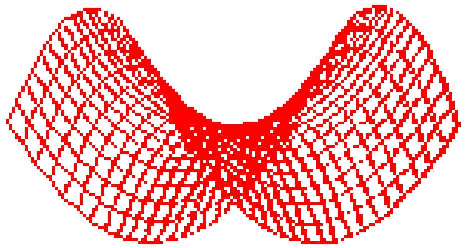
```
void DrawSphere(float R){
        Point3D P;   Point2D P1;        double Delta_U,Delta_V,u,v,Pi_2;
        Pi_2 = M_PI/2;Delta_U = 0.1; Delta_V = 0.1;
        for (v=-Pi_2; v<Pi_2 ;v+=Delta_V){
            for (u=0; u<2*M_PI; u+=Delta_U){
                P.x = R*cos(u)*cos(v);  P.y = R*sin(u)*cos(v);
                P.z = R*sin(v);            P1 = Chieu(KieuChieu,P);
                putpixel(xc+P1.x,yc+P1.y,GREEN);
            }
        }
}
```

Mặt tròn xoay

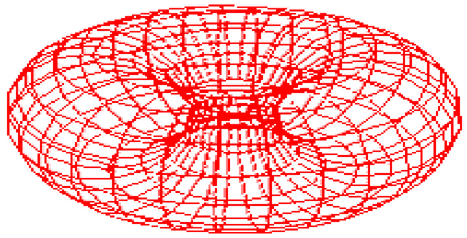
Mặt Ellipsoid: $\begin{cases} X(u,v) = R_x.\cos(u).\cos(v) \\ Y(u,v) = R_y.\sin(u).\cos(v) \\ Z(u,v) = R_z.\sin(v) \end{cases}$ vôùi $\begin{cases} -\pi/2 \le v \le \pi/2 \\ 0 \le u \le 2\pi \end{cases}$



Mặt Hypeboloid: $\begin{cases} X(u,v) = u \\ Y(u,v) = v \\ Z(u,v) = u^2 - v^2 \\ -1 \le u,v \le 1 \end{cases}$



Mặt xuyến toroid: $\begin{cases} X(u,v) = (R + a.\cos(v)).\cos(u) \\ Y(u,v) = (R + a.\cos(v)).\sin(u) \\ Z(u,v) = a.\sin(v) \\ 0 \le u \le 2\pi; \quad -\pi/2 \le v \le \pi/2 \end{cases}$



Mặt tròn xoay

Phương pháp chính ở đây là vẽ các đường đồng mức theo u và v.

Để vẽ một đường đồng mức u tại giá trị u’ khi v chạy từ V_{Min} đến V_{max} ta làm như sau:

-Tạo một tập hợp các giá trị $v[i] \in [V_{Min}, V_{Max}]$, xác định vị trí:

$$P[i] = (X(u',v[i]), Y(u',v[i]), Z(u',v[i])).$$

-Chiếu từng điểm P[i] lên mặt phẳng.

-Vẽ các đường gấp khúc dựa trên các điểm 2D P’[i].

vẽ một mặt cong, ta thực hiện các bước sau:

-Nhập các hệ số của phương trình mặt:

a, b, c, d, U_{min}, U_{max}, V_{min}, V_{max}.

-Tính các hàm 2 biến: X(u,v), Y(u,v), Z(u,v).

-Khởi tạo phép chiếu: Song song/Phối cảnh.

-Vẽ họ đường cong u.

-Vẽ họ đường cong v.

SUMMARY

- Đường cong

- Mặt cong

Enjoy the Course…!