Department of Applied Mathematics
and Computational Sciences
University of Cantabria
UC-CAGD Group

COMPUTER-AIDED GEOMETRIC DESIGN AND COMPUTER GRAPHICS:

HIGH LEVEL COMPUTATION PROGRAMS (*MATHEMATICA* AND *MATLAB*) IN CAGD AND COMPUTER GRAPHICS

Andrés Iglesias
e-mail: iglesias@unican.es
Web pages: http://personales.unican.es/iglesias
http://etsiso2.macc.unican.es/~cagd

Symbolic and Numerical computation programs

- In the last few years, the extraordinary advances in hardware and software have made possible the appearance of a new generation of Scientific Computation Programs (SCPs):

  - either symbolic, as Mathematica
  - or numerical, as Matlab

 # The SCPs are easier to use, because:

- they incorporate many mathematical and programming commands and libraries

- their algorithms are very optimized

- they have a powerful and user-friendly interface

 # The SCPs are very powerful, because:

- their programming languages incorporate not only the procedural but also the functional programming including, in several cases, pattern recognition and object-oriented programming.

- they have a very remarkable graphical capabilities.

 # The NCPs are very popular. Both Mathematica and Matlab are used for hundreds of thousands of industrial, government and academic users around the world.

Mathematica for CAGD and Computer Graphics

Mathematical expressions can be easily implemented in Mathematica; in most cases, this process consists of a simple translation of these expressions to its programming language.

We have developed a Mathematica package that incorporates the most usual curves and surfaces in CAGD and computer graphics.

First, loading the package

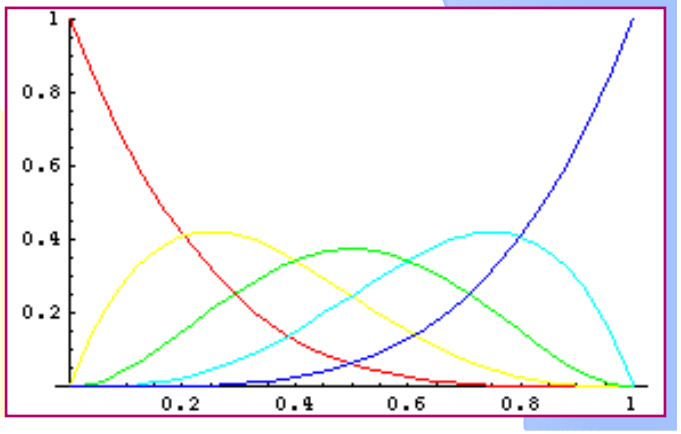
```
In[1]:= <<CAGD.m
```

```
Bernstein[i_,n_,v_]:=
    Binomial[n,i] (1-v)^(n-i) v^i
```

$$B_i^n(t) = \binom{n}{i}(1-t)^{n-i}t^i$$


```
In[3]:=  Bernstein[1,3,t]
Out[3]:= 3 (1-t)² t
```

```
In[4]:= Show[
          Table[Plot[Bernstein[i,4,t],
                {t,0,1},PlotRange->{0,1}],
          {i,0,4}]
        ]
```



Mathematica for CAGD and Computer Graphics

```
BezierCurve[pts:{{_,_}..}|{{_,_,_}..},v_]:=
 Module[{n=Length[pts]-1},
   Table[Bernstein[i,n,v],{i,0,n}].pts
          //Simplify
  ];
```

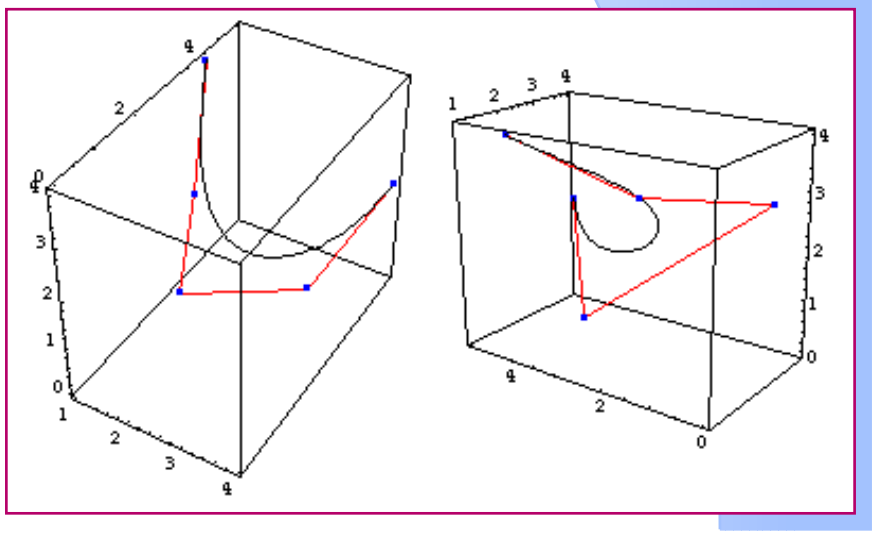
$$\sum_{i=0}^{n} \mathbf{P}_i B_i^n(t)$$

```
In[6]:= pts={{1,1},{2,4},{3,0},{4,3},{5,-1},{6,0}};

In[7]:=   BezierCurve[pts,t]
Out[7]:=  {1+5t, 1+15t-70t^2+140t^3-140t^4+54t^5}
```

We have obtained the symbolic expression for the Bézier curve !!!

The same commands work for two- and three- dimensional points.

```
In[8]:= ptos3D={{1,4,4},{2,2,3},
         {3,0,3},{3,4,0},{4,5,2}};

In[9]:= PlotBezierCurve[ptos3D,t]
```



Mathematica for CAGD and Computer Graphics

BEZIER SURFACES

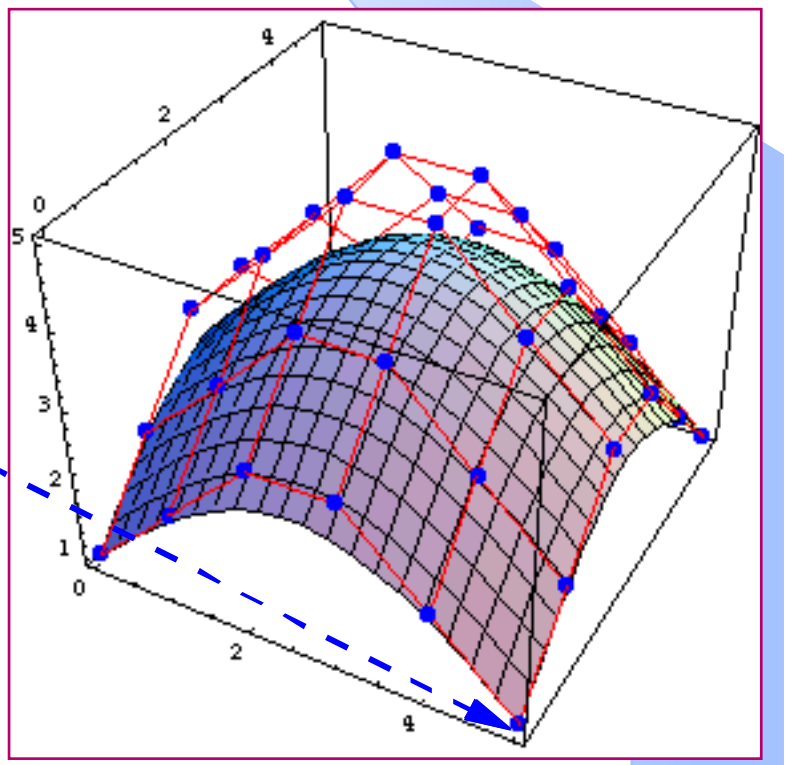
The previous commands can be easily generalized to the surface case. For example:

```
BezierSurf[pts:{{{_,_,_}..}..},{u_,v_}]:=
  Module[{m=Length[pts]-1,U,V,
          n=Length[First[pts]]-1},
      {U,V}=MapThread[
          Table[
            Bernstein[i,#1,#2],
              {i,0,1}]&,{{m,n},{u,v}}
           ];
        Plus @@ (U.pts*V) //Simplify
    ]
```

$$\mathbf{X}(u,v) = \sum_{i=0}^{n}\sum_{k=0}^{m} \mathbf{P}_{ik} B_i^n(u) B_k^m(v)$$

x	y	z	x	y	z	x	y	z	x	y	z	x	y	z	x	y	z
0	0	1	0	1	2	0	2	3	0	3	3	0	4	2	0	5	1
1	0	2	1	1	3	1	2	4	1	3	4	1	4	3	1	5	2
2	0	3	2	1	4	2	2	5	2	3	5	2	4	4	2	5	3
3	0	3	3	1	4	3	2	5	3	3	5	3	4	4	3	5	3
4	0	2	4	1	3	4	2	4	4	3	4	4	4	3	4	5	2
5	0	1	5	1	2	5	2	3	5	3	3	5	4	2	5	5	1

Table of 3-dimensional control points

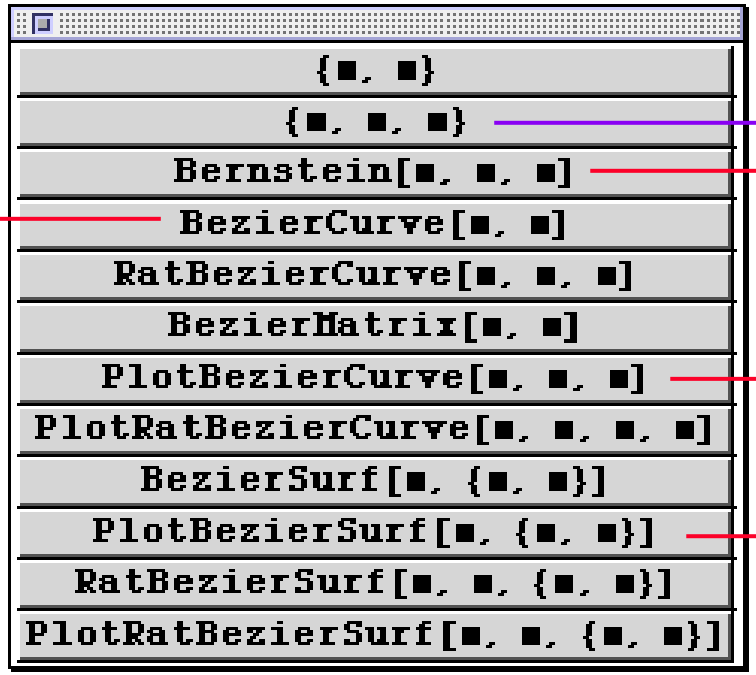


Mathematica for CAGD and Computer Graphics

MATHEMATICA PALETTES

In Mathematica 3.0 or higher users can easily create palettes for incorporating their own commands.

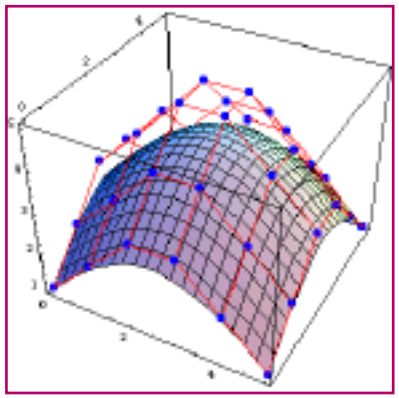
Mathematica 3.0 palettes




```
In[]:=  Bernstein[1,3,u]
Out[]:= 3 (1-u)^2 u
```

```
In[]:= ptos3D={{1,4,4},{2,2,3},
       {3,0,3},{3,4,0},{4,5,2}};


In[]:= PlotBezierCurve[ptos3D,t]
```

```
In[]:= PlotBezierSurf[ptos3D,{u,v}]
```



```
In[]:=  BezierCurve[pts,t]
Out[]:= {1+5t,
      1+15t-70t^2+140t^3-140t^4+54t^5}
```

Mathematica for CAGD and Computer Graphics

The CAGD.m package can be applied for educational purposes!!!

EXAMPLE

We know that the Bézier curves hold the convex hull property: the curve lies entirely within the convex polygon determined by its control points.

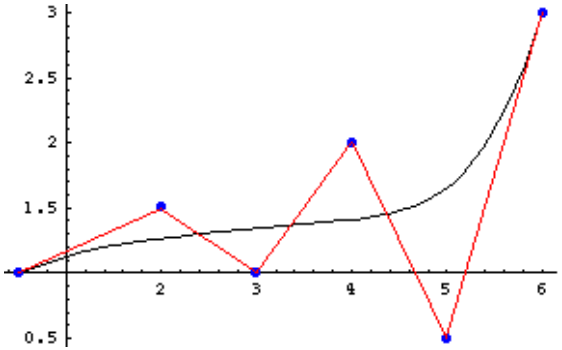
We introduce some control points.

```
{■, ■}
```

```
In[1]: p1={{1/2,1},{2,3/2},
        {3,1},{4,2},{5,1/2},{6,3}};
```

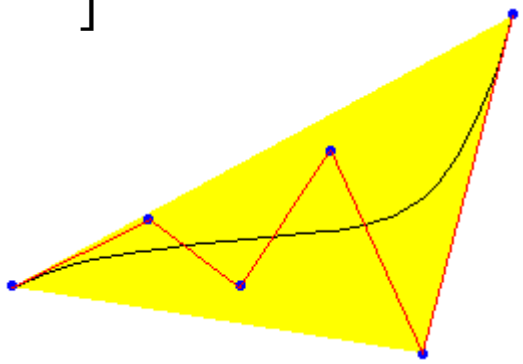
```
PlotBezierCurve[■, ■, ■]
```

```
In[2]: c1=PlotBezierCurve[p1,t];
```



We take advantage of the computational geometry Mathematica packages, avoiding to calculate the convex hull of the given list of points:

```
In[3]: <<DiscreteMath`
        ComputationalGeometry`

In[4]: ConvexHull[p1];
        Part[p1,#]& /@ %
Out[4]: {{6,3},{1/2,1},{5,1/2}}

In[5]: Show[
          {Graphics[
            {RGBColor[1,1,0],
             Polygon[%]}],c1}
        ]
```



Mathematica for CAGD and Computer Graphics

EXAMPLE: TWO DIMENSIONAL TRANSFORMATIONS

This example shows how the previous commands can be combined with two-dimensional transformations.

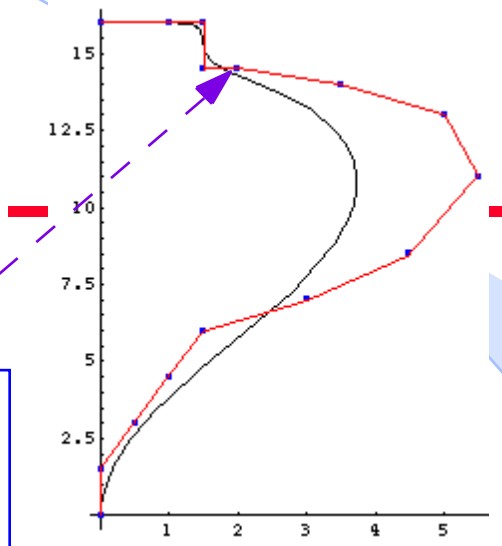
List of 22 control points

```
spitop={{0,16},{1,16},{1.5,16},{1.5,16},{1.5,16},
        {1.5,16},{1.5,14.5},{1.5,14.5},{1.5,14.5},
        {1.5,14.5},{1.5,14.5},{2,14.5},{3.5,14},
        {5,13},{5.5,11},{4.5,8.5},{3,7},{1.5,6},
         {1,4.5},{0.5,3},{0,1.5},{0,0}};
```

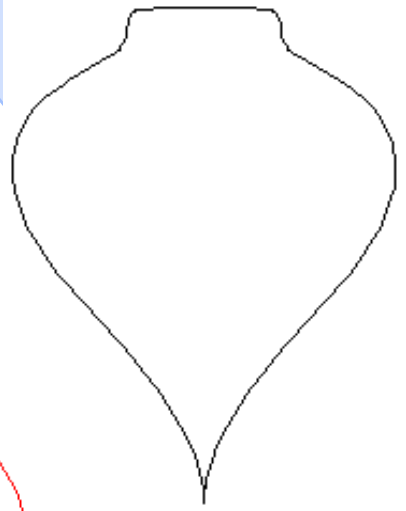
```
In[1]:= BezierCurve[spitop,t]
```

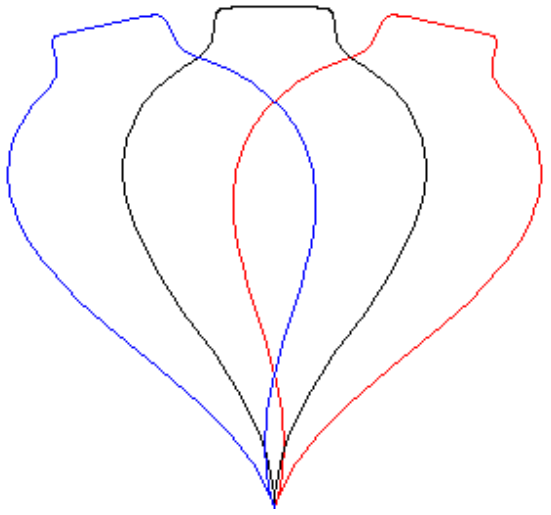
Out[1]: $\{21\,t - 105\,t^2 + 2992.5\,t^4 - 20349\,t^5 + \ldots + 4.36432\,106\,t^{19} - 737383\,t^{20} + 59270.5\,t^{21}\}$

A 21-degree polynomial !!!!



Reflection



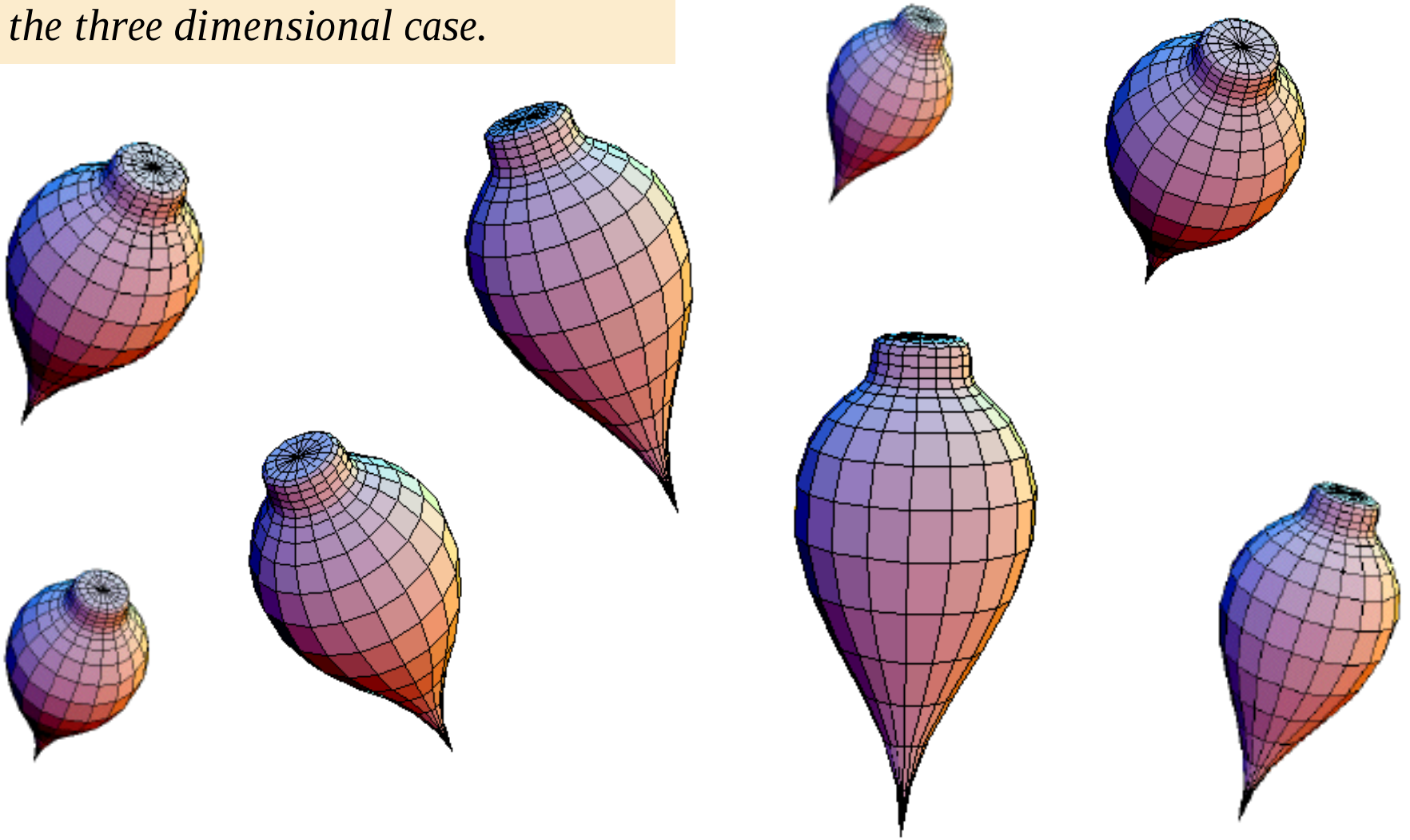


Rotations

Mathematica for CAGD and Computer Graphics

EXAMPLE: THREE DIMENSIONAL TRANSFORMATIONS

This previous example can be extended to the three dimensional case.



THE CONTEXT

Many of the most important programs for computer graphics and CAGD have been written in traditional programming languages (Fortran, Pascal, C, etc...)

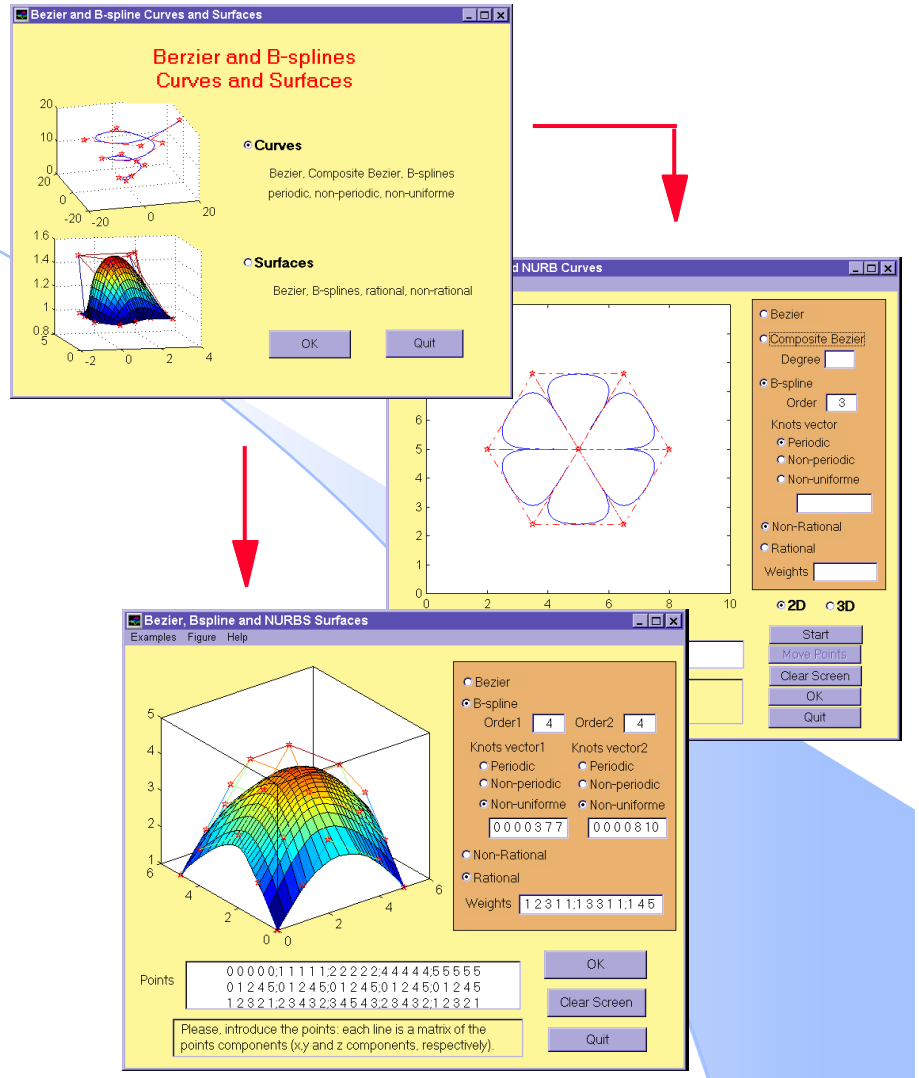
BUT...

NOWADAYS

the general-purpose numerical (Matlab, Scilab...) and symbolic (Mathematica, Maple, Axiom...) computation programs are gaining more and more popularity.

OUR TARGET...

To analyze the possibilities of applying a numerical computation program (MATLAB), instead of the traditional programming languages, to CAGD and the computer graphics fields.



An Interactive MATLAB Program for CAGD **(A. Iglesias, A. Gálvez)**
IX Int. Conf. on Comp. Graph. and Vision
Moscow, Russia, 26 Aug.-1 Sept. 1999

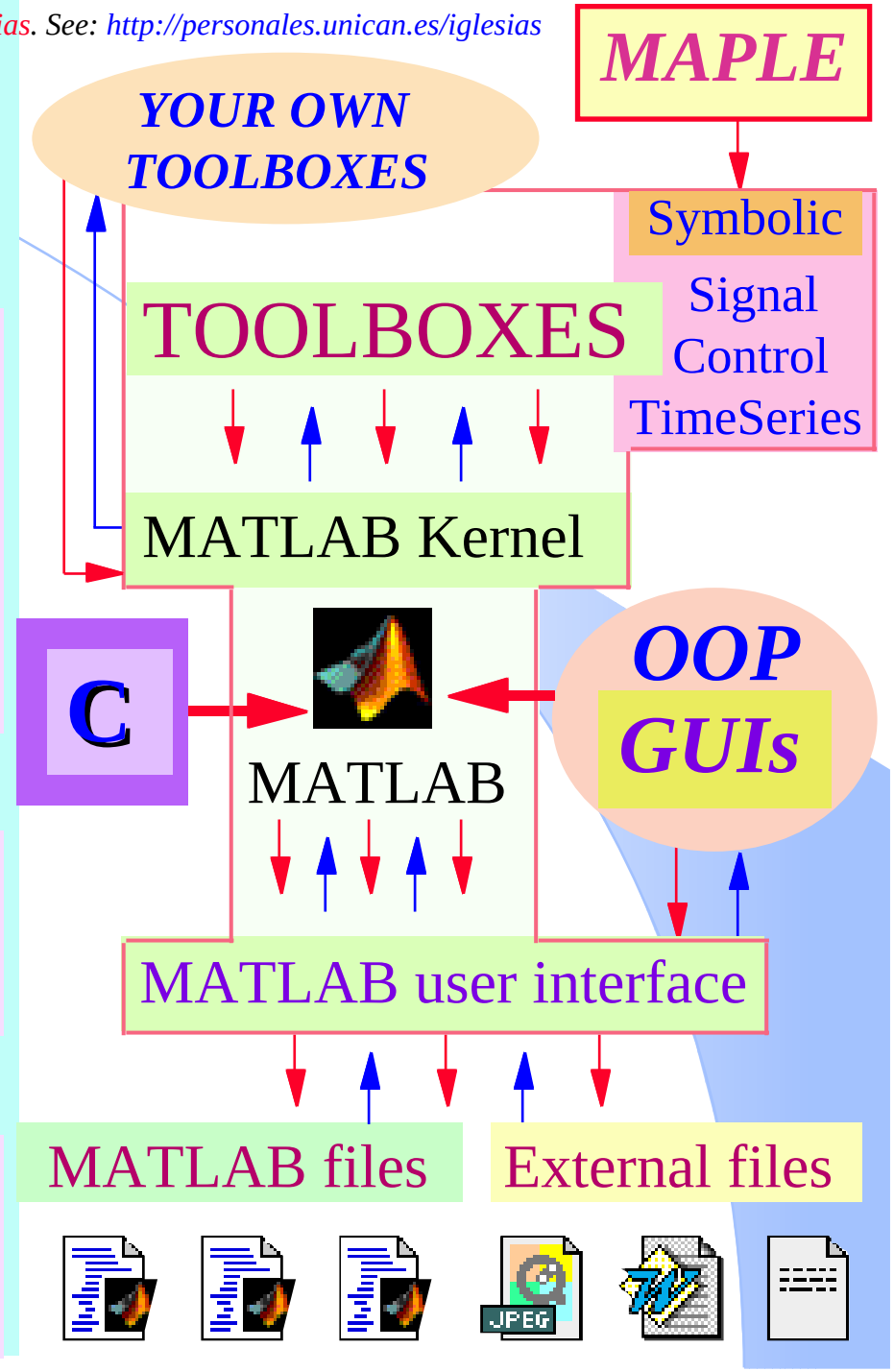
MATLAB

WEB PAGE: www.mathworks.com

- **Spreading:**
  - Hundreds of thousands of users.
  - Industrial, academic and research environments.
  - Available versions for Windows 95, 98 and NT, Macintosh, UNIX, VMS, Linux, Digital, etc...
- **Graphical capabilities.**

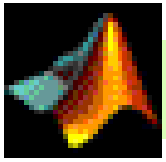
Plotting 2D and 3D data, patches, hidden line removal, colors, lighting, reflectances, texture mapping, files management, etc...

- **Programming language:**
  - It is based on C.
  - Arrays do not require dimensioning.
  - Incorporates functional programming.



Building numerical libraries for CAGD

MATLAB incorporates:



MATLAB Kernel

- Basic commands for interpolation:
  - 'nearest' - nearest neighbor
  - 'linear' - linear
  - 'spline' - cubic spline
  - 'cubic' - cubic

TOOLBOXES

- A *Spline Toolbox* by Carl de Boor.
  - difficult to understand
  - it is very limited
  - not so useful for industry
  - it lacks of many important commands in CAGD

TASK:

Implementation of an extensive set of numerical libraries for CAGD.

- Bézier curves and surfaces: both rational and nonrational Bézier and composite Bézier.

- B-splines curves and surfaces: for any order and knots vector (periodic, nonperiodic, nonuniform) and weights (including NURBS).

- Two- and three-dimensional transformations.

- Projections and perspectives.

An illustrative example: Bézier curves

[See: G. E. Farin. Curves and Surfaces for CAGD, 3rd ed., Academic Press, San Diego (1993)**]**

The main function:

```
function Bezier(ptos)
[n,d]=size(ptos);
n=n-1;
bt=ptos'*mij(n)*ti(n);
if d==2
plot(bt(1,:),bt(2,:),ptos(:,1),ptos(:,2),'r-.p')
else
plot3(bt(1,:),bt(2,:),bt(3,:), ...
      ptos(:,1),ptos(:,2),ptos(:,3),'r-.p')
end
rotate3d
```

Defining the factorial and binomial function:

```
function f=factorial(n)
if n==1
      f=1;
else
      f=n*factorial(n-1);
end
```

```
function c=binom(n,i)
if i==n | i==0
   c=1;
elseif i<n & i>=0
   c=factorial(n)/
         (factorial(i)*factorial(n-i));
else
   c=0;
end
```

Auxiliar functions:

```
function T=ti(n)
m=1;
t=0:0.05:m;    % step=0.05
T=[];
for i=0:n
   T= [T;t.^i];
end
```

```
function M = mij(n)
for i=0:n
   for j=0:n
     M(i+1,j+1)=(-1)^(ji)*binom(n,j)*binom(j,i);
   end
end
M=M(1:n+1,1:n+1);
```

The case of Bézier surfaces

[See: G. E. Farin. Curves and Surfaces for CAGD, 3rd ed., Academic Press, San Diego (1993)**]**

The main function:

```
function SupBezier(ptos)
[m,n,o]=size(ptos);
for k=1:3
b(:,:,k)=ti(m-1)'*mij(m-1)'*ptos(:,:,k)...
        *mij(n-1)*ti(n-1);
end
surf(b(:,:,1),b(:,:,2),b(:,:,3)), hold on,
mesh(ptos(:,:,1),ptos(:,:,2),ptos(:,:,3)),
hidden off
plot3(ptos(:,:,1),ptos(:,:,2),ptos(:,:,3),'bp')
rotate3d
```

Defining the factorial and binomial function:

```
function f=factorial(n)
if n==1
     f=1;
else
     f=n*factorial(n-1);
end
```

```
function c=binom(n,i)
if i==n | i==0
  c=1;
elseif i<n & i>=0
  c=factorial(n)/
        (factorial(i)*factorial(n-i));
else
  c=0;
end
```

Auxiliar functions:

```
function T=ti(n)
m=1;
t=0:0.05:m;    % step=0.05
T=[];
for i=0:n
  T= [T;t.^i];
end
```

```
function M = mij(n)
for i=0:n
  for j=0:n
    M(i+1,j+1)=(-1)^(ji)*binom(n,j)*binom(j,i);
  end
end
M=M(1:n+1,1:n+1);
```

Applications to industry

In an European project we propose the use of symbolic and numerical tools for CAGD in industry.

(Ref: 1FD97-0409)

UNICAN GD
Cantabria University
CAGD Group
Department of Applied Mathematics and Computational Sciences

Computer-Aided Geometric Design Group
University of Cantabria

http://etsiso2.macc.unican.es/~andres/UC-CAGD/

MATHEMATICA

MATLAB

www.candemat.com

MAPLE

C A G - Cantabria
Computational Algebra and Geometry

Computational Algebra and Geometry Group
University of Cantabria
http://matsun1.matesco.unican.es/CAG/

Applications to industry

CDM

CANDEMAT

IGES FILE

Numerical

MATLAB

Symbolic

Mathematica®

Curves and surfaces

1 Reading

2 Evaluation

3 Representation

Interactivity

Patches

Reflectance

Textures

Mesh

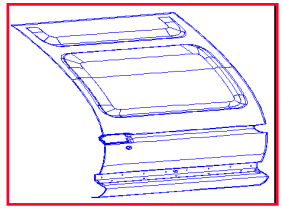
4 Visualization

Lighting

Color

Wireframe

Tasks to be done



IGES FILE

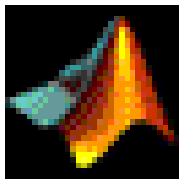
- *Build a IGES-MATLAB Converter*
  - *MATLAB file management*

(The formats TIFF, JPEG, BMP, PCX, XWD and HDF are available)

- *Applying the CAGD MATLAB Toolbox*
  - *We shall obtain both a numerical and a graphical output*
- *Visualization*
  - *Taking advantage of the MATLAB graphical capabilities*
  - *Generating animations*
  - *Converting to VRML language*

VRMLplot by Craig Sayers

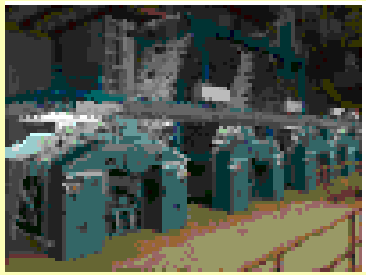
www.dsl.whoi.edu/DSL/sayers/VRMLplot



MATLAB

VRML file

WEB BROWSER



Interactive Graphical Output