

Lecture

Parametric design overview & geometry notions

Moran Mizrahi

Advanced Topics in Digital Design

67682

The Rachel and Selim Benin School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel

Outline

- Introduction to class
- Parametric design
 - Overview
 - Formal definition
 - Traditional classes of parametric models.
- Geometry notions (2D)
 - Formal definitions and primitives
 - Geometric transformations
 - Curves

Outline

- **Introduction to class**
- Parametric design
 - Overview
 - Formal definition
 - Traditional classes of parametric models.
- Geometry notions (2D)
 - Formal definitions and primitives
 - Geometric transformations
 - Curves

Introduction to class

- In this class, we will study and practice principles of computational design with emphasis on parametric design and computer-aided design SW.
- The class will include a theoretical discussion on hybrid design and craft and their intersections with the digital realm.

Requirements

- Attendance 5%
- Paper presentation 10%
- Assignments 30%
- Final project 55%

Books

- **Elements of Parametric Design.** Robert Woodbury
Routledge, 2010
- **Parametric Design for Architecture.** Wassim Jabi.
Laurence King Publishing, 3 Sep 2013
- **The Craft Reader.** Front Cover Glenn Adamson
Bloomsbury Academic, 15 Feb 2010

Outline

- Introduction to class
- **Parametric design**
 - Overview
 - Formal definition
 - Traditional classes of parametric models
- Geometry notions (2D)
 - Primitives
 - Geometric transformations
 - Curves

Outline

- Introduction to class
- Parametric design
 - **Overview**
 - Formal definition
 - Traditional classes of parametric models
- Geometry notions (2D)
 - Primitives
 - Geometric transformations
 - Curves

Parametric design overview

- Twenty years ago, the practice of computational modeling and design was accessible mostly for experts such as architects and mechanical engineers.
- Nowadays, because of the emergence of digital fabrication technologies, such as 3D printers, laser cutters, and CNC machines, the use of digital modelling software has become more common.
- There is a vast number of computer-aided design tools to build digital models, such as Blender, SolidWorks, Maya, Rhino, ZBrush, etc., and designers learn as part of their studies how to use it to fit their needs and to realize designs and ideas.

Parametric design vs. direct manipulation

Today, most CAD tools used to build digital models are based on either a ***direct manipulation*** paradigm or a ***parametric design*** paradigm.

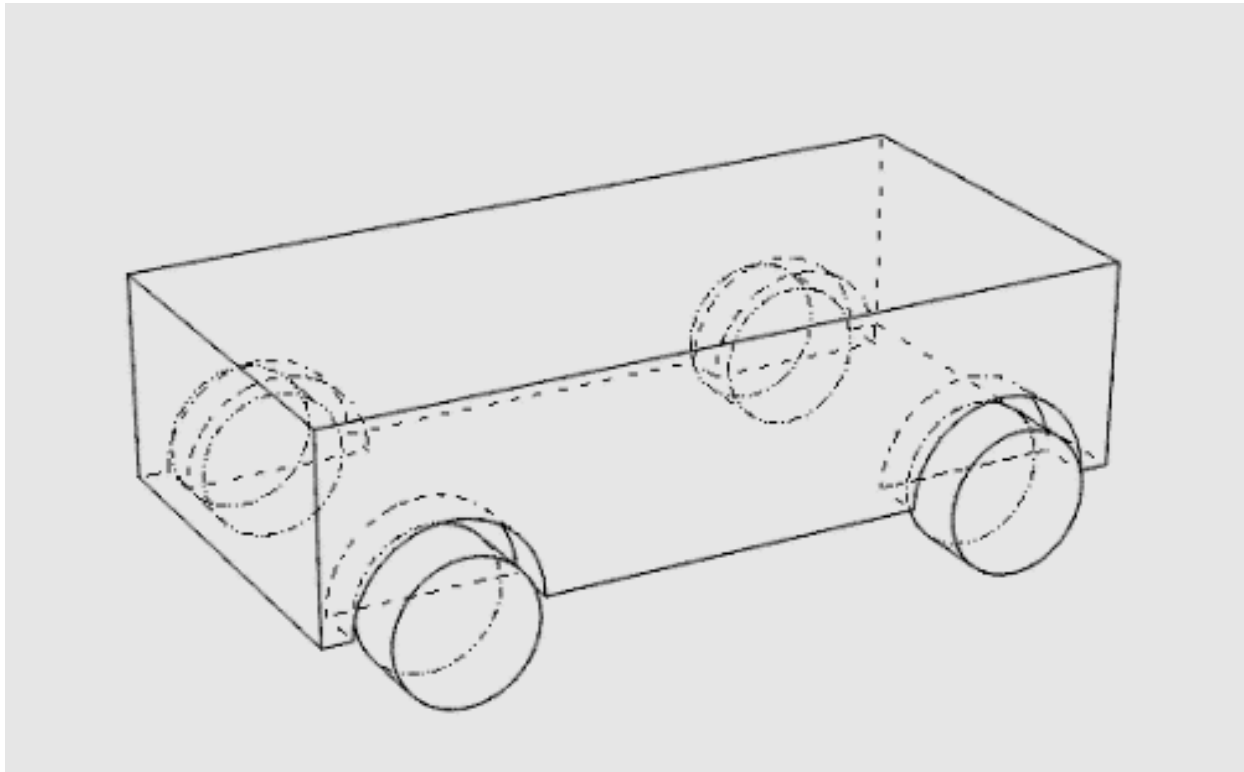
Parametric design vs. direct manipulation

Direct modeling manipulation tools (pros and cons):

- Intuitive and mimic the real-world design process (which is iterative, and consists of many top-down and bottom-up passes).
- In each iteration, the designer refines and tunes one of the components of the model based on his/her prior experience. However, a change to one aspect of the obtained model is cumbersome and time-consuming because it might require extensive low-level modifications. Of course, this problem becomes more severe with 3D geometrical complex models.

Parametric design vs. direct manipulation

Modeling a simple toy car:



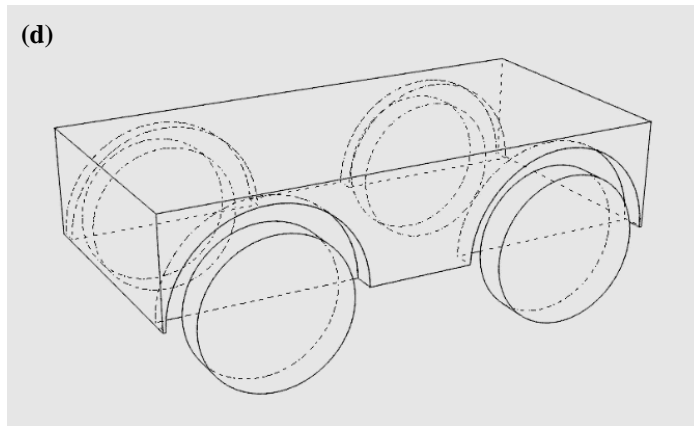
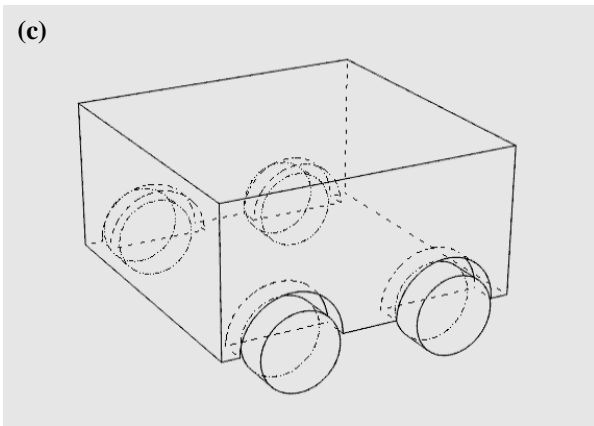
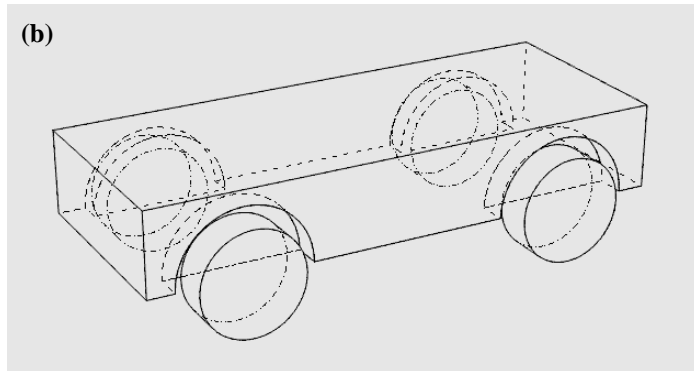
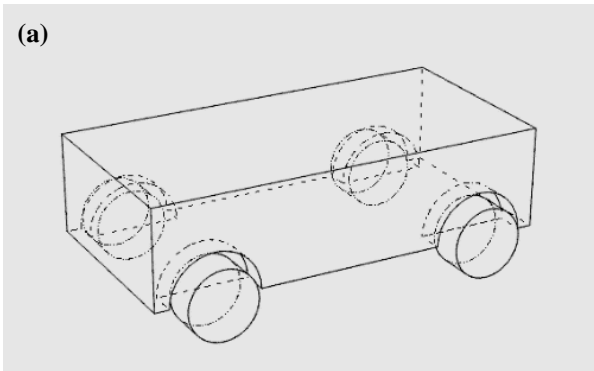
Parametric design vs. direct manipulation

Modeling a simple toy car (associative rules):

1. A modification to the radius of a wheel will result in the same modification for all other wheels.
2. A translation in one wheel location will result in a change in the other wheel locations respectively.
3. A modification to the radius of a wheel will result in a suitable modification to its bumper.
4. A modification to the radius of a wheel will result in a suitable modification to the car height.

Parametric design vs. direct manipulation

Modeling a simple toy car (parametric model):



Parametric design vs. direct manipulation

Parametric design tools (pros and cons):

- Less intuitive (the designer must define the free parameters and the relationships between them), requires a good geometrical understanding.
- A small change to the model does not require extensive modifications.

Parametric design

- Parametric design is not a new concept, and it constitutes an important modeling paradigm in computer-aided design.
- However, until recently, parametric design "was understood as highly sophisticated and expensive software made exclusively for manufacturing in the aerospace, shipping and automobile industries".
- The revolution in digital fabrication technology pushed the concept of parametric design beyond its original boundaries, and it can also be found today in standard common CAD tools.

Parametric design - resurgence

- The resurgence of the field in its new form started in architecture.
- Patrick Schumacher even coined the term "Parametricism" as the name of a new movement in architecture after "Modernism".
- In his paper "Parametricism", Patrick Schumacher defined the fundamental themes in parametric design as versioning, iteration, mass-customization, and continuous differentiation.

Parametric design - constraints

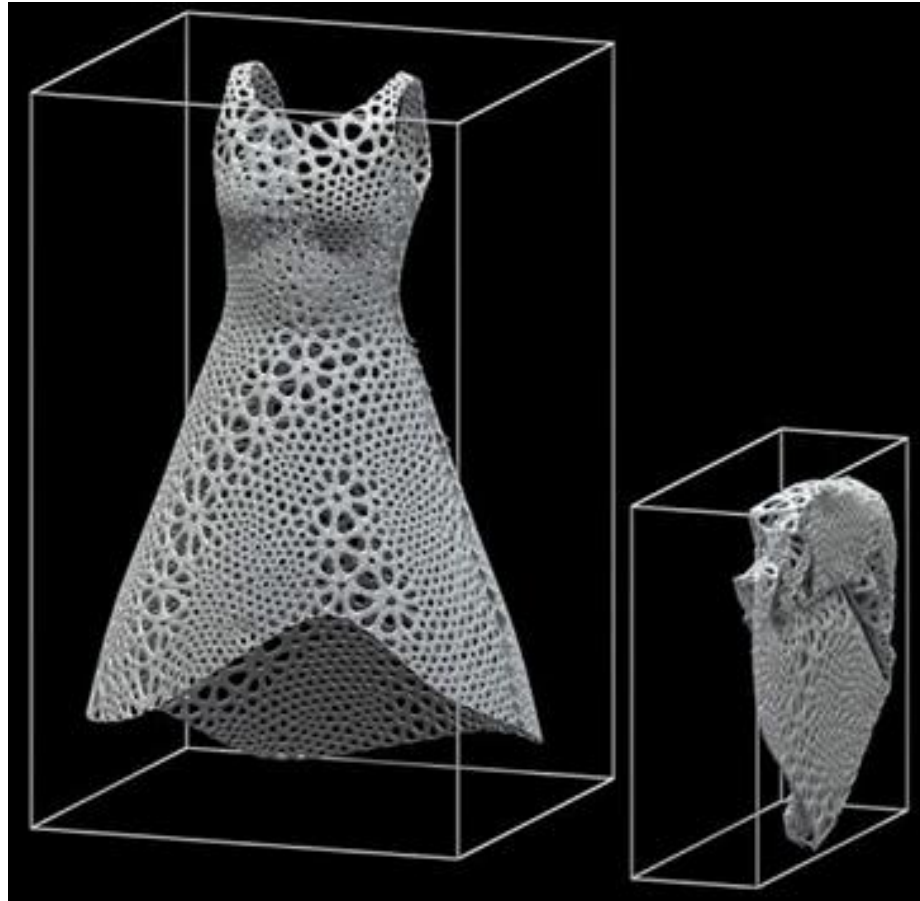
- Parametric design tools allow designers to predefine weak and strong constraints on their desired parametric model.
- These constraints can derive from the geometrical model, structural and thermal forces on the model, material behavior, contextual conditions, and many more factors.
- Defining a parametric model based on constraints and free parameters yields an infinite design space in which every modification of the free parameters results in a different variation of the model that fulfills the given constraints.

Parametric design - architecture



Examples of parametric buildings: the Aviva Stadium in Dublin (left); and the Al Bahr Towers in Abu Dhabi (right).

Parametric design – jewelry making and fashion design (Kinematics)

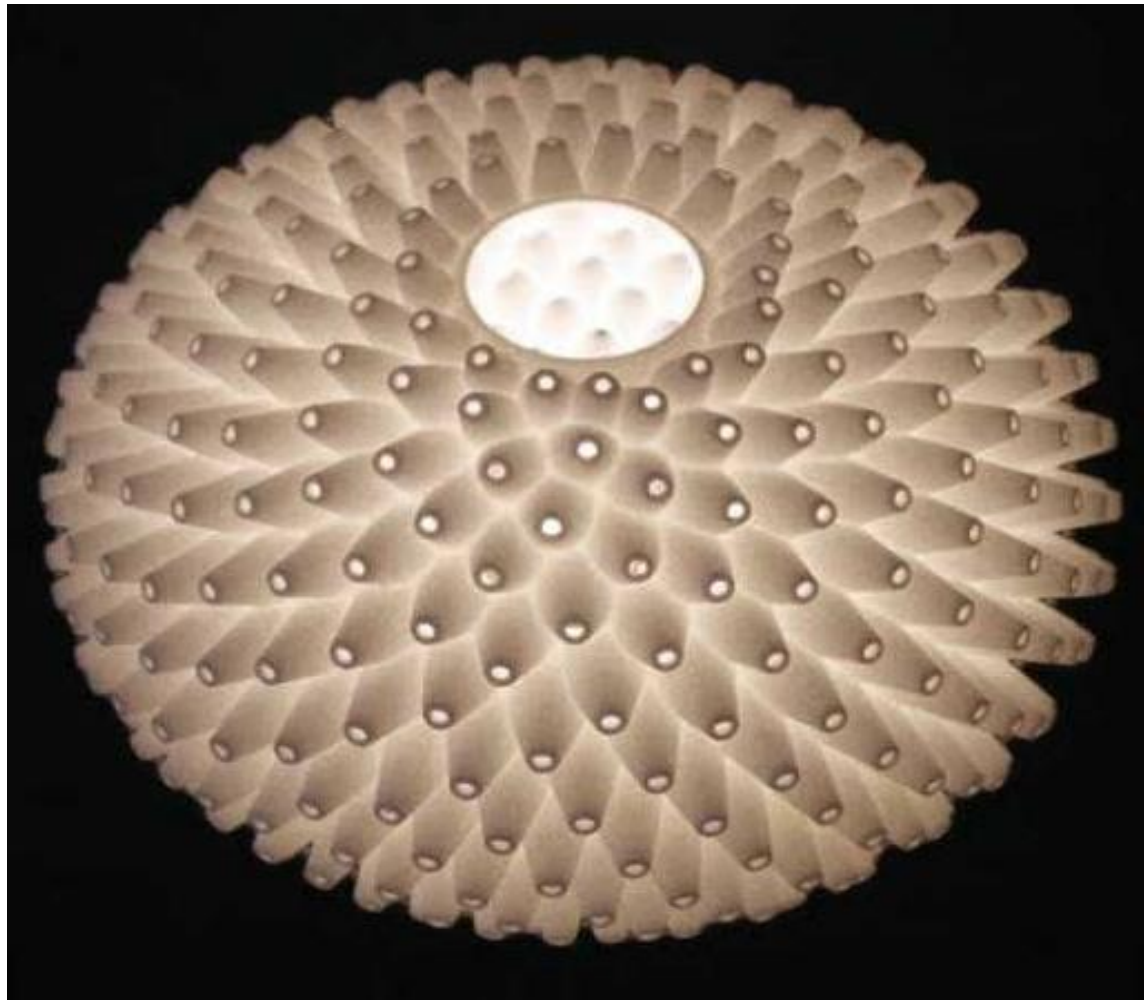


Kinematics project fold simulation (reduced the size of the dress by 85% as a single piece)

Parametric design – Nuala O'Donovan



Parametric design – Janne Kyttanen



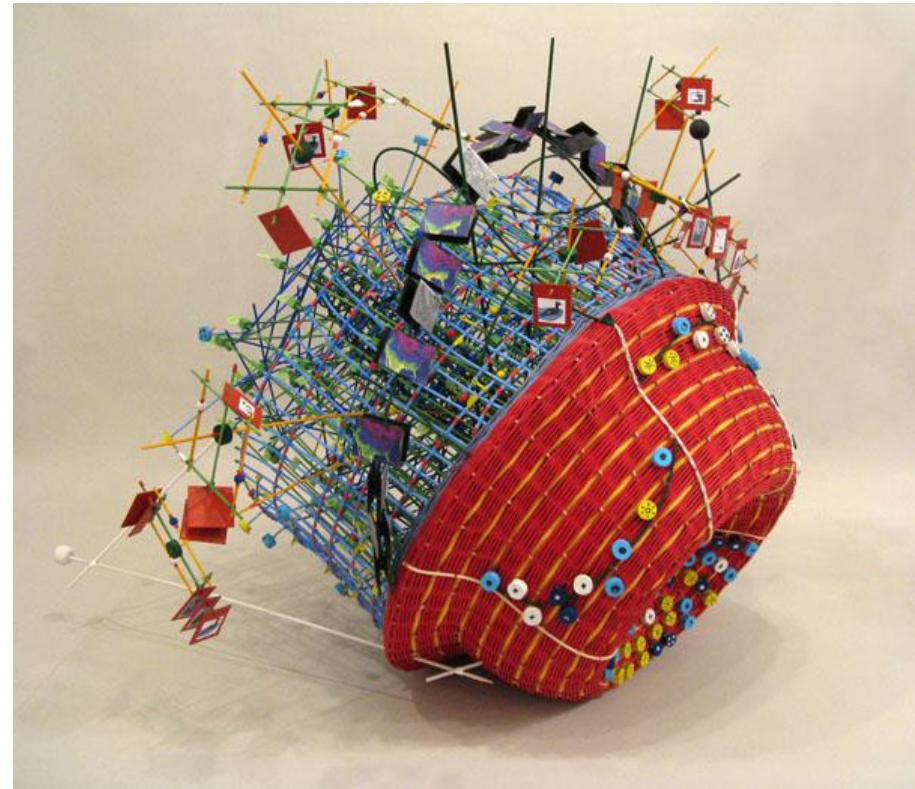
Parametric design – Michael Hansmeyer



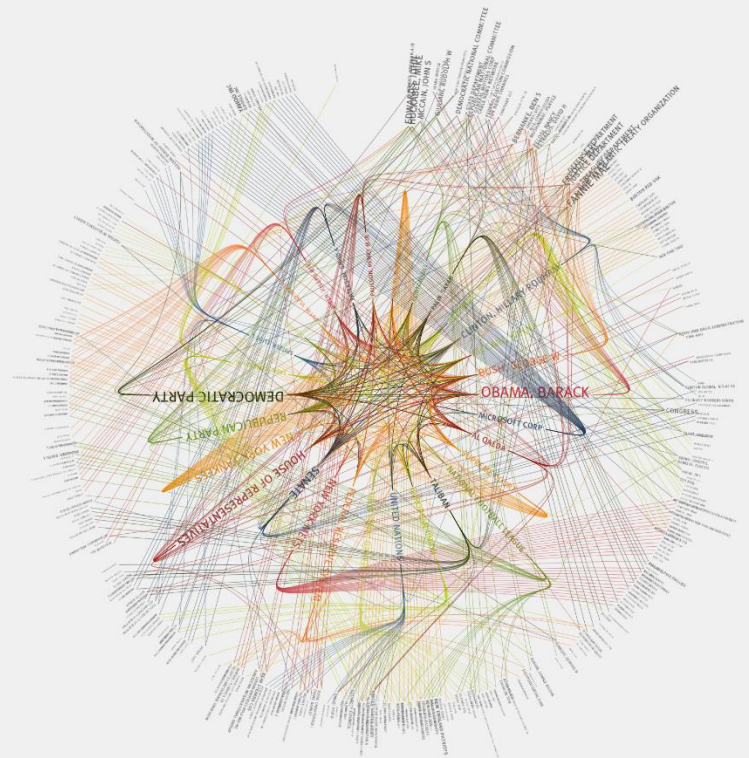
Parametric design – Data visualization



Natalie Miebach



1989



Outline

- Introduction to class
- Parametric design
 - Overview
 - **Formal definitions**
 - Traditional classes of parametric models
- Geometry notions (2D)
 - Primitives
 - Geometric transformations
 - Curves

Parametric modeling systems terms

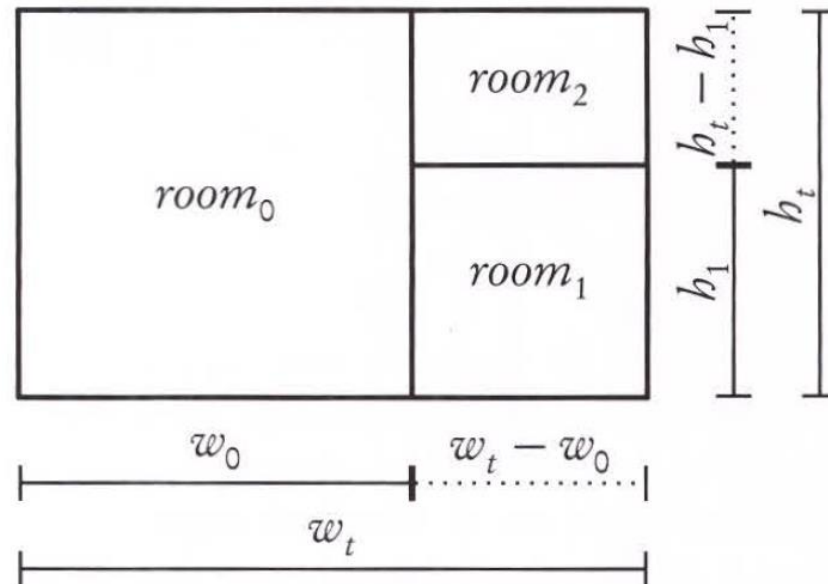
- A ***directed graph*** contains ***nodes (vertices)*** that are connected by ***links (directed edges)***.
- In parametric modeling, nodes are ***properties***.
- Nodes (properties) used in a constraint expression are predecessors of the node (property) holding the expression.
- A ***source node*** has no predecessor nodes. A ***sink node*** has no successor nodes. An ***internal node*** is neither source nor sink.

Parametric design definition

- A *parametric design* is a directed graph of properties (nodes) and links between them.
- A **well-formed parametric design** has no cycles - it is a directed acyclic graph.

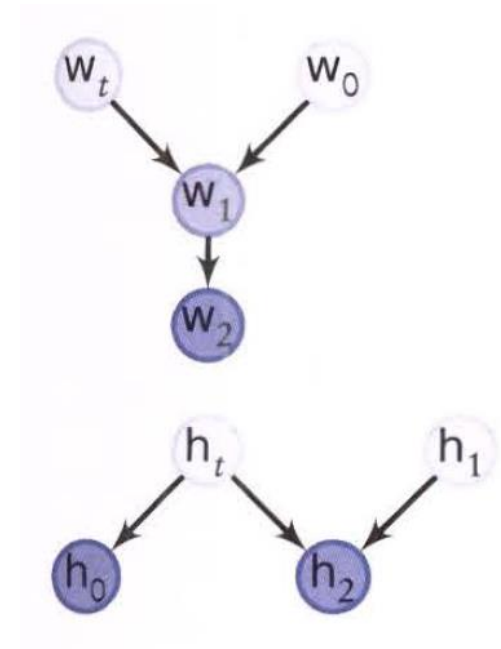
Parametric design definition - example

- Consider a plan of an apartment contains three rooms: $room_0$, $room_1$, $room_2$.
- Each $room_i$ has a width w_i and a height h_i .
- The total width is w_t ; the total height is h_t .




Parametric design definition - example

- w_t and w_0 are independent.
- w_1 and w_2 are dependent:
 $w_t - w_0 \rightarrow w_1$ and $w_1 \rightarrow w_2$.
- h_t and h_0 are independent.
- Whereas h_0 and h_2 are dependent:
 $h_t \rightarrow h_0$ and $h_t - h_1 \rightarrow h_2$.

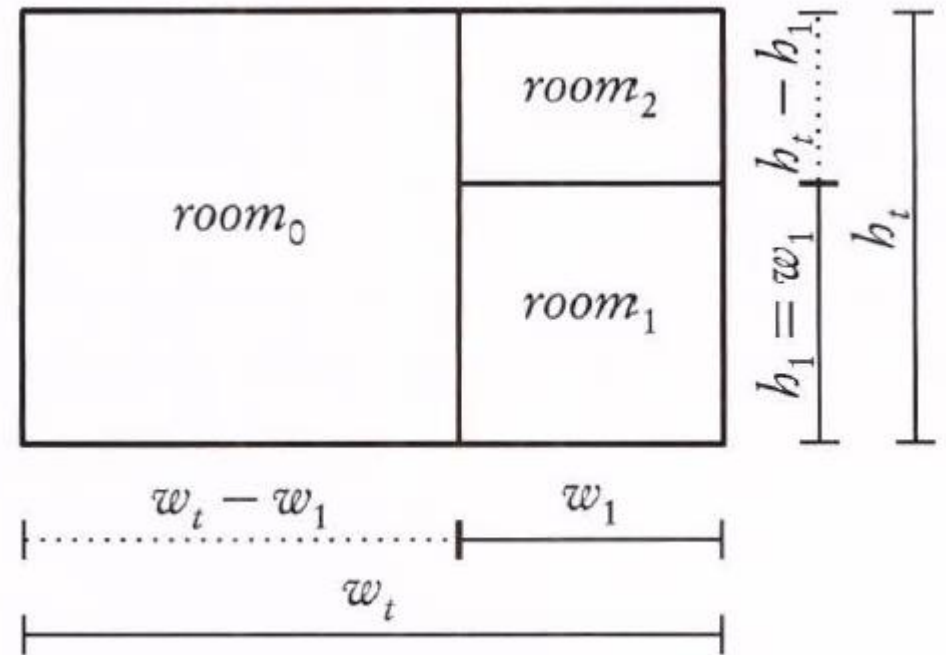
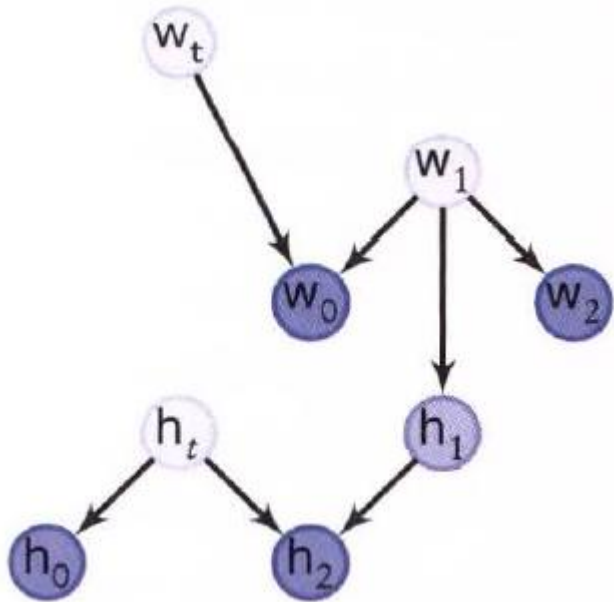


 - source node

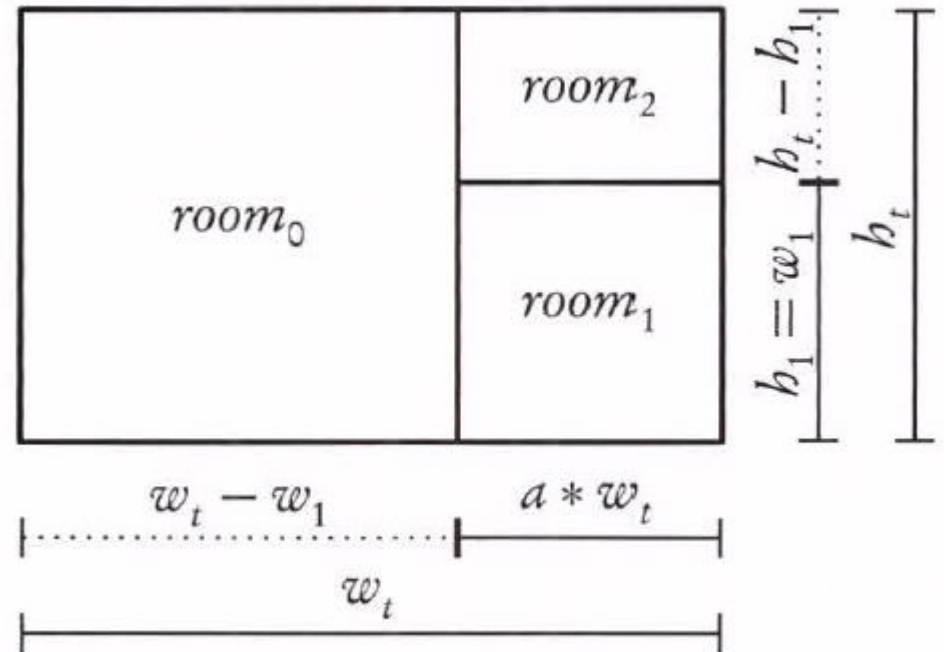
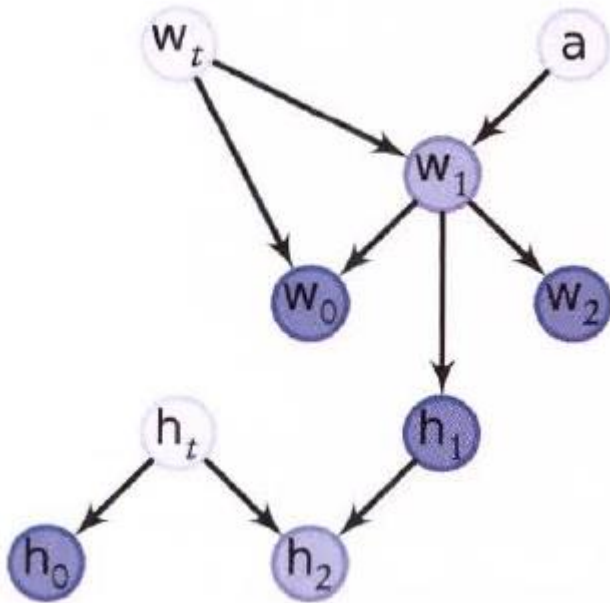
 - internal node

 - sink node

Parametric design definition - example



Parametric design definition - example



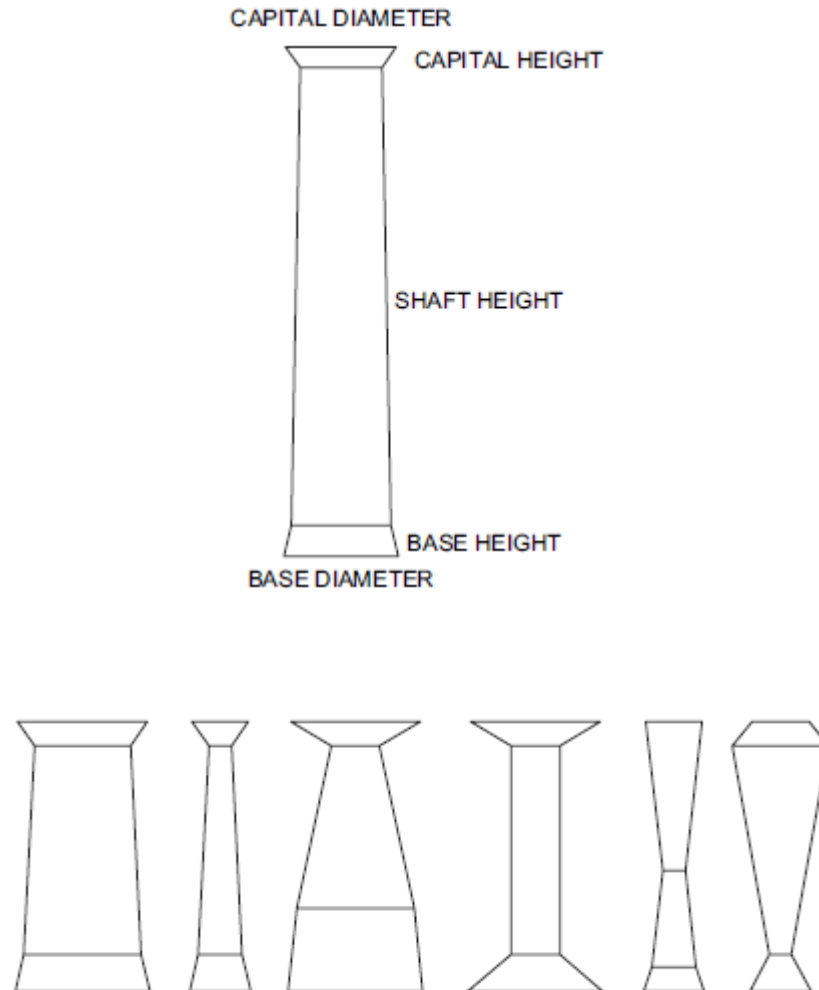
Outline

- Introduction to class
- Parametric design
 - Overview
 - Formal definitions
 - **Traditional classes of parametric models**
- Geometry notions (2D)
 - Formal definitions and primitives
 - Geometric transformations
 - Curves

Traditional classes of parametric models (PV)

- ***Parametric Variations (PV)*** is a kind of parametric model based on the declarative nature of the parameters to construct shapes.
- The designer creates a geometrical model of any kind, and its attributes are parameterized based on the desired behavior, thus creating a ***parameterized modeling schema***.
- A parametric modeling schema shows which attributes of a geometrical model are parameterized and how the designer can change the values of the parameters.

Traditional classes of parametric models (PV)

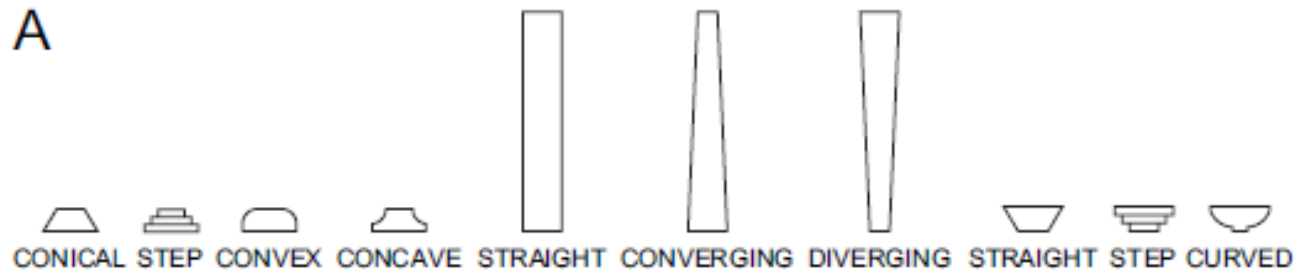


Traditional classes of parametric models (PC)

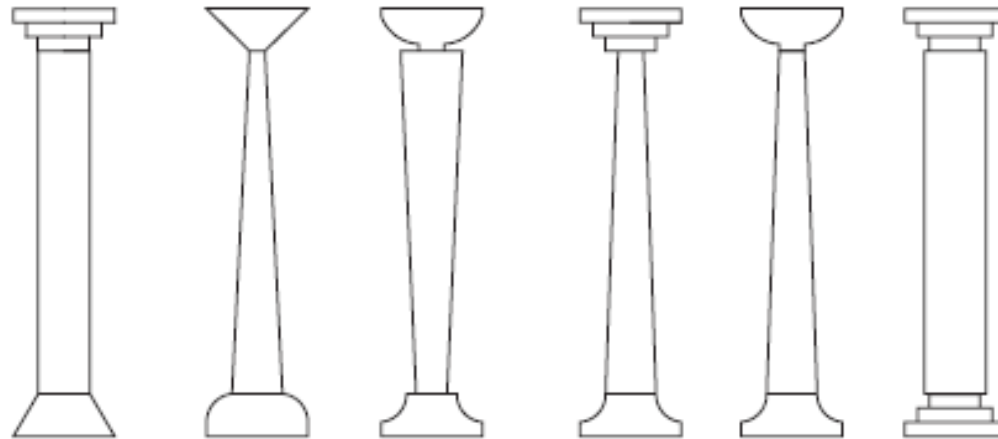
- ***Parametric Combinations (PC)*** is the second class of parametric models that is most used. A PC model is composed of a series of geometrical shapes that are arranged according to rules that create more complex structure.
- PC offers another degree of complexity beyond the parameterization of the geometrical components, which is done by constructing combinations according to specific rules.

Traditional classes of parametric models (PC)

A



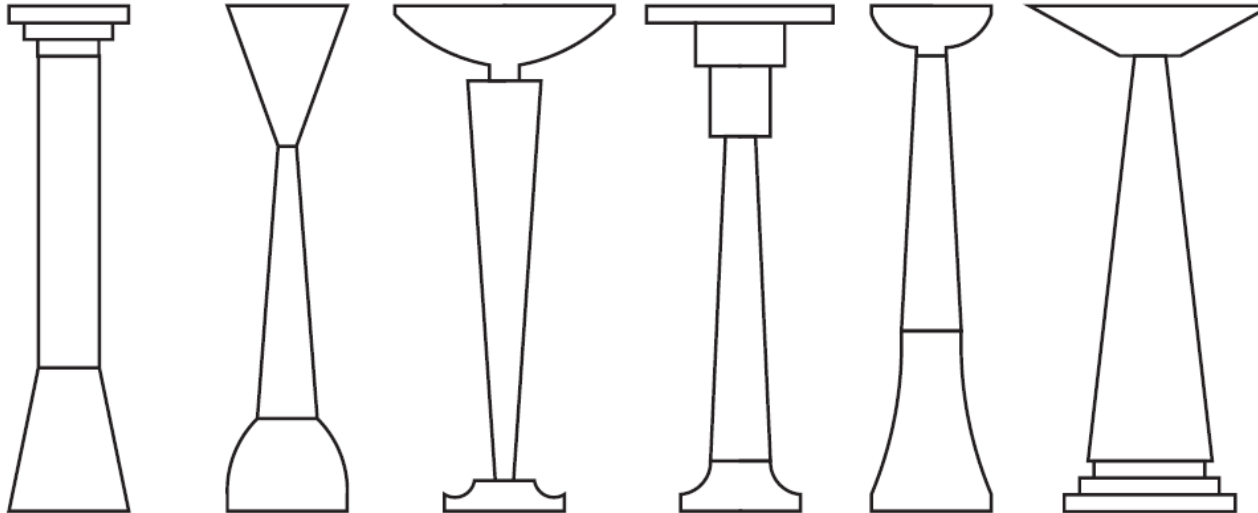
B



Traditional classes of parametric models (PH)

- ***Parametric Hybrid (PH)*** models are less used than Parametric Variations or Parametric Combinations, they offer the best of both and can be very robust for design exploration.
- However, they are difficult to construct, and in most cases it is better to construct and design with two models in parallel, one for variations and another for combinations.

Traditional classes of parametric models (PH)



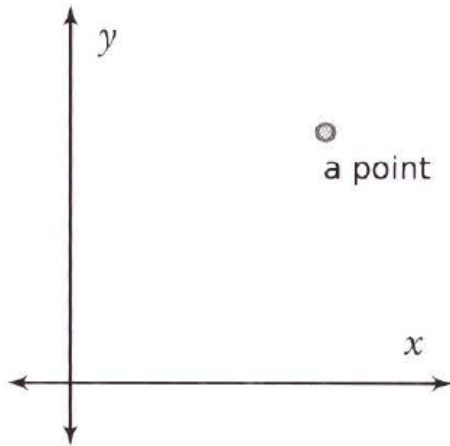
Outline

- Introduction to class
- Parametric design
 - Overview
 - Formal definition
 - Traditional classes of parametric models
- **Geometry notions (2D)**
 - Primitives
 - Geometric transformations
 - Curves

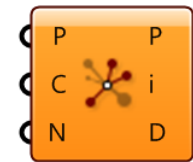
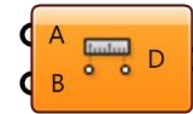
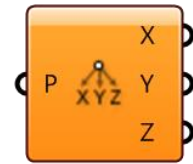
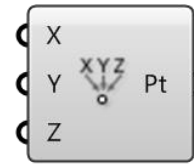
Outline

- Introduction to class
- Parametric design
 - Overview
 - Formal definition
 - Traditional classes of parametric models
- Geometry notions (2D)
 - **Primitives**
 - Geometric transformations
 - Curves

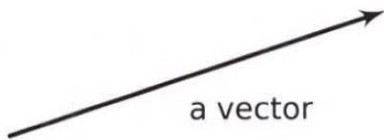
Points



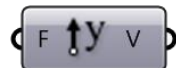
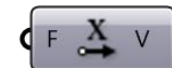
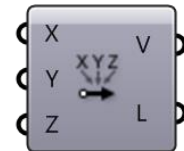
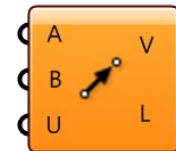
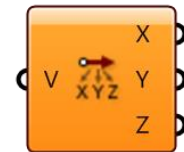
$$\dot{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



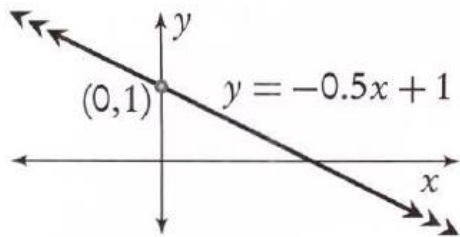
Vectors



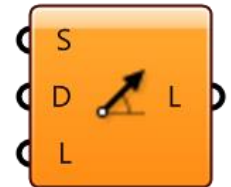
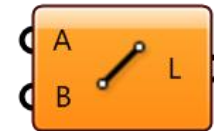
$$\vec{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



Lines



$$y = mx + b$$

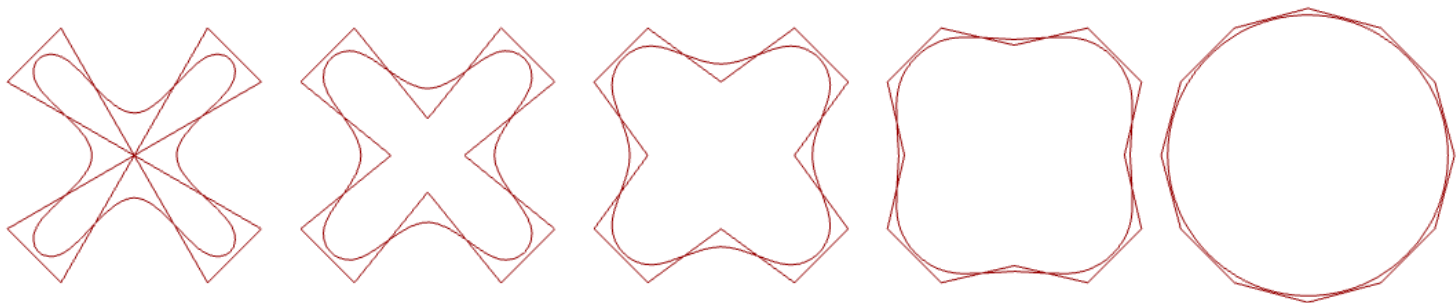
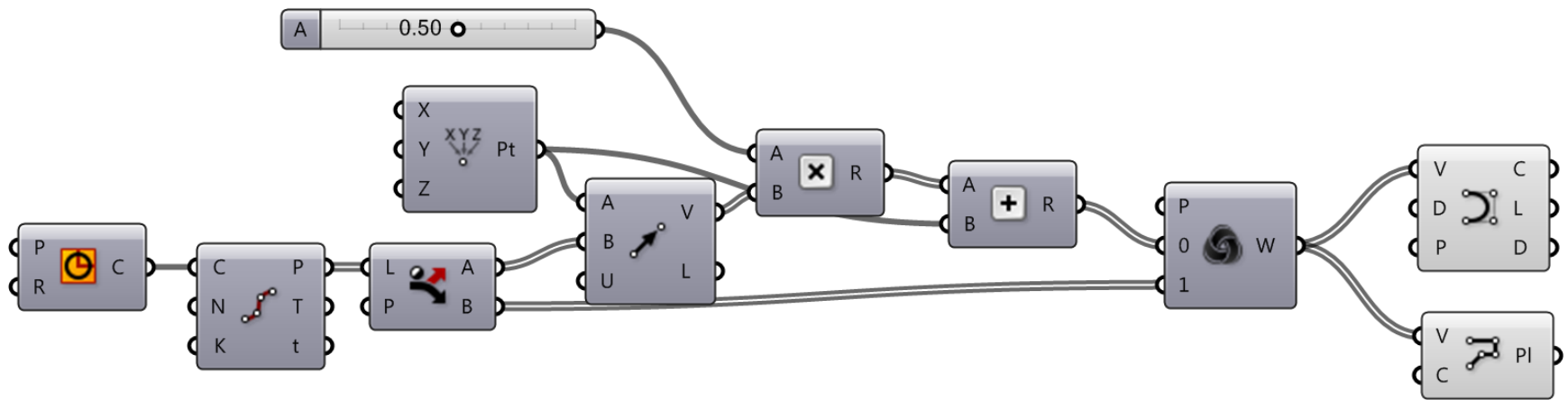


Lines (parametric equation)

A line can be uniquely defined by a point and a vector. Given a point p and a vector \vec{v} , any point $p(t)$ on the line has the functional equation:

$$p(t) = p + t \cdot \vec{v}$$

Lines (parametric equation - example)



$A = 0$

$A = 0.25$

$A = 0.5$

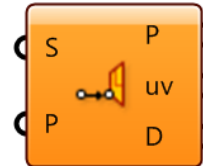
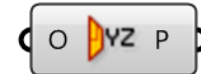
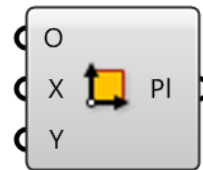
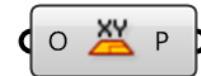
$A = 0.75$

$A = 1$

Planes

$$ax + by + cz + d = 0$$

$$\vec{n} \bullet (\vec{p} - \vec{q}) = 0$$



Outline

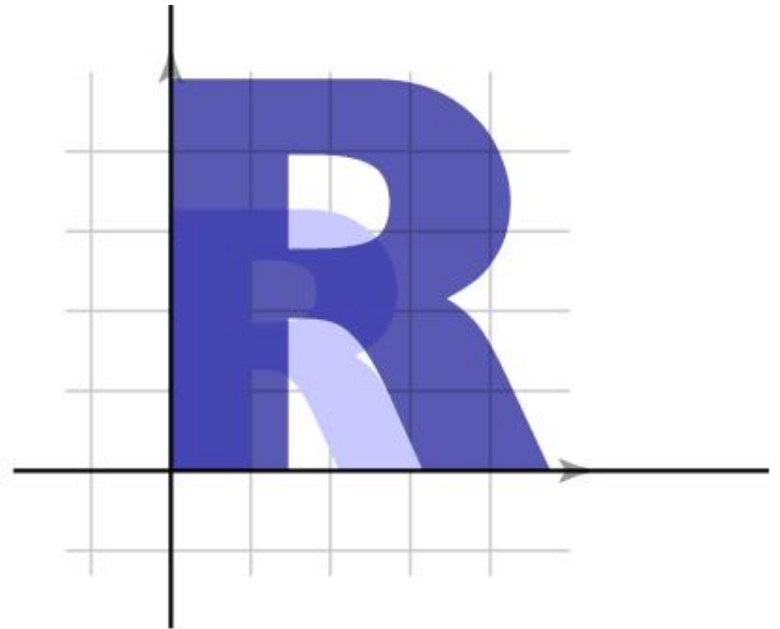
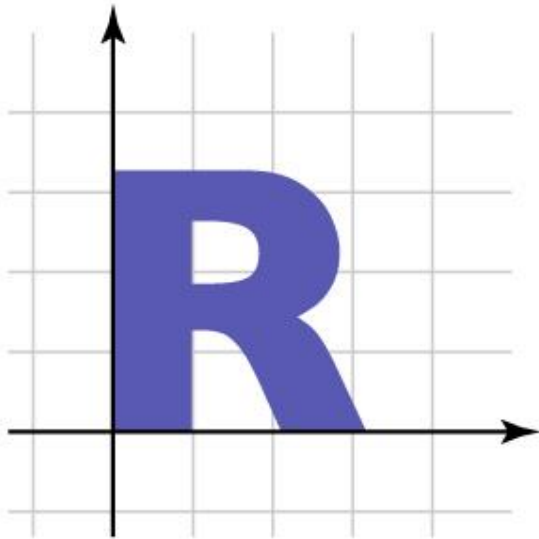
- Introduction to class
- Parametric design
 - Overview
 - Formal definition
 - Traditional classes of parametric models
- Geometry notions (2D)
 - Primitives
 - **Geometric transformations**
 - Curves

Matrices as transformations

- The geometric transformations we know such as scale, translate, rotate, shear can be represented as matrices.
- Performing several geometric transformations in a row is equivalent to matrix multiplication.
- Matrix multiplication is not commutative, therefore the order of the operations is important.

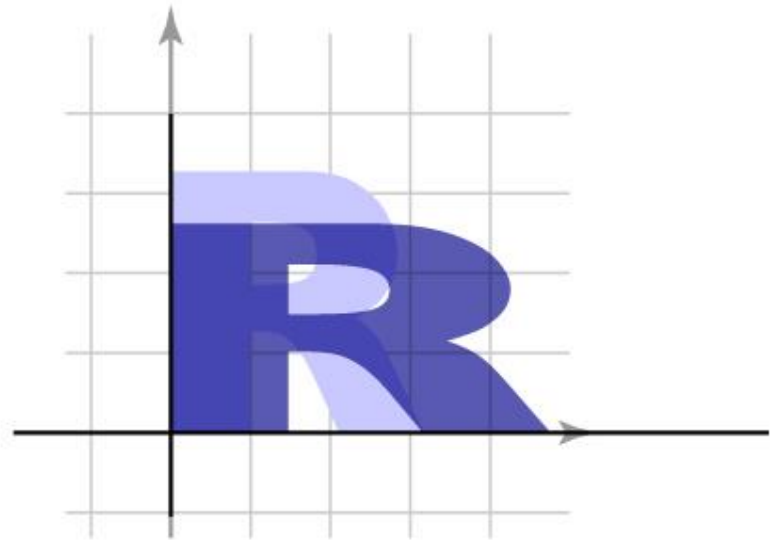
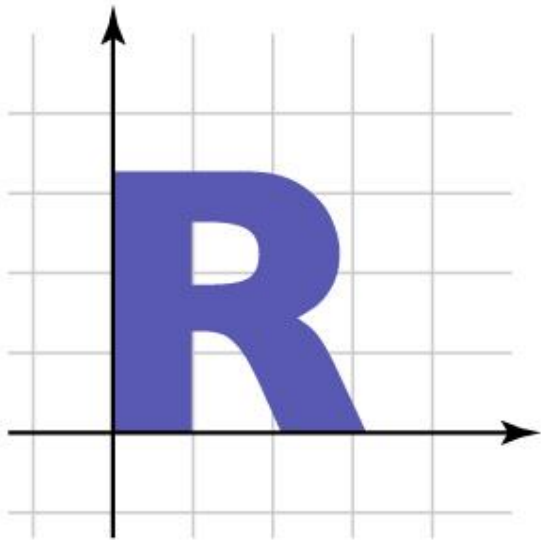
Uniform scale

$$\begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} sx \\ sy \end{bmatrix}$$



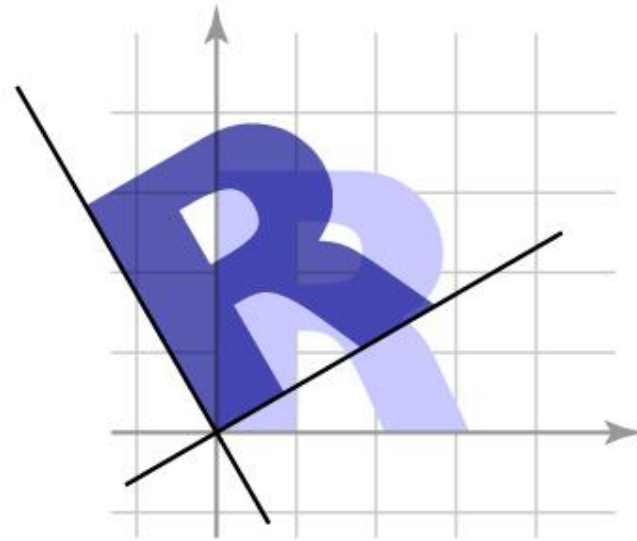
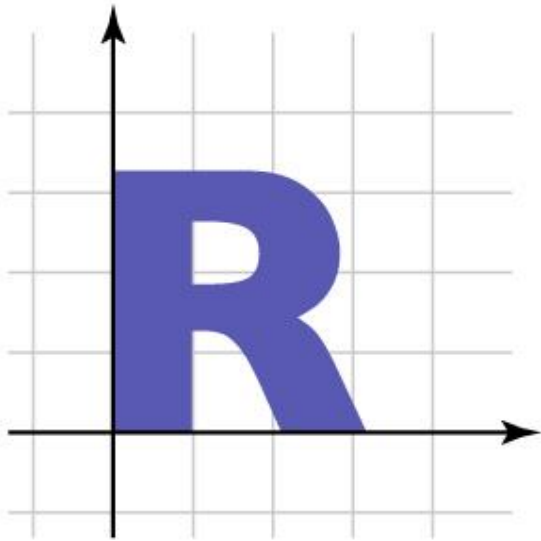
Non-uniform scale

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$



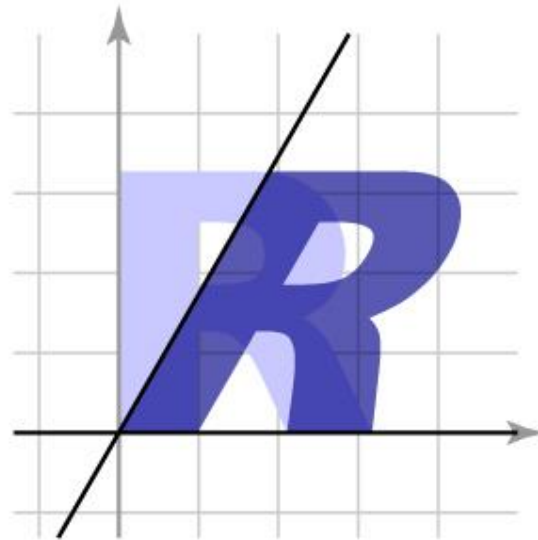
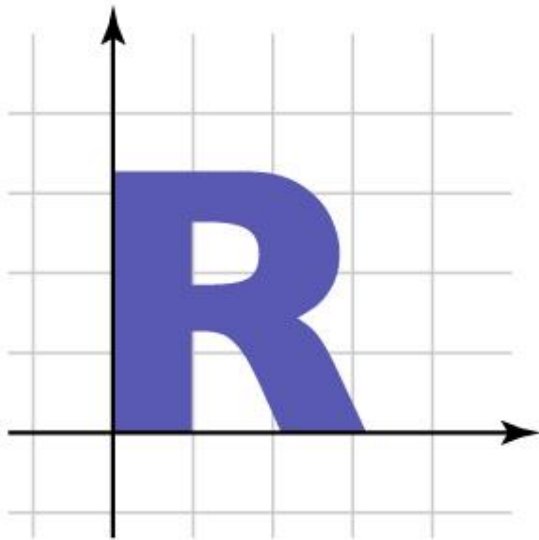
Rotation

$$\begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \end{bmatrix}$$



Shear

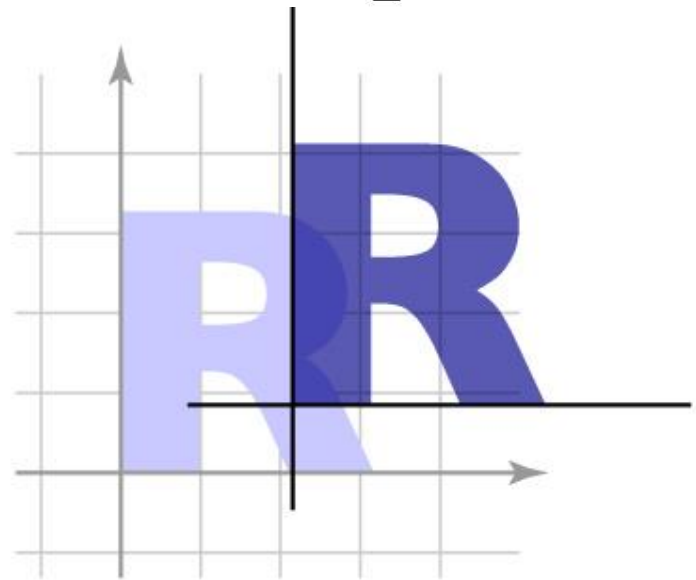
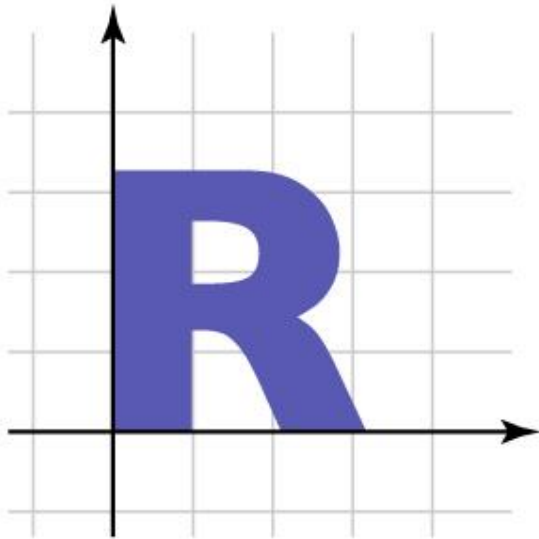
$$\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + ay \\ y \end{bmatrix}$$



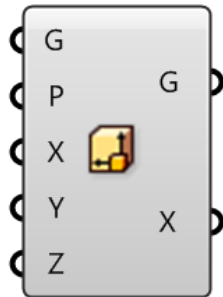
Translation

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 2.15 \\ 0 & 1 & 0.85 \\ 0 & 0 & 1 \end{bmatrix}$$



Geometric transformations

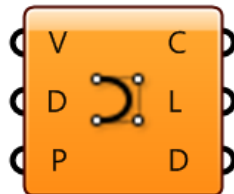
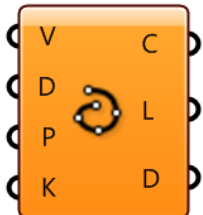


Outline

- Introduction to class
- Parametric design
 - Overview
 - Formal definition
 - Traditional classes of parametric models
- Geometry notions (2D)
 - Primitives
 - Geometric transformations
 - **Curves**

Interpolation vs. approximation

- Some curves algorithms ***interpolate***: they compute curves that do through the input points.
- Others ***approximate***: they place curves that are, in some sense, “near” the input points.
- We call the input points the ***control points***.



Bezier curves

- **A *Bezier curve*** of a degree n is defined by $n + 1$ control points p_0, p_1, \dots, p_n as follows:

$$B(t) = \sum_{i=0}^n \binom{n}{i} (1-t)^{n-i} t^i P_i =$$

for $t \in [0,1]$.

(Notice that: $B(0) = P_0$ and $B(1) = P_n$)

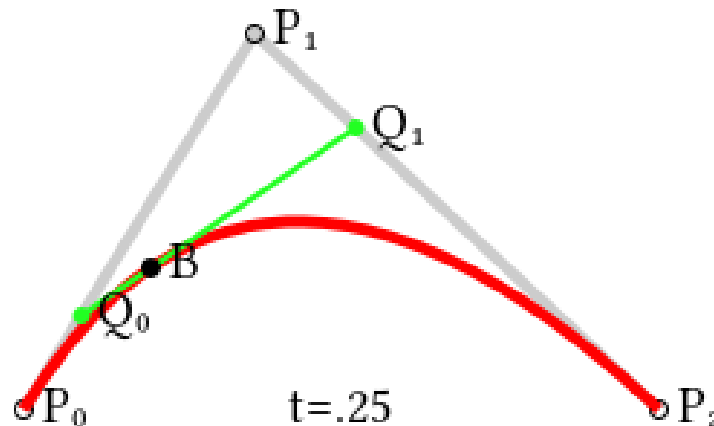
Bezier curves

- For 2 control points, the Bezier curve is just a line connecting the two points:

$$B(t) = (1 - t) \cdot P_0 + t \cdot P_1 \quad (t \in [0,1])$$

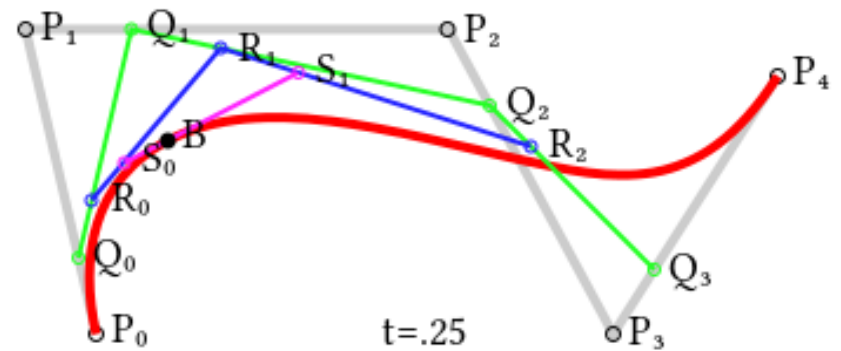
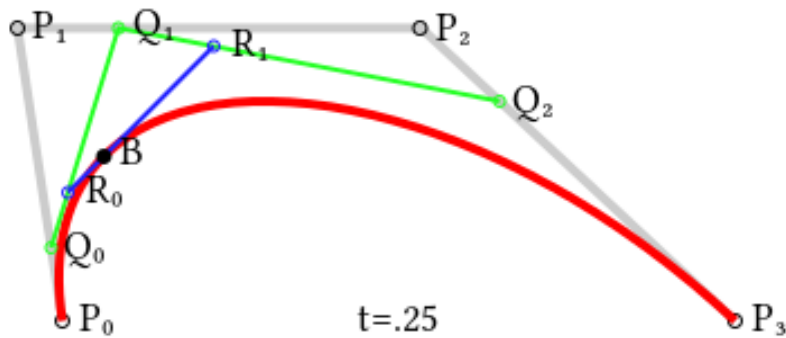
- For 3 control points the Bezier curve is:

$$B(t) = (1 - t)[(1 - t)P_0 + tP_1] + t[(1 - t)P_1 + tP_2] \quad (t \in [0,1])$$

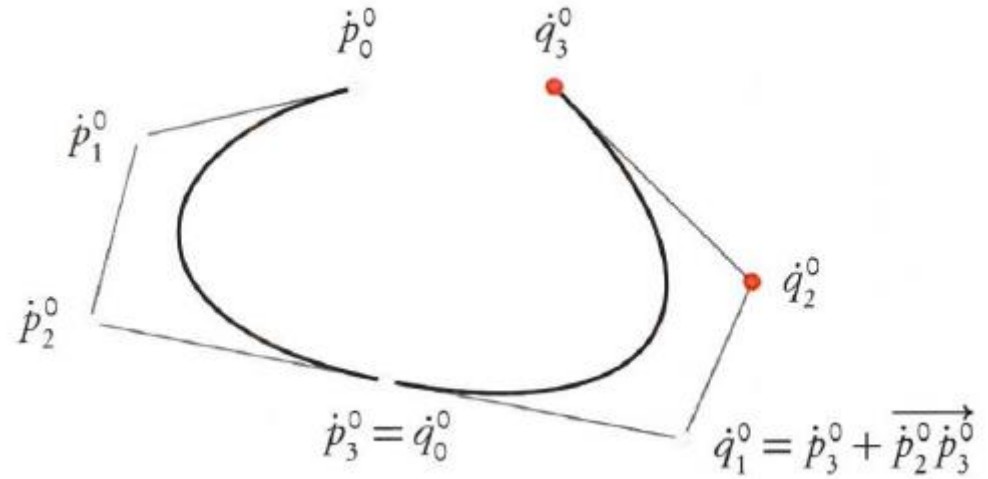
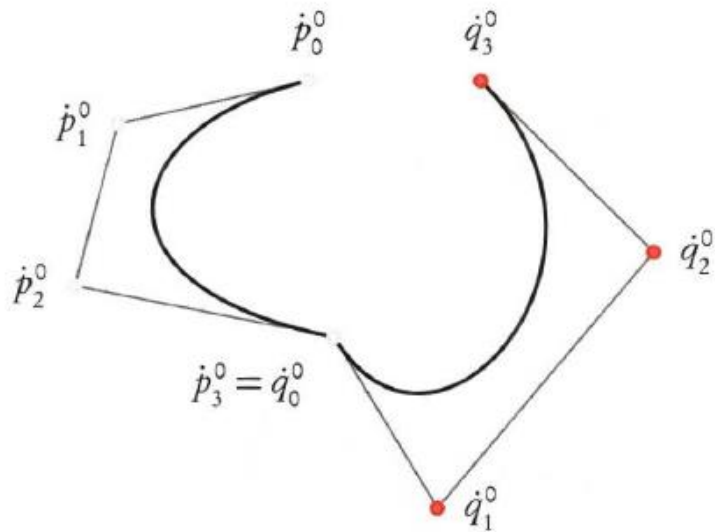


Bezier curves

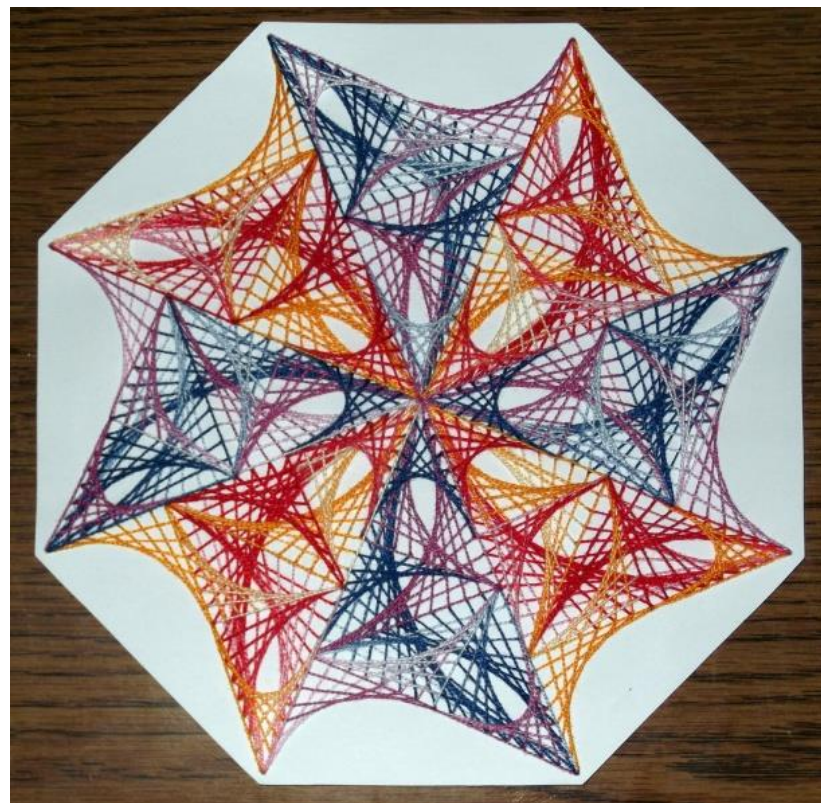
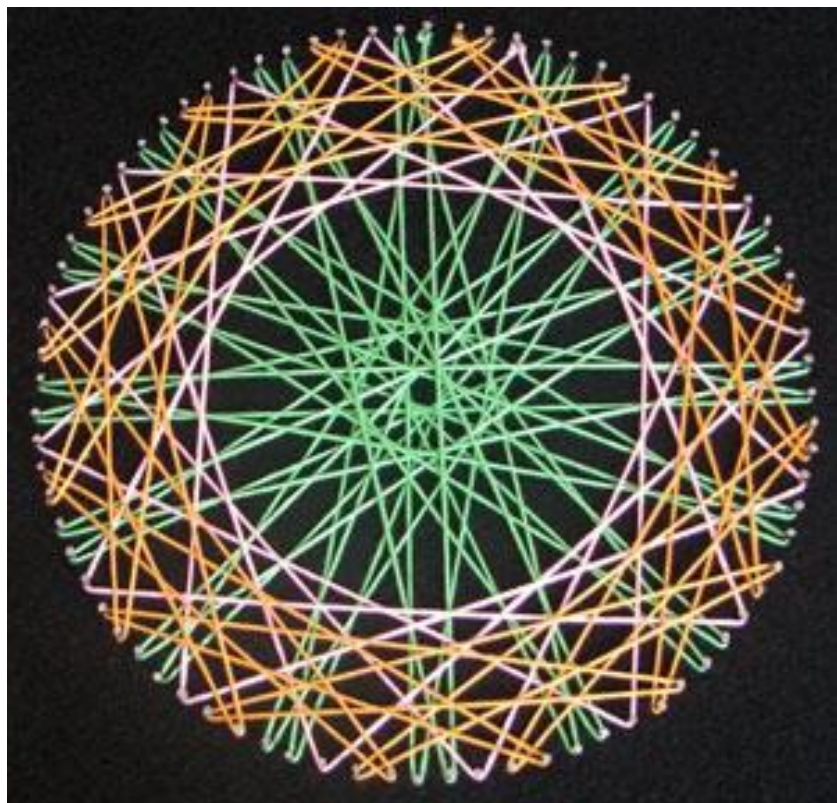
- The same is goes for 4, 5 control points and so on..



Continuity: when curves join



Bezier curves (example - string art)



B-Spline curves

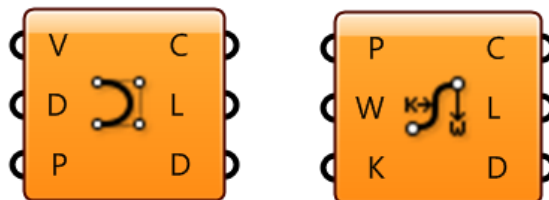
- Bezier curves are geometrically and mathematically simple, but they have deep problem: control is only pseudo-local, and the order of a Bezier curve is the number of points in the control polygon.
- ***Pseudo-local control*** means that all points in a curve change when any control point is changed.
- The link between order and control points means that a complex design must be done with high-order curves and this creates problems in interactive editing.
- The curve also may lie far from the control polygon, making it hard to predict how an editing action might affect the curve.

B-Spline curves

- B-Splines curves address all of these problems.
- Like Bezier curves, B-Splines are defined on a control polygon.
- The only constraint on the order of the curve is that it must be less than or equal to the number of points in the control polygon.

Non-uniform rational B-Spline curves (NURBS)

- Bezier curves and B-Splines are private cases of NURBS.
- NURBS curves have one more control over B-Splines: weights.
- NURBS curves are easier to work with, but less efficient to evaluate and the math behind them is hard to understand.
- NURBS curves are the power of Rhino!



Meshes vs. solids

- **A mesh** is triangulated polygons approximating the geometry (the more dense the triangles, the closer to the actual geometry).
- **A solid** is the actual mathematical expression of the geometry (NURBS and the like).
- Solids are parametric representations and are resolution independent. While a mesh has finite number of points and lines and will show as jagged when scaled up.
- We would prefer working with solids as long as we can. However, when we export a model for manufacture it ultimately ends up as a mesh.

Assignment #1

Portobello Mushroom (1st part) 3D Rhino model:

Take a Portobello Mushroom, photo it from three different directions, and 3D model it in Rhino. Don't use Grasshopper. Try to keep the mushroom geometric integrity as accurate as possible but don't get into texture or small details. Submit your Rhino model (3DM file) and photos.

Submission: by Monday 6.3.17 at 23:00 to Amit's mail (submission format: "Design class assignment #1" followed by your full name and id)