Polyhedral geometry 4

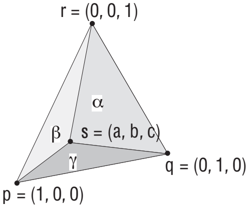
Computational Visual Design (CVD-Lab), DIA, “Roma Tre” University, Rome, Italy

Computational Graphics 2012

Convex coordinates and volume

The convex coordinates (α, β, γ) of a point $\mathbf{s} \in \mathrm{conv}\,(\mathbf{p}, \mathbf{q}, \mathbf{r}) \subset \mathbb{E}^d$ can be defined as the ratios between the areas of suitable triangles. In particular:

$$\alpha = \frac{\mathsf{area}(\mathbf{s}, \mathbf{q}, \mathbf{r})}{\mathsf{area}(\mathbf{p}, \mathbf{q}, \mathbf{r})}, \quad \beta = \frac{\mathsf{area}(\mathbf{p}, \mathbf{s}, \mathbf{r})}{\mathsf{area}(\mathbf{p}, \mathbf{q}, \mathbf{r})}, \quad \gamma = \frac{\mathsf{area}(\mathbf{p}, \mathbf{q}, \mathbf{s})}{\mathsf{area}(\mathbf{p}, \mathbf{q}, \mathbf{r})}.$$



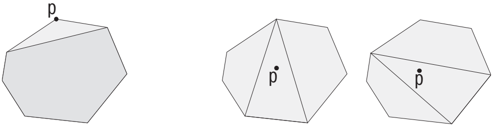
This property immediately generalizes to any convex set in any d-dimensional space, by using ratios of volumes of d-simplices.

examples of both affine or convex independence

1. two distinct points are independent in any $\mathbb{E}^d$, $d \geq 1$;
2. three non-colinear points are independent in any $\mathbb{E}^d$, $d \geq 2$;
3. four non-coplanar points are independent in any $\mathbb{E}^d$, $d \geq 3$;
4. $d+1$ points not belonging to the same affine subspace of codimension 1 (i.e.~dimension $d-1$) are independent in any $\mathbb{E}^D$, $D \geq d$;
5. m points are convexely independent if none of them is in the convex hull of the other $m-1$ points.

Convex polygon

The vertices of a convex polygon in $\mathbb{E}^2$ are convexly independent, as none of them can be generated as a convex combination of the others; idem for a convex polyhedron in $\mathbb{E}^d$



each vertex of a convex polygon is external to the convex hull of the others; convex coordinates of an internal point are not unique when the point is contained in more than one simplex

- points in $\mathrm{conv}\,\{\mathbf{p}_0, \ldots, \mathbf{p}_n\}$, with $\mathbf{p}_0, \ldots, \mathbf{p}_n \in \mathbb{E}^d$, have non unique convex coordinates when $d < n$

Parametric form of affine sets

The affine subspace generated by $d+1$ affinely independent points $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_d \in \mathbb{E}^n$, also called a d-flat, can be written in parametric form as an affine function of d independent real parameters:

$$S(\alpha_1, \ldots, \alpha_d) = \{\mathbf{p} \in \mathbb{E}^n | \ \mathbf{p} = \mathbf{p}_0 + \alpha_1(\mathbf{p}_1 - \mathbf{p}_0) + \cdots + \alpha_d(\mathbf{p}_d - \mathbf{p}_0)\}$$

Parametric segment

Consider the segment in $\mathbb{E}^d$ joining $\mathbf{p}_0$ and $\mathbf{p}_1$:

$$L(\alpha) = \{\mathbf{p} \in \mathbb{E}^d | \mathbf{p} = \mathbf{p}_0 + \alpha(\mathbf{p}_1 - \mathbf{p}_0)\}.$$

$L(\alpha)$ is a point-valued function of one real parameter $\alpha \in [0, 1] \subset \mathbb{R}$.

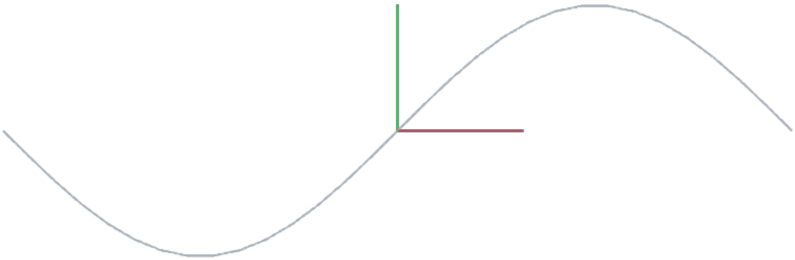
Parametric plane

A plane (or "2-flat") in $\mathbb{E}^n$ generated by points $\mathbf{q}, \mathbf{r}, \mathbf{s}$ is defined as:

$$P(\alpha, \beta) = \{\mathbf{p} \in \mathbb{E}^n | \ \mathbf{p} = (1 - \beta)((1 - \alpha)\mathbf{q} + \alpha\mathbf{r}) + \beta\mathbf{s}\}$$

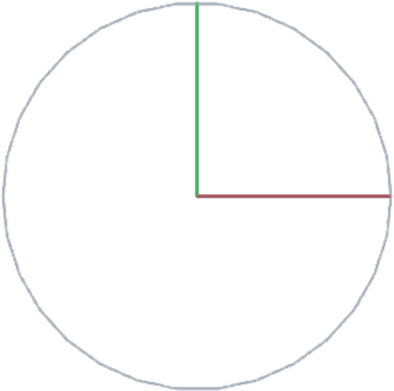
$P(\alpha, \beta)$ is a point-valued function of two real parameters $\alpha, \beta \in \mathbb{R}$.

Graph of a function



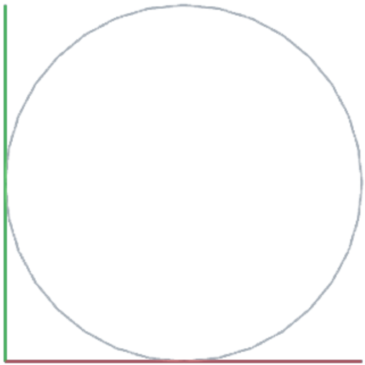
```
GRAPH = (f) -> ([a,b]) ->
    domain = T([0])([a]) INTERVALS(b-a)(30)
    MAP([ID, f]) domain

object = GRAPH(Math.sin)([-Math.PI, Math.PI])
model = viewer.draw object
```

Unit circle in 2D



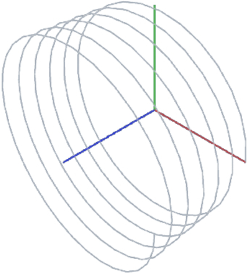
```
domain = INTERVALS(2*Math.PI)(30)
object = (MAP [Math.cos, Math.sin]) domain

model = viewer.draw object
```

Circle with radius r and center c_x, c_y



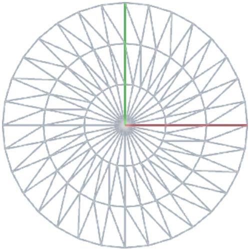
```
circle = (r) -> (c) -> (n) ->
    domain = INTERVALS(2*Math.PI)(n)
    x = (u) -> r * Math.cos u
    y = (u) -> r * Math.sin u
    object = T([0,1])(c) MAP([x, y])(domain)
```

Helix curve (r = 1, h = 1, 6 turns)



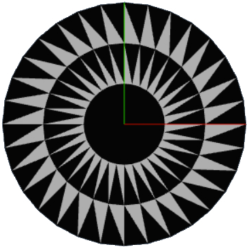
```
helix = (r=1, h=1, turns=6, n=180) ->
    domain = INTERVALS( turns*2*Math.PI )(n)
    x = (u) -> r * Math.cos u
    y = (u) -> r * Math.sin u
    z = (u) -> u * h/(turns*2*Math.PI)
    object = MAP([x, y, z])(domain)

model = viewer.draw helix()
```

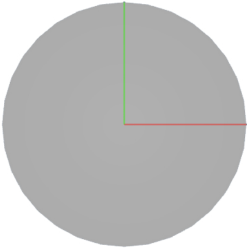
2D disk



```
DISK = (radius=1,n=32,m=1) ->
    domain = SIMPLEXGRID [REPEAT(n)(2*Math.PI/n), REPEAT(m
    fx = ([u,v]) -> v*Math.sin(u)
    fy = ([u,v]) -> v*Math.cos(u)
    MAP( [fx, fy] )(domain)

model = viewer.draw SKELETON(1) DISK(1,32,3)
```

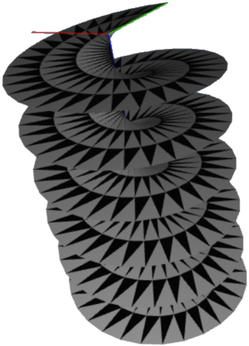
2D disk





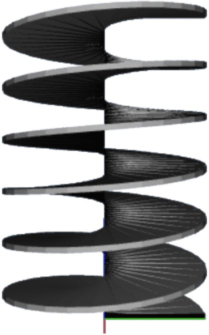
```
model = viewer.draw DISK(1,32,3)

model = viewer.draw R([0,2])(Math.PI) EMBED(1) DISK(1,32)
```

2D helicoid



```
helicoid = (radius=1, h=1, turns=6, n=180, m=1) ->
    domain = SIMPLEXGRID [REPEAT(n)(turns*2*Math.PI/n),
                            REPEAT(m)(radius/m)]
    fx = ([u,v]) -> v*Math.sin(u)
    fy = ([u,v]) -> v*Math.cos(u)
    fz = ([u,v]) -> u * h/(turns*2*Math.PI)
    object = MAP([fx, fy, fz])(domain)

model = viewer.draw helicoid(radius=1, h=3, turns=6, n=180, m=3)
```

3D solid helicoid



```
solidHelicoid = (width=0.1, radius=1, h=1, turns=6, n=180, m=1, p=1) ->
    domain = SIMPLEXGRID [REPEAT(n)(turns*2*Math.PI/n), REPEAT(m)(radiu
    fx = ([u,v,w]) -> v * Math.sin(-u)
    fy = ([u,v,w]) -> v * Math.cos(-u)
    fz = ([u,v,w]) -> width*w + (u * h/(turns*2*Math.PI))
    object = MAP([fx, fy, fz])(domain)

model = viewer.draw solidHelicoid(width=0.05, radius=1, h=3)
```

Section 1

Linear transformations

Linear transformations and tensors

The concept of tensor is very important for the PLaSM language

In such a language a geometric transformation (e.g. a rotation, a translation, a scaling) of an Euclidean space is just represented as a tensor.

Tensors are applied to geometric objects and assemblies as functions

Linear transformation

A linear transformation $\mathbf{T} : \mathcal{V}_1 \to \mathcal{V}_2$ is a function between vector spaces that preserves the linear combinations:

$$\mathbf{T}(\alpha_1 \mathbf{v}_1 + \cdots + \alpha_n \mathbf{v}_n) = \alpha_1 \mathbf{T} \mathbf{v}_1 + \cdots + \alpha_n \mathbf{T} \mathbf{v}_n.$$

Tensors

The term tensor is used as a synonym of invertible linear transformation $\mathbf{T}$ from a vector space $\mathcal{V}$ to itself.

In other words, a tensor is a linear function $\mathbf{T}$ which maps the vector $\mathbf{u}$ to the vector $\mathbf{Tu}$.

Tensors

The set of all tensors on $\mathcal{V}$ is a vector space, denoted as $\operatorname{lin}\mathcal{V}$, if addition of tensors and product of a tensor by a scalar are defined by:

$$
\begin{aligned}
(\mathbf{S}+\mathbf{T})\mathbf{v} &= \mathbf{S}\mathbf{v}+\mathbf{T}\mathbf{v}, \\
(\alpha\mathbf{S})\mathbf{v} &= \alpha(\mathbf{S}\mathbf{v}).
\end{aligned}
$$

The null tensor$\mathbf{0}$ and the identity tensor $\mathbf{I}$ map each vector $\mathbf{v}\in\mathcal{V}$ to the null vector and to itself, respectively:

$$
\begin{aligned}
\mathbf{0}\mathbf{v} &= \mathbf{0}, \\
\mathbf{I}\mathbf{v} &= \mathbf{v}.
\end{aligned}
$$

Tensor operations – Product

The $\{$*product*$\}$ *of tensors* $\mathbf{S}, \mathbf{T} \in \operatorname{lin}\mathcal{V}$ *is defined as composition of functions:*

$$\mathbf{ST} = \mathbf{S} \circ \mathbf{T}.$$

Hence $(\mathbf{ST})\mathbf{v} = \mathbf{S}(\mathbf{Tv})$ *for each* $\mathbf{v} \in \mathcal{V}$*.*

Tensor operations – Product

The $\{$*product*$\}$ *of tensors* $\mathbf{S}, \mathbf{T} \in \operatorname{lin} \mathcal{V}$ *is defined as composition of functions:*

$$\mathbf{S}\mathbf{T} = \mathbf{S} \circ \mathbf{T}.$$

Hence $(\mathbf{S}\mathbf{T})\mathbf{v} = \mathbf{S}(\mathbf{T}\mathbf{v})$ *for each* $\mathbf{v} \in \mathcal{V}$.

For the products of a tensor $\mathbf{S}$ *with itself we have*

$$\mathbf{S}^2 := \mathbf{S}\mathbf{S}, \qquad \mathbf{S}^3 := \mathbf{S}^2\mathbf{S}, \qquad \textit{etc.}$$

Tensor operations – tensor product of vectors

The tensor product of two vectors $\mathbf{a}, \mathbf{b} \in \mathcal{V}$ is the tensor $\mathbf{a} \otimes \mathbf{b} \in \operatorname{lin} \mathcal{V}$ that maps a vector $\mathbf{v}$ to the vector $(\mathbf{b} \cdot \mathbf{v})\mathbf{a}$. More formally:

$$\otimes : \mathcal{V}^2 \to \operatorname{lin} \mathcal{V} : (\mathbf{a}, \mathbf{b}) \mapsto \mathbf{a} \otimes \mathbf{b}$$

such that, for each $\mathbf{v} \in \mathcal{V}$

$$(\mathbf{a} \otimes \mathbf{b})\mathbf{v} = (\mathbf{b} \cdot \mathbf{v})\mathbf{a}.$$

Properties of tensor product of vectors

A few important properties of tensor product follow:

1. $(\mathbf{a} \otimes \mathbf{b})^T = (\mathbf{b} \otimes \mathbf{a})$;
2. $(\mathbf{a} \otimes \mathbf{b})(\mathbf{c} \otimes \mathbf{d}) = (\mathbf{b} \cdot \mathbf{c})\mathbf{a} \otimes \mathbf{d}$;
3. $(\mathbf{e}_i \otimes \mathbf{e}_i)(\mathbf{e}_j \otimes e_j) = \begin{matrix} \mathbf{0}, & i \neq j, \\ \mathbf{e}_i \otimes \mathbf{e}_i & i = j; \end{matrix}$
4. $\sum_i \mathbf{e}_i \otimes \mathbf{e}_i = \mathbf{I}$.

Example Directional decomposition of vectors

Let $\mathbf{e}, \mathbf{v} \in \mathcal{V}$, with $\mathbf{e}$ a unit vector. The tensor $\mathbf{e} \otimes \mathbf{e}$ applied to the vector $\mathbf{v}$ gives the projection of $\mathbf{v}$ onto the $\mathbf{e}$ direction:

$$(\mathbf{e} \otimes \mathbf{e})\mathbf{v} = (\mathbf{e} \cdot \mathbf{v})\mathbf{e}$$

Example Directional decomposition of vectors

Let $\mathbf{e}, \mathbf{v} \in \mathcal{V}$, with $\mathbf{e}$ a unit vector. The tensor $\mathbf{e} \otimes \mathbf{e}$ applied to the vector $\mathbf{v}$ gives the projection of $\mathbf{v}$ onto the $\mathbf{e}$ direction:

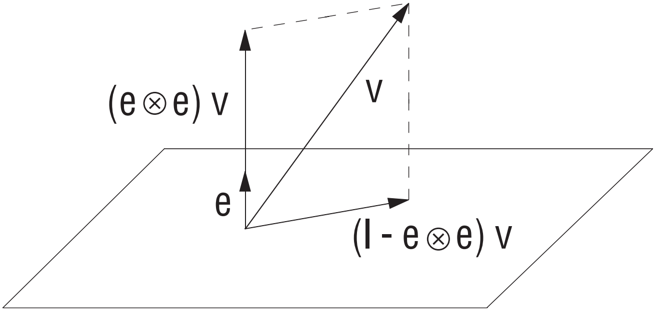
$$(\mathbf{e} \otimes \mathbf{e})\mathbf{v} = (\mathbf{e} \cdot \mathbf{v})\mathbf{e}$$

Conversely, the tensor $\mathbf{I} - \mathbf{e} \otimes \mathbf{e}$, when applied to $\mathbf{v}$, gives

$$(\mathbf{I} - \mathbf{e} \otimes \mathbf{e})\mathbf{v} = \mathbf{v} - (\mathbf{e} \cdot \mathbf{v})\mathbf{e},$$

which is the projection of $\mathbf{v}$ onto the linear subspace orthogonal to $\mathbf{e}$.

Example Directional decomposition of vectors



Directional and orthogonal projections of a vector

Section 2

Affine transformations

Affine transformations

affine transformation $\mathbf{T} : \mathbb{E}_1 \to \mathbb{E}_2$ is a function between affine spaces that preserves the affine action:

$$\mathbf{T}(\mathbf{x} + \alpha(\mathbf{y} - \mathbf{x})) = \mathbf{Tx} + \alpha(\mathbf{Ty} - \mathbf{Tx})$$

An affine transformation extends naturally to the underlying vector space, by defining

$$\mathbf{Tv} = \mathbf{Ty} - \mathbf{Tx}, \quad \text{where} \quad \mathbf{v} = \mathbf{y} - \mathbf{x}.$$