# CSIT 2013

# Rational Arithmetic with Floating Point


## Vaclav Skala

University of West Bohemia, Plzen, Czech Republic
VSB Technical University, Ostrava, Czech Republic
http://www.VaclavSkala.eu

# CSIT 2013

## Plzen (Pilsen) City

Plzen is an old city [first records of Plzen castle 976] city of culture, industry, and brewery.

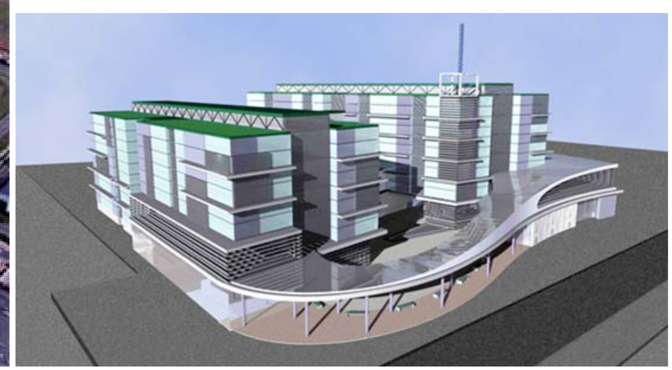
City, where today's beer fermentation process was invented
that is why today's beers are called Pilsner - world wide

# CSIT 2013

**University of West Bohemia 17530 students + 987 PhD students**

**Computer Science and Engineering** **Mathematics** (+ Geomatics)
**Physics** **Cybernetics** **Mechanics** (Computational)

- Over **50%** of income from research and application projects
- NTIS project (investment of 64 mil. EUR)
- 2nd in the ranking of Czech technical / informatics faculties 2009, 2012



# CSIT 2013

"Real science" in the XXI century

Courtesy of Czech Film, Barrandov

## Numerical systems

- Binary system is used nearly exclusively
- Octal & hexadecimal representation is used
- If we would be direct descendants of tetrapods – we would have a great advantage – “simple counting in hexadecimal system”

| | Name | Base | Digits | E min | E max |
|---|---|---|---|---|---|
| **BINARY** | | | | | |
| B 16 | Half | 2 | 10+1 | −14 | 15 |
| B 32 | Single | 2 | 23+1 | −126 | 127 |
| B 64 | Double | 2 | 52+1 | −1022 | 1023 |
| B 128 | Quad | 2 | 112+1 | −16382 | 16383 |
| **DECIMAL** | | | | | |
| D 32 | | 10 | 7 | −95 | 96 |
| D 64 | | 10 | 16 | −383 | 384 |
| D 128 | | 10 | 34 | −6143 | 6144 |

IEEE 758-2008 standard

Courtesy Clive "Max" Maxfield and Alvin Brown

The first tetrapods had eight fingers on each hand

# CSIT 2013

## Mathematically perfect algorithms fail due to instability

### Main issues

- stability, robustness of algorithms
- acceptable speed
- linear speedup – results depends on HW, CPU …. parameters !

### Numerical stability

- limited precision of float / double
- tests A ? B with floats

  **if** A = B **then** ….. **else** ….. ; **if** A = 0 **then** ….. **else** ….
  should be forbidden in programming languages

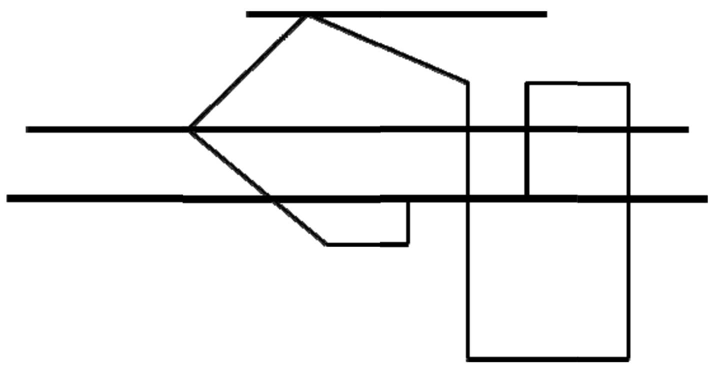
- division operation should be removed or postponed to the last moment if possible - “blue screens”, system resets

# CSIT 2013

## Typical examples of instability

- intersection of 2 lines in $E^3$
- point lies on a line in $E^2$ or a plane in $E^3$

  $Ax + By + C = 0$ or $Ax + By + Cz + D = 0$
- detection if a line intersects a polygon, touches a vertex or passes through

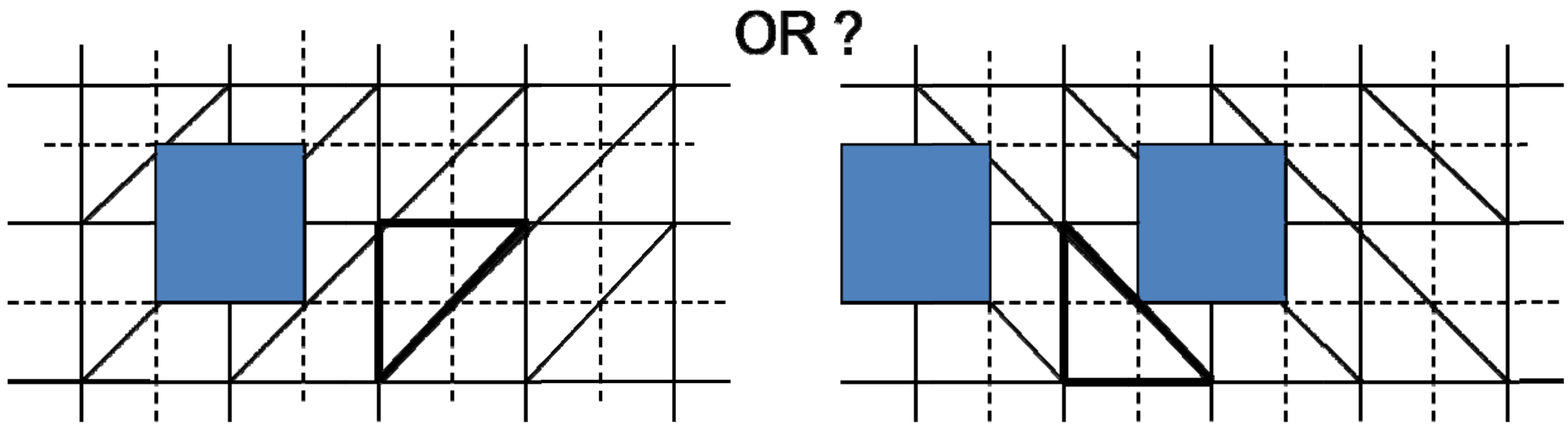


## Typical problem

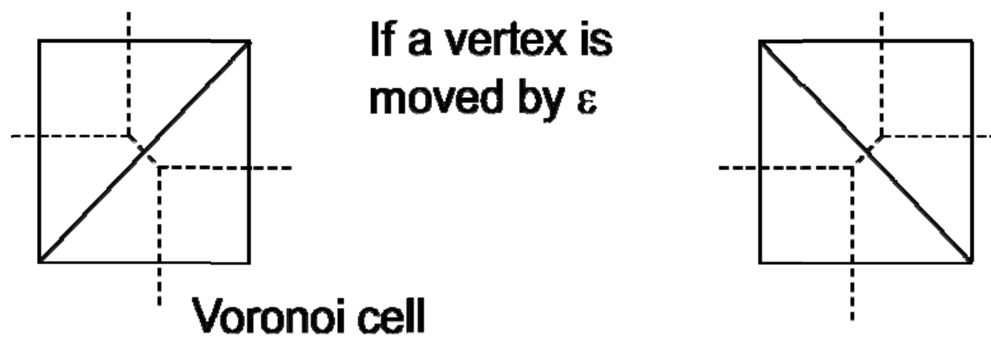
```
double x = -1; double p =  ….;
while ( x < +1)
{    if (x == p) Console.Out.WriteLine(" *** ")
     x += p;
}
/*     if p = 0.1 then no output,   if p = 0.25 then expected output */
```

## Delaunay triangulation & Voronoi diagram

Point inside of a circle given by three points – problems with meshing points in regular rectangular grid.




It can be seen that the DT & VD is very sensitive to a point position change




**?? ROBUSTNESS ??**

# CSIT 2013

## Floating point

- Not all numbers are represented correctly
- Logarithmic arithmetic
- Continuous fractions
- Interval arithmetic

$$\pi = \cfrac{4}{1 + \cfrac{1^2}{3 + \cfrac{2^2}{5 + \cfrac{3^2}{\cdots}}}}$$

$$\pi = [3; 7,15,1,292,1,1,1,2,1,3,1 \ldots]$$

Generally NOT valid identities due to limited precision

- $cos^2\alpha + cos^2\beta = 1$
  $[\ \alpha + \beta = \pi\ ]$
- $\mathrm{x}^2 - \mathrm{y}^2 = (x - y)(x + y)$

| | |
|---|---|
| x + y = [a + c, b + d] | x = [ a , b ] |
| x - y = [a - d, b - c] | y = [ c , d ] |
| x × y = [min(ac, ad, bc, bd), max(ac, ad, bc, bd)] | |
| x / y = [min(a/c, a/d, b/c, b/d), | |
| max(a/c, a/d, b/c, b/d)] if y ≠ 0 | |

# CSIT 2013

**Statements like**

**if** <float> = <float> **then ….** or **if** <float> ≠ <float> **then ….**

**should not be allowed in languages**

**Quadratic equation**

$at^2 + bt + c = 0$ usually solved as $t_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

If $b^2 \gg 4ac$ then

$$q = -(b + sign(b)\sqrt{b^2 - 4ac}\,)/2$$

$t_1 = {}^{q}/_{a}$ $t_2 = {}^{c}/_{q}$

**to get more reliable results.**

# CSIT 2013

**Function value computation** at $x = 77617,\ y = 33096$

$$f(x,y) = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

$f = 6.33835\ 10^{29}$ single precision

$f = 1{,}1726039400532$ double precision

$f = 1{,}172603940053178631858834904520183$ extended precision

The correct result is "somewhere" in the interval of

$$[-0{,}82739605994682136814116509547981629\mathbf{2005},\ -0{,}82739605994682136814116509547981629\mathbf{1986}]$$

Exact solution

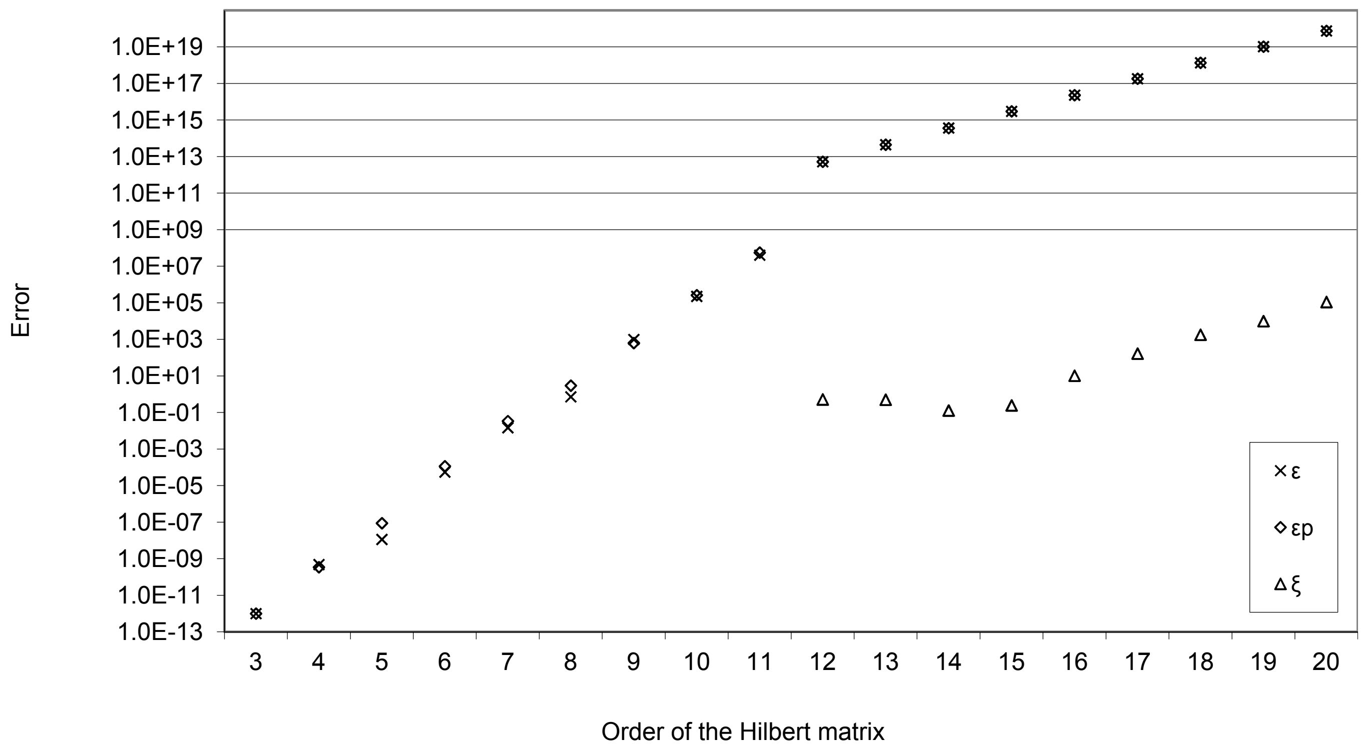
$$f(x,y) = -2 + \frac{x}{2y} = -\frac{54767}{66192}$$

# CSIT 2013

**Numerical computations** $\boldsymbol{Ax = b}$ $\boldsymbol{x = A^{-1}b}$

**Hilbert's Matrix** $H_{ij} = \frac{1}{i+j-1}$

$$H_{ij}^{-1} = (-1)^{i+j}(i+j-1)\binom{n+i-1}{n-j}\binom{n+j-1}{n-i}\binom{i+j-2}{i-1}^2$$

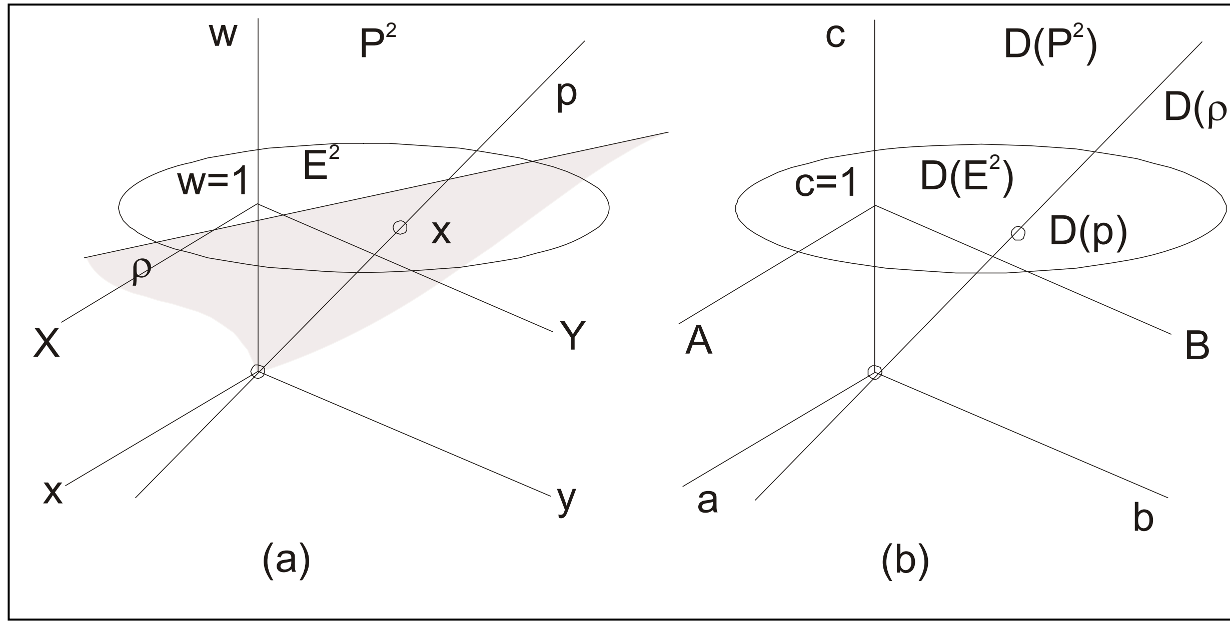


# CSIT 2013

## Projective Space

$\boldsymbol{X} = [X, Y]^T \quad \boldsymbol{X} \in E^2$

$\boldsymbol{x} = [x, y: w]^T \; \boldsymbol{x} \in P^2$




## Conversion:

$X = x / w \qquad Y = y / w$

$\& \; w \neq 0$

If $w = 0$ then $\boldsymbol{x}$ represents "an ideal point" - a point in infinity, i.e. it is a directional vector.

The Euclidean space $E^2$ is represented as a plane $w = 1$.

# CSIT 2013

## Points and vectors

- Vectors **are "freely movable"** – not having a fixed position

$$\boldsymbol{a}_1 = [x_1, y_1 : 0]^T$$

- Points are **not "freely movable"** – they are fixed to an origin of the current coordinate system

$$\boldsymbol{x}_1 = [x_1, y_1 : w_1]^T \quad \text{and} \quad \boldsymbol{x}_2 = [x_2, y_2 : w_2]^T$$

usually in textbooks $w_1 = w_2 = 1$

A vector $\boldsymbol{A} = \boldsymbol{X}_2 - \boldsymbol{X}_1$ in the Euclidean coordinate system - **CORRECT**

**Horrible "construction" DO NOT USE IT – IT IS TOTALLY WRONG**

$$\boldsymbol{a} = \boldsymbol{x}_2 - \boldsymbol{x}_1 = [x_2 - x_1, y_2 - y_1 : w_2 - w_1]^T = [x_2 - x_1, y_2 - y_1 : 1 - 1]^T$$
$$= [x_2 - x_1, y_2 - y_1 : 0]^T$$

This was presented as "How a vector" is constructed in the projective space $P^k$ in a textbook!! **WRONG, WRONG, WRONG**

*This construction has been found in SW as well*!!

## Points and vectors

A vector given by two points in the projective space

$$\boldsymbol{a} = \boldsymbol{x}_2 - \boldsymbol{x}_1 = [w_1 x_2 - w_2 x_1, w_1 y_2 - w_2 y_1 : w_1\, w_2]^T$$

This is the **CORRECT SOLUTION**, but what is the interpretation?

**A "difference" of coordinates of two points is a vector in the mathematical meaning and $w_1\, w_2$ is a "scaling" factor actually**

In the projective representation (if the vector length matters)

$$\boldsymbol{a} = \boldsymbol{x}_2 - \boldsymbol{x}_1 = [w_1 x_2 - w_2 x_1, w_1 y_2 - w_2 y_1 : w_1\, w_2]^T$$
$$\triangleq \left[\frac{w_1 x_2 - w_2 x_1}{w_1\, w_2}, \frac{w_1 y_2 - w_2 y_1}{w_1\, w_2} : 0\right]^T$$

We have to strictly distinguish if we are working with points, i.e. vector as a data structure represents the coordinates, or with a vector in the mathematical meaning stored in a vector data structure.

**VECTORS x FRAMES**

## Duality

For simplicity, let us consider a line $p$ defined as:

$aX + bY + c = 0$

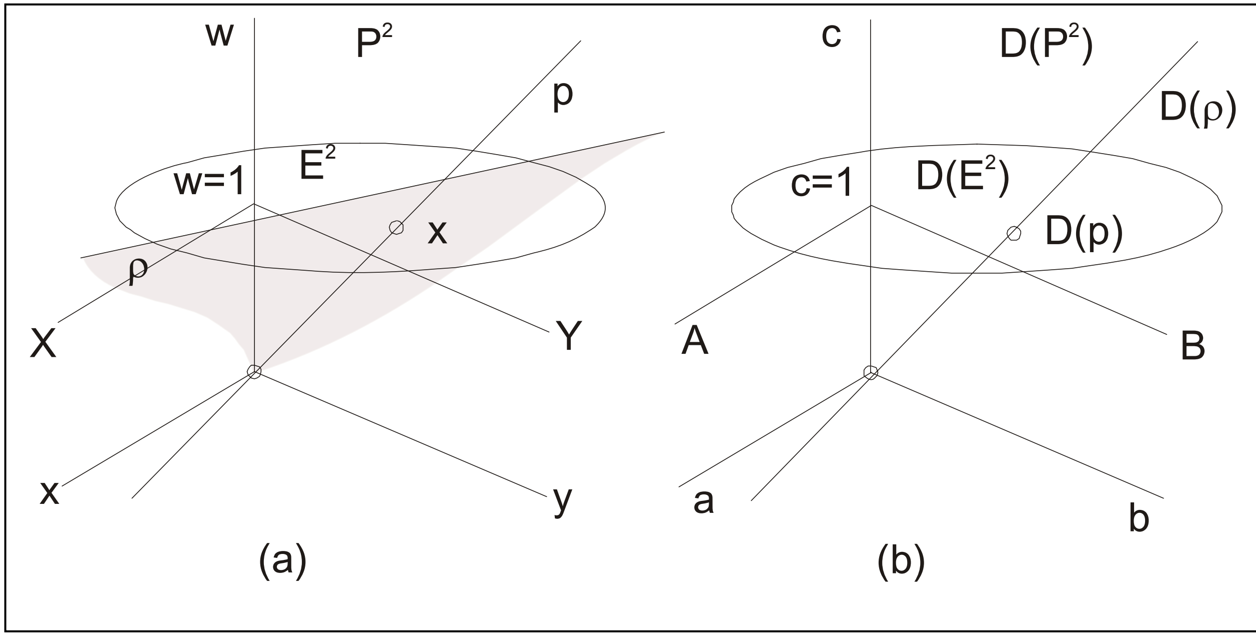
We can multiply it by $w \neq 0$ and we get:

$$ax + by + cw = 0$$

i.e. $\quad \boldsymbol{p}^T \boldsymbol{x} = 0$

$\boldsymbol{p} = [\, a, b: c]^{\mathrm{T}}$

$\boldsymbol{x} = [\, x, y: w]^{\mathrm{T}} = [\, wX, wY: w]^{\mathrm{T}}$




A line $p \in E^2$ is actually a plane in the projective space $P^2$ (point $[0,0:0]^{\mathrm{T}}$ excluded)

## Duality

From the mathematical notation $\boldsymbol{p}^{\boldsymbol{T}}\boldsymbol{x} = 0$

we cannot distinguish whether ***p*** is a line and ***x*** is a point or vice versa in the case of $P^2$. It means that

- a *point* and a *line* **are dual** in the case of $P^2$, and
- a *point* and a *plane* **are dual** in the case of $P^3$.

The principle of duality in $P^2$ states that:

Any theorem remains true when we interchange the words “point” and “line”, “lie on” and “pass through”, “join” and “intersection”, “collinear” and “concurrent” and so on.

**Once the theorem has been established, the dual theorem is obtained as described above.**

This helps a lot to solve some geometrical problems.

## Examples of dual objects and operators

| | Primitive | Dual primitive |
|---|---|---|
| $E^2$ | Point<br>Line | Line<br>Point |
| $E^3$ | Point<br>Plane | Plane<br>Point |

| Operator | Dual operator |
|---|---|
| Join<br>Intersect | Intersect<br>Join |

***Computational sequence for a problem is the same for a dual problem.***

# CSIT 2013

## Intersection of two lines

Let two lines $p_1$ and $p_2$ are given by

$$p_1 = [a_1, b_1 : c_1]^T \qquad \text{and} \qquad p_2 = [a_2, b_2 : \ c_2]^T$$

We have to solve a system of linear equations $\boldsymbol{Ax = b}$

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \qquad \text{and} \qquad \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} -c_1 \\ -c_2 \end{bmatrix} *$$

Then well known formula is used

$$X = \frac{Det_X}{Det} = \frac{det \begin{bmatrix} q_1 & b_1 \\ q_2 & b_2 \end{bmatrix}}{det \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix}} \qquad Y = \frac{Det_Y}{Det} = \frac{det \begin{bmatrix} a_1 & q_1 \\ a_2 & q_2 \end{bmatrix}}{det \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix}}$$

But what if $Det$ is small?

Usually a sequence like $\boldsymbol{if}\ abs(\det(..)) \leq eps\ \boldsymbol{then}$ is used. What is $eps$?

**Note** * usually a line is in its explicit form as $ax + by = q$ instead of $+by + c = 0$ , i.e. the implicit form

# CSIT 2013

**How a line is given by two $X_1$ and $X_2$ points?**

We have to solve a homogeneous system of linear equations

$$\begin{bmatrix} X_1 & Y_1 & 1 \\ X_2 & Y_2 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

i.e. $\boldsymbol{Ax} = \boldsymbol{0}$ . It means that there is ***one parametric set*** of solutions!

**NOW**

Computation of

- an intersection of two lines is given as $\boldsymbol{Ax} = \boldsymbol{b}$
- a line given by two points is given as $\boldsymbol{Ax} = \boldsymbol{0}$

**Different schemes for computation BUT**

**Those problems are DUAL, why algorithms are different??**

## Definition

The cross product of the two vectors

$\mathbf{x}_1 = [x_1, y_1 : w_1]^T$ and $\mathbf{x}_2 = [x_2, y_2 : w_2]^T$

is defined as:

$$\boldsymbol{x}_1 \times \boldsymbol{x}_2 = det \begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_1 & y_1 & w_1 \\ x_2 & y_2 & w_2 \end{bmatrix}$$

where: $\mathbf{i} = [1,0:0]^T$, $\mathbf{j} = [0,1:0]^T$, $\mathbf{k} = [0,0:1]^T$

Please, note that homogeneous coordinates are used.

# CSIT 2013

**Theorem**

Let two points $\mathbf{x}_1$ and $\mathbf{x}_2$ be given in the projective space. Then the coefficients of the $\boldsymbol{p}$ line, which is defined by those two points, are determined as the cross product of their homogeneous coordinates

$$\boldsymbol{p} = \boldsymbol{x}_1 \times \boldsymbol{x}_2 = [a, b : c]^T$$

**Proof**

Let the line $\boldsymbol{p} \in E^2$ be defined in homogeneous coordinates as (coefficient $d$ is used intentionally to have the same symbol representing a “distance” of the element from the origin for lines and planes)

$$ax + by + cw = 0$$

We are actually looking for a solution to the following equations:

$$\boldsymbol{p}^T \boldsymbol{x}_1 = 0 \qquad \boldsymbol{p}^T \boldsymbol{x}_2 = 0$$

where: $\mathbf{p} = [a, b : c]^T$

# CSIT 2013

It means that any point $\boldsymbol{x}$ that lies on the $p$ line must satisfy both the equation, i.e. $\boldsymbol{p}^T\boldsymbol{x}_1 = 0$ $\boldsymbol{p}^T\boldsymbol{x}_2 = 0$ and the equation $\boldsymbol{p}^{\boldsymbol{T}}\boldsymbol{x} = 0$ in other words the $\boldsymbol{p}$ vector is defined as

$$\boldsymbol{p} = \boldsymbol{x}_1 \times \boldsymbol{x}_2 = det\begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_1 & y_1 & w_1 \\ x_2 & y_2 & w_2 \end{bmatrix}$$

We can write

$$(\boldsymbol{x}_1 \times \boldsymbol{x}_2)^{\boldsymbol{T}}\boldsymbol{x} = 0 \qquad \text{i.e.} \quad det\begin{bmatrix} x & y & w \\ x_1 & y_1 & w_1 \\ x_2 & y_2 & w_2 \end{bmatrix} = 0$$

Note that **cross** product and **dot** product are instructions in Cg/HLSL on GPU

# CSIT 2013

Evaluating the determinant $det\begin{bmatrix} a & b & c \\ x_1 & y_1 & w_1 \\ x_2 & y_2 & w_2 \end{bmatrix} = 0$

we get the line coefficients of the line *p* as:

$$a = det\begin{bmatrix} y_1 & w_1 \\ y_2 & w_2 \end{bmatrix} \qquad b = -det\begin{bmatrix} x_1 & w_1 \\ x_2 & w_2 \end{bmatrix} \qquad c = det\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix}$$

**Note**:

1. A line $ax + by + c = 0$ is a one parametric set of coefficients

   $$\boldsymbol{p} = [a, b : c]^T$$

   From two values $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ we have to compute 3 values, coefficients $a$ , $b$ and $c$

2. For w = 1 we get the standard cross product formula and the cross product defines the *p* line, i.e. $\boldsymbol{p} = \boldsymbol{x}_1 \times \boldsymbol{x}_2$ where:

   $$\boldsymbol{p} = [a, b : c]^T$$

# CSIT 2013

We have seen, in the Euclidean space, the computation of

- an intersection of two lines is given as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$
- a line given by two points is given as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$

**If projective representation is used it is actually an application of the cross product.**

**Those problems are DUAL and algorithms are identical ??**

**Cross product is equivalent to a solution of a linear system of equations!**

**No division operations!**

# CSIT 2013

## DUALITY APPLICATION

In the projective space $P^2$ points and lines are dual. Due to duality we can directly intersection of two lines as

$$\boldsymbol{x} = \boldsymbol{p}_1 \times \boldsymbol{p}_2 = det\begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{bmatrix} = [x, y: w]^T$$

If the lines are parallel or close to parallel, the homogeneous coordinate $w \to 0$ and users have to take a decision – so there is no sequence in the code like $\boldsymbol{if}\ abs(\det(..)) \leq eps\ \boldsymbol{then}$ …in the procedure.

Generally computation can continue even if $w \to 0$ if projective space is used.

**Computation in Projective Space -** Barycentric coordinates

Let us consider a triangle with vertices $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$,

A position of any point $\mathbf{X} \in E^2$ can be expressed as

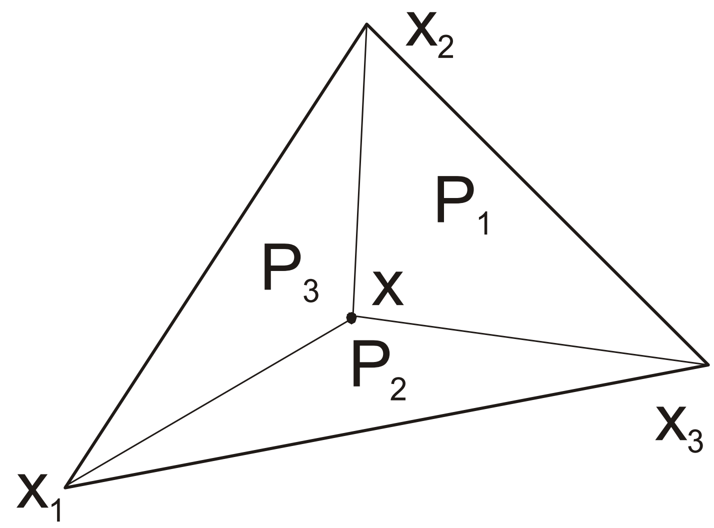
$$a_1 X_1 + a_2 X_2 + a_3 X_3 = X$$

$$a_1 Y_1 + a_2 Y_2 + a_3 Y_3 = Y$$

additional condition

$$a_1 + a_2 + a_3 = 1 \quad 0 \le a_i \le 1$$

$$a_i = \frac{P_i}{P} \quad \mathrm{i} = 1,\dots,3$$




A linear system of equations $\boldsymbol{Ax} = \boldsymbol{b}$ has to be solved

If points $\mathbf{x}_i$ are given as $[x_i, y_i, z_i: w_i]^T$ and $w_i \neq 1$ then $\mathbf{x}_i$ must be "normalized" to $w_i = 1$, i.e. 4 * 3 = 12 division operations

# CSIT 2013

## Computation in Projective Space

$$b_1 X_1 + b_2 X_2 + b_3 X_3 + b_4 X = 0$$

$$b_1 Y_1 + b_2 Y_2 + b_3 Y_3 + b_4 Y = 0$$

$$b_1 + b_2 + b_3 + b_4 = 0$$

$$b_i = -a_i b_4 \quad i = 1, \ldots, 3 \qquad b_4 \neq 0$$

Rewriting

$$\begin{bmatrix} X_1 & X_2 & X_3 & X \\ Y_1 & Y_2 & Y_3 & Y \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \mathbf{0}$$

$$\mathbf{b} = \boldsymbol{\xi} \times \boldsymbol{\eta} \times \mathbf{w}$$

$$\mathbf{b} = \left[ b_1, b_2, b_3, b_4 \right]^T$$

$$\boldsymbol{\xi} = \left[ X_1, X_2, X_3, X \right]^T$$

$$\boldsymbol{\eta} = \left[ Y_1, Y_2, Y_3, Y \right]^T$$

$$\mathbf{w} = \left[ 1,1,1,1 \right]^T$$

Solution of the linear system of equations (LSE) is equivalent to generalized cross product

$$\boldsymbol{b} = \boldsymbol{\xi} \times \boldsymbol{\eta} \times \boldsymbol{w}$$

# CSIT 2013

## Computation in Projective Space

if $w_i \neq 1$ or $w_i = 1$

$$\begin{bmatrix} x_1 & x_2 & x_3 & x \\ y_1 & y_2 & y_3 & y \\ w_1 & w_2 & w_3 & w \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \mathbf{0}$$

$$\mathbf{b} = \boldsymbol{\xi} \times \boldsymbol{\eta} \times \mathbf{w}$$

$$\mathbf{b} = \left[ b_1, b_2, b_3, b_4 \right]^T$$

$$\boldsymbol{\xi} = \left[ x_1, x_2, x_3, x \right]^T$$

$$\boldsymbol{\eta} = \left[ y_1, y_2, y_3, y \right]^T$$

$$\mathbf{w} = \left[ w_1, w_2, w_3, w \right]^T$$

$$0 \leq (-b_1 : w_2 w_3 w) \leq 1$$

$$0 \leq (-b_2 : w_3 w_1 w) \leq 1$$

$$0 \leq (-b_3 : w_1 w_2 w) \leq 1$$

=> **new entities**:

projective scalar, projective vector

- Skala,V.: Barycentric coordinates computation in homogeneous coordinates, Computers&Graphics, 2008)

## Computation in Projective Space

### Line in E3 as Two Plane Intersection

Standard formula in the Euclidean space

$$\boldsymbol{\rho}_1 = [a_1, b_1, c_1 : d_1]^T = [\boldsymbol{n}_1^T : d_1]^T \qquad \boldsymbol{\rho}_2 = [a_2, b_2, c_2 : d_2]^T = [\boldsymbol{n}_2^T : d_2]^T$$

Line given as an intersection of two planes

$$\boldsymbol{s} = \boldsymbol{n}_1 \times \boldsymbol{n}_2 \equiv [a_3, b_3, c_3 : 0]^T \qquad \boldsymbol{X}(\boldsymbol{t}) = \boldsymbol{X}_0 + \boldsymbol{s}t$$

$$X_0 = \frac{d_2 \begin{vmatrix} b_1 & c_1 \\ b_3 & c_3 \end{vmatrix} - d_1 \begin{vmatrix} b_2 & c_2 \\ b_3 & c_3 \end{vmatrix}}{DET} \qquad Y_0 = \frac{d_2 \begin{vmatrix} a_3 & c_3 \\ a_1 & c_1 \end{vmatrix} - d_1 \begin{vmatrix} a_3 & c_3 \\ a_2 & c_2 \end{vmatrix}}{DET}$$

$$Z_0 = \frac{d_2 \begin{vmatrix} a_1 & b_1 \\ a_3 & b_3 \end{vmatrix} - d_1 \begin{vmatrix} a_2 & b_2 \\ a_3 & b_3 \end{vmatrix}}{DET} \qquad DET = \begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix}$$

**The formula is quite “horrible” one and for students not acceptable as it is too complex and they do not see from the formula comes from.**

## Basic operations using projective

Robustness of computation is a key issue in many sophisticated computational algorithms as sophisticated engineering problems solved today might be ill conditioned. However different data structures have to be considered, i.e.

- projective scalar $[a_0 : a_1]$
- projective vector $[a_0 : a_1, \dots, a_n]$

## Fundamental operations with:

*A. scalars*

- addition, resp. subtraction $[a_0 : a_1] \pm [b_0 : b_1] = [a_0 b_0 : b_0 a_1 \pm a_0 b_1]$
- multiplication $[a_0 : a_1] * [b_0 : b_1] = [a_0 b_0 : a_1 b_1]$
- division $[a_0 : a_1] / [b_0 : b_1] = [a_0 b_1 : a_1 b_0]$

*B. vectors*

- addition resp. subtraction

$$[a_0 : a_1, \dots, a_n] \pm [b_0 : b_1, \dots, b_n] = [a_0 b_0 : \{b_0(a_1, \dots, a_n) \pm a_0(b_1, \dots, b_n)\}]$$

- scalar multiplication (dot product)

$$[a_0 : (a_1, \dots, a_n)] \cdot [b_0 : (b_1, \dots, b_n)] = [a_0 b_0 : \sum_{i=1}^{n} a_i b_i]$$

- vector multiplication (cross product)

$$[a_0 : (a_1, \dots, a_3)] \times [b_0 : (b_1, \dots, b_3)] = [a_0 b_0 : \{(a_1, \dots, a_3) \times (b_1, \dots, b_3)\}]$$

Note that the projective vector is different from a vector which consists of projective scalars.

*C. exponent normalization*

Exponents due to arithmetic operations tend to grow or become smaller. It means that the exponent overflow or underflow is to be check and exponents can be normalized. This is very simple operation as it means that the same value in the exponent is to be added or subtracted from both – numerator and denominator as well.

Extraction of an exponent for a single and double precision is defined

EXP := (FP_value **land** MASK) **shr** m;

where: **land** is bitwise and operation, **shr** is shift right, MASK is the binary mask and *m* is the argument for the shift operation.

| Precision | MASK | m | Exp_Digits |
|---|---|---|---|
| Single | &7FC0 | 4 | 255 |
| Double | &7F80 | 7 | 2047 |

*D. Comparison operation*

The comparison operation is a little bit tricky as the condition

$$a < b$$

i.e.

$$[a_0 : a_1] < [b_0 : b_1]$$

Projective scalars have to have homogeneous coordinate non-negative, i.e. $a_0 > 0$ and $b_0 > 0$

The condition is to be replaced as follows:

$$[b_0 a_1] < [a_0 b_1]$$

# CSIT 2013

## Advantages

- The mantissa is actually doubled due to the "hidden" division operation by the homogeneous value a there is a higher range of the fractional part
- The exponent range is higher. If a single precision is used, the range is $2^{-254}$ to $2^{254}$ , i.e.the range is actually $2^{508}$
- The division operation is eliminated by multiplication of a homogeneous value in which denominator is "hidden"
- Infinity can be handled properly, i.e. division by a value close or equal to zero does not cause "floating point overflow"
- If double precision for numerator and denominator is used is used, actually a quadruple extended precision is implemented; if quadruple precision is available we get more that 2 times better precision
- Simple implementation on vector-vector architectures, like GPU – available projective Library P-Lib [25]

# CSIT 2013

## Disadvantages

- Current hardware does not support projective rational floating point, but the additional computational cost of that is low, but should be considered
- Operations are approx. two times longer if not vector-vector architecture or SSE instructions are used
- Value of exponents have to be controlled – there is a possibility of exponent overflow or underflow, but easily solved by addition or subtraction to numerator and denominator in hardware.

Normalization can be made in software without a significant slowdown of computations.

- There is a significant difference between vector of projective scalar values and projective vector, i.e. representation of values

# CSIT 2013

| Vector of projective scalars | Projective vector |
| --- | --- |
| $\left[\frac{a_1}{a_0^1}, \dots, \frac{a_n}{a_0^n}\right] \asymp [(a_0^1 : a_1), \dots, (a_0^n : a_n)]$<br>$\asymp$ means equal projectively | $[a_0 : a_1, \dots, a_n]$ |

and users have to be careful in the mathematical and expression formulations.

- In the case of iterative methods on the current CPU longer computation time is to be expected. The given approach is not convenient for application of iterative methods on CPU due to exponent values control in software

Experimental verification

Let us consider following simple examples for demonstration of the proposed projective rational arithmetic in linear algebra.

## Gauss elimination

Gaussian elimination method is well known for solving a system of linear equations $\boldsymbol{Ax} = \boldsymbol{b}$ where:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

It generally produces an upper triangular matrix, *row echolon form*, and then solves the unknown $\boldsymbol{x}$ in the backward run.

The structure of the Gaussian elimination method is

**for** k := 1 **to** n
    **for** i := k+1 **to** m
        **for** j := k+1 **to** m

$$a_{ij} := a_{ij} - \frac{a_{ik} a_{kj}}{a_{kk}}$$

# CSIT 2013

The expression for $a_{ij}$ can be rewritten as

$$a_{ij} = \frac{a_{ij}a_{kk} - a_{ik}a_{kj}}{a_{kk}}$$

As the value of $a_{kk}$ can be very small, i.e. $a_{kk} \to 0$, and division by a denominator could cause significant inaccuracy or floating point overflow, exchange of rows is made in practice. If the projective notation is used

$$a_{ij} = [a_{kk}: a_{ij}a_{kk} - a_{ik}a_{kj}]$$

where: $a_{kk}$ is the homogeneous part of the expression. If $a = [a_0: a_1]$ and $\overline{a} = [a_1: a_0]$ is the reciprocal value on $a$, then we can write

$$a_{ij} = [\ \overline{a_{kk}} * (a_{ij}a_{kk} - a_{ik}a_{kj})]$$

It should be noted that a reciprocal value $\overline{a} = [a_1: a_0]$ is actually a swap of $a_0$ and $a_1$ values. The scalar value $0$ represented in the projective notation is $a = [1: 0]$, i.e. the homogeneous value is non-zero, usually $1$.

# CSIT 2013

It means that no division operation is needed, however it is still "hidden" in the homogeneous coordinate and pivoting has to be used.

As the solution of a system linear equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is equivalent to extended cross-product [9] [12] [14] we can write

$$\boldsymbol{\Omega}\, \boldsymbol{\xi} = \boldsymbol{0}$$

where

$$\boldsymbol{\Omega} = [-\boldsymbol{b}|\boldsymbol{A}] = \left[\begin{array}{c|ccc} -b_1 & a_{11} & \cdots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ -b_n & a_{n1} & \cdots & a_{nn} \end{array}\right] = \begin{bmatrix} \boldsymbol{\omega}_1 \\ \vdots \\ \boldsymbol{\omega}_n \end{bmatrix}$$

and

$$\boldsymbol{x} = \boldsymbol{\omega}_1 \times \boldsymbol{\omega}_2 \times \ldots \times \boldsymbol{\omega}_n$$

where $\boldsymbol{x}$ is the solution of the linear system of equations in the projective form, i.e. $\boldsymbol{x} = [x_0 : x_1, \ldots, x_n]$.

It can be seen that no division is required unless we need to express the computed value in the Euclidean space.

# CSIT 2013

As direct consequence of the equivalence we can easily solved many computational problems without the division operation.

The projective rational arithmetic with floating point has been verified experimentally for stability and precision of computation and inversion of the Hilbert matrix was used, which converge to a singular matrix with the growing size. The experiments proved the expected properties of the proposed approach, details can be found in [12]. [25].

## Conclusion

Projective rational arithmetic with floating point was described and fundamental arithmetic operations were described. The projective representation using homogeneous coordinates is used in computer graphics and computer vision and its application enabled to solve many problems in more effective way. As it was shown the projective representation is convenient for general numerical computation as well as it has several advantages of the standard single or double floating point representation. From the precision point of view, it offers higher range of exponents and also significantly wider range for a fraction representation.

The presented approach is convenient for vector-vector hardware architectures including GPU. If used on CPU with SSE instructions it is slightly slower than the computation with the Euclidean notation, but offers higher precision natively. If double representation is used it offers more than quadruple or extended representation.

## Acknowledgment

The author would like to thank to anonymous reviewers for their critical comments and hints that improved this paper significantly. Thanks also belong to students and colleagues at the University of West Bohemia, Plzen and VSB-Technical University, Ostrava for their suggestions. The project was supported by the Ministry of Education of the Czech Republic, project No. LH12181.

## Geometry algebra

$$\boldsymbol{ab} = \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \wedge \boldsymbol{b} \qquad \text{in } E^3 \qquad \boldsymbol{ab} = \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \times \boldsymbol{b}$$

**It is strange** – result of a dot product is a scalar value while result of the outer product (cross product) is a vector.

What is $\boldsymbol{ab}$**???**

Please, for details see

- http://geometricalgebra.zcu.cz/
- GraVisMa – recent workshops on Computer Graphics, Computer Vision & Mathematics http://www.GraVisMa.eu
- **WSCG – Conferences on Computer Graphics, Computer Vision & Visualization** since 1992 http://www.wscg.eu

# CSIT 2013

## References


- Skala,V.: Barycentric Coordinates Computation in Homogeneous Coordinates, Computers & Graphics, Elsevier, ISSN 0097-8493, Vol. 32, No.1, pp.120-127, 2008
- Skala,V.: Intersection Computation in Projective Space using Homogeneous Coordinates, Int. Journal of Image and Graphics, ISSN 0219-4678, Vol.7, No.4, pp.615-628, 2008
- Skala,V.: Length, Area and Volume Computation in Homogeneous Coordinates, Int. Journal of Image and Graphics, Vol.6., No.4, pp.625-639, ISSN 0219-4678, 2006
- Skala,V., Kaiser,J., Ondracka,V.: Library for Computation in the Projective Space, 6th Int.Conf. Aplimat, Bratislava, ISBN 978-969562-4-1, pp. 125-130, 2007
- Skala,V.: GPU Computation in Projective Space Using Homogeneous Coordinates , Game Programming GEMS 6 (Ed.Dickheiser,M.), pp.137-147, ISBN 1-58450-450-1, Charles River Media, 2006
- Skala,V.: A new approach to line and line segment clipping in homogeneous coordinates, The Visual Computer, ISSN 0178-2789, Vol.21, No.11, pp.905-914, Springer Verlag, 2005

- Generally: Authors publications with on-line DRAFTs via http://www.VaclavSkala.eu


### References related


- Yamaguchi,F.:Computer-Aided Geometric Design: A Totally Four-Dimensional Approach, Springer, 2002
- Agoston,M.K.: Computer Graphics and Geometric Modeling - Mathematics, ISBN 1-58233-817-2, Springer, 2005

# CSIT 2013

**Additional references for Geometric Algebra and Conformal Geometry**


- Bayro-Corrochano,E: Geometric Computing: For Wavelet Transforms, Robot Vision, Learning, Control and Action, Springer, 2010
- Bayro-Corrochano,E, Sobczyk,G.: Geometric Algebra with Application in Science and Engineering, Birkhauser, 2001
- Browne,J.: Grassmann Algebra - Draft Exploring Application of Extended Vector Algebra with Mathematica, Swinburn University of Technology, Melbourne, Australia, 2001
- Chevalley,C.: The Algebraic Theory of Spinors and Clifford Algebras, Springer, 1995
- Doran,Ch., Lasenby,A.: Geometric Algebra for Physicists, Cambridge Univ.Press, 2003
- Dorst,L., Fontine,D., Mann,S.: Geometric Algebra for Computer Science, Morgan Kaufmann, 2007
- Gu,X.D., Yau,S.-T.: Computational Conformal Geometry, International Press, Somerville, USA, 2008
- Hildenbrand,D.: Foundations of Geometric Algebra Computing, Springer 2012
- Li,H., Olver,P.J., Sommer,G.: Computer Algebra and Geometric Algebra with Applications, Springer
- Perwass,Ch.: Geometric Algebra with Applications in Engineering, Springer Verlag, 2008
- Sommer, G.: Geometric Computing with Clifford Algebras. Theoretical Foundations and Applications in Computer Vision and Robotics, Springer, Berlin, 2001
- Suter,J.: Geometric Algebra Primer [PDF] University of Twente, The Netherlands

# CSIT 2013


- Vince,J.: Geometric Algebra for Computer Science, Springer, 2008
- Vince,J.: Geometric Algebra: An Algebraic System for Computer Games and Animation, Springer, 2009
- Macdonald,A: A Survey of Geometric Algebra and Geometric Calculus, http://faculty.luther.edu/~macdonal, 2009
- Advances in Applied Clifford Algebras - Birkhauser-Springer journal [http://www.clifford-algebras.org]

- Digital papers on-line repository of WSCG since 1992 available from http://www.wscg.eu

# CSIT 2013

# Questions

**???**

**Contact**
**Vaclav Skala**
c/o University of West Bohemia
Faculty of Applied Sciences
Dept. of Computer Science and Engineering
CZ 306 14 Plzen, Czech Republic
**http://www.VaclavSkala.eu** skala@kiv.zcu.cz subj. CSIT 2013

Supported by the Ministry of Education of the Czech Republic, projects LH12181, LG13047