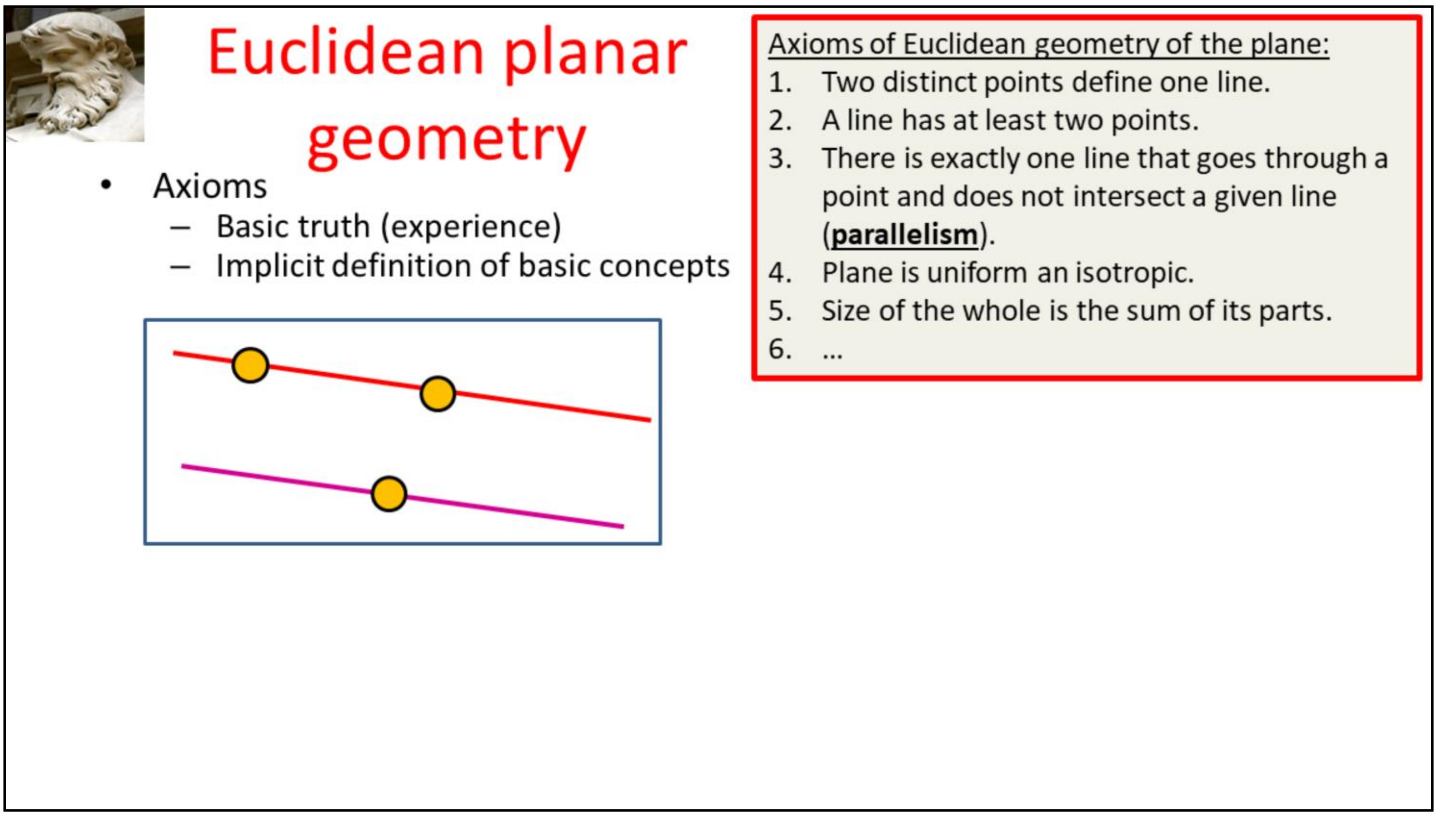
"In jeder besonderen Naturlehre nur so viel eigentliche Wissenschaft angetroffen werden könne, als darin Mathematik anzutreffen ist."
Immanuel Kant

Geometries and algebras
1. Classical geometries

Szirmay-Kalos László



Euclidean planar geometry

- Axioms
  - Basic truth (experience)
  - Implicit definition of basic concepts

<ins>Axioms of Euclidean geometry of the plane:</ins>

1. Two distinct points define one line.
2. A line has at least two points.
3. There is exactly one line that goes through a point and does not intersect a given line (**parallelism**).
4. Plane is uniform an isotropic.
5. Size of the whole is the sum of its parts.
6. …

Computer graphics works with shapes. The field of mathematics that describes shapes is the geometry, so geometry is essential in computer graphics.

Geometry, like other fields of formal science, has **axioms** that are based on experience and cannot be argued but must be accepted as true statements without arguments. From axioms other true statements, called theorems, can be deducted with logic reasoning.

For example, axioms of the **Euclidean geometry** include the postulates shown above.

Axioms have two purposes, on the one hand, they are accepted as true statements. On the other hand, axioms implicitly define **basic concepts** like points, lines etc. because they postulate their properties.

Euclidean geometry is **metric**, i.e. we can talk of the **distance** between points or separation of lines, called the **angle**, and size is an important concept. Additional axioms introduce the properties of metric quantities (distance and angle).

Having defined the axioms, we can start establishing theorems using logic inference. Such theorems will constitute the geometry. For example, theorems are:

T1: Two different lines may intersect each other at most one point.

T2: Two lines are parallel if and only if the angles in which a third line intersects them are equal.

T3: The sum of angles of a triangle is the half angle, i.e. 180 degrees.

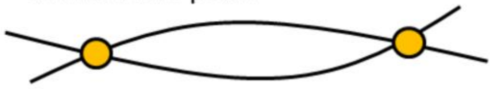
T4: Theorem of Pythagoras.

Prove them using axioms and already proven theorems!

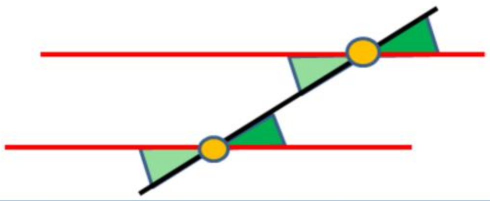
Euclidean planar geometry

- Axioms
  - Basic truth (experience)
  - Implicit definition of basic concepts
- Definitions and theorems

T1. Two different lines intersect each other at most one point



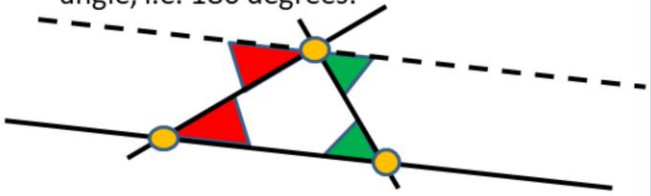
T2. Two lines are parallel if and only if the angles in which a third line intersects them are equal.



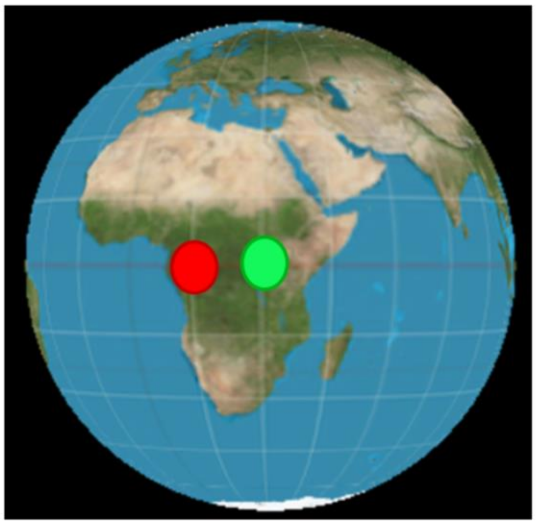
<u>Axioms of Euclidean geometry of the plane:</u>

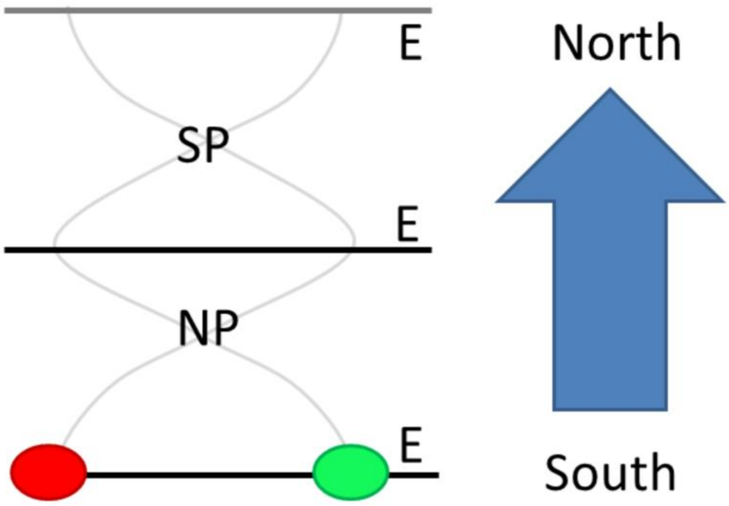
1. Two points define a line.
2. A line has at least two points.
3. There is exactly one line that goes through a point and does not intersect a given line (**parallelism**).
4. Plane is uniform an isotropic.
5. Size of the whole is the sum of its parts.
6. ...

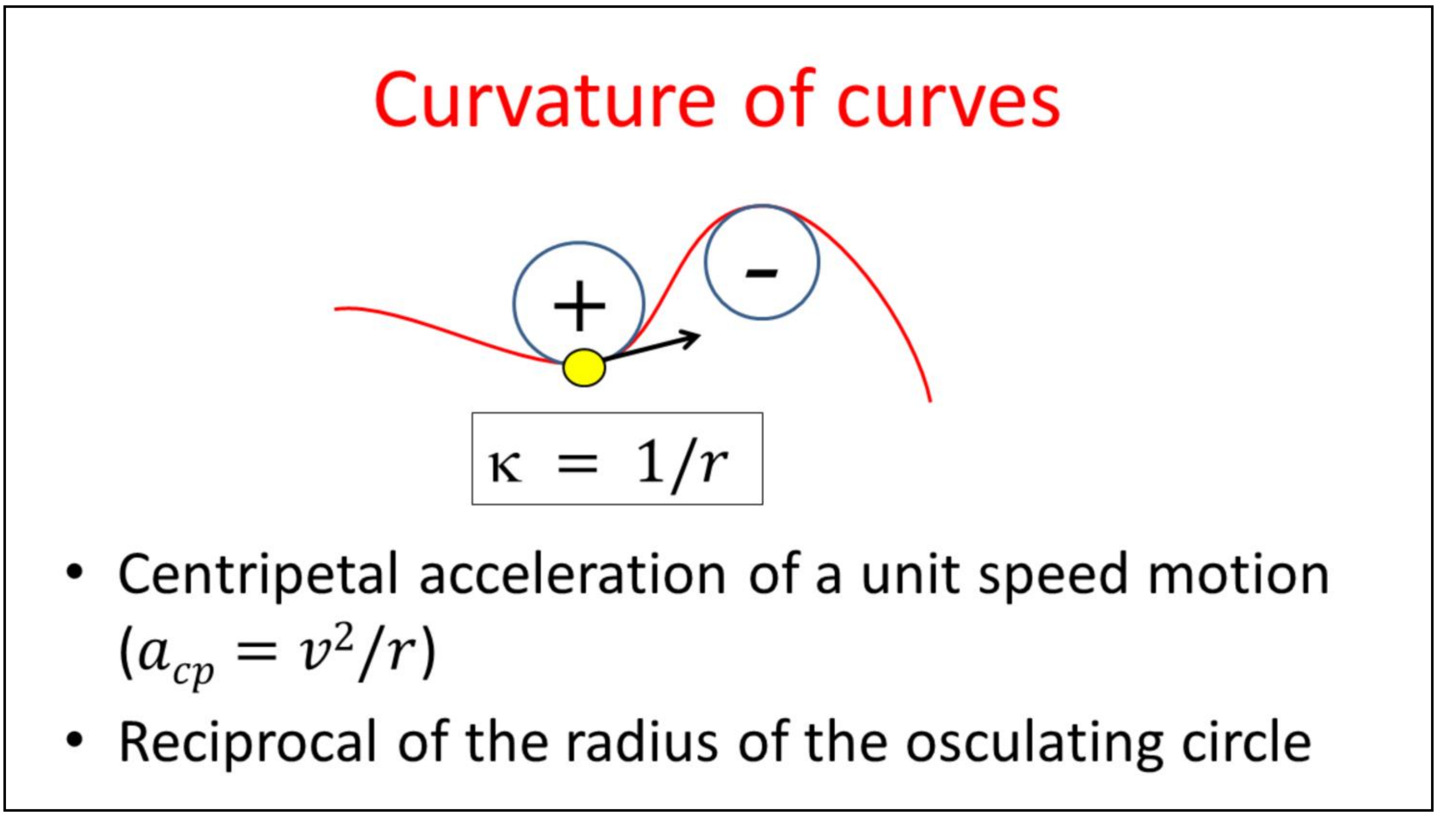
T3. The sum of angles of a triangle is the half angle, i.e. 180 degrees.



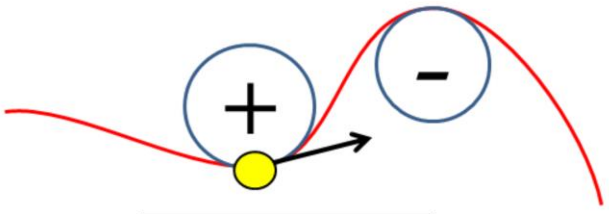
Is Euclidean geometry good for earth (Geo) measurement (meter)?







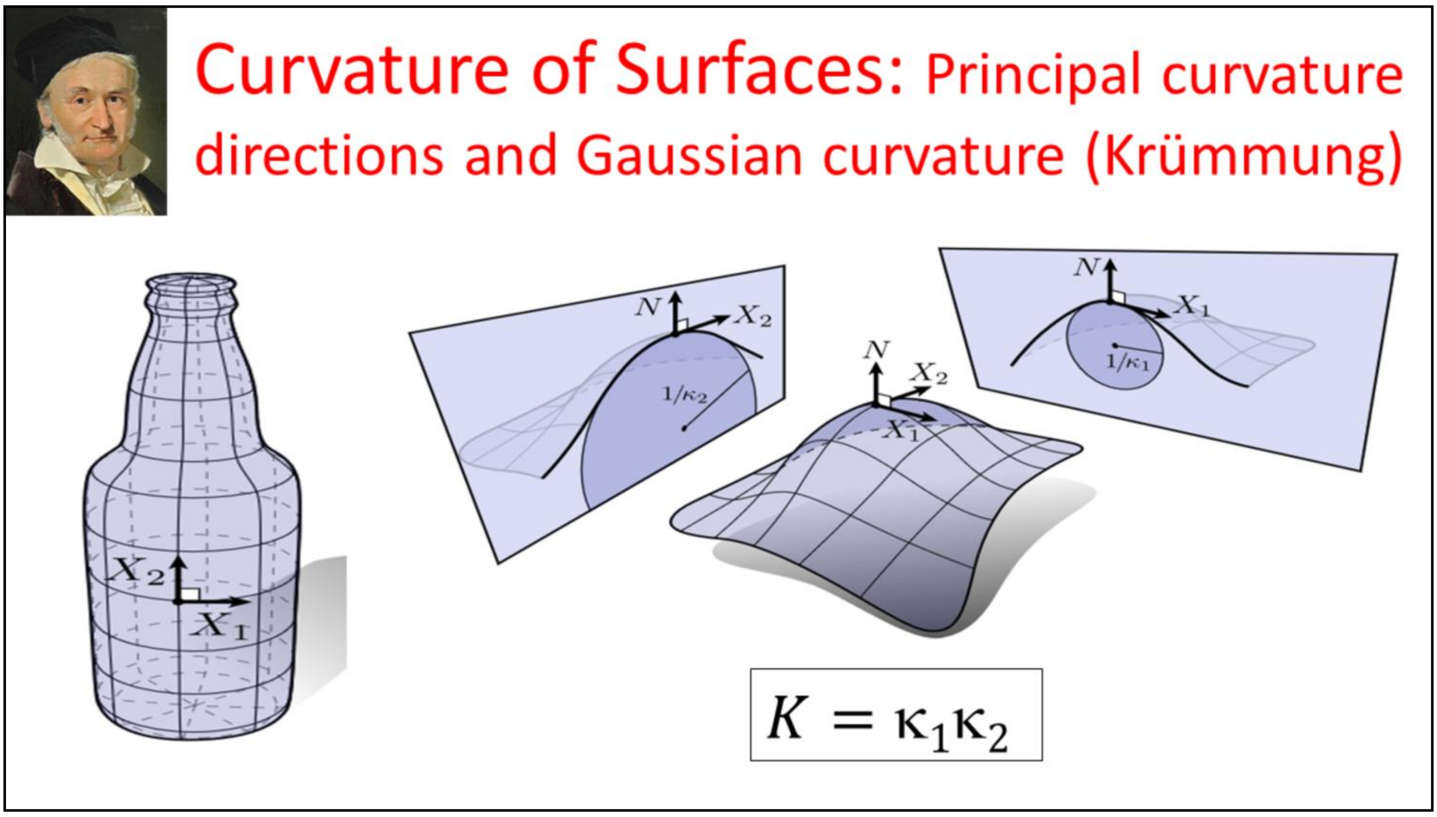
Curvature of curves



$$\boxed{\kappa = 1/r}$$

- Centripetal acceleration of a unit speed motion ($a_{cp} = v^2/r$)
- Reciprocal of the radius of the osculating circle

The **curvature** kappa of a 1D object (a curve) in one of its points is the reciprocal of the radius r of the **osculating circle**. The curvature is proportional to the centripetal force needed when we are travelling along the curve with constant unit velocity. To distinguish turning right and left, we can say that left turns have positive curvature and right turns have negative curvature. Curvature is also the second derivative of explicit function form of the function if its derivative is zero.



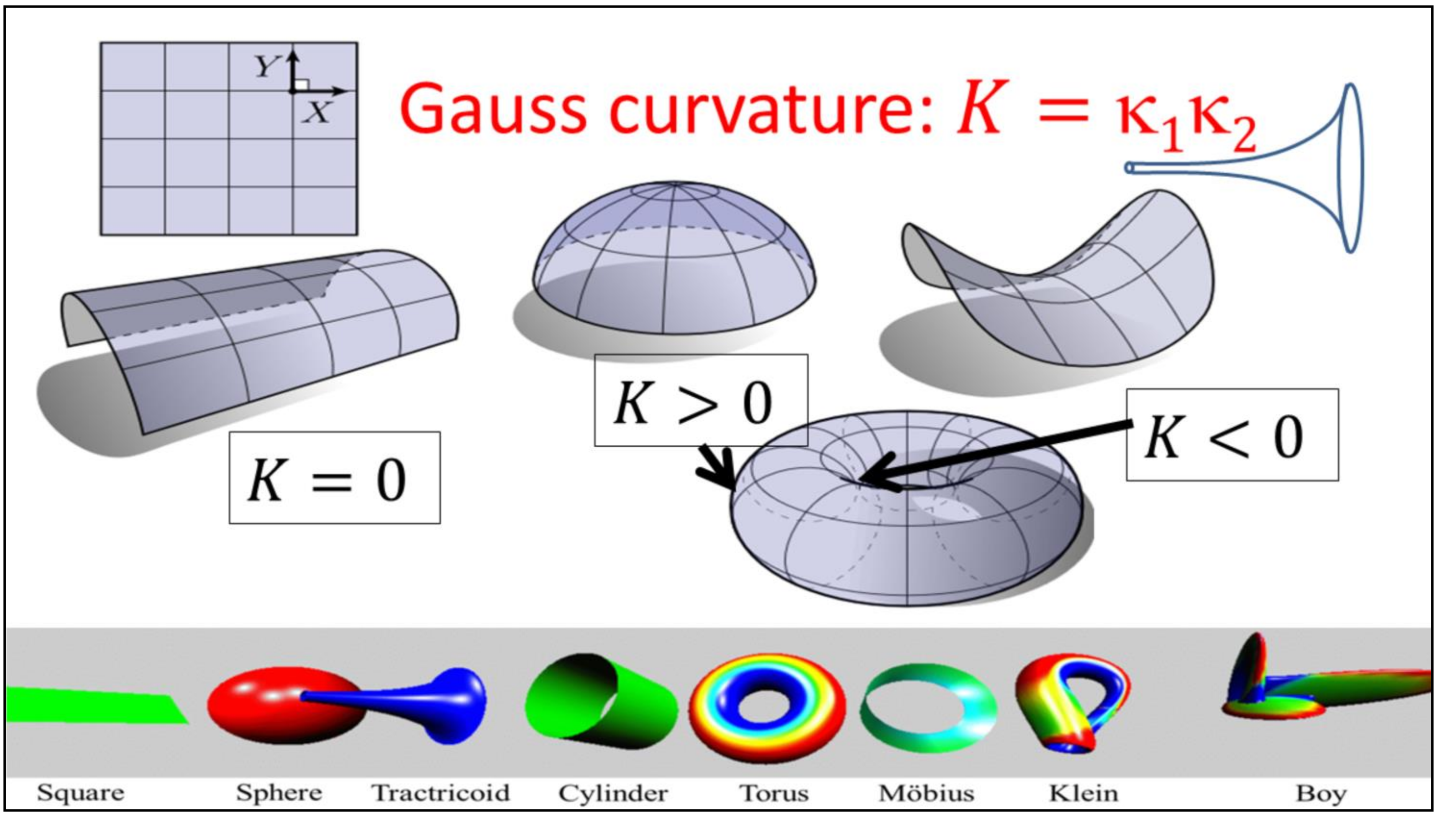
Curvature of Surfaces: Principal curvature directions and Gaussian curvature (Krümmung)

$$\boxed{K = \kappa_1 \kappa_2}$$

Let us examine the curvature K of 2D objects (surfaces) in a point. In this point, the surface has a **tangent plane** and a **normal vector** that is perpendicular to the tangent plane. The point and the normal vector define a line. Let us place a plane through this line. This plane intersects the surface and the intersection is a curve, for which the 1D curvature definition can be used. We call this directional curvature since it depends on the direction of the cutting plane. If we test all possible directions, we can notice that there are **two orthogonal directions** in which the curvature is maximum and minimum (try to prove it!). These directions are called the **principal curvature directions**.

The curvature of a surface can be described, for example, by the product of the minimum and maximum curvatures, which leads to the concept of **Gaussian curvature**.

What is the Gaussian curvature of a cylinder, plane, sphere or torus?



The **curvature of a plane** in all directions is zero, thus the Gaussian curvature of a plane is zero.

The principal curvature directions of a **cylinder** are parallel and perpendicular to the axis. In the parallel direction, the curvature is zero. In the orthogonal direction, the curvature is reciprocal of the radius of the cylinder. Thus, the Gaussian curvature of a cylinder is zero. Oops, planes and cylinders are relatives, both of them have zero Gaussian curvature. Indeed, we can unfold a cylinder and make it a plane without distortions.

The **curvature of a sphere** in all directions is the reciprocal of the sphere radius. Thus the Gaussian curvature is the reciprocal of the square of the radius. The Gaussian curvature of a sphere is positive, and thus is different from the Gaussian curvature of a plane, which is zero. It means that a sphere cannot be unfolded into a plane without distortions.

If in one of the principal curvature directions the intersection curve is above the tangent plane while it is below in the other principal direction, the two directional curvatures have opposite signs, so the **Gaussian curvature is negative**. Examples are the saddle or the trumpet, called tractricoid, which is the rotation of a tractrix. The Gaussian curvature of a torus changes from point to point. It is positive in the outer half and negative in the inner half. The figure at the bottom shows different objects colored according to the curvature information. We used rainbow colors, blue corresponds to negative curvature, green to zero curvature, and red to positive curvature.

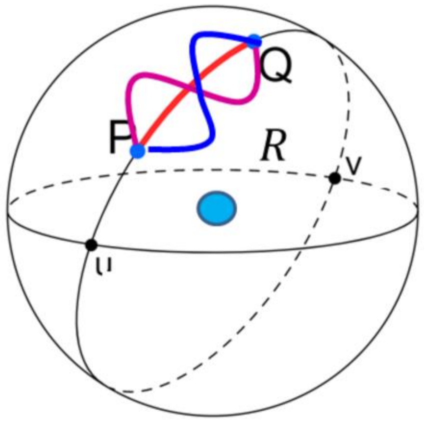
Spherical geometry

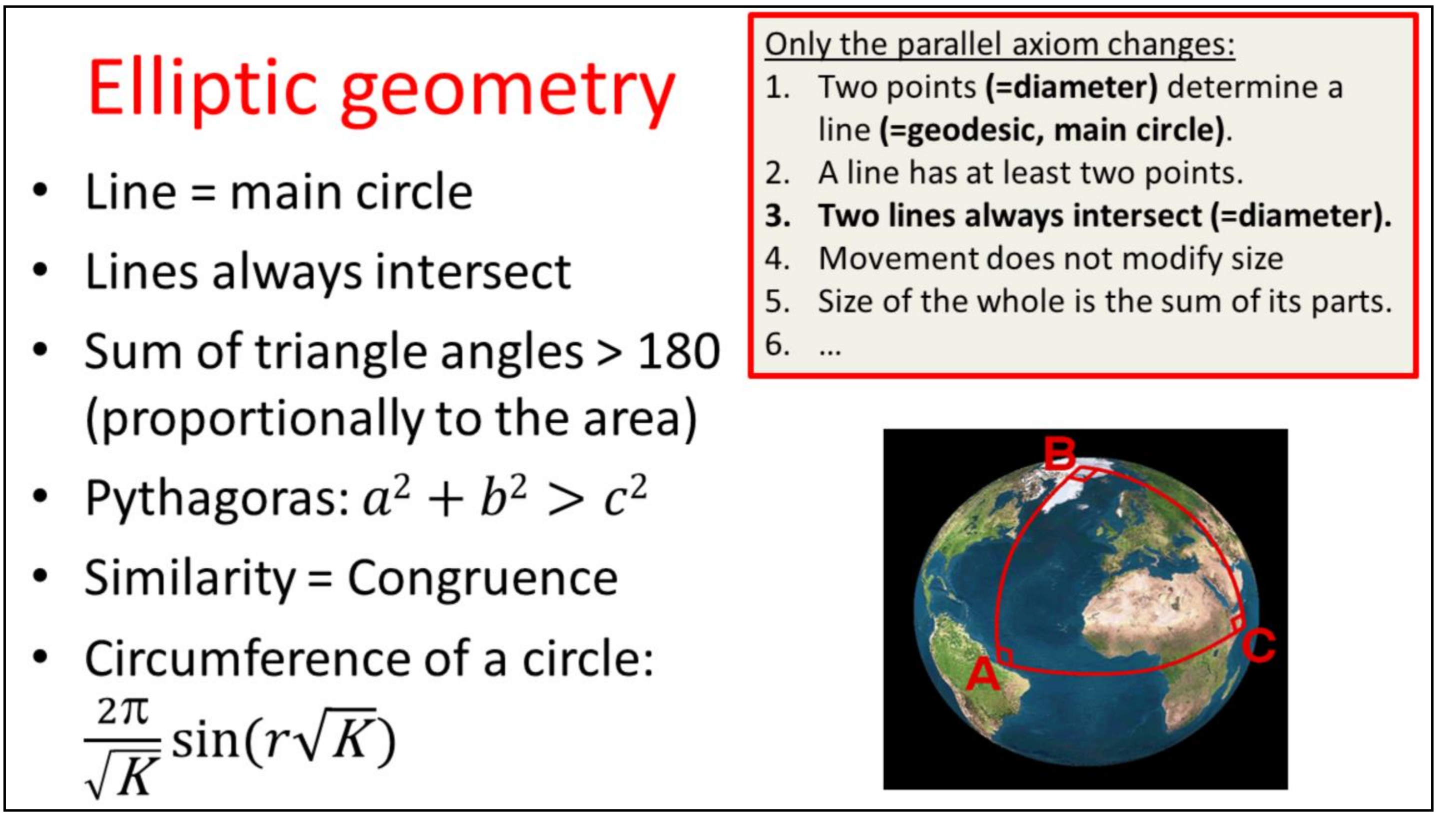
$$\boxed{x^2 + y^2 + z^2 = R^2}$$

Euclidean axioms are wrong:

1. Two points **not always** define a line.
2. A line has at least two points.
3. **Two lines always intersect.**
4. Plane is uniform and isotropic.
5. Size of the whole is the sum of its parts.
6. …

- Constant positive curvature
- Line = main circle
  = shortest path



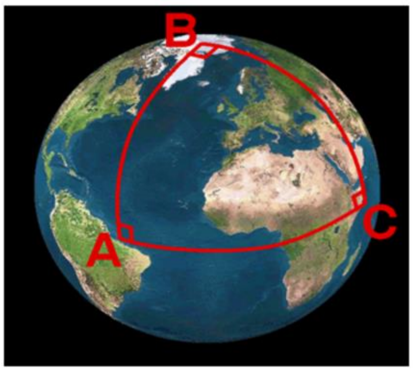


Elliptic geometry

- Line = main circle
- Lines always intersect
- Sum of triangle angles > 180 (proportionally to the area)
- Pythagoras: $a^2 + b^2 > c^2$
- Similarity = Congruence
- Circumference of a circle: $\frac{2\pi}{\sqrt{K}}\sin(r\sqrt{K})$

Only the parallel axiom changes:

1. Two points **(=diameter)** determine a line **(=geodesic, main circle)**.
2. A line has at least two points.
3. **Two lines always intersect (=diameter).**
4. Movement does not modify size
5. Size of the whole is the sum of its parts.
6. …



The axioms of the Euclidean geometry are based on experience gathered on walking on a flat terrain or not too large distances.

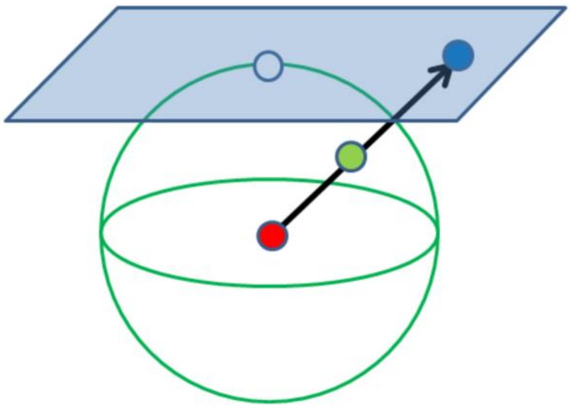
If Euclid had travelled through continents of the spherical Earth, he would have different experience and would have established different axioms. For example, on the Earth parallel lines meet at exactly 10 million km from start .

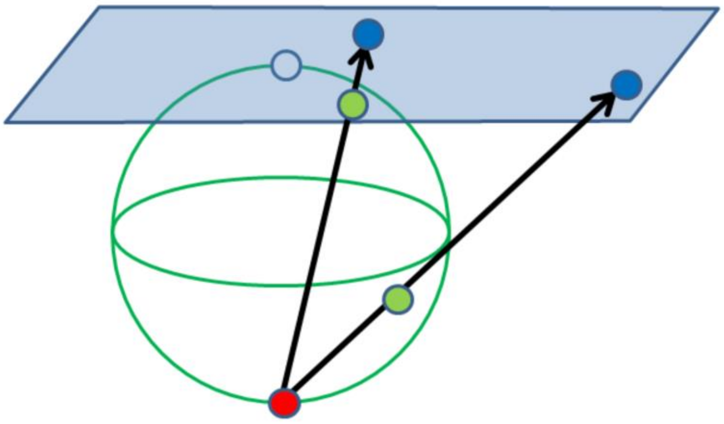
The line on a sphere, using the concept of the shortest path between two points, is the main circle, which is the intersection of the sphere and the plane defined by the two points and the center of the sphere. Airplanes fly along these spherical lines.

If we interpret spherical points as diameters, the first two axioms of the Euclidean geometry remain valid. However, the third axiom is invalid for spherical lines and points. Here lines always intersect in a diameter. Consequently, the theorems of Euclidean geometry are not true. For example, **the sum of the angles of a triangle is always larger than 180 degrees.**

Maps are Euclidean unfolding of the non-Euclidean (spherical) plane. As the curvature of the sphere is not zero, a map must distort distances and/or angles. There are different options, but all of them distort somehow. Mercator’s map, for example, preserves angles but apply drastic distance distortions

Maps of the sphere





Central projection

- Preserves lines

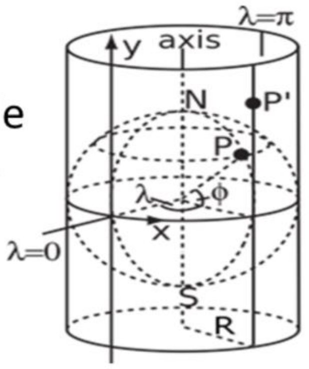
Stereographic projection

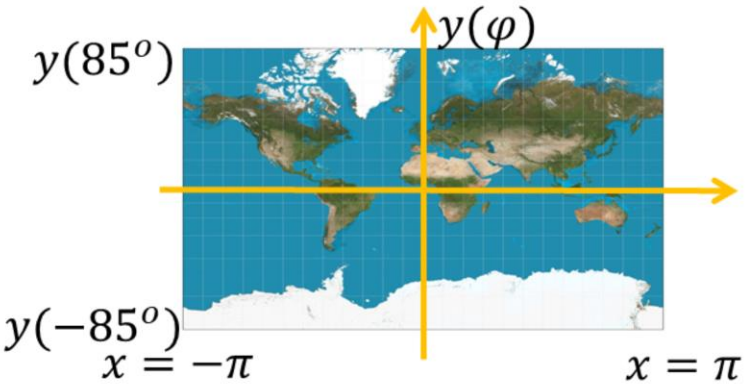
- Preserves circles and angles

(Gerardus) Mercator map

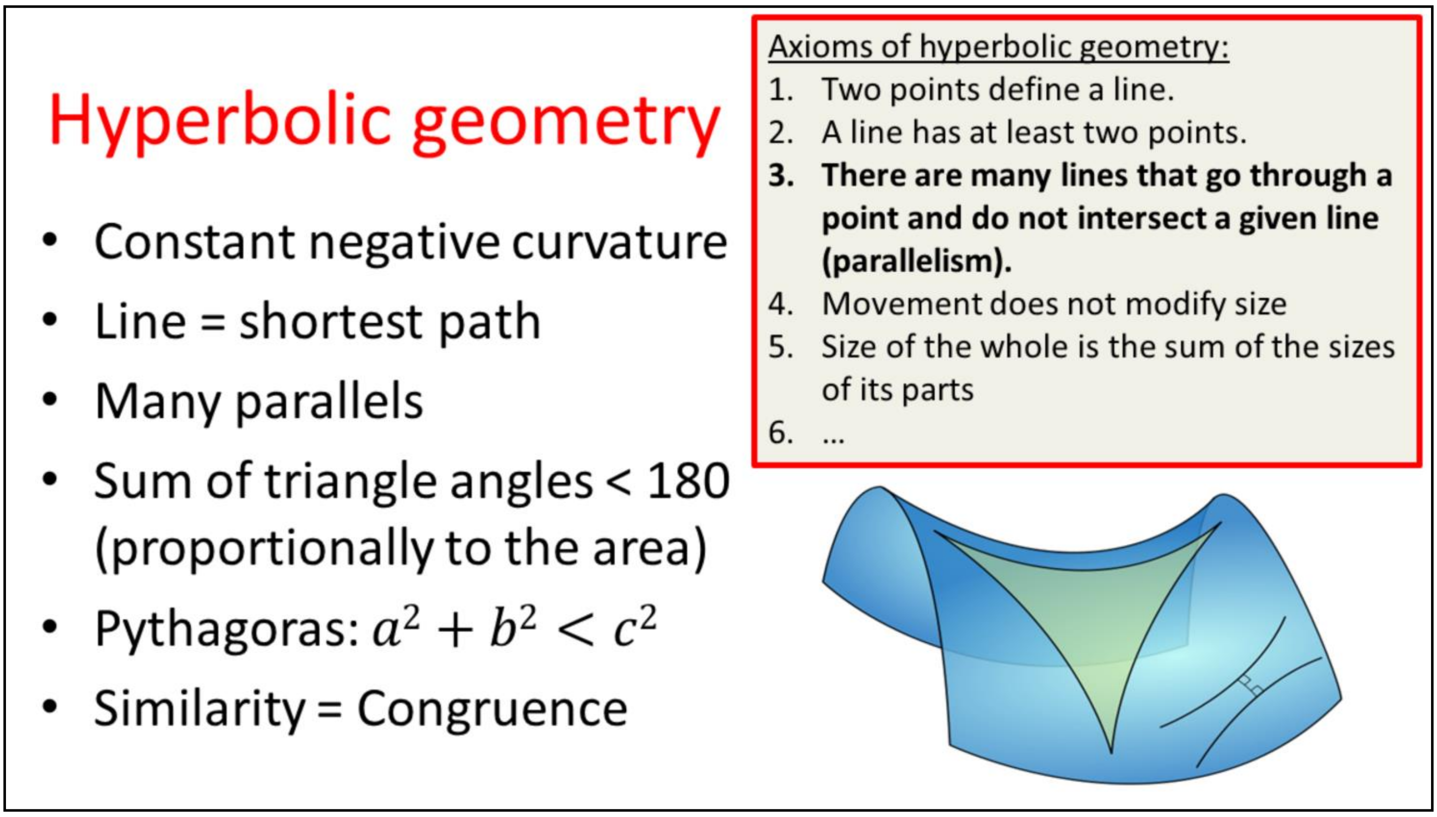
- Preserves angles but distorts distances

λ = longitude
ϕ = latitude





$$\boxed{\begin{aligned}&x = \lambda\\ &y = \log\left(\tan\left(\frac{\pi}{4} + \frac{\varphi}{2}\right)\right)\end{aligned}}$$



Hyperbolic geometry

- Constant negative curvature
- Line = shortest path
- Many parallels
- Sum of triangle angles < 180 (proportionally to the area)
- Pythagoras: $a^2 + b^2 < c^2$
- Similarity = Congruence

Axioms of hyperbolic geometry:

1. Two points define a line.
2. A line has at least two points.
3. **There are many lines that go through a point and do not intersect a given line (parallelism).**
4. Movement does not modify size
5. Size of the whole is the sum of the sizes of its parts
6. …

Spaces of negative curvature need another geometry called **hyperbolic geometry**. We should change just a single word in the axioms: there can be more than one parallel line crossing a given point. This small modification may invalidate most of the theorems of Euclidean geometry and lead into a "new world".

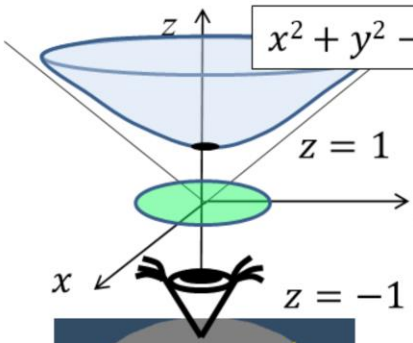
In **hyperbolic geometry the sum of angles of a triangle is less than 180 degrees, proportionally to the size.**

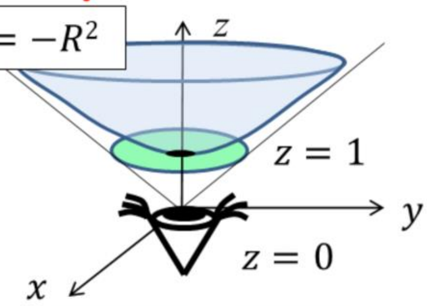
Hyperbolic geometry is also often studied with maps. Here we show the Beltrami-Poincare disc as a map, which is a conformal, i.e. angle preserving map of the hyperbolic plane. However, it is not distance preserving, infinity is at the boundary of the base circle. Esher loved this model and created many different artworks according to its rules.

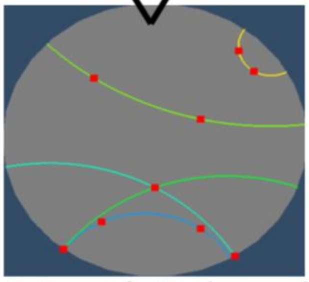
How can we figure it out whether our universe has positive, negative or zero curvature? This is an important question since it determines whether the universe will expand without limits or will start shrinking sooner or later.

Beltrami – Poincaré/Klein disc

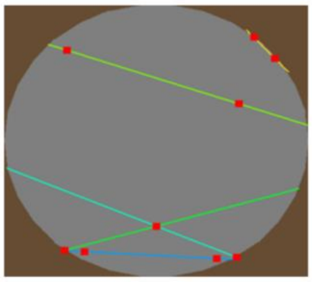
$$x^2 + y^2 - z^2 = -R^2$$



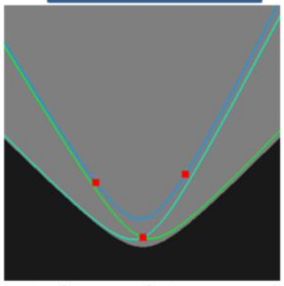




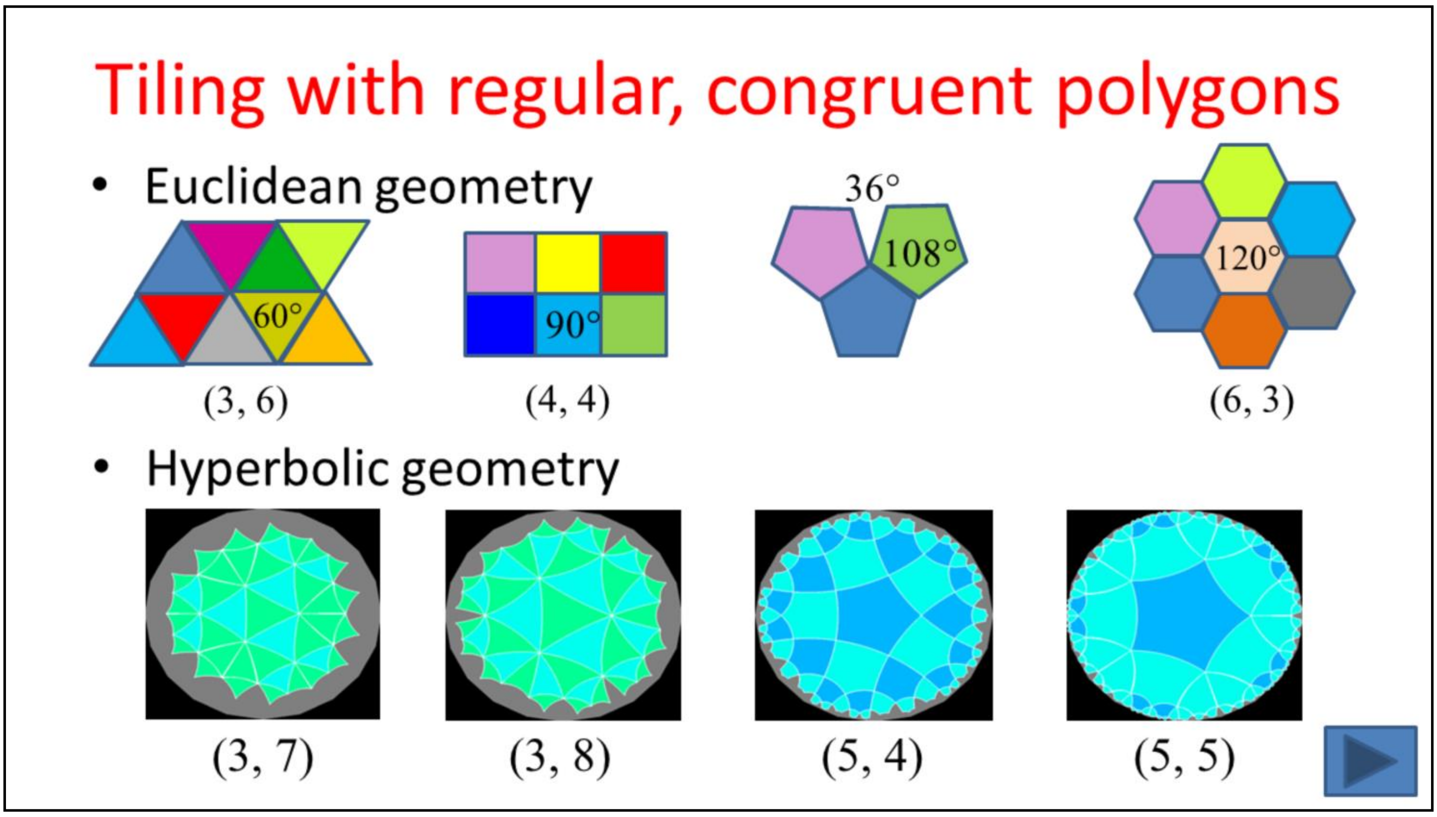
Poincaré: Angles and circles are preserved; Lines are circular arcs



Klein: Lines are preserved



Minkowski space
Sphere!



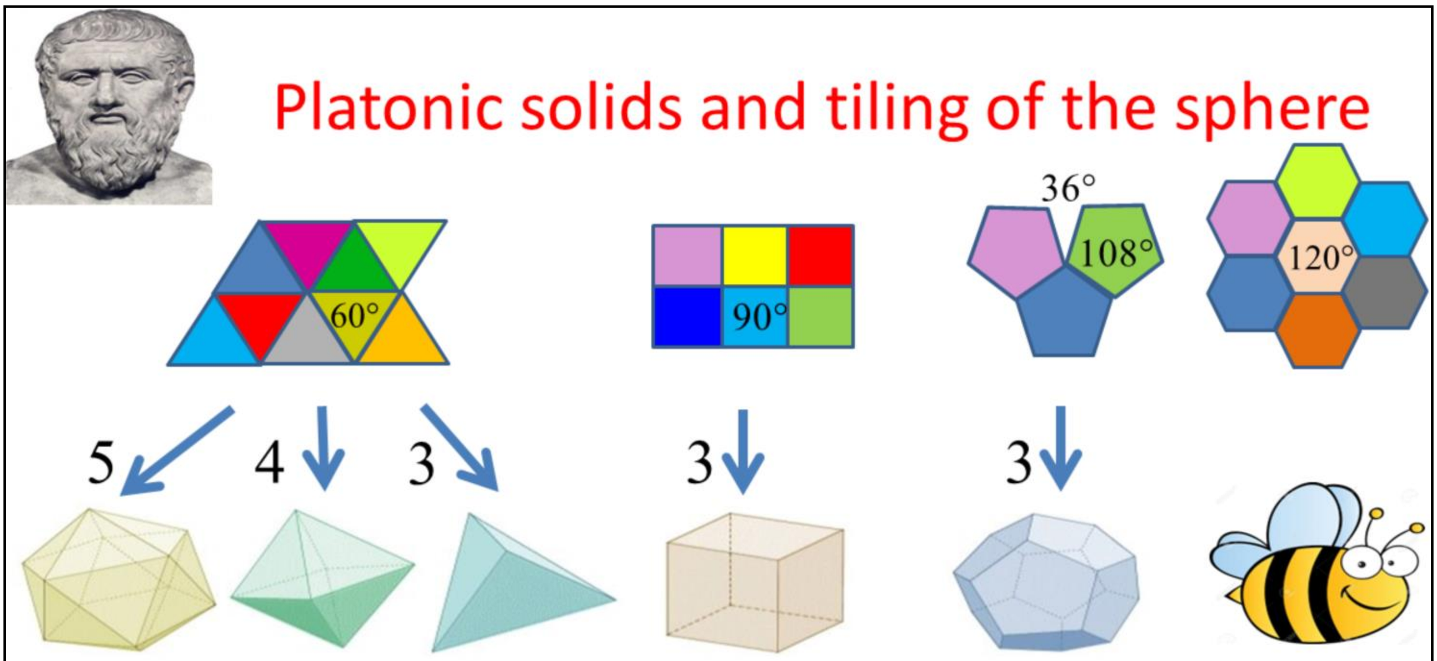
Tiling with regular, congruent polygons

- Euclidean geometry

- Hyperbolic geometry

The richness and beauty of hyperbolic geometry is often demonstrated with tiling, i.e. tessellation of the plane with, for example, regular, congruent polygons. The Euclidean plane has only three such tessellations: with triangles, squares, and hexagons. However, thanks to the fact that sum of angles of a triangle can be smaller than 180 degrees in hyperbolic geometry, a regular polygon with any number of vertices can tile the plane, and can do it with infinitely many ways. Just the size of the polygon should be set properly.

Platonic solids and tiling of the sphere

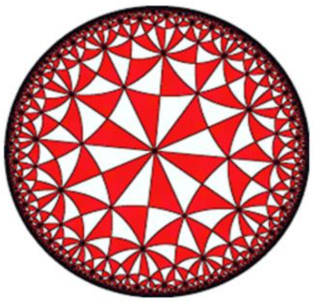


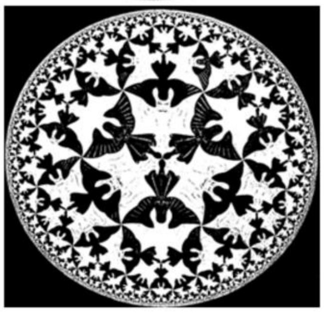
Escher and the Poincaré disc

Circle limit III

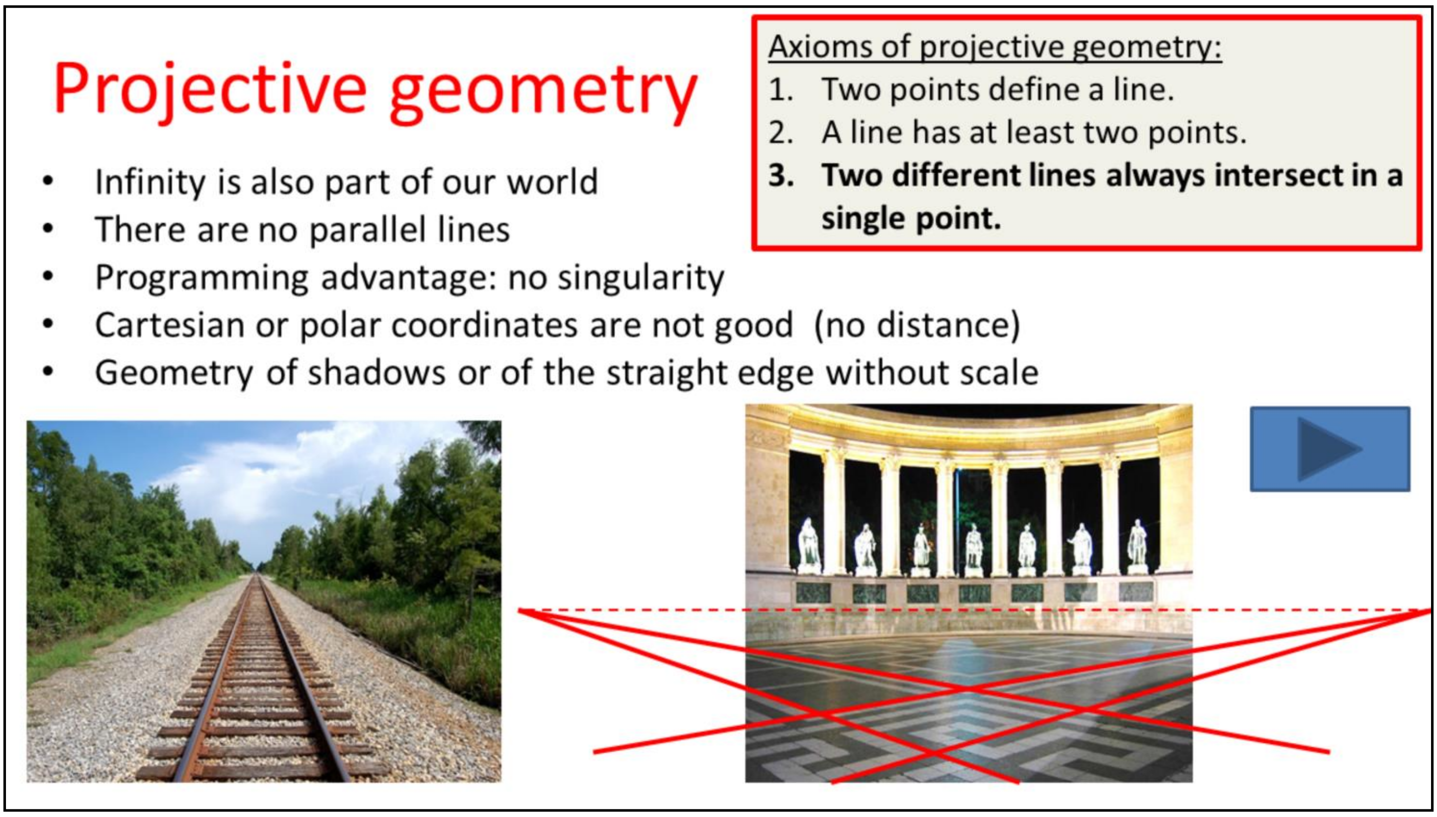
Circle limit I

Circle limit IV





Hyperbolic geometry



Projective geometry

- Infinity is also part of our world
- There are no parallel lines
- Programming advantage: no singularity
- Cartesian or polar coordinates are not good (no distance)
- Geometry of shadows or of the straight edge without scale

Axioms of projective geometry:
1. Two points define a line.
2. A line has at least two points.
3. **Two different lines always intersect in a single point.**

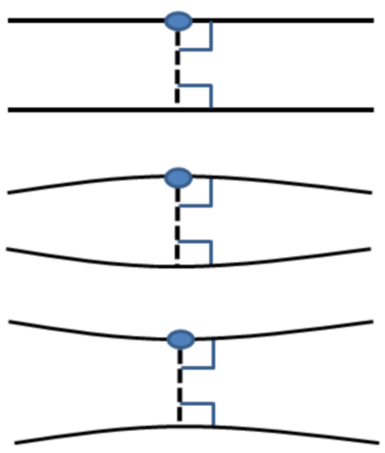
In Euclidean geometry parallel lines do not intersect, that is, a point at infinity (where parallel line would meet) is not part of the Euclidean plane. However, we can see the intersection of parallel lines, so a geometry where infinity is also included makes sense.

If we define axioms differently, **we can add points at infinity to the plane** making all lines, even parallel lines, intersecting. Clearly, this is a different geometry with different axioms and theorems, which is called the **projective geometry**. Projective geometry is not metric since distance cannot be defined in it. The reason is that the distance from points at infinity is infinite, but infinite is not a number. As a result, we cannot use coordinate systems that are based on the concept of distance, e.g. Cartesian coordinate systems are useless here. We should find another algebraic basis.

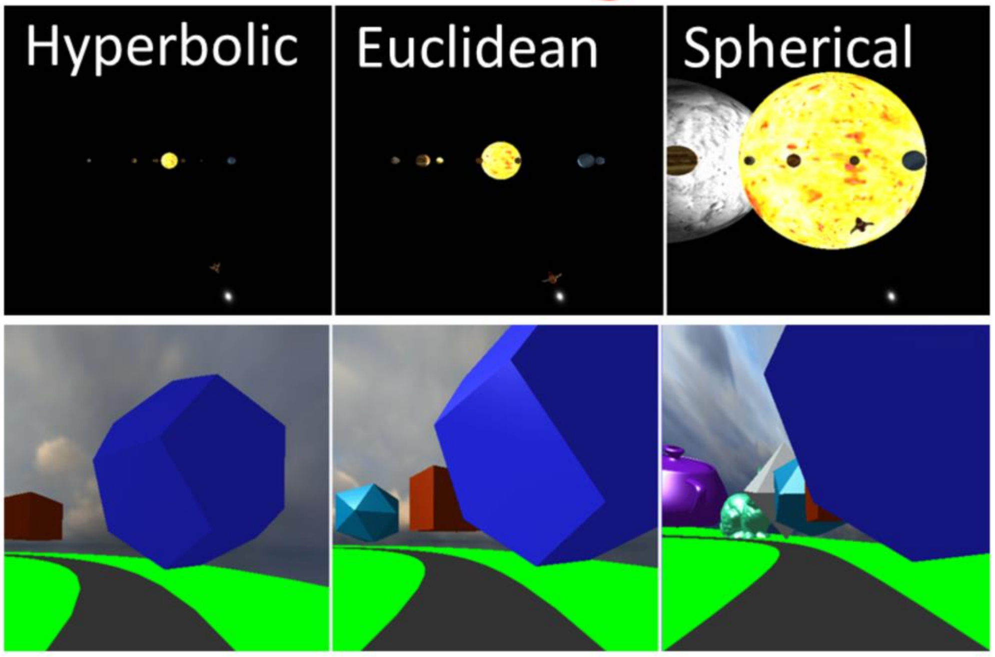
Note that the first three axioms of projective geometry are identical to those of the spherical geometry. They are close relatives. As a result, a projective line has similar properties to a circle. You can go around it.

Model geometries

- Euclidean
  - 1 non-intersecting line (parallel)
  - Zero curvature
- Spherical
  - 0 non-intersecting line
  - Positive curvature
- Hyperbolic
  - More than one non-intersecting line
  - Negative curvature
- Projective
  - 0 non-intersecting line
  - Not metric: includes infinity



Feeling

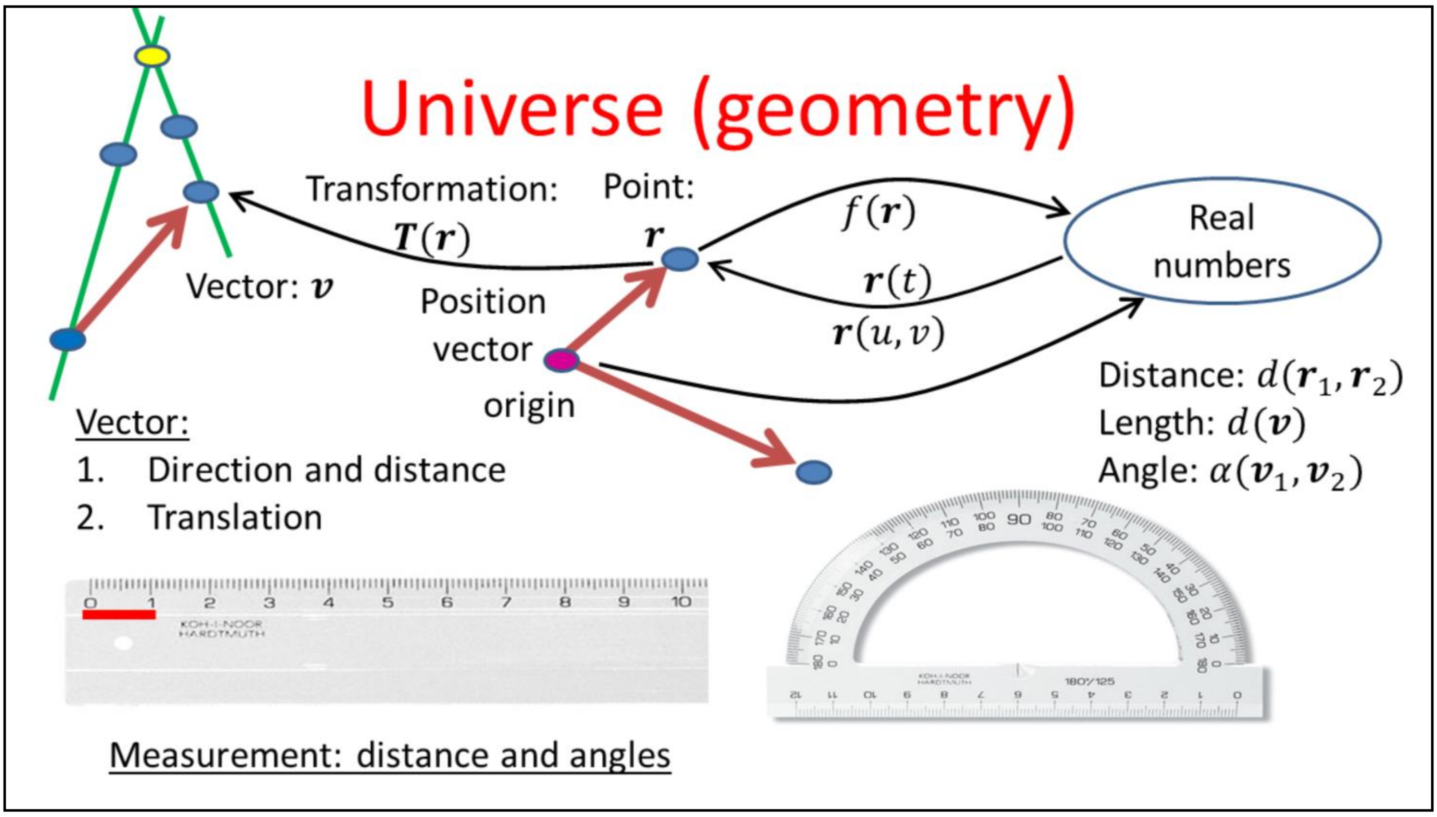


"Tant que l'Algèbre et la Géométrie ont été séparées, leurs progrès ont été lents et leurs usages bornés; mais lorsque ces deux sciences se sont réunies, elles se sont prêté des forces mutuelles et ont marché ensemble d'un pas rapide vers la perfection."

Joseph-Louis Lagrange

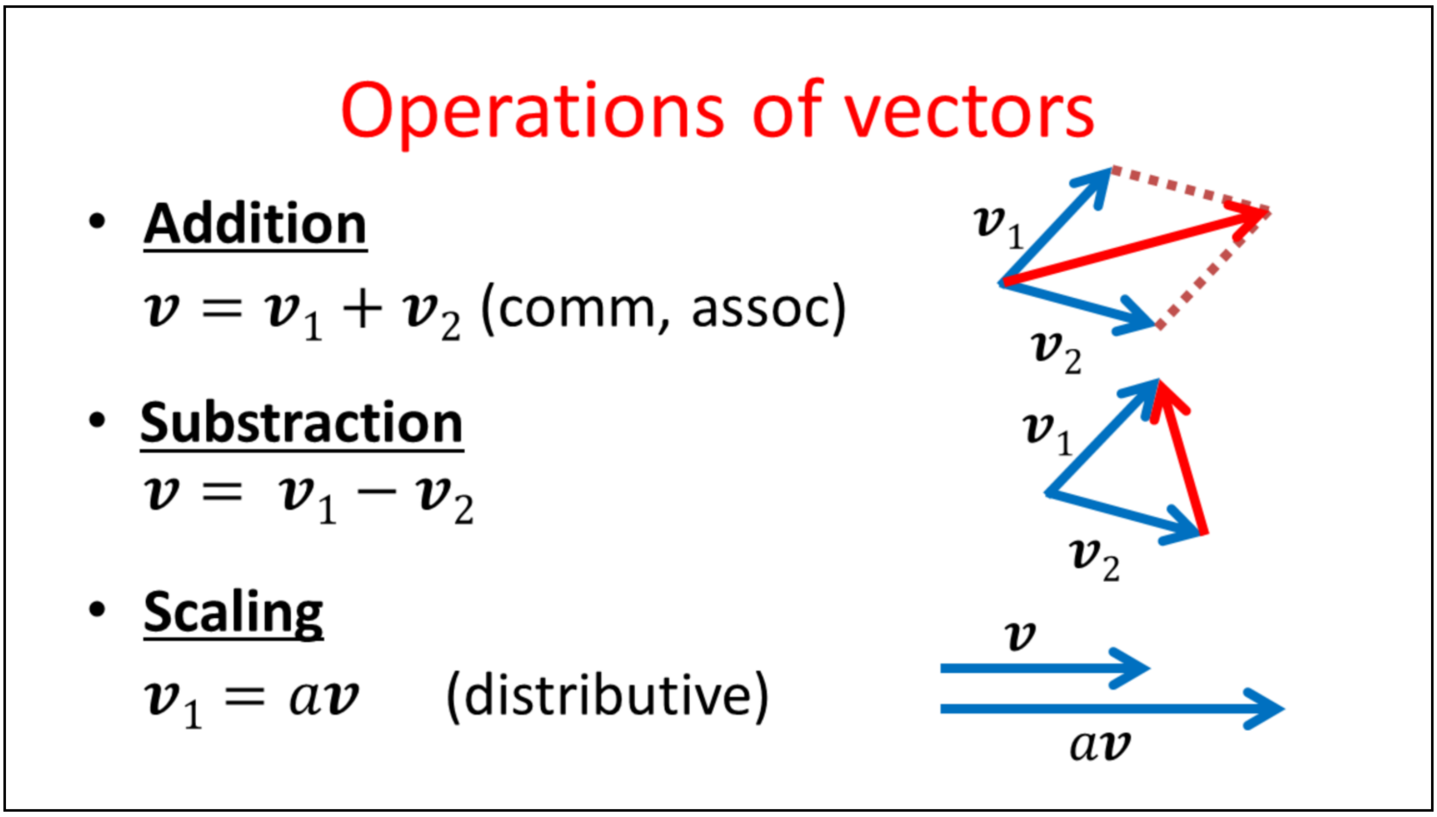
Geometries and algebras
2. Vector algebra

Szirmay-Kalos László


Universe (geometry)
Transformation: $\boldsymbol{T}(\boldsymbol{r})$
Point: $\boldsymbol{r}$
$f(\boldsymbol{r})$
Real numbers
$\boldsymbol{r}(t)$
$\boldsymbol{r}(u, v)$
Vector: $\boldsymbol{v}$
Position vector
origin
Distance: $d(\boldsymbol{r}_1, \boldsymbol{r}_2)$
Length: $d(\boldsymbol{v})$
Angle: $\alpha(\boldsymbol{v}_1, \boldsymbol{v}_2)$
Vector:
1. Direction and distance
2. Translation
Measurement: distance and angles

By definition a translation is a **vector**, which has **direction** and **length**. The length is denoted by the absolute value of the vector.

If we select a special reference point, called the **origin**, then every point has a unique vector that translates the origin to here, or from the other point of view, every vector unambiguously defines a point that is reached if the origin is translated by this vector. Such vectors are called **position vectors**. The fact that there is a one-to-one correspondence between points and position vectors does not mean that points and vectors would be identical objects (wife and husband are also strongly related and unambiguously identify each other, but are still different objects with specific operations).



Operations of vectors

- **Addition**

  $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$ (comm, assoc)

- **Substraction**

  $\boldsymbol{v} = \boldsymbol{v}_1 - \boldsymbol{v}_2$

- **Scaling**

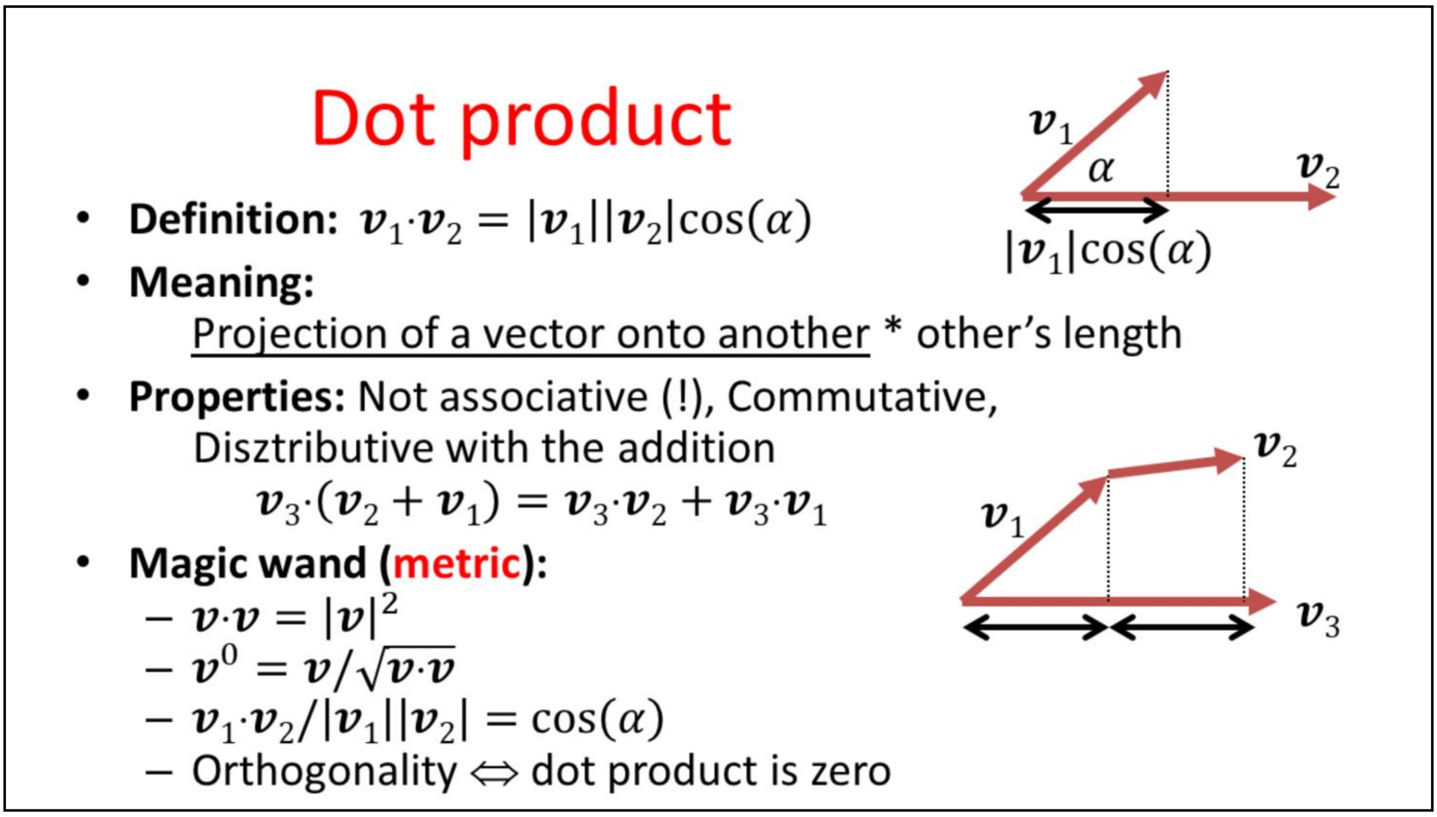
  $\boldsymbol{v}_1 = a\boldsymbol{v}$ (distributive)

$\boldsymbol{v}_1$
$\boldsymbol{v}_2$
$\boldsymbol{v}_1$
$\boldsymbol{v}_2$
$\boldsymbol{v}$
$a\boldsymbol{v}$

Concerning vector operations, we can talk of **addition** that means the execution of the two translations one after the other. The resulting translation is independent of the order, so vector addition is **commutative** (parallelogram rule). If we have more than two vectors, parentheses can rearranged so it is also **associative.** Vector addition has an inverse, because we can ask which vector completes the translation of v2 to get a resulting translation v.

Vectors can be **multiplied by a scalar**, which scales the length but does not modify the direction. Scaling is **distributive**, i.e. scaling a sum of two vectors results in the same vector as adding up the two scaled versions.

We have to emphasize that the nice properties of commutativity, associativity, and distributivity are usually not evident and sometimes not even true for vector operations. They must be proven using the axioms of Euclidean geometry.



Dot product

- **Definition:** $\boldsymbol{v}_1 \cdot \boldsymbol{v}_2 = |\boldsymbol{v}_1||\boldsymbol{v}_2|\cos(\alpha)$
- **Meaning:**
  Projection of a vector onto another * other's length
- **Properties:** Not associative (!), Commutative,
  Disztributive with the addition
  $\boldsymbol{v}_3 \cdot (\boldsymbol{v}_2 + \boldsymbol{v}_1) = \boldsymbol{v}_3 \cdot \boldsymbol{v}_2 + \boldsymbol{v}_3 \cdot \boldsymbol{v}_1$
- **Magic wand (metric):**
  - $\boldsymbol{v} \cdot \boldsymbol{v} = |\boldsymbol{v}|^2$
  - $\boldsymbol{v}^0 = \boldsymbol{v}/\sqrt{\boldsymbol{v} \cdot \boldsymbol{v}}$
  - $\boldsymbol{v}_1 \cdot \boldsymbol{v}_2 / |\boldsymbol{v}_1||\boldsymbol{v}_2| = \cos(\alpha)$
  - Orthogonality $\Leftrightarrow$ dot product is zero

Vectors can be multiplied in different ways. The first possibility is the **scalar product** (aka **dot** or inner product) that assigns a scalar to two vectors. By definition, the resulting scalar is equal to the product of the lengths of the two vectors and the cosine of the angle between them.

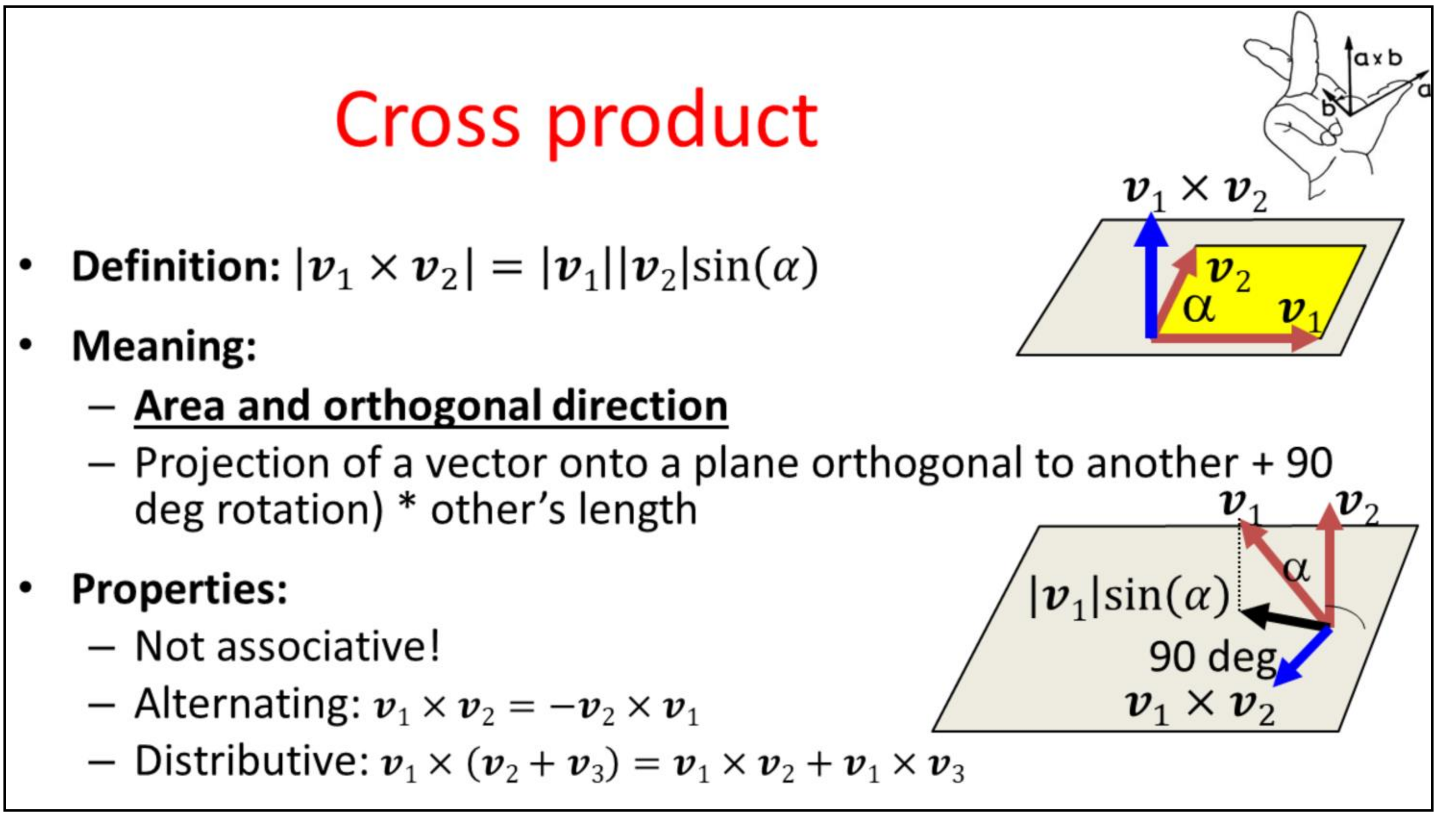
The geometric meaning of scalar product is the length of projection of one vector onto the other, multiplied by the lengths of the others.

Scalar product is **commutative** (symmetric), which is obvious from the definition.

Scalar product is **distributive with the vector addition**, which can be proven by looking at the geometric interpretation. Projection is obviously distributive (the projection of the sum of two vectors is the same as the sum of the two projections.

Scalar product is **NOT associative**!

There is a direct relationship between dot product and the absolute value. The scalar product of a vector with itself is equal to the square of its length according to the definition since cos(0)=1.



Cross product

- **Definition:** $|\boldsymbol{v}_1 \times \boldsymbol{v}_2| = |\boldsymbol{v}_1||\boldsymbol{v}_2|\sin(\alpha)$
- **Meaning:**
  - **Area and orthogonal direction**
  - Projection of a vector onto a plane orthogonal to another + 90 deg rotation) * other's length
- **Properties:**
  - Not associative!
  - Alternating: $\boldsymbol{v}_1 \times \boldsymbol{v}_2 = -\boldsymbol{v}_2 \times \boldsymbol{v}_1$
  - Distributive: $\boldsymbol{v}_1 \times (\boldsymbol{v}_2 + \boldsymbol{v}_3) = \boldsymbol{v}_1 \times \boldsymbol{v}_2 + \boldsymbol{v}_1 \times \boldsymbol{v}_3$

a x b
a
b
$\boldsymbol{v}_1 \times \boldsymbol{v}_2$
$\boldsymbol{v}_2$
α
$\boldsymbol{v}_1$
$\boldsymbol{v}_1$
$\boldsymbol{v}_2$
α
$|\boldsymbol{v}_1|\sin(\alpha)$
90 deg
$\boldsymbol{v}_1 \times \boldsymbol{v}_2$

Vectors can be multiplied with the rules of the **vector (aka cross) product** as well. The result is a vector of length equal to the product of the lengths of the two vectors and the sine of their angle. The resulting vector is perpendicular to both operands and points into the direction of the middle finger of our right hand if our thumb points into the direction of the first operand and our index finger into the direction of the second operand (**right hand rule**).

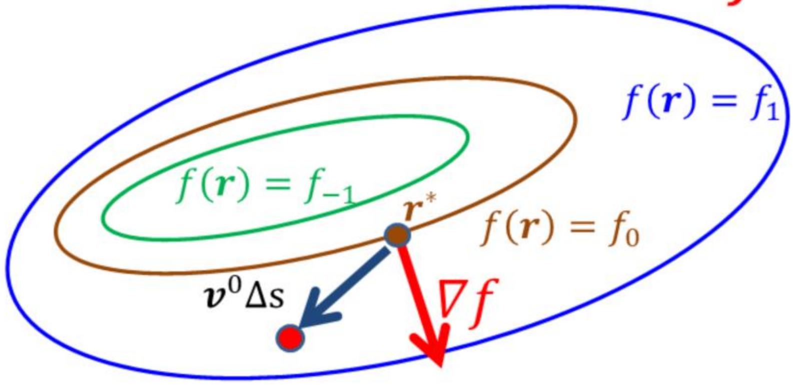
Cross product can be given two different geometric interpretations. The first is a vector meeting the requirements of the right hand rule and of length equal to the area of the parallelogram of edge vectors of the two operands.

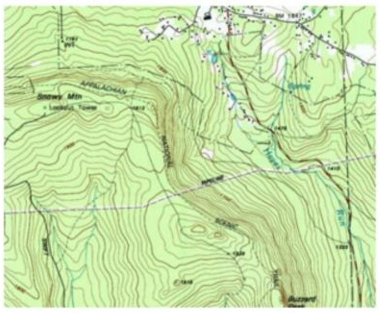
The second geometric interpretation is the projection of the second vector onto the plane perpendicular to the first vector, rotating the projection by 90 degrees around the first vector, and finally scaling the result with the length of the first vector.

Cross product is **NOT commutative** but **anti-symmetric** or alternating, which means that exchanging the two operands the result is multiplied by -1.

Cross product is **distributive with the addition**, which can be proven by considering its second geometric interpretation. Projection onto a plane is distributive with addition, so are rotation and scaling. Cross product is **NOT associative**.

Gradient: ∇f

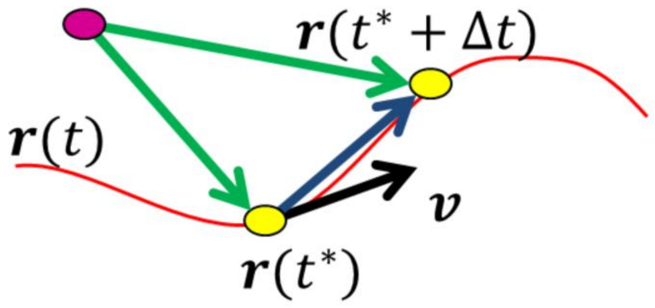

$f(\boldsymbol{r}) = f_1$
$f(\boldsymbol{r}) = f_{-1}$
$\boldsymbol{r}^*$
$f(\boldsymbol{r}) = f_0$
$\boldsymbol{v}^0 \Delta s$
∇f



$\lim \frac{f\left(\boldsymbol{r}^* + \boldsymbol{v}^0 \Delta s\right) - f(\boldsymbol{r}^*)}{\Delta s} = \frac{\mathrm{d} f\left(\boldsymbol{r}^* + \boldsymbol{v}^0 s\right)}{\mathrm{d} s}$ **directional derivative**

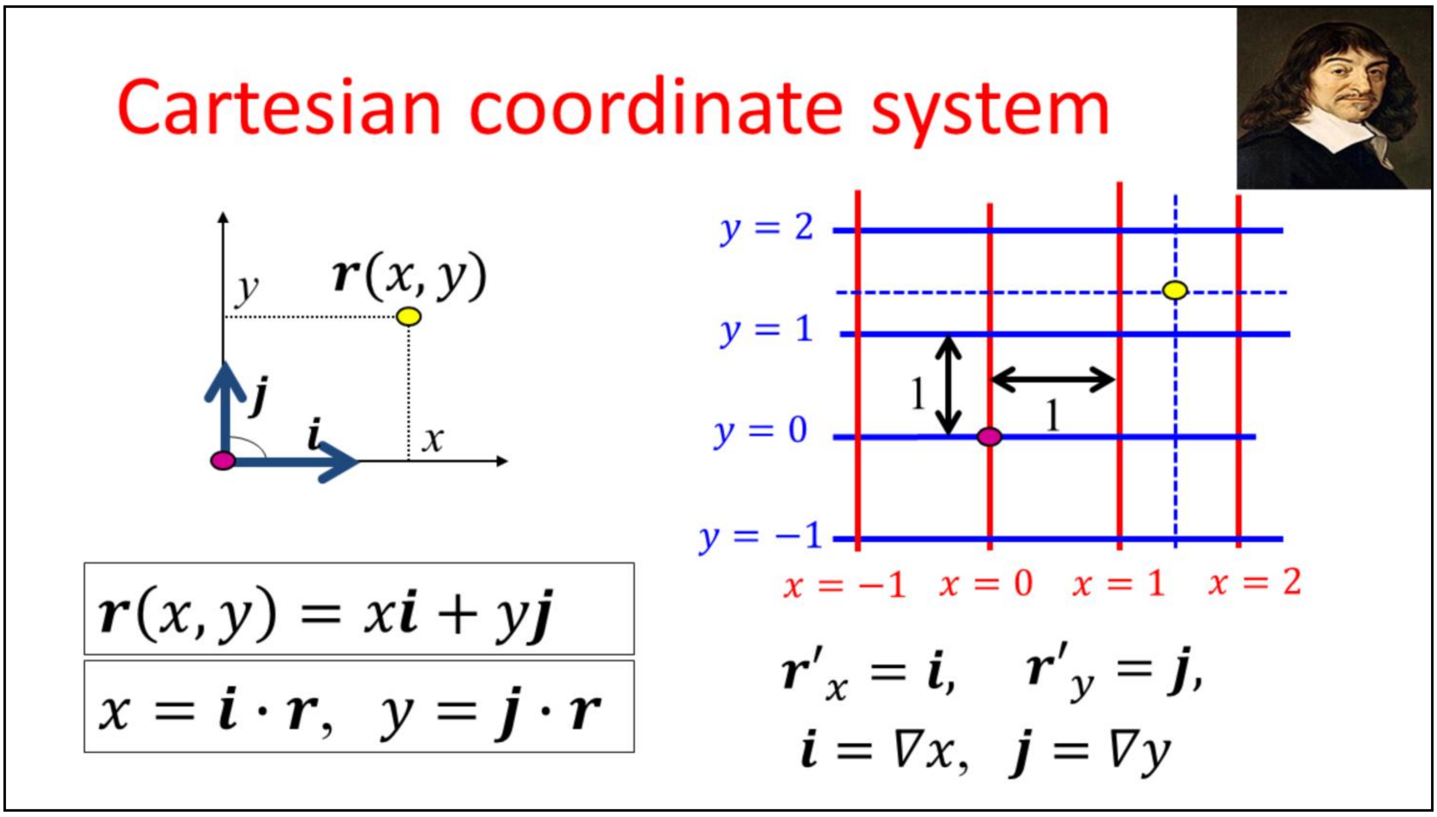
$\frac{\mathrm{d} f}{\mathrm{d}\boldsymbol{r}} = \nabla f$ is the direction and rate of the maximal increase

Derivation with respect to the parameter



Velocity (tangent) vector:

$$\boldsymbol{v} = \frac{\mathrm{d}\boldsymbol{r}(t)}{\mathrm{d}t} = \lim \frac{\boldsymbol{r}(t^* + \Delta t) - \boldsymbol{r}(t^*)}{\Delta t}$$


Cartesian coordinate system

$$\boxed{\boldsymbol{r}(x,y) = x\boldsymbol{i} + y\boldsymbol{j}}$$

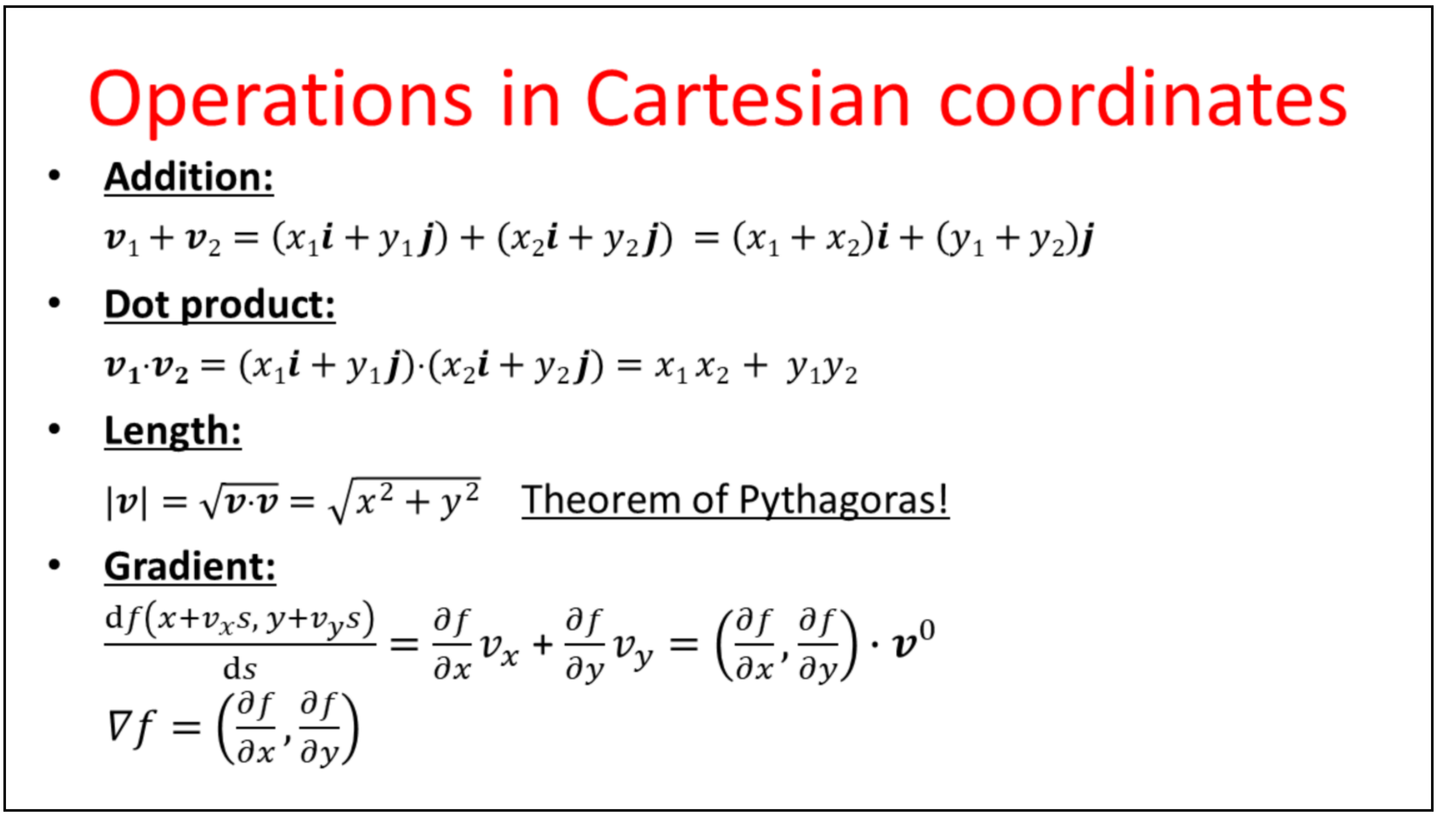
$$\boxed{x = \boldsymbol{i}\cdot\boldsymbol{r},\quad y = \boldsymbol{j}\cdot\boldsymbol{r}}$$

$$\boldsymbol{r}'_x = \boldsymbol{i},\quad \boldsymbol{r}'_y = \boldsymbol{j},$$

$$\boldsymbol{i} = \nabla x,\quad \boldsymbol{j} = \nabla y$$

Having vectors and operations, we are ready to establish a **Cartesian coordinate system**. Let us select one point of the plane and two unit (length) vectors ***i*** and ***j*** that are perpendicular to each other. A vector has unit length if its scalar product with itself is 1 and two vectors are perpendicular if their scalar product is zero since cos(90)=0 (formally:$\boldsymbol{i}\cdot\boldsymbol{i} = \boldsymbol{j}\cdot\boldsymbol{j} = 1$ and $\boldsymbol{i}\cdot\boldsymbol{j} = 0$).

Now, any position vector ***v*** can be unambiguously given as a linear combination of basis vector ***i*** and ***j***, i.e. in the form $\boldsymbol{v} = x\boldsymbol{i} + y\boldsymbol{j}$, where x and y are scalars, called the **coordinates**. Having ***v***, scalar products determine the appropriate coordinates: $x = \boldsymbol{v}\cdot\boldsymbol{i}$, $y = \boldsymbol{v}\cdot\boldsymbol{j}$. To prove this, let us multiply both sides of $\boldsymbol{v} = x\boldsymbol{i} + y\boldsymbol{j}$ by ***i*** and ***j***.



Operations in Cartesian coordinates

- **Addition:**

  $\boldsymbol{v}_1 + \boldsymbol{v}_2 = (x_1\boldsymbol{i} + y_1\boldsymbol{j}) + (x_2\boldsymbol{i} + y_2\boldsymbol{j}) \; = (x_1 + x_2)\boldsymbol{i} + (y_1 + y_2)\boldsymbol{j}$
- **Dot product:**

  $\boldsymbol{v_1}\cdot\boldsymbol{v_2} = (x_1\boldsymbol{i} + y_1\boldsymbol{j})\cdot(x_2\boldsymbol{i} + y_2\boldsymbol{j}) = x_1x_2 + \; y_1y_2$
- **Length:**

  $|\boldsymbol{v}| = \sqrt{\boldsymbol{v}\cdot\boldsymbol{v}} = \sqrt{x^2 + y^2}$ Theorem of Pythagoras!
- **Gradient:**

  $$\frac{\mathrm{d}f(x+v_xs,\, y+v_ys)}{\mathrm{d}s} = \frac{\partial f}{\partial x}v_x + \frac{\partial f}{\partial y}v_y = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)\cdot\boldsymbol{v}^0$$

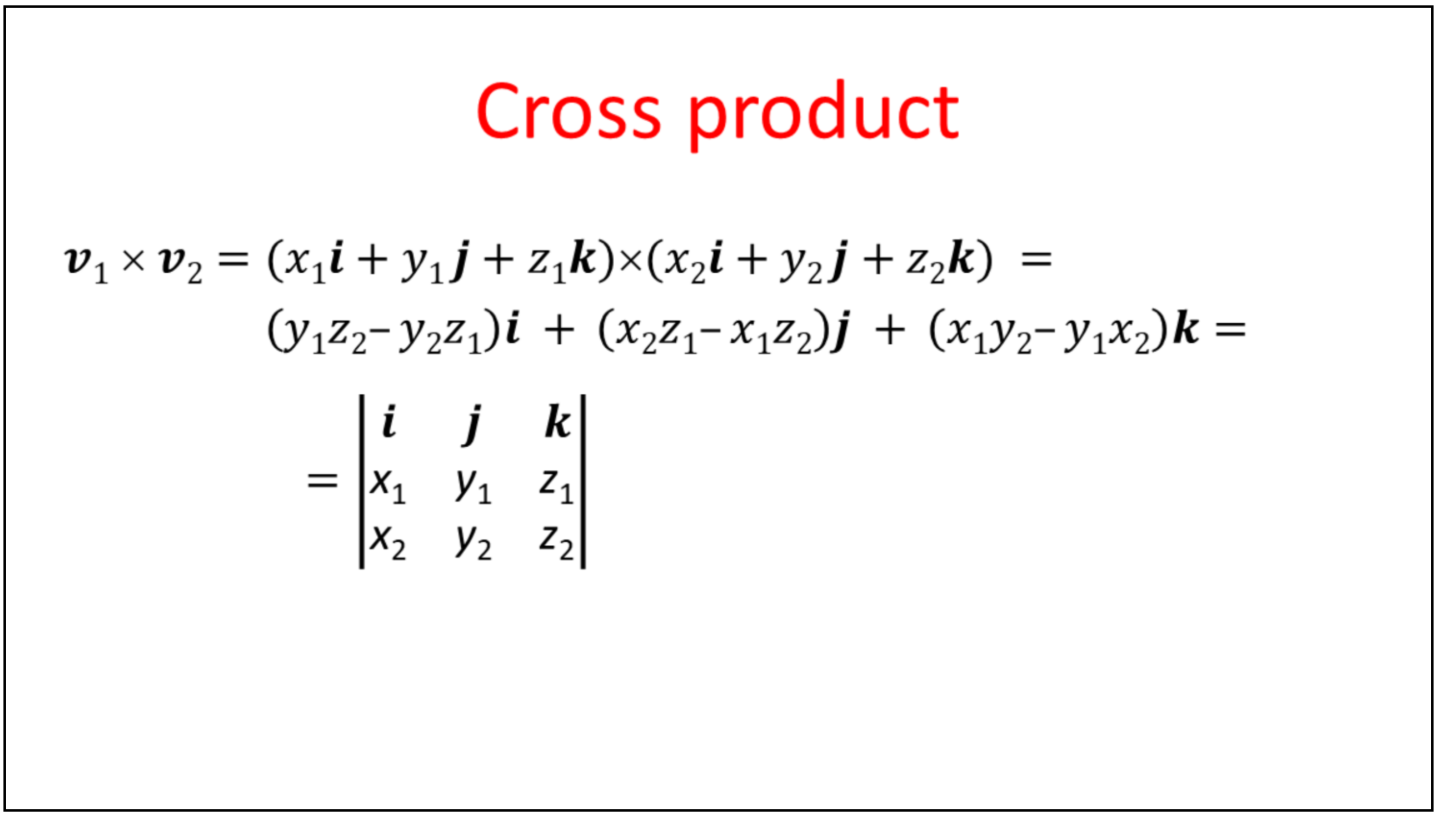
  $$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

As there is a one-to-one correspondence between vectors and coordinate pairs in 2D (and coordinate triplets in 3D), vectors can be represented by coordinates in all operations.

Based on the associative property of vector addition and on distributive property of multiplying a vector by a scalar with addition, we can prove that coordinates of the sum of two vectors are the sums of the respective coordinates of the two vectors.

Similarly, based on the distributive property of dot product with vector addition, we can prove that the dot product equals to the sum of the products of respective coordinates. Here we also exploit that $\boldsymbol{i}\cdot\boldsymbol{i} = \boldsymbol{j}\cdot\boldsymbol{j} = 1$ and $\boldsymbol{i}\cdot\boldsymbol{j} = 0$.

The absolute value of a vector is the square root of the scalar product of the vector with itself. Note that we get the Pythagoras theorem for free.



Cross product

$$\boldsymbol{v}_1 \times \boldsymbol{v}_2 = (x_1\boldsymbol{i} + y_1\boldsymbol{j} + z_1\boldsymbol{k})\times(x_2\boldsymbol{i} + y_2\boldsymbol{j} + z_2\boldsymbol{k}) =$$

$$(y_1z_2 - y_2z_1)\boldsymbol{i} + (x_2z_1 - x_1z_2)\boldsymbol{j} + (x_1y_2 - y_1x_2)\boldsymbol{k} =$$

$$= \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{vmatrix}$$

Finally, based on the distributive property of the cross product with vector addition, we can also express the cross product of two vector with their coordinates. We should also use the cross products of the base vectors $\boldsymbol{i} \times \boldsymbol{i} = 0$, $\boldsymbol{i} \times \boldsymbol{j} = \boldsymbol{k}$, etc. The result can be memorized as a determinant where the first row contains the three basis vectors, the second the coordinates of the first operand, the third the coordinates of the second operand.

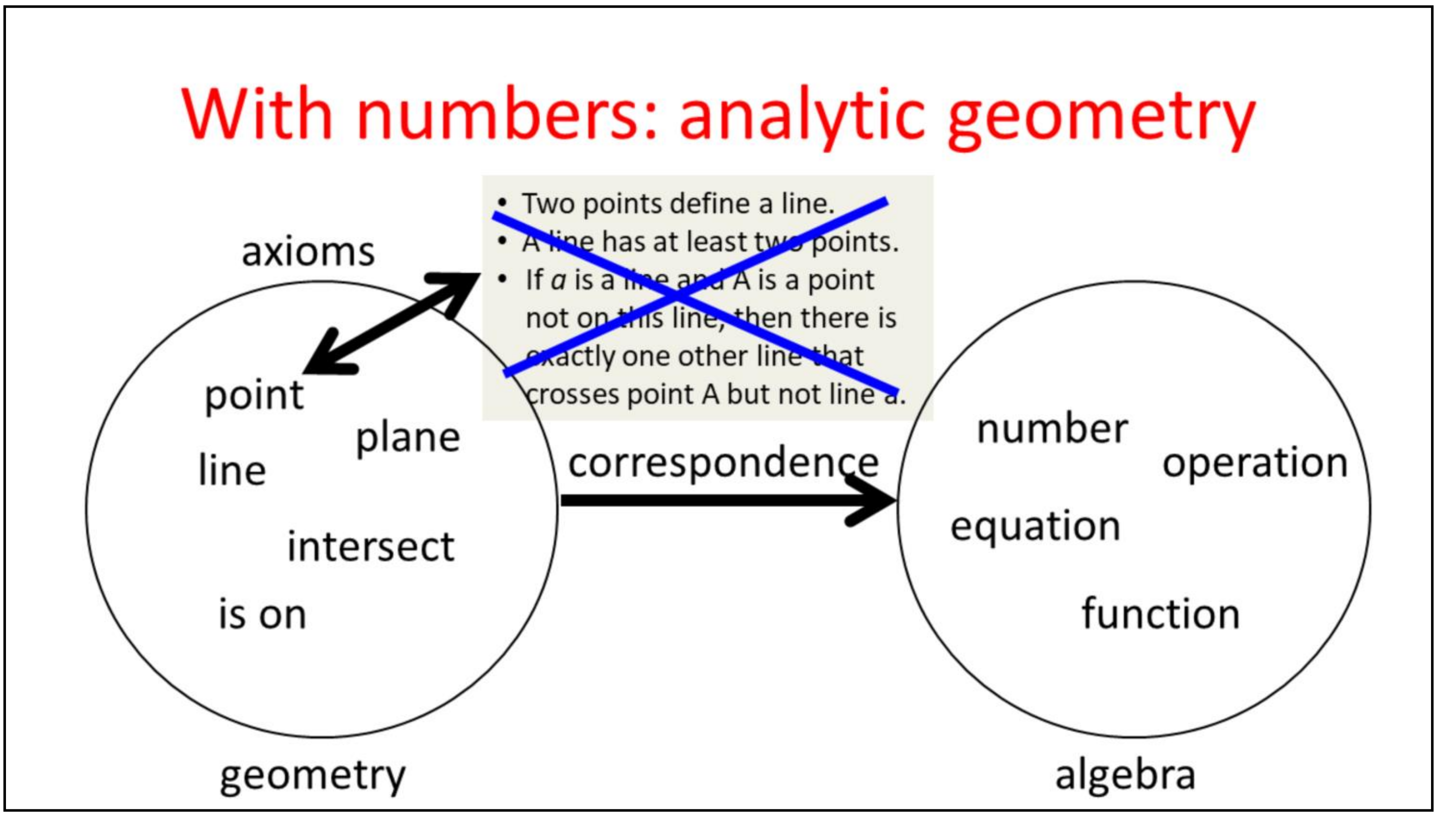
The absolute value of a vector is the square root of the scalar product of the vector with itself. Note that we get the Pythagoras theorem for free.

"Algebra is the offer made by the devil: Give me your soul, give up geometry and you will have this marvelous machine."

Michael Francis Atiyah

Geometries and algebras
4. Analytic geometry

Szirmay-Kalos László



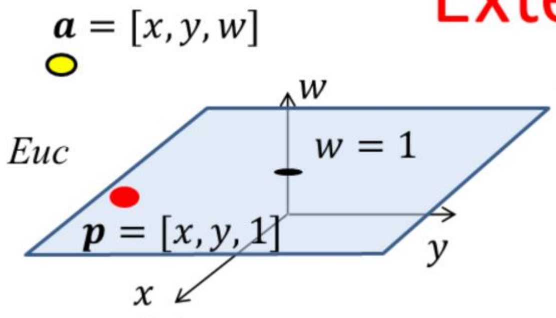
With numbers: analytic geometry

- ~~Two points define a line.~~
- ~~A line has at least two points.~~
- ~~If *a* is a line and A is a point not on this line, then there is exactly one other line that crosses point A but not line *a*.~~

In computer graphics, we should also take into account that a computer is programmed, which cannot do anything else but making calculations with numbers. A computer is definitely not able to understand abstract concepts like point, line etc. So for the application of a computer, geometric concepts must be translated to numbers, calculation and algebra.

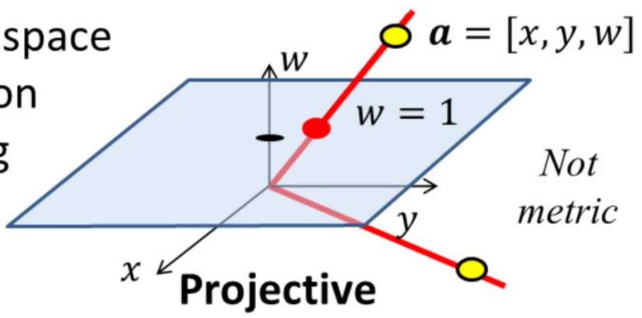
A geometry based on algebra, equations and numbers is called **analytic geometry** or **coordinate geometry**. To establish an analytic version of a geometry, we have to find correspondences between geometric concepts and concepts of algebra in a way that axioms of the geometry will not contradict to the concepts of algebra. If it is done, we can forget the original axioms and work only with numbers and equations.

External view

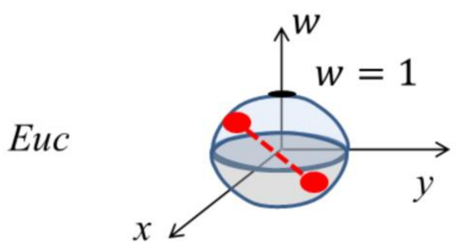


Euclidean: $w = 1$

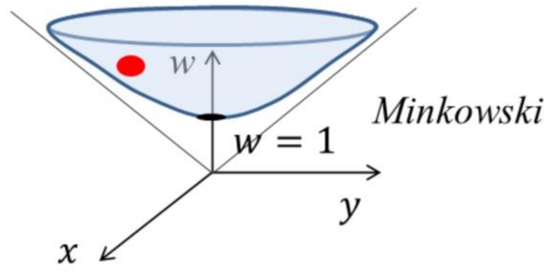
Ambient space
- addition
- scaling



Projective

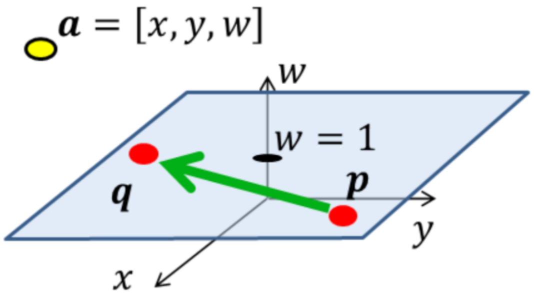


Elliptic: $x^2 + y^2 + w^2 = 1$



Hyperbolic: $x^2 + y^2 - w^2 = -1$

Euclidean planar geometry



Dot product in ambient space:

$$\boxed{\boldsymbol{a}_1 \cdot \boldsymbol{a}_2 = x_1 x_2 + y_1 y_2 + w_1 w_2}$$

- Commutative: $\boldsymbol{a} \cdot \boldsymbol{b} = \boldsymbol{b} \cdot \boldsymbol{a}$
- Distributive: $\boldsymbol{a} \cdot (\boldsymbol{b} + \boldsymbol{c}) = \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \cdot \boldsymbol{c}$
- Scaling: $(s\boldsymbol{a}) \cdot \boldsymbol{b} = s(\boldsymbol{a} \cdot \boldsymbol{b})$

Elements of ambient space:

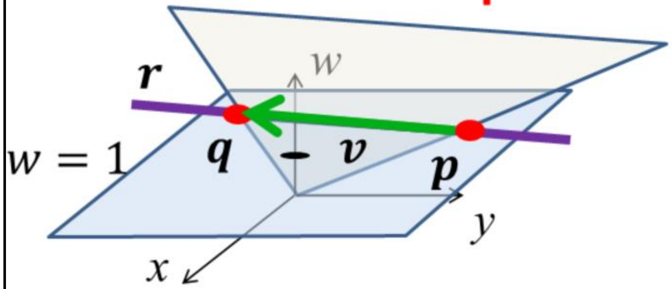
- **Points:** $\boxed{\boldsymbol{p} = [x, y, 1]}$

  Why not $w = 0$? *Translation would not be linear*
- **Vectors:** $\boldsymbol{v} = \boldsymbol{q} - \boldsymbol{p}$, $\boxed{\boldsymbol{v} = [x, y, 0]}$
- Other w values: neither points not vectors

Properties of vectors

- Separation of two points: $\boldsymbol{v} = \boldsymbol{q} - \boldsymbol{p}$
- Element of ambient space: $\boldsymbol{v} = [x, y, 0]$
- Length: $|\boldsymbol{v}| = \sqrt{\boldsymbol{v} \cdot \boldsymbol{v}} = \sqrt{x^2 + y^2}$
- Unit vector: $|\boldsymbol{v}^0| = 1,\ \boldsymbol{v}^0 \cdot \boldsymbol{v}^0 = 1, \boldsymbol{v}^0 = \frac{\boldsymbol{v}}{\sqrt{\boldsymbol{v} \cdot \boldsymbol{v}}}$
- Orthogonality: $\boldsymbol{v}_\perp \cdot \boldsymbol{v} = 0,\ x_\perp x + y_\perp y = 0,$
  $\boldsymbol{v}_\perp = [-y, x, 0]\lambda$
- Parallelism: $\boldsymbol{v}_\parallel = \lambda \boldsymbol{v},\ \ \boldsymbol{v}_\parallel = [x, y, 0]\lambda$

Lines: parametric equation



Constant speed motion:

$$\boxed{\boldsymbol{r}(t) = \boldsymbol{p} + \boldsymbol{v}t}$$

Velocity: $\dot{\boldsymbol{r}}(t) = \boldsymbol{v}$

Acceleration: $\ddot{\boldsymbol{r}}(t) = 0$

Zero curvature

- Constant speed motion with coordinates:

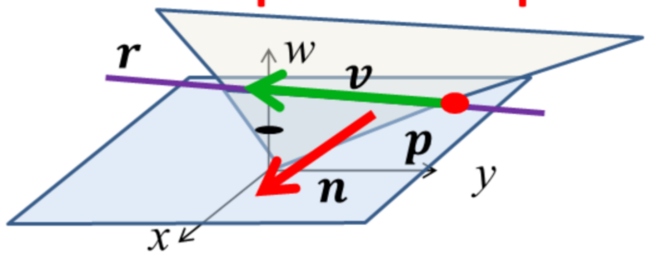
$$[x(t), y(t), w(t)] = [p_x, p_y, 1] + [v_x, v_y, 0]t$$

- Combination of two points: $\boldsymbol{r}(t) = \boldsymbol{p} + (\boldsymbol{q} - \boldsymbol{p})t = \boldsymbol{p}(1-t) + \boldsymbol{q}t$

$$[x(t), y(t), w(t)] = [p_x, p_y, 1](1-t) + [q_x, q_y, 1]t$$

- Intersection of the plane (origin, $\boldsymbol{p}, \boldsymbol{q}$) and the plane of $w = 1$

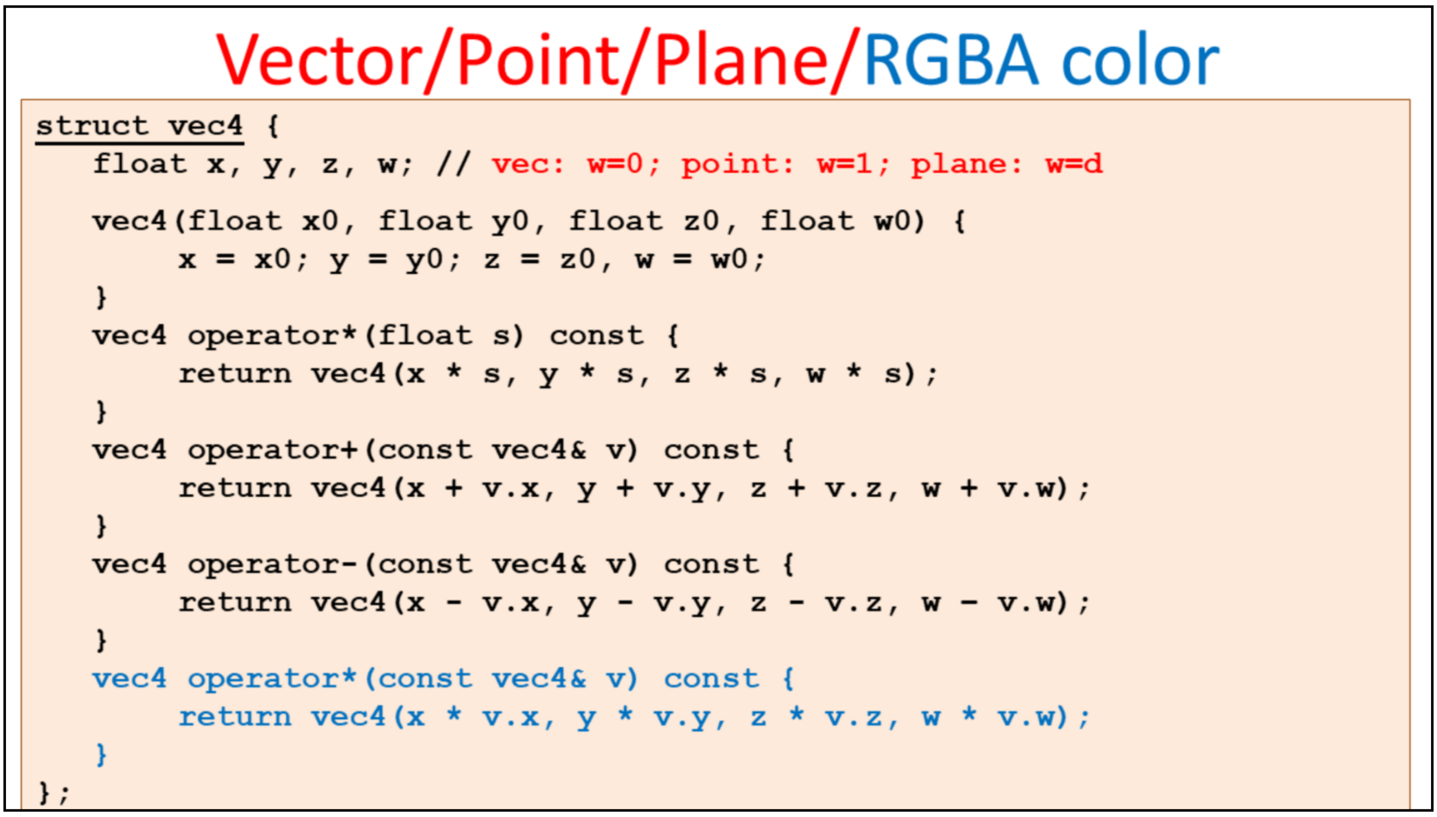
Lines: implicit equation



- **Implicit equation**: $\boldsymbol{n} = \boldsymbol{v}_\perp \Rightarrow \boldsymbol{n} \cdot \boldsymbol{v} = 0 \Rightarrow \boldsymbol{n} \cdot (\boldsymbol{r} - \boldsymbol{p}) = 0$
  $[n_x, n_y, 0] \cdot [x - p_x, y - p_y, 0] = \boxed{n_x x + n_y y + d = 0}$
  where $d = -\boldsymbol{n} \cdot \boldsymbol{p} = -n_x p_x - n_y p_y$
- Element of the ambient space: $\boldsymbol{N} = [n_x, n_y, d]$
  $\boldsymbol{N} \cdot \boldsymbol{r} = [n_x, n_y, d] \cdot [x, y, w] = n_x\, x + n_y y + dw = 0,$
  where $\boldsymbol{r}$ is point, i.e. $w = 1$. Line = intersection of two planes.
- Two lines are the same if $\boldsymbol{N}_1 = \boldsymbol{N}_2 \lambda$

Euclidean geometry of the 3D space

- Ambient space is 4D: $\boldsymbol{a} = [x, y, z, w]$
- Dot product: $\boldsymbol{a}_1 \cdot \boldsymbol{a}_2 = x_1 x_2 + y_1 y_2 + z_1 z_2 + w_1\, w_2$
- Points: $\boldsymbol{p} = [x, y, z, 1]$
  - Distance: $d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{(\boldsymbol{q} - \boldsymbol{p}) \cdot (\boldsymbol{q} - \boldsymbol{p})}$
  - Angle: $\alpha(\boldsymbol{u}, \boldsymbol{v}) = \arccos\left(\frac{\boldsymbol{u} \cdot \boldsymbol{v}}{|\boldsymbol{u}||\boldsymbol{v}|}\right)$
- Vectors: $\boldsymbol{v} = [x, y, z, 0]$
  - Lengths: $|\boldsymbol{v}| = \sqrt{\boldsymbol{v} \cdot \boldsymbol{v}} = \sqrt{x^2 + y^2 + z^2}$
- Planes in 3D are like the lines in 2D geometry



Vector/Point/Plane/RGBA color

```
struct vec4 {
   float x, y, z, w; // vec: w=0; point: w=1; plane: w=d

   vec4(float x0, float y0, float z0, float w0) {
        x = x0; y = y0; z = z0, w = w0;
   }
   vec4 operator*(float s) const {
        return vec4(x * s, y * s, z * s, w * s);
   }
   vec4 operator+(const vec4& v) const {
        return vec4(x + v.x, y + v.y, z + v.z, w + v.w);
   }
   vec4 operator-(const vec4& v) const {
        return vec4(x - v.x, y - v.y, z - v.z, w - v.w);
   }
   vec4 operator*(const vec4& v) const {
        return vec4(x * v.x, y * v.y, z * v.z, w * v.w);
   }
};
```

Using vec4, i.e. four-element vectors, instead of vec3, we can represent not only points and vectors, but also planes. The fourth element should be set depending on the actual type.

Note that the test whether or not a point is on a plane is a dot product of the two 4-element vectors if the fourth component of a point is 1 and the four components of a plane is a, b, c, d, the plane parameters.

We can preserve the validity of vector operations if the fourth element of a vector is zero.

And now, the big news: those operations that are applicable for points remain valid even when the fourth element is 1.

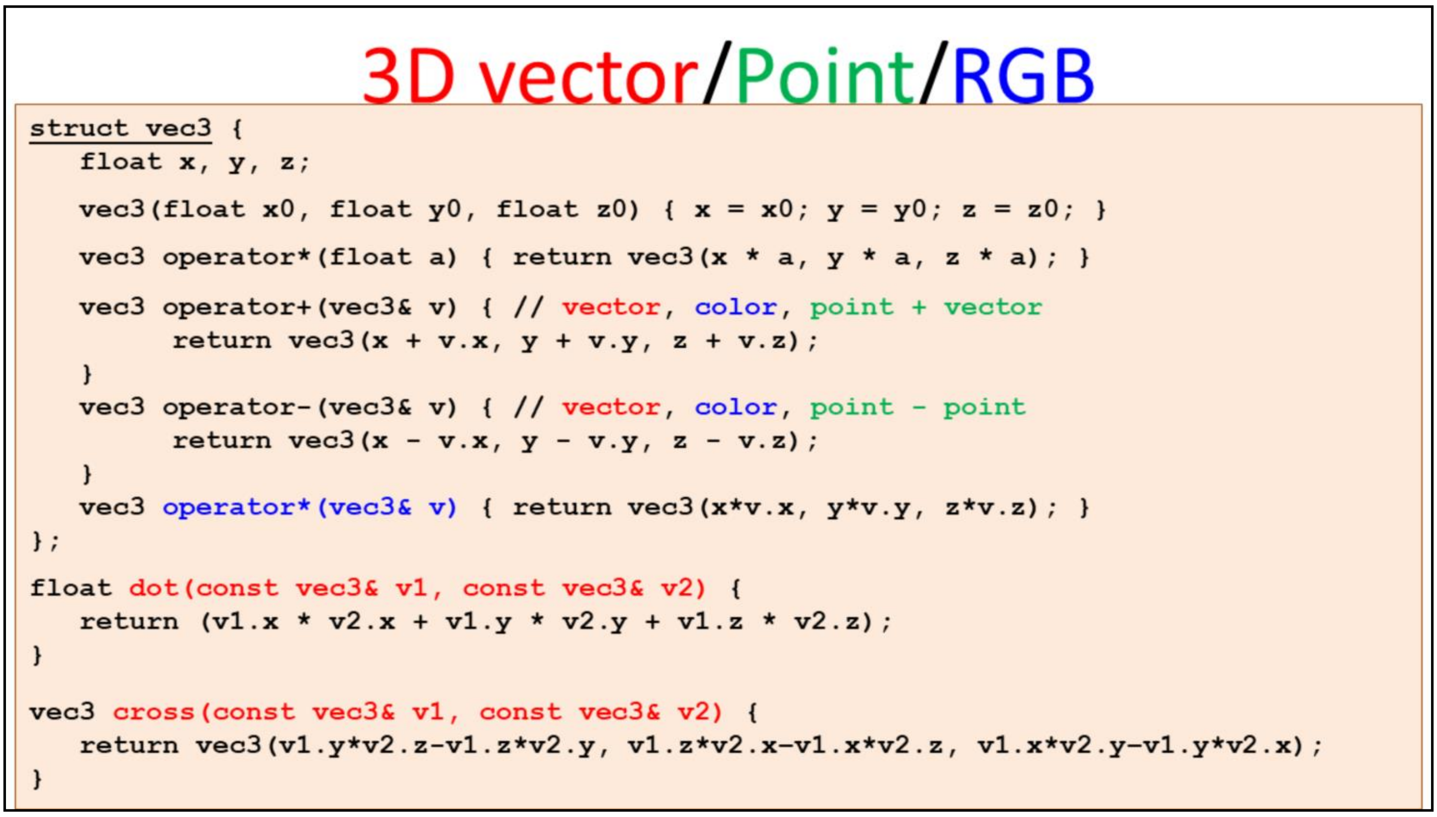
vec4 operations

```
float dot(const vec4& a, const vec4& b) {
   return a.x * b.x + a.y * b.y + a.z * b.z + a.w * b.w;
}

float length(const vec4& v) {
   return sqrtf(dot(v, v));
}

vec4 normalize(const vec4& v) {
   return v * (1/length(v));
}

vec4 lerp(const vec4& p, const vec4& q, float t) {
   return p * (1 - t) + q * t;
}
```



3D vector/Point/RGB

```
struct vec3 {
   float x, y, z;

   vec3(float x0, float y0, float z0) { x = x0; y = y0; z = z0; }

   vec3 operator*(float a) { return vec3(x * a, y * a, z * a); }

   vec3 operator+(vec3& v) { // vector, color, point + vector
        return vec3(x + v.x, y + v.y, z + v.z);
   }
   vec3 operator-(vec3& v) { // vector, color, point - point
        return vec3(x - v.x, y - v.y, z - v.z);
   }
   vec3 operator*(vec3& v) { return vec3(x*v.x, y*v.y, z*v.z); }
};

float dot(const vec3& v1, const vec3& v2) {
   return (v1.x * v2.x + v1.y * v2.y + v1.z * v2.z);
}

vec3 cross(const vec3& v1, const vec3& v2) {
   return vec3(v1.y*v2.z-v1.z*v2.y, v1.z*v2.x-v1.x*v2.z, v1.x*v2.y-v1.y*v2.x);
}
```

The implementation of the theory discussed so far is a single C++ class representing a 3D point or a vector with three Cartesian coordinates. Using operator overloading, the discussed vector (and point) operations are also.

Note that – similarly to the GLSL language – we use the same type to represent points and vectors. The programmer should be aware whether an object is a point or a vector and execute operations valid for this particular type. Mixing different types in a single class, we can extend the concept to colors as well. A color can be define as additive mixture of red, green, and blue components, so a three-dimensional vector is just a right representation.

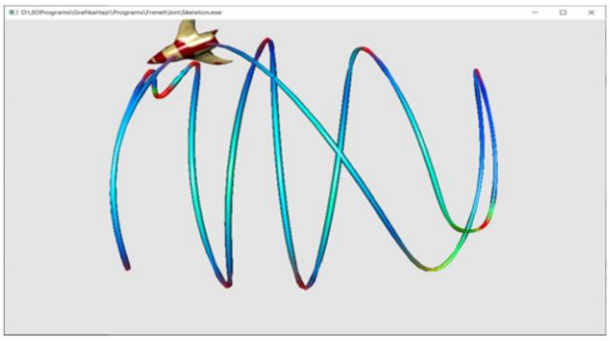
Analytic Euclidean geometry

- Numbers only please
- Coordinates
  - External view, +1 coordinate
  - Ambient space, +1 coordinate (point=1, vector=0)
  - Metric (distance, angle) = dot product
- Line:
  - Unit speed motion
  - Shortest path (geodesic)
- Implementation: vec4
  - Can be vec3, vec2 if we keep the missing coordinates in mind

"Navigare necesse est, vivere non est necesse."
Cnaeus Pompeius Magnus

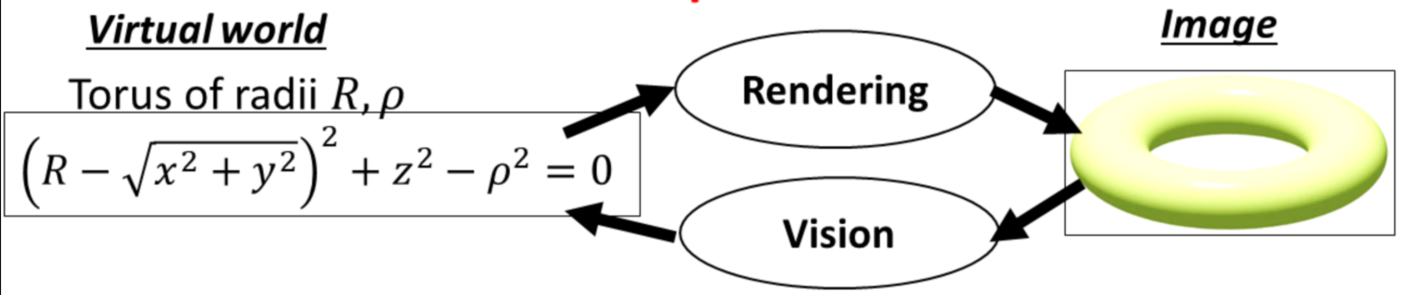
Automatic derivation

Szirmay-Kalos László

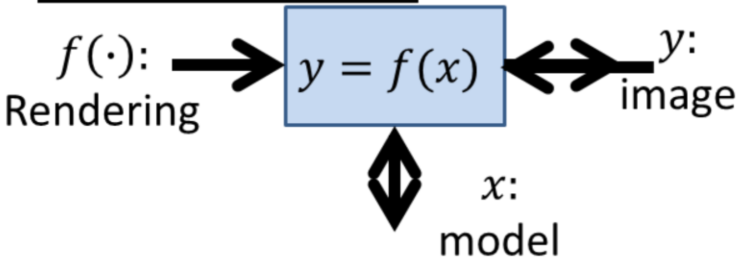


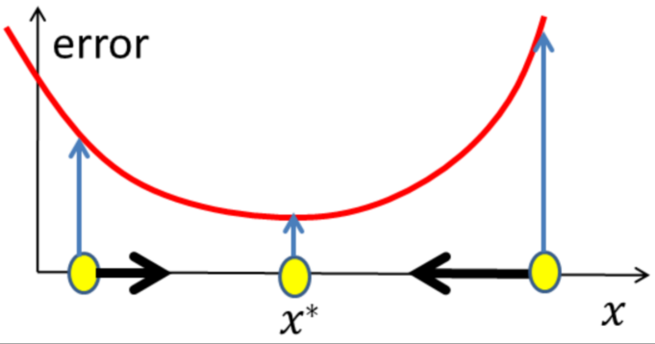
Example: shading, normal vector

Inverse problems



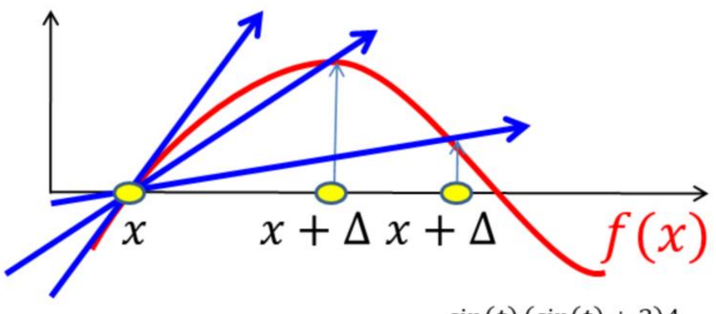
Function inverse





How **Not** to derivate!

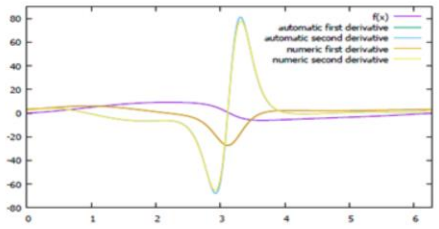
$$f'(x) \approx \frac{f(x+\Delta) - f(x)}{\Delta}$$



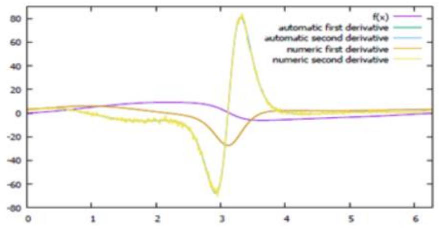
Subtraction: 💣☠✝

1234.569?????	7 valuable digits
−1234.567?????	7 valuable digits
0.002?????	1 valuable digits

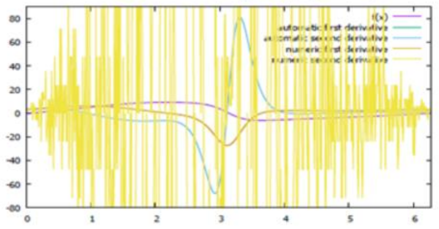
$$\frac{\sin(t)\,(\sin(t)+3)4}{\tan(\cos(t)+2)}$$



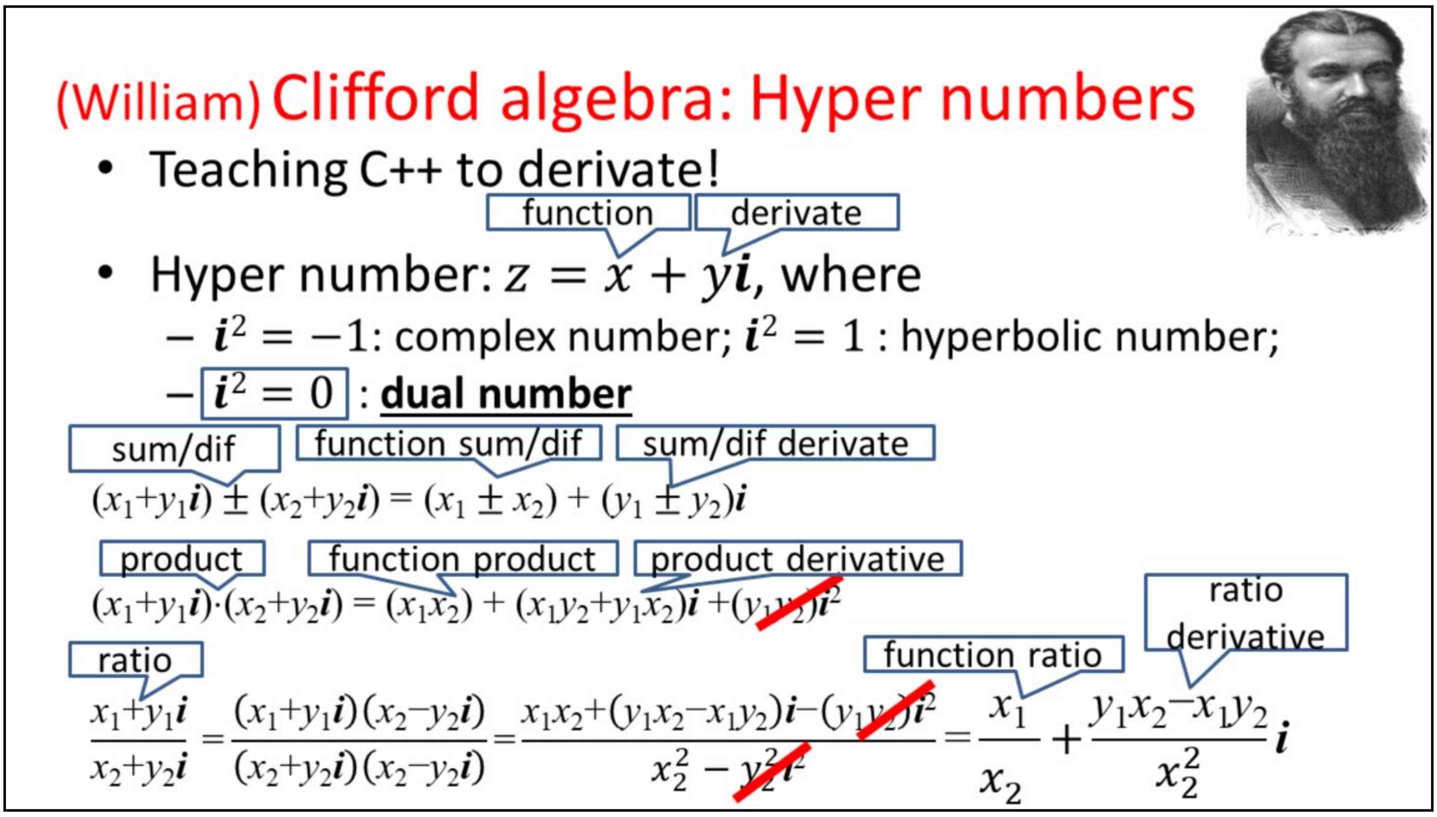
$\Delta = 0.1$



$\Delta = 0.001$



$\Delta = 0.0001$



(William) Clifford algebra: Hyper numbers

- Teaching C++ to derivate!
- Hyper number: $z = x + y\boldsymbol{i}$, where
  - $\boldsymbol{i}^2 = -1$: complex number; $\boldsymbol{i}^2 = 1$: hyperbolic number;
  - $\boxed{\boldsymbol{i}^2 = 0}$: **dual number**

function, derivate

sum/dif, function sum/dif, sum/dif derivate

$$(x_1+y_1\boldsymbol{i}) \pm (x_2+y_2\boldsymbol{i}) = (x_1 \pm x_2) + (y_1 \pm y_2)\boldsymbol{i}$$

product, function product, product derivative

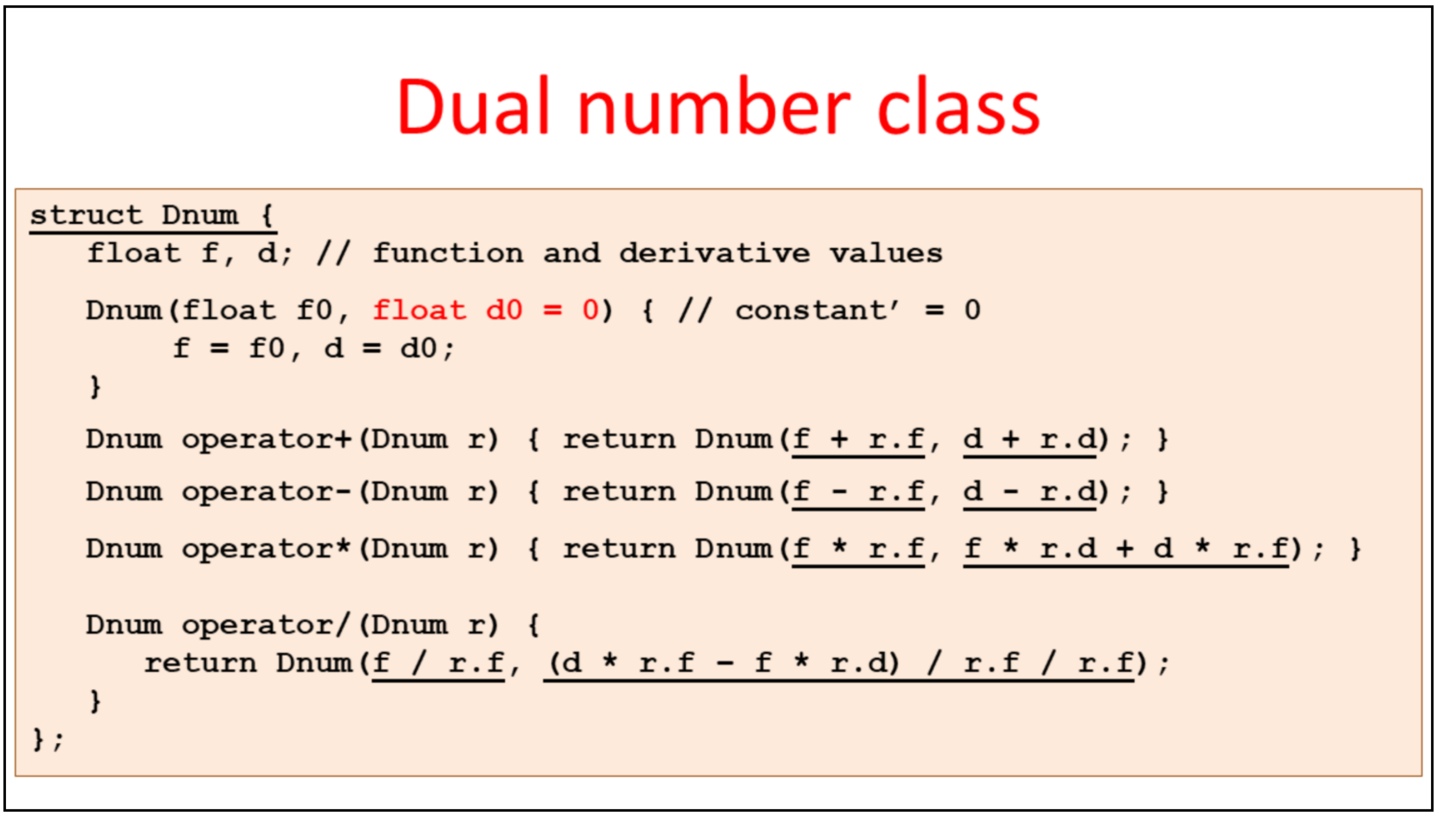
$$(x_1+y_1\boldsymbol{i})\cdot(x_2+y_2\boldsymbol{i}) = (x_1x_2) + (x_1y_2+y_1x_2)\boldsymbol{i} + \cancel{(y_1y_2)}\boldsymbol{i}^2$$

ratio, function ratio, ratio derivative

$$\frac{x_1+y_1\boldsymbol{i}}{x_2+y_2\boldsymbol{i}} = \frac{(x_1+y_1\boldsymbol{i})(x_2-y_2\boldsymbol{i})}{(x_2+y_2\boldsymbol{i})(x_2-y_2\boldsymbol{i})} = \frac{x_1x_2+(y_1x_2-x_1y_2)\boldsymbol{i}-\cancel{(y_1y_2)}\boldsymbol{i}^2}{x_2^2 - \cancel{y_2^2\boldsymbol{i}^2}} = \frac{x_1}{x_2} + \frac{y_1x_2-x_1y_2}{x_2^2}\boldsymbol{i}$$

The set of complex numbers is a special case of hyper numbers of Clifford algebras that deal with numbers of form x+yi where x and y are real numbers and i is a symbol, and the same arithmetic rules apply for such numbers as for real ones. To use the arithmetic rules, we should decide what is the value of i^2. Complex numbers define it as -1, in Clifford algebra, this is up to us.

If we take the option of i^2=0, then we have dual numbers and the rules will be equivalent to the rules of derivatives assuming x the function value and y the derivative. Thus, we can derivate without approximations specifying only the function, the derivative is obtained automatically.



Dual number class

```
struct Dnum {
   float f, d; // function and derivative values

   Dnum(float f0, float d0 = 0) { // constant' = 0
       f = f0, d = d0;
   }

   Dnum operator+(Dnum r) { return Dnum(f + r.f, d + r.d); }

   Dnum operator-(Dnum r) { return Dnum(f - r.f, d - r.d); }

   Dnum operator*(Dnum r) { return Dnum(f * r.f, f * r.d + d * r.f); }

   Dnum operator/(Dnum r) {
      return Dnum(f / r.f, (d * r.f - f * r.d) / r.f / r.f);
   }
};
```

Dnum implements the arithmetic rules of derivation.

Application of dual numbers

- Without derivative:

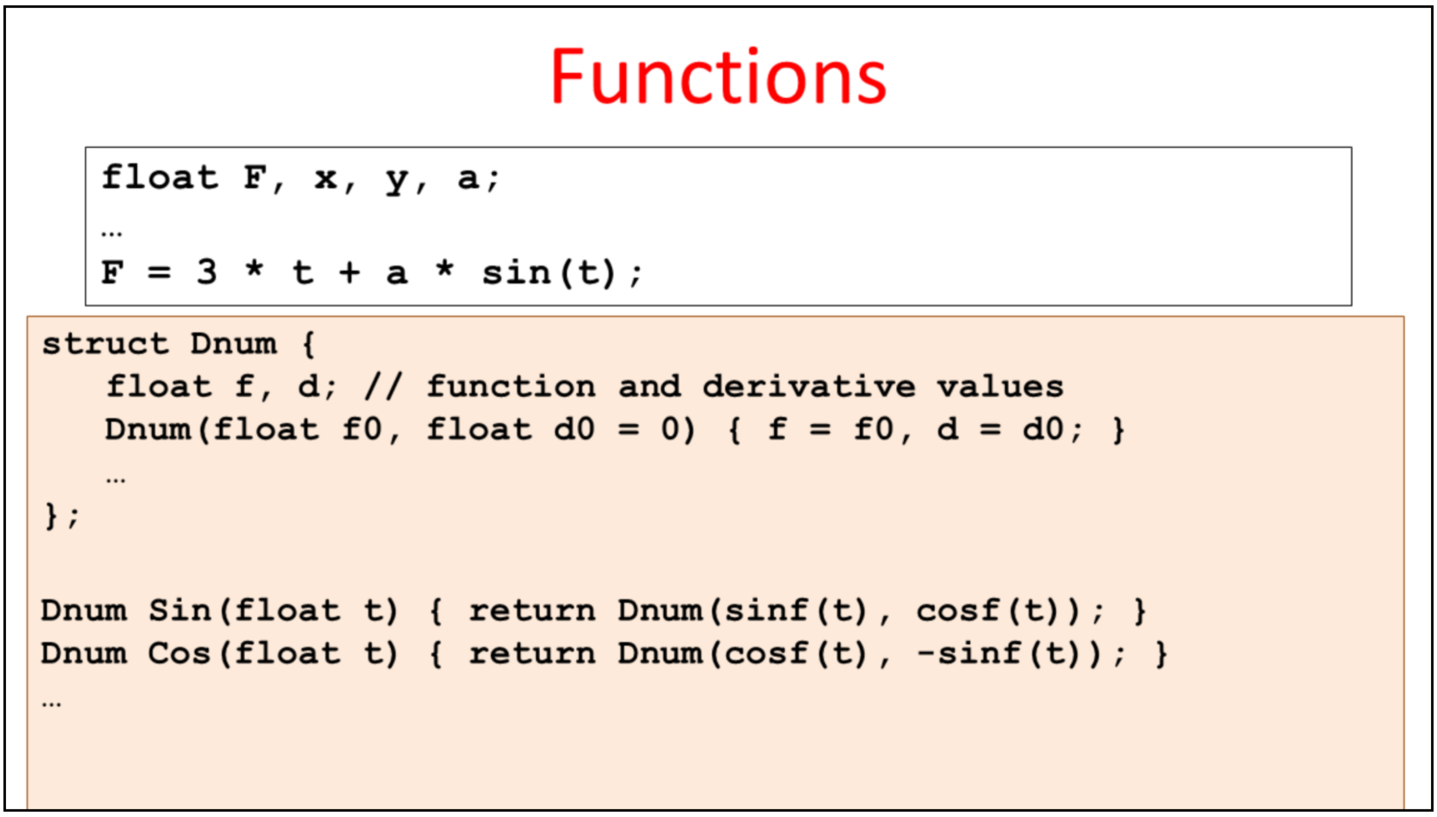
```
float t = value;
float F = t * a / (t * t + b);
```

- With derivative:

```
Dnum F = Dnum(t,1) * Dnum(a,0) /
         (Dnum(t,1) * Dnum(t,1) + Dnum(b,0));
```

- With derivative and more elegantly, exploiting the default parameterization in the constructor:

```
Dnum t(value, 1);
Dnum F = t * a / (t * t + b);
```



Functions

```
float F, x, y, a;
…
F = 3 * t + a * sin(t);
```

```
struct Dnum {
   float f, d; // function and derivative values
   Dnum(float f0, float d0 = 0) { f = f0, d = d0; }
   …
};

Dnum Sin(float t) { return Dnum(sinf(t), cosf(t)); }
Dnum Cos(float t) { return Dnum(cosf(t), -sinf(t)); }
…
```

We also need global functions to specify the derivatives of elementary functions.

Composite functions

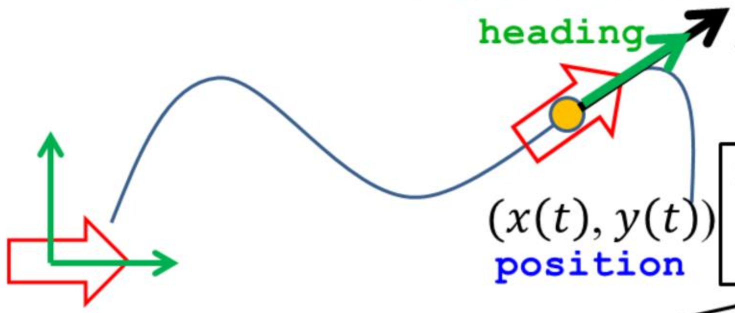
```
float F, x, y, a;
…
F = 3 * t + a * sin(t) + cos(y * log(t) + 2);
```

$$\frac{\mathrm{d}f(g(t))}{\mathrm{d}t} = \frac{\mathrm{d}f}{\mathrm{d}g} \cdot \frac{\mathrm{d}g}{\mathrm{d}t}$$

```
struct Dnum {
   float f, d; // function and derivative values
   Dnum(float f0, float d0 = 0) { f = f0, d = d0; }
   …
};

Dnum Sin(Dnum g) { return Dnum(sinf(g.f), cosf(g.f) * g.d); }
Dnum Cos(Dnum g) { return Dnum(cosf(g.f), -sinf(g.f) * g.d); }
Dnum Tan(Dnum g) { return Sin(g)/Cos(g); }
Dnum Log(Dnum g) { return Dnum(logf(g.f), 1/g.f * g.d); }
Dnum Exp(Dnum g) { return Dnum(expf(g.f), expf(g.f) * g.d); }
Dnum Pow(Dnum g, float n) {
    return Dnum(powf(g.f n), n * powf(g.f,n-1) * g.d);
}
```

Path animation



$$v(t) = (\dot{x}(t), \dot{y}(t))$$

$$\frac{48\sin^2(x)\sin^2(\cos(x)+2)}{(\cos(2(\cos(x)+2))-1)^2} + \frac{16\sin^3(x)\sin^2(\cos(x)+2)}{(\cos(2(\cos(x)+2))-1)^2} - \frac{8\sin(x)\cos(x)\sin(2(\cos(x)+2))}{\cos(2(\cos(x)+2))-1} - \frac{12\cos(x)\sin(2(\cos(x)+2))}{\cos(2(\cos(x)+2))-1}$$

Pálya: $x(t) = \frac{\sin(t)(\sin(t)+3)4}{\tan(\cos(t)+2)}, \quad y(t) = \frac{(\cos(\sin(t))8+1)12+2}{(\sin(t)\sin(t))^3+2}$

```
Dnum t(tt, 1);
Dnum x = Sin(t)*(Sin(t)+3)*4 / (Tan(Cos(t))+2);
Dnum y = (Cos(Sin(t)*8+1)*12+2)/(Pow(Sin(t)*Sin(t),3)+2);
vec2 position(x.f, y.f), velocity(x.d, y.d);
vec2 heading = normalize(velocity);
Draw(position, heading);
```

Suppose we want to animate an object along a path defined by (x(t), y(t)). The position of the object is given by inserting time t into these two functions.

Objects like trains, cars, birds, etc. follow their front (head, beak) during animation, so we should often rotate the object to transform its heading direction into the current direction of movement, which is the current velocity, i.e. the derivative of the path.