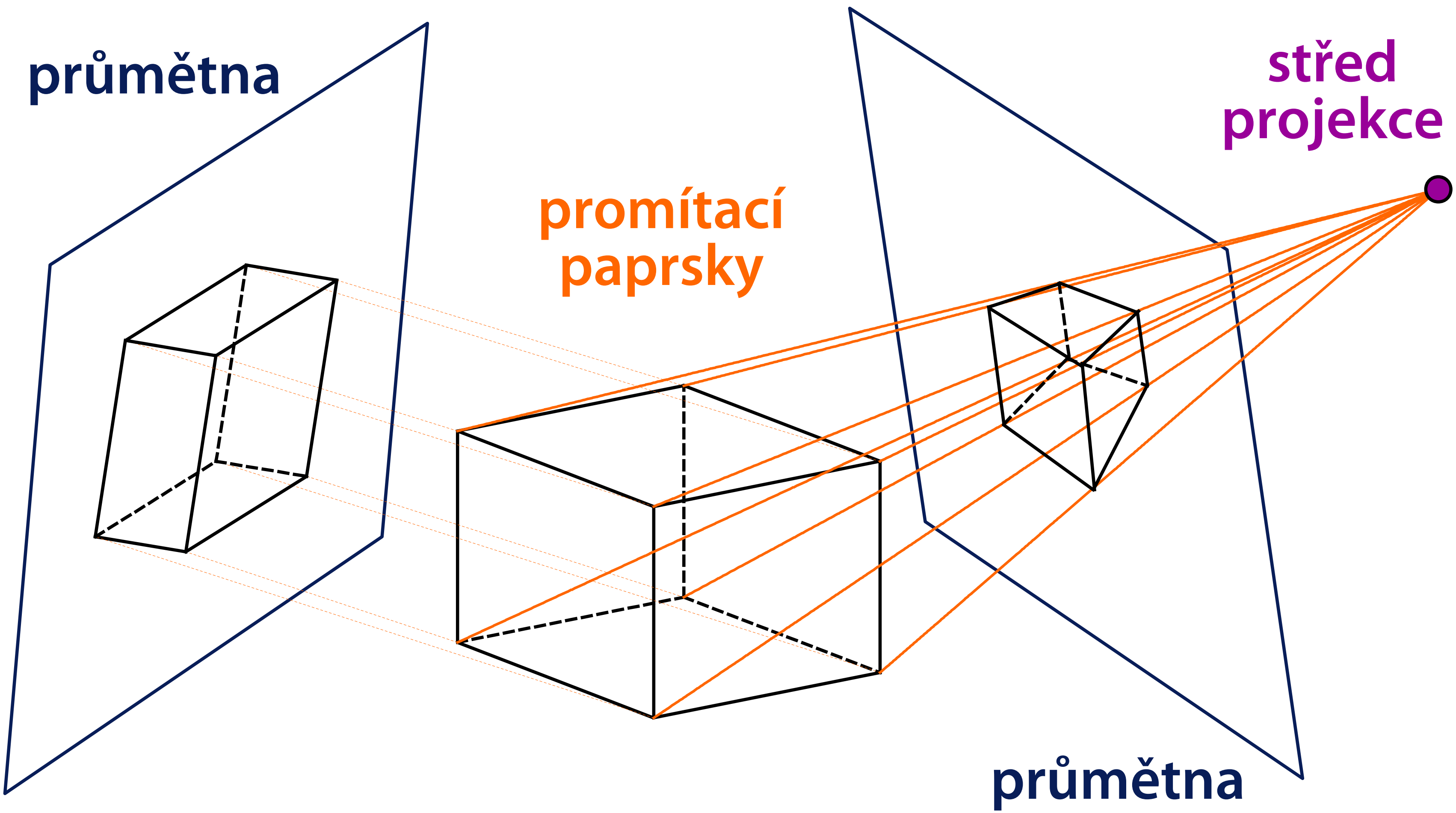
Zobrazovací projekce

© 1995-2019 Josef Pelikán
CGG MFF UK Praha

pepca@cgg.mff.cuni.cz
https://cgg.mff.cuni.cz/~pepca/

Základní pojmy



Klasifikace lineárních projekcí

Rovnoběžné projekce

- promítací paprsky jsou navzájem rovnoběžné

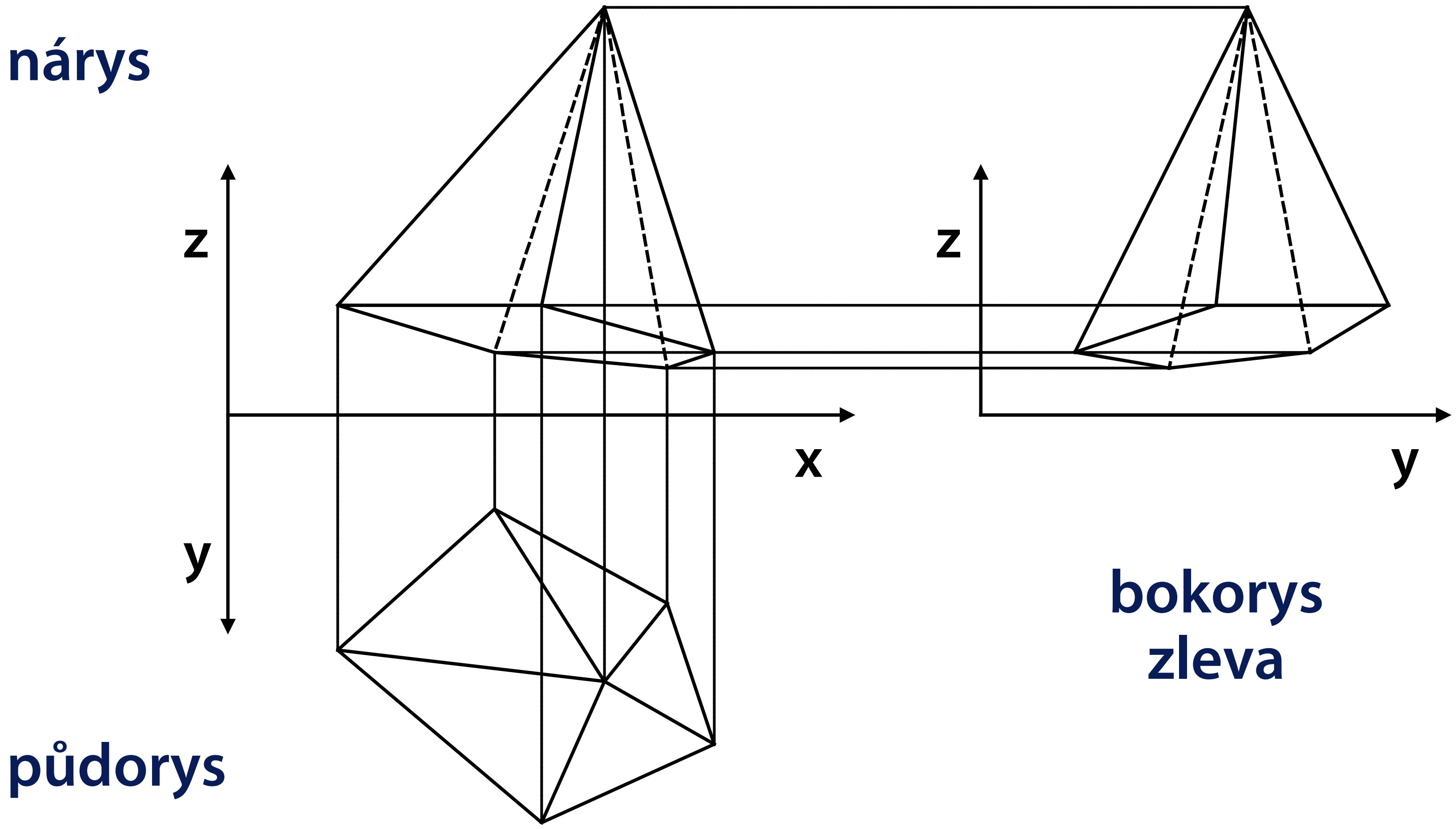
Kolmé projekce

- promítací paprsky jsou kolmé na průmětnu
- Mongeova projekce, půdorys, nárys, bokorys
- axonometrie (obecná kolmá projekce)

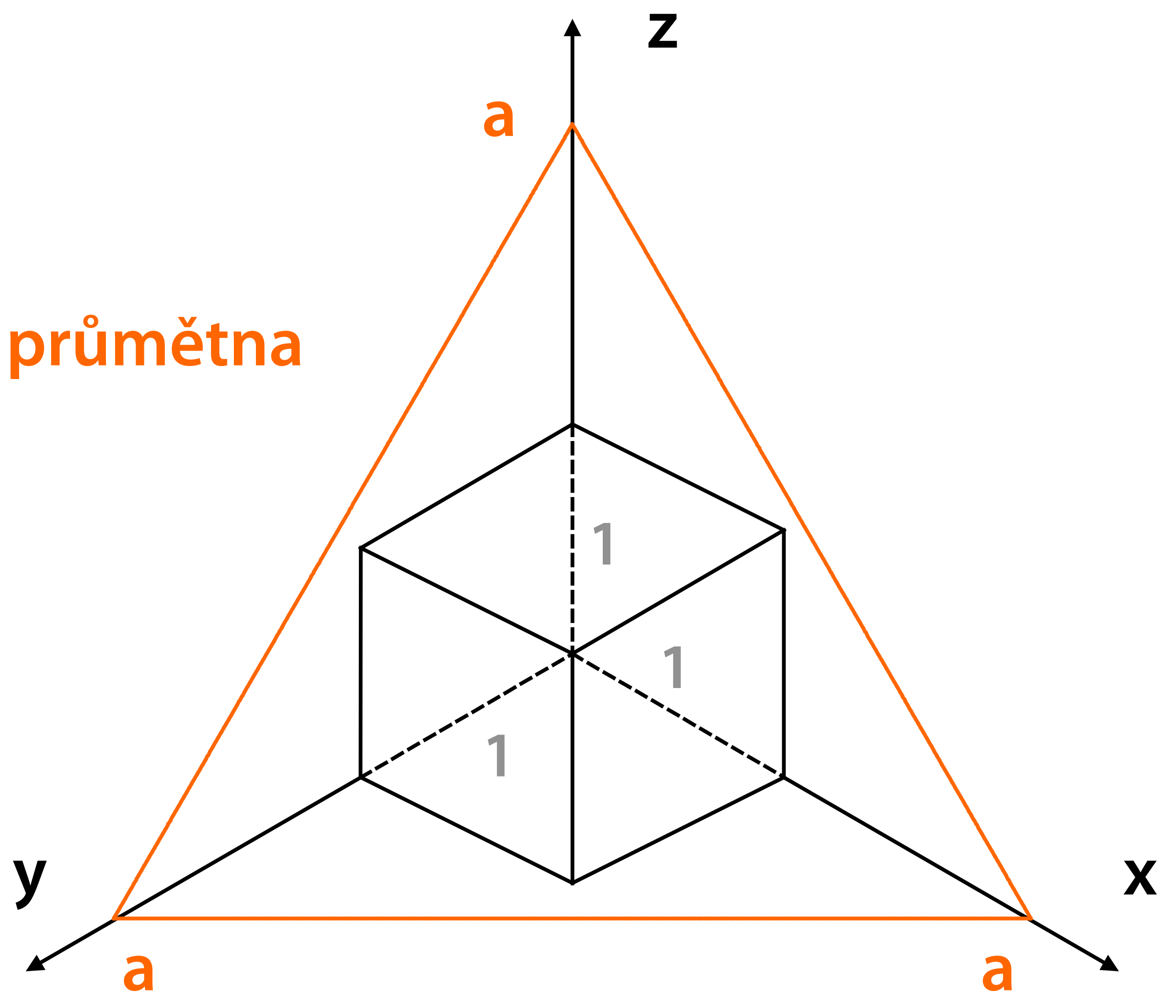
Kosoúhlé projekce

- kabinetní projekce (zkrácení měřítka osy **z** na **1/2**)
- kavalírní projekce (stejné měřítko na všech osách)

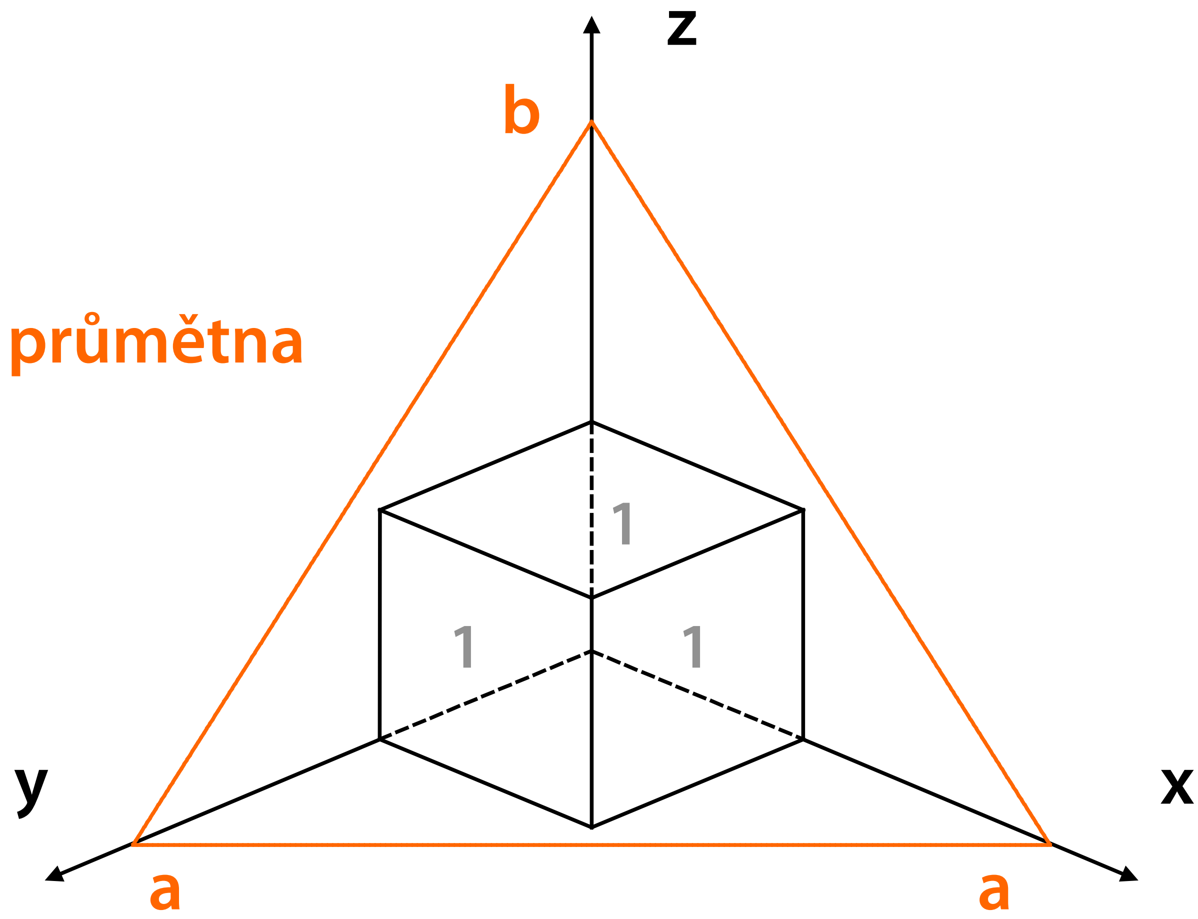
Mongeova projekce



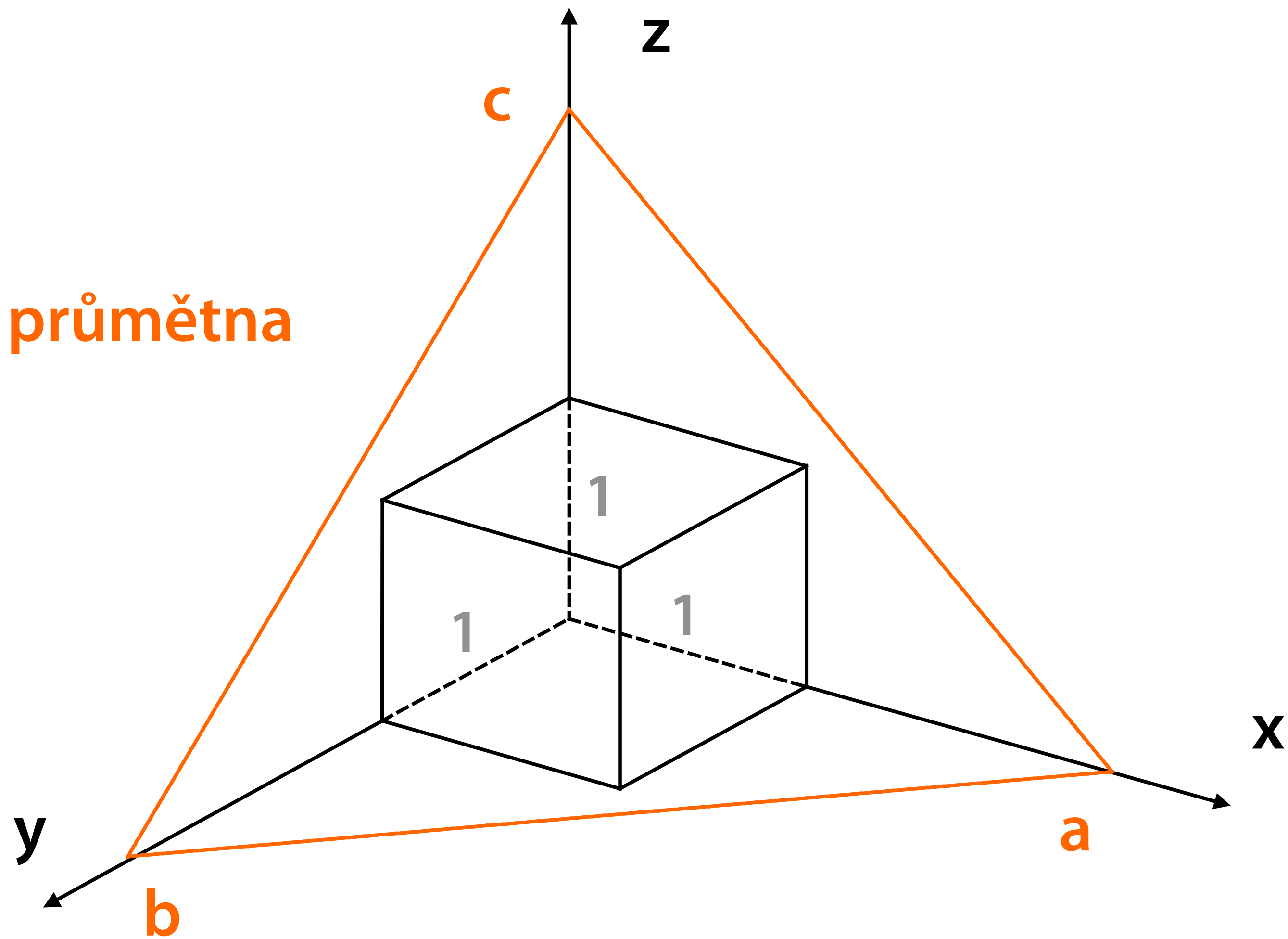
Axonometrie – isometrie



Axonometrie – dimetrie

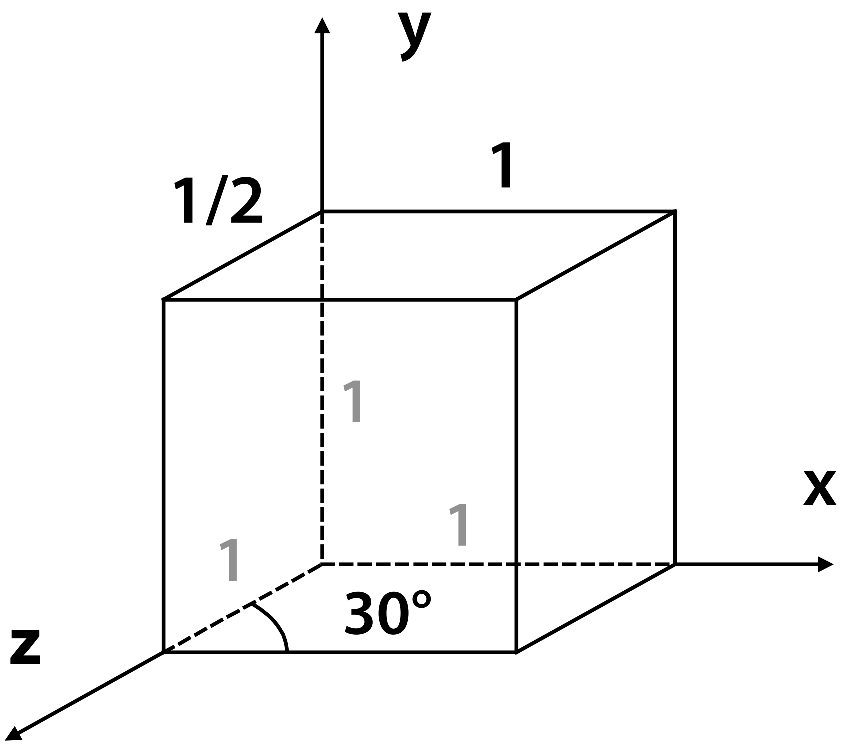


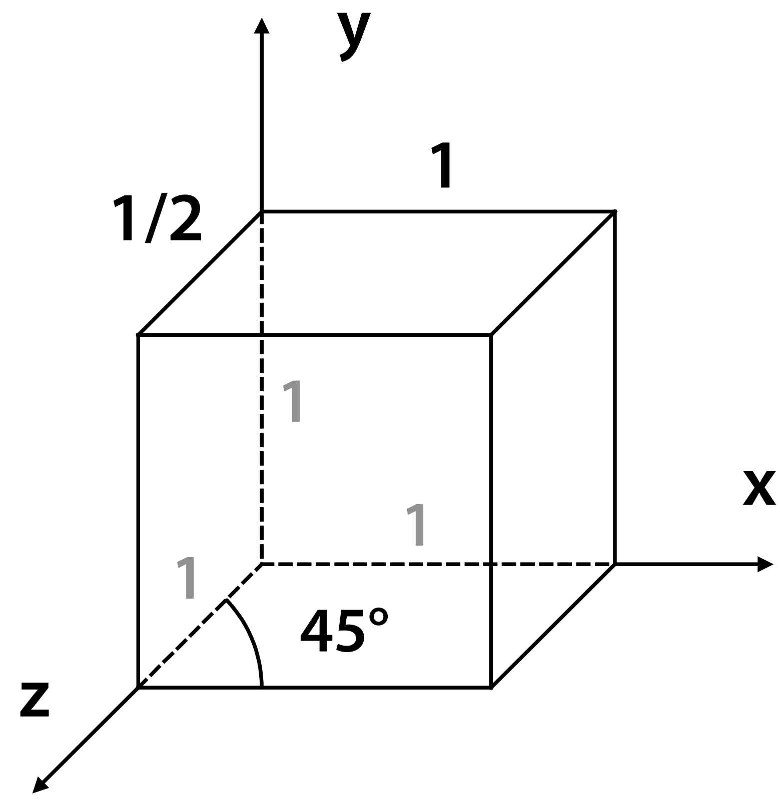
Axonometrie – trimetrie



Kabinetní projekce

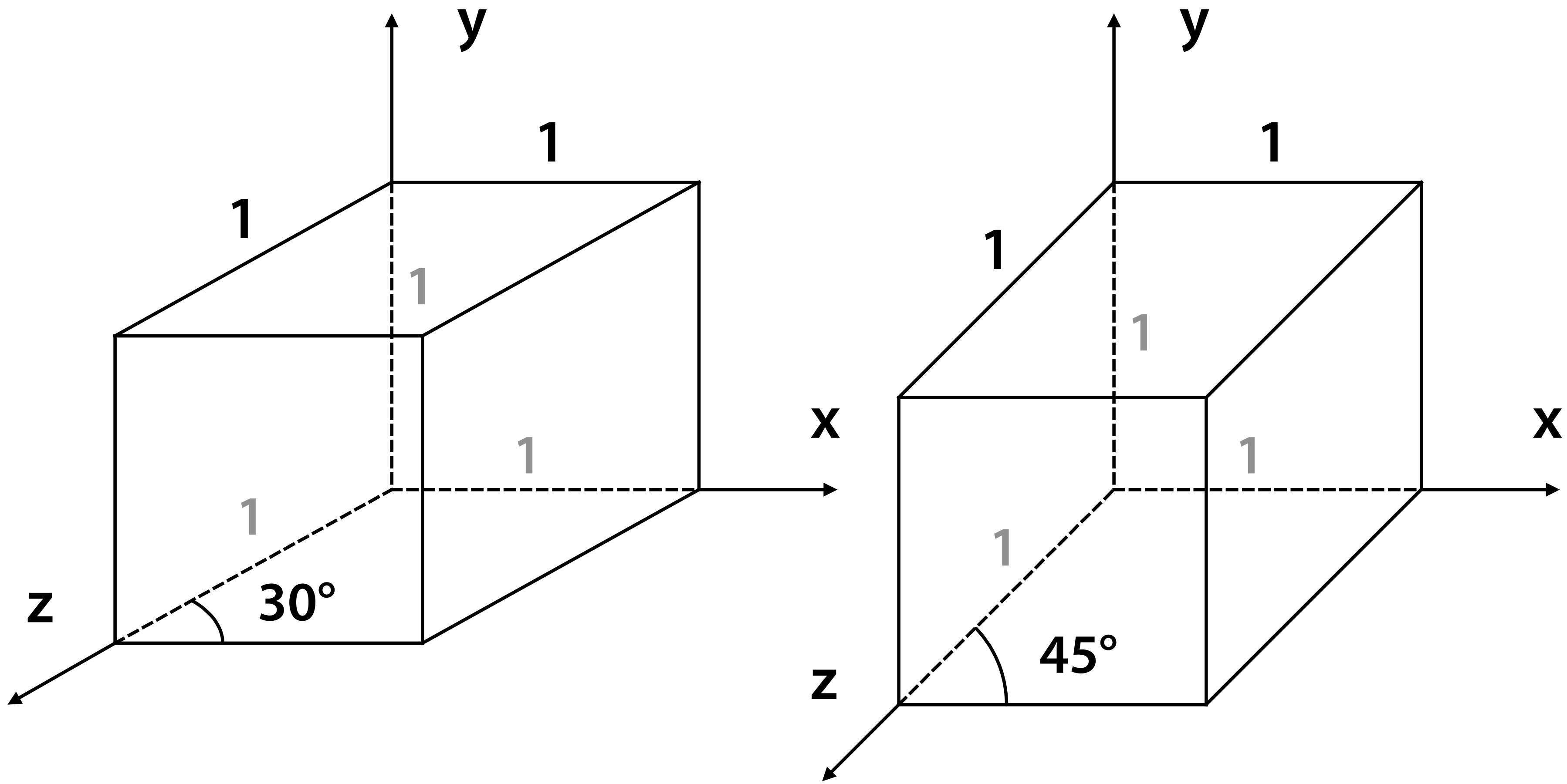
průmětna = xy





Kavalírní projekce

průmětna = xy



Klasifikace lineárních projekcí

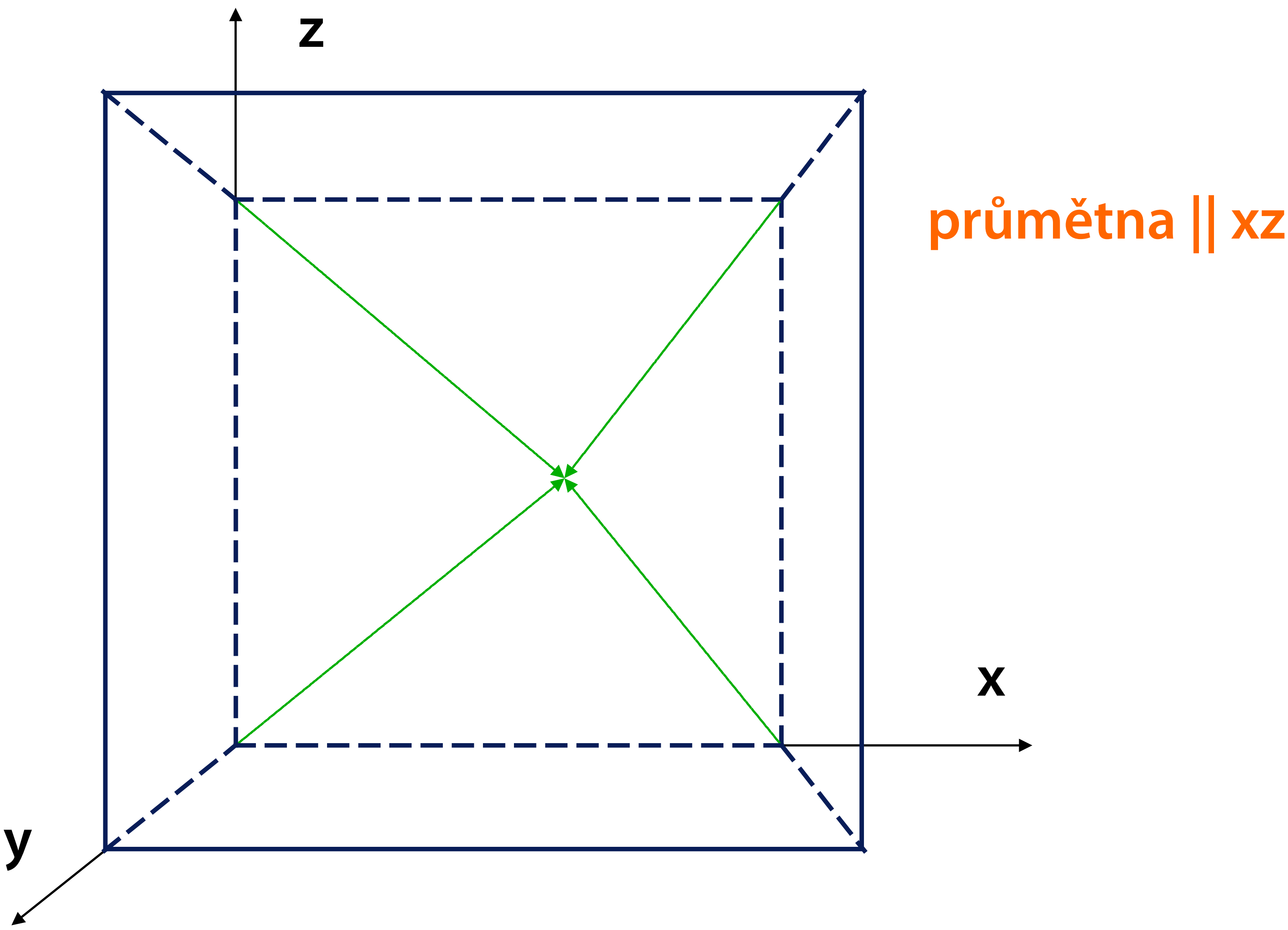
Středové projekce („perspective projections“)

- promítací paprsky tvoří svazek procházející jedním bodem, **středem projekce**
- nezachovává se rovnoběžnost (úběžníky)

Jednobodová perspektiva

- průmětna je rovnoběžná se dvěma souřadnými osami
- rovnoběžky se třetí osou se protínají v jednom hlavním úběžníku

Jednobodová perspektiva



Další středové projekce

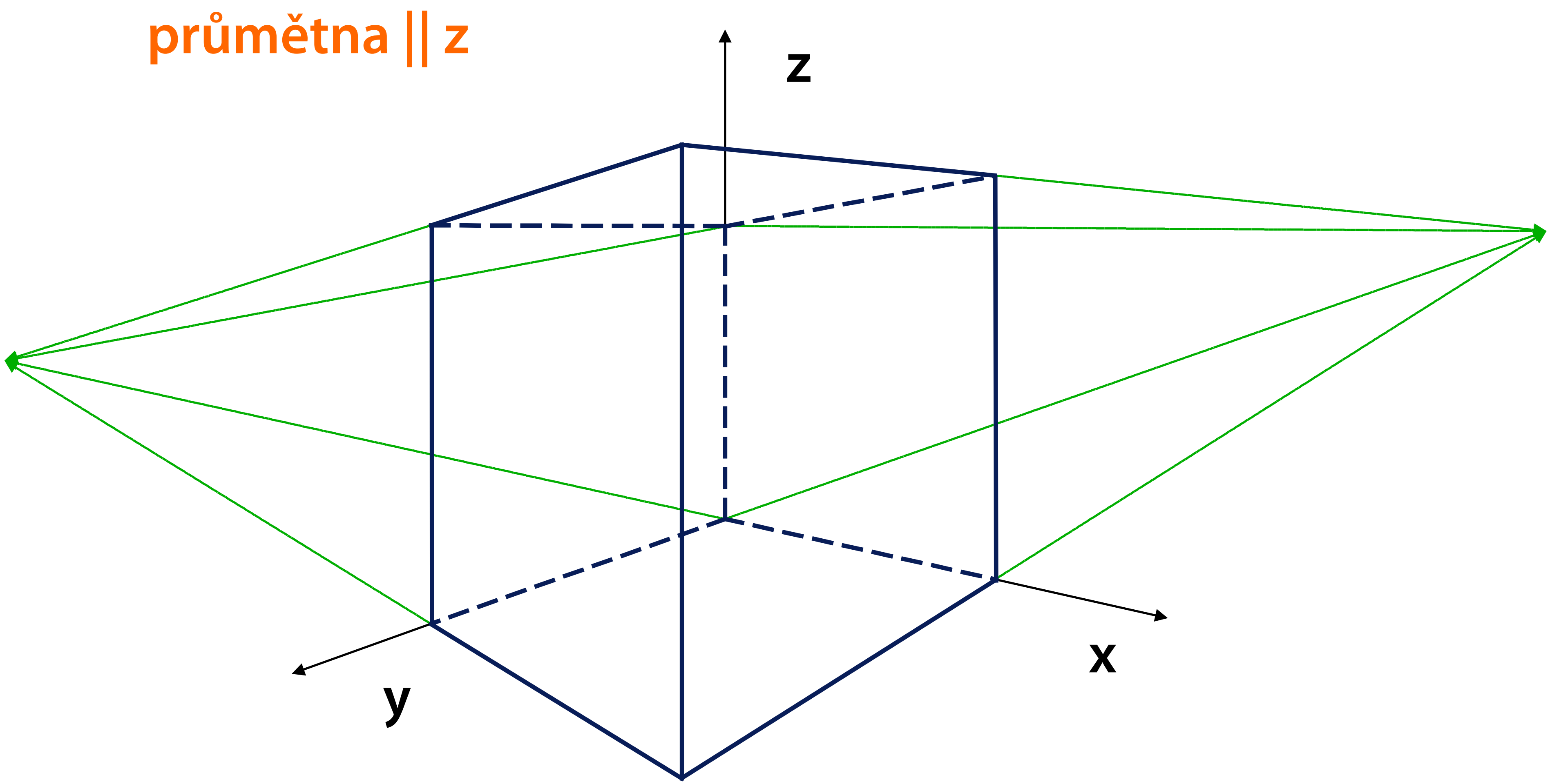
Dvoubodová perspektiva

- průmětna je rovnoběžná s jednou souřadnou osou
- rovnoběžky s ostatními osami se protínají ve dvou hlavních úběžnících

Tříbodová perspektiva

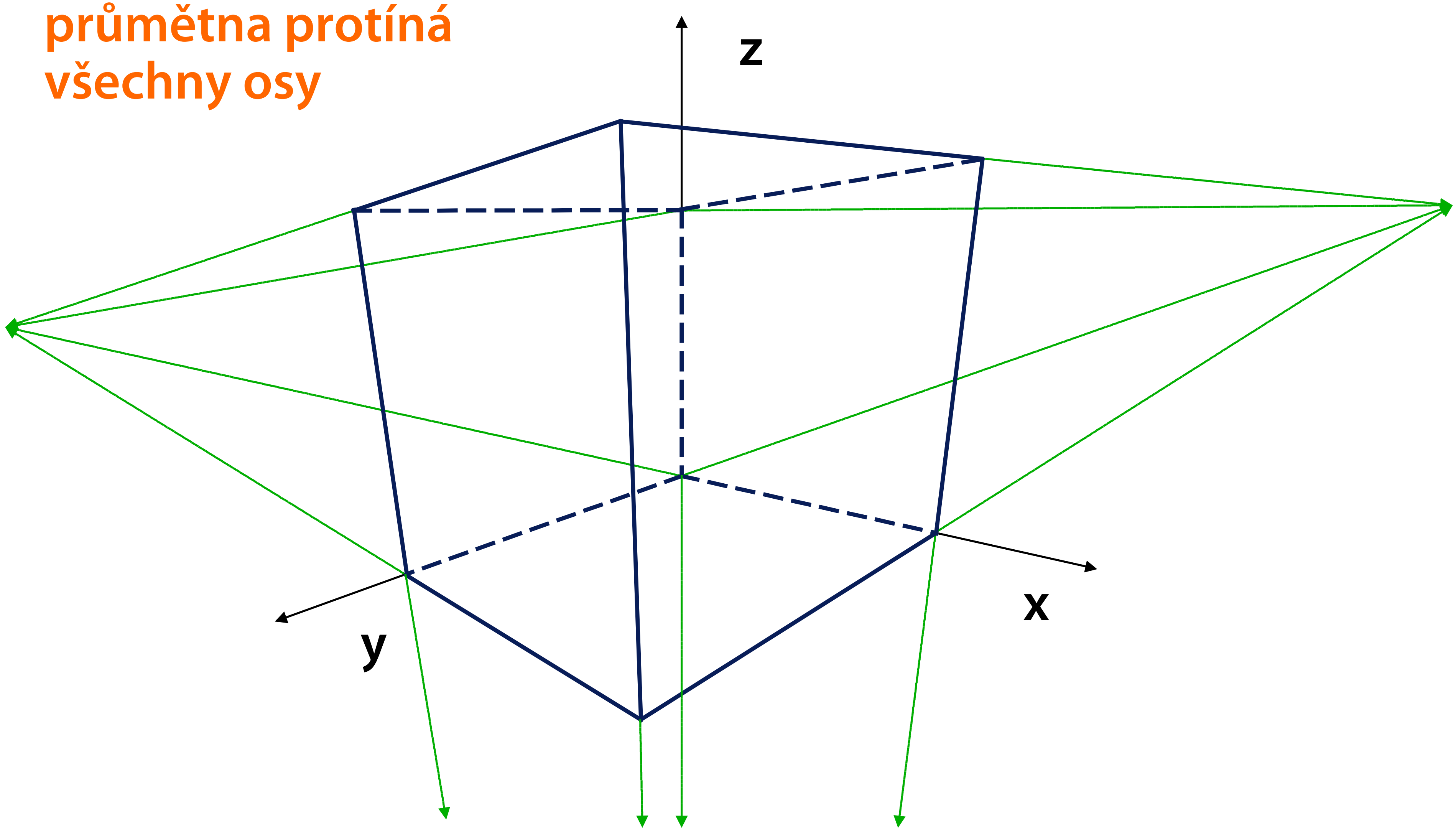
- průmětna má zcela obecnou polohu
- rovnoběžky se souřadnými osami se protínají ve třech hlavních úběžnících

Dvoubodová perspektiva



Tříbodová perspektiva

průmětna protíná
všechny osy



Implementace kolmé projekce

[x, y] budou souřadnice bodu v průmětu, **z** jeho hloubka (vzdálenost od pozorovatele)

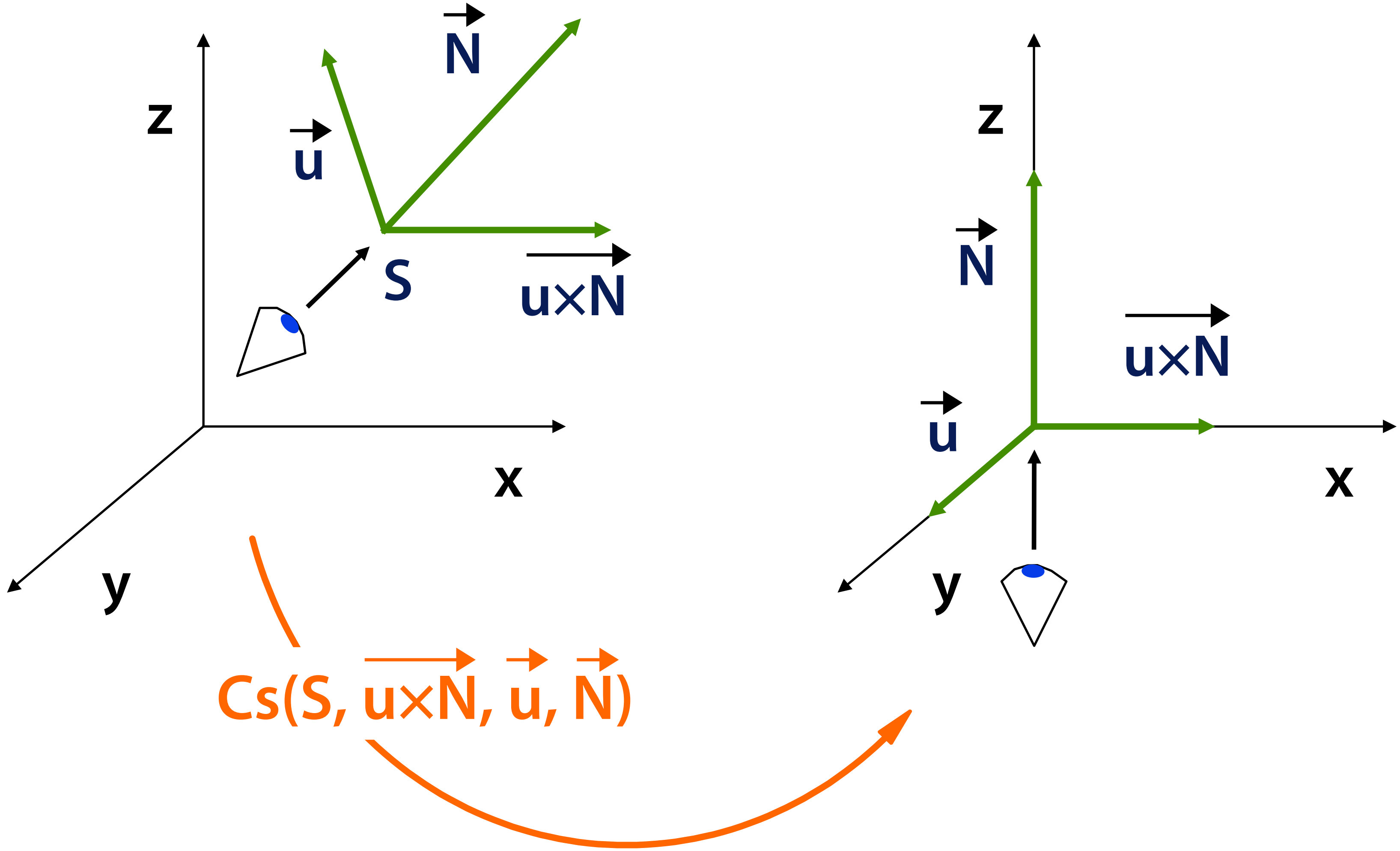
Základní pohledy (půdorys, nárys, bokorys)

– pouze permutace složek **x**, **y** a **z** (s příp. změnou znaménka)

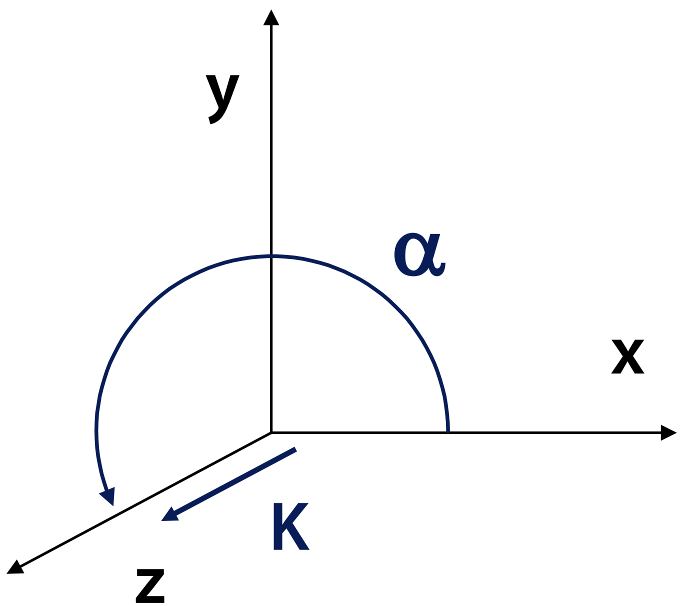
Obecná kolmá projekce (axonometrie)

– **směr pohledu** (normálový vektor průmětny): $\vec{\mathbf{N}}$
– **svislý vektor**: $\vec{\mathbf{u}}$
– převedení do základního pohledu: $\mathbf{Cs}(\mathbf{S}, \overrightarrow{\mathbf{u}\times\mathbf{N}}, \vec{\mathbf{u}}, \vec{\mathbf{N}})$

Obecná kolmá projekce



Implementace kosoúhlé projekce



průmětna: xy
koeficient zkrácení: K
úhel průmětu osy z: α

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ K\cdot\cos\alpha & K\cdot\sin\alpha & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Implementace středové projekce

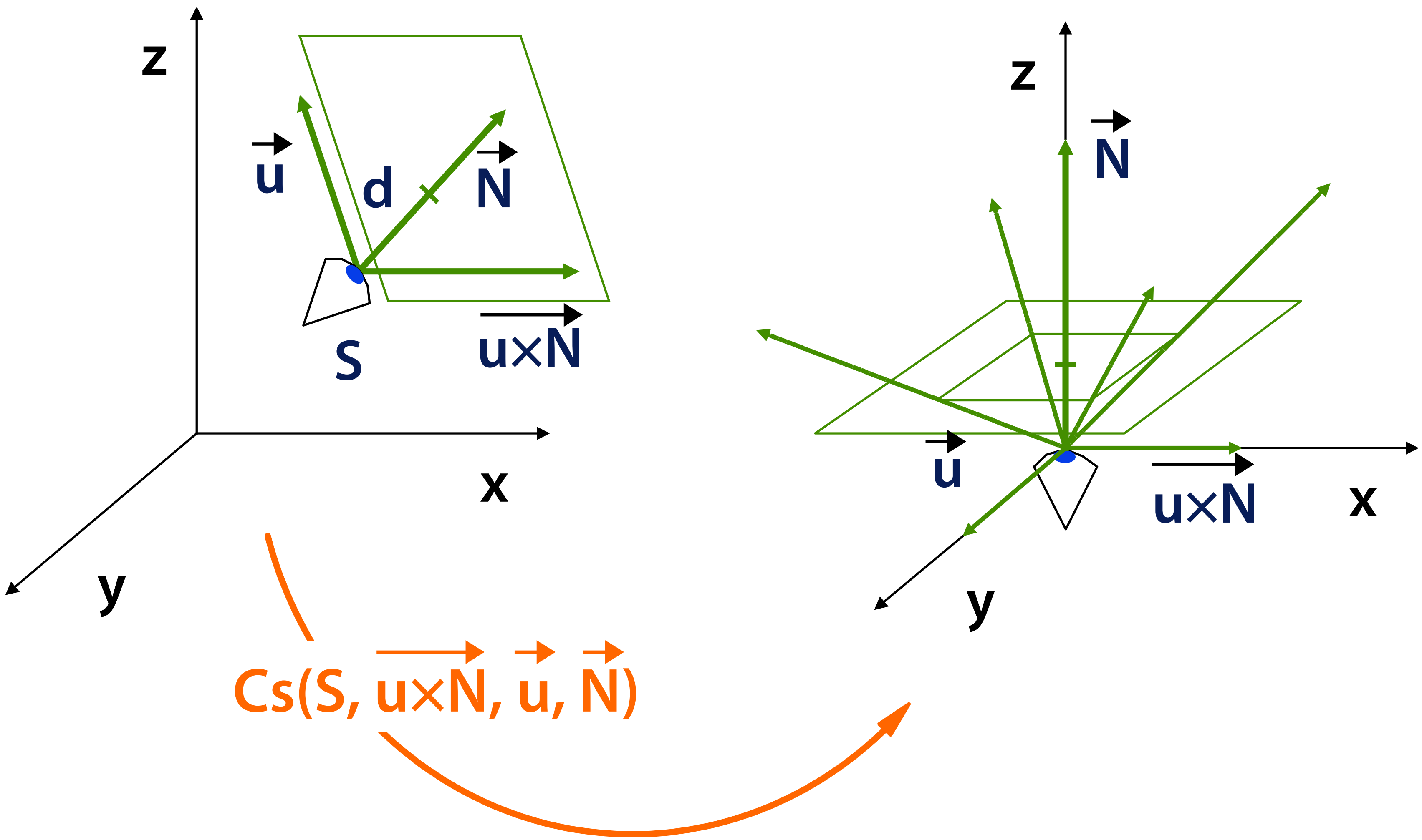
Obecná středová projekce

– střed projekce: **S**
– směr pohledu (normálový vektor průmětny): $\vec{\mathbf{N}}$
– vzdálenost průmětny od středu projekce: **d**
– svislý vektor: $\vec{\mathbf{u}}$

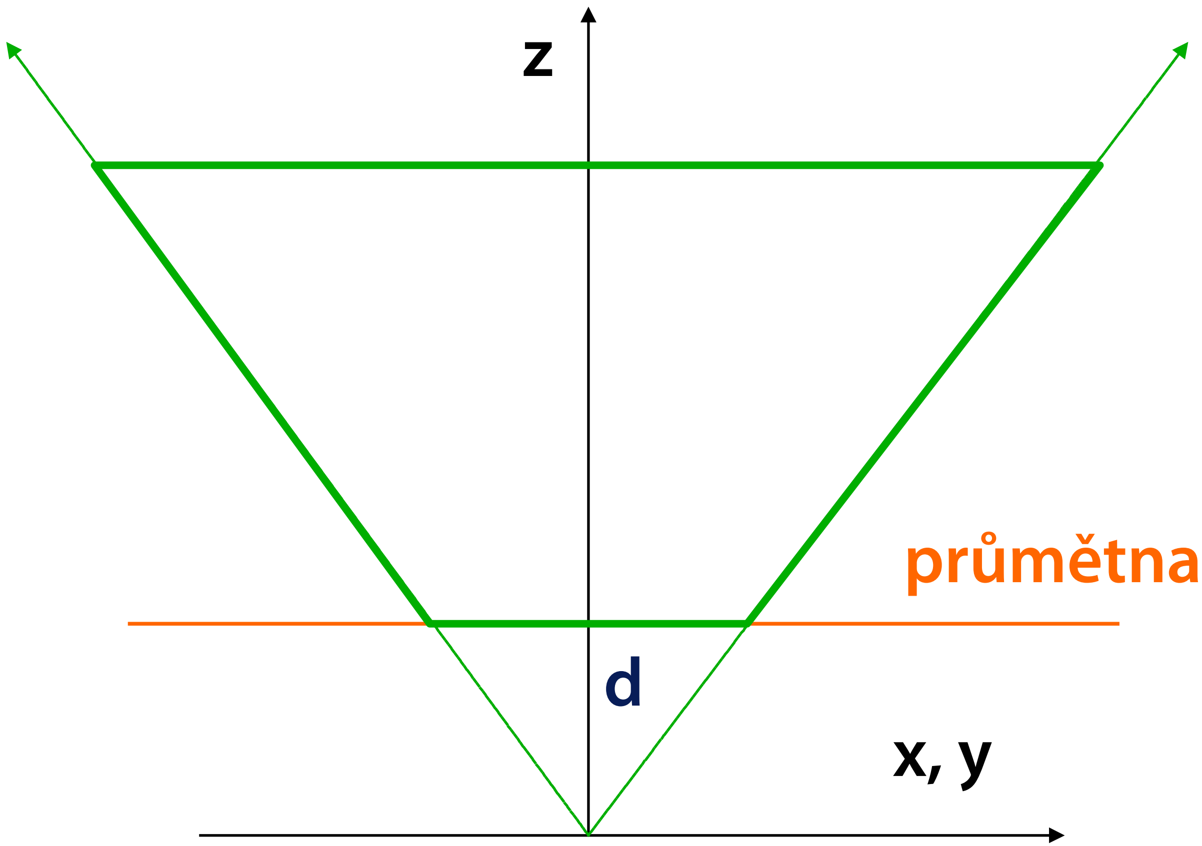
Promítací transformace

– převedení do **základní polohy** (střed projekce do počátku, směr pohledu do osy **z**): $\mathbf{Cs}(\mathbf{S}, \overrightarrow{\mathbf{u} \times \mathbf{N}}, \vec{\mathbf{u}}, \vec{\mathbf{N}})$
– **perspektivní projekce**: např. $[\, \mathbf{x \cdot d/z}, \mathbf{y \cdot d/z}, \mathbf{z} \,]$

Převedení do základní polohy



Perspektivní transformace



Nezachovává linearitu útvarů!

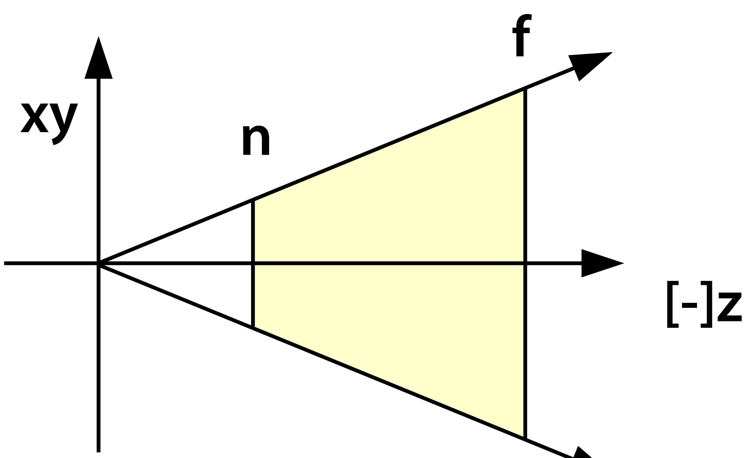
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & {}^{1}\!/_{d} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Perspektivní projekce na GPU

Vzdálený bod **f** může být i <u>v nekonečnu</u>

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2n}{t-b} & 0 & 0 \\ -\frac{r+l}{r-l} & -\frac{t+b}{t-b} & \frac{f+n}{f-n} & 1 \\ 0 & 0 & -\frac{2fn}{f-n} & 0 \end{bmatrix}$$

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2n}{t-b} & 0 & 0 \\ -\frac{r+l}{r-l} & -\frac{t+b}{t-b} & 1 & 1 \\ 0 & 0 & -2n & 0 \end{bmatrix}$$



Transformace lineárních útvarů

Perspektivní transformace úsečky Per

– je zřejmé, že **neplatí** rovnost

$$\mathbf{Per(A + t \cdot [B - A]) = Per(A) + t \cdot [Per(B) - Per(A)]}$$

Použití **diferenčních algoritmů** (DDA) při výpočtu viditelnosti

– mějme bod **C(u)** na úsečce **Per(A)Per(B)**

$$\mathbf{C(u)_{x,y} = Per(A)_{x,y} + u \cdot [Per(B)_{x,y} - Per(A)_{x,y}]}$$

– potřebujeme, aby i pro hloubku **z** platilo

$$\mathbf{C(u)_z = Per(A)_z + u \cdot [Per(B)_z - Per(A)_z]}$$

Interpolation in the screen space

Concerns **clip-space** and next spaces

- clip space: [x, y, z, w]
- NDS: [x/w, y/w, z/w, w] ("w" could be preserved)
- window space (fragments): $[x_i, y_i, z_i, w]$

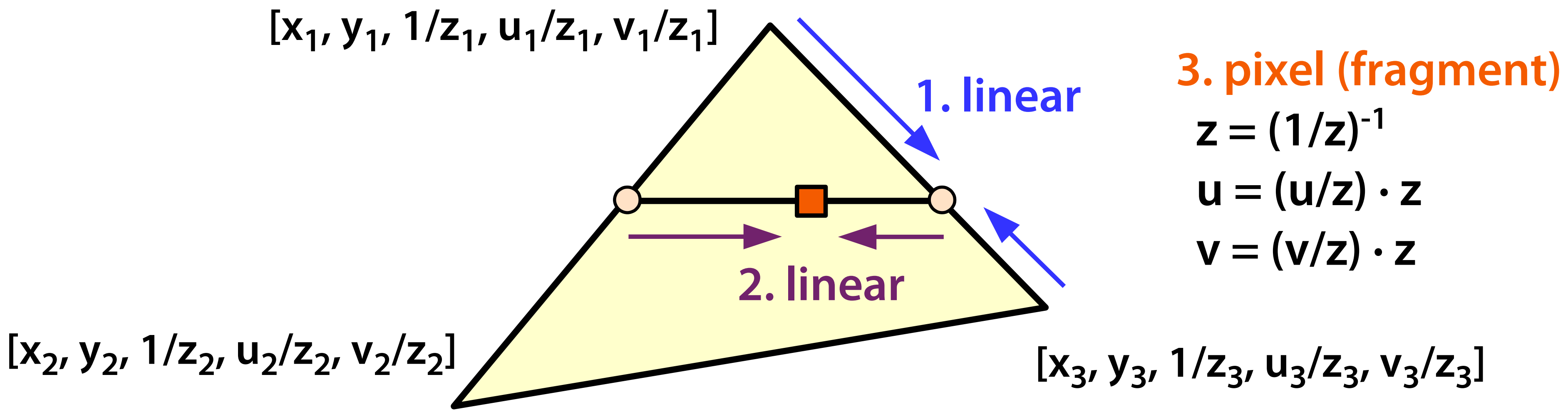
Projective perspective transformation maps depth **z** to NDS **non-linearly!**

− **nonuniform accuracy** of z-buffer (distant parts could be less accurate: minimization of **f/n** ratio!)

\+ **interpolation of depth 1/z** can be done in the screen space **linearly**

„W-buffer" instead of „z-buffer" (not frequently used)

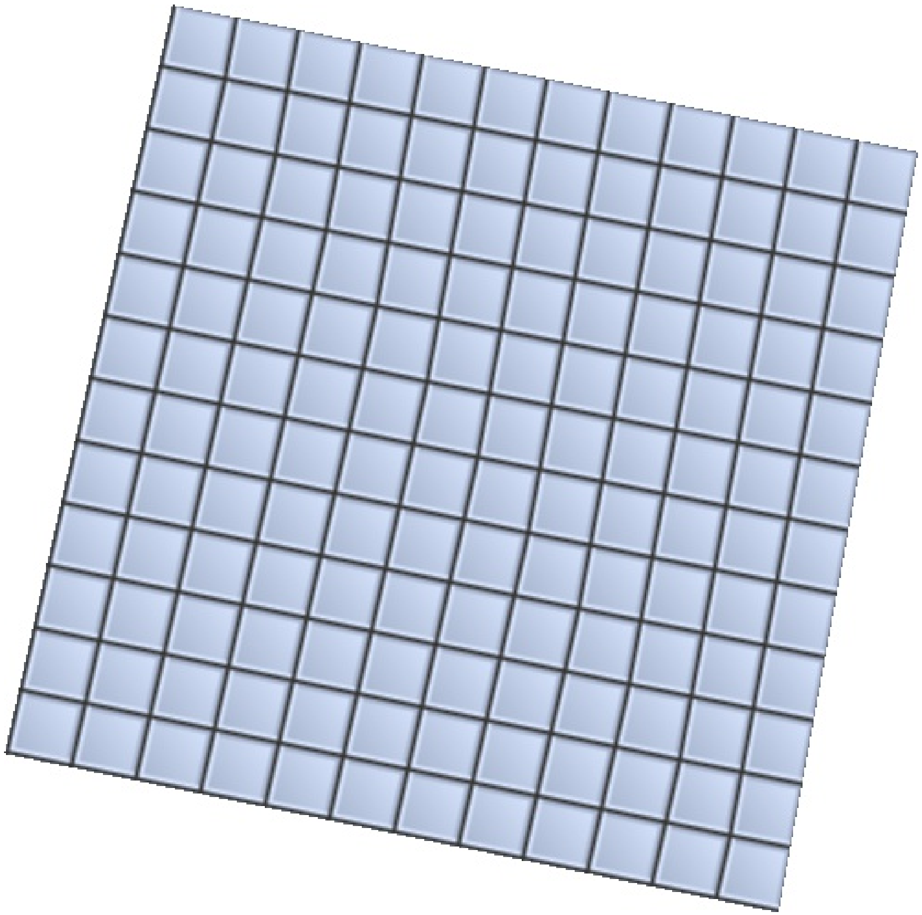
Perspective-correct interpolation



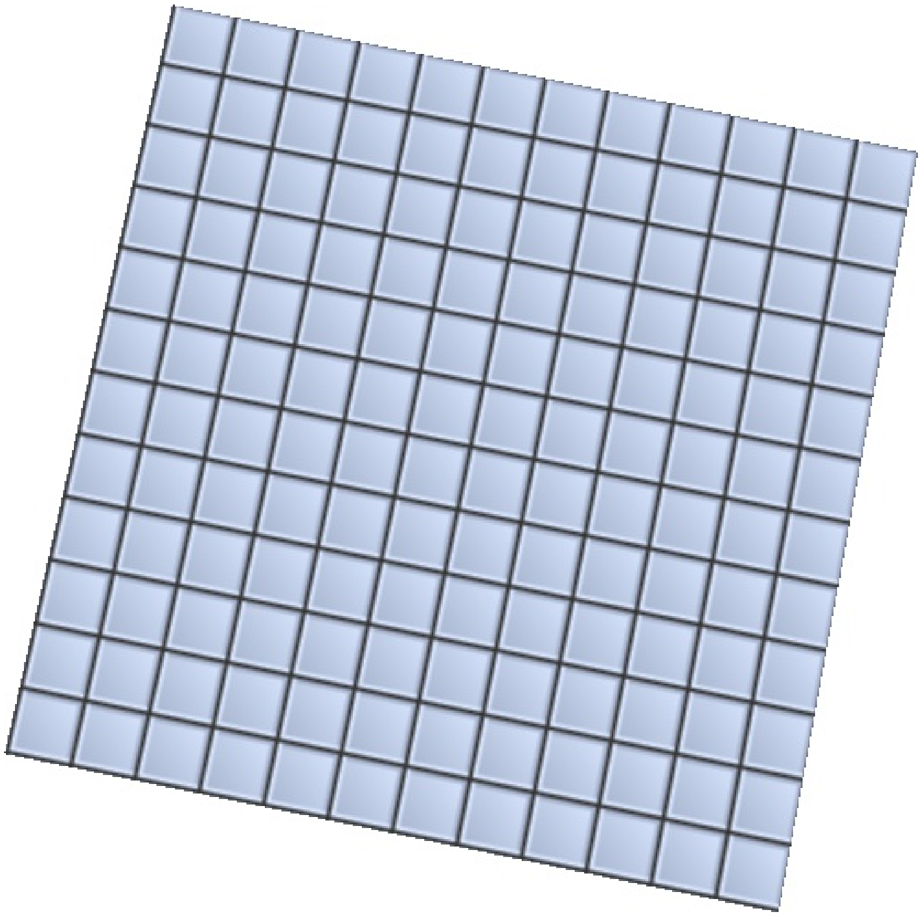
Linear interpolation + division (hyperbolic interpolation)

- **reciprocal depth "1/z"** can be interpolated linearly
- for additional attributes (**texture coordinates**) perspective-correct adjustment must be used
- quantities **"1/z"**, **"u/z"**, **"v/z"**… are linearly interpolated, **[u, v]** is then computed by division

Interpolation example I

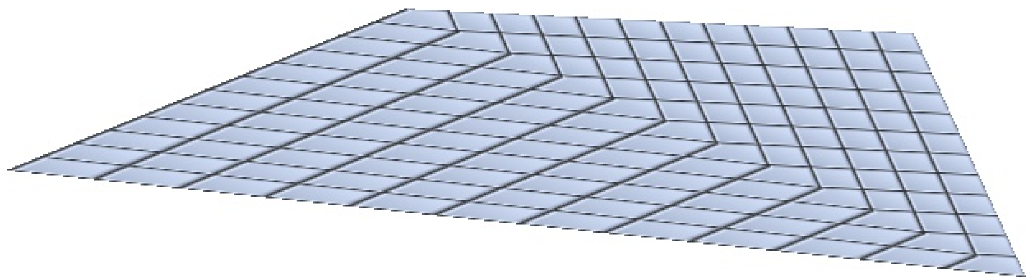


Affine

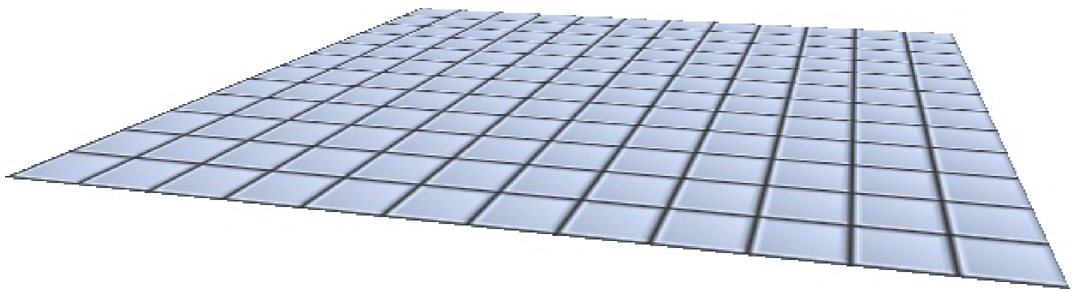


Correct

Interpolation example II

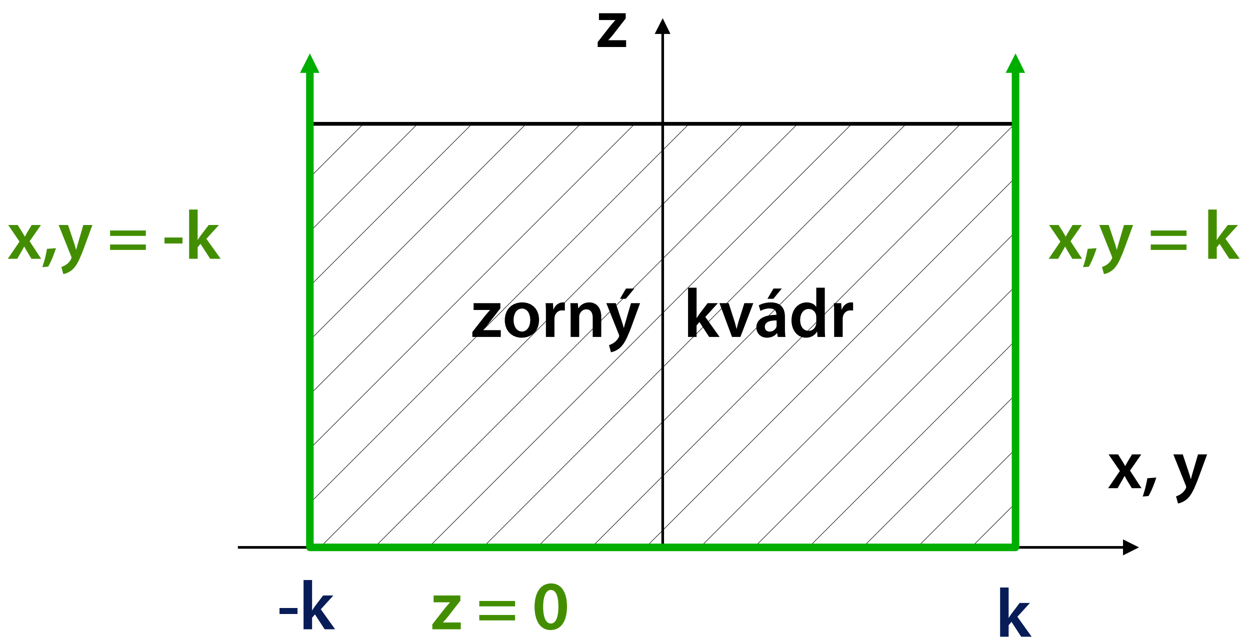


Affine



Correct

4D ořezávání



Hraniční nadroviny

x = -kw, x = kw, y = -kw, y = kw, z = 0

pro **w > 0: -kw < x < kw, -kw < y < kw, 0 < z**

Literatura

J. Foley, A. van Dam, S. Feiner, J. Hughes: *Computer Graphics, Principles and Practice*, 229-283

Jiří Žára a kol.: *Počítačová grafika, principy a algoritmy*, 277-291

Kok-Lim Low: *Perspective-Correct Interpolation* (report, proof), University of North Carolina at Chapel Hill, 2002