

# Základy počítačové grafiky

## Přednáška 1

Martin Němec

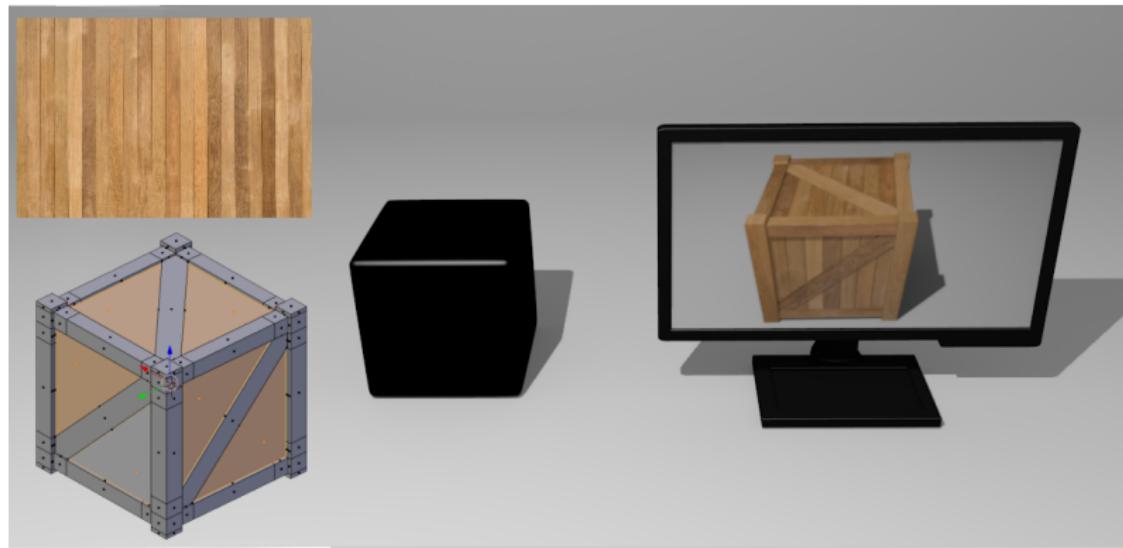
VŠB-TU Ostrava

2024

# Cíl předmětu

Seznámit posluchače se základními principy počítačové grafiky a vývoje grafických aplikací.

- **vstup** - modely, textury, scéna, transformace, shadery atd.
- **výstup** - realtime renderování a procházení 3D scény.



Cíl: Vytvořit aplikaci s využitím OpenGL (máme na to X týdnů)

## Zápočet celkem 45 bodů (min. 20 bodů)

- Postupné vytváření 3D aplikace (C++ a OpenGL), obhajoba v zápočtovém týdnu.

## Zkouška celkem 55 bodů (min. 20 bodů)

- Písemná část - 25 bodů (min. 10 bodů)  
zápočtový týden na přednášce.
- Ústní zkouška - 30 bodů (min. 10 bodů).  
zkouškové období (2x otázka).

# Obsah přednášek

- Úvod do standardního zobrazovacího řetězce (OpenGL).
- Homogenní souřadný systém, transformace.
- Promítání, kamera, souřadné systémy.
- Světlo, intenzita osvětlení, lokální osvětlovací modely.
- Textury v OpenGL, mapování.
- Řešení viditelnosti (z-buffer, malířův algoritmus), skybox.
- Optická iluze nerovnosti, normal mapping.
- Identifikace objektů.
- Stíny v počítačové grafice.
- Úvod do křivek a ploch.

Obsah i pořadí se může mírně změnit.

# Státnice - Počítačová grafika (ZPG, URO)

- Metody a nástroje pro realizaci grafických uživatelských rozhraní (kognitivní schopnosti člověka, mentální modely, základní pravidla designu, barevné prostory, volba barev a prezentace textu).
- Standardní zobrazovací řetězec (realizace jednotlivých kroků řetězce, modelovací a zobrazovací transformace, Phongův osvětlovací model, řešení viditelnosti, identifikace těles, stručná charakteristika standardu OpenGL a jazyka GLSL).
- Geometrické modelování (affinní a projektivní prostory, popis těles a možnosti jejich reprezentace, základní křivky používané v počítačové grafice, jejich vyjádření, vlastnosti a použití, Fergusonova kubika, Bézierova křivka).

Příklad otázky: Popište možnosti reprezentace těles a způsob jejich vykreslení pomocí standardního zobrazovacího řetězce v kontextu grafického rozhraní OpenGL.

## **Bakalářské studium**

Základy počítačové grafiky

Modelování v grafických aplikacích

## **Magisterské studium**

Počítačová grafika I

Počítačová grafika II

Analýza obrazu

Geometrie pro počítačovou grafiku

Vizualizace dat

Digitální zpracování obrazu

Algoritmizace geometrických úloh

# Zápočet

Během semestru budeme postupně na cvičeních vytvářet aplikaci (jazyk C++ a OpenGL 3.3+), která bude umožňovat načíst 3D scénu (modely, textury, světla apod.), načtenou scénu následně vizualizovat a procházet s využitím základního vykreslovacího řetězce.

Seznámíme se s jednotlivými kroky OpenGL, transformacemi, osvětlením, animacemi, shadery atd.

Budeme předpokládat, že student má:

- znalost probrané SŠ a VŠ matematiky;
- znalost programování v C++;
- znalost OOP programování.

# Zápočet

Na OpenGL aplikaci budeme pracovat průběžně na jednotlivých cvičeních, včetně kontroly přes <http://kelvin.cs.vsb.cz>.

## Úkol 1

Select multiple files using CTRL or SHIFT or drag them to this window

Assignment

**Zadání**

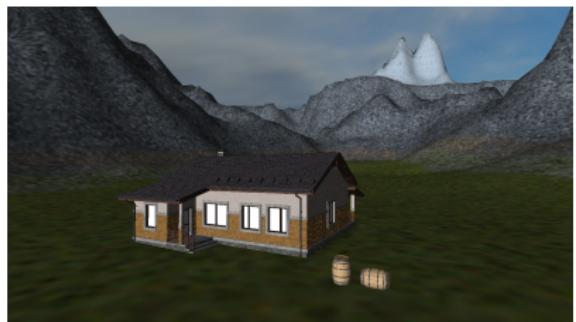
- Upravte stávající projekt z prvního cvičení na moderní OpenGL a objektový kód (viz přednáška a cvičení).
- Funkce `main` bude mít pouze inicializaci základního objektu `Application` a případné nastavení a spouštění. Doplňte třídy minimálně pro modely, shadery, kontrolu callback funkcí atd. Základní požadavky byly probrány na přednášce.

**Odevzdání úkolu**

- Odešlete zdrojové soubory `CPP` a `H` (neposílejte knihovny), které můžete zabalit do `zip` souboru.
- Přiložte screen obrazovky výsledky (trojúhelník s pozicí na barvě, čtverec atd.).

# Zápočet

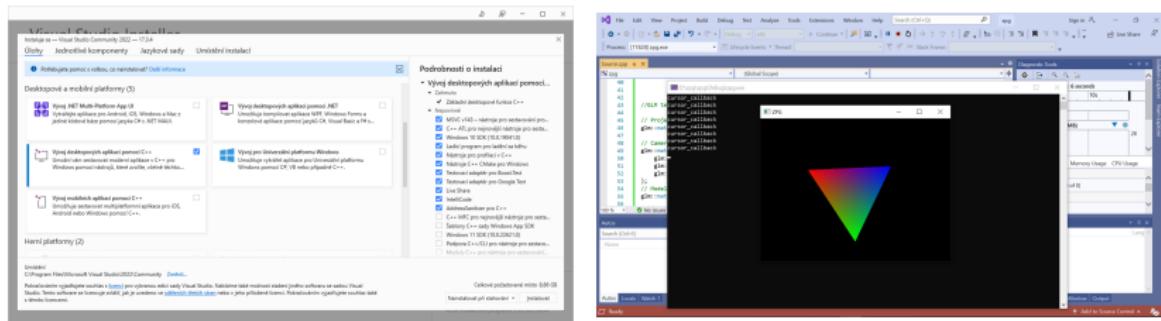
V aplikaci se pokusíme vykreslit základní 3D prostředí (světla, modely, textury, pozadí atd.), které budeme moci pomocí myši a klávesnice procházet. V projektu budeme používat různé modely (obvykle cizí), ale zkusíme si vytvořit i vlastní.



V předmětu MGA (volitelný, příští semestr) budeme modelovat v prostředí Blender a s využitím Unreal Enginu vlastní 3D prostředí (vizualizaci za nás bude dělat UE).

# Cvičení 1

Primárně budeme používat vývojové prostředí Visual Studio Community, které je na všech učebnách a je zdarma. Lze použít i jiné vývojové prostředí, podle preferencí, operačního systému apod. Budeme potřebovat externí knihovny: pro práci s okny, matematické operace, načítání obrázků, modelů atd.



# Výsledná 3D aplikace

- Cílem předmětu není jen počítačová grafika, ale zopakovat i dříve získané znalosti (počítačová grafika, programování, matematika, geometrie atd.).
- Vytvoříme si čistý C++ projekt a postupně jej budeme rozšiřovat.
- Nejde jen o to dopsat někam kousek kódu, ale vše si od začátku vytvořit.
- Budeme se snažit nejen "něco zobrazit", ale porozumět tomu co programujeme.
- Součástí cvičení bude i refactoring, optimalizace, testování atd.

Mezi nejčastěji používané rozhraní pro programování grafických aplikací, tzv. API (Application Programming Interface) patří:

- **OpenGL** - (Khronos Group) Vyšší úroveň abstrakce, méně efektivní, vhodnější pro jednodušší aplikace. Jednodušší, vhodné pro začátečníky.
- **Vulkan** - (Khronos Group) Považován za nástupce OpenGL, nízkoúrovňové API, navržen pro vysoký výkon na moderním vícevláknovém hardwaru.
- **DirectX** - (Microsoft) Obdobně jako Vulkán je DirectX 12 nízkoúrovňové API, optimalizované pro moderní vícejádrové procesory a grafické karty. Primárně pro Windows a Xbox.

# Vznik OpenGL

**Open Graphics Library** (OpenGL) je multiplatformní rozhraní (API) pro tvorbu grafických aplikací. Od roku 2006 spadá pod Kronos Group.

- první vydání v červnu 1992 (Silicon Graphics);
- zaměřen čistě na vykreslování (neřeší ovládání, okna atd.);
- během vývoje postupně upravován (fixní a programovatený);
- aktuální verze 4.6 (2017);

**OpenGL Shading Language** (zkráceně GLSL) je programovací jazyk pro psaní shaderů.

**Vulkan** - nástupcem OpenGl, nízkoúrovňové API pro moderní grafické karty (paralelizace).

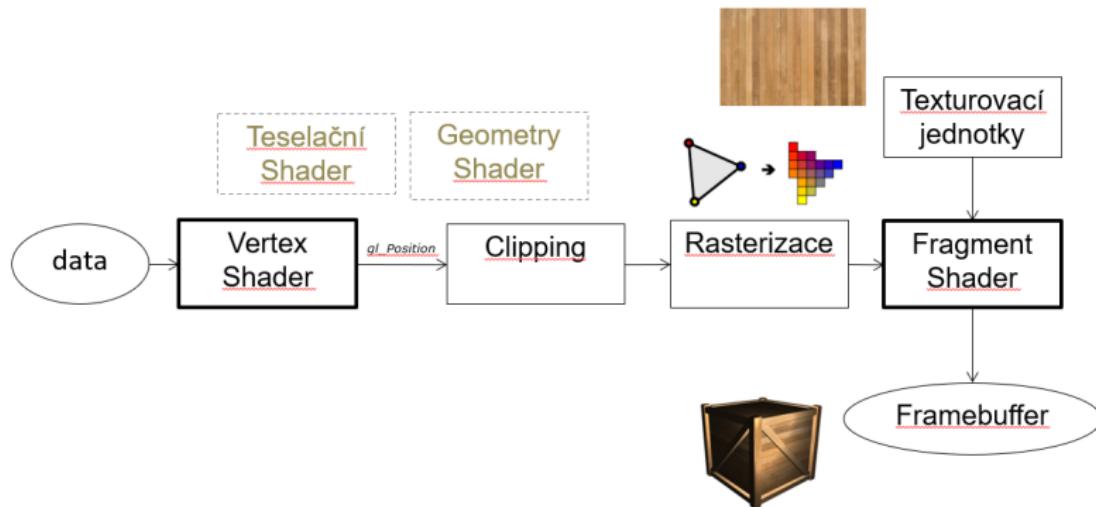
# Projekt

Praktická cvičení budeme věnovat:

- vytvoření projektu;
- vykreslení trojúhelníku;
- práce s shadery;
- transformace;
- kamera;
- osvětlení;
- načítání modelů (FBX);
- identifikace modelů;
- skybox;
- atd.

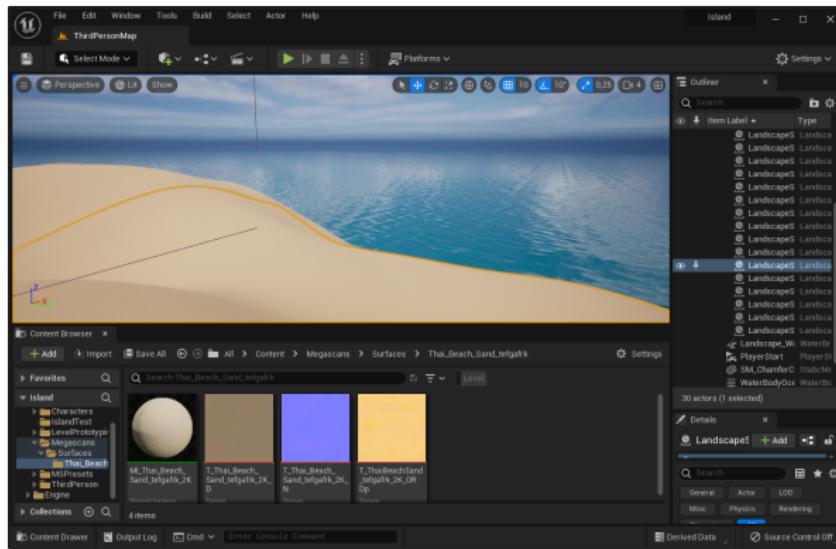
# Vykreslovací řetězec

- Vertex Shader - převážně k aplikaci transformací (`gl_Position`)
- Clipping - ořezání neviditelných částí
- Rasterizace - rozklad těles na fragmenty (vektor-rastr)
- Fragment Shader - barvení, hloubka apod.



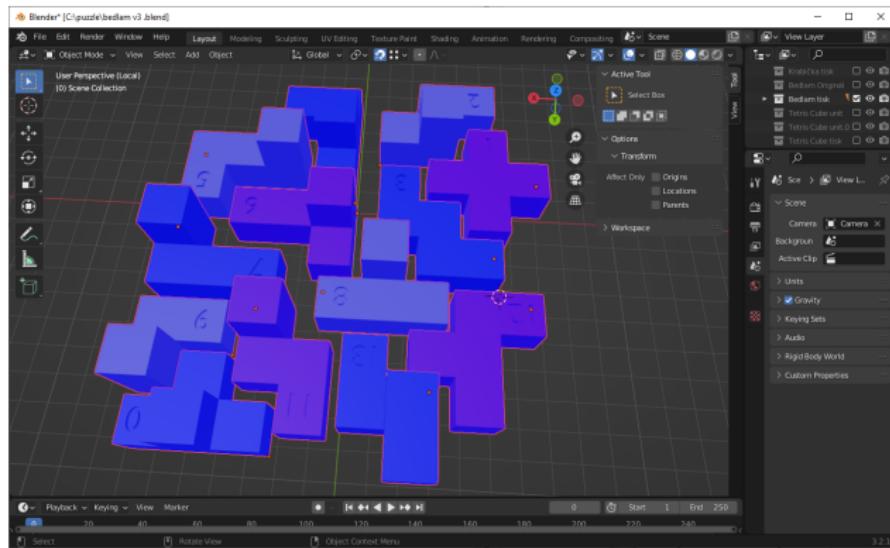
# Unreal Engine

Proč si implementovat vlastní vykreslovací řetězec, když lze profesionální 3D aplikace využít i jinak?



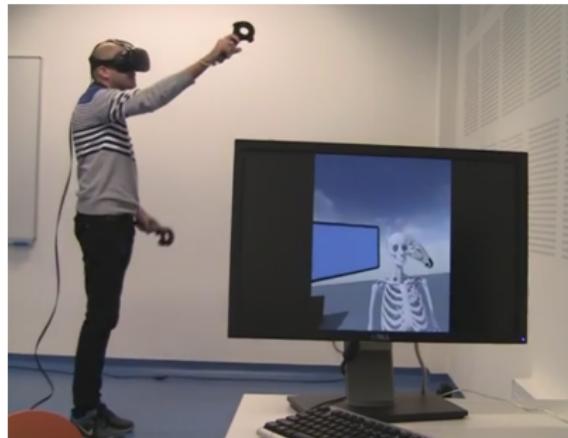
# Blender

Velmi populární open source nástroj pro 3D modelování a tvorbu animací. Bude mu věnován rozšiřující předmět MGA.



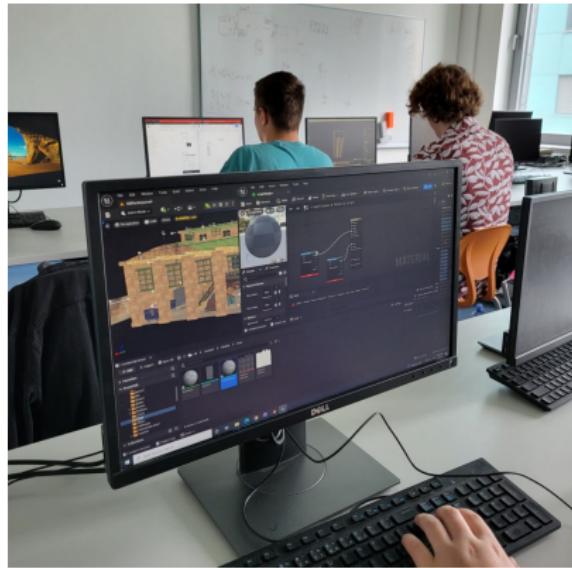
# Co nestihneme?

Počítačová grafika má spoustu zajímavých částí, ale vše nestihneme.



# The Factory

Projekt se středoškoláky, který vznikl během dvou týdnů.



# Hobo: Tough Life

Perun Creative (Jan Polach a Jiří Vašica)



# Mall Craze

Milan Timčenko



Datum vydání: 19. září 2023

# Kvark

Radovan Šťastný, Petr Pavlík, Perun Creative



Datum vydání: 2. června 2023

# Učebna

## Učebna EB408



Processor: 13th Gen Intel(R) Core(TM) i7-13700KF, 3400 MHz, 16 Core(s), 24 Logical Processor(s). RAM: 64.0 GB. Graphics card: NVIDIA Geforce RTX 4080.

# Počítačová grafika

Počítačová grafika je obor informatiky, který se od svých počátků v 70. letech rozvinul do samotné vědní disciplíny.

- Vývoj je úzce propojen s výpočetními a zobrazovacími vlastnostmi počítačů (velký rozvoj v posledních letech).
- Počítačová grafika pronikla do různých odvětví (film, lékařství, vojenství, stavebnictví, projektování, modelování, apod.).
- Dnes se stává již nepostradatelnou součástí.



# Různá rozdělení

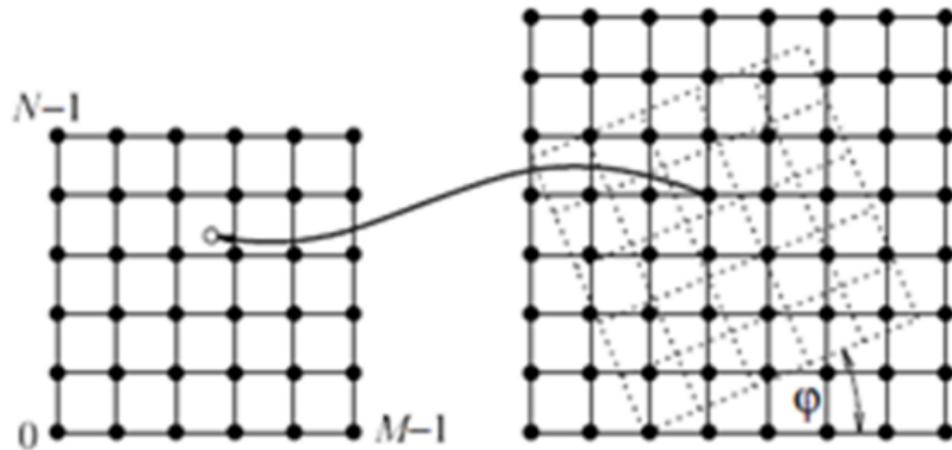
Počítačovou grafiku můžeme dělit podle různých kritérií.

- Podle dimenze - 2D grafika, 3D grafika.
- Podle popisu - rastrová grafika, vektorová grafika;
- Podle rychlosti - real-time rendering, fotorealistický rendering.



# Afinní transformace - rotace

Diskrétní obraz (DZO - digitální zpracování obrazu)

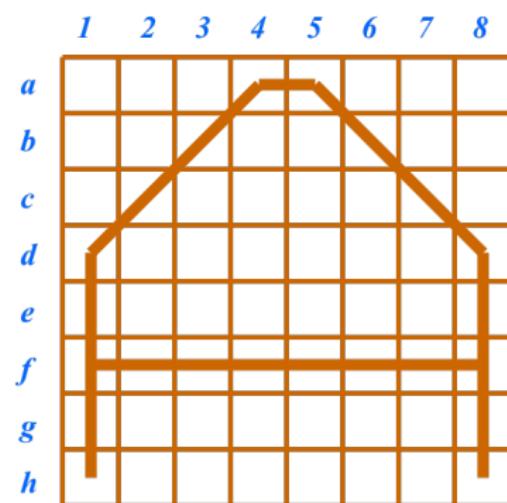
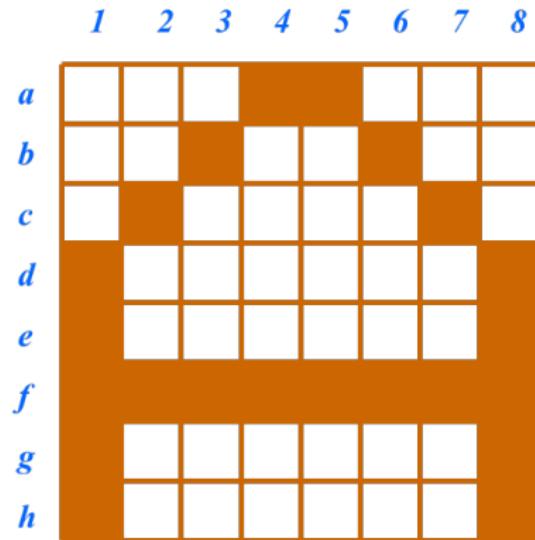


$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = \begin{bmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + (N-1) \begin{bmatrix} -\sin(\varphi) \cos(\varphi) \\ -\sin^2(\varphi) \end{bmatrix}$$

# Rastrový vs. vektorový popis

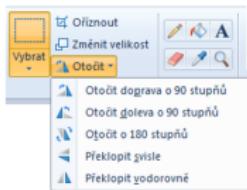
Rastrový formát - obrázek je popsán pomocí jednotlivých barevných bodů (pixelů).

Vektorový formát - obraz je složen z matematicky definovaných objektů (bod, kružnice, úsečka, křivka atd.).



# Rastrový vs. vektorový obraz

Srovnání výhod a nevýhod.



## Jak jsou řešené fonty?

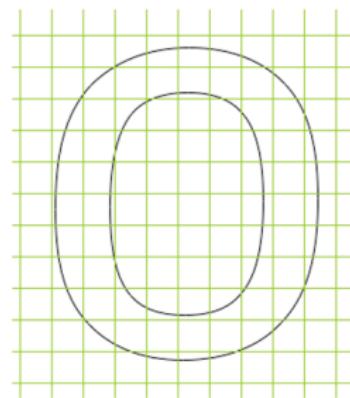
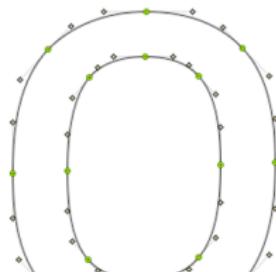
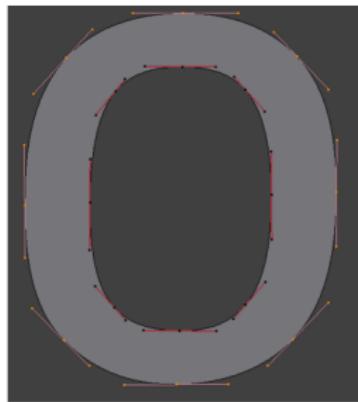


ab ab ab ab ab ab ab

TrueType fonty - kvadratické Bézierovy křivky  
Postscriptové fonty - kubické Bézierovy křivky

# Rasterizace

Převod z vektorového popisu (obrazu) na rastrový popis (mřížku).  
Rastr musí mít zadanou velikost.



# Vektor

Neformálně je **vektor** prvek vektorového prostoru, který má velikost a směr.

$$\vec{v} = (v_1, \dots, v_n)$$

Velikost vektoru, někdy také nazývaná norma vektoru (Euklidovská norma)  $\|\vec{u}\|$  vektoru  $\vec{u} \in \mathbb{R}^n$  je dána jako:

$$\|\vec{u}\| = \sqrt{u_1^2 + \dots + u_n^2}.$$



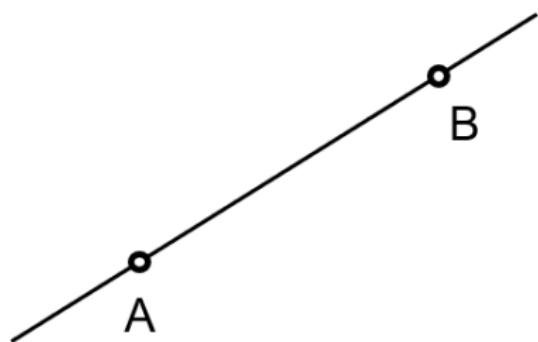
# Parametrické vyjádření přímky

Základní rovnice pro popis přímky v rovině nebo v prostoru.

$$\mathbf{X}(t) = \mathbf{A} + \vec{u}t$$

$$\mathbf{X}(t) = \mathbf{A} + (\mathbf{B} - \mathbf{A})t$$

$$\mathbf{X}(t) = (1 - t)\mathbf{A} + t\mathbf{B}$$



$$x(t) = a_x + \vec{u}_x t$$
$$y(t) = a_y + \vec{u}_y t$$

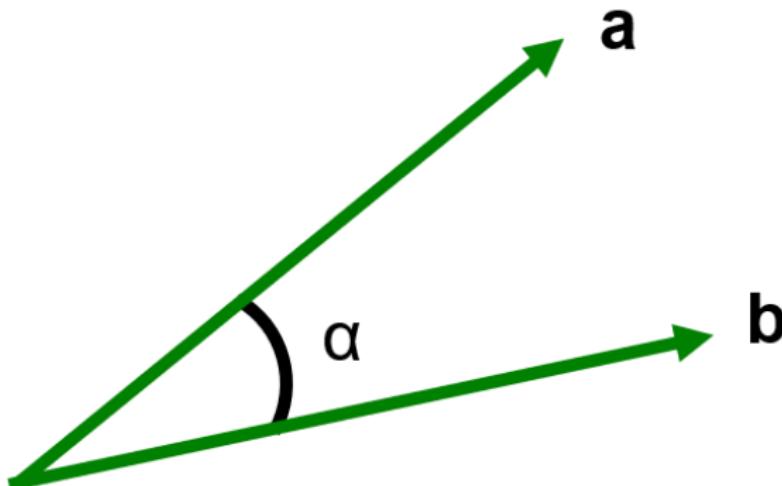
Jak je zadaná přímka, úsečka, polopřímka?

## Skalární součin

Definuje se mezi dvěma vektory a zachycuje vztah mezi velikostí těchto vektorů a jejich úhlem.

$$\vec{a} \cdot \vec{b} = |\vec{a}| |\vec{b}| \cos \alpha$$

$$\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2 + a_3 b_3$$

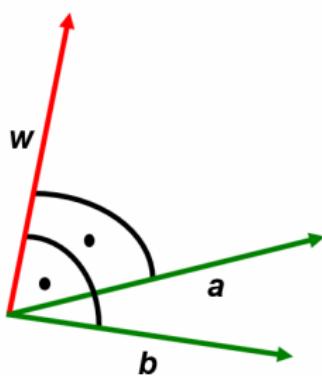


# Vektorový součin

Vektor kolmý k vstupním vektorům s velikostí, která je rovna obsahu rovnoběžníku, který oba vektory určují.

$$\vec{a} \times \vec{b} = \begin{vmatrix} w_1 & w_2 & w_3 \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix}$$

$$\vec{a} \times \vec{b} = (a_2 b_3 - a_3 b_2, a_3 b_1 - a_1 b_3, a_1 b_2 - a_2 b_1)$$



- J. Žára a kol.: Počítačová grafika. Computer Press, 2005, ISBN 80-251-0454-0
- E. Sojka, M. Němec, T. Fabián: Matematické principy PG. Ostrava
- Další informace: <http://lms.vsb.cz>

**Dotazy?**