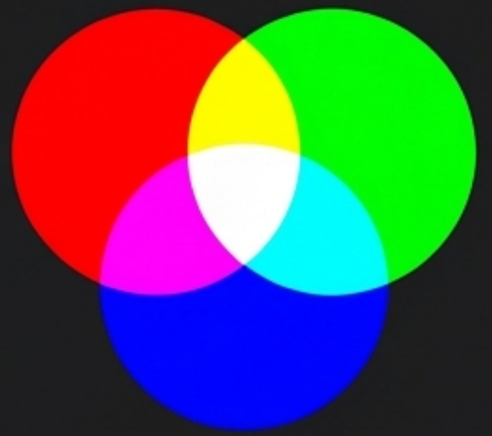
Jak hacknout realitu: Magie obrázkových filtrů

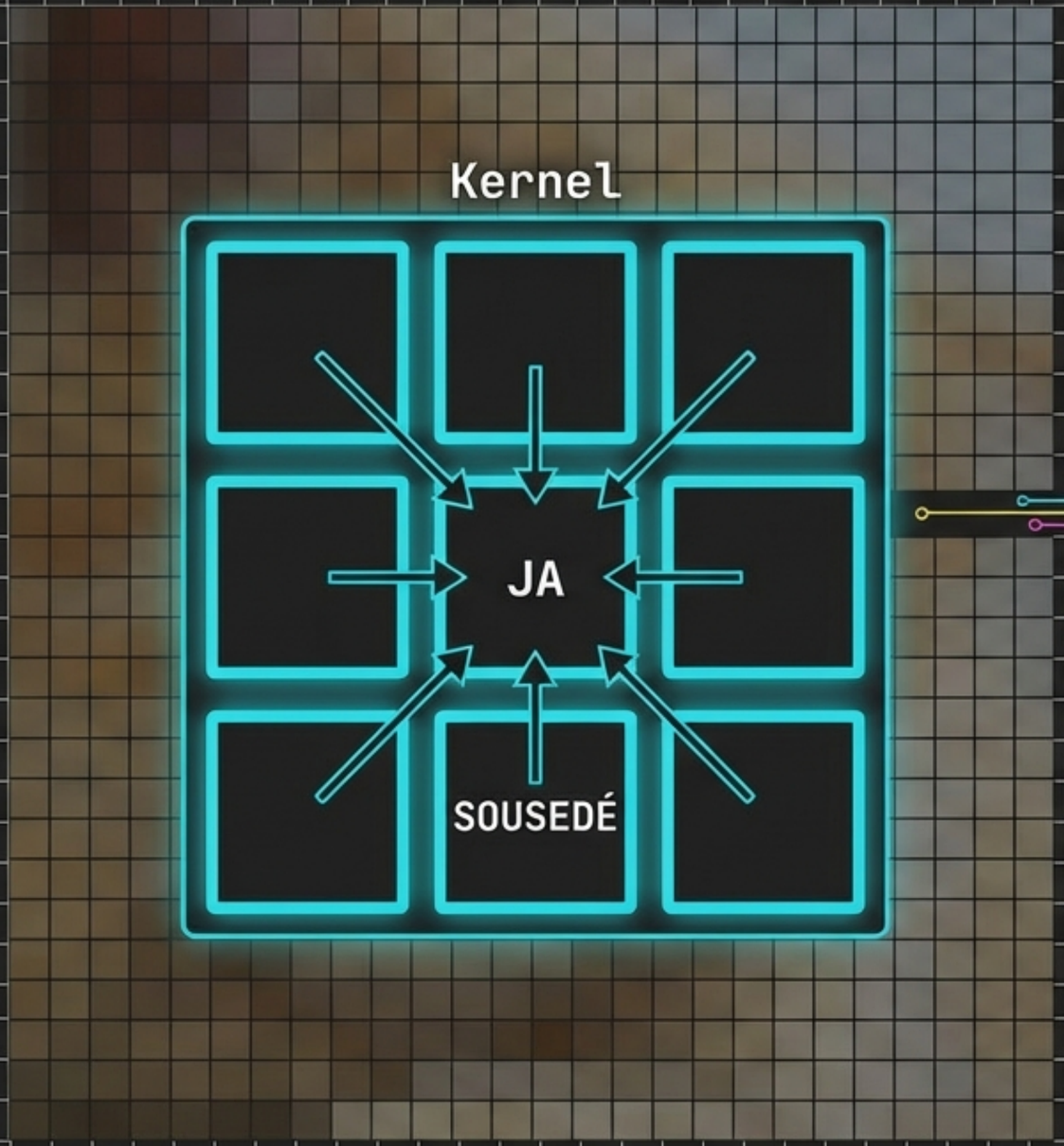
Od pixelů k Python kódu – co se děje uvnitř Instagramu?

Filtry nejsou kouzla. Je to matematika. Dnes se podíváme ‘pod kapotu’ a naučíme se, jak tyto efekty naprogramovat v Pythonu pomocí knihovny Pillow.

Co je to pixel? (Atom digitálního světa)

- Počítač nevidí 'kočku' ani 'tramvaj'. Vidí jen mřížku čísel.
- Každý bod (pixel) má svou adresu [x, y] a barvu.
- RGB Model: Barva je míchána ze tří složek – Červená (Red), Zelená (Green), Modrá (Blue).
- Bílá = (255, 255, 255), Černá = (0, 0, 0).

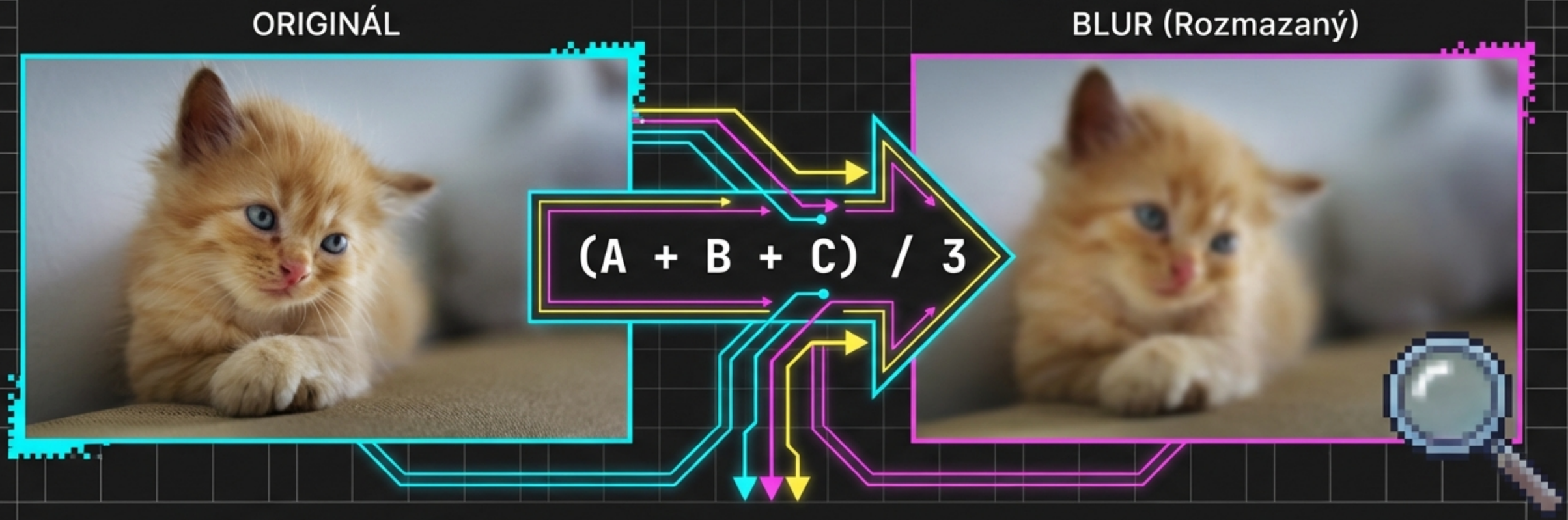




Tajemství je v sousedech (Konvoluce)

- Jak změníme celý obrázek? Pixel po pixelu.
- Ale pixel se nemění náhodně. Rozhoduje se podle svých sousedů.
- Tomuto pravidlu říkáme Kernel (nebo “Plovoucí okno”).
- Analogie: Je to jako tlak okolí. Pokud jsou všichni tví sousedé bílí, pravděpodobně budeš muset být bílý taky.

Filtr 1: BLUR (Rozmazání)



Princip: Průměrování (Averaging).

- Pokud je pixel červený a sousedé bílí, smícháme je do růžové.
- Tím se ztratí ostré hrany a obrázek se zjemní.
- V Pythonu: **ImageFilter.BLUR**

Filtr 2: SHARPEN (Zostření)

ORIGINÁL

SHARPEN (Zostřený)

Princip: Zvýraznění rozdílů (Contrast).

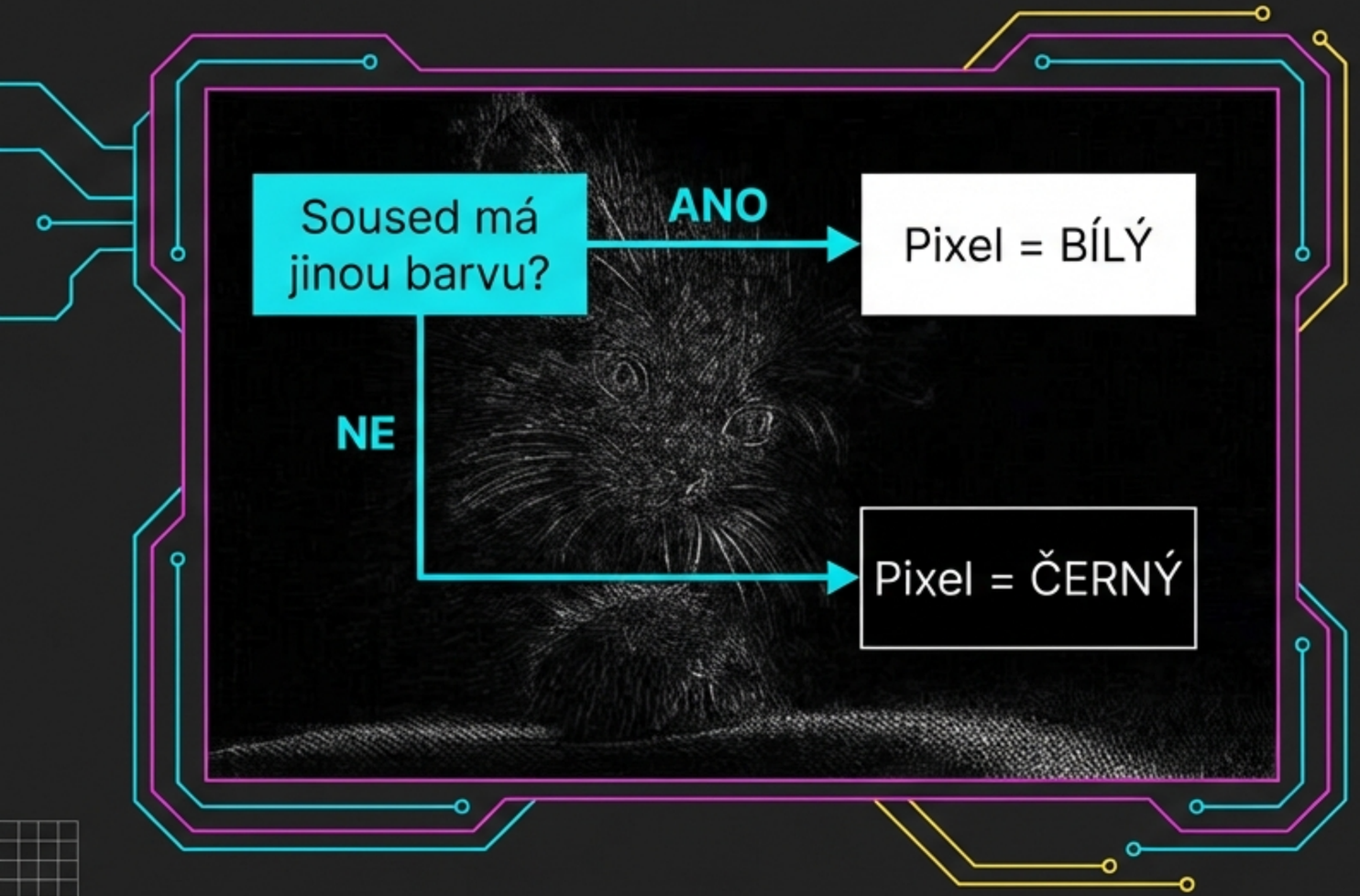
Opačný proces než rozmazání.

Pokud je pixel světlejší než soused, uděláme ho ještě světlejším. Pokud je tmavší, uděláme ho ještě tmavším.

Filtry v Pythonu:

- `ImageFilter.SHARPEN` (Jemné doostření)
- `ImageFilter.EDGE_ENHANCE_MORE` (Agresivní zvýraznění hran)

Filtr 3: FIND_EDGES (Detektiv hran)



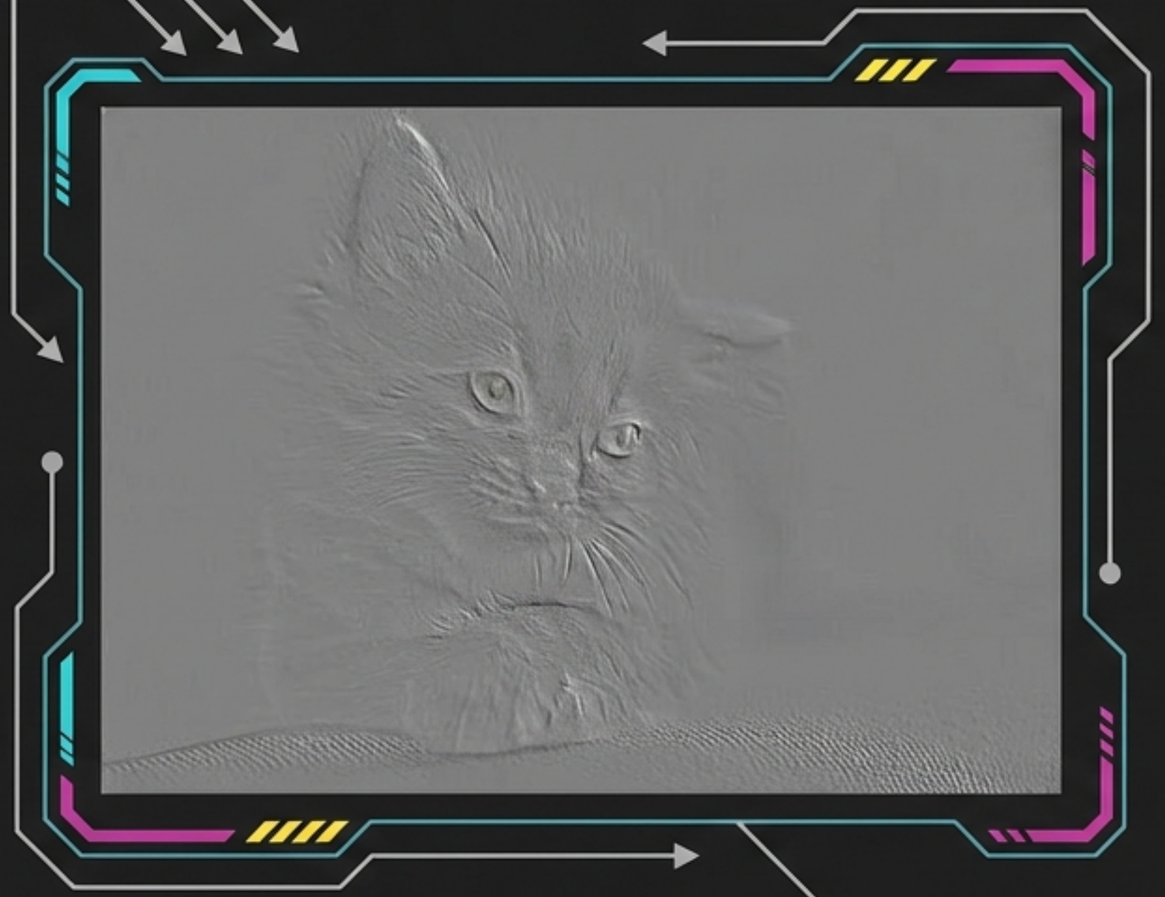
Princip: Hledání změn (Gradient).

Algoritmus se ptá: 'Je mezi mnou a sousedem velký rozdíl v barvě?'

Filtry v Pythonu:

- ImageFilter.FIND_EDGES (Vytvoří černou mapu s bílými obrysy)
- ImageFilter.CONTOUR (Vypadá jako kresba tužkou)

Filtr 4: EMBOSS (Reliéf)



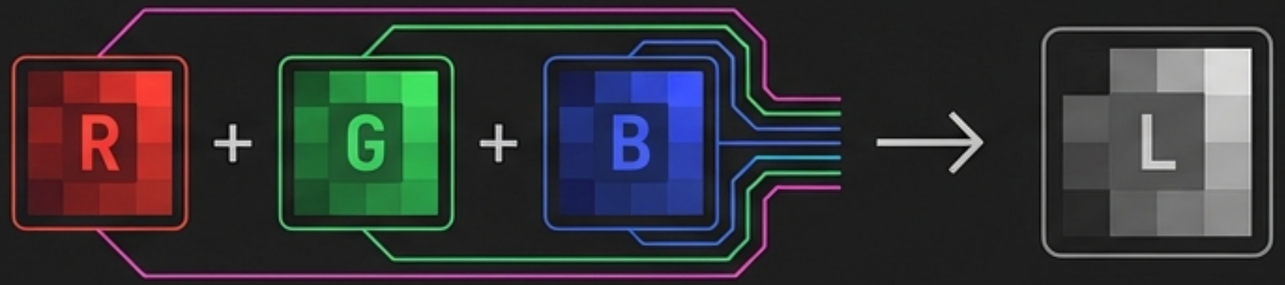
Princip: Simulace 3D světla.

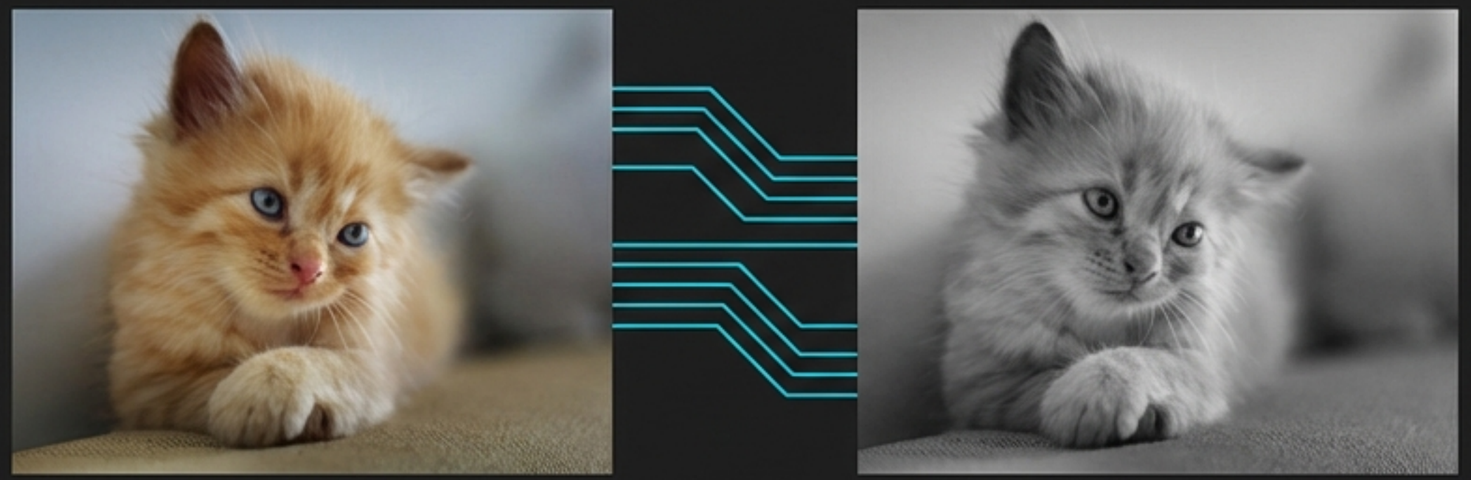
Filtr vypočítá stíny a světla, jako by na obrázek svítilo slunce z boku.

- Plochy bez změny -> Šedá.
- Hrany -> Světlá/Tmavá podle směru světla.

V Pythonu: `ImageFilter.EMBOSS`

Černobílá magie (Příprava dat)





Než začneme hledat hrany, často převedeme obrázek na odstíny šedi.

Proč? Počítač lépe analyzuje intenzitu světla (jas) než složité barvy.

`Kód: image.convert('L')`

Režim "L" znamená Luminance (Jas) – hodnoty **0** (černá) až 255 (bílá).

Jak to napsat v Pythonu?

S knihovnou Pillow (PIL) nepotřebujete počítat matice ručně. Stačí 3 řádky kódu.

1. Načíst knihovnu a obrázek.
2. Aplikovat filtr.
3. Zobrazit výsledek.

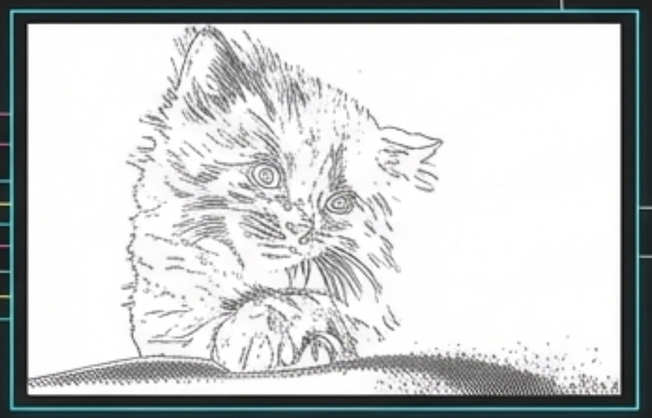
```python
from PIL import Image, ImageFilter

# 1. Otevři obrázek
img = Image.open('kocka.jpg')

# 2. Aplikuj filtr
img_upraveny = img.filter(ImageFilter.FIND_EDGES)

# 3. Zobraz výsledek
img_upraveny.show()
```

Galerie efektů: Porovnání

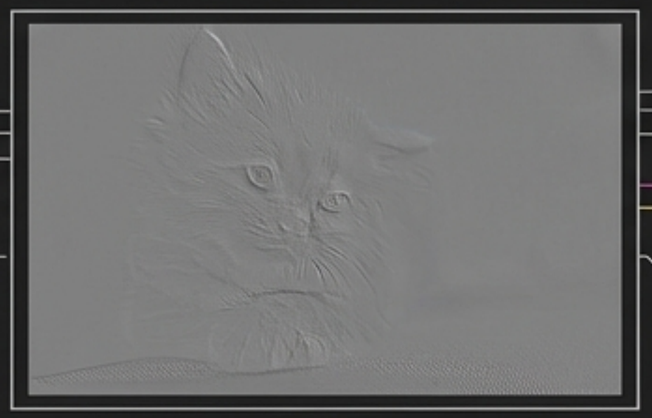
BLUR

SHARPEN

EDGE_ENHANCE_MORE



CONTOUR

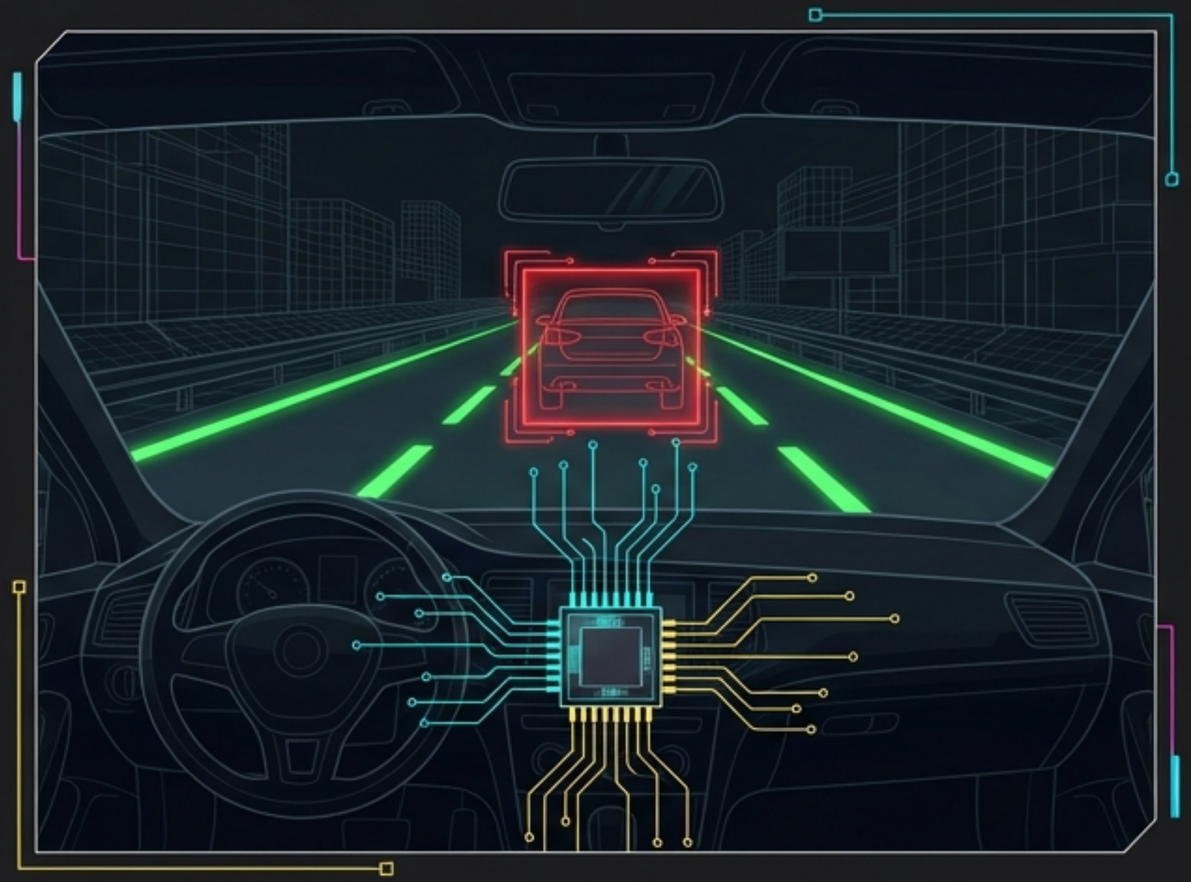


EMBOSS

FIND_EDGES

Každý filtr používá jinou 'matematickou mřížku' (Kernel), ale vstupní data jsou stejná.

K čemu to je dobré? (Víc než jen Instagram)



Tyto jednoduché filtry jsou základem pro Umělou inteligenci a Počítačové vidění.

- **Autonomní auta:** Používají detekci hran (FIND_EDGES), aby viděla pruhy na silnici.
- **Rozpoznávání obličejů:** Používá zjednodušení a kontury, aby našla oči a ústa.
- **Lékařství:** Zostření rentgenových snímků pro lepší diagnózu.

Shrnutí · Shrnutí & Výzva

Pixel (Data) Kernel (Filtr) Pillow (Výsledek)

1. Pixel je jen číslo v mřížce.
2. Filtr (Kernel) se dívá na sousedy a počítá novou barvu.
3. Pillow nám umožňuje měnit realitu pár řádky kódu.

Experiment: Co se stane, když filtry zkombinujete? Zkuste obrázek nejdřív rozmazat (BLUR) a teprve potom najít hrany (FIND_EDGES). Jak se výsledek změní?