Robotika: Základy vidění

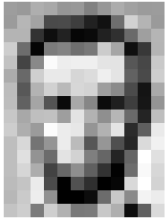
Vladimír Petrík

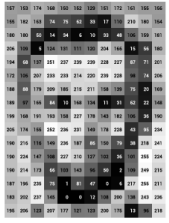
vladimir.petrik@cvut.cz

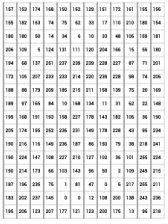
23.10.2023

Co je to obrázek?

- Kamera připojená k počítači vytváří obrázky
- Obrázek je pole čísel[1]



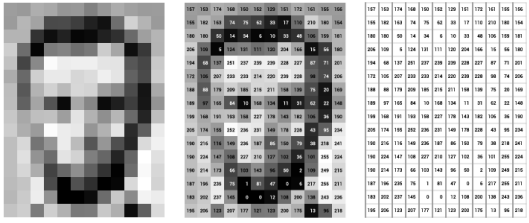


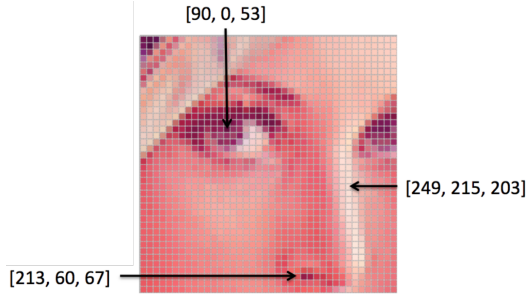


[1]Obrázky jsou z: `https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html`

Co je to obrázek?

- Kamera připojená k počítači vytváří obrázky
- Obrázek je pole čísel[1]

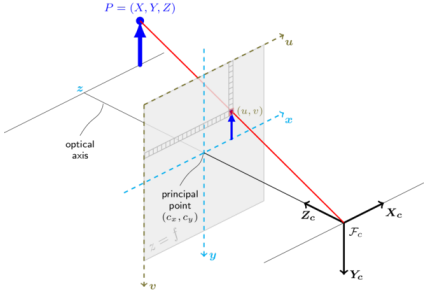




[1] Obrázky jsou z: `https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html`

Jak vzniká obraz?

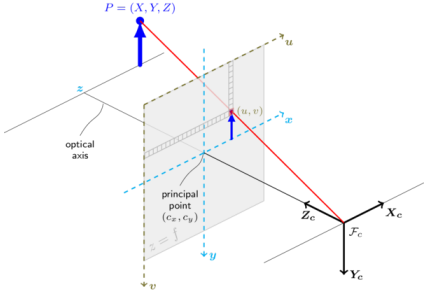
- Perspektivní kamera
  - model dírkové kamery/středové promítání[2]
  - promítá prostorový bod $\boldsymbol{x}_c$ do obrazového bodu $\boldsymbol{u} = \begin{pmatrix} u & v \end{pmatrix}^\top$ protnutím
    - obrazové roviny a
    - přímky spojující $\boldsymbol{x}_c$ se středem projekce
  - všechny body na paprsku se promítají do stejného pixelu

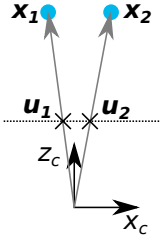


[2] docs.opencv.org

Jak vzniká obraz?

- Perspektivní kamera
  - model dírkové kamery/středové promítání[2]
  - promítá prostorový bod $\boldsymbol{x}_c$ do obrazového bodu $\boldsymbol{u} = \begin{pmatrix} u & v \end{pmatrix}^\top$ protnutím
    - obrazové roviny a
    - přímky spojující $\boldsymbol{x}_c$ se středem projekce
  - všechny body na paprsku se promítají do stejného pixelu





[2] `docs.opencv.org`

Projekce

- $\boldsymbol{u}_H = K\boldsymbol{x}_c$
  - $\boldsymbol{u}_H$ je pixel v homogenních souřadnicích
  - pokud $\boldsymbol{u}_H = \begin{pmatrix} u_H & v_H & w_H \end{pmatrix}^\top$, pak souřadnice pixelu jsou $\begin{pmatrix} u_H/w_H & v_H/w_H \end{pmatrix}^\top$

Projekce

- $\boldsymbol{u}_H = K\boldsymbol{x}_c$
  - $\boldsymbol{u}_H$ je pixel v homogenních souřadnicích
  - pokud $\boldsymbol{u}_H = \begin{pmatrix} u_H & v_H & w_H \end{pmatrix}^\top$, pak souřadnice pixelu jsou $\begin{pmatrix} u_H/w_H & v_H/w_H \end{pmatrix}^\top$
  - alternativně můžeme reprezentovat jako: $\lambda \begin{pmatrix} u, v, 1 \end{pmatrix}^\top = K\boldsymbol{x}_c$

Projekce

- $\boldsymbol{u}_H = K\boldsymbol{x}_c$
  - $\boldsymbol{u}_H$ je pixel v homogenních souřadnicích
  - pokud $\boldsymbol{u}_H = \begin{pmatrix} u_H & v_H & w_H \end{pmatrix}^\top$, pak souřadnice pixelu jsou $\begin{pmatrix} u_H/w_H & v_H/w_H \end{pmatrix}^\top$
  - alternativně můžeme reprezentovat jako: $\lambda \begin{pmatrix} u, v, 1 \end{pmatrix}^\top = K\boldsymbol{x}_c$
- K je matice kamery
  - $K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$

Projekce

- $\boldsymbol{u}_H = K\boldsymbol{x}_c$
  - $\boldsymbol{u}_H$ je pixel v homogenních souřadnicích
  - pokud $\boldsymbol{u}_H = \begin{pmatrix} u_H & v_H & w_H \end{pmatrix}^\top$, pak souřadnice pixelu jsou $\begin{pmatrix} u_H/w_H & v_H/w_H \end{pmatrix}^\top$
  - alternativně můžeme reprezentovat jako: $\lambda \begin{pmatrix} u, v, 1 \end{pmatrix}^\top = K\boldsymbol{x}_c$
- K je matice kamery
  - $K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$
  - Co představuje λ?

Projekce

- $\boldsymbol{u}_H = K\boldsymbol{x}_c$
  - $\boldsymbol{u}_H$ je pixel v homogenních souřadnicích
  - pokud $\boldsymbol{u}_H = \begin{pmatrix} u_H & v_H & w_H \end{pmatrix}^\top$, pak souřadnice pixelu jsou $\begin{pmatrix} u_H/w_H & v_H/w_H \end{pmatrix}^\top$
  - alternativně můžeme reprezentovat jako: $\lambda \begin{pmatrix} u, v, 1 \end{pmatrix}^\top = K\boldsymbol{x}_c$
- K je matice kamery
  - $K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$
  - Co představuje λ?
    - λ je nenulové reálné číslo
    - pokud znáte hodnotu λ, můžete vypočítat kartézskou souřadnici $\boldsymbol{x} = \lambda K^{-1}\boldsymbol{u}$
      $\boldsymbol{x} = \lambda K^{-1}\boldsymbol{u}$
    - v opačném případě lze vypočítat pouze paprsek

Projekce

- $\boldsymbol{u}_H = K\boldsymbol{x}_c$
  - $\boldsymbol{u}_H$ je pixel v homogenních souřadnicích
  - pokud $\boldsymbol{u}_H = \begin{pmatrix} u_H & v_H & w_H \end{pmatrix}^\top$, pak souřadnice pixelu jsou $\begin{pmatrix} u_H/w_H & v_H/w_H \end{pmatrix}^\top$
  - alternativně můžeme reprezentovat jako: $\lambda \begin{pmatrix} u, v, 1 \end{pmatrix}^\top = K\boldsymbol{x}_c$
- K je matice kamery
  - $K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$
  - Co představuje λ?
    - λ je nenulové reálné číslo
    - pokud znáte hodnotu λ, můžete vypočítat kartézskou souřadnici $\boldsymbol{x} = \lambda K^{-1}\boldsymbol{u}$
      $\boldsymbol{x} = \lambda K^{-1}\boldsymbol{u}$
    - v opačném případě lze vypočítat pouze paprsek
  - jak zjistit K z bodů?

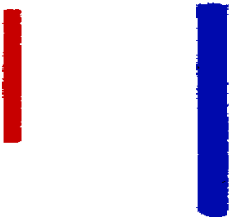
Co můžeme studovat na obrázcích?

Co můžeme studovat na obrázcích?

- Segmentační masky (kde jsou objekty zájmu)

Co můžeme studovat na obrázcích?

- Segmentační masky (kde jsou objekty zájmu)
- Klasifikace objektů (označování)



Segmentační masky - prahování barev

- Prahování
  - hodnoty pixelů RGB pro souřadnice $\boldsymbol{u}$: $I_{\mathsf{RGB}}(\boldsymbol{u})$

Segmentační masky - prahování barev

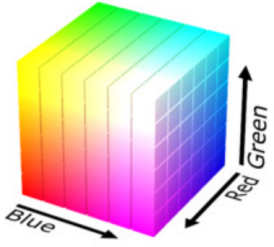
- Prahování
  - hodnoty pixelů RGB pro souřadnice $\boldsymbol{u}$: $I_{\mathsf{RGB}}(\boldsymbol{u})$
  - $M(\boldsymbol{u}) = 1,$ pokud $I_{\mathsf{RGB}}(\boldsymbol{u}) = \begin{pmatrix} 0 & 255 & 0 \end{pmatrix}^\top$?

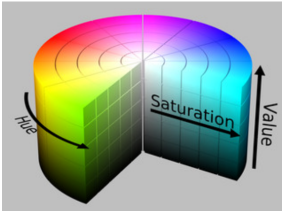
Segmentační masky - prahování barev

- Prahování
  - hodnoty pixelů RGB pro souřadnice $\boldsymbol{u}$: $I_{\mathsf{RGB}}(\boldsymbol{u})$
  - $M(\boldsymbol{u}) = 1$, pokud $I_{\mathsf{RGB}}(\boldsymbol{u}) = \begin{pmatrix} 0 & 255 & 0 \end{pmatrix}^\top$?
  - $M(\boldsymbol{u}) = 1$, pokud $\boldsymbol{\tau}_l < I_{\mathsf{RGB}}(\boldsymbol{u}) < \boldsymbol{\tau}_u$, pro všechny kanály

Segmentační masky - prahování barev

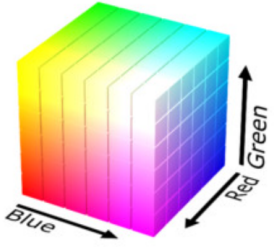
- Prahování
  - hodnoty pixelů RGB pro souřadnice $\boldsymbol{u}$: $I_{\mathsf{RGB}}(\boldsymbol{u})$
  - $M(\boldsymbol{u}) = 1,$ pokud $I_{\mathsf{RGB}}(\boldsymbol{u}) = \begin{pmatrix} 0 & 255 & 0 \end{pmatrix}^\top$?
  - $M(\boldsymbol{u}) = 1,$ pokud $\boldsymbol{\tau}_l < I_{\mathsf{RGB}}(\boldsymbol{u}) < \boldsymbol{\tau}_u$, pro všechny kanály
  - $M(\boldsymbol{u}) = 1,$ pokud $\boldsymbol{\varphi}_l < I_{\mathsf{HSV}}(\boldsymbol{u}) < \boldsymbol{\varphi}_u$, pro všechny kanály

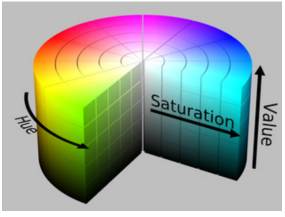




Segmentační masky - prahování barev

- Prahování
  - hodnoty pixelů RGB pro souřadnice $\boldsymbol{u}$: $I_{\mathsf{RGB}}(\boldsymbol{u})$
  - $M(\boldsymbol{u}) = 1$, pokud $I_{\mathsf{RGB}}(\boldsymbol{u}) = \begin{pmatrix} 0 & 255 & 0 \end{pmatrix}^\top$?
  - $M(\boldsymbol{u}) = 1$, pokud $\boldsymbol{\tau}_l < I_{\mathsf{RGB}}(\boldsymbol{u}) < \boldsymbol{\tau}_u$, pro všechny kanály
  - $M(\boldsymbol{u}) = 1$, pokud $\boldsymbol{\varphi}_l < I_{\mathsf{HSV}}(\boldsymbol{u}) < \boldsymbol{\varphi}_u$, pro všechny kanály
- Následné zpracování
  - výpočet spojených komponent
  - odstranit malé nebo deformované segmenty
  - přiřadit označení na základě prahových hodnot





Segmentační masky pro známé 3D objekty

- Neuronová síť (např. Mask R-CNN)

Segmentační masky pro známé 3D objekty

- Neuronová síť (např. Mask R-CNN)
- Trénovací vstupy:
  - datová sada obrázků, masek a štítků nebo

Segmentační masky pro známé 3D objekty

- Neuronová síť (např. Mask R-CNN)
- Trénovací vstupy:
  - datová sada obrázků, masek a štítků nebo
  - datová sada známých 3D objektů (mesh)

Segmentační masky pro známé 3D objekty

Segmentační masky pro známé 3D objekty

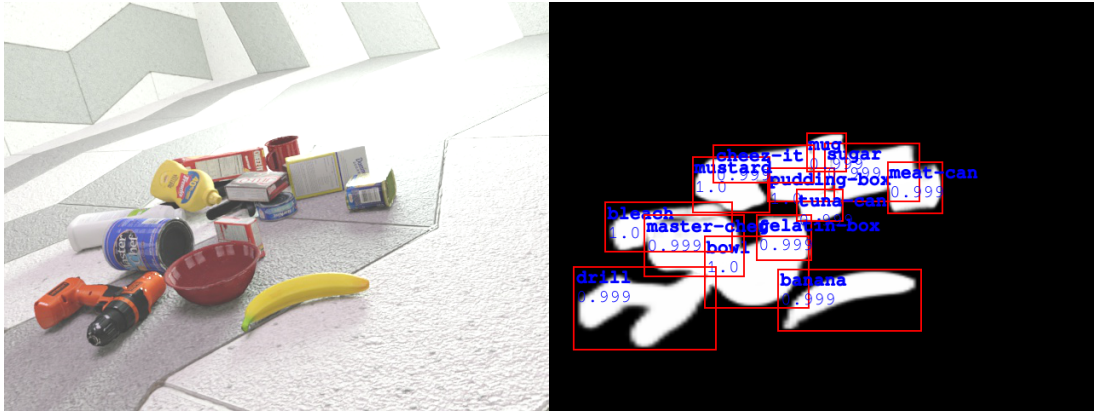
Segmentační masky pro známé 3D objekty

- Neuronová síť (např. Mask R-CNN)
- Trénovací vstupy:
  - datová sada obrázků, masek a štítků nebo
  - datová sada známých 3D objektů (mesh)
  - kvalita závisí na trénovacích datech (augumentace)

Segmentační masky pro známé 3D objekty

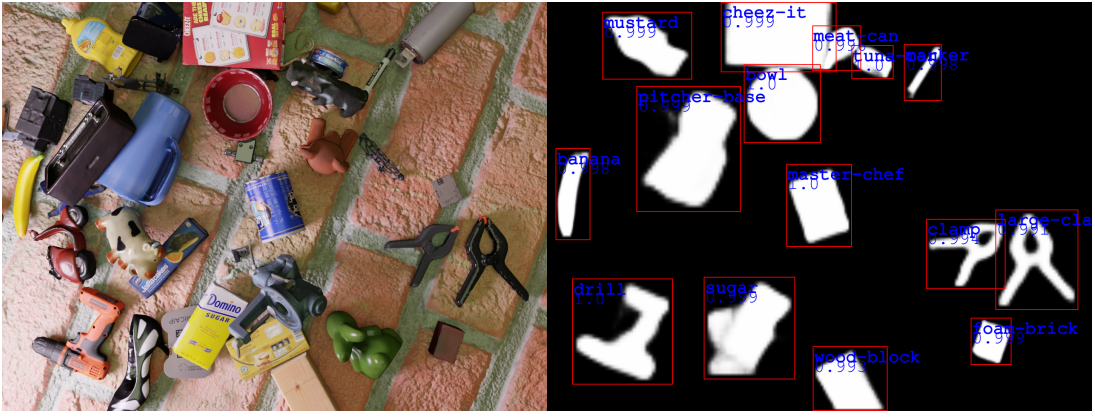
- Neuronová síť (např. Mask R-CNN)
- Trénovací vstupy:
  - datová sada obrázků, masek a štítků nebo
  - datová sada známých 3D objektů (mesh)
  - kvalita závisí na trénovacích datech (augumentace)
- Inference:
  - Vstup: obrázek
  - Výstup: segmentační maska, ohraničující rámeček, označení a konfidence

Mask R-CNN výsledky

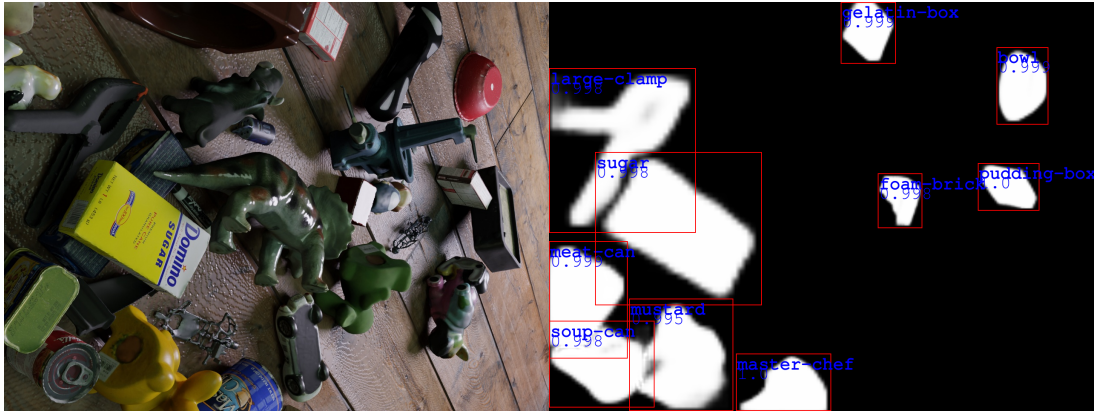


Mask R-CNN výsledky

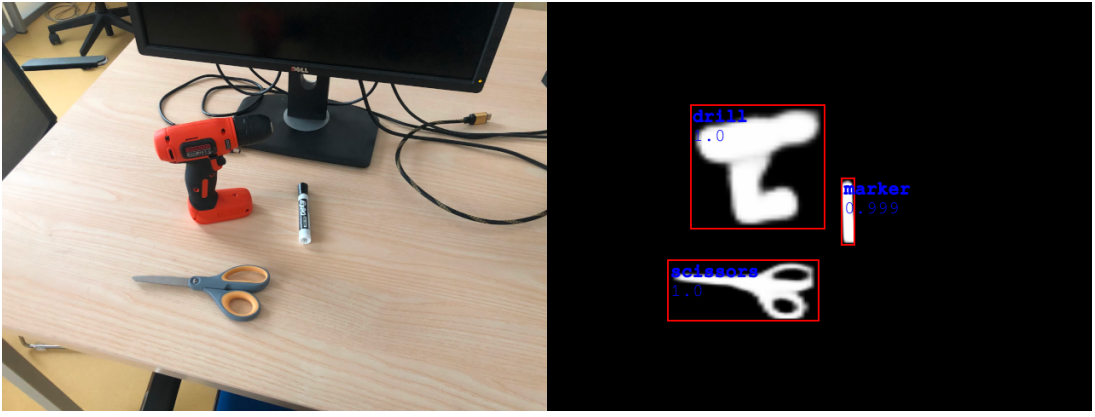
Mask R-CNN výsledky



Mask R-CNN výsledky



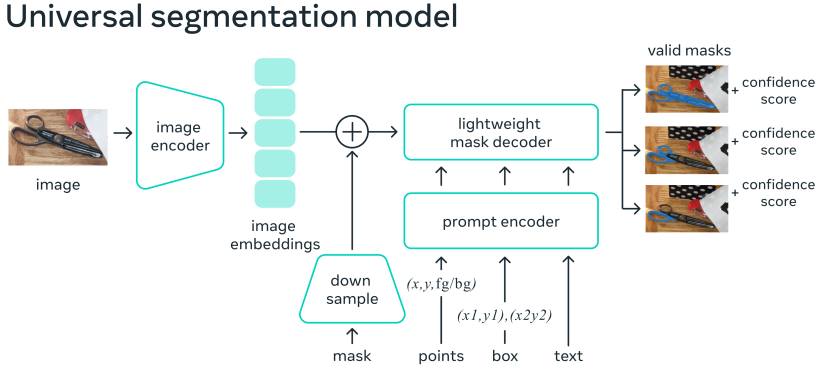
Mask R-CNN výsledky



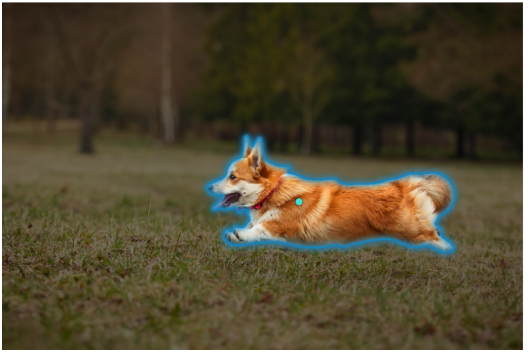
Mask R-CNN výsledky

Segmentační masky bez pretrénování

- Segment Anything Model (SAM)
  - segmentace libovolného objektu na libovolném snímku jediným kliknutím
  - datová sada 10 milionů obrázků, 1 miliarda masek



SAM výsledky



SAM výsledky

SAM výsledky

SAM výsledky

Segmentace

- Segmentace nalezne objekty v obraze
  - segmentační maska
  - bounding box (ohraničující rámeček)
  - label (označení)
  - skóre důvěryhodnosti/konfidence

Segmentace

- Segmentace nalezne objekty v obraze
  - segmentační maska
  - bounding box (ohraničující rámeček)
  - label (označení)
  - skóre důvěryhodnosti/konfidence
- Informace pouze v obrazovém prostoru
- Jak používat v prostoru robota?

Externí kamera

- Předpokládáme kameru pevně připevněnou k referenčnímu rámci
  - pokud známe K a T_{RC}, jak promítnout body $\boldsymbol{x}_R$ do obrazu?

Externí kamera

- Předpokládáme kameru pevně připevněnou k referenčnímu rámci
  - pokud známe K a T_{RC}, jak promítnout body $\boldsymbol{x}_R$ do obrazu?
- Neznámé K a T_{RC} a planární problém
  - např. kostky se stejnou výškou na stole
  - jaká je poloha kostky na 2D stole vzhledem k 2D souřadnicím obrazu/pixelů?

Externí kamera

- Předpokládáme kameru pevně připevněnou k referenčnímu rámci
  - pokud známe K a T_{RC}, jak promítnout body $\boldsymbol{x}_R$ do obrazu?
- Neznámé K a T_{RC} a planární problém
  - např. kostky se stejnou výškou na stole
  - jaká je poloha kostky na 2D stole vzhledem k 2D souřadnicím obrazu/pixelů?
  - analyzováno pomocí **homografie**

Homografie

- Homografická matice H je matice 3×3 , která mapuje body z jedné roviny do druhé
  - obrazová rovina na stůl
  - jedna obrazová rovina do jiné obrazové roviny (jiný pohled)

Homografie

- Homografická matice H je matice 3×3 , která mapuje body z jedné roviny do druhé
  - obrazová rovina na stůl
  - jedna obrazová rovina do jiné obrazové roviny (jiný pohled)
- $s \begin{pmatrix} x & y & 1 \end{pmatrix}^\top = H \begin{pmatrix} u & v & 1 \end{pmatrix}^\top$
  - x, y jsou souřadnice v první rovině
  - u, v jsou souřadnice ve druhé rovině

Homografie

- Homografická matice H je matice 3×3 , která mapuje body z jedné roviny do druhé
  - obrazová rovina na stůl
  - jedna obrazová rovina do jiné obrazové roviny (jiný pohled)
- $s \begin{pmatrix} x & y & 1 \end{pmatrix}^\top = H \begin{pmatrix} u & v & 1 \end{pmatrix}^\top$
  - x, y jsou souřadnice v první rovině
  - u, v jsou souřadnice ve druhé rovině
- 9 prvků, ale pouze 8 DoF, obvykle se přidává omezení $h_{33} = 1$
- Jak najít H?

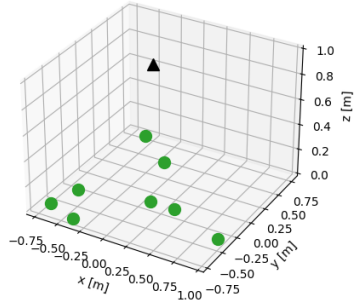
Homografie

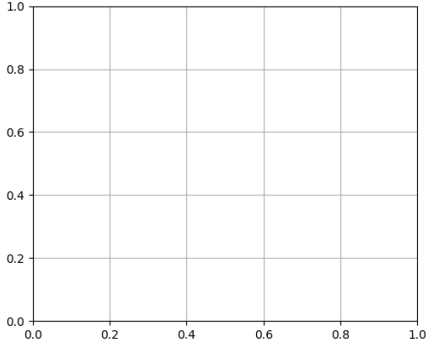
- Homografická matice H je matice 3×3 , která mapuje body z jedné roviny do druhé
  - obrazová rovina na stůl
  - jedna obrazová rovina do jiné obrazové roviny (jiný pohled)
- $s \begin{pmatrix} x & y & 1 \end{pmatrix}^\top = H \begin{pmatrix} u & v & 1 \end{pmatrix}^\top$
  - x, y jsou souřadnice v první rovině
  - u, v jsou souřadnice ve druhé rovině
- 9 prvků, ale pouze 8 DoF, obvykle se přidává omezení $h_{33} = 1$
- Jak najít H?
  - `H, _ = cv2.findHomography(U, X)`
  - U, X jsou $N \times 2$ korespondenční body

Homografie

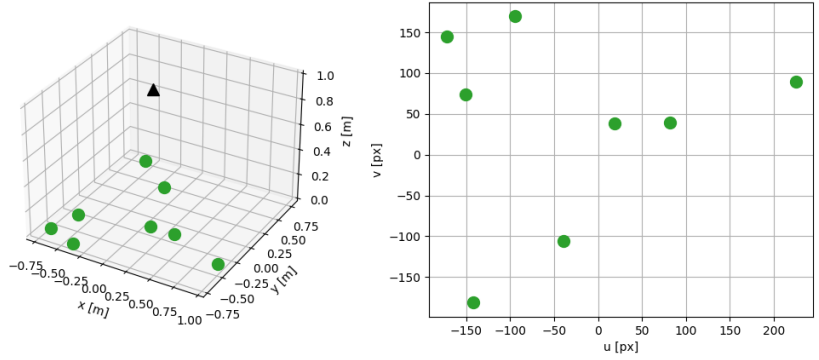
- Homografická matice H je matice 3×3 , která mapuje body z jedné roviny do druhé
  - obrazová rovina na stůl
  - jedna obrazová rovina do jiné obrazové roviny (jiný pohled)
- $s \begin{pmatrix} x & y & 1 \end{pmatrix}^\top = H \begin{pmatrix} u & v & 1 \end{pmatrix}^\top$
  - x, y jsou souřadnice v první rovině
  - u, v jsou souřadnice ve druhé rovině
- 9 prvků, ale pouze 8 DoF, obvykle se přidává omezení $h_{33} = 1$
- Jak najít H?
  - `H, _ = cv2.findHomography(U, X)`
  - U, X jsou $N \times 2$ korespondenční body
  - např. ručně změříme
    - poloha středu krychle vůči rohu stolu
    - pozice středu krychle v obrázku

Příklad homografie

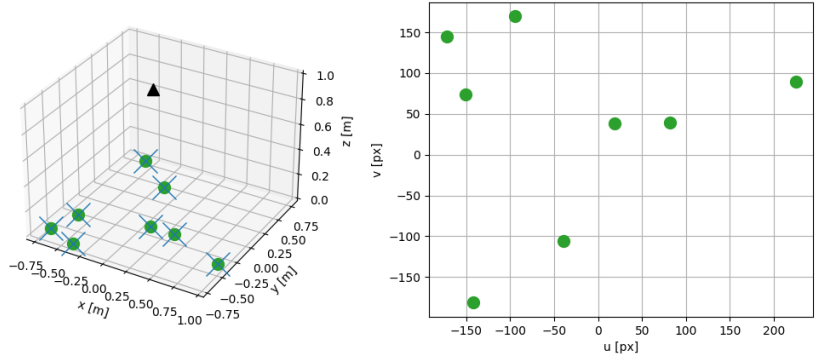




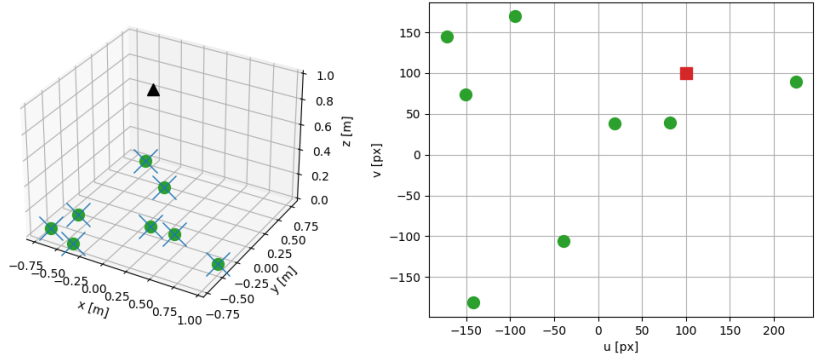
Příklad homografie



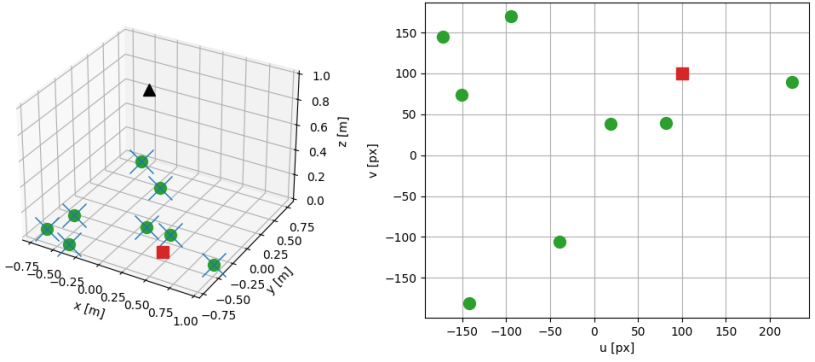
Příklad homografie



Příklad homografie



Příklad homografie

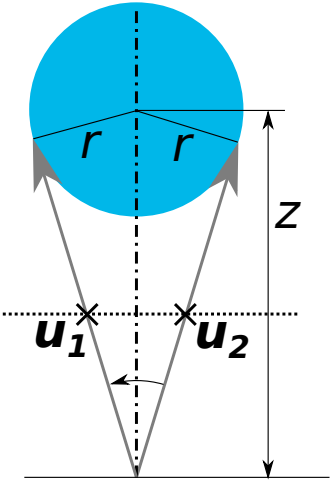


Neplanární odhad polohy/orientace

- Homografie mapuje pouze rovinu na rovinu
- Obecnější odhad polohy objektu ve s.s **kamery**
  - získání hloubky mapováním z plochy v pixelech na hloubku pro objekty pevné velikosti
  - získání hloubky pomocí dodatečných informací o scéně, např. známá velikost/model objektu
  - RGBD kamera
  - dodatečné značky (markery)

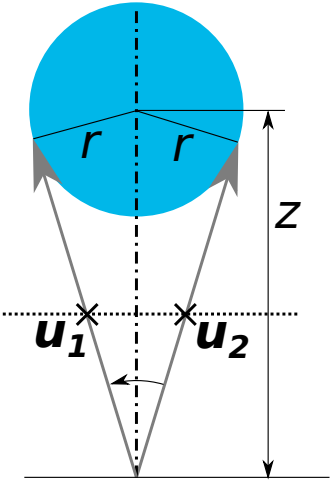
Využití znalostí o velikosti

- Víme, že poloměr je fixní



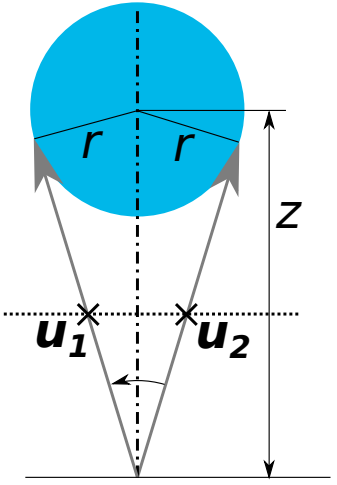
Využití znalostí o velikosti

- Víme, že poloměr je fixní
- Z detekovaných pixelů $\boldsymbol{u}_1, \boldsymbol{u}_2$, můžeme vypočítat paprsky $\boldsymbol{x}_1, \boldsymbol{x}_2$: $\frac{1}{\lambda_i}\boldsymbol{x}_i = K^{-1}\boldsymbol{u}_i$



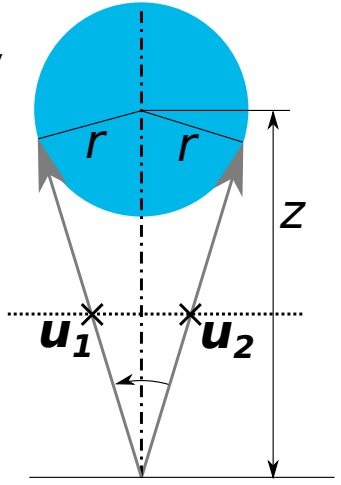
Využití znalostí o velikosti

- Víme, že poloměr je fixní
- Z detekovaných pixelů $\boldsymbol{u}_1, \boldsymbol{u}_2$, můžeme vypočítat paprsky $\boldsymbol{x}_1, \boldsymbol{x}_2$: $\frac{1}{\lambda_i}\boldsymbol{x}_i = K^{-1}\boldsymbol{u}_i$
- Úhel mezi vektory: $\cos\alpha = \frac{\frac{1}{\lambda_1\lambda_2}}{\frac{1}{\lambda_1\lambda_2}} \frac{\boldsymbol{x}_1 \cdot \boldsymbol{x}_2}{\|\boldsymbol{x}_1\|\|\boldsymbol{x}_2\|}$



Využití znalostí o velikosti

- Víme, že poloměr je fixní
- Z detekovaných pixelů $\boldsymbol{u}_1, \boldsymbol{u}_2$, můžeme vypočítat paprsky $\boldsymbol{x}_1, \boldsymbol{x}_2$: $\frac{1}{\lambda_i}\boldsymbol{x}_i = K^{-1}\boldsymbol{u}_i$
- Úhel mezi vektory: $\cos\alpha = \frac{\frac{1}{\lambda_1\lambda_2}}{\frac{1}{\lambda_1\lambda_2}} \frac{\boldsymbol{x}_1 \cdot \boldsymbol{x}_2}{\|\boldsymbol{x}_1\|\|\boldsymbol{x}_2\|}$
- Hloubka: $z = \frac{r}{\sin(\alpha/2)}$

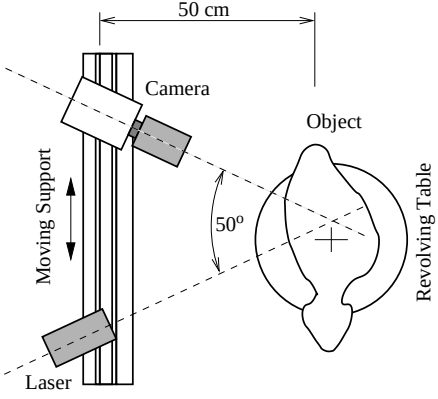


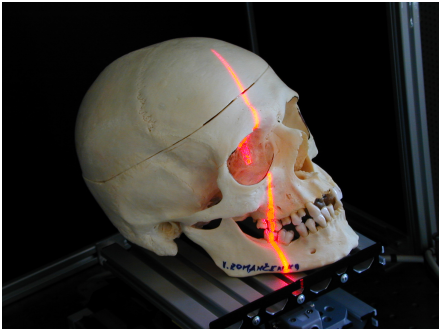
Použití hloubkového senzoru

- RGB**D** senzory
  - RGB obrázek $(H \times W \times 3)$
  - Hloubková mapa $(H \times W \times 1)$, vzdálenost v metrech pro každý pixel
  - Strukturované mračno bodů $(H \times W \times 3)$, $\begin{pmatrix} x_c & y_c & z_c \end{pmatrix}$ pro každý pixel

Jak funguje hloubkový senzor

- Laser promítá vzor a kamera ho rozpozná
- Informace o hloubce se vypočítá pomocí triangulace



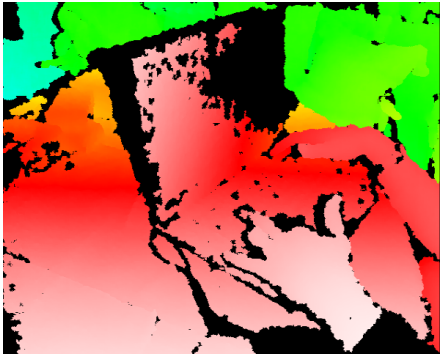


2D hloubkové senzory

- Na základě strukturovaného světla
- Projektuje 2D infračervené vzory
- Jeden projektor a dvě kamery (RGB + IR)

Problémy se snímači hloubky

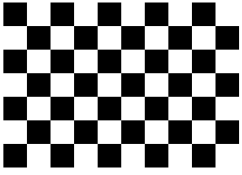
- Rekonstrukce hloubky není dokonalá (černé oblasti na snímku[3])
- V jazyce python reprezentováno hodnotou NaN
- Ne každý pixel v RGB má rekonstruovanou hodnotu hloubky
- Data RGB a hloubky nejsou zarovnána (je třeba je kalibrovat)

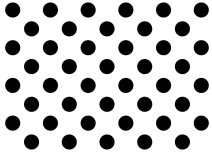


[3] https://commons.wikimedia.org, User:Kolossos

Dodatečné značky

- Můžeme vypočítat polohu/orientaci obrazců[4]?

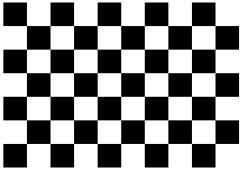


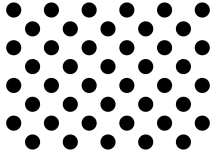


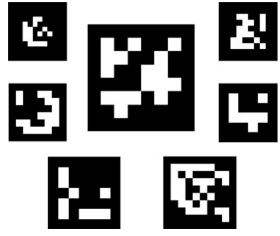
[4] docs.opencv.org

Dodatečné značky

- Můžeme vypočítat polohu/orientaci obrazců[4]?
  - je třeba znát velikost a strukturu
  - subpixelová přesnost
  - musí být zcela viditelný
- Můžeme vypočítat polohu značek ArUco?



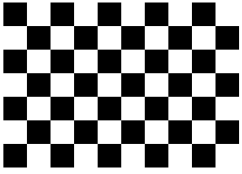


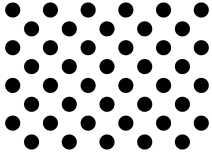


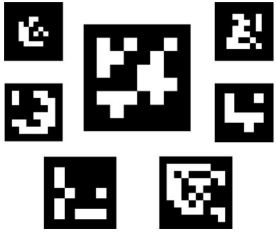
[4] docs.opencv.org

Dodatečné značky

- Můžeme vypočítat polohu/orientaci obrazců[4]?
  - je třeba znát velikost a strukturu
  - subpixelová přesnost
  - musí být zcela viditelný
- Můžeme vypočítat polohu značek ArUco?
  - méně přesné než pravidelné obrazce
  - poskytuje identifikátor značky a pózu
  - musí být zcela viditelný

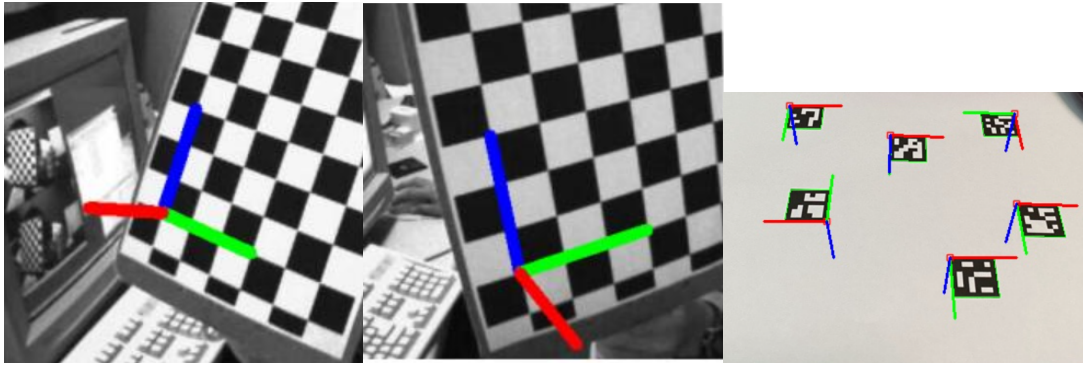






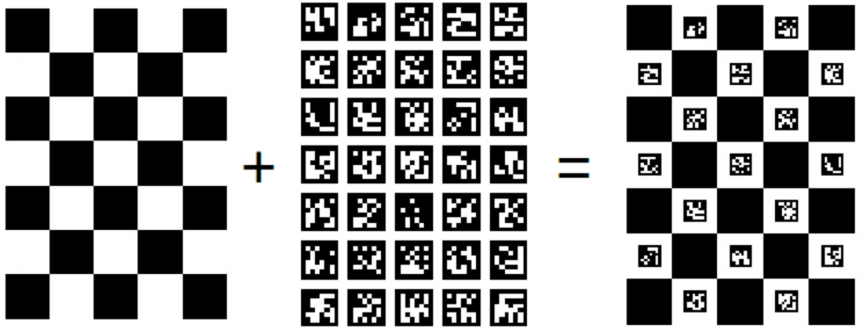
[4] docs.opencv.org

Příklad umístění značek



ChArUco deska pro kalibraci

- Kombinuje přesnost pravidelného vzoru s detekcí ArUco
- Detekce dílčích částí / částečná viditelnost



Charuco definition

Odhad matice kamery pomocí kalibračních desek

- Můžeme odhadnout matici kamery z korespondencí v obrazovém a prostorovém prostoru
  - nasbíráme snímky tabule z různých pohledů
  - detekujeme desky
  - vypočítáme korespondence mezi obrazovými body a body s.s. tabule
  - ```
    _, K, dist_coeffs, rvecs, tvecs = cv2.calibrateCamera(
        obj_points, img_points, img_shape)
    ```
- Kromě toho získáme
  - koeficienty zkreslení, které kompenzují vady objektivu

    ```
    Knew, roi = cv.getOptimalNewCameraMatrix(K, dist_coeffs,
        img_shape, 1, img_shape)
    img_undistorted = cv.undistort(img, K, dist_coeffs, None, Knew)
    ```
  - $SE(3)$ pozice desek v s.s. kamery
    ```

Odhad pozice z RGB(D)

- Metody odhadu polohy
  - používají předchozí znalosti o úloze, např. objekty s pevnou výškou v rovině
  - používají předchozí znalosti o objektech (velikost)
  - používají hloubkový senzor
  - používají značky ArUco

Odhad pozice z RGB(D)

- Metody odhadu polohy
  - používají předchozí znalosti o úloze, např. objekty s pevnou výškou v rovině
  - používají předchozí znalosti o objektech (velikost)
  - používají hloubkový senzor
  - používají značky ArUco
- Kde je robot?

Odhad pozice z RGB(D)

- Metody odhadu polohy
  - používají předchozí znalosti o úloze, např. objekty s pevnou výškou v rovině
  - používají předchozí znalosti o objektech (velikost)
  - používají hloubkový senzor
  - používají značky ArUco
- Kde je robot?
  - homografie odhaduje polohy objektů vzhledem k s.s. roviny

Odhad pozice z RGB(D)

- Metody odhadu polohy
  - používají předchozí znalosti o úloze, např. objekty s pevnou výškou v rovině
  - používají předchozí znalosti o objektech (velikost)
  - používají hloubkový senzor
  - používají značky ArUco
- Kde je robot?
  - homografie odhaduje polohy objektů vzhledem k s.s. roviny
  - ostatní metody odhadují polohy v s.s. kamery

Odhad pozice z RGB(D)

- Metody odhadu polohy
  - používají předchozí znalosti o úloze, např. objekty s pevnou výškou v rovině
  - používají předchozí znalosti o objektech (velikost)
  - používají hloubkový senzor
  - používají značky ArUco
- Kde je robot?
  - homografie odhaduje polohy objektů vzhledem k s.s. roviny
  - ostatní metody odhadují polohy v s.s. kamery
  - potřebujeme odhadnout/kalibrovat T_{RC}

Kalibrace RukaOko (HandEye)

- Kamera může být namontována na
  - základně robota (kalibrace oko-ruka, eye-to-hand)
  - chapadle (kalibrace oko-v-ruce, eye-in-hand)

Kalibrace RukaOko (HandEye)

- Kamera může být namontována na
  - základně robota (kalibrace oko-ruka, eye-to-hand)
  - chapadle (kalibrace oko-v-ruce, eye-in-hand)
- Řešíme $A^i X = Y B^i$
  - měření: $A^i, B^i \in SE(3)$
  - odhadnuté parametry: $X, Y \in SE(3)$

Kalibrace RukaOko (HandEye)

- Kamera může být namontována na
  - základně robota (kalibrace oko-ruka, eye-to-hand)
  - chapadle (kalibrace oko-v-ruce, eye-in-hand)
- Řešíme $A^i X = Y B^i$
  - měření: $A^i, B^i \in SE(3)$
  - odhadnuté parametry: $X, Y \in SE(3)$
- `X, Y = calibrateRobotWorldHandEye(A, B)`

Kalibrace RukaOko (HandEye)

- Kamera může být namontována na
  - základně robota (kalibrace oko-ruka, eye-to-hand)
  - chapadle (kalibrace oko-v-ruce, eye-in-hand)
- Řešíme $A^i X = Y B^i$
  - měření: $A^i, B^i \in SE(3)$
  - odhadnuté parametry: $X, Y \in SE(3)$
- `X, Y = calibrateRobotWorldHandEye(A, B)`
- Kalibrace oko-ruka, eye-to-hand
  - $A^i = T^i_{\mathsf{RG}}$
  - $B^i = T^i_{\mathsf{CT}}$
  - $X = T_{\mathsf{GT}}$
  - $Y = T_{\mathsf{RC}}$

Kalibrace RukaOko (HandEye)

- Kamera může být namontována na
  - základně robota (kalibrace oko-ruka, eye-to-hand)
  - chapadle (kalibrace oko-v-ruce, eye-in-hand)
- Řešíme $A^i X = Y B^i$
  - měření: $A^i, B^i \in SE(3)$
  - odhadnuté parametry: $X, Y \in SE(3)$
- `X, Y = calibrateRobotWorldHandEye(A, B)`
- Kalibrace oko-ruka, eye-to-hand
  - $A^i = T^i_{\mathsf{RG}}$
  - $B^i = T^i_{\mathsf{CT}}$
  - $X = T_{\mathsf{GT}}$
  - $Y = T_{\mathsf{RC}}$
- Kalibrace oko-v-ruce, eye-in-hand
  - $A^i = T^i_{\mathsf{CT}}$
  - $B^i = T^i_{\mathsf{GR}}$
  - $X = T_{\mathsf{TR}}$
  - $Y = T_{\mathsf{CG}}$

Shrnutí

- Reprezentace obrázku
- Promítání do/z obrazu
- Segmentace v obrazovém prostoru
- Homografie
- Odhad polohy z obrazu
- Kalibrace kamery

Cvičení

- Tento týden žádný nový domácí úkol
- Odhad homografie na příkladu v Pythonu/OpenCV
- Kalibrace oko-ruka na příkladu v Pythonu/OpenCV