



Robotika: Základy vidění

Vladimír Petřík

vladimir.petrík@cvut.cz

6.10.2025

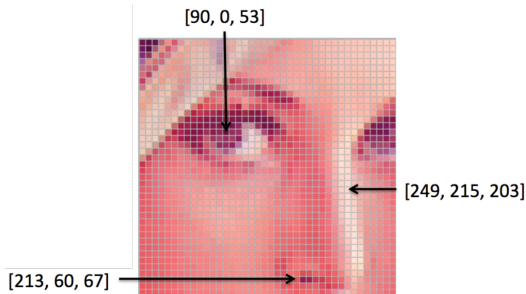
Co je to obrázek?

- ▶ Kamera připojená k počítači vytváří obrázky
- ▶ Obrázek je pole čísel¹



187	189	174	148	160	162	129	181	170	163	162	160
185	182	149	14	75	63	55	17	170	210	180	164
180	180	52	14	34	6	10	33	40	146	169	181
206	100	6	154	131	111	120	254	146	16	64	180
194	68	137	281	237	239	239	228	227	87	71	201
172	100	207	233	233	214	220	239	238	54	74	206
188	88	178	209	185	216	211	158	190	75	50	188
189	97	188	14	12	168	126	11	21	62	22	148
199	148	191	193	198	227	178	143	182	106	86	190
205	174	195	292	236	231	149	178	238	43	51	204
190	214	116	149	236	187	65	160	79	38	218	241
190	224	147	186	227	210	127	103	36	108	285	224
190	214	173	86	103	143	95	50	2	108	249	215
187	180	238	76	1	81	47	5	6	217	286	211
183	202	237	195	6	0	12	108	200	138	243	236
196	206	123	287	177	121	139	200	178	18	81	218

187	189	174	148	160	162	129	181	170	163	162	160
185	182	149	14	75	63	55	17	170	210	180	164
180	180	52	14	34	6	10	33	40	146	169	181
206	100	6	154	131	111	120	254	146	16	64	180
194	68	137	281	237	239	239	228	227	87	71	201
172	100	207	233	233	214	220	239	238	54	74	206
188	88	178	209	185	216	211	158	190	75	50	188
189	97	188	14	12	168	126	11	21	62	22	148
199	148	191	193	198	227	178	143	182	106	86	190
205	174	195	292	236	231	149	178	238	43	51	204
190	214	116	149	236	187	65	160	79	38	218	241
190	224	147	186	227	210	127	103	36	108	285	224
190	214	173	86	103	143	95	50	2	108	249	215
187	180	238	76	1	81	47	5	6	217	286	211
183	202	237	195	6	0	12	108	200	138	243	236
196	206	123	287	177	121	139	200	178	18	81	218

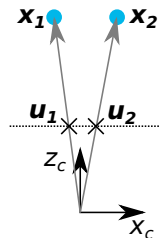
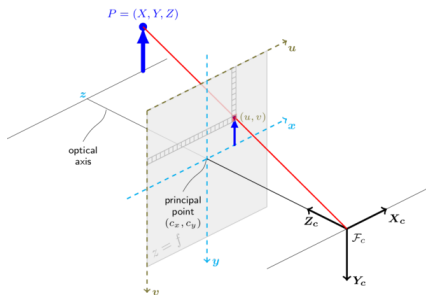


¹Obrázky jsou z: <https://ai.stanford.edu/~syueung/cvweb/tutorial1.html>



Jak vzniká obraz?

- ▶ Perspektivní kamera
 - ▶ model dírkové kamery/středové promítání²
 - ▶ promítá prostorový bod x_c do obrazového bodu $u = (u \ v)^T$ protnutím
 - ▶ obrazové roviny a
 - ▶ přímky spojující x_c se středem projekce
 - ▶ všechny body na paprsku se promítají do stejného pixelu



²docs.opencv.org

Projekce

- ▶ $\mathbf{u}_H = K \mathbf{x}_c$
 - ▶ $\mathbf{u}_H$ je pixel v homogenních souřadnicích
 - ▶ pokud $\mathbf{u}_H = (u_H \ v_H \ w_H)^\top$, pak souřadnice pixelu jsou $(u_H/w_H \ v_H/w_H)^\top$
 - ▶ alternativně můžeme reprezentovat jako: $\lambda (u, v, 1)^\top = K \mathbf{x}_c$
- ▶ 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 $\mathbf{x} = \lambda K^{-1} \mathbf{u}$
 $\mathbf{x} = \lambda K^{-1} \mathbf{u}$
 - ▶ v opačném případě lze vypočítat pouze paprsek
 - ▶ jak zjistit K z bodů?



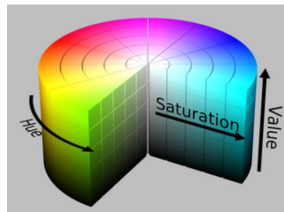
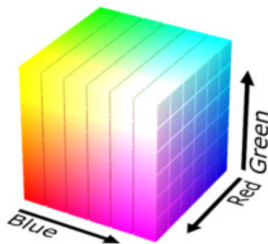
Co můžeme studovat na obrázcích?

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



Segmentační masky - prahování barev

- ▶ Prahování
 - ▶ hodnoty pixelů RGB pro souřadnice $\mathbf{u}$: $I_{\text{RGB}}(\mathbf{u})$
 - ▶ $M(\mathbf{u}) = 1$, pokud $I_{\text{RGB}}(\mathbf{u}) = (0 \ 255 \ 0)^T$?
 - ▶ $M(\mathbf{u}) = 1$, pokud $\tau_l < I_{\text{RGB}}(\mathbf{u}) < \tau_u$, pro všechny kanály
 - ▶ $M(\mathbf{u}) = 1$, pokud $\varphi_l < I_{\text{HSV}}(\mathbf{u}) < \varphi_u$, pro všechny kanály
- ▶ Následné zpracování
 - ▶ výpočet spojených komponent
 - ▶ odstranit malé nebo deformované segmenty
 - ▶ přiřadit označení na základě prahových hodnot

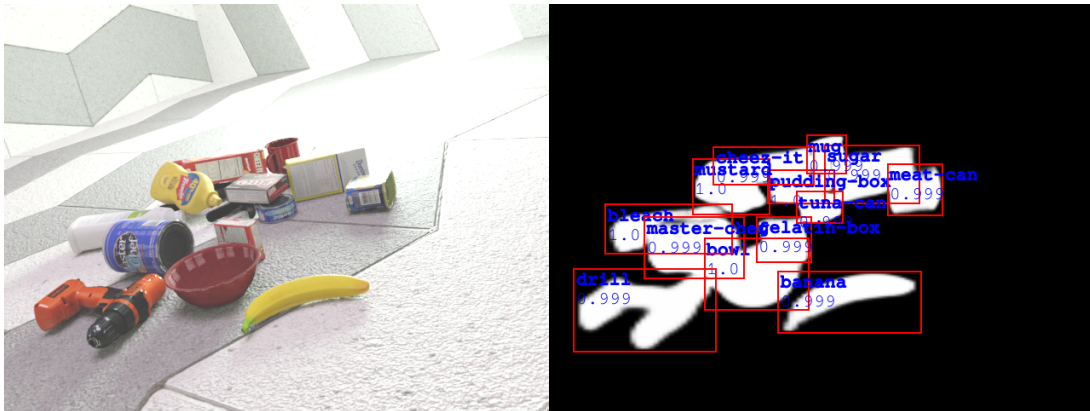


Segmentační masky pro známé 3D objekty

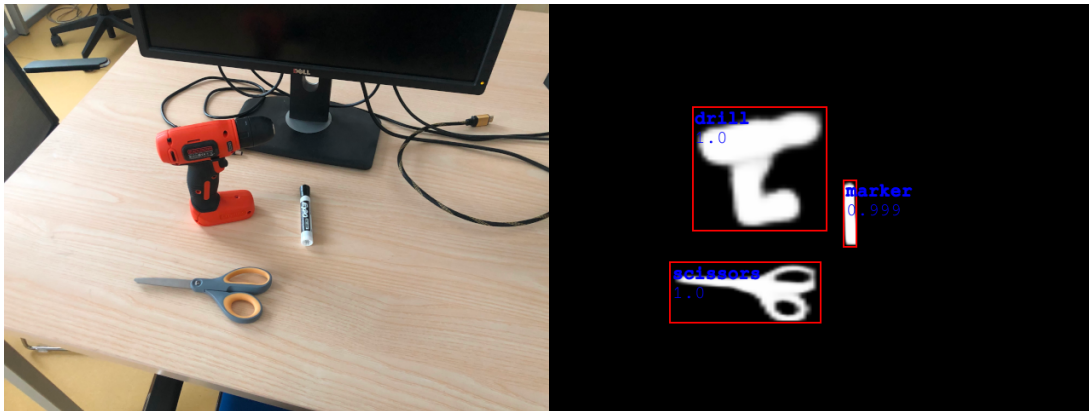
- ▶ Neuronová síť (např. Mask R-CNN)
- ▶ Trénovací vstupy:
 - ▶ datová sada obrázků, masek a štítků nebo
 - ▶ datová sada známých 3D objektů (mesh)
 - ▶ kvalita závisí na trénovacích datech (augmentation)
- ▶ Inference:
 - ▶ Vstup: obrázek
 - ▶ Výstup: segmentační maska, ohraničující rámeček, označení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



Mask R-CNN výsledky



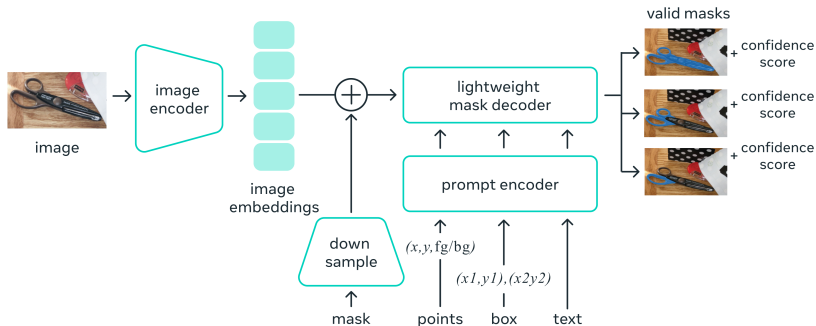
Mask R-CNN výsledky



Segmentační masky bez pretrénování

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

Universal segmentation model



SAM výsledky



SAM výsledky





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 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**

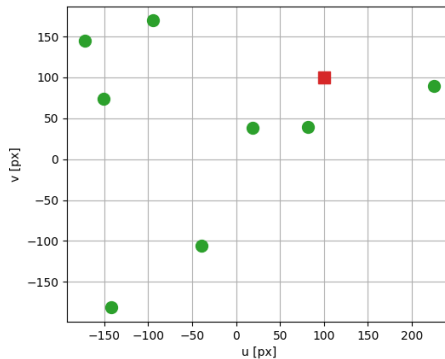
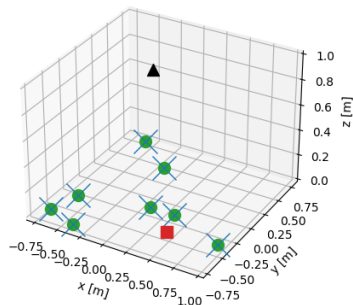


Homografie

- ▶ 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)
- ▶ $s \begin{pmatrix} x & y & 1 \end{pmatrix}^T = H \begin{pmatrix} u & v & 1 \end{pmatrix}^T$
 - ▶ 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 ?
 - ▶ $H, _ = \text{cv2.findHomography}(U, X)$
 - ▶ U, X jsou $N \times 2$ korespondenční body
 - ▶ např. ručně změříme
 - ▶ poloha středu krychle vůči rohu stolu
 - ▶ pozice středu krychle v obrázku



Příklad homografie



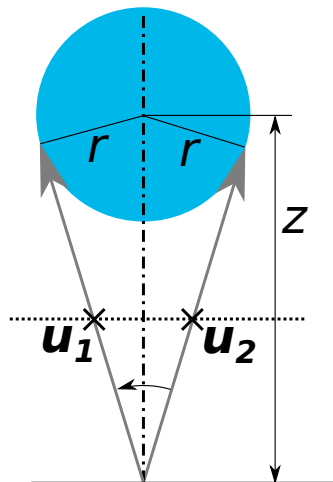
Neplanární odhad polohy/orientace

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



Použití hloubkového senzoru

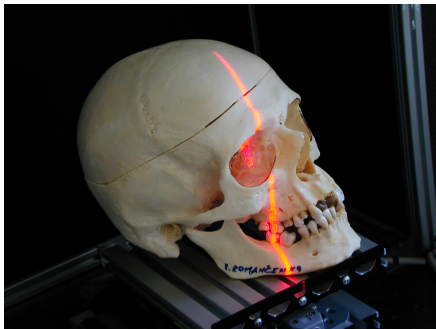
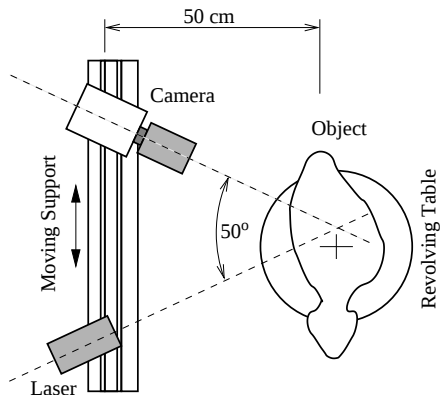
► RGBD senzory

- RGB obrázek ($H \times W \times 3$)
- Hloubková mapa ($H \times W \times 1$), vzdálenost v metrech pro každý pixel
- Strukturované mračno bodů ($H \times W \times 3$), $(x_c \ y_c \ z_c)$ pro každý pixel



Jak funguje hloubkový senzor

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



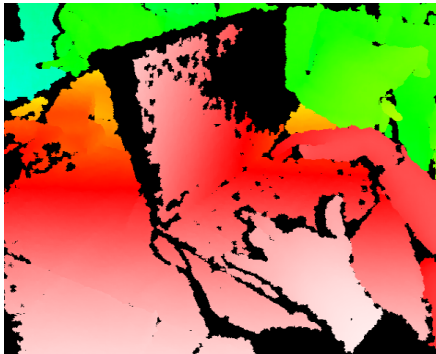
2D hloubkové senzory

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



Problémy se snímači hloubky

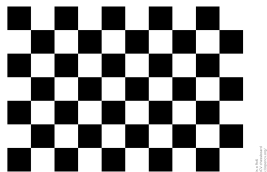
- ▶ Rekonstrukce hloubky není dokonalá (černé oblasti na snímku³)
- ▶ V jazyce python reprezentováno hodnotou NaN
- ▶ Ne každý pixel v RGB má rekonstruovanou hodnotu hloubky
- ▶ Data RGB a hloubky nejsou zarovnána (je třeba je kalibrovat)



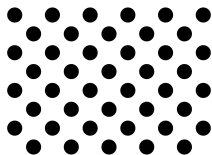
³<https://commons.wikimedia.org>, User:Kolossos

Dodatečné značky

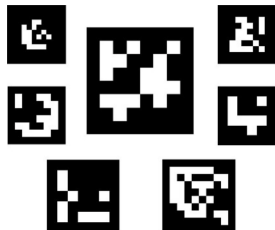
- ▶ Můžeme vypočítat polohu/orientaci obrazců⁴?
 - ▶ je třeba znát velikost a strukturu
 - ▶ subpixelová přesnost
 - ▶ musí být zcela viditelný
- ▶ Můžeme vypočítat polohu značek ArUco?
 - ▶ méně přesné než pravidelné obrazce
 - ▶ poskytuje identifikátor značky a pózu
 - ▶ musí být zcela viditelný



OpenCV

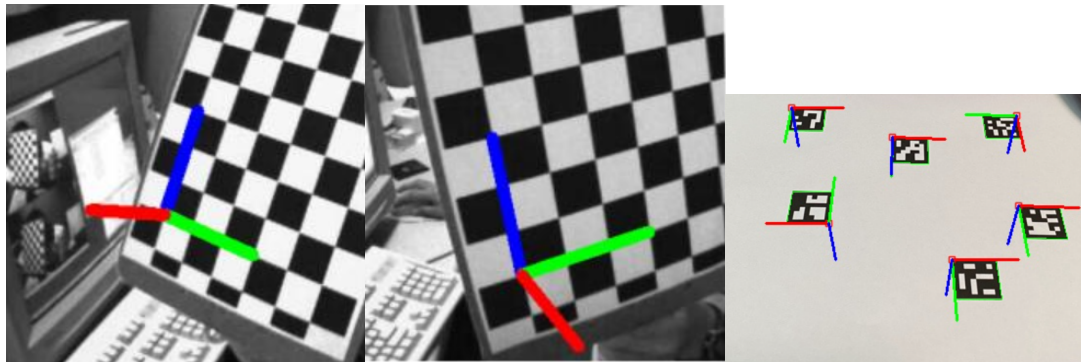


OpenCV



⁴docs.opencv.org

Příklad umístění značek

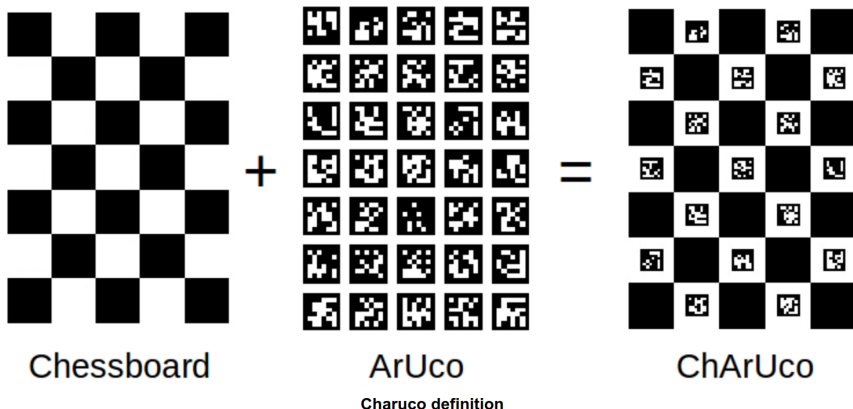


Příklad značek z reálného světa



ChArUco deska pro kalibraci

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



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
- ▶ Kde je robot?
 - ▶ homografie odhaduje polohy objektů vzhledem k s.s. rovině
 - ▶ ostatní metody odhadují polohy v s.s. kamery
 - ▶ potřebujeme odhadnout/kalibrovat T_{RC}



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
- ▶ Povinné cvičení - BOZP
 - ▶ CIIRC, B670



Co dál?

- ▶ Za týden začneme s testem
 - ▶ Otázky z prvních dvou přednášek
 - ▶ SE2 a SE3 transformace a vlastnosti
 - ▶ Přímá kinematika a DoF

