

Robotika: Kalibrace

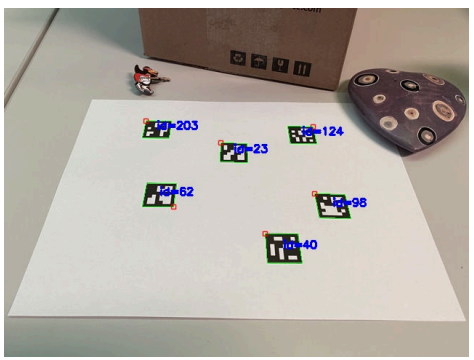
Vladimír Petrík

vladimir.petrik@cvut.cz

19. 10. 2026

Co víme

- Jak detekovat objekty v obraze
  - šachovnice
  - ArUco markery

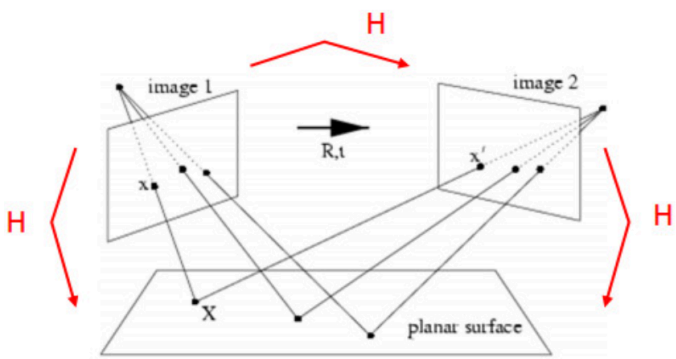




Obrázky z docs.opencv.org

Co víme

- Jak detekovat objekty v obraze
  - šachovnice
  - ArUco markery
- Jak odhadnout homografii
  - mapuje body mezi rovinami
  - odhaduje se z korespondencí
  - robustní – nevyžaduje odhad hloubky



Obrázky z docs.opencv.org

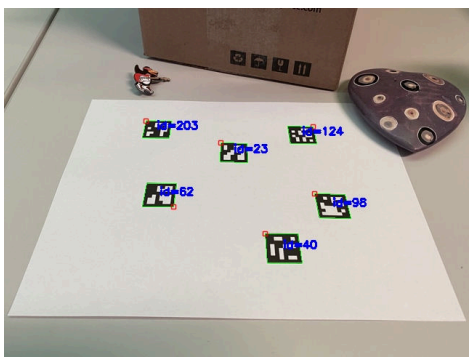
Co víme

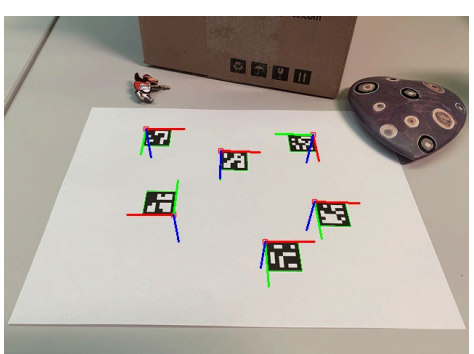
- Jak detekovat objekty v obraze
  - šachovnice
  - ArUco markery
- Jak odhadnout homografii
  - mapuje body mezi rovinami
  - odhaduje se z korespondencí
  - robustní – nevyžaduje odhad hloubky
- Kalibrace vnitřních parametrů kamery
  - použijeme obrázky šachovnice
  - odhadneme matici kamery K

Obrázky z docs.opencv.org

Co víme

- Jak detekovat objekty v obraze
  - šachovnice
  - ArUco markery
- Jak odhadnout homografii
  - mapuje body mezi rovinami
  - odhaduje se z korespondencí
  - robustní – nevyžaduje odhad hloubky
- Kalibrace vnitřních parametrů kamery
  - použijeme obrázky šachovnice
  - odhadneme matici kamery K
- Odhad polohy
  - odhaduje polohu kamery vůči objektu
  - používá algoritmus PnP
  - vyžaduje K a korespondence





Obrázky z docs.opencv.org

Kde je robot?

- Uvažujeme homografii – jak získáme 3D body v souřadnicovém systému robota?

- Omezeno na mapování roviny na rovinu

Kde je robot?

- Uvažujeme homografii – jak získáme 3D body v souřadnicovém systému robota?
  1. Ručně změříme pozice ArUco markerů vzhledem k základně
     - umístíme ArUco markery do roviny, která nás zajímá
     - může být obtížné změřit přesně

- Omezeno na mapování roviny na rovinu

Kde je robot?

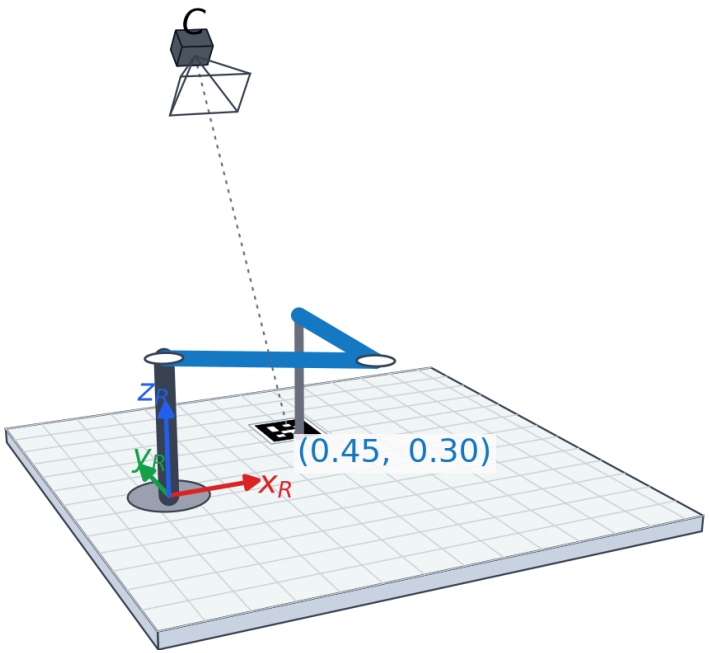
- Uvažujeme homografii – jak získáme 3D body v souřadnicovém systému robota?
  1. Ručně změříme pozice ArUco markerů vzhledem k základně
     - umístíme ArUco markery do roviny, která nás zajímá
     - může být obtížné změřit přesně
  2. Ručně umístíme robota ke kalibračnímu objektu
     - kombinujeme známé 3D body kalibračního objektu s FK

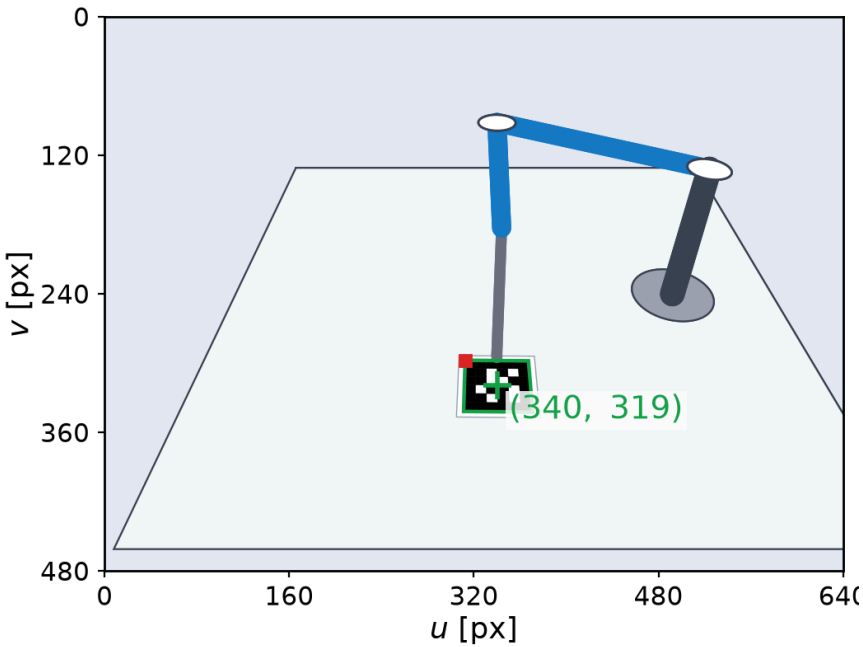
- Omezeno na mapování roviny na rovinu

Kde je robot?

- Uvažujeme homografii – jak získáme 3D body v souřadnicovém systému robota?
  1. Ručně změříme pozice ArUco markerů vzhledem k základně
     - umístíme ArUco markery do roviny, která nás zajímá
     - může být obtížné změřit přesně
  2. Ručně umístíme robota ke kalibračnímu objektu
     - kombinujeme známé 3D body kalibračního objektu s FK
  3. Použijeme ArUco marker uchopený robotem
     - použijeme FK k výpočtu pozice značky v souřadnicovém systému robota
     - musíme se pohybovat v rovině, která nás zajímá
     - nebudeme mít přesný odhad značky vzhledem k ruce
- Omezeno na mapování roviny na rovinu

Homografie z pohybu robota

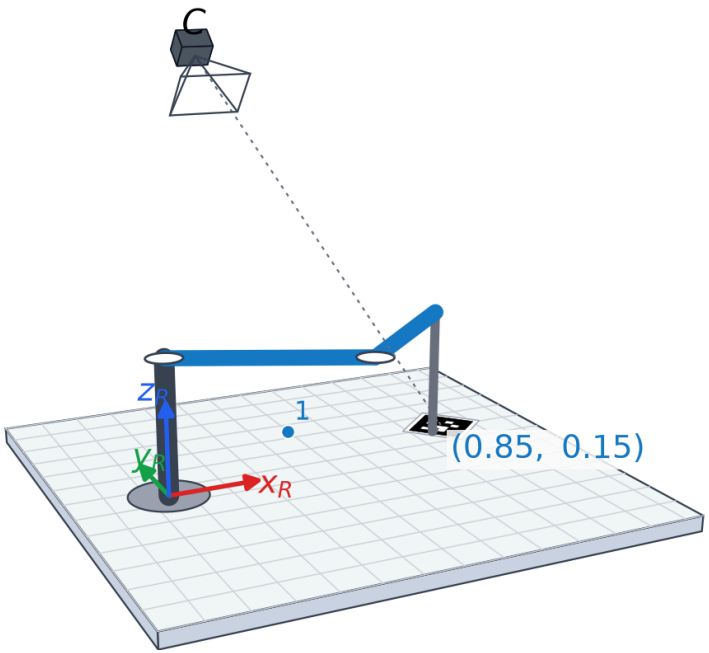


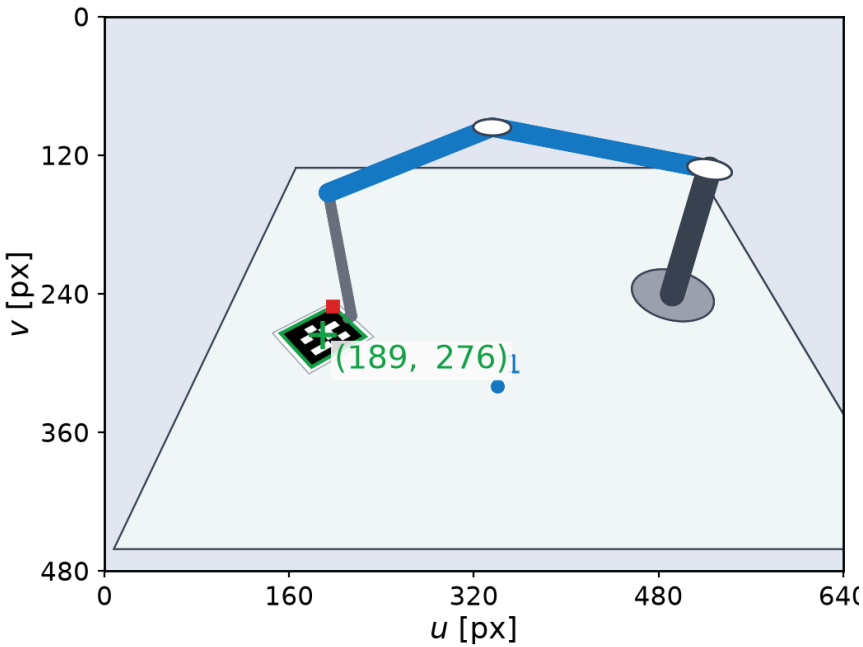


- Marker v chapadle: FK dává jeho polohu (x_R, y_R), detekce jeho pixel (u, v)

robot (x_R, y_R): $(0.45, 0.30)$
obraz (u, v): $(340, 319)$

Homografie z pohybu robota

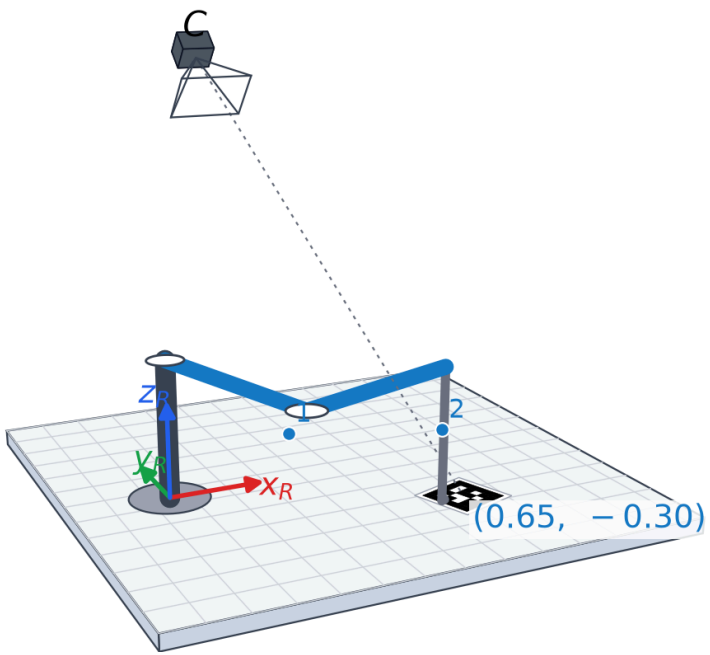


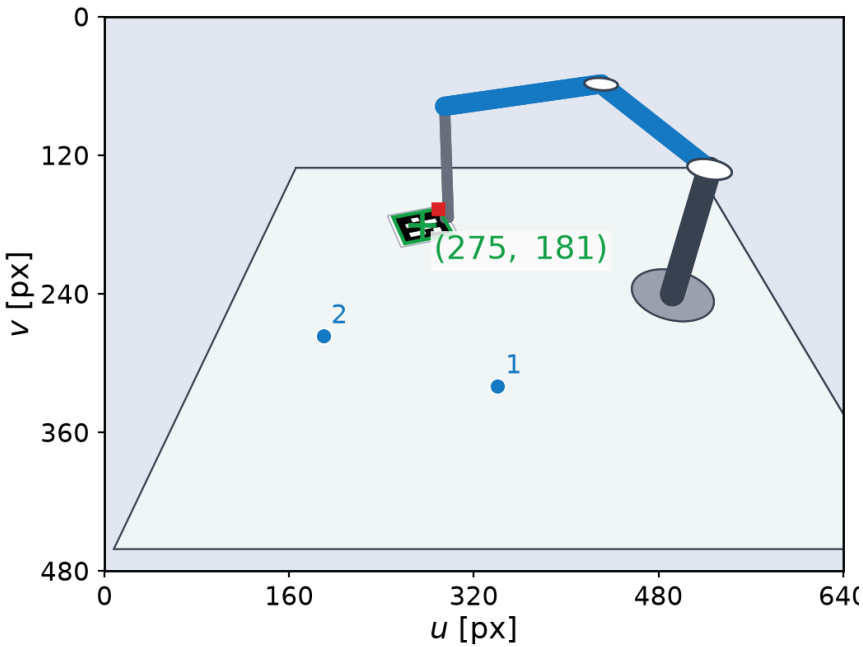


- Pohneme robotem v rovině a detekujeme znovu: jedna korespondence na polohu

robot (x_R, y_R):	$(0.45, 0.30)$	$(0.85, 0.15)$
obraz (u, v):	$(340, 319)$	$(189, 276)$

Homografie z pohybu robota

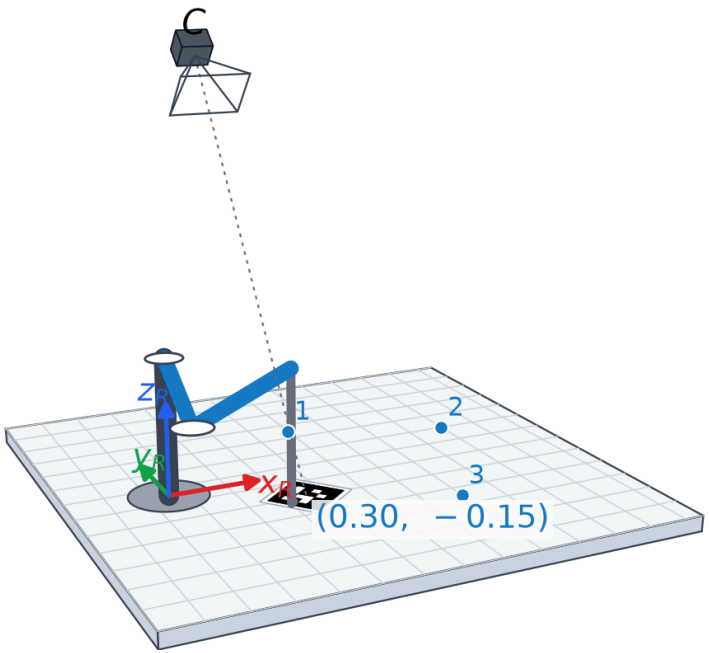


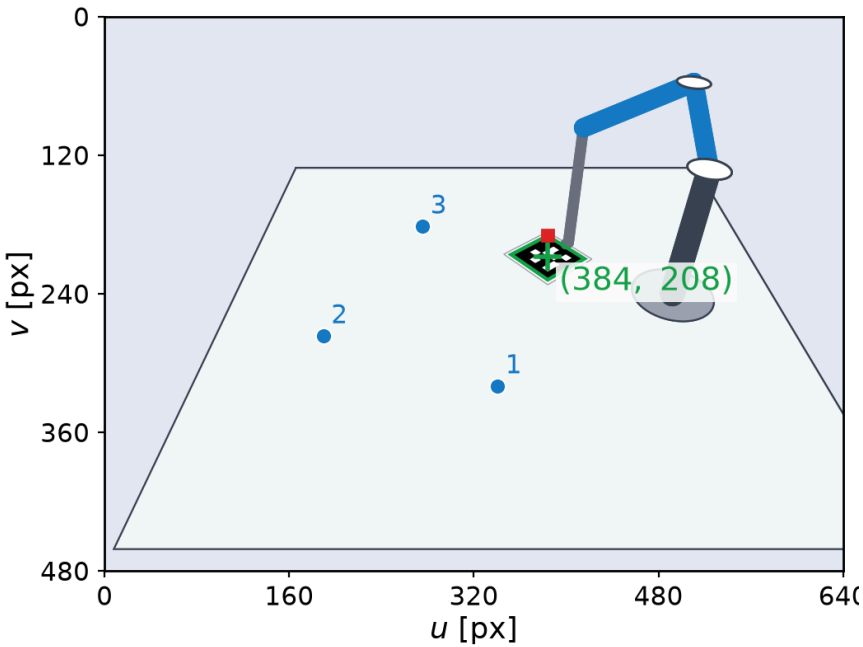


- Pohneme robotem v rovině a detekujeme znovu: jedna korespondence na polohu

robot (x_R, y_R):	$(0.45, 0.30)$	$(0.85, 0.15)$	$(0.65, -0.30)$
obraz (u, v):	$(340, 319)$	$(189, 276)$	$(275, 181)$

Homografie z pohybu robota

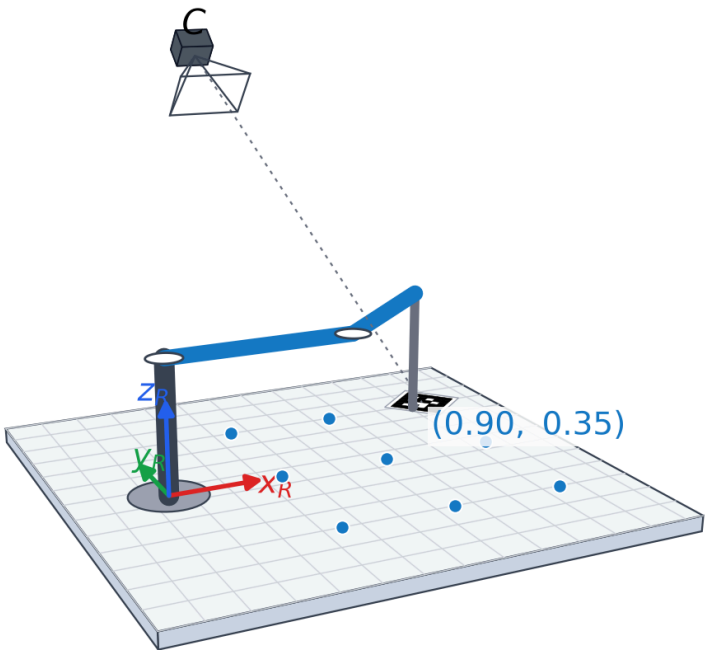


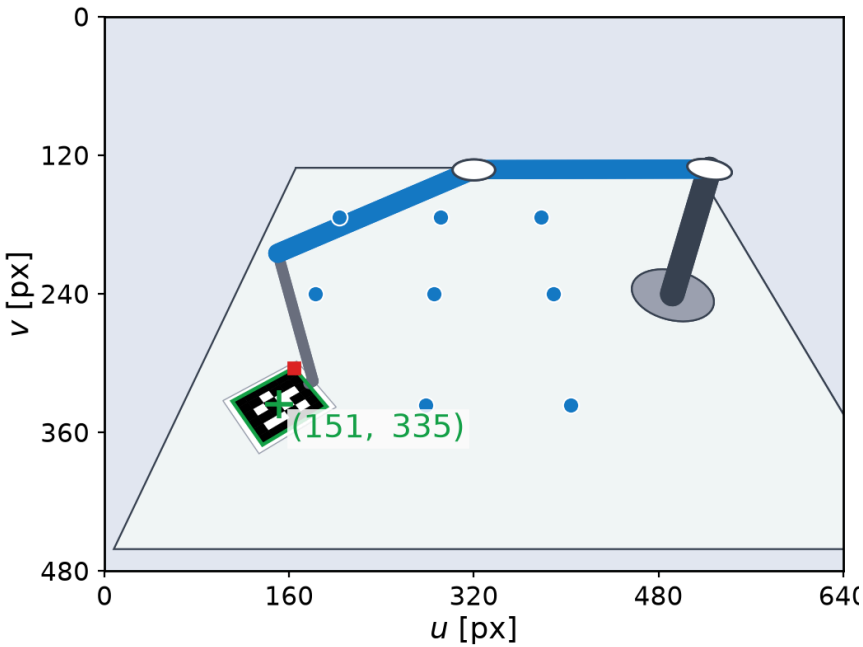


- Čtyři korespondence určují H (8 DoF)

robot (x_R, y_R):	$(0.45, 0.30)$	$(0.85, 0.15)$	$(0.65, -0.30)$	$(0.30, -0.15)$
obraz (u, v):	$(340, 319)$	$(189, 276)$	$(275, 181)$	$(384, 208)$

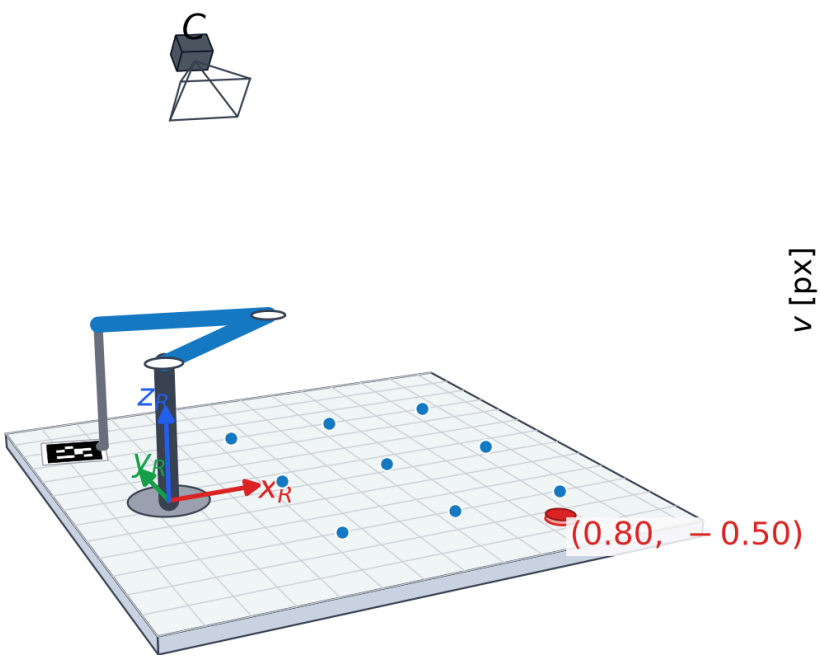
Homografie z pohybu robota

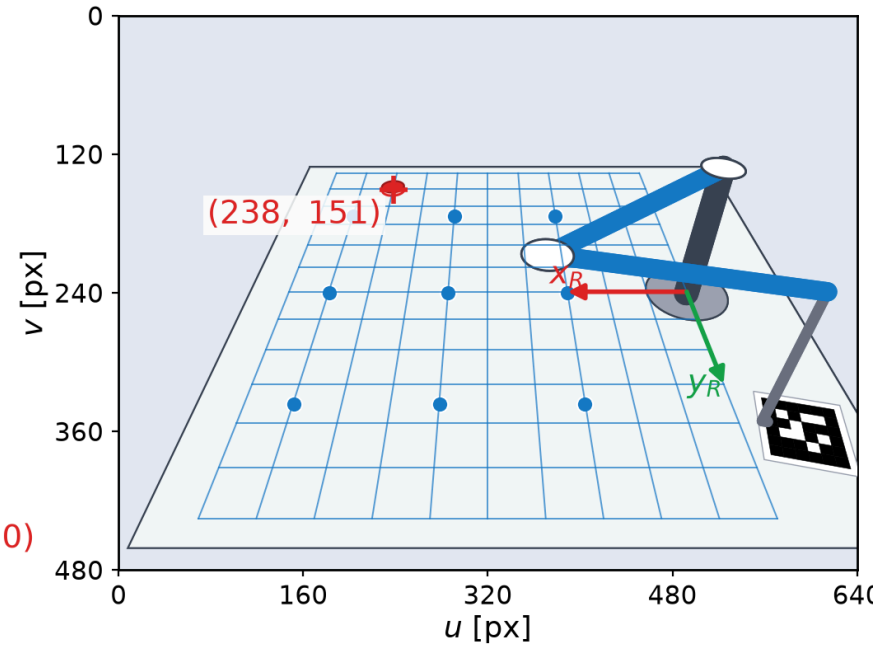




- Více poloh rozmístěných po pracovním prostoru: odhad nejmenšími čtverci, `cv2.findHomography`
- Marker musí zůstat v rovině, která nás zajímá, zde stůl

Homografie z pohybu robota





- H^{-1} mapuje libovolný pixel roviny stolu do s.s. robota
- Objekt detekovaný v $(238, 151) \rightarrow (0.80, -0.50)$: cíl pro robota

Kalibrace polohy/orientace kamery

- Kalibrace oko-ruka
- Řešíme $A^i X = Y B^i$
  - měření: $A^i, B^i \in SE(3)$
  - odhadnuté parametry: $X, Y \in SE(3)$

Kalibrace polohy/orientace kamery

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

Kalibrace polohy/orientace kamery

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

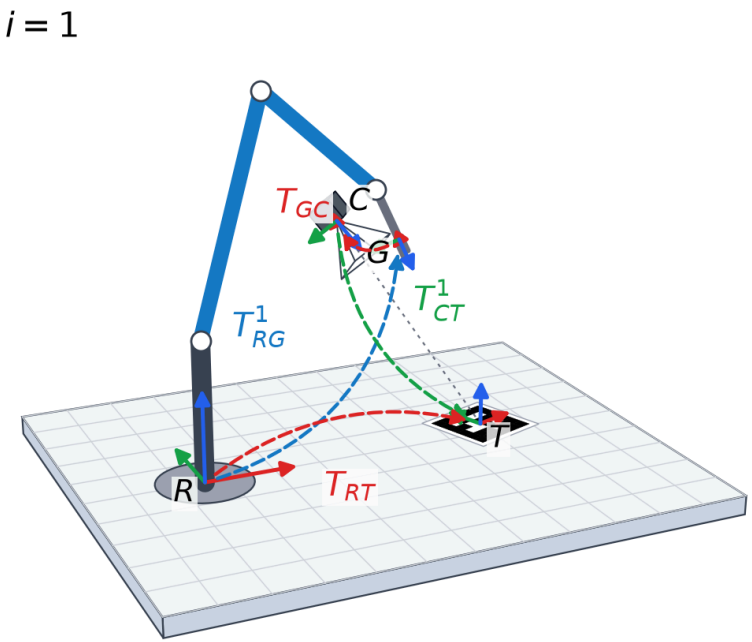
- Kalibrace oko-ruka, eye-to-hand
  - $A^i = T^i_{RG}$
  - $B^i = T^i_{CT}$
  - $X = T_{GT}$
  - $Y = T_{RC}$

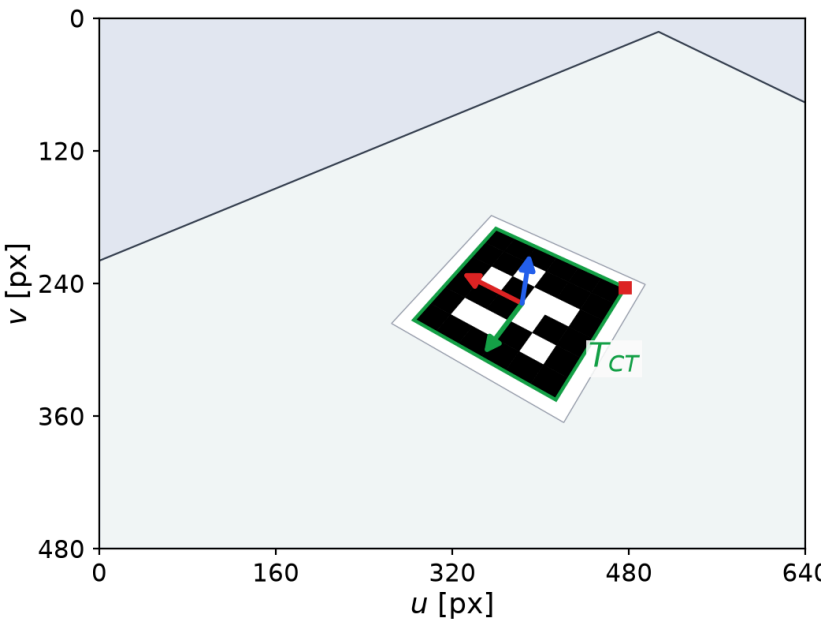
Kalibrace polohy/orientace kamery

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

- Kalibrace oko-ruka, eye-to-hand
  - $A^i = T^i_{RG}$
  - $B^i = T^i_{CT}$
  - $X = T_{GT}$
  - $Y = T_{RC}$
- Kalibrace oko-v-ruce, eye-in-hand
  - $A^i = T^i_{CT}$
  - $B^i = T^i_{GR}$
  - $X = T_{TR}$
  - $Y = T_{CG}$

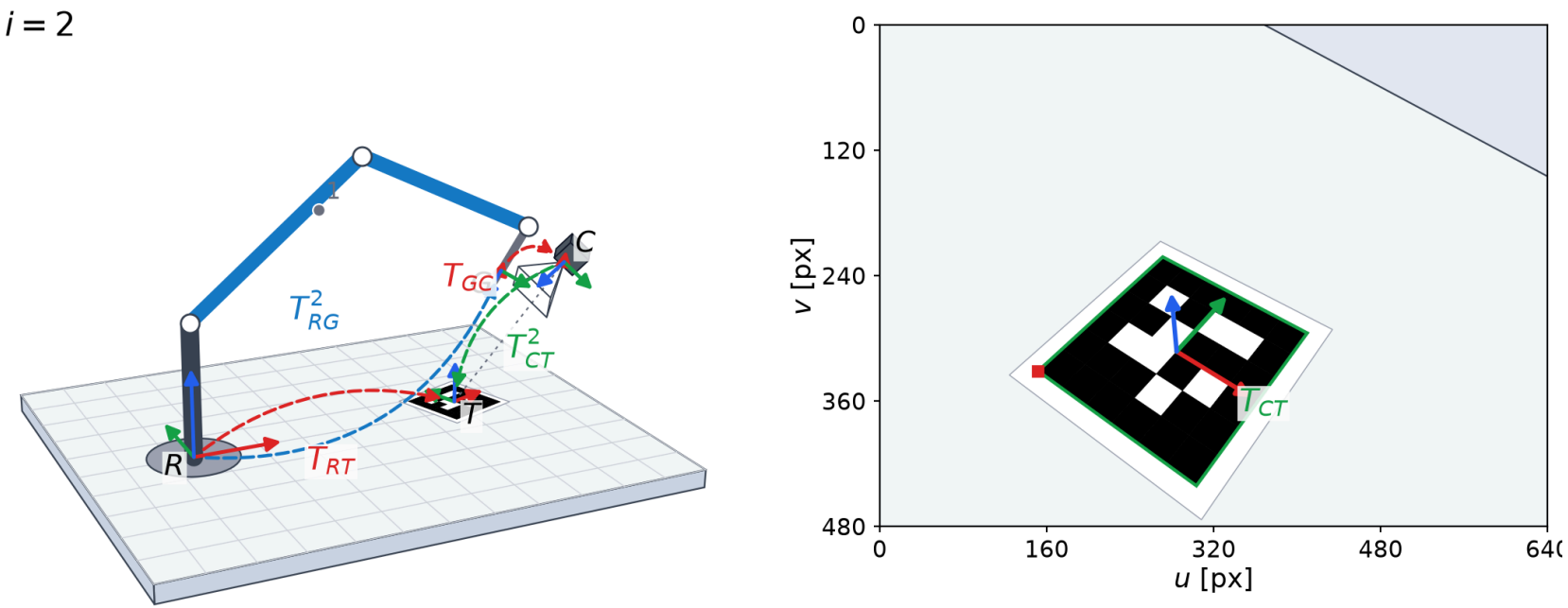
Kalibrace oko-v-ruce





- Kamera na chapadle, ArUco terč pevně na stole: T^1_{RG} z FK, T^1_{CT} z odhadu polohy
- T_{GC} a T_{RT} neznáme, ale jsou konstantní

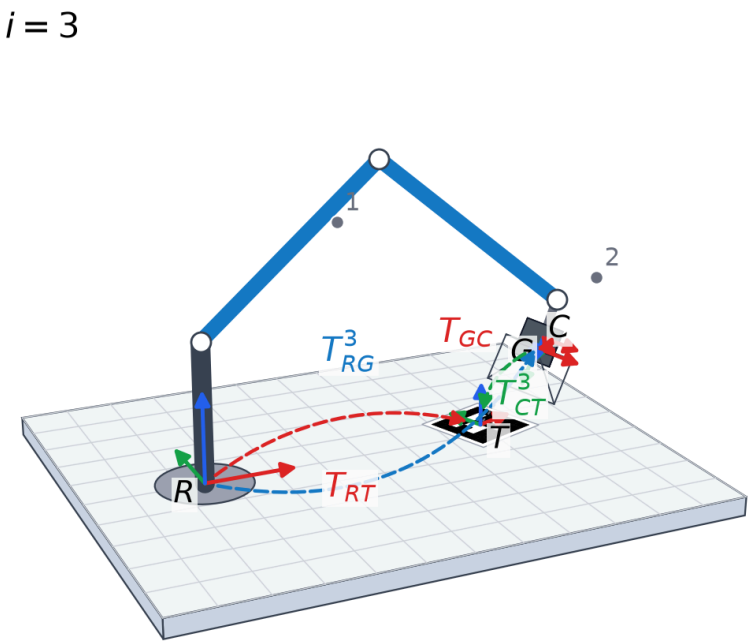
Kalibrace oko-v-ruce

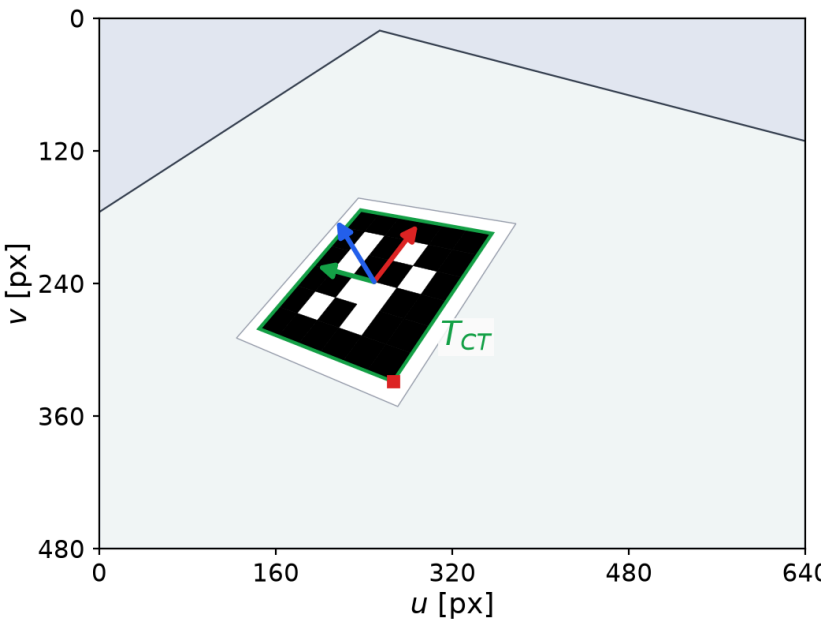


- Pohneme robotem: nové T^2_{RG} a T^2_{CT}, smyčka se uzavírá pro každou polohu i

$$T^i_{RG}\, T_{GC}\, T^i_{CT} = T_{RT}$$

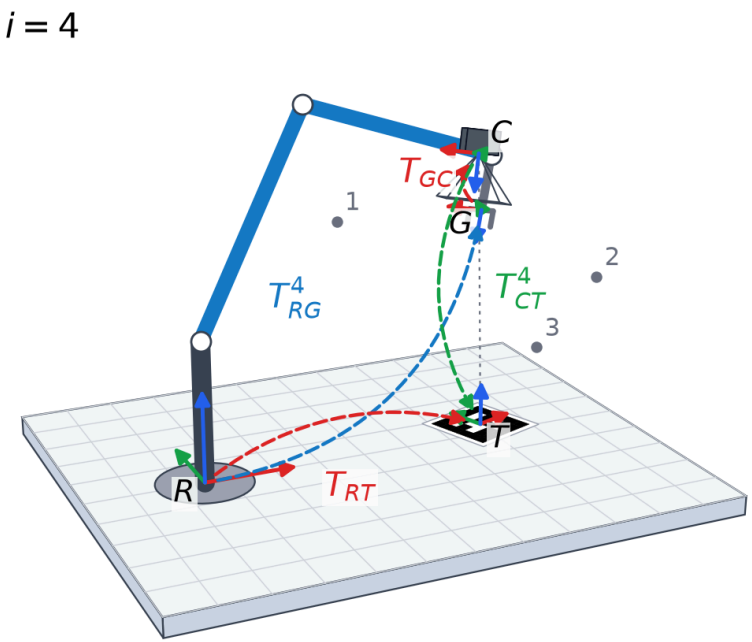
Kalibrace oko-v-ruce

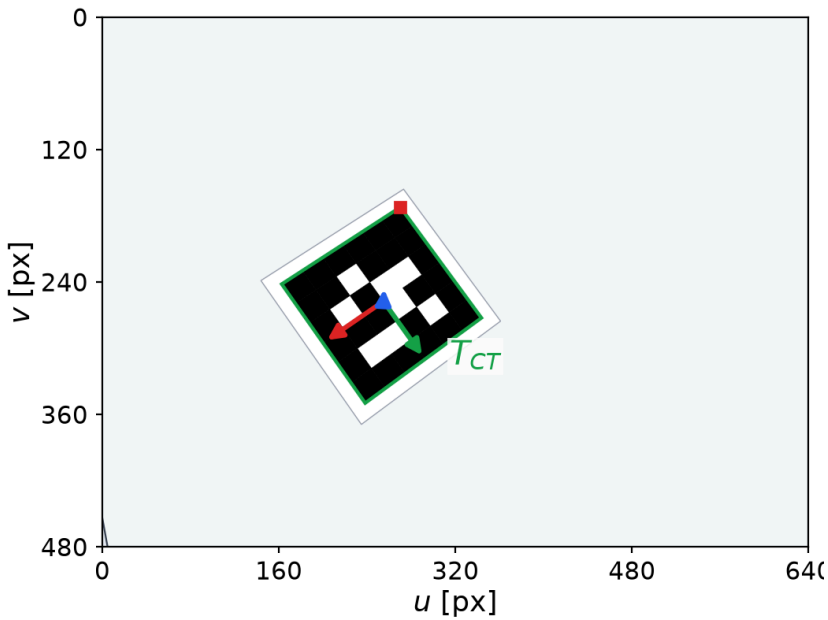




- Otáčíme kamerou kolem různých os: rotace T^i_{RG} určují rotaci T_{GC}
- Čisté translace nebo rotace kolem jediné osy jsou degenerované

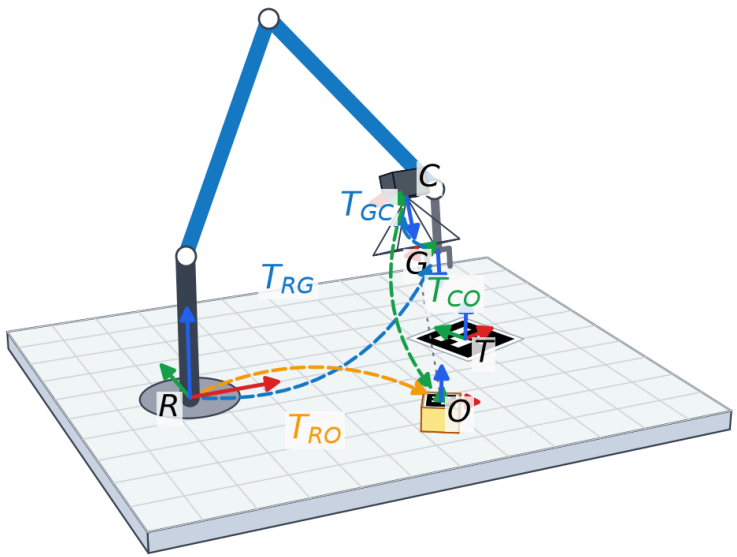
Kalibrace oko-v-ruce

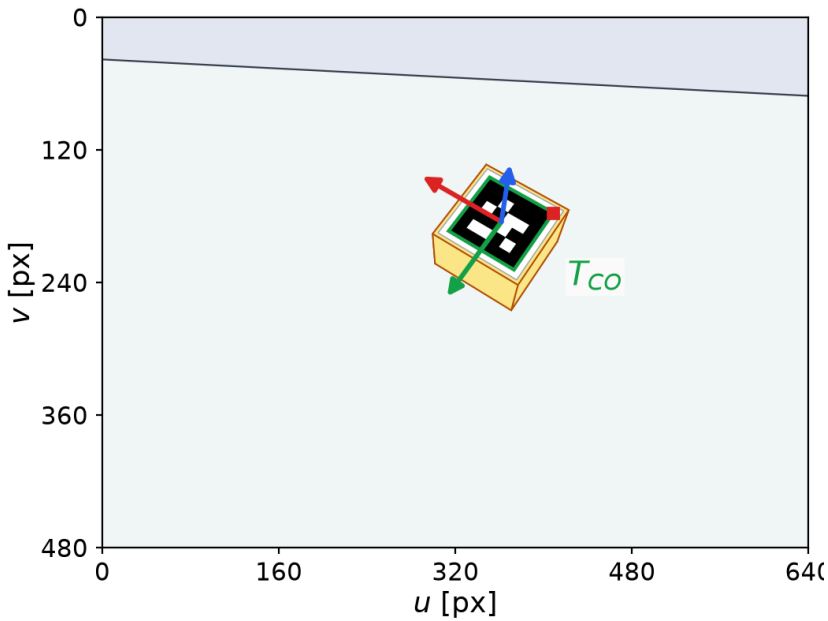




- Aspoň 3 polohy s různými rotacemi, více poloh pro odhad nejmenšími čtverci
- $A^iX = YB^i$ s $A^i = T^i_{CT}$, $B^i = T^i_{GR}$, $X = T_{TR}$, $Y = T_{CG}$: `cv2.calibrateRobotWorldHandEye`

Kalibrace oko-v-ruce





- Kalibrované T_{GC}: každý marker viděný kamerou dostane polohu v s.s. robota

$$T_{RO} = T_{RG}\, T_{GC}\, T_{CO}$$

Kód pro kalibraci $AX = YB$

```python
def solve_AX_YB(a: list[SE3], b: list[SE3]) -> tuple[SE3, SE3]:
    """Solve A^i X = Y B^i, return X, Y"""

    rvec_a = [T.rotation.log() for T in a]
    tvec_a = [T.translation for T in a]
    rvec_b = [T.rotation.log() for T in b]
    tvec_b = [T.translation for T in b]

    Rx, tx, Ry, ty = cv2.calibrateRobotWorldHandEye(rvec_a, tvec_a, rvec_b, tvec_b)
    return SE3(tx[:, 0], SO3(Rx)), SE3(ty[:, 0], SO3(Ry))
```

Vliv velikosti ArUco markerů

- Odhad hloubky malých ArUco markerů je obtížný
  - velké chyby vedou k nepřesné kalibraci
  - velké markery omezují pohyb

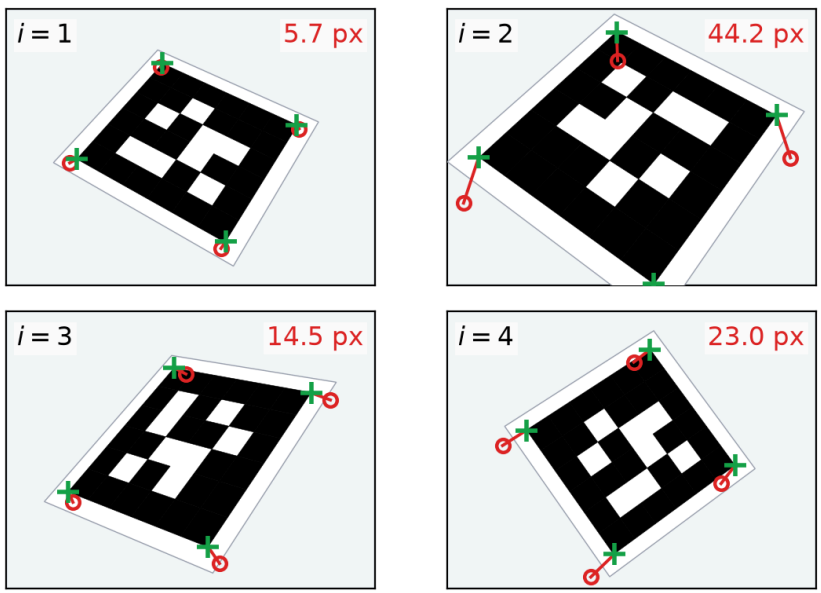
Vliv velikosti ArUco markerů

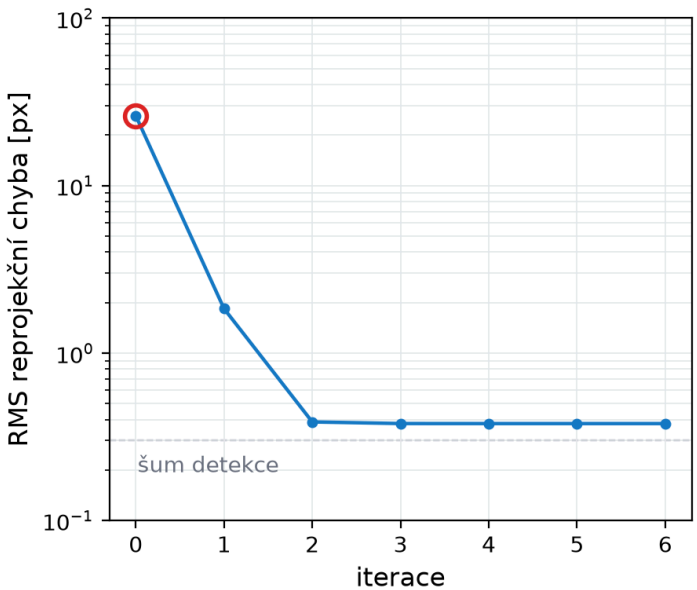
- Odhad hloubky malých ArUco markerů je obtížný
  - velké chyby vedou k nepřesné kalibraci
  - velké markery omezují pohyb
- Řešení: minimalizace reprojekční chyby
  - původní měření jsou v pixelech
  - můžeme provést jemnou optimalizaci v prostoru pixelů
  - reprojekční chyba:

$$X^*, Y^* = \arg\min_{X,Y} \sum_i \|\boldsymbol{u}_i - \pi(\boldsymbol{x}_i(X, Y))\|^2$$

    - $\boldsymbol{u}_i$ – detekovaný 2D bod, $\boldsymbol{x}_i$ – 3D bod v souřadnicovém systému kamery, π – projekce
    - $\boldsymbol{x}_i$ je spočteno pomocí FK a poloh X a Y
    - počáteční odhad je z kalibrace oko-ruka

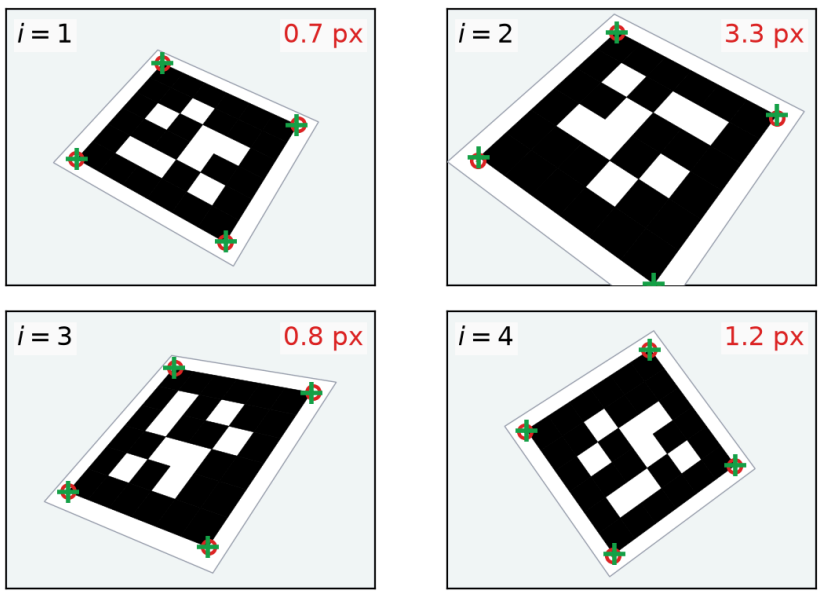
Minimalizace reprojekční chyby

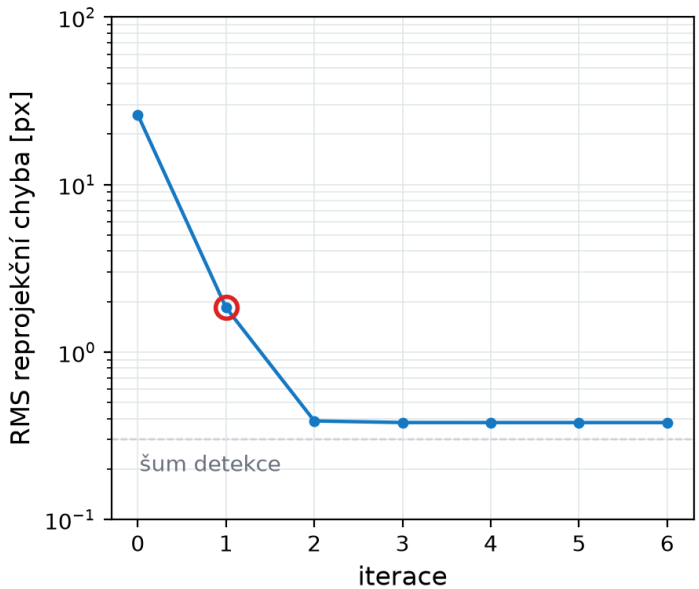




- Počáteční X, Y z kalibrace oko-ruka: detekované rohy $\boldsymbol{u}_i$ vs. reprojektované rohy $\pi(\boldsymbol{x}_i(X, Y))$
- Chyby desítek pixelů, v každém obrázku jiné

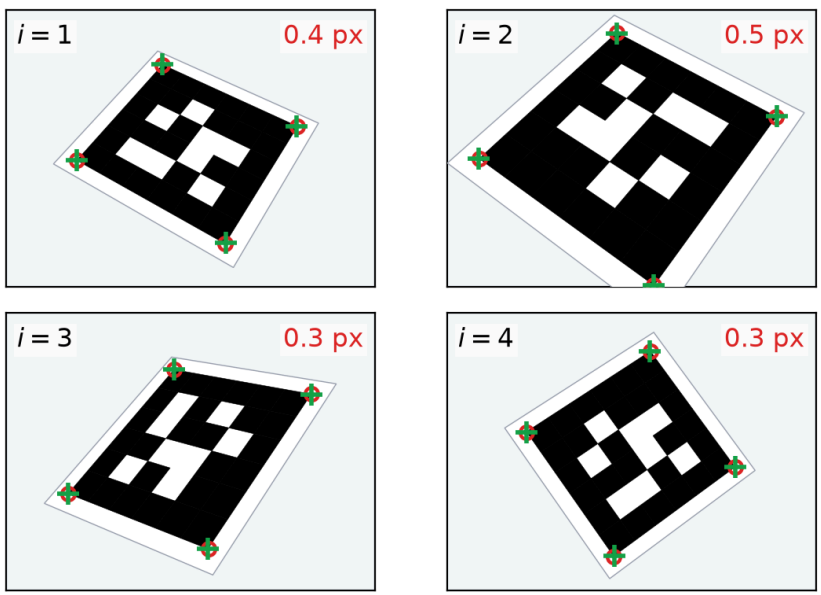
Minimalizace reprojekční chyby

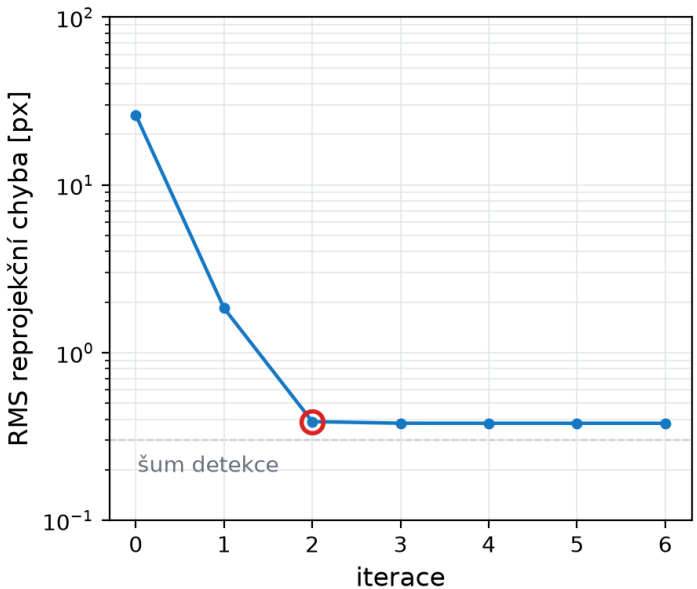




- Krok Gauss–Newton: linearizujeme rezidua ve 12 parametrech X, Y, vyřešíme lineární nejmenší čtverce, aktualizujeme

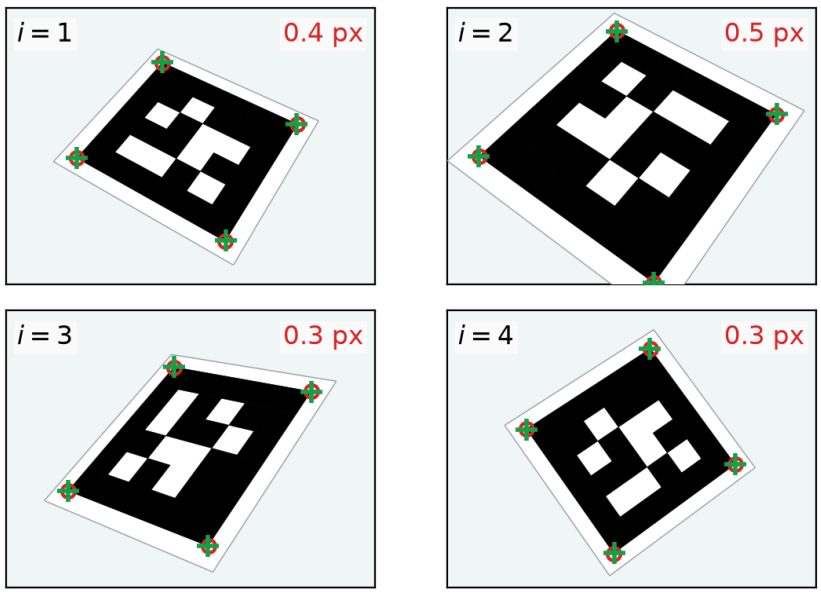
Minimalizace reprojekční chyby

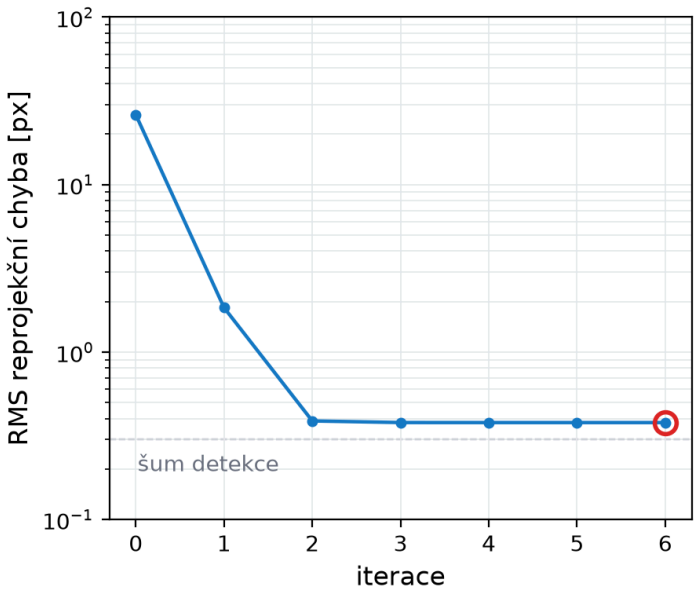




- Druhá iterace: chyba klesne pod jeden pixel

Minimalizace reprojekční chyby





- Konvergence po třech iteracích: reziduum je na úrovni šumu detekce
- V praxi: `scipy.optimize.least_squares` s funkcí reziduí

Vliv kinematiky robota

- Kinematika robota také není perfektní
  1. různé konfigurace robota mají různé chyby
  2. často chyby v offsetech kloubů

Vliv kinematiky robota

- Kinematika robota také není perfektní
  1. různé konfigurace robota mají různé chyby
  2. často chyby v offsetech kloubů
- Řešení #1: omezit se na podmnožinu konfigurací
  - při řešení IK používat pouze podobné konfigurace
  - pro kalibraci, ale také pro provoz robota!

Vliv kinematiky robota

- Kinematika robota také není perfektní
  1. různé konfigurace robota mají různé chyby
  2. často chyby v offsetech kloubů
- Řešení #1: omezit se na podmnožinu konfigurací
  - při řešení IK používat pouze podobné konfigurace
  - pro kalibraci, ale také pro provoz robota!
- Řešení #2: kalibrovat offsety kloubů
  - stejně jako při minimalizaci reprojekční chyby

$$X^*, Y^*, \boldsymbol{\theta}^*_{\text{offset}} = \underset{X,Y,\boldsymbol{\theta}_{\text{offset}}}{\arg\min} \sum_i \left\| \boldsymbol{u}_i - \pi(\boldsymbol{x}_i(X, Y, \boldsymbol{\theta}_{\text{offset}})) \right\|^2$$

  - FK je upraveno tak, aby zahrnovalo offsety kloubů

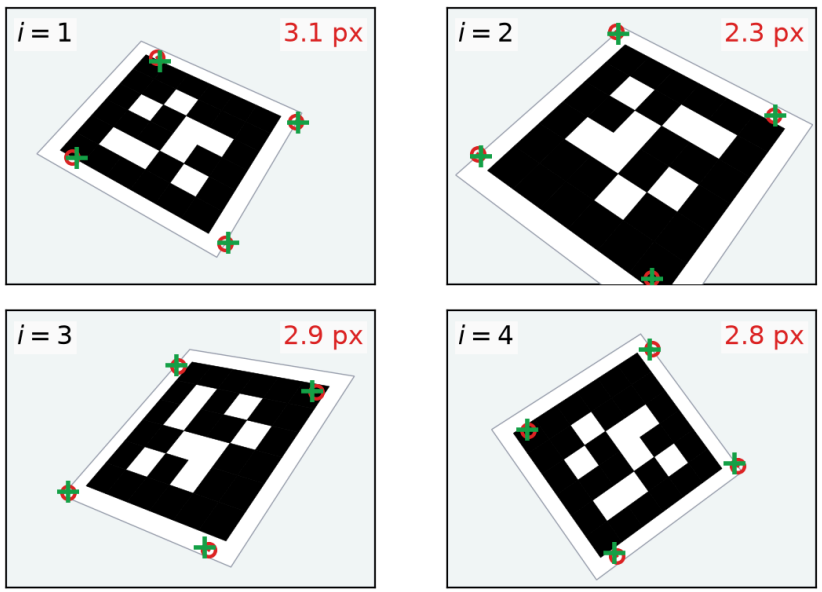
Vliv kinematiky robota

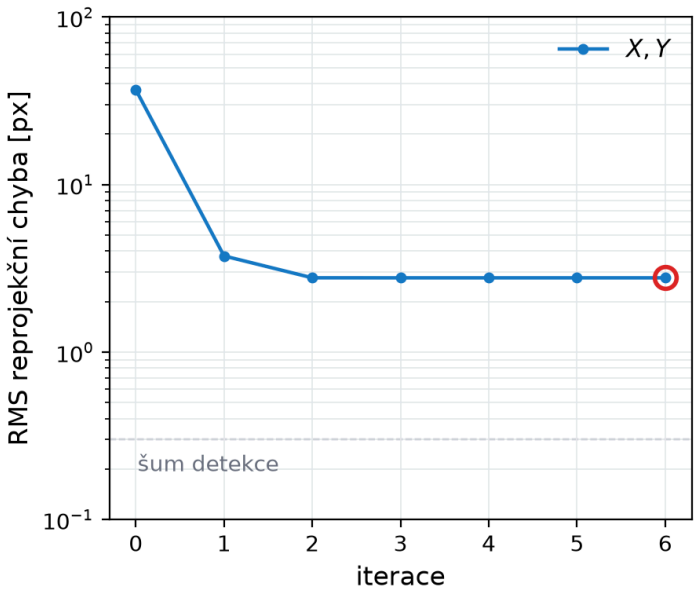
- Kinematika robota také není perfektní
  1. různé konfigurace robota mají různé chyby
  2. často chyby v offsetech kloubů
- Řešení #1: omezit se na podmnožinu konfigurací
  - při řešení IK používat pouze podobné konfigurace
  - pro kalibraci, ale také pro provoz robota!
- Řešení #2: kalibrovat offsety kloubů
  - stejně jako při minimalizaci reprojekční chyby

$$X^*, Y^*, \boldsymbol{\theta}^*_{\text{offset}} = \arg\min_{X,Y,\boldsymbol{\theta}_{\text{offset}}} \sum_i \left\| \boldsymbol{u}_i - \pi(\boldsymbol{x}_i(X, Y, \boldsymbol{\theta}_{\text{offset}})) \right\|^2$$

  - FK je upraveno tak, aby zahrnovalo offsety kloubů
- Kalibrace není jednorázový úkol
  - kalibrovat robota před každým důležitým úkolem
  - kalibrovat robota po každé významné změně

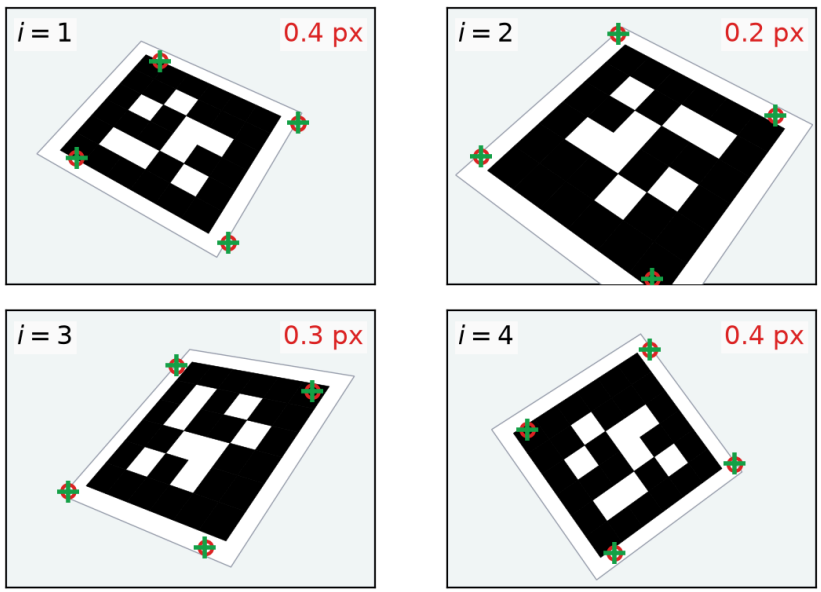
Kalibrace offsetů kloubů

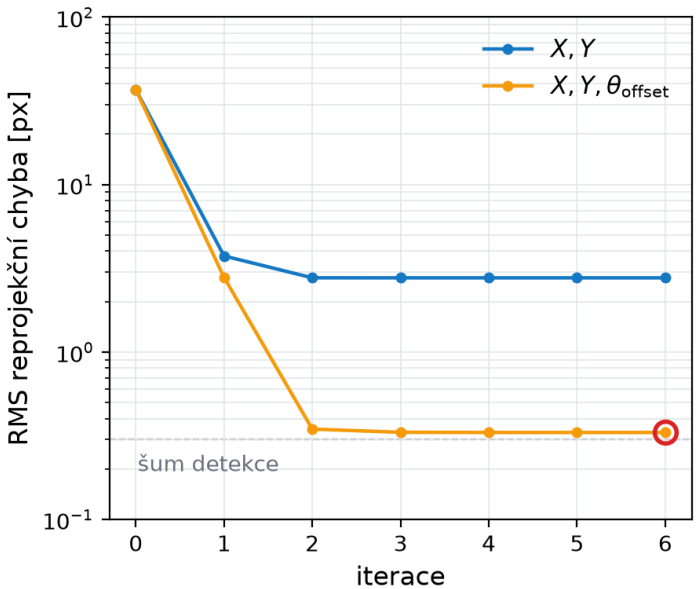




- Robot s offsety kloubů (zde $-1.5°$ a $2.5°$ na dvou kloubech): samotné X, Y měření nevysvětlí
- Reziduum zůstává nad šumem detekce, v každé konfiguraci jinak

Kalibrace offsetů kloubů





- Přidáme θ_{offset} mezi parametry: reziduum klesne na úroveň šumu a offsety se odhadnou
- Více poloh a konfigurací zlepšuje pozorovatelnost offsetů

Závěr

- Kalibrace závisí na úkolu
  - homografie
  - vnitřní parametry kamery
  - poloha kamery
  - kinematika robota
- Kalibrace není jednorázový úkol
  - dobré ji automatizovat

Cvičení

- 5. a 6. týden: individuální práce na semestrálním projektu, konzultace s cvičícími
- hw03 (transformace kamera–robot, homografie) odevzdat do 25. 10.
- Roboty rezervujte v BRUTE