



FACULTY OF MATHEMATICS,
PHYSICS AND INFORMATICS
Comenius University
Bratislava

3D Lokalizácia

Praktikum zo strojového učenia a umelej inteligencie na vizuálnych dátach

Ing. Viktor Kocur, PhD.

24.11.2022

3D Lokalizácia Snímok

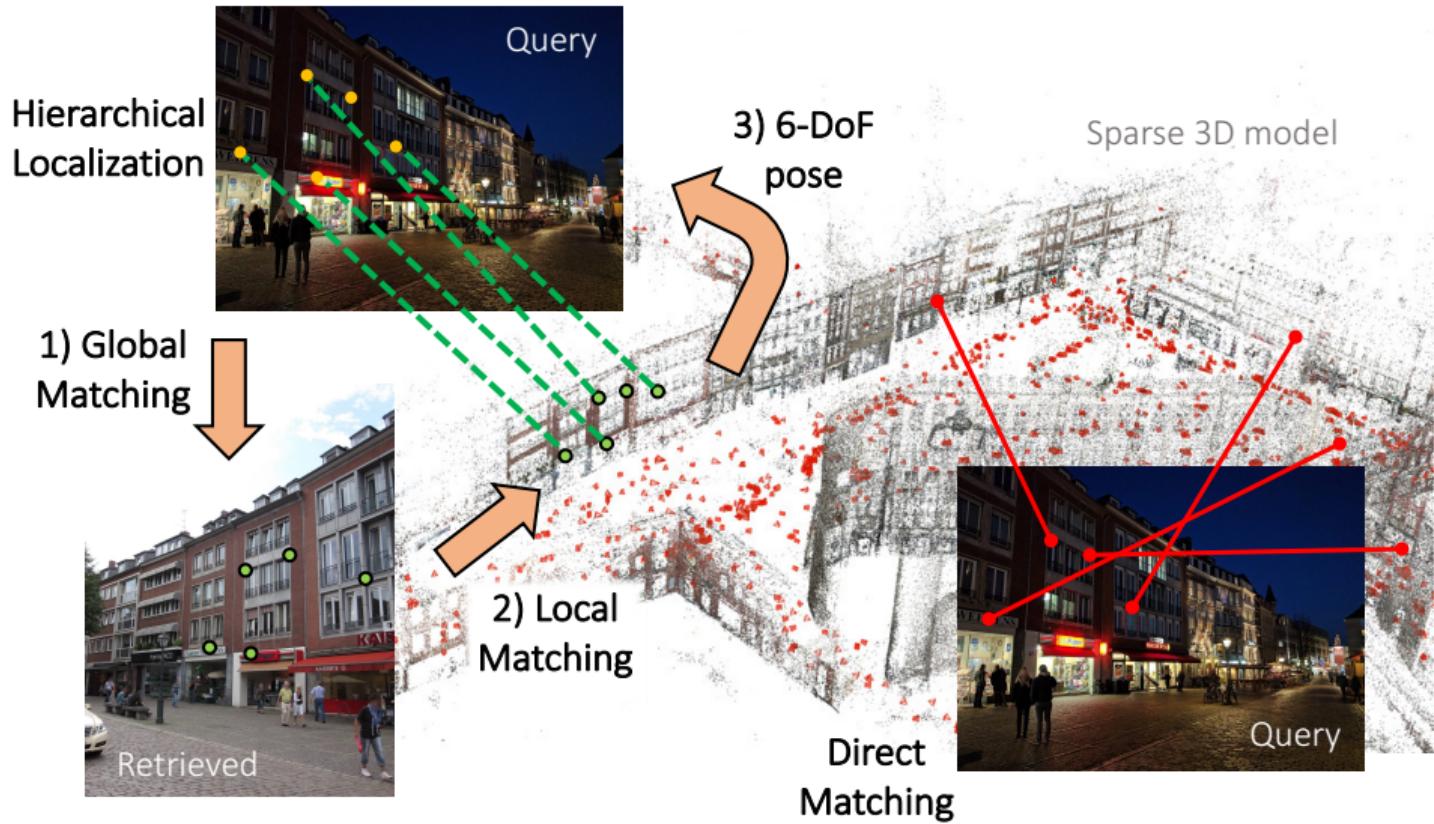


Formálny popis

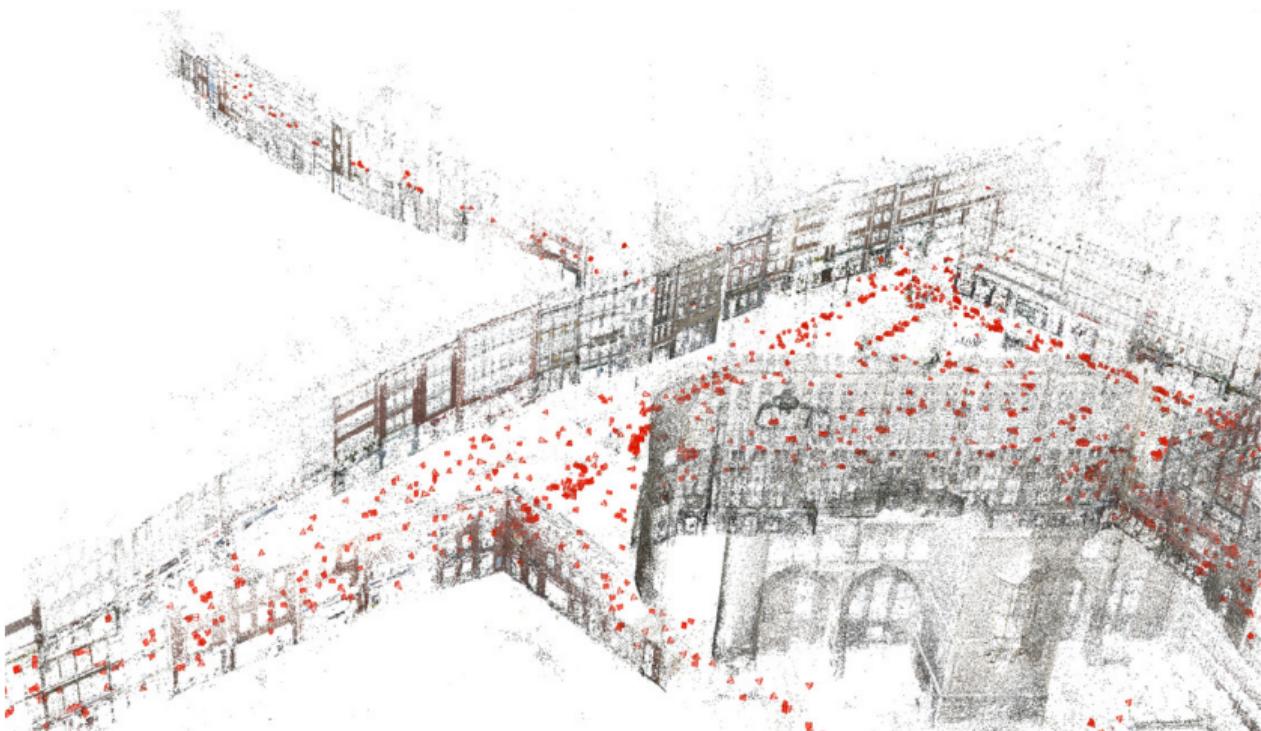


Majme definovanú 3D súradnu sústavu v reálnom svete a fotografiu. Potom sa snažíme pre danú fotografiu nájsť rotačnú maticu $R \in SO(3)$ a translačný vektor $\mathbf{t} \in \mathbb{R}^3$, ktoré popisujú pozíciu optického stredu kamery v čase záberyenia fotografie.

Ako na to?



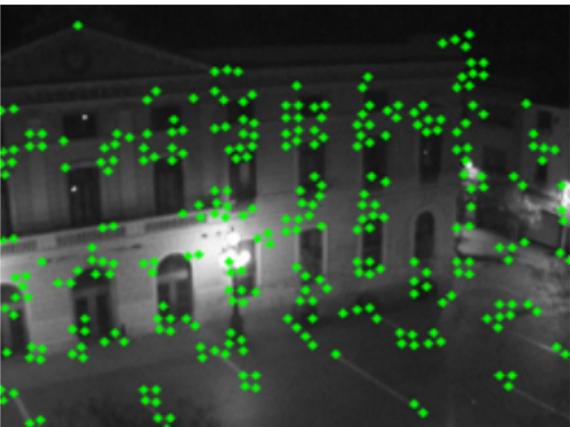
Ako na to? - 3D model



Structure from Motion



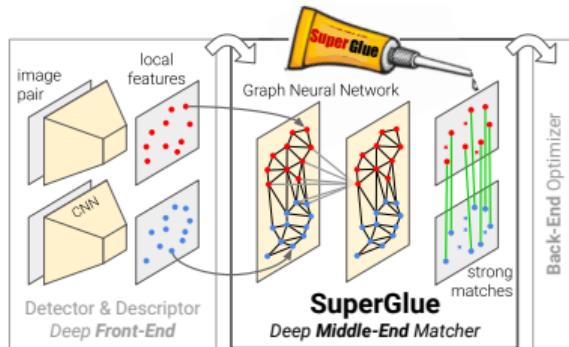
1. Detekcia klúčových bodov v obrázkoch z datasetu



Structure from Motion



1. Detekcia klúčových bodov v obrázkoch z datasetu
2. Párovanie bodov medzi obrázkami





1. Detekcia klúčových bodov v obrázkoch z datasetu

$$\mathbf{x}'^T F \mathbf{x} = 0$$

2. Párovanie bodov medzi obrázkami

$$\det(F) = 0$$

3. Geometrická verifikácia korešpondencií - RANSAC

Structure from Motion



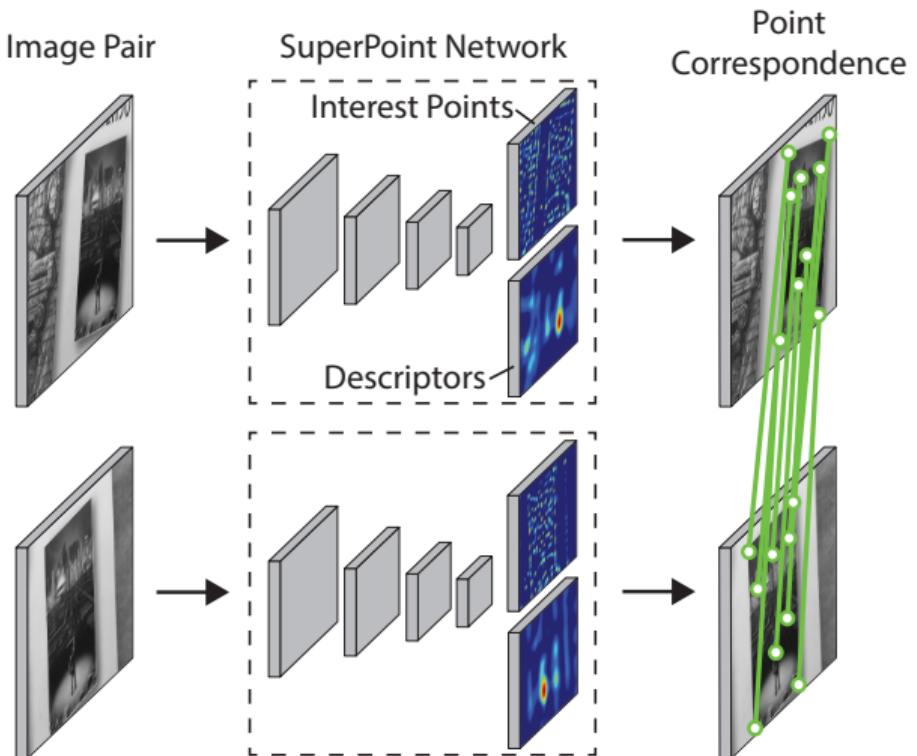
1. Detekcia klúčových bodov v obrázkoch z datasetu
2. Párovanie bodov medzi obrázkami
3. Geometrická verifikácia korešpondencií - RANSAC
4. Optimalizácia modelu - Bundle Adjustment

$$\arg \min_{\mathbf{X}_1, \mathbf{X}_2, \dots} \sum_{i=1}^m \sum_{j=1}^n \|x_{ij} - P_i \mathbf{X}_j\|$$

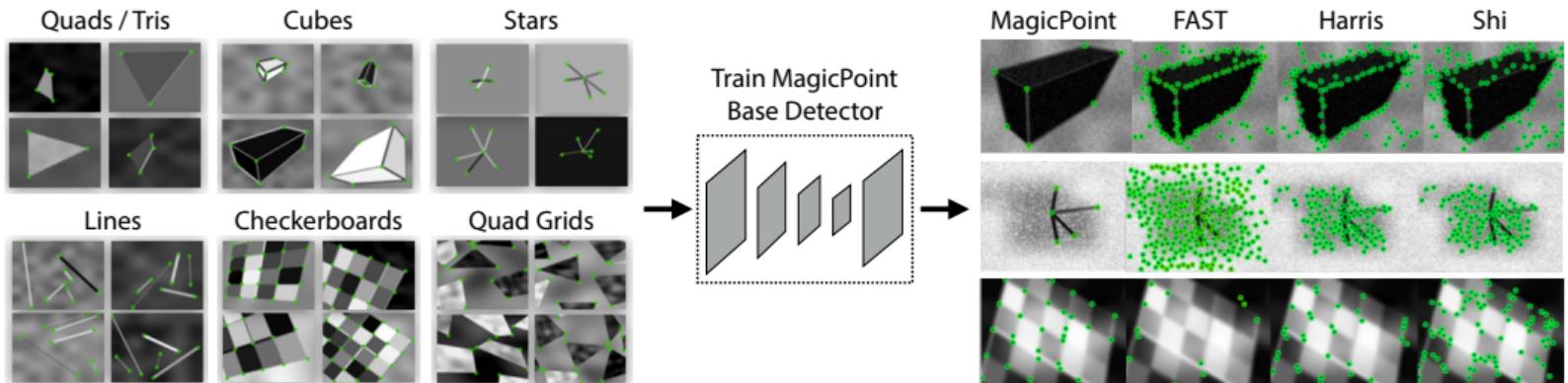


1 - Detekcia kľúčových bodov

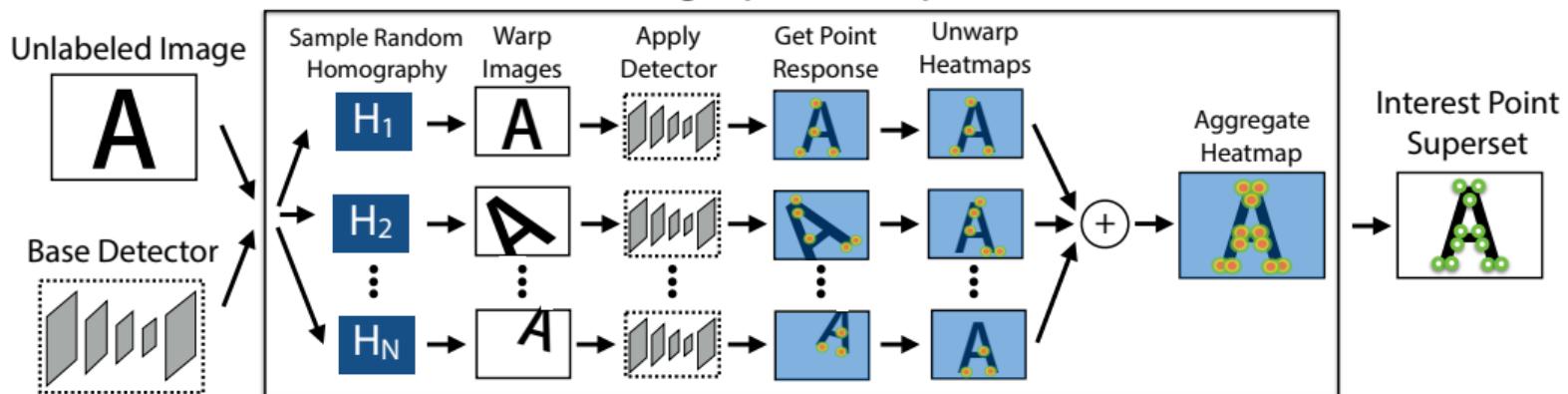
Klúčové body - SuperPoint



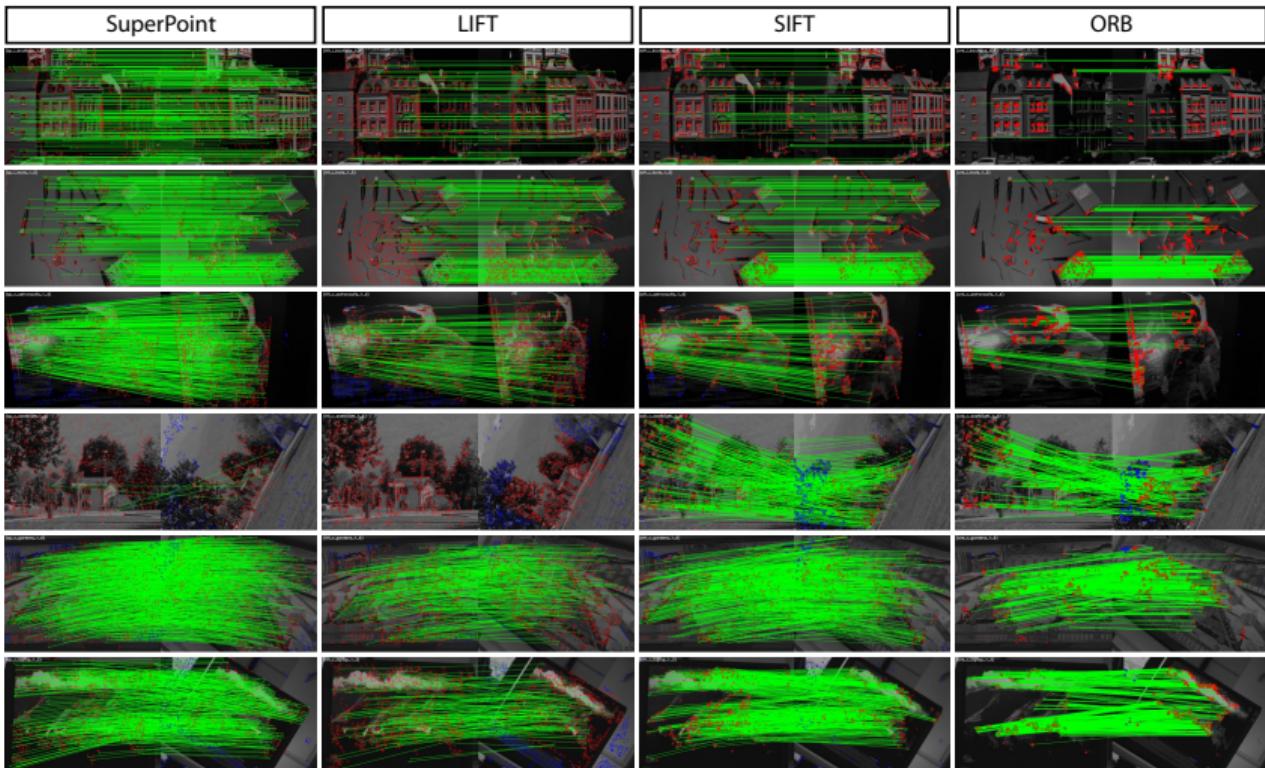
SuperPoint - Tréning I



Homographic Adaptation



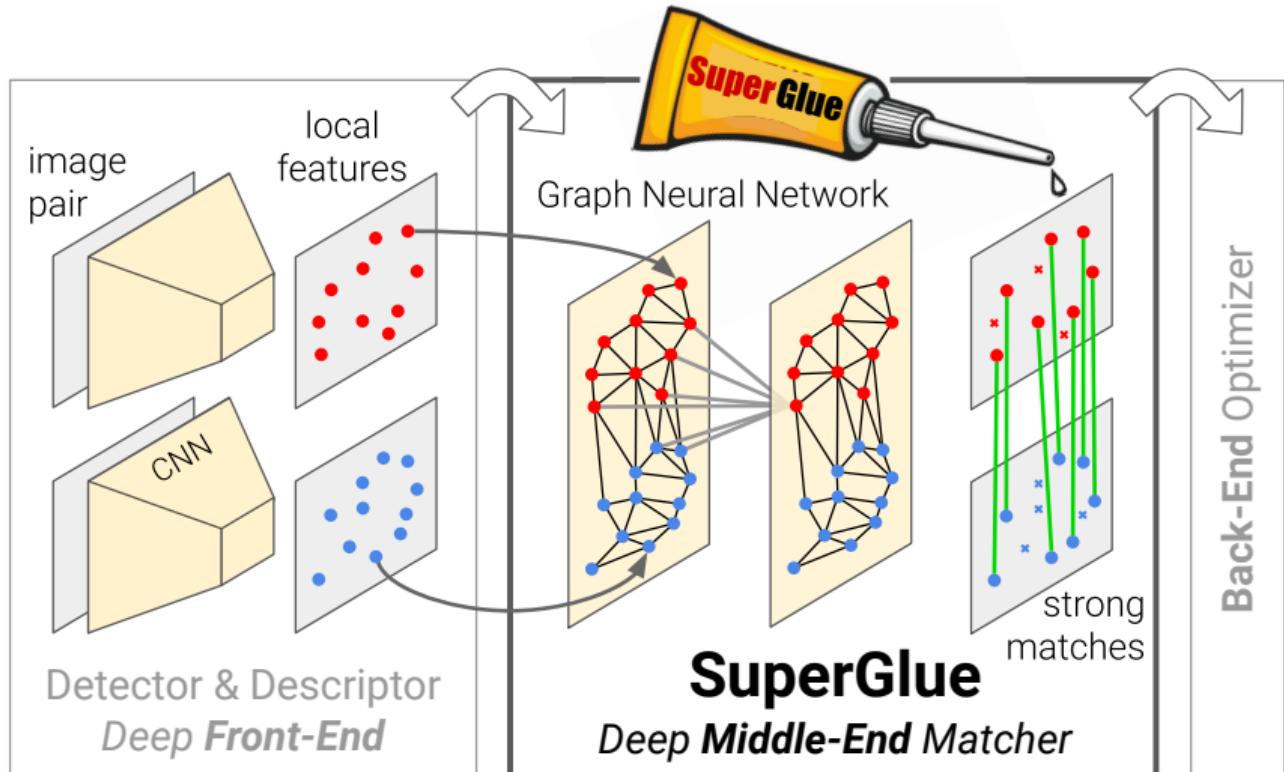
SuperPoint Porovnanie





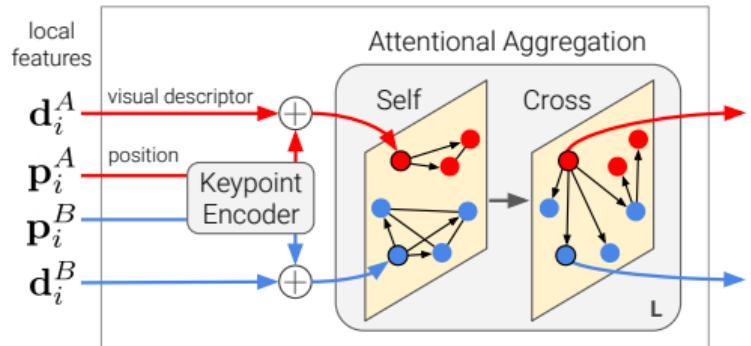
2 - Párovanie bodov

Lepší Matching - SuperGlue

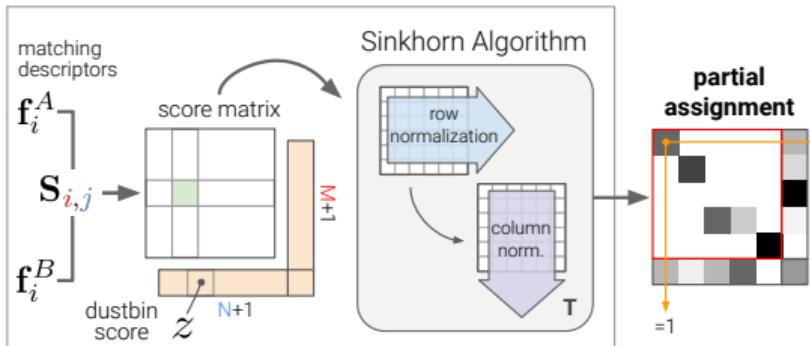




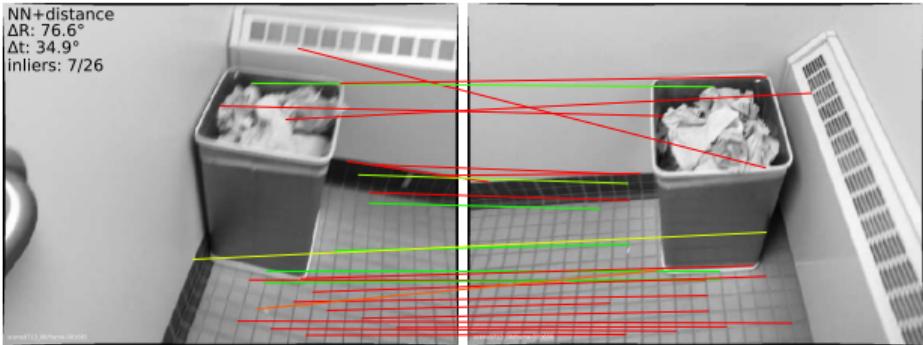
Attentional Graph Neural Network



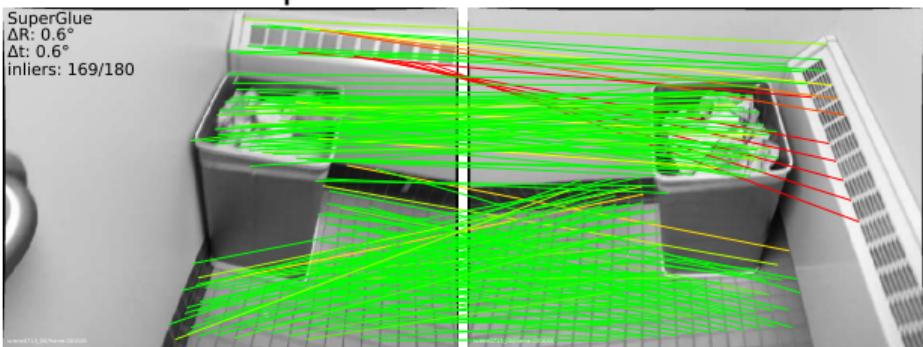
Optimal Matching Layer



SuperGlue



SuperPoint + NN + Dist



SuperPoint + SuperGlue

Structure from Motion



1. Detekcia klúčových bodov v obrázkoch z datasetu
2. Párovanie bodov medzi obrázkami



1. Detekcia kľúčových bodov v obrázkoch z datasetu
2. Párovanie bodov medzi obrázkami
3. Geometrická verifikácia korešpondencií - RANSAC
4. Optimalizácia modelu - Bundle Adjustment



Matematická vsuvka
Projektívna geometria

Homogénne súradnice



Na nasledujúcich slidoch budeme používať namiesto štandardných tzv. homogénne (projektívne) súradnice. Tie fungujú veľmi jednoducho. V prípade ak máme dvojrozmerný priestor, tak každý bod budeme reprezentovať trojicou (x, y, w) a bude platiť nasledovné



Na nasledujúcich slidoch budeme používať namiesto štandardných tzv. homogénne (projektívne) súradnice. Tie fungujú veľmi jednoducho. V prípade ak máme dvojrozmerný priestor, tak každý bod budeme reprezentovať trojicou (x, y, w) a bude platiť nasledovné

1. Ak w nie je 0, tak trojica (x, y, w) reprezentuje reálny bod $(\frac{x}{w}, \frac{y}{w})$.



Na nasledujúcich slidoch budeme používať namiesto štandardných tzv. homogénne (projektívne) súradnice. Tie fungujú veľmi jednoducho. V prípade ak máme dvojrozmerný priestor, tak každý bod budeme reprezentovať trojicou (x, y, w) a bude platiť nasledovné

1. Ak w nie je 0, tak trojica (x, y, w) reprezentuje reálny bod $(\frac{x}{w}, \frac{y}{w})$.
2. Ak w je 0, a $x \neq 0 \vee y \neq 0$ tak trojica $(x, y, 0)$ reprezentuje bod v nekonečne.



Na nasledujúcich slidoch budeme používať namiesto štandardných tzv. homogénne (projektívne) súradnice. Tie fungujú veľmi jednoducho. V prípade ak máme dvojrozmerný priestor, tak každý bod budeme reprezentovať trojicou (x, y, w) a bude platiť nasledovné

1. Ak w nie je 0, tak trojica (x, y, w) reprezentuje reálny bod $(\frac{x}{w}, \frac{y}{w})$.
2. Ak w je 0, a $x \neq 0 \vee y \neq 0$ tak trojica $(x, y, 0)$ reprezentuje bod v nekonečne.
3. Súradnice $(0, 0, 0)$ nereprezentujú žiadny bod.

Homogénne súradnice



Na nasledujúcich slidoch budeme používať namiesto štandardných tzv. homogénne (projektívne) súradnice. Tie fungujú veľmi jednoducho. V prípade ak máme dvojrozmerný priestor, tak každý bod budeme reprezentovať trojicou (x, y, w) a bude platiť nasledovné

1. Ak w nie je 0, tak trojica (x, y, w) reprezentuje reálny bod $(\frac{x}{w}, \frac{y}{w})$.
2. Ak w je 0, a $x \neq 0 \vee y \neq 0$ tak trojica $(x, y, 0)$ reprezentuje bod v nekonečne.
3. Súradnice $(0, 0, 0)$ nereprezentujú žiadny bod.

Je to rovnaké aj v 3D, ale tam máme štvorce (X, Y, Z, W) - pre prehľadnosť ich budeme značiť veľkými písmenami.

Homogénne súradnice



Z bežného bodu dostaneme homogénne súradnice pridaním 1 na koniec:
 $(1.4, 2.8) \rightarrow (1.4, 2.8, 1)$, alebo $(2, 5, 7) \rightarrow (2, 5, 7, 1)$.

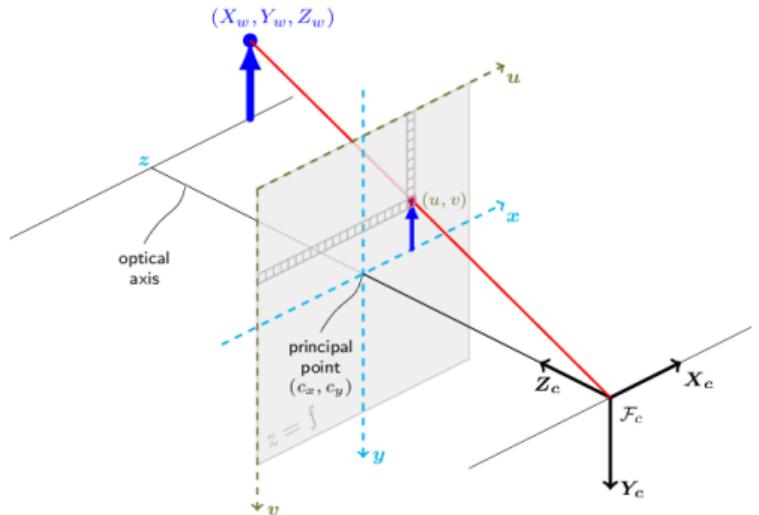
Homogénne súradnice



Z bežného bodu dostaneme homogénne súradnice pridaním 1 na koniec:
 $(1.4, 2.8) \rightarrow (1.4, 2.8, 1)$, alebo $(2, 5, 7) \rightarrow (2, 5, 7, 1)$.

Naopak len vydelíme posledným číslom:
 $(3, 5, 2) \rightarrow (1.5, 2.5)$, alebo $(8, 4.5, 9, 0.5) \rightarrow (16, 9, 18)$

Projekcia bodov



$$\begin{bmatrix} u \\ v \end{bmatrix} \simeq \begin{bmatrix} x \\ y \\ w \end{bmatrix} = K \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (1)$$



3 - Geometrická verifikácia korešpondencií



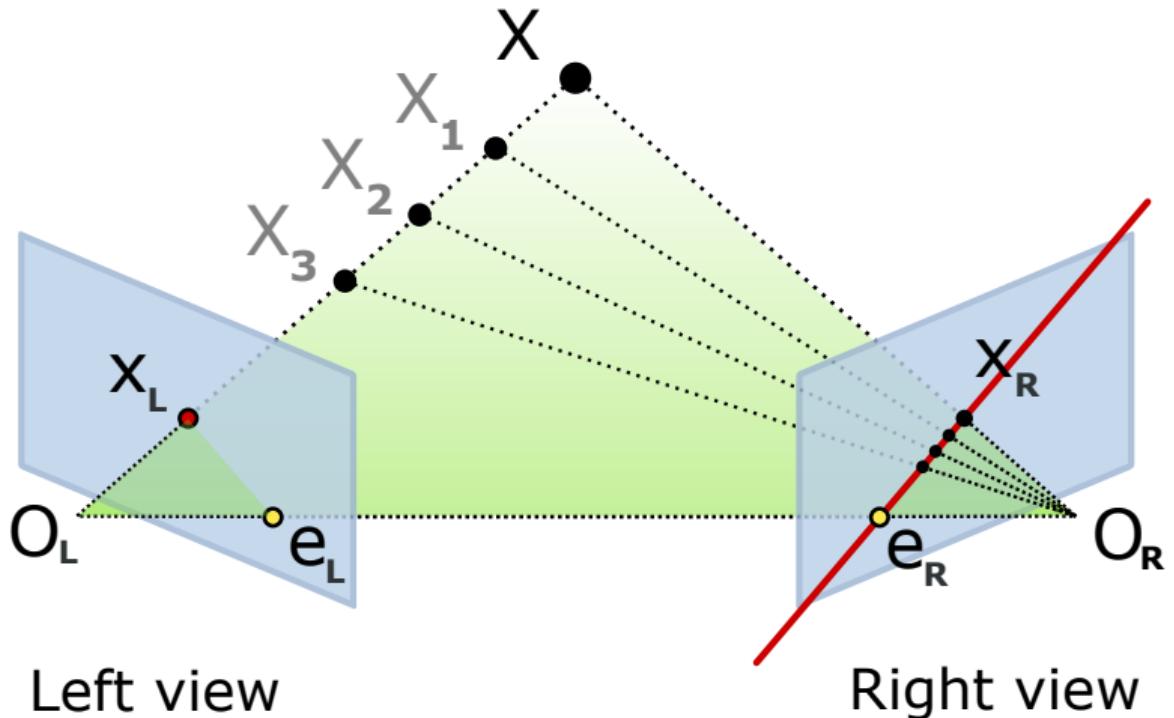
$$P = K \begin{bmatrix} R & \mathbf{t} \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \quad (2)$$
$$\mathbf{t} \in \mathbb{R}^3, R \in so(3)$$



$$P = K \begin{bmatrix} R & \mathbf{t} \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \quad (2)$$
$$\mathbf{t} \in \mathbb{R}^3, R \in so(3)$$

V skutočnosti je to zložitejšie, lebo kamery majú často ešte skreslenie, ale pre ilustráciu nám to bude stačiť.

Epipolárna geometria



Fundamentálna a esenciálna matica



Ak predpokladáme, že máme dve matice s projektívnymi maticami:

$$P = K \begin{bmatrix} \mathbb{I} & \mathbf{0} \end{bmatrix} \quad (3)$$

$$P' = K' \begin{bmatrix} R & \mathbf{t} \end{bmatrix} \quad (4)$$

Fundamentálna a esenciálna matica



Ak predpokladáme, že máme dve matice s projektívnymi maticami:

$$P = K \begin{bmatrix} \mathbb{I} & \mathbf{0} \end{bmatrix} \quad (3)$$

$$P' = K' \begin{bmatrix} R & \mathbf{t} \end{bmatrix} \quad (4)$$

tak potom všetko, čo vieme o geometrii medzi bodmi zobrazenými oboma kamerami sa dá obsiahnúť do tzv. *fundamentálnej matice* F :

$$F = K'^{-T} E K^{-1} = K'^{-T} [\mathbf{t}]_{\times} R K^{-1} \quad (5)$$

Prípadne ak poznáme vnútorné parametre matíc tak môžme využiť *esenciálnu maticu* E :

$$E = K'^T F K \quad (6)$$

Výpočet fundamentálnej matice



Na to aby sme vypočítali fundamentálnu maticu budeme potrebovať sedem korešpondencií $\mathbf{x}' \leftrightarrow \mathbf{x}$ bodov. Budeme potom riešiť systém pozostávajúci z rovníc:

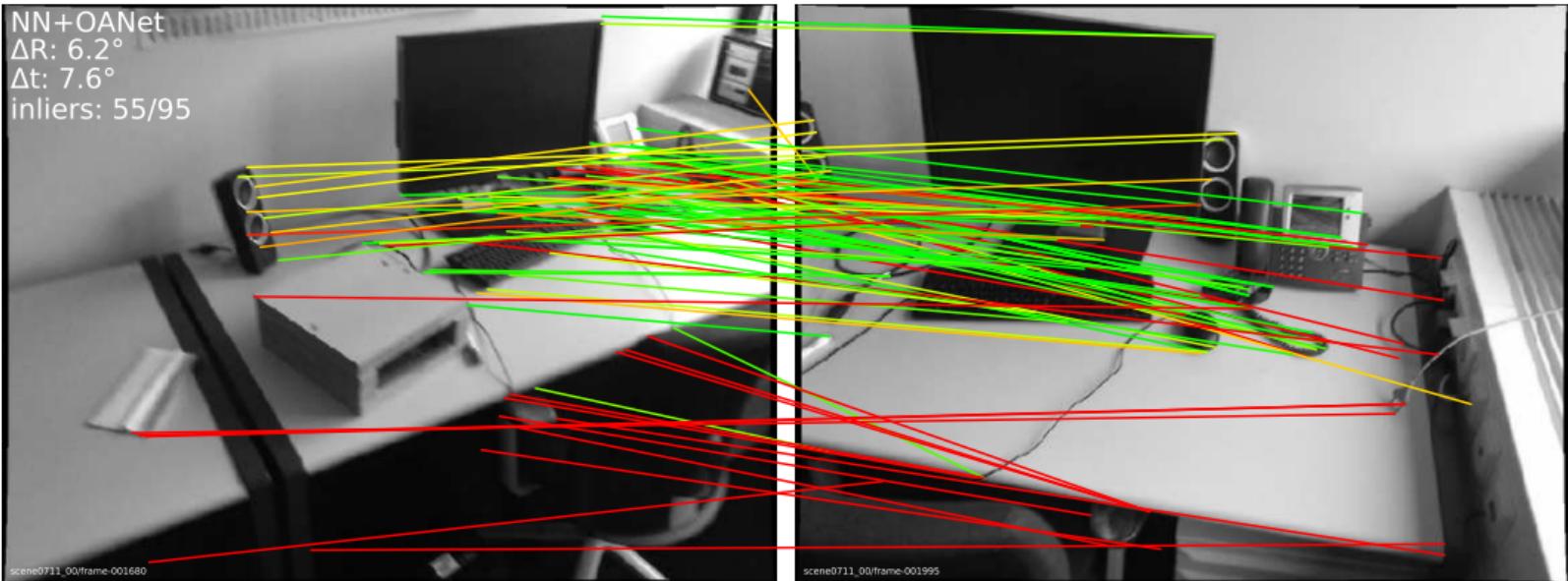
$$\mathbf{x}'^T E \mathbf{x} = \mathbf{0} \quad (7)$$

$$\det(F) = 0 \quad (8)$$

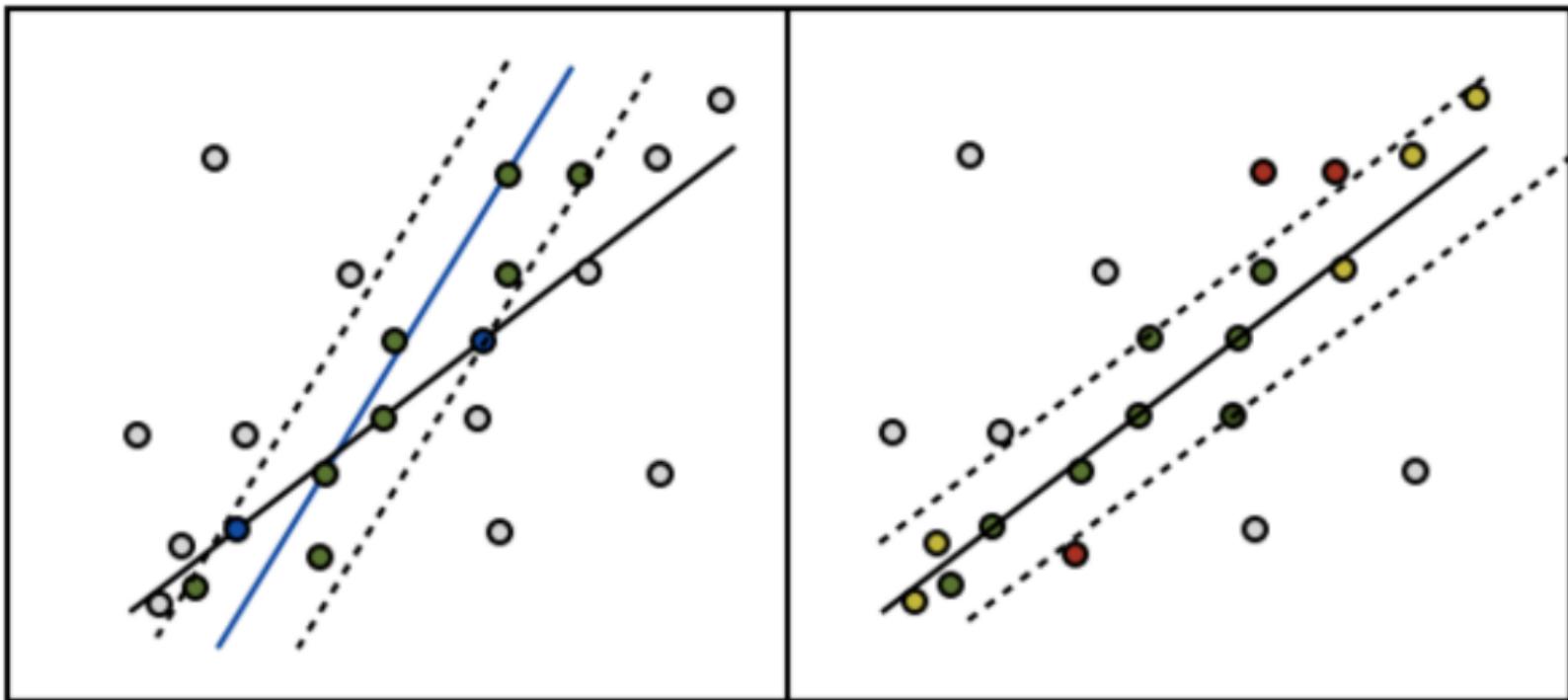
Korešpondencie



NN+OANet
 ΔR : 6.2°
 Δt : 7.6°
inliers: 55/95



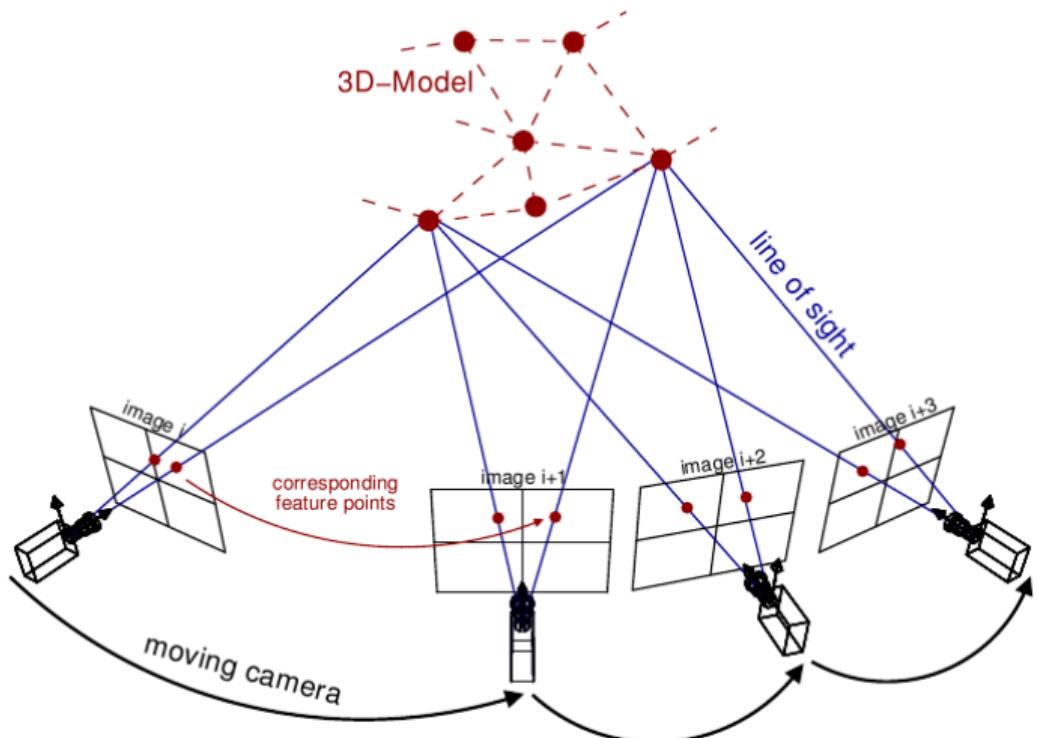
Ransac





1. Vyberieme 7 korešpondencií.
2. Vypočítame F podľa (7) a (8).
3. Podľa (7) spočítame kol'ko ostatných bodov suhlasí s f (v rámci nejakého prahu).
4. Bud' má F dostatok súhlasných bodov a končíme, alebo opakujem až do nejakého limitu opakovania a vyberieme najlepšie F .
5. Z bodov ktoré súhlasia s našim F môžeme ešte urobiť lepší odhad F pomocou metódy najmenších štvorcov.

Na čo nám to je?





1. Najprv začneme model pomocou dvoch obrazov:



1. Najprv začneme model pomocou dvoch obrazov:

1.1 Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie. Získame tak aj relatívnu pozíciu dvoch kamier.



1. Najprv začneme model pomocou dvoch obrazov:
 - 1.1 Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie. Získame tak aj relatívnu pozíciu dvoch kamier.
 - 1.2 Pomocou triangulácie spočítame zo súhlasných korešpondencií 3D body.



1. Najprv začneme model pomocou dvoch obrazov:

 - 1.1 Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie. Získame tak aj relatívnu pozíciu dvoch kamier.
 - 1.2 Pomocou triangulácie spočítame zo súhlasných korešpondencií 3D body.
2. Potom pridávame k rekonštrukcii vždy po jednom ďalšie obrazy:



1. Najprv začneme model pomocou dvoch obrazov:

 - 1.1 Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie. Získame tak aj relatívnu pozíciu dvoch kamier.
 - 1.2 Pomocou triangulácie spočítame zo súhlasných korešpondencií 3D body.
2. Potom pridávame k rekonštrukcii vždy po jednom ďalšie obrazy:

 - 2.1 Podobným postupom ako v 1.1 spočítame relatívnu pozíciu kamery voči 3D modelu pomocou 2D-3D korešpondencií (PnP)



1. Najprv začneme model pomocou dvoch obrazov:
 - 1.1 Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie. Získame tak aj relatívnu pozíciu dvoch kamier.
 - 1.2 Pomocou triangulácie spočítame zo súhlasných korešpondencií 3D body.
2. Potom pridávame k rekonštrukcii vždy po jednom ďalšie obrazy:
 - 2.1 Podobným postupom ako v 1.1 spočítame relatívnu pozíciu kamery voči 3D modelu pomocou 2D-3D korešpondencií (PnP)
 - 2.2 Do 3D modelu pridáme nové korešpondencie pomocou triangulácie



4 - Globálna optimalizácia

Problémy doterajšieho riešenia



V doterajšom výpočte budú určite chyby:

- Body z korešpondencií nebudú určené presne - limitácia na pixely + šum
- Matematický model kamery je inherentným zjednodušením reálneho fyzikálneho procesu.

Tieto chyby sa navyše môžu nabalovať a pri inkrementálnom pridávaní obrazov už nie je jednoznačné ako späťne meniť pozície kamier, resp. pozície triangulovaných bodov.

Riešenie: Budeme optimalizovať naraz polohu všetkých 3D bodov z modelu a kamier.



Budeme teda hľadať takú konfiguráciu, ktorá minimalizuje reprojekčnú chybu:

$$\arg \min_{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n} \sum_{i=1}^m \sum_{j=1}^n \|\alpha_{ij} \mathbf{x}_{ij} - P_i \mathbf{X}_j\|$$

Tento problém potom riešime pomocou Levenberg–Marquardtovho algoritmu. Na inicializáciu použijeme konfiguráciu, ktorú sme získali predchádzajúcim postupom.

Bundle Adjustment



Budeme teda hľadať takú konfiguráciu, ktorá minimalizuje reprojekčnú chybu:

$$\arg \min_{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n, R_1, \mathbf{t}_1, R_2, \mathbf{t}_2, \dots, R_m, \mathbf{t}_m} \sum_{i=1}^m \sum_{j=1}^n \|\alpha_{ij} \mathbf{x}_{ij} - P_i \mathbf{X}_j\|$$

Tento problém potom riešime pomocou Levenberg–Marquardtovho algoritmu. Na inicializáciu použijeme konfiguráciu, ktorú sme získali predchádzajúcim postupom.

Bundle Adjustment



Budeme teda hľadať takú konfiguráciu, ktorá minimalizuje reprojekčnú chybu:

$$\arg \min_{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, P_1, P_2, \dots, P_m} \sum_{i=1}^m \sum_{j=1}^n \|\alpha_{ij} \mathbf{x}_{ij} - P_i \mathbf{x}_j\|$$

Tento problém potom riešime pomocou Levenberg–Marquardtovho algoritmu. Na inicializáciu použijeme konfiguráciu, ktorú sme získali predchádzajúcim postupom.



- 1.** Najprv začneme model pomocou dvoch obrazov:
 - 1.1** Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie.
Získame tak aj relatívnu pozíciu dvoch kamier.
 - 1.2** Pomocou triangulácie spočítame zo súhlasných korešpondencií 3D body.
- 2.** Potom pridávame k rekonštrukcii vždy po jednom ďalšie obrazy:
 - 2.1** Podobným postupom ako v 1.1 spočítame relatívnu pozíciu kamery voči 3D modelu pomocou 2D-3D korešpondencií (PnP)
 - 2.3** Do 3D modelu pridáme nové korešpondencie pomocou triangulácie

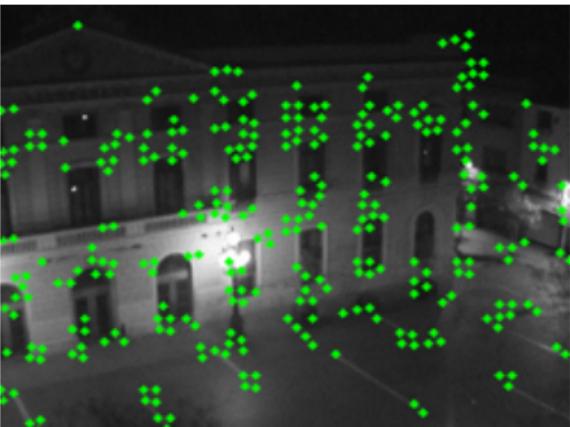


1. Najprv začneme model pomocou dvoch obrazov:
 - 1.1 Pre tieto dva obrazy spočítame F resp. E a k nim súhlasné korešpondencie.
Získame tak aj relatívnu pozíciu dvoch kamier.
 - 1.2 Pomocou triangulácie spočítame zo súhlasných korešpondencií 3D body.
 - 1.3 **Bundle Adjustment**
2. Potom pridávame k rekonštrukcii vždy po jednom ďalšie obrazy:
 - 2.1 Podobným postupom ako v 1.1 spočítame relatívnu pozíciu kamery voči 3D modelu pomocou 2D-3D korešpondencií (PnP)
 - 2.2 **Bundle Adjustment**
 - 2.3 Do 3D modelu pridáme nové korešpondencie pomocou triangulácie
 - 2.4 **Bundle Adjustment**



Rekapitulácia

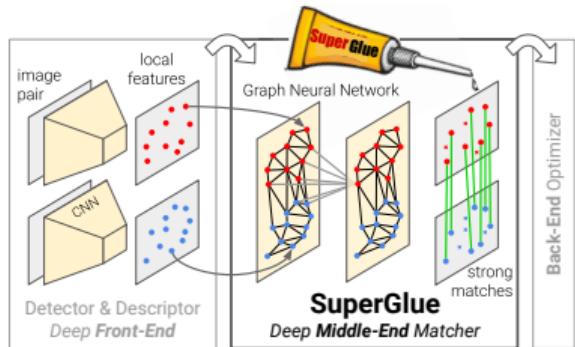
1. Detekcia klúčových bodov v obrázkoch z datasetu



Structure from Motion



1. Detekcia klúčových bodov v obrázkoch z datasetu
2. Párovanie bodov medzi obrázkami





1. Detekcia klúčových bodov v obrázkoch z datasetu

$$\mathbf{x}'^T F \mathbf{x} = 0$$

2. Párovanie bodov medzi obrázkami

$$\det(F) = 0$$

3. Geometrická verifikácia korešpondencií - RANSAC



1. Detekcia klúčových bodov v obrázkoch z datasetu
2. Párovanie bodov medzi obrázkami
3. Geometrická verifikácia korešpondencií - RANSAC
4. Optimalizácia modelu - Bundle Adjustment

$$\arg \min_{\mathbf{X}_1, \mathbf{X}_2, \dots} \sum_{i=1}^m \sum_{j=1}^n \|x_{ij} - P_i \mathbf{X}_j\|$$

Lokalizácia



Lokalizáciu d'alšiho obrazu potom môžeme realizovať rovnako ako pri iteratívnom pridávaní obrazov do modelu!

Demo



Demo nájdeme na adrese: <https://colab.research.google.com/drive/1MrVs9b8aQYODt0GkoaNf9Nji3sbCNMQ>

Skúste si rekonštrukciu matfyzu pomocou linku:

https://liveuniba-my.sharepoint.com/:u/g/personal/kocur15_uniba_sk/EVW4VwHy5ldBhSEtRw2Q0moBAi_Evee3LhDWc5ae81zRrQ?e=MJdH7o\&download=1



Ked'že vieme dostať pozíciu kamery, tak vlastne zároveň vieme aj relatívnu pozíciu objektu voči kamere! Skúsime si to teda v úlohe.

Stiahnite si dátu na adrese:

https://liveuniba-my.sharepoint.com/:u/g/personal/kocur15_uniba_sk/EUb4y4ho8WxFmvTmU4EOSxYBwLyfpbRfM_U_R0ioND0vsg?e=Z21TYA&download=1

Ciel' úlohy



Cieľom úlohy bude dokresliť si do query obrázkov (nepoužijeme ich na vytvorenie modelu) stred bodov z modelu a vyznačiť aj orientáciu objektu pomocou hlavných komponentov, ktoré získame vd'aka PCA.



3D body z modelu môžeme získať nasledovným spôsobom z objektu `model`, ktorý sme získali počas rekonštrukcie:

```
pts = np.array([p.xyz for p in model.points3D.values()])
```

Z týchto bodov potom spočítame ich priemer a tri komponenty pomocou PCA. Na PCA môžete použiť napríklad <https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>.

Následne do obrázku nakreslíme čiary idúce zo stredu smerom v troch hlavných komponentoch. Kresliť môžete bud' pomocou matplolibu, OpenCV, alebo akýmkol'vek iným spôsobom.



Aby sme si vykreslili 3D body do obrázkov, tak si ich budeme musieť premietnuť do roviny obrazu. To môžeme spraviť keďže vieme získať aj vnútorné parametre kamery (objekt camera z dema) a aj jej polohu (pose).

```
P_img_3d = pose.transform_to_image(P_3d)  
p_2d = camera.world_to_image(P_img_3d[:2]/P_img_3d[2])
```

Pričom P_3d je numpy pole dĺžky 3 reprezentujúce bod ktorý chceme zobrazit'. Bod p_2d potom bude bod v pixelových súradniciach.