9. Vizuális odometria és SLAM

Kató Zoltán

Képfeldolgozás és Számítógépes Grafika tanszék
SZTE
(http://www.inf.u-szeged.hu/~kato/teaching/)

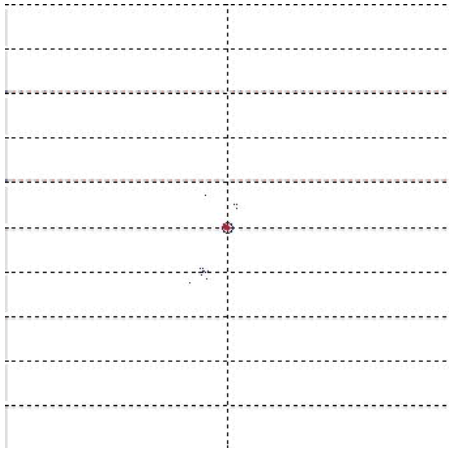
Vizuális odometria és SLAM fogalma

- Vizuális odometria (VO): a mozgó robot/jármű helyzetének inkrementális becslése a reá szerelt kamera (vagy kamerák) képében a mozgás által előidézett változás alapján
- Vizuális SLAM (Simultaneous Localization and Mapping): A globális útvonal konzisztens meghatározása és feltérképezése (3D struktúra)
  - A térkép lényegében 3D pozícióval rendelkező jellemzők gyűjteménye, melyet többféleképpen reprezentálhatunk
  - Fontos része a visszatérési pontok felismerése, és a körutak zárása

input

Képsorozat (videófolyam) a mozgó robotra szerelt kamerá(k)ból

output



Kamera útvonal (3D struktúra is előállítható):

$R_0, R_1, \ldots, R_i$

$t_0, t_1, \ldots, t_i$

Vizuális odometria és SLAM fogalma

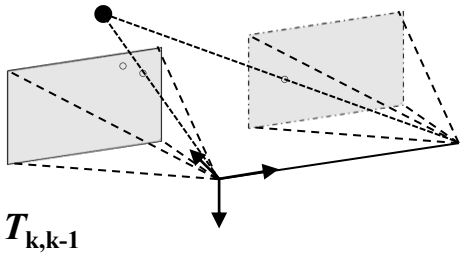
Vizuális odometria

- lokális konzisztencia
  - Néhány képkockára kiterjedő útvonal
- SLAM egy építőeleme
- Egyszerűbb algoritmus
  - Nincs szükség a teljes útvonal nyilvántartására
- Valós idejű futás, de globálisan inkozisztens

SLAM

- globális konzisztencia
  - A teljes útvonalra és térképre megkövetelt
- Körutak bezárása, és ez alapján az akkumulált hiba korrigálása
- Komplex algoritmus
  - pl. isszatérési helyek azonosítása
- Hosszabb futásidő, de konzisztens eredmény

A VO feladata



- Mozgó robotra szerelt kamerarendszer *k* diszkrét időpontokban képet készít az általa látott környezetről
  - Lehet egyetlen kamera, vagy szeteró kamera pár, esetleg több kamerából álló rendszer
  - A kamerák minden esetben mereven rögzítettek, önmagukban nem mozognak
  - A rögzített képsorozatot $\{I_0, I_1, \ldots, I_n\}$ jelöljük
- A kamera k-1 és k időpillanatokban vett pozíciói között egy T_k 3D merev test transzformáció hat:

$$T_k = (R_{k,k-1}, t_{k,k-1})$$

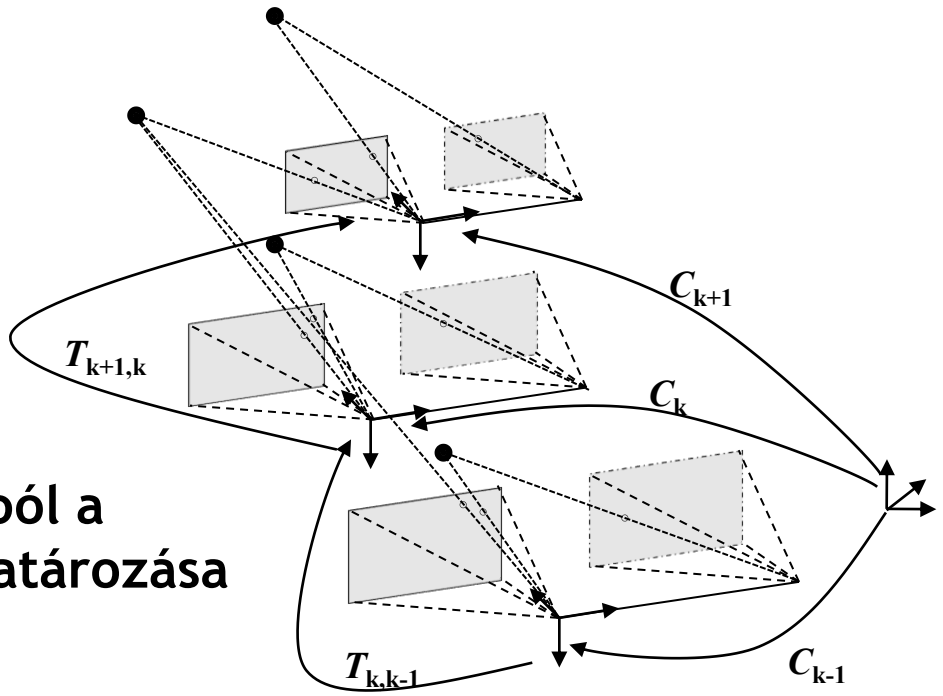
  - $\{T_1, T_2, \ldots, T_n\}$ sorozat a kamera által megtett minden mozgást leír
- A $\{C_0, C_1, \ldots, C_n\}$ kamera pozíciók pedig megadják a kamerák helyzetét a kezdeti k=0 pozícióhoz képest

A VO feladata

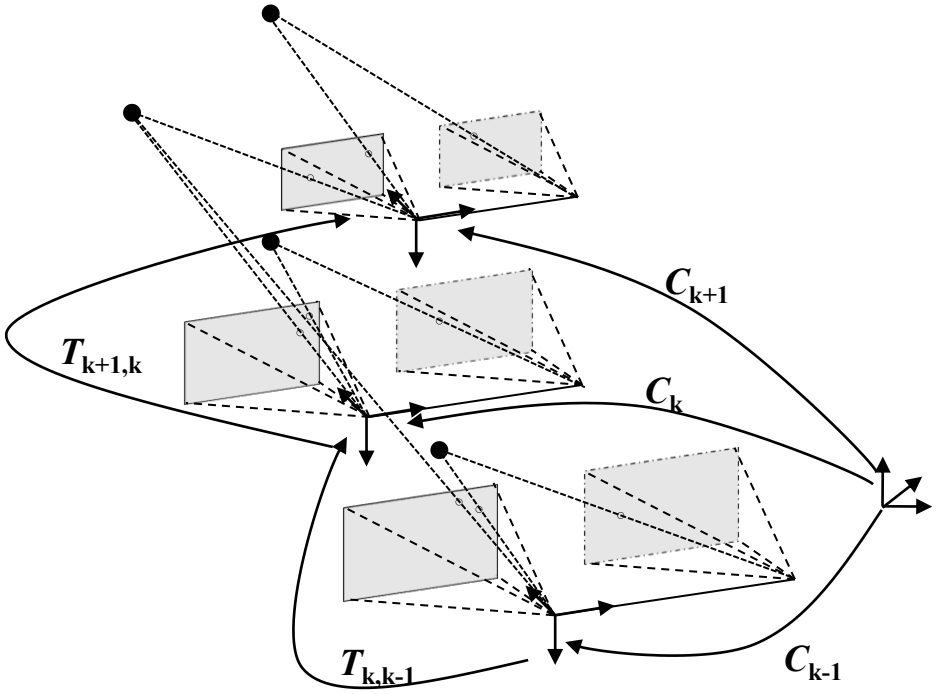
- A C_0 kezdeti helyzet megválasztható önkényesen a felhasználó által
  - lényegében a világ koordináta rendszer megadásáról van szó
- Ezek után C_n megkapható a T_k, k=1,…,n transzformációk konkatenációjával, vagyis:

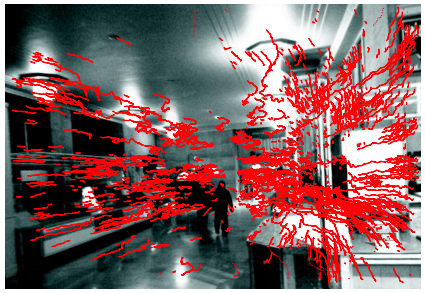
$$C_n = T_n C_{n-1}$$

- A VO feladata tehát
  - Az egymást követő kamera helyzetek közötti T_k transzformáció meghatározása az I_k és I_{k-1} képkockákból,
  - majd ezek konkatenációjából a $C_{0\rightarrow n}$ kamera útvonal meghatározása



Vizuális odometria algoritmusa

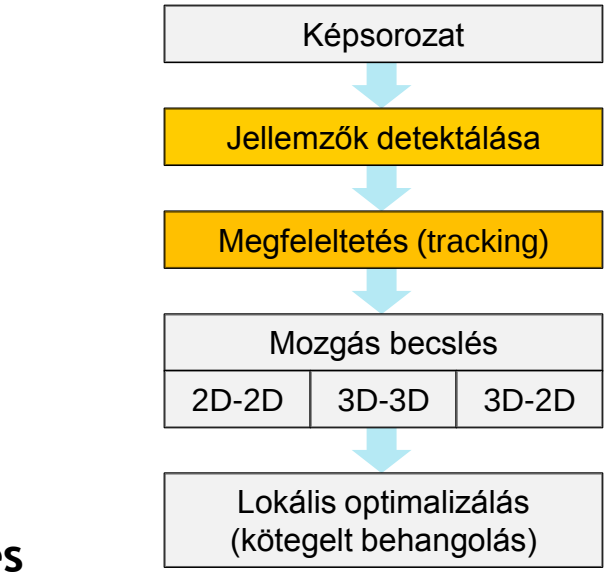


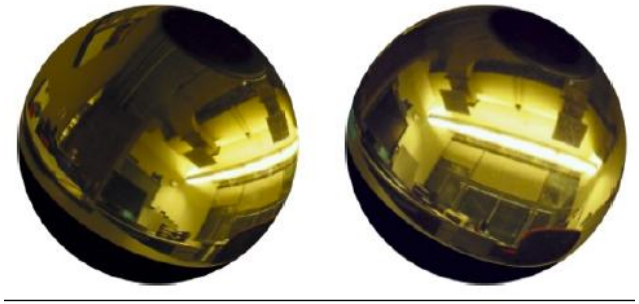


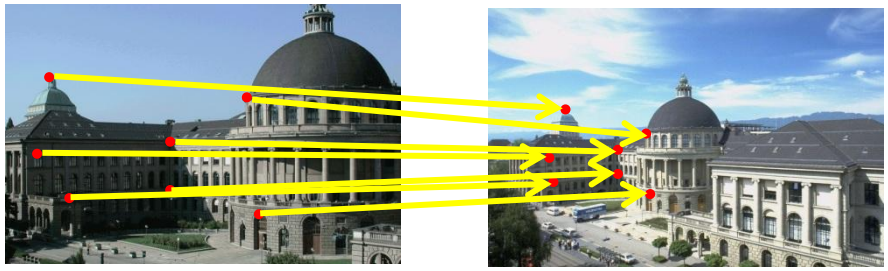
SIFT jellemzők követése

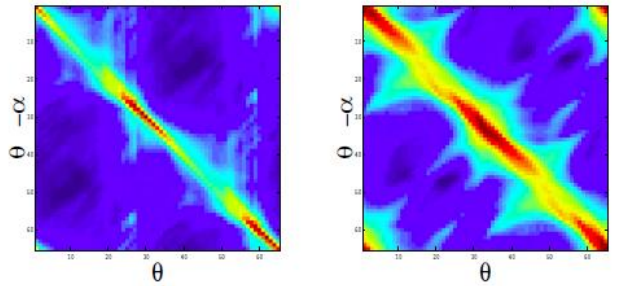
Jellemzők megfeleltetése

- **Kétféle módszer:**
  - **Korreláció alapú sűrű megfeleltetés (hasonlóan a sűrű sztereó megfeleltetéshez)**
    - **Kevésbé elterjedt, mert számításigényes és általában pontatlanabb is**
  - **Jellemző pontok megfeleltetése és mozgásának követése**
    - **Tipikusan ezt használják robusztus megfeleltetéssel/követéssel**

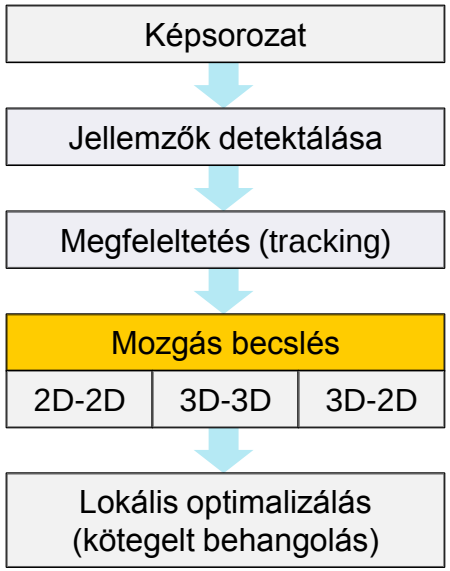








Mozgás becslés

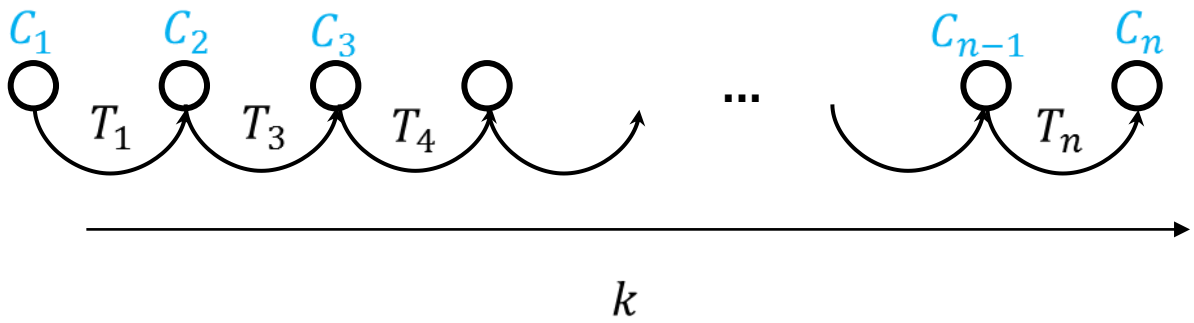


- A kinyert f_{k-1}, f_k megfeleltetések alapján a

$$\mathbf{T}_k = \begin{bmatrix} \mathbf{R}_{k,k-1} & \mathbf{t}_{k,k-1} \\ 0 & 1 \end{bmatrix}$$

kamera elmozdulás kiszámolása az aktuális és az előző kép között

- Majd ezek konkatenációjából a teljes kamera útvonal rekonstrukciója:



Mozgás becslés

- 2D-2D: az f_{k-1} és f_k jellemzők 2D képkoordinátában adottak

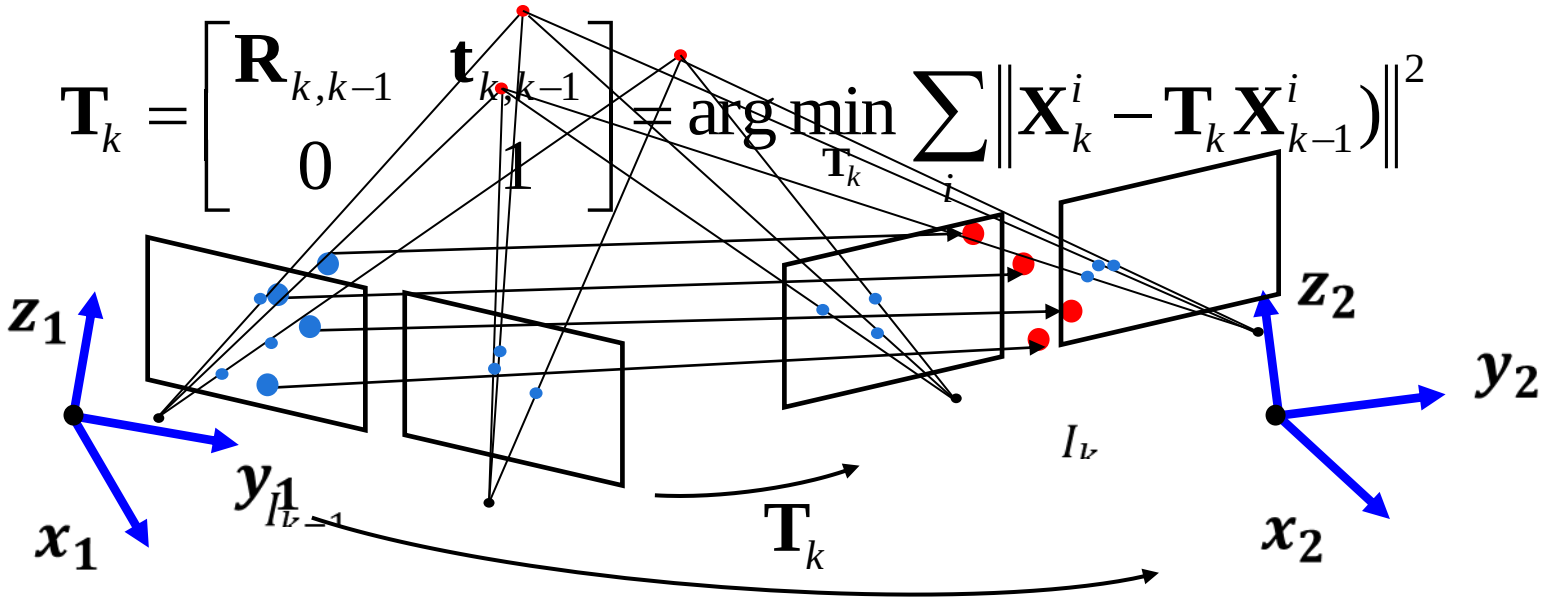
- 3D-3D: az f_{k-1} és f_k jellemzők 3D koordináták. Ekkor a 3D pontokat minden időpillanatban háromszögeléssel elő kell állítani (pl. szeteró kamerák alkalmazásával)

- 3D-2D: az f_{k-1} 3D koordinátákkal adott, míg f_k a pontok megfelelői (visszavetítettjei) az I_k képen
  - A 3D pontokat szintén elő kell állítani, de ezt most megtehetjük egyetlen kamera alkalmazásával is, ahol az előző pozíciókban meghatározott kameraképekből rekonstruáljuk az előző képkocka pontjait.

3D – 3D

- f_{k-1} és f_k jellemzők 3D koordinátában adottak
  - Külön-külön háromszögeléssel meghatároztuk szeteró kamerák alkalmazásával
- Minimum 3 pontmegfeleltetés szükséges (lineárisan független pontokban)
- A transzformáció a legjobb regisztráció lesz, ami minimalizálja az X^i pontok 3D távolságát:

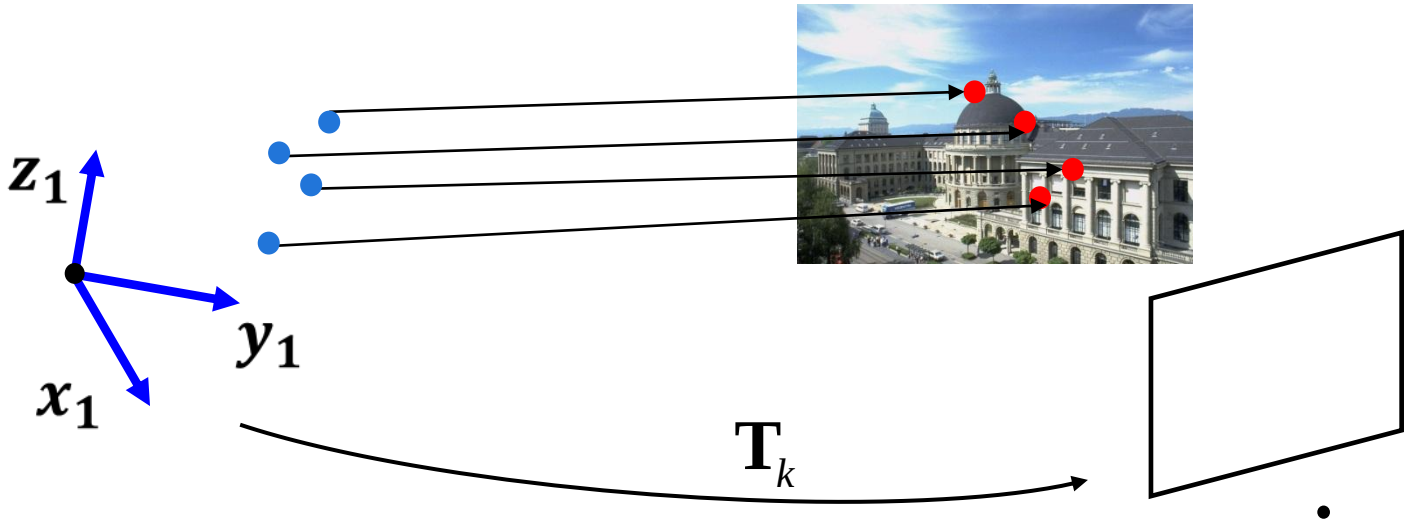
$$\mathbf{T}_k = \begin{bmatrix} \mathbf{R}_{k,k-1} & \mathbf{t}_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{\mathbf{T}_k} \sum_i \left\| \mathbf{X}_k^i - \mathbf{T}_k \mathbf{X}_{k-1}^i \right\|^2$$



3D - 2D

- f_{k-1} 3D és f_k 2D koordinátában adottak
  - Egy kamera esetén f_{k-1} 3D pontokat az előző két pozícióból készült képek (I_{k-1} és I_{k-2}) alapján kapjuk háromszögeléssel
- Minimum 3 pontmegfeleltetés szükséges a megoldáshoz
- A megoldást a visszavetítési hiba minimalizálásával kapjuk:

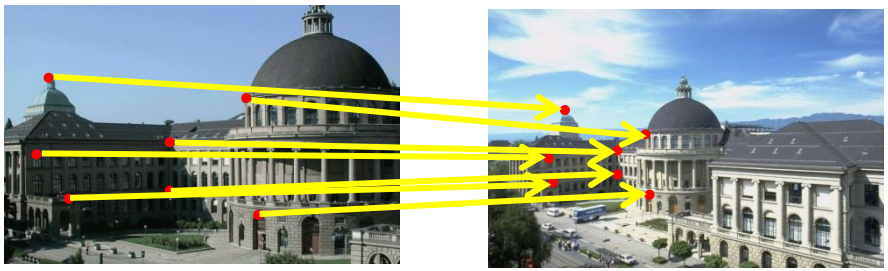
$$\mathbf{T}_k = \begin{bmatrix} \mathbf{R}_{k,k-1} & \mathbf{t}_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{\mathbf{C}_k} \sum_i \left\| \mathbf{p}_k^i - g(\mathbf{X}_i, \mathbf{C}_k) \right\|^2$$



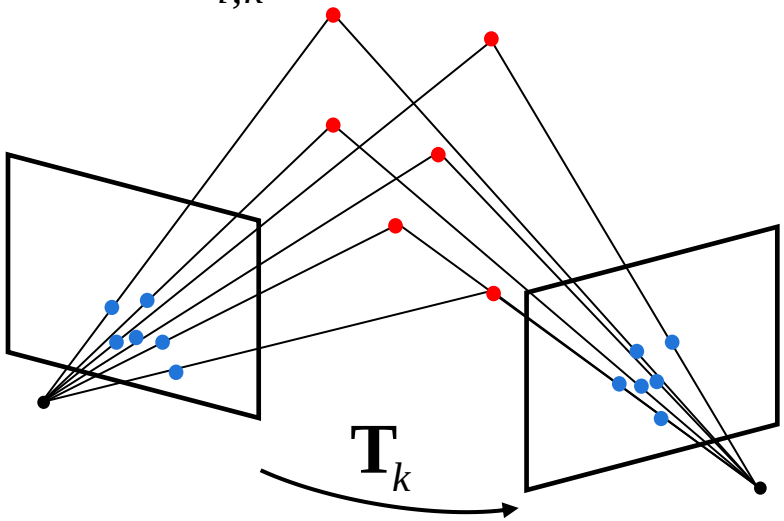
2D - 2D

- f_{k-1} és f_k jellemzők 2D képkoordinátában adottak
- Minimum 5 pontmegfeleltetés szükséges (de a gyakorlatban inkább 8-pontos algoritmust használnak)
- A transzformációt a háromszögeléssel kapott 3D X_i pontok visszavetítési hibájának minimalizálásával kapjuk:

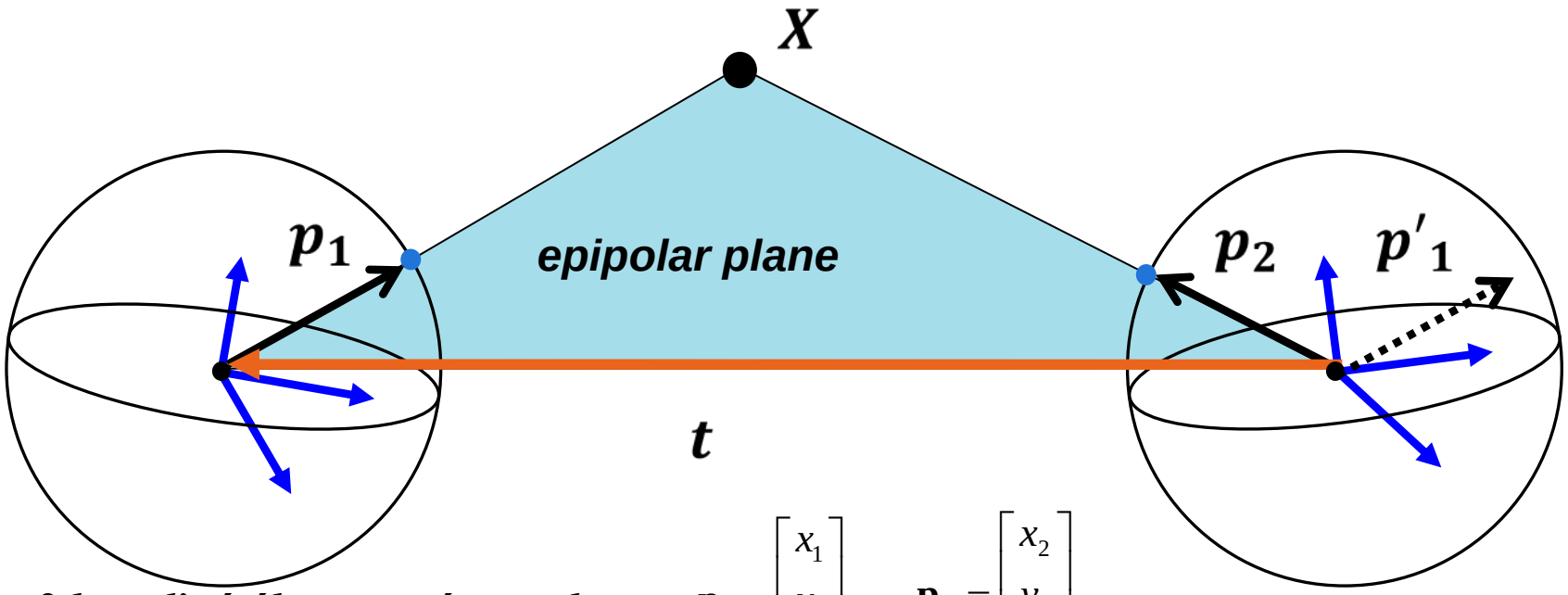
$$\mathbf{T}_k = \begin{bmatrix} \mathbf{R}_{k,k-1} & \mathbf{t}_{k,k-1} \\ 0 & 1 \end{bmatrix} = \arg\min_{\mathbf{X}_i, \mathbf{C}_k} \sum_{i,k} \left\| \mathbf{p}_k^i - g(\mathbf{X}_i, \mathbf{C}_k) \right\|^2$$



I_{k-1} I_k



2D - 2D epipoláris geometria



P1, p2 koordináták az egység gömbön: $\mathbf{p}_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$ $\mathbf{p}_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$

P1,p2, t egy síkba esik

$$\mathbf{p}_2^T \cdot (\mathbf{t} \times \mathbf{p}_1') = 0 \qquad \Rightarrow \mathbf{p}_2^T \cdot (\mathbf{t} \times (\mathbf{R}\mathbf{p}_1)) = 0$$

$$\Rightarrow \mathbf{p}_2^T [\mathbf{t}]_\times \mathbf{R}\mathbf{p}_1 = 0 \qquad \Rightarrow \mathbf{p}_2^T \mathbf{E}\, \mathbf{p}_1 = 0$$

Epipoláris kényszer: E meghatározható a 8-pontos algoritmussal (SVD)

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$$

Esszenciális mátrixból R,t faktorizálható (SVD)

2D – 2D mozgás becslés algoritmusa

1. Új kép készítése ➔ I_k
2. Jellemzők kinyerése és megfeleltetése I_k és I_{k-1} között
3. Esszenciális mátrix meghatározása, amit faktorizálva megkapjuk R_k és t_k ➔ T_k
4. Számítsuk ki a relatív skálázást és ezzel skálázzuk t_k -t
5. Kamera helyzetének meghatározása: $C_k = T_k C_{k-1}$
6. Vissza az 1. lépésre.

- Hogyan számoljuk a relatív skálát?

Relatív skála meghatározása

- Két kép alapján az eltolás abszolút skálája nem határozható meg
- Relatív skála az egymást követő elmozdulások között azonban meghatározható:
  - Háromszögeléssel határozzunk meg X_{k-1} és X_k pontokat a két egymást követő képből
  - Egymásnak megfelelő pontokból kiszámíthatjuk a relatív távolságot
  - A relatív skála ezután megkapható egy X_{k-1} és X_k pontpár távolságának arányából:

$$r = \frac{||X_{k-1,i} - X_{k-1,j}||}{||X_{k,i} - X_{k,j}||}$$

  - Robusztusság : több pontpárra is számoljuk r értékét, majd átlagolunk, vagy mediant veszünk (kilógó adatokra kevésbé érzékeny)

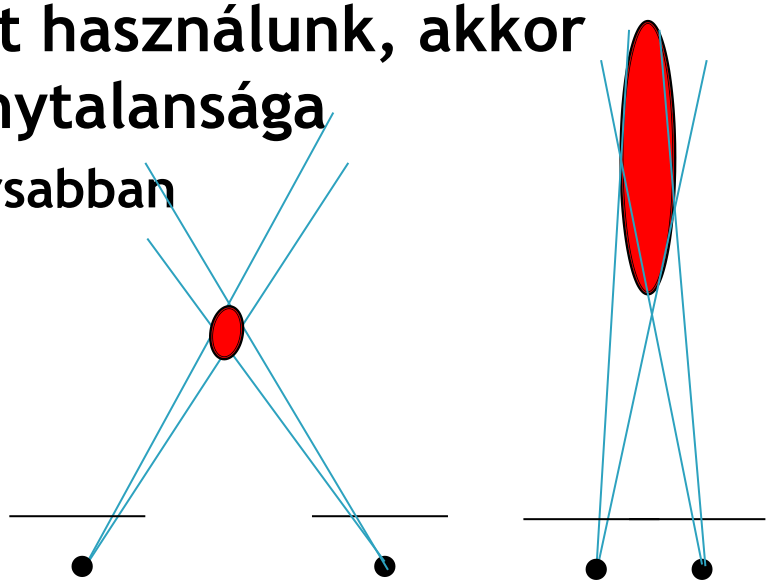
Mozgás becslés

- Összefoglalva: a mozgásmegfeleltetések típusa és az alkalmazott kamerarendszer közötti összefüggés:

Megfeleltetés típusa	Egyetlen kamera	Sztereó kamerapár
2D-2D	X	X
3D-3D		X
3D-2D	X	X

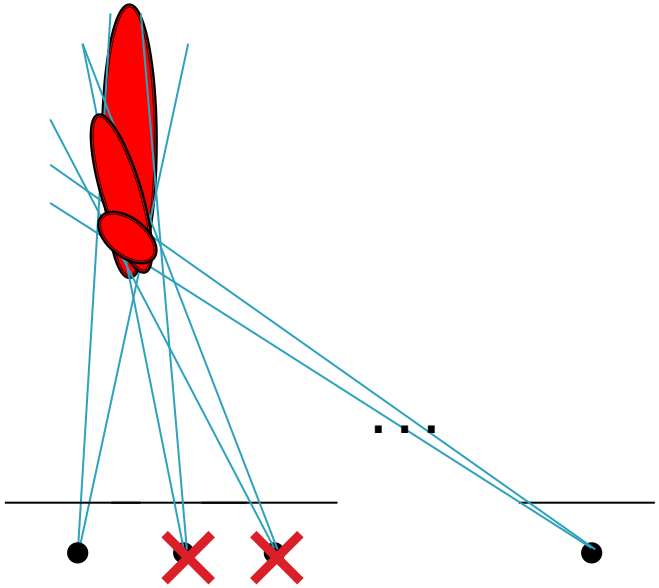
Háromszögelés

- A 3D alapú mozgáskövetéshez szükséges a pontok rekonstrukciója háromszögeléssel
  - Ezt legalább két megfeleltetett képpont visszavetítésével határozzuk meg
  - Ezek a sugarak azonban a valóságban nem fogják egymást metszeni (zaj, kamera mátrix és megfeleltetési hibák miatt)
  - Ezért a sugarakhoz legközelebbi 3D pontot vesszük
- Ha egymáshoz közeli pozíciókat használunk, akkor nagy lesz a rekonstrukció bizonytalansága
  - Ezért a 3D-3D mozgásbecslés gyorsabban növeli az akkumulált hibát



Háromszögelés

- A háromszögelésből eredő hibát minimalizálhatjuk, ha a túl közeli képeket kihagyjuk mindaddig, amig az átlagos rekonstrukciós hiba egy küszöb alá nem csökken
  - Az így kapott képkockát *kulcskockának* nevezzük.
  - Ez a VO nagyon fontos lépése, és mindig a mozgásbecslés előtt kell végrehajtani.

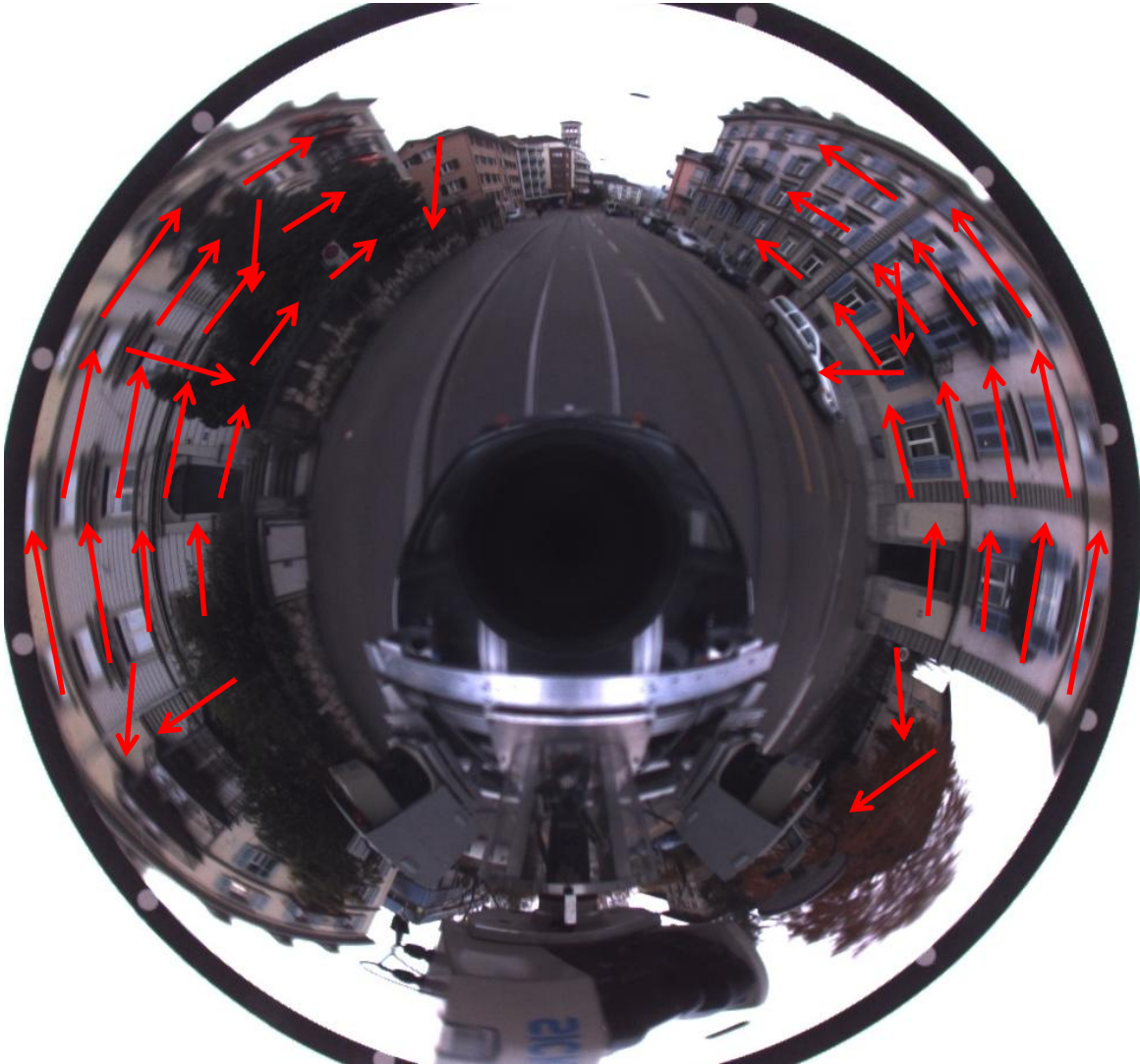


Sztereo vagy mono kamera rendszer?

- Sztereo kamerarendszer előnye, hogy egyszerre kapjuk meg a 3D struktúrát (térképezés) és a pozíciót metrikus térben (vagyis helyes skálán)
  - 3D-2D összességében kisebb hibát eredményez, mint a 3D-3D
  - A *kulcskockák* kiválasztása kritikus a hiba minimalizálásához
- Ha a bázistávolság elhanyagolható az objektumokhoz, akkor a sztereo rendszer egyenértékű lesz egyetlen kamerával.
- Mindkét esetben fontos azonban a kilógó adatok (outlierek) kezelése ➔ RANSAC

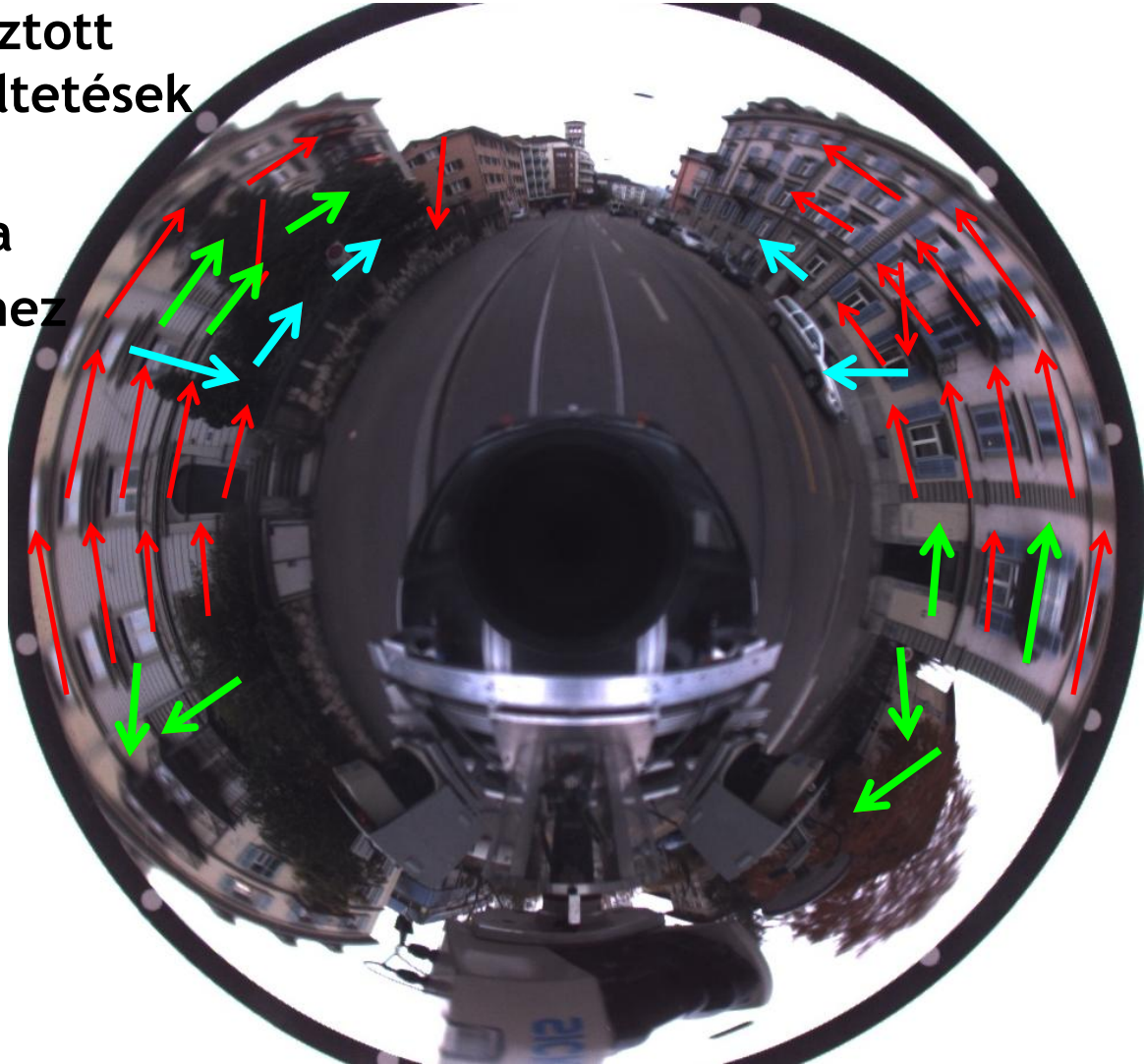
RANSAC

- Outlierek esetében standard módszer mozgásbecslésre



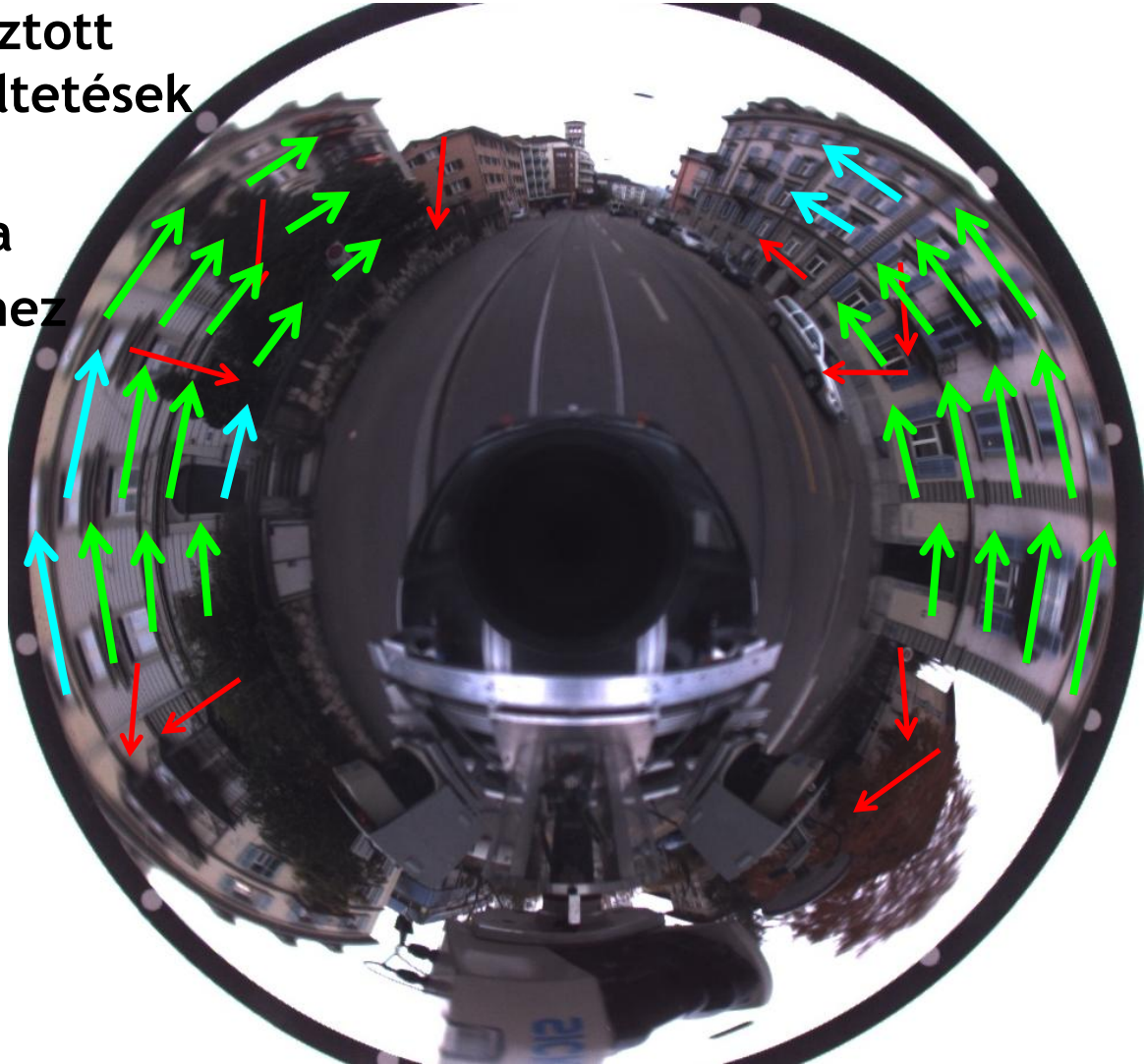
RANSAC

- Outlierek esetében standard módszer mozgásbecslésre
  1. Véletlenül kiválasztott minimális megfeleltetések
  2. Mozgásbecslés és inlierek számlálása
  3. Vissza az 1. lépéshez



RANSAC

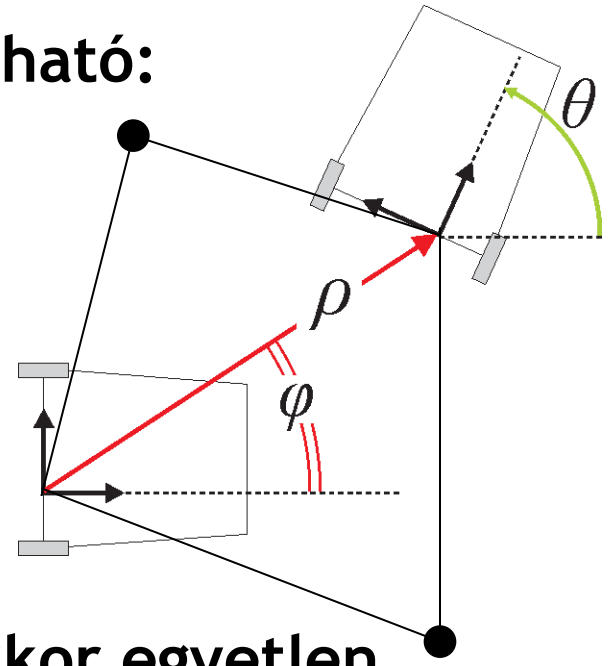
- Outlierek esetében standard módszer mozgásbecslésre
  1. Véletlenül kiválasztott minimális megfeleltetések
  2. Mozgásbecslés és inlierek számlálása
  3. Vissza az 1. lépéshez
- Az iterációk száma exponenciálisan nő az outlierek számával



Minimálisan hány pontpár kell?

- Egy sík mozgása 3 paraméterrel leírható:

  $\theta, \varphi, \rho \Rightarrow 3\ DoF$

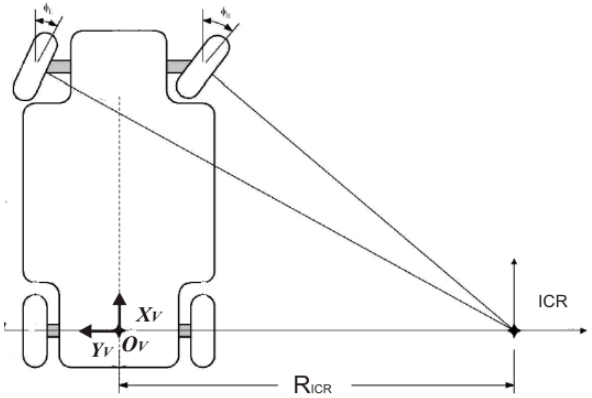


- Ezért két pont elegendő

- Ha további megszorítással élünk, akkor egyetlen pont is elegendő
  - Járművek esetén *nemholonom* kényszermozgást tételezhetünk fel

Nemholonom kényszermozgás

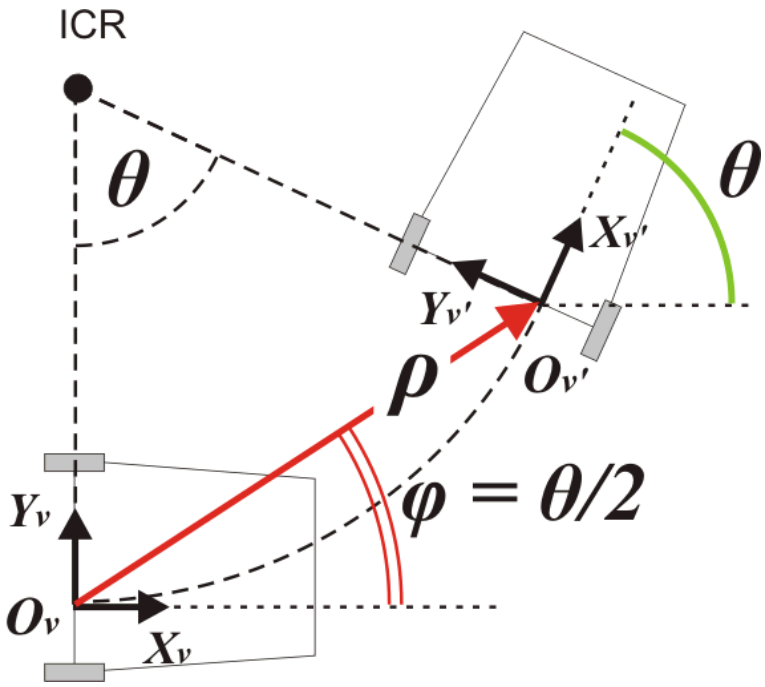
- **A járművek lokálisan körpályán mozognak a pillanatnyi tengely körül**



Example of Ackerman steering principle

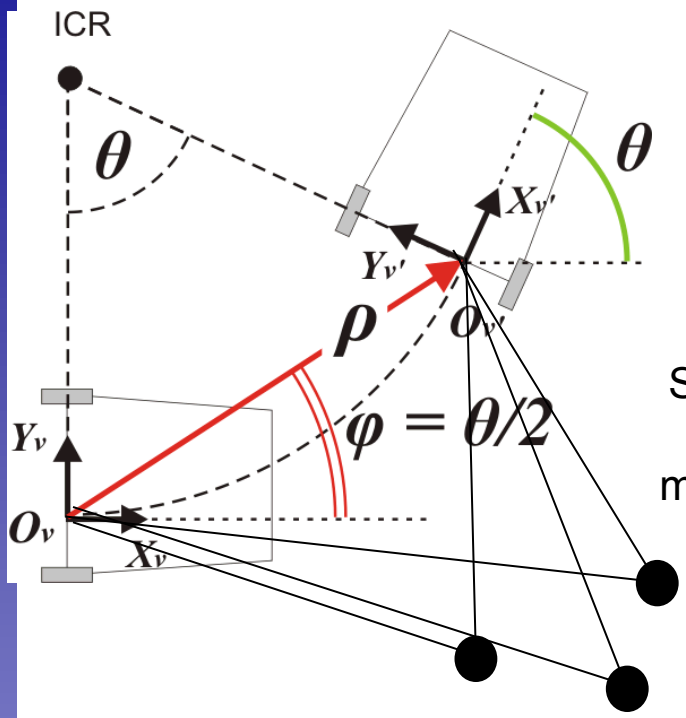
Nemholonom kényszermozgás

- A járművek lokálisan körpályán mozognak a pillanatnyi tengely körül
- φ = θ/2 => csak 2 parameter (θ, ρ) becslése szükséges
- Tehát1 pont is elegendő
- Ez az elérhető legkisebb paraméterezés és így a leghatékonyabb RANSAC algoritmust kapjuk

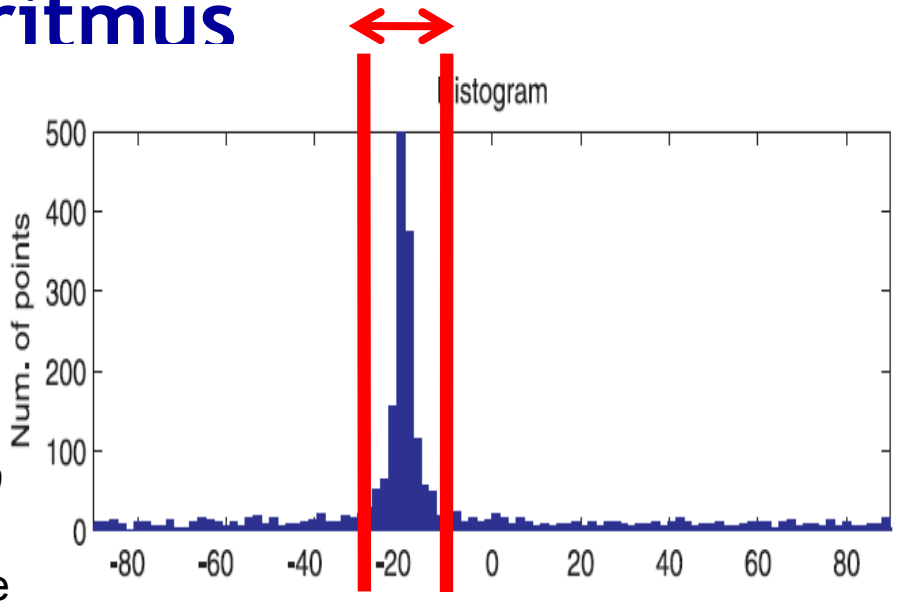


Locally circular motion

1 pontos RANSAC algoritmus

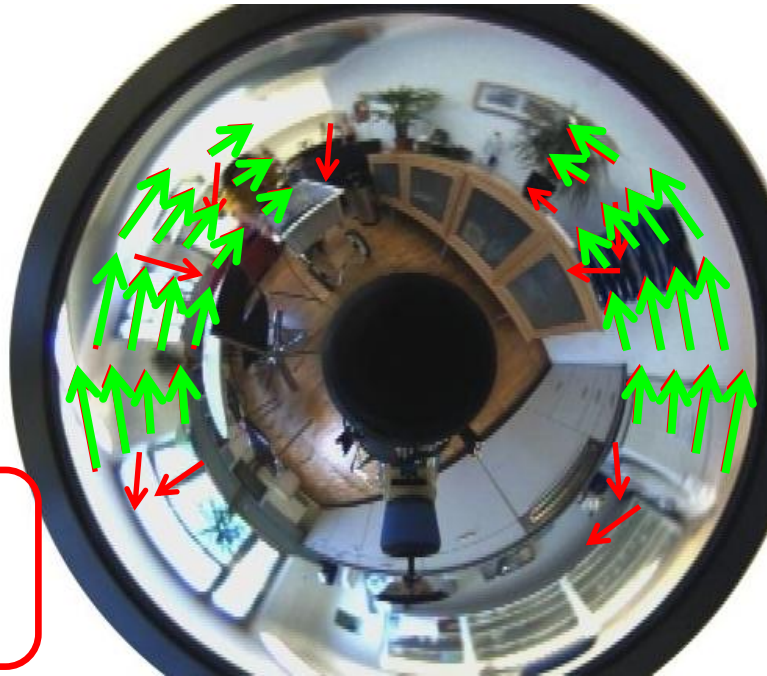


Számoljuk ki θ minden pont megfeleltetésre



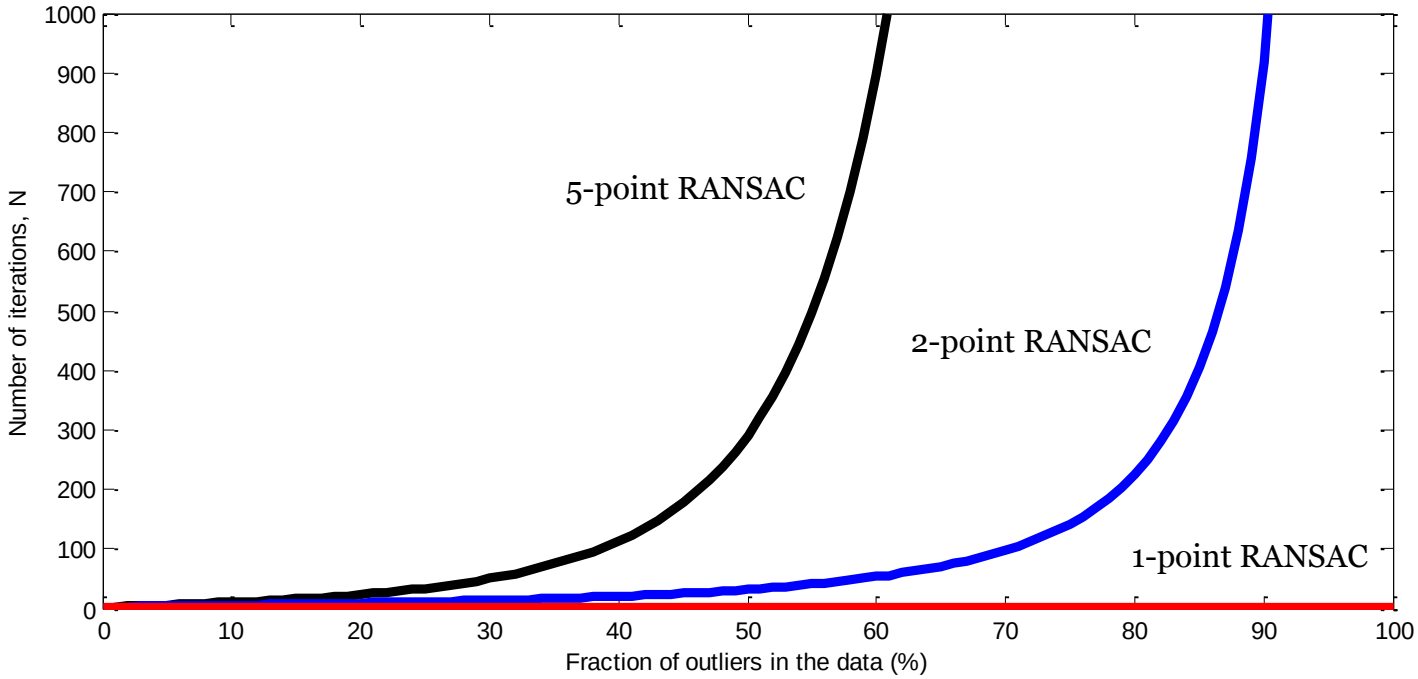
Cak 1 iteráció!

800 Hz rátával képes outlireket szűrni



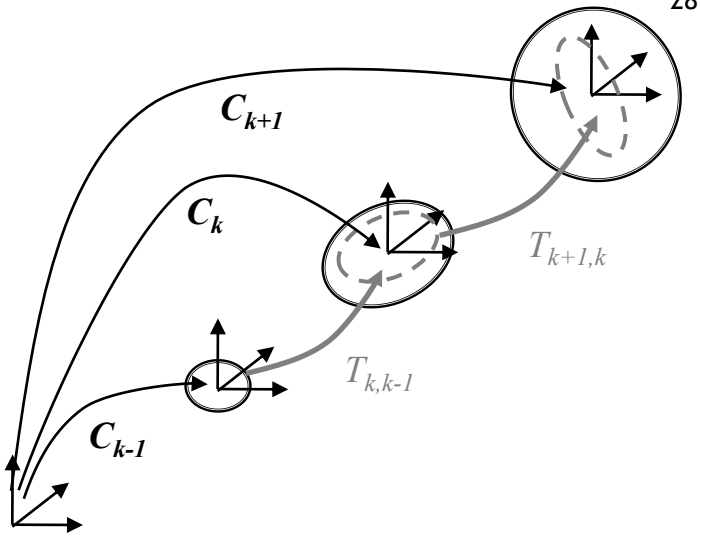
1 pontos RANSAC kizárólag outlierek szűrésére.
A mozgás már 6DOF becsléssel számoljuk

RANSAC hatékonyság



	5-Point RANSAC [Nister'03]	2-Point RANSAC [Ortin'01]	1-Point RANSAC [Scaramuzza, IJCV'11, JFR'11]
Iterációk	**~1000**	**~100**	**1**

Hiba propagálása

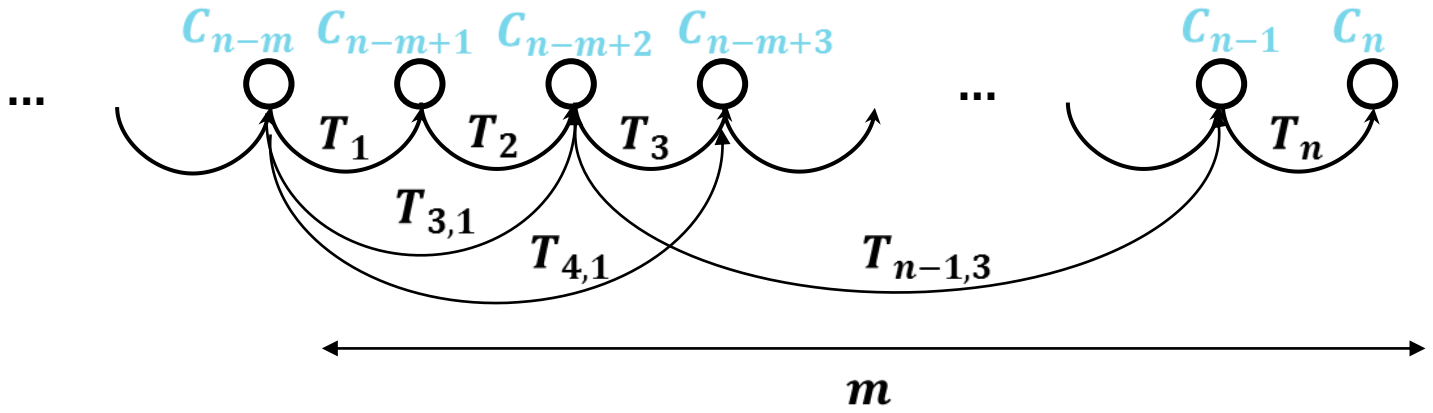


- A $C_k = f(C_{k-1}, T_k)$kamera helyzet bizonytalansága
  - az előző kamera helyzetének Σ_{k-1} kovarianciája és
  - a T_k transzfromáció $\Sigma_{k,k-1}$ kovarianciájának kombinációja
- A kombinált kovariancia mátrixot első rendű Taylor approximációval kapjuk:

$$\Sigma_k = \mathbf{J}\begin{bmatrix}\Sigma_{k-1} & 0\\ 0 & \Sigma_{k,k-1}\end{bmatrix}\mathbf{J}^T = \mathbf{J}_{C_{k-1}}\Sigma_{k-1}\mathbf{J}_{C_{k-1}}^T + \mathbf{J}_{T_k}\Sigma_{k,k-1}\mathbf{J}_{T_k}^T$$

  - Ahol $J_{C_{k-1}}$ és J_{T_k} az f C_{k-1} és T_k szerinti Jacobi mátrixai.
- Tehát a hiba mindig növekszik.

Kamera helyzetének optimalizálása

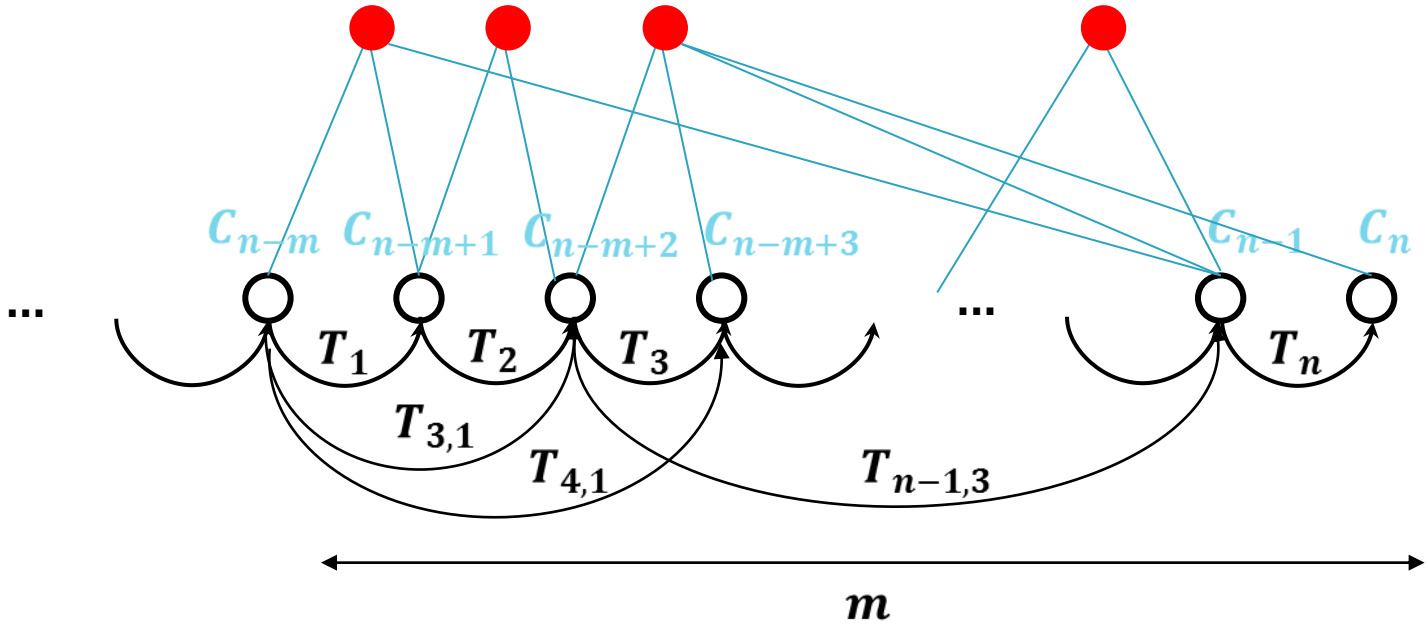


- A transzformációt nem csak szomszédos képkockák között számolhatjuk ki
  - Ezzel újabb kényszerfeltételeket nyerünk
- Ezekkel a távoli tarnszformációkkal egy újabb hibafüggvényt írhatunk fel az utolsó *m* képre:

$$\sum_{e_{ij}} \|C_i - T_{e_{ij}} C_j\|^2$$

  - minimalizálás a Levenberg-Marquardt algoritmussal

Kötegelt behangolás (Bundle Adjustment)

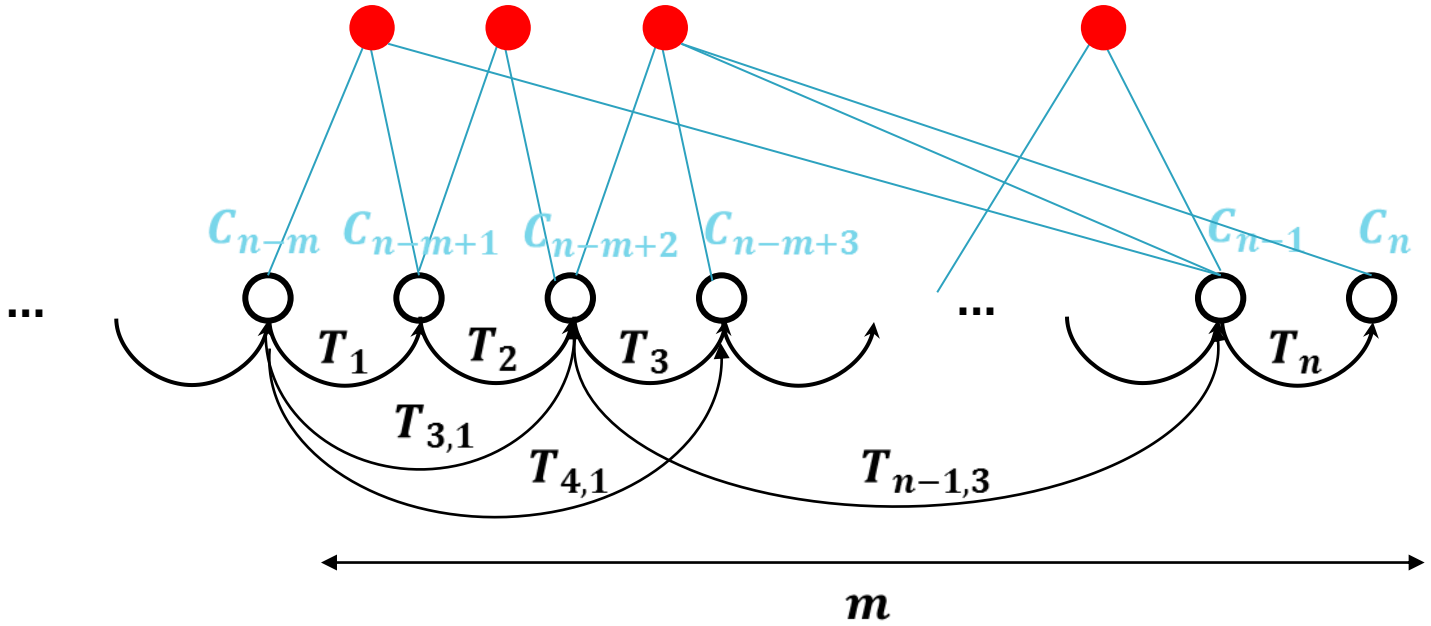


- **Hasonló az előző esethez, de itt egyszerre minimalizáljuk a 3D struktúra és a kamera helyzetének hibáját:**

$$\arg\min_{X^i, C_k} \sum_{i,k} \| p_k^i - g(X^i, C_k) \|^2$$

  - **Itt is a Levenbeg-Marquardt algoritmus használható. A lokális minimumok elkerüléséhez fontos a jó inicializálás.**

Kötegelt behangolás (Bundle Adjustment)



- A két nézet alapján kiszámolt VO-hoz képest javulás
- Pontosabb mint csak a kamera helyzetének meghatározásán alapuló eljárás
- Complexitása: $O((qN+lm)^3)$, ahol N a pontok száma, m a képek száma, q és l pedig a pontok illetve kamerák paramétereinek a száma
  - m értékét a rendelkezésre álló számításidő függvényében kell meghatározni

SLAM

- VO a SLAM építőeleme
  - A kötegelt behangolást kiterjesztjük a teljes útvonalra, nem csak az utolsó *m* képkockára
  - A körutak detektálása kritikus, hiszen ezzel két távoli képkocka között teremtünk kapcsolatot, másrészt a kötegelt behangolást a teljes körútra elvégezve megkapjuk a legpontosabb becslést.
- Körutak detektálásának elve:
  - Egy régen nem látott jellemző újra feltűnik, vagy
  - Visszatérés egy korábban már feltérképezett területre
- Visszatérés detektálása:
  - Lokális vagy globális képleírók alapján a vizuális hasonlóság detektálása a jelenlegi és egy korábbi képkocka között.

First observation

Second observation after a loop

Hasznos szoftverek, adatok

SOFTWARE AND DATASETS

Author	Description	Link
Willow Garage	OpenCV: A computer vision library maintained by Willow Garage. The library includes many of the feature detectors mentioned in this tutorial (e.g., Harris, KLT, SIFT, SURF, FAST, BRIEF, ORB). In addition, the library contains the basic motion-estimation algorithms as well as stereo-matching algorithms.	http://opencv.willowgarage.com
Willow Garage	ROS (Robot Operating System): A huge library and middleware maintained by Willow Garage for developing robot applications. Contains a visual-odometry package and many other computer-vision-related packages.	http://www.ros.org
Willow Garage	PCL (Point Cloud Library): A 3D-data-processing library maintained from Willow Garage, which includes useful algorithms to compute transformations between 3D-point clouds.	http://pointclouds.org
Henrik Stewenius et al.	5-point algorithm: An implementation of the 5-point algorithm for computing the essential matrix.	http://www.vis.uky.edu/~stewe/FIVEPOINT/
Changchang Wu et al.	SiftGPU: Real-time implementation of SIFT.	http://cs.unc.edu/~ccwu/siftgpu
Nico Cornelis et al.	GPUSurf: Real-time implementation of SURF.	http://homes.esat.kuleuven.be/~ncorneli/gpusurf
Christopfer Zach	GPU-KLT: Real-time implementation of the KLT tracker.	http://www.inf.ethz.ch/personal/chzach/opensource.html
Edward Rosten	Original implementation of the FAST detector.	http://www.edwardrosten.com/work/fast.html

Hasznos szoftverek, adatok

Michael Calonder	Original implementation of the BRIEF descriptor.	http://cvlab.epfl.ch/software/brief/
Leutenegger et al.	BRISK feature detector.	http://www.asl.ethz.ch/people/lestefan/personal/BRISK
Jean-Yves Bouguet	Camera Calibration Toolbox for Matlab.	http://www.vision.caltech.edu/bouguetj/calib_doc
Davide Scaramuzza	OCamCalib: Omnidirectional Camera Calibration Toolbox for MATLAB.	https://sites.google.com/site/scarabotix/ocamcalib-toolbox
Christopher Mei	Omnidirectional Camera Calibration Toolbox for MATLAB	http://homepages.laas.fr/~cmei/index.php/Toolbox
Mark Cummins	FAB-MAP: Visual-word-based loop detection.	http://www.robots.ox.ac.uk/~mjc/Software.htm
Friedrich Fraundorfer	Vocsearch: Visual-word-based place recognition and image search.	http://www.inf.ethz.ch/personal/fraundof/page2.html
Manolis Lourakis	SBA: Sparse Bundle Adjustment	http://www.ics.forth.gr/~lourakis/sba
Christopher Zach	SSBA: Simple Sparse Bundle Adjustment	http://www.inf.ethz.ch/personal/chzach/opensource.html
Rainer Kuemmerle et al.	G2O: Library for graph-based nonlinear function optimization. Contains several variants of SLAM and bundle adjustment.	http://openslam.org/g2o
RAWSEEDS EU Project	RAWSEEDS: Collection of datasets with different sensors (lidars, cameras, IMUs, etc.) with ground truth.	http://www.rawseeds.org
SFLY EU Project	SFLY-MAV dataset: Camera-IMU dataset captured from an aerial vehicle with Vicon data for ground truth.	http://www.sfly.org
Davide Scaramuzza	ETH OMNI-VO: An omnidirectional-image dataset captured from the roof of a car for several kilometers in a urban environment. MATLAB code for visual odometry is provided.	http://sites.google.com/site/scarabotix

Felhasznált anyagok

- **Davide Scaramuzza: A Tutorial on Visual Odometry**
  - **https://sites.google.com/site/scarabotix/**
- **További források az egyes diákon feltüntetve.**