*Luka Simić i Petar Marković*

# Poređenje performansi i uspešnosti Structure from Motion algoritama

*Cilj ovog rada je poređenje performansi i uspešnosti kombinacija algoritama koji se mogu koristiti za rekonstruisanje 3D prostora iz sekvenci 2D slika (Structure from Motion). Algoritmi za poređenje su podeljeni na tri kategorije u zavisnosti od funkcije: prepoznavanje karakterističnih tačaka, povezivanje/praćenje karakterističnih tačaka i rekonstrukcija prostora. Upoređene su implementacije algoritama u biblioteci OpenCV u programskom jeziku C++, koristeći KITTI bazu slika. Za prepoznavanje karakterističnih tačaka korišćeni su algoritmi ORB, SURF i SIFT. Za praćenje pomeraja karakterističnih tačaka korišćen je algoritam zasnovan na FLANN biblioteci kao i brute-force algoritam. Za procenu transformacije pomeraja neophodne su dobro povezane karakteristične tačke, stoga su iste filtrirane korišćenjem RANSAC algoritma. Dobijeni rezultati pokazuju da kombinacija SURF algoritma za prepoznavanje karakterističnih tačaka i brute-force algoritma za njihovo povezivanje daje najtačniji oblak tačaka (point cloud) na osnovu sličnosti određene ICP algoritmom, ali je ta kombinacija takođe i najsporija.*

## Uvod

Mogućnost da se robot lokalizuje i mapira prostor u kom se nalazi je jako bitna za razne primene, poput automatskog pravljenja topoloških mapa ili nacrta unutrašnjosti nekog prostora. Pokazalo se da su metode koje koriste računarsku viziju efektivnije od ostalih zato što su precizne, a aparatura za njih je relativno jeftina.

Structure from Motion (SfM ) je fotogrametrijska tehnika koja se koristi za procenjivanje trodimenzionalnog prostora na osnovu sekvence dvodimenzionalnih slika tog prostora napravljenih pri kretanju.

Cilj ovog rada jeste da uporedi preciznost i performanse SfM varijacija. Preciznost se određuje kako bi se znalo koliko će tačna biti lokalizacija nekog robota u prostoru, a performanse se određuju kako bi se znalo koliko bi se brzo taj robot kretao po prostoru i u odnosu na specifikacije njegovog procesora odrediti okvirno vreme njegovog kretanja za određenu operaciju.

## Metod

Structure from Motion tehnika obično podrazumeva kombinacije više algoritama, konkretno:

- Algoritme za pronalaženje karakterističnih tačaka na slikama (feature detection algoritmi; Li 2017)
- Algoritme za povezivanje karakterističnih tačaka sa više slika (feature matching i feature tracking algoritmi)
- Rekonstrukciju 3D prostora iz povezanih karakterističnih tačaka (triangulacija)

Obrada podataka iz baze tekla je na sledeći način (šema na slici 2):

1. Stereo parovi slika se učitavaju iz baze, frejm po frejm.

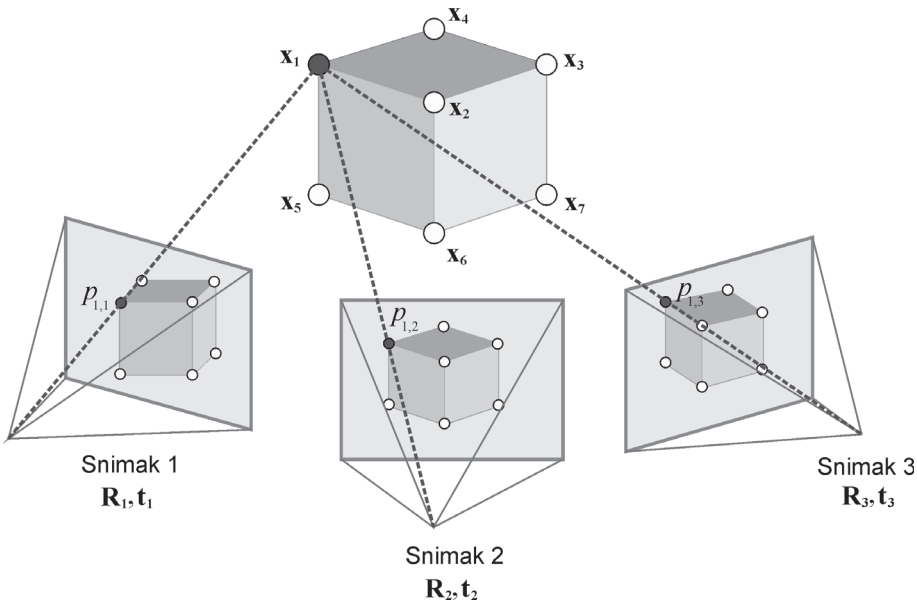
*Luka Simić (2000), Novi Beograd, Nehruova 135, učenik 4. razreda Računarska gimnazije u Beogradu*

*Petar Marković (2002), Novi Sad, Vase Jovanovića 1, učenik 2. razreda Gimnazije „Jovan Jovanović Zmaj" u Novom Sadu*

*MENTORI:*

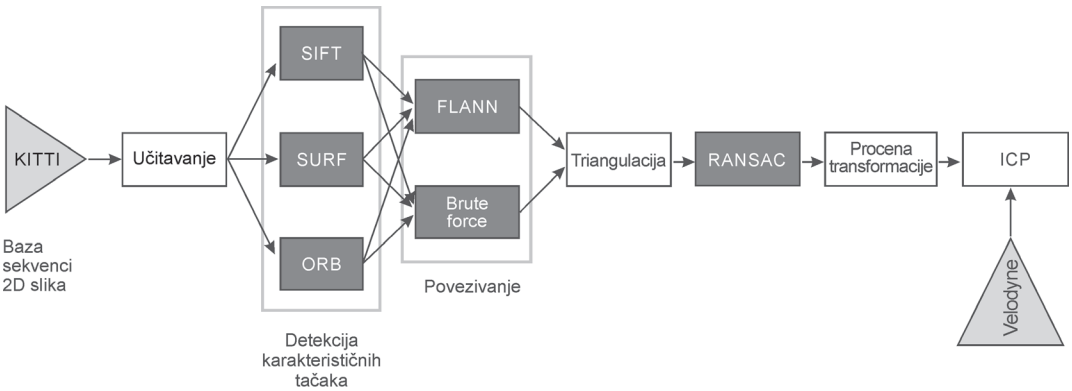
*Damjan Dakić, Microsoft Beograd, Data Scientist*

*Nikola Milenić, student Elektrotehničkog fakulteta Univerziteta u Beogradu*



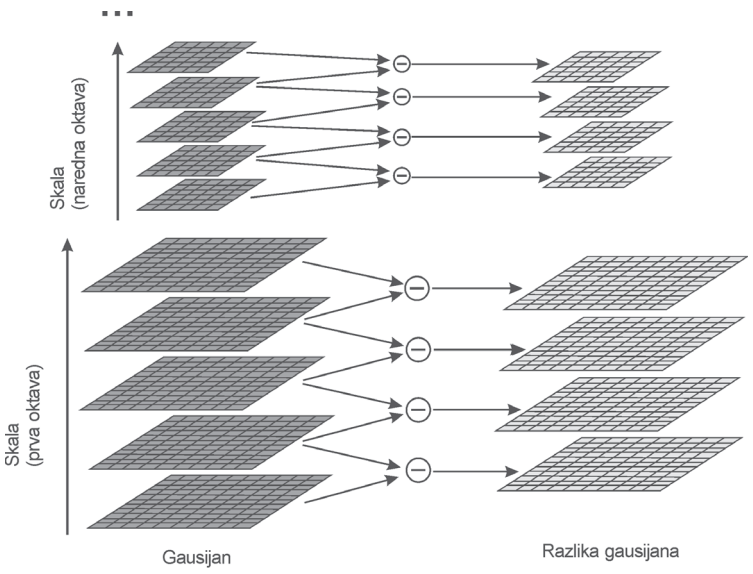

Slika 1. Ilustracija rada Structure from Motion tehnike. Na kamere 1, 2 i 3 se projektuje isti objekat, s tim što se vidi iz različitih pozicija. Na osnovu poznatih projekcija karakterističnih tačaka na platna kamera ( $p_{ij}$, $i$ = 1, 2, …, 7, $j$ = 1, 2 i 3) i pozicija kamera u prostoru datih preko matrica rotacije i transformacije ($\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}_3$; $\mathbf{t}_1$, $\mathbf{t}_2$, $\mathbf{t}_3$) u preseku pravih koje povezuju karakteristične tačke sa pozicijama kamere dobijaju se pozicije karakterističnih tačaka u prostoru ($\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_7$). Bitno je pomenuti da treća kamera nije neophodna osim u slučaju kada rekonstruišemo i dubinu sa slika. (Adaptirano prema Yilmaz i Karakus 2013)

Figure 1. Illustration of the Structure from Motion technique. An object is projected to cameras 1, 2 and 3. Based on keypoint projections to camera screens ( $p_{ij}$, $i$ = 1, 2, …, 7, $j$ = 1, 2 and 3) and camera positions in space given through rotation and transformation matrices ($\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}_3$; $\mathbf{t}_1$. $\mathbf{t}_2$, $\mathbf{t}_3$) , 3D keypoints' positions ($\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_7$) can be found in the intersection of lines connecting camera positions with keypoint projections. It is important to note that the third camera is only needed in case we are trying to estimate depth from image as well. (Source: Yilmaz & Karakus 2013)




Slika 2. Dijagram kretanja podataka i redosleda izvršavanja algoritama

Figure 2. Diagram of the data flow and execution order of algorithms




Slika 3. Scale Space, proceduralno zamućivanje frejma gausovim blurom na jednoj skali, te smanjivanje rezolucije po oktavama. Oduzimanjem vrednosti piksela susednih slika jedne oktave omogućava se nalaženje robusnih karakterističnih tačaka. (Izvor: Sinha 2010)

Figure 3. Scale Space, procedurally applying Gaussian blur to the frame within one scale and downsampling it across different octaves. By deducing neighbour images of one octave it becomes possible to find robust keypoints. (Source: Sinha 2010)

2. Nad učitanim slikama iz KITTI baze izdvajaju se karakteristične tačke koristeći SIFT, SURF ili ORB algoritam.

3. Karakteristične tačke se, prema deskriptorima, povezuju između frejmova (slika 6), kao i unutar jednog frejma (između leve i desne slike stereo para) algoritmima za povezivanje karakterističnih tačaka (FLANN i brute-force), nakon čega se odbacuju karakteristične tačke koje nemaju par.

4. Parovi povezanih karakterističnih tačaka bivaju triangulisani. Rezultati triangulacije u jednom frejmu su pozicije tih tačaka u prostoru. Rezultati triangulacije između frejmova su zapravo matrice rotacije i transformacije koje opisuju pomeraj kamere između ta dva frejma.

5. Akumulacijom matrica transformacije dobijamo opis putanje kojom se kamera kretala. Primenjujući ovaj opis na dobijene pozicije triangulisanih tačaka u prostoru, rekonstruišemo oblak tačaka.

6. Dobijeni oblak tačaka se poredi sa referentnim oblacima korišćenjem ICP metode (Iterative Closest Point), kako bi se referentni i rezultujući oblaci što više približili pre računanja greške kao sume kvadrata razdaljina između njihovih tačaka (Stojanović i Arbanas 2012).

## Prepoznavanje karakterističnih tačaka

Za detekciju karakterističnih tačaka na slikama korišćena (i poređena) su tri algoritma:

- SIFT (Introduction to SIFT 2013)
- SURF (Bay *et al*. 2006)
- ORB (Oriented FAST and rotated BRIEF)

**SIFT** (Scale Invariant Feature Transform – transformacija u karakteristične tačke nezavisne od skaliranja) je algoritam koji koristi Scale-Space (Sinha 2010, ilustrovano na slici 3) da bi dobio karakteristične tačke koje ne zavise od rotacije i pomeraja kamere. Ideja je da različiti nivoi zamagljenja u skalama i oktavama daju garanciju da će izabrane karakteristične tačke biti kvalitetne, tj. da će biti prepoznatljive i u ostalim slikama. Lokalni maksimumi ili minimumi razlike slika sa različitim nivoima Gausovog zamućenja (Gaussian blur) se uzimaju za karakteristične tačke.

**SURF** (Speeded Up Robust Features) je algoritam sličan SIFT-u. SURF umesto razlike gausijana, za aproksimaciju Laplasove transformacije Gausovog zamućenja koristi Box Filter (Bay *et al*. 2006, ilustrovano na slici 4). Povezivanje karakterističnih tačaka nađenih od strane


Slika 4. Box filtar: 9 × 9 box filtri su aproksimacija Gausovog drugog izvoda (izvor: Grauman i Leibe 2011)

Figure 4. Box filter: the 9 × 9 box filters are approximations for Gaussian second order derivatives (Source: Grauman & Leibe 2011)

Slika 5. Prikaz parova označenih atributa (karakterisitičnih tačaka) na susednim slikama

Figure 5. Some of the matched points shown

Slika 6. Optički protok, crne linije pokazuju pomeraj karakterističnih tačaka između dva susedna frejma

Figure 6. Optical flow, black lines show the shift of each keypoint between two neighbor frames

SURF-a je zbog toga mnogo brže nego kod SIFT-a (*ibid.*).

**ORB** (Oriented FAST and rotated BRIEF, Rublee *et al*. 2011) je algoritam koji koristi FAST algoritam za prepoznavanje karakterističnih tačaka i izmenjen BRIEF algoritam za generisanje opisa prepoznatih karakterističnih tačaka.

## Povezivanje karakterističnih tačaka

Nakon detekcije karakterističnih tačaka na slikama one bivaju povezane između slika sa iste kamere pomoću algoritama za povezivanje karakterističnih tačaka (slika 5). Dva algoritma koja su poređena su:

– algoritam zasnovan na FLANN (Fast Library for Approximate Nearest Neighbors) biblioteci, i
– algoritam koji poredi razdaljinu između svake dve karakteristične tačke na slikama i povezuje one sa najmanjom razdaljinom (brute-force matcher, u daljem tekstu BF).

## Praćenje karakterističnih tačaka

Korišćenjem istog algoritma za povezivanje ključnih tačaka, karakteristične tačke su povezane između slika slikanih iz različitih pozicija, ali istog ugla. Na taj način su povezane zajedničke karakteristične tačke za sve tri slike koje se nakon ovog koraka koriste za sve ostale algoritme umesto tačaka između samo dve slike. Takođe, na osnovu prethodno povezanih slika generisan je optički protok (optical flow) između slika jedne sekvence radi vizualizacije povezanih karakterističnih tačaka.

## Triangulacija

Triangulacija je proces dobijanja tačaka u 3D prostoru iz koordinata tih tačaka u 2D prostoru i intrinzika kamera koje su ih slikale. U ovom koraku, triangulacija je izvršena između karakterističnih tačaka sa dve slike jednog frejma (leva i desna slika stereo kamere) i dobijene su 3D koordinate u sistemu trenutne pozicije kamera (Lindstrom 2010).

## Procena transformacije

Kako bi se videlo kako se kamera pomerala tokom vremena, i na taj način transformisale 3D koodinate iz koordinatnih sistema trenutnih pozicija kamera u koordinatni sistem prve pozicije kamera, potrebno je pronaći relevantnu matricu transformacije i potom pomnožiti sve koordinate s njom. Ovaj problem je poznatiji kao PnP problem (Projection ‘n’ Perspective; Wu i Hu 2006).

Za rešavanje ovog problema korišćena je OpenCV funkcija solvePnP koja za parametre uzima tačke u prostoru koje su dobijene triangulacijom, odgovarajuće tačke na sledećem frejmu i parametre kamere, a to su matrica kamere (VM 2006) i koeficijenti distorzije koji su u našem slučaju bili 0. Povratna vrednost funkcije su dva vektora koji predstavljaju promenu ugla (rotaciju) i pomeraj pozicije (translaciju). Radi reprojektovanja tačaka nazad u ravan kamere dati vektor rotacije se prebacuje u matricu rotacije. Matrica rotacije i vektor translacije se dalje spajaju u jednu matricu transformacije, čijim se množenjem sa koordinatama tačaka slike dobijaju reprojektovane tačke.

**RANSAC**. Pošto se često dešava da se pogrešno povežu karakteristične tačke između slika, mogu se videti velike transformacije tačaka i to uzrokuje da transformaciona matrica na nekim mestima značajno odstupi od tačne, pa stoga i krajnja putanja biva isečena na više mesta. Ovakve greške značajno utiču na krajnji rezultat, i protiv njih je primenjivan RANSAC (Random Sample Consensus) algoritam. Pošto OpenCV implementacija RANSAC algoritma nije davala vidljiva poboljšanja u rezultatima, za potrebe ovog rada implementirana je i nova metoda za RANSAC filtriranje odstupanja, koja je kasnije upoređena sa OpenCV verzijom.

## Aparatura

**Hardver**. Merenja su vršena na laptopu sledećih specifikacija:

CPU: AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx (8) @ 2.300GHz
GPU: AMD ATI 05:00.0 Picasso
RAM: 5950MiB
OS: Arch Linux x86_64
Kernel: 5.8.14-arch1-1

**Softver**. Implementacija je realizovana u programskom jeziku C++ uz pomoć OpenCV biblioteke. Implementacija softvera korišćenog u ovom radu može se naći na repozitorijumu (Simić i Marković 2019) .

Korišćeni su podaci iz baze za vizuelnu odometriju KITTI, koji sadrži 11 sekvenci uzastopnih slika (iz grada Karlsrue u Nemačkoj), slikanih iz dva ugla sa referentnim pozicijama automobila zabeleženim GPS-om i oblacima tačaka okolnog prostora zabeleženih pomoću Velodyne LiDAR sistema, kao i kalibracijama stereo kamera koje su korišćene za snimanje.

Za vizualizaciju oblaka tačaka korišćen je softver za rekonstrukciju trodimenzionalnih prostora preko 2D slika, COLMAP.

# Rezultati

Metrike za poređenje uspešnosti algoritama su preuzete sa KITTI evaluation benchmark-a (Geiger *et al*. 2012). Pored KITTI EB je korišćen i Evo set Python skripti za evaluaciju VO i SLAM algoritama (Grupp 2019). Pored toga, vršeno je i poređenje klastera pomoću PCL biblioteke (Point Cloud Library) i ICP algoritma (Iterative Closest Point), a kao greška je uzeta vrednost zbira svih euklidskih rastojanja između tačaka generisanih point cloud-ova po frejmu i njima najbližih tačaka iz Velodyne point cloud--ova (takođe poznat kao fitness score).

## Merenje performansi algoritama

Performanse algoritama su generalno merene na sekvenci 4 odometrijskih podataka iz KITTI baze jer je najkraća (271 frejm) i zato najbrža za izvlačenje rezultata merenja performansi poređenih algoritama (tabela 1). Merena su vremena izvršavanja nad svakim pojedinačnim frejmom, ukupno vreme izvršavanja, prosečno vreme izvršavanja jednog frejma, minimalno i maksimalno vreme izvršavanja jednog frejma i za svaki je zabeležen prosečan broj pronađenih karakterističnih tačaka. RANSAC kolona se odnosi na to da li je korišćena naša RANSAC implementacija ili `cv::solvePnPRansac` metoda, a BF označava da je korišćen brute-force algoritam. Prilikom merenja performansi (kao i svih narednih merenja) povećan je prag razdaljine između uparenih karakterističnih tačaka (kako bi one bile smatrane parom) sa 0.7 na 0.8. ORB takođe ne radi sa algoritmom za povezivanje karakterističnih tačaka zasnovani na FLANN.

## Uspešnost rekonstrukcije

**Relativna greška pozicije**. Greška pozicije se sastoji iz dve komponente:

- R – greška rotacije, izražena u radijanima po frejmu
- T – greška translacije, izražena u metrima po frejmu

Prilikom merenja relativne greške pozicije korišćena je ista sekvenca kao kod merenja performansi. Rezultati merenja (tabela 2) se sastoje od minimalne, maksimalne i prosečne relativne greške rotacije i translacije. Zbog komplikacija pri implementaciji, nismo uspeli da dobijemo rezultate merenja sa ORB detektorom karakterističnih tačaka, brute-force algoritmom za njihovo povezivanje i našom implementacijom RANSAC-a.

Tabela 1. Rezultati merenja performansi kombinacija algoritama (prve tri kolone) nad sekvencom 4 odometrijskih podataka iz KITTI baze

| | | | Vreme izvršavanja (s) | | | | |
|---|---|---|---|---|---|---|---|
| Detekcija | Povezivanje | RANSAC | Ukupno | Srednje | Min. | Maks. | Sred. br. KT |
| SIFT | FLANN | Da | 414.6 | 1.54(1) | 1.18 | 1.79 | 120(1) |
| SIFT | FLANN | Ne | 399.7 | 1.481(8) | 1.169 | 1.719 | 120(1) |
| SIFT | BF | Da | 415.5 | 1.539(8) | 1.150 | 1.804 | 119(1) |
| SIFT | BF | Ne | 402.3 | 1.490(8) | 1.136 | 1.748 | 119(1) |
| SURF | FLANN | Da | 643.0 | 2.38(3) | 1.57 | 3.5 | 362(4) |
| SURF | FLANN | Ne | 627.7 | 2.32(3) | 1.28 | 3.32 | 362(4) |
| SURF | BF | Da | 693.2 | 2.57(4) | 1.66 | 3.71 | 354(4) |
| SURF | BF | Ne | 686.2 | 2.54(4) | 1.65 | 3.72 | 354(4) |
| ORB | BF | Da | 52.5 | 0.194(2) | 0.0 | 0.272 | 46.1(8) |
| ORB | BF | Ne | 51.9 | 0.1923(16) | 0.1432 | 0.2626 | 46.4(0.8) |

Sred. br. KT – srednji broj karakterističnih tačaka; BF – brute force algoritam. Cifre u zagradi označavaju grešku: 1.54(1) znači isto što i $1.54 \pm 0.01$, a 0.1923(16) isto što i $0.1923 \pm 0.0016$).

Tabela 2. Rezultati merenja relativne greške pozicije kombinacija algoritama nad sekvencom 4 odometrijskih podataka iz KITTI baze.

| Detekcija | Povezivanje | RANSAC | Rotacija ($\times 10^{-3}$) | | | Translacija | | |
|---|---|---|---|---|---|---|---|---|
| | | | Srednja | Min. | Maks. | Srednja | Min. | Maks. |
| SIFT | FLANN | Da | 1.5(1) | 0.0 | 12.1 | 0.085(13) | 0.006 | 1.723 |
| SIFT | FLANN | Ne | 0.95(5) | 0.0 | 5.44 | 0.0384(16) | 0.0032 | 0.1612 |
| SIFT | BF | Da | 1.50(9) | 0.0 | 9.39 | 0.082(13) | 0.006 | 1.712 |
| SIFT | BF | Ne | 0.96(5) | 0.0 | 4.64 | 0.0378(17) | 0.0033 | 0.2488 |
| SURF | FLANN | Da | 0.90(4) | 0.0 | 4.02 | 0.035(1) | 0.007 | 0.103 |
| SURF | FLANN | Ne | 0.70(4) | 0.0 | 2.72 | 0.036(1) | 0.006 | 0.101 |
| SURF | BF | Da | 0.88(4) | 0.0 | 4.14 | 0.035(1) | 0.008 | 0.106 |
| SURF | BF | Ne | 0.68(4) | 0.0 | 2.57 | 0.036(1) | 0.005 | 0.104 |
| ORB | BF | Da | – | – | – | – | – | – |
| ORB | BF | Ne | 14(9) | 0.0 | 222.1 | 16(16) | 0 | 4308 |

BF – brute-force algoritam. Cifre u zagradi označavaju grešku: 14(9) znači isto što i $14 \pm 9$, a 0.0378(17) isto što i $0.0378 \pm 0.0017$

Tabela 3. Fitness score na sekvenci 0

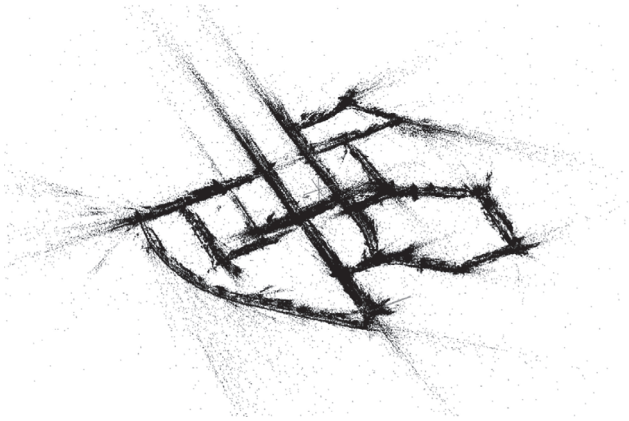
| Detekcija | Povezivanje | RANSAC | Ukupno | Prosek |
|---|---|---|---|---|
| SIFT | FLANN | Da | 459384.0 | 101.2(9) |
| SIFT | FLANN | Ne | 452289.9 | 99.6(8) |
| SIFT | BRUTEFORCE | Da | 452673.3 | 99.7(8) |
| SURF | FLANN | Da | 437168.2 | 96.3(9) |
| SURF | BRUTEFORCE | Da | 434250.6 | 95.6(9) |
| ORB | BRUTEFORCE | Da | – | – |
| ORB | BRUTEFORCE | Ne | – | – |

Cifre u zagradi označavaju grešku: 101.2(9) znači isto što i $101.2 \pm 0.9$

**Fitness score**. Fitness score predstavlja srednju vrednost kvadrata razdaljina između najbližih tačaka dva point cloud-a, i njome je merena tačnost rekonstruisanog point cloud-a. Testiranje je izvršeno samo na sekvenci 0, jer su datoteke za referentne point cloud-ove prilično velike, i jedino smo za tu sekvencu mogli da sačuvamo ceo referentni point cloud. Razdaljine između rekonstruisanog i referentnog point cloud-a su merene za svaki frejm ponaosob, jer je ceo spojeni point cloud bio previše veliki i suviše gust da bi se kao takav u celosti mogao porediti. Kao krajnji rezultat računali smo ukupan i prosečan fitness score za sve frejmove (tabela 3).

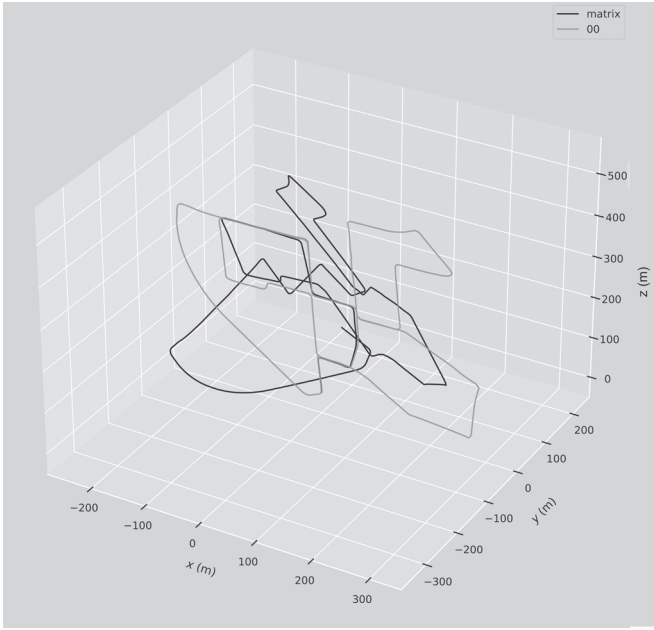
## Kvalitativni rezultati

Zbog akumuliranja greške rotacije koja se naročito povećava pri velikim pomerajima kadra (na primer, kada KITTI automobil skreće) spojeni point cloud je dosta odstupao od pravog izgleda terena iz sekvence nad kojom je pokretan. Međutim, prilikom primene KITTI transformacija položaja na naše triangulisane tačke, dobije se point cloud (slika 8) koji liči na teren po kojem se KITTI automobil kretao. Takođe, prilikom poređenja KITTI transformacija sa našim transformacijama (slika 9) može se videti da su najveća odstupanja u rotaciji, ali da oblik naše putanje liči na oblik putanje KITTI automobila.



Slika 7. Point cloud generisan koristeći KITTI transformacije položaja

Figure 7. Point cloud generated using KITTI poses




Slika 8. Procena pozicije: crnom linijom označene su dobijene, a sivom tačne pozicije (generisano koristeći SURF i algoritam za povezivanje tačaka zasnovan na FLANN, bez korišćenja RANSAC implementacije).

Figure 8. Postion estimation: black color marks generated positions, grey marks ground truth positions (generated using SURF and FLANN-based matcher, without using RANSAC).

# Zaključak

Kada se za detekciju karakterističnih tačaka koristi SIFT, prosečna dužina izvršavanja za jedan frejm opada za skoro 60% u odnosu na SURF. Ovo se dešava zbog toga što SURF detektuje skoro trostruko više karakterističnih tačaka što usporava povezivanje karakterističnih tačaka i RANSAC.

Takođe, brzina izvršavanja za ORB detekciju karakterističnih tačaka je osam puta veća nego sa SIFT-om, ali pokazalo se da karakteristične tačke dobijene ORB-om nisu dovoljno sigurne da bi se sa njima radila rekonstrukcija. ORB je povoljno koristiti kada je u postavi rig i gde se pozicije kamere uvek znaju, tako da neće biti problema oko generisanja ekstrinzika za susedne frejmove. U slučaju kada je potrebno izračunati pomeraj kamere povoljnije je koristiti SIFT ili SURF zajedno sa brute-force algoritmom jer su pokazali veću pouzdanost nego ORB. Pokazalo se da ORB daje najnetačniju estimaciju promene pozicije kamere, ali da su ORB-ove karakteristične tačke dovoljno dobre za generisanje point cloud-a.

## Literatura


Bay H., Tuytelaars T., Van Gool L. 2006. SURF: Speeded Up Robust Features. U *Computer Vision – ECCV 2006, lecture notes in computer science* (ur. A. Leonardis *et al*.). Springer, str. 404–17.

Geiger A., Lenz P., Urtasun R. 2012. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. U 2*012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, str. 3354–61.

Grauman K., Leibe B. 2011. *Visual Object Recognition*. Morgan & Claypool Publishers

Grupp M. 2019. Evo: Python Package for the Evaluation of Odometry and SLAM. https://github.com/MichaelGrupp/evo (12. avgust 2019).

Li S. 2017. A Review of Feature Detection and Match Algorithms for Localization and Mapping. *IOP Conference Series: Materials Science and Engineering*, **231**: 012003.

Lindstrom P. 2010. Triangulation Made Easy. U *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, str. 1554–61.

Introduction to SIFT (Scale-Invariant Feature Transform) 2013. OpenCV-Python Tutorials. Documentation.

Rublee E., Rabaud V., Konolige K., Brodski G. 2011. Tutorial on ORB. https://gilscvblog.com/2013/10/04/a-tutorial-on-binary-descriptors-part-3-the-orb-descriptor/

Simić L., Marković P. 2019. SfM. https://github.com/KockaAdmiralac/SfM

Sinha U. 2010. SIFT: theory and practice – introduction. http://aishack.in/tutorials/sift-scale-invariant-feature-transform-introduction/

Stojanović M., Arbanas M. 2012. Mapiranje prostora pomoću Turtlbota. *Petničke sveske*, 70: 137.

VM 2006. Camera models and parameters. http://ftp.cs.toronto.edu/pub/psala/VM/camera-parameters.pdf

Wu Y., Hu Z. 2006. PnP Problem Revisited. *Journal of Mathematical Imaging and Vision*, **24**: 131.

Yilmaz O., Karakus F. 2013. Stereo and kinect fusion for continuous 3D reconstruction and visual odometry. U *2013 International Conference on Electronics, Computer and Computation (ICECCO)*. IEEE, str. 115-118.


---


*Luka Simić and Petar Marković*


## Comparison of Performance and Accuracy of Structure from Motion Algorithms


The objective of this paper is to compare the performance and success rate of algorithm combinations that can be used for reconstructing a 3D space from 2D images (also known as the Structure from Motion technique). The algorithms being compared are separated into three categories by their functionality: feature detection, feature matching/tracking and space reconstruction. Implementations of these algorithms in the C++ programming language using the OpenCV library were compared on the KITTI dataset. For feature detection ORB, SURF and SIFT algorithms were used, and for feature tracking FLANN-based and brute-force matchers were used. Correctly matched features are a key part of the transformation estimation for the change in position, so all features were filtered using RANSAC. Results show that the combination of SURF and a brute-force matcher yields the most accurate point cloud in terms of ICP fitness score, but that combination is also the slowest.