MINISTERUL EDUCAȚIEI

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

LIPIREA IMAGINILOR FOLOSIND DIFERITE TIPURI DE PUNCTE CHEIE

Lucrare De Licență

Absolvent: Adrian BÎZ

Coordonator științific: Șl. Dr. Ing. Robert VARGA

Introducere

În domeniul procesării de imagini și viziunii computerizate, detectarea de elemente cheie și extragerea acestora reprezintă una dintre operațiile de bază. Acestea jucând un rol critic pentru procese precum recunoașterea de obiecte, analizarea scenelor sau lipirea de imagini.

Lipirea imaginilor

Lipirea imaginilor constă în combinarea a două sau mai multe imagini ce au elemente comune și perspective parțial similare pentru a produce o panoramă sau o imagine de rezoluție mare.

Procedeul de lipire a două imagini

- Detectarea punctelor cheie.
- Descrierea modelelor definite de punctele cheie.
- Împerecherea punctelor ce reprezintă același model.
- Crearea omografiei imaginilor și potrivirea acestora.

Punctele cheie sunt puncte de interes într-o imagine, acestea sunt distincte și pot fi identificate fiabil, chiar și sub diverse transformări.

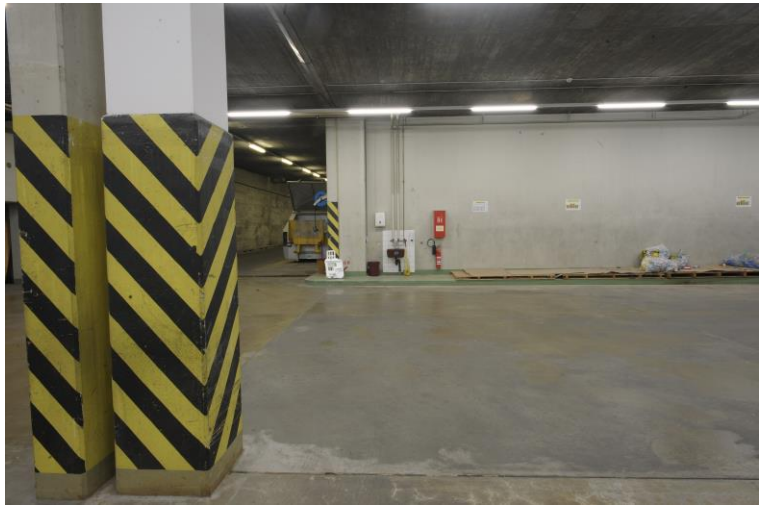
Aceste puncte servesc ca referințe pentru efectuarea sarcinilor de recunoaștere a obiectelor, potrivirea imaginilor, reconstrucția tridimensională și multe altele.

Descriere problemă

Detectorii standard de puncte cheie, deși sunt utilizați în diverse aplicați și pe scară largă, aceștia nu sunt perfecți. Printre limitele lor se numără:

- Sensibilitate la schimbări de scală (Harris).
- Sensibilitate la schimbări de rotație (Shi-Tomasi).
- Sensibilitate la zgomot (FAST).
- Cost computațional ridicat (SIFT).
- Sensibilitate la schimbări de lumină (ORB)

În cazul detectorilor mai dezvoltați aceștia sunt mai stabili, dar implementările lor sunt limitate la anumite limbaje de programare sau protejate de licențe.



Scopul lucrări

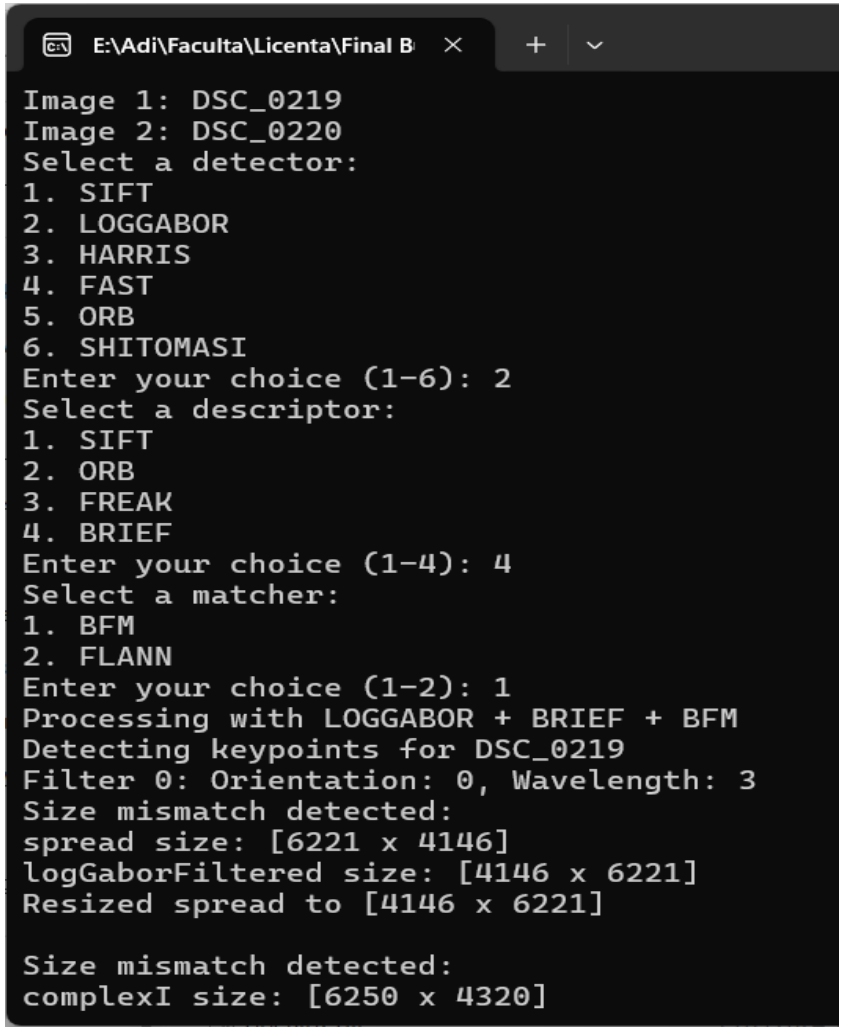
Aplicația dezvoltată în cadrul acestui proiect realizează lipirea a două imagini folosind diferite combinații de algoritmi.

Rolul Aplicației

- să ofere rezultate comparative pentru seturile de algoritmi folosiți;
- să arate utilitatea algoritmului implementat în cadrul aplicației față de cei standard;
- să ofere o alternativă în limbajul C++ a implementării filtrelor Log-Gabor.

Aplicația

- Aplicația dezvoltată este un sistem de procesare de imagini ce se folosește de tehnici tradiționale și metode avansate de detectare a punctelor cheie.
- Aceasta poate să se folosească de diferite combinații de detectori, descriptori și împerechetori pentru a obține o imagine panoramică.



Algoritmii folosiți în aplicație

Detecție

- Detectorul de colțuri Harris
- Detectorul de colțuri Shi-Tomasi
- FAST
- ORB
- SIFT
- Detector bazat pe filtrele Log-Gabor (implementat în cadrul aplicației)

Descriere

- BRIEF
- FREAK
- ORB
- SIFT

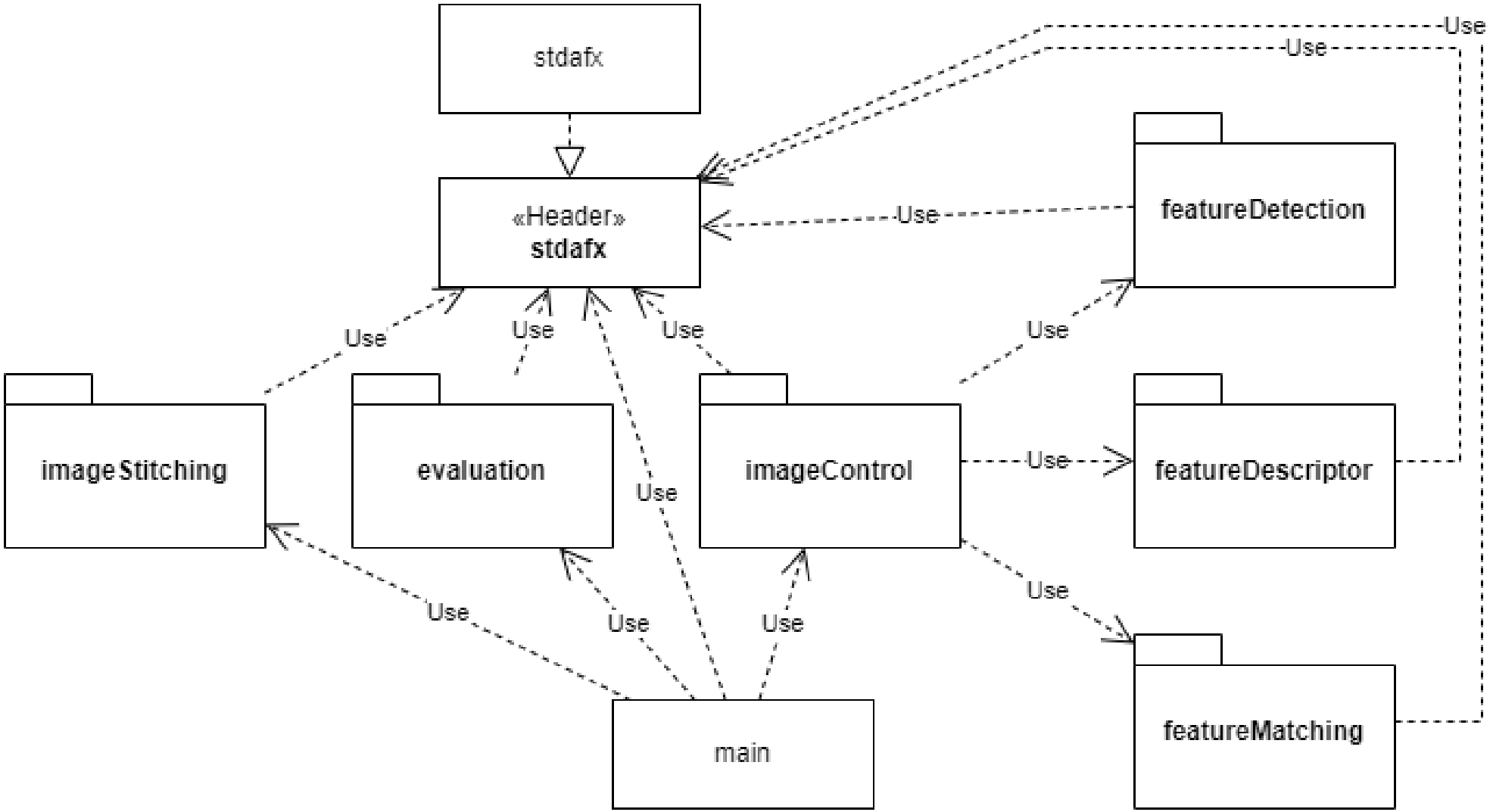
Împerechere

- Potrivirea prin forță brută (BFM)
- Potrivirea FLANN

Omografie

- RANSAC

Structura aplicației



Principalele Pachete

Pachetul detectorilor

featureDetection

Pachetul descriptorilor

featureDescriptor

Pachetul împerechetorilor

featureMatching

Pachetul metricilor

evaluation

Conține funcțiile de verificare, evaluare și validare a algoritmilor folosiți

Pachetul de control

imageControl

Conține funcțiile de manipulare a imaginilor

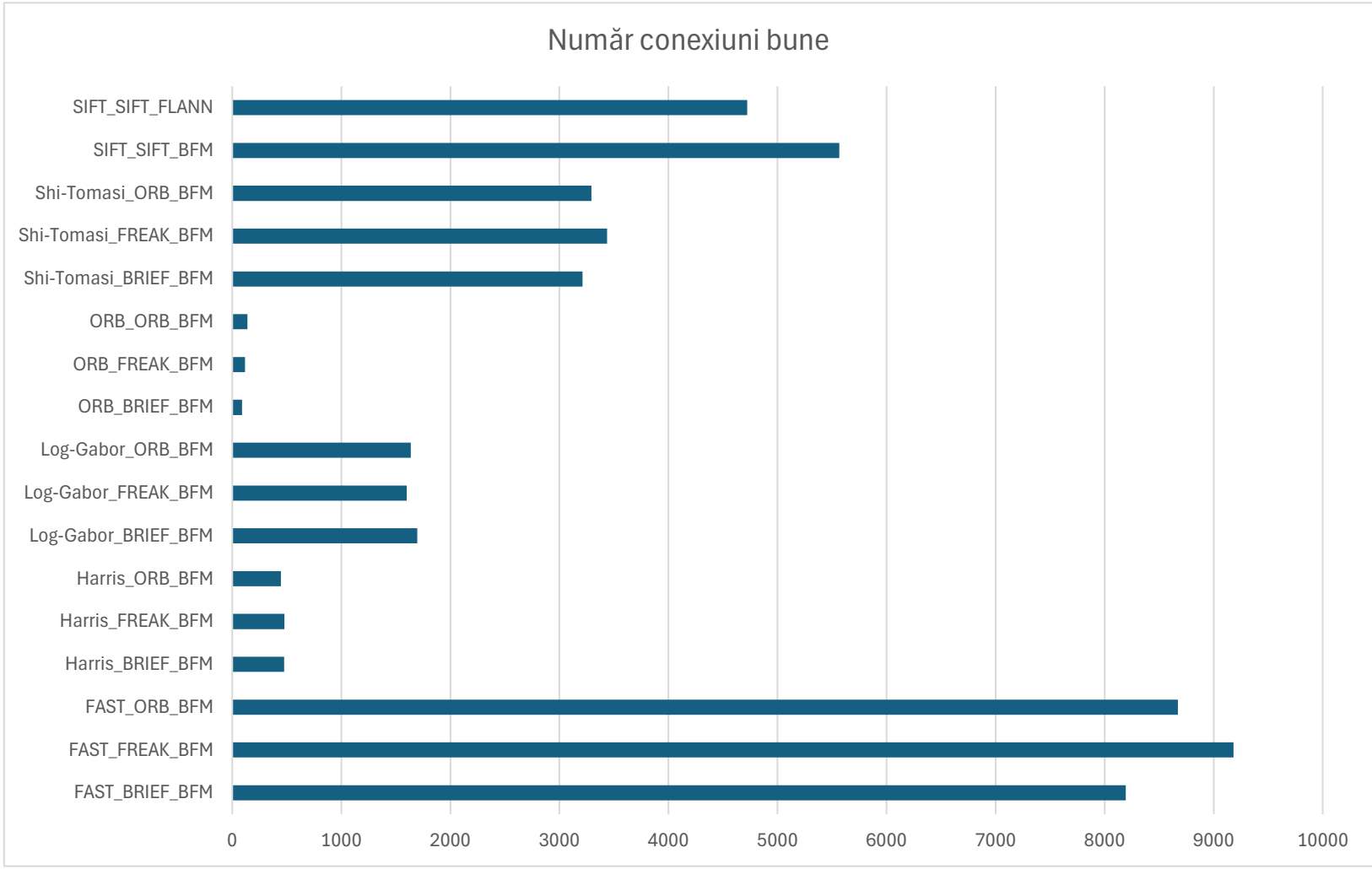
Pachetul de lipire

imageStitching

Conține funcțiile necesare lipirii imaginilor considerând că punctele cheie au fost detectate și împerecheate

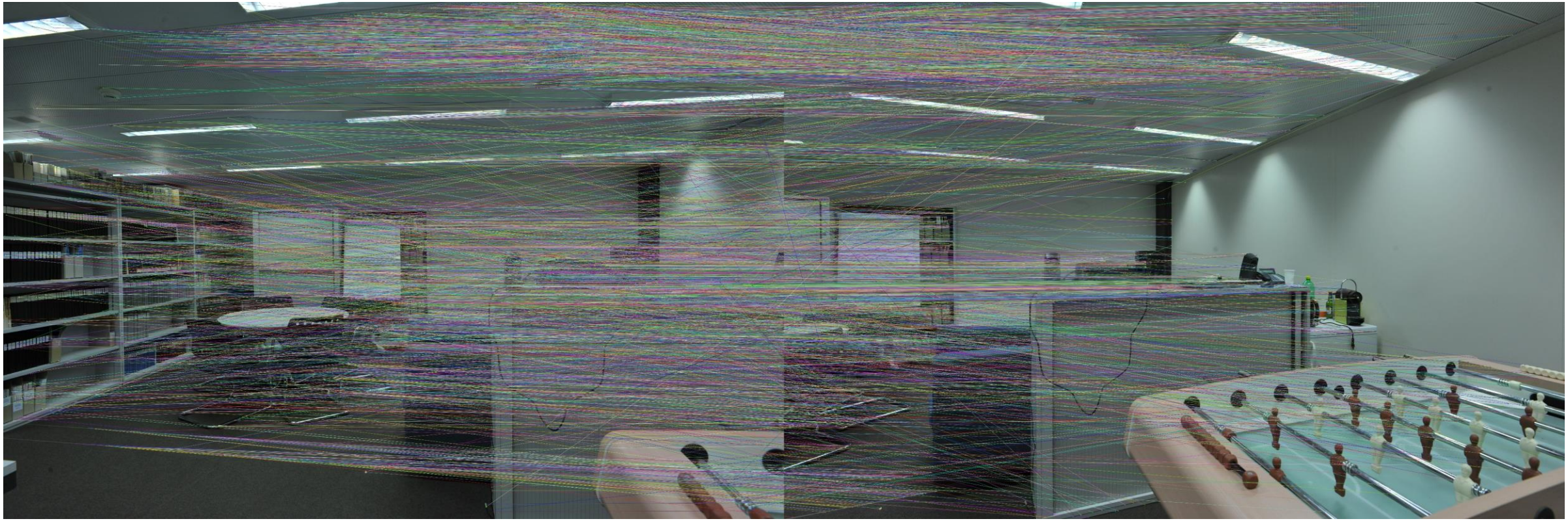
Funcționarea aplicației

- Aplicația necesită două imagini ce au câmpuri vizuale suprapuse.
- Aceasta va salva pe lângă imaginea rezultată lipirii celor două, rezultate parțiale și date despre eficiența întregului proces: punctele cheie detectate, numărul de potriviri și metrice de evaluare.



Rezultate parțiale

- Punctele cheie detectate pentru fiecare imagine.
- Corelațiile între imagini
- Filtrele Log-Gabor folosite
- Imaginile înmulțite cu filtrul Log-Gabor

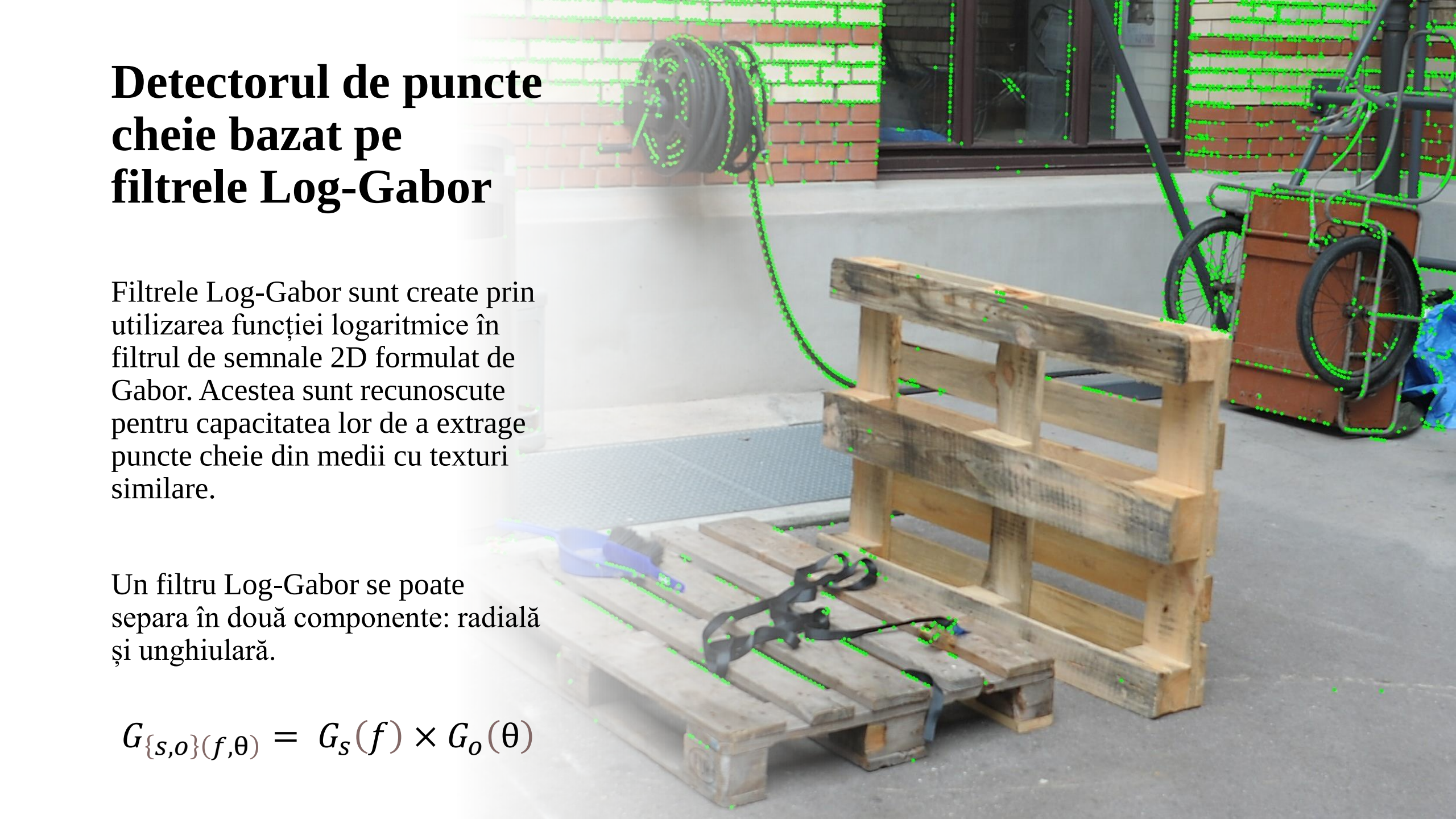


Detectorul de puncte cheie bazat pe filtrele Log-Gabor

Filtrele Log-Gabor sunt create prin utilizarea funcției logaritmice în filtrul de semnale 2D formulat de Gabor. Acestea sunt recunoscute pentru capacitatea lor de a extrage puncte cheie din medii cu texturi similare.

Un filtru Log-Gabor se poate separa în două componente: radială și unghiulară.

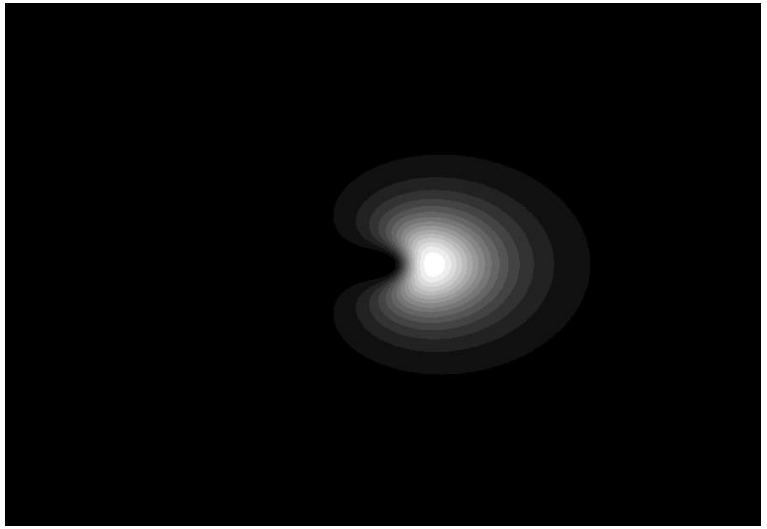
$$G_{\{s,o\}(f,\theta)} = G_s(f) \times G_o(\theta)$$



Detectorul bazat pe filtre Log-Gabor

Imagine originală

Filtrul



Imaginea X Filtru

Formula componentei radiale:

- $fs = 1/\lambda$: Frecvența radină a filtrului.
- λ : Lungimea de undă a filtrului.
- σ_{fs} : Lățimea de bandă radială de Bf în octave definite ca:

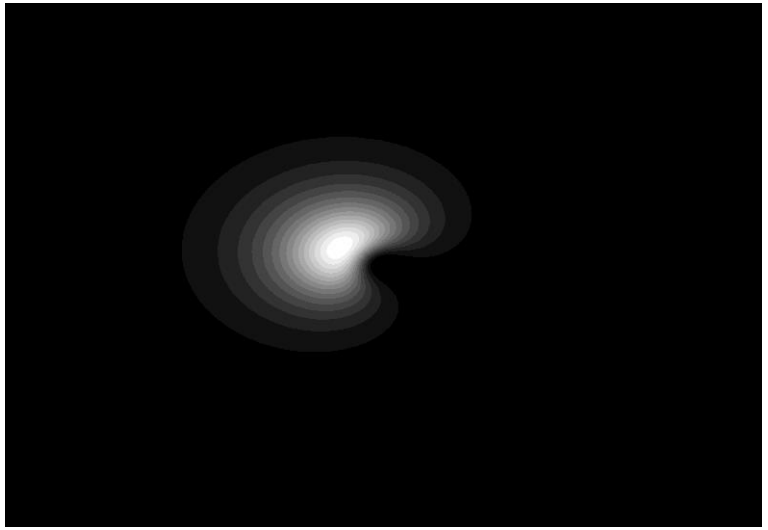
$$B_f = 2\sqrt{2\backslash log(2)}\left|\, log\left(\sigma_f / f_s\right)\right|$$

$$G_{s(f)} = exp\left(-\frac{\left(log\left(\frac{f}{f_s}\right)\right)^2}{2\left(log\left(\frac{\sigma_f}{f_s}\right)\right)^2}\right)$$

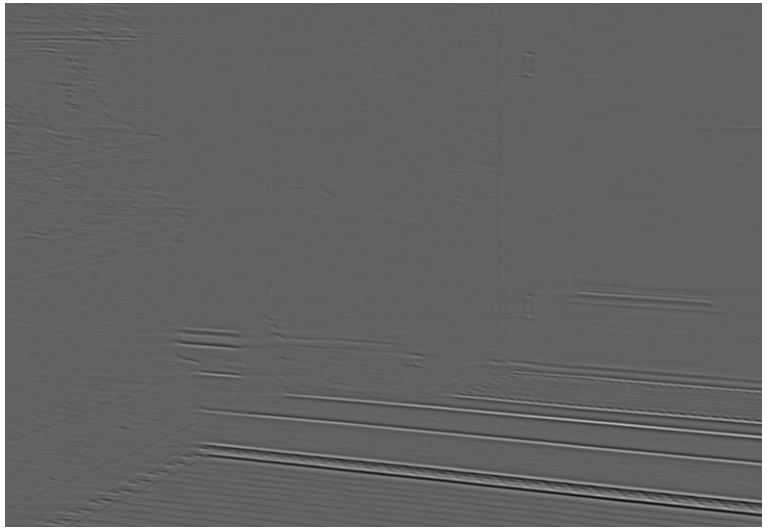
Detectorul bazat pe filtre Log-Gabor

Imagine originală

Filtrul



Imaginea X Filtru



Formula componentei unghiulare:

- θo : Unghiul de orientare al filtrului
- $\sigma\theta o$: Lățimea de bandă a lui $B\theta$ în octave.

$$G_o(\theta) = exp\left(-\frac{(\theta - \theta_o)^2}{2\sigma^2_{\{\theta_o\}}}\right) \qquad B_\theta = 2\sigma_\theta\sqrt{2\backslash log(2)}$$

Din cauza limitării limbajului C++ a fost necesară alternativa:

$$G_o(\sigma) = \exp(-|atan2(sin(\theta)\cdot cos(angl) - cos(\theta)\cdot sin(angl), cos(\theta)\cdot cos(angl) + sin(\theta)\cdot sin(angl))|^2 \times \frac{1}{2\sigma^2})$$

Unde : $\theta = atan2(-y, x)$

Modulul detectorului bazat pe filtre Log-Gabor

- createNormalizedRadius: construiește o matrice care conține distanțele normalizate de la fiecare punct.
- createLogGaborFilter: construiește componenta radială a filtrului.
- createLowPassFilter: construiește un filtru trece-jos.
- createAngularComponent: construiește componenta unghiulară a filtrului.
- findKeypoints: are rolul de a extrage punctele de interes.

Modulul detectorului bazat pe filtre Log-Gabor

- filterKeypointsByDistance: elimină din punctele de interes cele ce sunt prea apropiate.
- checkAndResizeIfNeeded: verifică și ajustează dimensiunile a două matrici.
- generateFilterBankParameters: generează multiple seturi de parametrii pentru filtrele Log-Gabor.
- detectLogGaborKeypoints: constă în apelarea aplicării filtrului Log-Gabor și găsirea punctelor de interes.
- applyLogGaborFilter: generează un filtru Log-Gabor și-l aplică unei imagini.

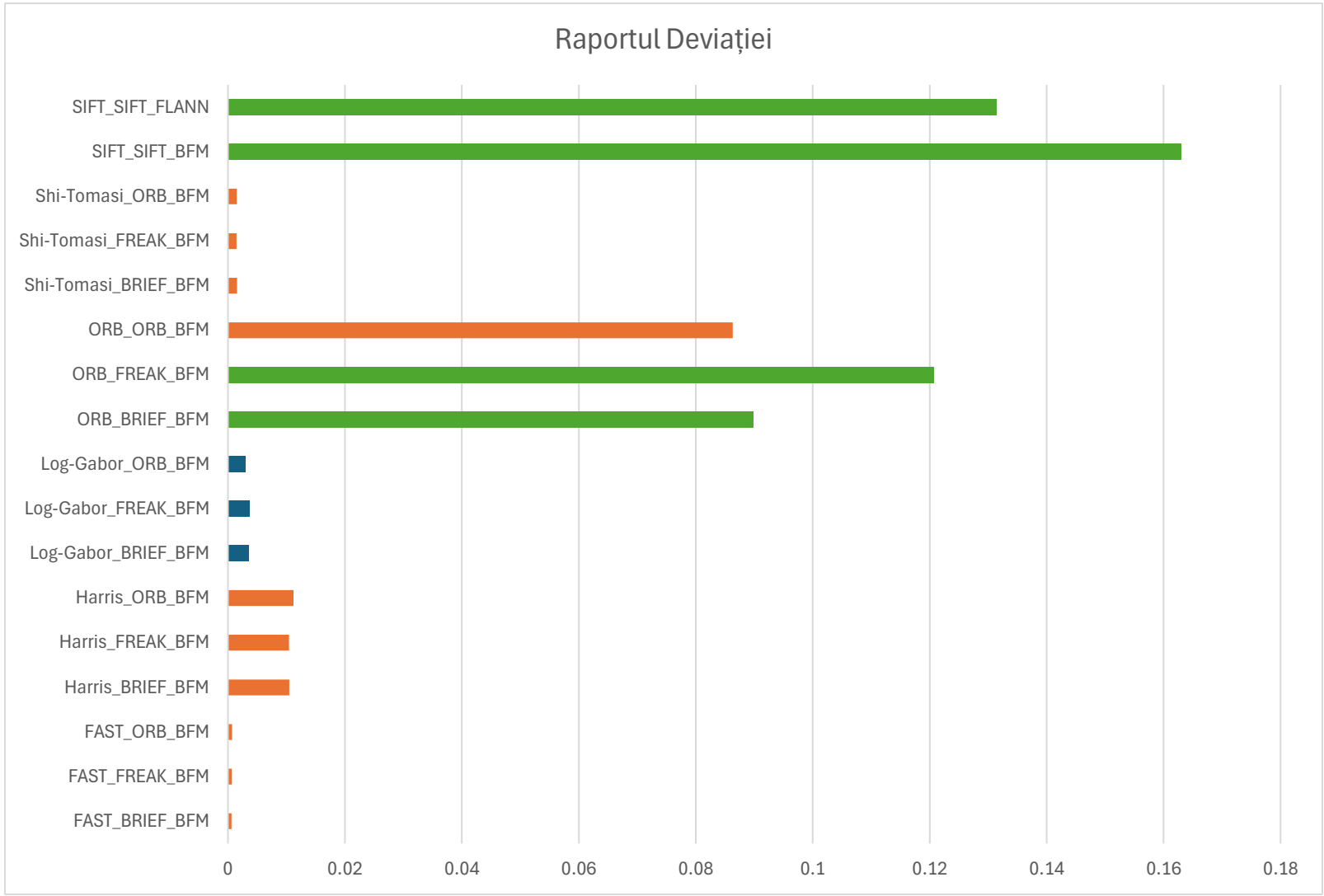
Pseudocodul pentru generarea și aplicarea unui filtru Log-Gabor

```
Function applyLogGaborFilter(src_gray, angl, sig_fs, lam, theta_o, threshold, cutoff, sharpness, imageName, id)
        radius := createNormalizedRadius(rows, cols);
        logGabor := createLogGaborFilter(radius, lam, sig_fs);
        lowPassFilter := createLowPassFilter(radius, cutoff, sharpness);
        multiply(logGabor, lowPassFilter, logGaborFiltered);
        spread := createAngularComponent(cols, rows, angl, theta_o);
        multiply(spread, logGaborFiltered, filter);
        m := getOptimalDFTSize(rows);
        n := getOptimalDFTSize(cols);
        copyMakeBorder(src_gray, padded, 0, m - rows, 0, n - cols);
        dft(complexI, complexI);
        fftShift(complexI, complexI);
        applyFilterManually(complexI, filter);
        fftShift(complexI, complexI);
        idft(complexI, imgf, DFT_REAL_OUTPUT);
        normalize(response, response, 0, 1, NORM_MINMAX);
return response;
```

Rezultate statistice

În graficul raportului deviației se poate vedea:

- cu verde valorile seturilor ce au obținut o lipire acceptabilă din punct de vedere vizual
- cu albastru valorile seturilor ce au folosit detectorul bazat pe filtre Log-Gabor

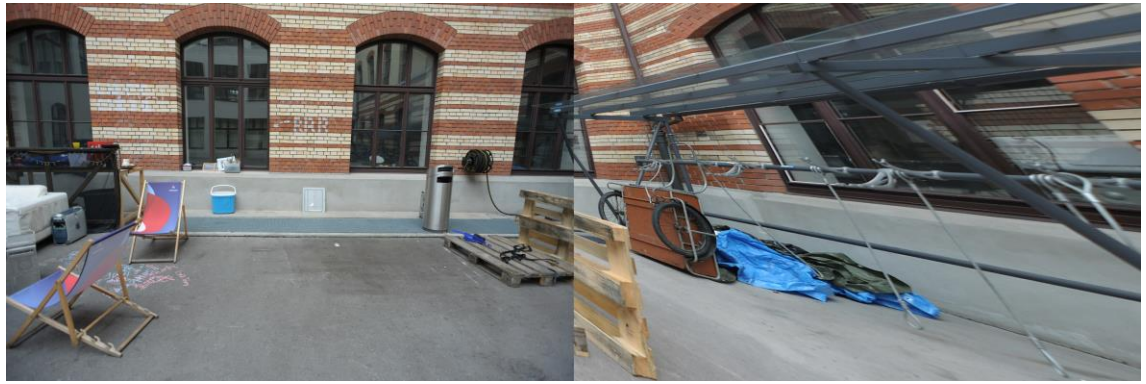


Rezultate lipiri

SIFT_SIFT_BFM

ORB_BRIEF_BFM

ORB_FREAK_BFM



SIFT_SIFT_FLANN

Concluzii

Observând rezultatele se pot trage următoarele concluzii:

- Implementarea detectorului bazat pe filtrele Log-Gabor este stabilă, aceasta putând să ofere rezultate constante în combinație cu diferiți algoritmi de detecție.
- Detectorul bazat pe filtre Log-Gabor deși are o bază stabilă, necesită calibrări și îmbunătățiri ca să poată depăși calitatea și eficiența celorlalte opțiuni.
- Aplicația este funcțională și are potențialul de a servi ca punct de plecare următoarelor dezvoltări a detectoarelor de puncte cheie în limbajul C++.

Bibliografie

- Definirea filtrului Log-Gabor: DJ, F. (1987). Relations Between the Statistics of Natural Images and the Response Properties of Cortical Cells. *Journal of the Optical Society of America A*, *4*(12), 2379-2394.
- Pentru formula alternativă: Kovesi, P. (n.d.). *What Are Log-Gabor Filters and Why Are They Good?* Retrieved from https://peterkovesi.com/matlabfns/PhaseCongruency/Docs/convexpl.html
- Articolul de start: Tze Kian Jong, D. B. (2023). An effective feature detection approach for image stitching of near-uniform scenes. *Signal Processing: Image Communication, 110*, 116872.
- Imagini: Computer Vision and Geometry Group, “ETH3D”, ETH Zurich, [Interactiv]. Available: https://www.eth3d.net/datasets#high-res-multi-view

Vă mulțumesc pentru atenție.