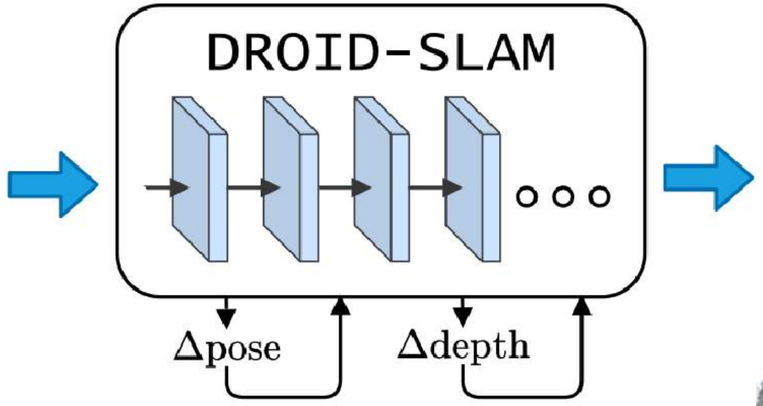
Localizare și mapare simultană folosind rețele neuronale



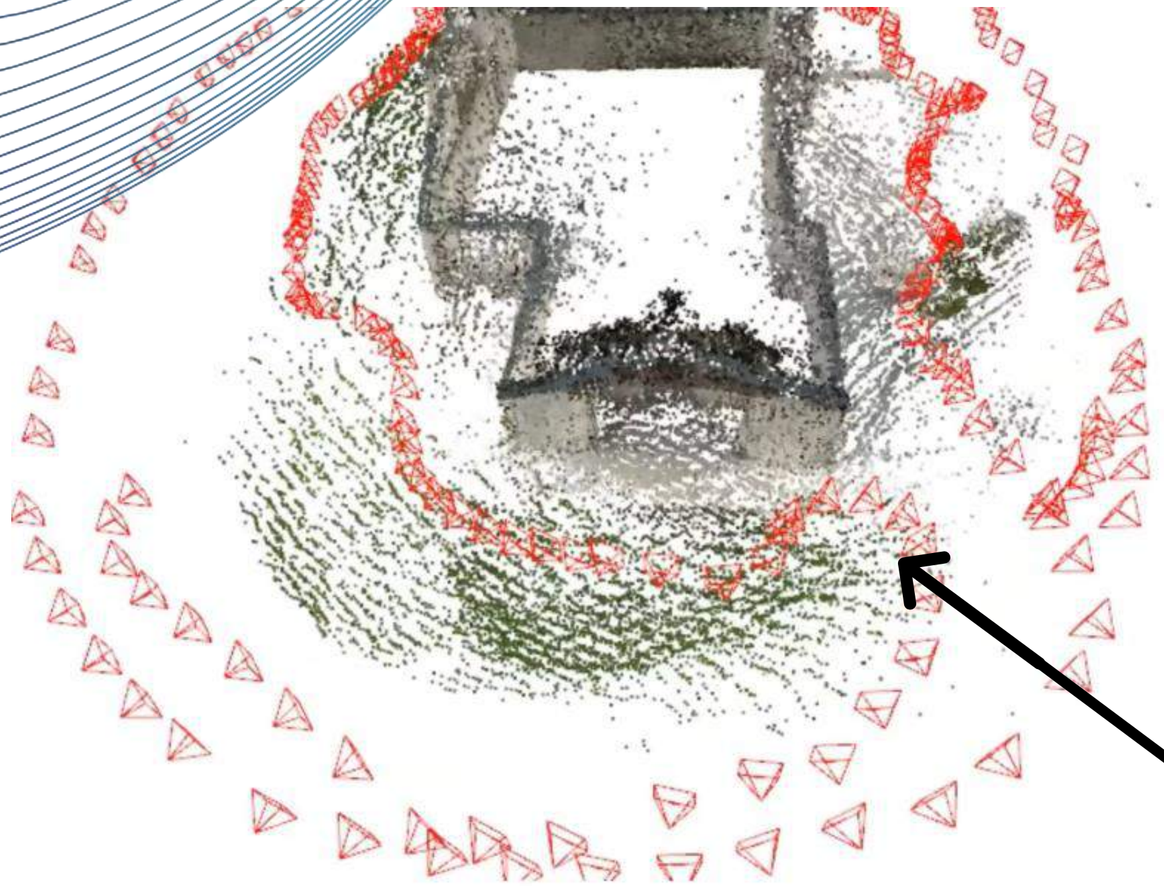
Student: Micudă Florin

Cadru didactic îndrumător:
Prof. Dr. Ing. Sorin Grigorescu

CUPRINS

- Introducere în SLAM și importanța acestei tehnologii
- Structure from Motion
- Lucrări relevante - Laboratorul de Viziune al Universității Princeton
- Arhitectura DROID-SLAM
- Tehnologii folosite
- Dataset-uri și experimente

Introducere în SLAM și importanța acestei tehnologii



Localizarea și maparea simultană (SLAM) este o tehnologie fundamentală în robotică și în sistemele autonome. SLAM permite unui dispozitiv să construiască o hartă a unui mediu necunoscut și, în același timp, să își determine locația în cadrul acestei hărți.

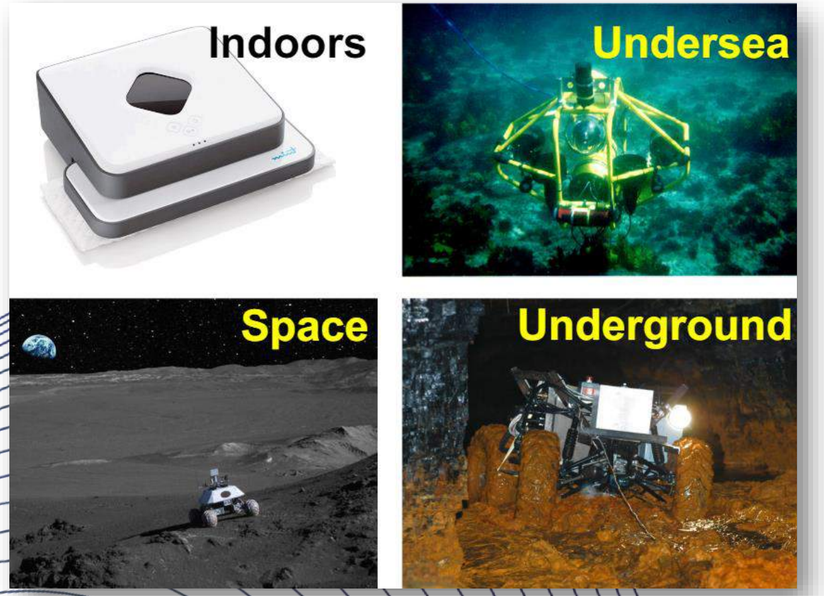
Localizarea în spațiu – poziția agentului

Reconstrucție 3D densă

Importanța SLAM

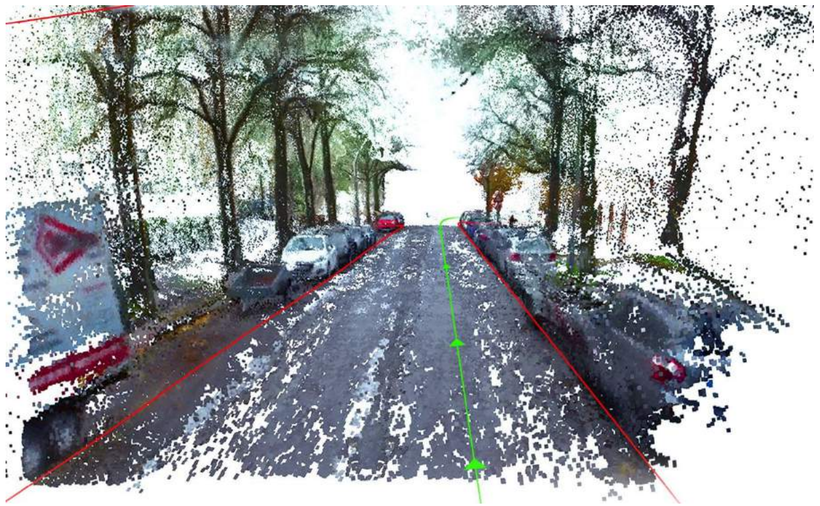
Robotică

Permite roboților să navigheze și să opereze în mod autonom în medii complexe.



Conducere Autonomă

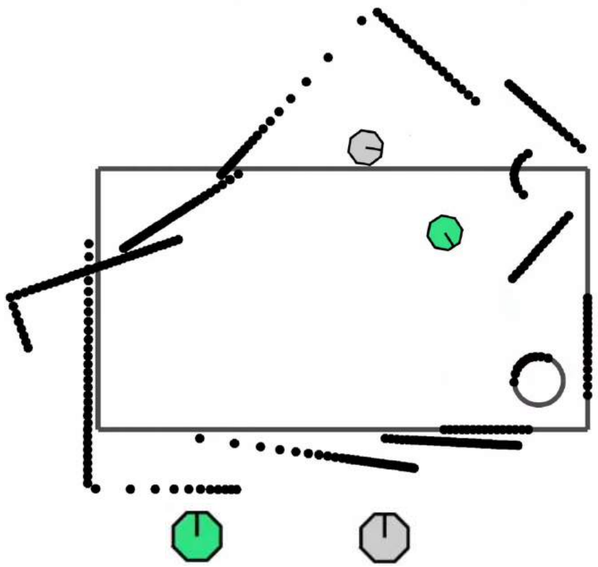
Esențial pentru ca mașinile care se conduc singure să poată naviga în siguranță fără GPS.



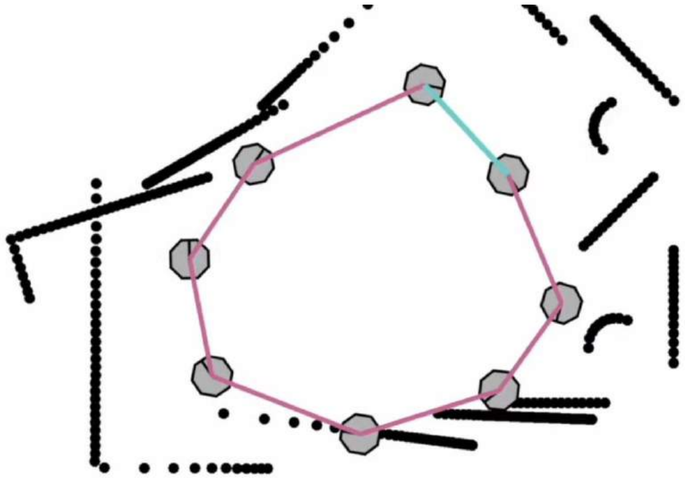
AR/VR

Îmbunătățește experiența utilizatorului prin maparea precisă și interacțiunea cu lumea reală.

Procesul general SLAM



Valorile **reale** și **estimate** ale agentului



Pose Graph Optimization

01 → **02** → **03** → **04** → **05**

Date Senzoriale – În cazul nostru datele de intrare vin de la camere RGB-D

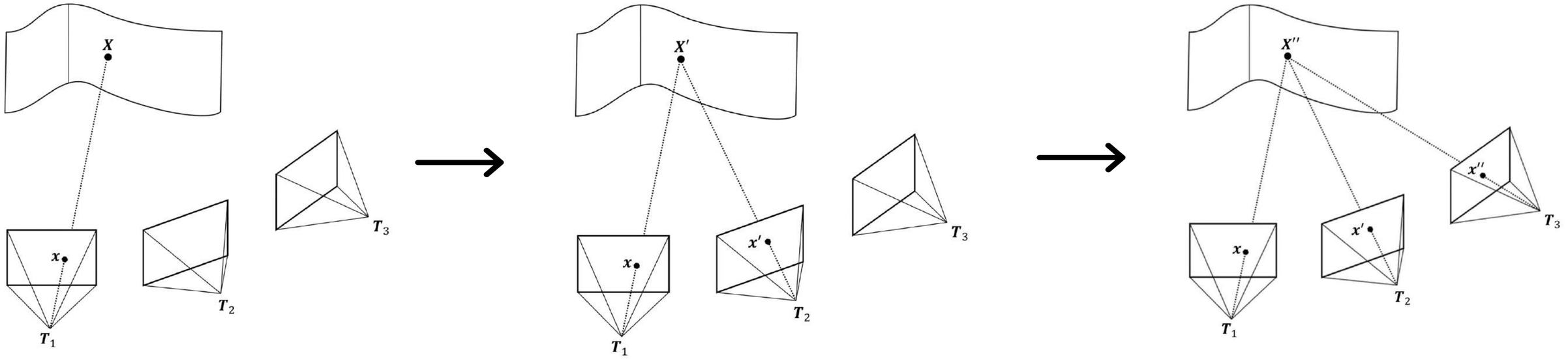
Feature Extraction și Feature Match

Pose Estimation

Loop Closure

Bundle Adjustment

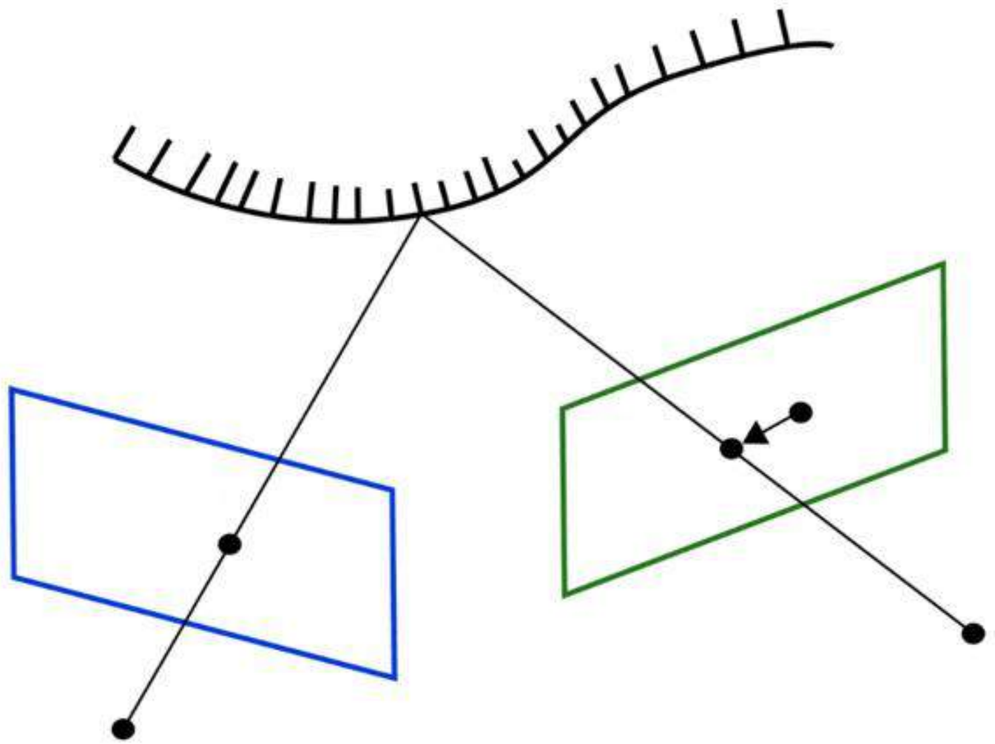
Structure from Motion și Flow



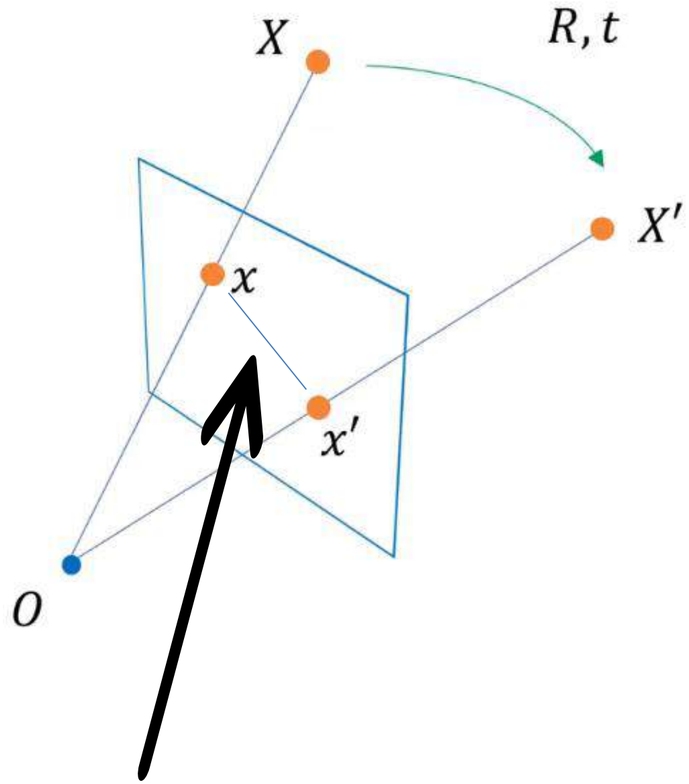
$$\boldsymbol{X} = \begin{pmatrix} x/d \\ y/d \\ 1/d \end{pmatrix} \qquad \boldsymbol{X'} = \boldsymbol{T}_2\boldsymbol{T}_1^{-1}\boldsymbol{X} \qquad \boldsymbol{x'} = \begin{pmatrix} X'/Z' \\ Y'/Z' \end{pmatrix} \longrightarrow \omega_{ij}(\boldsymbol{T}, \boldsymbol{d})$$

Structure from Motion și Flow

Optical Flow este o funcție analitică a adâncimii și poziției camerei



Structure from Motion și Flow



Flow Indus = x – x′

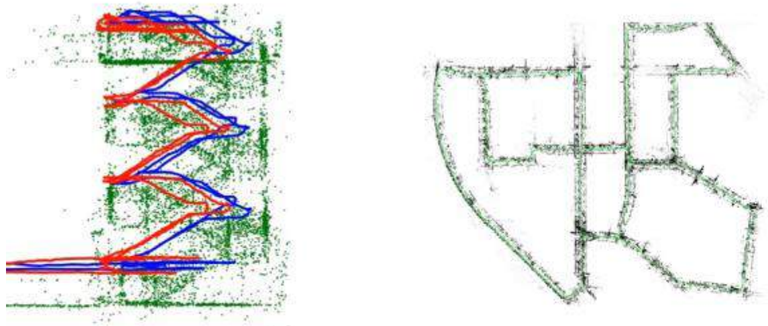
Lucrări SLAM anterioare

SLAM Indirect

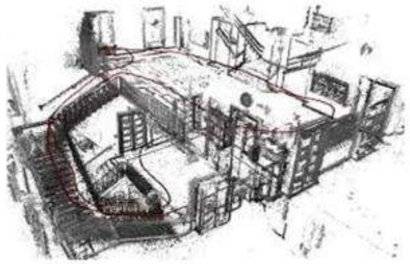
- detecteaza și potriveste features
- minimizeaza eroarea geometrică între punctele 2D observate și intre punctele 2D induse de soluție

SLAM Direct

- potrivește direct pixelii
- minimizeaza eroarea fotometrică între intensitățiile pixelilor observați în noul cadru și între pixelii induși de soluție

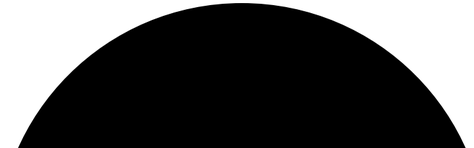


ORB-SLAM [Mur-Artal et al., 2015]
SVO [Forster et al. 2014]



LSD-SLAM [Engel et al., 2014]
DSO [Engel et al. 2016]

Robustitate insuficientă: Eșecuri dese și catostrofale



Lucrării relevante - Laboratorul de Viziune al Universității Princeton

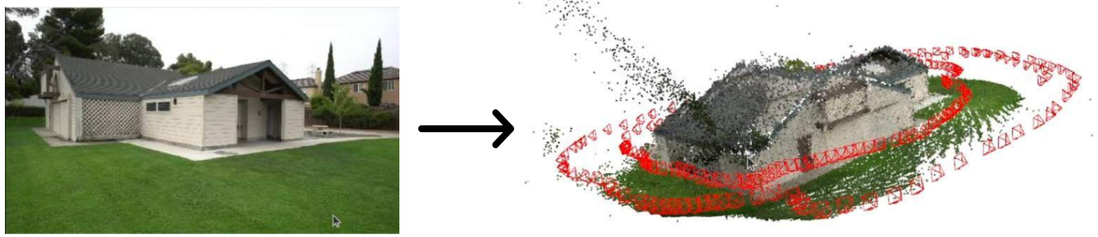
Optical Flow | Scene Flow | Stereo | SLAM / SfM | Tools

RAFT

Recurrent All Pairs Field Transforms for **Optical Flow**

DROID-SLAM

Differentiable recurrent optimization-inspired design



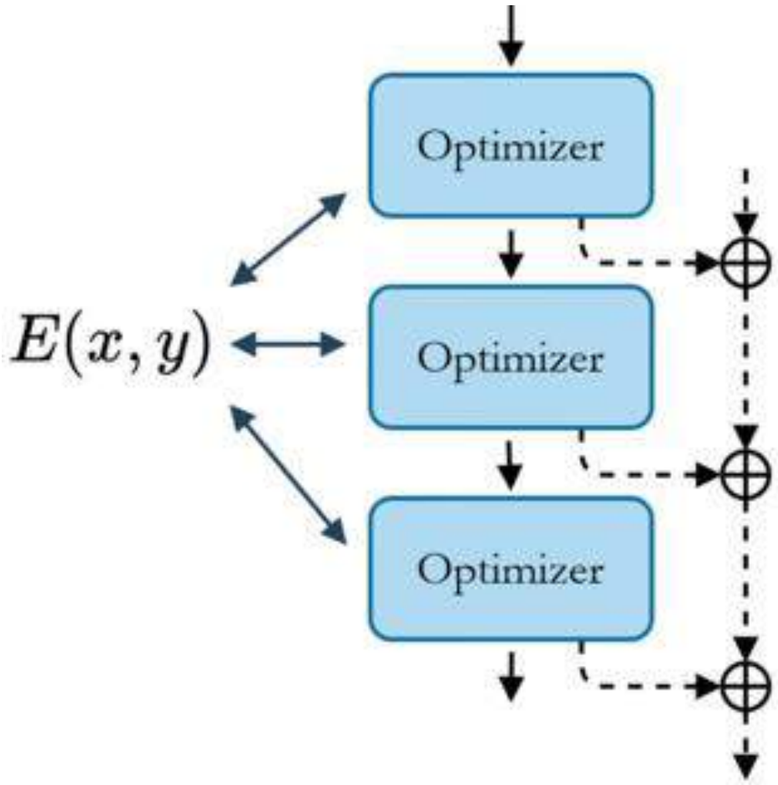
Lie-Torch: Tangent Space Backpropagation for 3D Transformation Groups

Arhitectura de bază

$$y^* = \min_{y \in Y} \boldsymbol{E}(x, y)$$

Optimization Variable
(e.g flow, disparity, pose)

Data
(e.g image, video, stereo frames)

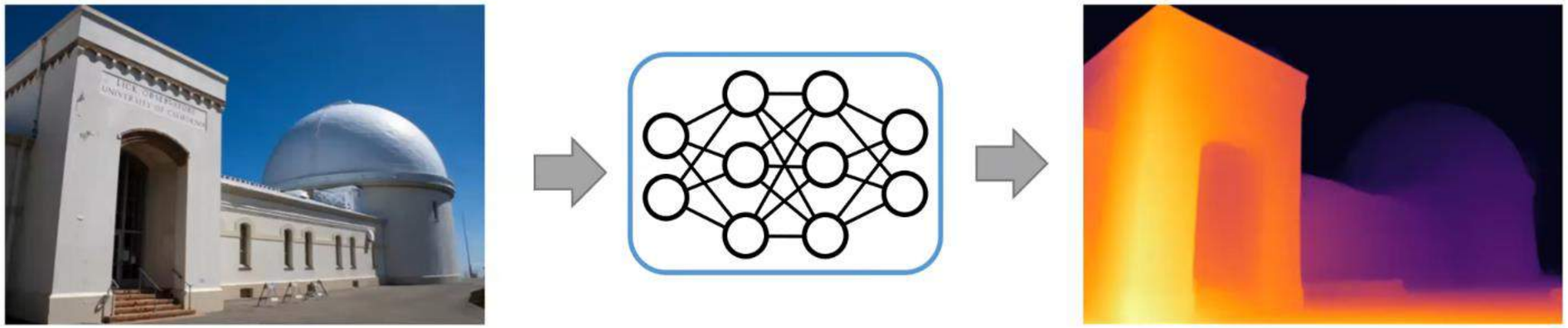


Funcția obiectiv: masurărea calitații soluției
Optimizer: algoritm de căutare care minimizează obiectivul

Limitare principala: Probleme de Multiview ridică probleme de optimizare extrem de complicate

Se vrea design-ul unei rețele neuronale care să se comporte ca o problemă continuă de optimizare "end-to-end"

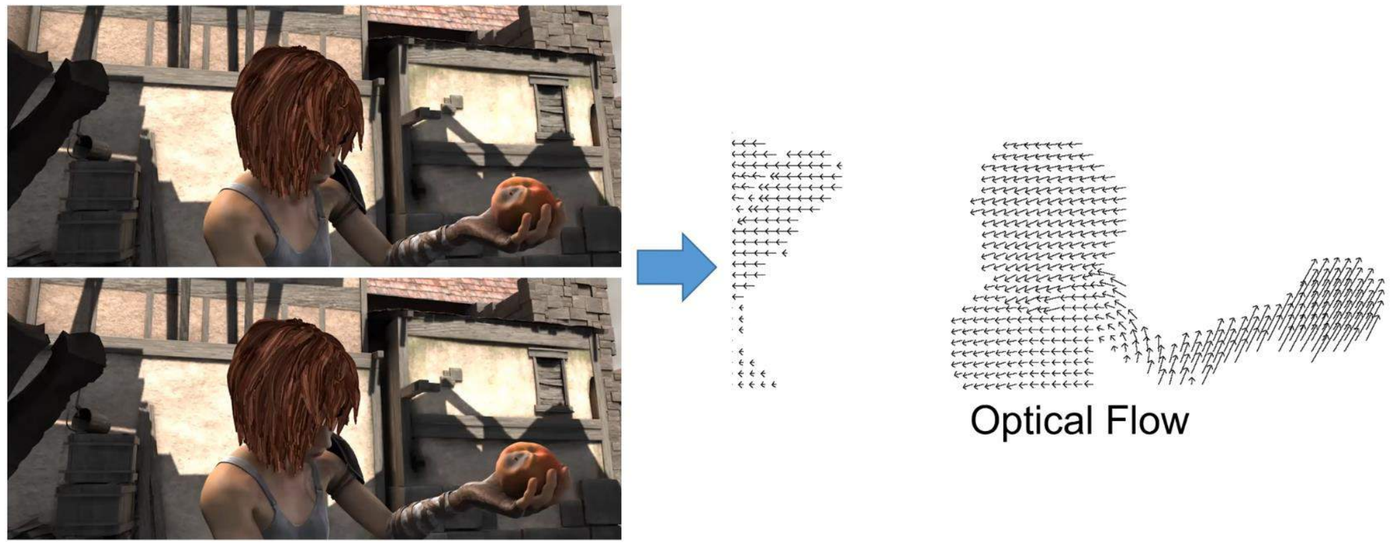
Arhitectura de bază



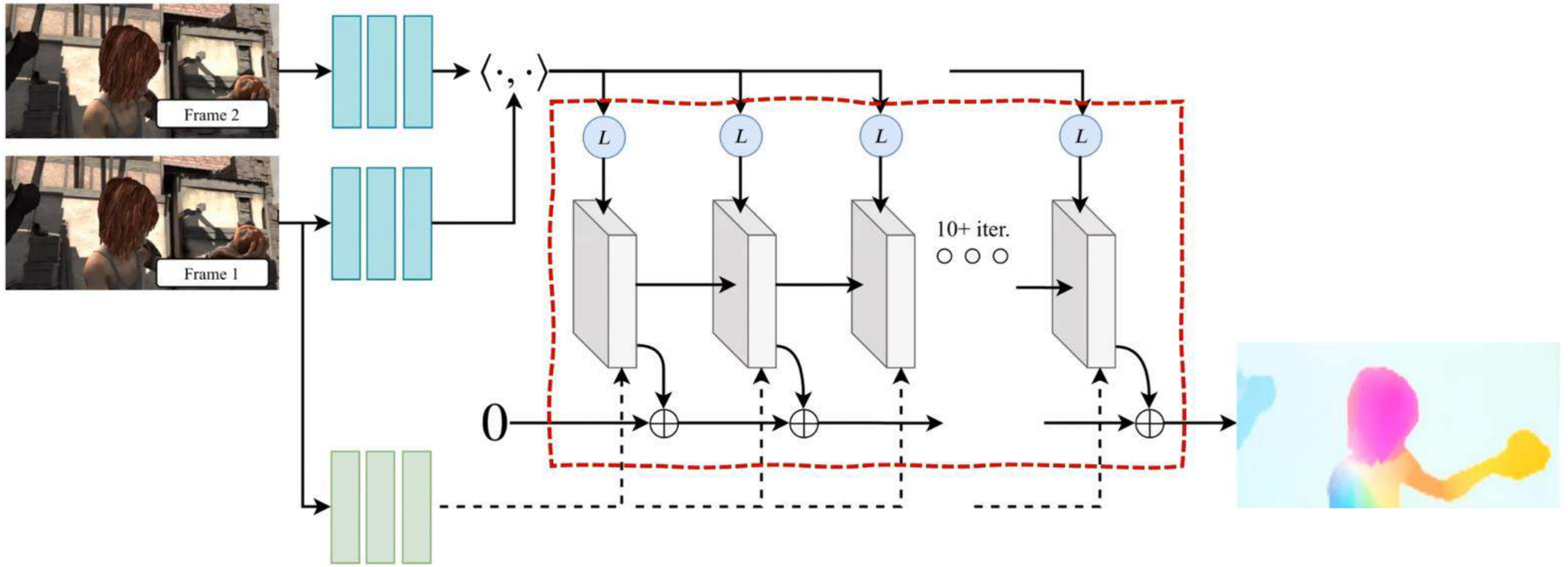
O rețea neuronală de tip deep learning poate fii antrenată pentru a estima adâncimea dintr-o imagine sau dintr-un video

Lucrării relevante – RAFT

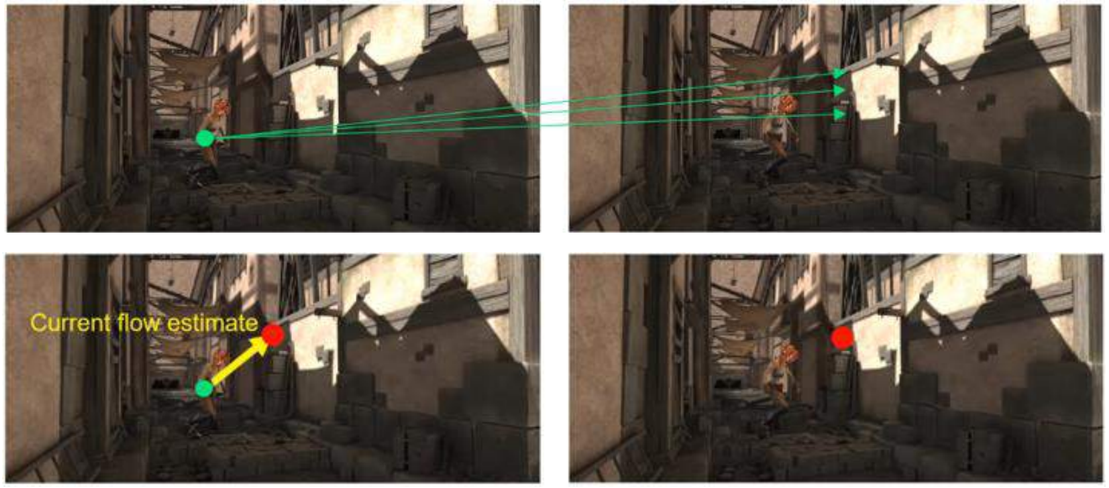
Mișcare 2D densă în funcție de pixeli între o pereche de frame-uri

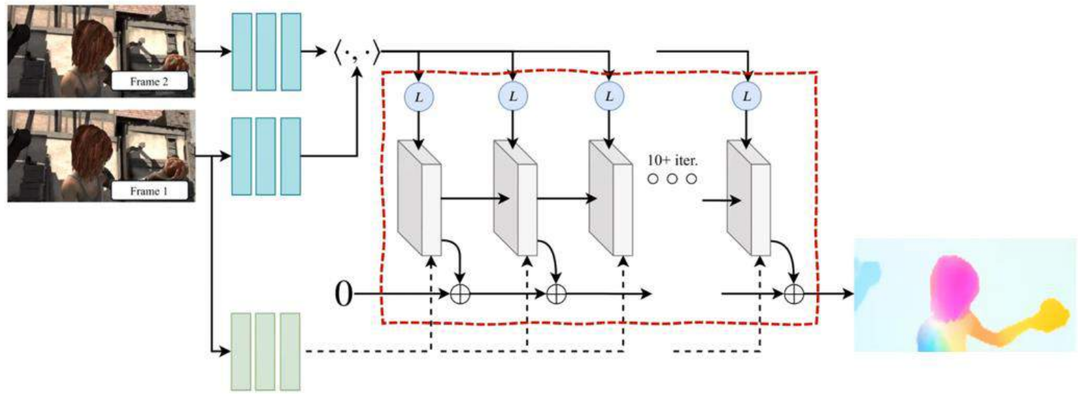


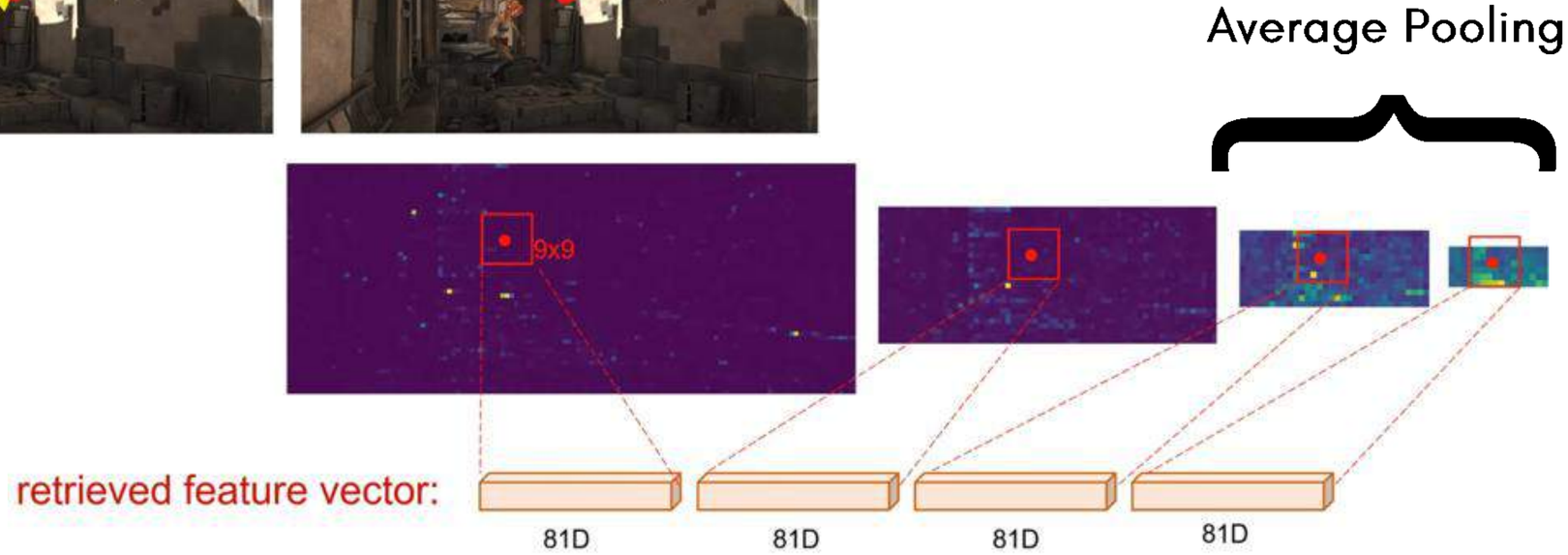
Arhitectura de bază - RAFT



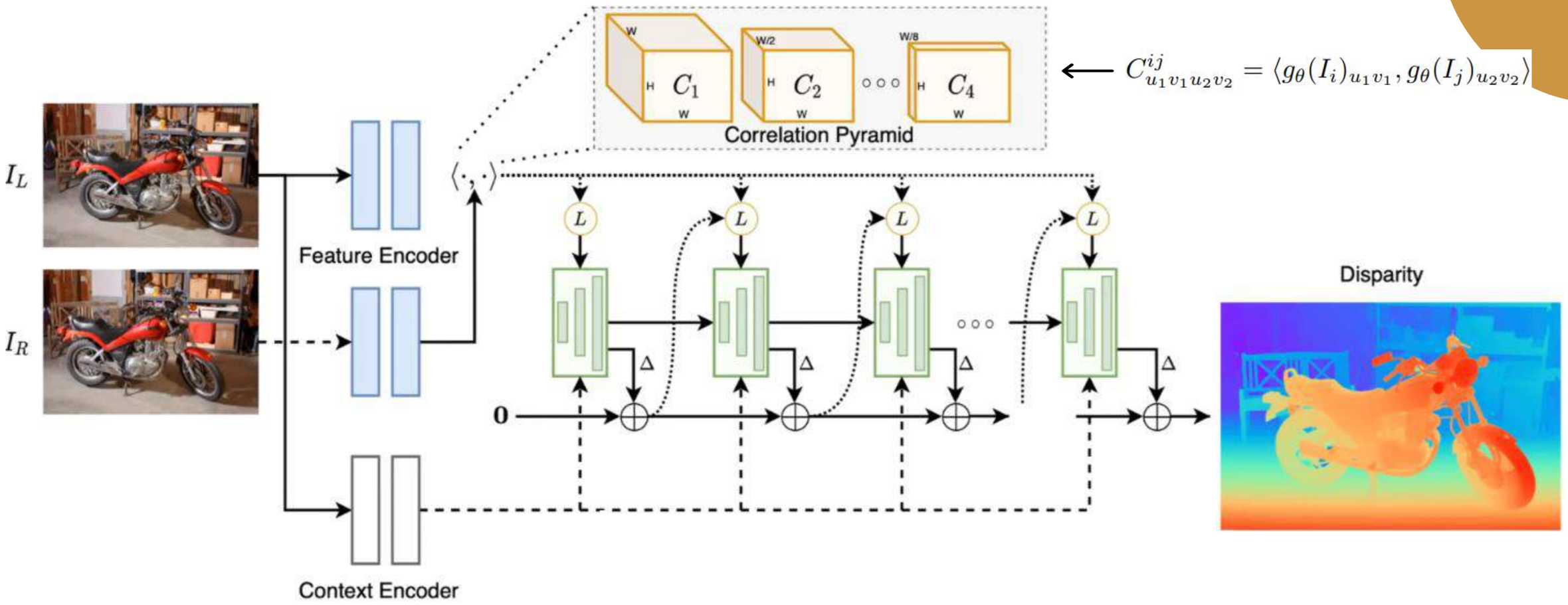
Extragerea caracteristicilor



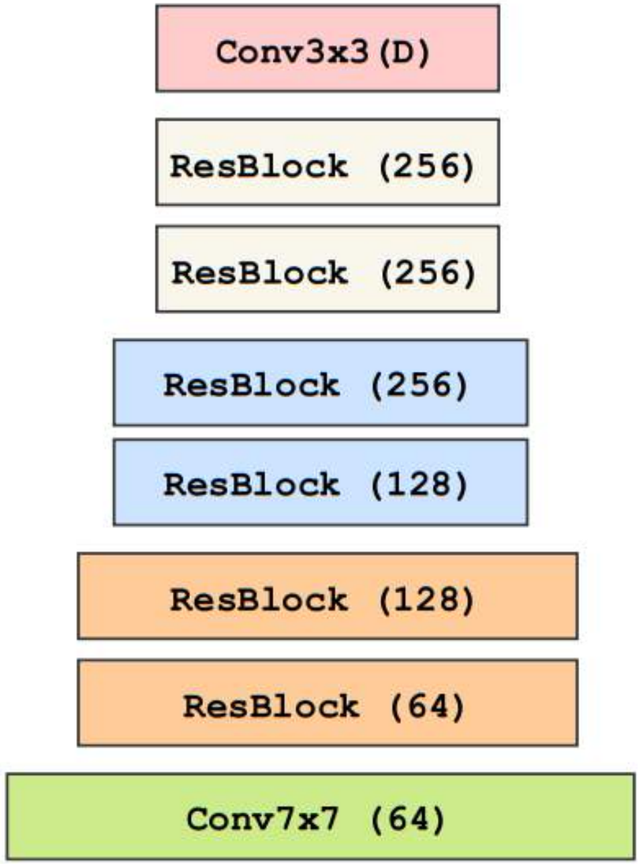




Arhitectura pentru detecția STEREO



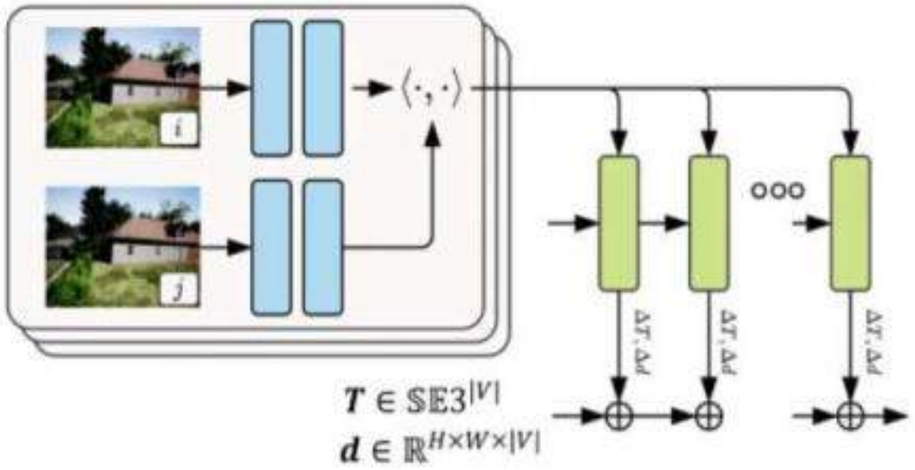
Arhitectura rețelei



Arhitectura codificatoarelor de caracteristici și de context. Ambele extrag caracteristici la 1/8 din rezoluția imaginii de intrare folosind un set de 6 blocuri reziduale de bază. Normalizarea instanțelor este utilizată în cazul caracteristicilorcodificatorul de caracteristici; în codificatorul de context nu se utilizează nicio normalizare. Codificatorul de caracteristici scoate caracteristici cu dimensiunea D=128, iar codificatorul de context scoate caracteristici cu dimensiunea D=256.

Actualizarea poziției

Se actualizează poziția camerei $G_t \in SE(3)$ și adâncimea inversă $d_t \in \mathbb{R}^{H \times W}$, iar co-vizibilitatea este modelată printr-un graf de cadre $V, \mathcal{E}$.



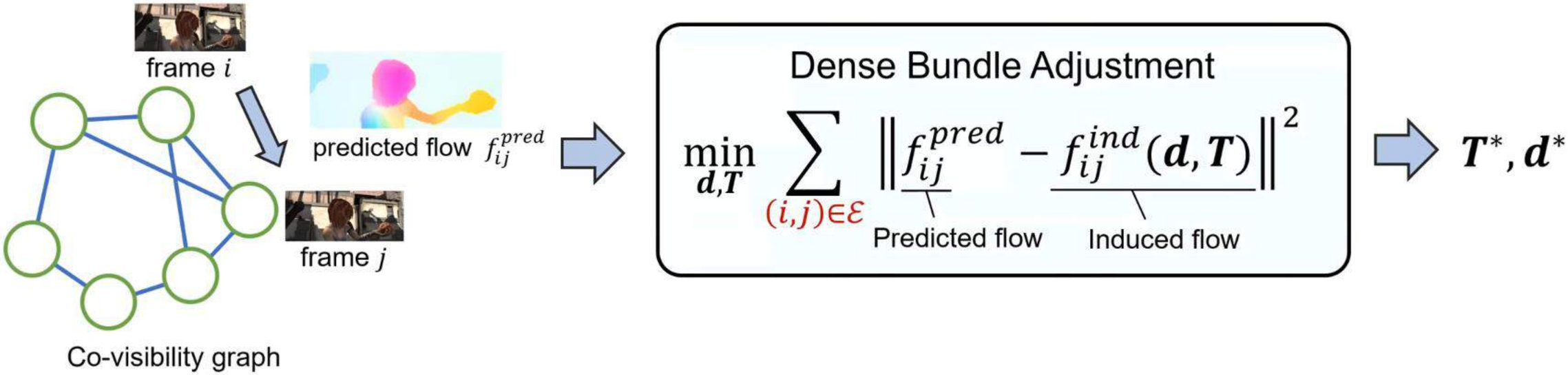
$$\boldsymbol{x}'_{ij} = \omega_{ij}\left(\boldsymbol{T}^{(k)}, \boldsymbol{d}^{k}\right) + \boldsymbol{r}_{ij}$$

$$\boldsymbol{T}^{(k+1)}, \boldsymbol{d}^{(k+1)} = \min_{\boldsymbol{T},\boldsymbol{d}} \sum_{(i,j)\in E} \left\| \omega_{ij}(\boldsymbol{T}, \boldsymbol{d}) - \boldsymbol{x}'_{ij} \right\|^2_{\mathrm{diag}(w_{ij})}$$

Dense Bundle Adjustment

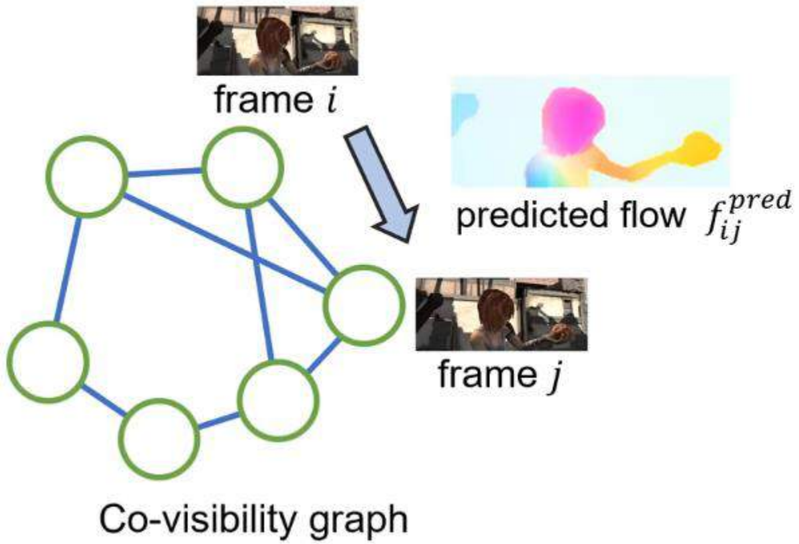
Input: graph de co-vizibilitate $(\mathcal{V}, \mathcal{E})$ și f_{ij}^{pred}

Vrem: mape de adâncimii $d = (d_1, \dots, d_i, \dots)$, pozițiile camerei $T = (T_1, \dots, T_i, \dots)$



Stratul de Dense Bundle Adjustment (DBA) ajustează poziția și adâncimea printr-o funcție de cost unde se folosesc metode de optimizare neliniară precum Gauss-Newton.

Dense Bundle Adjustment



Dense Bundle Adjustment

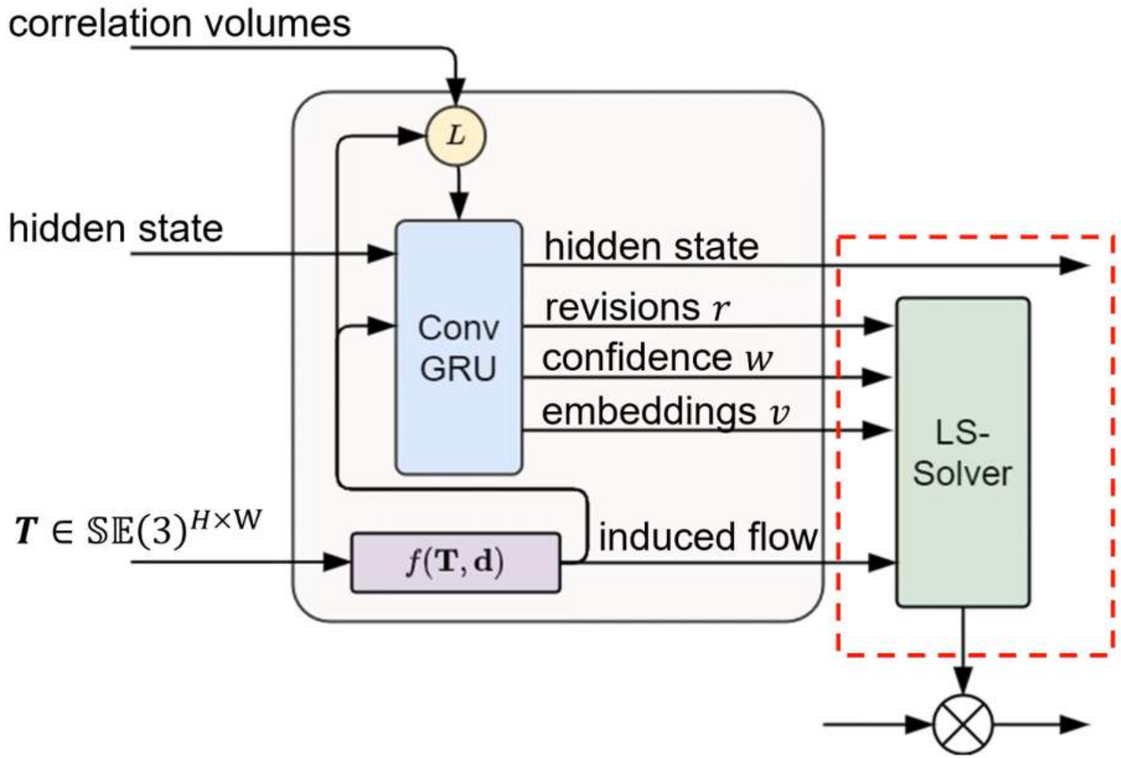
$$\min_{\boldsymbol{d},\boldsymbol{T}} \sum_{(i,j)\in\mathcal{E}} \left\| \underbrace{f_{ij}^{pred}}_{\text{Predicted flow}} - \underbrace{f_{ij}^{ind}(\boldsymbol{d},\boldsymbol{T})}_{\text{Induced flow}} \right\|^2 \Rightarrow \boldsymbol{T}^*, \boldsymbol{d}^*$$

$$\mathbf{E}(\mathbf{G}', \mathbf{d}') = \sum_{(i,j)\in\mathcal{E}} \left\| \mathbf{p}_{ij}^* - \Pi_c \left(\mathbf{G}_{ij}' \circ \Pi_c^{-1}(\mathbf{p}_i, \mathbf{d}_i') \right) \right\|_{\Sigma_{ij}}^2 \qquad \Sigma_{ij} = \operatorname{diag} \mathbf{w}_{ij}$$

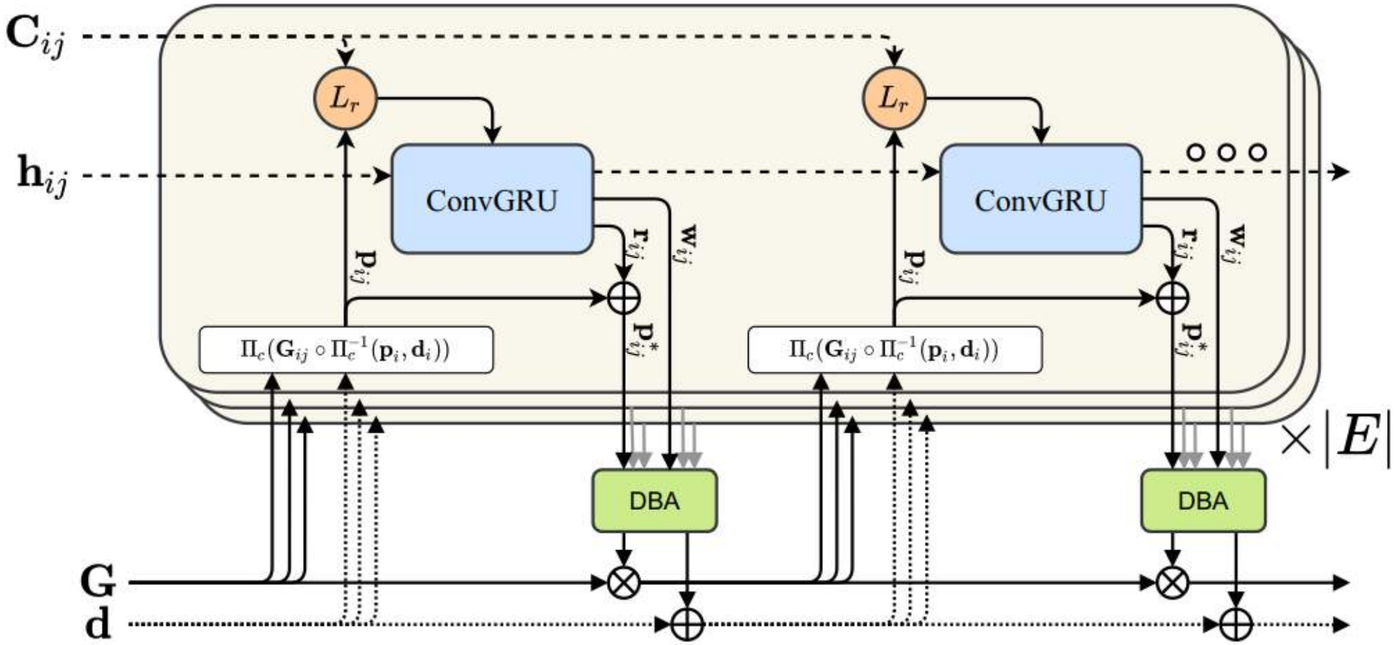
Componenta principală a sistemului SLAM este un operator de actualizare:

un **GRU convoluțional 3 x 3**

La fiecare iterație, GRU primește date de fundal, corelație și flux pentru a actualiza **starea ascunsă** $\boldsymbol{h}$ și a calcula revizia și ponderile de confidenta w.



Operatorul de actualizare



Actualizările poziției și adâncimii sunt aplicate estimărilor curente ale adâncimii și poziției:

$$\mathbf{G}^{(k+1)} = \exp\left(\Delta\xi^{(k)}\right) \circ \mathbf{G}^{(k)}$$

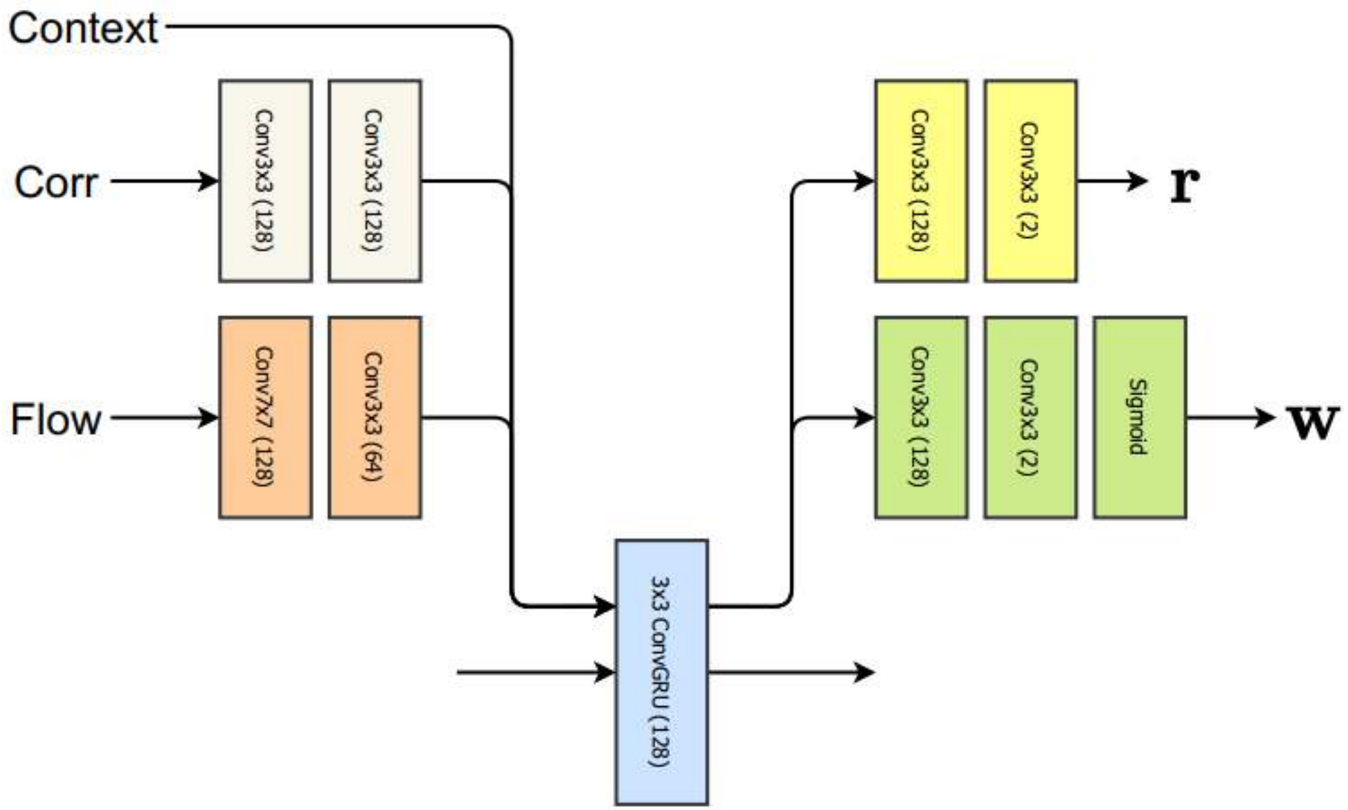
$$d^{(k+1)} = \Delta d^{(k)} + d^{(k)}$$

La începutul fiecărei iterații se folosesc estimările curente ale pozițiilor și adâncimilor pentru a estima corespondența. Având o rețea de coordonate de pixeli $\mathbf{p}_i \in \mathbb{R}^{H \times W \times 2}$ în cadrul i, calculăm câmpul de corespondență densă $\mathbf{p}_{ij}$.

$$\mathbf{p}_{ij} = \Pi_c \left(\mathbf{G}_{ij} \circ \Pi_c^{-1}(\mathbf{p}_i, d_i) \right)$$

$$\mathbf{G}_{ij} = \mathbf{G}_j \circ \mathbf{G}_i^{-1}$$

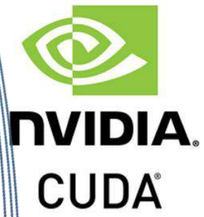
Arhitectura operatorului de actualizare



Tehnologii folosite

Se valorifică eficiența **Python** și bibliotecile sale

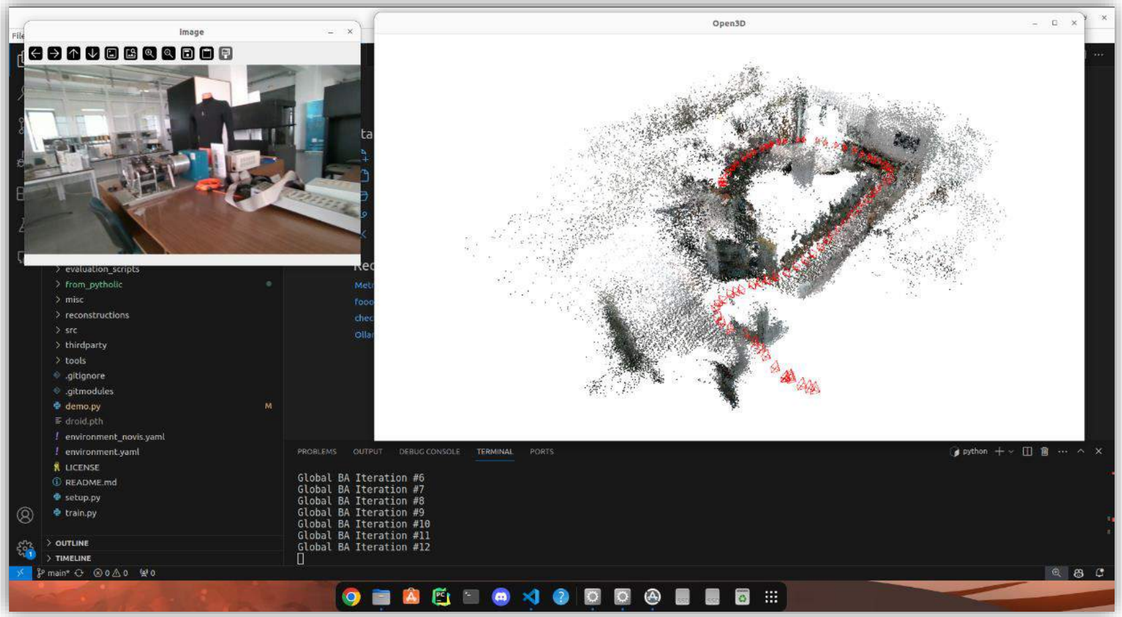
PyTorch, folosit pentru antrenarea rețelelor neuronale, facilitând deep learning-ul și prelucrarea avansată a datelor

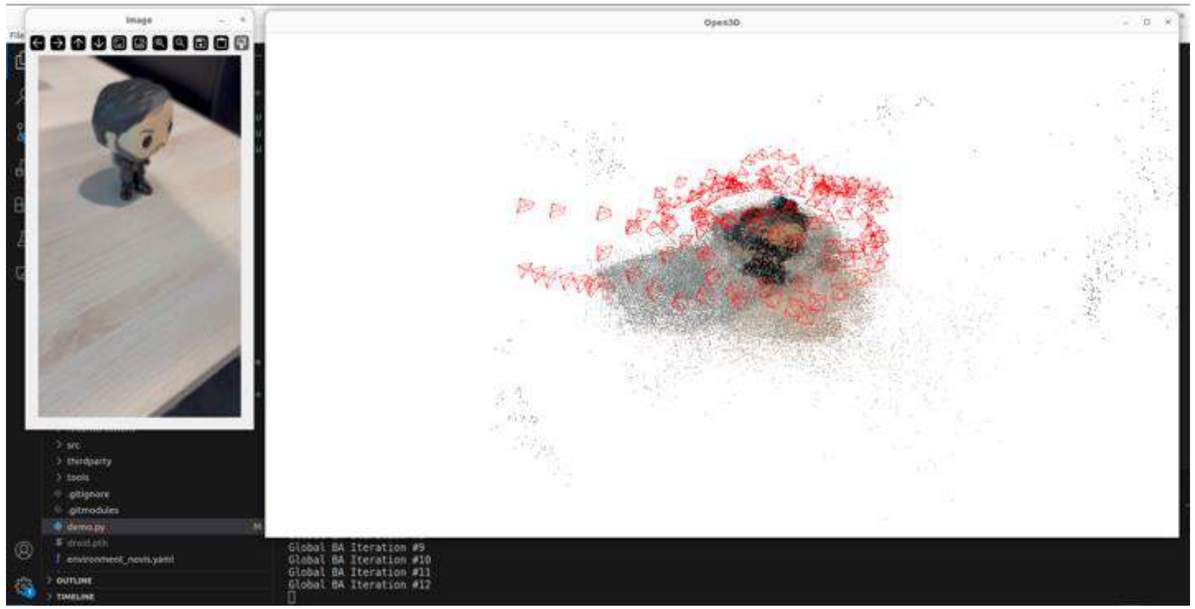


Accelerează calculul matematic si analiza rapida pentru SLAM, folosind GPU-ul

- **Lietorch:** optimizează transformările spațiale
- **RAFT:** urmărirea dinamica mediului
- **OPENCV și OPEN3D:** manipularea imaginilor, respectiv a structurilor 3D

Seturi de date și experimente





Printre seturile de date utilizate se numără KITTI, EuRoc și Freiburg.

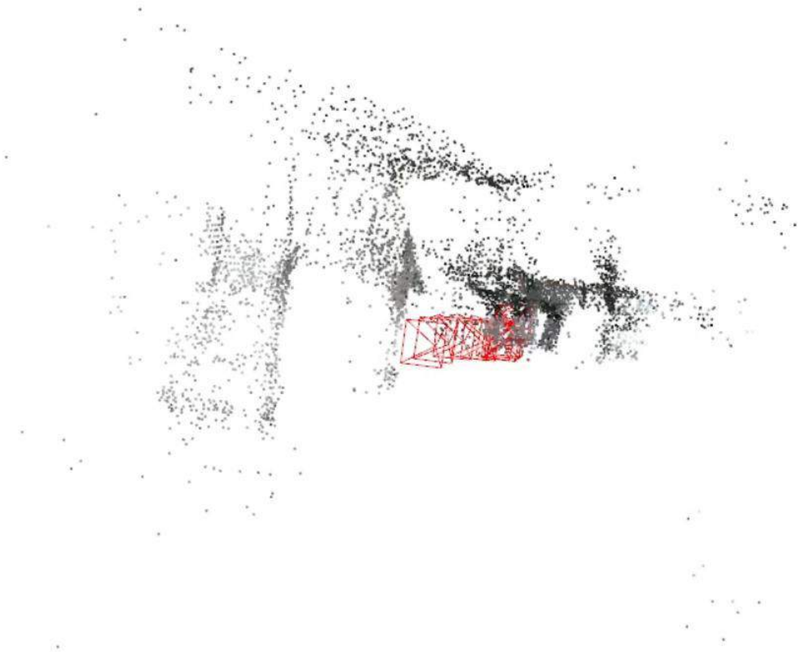
De asemenea, am folosit camera Intel RealSense pentru capturarea datelor 3D în timp real.

Am utilizat telefonul mobil, iPhone, echipat cu un senzor LiDAR 3D integrat.

Tipuri de camere folosite pentru achiziția de date

Matricea parametriilor intrinseci $K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$

Inferență



Vă Mulțumesc!