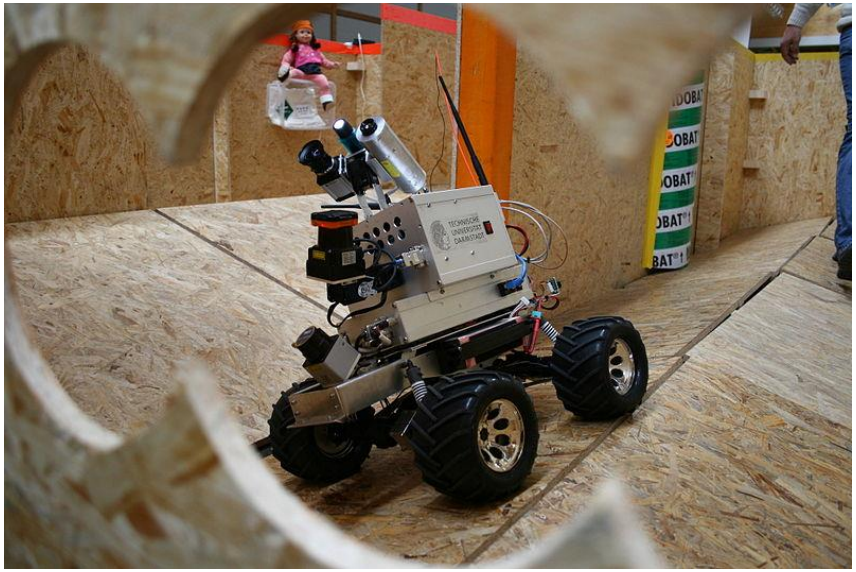
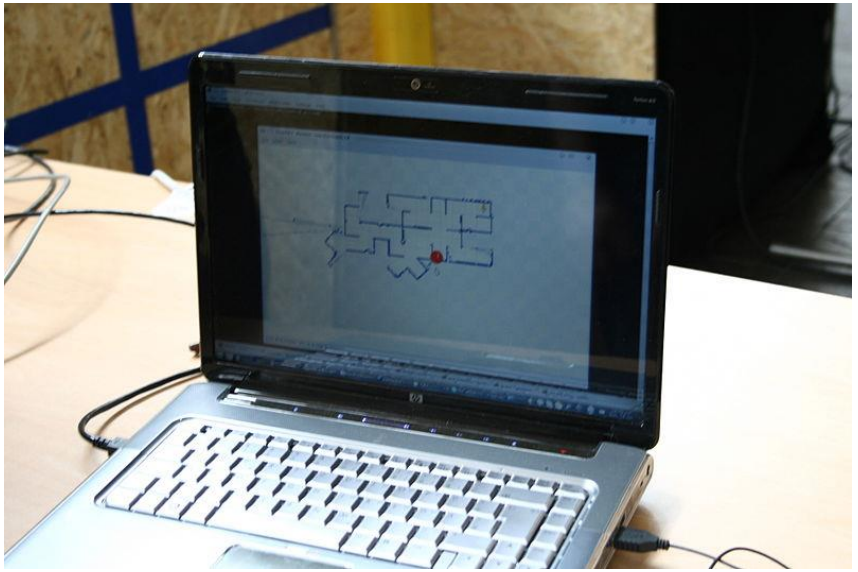
实时摄像机跟踪

章国锋
浙江大学CAD&CG实验室

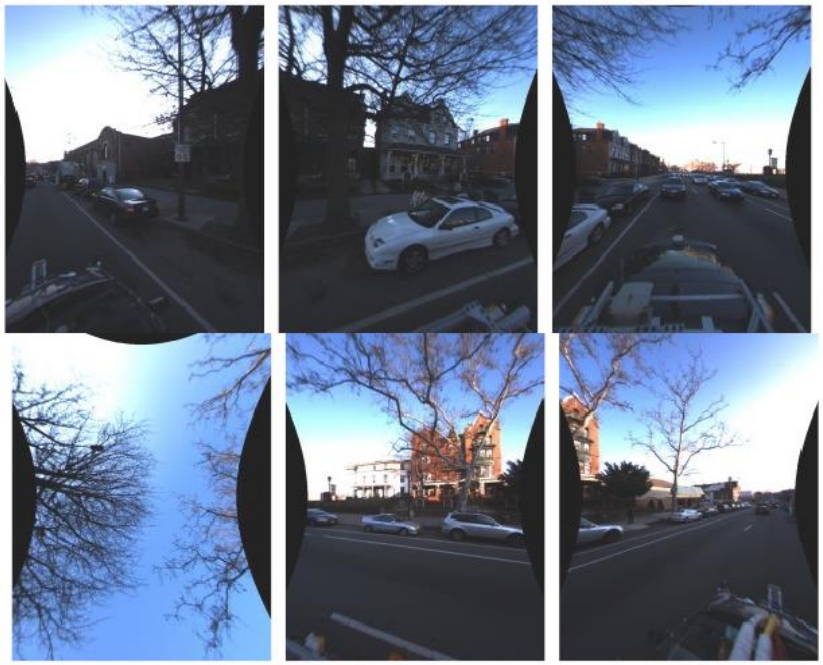
SLAM

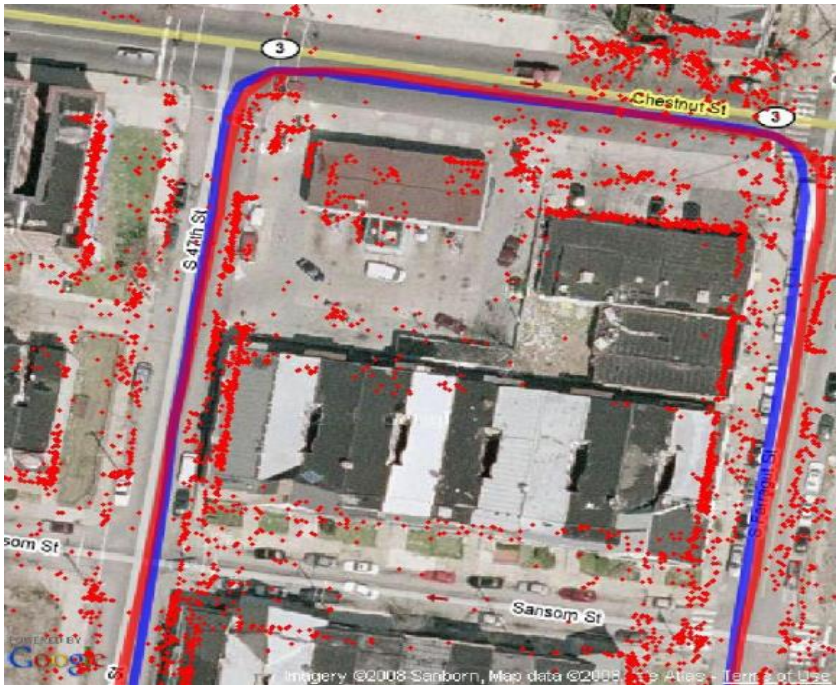
- Simultaneous Localization and Mapping
  - Estimate the environment structure and the camera trajectory online, under a highly nonlinear partial observation model.





SLAM for Visual Odometry





Jean-Philippe Tardif, Yanis Pavlidis, Kostas Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. IROS, pp. 2531-2538, 2008.

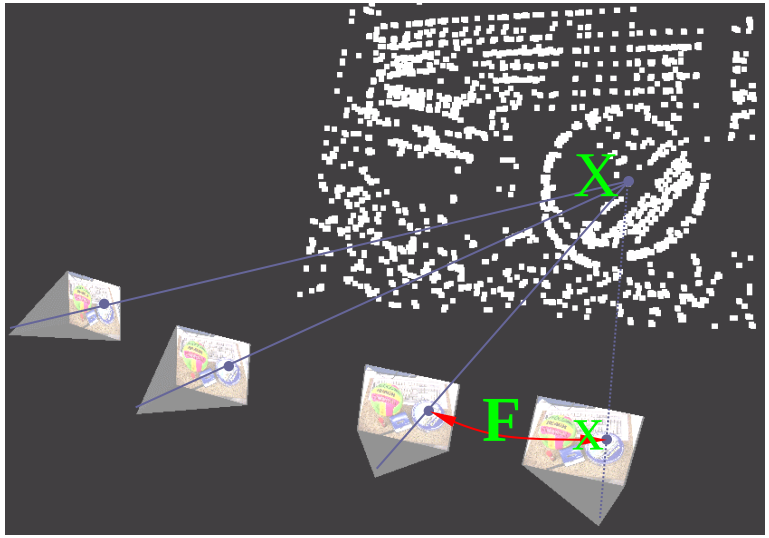
SLAM for Augmented Reality

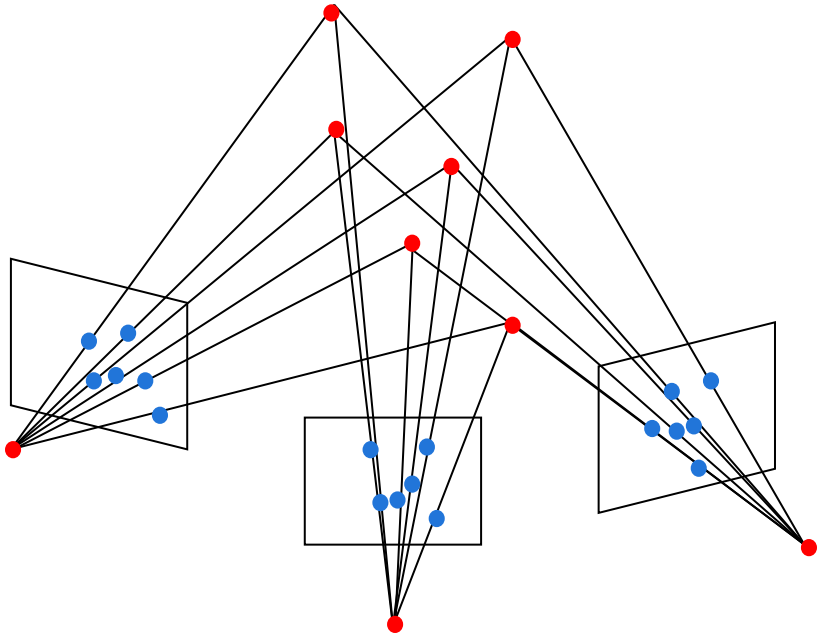
Multi-View Geometry

- Structure-from-Motion
  - Automatically recover the camera parameters and 3D structure from multiple images or video sequences.

Snavely et al. 2006

Multi-View Geometry





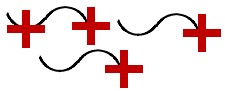
$$\mathbf{x}_{ij} = \pi(\mathbf{P}_i X_j)$$

Projection Function $\pi(x, y, z) = (x/z, y/z) \quad \mathbf{P}_i = \mathbf{K}_i[\mathbf{R}_i|\mathbf{T}_i]$

Structure-from-Motion

- Pipeline
  - Feature Tracking
    - Obtain a set of feature tracks

$$\mathcal{X} = \{\mathbf{x}_i|_{i=1,\ldots,m}\}$$



  - Structure from Motion
    - Solve the camera parameters and 3D points of tracks

$$\mathbf{x}_{ij} = \pi(\mathbf{P}_i X_j) \qquad \mathbf{P}_i = \mathbf{K}_i[\mathbf{R}_i|\mathbf{T}_i]$$

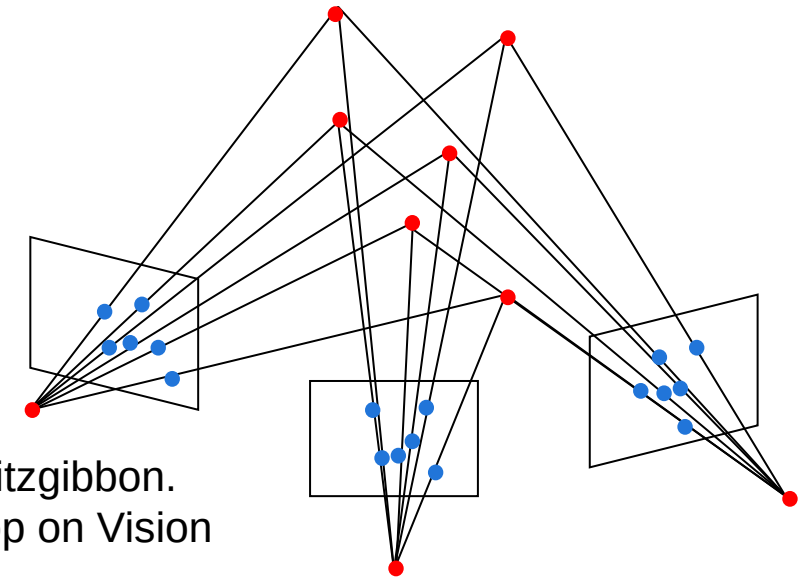
$$E(\mathbf{P}_1, \ldots, \mathbf{P}_m, X_1, \ldots, X_n) = \sum_{i=1}^{m}\sum_{j}^{n} w_{ij}||\pi(\mathbf{P}_i X_j) - \mathbf{x}_{ij}||^2$$

- SLAM: real-time SfM

Bundle Adjustment

- Definition
  - Refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter (camera pose and/or calibration) estimates.

$$\arg\min_{\mathbf{P}_k, \mathrm{X}_i} \sum_{k=1}^{m} \sum_{i=1}^{n} D\left(\mathrm{x}_{ki}, \mathbf{P}_k\left(\mathrm{X}_i\right)\right)^2$$



B. Triggs, P. F. McLauchlan, R. I. Hartley, and A.W. Fitzgibbon. Bundle adjustment - a modern synthesis. In Workshop on Vision Algorithms, pages 298–372, 1999.

Real-Time Structure-from-Motion

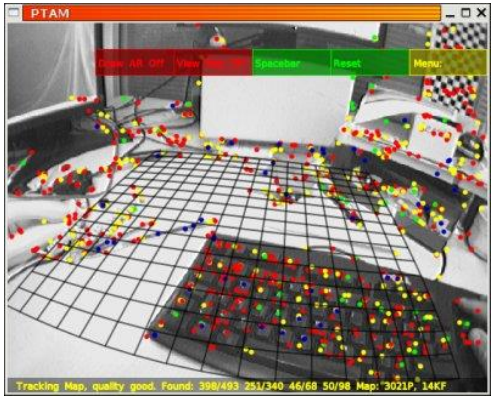
- With a set of 2D-3D correspondences, quickly compute the camera pose by solving

$$\mathbf{P}_i = \arg\min_{\mathbf{P}_i} \sum_j ||\pi(\mathbf{P}_i X_j) - \mathbf{x}_{ij}||^2$$

  - The computational complexity is low and can be performed in real time.
- Parallel Tracking and Mapping
  - Foreground thread: track features and compute the camera pose with the estimated 3D points
  - Background thread: BA for map refinement

Related Work

- Filter-based SLAM
  - Davison et al.2007, Eade and Drummond 2006
- Keyframe-based SLAM
  - Klein and Murray 2007,2008, Castle et al.2008
- SLAM in Dynamic Environments
  - Shimamura et al.2011, Zou and Tan,2013
  - Tan et al. 2013



Extended Kalman Filter

- State at time k, model as multivariate Gaussian

$$x_k \sim N(\hat{x}_k, P_k)$$

mean covariance

- State transition model

$$x_k = f(x_{k-1}) + w_k$$

$$w_k \sim N(0, Q_k)$$ Process noise

- State observation model

$$z_k = h(x_k) + v_k$$

$$v_k \sim N(0, R_k)$$ Observation noise

Extended Kalman Filter

- Predict

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1})$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

$$F_k = \partial f / \partial x \big|_{\hat{x}_{k-1|k-1}}$$

- Update

$$S_k = H_k P_{k|k-1} H_k^T + R_k$$ Innovation covariance

$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$

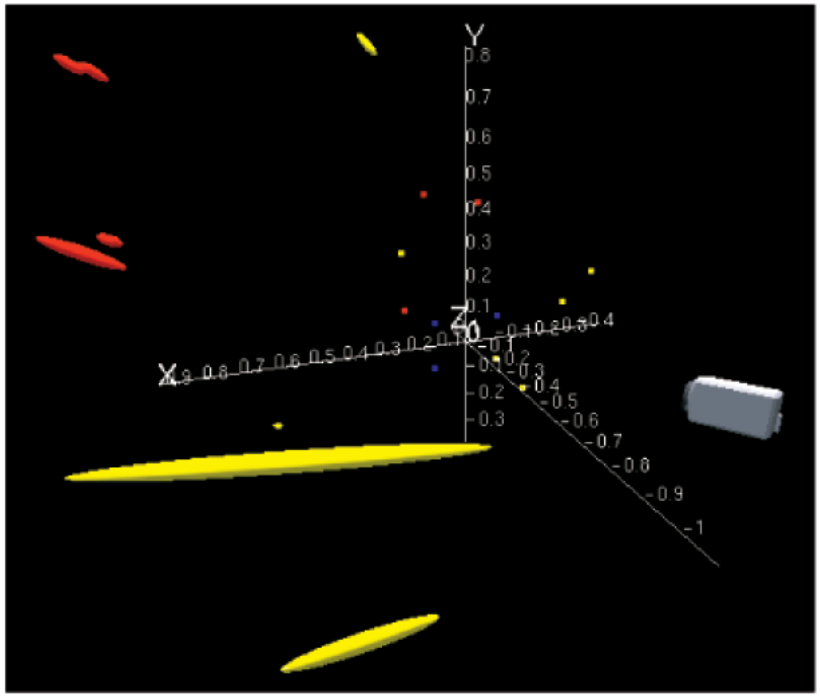
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (z_k - h(\hat{x}_{k|k-1}))$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}$$

$$H_k = \partial h / \partial x \big|_{\hat{x}_{k|k-1}}$$

MonoSLAM

- Map representation



$$x = \begin{pmatrix} C \\ X \end{pmatrix} = \begin{pmatrix} C \\ X_1 \\ X_2 \\ \vdots \end{pmatrix}$$

camera state

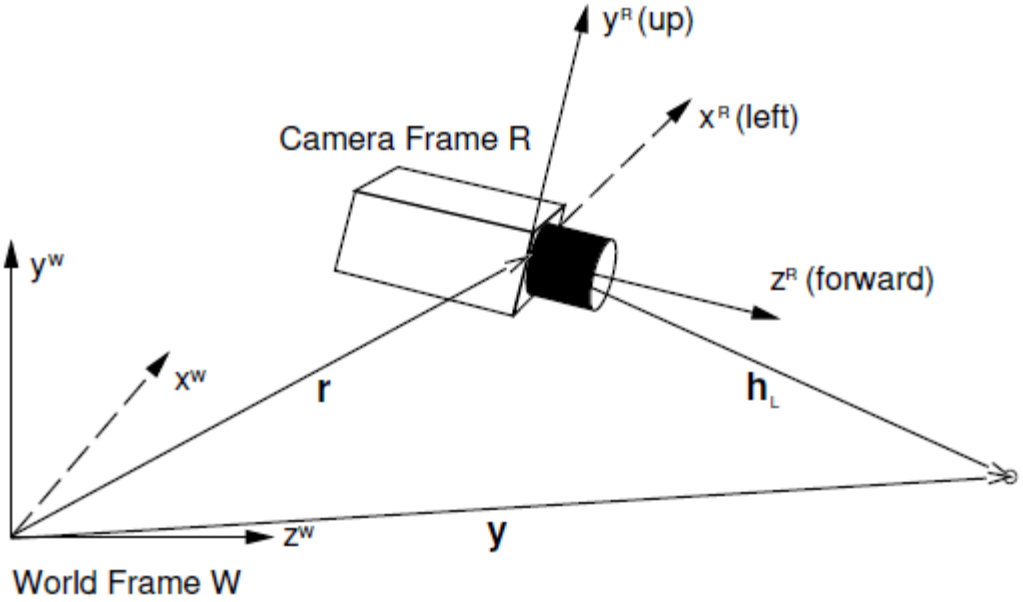
point state

$$P = \begin{pmatrix} P_{CC} & P_{CX_1} & P_{CX_2} & \cdots \\ P_{X_1C} & P_{X_1X_1} & P_{X_1X_2} & \cdots \\ P_{X_2C} & P_{X_2X_1} & P_{X_2X_2} & \cdots \\ \vdots & \vdots & \vdots & \end{pmatrix}$$

A. J. Davison, N. D. Molton, I. Reid, and O. Stasse. MonoSLAM: Real-time single camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 29(6):1052{1067, 2007.

MonoSLAM

- Camera state



$$C_k = \begin{pmatrix} p_k \\ q_k \\ v_k \\ \omega_k \end{pmatrix}$$

camera position

orientation quaternion

linear velocity

angular velocity

MonoSLAM

- Predict

$$w_k = \begin{pmatrix} a_k \\ \alpha_k \end{pmatrix}$$ linear acceleration

angular acceleration

$$w_k \sim N\left(0, \mathrm{diag}(Q_a, Q_\alpha)\right)$$

$$C_k = \begin{pmatrix} p_k \\ q_k \\ v_k \\ \omega_k \end{pmatrix} = \begin{pmatrix} p_{k-1} + (v_{k-1|} + a_k)\Delta t \\ q((\omega_{k-1} + \alpha_k)\Delta t) \otimes q_{k-1} \\ v_{k-1|} + a_k \\ \omega_{k-1} + \alpha_k \end{pmatrix}$$

$$X_k = X_{k-1}$$

MonoSLAM

- Predicted features position

$$z_i = \pi(X_i, C) + v_i$$

$$v_i \sim N(0, R)$$

- Innovation covariance
  - Elliptical feature search region

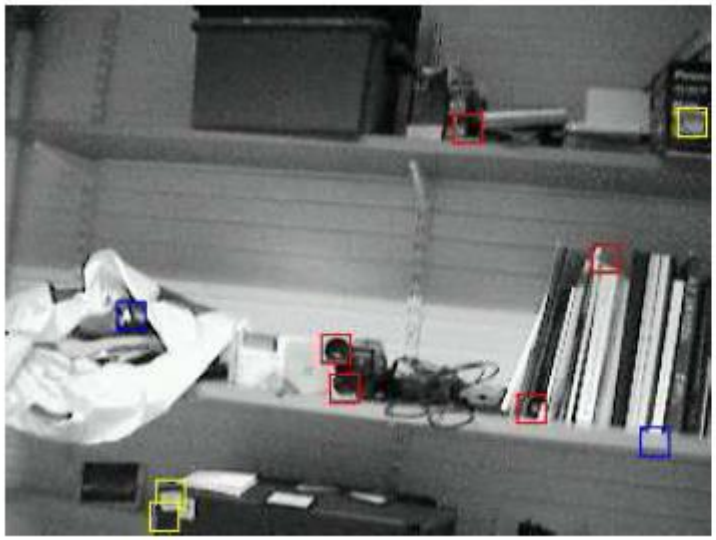
$$S_i = J_C P_{CC} J_C^T + J_C P_{CX_i} J_{X_i}^T + J_{X_i} P_{X_i C} J_C^T + J_{X_i} P_{X_i X_i} J_{X_i}^T + R$$

$$J_C = \frac{\partial z_i}{\partial C}$$

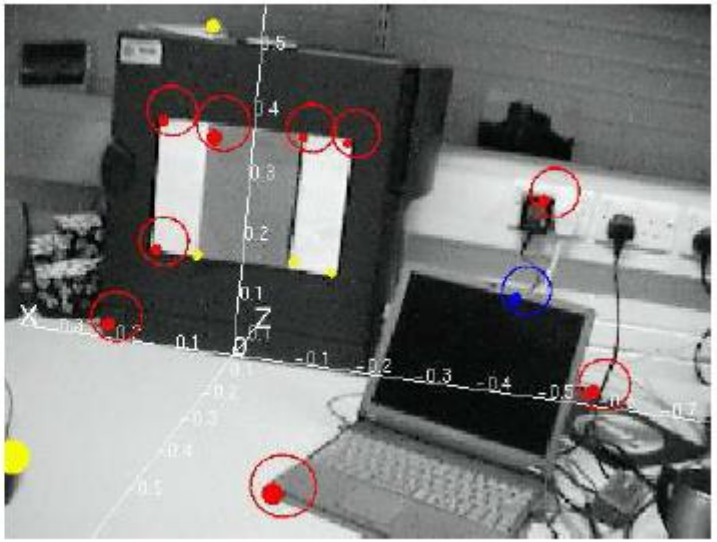
$$J_{X_i} = \frac{\partial z_i}{\partial X_i}$$

MonoSLAM

- Active search



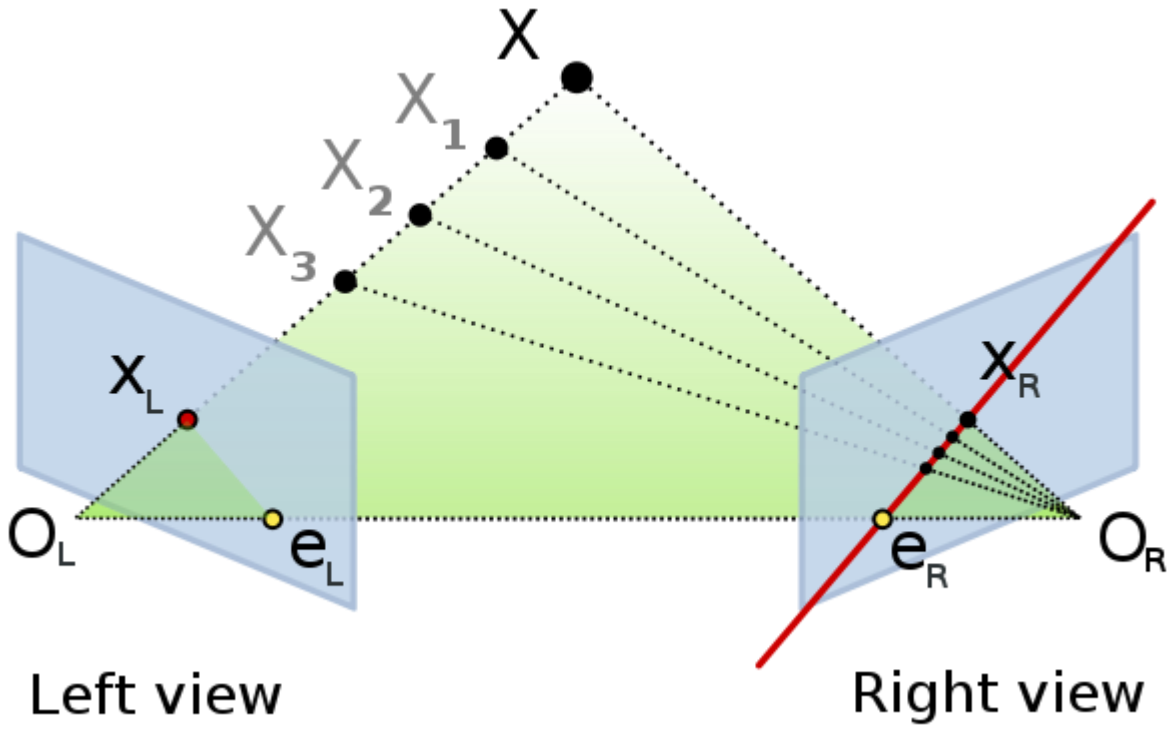
Shi and Tomasi Feature



Elliptical search region

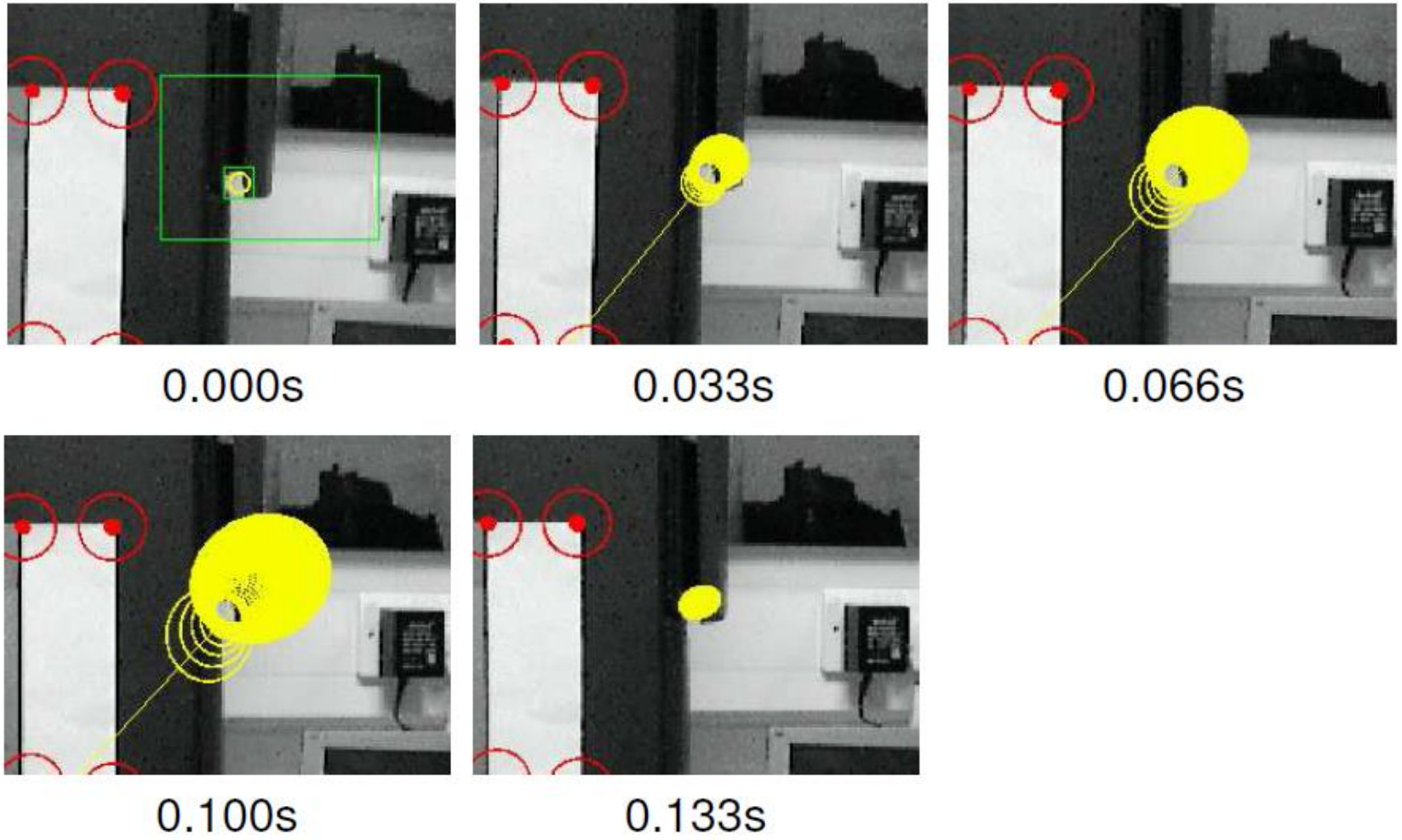
MonoSLAM

- Initialization
  - 2D feature → 3D ray
  - Uniformly sample 100 depths in [0.5m, 5.0m]



MonoSLAM

- Initialization

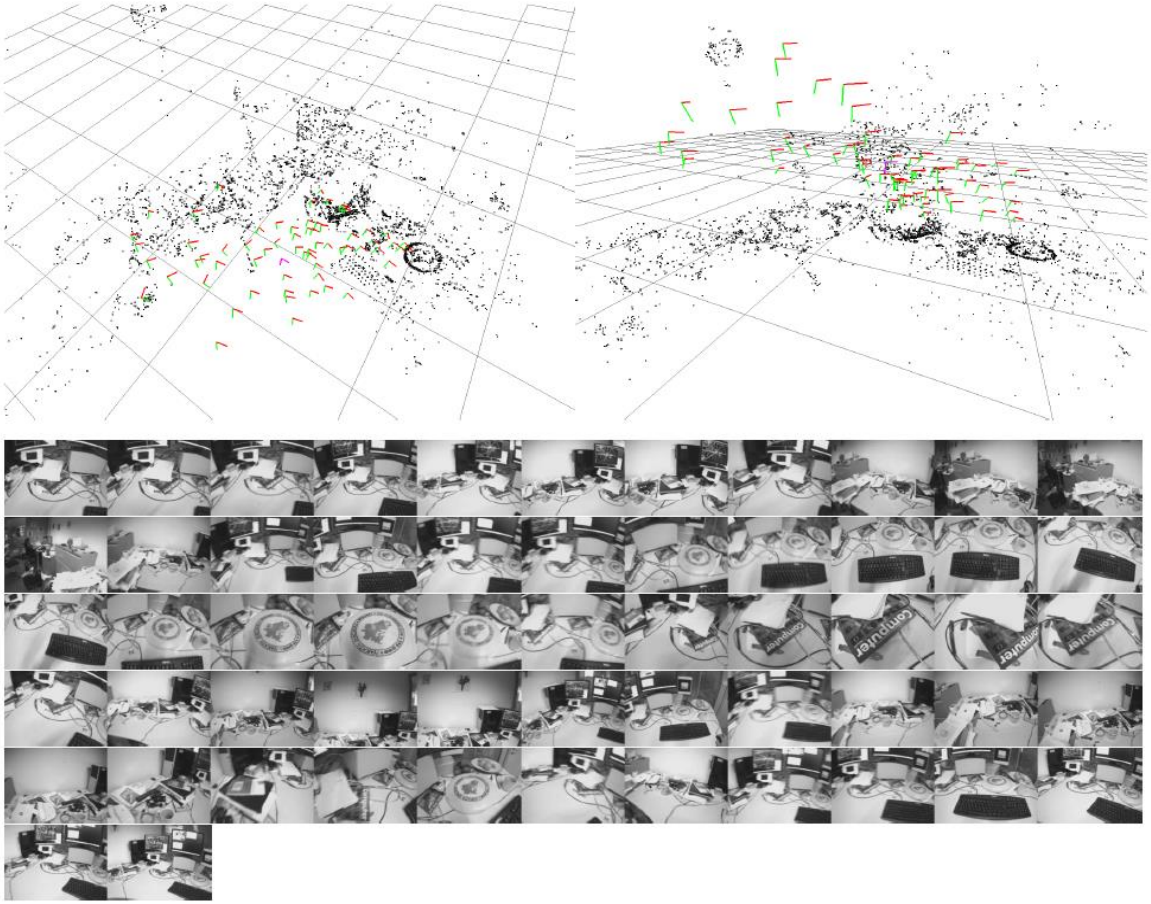


MonoSLAM

- Complexity
  - $O(N^3)$ per frame
- Scalability
  - hundreds of points

PTAM

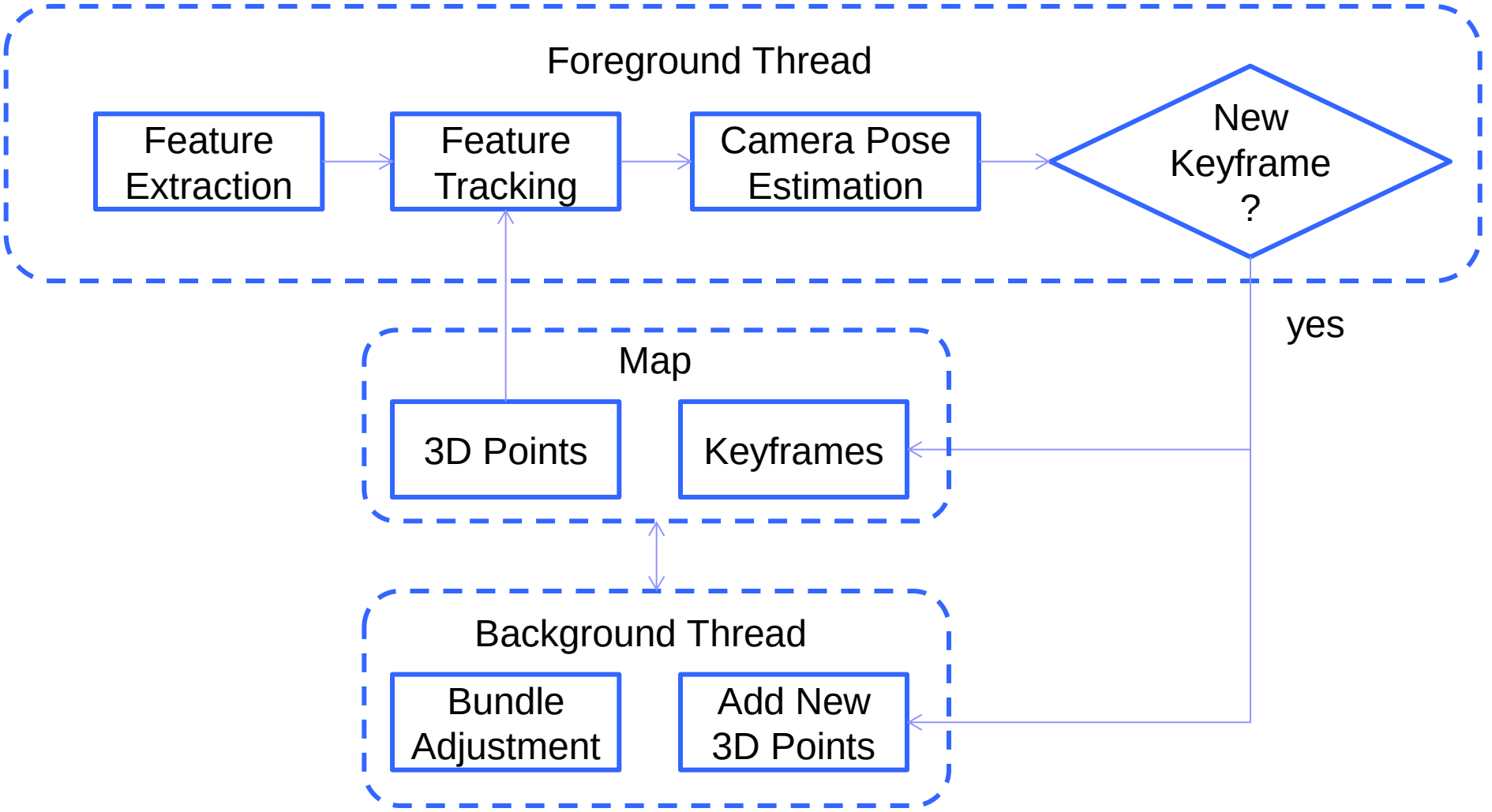
- Map representation



G. Klein and D. W. Murray. Parallel tracking and mapping on a camera phone. In Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2009.

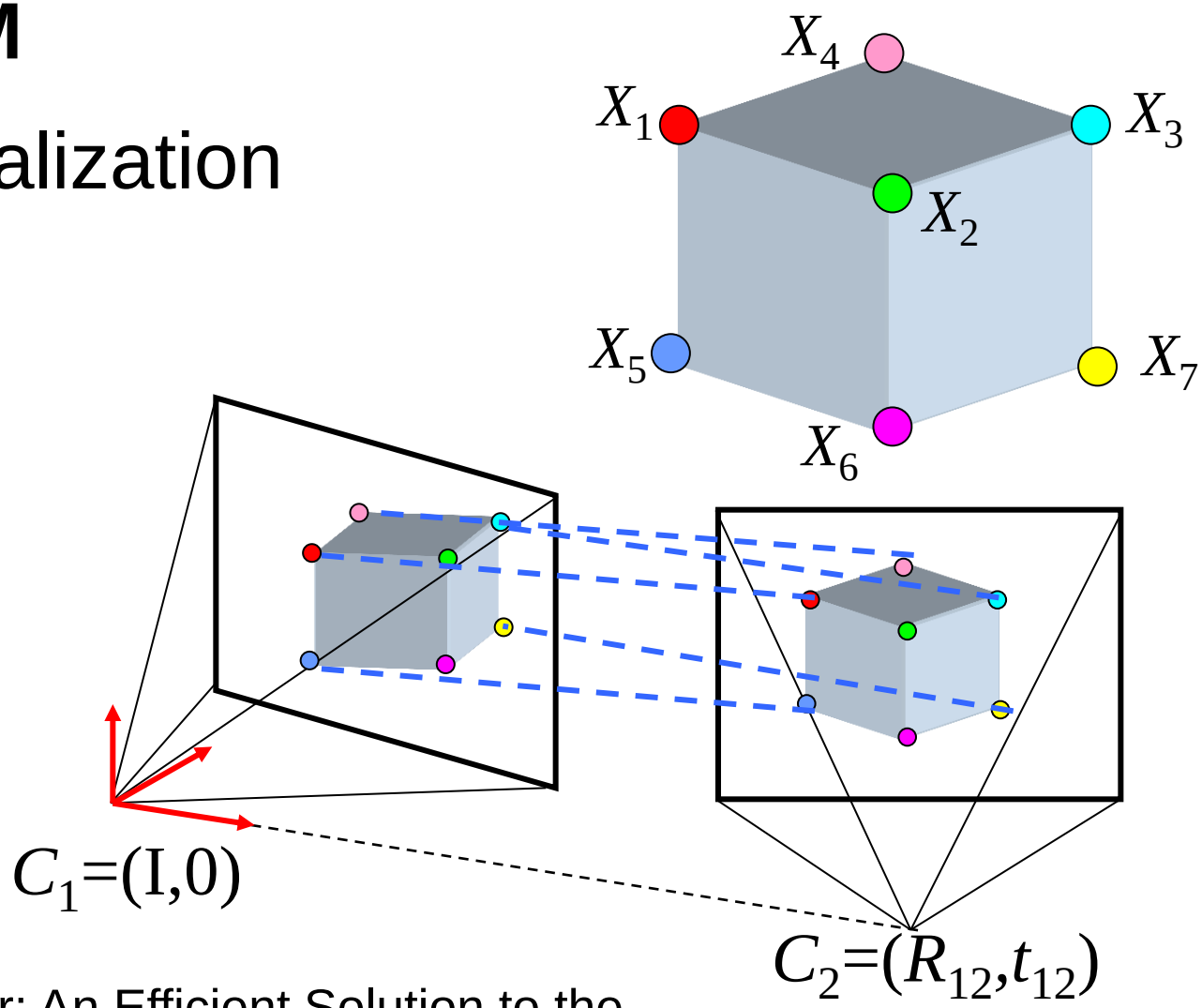
PTAM

- Overview



PTAM

- Initialization

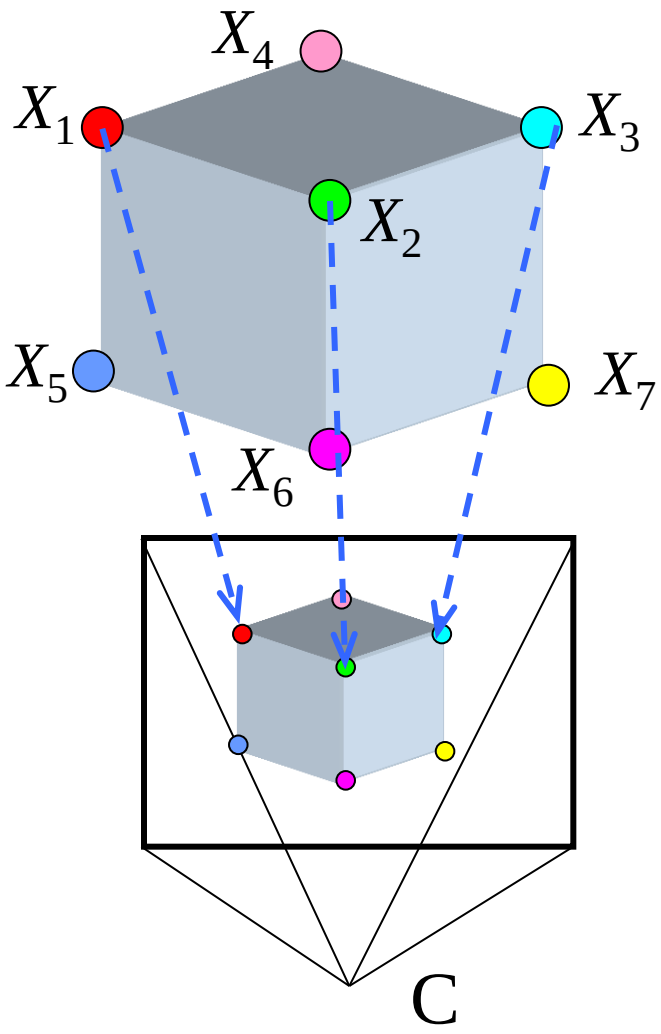


David Nistér: An Efficient Solution to the Five-Point Relative Pose Problem (PAMI), 2004

PTAM

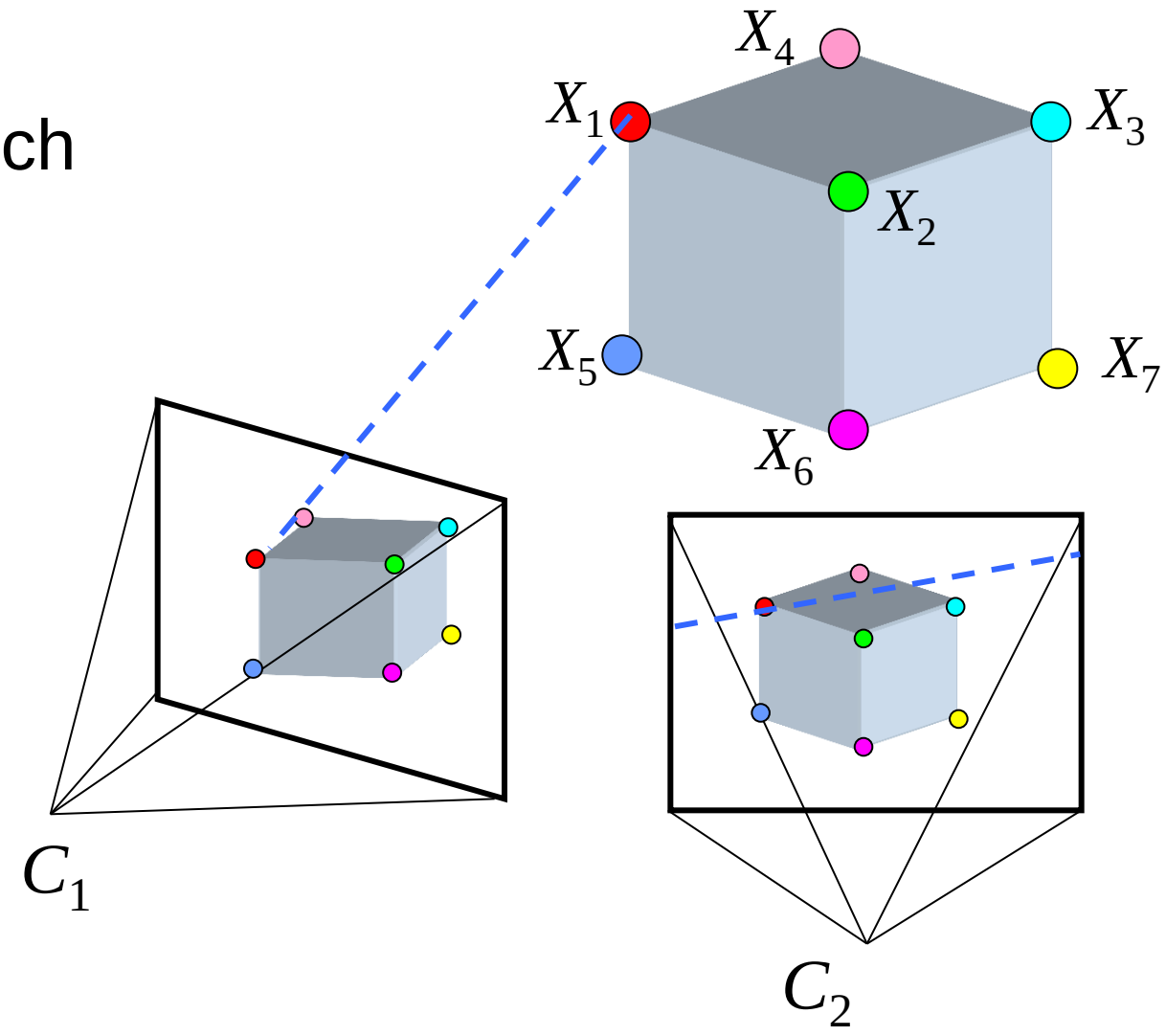
- Camera tracking

$$\arg\min_{C} \sum_{i} \left\| \pi(X_i, C) - x_i \right\|^2$$



PTAM

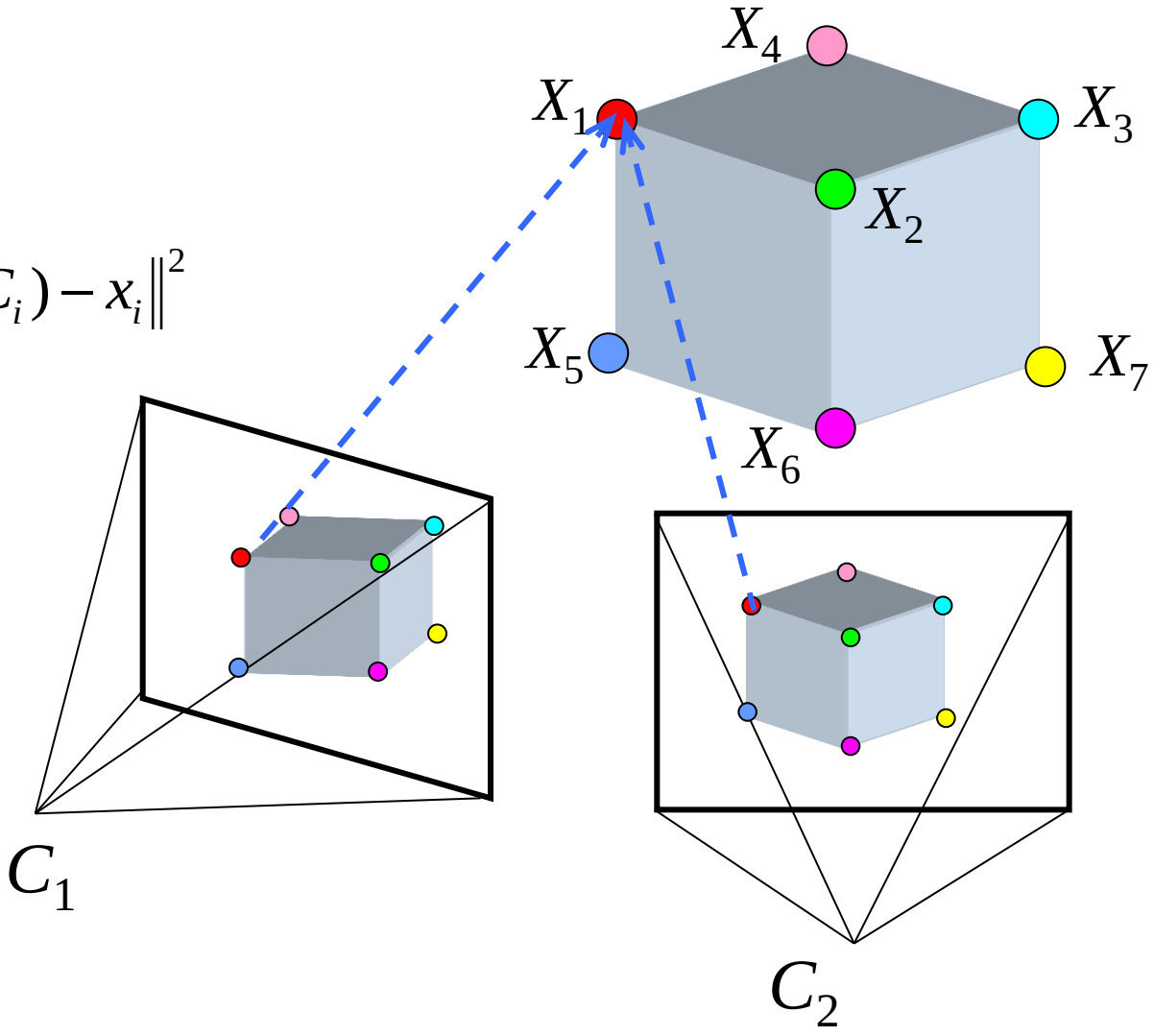
- Epipolar search



PTAM

- Triangulation

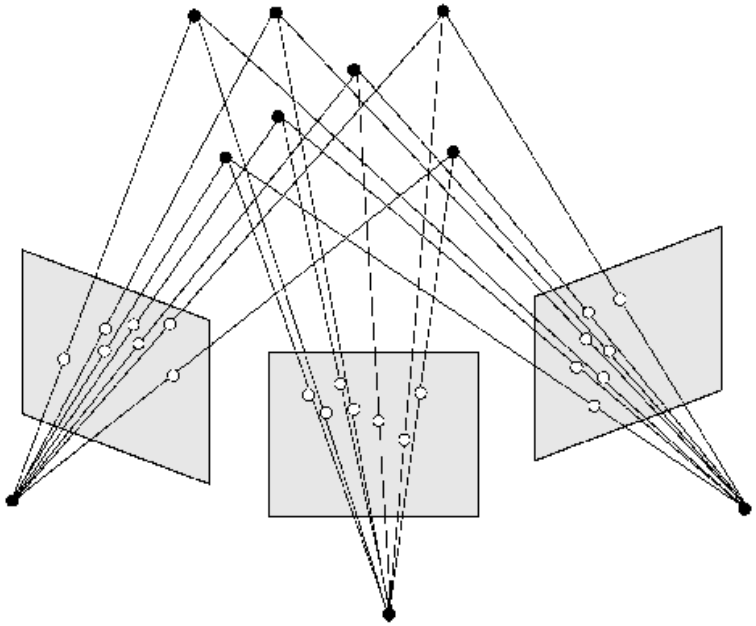
$$\arg\min_{X} \sum_{i} \left\| \pi(X, C_i) - x_i \right\|^2$$



Bundle Adjustment

- Jointly optimize all cameras and points

$$\underset{C_1,\ldots C_{N_c},X_1,\ldots,X_{N_p}}{\arg\min} \sum \left\| \pi(X_i, C_j) - x_{ij} \right\|^2$$



Triggs, B., Mclauchlan, P., Hartley, R., and Fitzgibbon, A. 1999. Bundle adjustment—a modern synthesis. In Proceedings of the International Workshop on Vision Algorithms: Theory and Practice. 298–372.

Nonlinear Least Squares

- Gaussian Newton

$$x^* = \arg\min_x \left\| \varepsilon(x) \right\|^2$$

$$\varepsilon(x^*) = \varepsilon(\hat{x} + \delta_x) \approx \varepsilon(\hat{x}) + J\delta_x$$

$$J = \partial\varepsilon / \partial x \Big|_{x=\hat{x}}$$ Jacobian matrix

$$J^T J \delta_x = -J^T \varepsilon(\hat{x})$$

first order approximation to Hessian

- Levenberg-Marquardt

$$(J^T J + \mu I)\delta x = -J^T \varepsilon(\hat{x})$$

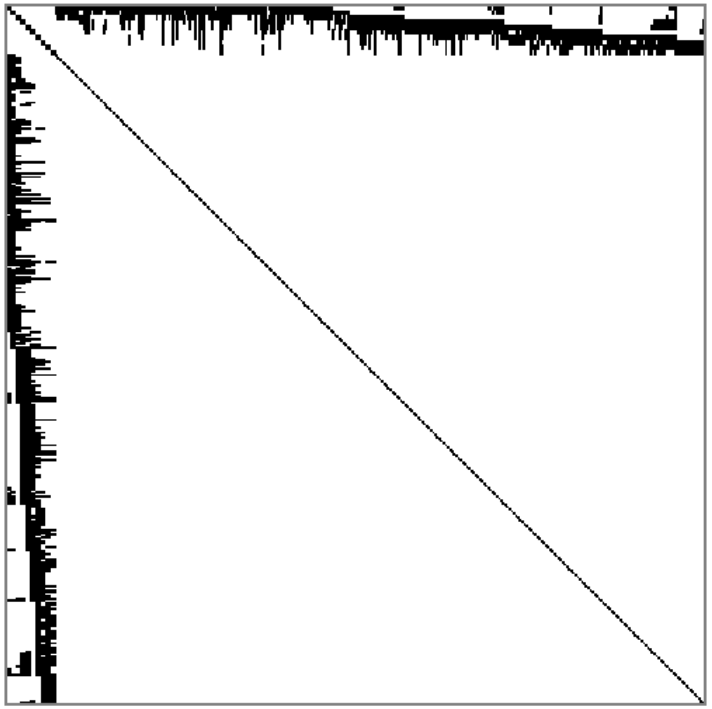
Sparse Bundle Adjustment

$$\underset{C_1,\ldots C_{N_c},X_1,\ldots,X_{N_p}}{\arg\min} \sum \left\| \pi(X_i, C_j) - x_{ij} \right\|^2$$

1 Point

1 Camera

Sparsity patten of Hessian



Manolis I. A. Lourakis, Antonis A. Argyros: SBA: A software package for generic sparse bundle adjustment. ACM Trans. Math. Softw. 36(1) (2009)

Sparse Bundle Adjustment

- An simple example
  - 4 points
  - 3 cameras
  - all points are visible in all cameras

Sparse Bundle Adjustment

$$J = \begin{pmatrix} A_{11} & 0 & 0 & B_{11} & 0 & 0 & 0 \\ 0 & A_{12} & 0 & B_{12} & 0 & 0 & 0 \\ 0 & 0 & A_{13} & B_{13} & 0 & 0 & 0 \\ A_{21} & 0 & 0 & 0 & B_{21} & 0 & 0 \\ 0 & A_{22} & 0 & 0 & B_{22} & 0 & 0 \\ 0 & 0 & A_{23} & 0 & B_{23} & 0 & 0 \\ A_{31} & 0 & 0 & 0 & 0 & B_{31} & 0 \\ 0 & A_{32} & 0 & 0 & 0 & B_{32} & 0 \\ 0 & 0 & A_{33} & 0 & 0 & B_{33} & 0 \\ A_{41} & 0 & 0 & 0 & 0 & 0 & B_{41} \\ 0 & A_{42} & 0 & 0 & 0 & 0 & B_{42} \\ 0 & 0 & A_{43} & 0 & 0 & 0 & B_{43} \end{pmatrix}, \varepsilon = \begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{12} \\ \varepsilon_{13} \\ \varepsilon_{21} \\ \varepsilon_{22} \\ \varepsilon_{23} \\ \varepsilon_{31} \\ \varepsilon_{32} \\ \varepsilon_{33} \\ \varepsilon_{41} \\ \varepsilon_{42} \\ \varepsilon_{43} \end{pmatrix}$$

Sparse Bundle Adjustment

$$J^T J \delta_x = -J^T \varepsilon$$

$$J^T J = \begin{pmatrix} U & W \\ W^T & V \end{pmatrix} = \begin{pmatrix} U_1 & 0 & 0 & W_{11} & W_{21} & W_{31} & W_{41} \\ 0 & U_2 & 0 & W_{12} & W_{22} & W_{32} & W_{42} \\ 0 & 0 & U_3 & W_{13} & W_{23} & W_{33} & W_{43} \\ W_{11}^T & W_{12}^T & W_{13}^T & V_1 & 0 & 0 & 0 \\ W_{21}^T & W_{22}^T & W_{23}^T & 0 & V_2 & 0 & 0 \\ W_{31}^T & W_{32}^T & W_{33}^T & 0 & 0 & V_3 & 0 \\ W_{41}^T & W_{42}^T & W_{43}^T & 0 & 0 & 0 & V_4 \end{pmatrix}$$

$$U_j = \sum_{i=1}^{4} A_{ij}^T A_{ij}, V_i = \sum_{j=1}^{3} B_{ij}^T B_{ij}, W_{ij} = A_{ij}^T B_{ij}$$

Sparse Bundle Adjustment

$$J^T J \delta_x = -J^T \varepsilon$$

$$\delta_x = \begin{pmatrix} \delta_C \\ \delta_X \end{pmatrix} = \begin{pmatrix} \delta_{C_1}^T & \delta_{C_2}^T & \delta_{C_3}^T & \delta_{X_1}^T & \delta_{X_2}^T & \delta_{X_3}^T & \delta_{X_4}^T \end{pmatrix}^T$$

Sparse Bundle Adjustment

$$J^T J \delta_x = -J^T \varepsilon$$

$$J^T \varepsilon = \begin{pmatrix} \varepsilon_C \\ \varepsilon_X \end{pmatrix} = \begin{pmatrix} \varepsilon_{C_1}^T & \varepsilon_{C_2}^T & \varepsilon_{C_3}^T & \varepsilon_{X_1}^T & \varepsilon_{X_2}^T & \varepsilon_{X_3}^T & \varepsilon_{X_4}^T \end{pmatrix}^T$$

$$\varepsilon_{C_j} = \sum_{i=1}^{4} A_{ij}^T \varepsilon_{ij}$$

$$\varepsilon_{X_i} = \sum_{j=1}^{3} B_{ij}^T \varepsilon_{ij}$$

Sparse Bundle Adjustment

$$J^T J \delta_x = -J^T \varepsilon$$

$$\begin{pmatrix} U & W \\ W^T & V \end{pmatrix} \begin{pmatrix} \delta_C \\ \delta_X \end{pmatrix} = - \begin{pmatrix} \varepsilon_C \\ \varepsilon_X \end{pmatrix}$$

$$\begin{pmatrix} U - WV^{-1}W^T & 0 \\ W^T & V \end{pmatrix} \begin{pmatrix} \delta_C \\ \delta_X \end{pmatrix} = - \begin{pmatrix} \varepsilon_C - WV^{-1}\varepsilon_X \\ \varepsilon_X \end{pmatrix}$$

$S = U - WV^{-1}W^T$ Schur Complement

$S\delta_C = -(\varepsilon_C - WV^{-1}\varepsilon_X)$ Compute cameras first (# cameras << # points)

$V\delta_X = -\varepsilon_X - W^T\delta_C$ back substitution for points

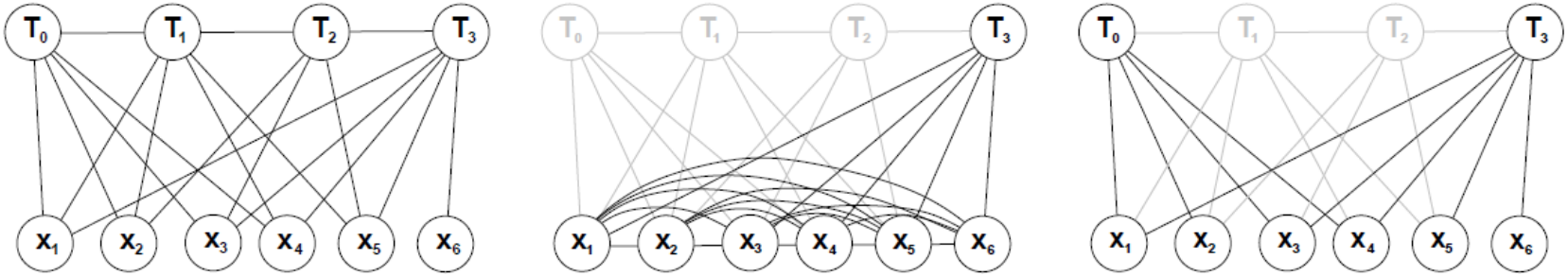
Sparse Bundle Adjustment

- In general, NOT all points are visible in all cameras

$$U_j = \sum_{i=1}^{4} A_{ij}^T A_{ij}, V_i = \sum_{j=1}^{3} B_{ij}^T B_{ij}, W_{ij} = A_{ij}^T B_{ij}$$

  - $A_{ij} = B_{ij} = 0$ if *i*-th points is invisible (or not matched) in *j*-th camera
  - More sparse structure, more speed-up

Filter vs BA

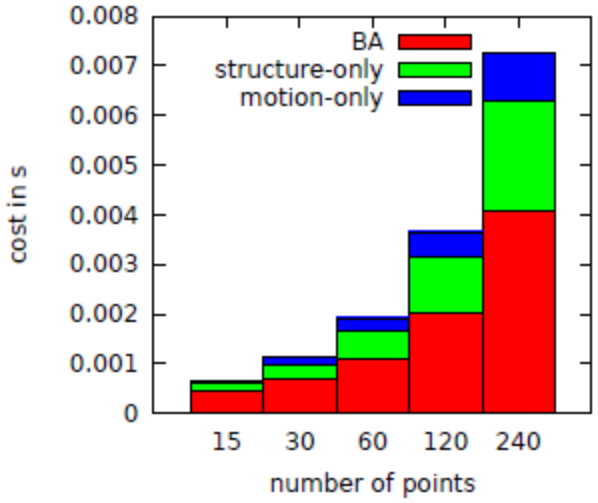


(a) Markov Random Field (b) Filter (c) Keyframe BA

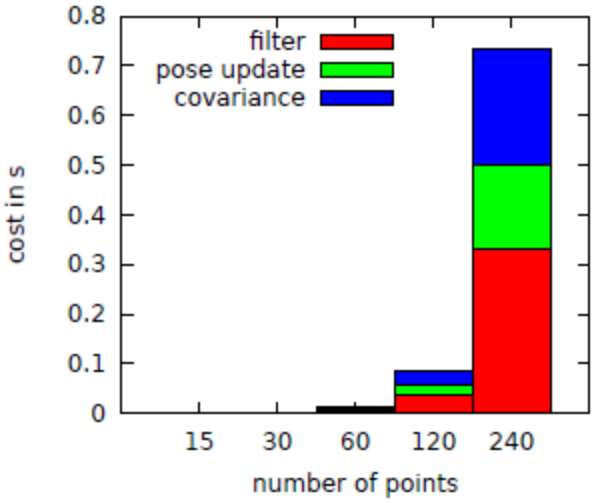
H. Strasdat, J. Montiel, and A. J. Davison. Visual slam: Why filter? Image and Vision Computing, 30:65.77, 2012.

Filter vs BA

- Computational cost



(a) BA-SLAM

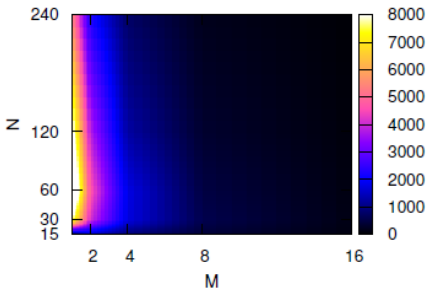


(b) Filter-SLAM

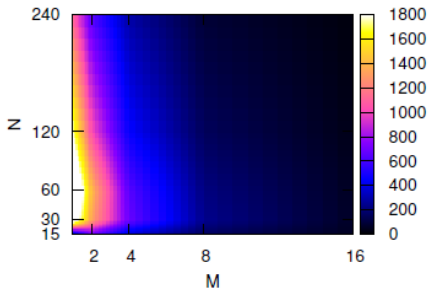
Filter vs BA

- Accuracy/cost in bits per second (bps)

Setting (i):

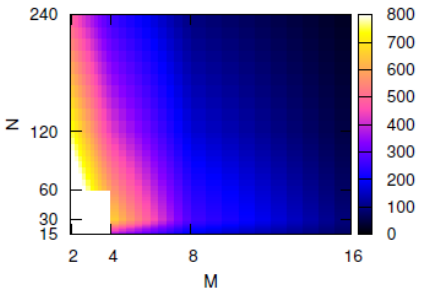


(a) Stereo BA

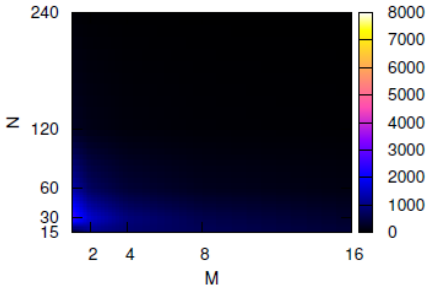


(c) Mono BA

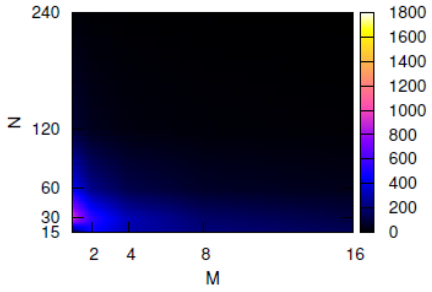
Setting (ii):



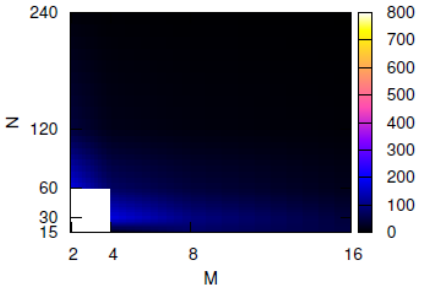
(e) Monocular BA



(b) Stereo Filter



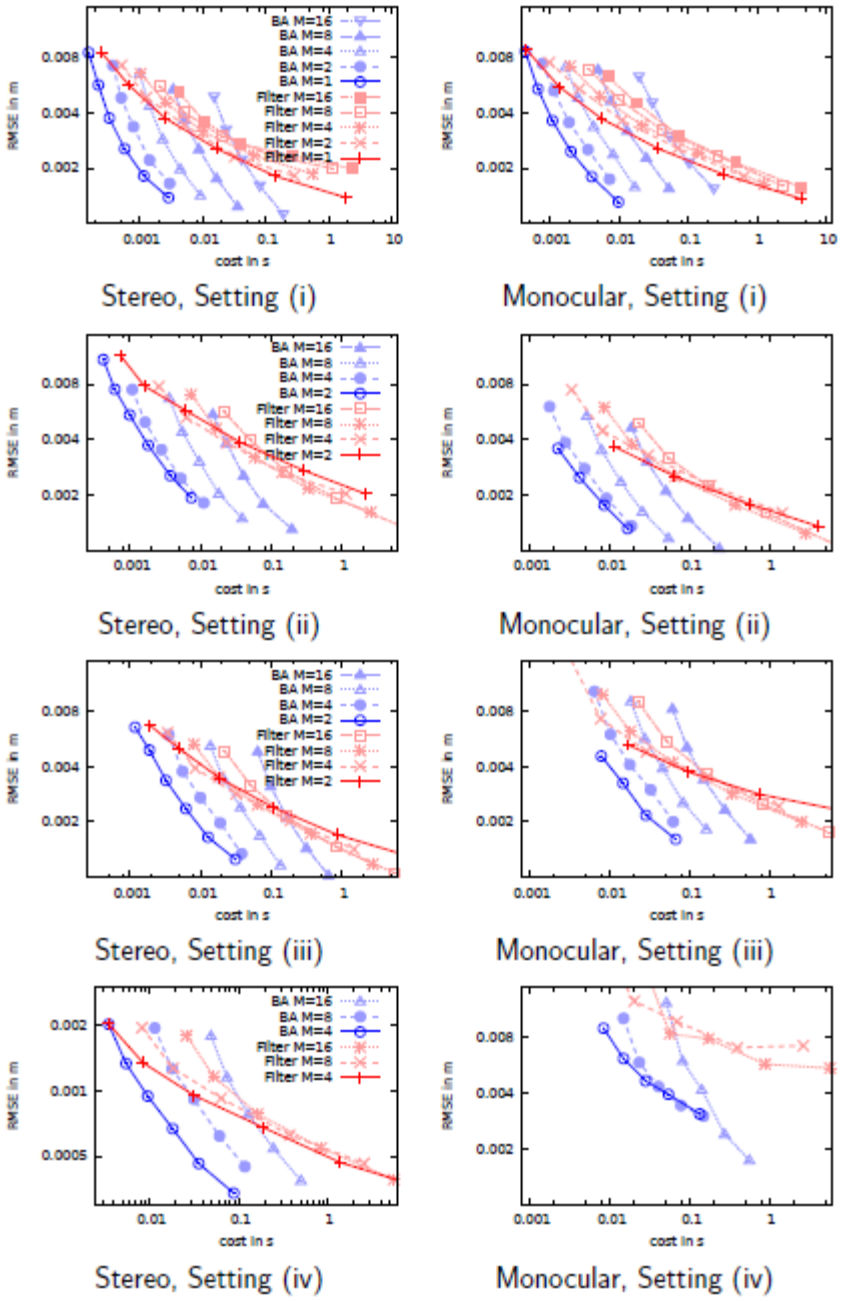
(d) Monocular Filter



(f) Monocular Filter

Filter vs BA

- Error versus cost on a logarithmic scale.



Filter vs BA

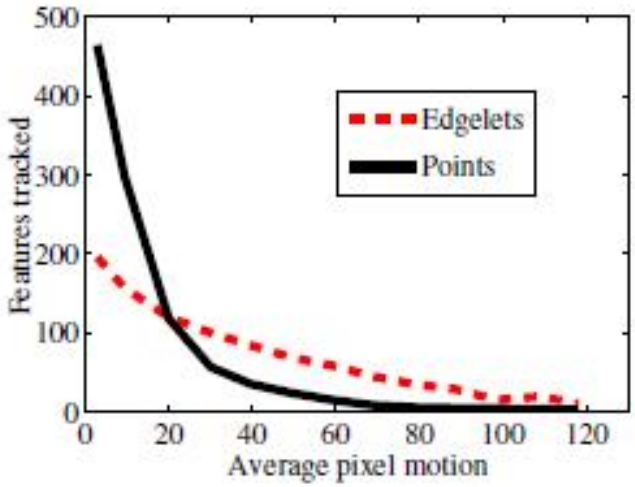
- Conclusion
  - keyframe bundle adjustment outperforms filtering, since it gives the most accuracy per unit of computing time

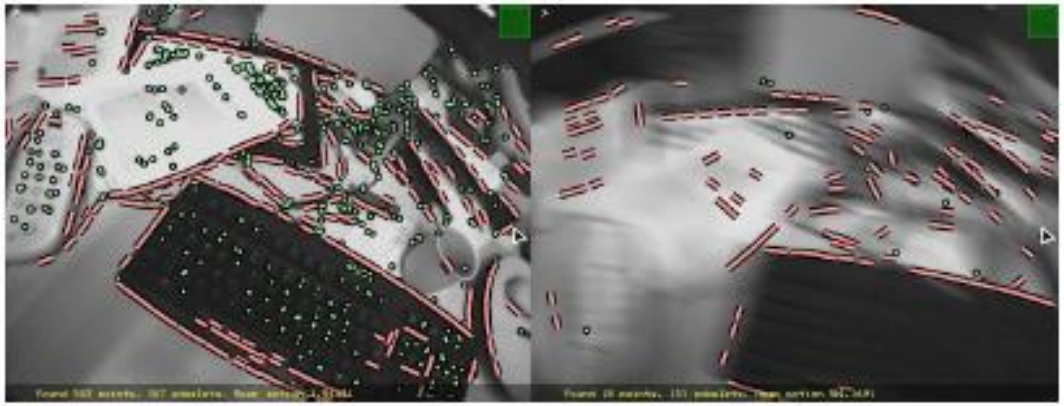
H. Strasdat, J. Montiel, and A. J. Davison. Visual slam: Why filter? Image and Vision Computing, 30:65.77, 2012.

Challenge of BA based SLAM

- Agility
  - blur
  - strong dependency on feature
- Scalability
  - thousands of features
- Error accumulation
  - loop closure

Improving the Agility

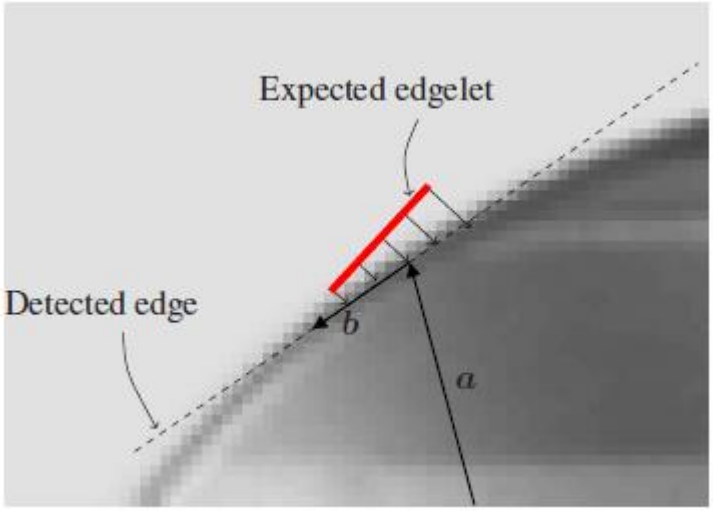




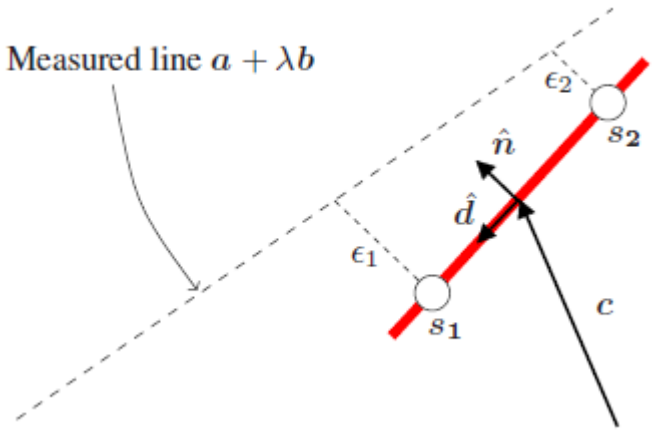
Georg Klein and David Murray. Improving the Agility of Keyframe-Based SLAM (ECCV) 2008.

Improving the Agility

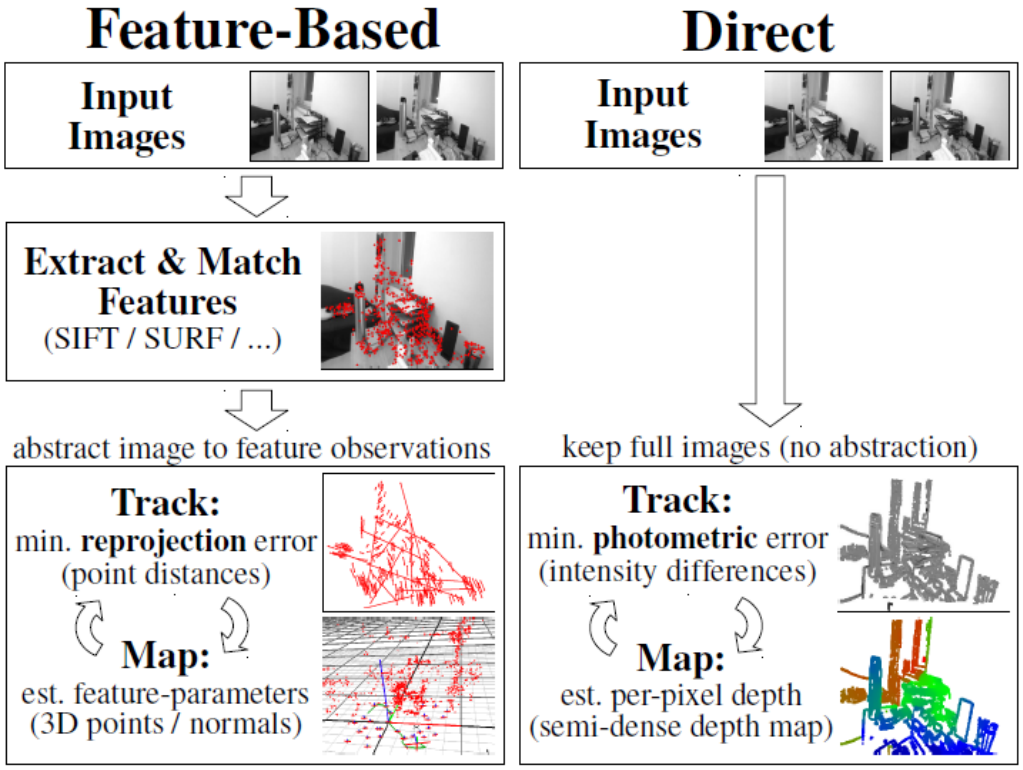
- Edgelet measurement



- Objective function



Direct Tracking



Thomas Schops, Jakob Engel, Daniel Cremers: Semi-dense visual odometry for AR on a smartphone. ISMAR 2014: 145-150

Direct Tracking

- Goal
  - Estimate the camera motion ξ by aligning intensity images I_1 and I_2 with depth map Z_1 of I_1
- Assumption

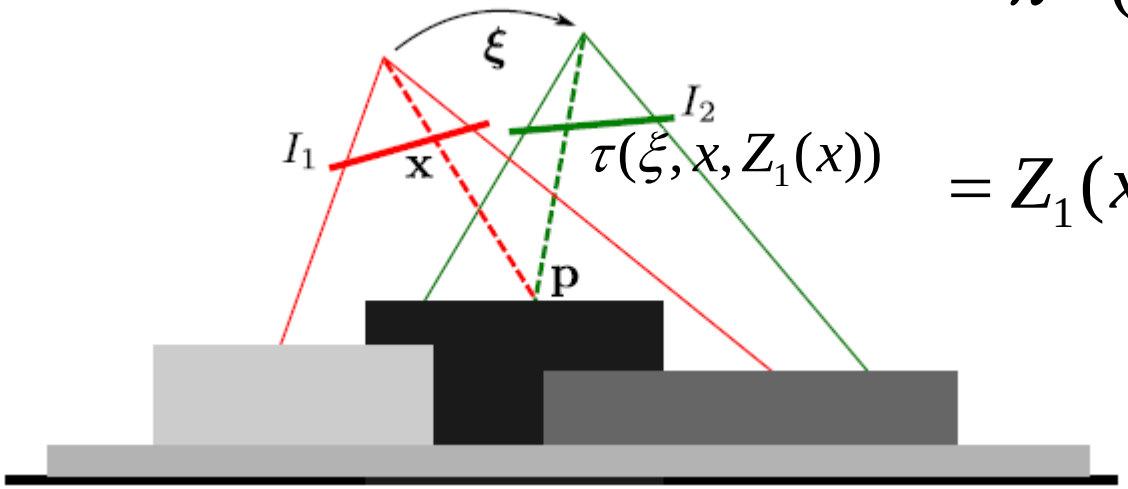
$$I_1(x) = I_2(\tau(\xi, x, Z_1(x)))$$

warping function: maps a pixel from I_1 to I_2

Direct Tracking

- Warping function

$$p = \pi^{-1}(x, Z_1(x))$$
$$= \pi^{-1}((u,v)^T, Z_1(x))$$
$$= Z_1(x)\left(\frac{u - c_x}{f_x}, \frac{v - c_y}{f_y}\right)^T$$

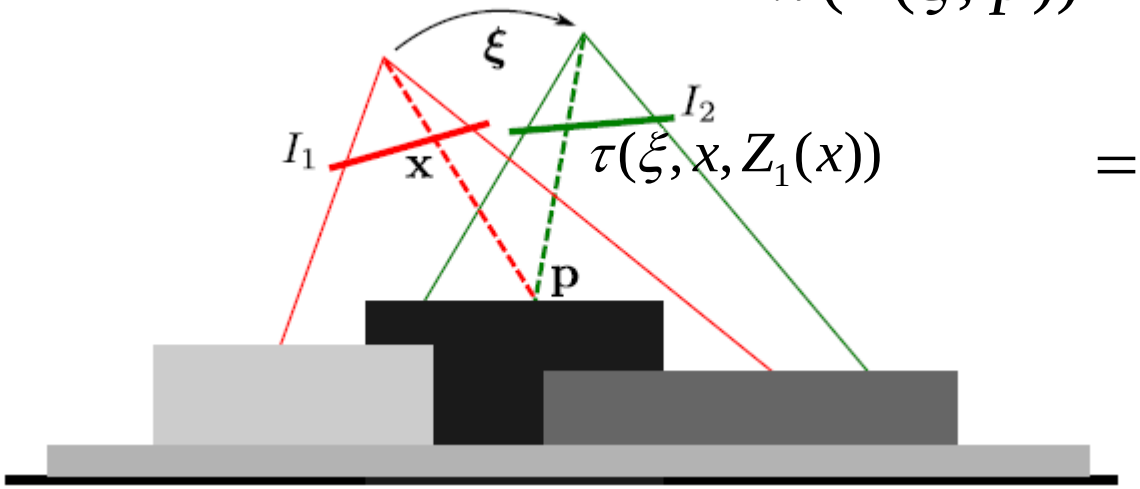


Christian Kerl, Jürgen Sturm, Daniel Cremers: Robust odometry estimation for RGB-D cameras. ICRA 2013: 3748-3754

Direct Tracking

- Warping function $T(\xi, p) = Rp + t$

$$\pi(T(\xi, p)) = \pi((X, Y, Z)^T)$$



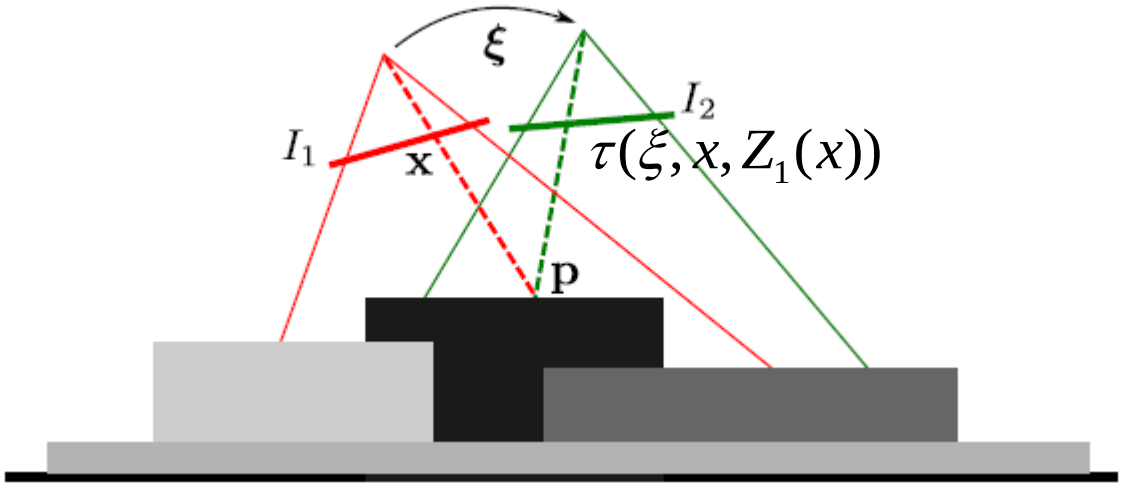
$$= \left(\frac{f_x X}{Z} + c_x, \frac{f_y Y}{Z} + c_y \right)^T$$

Christian Kerl, Jürgen Sturm, Daniel Cremers: Robust odometry estimation for RGB-D cameras. ICRA 2013: 3748-3754

Direct Tracking

- Warping function

$$\tau(\xi, x, Z_1(x)) = \pi(T(\xi, p))$$
$$= \pi(T(\xi, \pi^{-1}(x, Z_1(x))))$$



Christian Kerl, Jürgen Sturm, Daniel Cremers: Robust odometry estimation for RGB-D cameras. ICRA 2013: 3748-3754

Direct Tracking

- Residual of the *k*-th pixel

$$r_k(\xi) = I_2(w(\xi, x_k, Z_1(x_k))) - I_1(x_k)$$

- Posteriori likelihood

$$p(\xi \mid r) = \frac{p(r \mid \xi)\, p(\xi)}{p(r)} = \frac{\left(\prod_k p(r_k \mid \xi) \right) p(\xi)}{p(r)}$$

MAP Estimation

- Maximum A Posteriori (MAP)

$$\xi_{\text{MAP}} = \arg\max_{\xi} p(\xi \mid r)$$

$$= \arg\max_{\xi} \left(\prod_{k} p(r_k \mid \xi) \right) p(\xi)$$

$$= \arg\min_{\xi} - \left(\sum_{k} \log p(r_k \mid \xi) \right) - \log p(\xi)$$

MAP Estimation

- Minimum is found by taking

$$\sum_k \frac{\partial \log p(r_k \mid \xi)}{\partial \xi} = \sum_k \frac{\partial \log p(r_k)}{\partial r_k} \frac{\partial r_k}{\partial \xi}$$

$$= \sum_k \left(\frac{\partial \log p(r_k)}{\partial r_k} \frac{1}{r_k} \right) r_k \frac{\partial r_k}{\partial \xi} = 0$$

- Weighting function

$$w(r_k) = \frac{\partial \log p(r_k)}{\partial r_k} \frac{1}{r_k} \Rightarrow \sum_k w(r_k) r_k \frac{\partial r_k}{\partial \xi} = 0$$

MAP Estimation

- Equivalent weighted least squares problem

$$\sum_k w(r_k) r_k \frac{\partial r_k}{\partial \xi} = 0 \Leftrightarrow \xi_{\text{MAP}} = \arg\min_{\xi} \sum_k w(r_k) r_k^2(\xi)$$

- A special case

$$p(r_k) \propto \exp(-r_k^2 / \sigma^2) \Rightarrow \xi_{\text{MAP}} = \arg\min_{\xi} \sum_k r_k^2(\xi)$$

normal least squares

MAP Estimation

- Gaussian Newton

$$r_k(\hat{\xi}+\delta_\xi) \approx r_k(\hat{\xi}) + J_k \delta_\xi$$

$$J^T W J \delta_\xi = -J^T W r(\hat{\xi})$$

$$J = \left(\ldots, J_k^T, \ldots\right)^T$$

$$W = \mathrm{diag}(\ldots, w(r_k), \cdots)$$

- Levenberg-Marquardt

$$(J^T W J + \mu I)\delta_\xi = -J^T W r(\hat{\xi})$$

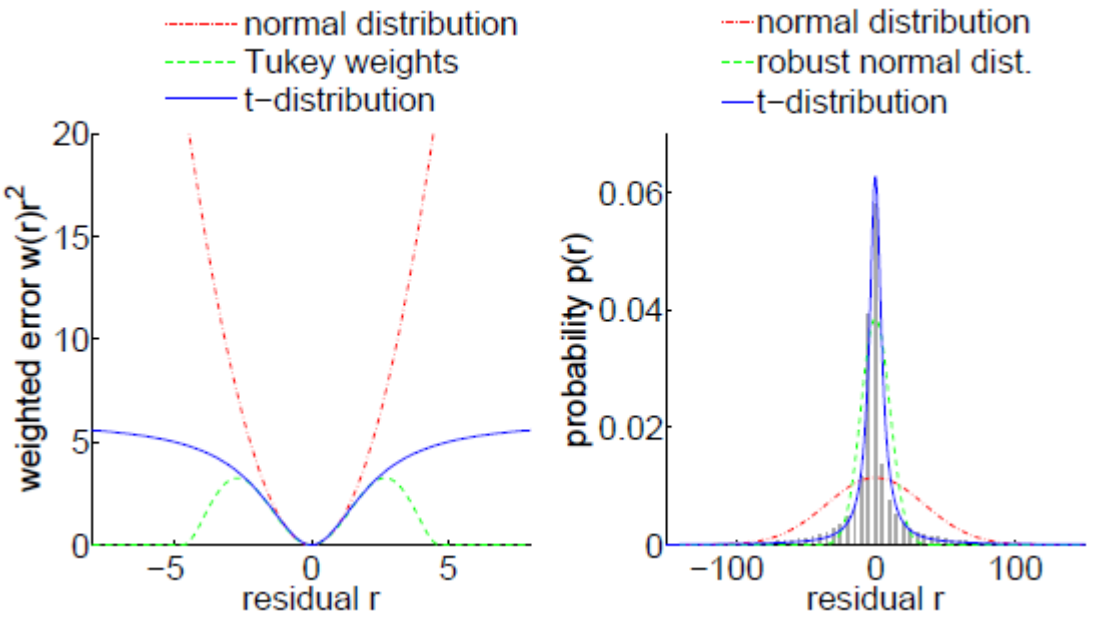
MAP Estimation

- Estimated error covariance

$$\xi \sim N\left(\xi_{\text{MAP}}, \Sigma_{\xi}\right)$$

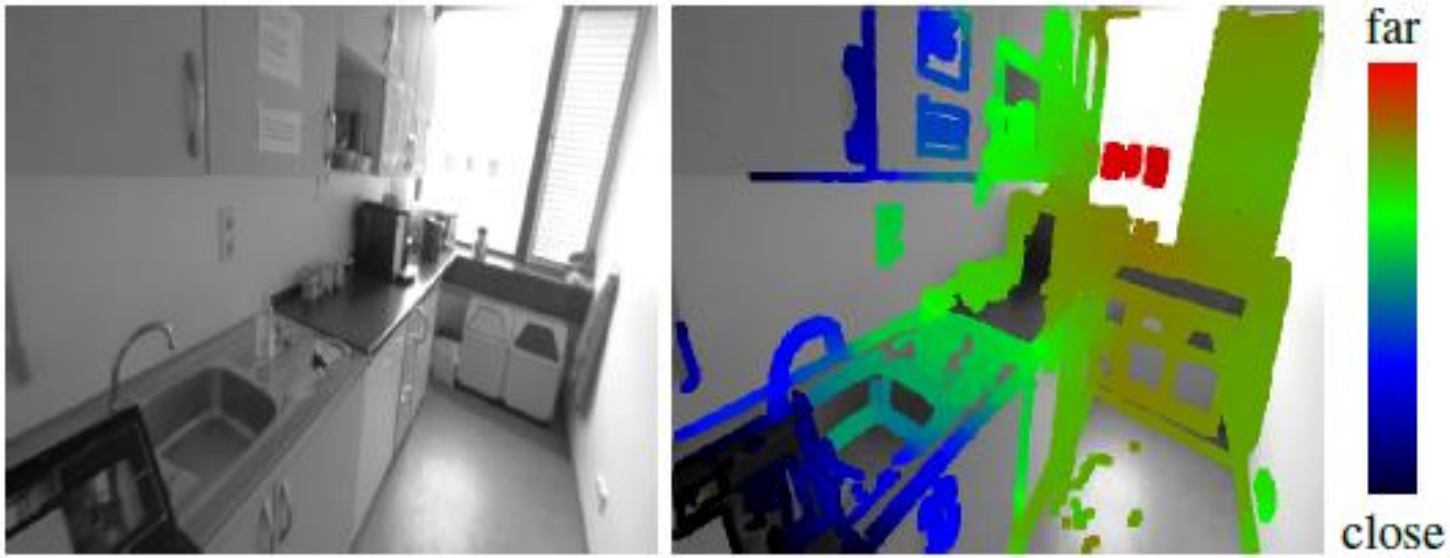
$$\Sigma_{\xi} = (J^{T} W J)^{-1}$$

Different Weighting Functions



Christian Kerl, Jürgen Sturm, Daniel Cremers: Robust odometry estimation for RGB-D cameras. ICRA 2013: 3748-3754

Semi-Dense Visual Odometry



Jakob Engel, Jürgen Sturm, Daniel Cremers: Semi-dense Visual Odometry for a Monocular Camera. ICCV 2013: 1449-1456

Semi-Dense Visual Odometry

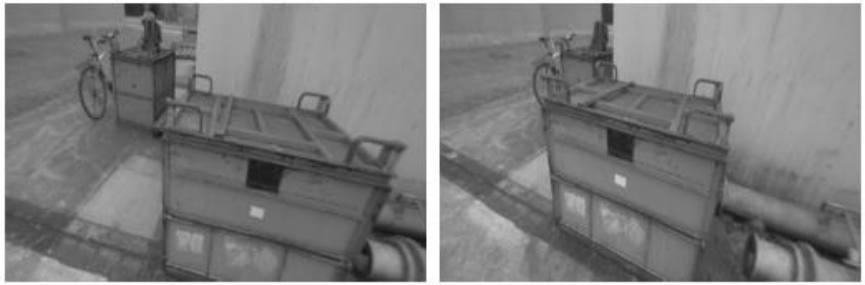
- Keyframe representation

$$K_i = (I_i, D_i, V_i)$$

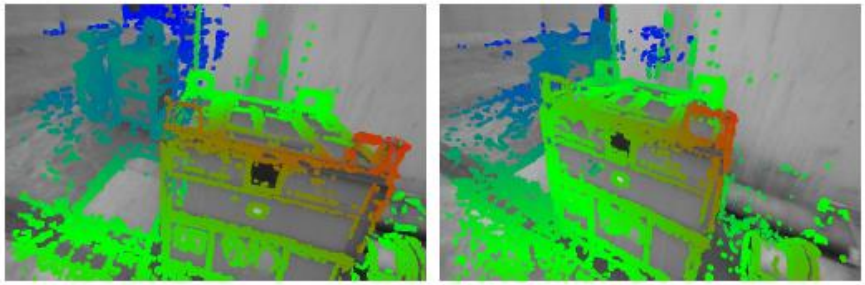
$i_i = I_i(x)$ image intensity

$d_i = D_i(x)$ inverse depth

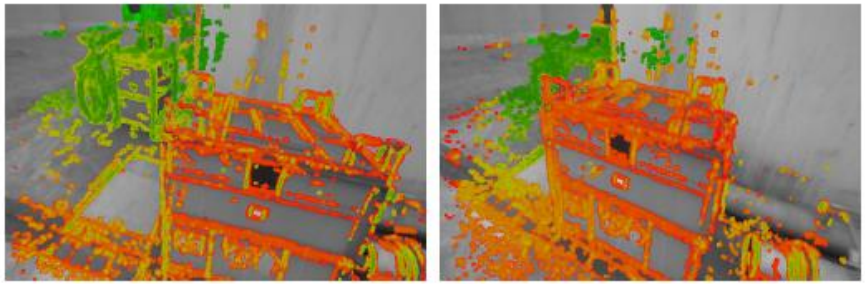
$\sigma_{d_i}^2 = V_i(x)$ inverse depth variance



(a) camera images I



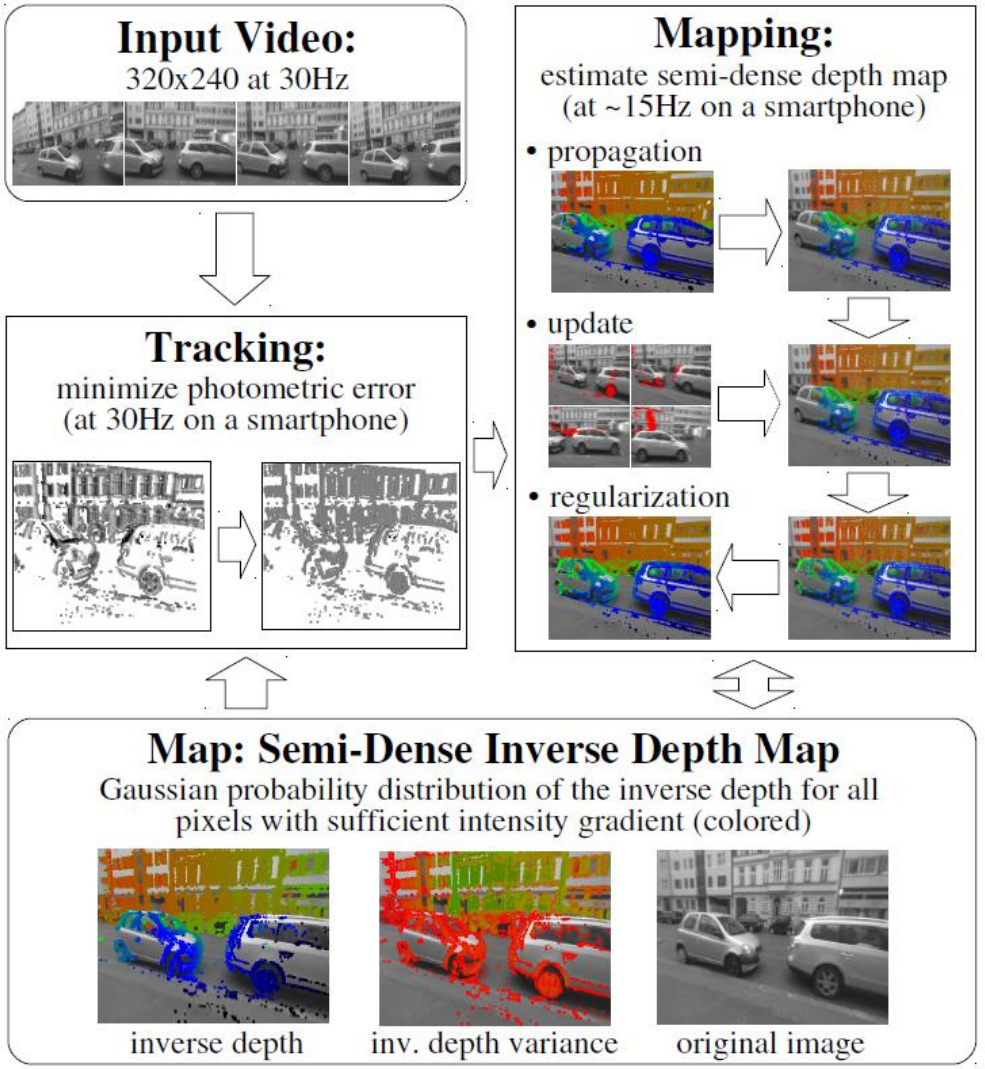
(b) estimated inverse depth maps D



(c) inverse depth variance V

Semi-Dense Visual Odometry

- Overview



Semi-Dense Visual Odometry

- Direct tracking
  - from keyframe $K_i = (I_i, D_i, V_i)$ to current image I_j

$$\xi_{ji}^* = \arg\min_{\xi_{ji}} \sum_k \left\| w(r_k) r_k^2 \right\|_\delta$$

$$r_k = I_j(\tau(\xi_{ji}, x_k, 1/d_i)) - I_i(x_k)$$

$$w(r_k) = 1 \Big/ \left(2\sigma_I^2 + \left(\frac{\partial r_k}{\partial d_i} \right)^2 \sigma_{d_i}^2 \right)$$

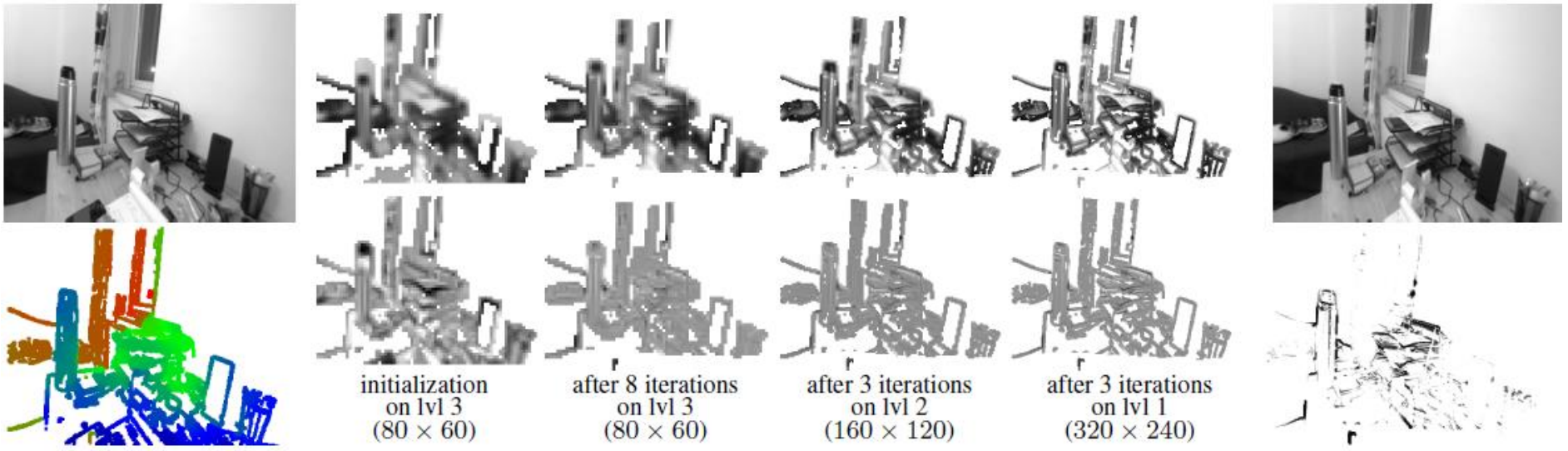
$$\left\| r^2 \right\|_\delta = \begin{cases} r^2 / 2\delta & \text{if } |r| \le \delta \\ |r| - \delta/2 & \text{otherwise} \end{cases}$$

Huber norm

Semi-Dense Visual Odometry

- Coarse-to-fine

keyframe K_i warped image $I_j(\tau(\xi_{ji}, x_k, 1/d_i))$ current image I_j



residual image r_k Weight image $w(r_k)$

Semi-Dense Visual Odometry

- Depth propagation

$$d_j = 1/T_Z(\xi_{ji}, \pi^{-1}(x, 1/d_i))$$

$$\sigma_{d_j}^2 = J_{\xi_{ji}} \Sigma_{\xi_{ji}} J_{\xi_{ji}}^T + J_{d_i} \sigma_{d_i}^2 J_{d_i}^T$$

$$J_{\xi_{ji}} = \frac{\partial d_j}{\partial \xi_{ji}}$$

$$J_{d_i} = \frac{\partial d_j}{\partial d_i}$$

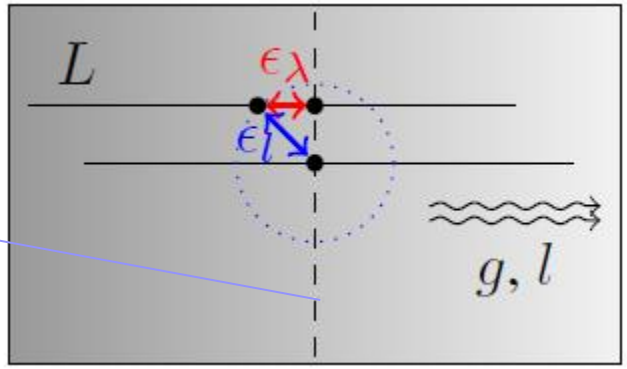
Semi-Dense Visual Odometry

- Stereo matching
  - Step 1: compute epipolar line $L = \{l_0 + \lambda(l_x, l_y)^T\}$
  - Step 2: search for best disparity λ^* in interval $d_i \pm 2\delta_i$
  - Step 3: compute inverse depth d_i^* from disparity λ^*
- Error source
  - Step 1: geometric error from error of ξ_{ji}
  - Step 2: photometric error from noise in I_i and I_j
  - Step 3: a factor depends on baseline

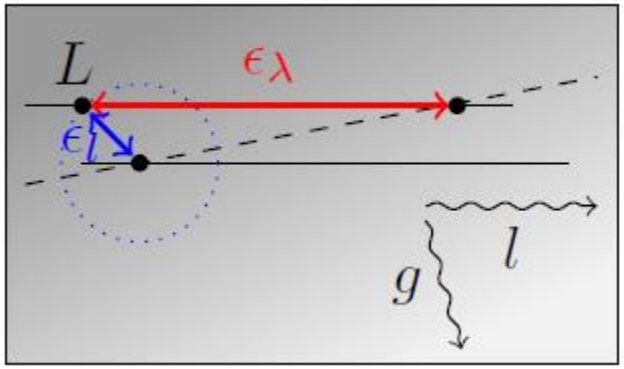
Semi-Dense Visual Odometry

- Geometric error $\varepsilon_l \sim N(0,\sigma_l^2) \Rightarrow \varepsilon_{\lambda(\xi)} \sim N(0,\sigma_{\lambda(\xi)}^2)$
  - Only consider position error of l_0
  - Ignore direction error of $l = (l_x, l_y)^T$

matching isocurve



l parallel to image gradient *g*,
ϵ_l cause small disparity error ϵ_λ



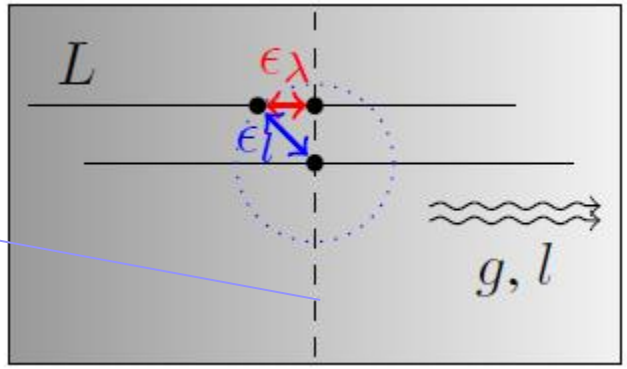
l nearly perpendicular to *g*,
ϵ_l cause large disparity error ϵ_λ

Semi-Dense Visual Odometry

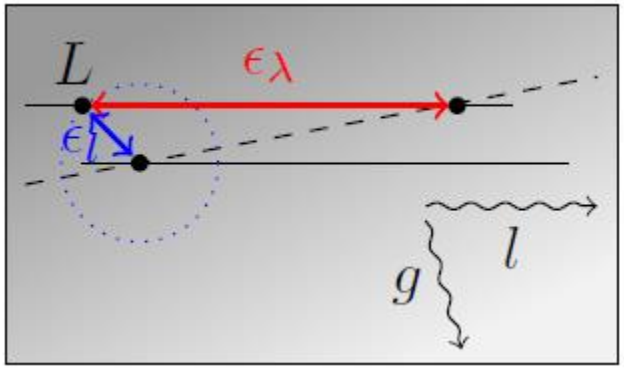
- Geometric error $\varepsilon_l \sim N(0,\sigma_l^2) \Rightarrow \varepsilon_{\lambda(\xi)} \sim N(0,\sigma_{\lambda(\xi)}^2)$
  - $l_0 + \lambda^*(l_x,l_y)^T = g_0 + \gamma(-g_y,g_x)^T$

$$\Rightarrow \lambda^* = \frac{\langle g, g_0 - l_0 \rangle}{\langle g,l \rangle} \Rightarrow \sigma_{\lambda(\xi)}^2 = \frac{\sigma_l^2}{\langle g,l \rangle^2}$$

matching isocurve



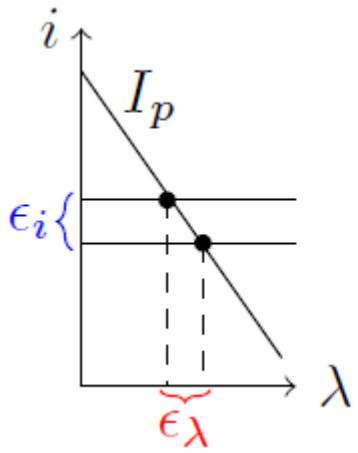
l parallel to image gradient *g*,
ϵ_l cause small disparity error ϵ_λ



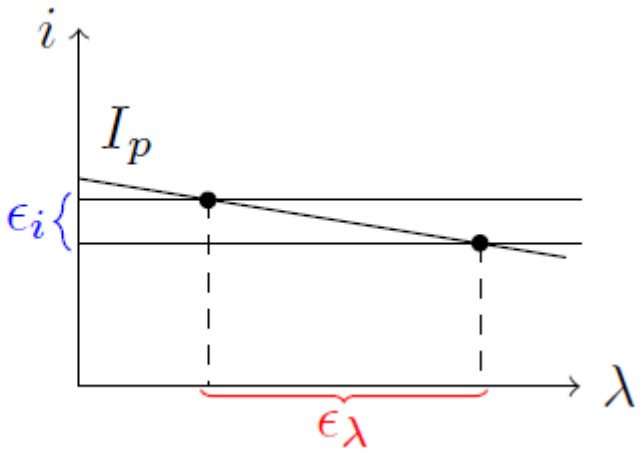
l nearly perpendicular to *g*,
ϵ_l cause large disparity error ϵ_λ

Semi-Dense Visual Odometry

- Photometric error $\varepsilon_i \sim N(0, \sigma_i^2) \Rightarrow \varepsilon_{\lambda(I)} \sim N(0, \sigma_{\lambda(I)}^2)$
  - $I_p(\lambda)$: along epipolar line at disparity λ
  - g_p : gradient of $I_p(\lambda)$



When g_p is large, ϵ_i cause small disparity error ϵ_λ

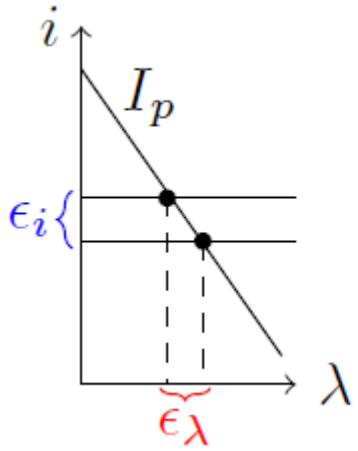


When g_p is small, ϵ_i cause large disparity error ϵ_λ

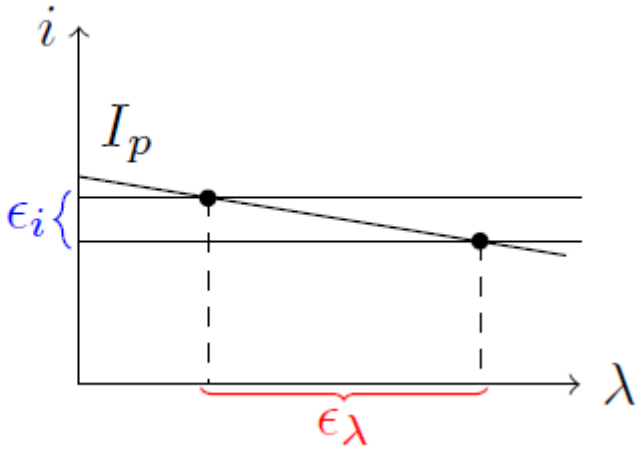
Semi-Dense Visual Odometry

- Photometric error $\varepsilon_i \sim N(0,\sigma_i^2) \Rightarrow \varepsilon_{\lambda(I)} \sim N(0,\sigma_{\lambda(I)}^2)$
  - $\lambda^* = \arg\min_{\lambda} \left(i_i - I_p(\lambda)\right)^2 \approx \lambda_0 + \left(i_i - I_p(\lambda)\right) g_p^{-1}$

    $\Rightarrow \sigma_{\lambda(I)}^2 = 2\sigma_i^2 / g_p^2$



When g_p is large, ϵ_i cause small disparity error ϵ_λ



When g_p is small, ϵ_i cause large disparity error ϵ_λ

Semi-Dense Visual Odometry

- Disparity to inverse depth conversion

$$d_{i,\text{obs}} = d_i^*$$

$$\sigma^2_{d_i,\text{obs}} = \alpha^2(\sigma^2_{\lambda(\xi)} + \sigma^2_{\lambda(I)})$$

$$\alpha = \frac{\delta_d}{\delta_\lambda}$$

δ_d — Length of the searched inverse depth interval

δ_λ — Length of the searched epipolar segment

Semi-Dense Visual Odometry

- Depth update
  - Stereo matching obtains new depth observation

$$N(d_{i,\text{obs}}, \sigma^2_{d_i,\text{obs}})$$

  - Fuse to the existing depth state (Kalman filter)

$$N(d_i, \sigma^2_{d_i}) \rightarrow N\left(\frac{\sigma^2_{d_i} d_{i,\text{obs}} + \sigma^2_{d_i,\text{obs}} d_i}{\sigma^2_{d_i} + \sigma^2_{d_i,\text{obs}}}, \frac{\sigma^2_{d_i} \sigma^2_{d_i,\text{obs}}}{\sigma^2_{d_i} + \sigma^2_{d_i,\text{obs}}}\right)$$

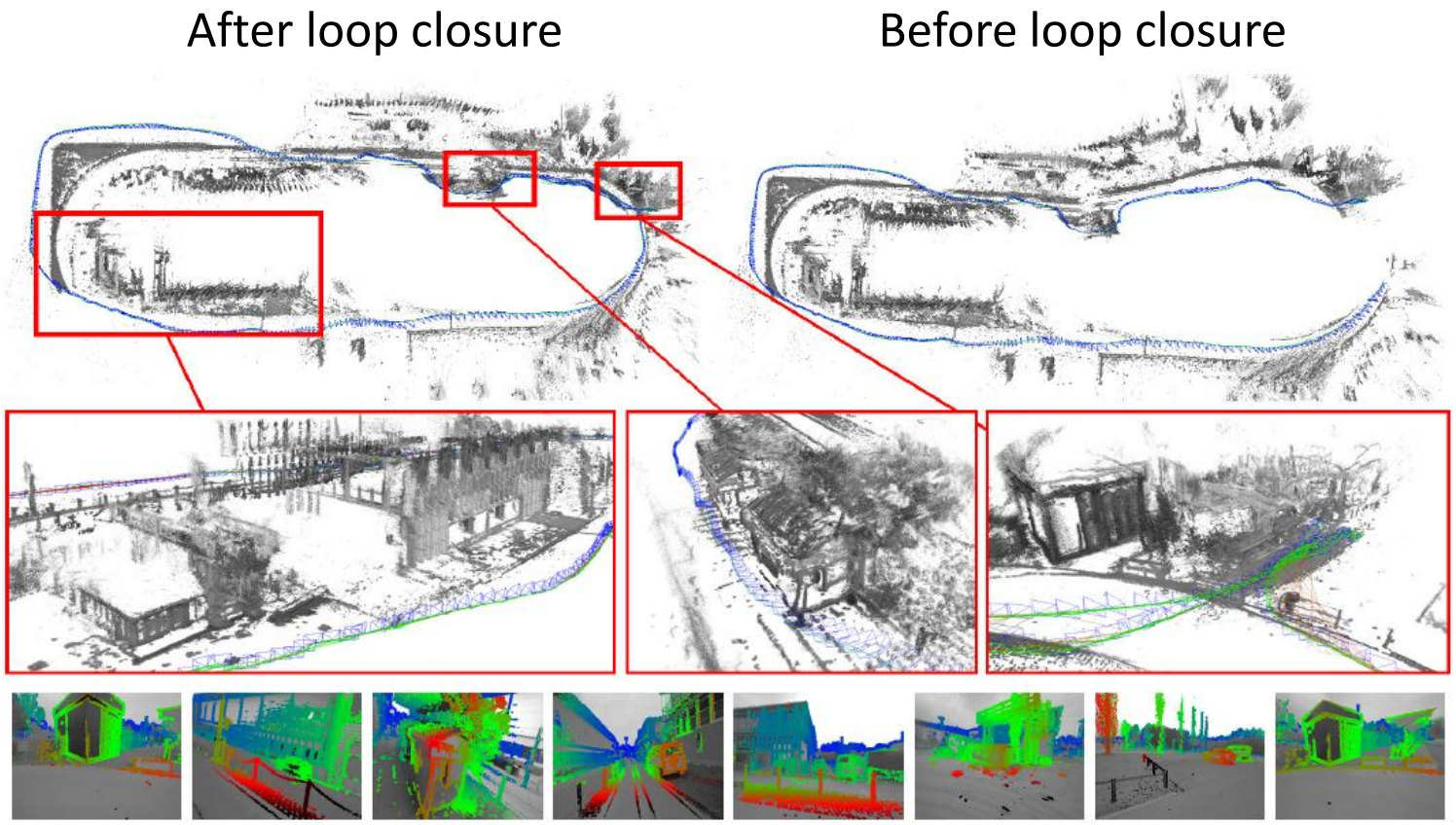
Semi-Dense Visual Odometry

- Depth regularization

$$D_r(x_i) = \frac{\sum_{x_j \in \Omega(x_i)} V(x_j) D(x_j)}{\sum_{x_j \in \Omega(x_i)} V(x_j)}$$

$\Omega(x_i)$: neighbors of x_i satisfying $\left| D(x_j) - D(x_j) \right| < 2\sigma$

LSD-SLAM



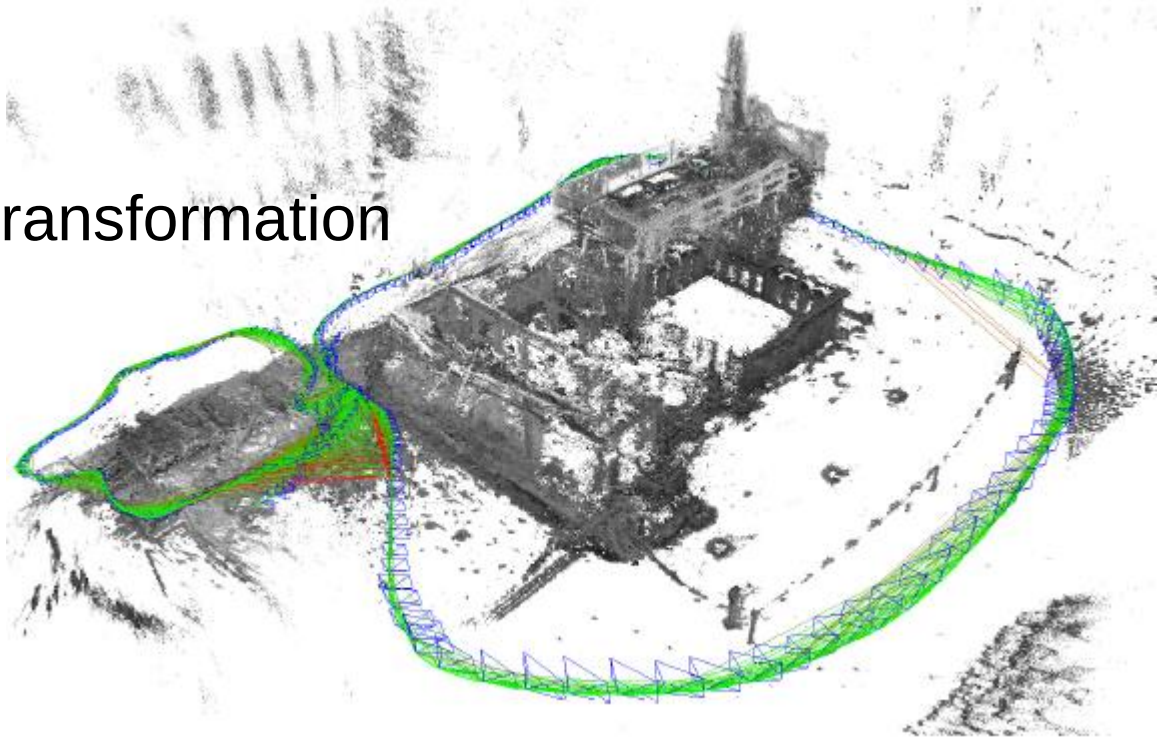
Jakob Engel, Thomas Schops, Daniel Cremers: LSD-SLAM: Large-Scale Direct Monocular SLAM. ECCV (2) 2014: 834-849

LSD-SLAM

- Map representation
  - Pose graph of keyframes
  - Node: keyframe

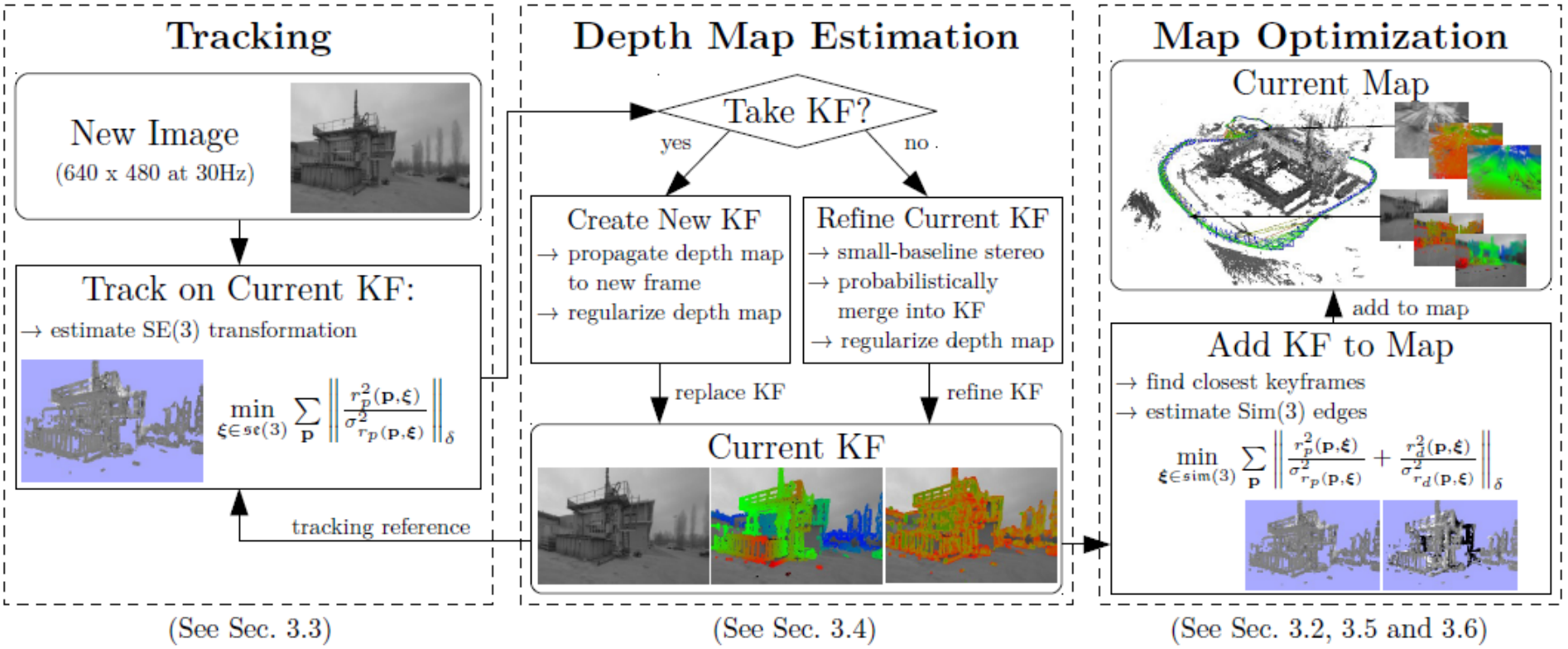
    $K_i = (I_i, D_i, V_i)$
  - Edge: similarity transformation

    $\xi_{ji} \in \mathrm{sim}(3)$



LSD-SLAM

- Overview



LSD-SLAM

- Direct sim(3) image alignment

$$\xi_{ji}^{*} = \arg\min_{\xi_{ji}} \sum_{p} \left\| \frac{r_p^2(p,\xi_{ji})}{\sigma^2_{r_p^2(p,\xi_{ji})}} + \frac{r_d^2(p,\xi_{ji})}{\sigma^2_{r_d^2(p,\xi_{ji})}} \right\|_{\delta}$$

$$r_p(p,\xi_{ji}) = I_j(\tau(\xi_{ji}, p, 1/d_i)) - I_i(p)$$

$$\sigma^2_{r_p^2(p,\xi_{ji})} = 2\sigma_I^2 + \left(\frac{\partial r_p}{\partial d_i}\right)^2 \sigma^2_{d_i}$$

$$r_d(p,\xi_{ji}) = 1/T_Z(\xi_{ji}, \pi^{-1}(p, 1/d_i)) - D_j(p_\tau)$$

$$\sigma^2_{r_d^2(p,\xi_{ji})} = V_j(p_\tau)\left(\frac{\partial r_d}{D_j(p_\tau)}\right)^2 + V_i(p)\left(\frac{\partial r_d}{D_i(p)}\right)^2$$

$$p_\tau = \tau(\xi_{ji}, p, 1/d_i)$$

LSD-SLAM

- Pose graph optimization
  - Energy function:

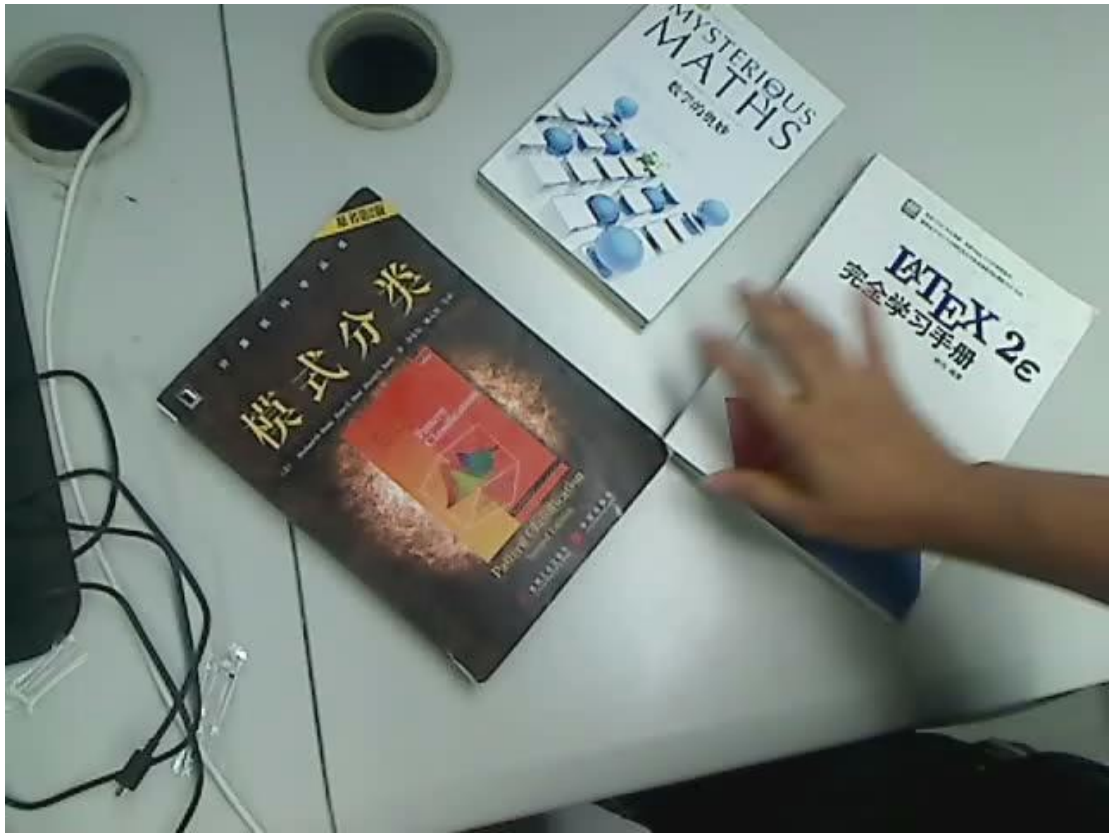
$$E(\boldsymbol{\xi}_{W1} \dots \boldsymbol{\xi}_{Wn}) := \sum_{(\boldsymbol{\xi}_{ji}, \boldsymbol{\Sigma}_{ji}) \in \mathcal{E}} (\boldsymbol{\xi}_{ji} \circ \boldsymbol{\xi}_{Wi}^{-1} \circ \boldsymbol{\xi}_{Wj})^T \boldsymbol{\Sigma}_{ji}^{-1} (\boldsymbol{\xi}_{ji} \circ \boldsymbol{\xi}_{Wi}^{-1} \circ \boldsymbol{\xi}_{Wj})$$

Kummerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g2o: A general framework for graph optimization. In: Intl. Conf. on Robotics and Automation(ICRA) (2011)

Robust Monocular SLAM in Dynamic Environments

Key Issues for SLAM in Dynamic Environments

- Gradually changing



Key Issues for SLAM in Dynamic Environments

- Gradually changing
- Object Occlusion
  - Viewpoint Change

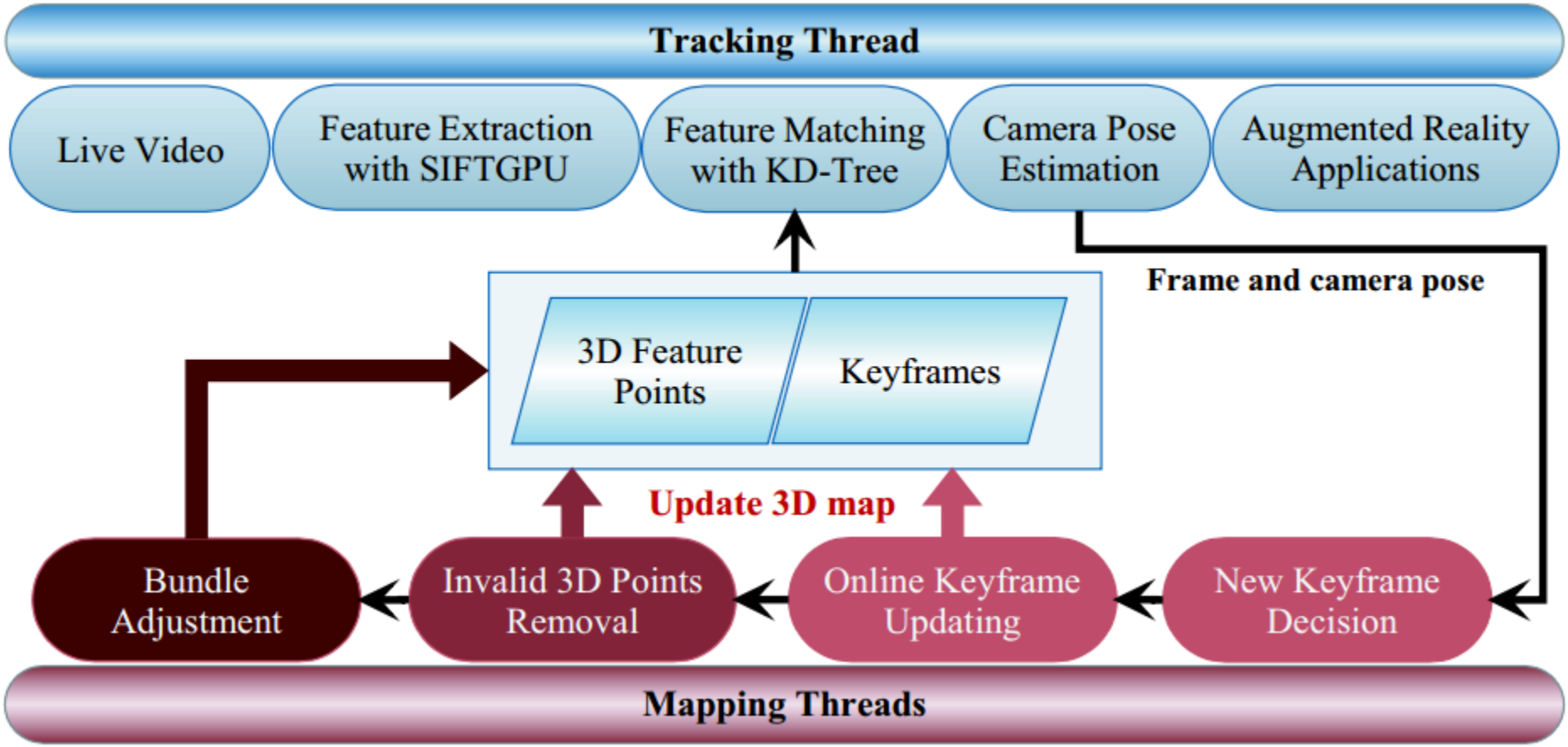
Key Issues for SLAM in Dynamic Environments

- Gradually changing
- Object Occlusion
  - Viewpoint Change
  - Dynamic Objects

Key Issues for SLAM in Dynamic Environments

- Gradually changing
- Object Occlusion
  - Viewpoint Change
  - Dynamic Objects
- Very low inlier ratio

Our Framework

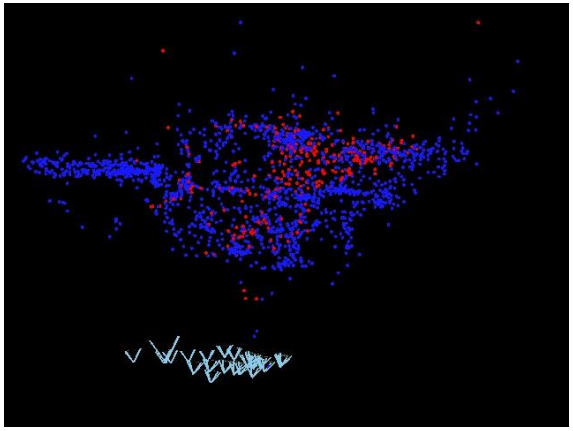


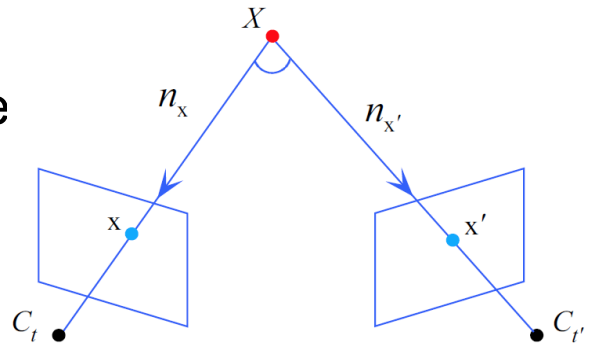
Roadmap

- Background and Related Work
- Key Issues for SLAM in Dynamic Scenes
- System Overview
- Online 3D Points and Keyframes Updating
- Prior-based Adaptive RANSAC
- Results and Comparison

Online 3D Points and Keyframes Updating

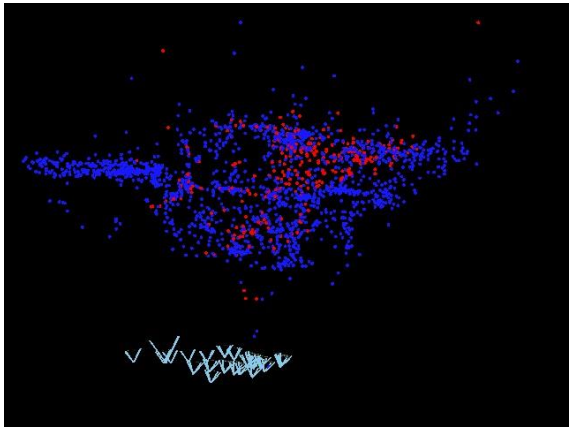
- Keyframe representation
- 3D Change detection
  - Select 5 closest keyframes for online image.
  - For each valid feature point x in each selected keyframe,
    - Compute its projection x' in current frame
    - If $n_{\mathbf{X}}^{\top} \hat{n}_{\mathbf{X}'} < \tau_n$, compute the appearance difference $D_c(X) = \min_d \sum_{\mathbf{y} \in W(\mathbf{x})} |I_{\mathbf{y}} - I_{\mathbf{y}'+d}|$

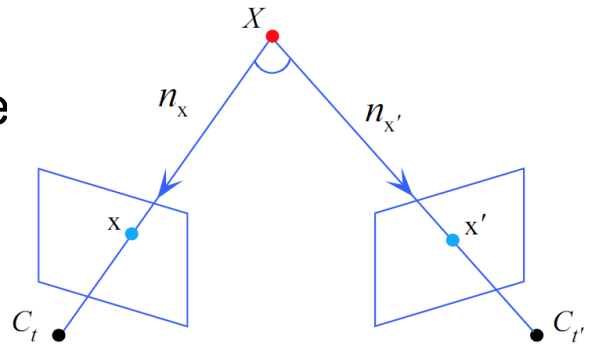




Online 3D Points and Keyframes Updating

- Keyframe representation
- 3D Change detection
  - Select 5 closest keyframes for online image.
  - For each valid feature point x in each selected keyframe,
    - Compute its projection x' in current frame
    - If $n_{\mathbf{X}}^{\top} \hat{n}_{\mathbf{X}'} < \tau_n$, compute the appearance difference $D_c(X) = \min_d \sum_{\mathbf{y} \in W(\mathbf{x})} |I_{\mathbf{y}} - I_{\mathbf{y}'+d}|$
      - If $D_c(X) > \tau_c$, then find a set of feature points y close to x'.

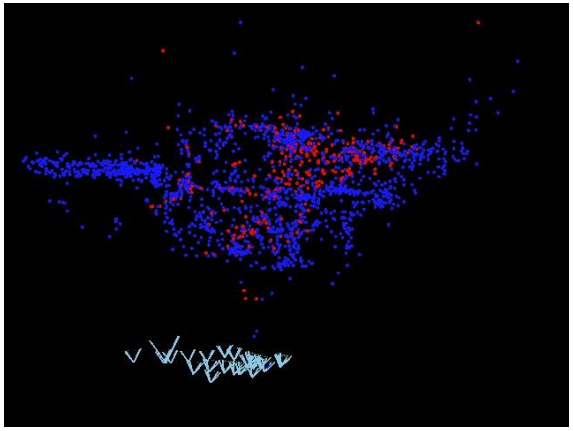


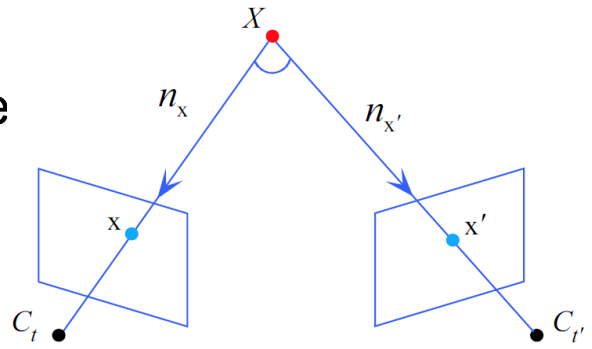


Since dynamic points cannot be triangulated, the occlusion caused by dynamic objects can be excluded here.

Online 3D Points and Keyframes Updating

- Keyframe representation
- 3D Change detection
  - Select 5 closest keyframes for online image.
  - For each valid feature point x in each selected keyframe,
    - Compute its projection x' in current frame
    - If $n_{\mathbf{X}}^{\top} \hat{n}_{\mathbf{X}'} < \tau_n$, compute the appearance difference $D_c(X) = \min_d \sum_{\mathbf{y} \in W(\mathbf{x})} |I_{\mathbf{y}} - I_{\mathbf{y}'+d}|$
      - If $D_c(X) > \tau_c$, then find a set of feature points y close to x'.
    - If $z_{X_{\mathbf{y}}} \geq z_X$ or their depths are very close, set V(X)=0.

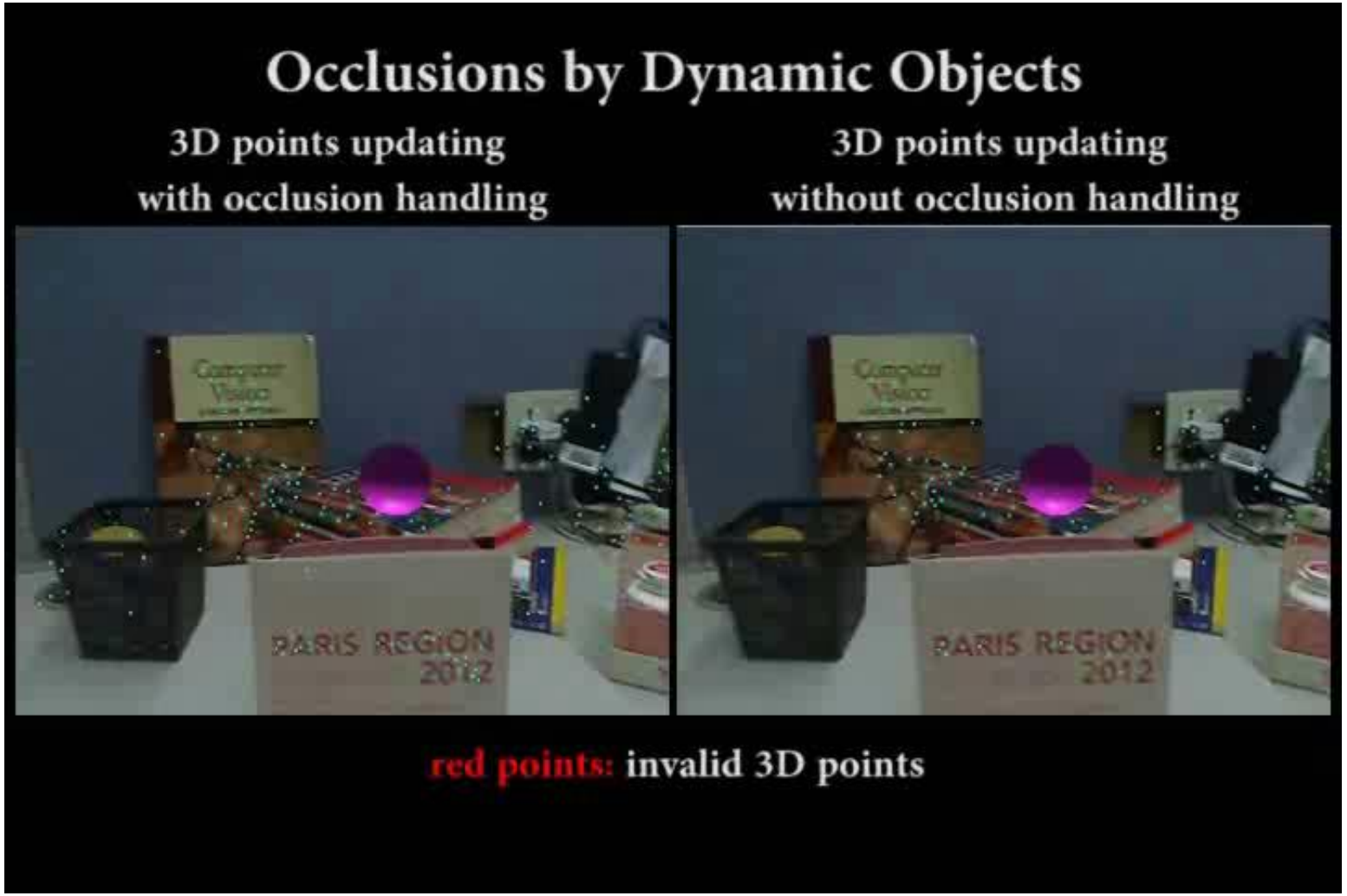




Since dynamic points cannot be triangulated, the occlusion caused by dynamic objects can be excluded here.

The occlusions caused by static objects are also excluded.

Occlusion Handling



Occlusion Handling

(a) The SLAM result without occlusion handling.
(b) The SLAM result with occlusion handling.

Roadmap

- Background and Related Work
- Key Issues for SLAM in Dynamic Scenes
- System Overview
- Online 3D Points and Keyframes Updating
- Prior-based Adaptive RANSAC
- Results and Comparison

Random Sample Consensus (RANSAC)

[Fischler and Bolles, 1981]

Objective Robust fit of a model to a data set S which contains outliers.

Step 1. Compute a set of potential matches

Step 2. While T(#inliers, #samples) < 95% do

step 2.1 select minimal sample (6 matches)

step 2.2 compute solutions for *P*

step 2.3 determine inliers

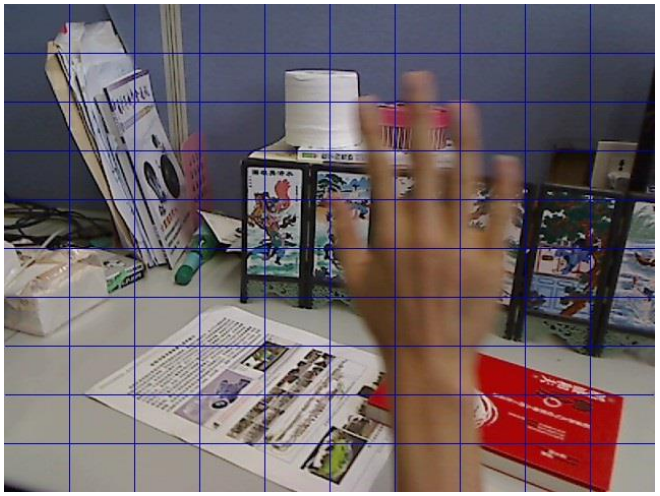
Step 3. Refine *P* based on all inliers

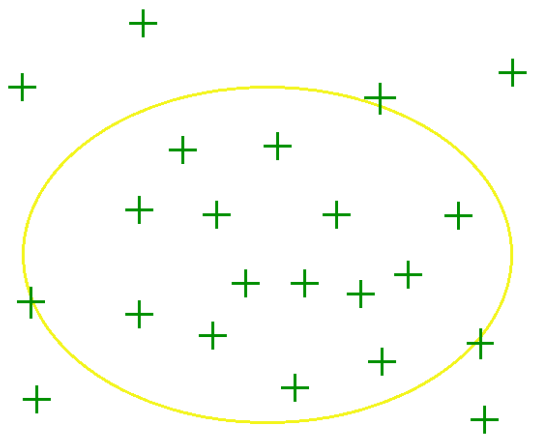
Prior-based Adaptive RANSAC

- Sample generation
  - 10x10 bins
  - Prior probability $p_i = \varepsilon_i^* / \sum_j \varepsilon_j^*$
- Hypothesis evaluation

$$s = (\sum_i \varepsilon_i) \frac{\pi \sqrt{\det(C)}}{A}$$

  - Inliers number $N \approx \sum_i \varepsilon_i$
  - Inliers distribution, i.e., distribution ellipse C



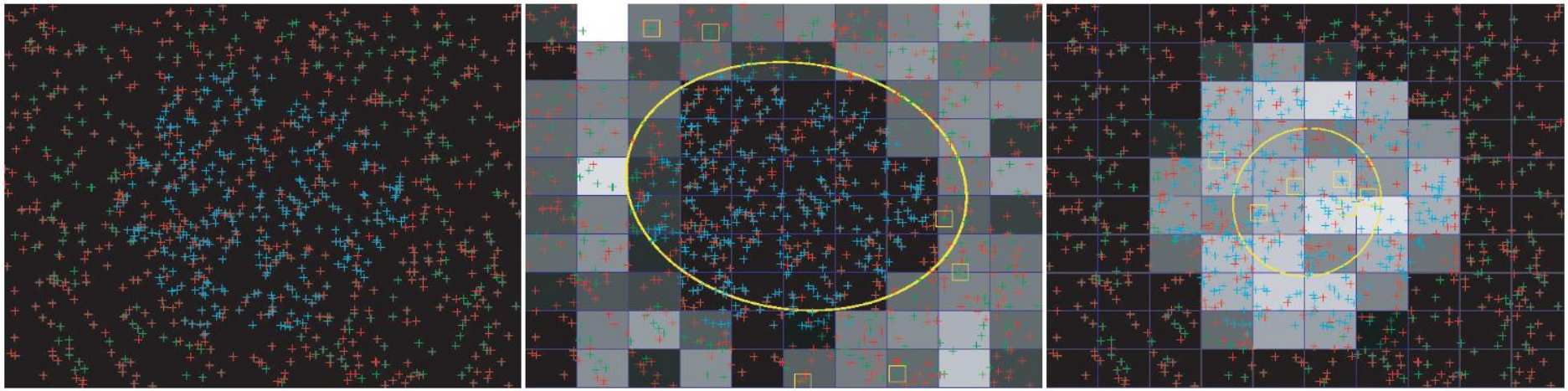


Prior-based Adaptive RANSAC

- Hypothesis evaluation

$$s = (\sum_i \varepsilon_i) \frac{\pi \sqrt{\det(C)}}{A}$$

$\sum_i \varepsilon_i$=24.94 $\sum_i \varepsilon_i$=21.77



200 green points on the static background, 300 cyan points on the rigidly moving object, 500 red points are randomly moving.

Prior-based Adaptive RANSAC

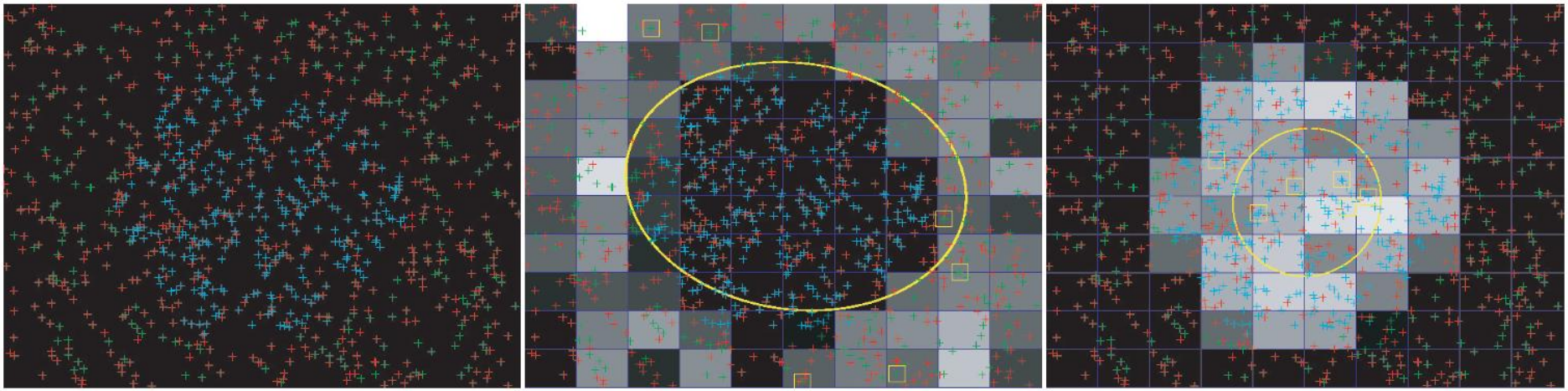
- Hypothesis evaluation

$$s = (\sum_i \varepsilon_i) \frac{\pi \sqrt{\det(C)}}{A}$$

S1 = 8.31 > S2 = 1.98

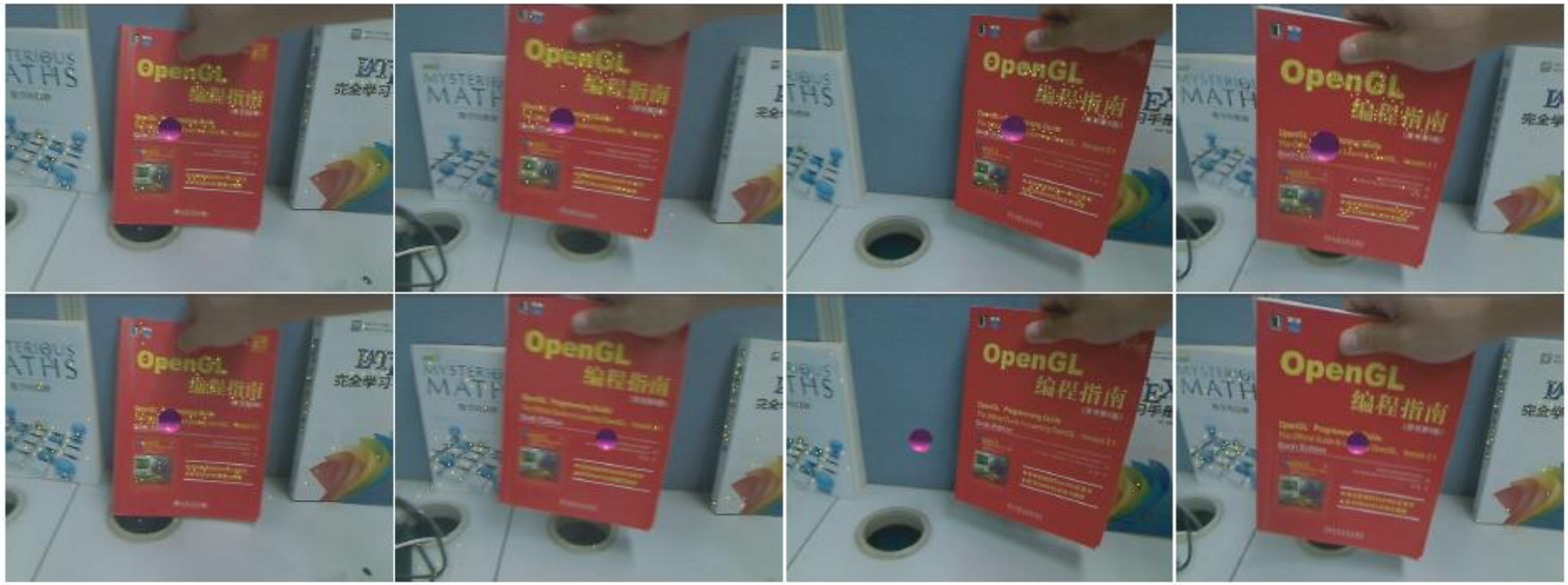
$\sum_i \varepsilon_i$=24.94

$\sum_i \varepsilon_i$=21.77



200 green points on the static background, 300 cyan points on the rigidly moving object, 500 red points are randomly moving.

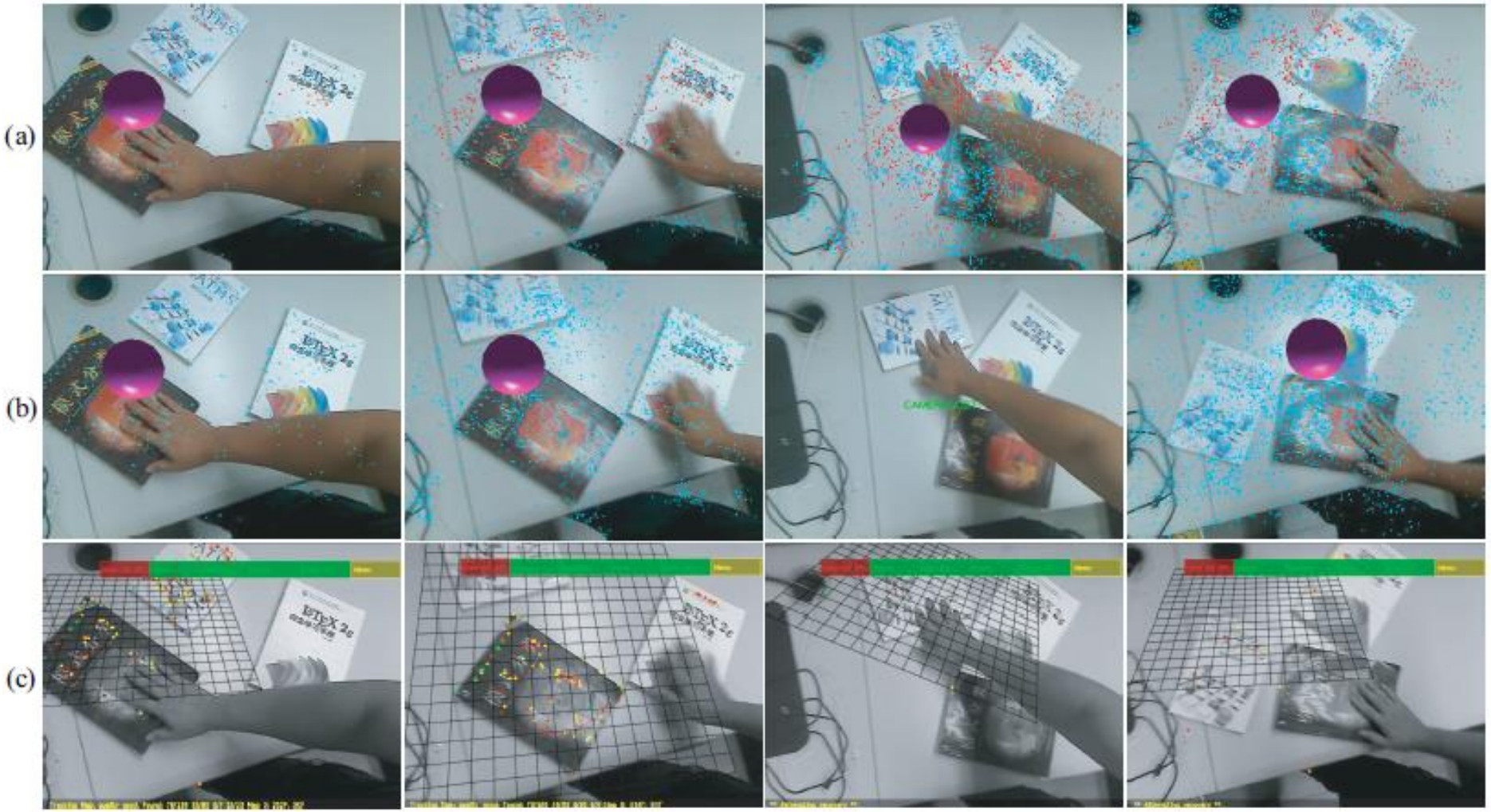
Result Comparison



(a) The SLAM result with standard RANSAC
(b) The SLAM result with our PARSAC

Results and Comparison

Results and Comparison



RDSLAM: Robust Dynamic SLAM

Home

Description

RDSLAM is a real-time simultaneous localization and mapping system which can robustly work in dynamic environments. **It is for non-commercial research and educational use ONLY. Not for reproduction, distribution or commercial use.** If you use this executable for your academic publication, please acknowledge our work. This program is tested on Win7, but is still not guaranteed to be bug-free and work properly with all versions of Windows. You are welcome to report any suggestions or bugs. We will actively update the program. Please email Guofeng Zhang if you have any questions.

Release (RDSLAM1.0 released on Dec. 11, 2013)

RDSLAM1.0 is implemented based on the following paper:

Wei Tan, Haomin Liu, Zilong Dong, Guofeng Zhang* and Hujun Bao. Robust Monocular SLAM in Dynamic Environments. International Symposium on Mixed and Augmented Reality (ISMAR), 2013.

Changelog

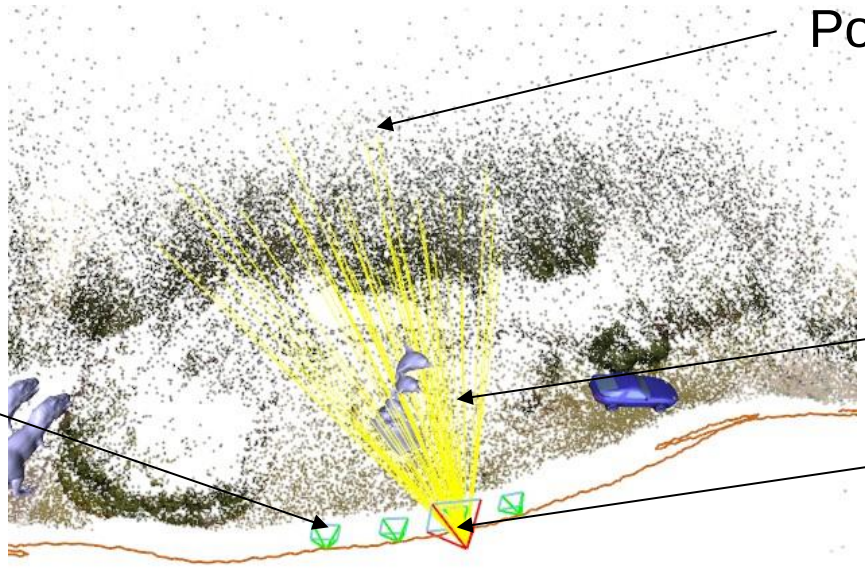
http://www.zjucvg.net/rdslam/rdslam.html

Real-Time Global Relocalization

- Traditional Structure-from-Motion
  - The complexity is between linear and quadratic in the number of processed frames.
  - Difficult to run at real-time in large-scale scene.
- Alternative Strategy for Real-Time Camera Tracking
  - If the 3D structure of the scene is known
    - Real-time camera tracking can be much easier.
  - Limited Real-Time Camera Tracking
    - Offline environment modeling
    - Online camera tracking

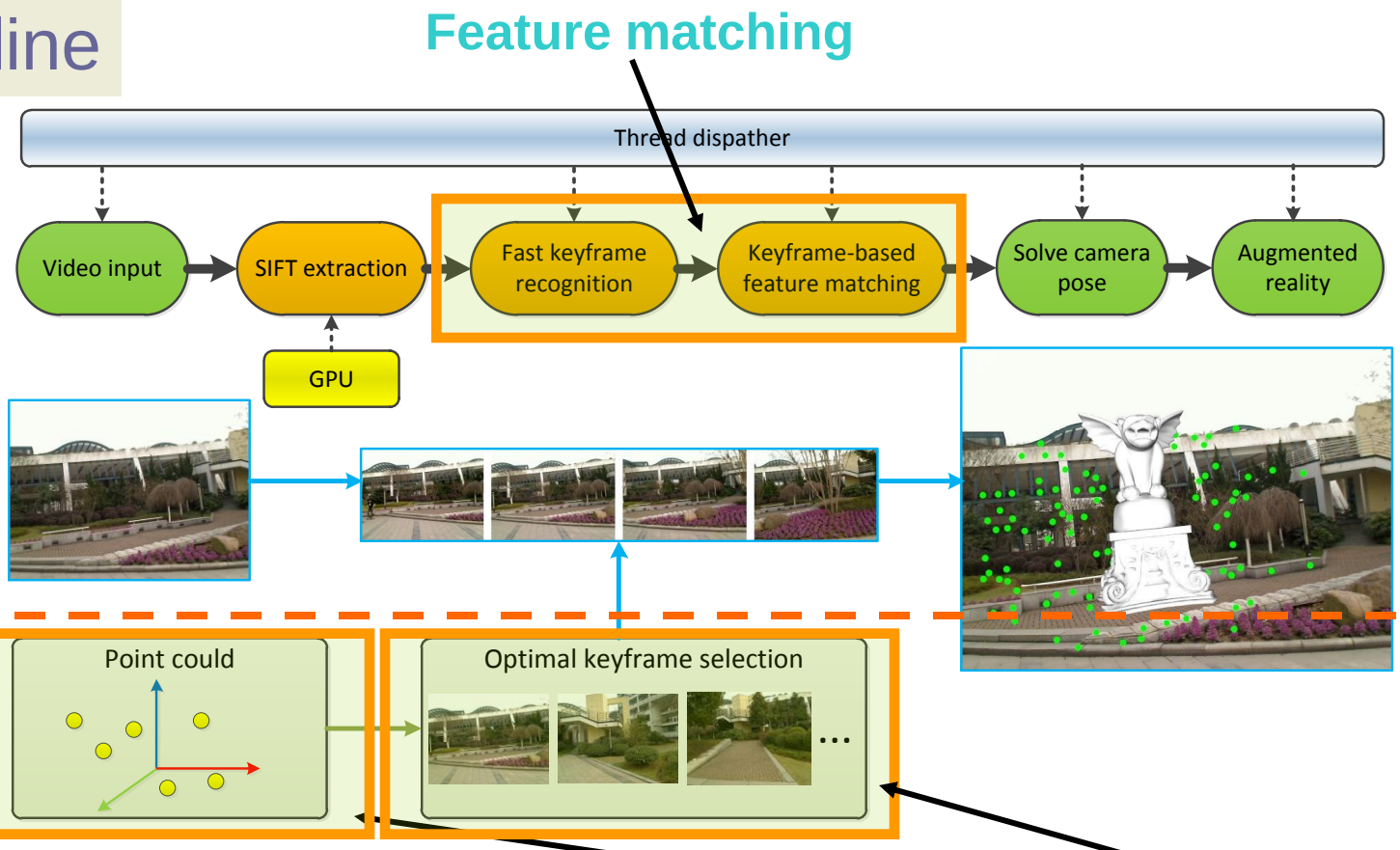
Real-Time Global Relocalization

- Use Invariant features，e.g. SIFT, SURF
  - Scale-invariant
  - Rotation-invariant
  - Illumination-invariant



Framework

Online



Offline

Reconstruct point could using SFM

Automatic keyframe selection

Offline Reference 3D points Reconstruction

- Input reference sequence and reconstructed point could

Automatic keyframe selection

- Motivation
  - Efficiency of feature matching
  - Distinctiveness of SIFT description
- Objective

Completeness

keep as many features as possible

$$E_c(F)$$

Redundancy

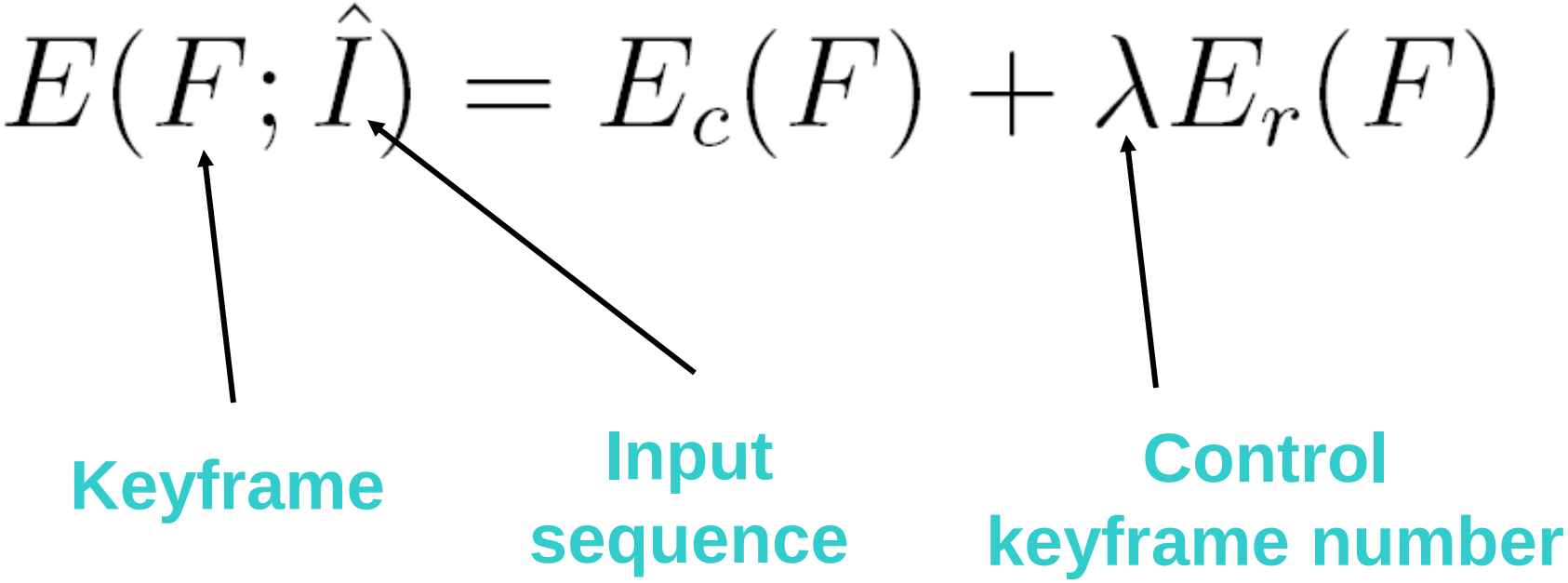
keyframe number be small

$$E_r(F)$$

Objective function

- Solved by Greedy method

$$E(F; \hat{I}) = E_c(F) + \lambda E_r(F)$$



Statistics of keyframe selection

- Keyframes of 1937 input images

λ	keyframes	E_c	E_r	ratio of points
0.1	123	0.020207	0.584265	95.7998%
1.0	65	0.194271	0.142999	70.2325%
2.0	51	0.289593	0.071128	58.4981%
5.0	33	0.443603	0.022406	42.1656%
10	24	0.558724	0.006362	31.1694%
100	12	0.739066	0.000220	16.4674%

Result of keyframe selection

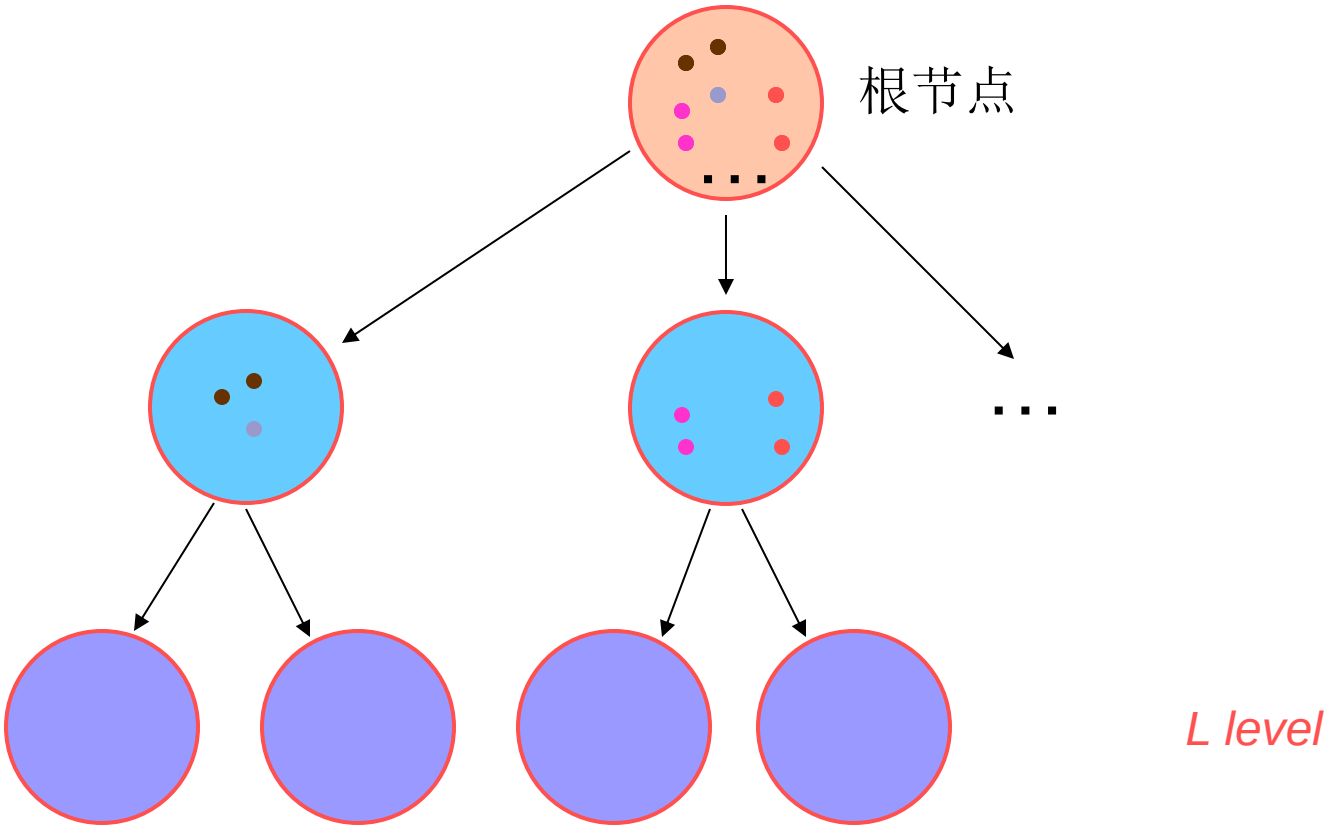
Fast keyframe recognition

- Image recognition method
  - Vocabulary tree

Candidate keyframe

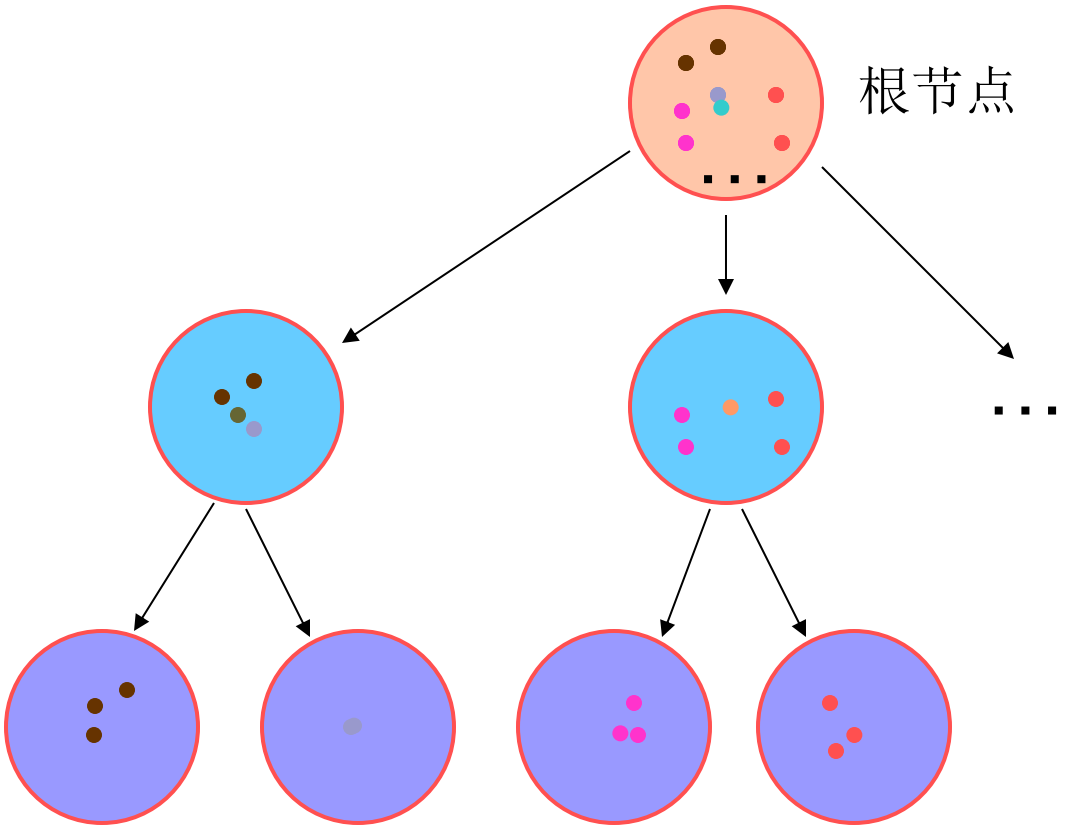
Fast keyframe recognition

- Building vocabulary tree（KMeans）



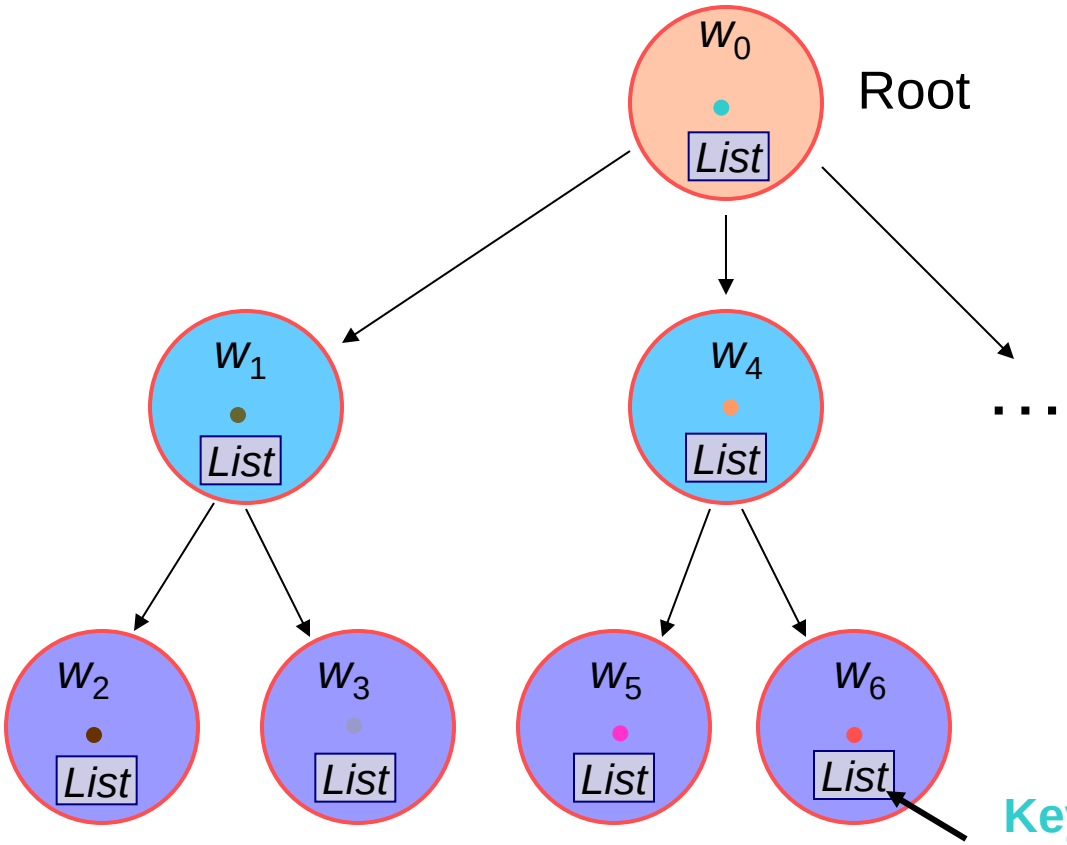
Fast keyframe recognition

- Building vocabulary tree



Fast keyframe recognition

- Building vocabulary tree

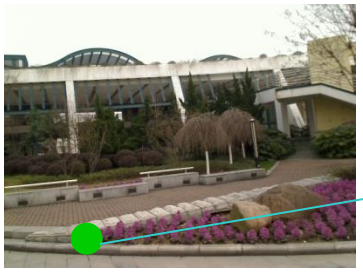


Total keyframe number

$$w_i = \log \frac{K}{|L_i|}$$

Keyframes covered by the features in the node

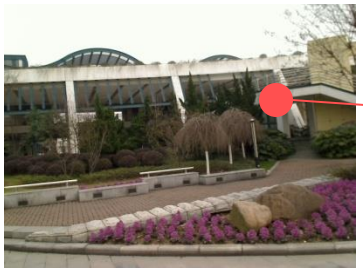
Fast keyframe recognition



Input image

Keyframe list

Fast keyframe recognition



Input image

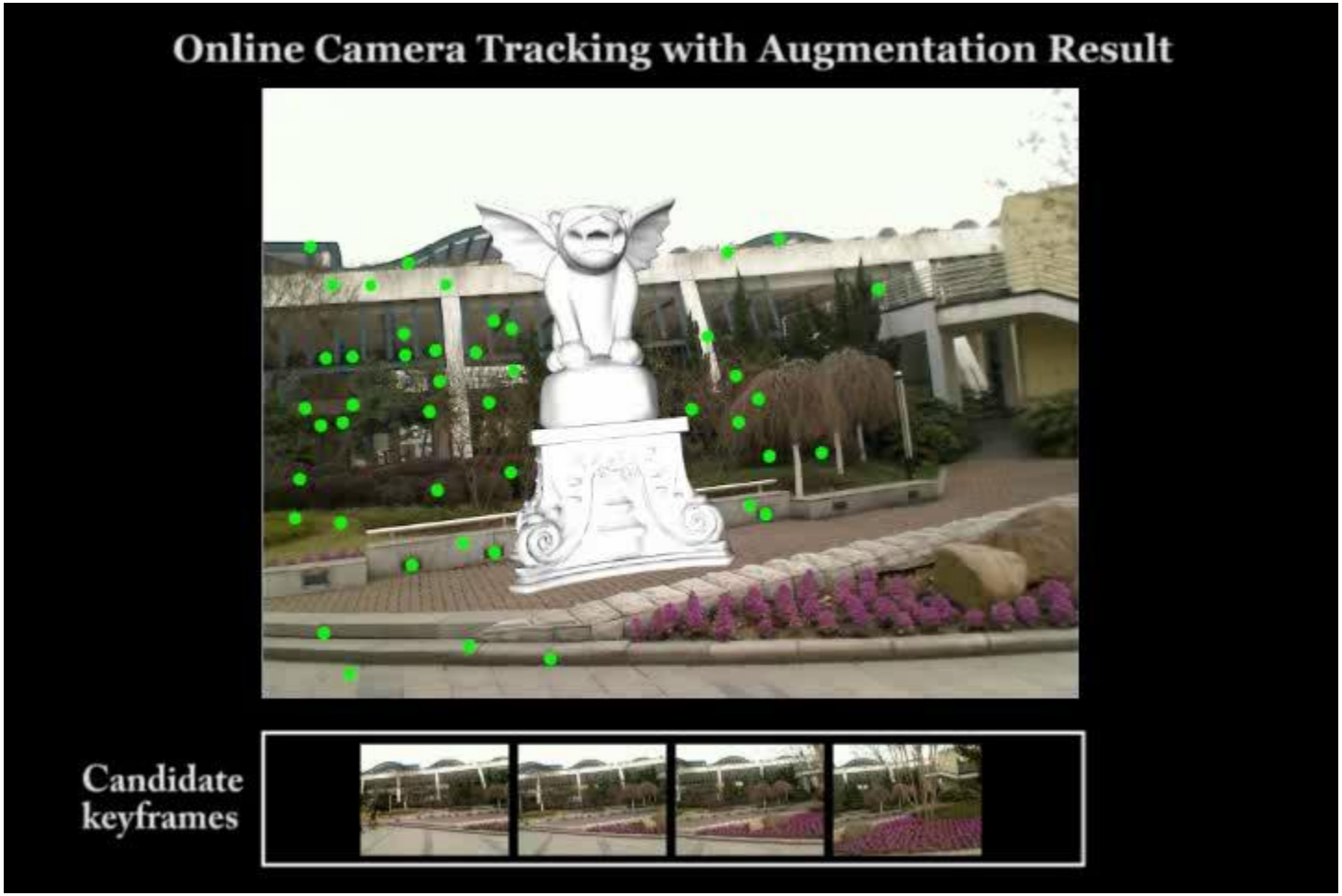
Keyframe-based matching

System timing

- Intel(R) Core(TM)2 Quad CPU Q9550 @ 2.83GHz

Module	**Average time** $[ms]$
Feature extraction (320×240)	23
Keyframe recognition	2
Keyframe-based matching	$4 \times \mathcal{K}$
Camera pose estimation	5

Augmented result



Thank you!