Visual SLAM:
a tour from sparse to dense

Zhaoyang Lv

Advised by Prof. Frank Dellaert
1st year PhD in Robotics
Interactive Computing

OUTLINE

- Introduction to visual SLAM
- Behind front-end tracking and back-end mapping
- Sparse tracking and mapping
- Dense Tracking and mapping
- Challenges and Opportunities

SLAM in General

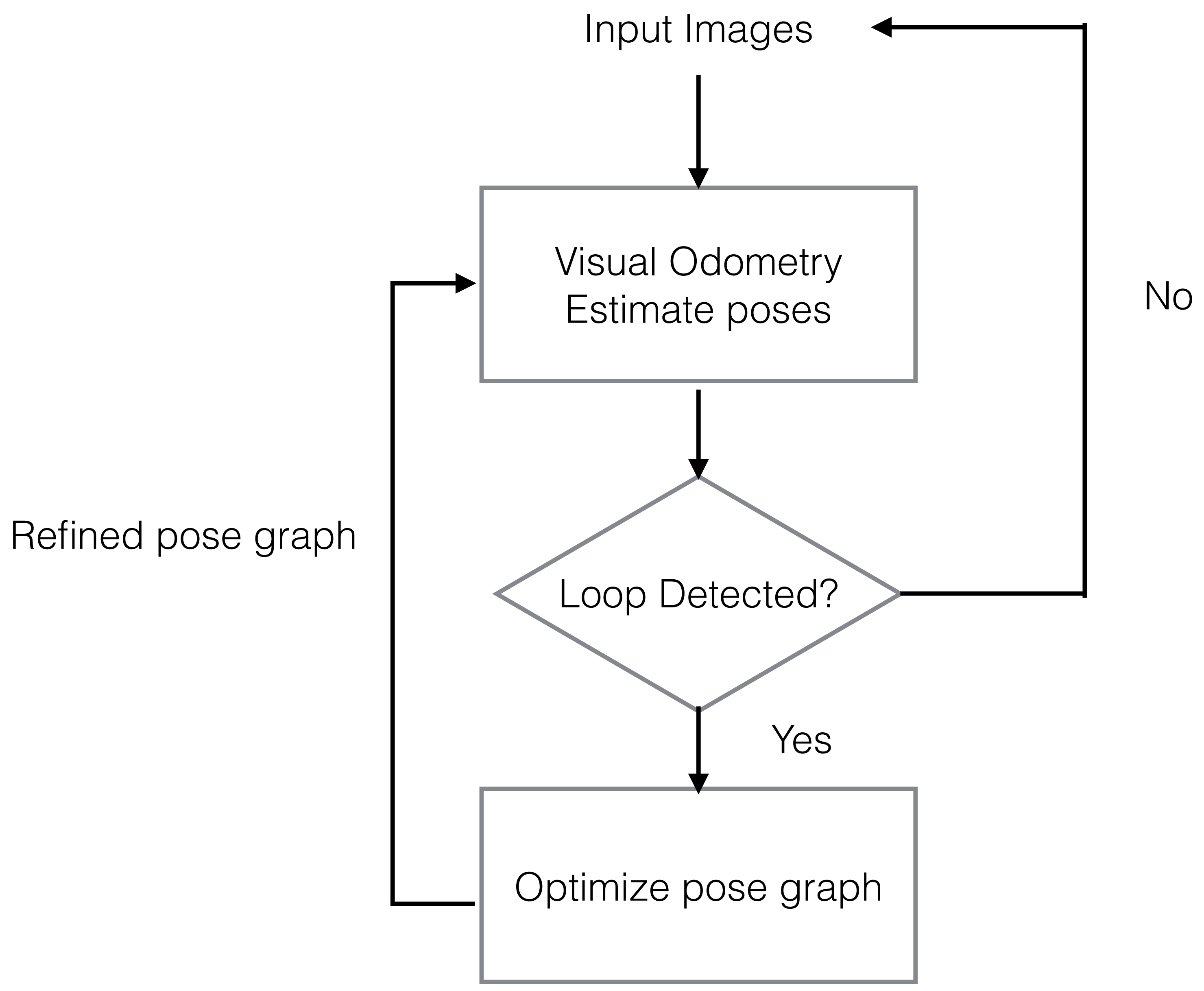
What is SLAM?
Simultaneous Localization and Mapping

- **A general problem:** A robot with quantitative sensors, navigating in a previously unknown environment, mapping the environment and calculate its ego-motions.

- Make it simple: estimate the **robot poses,** and meanwhile **map the scene**.

- In visual SLAM, the only sensor we use is camera.

General Visual SLAM pipeline



Visual Odometry

- We want to estimate 6-DoF camera pose [**R**|**T**] incrementally
- Recall what we have in two-view geometry class. We use epipolar geometry to estimate relative camera motions:

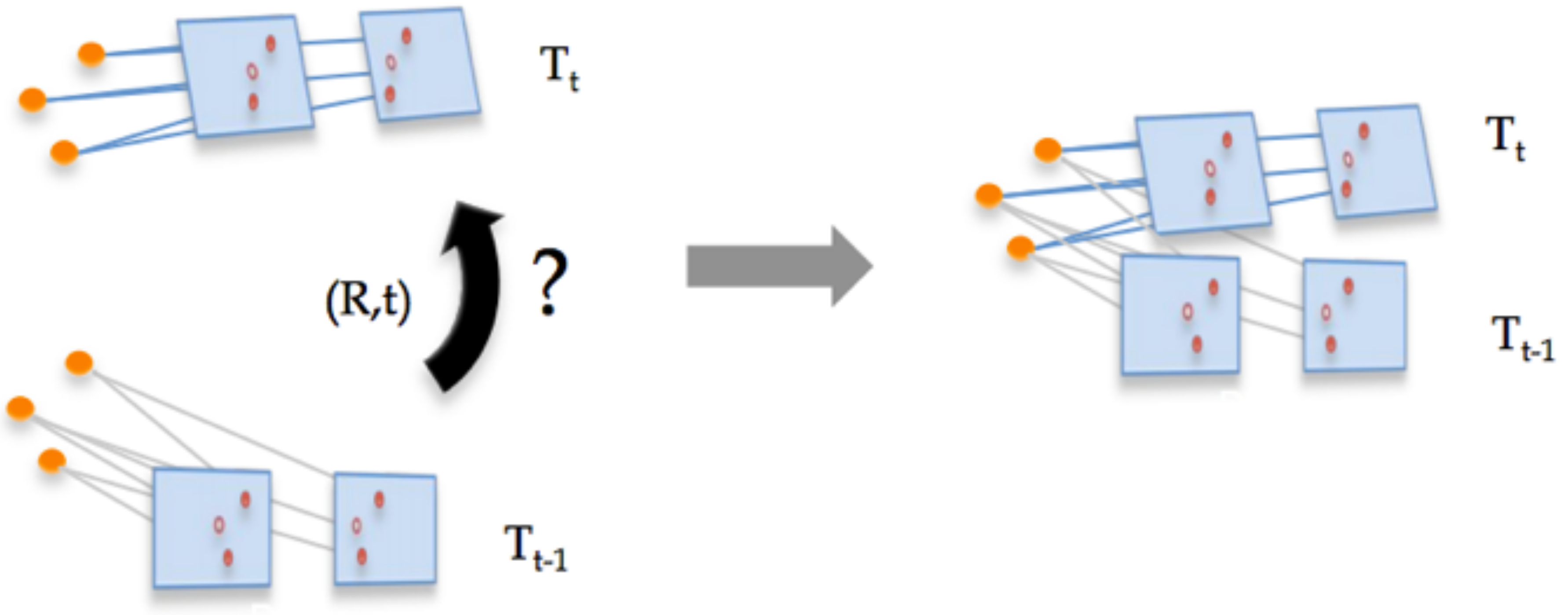
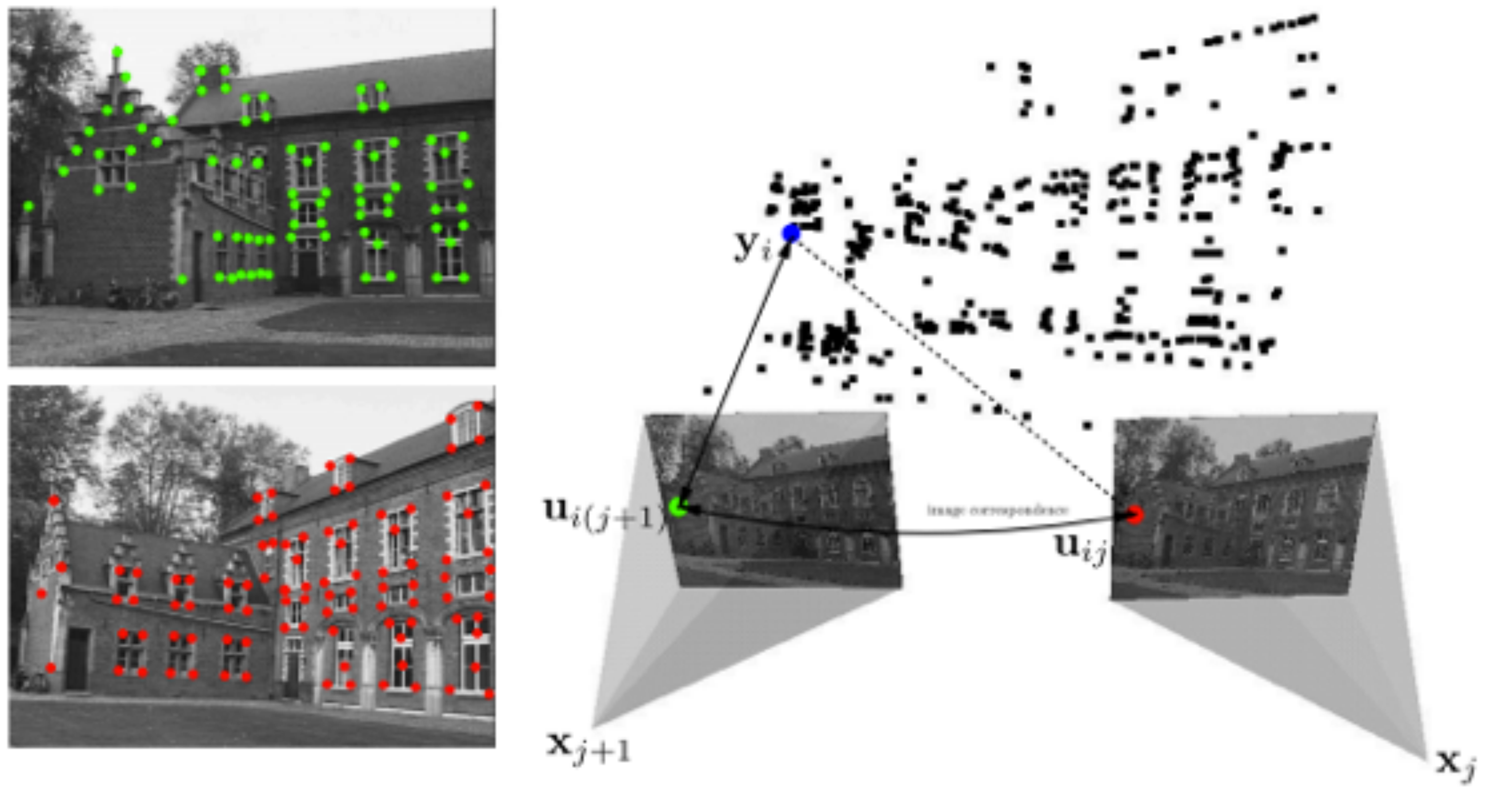


Image from [C.Beall, Stereo VO, CVPR'14 workshop]

Visual Odometry

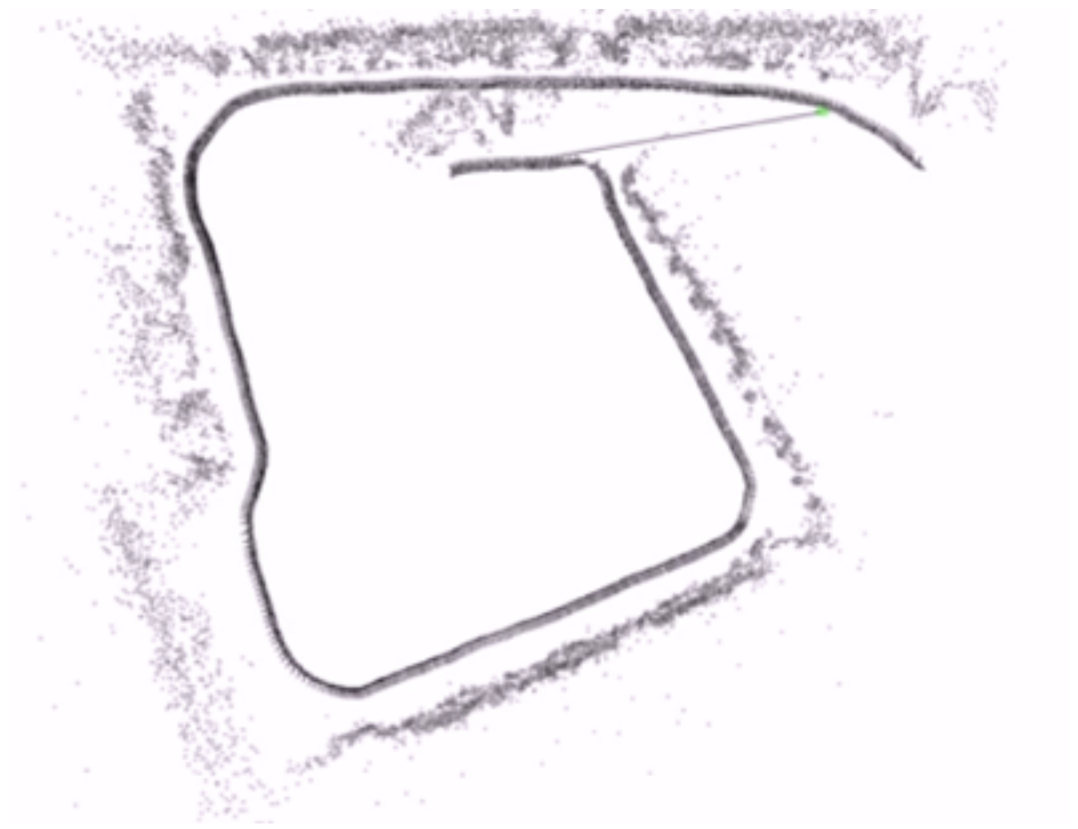
- For monocular camera, triangulation can be done in consecutive frames.

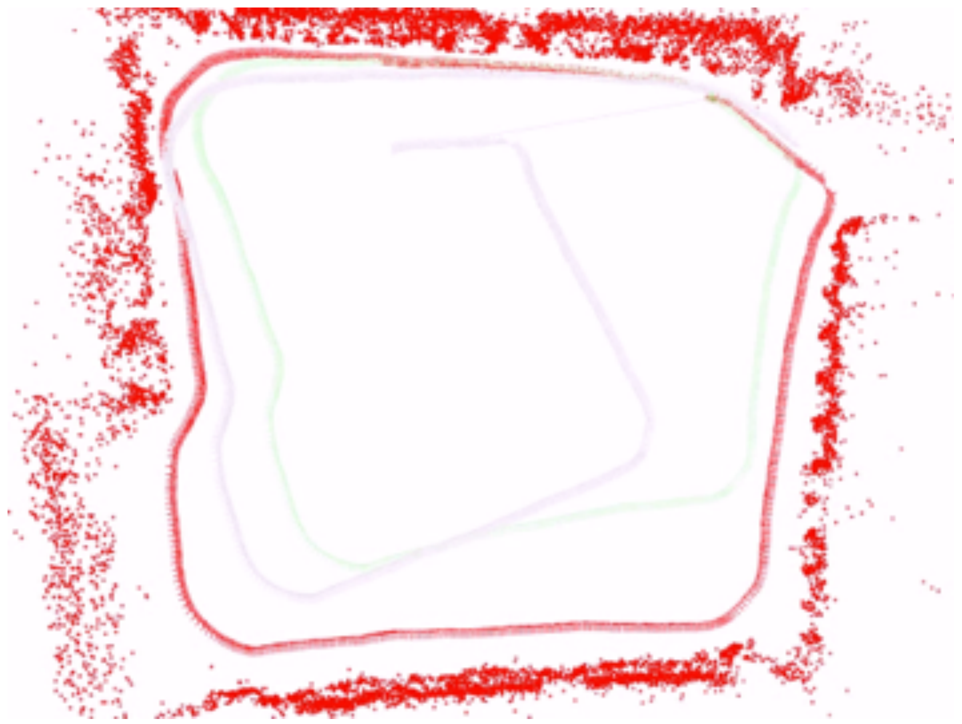


[M.Pollefeys, Hand-held acquisition of 3D models with a video camera., 1999, 3DIM]

Graph Relaxation

- The robot localization is a belief propagation.
- Uncertainty increases! Odometry will drift!
- How do we solve it?



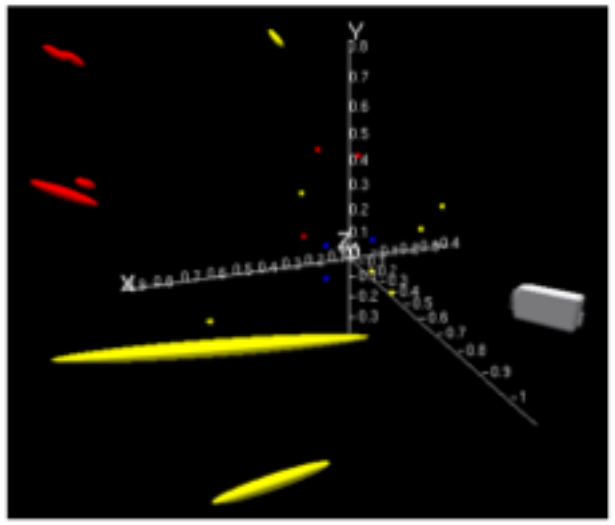


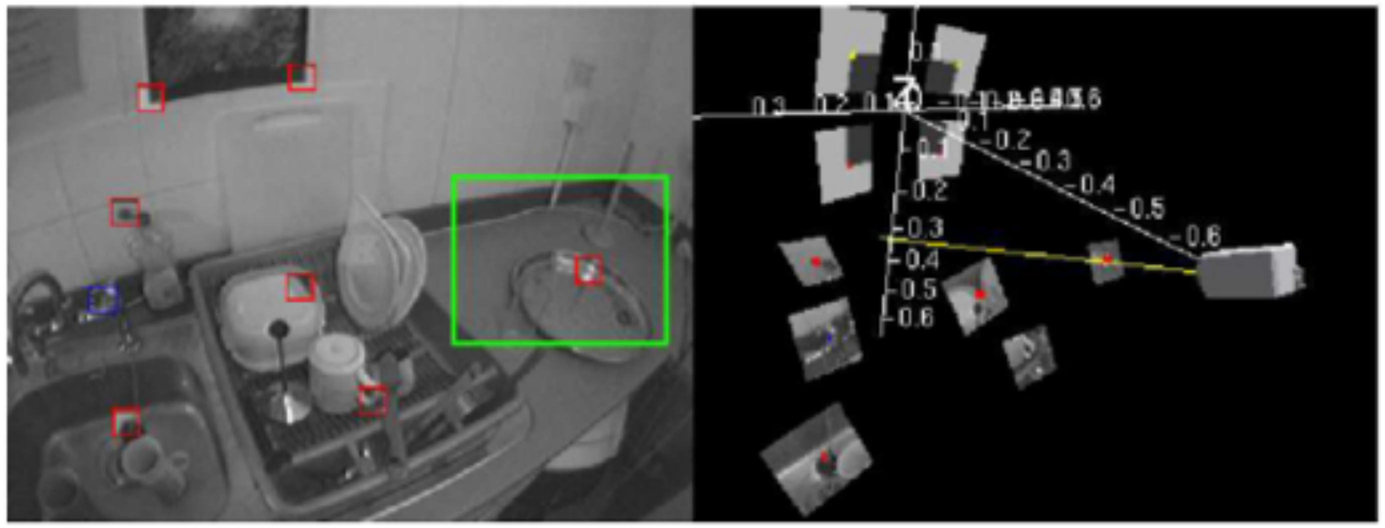
[H.Strasdat, et al. Scale Drift-Aware Large Scale Monocular SLAM, 2010, RSS]

Belief Propagation using Filtering

- Extended Kalman Filter model.
- States including both camera motions and features.

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_v \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix}, \quad \mathrm{P} = \begin{bmatrix} \mathrm{P}_{xx} & \mathrm{P}_{xy_1} & \mathrm{P}_{xy_2} & \cdots \\ \mathrm{P}_{y_1x} & \mathrm{P}_{y_1y_1} & \mathrm{P}_{y_1y_2} & \cdots \\ \mathrm{P}_{y_2x} & \mathrm{P}_{y_2y_1} & \mathrm{P}_{y_2y_2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}.$$





[A.Davison et al. MonoSLAM: real-time single camera SLAM, 2007, PAMI]

Belief Propagation using Filtering — Good?

- Tracking and mapping here are coupled at each frame. Work well at high frame rates, but may loose robustness when frame-rate drop.
- Either tracking or mapping fail, the system fails.
- Number of landmarks are limited. Unpractical in a large-scale environment. Fatal!

Can we do better?
Recap: Structure from Motion

- **What we have:** incremental increasing images.
- **What's the same:** estimate the 3D structure from the image, and meanwhile get their camera poses.
- **What's different:** we need a real-time incremental pose estimation. (Or maybe not! Some people think they are the same!)

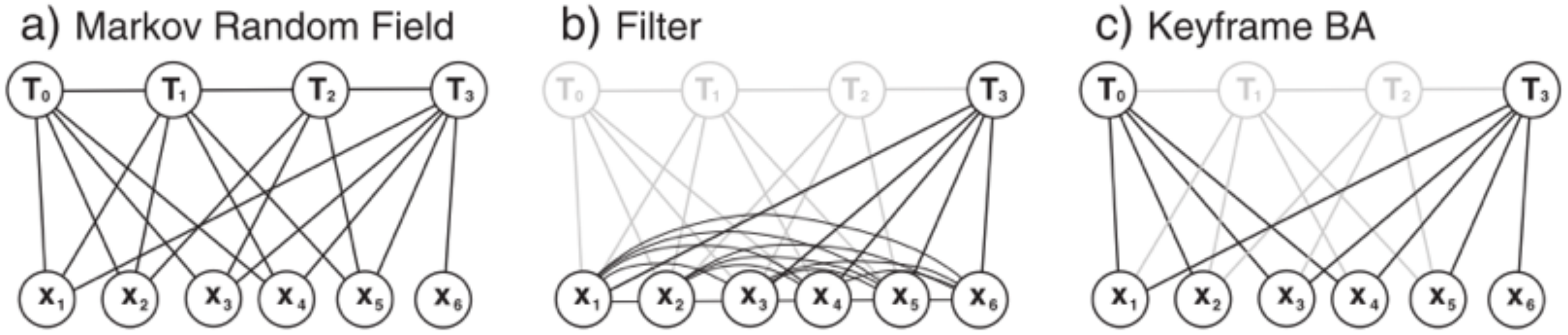
Photo Tourism
[Snavely et al. SIGGRAPH'06]

Build Rome in a Day
[Agarwal et al. ICCV'09]

Optimization: Bundle Adjustment

- Good initial pose estimation? Yes, from the multi-view geometry! Then optimize it.
- Too many frames is a pain! Select **key frames** and set up correspondence.
- We can either do it **locally**, eg. 5 local key frames (refinement to odoemtry), or **globally** (refinement to whole pose graph).
- The same model, as a minimization problem:

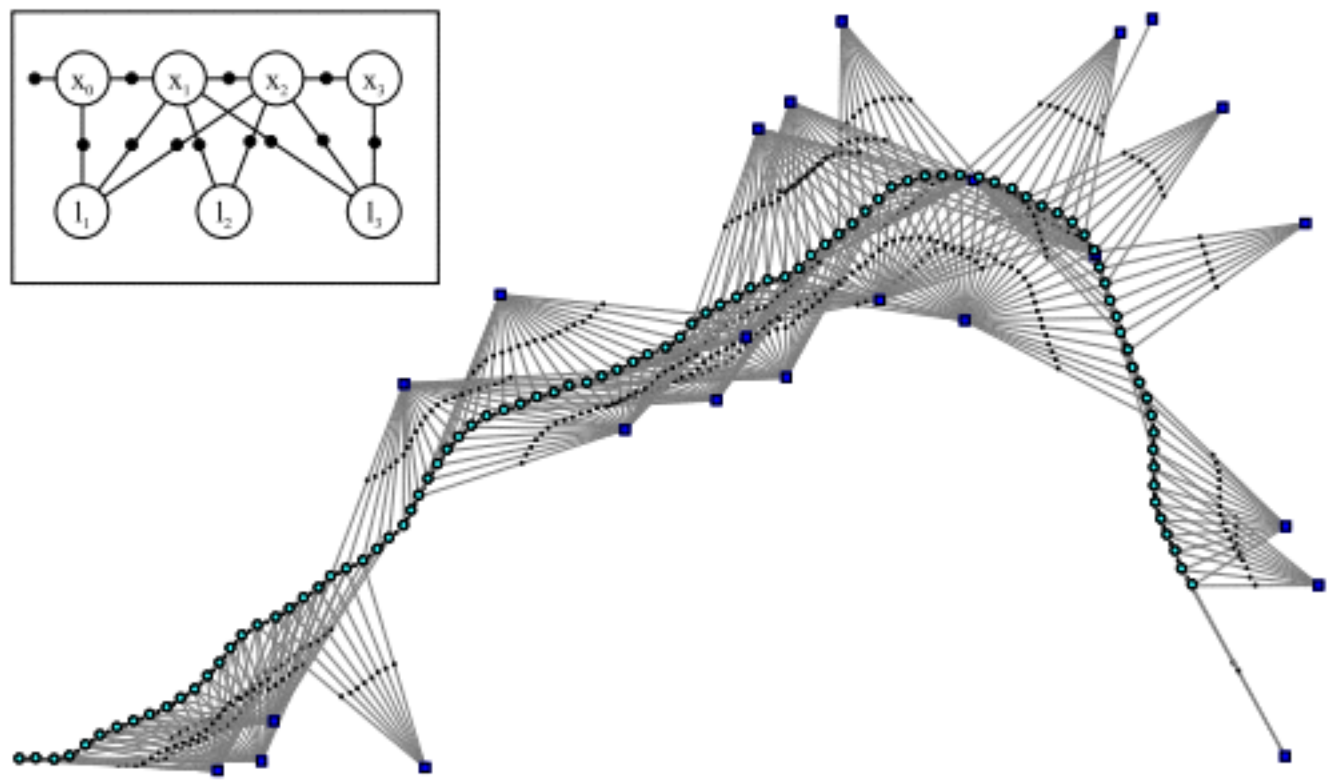
$$\min_{\mathbf{a}_j, \mathbf{b}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} d(\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i), \; \mathbf{x}_{ij})^2,$$



[H.Strasdat et al.Visual SLAM: Why Filter?, 2010, ICRA]

SLAM as Factor Graph

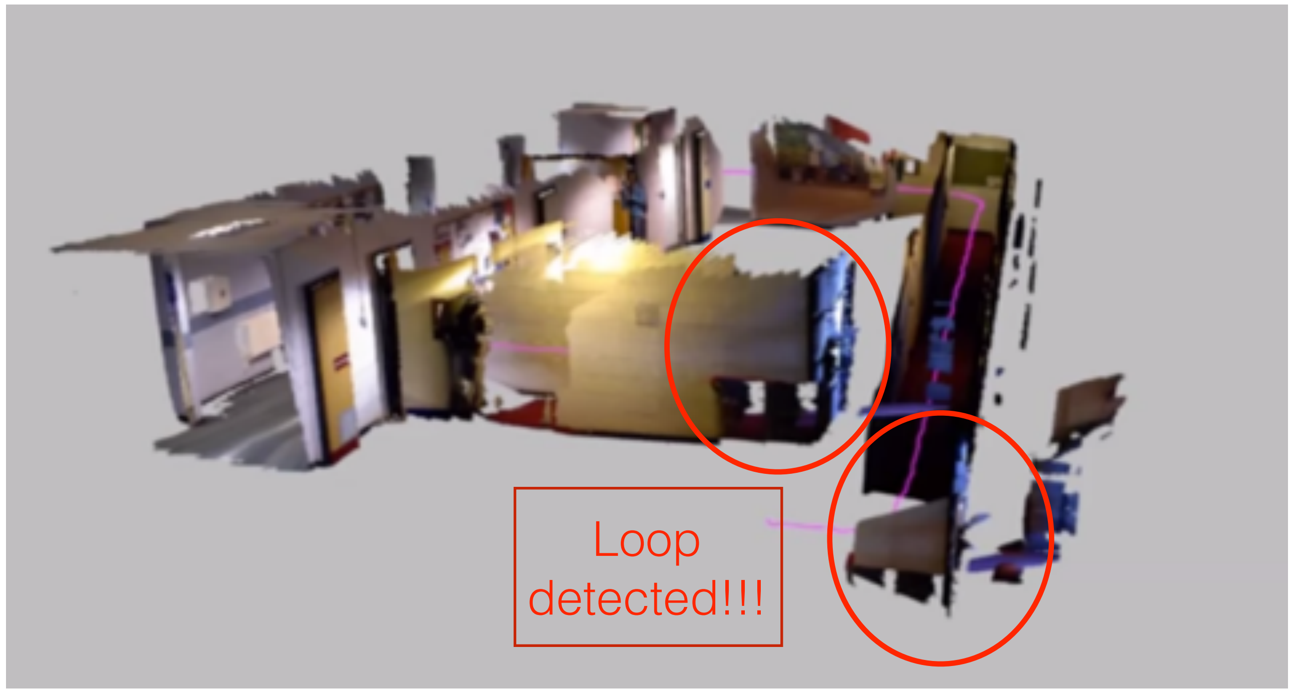
- A more power representation for SLAM.
- **A pure optimization problem:** Inference? Factorize the matrix? Eliminate the number of factors? Global Optimal?
- **Use GTSAM to solve it!**

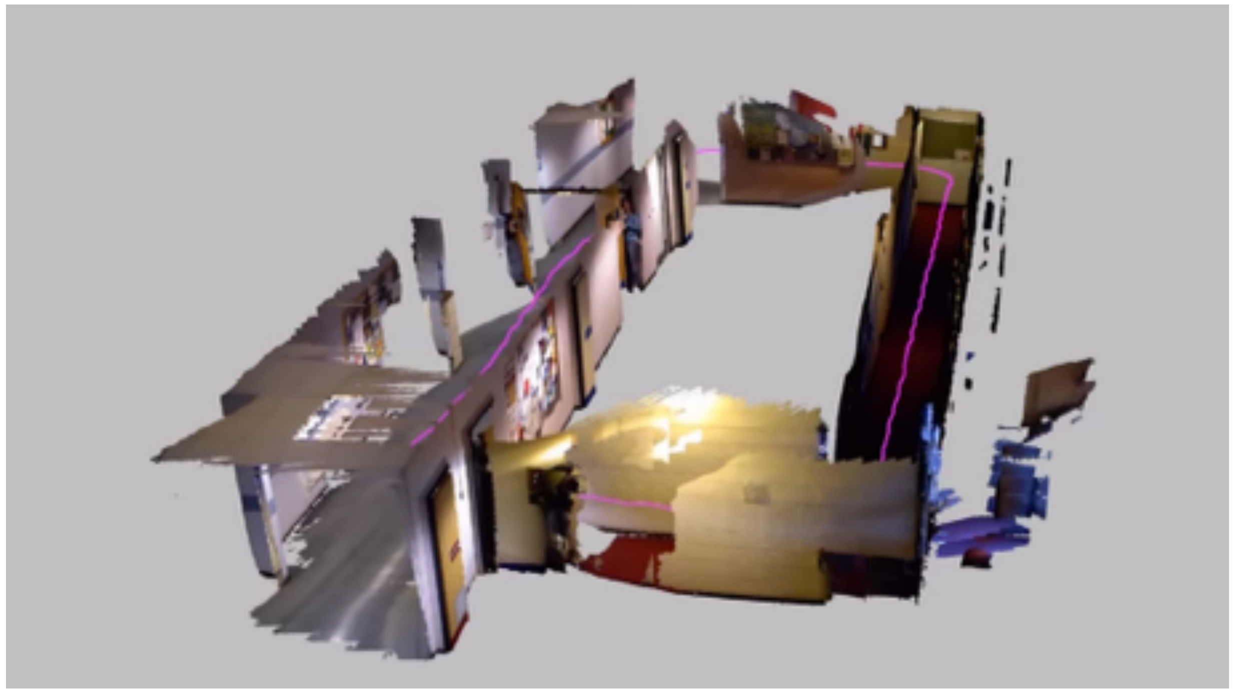


[F.Dellaert et al. Square Root SAM Simultaneous Localization and Mapping via Square Root Information Smoothing, 2006, IJRR]

What's in between — Loop Detection

- Optimization works the best if we have global correspondence from landmarks.
- When a loop is detected, we can set up the global correspondence.

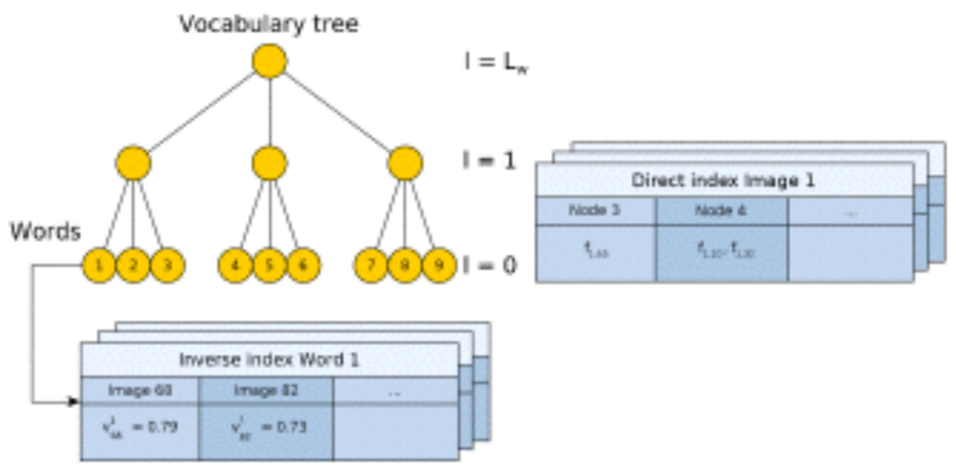


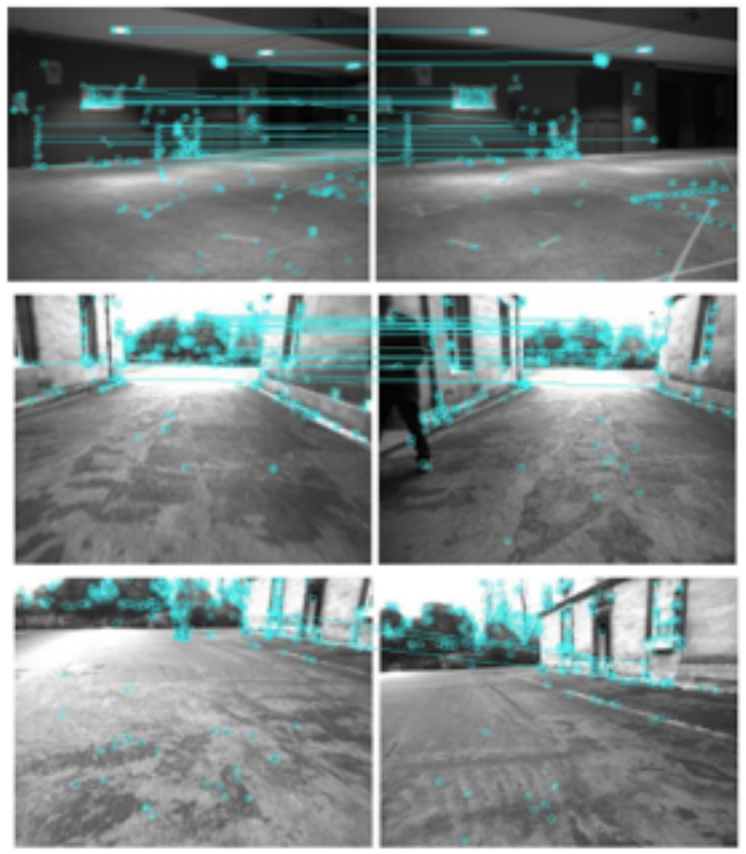


[T.Whelan et al. Deformation-based Loop Closure for Large Scale Dense RGB-D SLAM, 2013, IROS]

What’s in between — Loop Detection

- A loop detection is an image-matching step: detected the visited the image content and estimate its relative pose.
- The number of features increase incrementally, especially in large-scale environment. How to manage?
- Bag-of-Words system, e.g. Vocabulary tree.

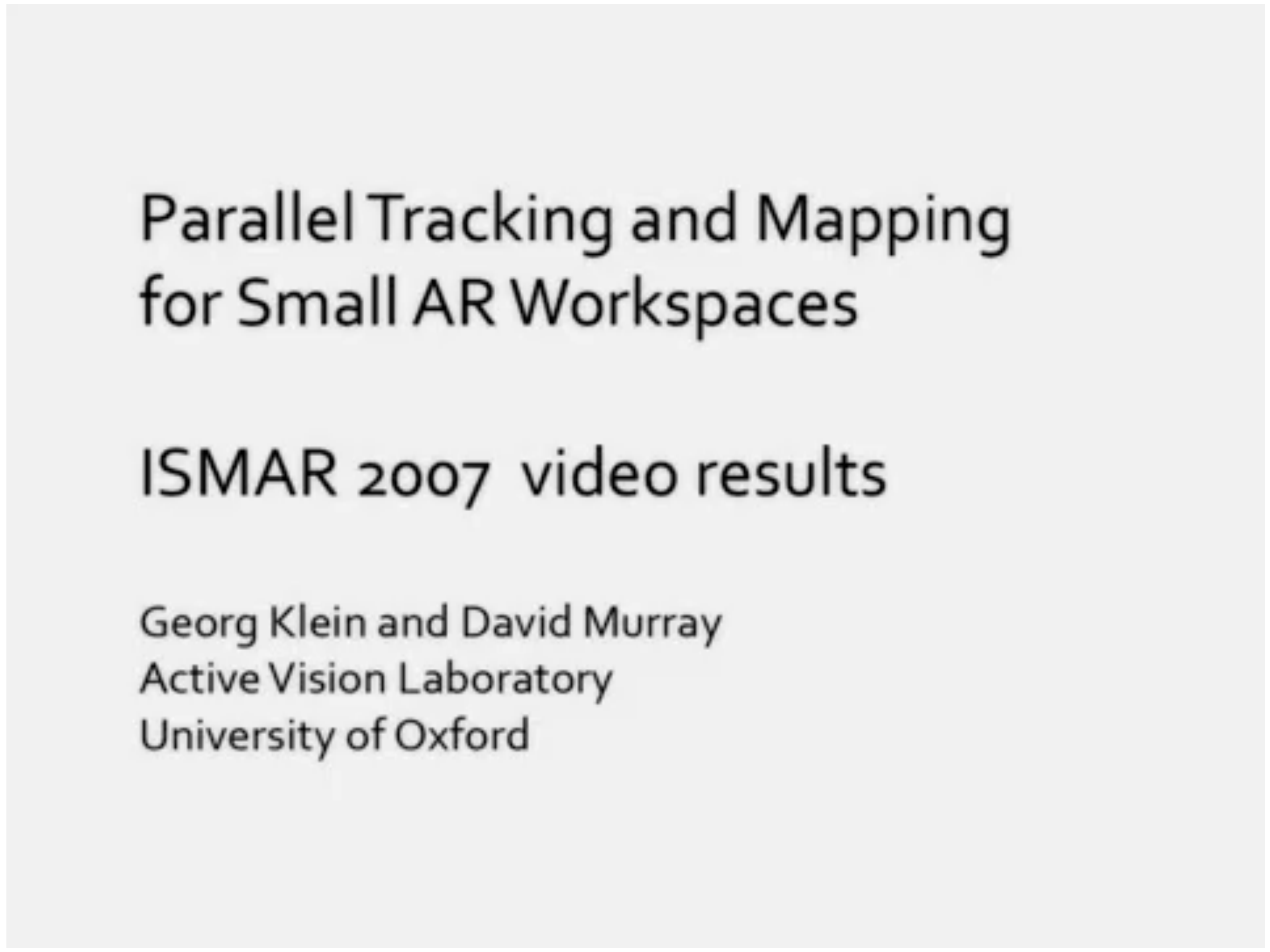




[D.Lopes et al. Bags of Binary Words for Fast Place Recognition in Image Sequences, 2012, TRO]

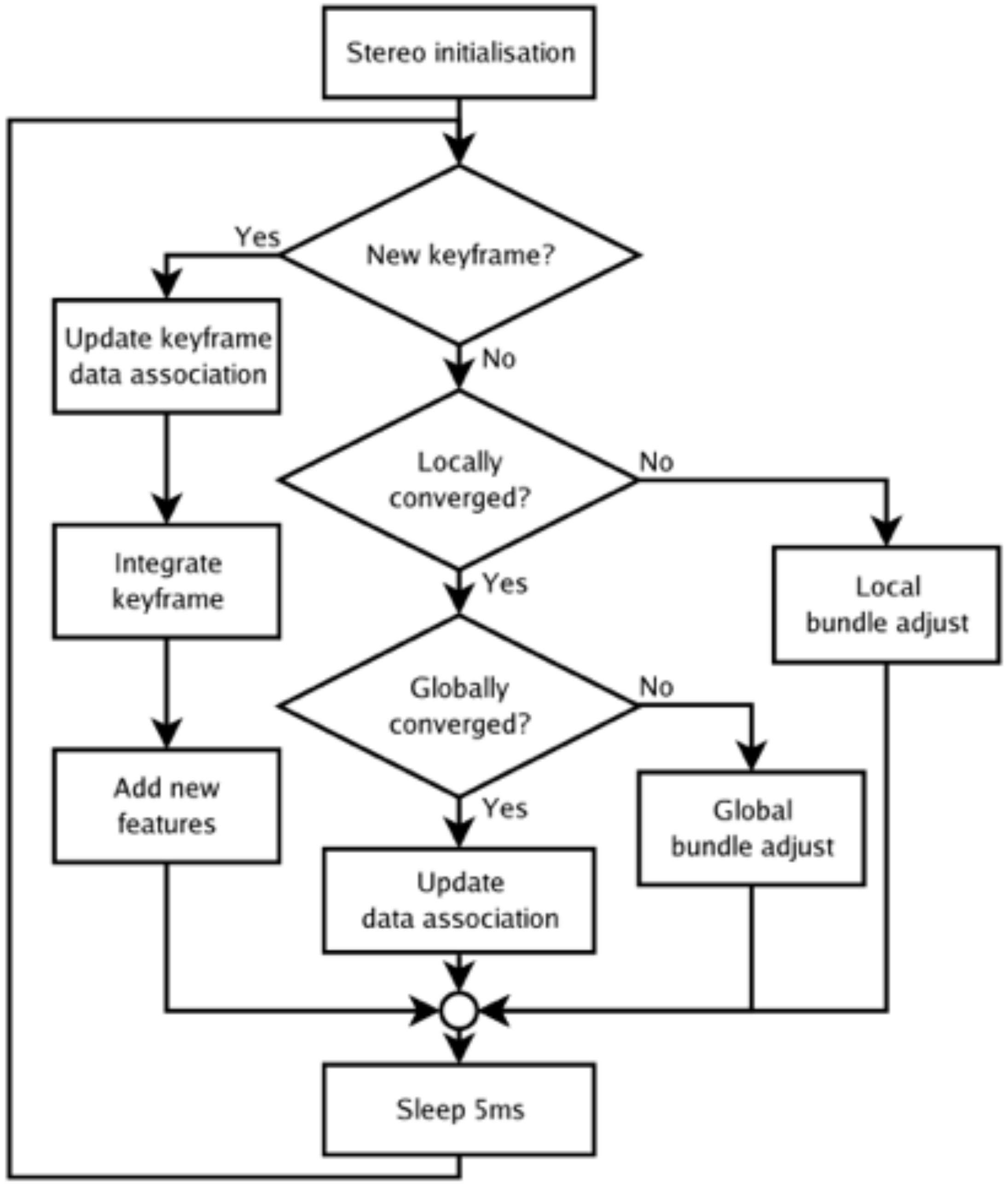
Sparse Tracking and Mapping

PTAM: Parallel Tracking and Mapping for Small AR Workspaces


Parallel Tracking and Mapping for Small AR Workspaces

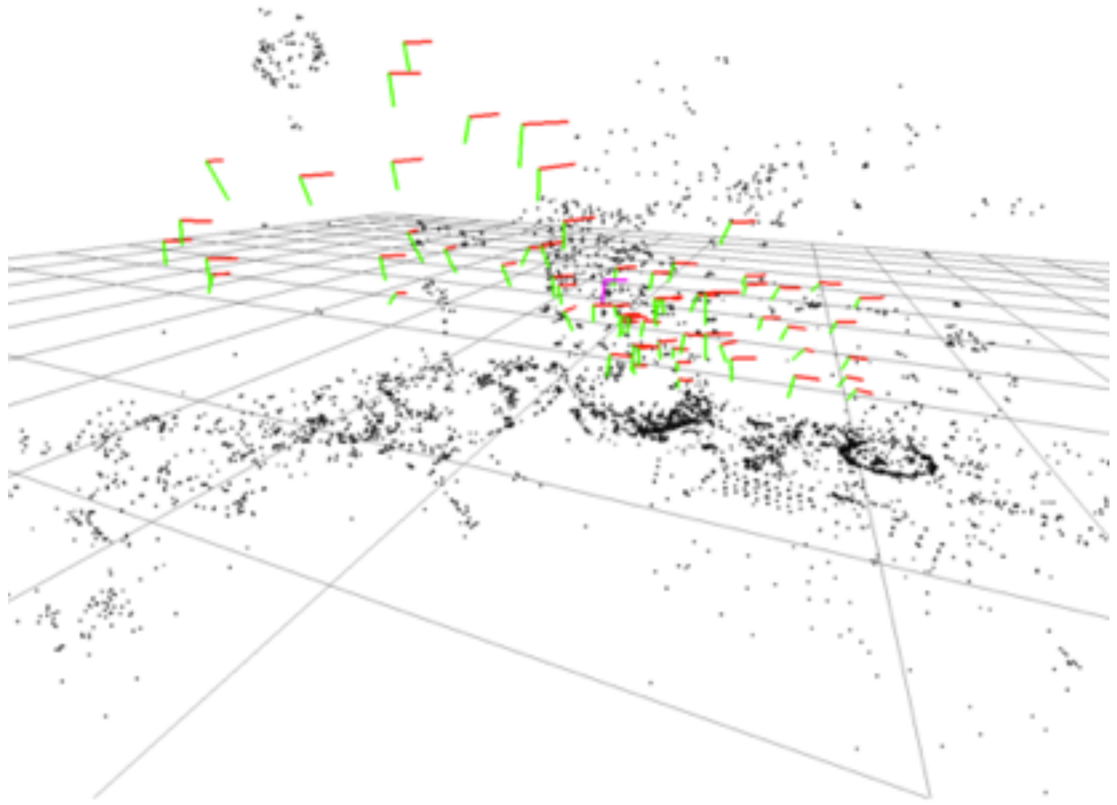
ISMAR 2007 video results

Georg Klein and David Murray
Active Vision Laboratory
University of Oxford



PTAM: Parallel Tracking and Mapping for Small AR Workspaces

- Tracking and Mapping are in two parallel threads.
- Tracking is frame-to-model, against the point clouds in the world.
- The global point clouds are initialized with epipolar geometry.
- Once we found feature points in key frames are not in the global frame, add them into the global point clouds.



PTAM: Parallel Tracking and Mapping for Small AR Workspaces

- Tracking is a two-step process:
  - A coarse-to-fine feature matching to estimate a initial camera pose;
  - Use local bundle adjustment to refine the pose.
- Project the global points to the image frame (eg. five frames). Minimize this error：

$$\{\{\boldsymbol{\mu}_{x\in X}\}, \{\boldsymbol{p}'_{z\in Z}\}\} = \underset{\{\{\boldsymbol{\mu}\},\{\boldsymbol{p}\}\}}{\operatorname{argmin}} \sum_{i\in X\cup Y} \sum_{j\in Z\cap S_i} \operatorname{Obj}(i,j).$$

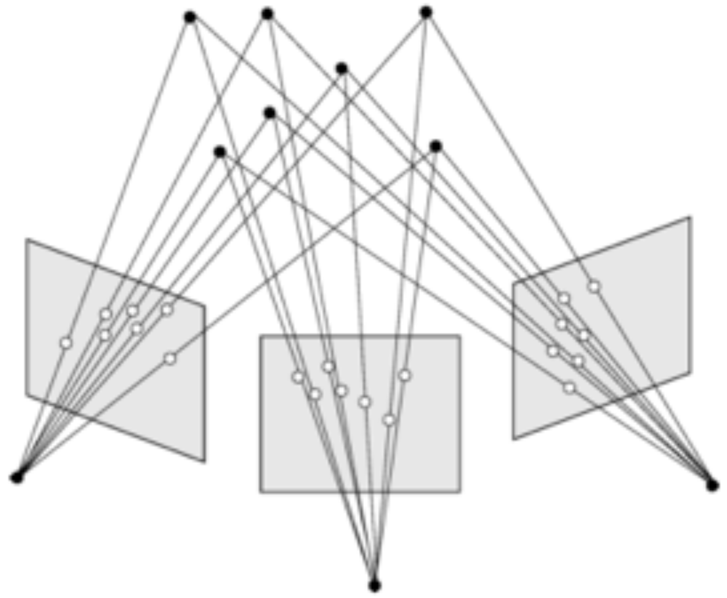
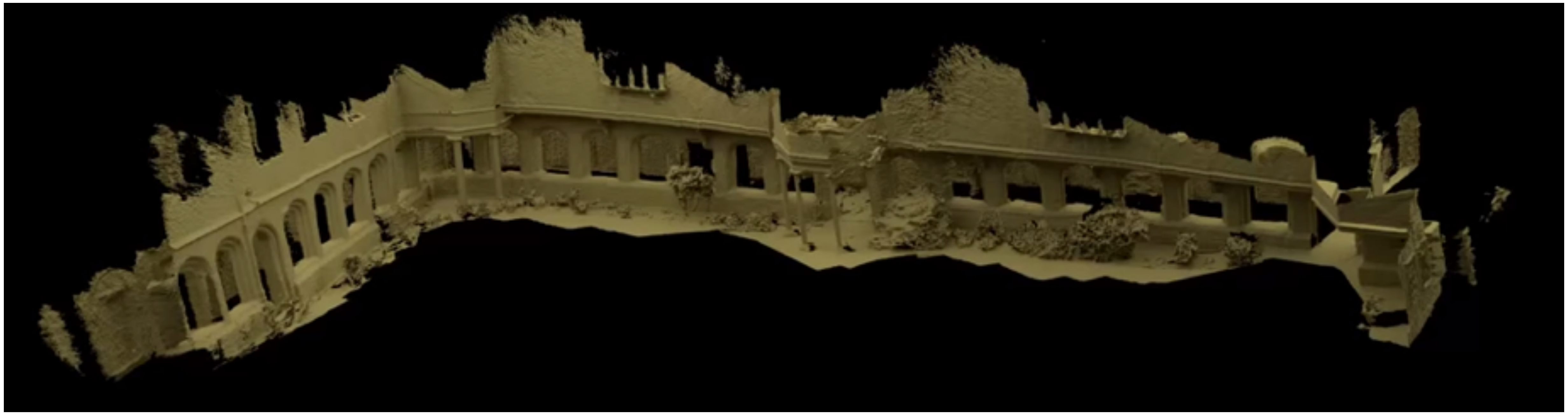


Image from [M.Lourakis et al,SBA: A Software Package for Generic Sparse Bundle Adjustment, 2009, MS]

Dense Tracking and Mapping

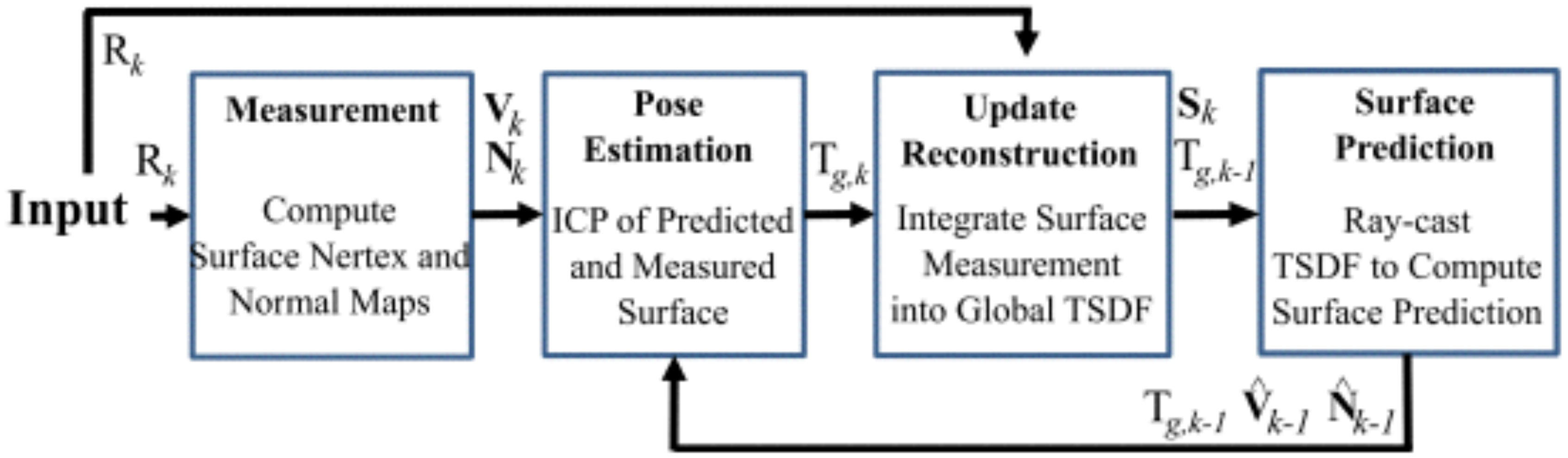
Why Dense Mapping?

- Tracking from frame-to-model: more dense scene, more information for tracking.
- Robot can approach perception at human-level.
- The secrets: RGB-D cameras and GPGPU!

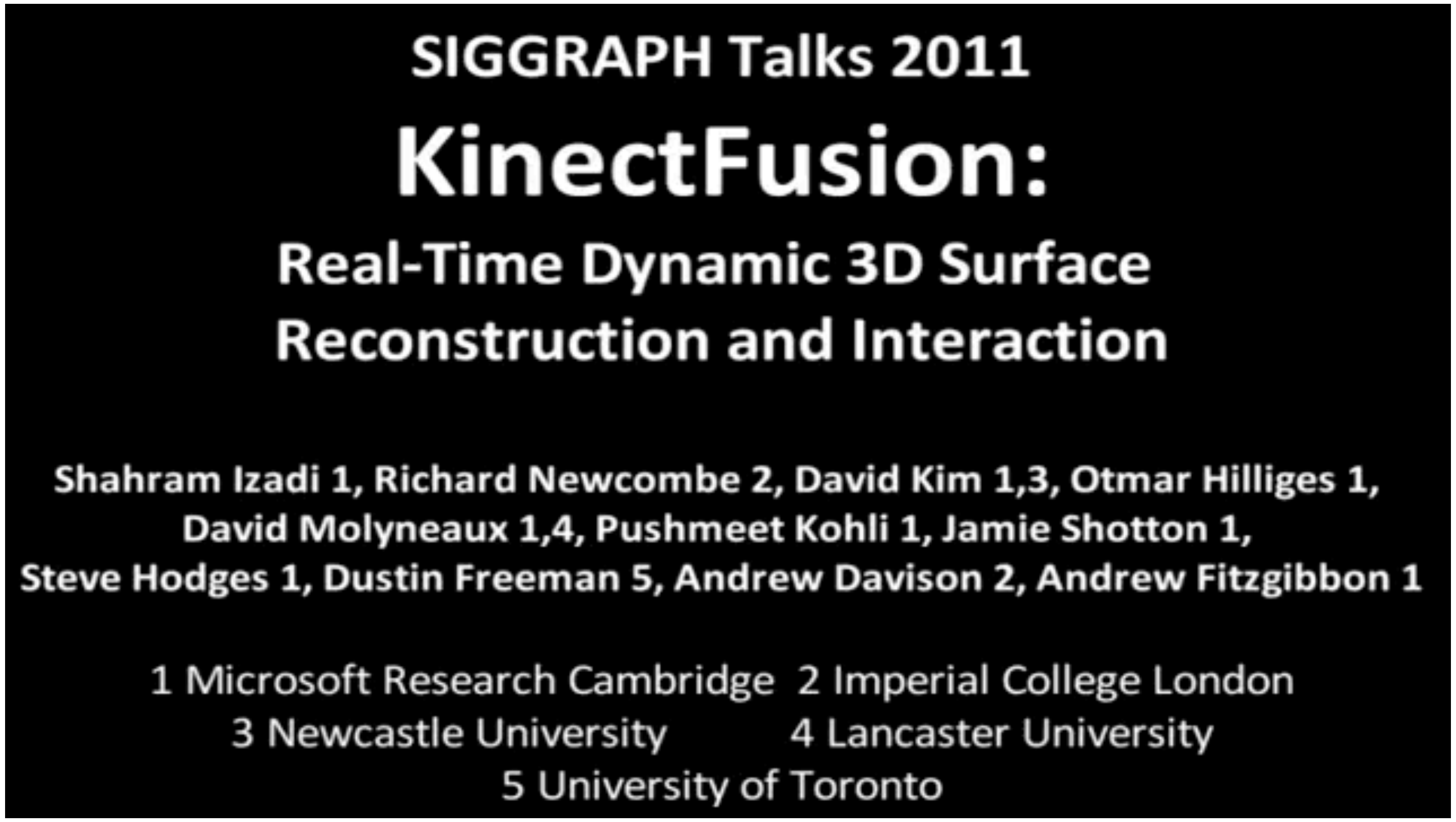


[M.NieBner et al, Real-time 3D Reconstruction at Scale Using Voxel Hashing, 2013, SIGGRAPH]

KinectFusion: Real-time Dense Surface Mapping and Tracking



R_k
Input
R_k
Measurement
Compute Surface Nertex and Normal Maps
V_k
N_k
Pose Estimation
ICP of Predicted and Measured Surface
$T_{g,k}$
Update Reconstruction
Integrate Surface Measurement into Global TSDF
S_k
$T_{g,k-1}$
Surface Prediction
Ray-cast TSDF to Compute Surface Prediction
$T_{g,k-1}$ $\hat{V}_{k-1}$ $\hat{N}_{k-1}$



SIGGRAPH Talks 2011

KinectFusion:

Real-Time Dynamic 3D Surface Reconstruction and Interaction

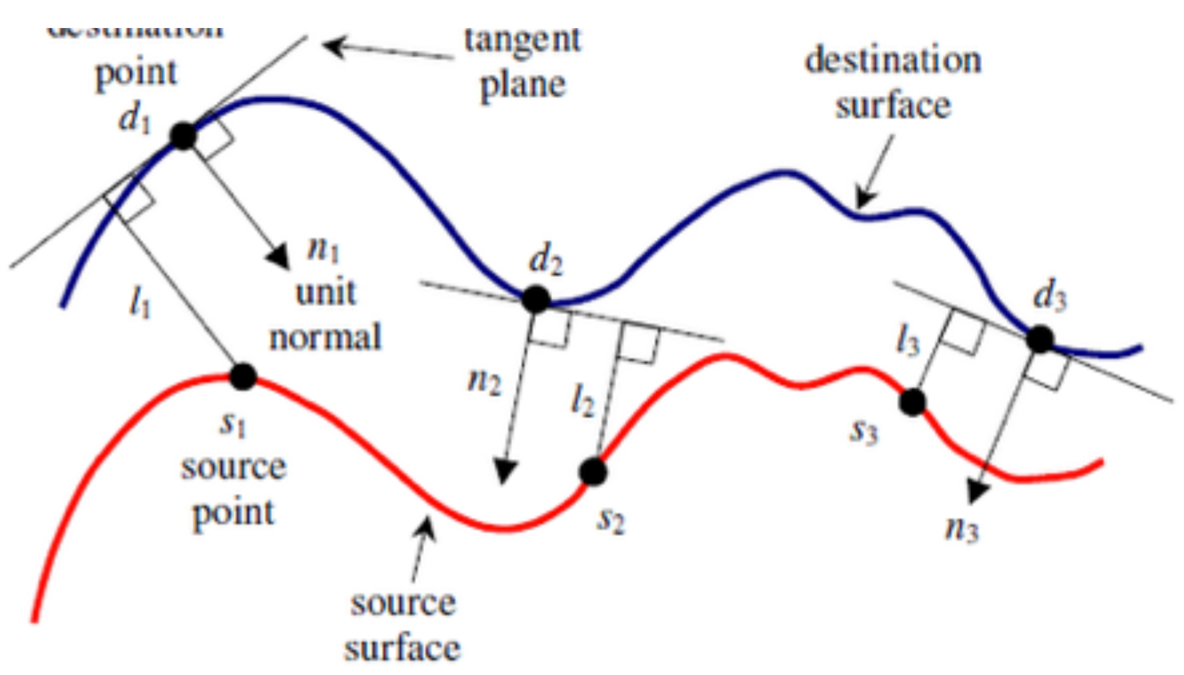
Shahram Izadi 1, Richard Newcombe 2, David Kim 1,3, Otmar Hilliges 1, David Molyneaux 1,4, Pushmeet Kohli 1, Jamie Shotton 1, Steve Hodges 1, Dustin Freeman 5, Andrew Davison 2, Andrew Fitzgibbon 1

1 Microsoft Research Cambridge 2 Imperial College London
3 Newcastle University 4 Lancaster University
5 University of Toronto

KinectFusion Pipelines
Depth Tracking: ICP matching

- What we have:
  - a global surface model;
  - a local surface model, generated by input point clouds;
  - an estimated camera pose from last frame, a good initial estimation when movement is small.
- Utilising these, we optimize the global point-plane energy to estimate the camera pose:

$$\mathbf{E}(\mathrm{T}_{g,k}) = \sum_{\substack{\mathbf{u}\in\mathscr{U}\\ \Omega_k(\mathbf{u})\neq\text{null}}} \left\| \left(\mathrm{T}_{g,k}\dot{\mathbf{V}}_k(\mathbf{u}) - \hat{\mathbf{V}}^g_{k-1}(\hat{\mathbf{u}})\right)^{\top} \hat{\mathbf{N}}^g_{k-1}(\hat{\mathbf{u}}) \right\|_2$$

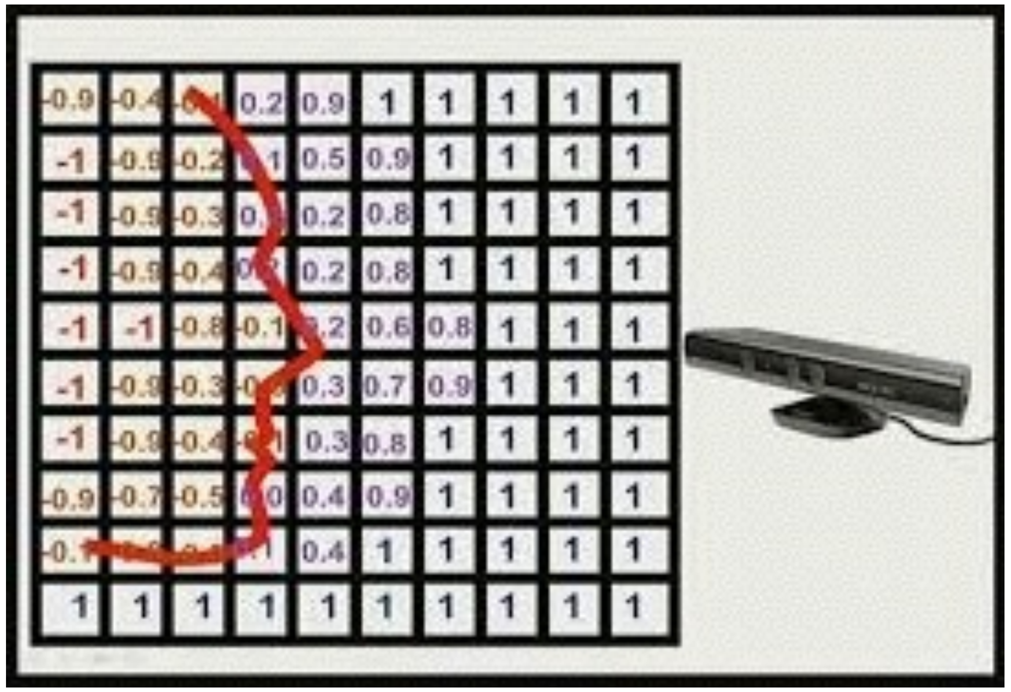


[K.Low, Linear Least-Squares Optimization for
Point-to-Plane ICP Surface Registration, 2003, Tech Report]

KinectFusion Pipelines
Fuse the Model: TSDF Volume

- Each Voxel stores a pair of values, signed distance value (SDF) F, and weight W: $\mathbf{S}_k(\mathbf{p}) \mapsto [\mathrm{F}_k(\mathbf{p}), \mathrm{W}_k(\mathbf{p})]$.
- SDF value represents the uncertainty around the surface
    - 0 means the voxel lies on the surface
    - 1 means empty space, -1 means behind the surface.
- At each frame, update the SDF value by its weight:

$$\begin{aligned}\mathrm{F}_k(\mathbf{p}) &= \frac{\mathrm{W}_{k-1}(\mathbf{p})\mathrm{F}_{k-1}(\mathbf{p}) + \mathrm{W}_{\mathrm{R}_k}(\mathbf{p})\mathrm{F}_{\mathrm{R}_k}(\mathbf{p})}{\mathrm{W}_{k-1}(\mathbf{p}) + \mathrm{W}_{\mathrm{R}_k}(\mathbf{p})} \\ \mathrm{W}_k(\mathbf{p}) &= \mathrm{W}_{k-1}(\mathbf{p}) + \mathrm{W}_{\mathrm{R}_k}(\mathbf{p})\end{aligned}$$



-0.9	-0.4	[illegible]	0.2	0.9	1	1	1	1	1
-1	-0.9	-0.2	[illegible]	0.5	0.9	1	1	1	1
-1	-0.9	-0.3	[illegible]	0.2	0.8	1	1	1	1
-1	-0.9	-0.4	[illegible]	0.2	0.8	1	1	1	1
-1	-1	-0.8	-0.1	[illegible]	0.6	0.8	1	1	1
-1	-0.9	-0.3	[illegible]	0.3	0.7	0.9	1	1	1
-1	-0.9	-0.4	[illegible]	0.3	0.8	1	1	1	1
-0.9	-0.7	-0.5	[illegible]	0.4	0.9	1	1	1	1
[illegible]	[illegible]	[illegible]	[illegible]	0.4	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1

Both images from [Point Cloud Library]

KinectFusion Pipelines
Get the 3D Global Map: Ray-casting

- Each pixel corresponds to a camera ray.
- The ray starts from the minimum depths for the pixel and stops at the zero crossings. (zero crossing is the surface).
- Voxel value might not correspond to exact zero. To find the exact value, we do interpolation.
- The output here is a 3D global surface. Use this to estimate the next frame camera pose.

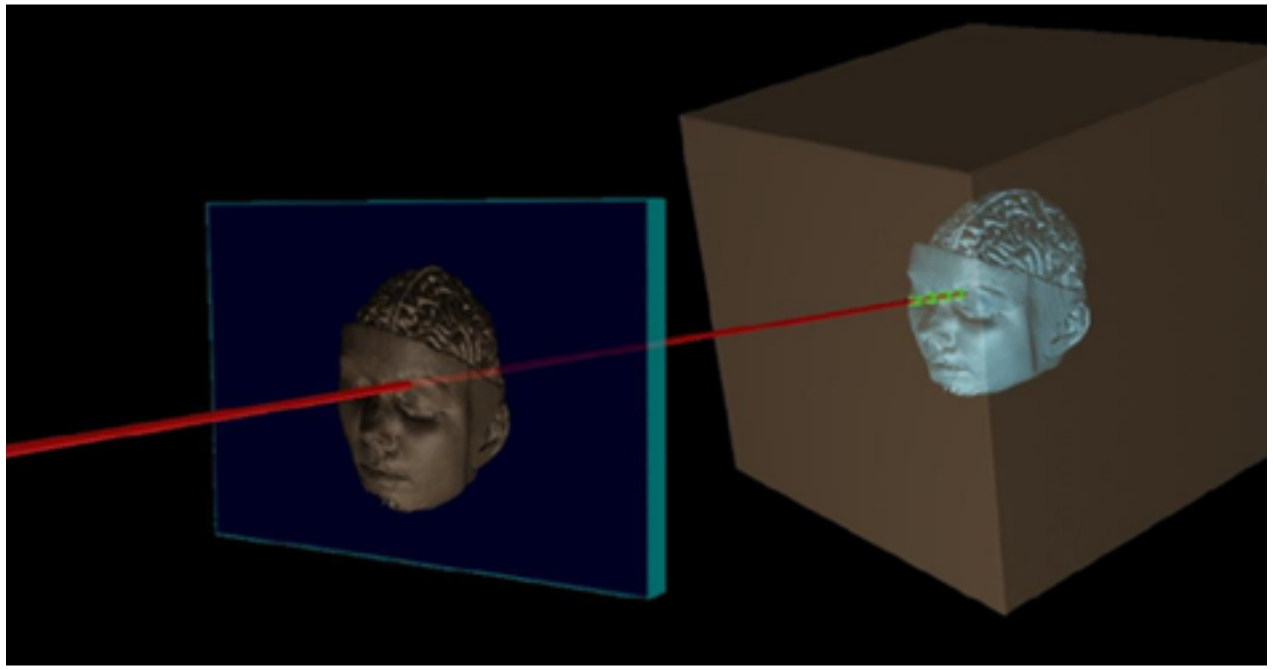
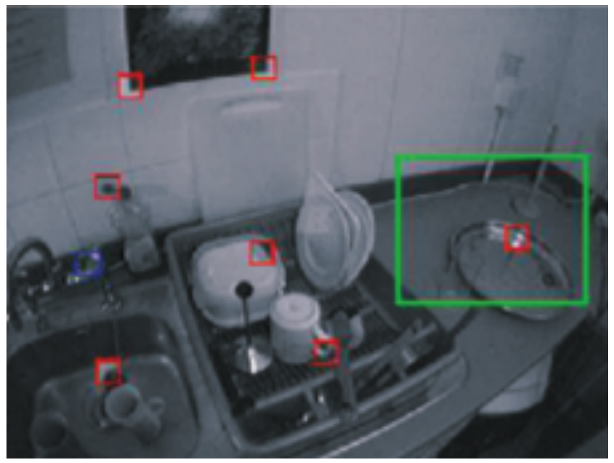
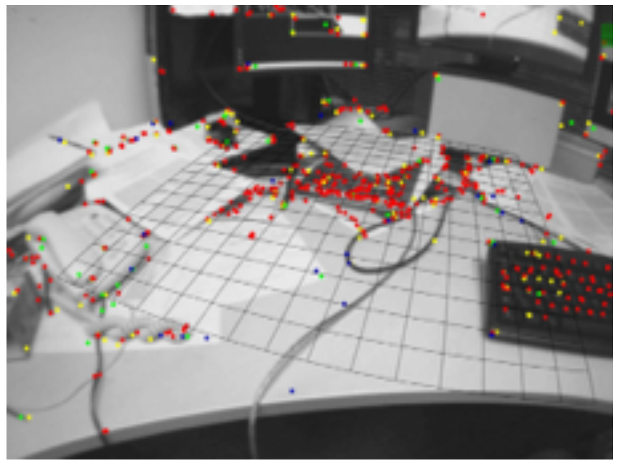


Image from [Volume Graphics Library http://vg.swan.ac.uk/gallery/]

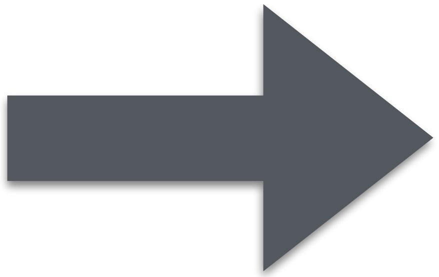
Challenges
&
Opportunities

What happens in the last decade

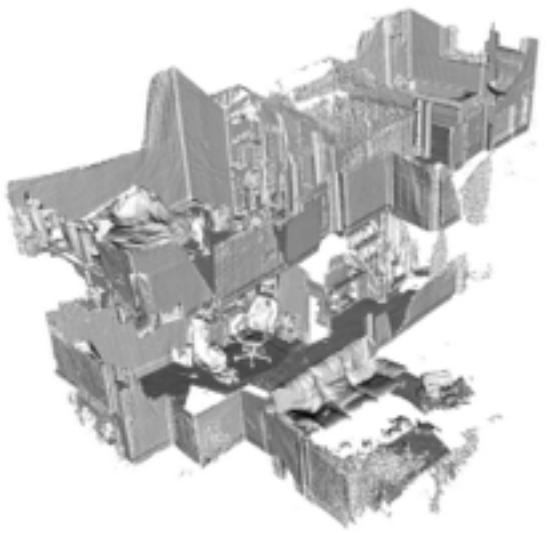

monoSLAM
[A.Davison, ICCV'03]

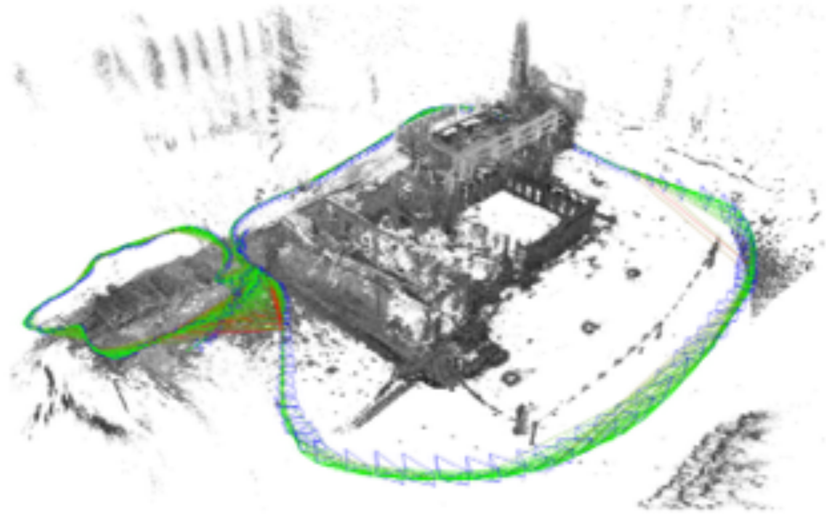

PTAM
[G.Klein et al. ISMAR'07]

More dense!
More accurate!
Larger!



KinectFusion
[R.Newcombe et al, ISMAR'11]


Kintinuous
[T.Whelan et al, MIT Tech'12]


LSD-SLAM
[J.Engel ECCV'14]

Challenges & Opportunities

- A more dense map in real-time? Approach real-time 3D reconstruction.
    - A more smooth mapping in large area?
    - Graph deformation is hard. The complexity grows!
- City-wide SLAM, Life-long SLAM, or season-wide SLAM?
    - Features are variant to scenes.
    - How to manage millions of features? How to fast retrieve?
- SLAM assumes static scenes. How to tackle with a dynamic environment?
    - Remove outliers?
    - Could we also track the object and the scene?
    - My previous work (youtube demo): https://www.youtube.com/watch?v=gphF88LtHuM&list=UUx-DbdC03CZiQqMhtliAwhg
- SLAM is perception. How perception can help learning and planning?
    - Sematic SLAM: understanding the scene from segmentation, clustering, etc.
    - Distributed SLAM: map the environment from multiple robots.
    - Active SLAM: actively select the landmarks, active tracking etc.

For more visual SLAM tutorials:

http://www.cc.gatech.edu/~dellaert/FrankDellaert/Frank_Dellaert/Entries/2014/6/28_Visual_SLAM_Tutorial_at_CVPR.html

or Come to BORG lab to discuss!