Dense Visual SLAM: Greedy Algorithms

Richard Newcombe

June 28, 2014

Outline

What is Dense Visual SLAM?

- We are interested in modelling geometry of a scene

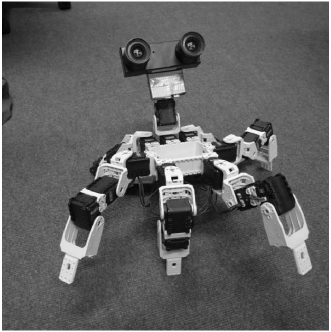
[Noah Snavely, Steven M. Seitz, Richard Szelisk Sigraph 2006]

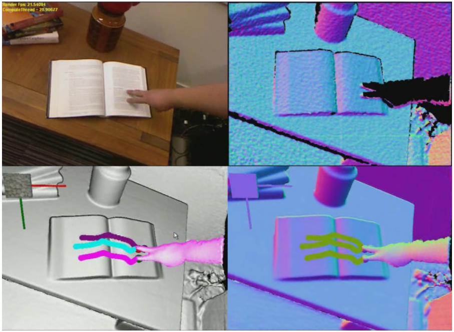
- *However* we want **surfaces**: not just **sparse point geometry**
- *And* we want it to be **causally estimated** in real-time
- not after all data has been collected after many hours.

Dense SLAM Motivation 1: Scene Interaction

Scene interaction vs. Obstacle avoidance/navigation

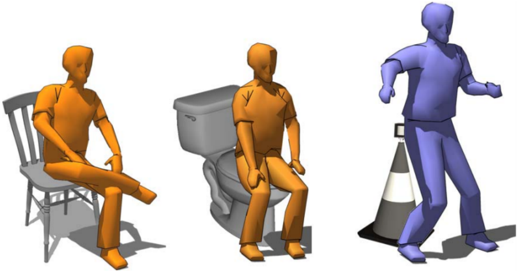
Building and keeping up to a date a model of the world enables robot interaction. A similar goal is enabling Human-Computer interaction.





Motivation 2: Physically constrained vision

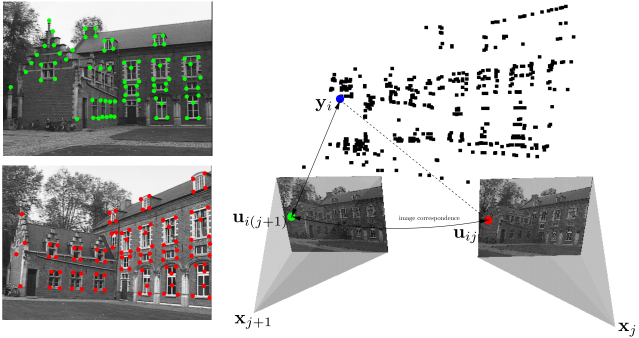
We can usefully recognize an object by utilising physical model properties – for example when we ask:
"Where is (the) chair?" (Visual recognition/search problem),
Do we really mean
"Where can I sit?" (Physically constrained embodied problem).



[Grabner, Ga, Van Gool"What makes a chair a chair?", CVPR 2011]

Can we just Scale up SfM to Real-time Dense SLAM?

-SFM (Structure from Motion)
-Obtain image correspondences across N views

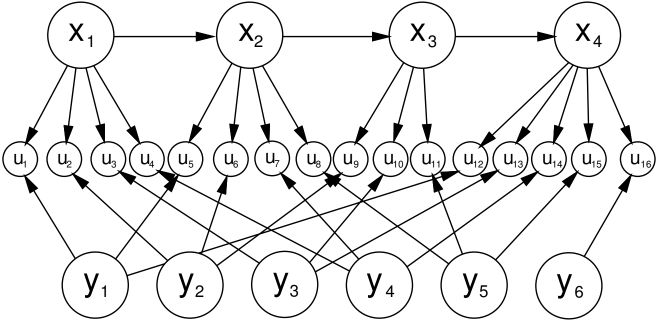


[Adapted from Pollefeys et al. 1999]

-Estimate both 3D points **y** and camera poses **x**
-Solve by minimising non-linear **2D Point Reprojection Error**

Can we just Scale up SfM to Real-time Dense SLAM?

-**Full SLAM** as Bayesian network: graphical representation shows structure

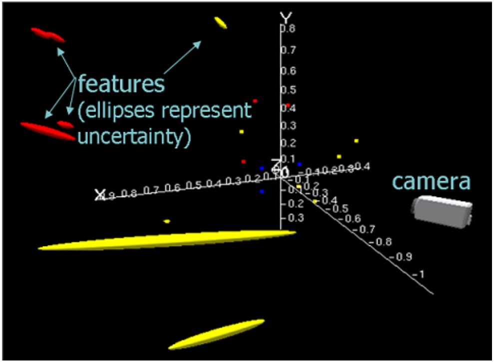


[Adapted from Dellaert and Kaess (2006)]

-**Can we** trivially scale to dense correspondences through video data?
-**Explosion of constraints** can the full optimisation problem be solved incrementally?

Example: first real-time SfM Kalman Filter

2003 Davison's Monoslam: importance of a cheap comodity sensor. Modelled and propagated joint uncertainty in real-time.



[Real-Time Simultaneous Localisation and Mapping with a Single Camera, Davison, ICCV 2003]

- Joint Gaussian distribution
- Covariance matrix is $(n+6) \times (n+6)$ growing with n structure.
- If we attempt to increase the density of the point cloud, it quickly becomes infeasible to solve in real-time due to fill in of the covariance matrix.

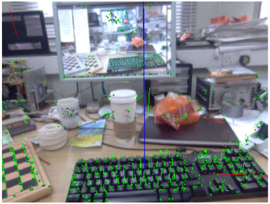
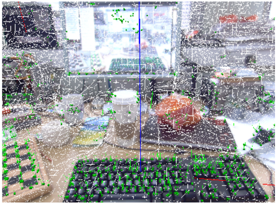
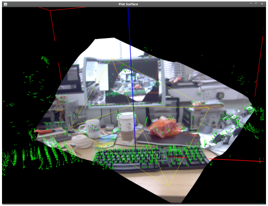
Example: Parallel Tracking and Mapping

-**2007,2008** Klein and Murray's PTAM
-Can handle **denser structure** estimation

[Parallel Tracking and Mapping, Klein and Murray, ISMAR 2007]

Building a dense model with the PTAM point cloud

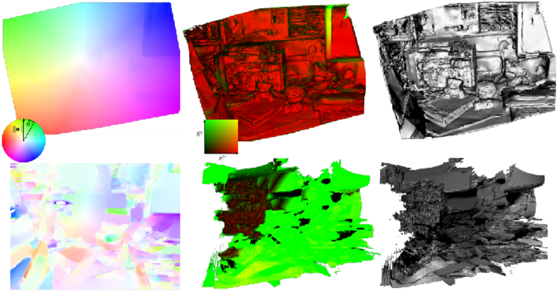
- Point cloud surface fitting techniques, e.g. implicit surface defined as a hierachical sum of Compactly Supported Basis Function weighted quadrics etc.
- Alternative method is to tetrahedralise the point cloud (proForma: Qi Pan et al 2009)., utilise the space carving property possible with point and line observations.

Hybrid Approach PTAM + Dense Optic Flow (Newcombe & Davison, CVPR 2010)

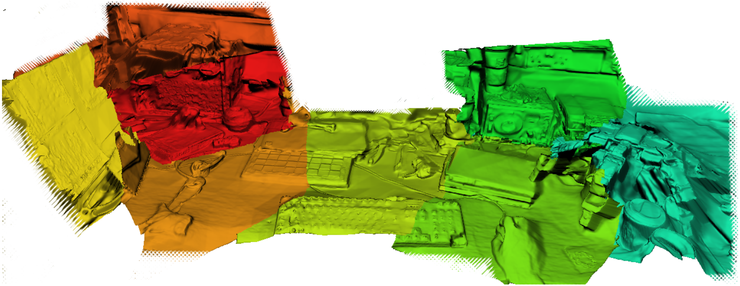
Optical flow initialised with surface prediction

We found that the coarse surface prediction (from a PTAM point cloud) greatly improves optic flow quality.



Real time, commodity SLAM system evolution

The depth maps are stiched sequentially into a global frame:

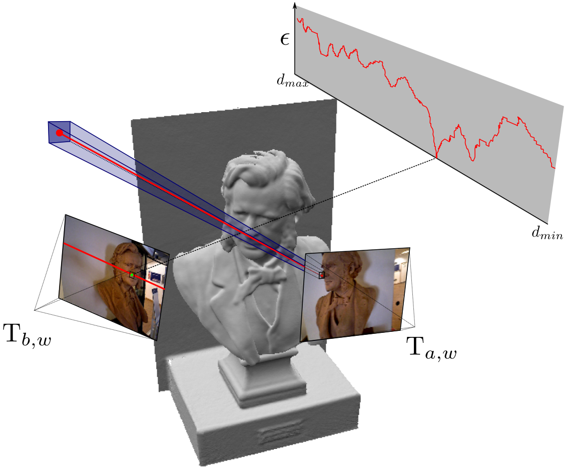


J. Stuehmer et al 2010, also augment the real-time SFM system but obtain real-time depth maps (without stiching/fusion). Also early work by Pollefeys et al 2007, on real-time reconstruction of Urban scenes.

Replace Sparse Mapping with Dense MVS

Multiple View Stereo

-A Reference pixel induces a photo-consistency error function



-Correspondence exists along epipolar line (if not occluded).

Cost volume data term

Build a cost volume from lots of weak data terms, and then using a simple discontinuity preserving smoothness prior, optimise global energy.

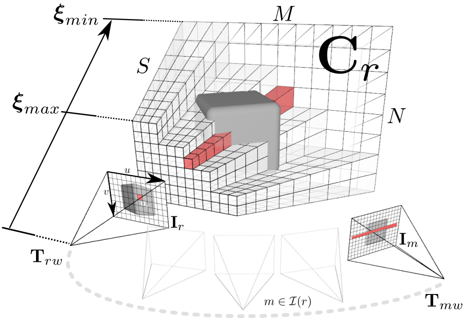


Figure : The cost volume for a given Depth map.

The sum over photometric errors is

$$\mathbf{C}_r(\mathbf{u}, d) = \frac{1}{|\mathcal{I}(r)|} \sum_{m \in \mathcal{I}(r)} \|\rho_r(\mathbf{I}_m, \mathbf{u}, d)\|_1 ,$$

$$\rho_r(\mathbf{I}_m, \mathbf{u}, d) = \mathbf{I}_r(\mathbf{u}) - \mathbf{I}_m\left(\pi\left(\mathrm{K}\mathrm{T}_{mr}\pi^{-1}(\mathbf{u}, d)\right)\right) ,$$

Using all possible frames from the live camera

-Combine lots of weak data-terms

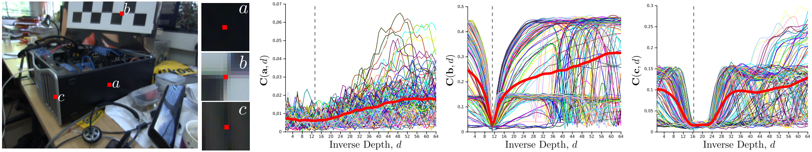


Figure : MVS Errors for single pixel photometric functions

Regularisation of the MVS cost

Energy = (**Data Term** Error) + (Spatial **Regularisation Term** Cost)

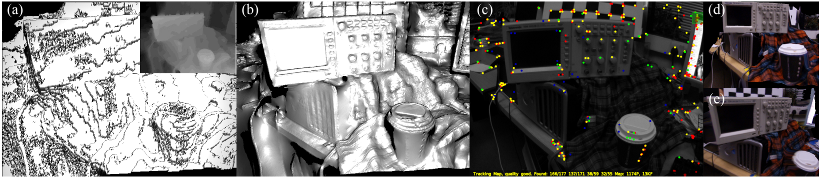
$$E_{\xi} = \int_{\Omega} \Big\{ \lambda \mathbf{C}\left(\mathbf{u}, \xi(\mathbf{u})\right) + g(\mathbf{u}) \|\nabla \xi(\mathbf{u})\|_{\epsilon} \Big\} \mathrm{d}\mathbf{u} \ .$$

- **non-convex** energy function
- Can iteratively linearise the data-term
- OR solve through splitting variable and exploit point-wise data-term.
- **Trivially paralellizable solution:** use GPGPU

Figure : Per pixel inverse depth minimum and lastly Regularisation

Dense Visual SLAM: Dense Tracking and Mapping

-**Single Passive Camera** system



[DTAM: Dense Tracking and Mapping in Real-time, Newcombe, Lovegrove, Davison, ICCV 2011]

Dense Mapping

-Create dense model using **multiple-view stereo** using estimated camera poses.

Dense Tracking

-Dense 6DoF tracking against current textured Model
-Enables **elegant occlusion handling**

Given current dense **textured** model :

-**Predict** View depth $\xi_v(\mathbf{u})$

-**Predict** View appearance $\mathbf{I}_v(\mathbf{u})$

To Estimate current view pose ψ (6DoF)

-**Minimise** cost over per pixel data error in live image $\mathbf{I}_l(\mathbf{u})$

$$f_{\mathbf{u}}(\psi) = \mathbf{I}_l\left(\pi\left(\mathrm{KT}_{lv}(\psi)\pi^{-1}\left(\mathbf{u}, \xi_v\left(\mathbf{u}\right)\right)\right)\right) - \mathbf{I}_v\left(\mathbf{u}\right).$$

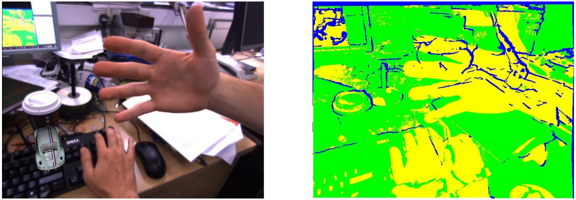


Figure : Gating given the predicted and live image (shown left).

Then along came commodity Depth Cameras: What's changed?

-Depth cameras provide real-time dense depth estimation
-Have become commodity devices!
-Two important technological changes in real-time vision:

Amazing commodity hardware capabilities

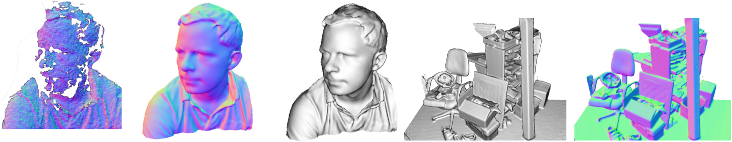
Kinect camera:
Real-time depth measurement

GPGPU:
Massive processing capabilities

This pairing of New technology changes what makes a solution scalable or elegant for SLAM.

KinectFusion: Real-Time Dense Surface Mapping and Tracking

ISMAR,UIST 2011 work while at MSRC.
-Use **structured light** based kinect device
-Exploit **real-time depth estimation** by fusing the data into a global implicit surface



KinectFusion Idea

-Use all depth frames and **build volumetric surface model**
-Perform full depth **frame to model** alignment as pose estimation
-Choice of representation to map efficiently to GPGPU computation.

Real time, commodity SLAM system evolution

KinectFusion uses only depth data, enabling operation of SLAM in complete darkness.

Dense Mapping as Surface Reconstruction

- Many techniques available for estimating a complete surface from a noisy point cloud.
- **Representation is important**: we don't want to be restricted in surface **topology** or precision.

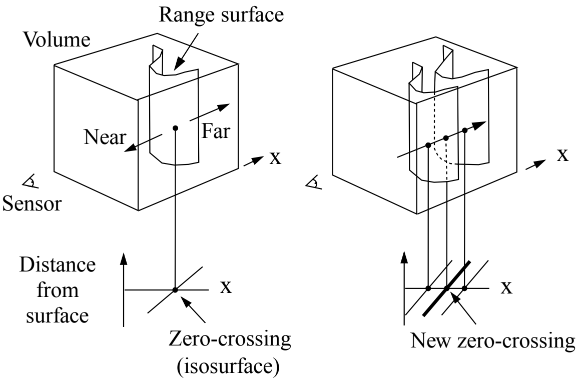
Use all data

We want to integrate over $640 \times 480 \times 30 \approx 9.2$ Million depth measurements per second on commodity hardware.

- Point clouds are *not* surfaces and meshes
- Updating surface topology is not trivial with explicit triangle meshes.

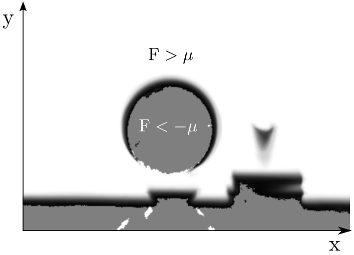
Surface reconstruction via depth map fusion

-Curless and Levoy (1996) introduced elegant method for fusing depth maps into a global surface.
-Use the **signed distance function (SDF)** representation of the depth measurement
-Robustly average the measurements together into a single SDF



Signed Distance Function surface representations

We use a *truncated signed distance* function representation, $\mathrm{F}(\vec{x}) : \mathbb{R}^3 \mapsto \mathbb{R}$ for the estimated surface where $\mathrm{F}(\vec{x}) = 0$.



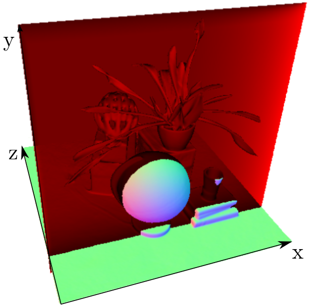
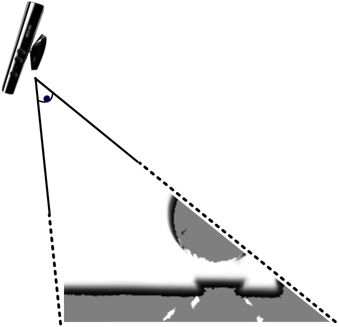
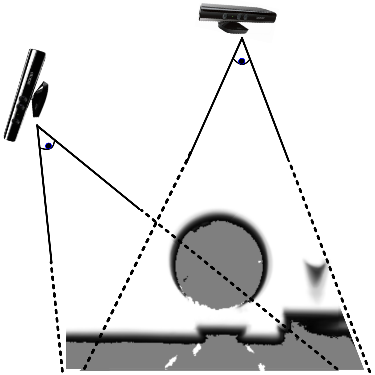


Figure : A cross section through a 3*D* Signed Distance Function of the surface shown.

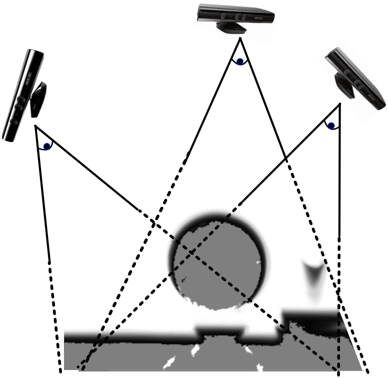
SDF Fusion



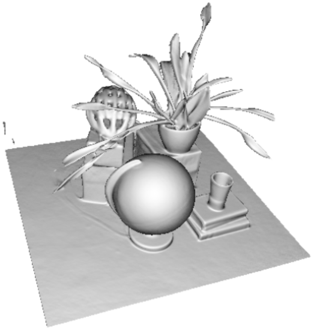
SDF Fusion



SDF Fusion

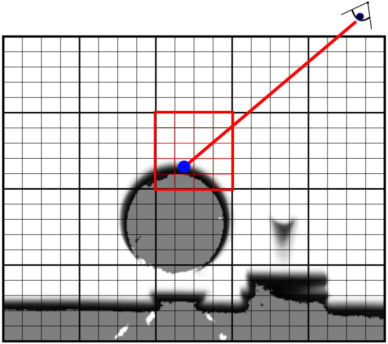


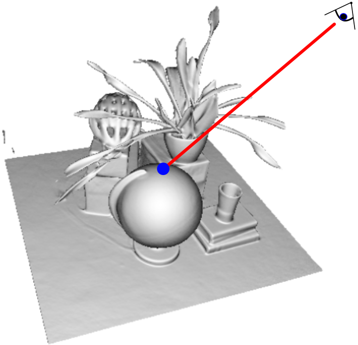
SDF Fusion



-Similar to volumetric denoising of the SDF under an ℓ_2 norm data-cost with no regularisation:
-Trivial to compute in an online manner as data comes in using weighted average.

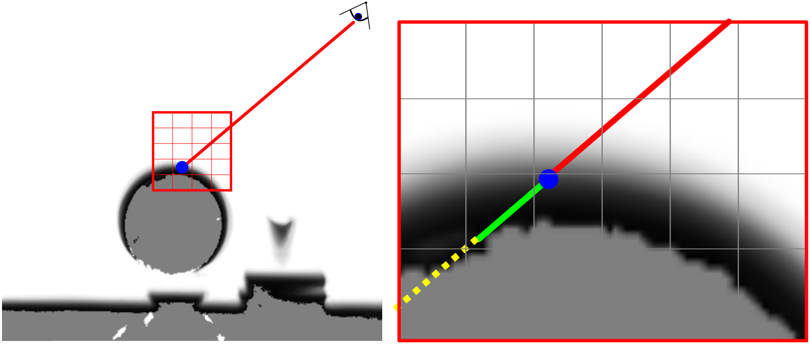
Rendering a surface represented in SDF





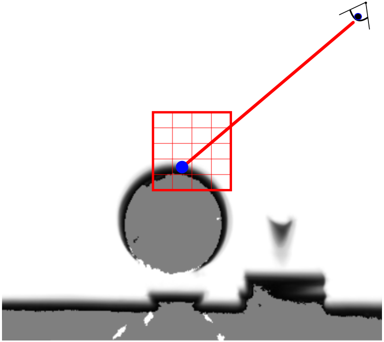
A regular grid holds a discretistion of the SDF. Ray-casting of iso-surfaces (S. Parker et al. 1998) is an established technique in graphics.

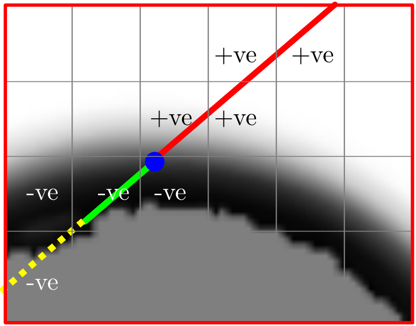
Rendering a surface represented in SDF



A regular grid holds a discretistion of the SDF. Ray-casting of iso-surfaces S. (Parker et al. 1998) is an established technique in graphics.

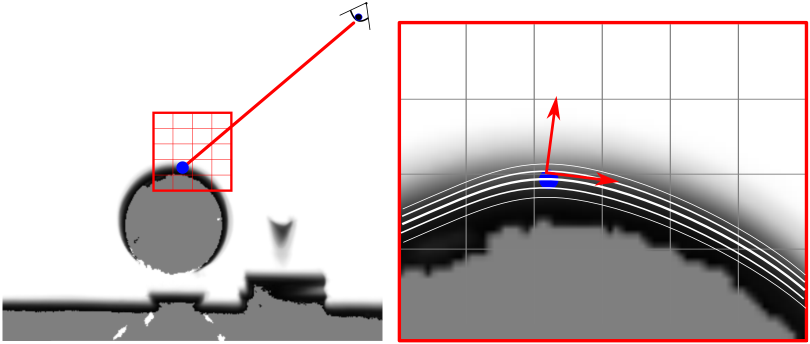
Rendering a surface represented in SDF





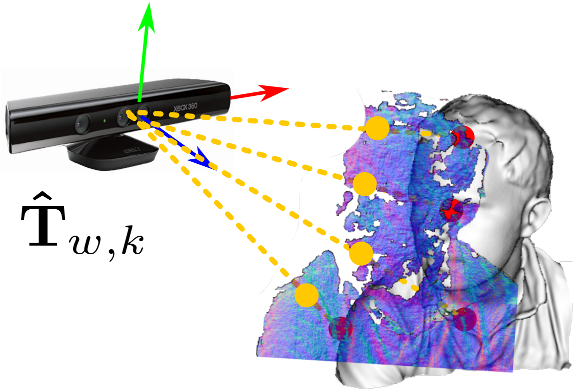
Interpolation reduces quantisation artefacts, and we can use the SDF value in a given voxel to skip along the ray if we are far from a surface.

Rendering a surface represented in SDF

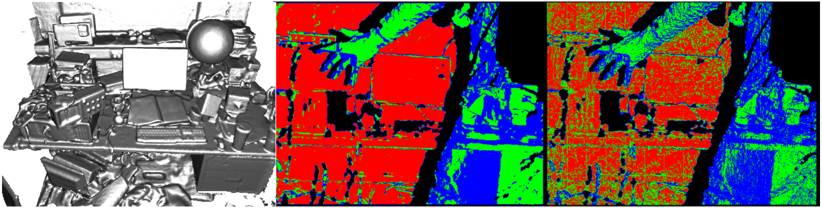


Near the level sets near the zero crossing are parallel. The SDF field implicitly represents the surface normal.

Predict current depth map and use dense ICP *(see Dense Tracking Talk)*



Using the Dense Prediction



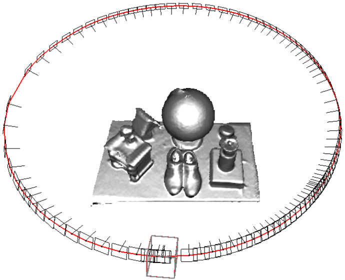
Dense inliers/outerliers

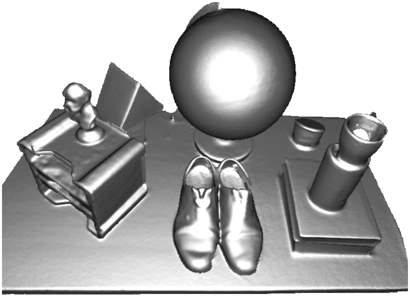
- ICP compatibility testing on the current surface model for **tracking** robustness
- Can use SDF distance check for **interaction** between moving unmapping objects in the scene.

Frame-Frame vs. *Frame-Model* Tracking

Low Drift Tracking with KinectFusion

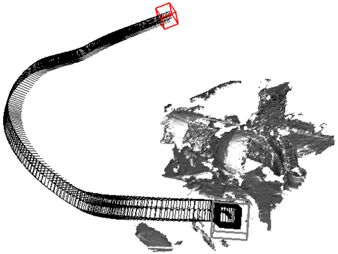
Frame-Model tracking provides drift free, higher accuracy tracking than Frame-Frame (Scan matching).

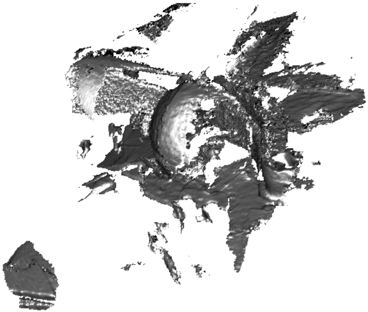




Frame-Frame vs. Frame-Model Tracking

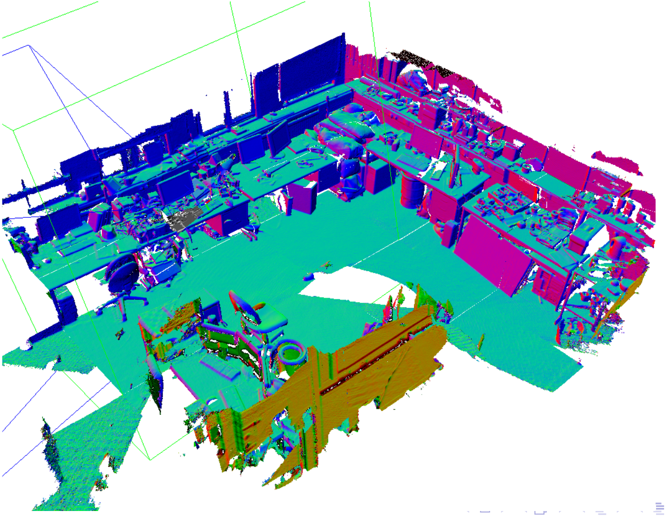
Frame-Frame tracking results in drift as pose errors are continuous integrated into the next frame.





Scalability

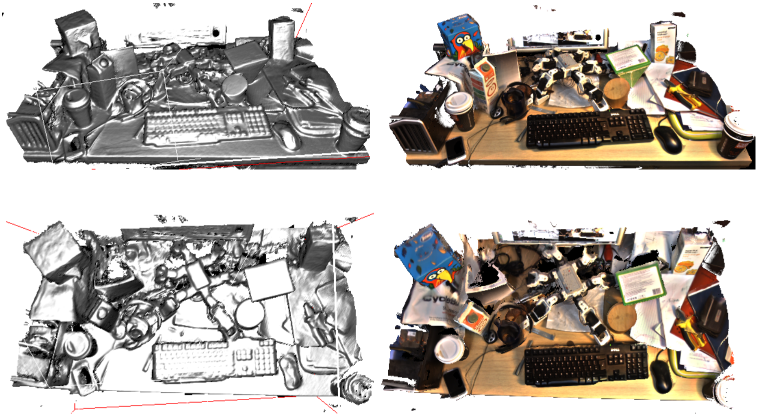
Sub-mapping techniques and **multi-scale** SDF representations to allow models to scale up for larger scenes (but note the system is still only greedily optimitising, hence drift can build up):



Can we do surface fusion with a single passive camera?

Yes

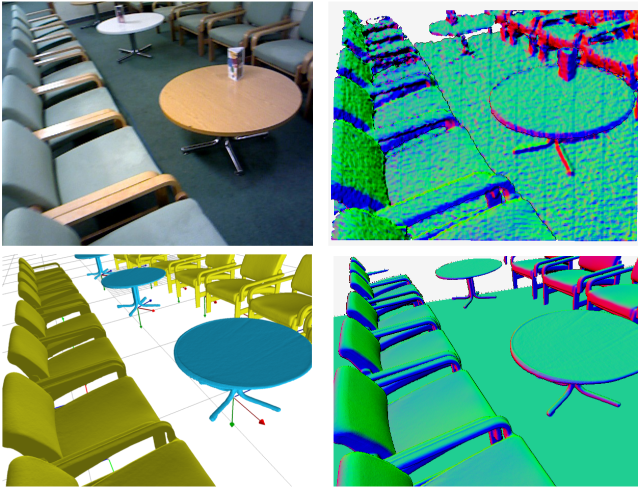
-Speed up single camera depth estimation to real-time
-Use the appearance based whole-image alignment tracking



Next Session: Beyond Surfaces to Object SLAM

-Can we bring object recognition into real-time Dense SLAM?

-No need reconstruction from scratch previously seen objects:



[CVPR 2013 : Salas-Moreno, Newcombe, Strasdata, Davison]

Dense Greedy SLAM Conclusions

Dense SLAM Key

Using denser surface model representation leads to trivially enabling all of the measurement data to be used.

- Using dense surface measurements leads to more robust tracking.
- Tracking from the current model can pose estimates good enough for dense MVS.
- Dense Models are more useful for robotics and augmented reality applications.
- But, we should start to incorporate more prior knowledge about the environment geometry: scene and object modelling.

Thankyou, Questions?