KinectFusion: Real-Time Dense Surface Mapping and Tracking

Ivy, Tian Jin

Outline

- Introduction
- Background
- Method
- Experiments
- Summary

Introduction

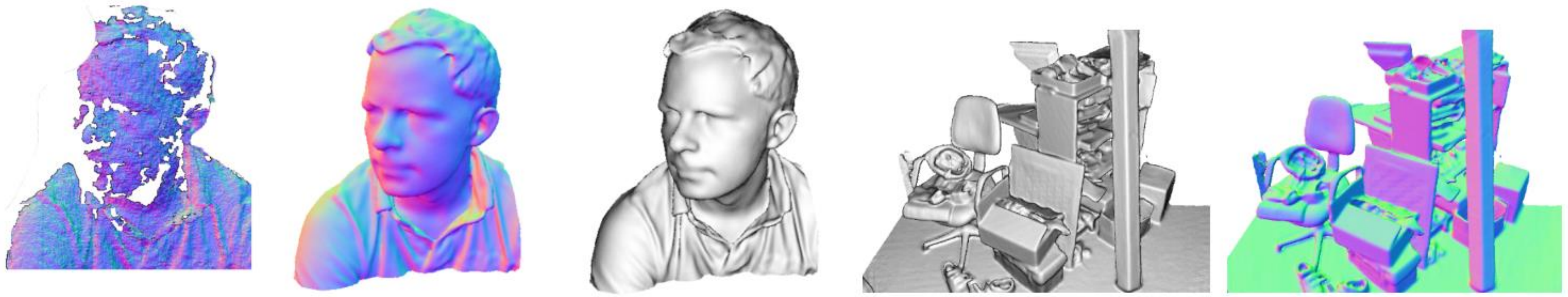


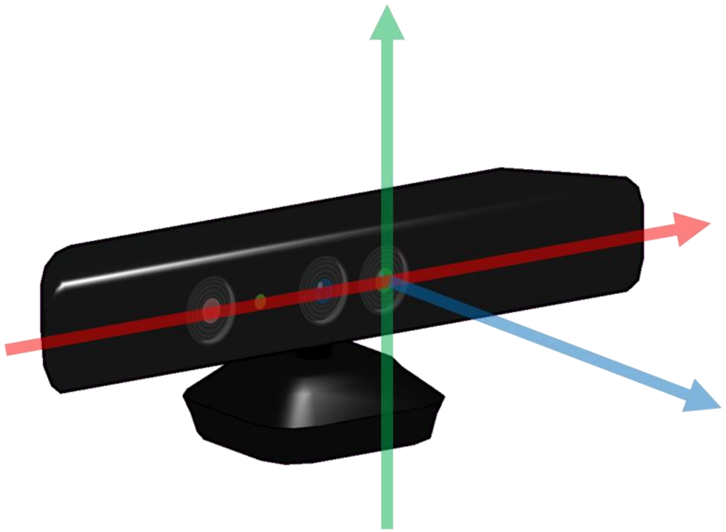
Figure 1: Example output from our system, generated in real-time with a handheld Kinect depth camera and no other sensing infrastructure. Normal maps (colour) and Phong-shaded renderings (greyscale) from our dense reconstruction system are shown. On the left for comparison is an example of the live, incomplete, and noisy data from the Kinect sensor (used as input to our system).

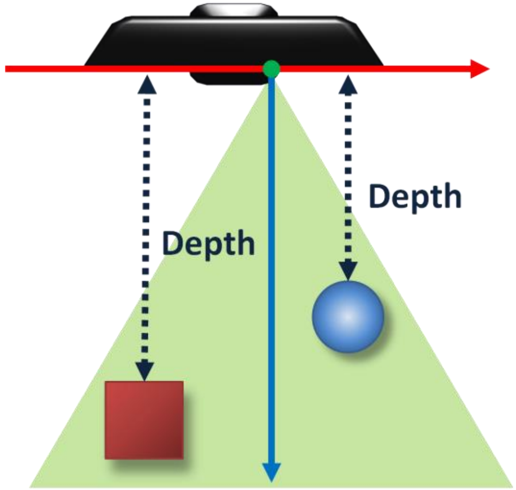
- Real-time mapping
- Regardless of lighting conditions
- Low-cost commodity camera and graphics hardware

→Kinect sensor

Background

- Kinect sensor





Method

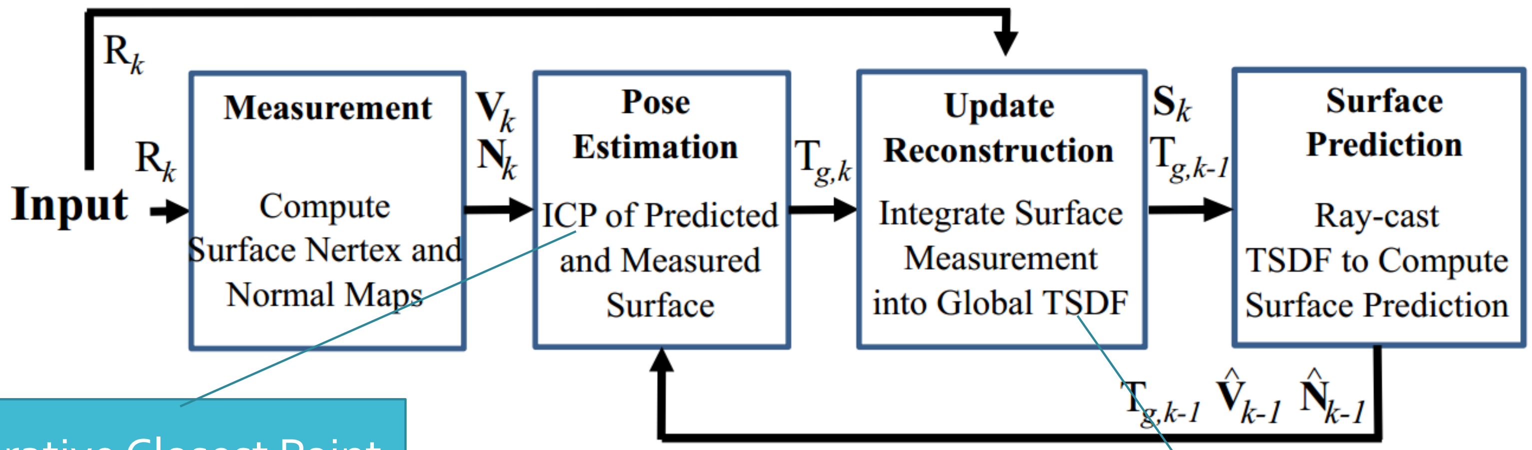


Figure 3: Overall system workflow.

Iterative Closest Point

Truncated Signed Distance Function

- Surface measurement
- Sensor pose estimation
- Surface reconstruction update
- Surface prediction

Method

Preliminaries

- Time: k
- Camera pose (a rigid body transformation matrix):

$$\mathrm{T}_{g,k} = \begin{bmatrix} \mathrm{R}_{g,k} & \mathbf{t}_{g,k} \\ \mathbf{0}^\top & 1 \end{bmatrix} \in \mathbb{SE}_3 , \quad (1)$$

where the Euclidean group $\mathbb{SE}_3 := \{\mathrm{R}, \mathbf{t} \mid \mathrm{R} \in \mathbb{SO}_3, \mathbf{t} \in \mathbb{R}^3\}$. This

- Transfer a point in the camera frame into the global co-ordinate frame

$$\mathbf{p}_g = \mathrm{T}_{g,k}\mathbf{p}_k .$$

- Using **K** to denote the camera calibration matrix

  which transforms points on the sensors plane into image pixels.

- homogeneous vectors $\dot{\mathbf{u}} := (\mathbf{u}^\top | 1)^\top$

Method

Surface measurement

- Pre-processing stage
- Generate a dense vertex map and normal map pyramid

Method

Surface measurement

Raw depth map R_k

Bilateral filter

Weighted value

$$D_k(\mathbf{u}) = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in \mathscr{U}} \mathcal{N}_{\sigma_s}(\|\mathbf{u}-\mathbf{q}\|_2)\,\mathcal{N}_{\sigma_r}(\|R_k(\mathbf{u}) - R_k(\mathbf{q})\|_2)\,R_k(\mathbf{q}) \,, \quad (2)$$

where $\mathcal{N}_\sigma(t) = \exp(-t^2\sigma^{-2})$ and $W_{\mathbf{p}}$ is a normalizing constant.

Depth map with reduced noise D_k

Back-project the filtered depth values into the sensors frame of reference k

$$V_k(\mathbf{u}) = D_k(\mathbf{u})\mathbf{K}^{-1}\dot{\mathbf{u}} \,. \quad (3)$$

Vertex map V_k

Cross product the neighboring vertices to compute normal vectors

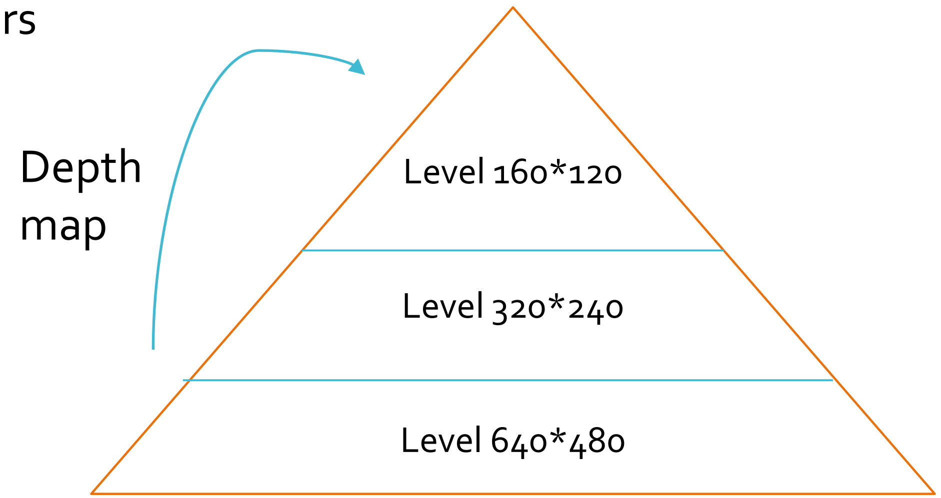
$$\mathbf{N}_k(\mathbf{u}) = \nu\big[(\mathbf{V}_k(u+1,v) - \mathbf{V}_k(u,v)) \times (\mathbf{V}_k(u,v+1) - \mathbf{V}_k(u,v))\big] \,, \quad (4)$$

where $\nu[\mathbf{x}] = \mathbf{x}/\|\mathbf{x}\|_2$.

Normal map N_k

Vertex and normal map pyramid Level = 3

To compute camera pose from coarse to fine, which can speed up the computation



Method

Mapping as Surface Reconstruction

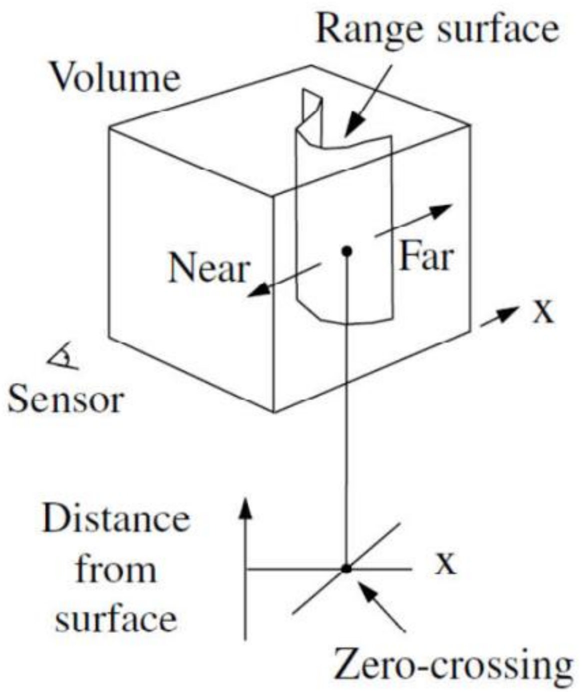
- The global scene fusion process
- Truncated Signed Distance Function (TSDF) representation

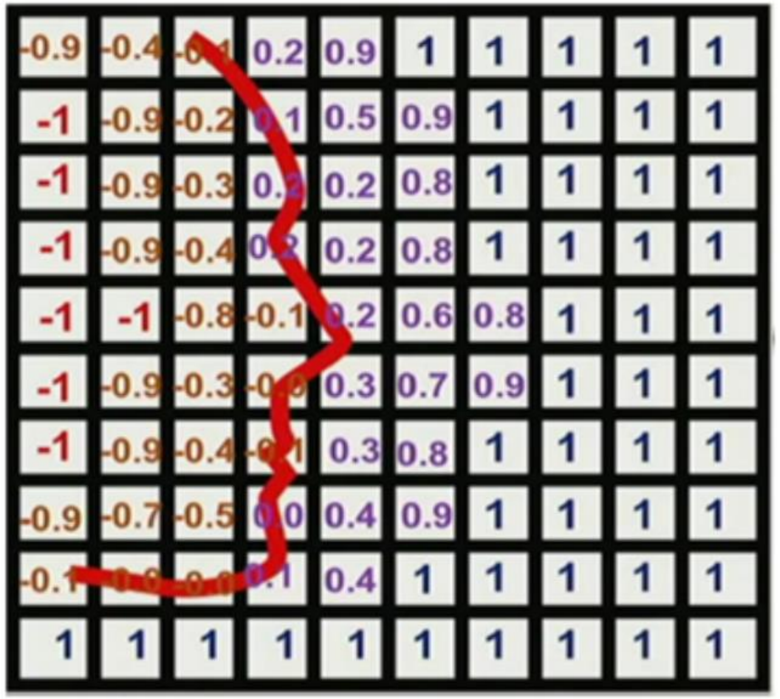
- “Each consecutive depth frame, with an associated live camera pose estimate is fused into one single 3D reconstruction”

Method

Mapping as Surface Reconstruction

- TSDF representation





Two components are stored:

$$\mathbf{S}_k(\mathbf{p}) \mapsto [\mathrm{F}_k(\mathbf{p}), \mathrm{W}_k(\mathbf{p})] \,. \qquad (5)$$

Method

Mapping as Surface Reconstruction

- How to compute the distance value?

3D point (global coordinates)

The ray from the camera center to the 3D point p

$$F_{R_k}(\mathbf{p}) = \Psi\left(\lambda^{-1}\|(\mathbf{t}_{g,k} - \mathbf{p}\|_2 - R_k(\mathbf{x})\right), \quad (6)$$

Normalization factor

$$\lambda = \|K^{-1}\dot{\mathbf{x}}\|_2, \quad (7)$$

2D pixel point

$$\mathbf{x} = \left\lfloor \pi\left(KT_{g,k}^{-1}\mathbf{p}\right)\right\rfloor, \quad (8)$$

$$\Psi(\eta) = \begin{cases} \min\left(1, \frac{\eta}{\mu}\right)\mathrm{sgn}(\eta) & \text{iff } \eta \geq -\mu \\ null & otherwise \end{cases} \quad (9)$$

SDF truncation

- Weighted value?

Proportional to $cos(\theta)/R_k(\mathbf{x})$

Method

Mapping as Surface Reconstruction

- Global TSDF?
- ------weighted average

Global TSDF

TSDF of current frame

$$F_k(\mathbf{p}) = \frac{W_{k-1}(\mathbf{p})F_{k-1}(\mathbf{p}) + W_{R_k}(\mathbf{p})F_{R_k}(\mathbf{p})}{W_{k-1}(\mathbf{p}) + W_{R_k}(\mathbf{p})} \quad (11)$$

$$W_k(\mathbf{p}) = W_{k-1}(\mathbf{p}) + W_{R_k}(\mathbf{p}) \quad (12)$$

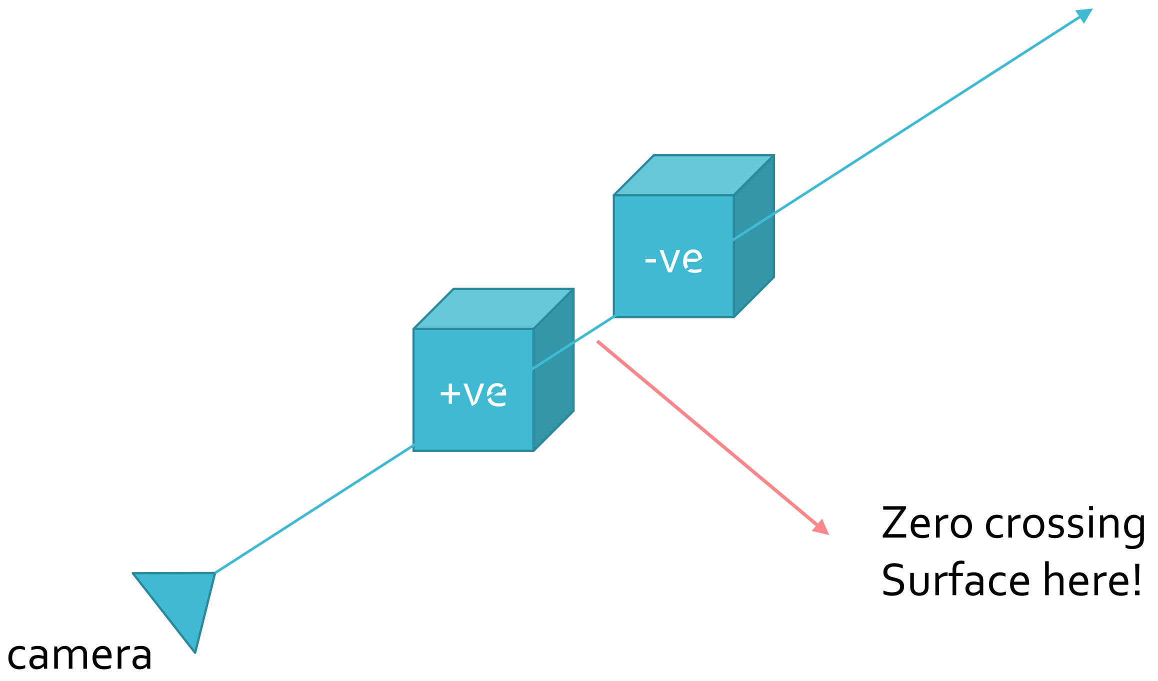
- Truncated weights

$$W_k(\mathbf{p}) \leftarrow \min(W_{k-1}(\mathbf{p}) + W_{R_k}(\mathbf{p}), W_\eta)\ , \quad (13)$$

Method

Surface prediction from ray casting the TSDF

- TSDF distance value = 0 → surface
- A per-pixel ray-casting is performed



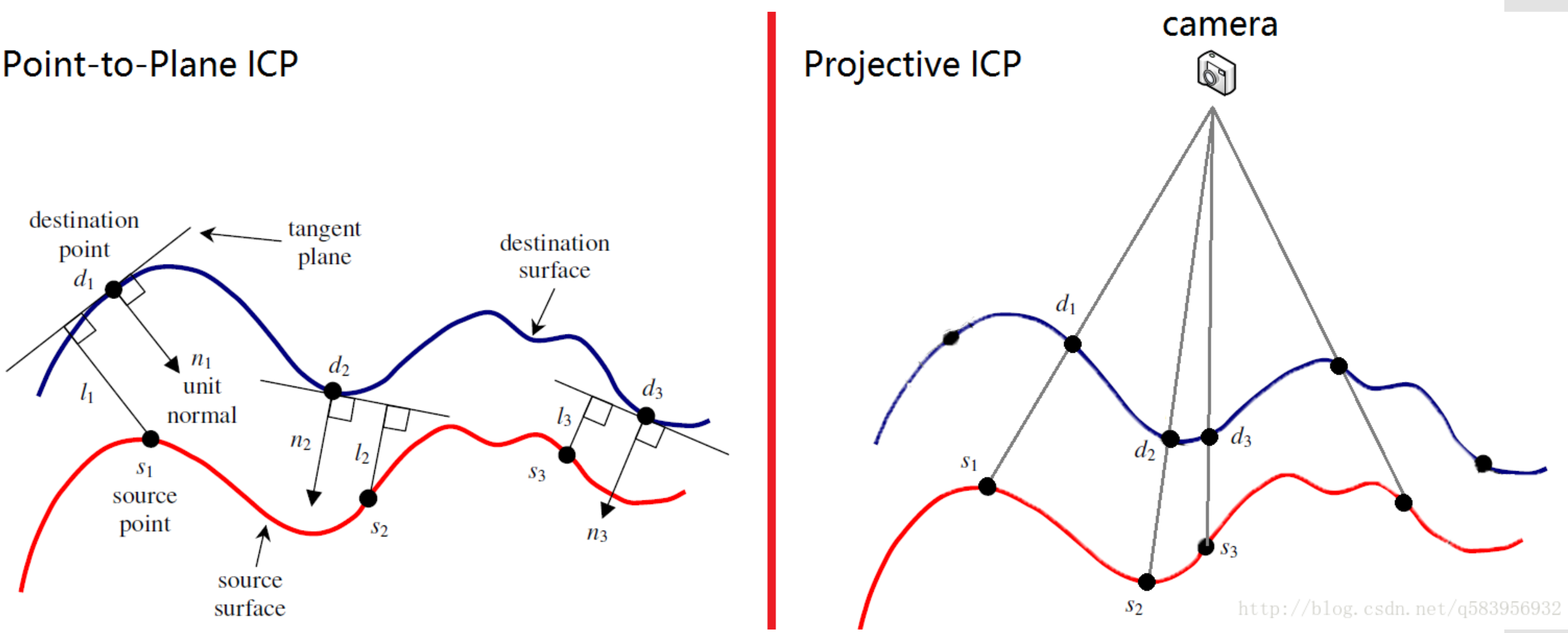
Method

Sensor pose estimation

- Iterative closest point (ICP) algorithm
- →can be used to compute the rigid body transformation

Point-to-Plane ICP

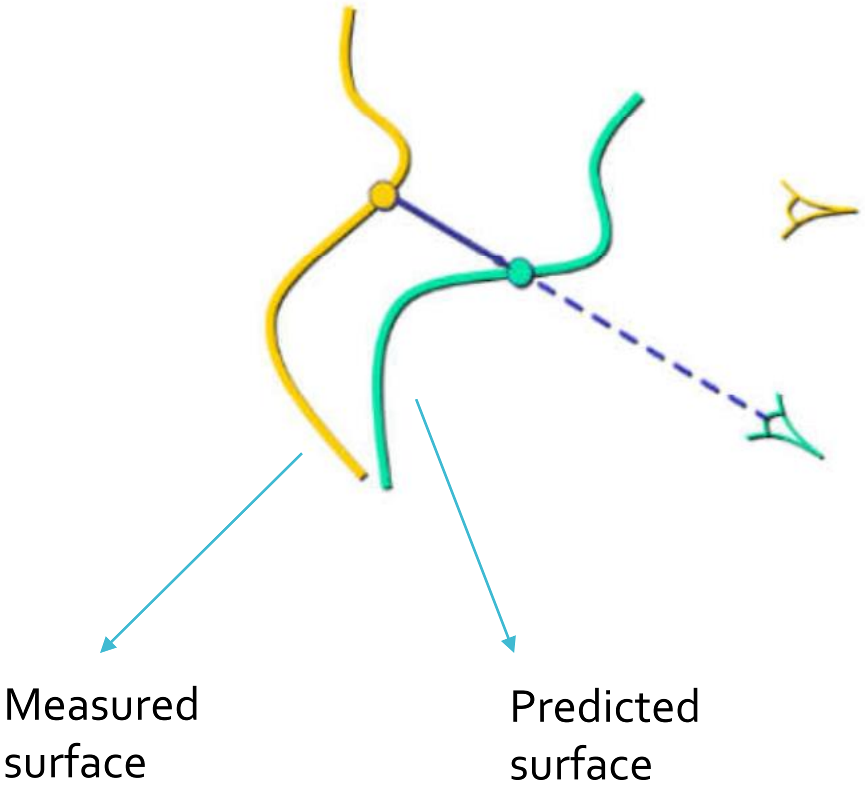
Projective ICP



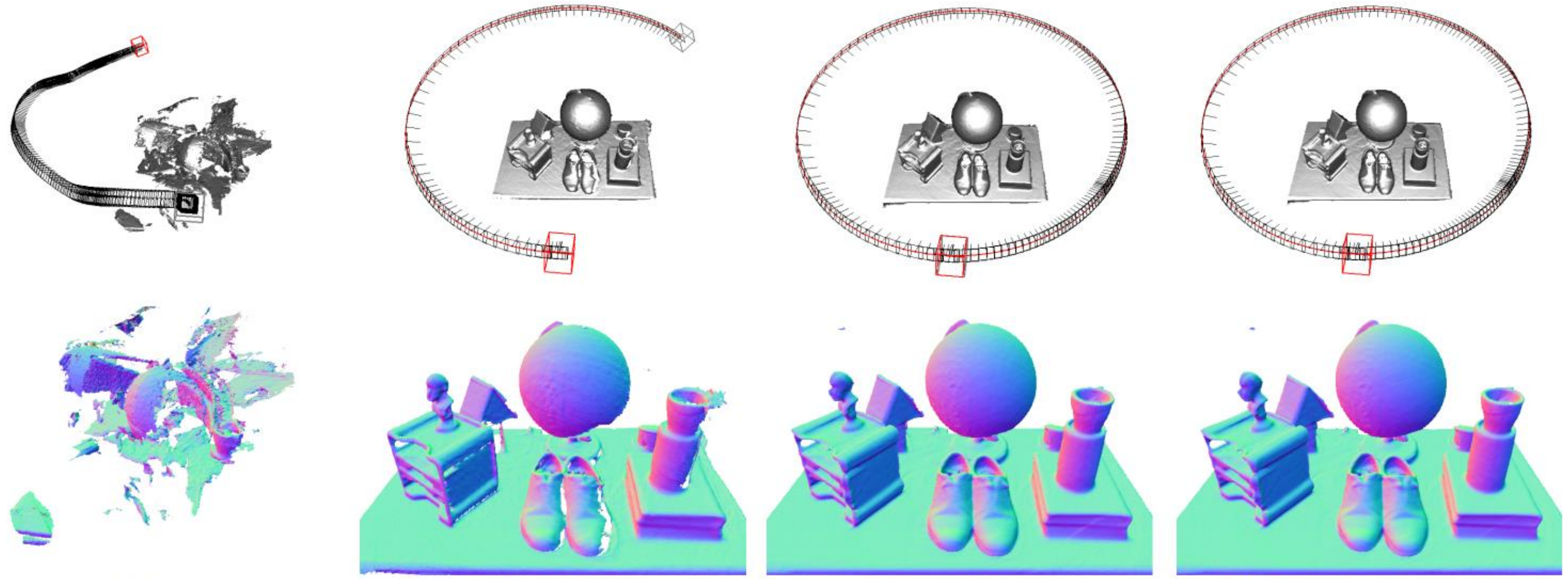
Method

Sensor pose estimation

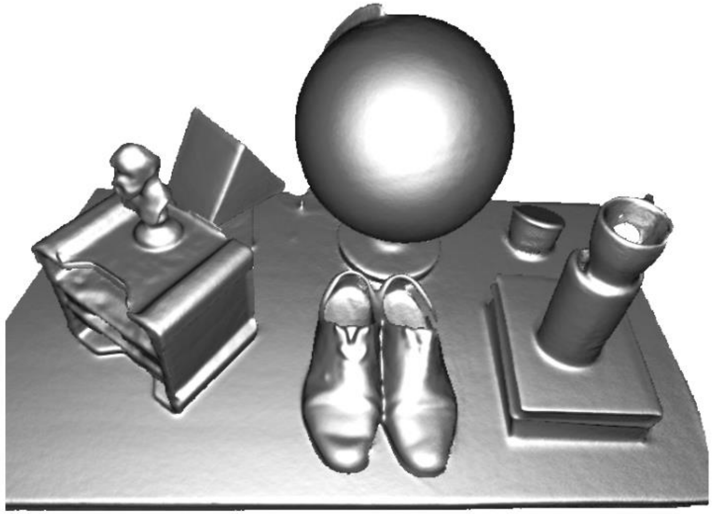
- Frame-to-model ICP?



Experiments



(a) Frame to frame tracking (b) Partial loop (c) Full loop (d) M times duplicated loop



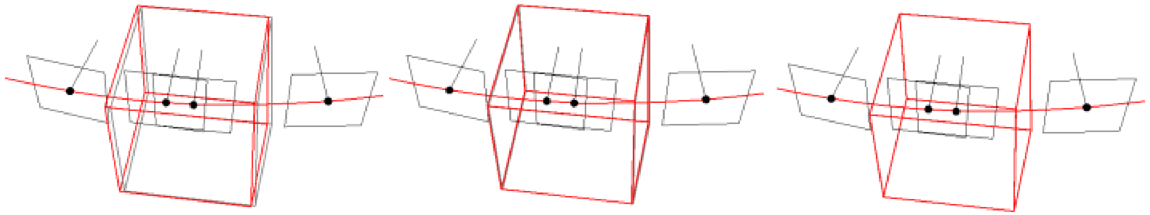
(e) MN frames without precise repetition

- The Kinect sensor is placed on a fixed turntable

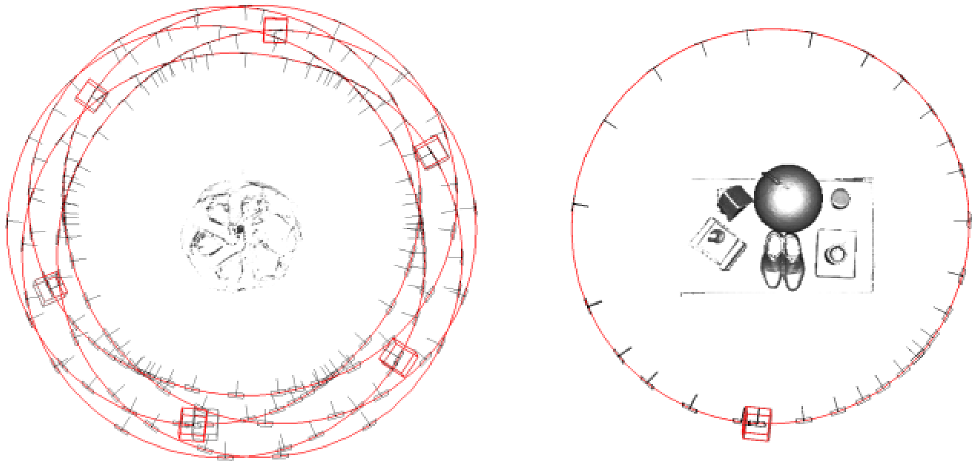
- Capture every 19s
- N=560 frames

Experiments

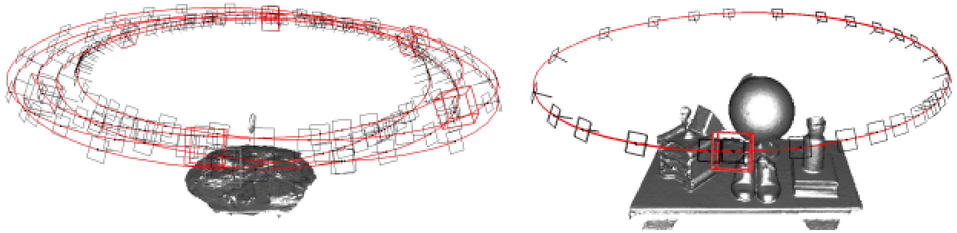
Alignment



(a) M=1 (b) M=2 (c) M=4



(a) Frame to frame tracking (b) Frame to model tracking



- Drift-free

Experiments ------ Processing time

-Constant

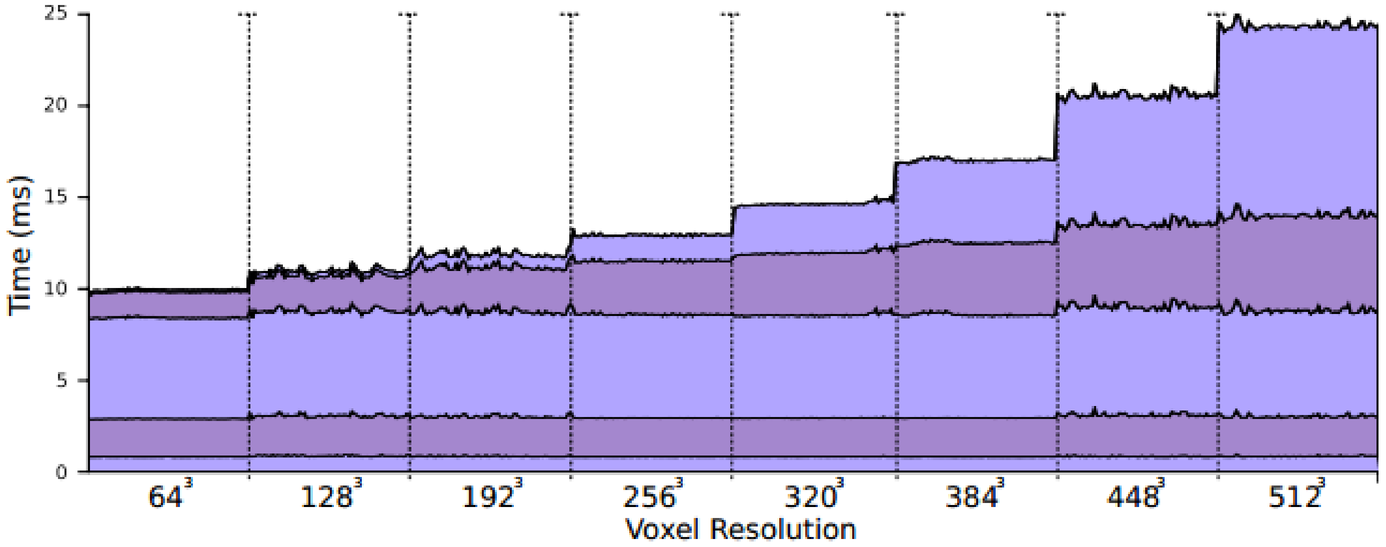


Figure 13: Real-time cumulative timing results of system components, evaluated over a range of resolutions (from 64^3 to 512^3 voxels) as the sensor reconstructs inside a volume of $3m^3$. Timings are shown (from bottom to top of the plot) for: pre-processing raw data, multi-scale data-associations; multi-scale pose optimisations; raycasting the surface prediction and finally surface measurement integration.

Experiments

Observations and Failure Modes

- Robustness to lighting indoor scene

- Main failure:
- Large planar scene

- No features
- Hard to align

Summary

- Key concept
  - Up-to-date surface representations fusing all data from previous scans with TSDF
  - Frame-to-model
  - Fully parallel algorithms

- Challenges
  - <= $7m^3$

Thanks!

Q&A?