Pagliari et al.
Kinect Fusion Improvement Using Depth Camera Calibration

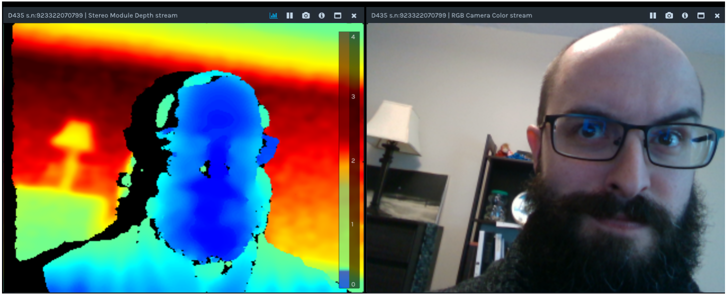
COMP5115 - Fall 2019

Outline

- Introduction and Motivation
- Related Works
- The KinectFusion Method
    - Measurement
    - Surface Reconstruction
    - Surface Prediction
    - Sensor Pose Estimation
- Results and Conclusion

Introduction and Motivation

- Purchased an Intel RealSense D435 Camera.
- Studied STAR paper by Zollhöfer to see how it could be used.
- Realized that article was too high-level (and advanced).
- KinectFusion seems to have established current paradigm. Explains math bits nicely, so a good starting paper.


D435 s.n:923322070799 | Stereo Module Depth stream
4
3
2
1
0
D435 s.n:923322070799 | RGB Camera Color stream

Problem Statement

Problem: process a stream of RGB-D frames for Simultaneous Localization and Mapping (and do it in real time!)

- Tracking: estimate the pose (position + orientation) of the camera. Camera presumed moving through space – need to keep track of position and which way it's pointing.
- Mapping: (incrementally) build a model of the scene captured by camera.

Challenges

- High volume of data (640x480 @ 30fps = 9 million points per sec)
- Occlusion (stuff in the way), holes
- Measurement errors: incident angles, shiny or transparent materials
- Potentially erratic camera movement: blurry measurements
- Dynamic scenes, moving objects
- Camera drift: accumulation of errors in pose estimation

Related Works I

Beardsley, et al. 1997
Sequential updating of projective and affine structure from motionof projective and affine structure from motion.

Fitzgibbon et al. 1998
Automatic camera recovery for closed or open image sequences.

A. J. Davison. 2003
Real-time simultaneous localisation and mapping with a single camera.

Related Works II

G. Klein and D. W. Murray. 2007
Parallel tracking and mapping for small AR workspaces.

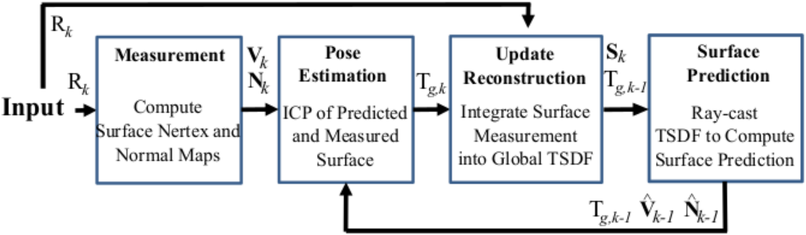
R. A. Newcombe and A. J. Davison. 2010
Live dense reconstruction with a single moving camera.

Method – Overview

Dense SLAM with Active Depth Sensing is an online scene reconstruction system composed of 4 steps:

1. Surface Measurement: pre-processing, generate vertex data & normals
2. Surface Reconstruction Update: use pose estimation to integrate new surface measurements into global scene model (TSDF).
3. Surface Prediction: generate dense surface prediction to align new depth maps.
4. Sensor Pose Estimation: multi-scale ICP align between predicted surface and current measurement.

Method – Overview 2



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
$T_{g,k-1}$ $\hat{V}_{k-1}$ $\hat{N}_{k-1}$

Figure 3: Overall system workflow.

Method – Math Preliminaries

6 degree of freedom pose estimation representated as matrix

$$T_{g,k} = \begin{bmatrix} R_{g,k} & t_{g,k} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

(an element Special Euclidean group – translations & rotations but not reflections)
It maps camera coordinate frame at time k into global frame g. Point $p_k \in \mathbb{R}^3$ in camera space is transferred to global coordinate space via

$$p_g = T_{g,k} p_k$$

Method – Math Preliminaries 2

Three different reference frames for points: camera frustum, projective space (camera pixels) and global model.
Camera matrix K transforms points on the depth surface into image pixels. and $\pi(p)$ performs perspective projection (dehomogenization) to obtain camera pixel $q \in \mathbb{R}^2 = (x/z, y/z)^T$

Method – Surface Measurement

Raw depth map R_k at time k gives calibrated depth $R_k(u) \in \mathbb{R}$ at each pixel $u = (u, v)^T$ for $u \in \mathcal{U} \subset \mathbb{R}^2$ (camera pixel space).

$$\mathbf{p_k} = R_k(\mathbf{u})K^{-1}\dot{\mathbf{u}}$$

p_k is a metric point measurement in sensor frame k.

Method – Surface Measurement 2

Apply *bilateral filter* to raw depth map to smooth noise.

$$D_k(\mathbf{u}) = \frac{1}{W_p} \sum_{\mathbf{q} \in \mathcal{U}} \mathcal{N}_{\sigma_s}(||\mathbf{u} - \mathbf{q}||_2) \mathcal{N}_{\sigma_r}(||R_k(\mathbf{u}) - R_k(\mathbf{q})||_2) R_k(\mathbf{q})$$

Where W_p is a normalizing constant (two Gaussians) and σ_r and σ_s are parameters.

Method – Surface Measurement 3

Vertex & Normal Maps

Create vertex map V_k by projecting filtered depth values back into sensor's frame of reference:

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

Depth sensor frames are measurements on a regular grid so can approximate normals using neighbours easily:

$$N_k(\mathbf{u}) = v\left[(V_k(u+1,v) - V_k(u,v)) \times V_k(u,v+1) - V_k(u,v)\right]$$

where $v[x] = \hat{x}$

Method – Surface Measurement 4

Validity Mask

Also need to keep track of sensor failures. Use *validity mask*

$$M_k(\mathbf{u}) = \begin{cases} 1 & \text{depth measure transforms to valid vertex?} \\ 0 & \text{otherwise} \end{cases}$$

Finally, create “multi-scale representation of surface measurement in form of a vertex and normal pyramid.”

Depth *pyramid* is a sequence $D^{l \in [1 \ldots L]}$ created by stacking depth map with sub-sample layers created by block-averaging (convolution?).

Method – Surface Measurement 5

Authors use $L = 3$ and are careful to “discard depth values more than $3\sigma_r$ of the central pixel to avoid smoothing over depth boundaries”. Vertex and normal pyramids are then $V^{l\in[1\dots L]}$ and $N^{l\in[1\dots L]}$ computed using corresponding depth pyramid layer.

$$V_g^k(\mathbf{u}) = T_{g,k} V_k \dot{(}\mathbf{u})$$

$$N_g^k(\mathbf{u}) = R_{g,k} N_k(\mathbf{u})$$

Method – Mapping as Surface Reconstruction

Global & Current TSDF

Function $S_k(\mathbf{p})$ is a fusion of TSDFs estimated from frames $1 \ldots k$ (where $p \in \mathbb{R}^3$ a global frame point in 3D volume).

$$S_k(\mathbf{p}) \mapsto [F_k(p), W_k(p)]$$

Assuming sensor error μ, dense surface measurement provides two constraints

$$r \overset{?}{<} (\lambda R_k(\mathbf{u}) - \mu)$$

where $\lambda = ||K^{-1}\dot{u}||$.
If less, detected free space. No surface information is obtained in reconstruction volume. Discard these values.

Method – Mapping as Surface Reconstruction 2

For raw map R_k with known pose $T_{g,k}$, its global frame projective TSDF is $[F_{R_k}, W_{R_k}]$ at a point $\mathbf{p}$ in the global frame is computed as

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

$$\lambda = ||K^{-1}\dot{x}||_2$$

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

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

Where $W_{R_k} \propto \cos(\theta)/R_k(x)$ and θ angle between associated pixel ray direction and surface normal in local frame. Represents weight for uncertainty of measurement.

Method – Mapping as Surface Reconstruction 3

TSDF are fused together by taking weighted sum of TSDF using raw depth data (current frame) and global surface model at previous step:

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

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

Method – Surface Prediction

Predictions stored as vertex and normal maps denoted $\hat{V}_k$ and $\hat{N}_k$.
Uses dense surface reconstruction (global TSDF) to do per-pixel raycasting.
March along each pixel ray until the SDF transition from positive to negative.
For points on surface $F_k(\mathbf{p}) = 0$, can assume gradient of TSDF at $\mathbf{p}$ is orthogonal to level set. So normals obtained using gradient:

$$\nabla F = \left[\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial F}{\partial z}\right]$$

Method – Sensor Pose Estimation

Estimating current camera pose $T_{w,k} \in \mathbb{SE}_3$. Track sensor by aligning live surface measurement (V_k, N_k) against model prediction from previous frame $(V\hat{}_{k-1}, N\hat{}_{k-1})$.

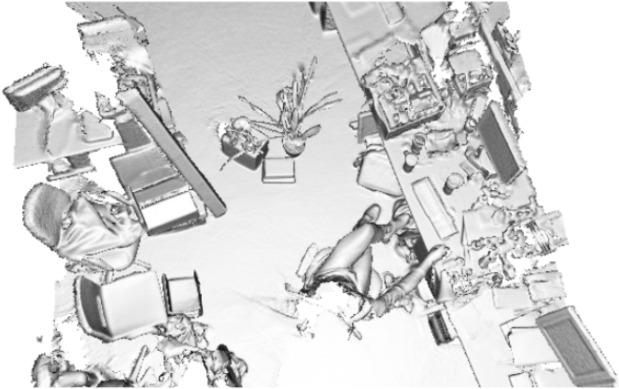
- No feature selection: use all depth data.
- Dense iterated closest point based technique.
- High tracking rate: can assume small motion between frames.
- Uses “fast projective data association”.
- Uses point-to-plane metric.

An iterative solution is obtained by minimizing a linearized energy function:

$$E(T_{g,k}) = \sum_{u \in \mathcal{U}} ||(T_{g,k})V_k(u) - V^{\hat{g}}_{k-1}(\hat{u})^T N^{\hat{g}}_{k-1}(\hat{u})||_2$$

Results

Impressive real-time reconstruction of scenes:



Results 2

Demos of KinectFusion on YouTube:

`https://www.youtube.com/watch?v=KOUSSlKUJ-A`

STAR: More difficult variation: dynamic scenes

`https://www.youtube.com/watch?v=N5bFhtlgRCc`

`https://www.youtube.com/watch?v=Zg0Zaiarlpk`

`https://www.youtube.com/watch?v=2dkcJ1YhYw4`

Also: simultaneous capture of lighting, albedo, material.

Conclusion

- KinectFusion defined pipeline for real-time geometry processing.
- Paves the way for cool applications in VR and AR.
- Cleanly written, well-chosen notation.

References I

Zollhöfer, Michael et al. (2018)
State of the Art on 3D Reconstruction with RGB-D Cameras
Computer Graphics Forum

Pagliari, Diana and Menna, Fabio and Roncella, R and Remondino, Fabio and Pinto, Livio (2011)
Kinect Fusion improvement using depth camera calibration
Photogrammetry, Remote Sensing and Spatial Information Sciences