University of Zurich UZH | ETH zürich

Institute of Informatics – Institute of Neuroinformatics

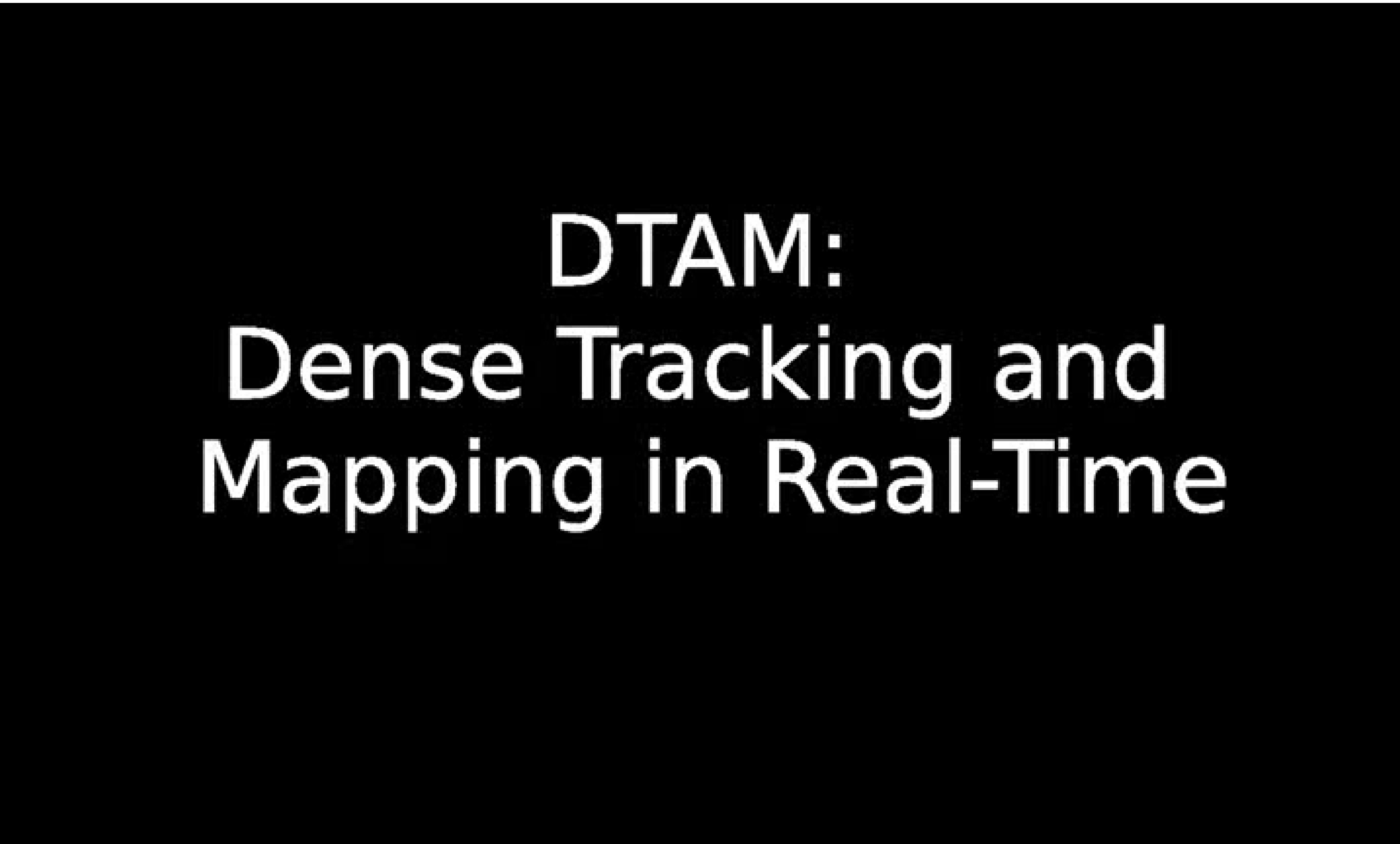
ROBOTICS & PERCEPTION GROUP

Lecture 10
Dense 3D Reconstruction

Davide Scaramuzza

http://rpg.ifi.uzh.ch/

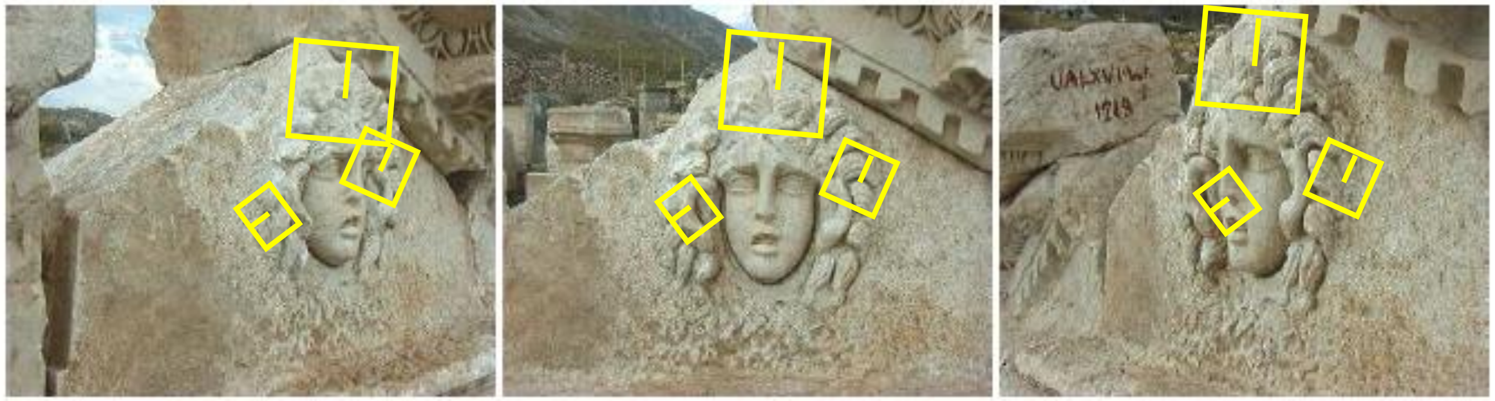
DTAM: Dense Tracking and Mapping in Real-Time, ICCV'11 by Newcombe, Lovegrove, Davison

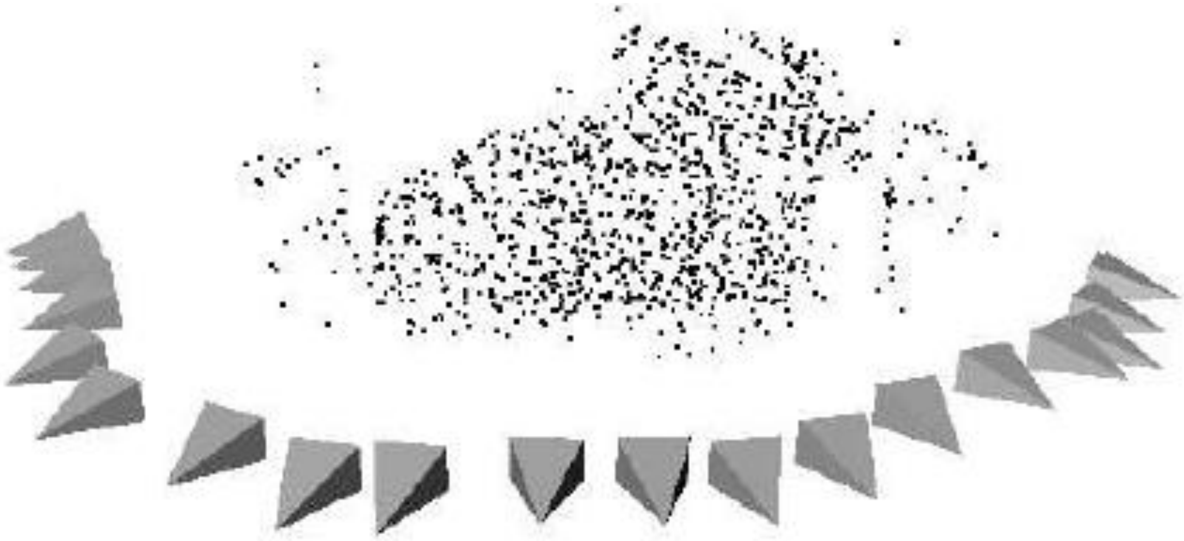


DTAM:
Dense Tracking and
Mapping in Real-Time

Sparse Reconstruction

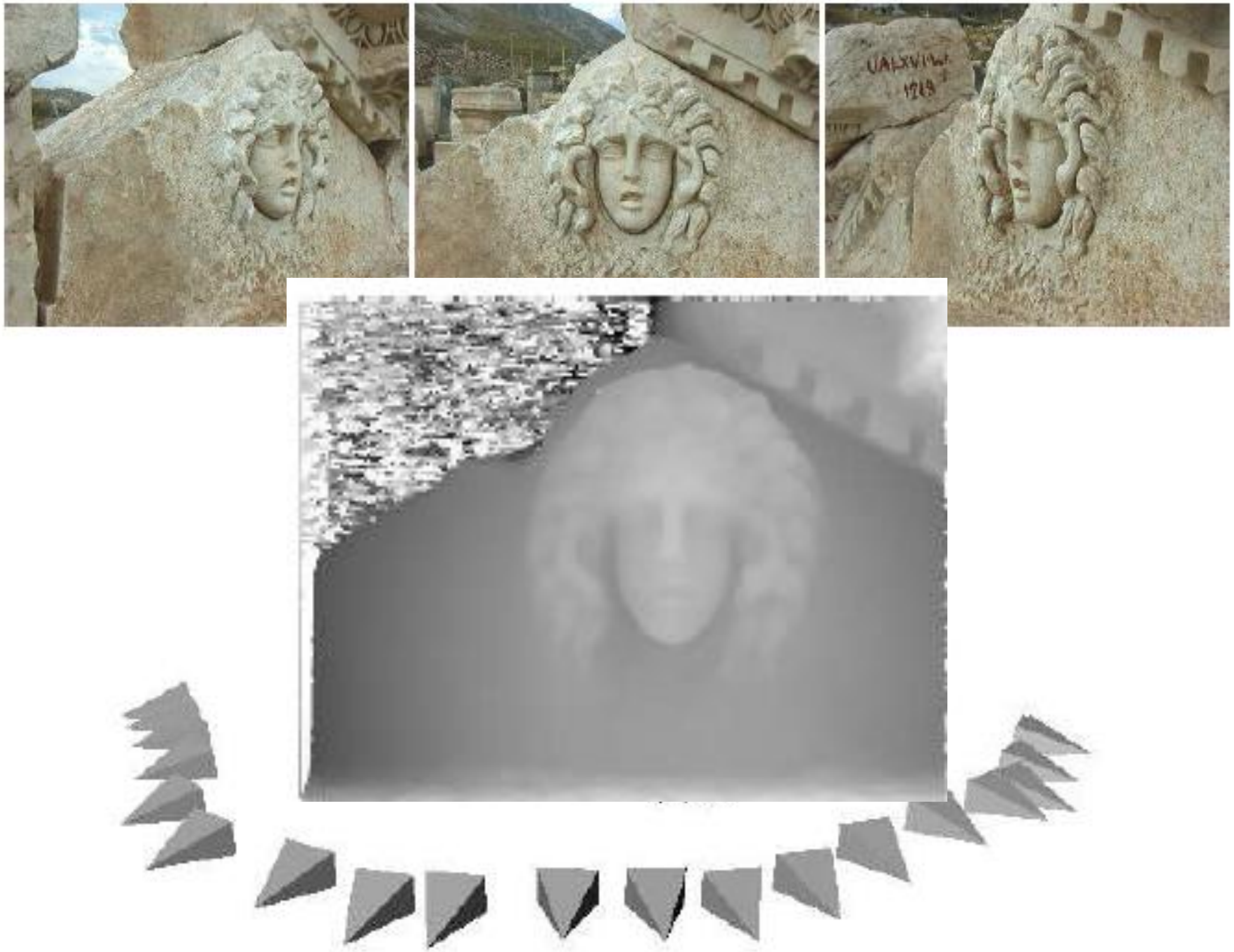
- Estimate the structure from a "sparse" set of features





Dense Reconstruction

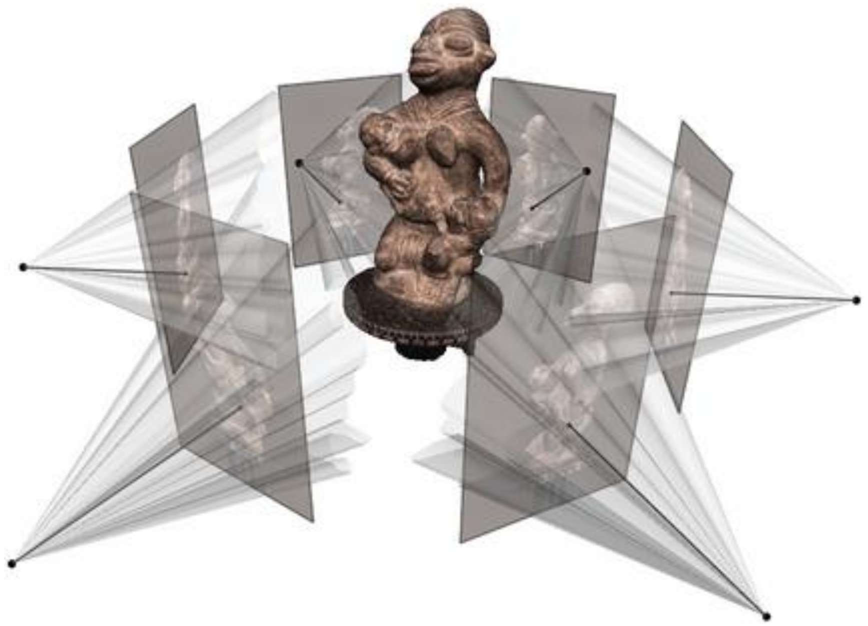
- Estimate the structure from a "dense" region of pixels



Dense Reconstruction (or Multi-view stereo)

Problem definition:

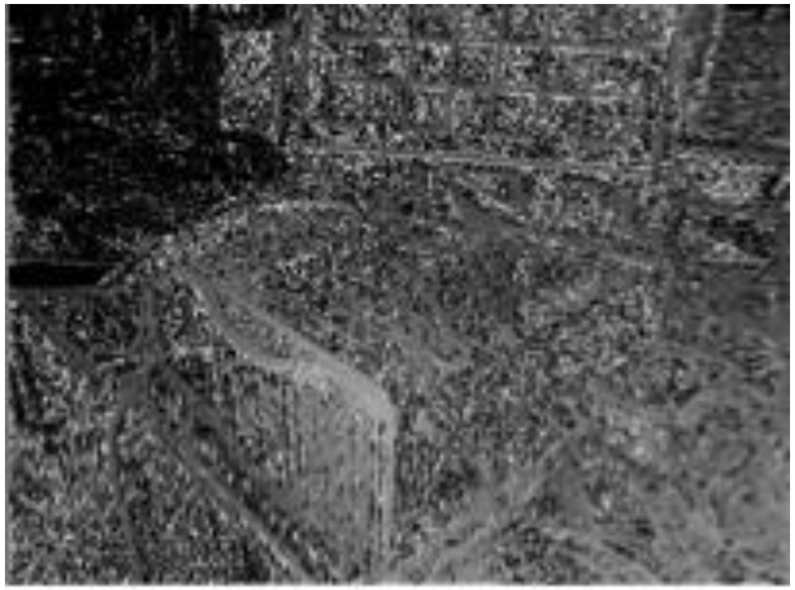
- **Input**: calibrated images from several viewpoints (i.e., K, R, T are known for each camera, e.g., from SFM)
- **Output**: 3D object **dense** reconstruction

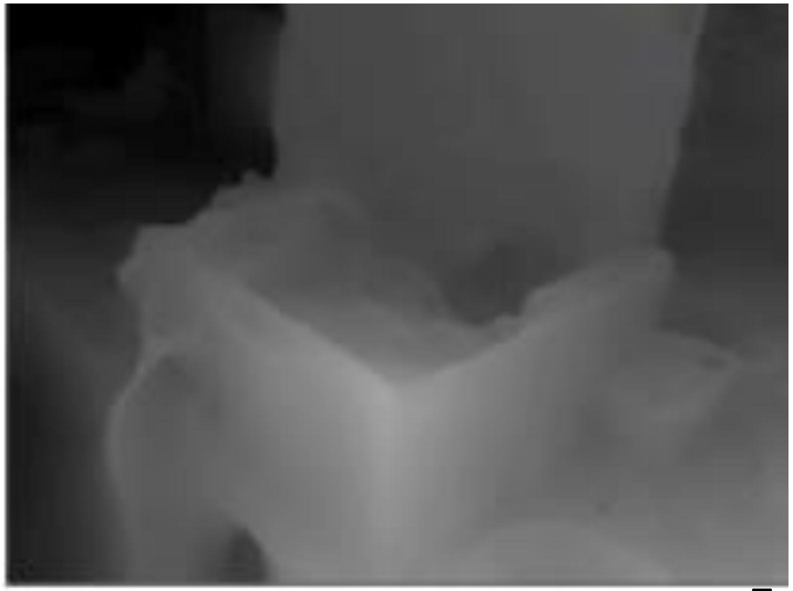


Dense reconstruction workflow

Step 1: Local methods

- Estimate depth for every pixel independently (how do we compute correspondences for *every* pixel?)



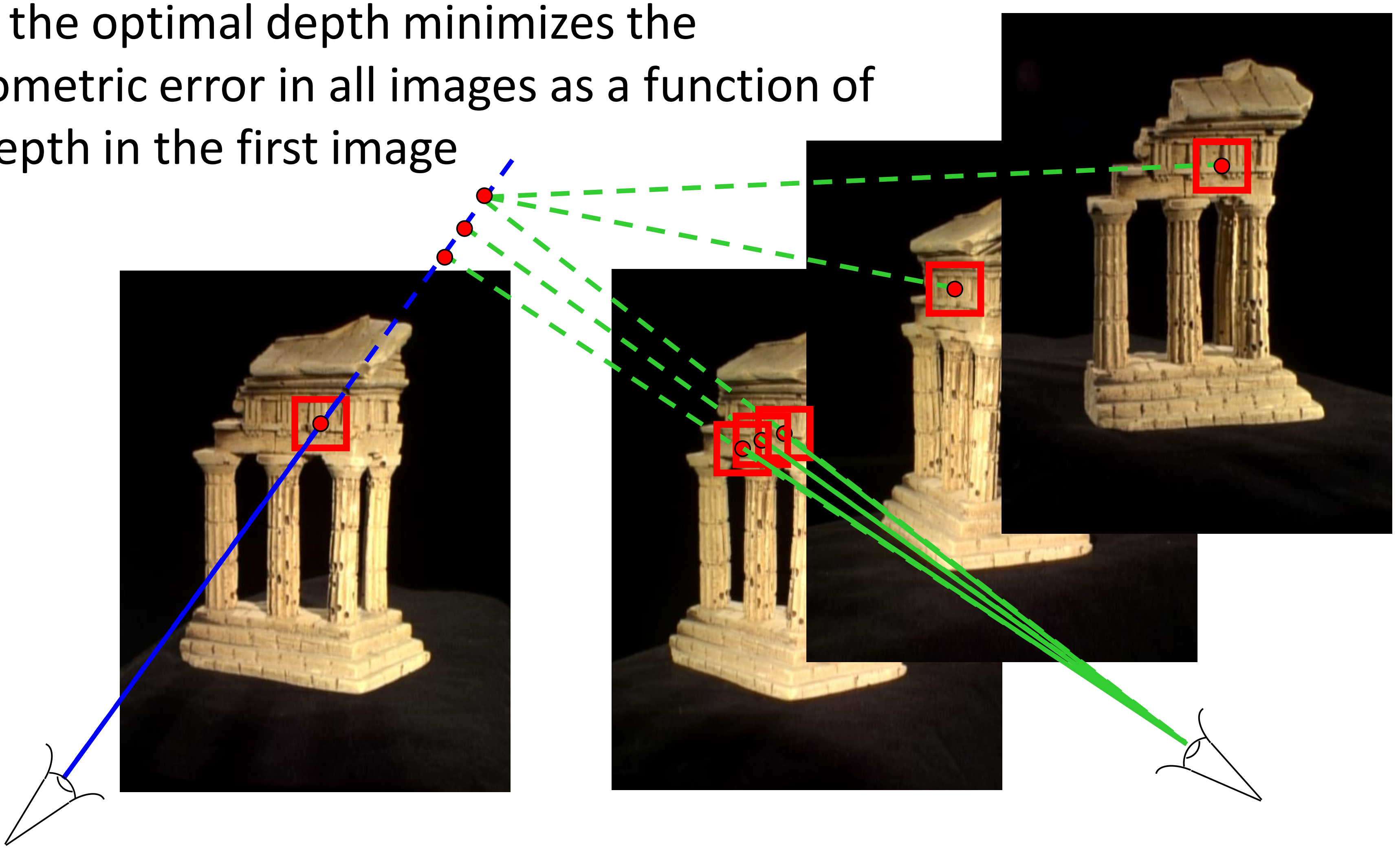


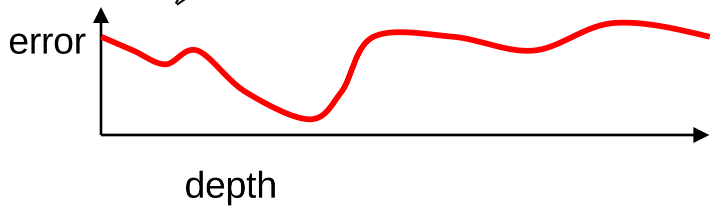
Step 2: Global methods

- Refine the depth surface as a whole by enforcing smoothness constraint

Photometric error (SSD)

IDEA: the optimal depth minimizes the photometric error in all images as a function of the depth in the first image

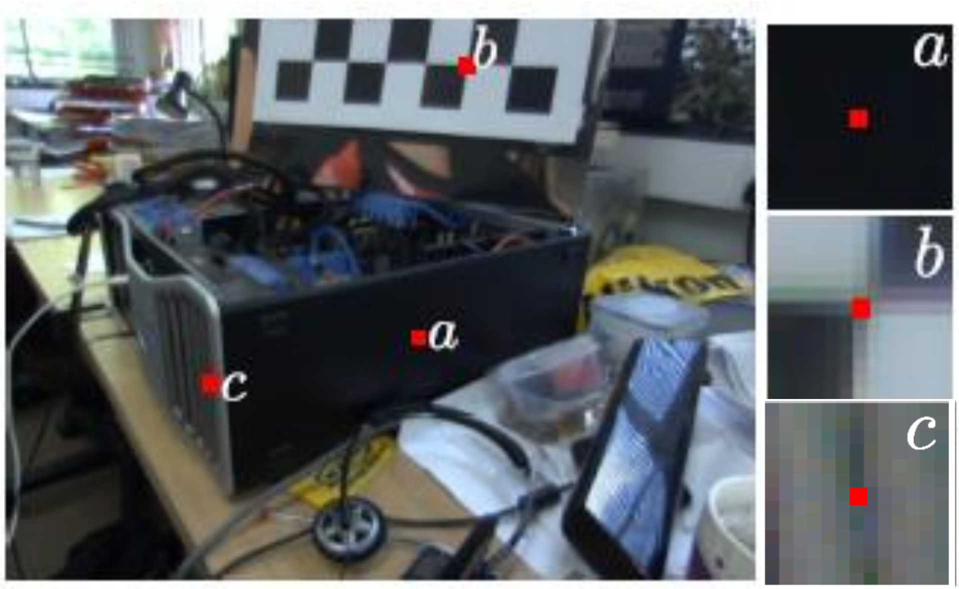




This error plot is derived for every combination of the reference image and any subsequent image

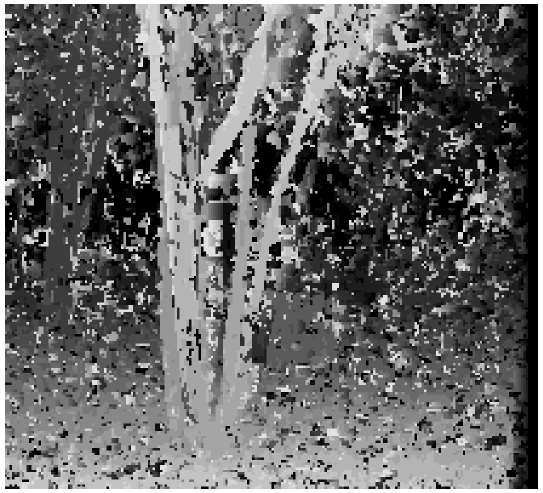
Aggregated Photometric Error

- Dense reconstruction requires establishing dense correspondences
- Correspondences are computed based on photometric error:
  - SSD between corresponding patches of intensity values (min patch size: 1x1 pixels)
  - What are the pros and cons of using small or large patches? (recall from stereo: see next slide)
- Not all the pixels can be matched reliably
  - Viewpoint and illumination changes, occlusions
- Take advantage of many small-baseline views where high quality matching is possible (why?)

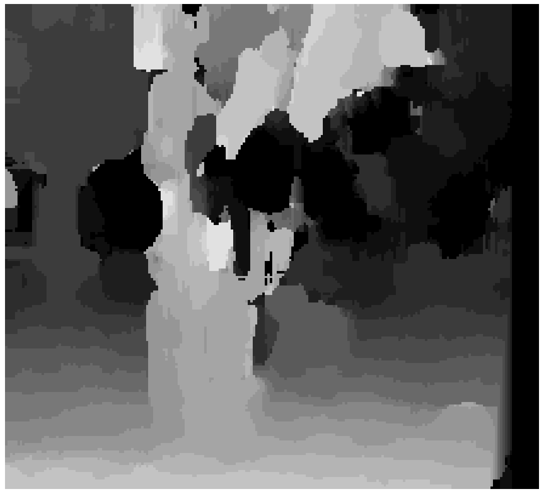


[Newcombe et al. 2011]

Review from stereo vision: Effects of window size



W = 3

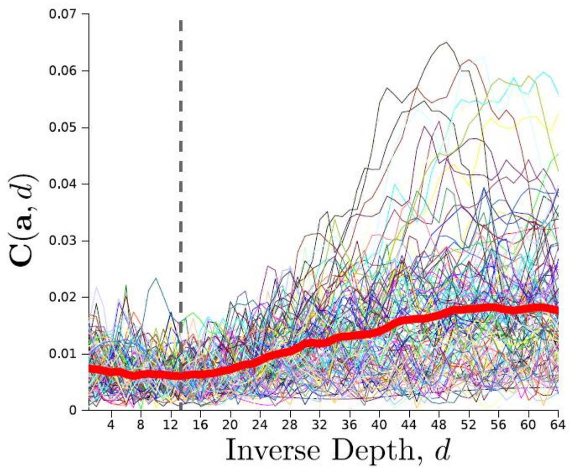


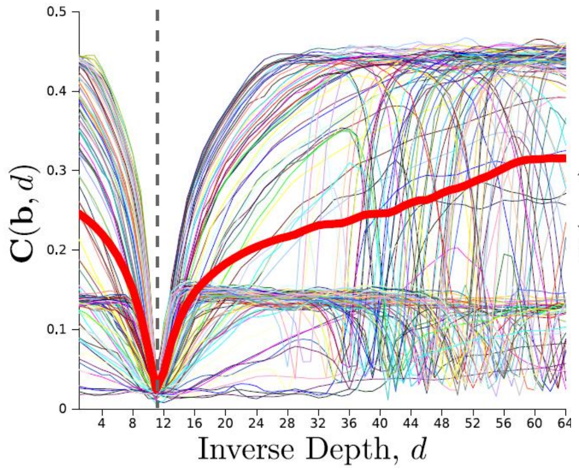
W = 20

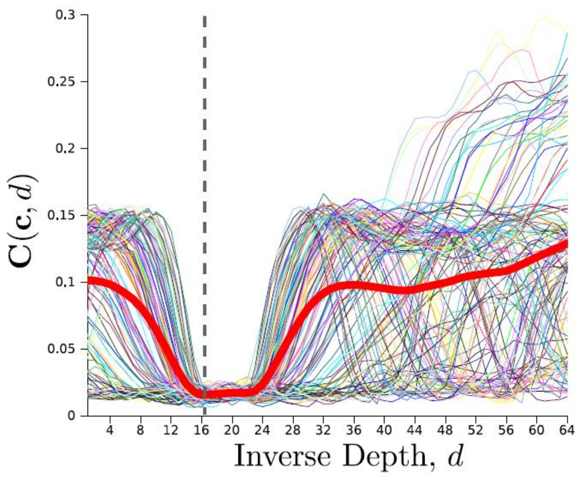
- Smaller window
  - + More detail
  - – More noise

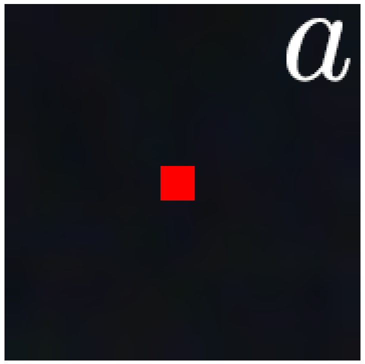
- Larger window
  - + Smoother disparity maps
  - – Less detail

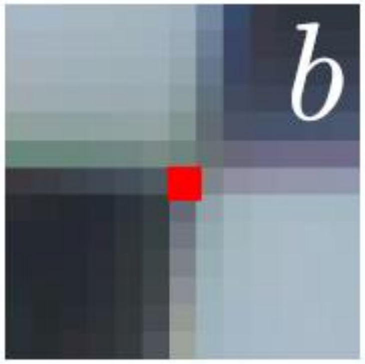
Aggregated Photometric Error

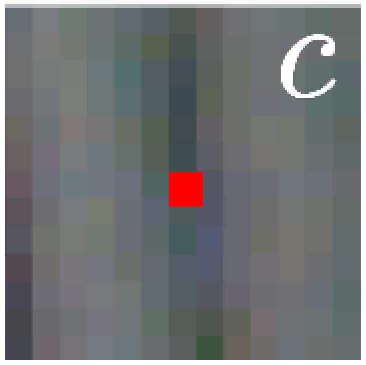












- Aggregated photometric error for flat regions (a) and *edges parallel to the epipolar line* (c) show flat valleys (plus noise)
- For distinctive features (corners as in (b) or blobs), the aggregated photometric error has one clear minimum.
- Non distinctive features (e.g., from repetitive texture) will show multiple minima

Disparity Space Image (DSI)

- For a given image point (u, v) and for discrete depth hypotheses d, the aggregated photometric error $C(u, v, d)$ with respect to the reference image I_R can be stored in a volumetric 3D grid called the **Disparity Space Image (DSI)**, where each voxel has value:

$$C(u, v, d) = \sum_{k=R+1}^{R+n-1} \rho\big(I_R(u, v) - I_k(u', v', d)\big)$$

  Where n is the number of images considered and $I_k(u', v', d)$ is the patch of intensity values in the k-th image centered on the pixel (u', v') corresponding to the patch $I_R(u, v)$ in the reference image I_R and depth hypothesis d; thus, formally:

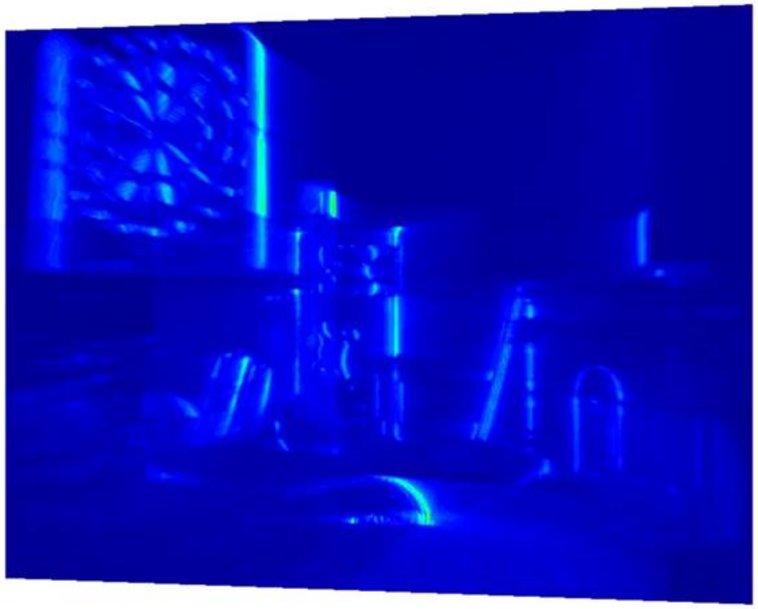
$$I_k(u', v', d) = I_k\left(\pi\left(T_{k,R}(\pi^{-1}(u, v) \cdot d)\right)\right)$$

- $\rho(\cdot)$ is the photometric error (SSD) (e.g. L_1, L_2, Tukey, or Huber norm)

Disparity Space Image (DSI)

Reference image

DSI



240 x 180 x 100 voxels

- Image resolution: 240x180 pixels
- Number of disparity (depth) levels: 100
- DSI:
  - size: 240x180x100 voxels; each voxel contains the aggregated photometric cost $C(u, v, d)$
  - white = low aggregated photometric error
  - blue = high aggregated photometric error

Solution

The solution to the depth estimation problem is to find *a* **function $\boldsymbol{d(u,v)}$** in the DSI that satisfies two criteria:

Minimum aggregated photometric error:

$$arg\min_{d}\sum\nolimits_{(u,v)} C(u,v,d(u,v)))$$ (local methods)

AND

Piecewise smooth (global methods)

Solution

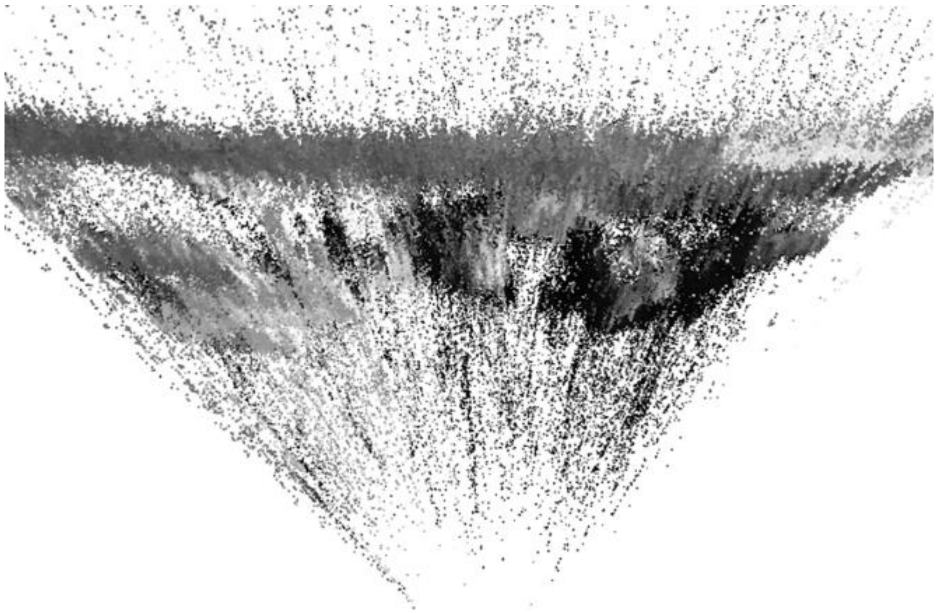
The solution to the depth estimation problem is to find *a* **function** $\boldsymbol{d(u,v)}$ in the DSI that satisfies two criteria:

Minimum aggregated photometric error:

$$arg \min_{d} \sum_{(u,v)} C(u, v, d(u, v)))$$ (local methods)

AND

Piecewise smooth (global methods)



First reconstruction via local methods

Solution

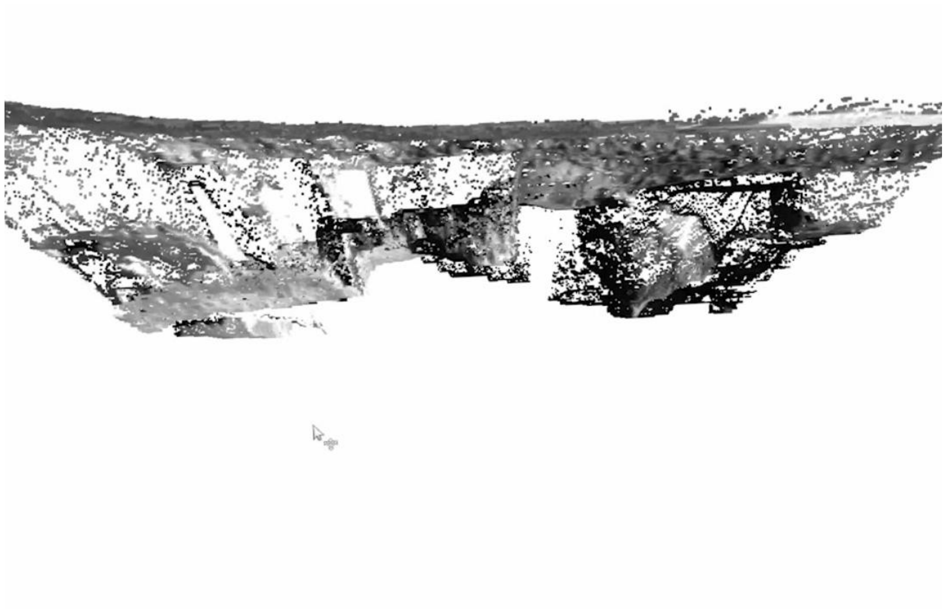
The solution to the depth estimation problem is to find *a* **function $\boldsymbol{d(u,v)}$** in the DSI that satisfies two criteria:

Minimum aggregated photometric error:

$$arg\min_{d}\sum_{(u,v)} C(u,v,d(u,v)))$$ (local methods)

AND

Piecewise smooth (global methods)



Effect of global methods: soothing

Solution

Global methods

- Formulated in terms of energy minimization
- The objective is to find a *surface* $d(u,v)$ that minimizes a global energy

$$E(d) = \underbrace{E_d(d)}_{\text{Data term}} + \underbrace{\lambda E_s(d)}_{\text{Regularization term}}$$

Data term: $$E_d(d) = \sum_{(u,v)} C(u,v,d(u,v))$$

Regularization term: $$E_s(d) = \sum_{(u,v)} \left(\frac{\partial d}{\partial u}\right)^2 + \left(\frac{\partial d}{\partial v}\right)^2$$

where:

- λ controls the tradeoff data / regularization. What happens as λ increases?

Regularized depth maps

- **The regularization term** $E_s(d)$
  - ***Smooths*** *non smooth surfaces* (results of noisy measurements) as well as discontinuities
  - ***Fills the holes***

Final depth image for increasing λ
[Newcombe et al. 2011]

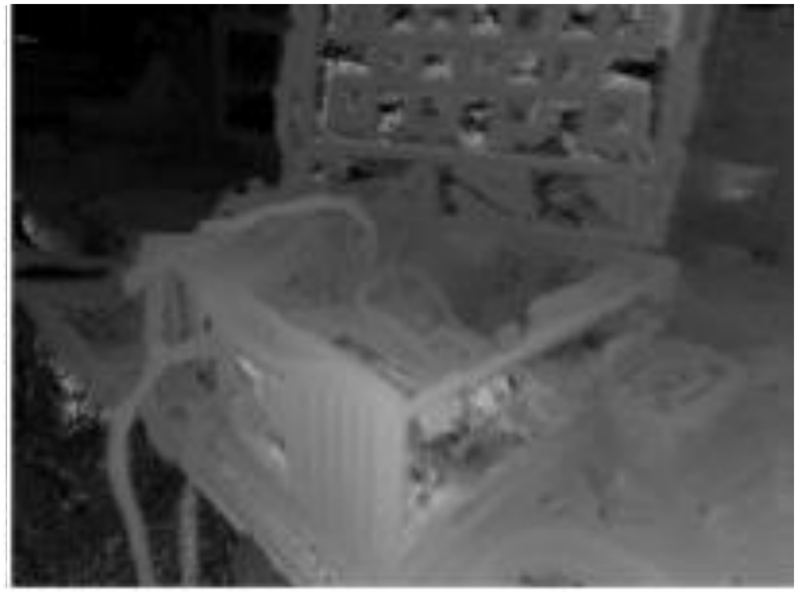
Regularized depth maps

- **The regularization term** $E_s(d)$
  - ***Smooths*** *non smooth surfaces* (results of noisy measurements) as well as discontinuities
  - ***Fills the holes***

Final depth image for increasing λ
[Newcombe et al. 2011]

Regularized depth maps

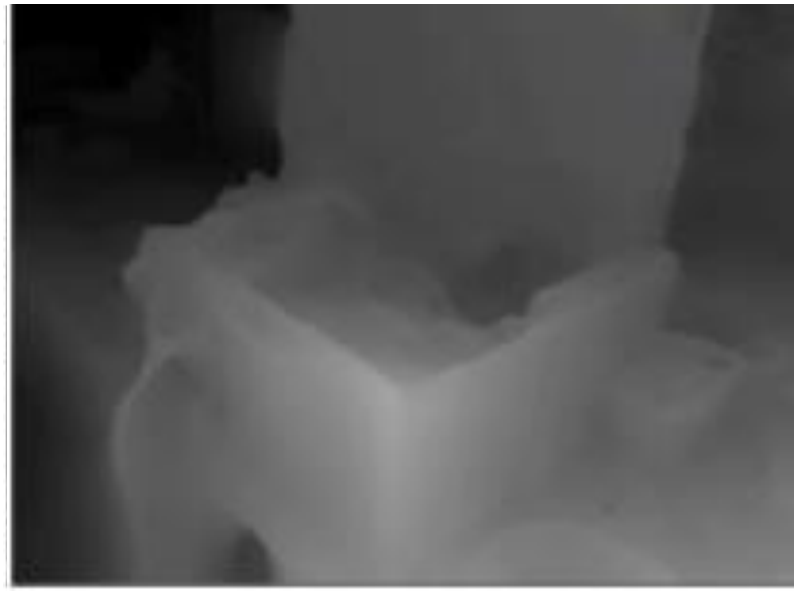
- **The regularization term** $E_s(d)$
  - ***Smooths*** *non smooth surfaces* (results of noisy measurements) as well as discontinuities
  - ***Fills the holes***



Final depth image for increasing λ
[Newcombe et al. 2011]

Regularized depth maps

- **The regularization term** $E_s(d)$
  - ***Smooths*** *non smooth surfaces* (results of noisy measurements) as well as discontinuities
  - ***Fills the holes***



Final depth image for increasing λ
[Newcombe et al. 2011]

How to deal with actual scene depth discontinuities?

- **Problem:** since we don't know a priori where there depth discontinuities, we can make the following assumption:

depth *discontinuities* ***coincide*** with **intensity** *discontinuities* (i.e., image gradients)

- **Solution: control regularization term** according to image gradient

$$E_s(d) = \sum_{(u,v)} \left(\frac{\partial d)}{\partial u}\right)^2 \rho_I \left(\frac{\partial I)}{\partial u}\right)^2 + \left(\frac{\partial d)}{\partial v}\right)^2 \rho_I \left(\frac{\partial I)}{\partial v}\right)^2$$

where ρ_I is some monotonically decreasing function of image gradients:

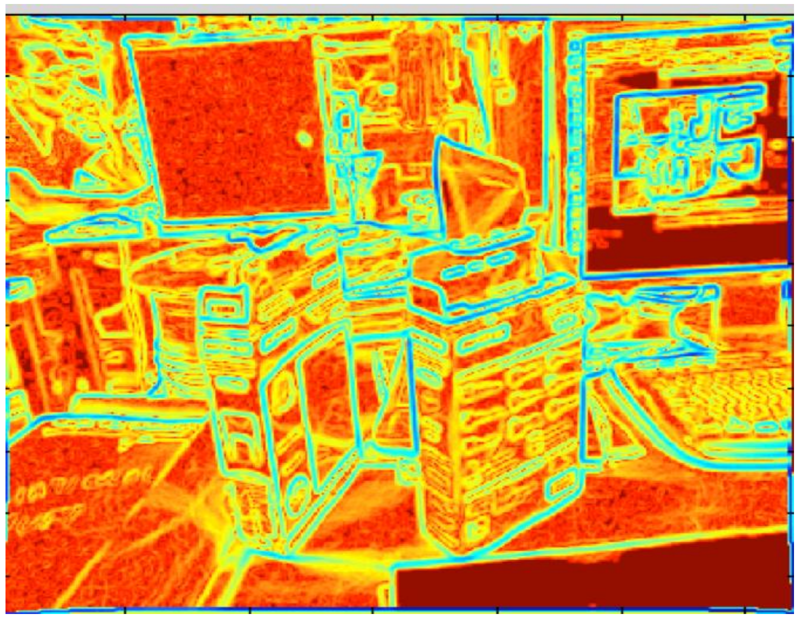
– **high for small image gradients** (i.e., regularization term dominates)

– **low** for **high image gradients** (i.e., data term dominates)

Effect of ρ_I on intensity discontinuities

Reference image

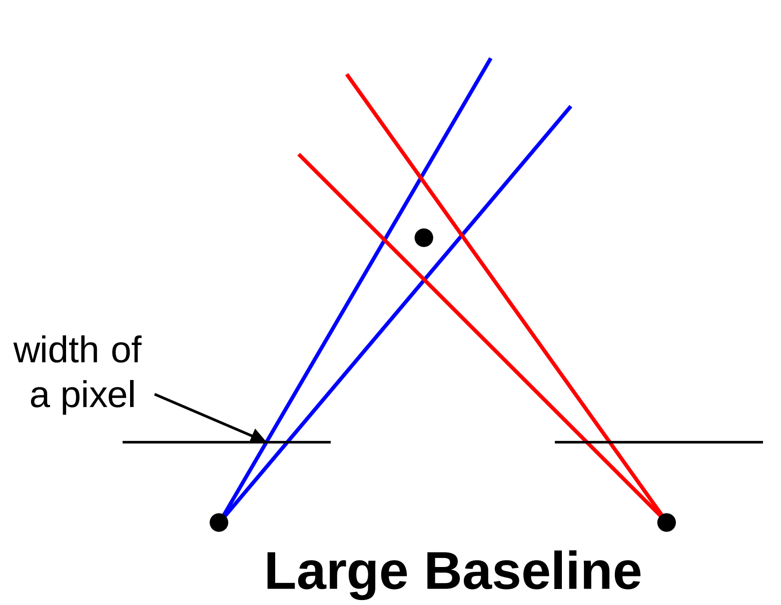
ρ_I

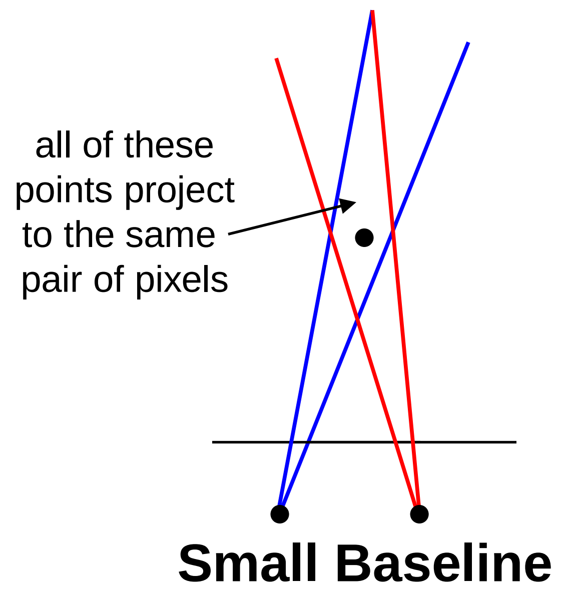


ρ_I is some monotonically decreasing function of image gradients:

- **high for small image gradients** (i.e., regularization term dominates)
- **low** for **high image gradients** (i.e., data term dominates)

Choosing the baseline between subsequent frames





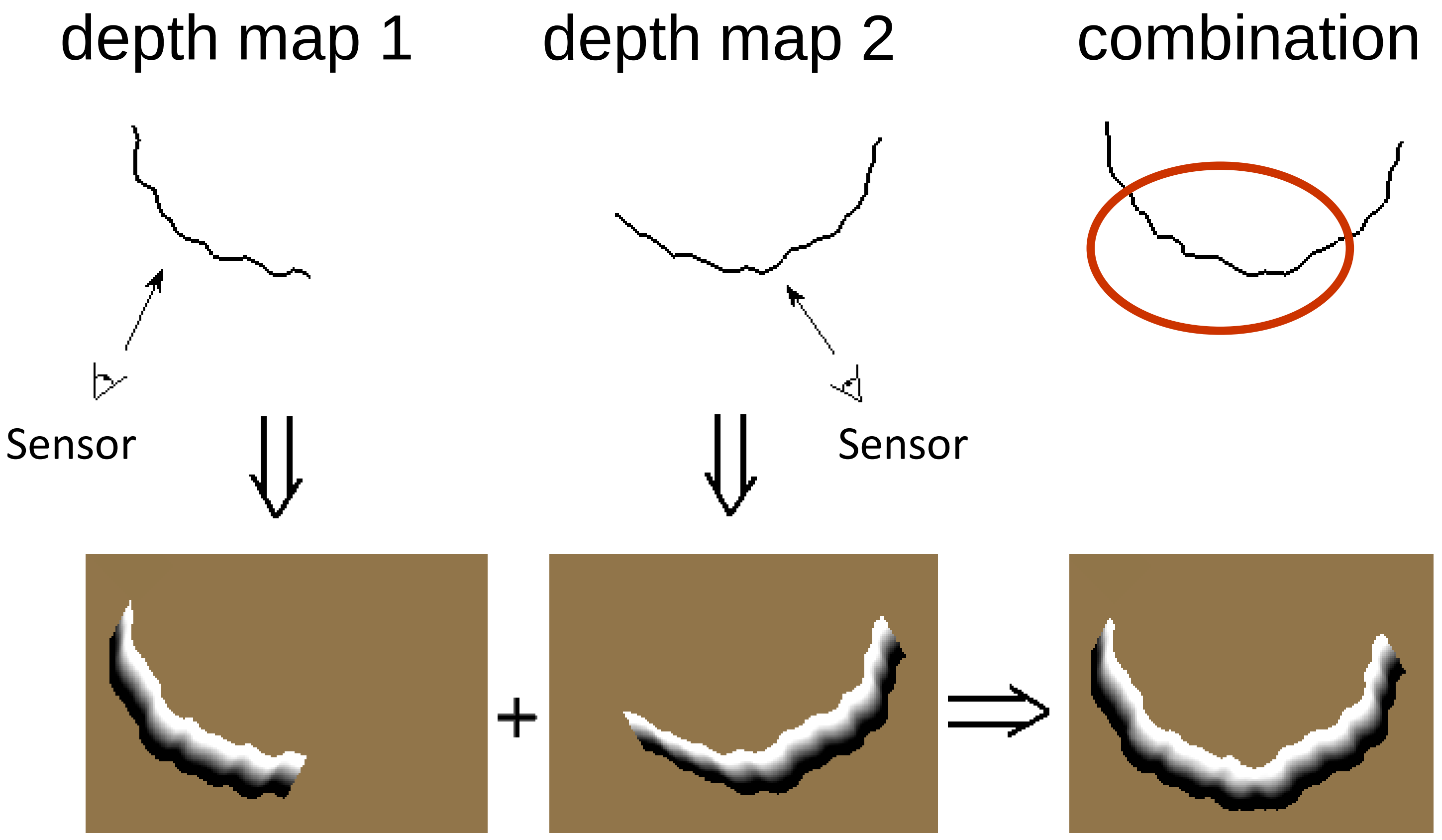
What's the optimal baseline ?

- Too large: ***difficult search problem*** due to wide view point changes
- Too small: ***large depth error***

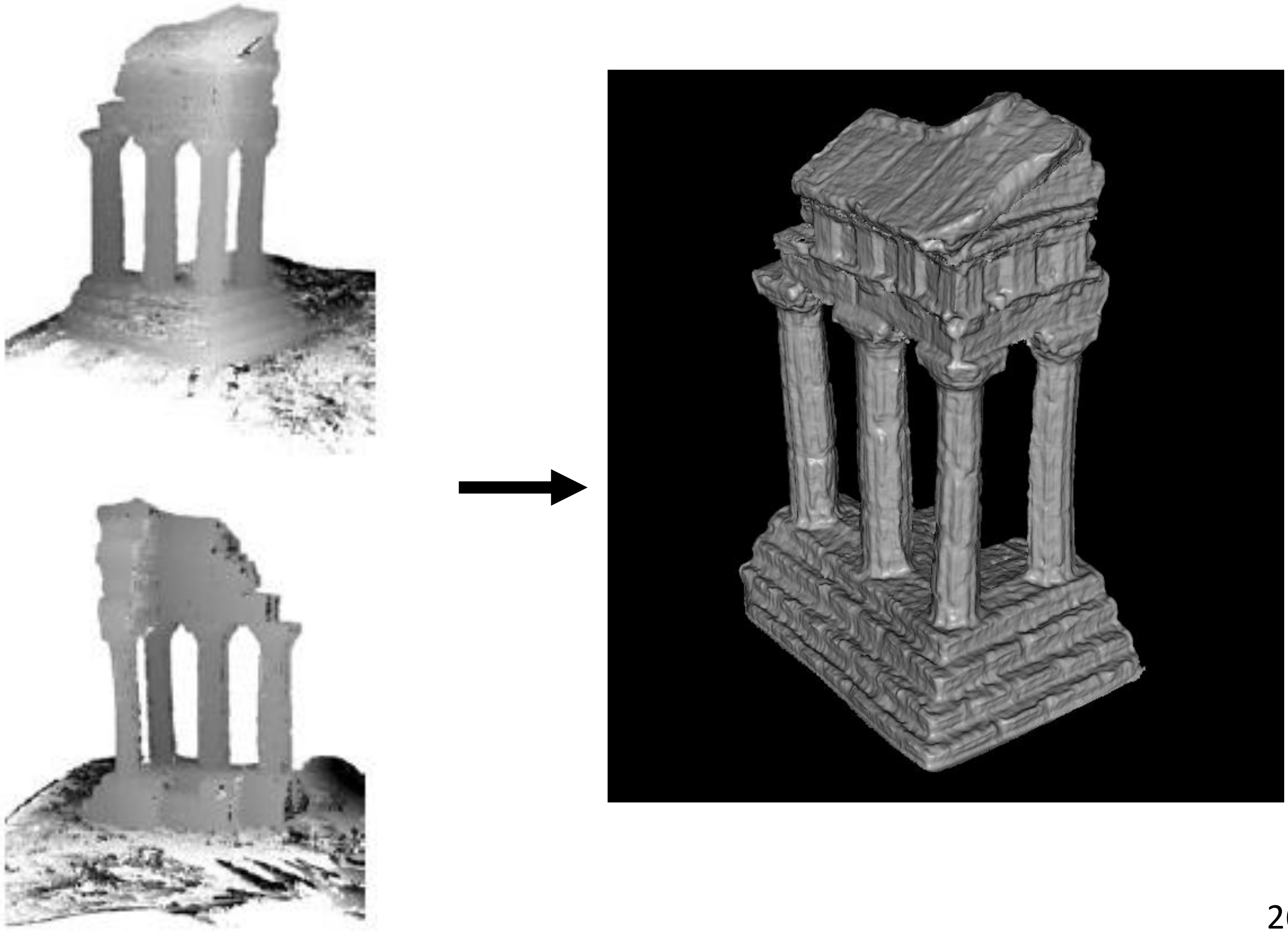
Solution

- Obtain depth map from **small baselines**
- When baseline becomes large (e.g., >10% of the avg scene depth), then **create new reference frame** (keyframe) and start a new depth map computation

Fusion of multiple depth maps

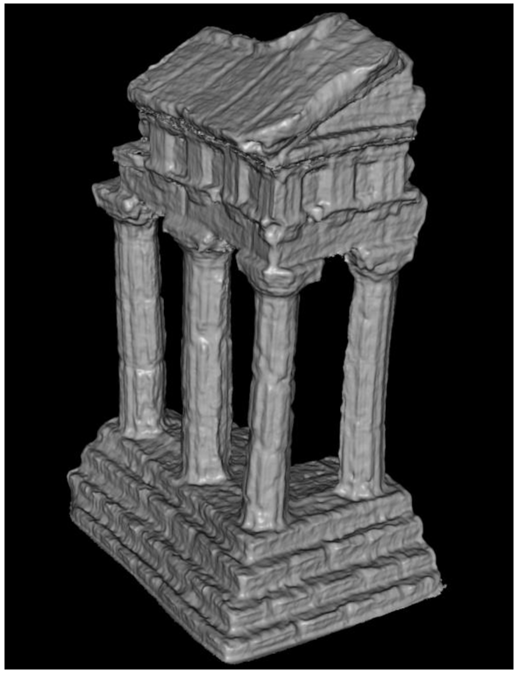


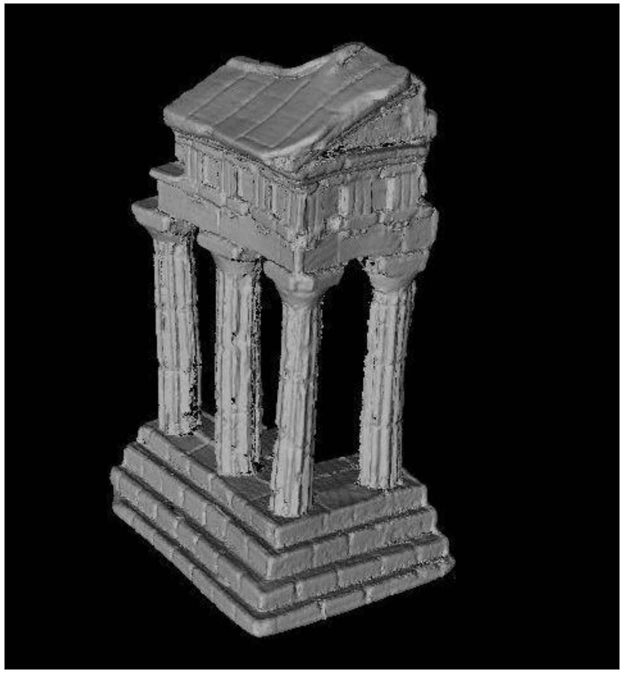
Fusion of multiple depth maps



Depth map fusion

input image

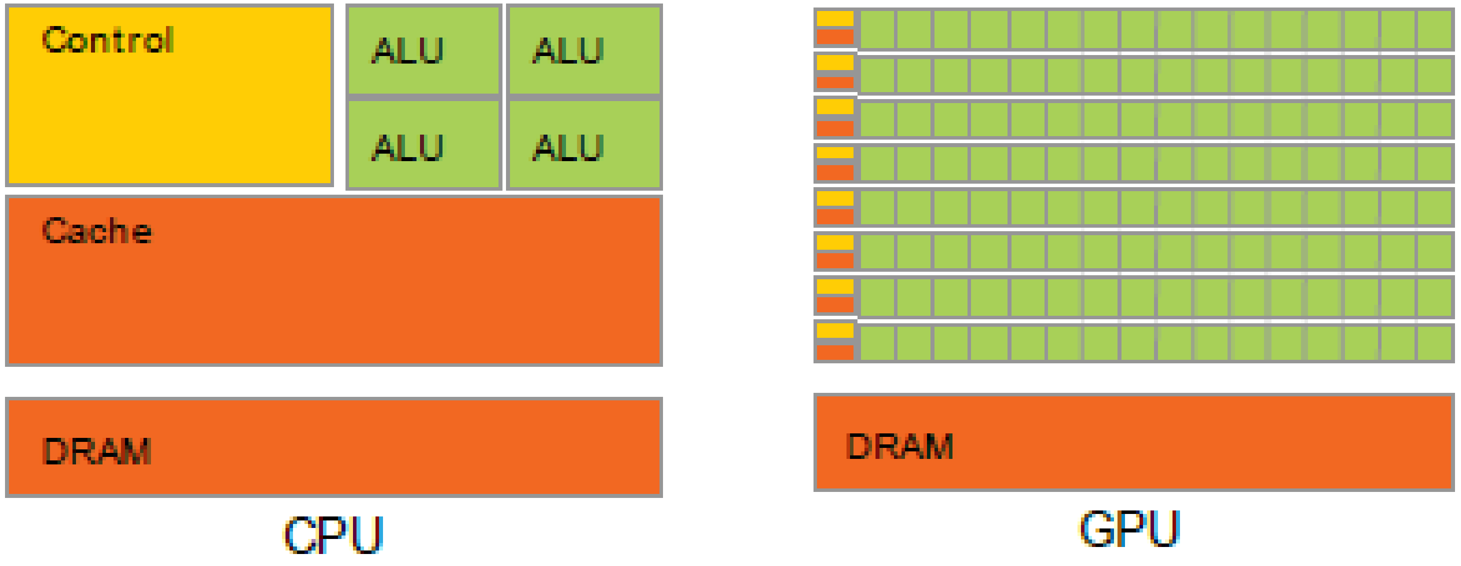

317 images (hemisphere)


ground truth model

Goesele, Curless, Seitz, 2006

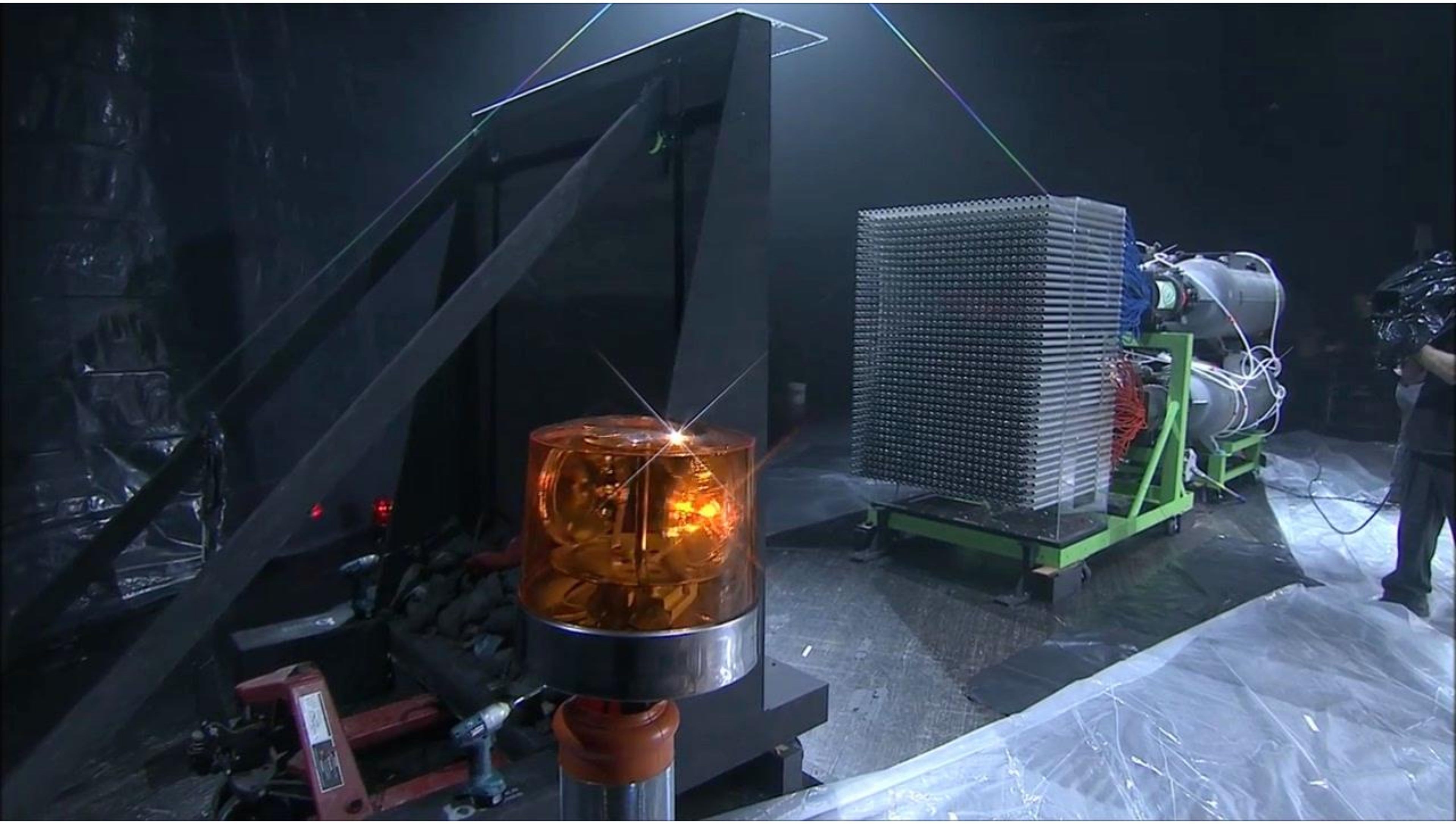
GPU: Graphics Processing Unit

- GPU performs calculations ***in parallel*** on thousands of cores (on a CPU a few cores optimized for *serial* processing)
- More transistors devoted to data processing
- More info: http://www.nvidia.com/object/what-is-gpu-computing.html#sthash.bW35IDmr.dpuf



ALU: Arithmetic Logic Unit

GPU: Graphics Processing Unit

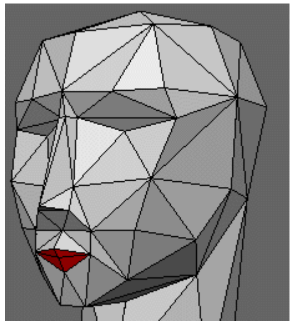


https://www.youtube.com/watch?v=-P28LKWTzrI

GPU Capabilities

- **Fast pixel processing**
  - Ray tracing, draw textures, shaded triangles faster than CPU
- **Fast matrix / vector operations**
  - Transform vertices
- **Programmable**
  - Shading, bump mapping
- **Floating-point support**
  - Accurate computations
- **Deep Learning**

Bump mapping


Shaded triangles

GPU for 3D Dense Reconstruction

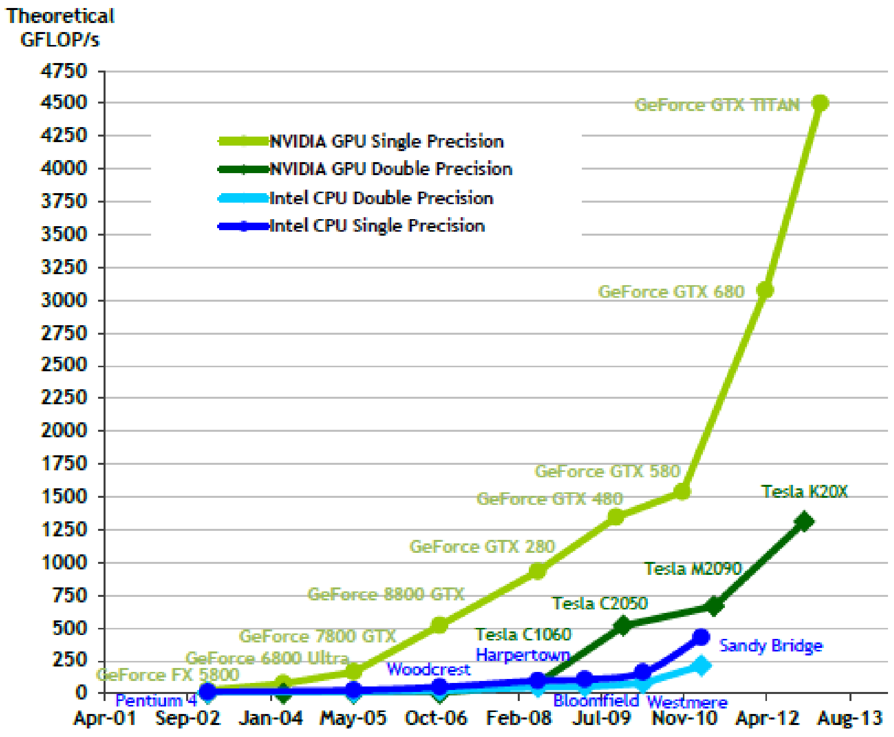
- **Image processing**
  - Filtering & Feature extraction (i.e., **convolutions**)
  - Warping (e.g., **epipolar rectification**, **homography**)

- **Multiple-view geometry**
  - Search for **dense correspondences**
    - *Pixel-wise* operations (SAD, **SSD**, NCC)
    - **Matrix and vector operations** (epipolar geometry)
  - **Aggregated Photometric Error** for multi-view stereo

- **Global optimization**
  - *Variational methods (i.e., regularization (smoothing))*
    - ***Parallel, in-place*** operations for gradient / divergence computation

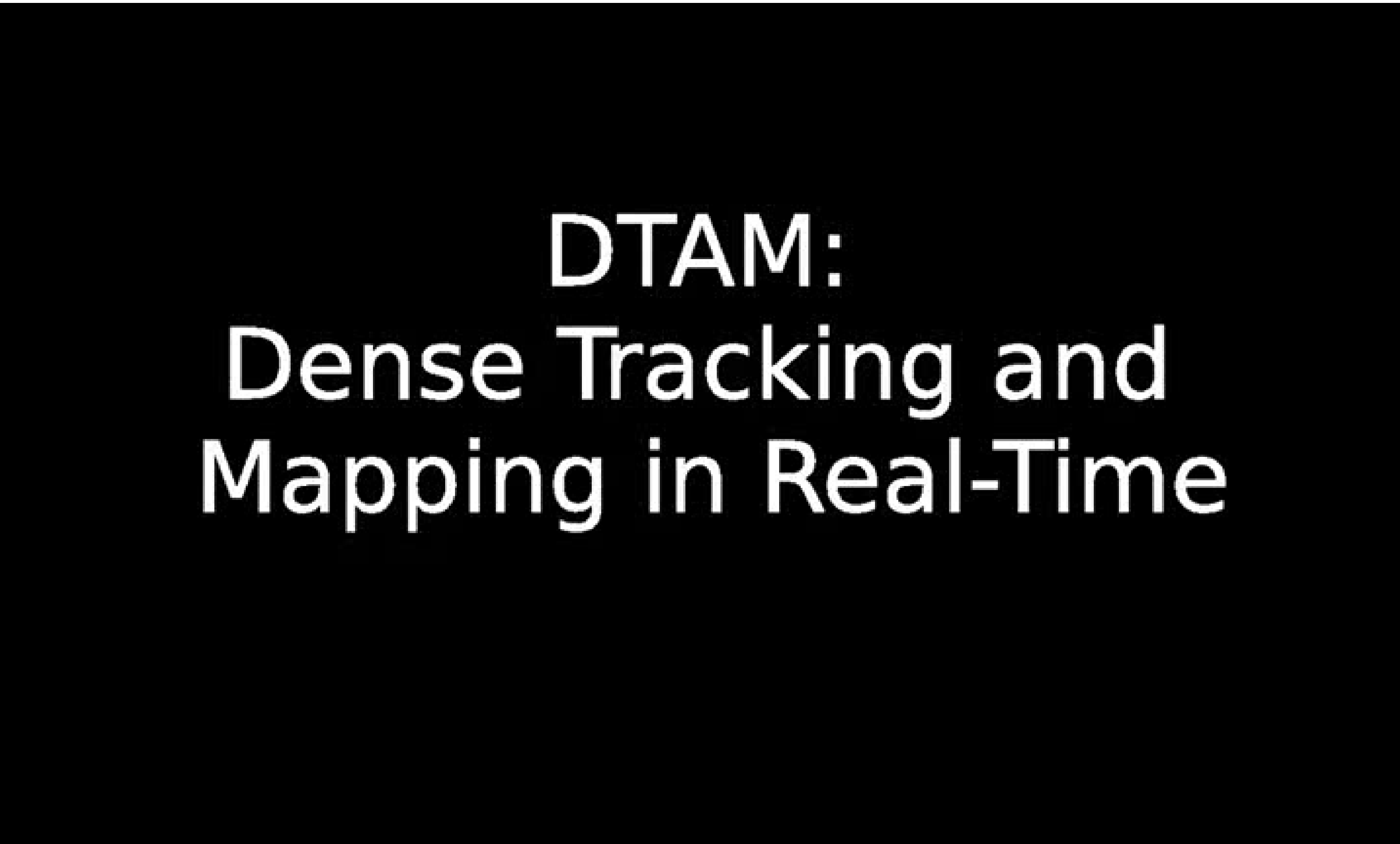
Why GPU

- GPUs run *thousands of lightweight threads **in parallel***
  - *Typically* on consumer hardware: 1024 threads per multiprocessor; 30 multiprocessor => **30k threads**.
  - Compared to CPU: 4 cores support 32 threads (with HyperThreading).
- Well suited for **data-parallelism**
  - The same instructions executed on multiple data in parallel
  - High **arithmetic intensity**: *arithmetic operations / memory operations*



[Source: nvidia]

DTAM: Dense Tracking and Mapping in Real-Time, ICCV'11 by Newcombe, Lovegrove, Davison

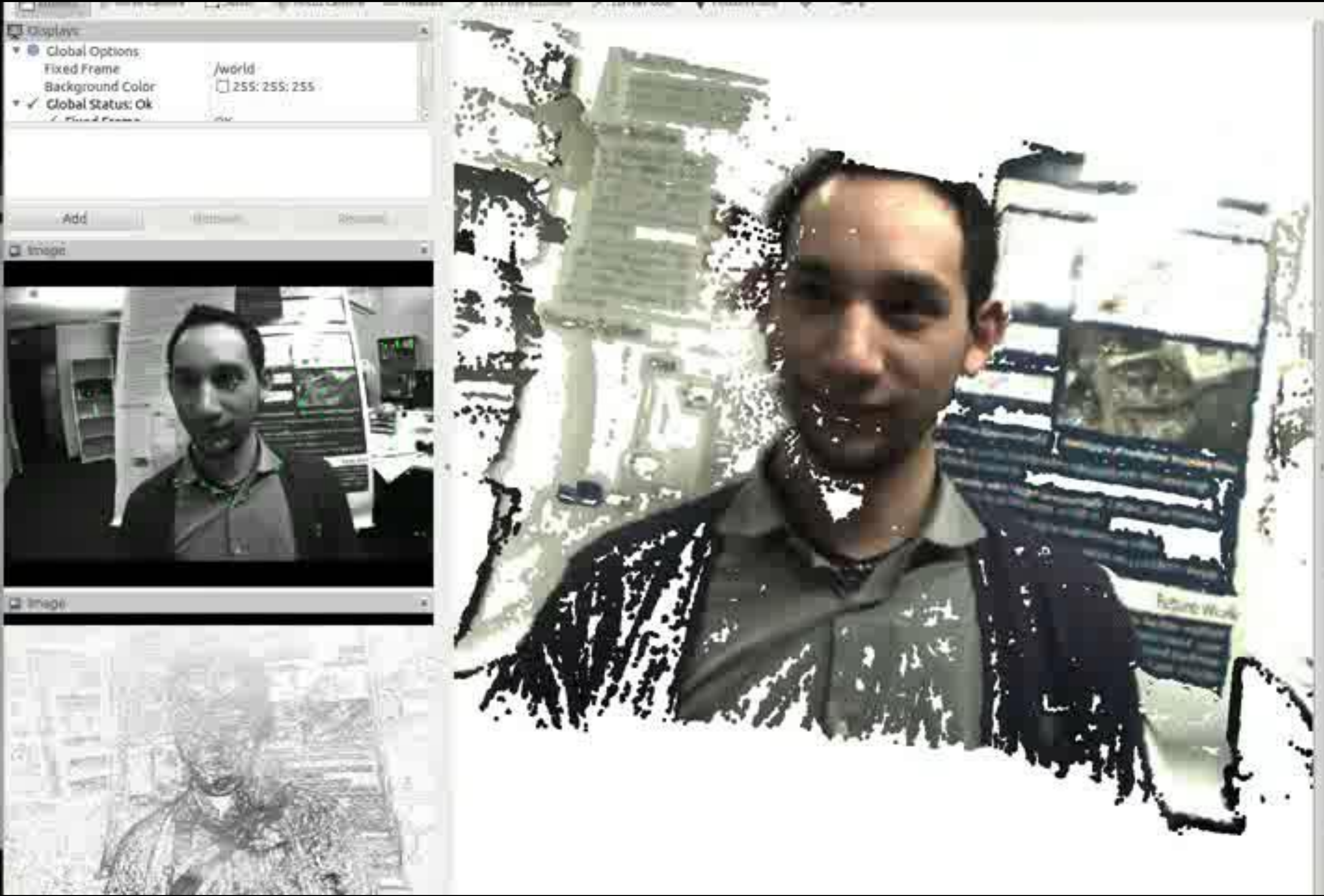


DTAM:
Dense Tracking and
Mapping in Real-Time

REMODE:
Regularized Monocular Dense Reconstruction

[M. Pizzoli, C. Forster, D. Scaramuzza, REMODE: Probabilistic, Monocular Dense Reconstruction in Real Time, IEEE International Conference on Robotics and Automation 2014]

Open source: *https://github.com/uzh-rpg/rpg_open_remode*



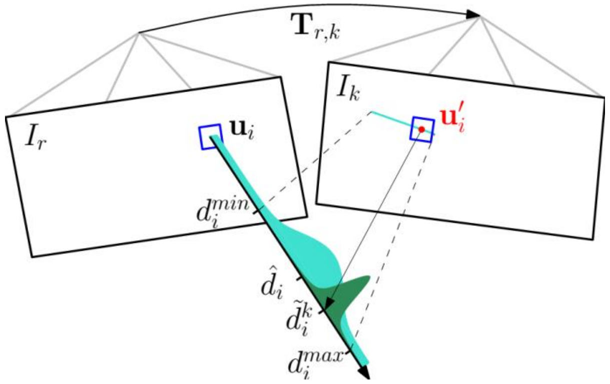
Global Options
Fixed Frame /world
Background Color 255; 255; 255
Global Status: Ok

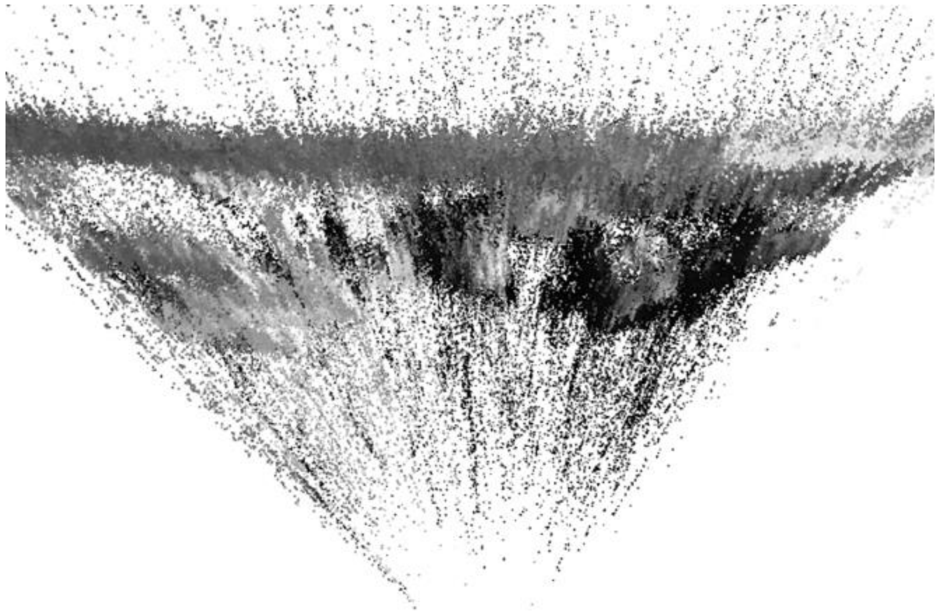
Monocular dense reconstruction in real-time from a hand-held camera

Stage-set from Gruber et al., "The City of Sights", ISMAR'10.

REMODE: Probabilistic, Monocular Dense Reconstruction in Real Time

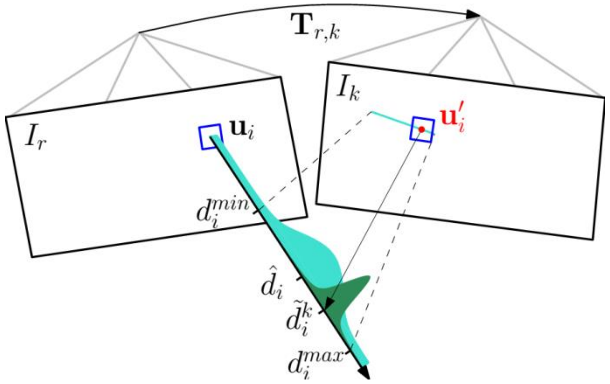
- Tracks every pixel (like DTAM) but **Probabilistically**
- Runs live on video streamed from MAV (50 Hz on GPU)
- Copes well with low texture surfaces





REMODE: Probabilistic, Monocular Dense Reconstruction in Real Time

- Tracks every pixel (like DTAM) but **Probabilistically**
- Runs live on video streamed from MAV (50 Hz on GPU)
- Copes well with low texture surfaces

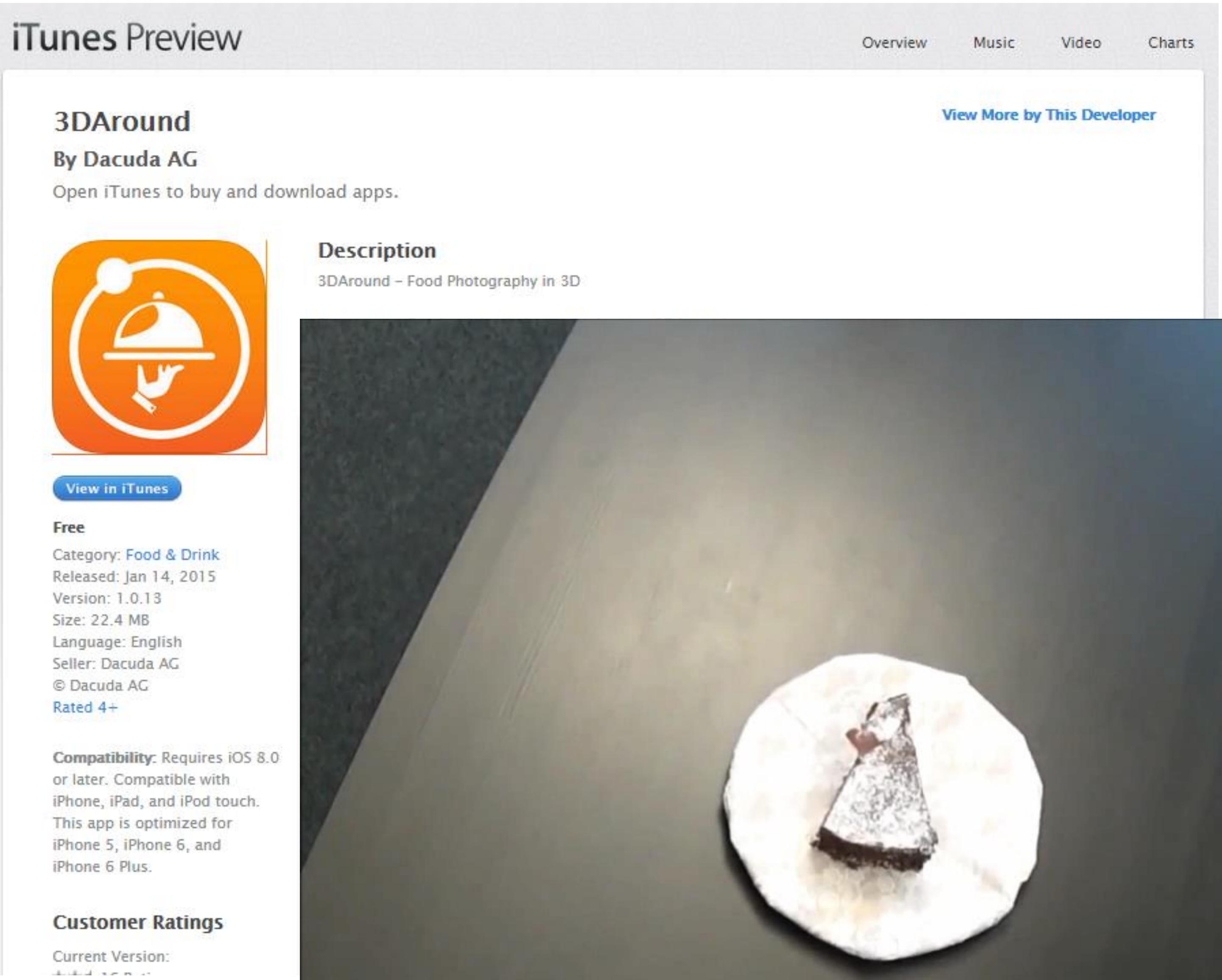


REMODE applied to autonomous flying 3D scanning

Live demonstration at the Firefighter Training Area of Zurich

3DAround iPhone App

Dacuda



iTunes Preview

3DAround

By Dacuda AG

Open iTunes to buy and download apps.

Description

3DAround – Food Photography in 3D

Free

Category: Food & Drink
Released: Jan 14, 2015
Version: 1.0.13
Size: 22.4 MB
Language: English
Seller: Dacuda AG
© Dacuda AG
Rated 4+

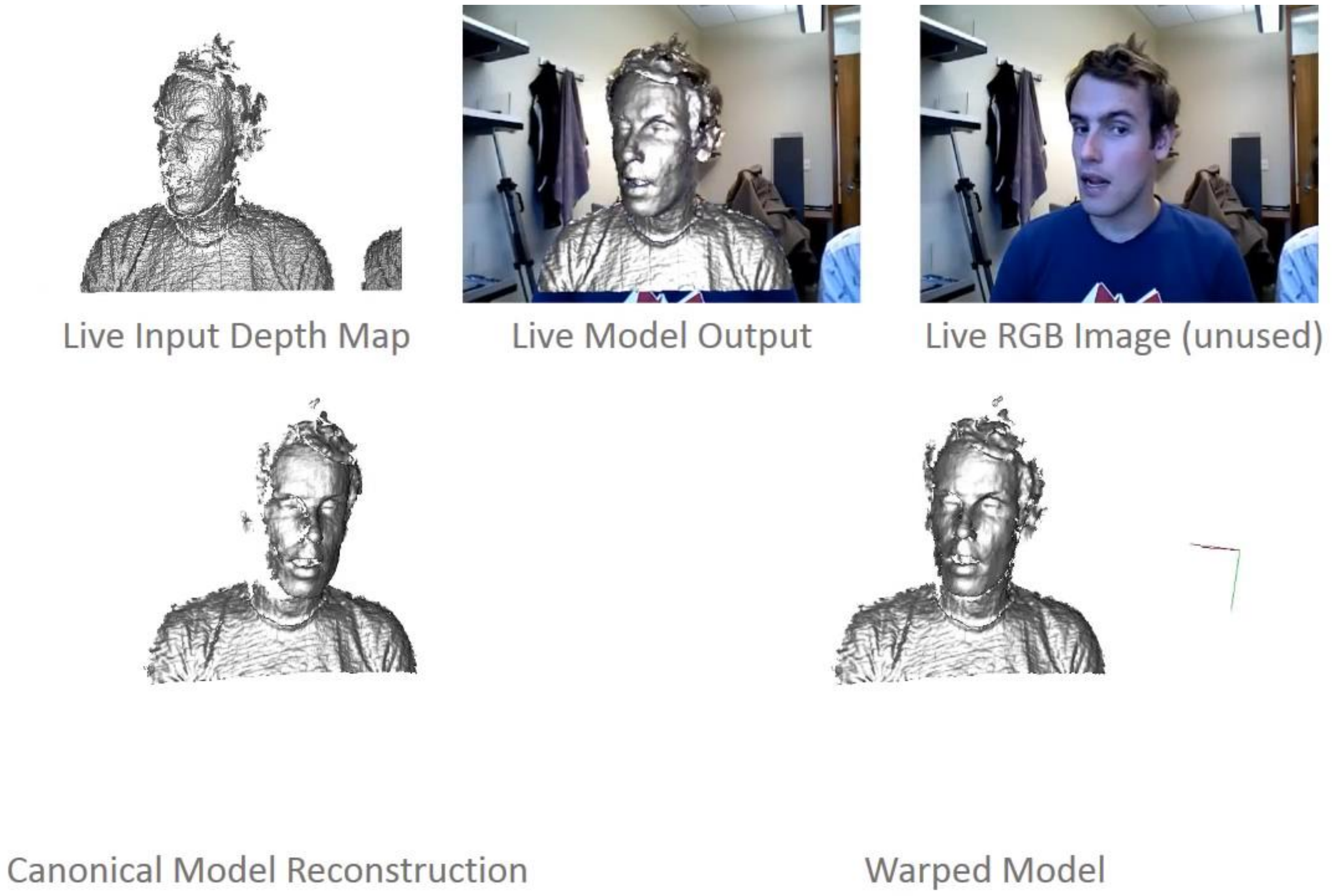
Compatibility: Requires iOS 8.0 or later. Compatible with iPhone, iPad, and iPod touch. This app is optimized for iPhone 5, iPhone 6, and iPhone 6 Plus.

Customer Ratings

Current Version:

DynamicFusion

- Reconstruction deforming scenes and tracking camera pose simultaneously with a RGBD camera

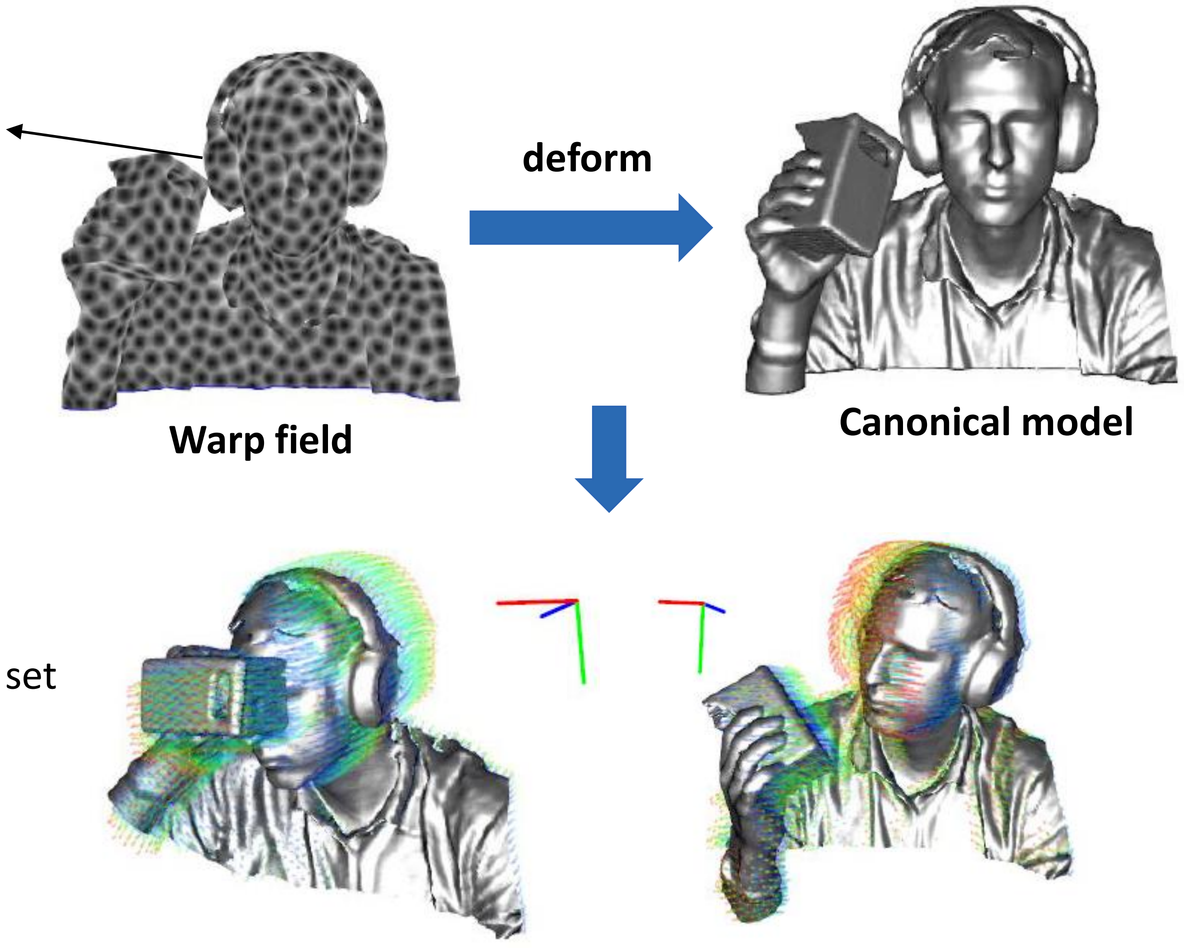


Newcombe et.al. DynamicFusion: Reconstruction and Tracking of Non-rigid Scenes in Real-Time. CVPR 2015, Best Paper Award.

DynamicFusion: scene representation

- How to represent the deformation of the scene?
  → Dense warp field

Each node stands for a rigid body motion that transforms (locally) the canonical model to the live frame.



We need to estimate a set of sparse nodes in the warp field per frame.

DynamicFusion: tracking and model update

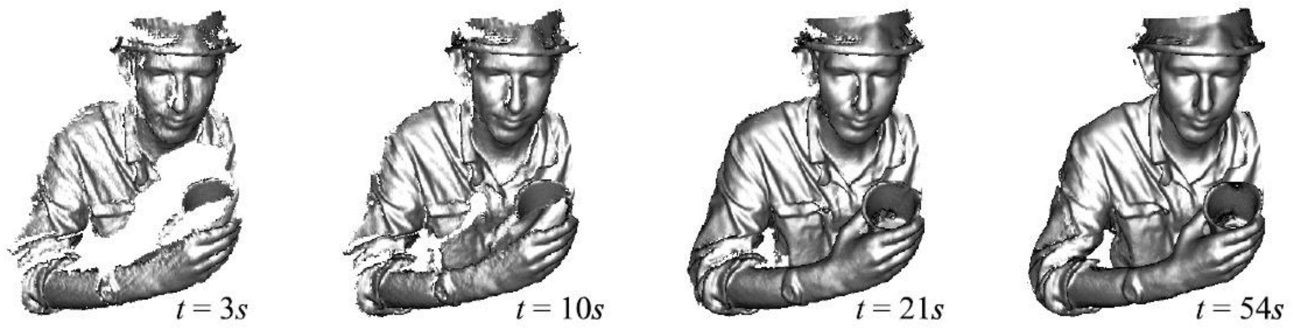
- Tracking: many parameters to optimize
  - Camera motion
  - The nodes in the warp field

W_t: warp field
D_t: depth map
V: canonical model

$$E(\mathcal{W}_t, \mathcal{V}, D_t, \mathcal{E}) = \mathbf{Data}(\mathcal{W}_t, \mathcal{V}, D_t) + \lambda \mathbf{Reg}(\mathcal{W}_t, \mathcal{E})$$

- **Data** term: The warped model should agrees well with the depth map.
- **Reg**ularization term: The warp field should be smooth.

- Model update: update the canonical model recursively
  → do not need to store all the depth images



Things to remember

- Aggregated Photometric Error
- Disparity Space Image
- Effects of regularization
- Handling discontinuities
- GPU

- Readings:
  - Chapter: 11.6 of Szeliski’s book

Understanding Check

Are you able to answer the following questions?

- ➢ Are you able to describe the multi-view stereo working principle? (aggregated photometric error)
- ➢ What are the differences in the behavior of the aggregated photometric error for corners, flat regions, and edges?
- ➢ What is the disparity space image (DSI) and how is it built in practice?
- ➢ How do we extract the depth from the DSI?
- ➢ How do we enforce smoothness (regularization) and how do we incorporate depth discontinuities (mathematical expressions)?
- ➢ What happens if we increase lambda (the regularization term)? What if lambda is 0? And if lambda is too big?
- ➢ What is the optimal baseline for multi-view stereo?
- ➢ What are the advantages of GPUs?