Occlusion-robust photo-consistency

George Vogiatzis
(g.vogiatzis@aston.ac.uk)

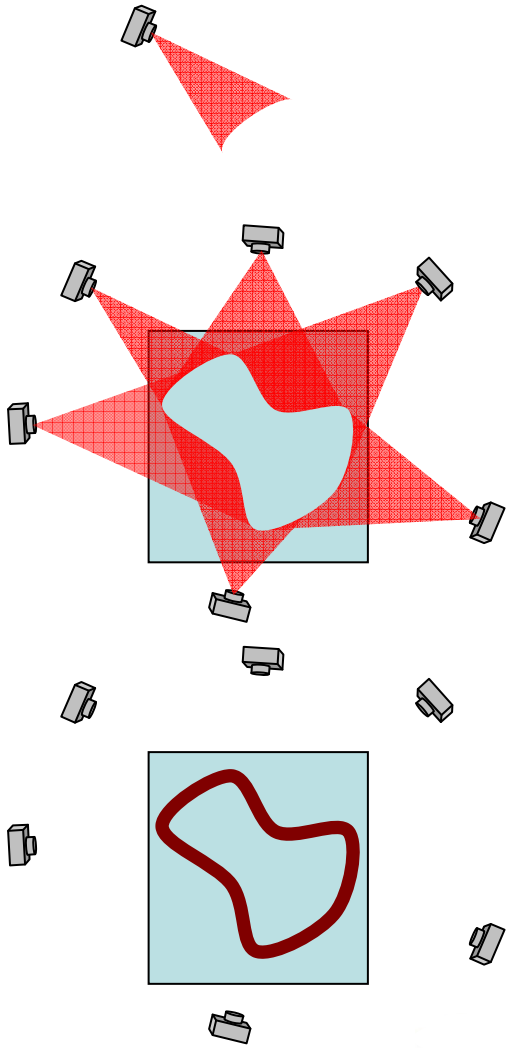
Aston University
Birmingham

Occlusion-robust photo-consistency

1. Compute depth-maps from each view using neighbouring views

2. Merge depth-maps into single volume

3. Extract 3d surface from this volume

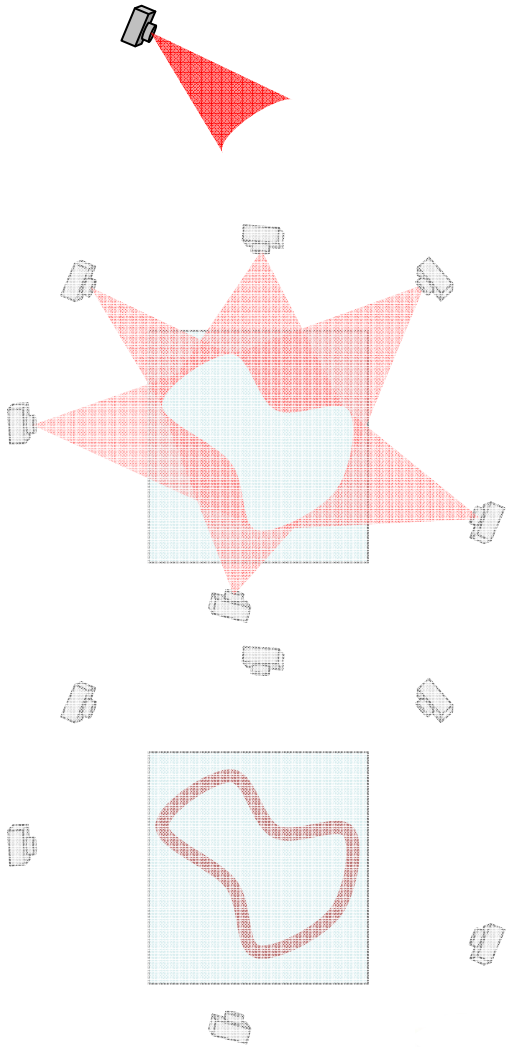


Occlusion-robust photo-consistency

1. Compute depth-maps from each view using neighbouring views

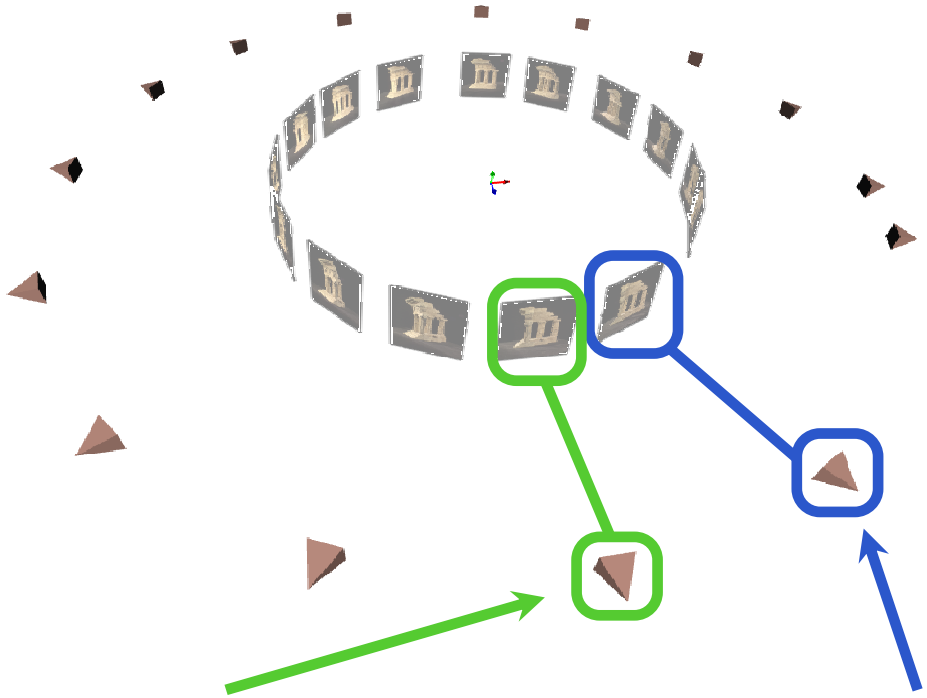
2. Merge depth-maps into single volume

3. Extract 3d surface from this volume



1. Compute depth-maps

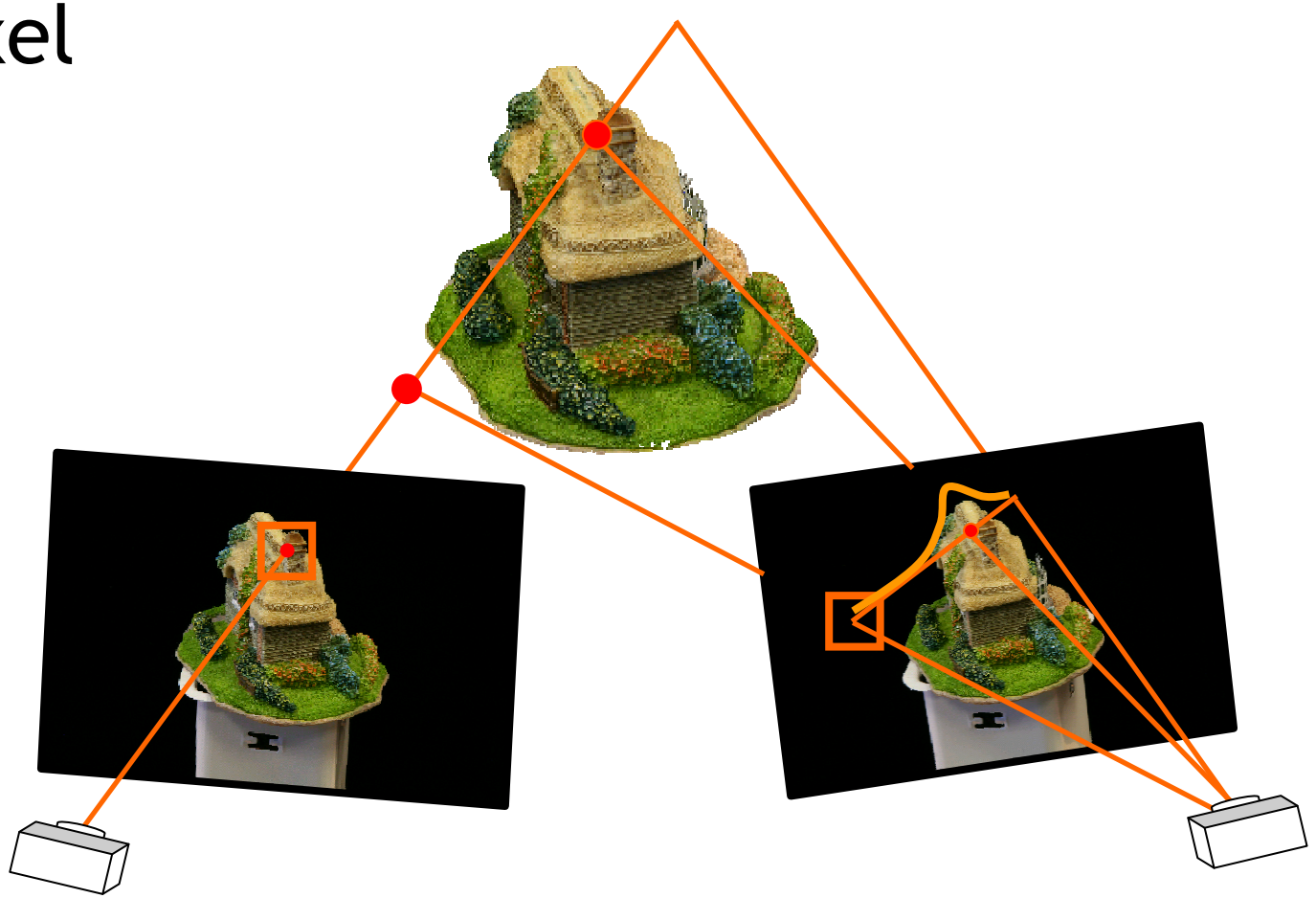
- Occlusion effects are small between neighbouring views



Select a reference image and a neighbouring image

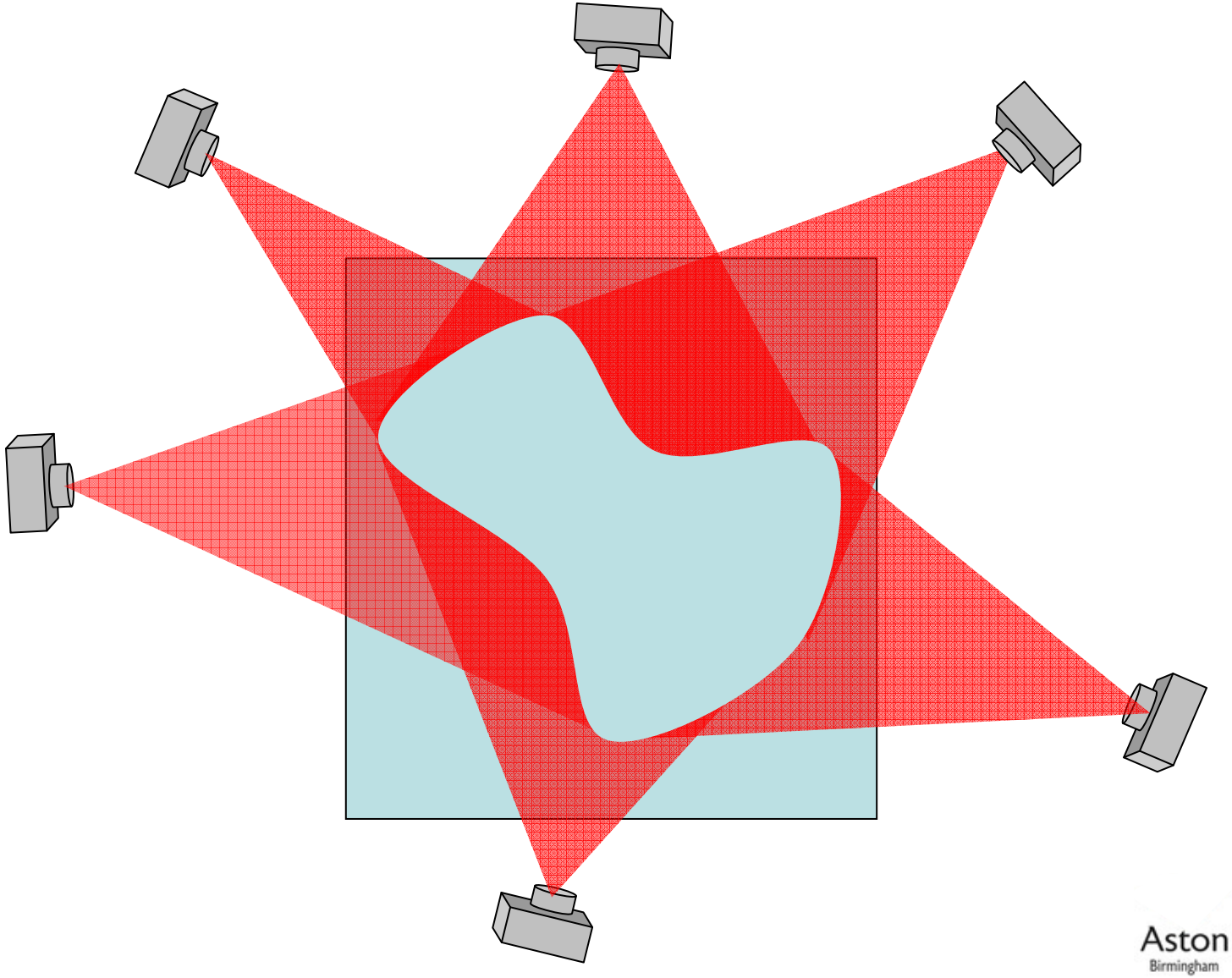
1. Compute depth-maps

- Use matching score to find best depth at pixel



2. Merge depth-maps

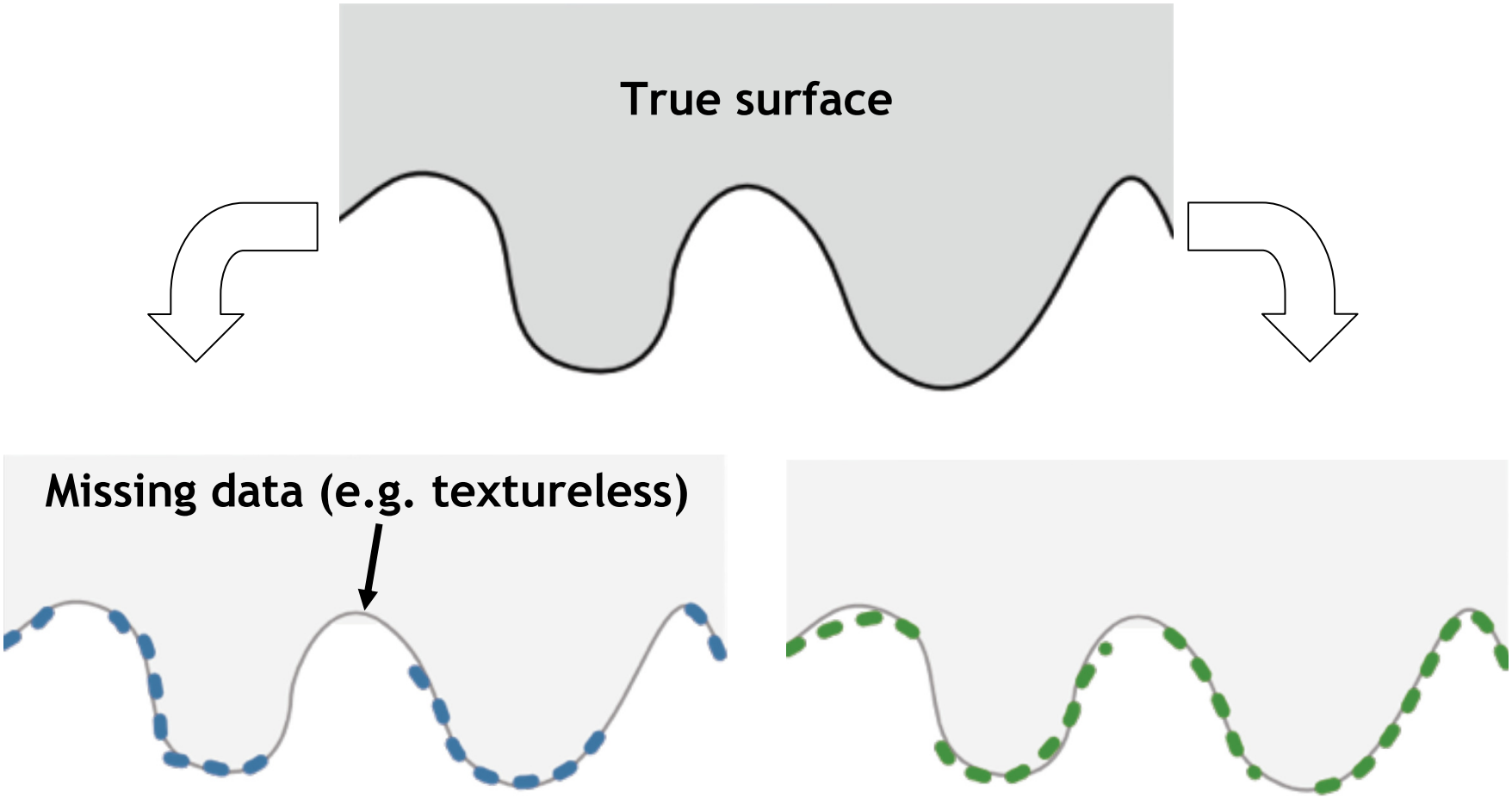
- Use efficient volumetric data-structure



Depth-map computation

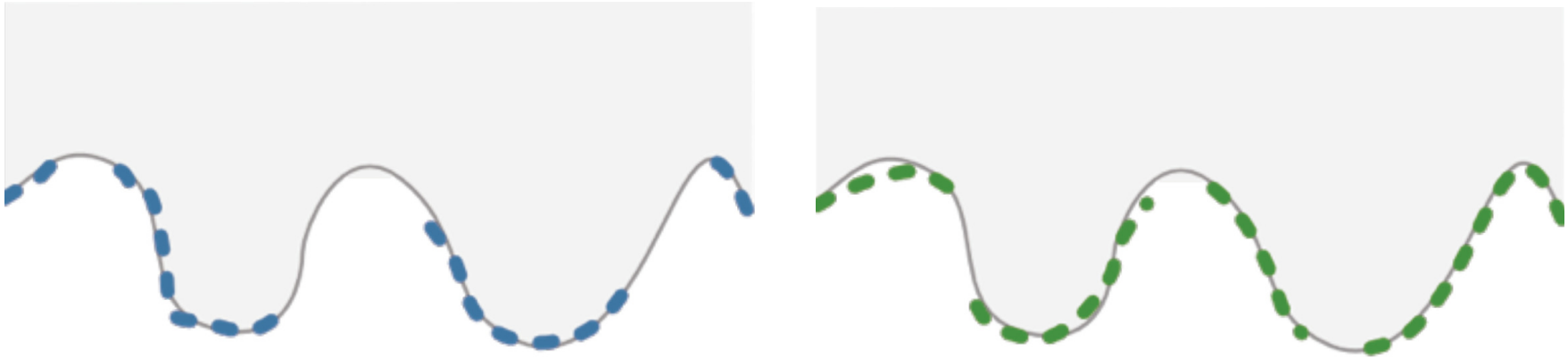
- Why not use the best binocular stereo algorithm?
- Makes use of spatial regularisation to cope with e.g. textureless regions
- Assumes only local image data is available
- Regularises too early

Early regularisation

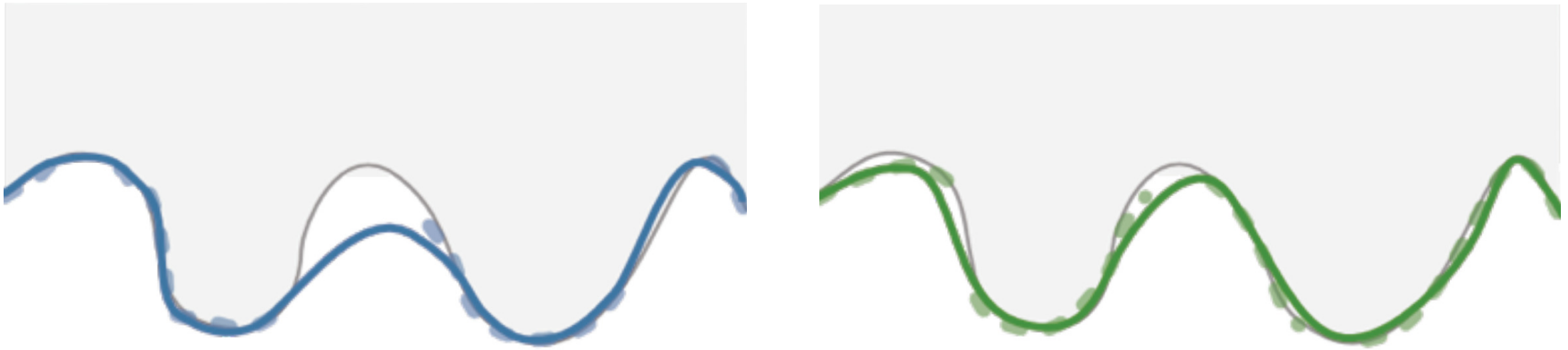


Two depth-maps from neighbouring views

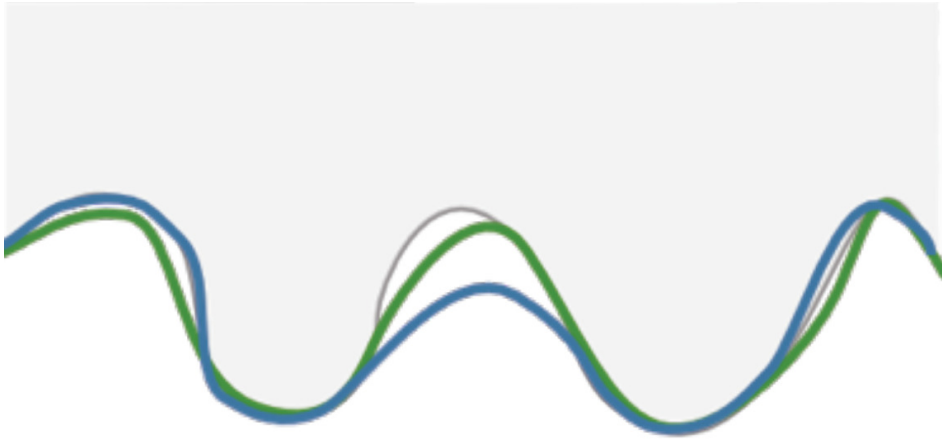
Early regularisation



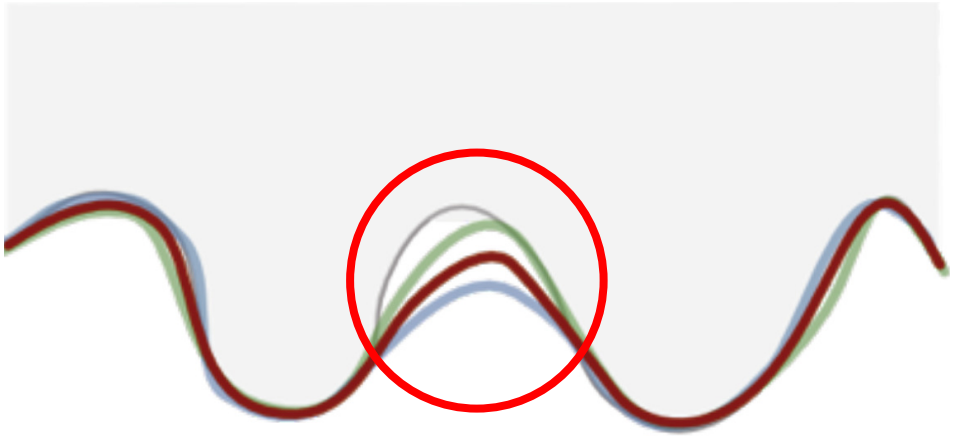
Independent regularisation for each depth-map



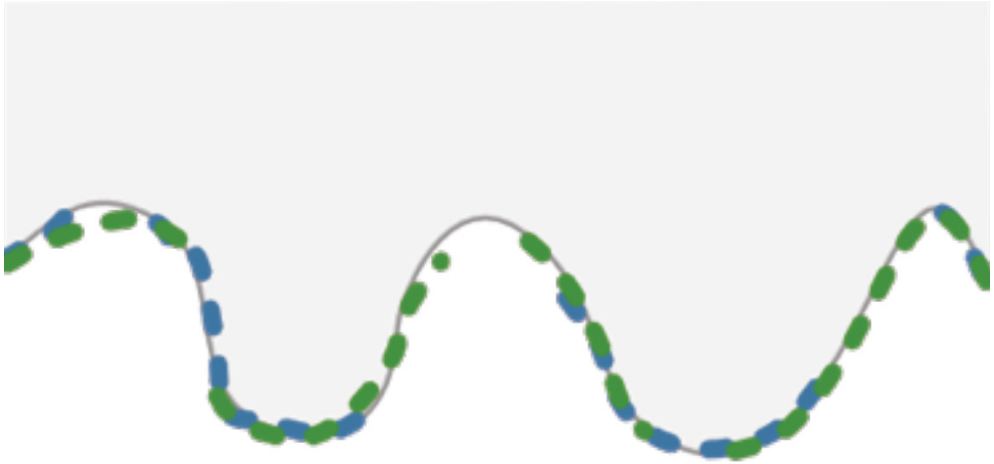
Early regularisation



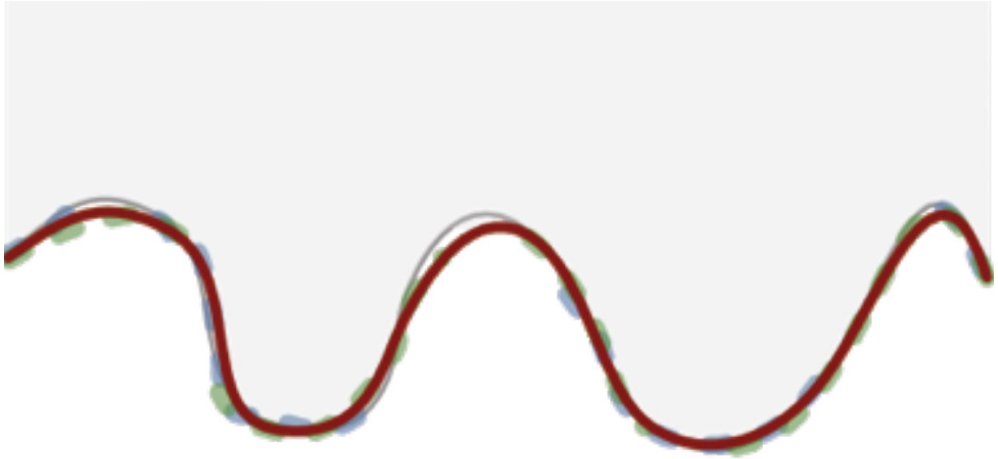
Combine the two estimates



Early regularisation

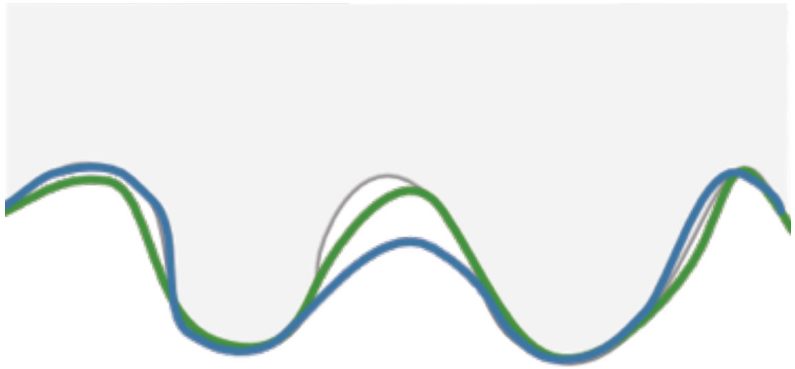


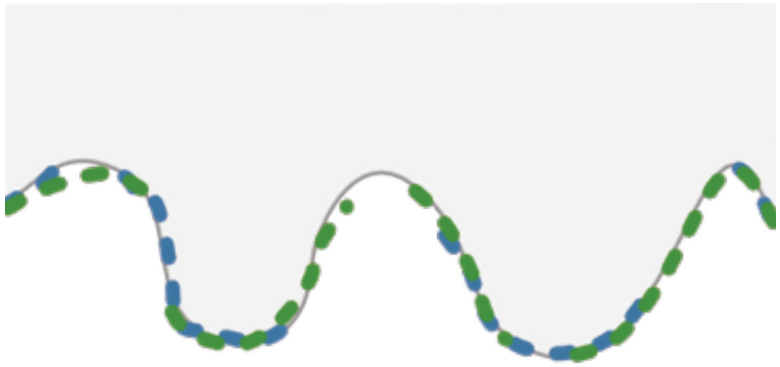
Combine the depth-maps and THEN regularise



Early regularisation

Before combining





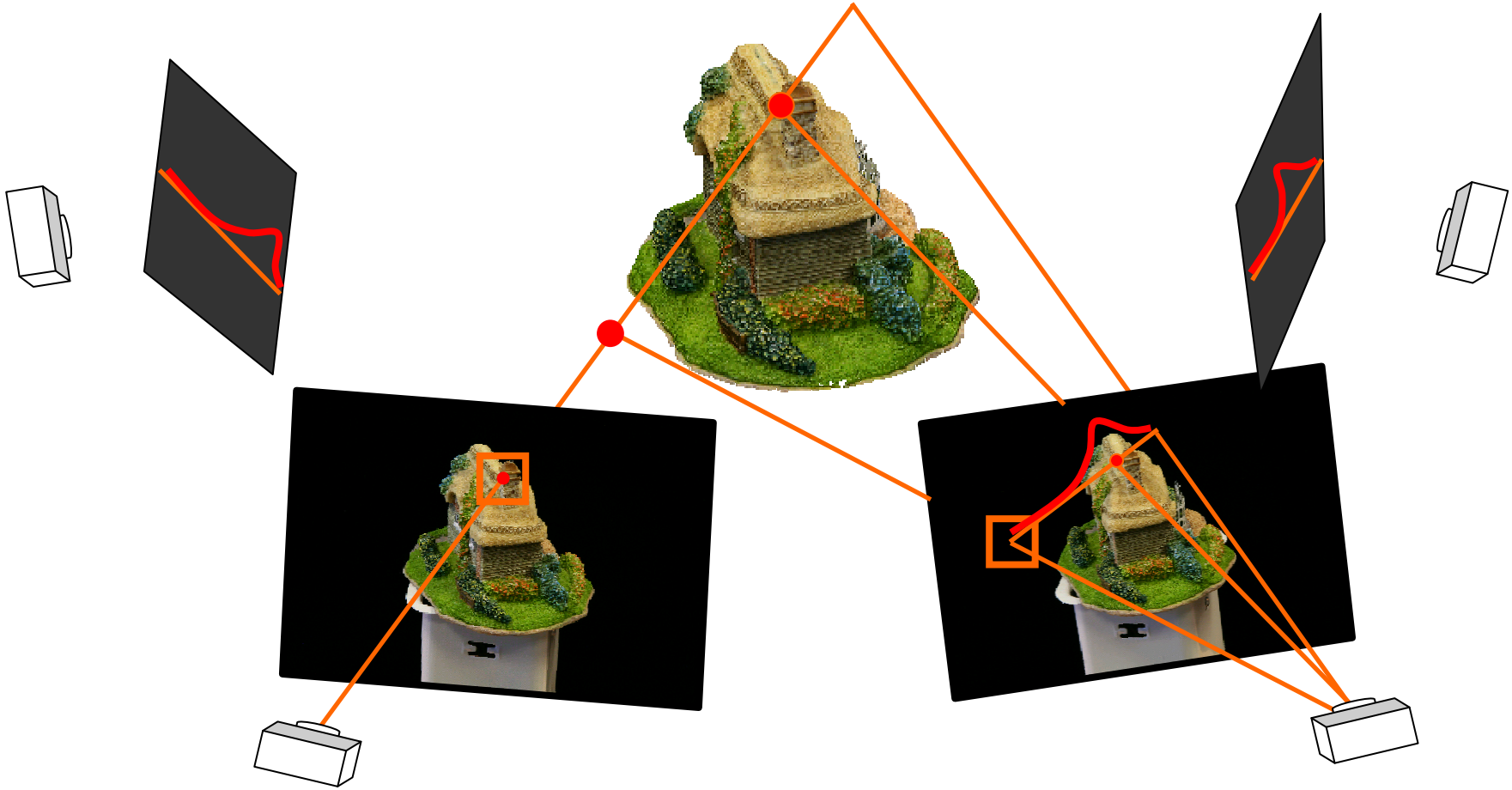
Final result

Depth-map computation

- Compute completely **unregularised** depth-maps independently for each viewpoint

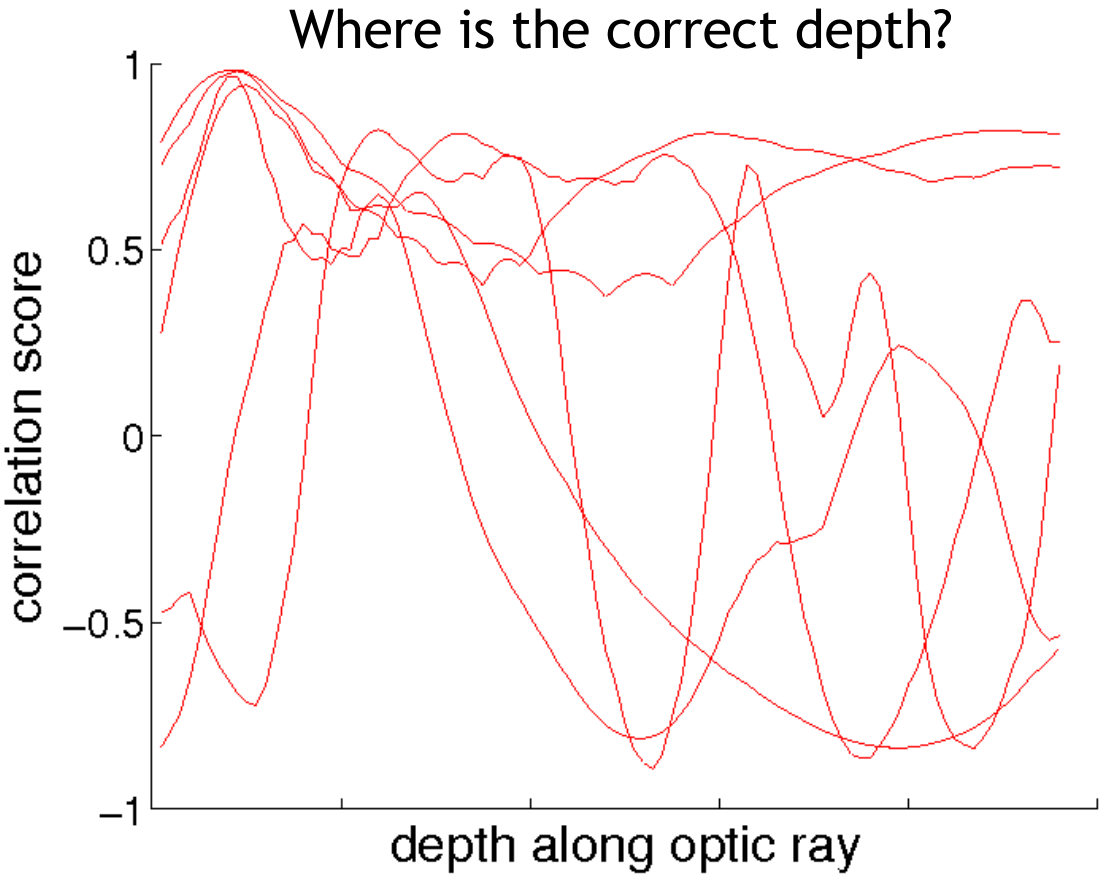
- Only regularise at final stage

Find independent depth of each pixel

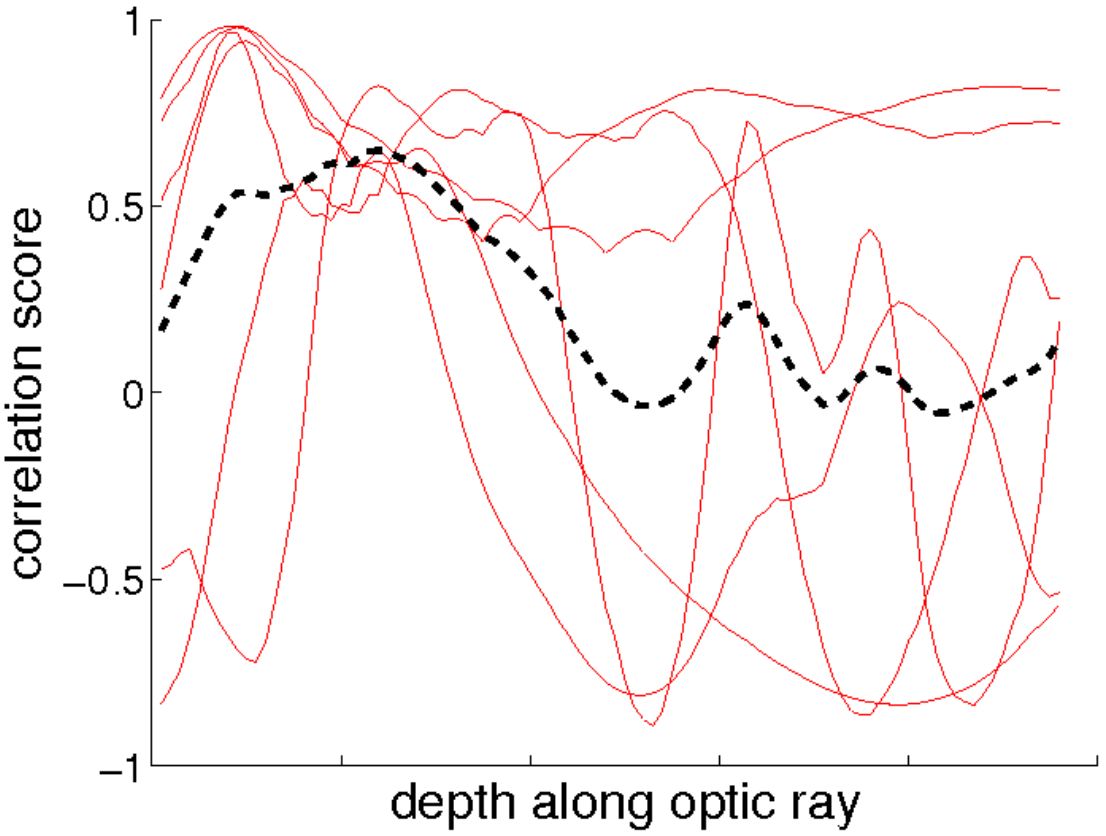


Combining Matching scores

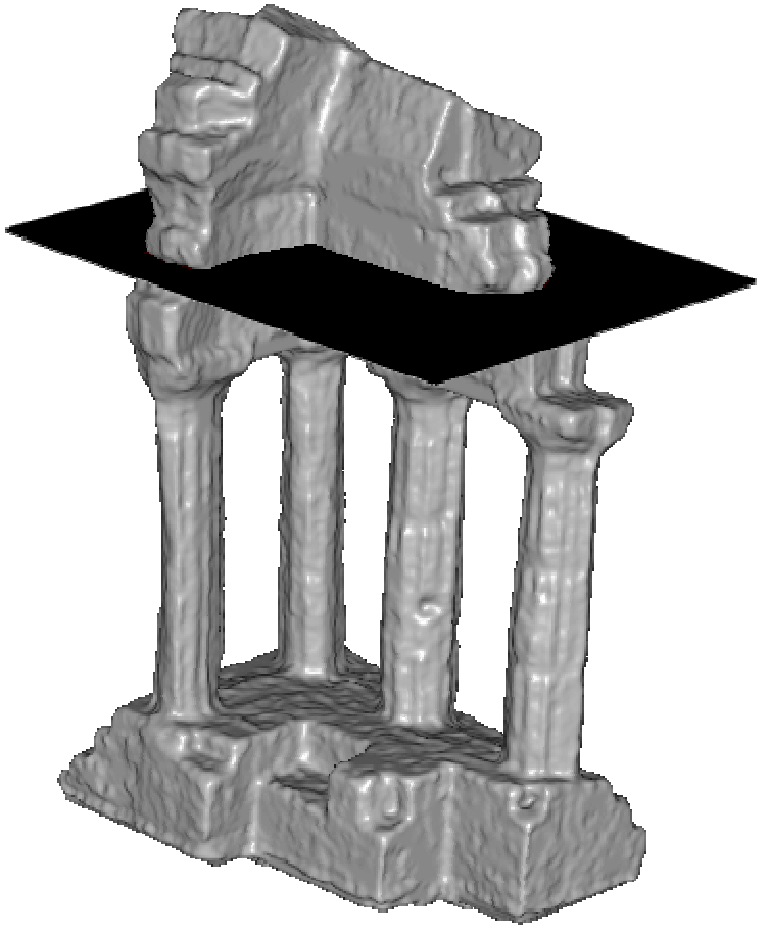
- Compute one correlation curve per image

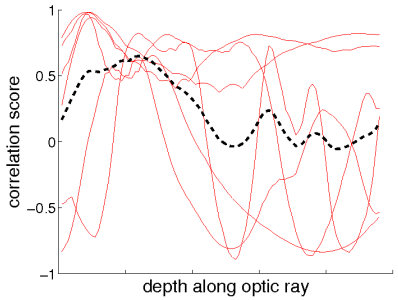


Simple averaging

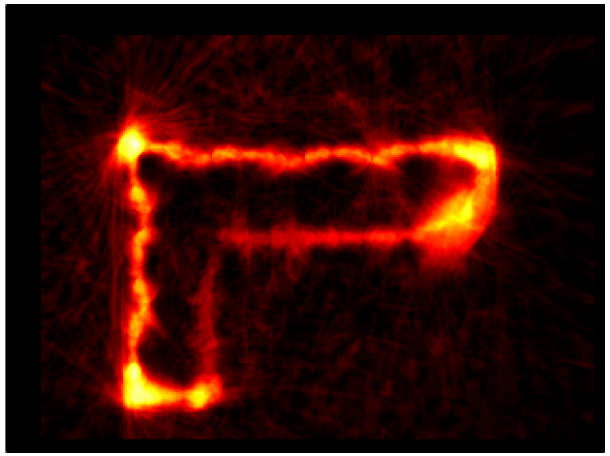


Averaged NCC





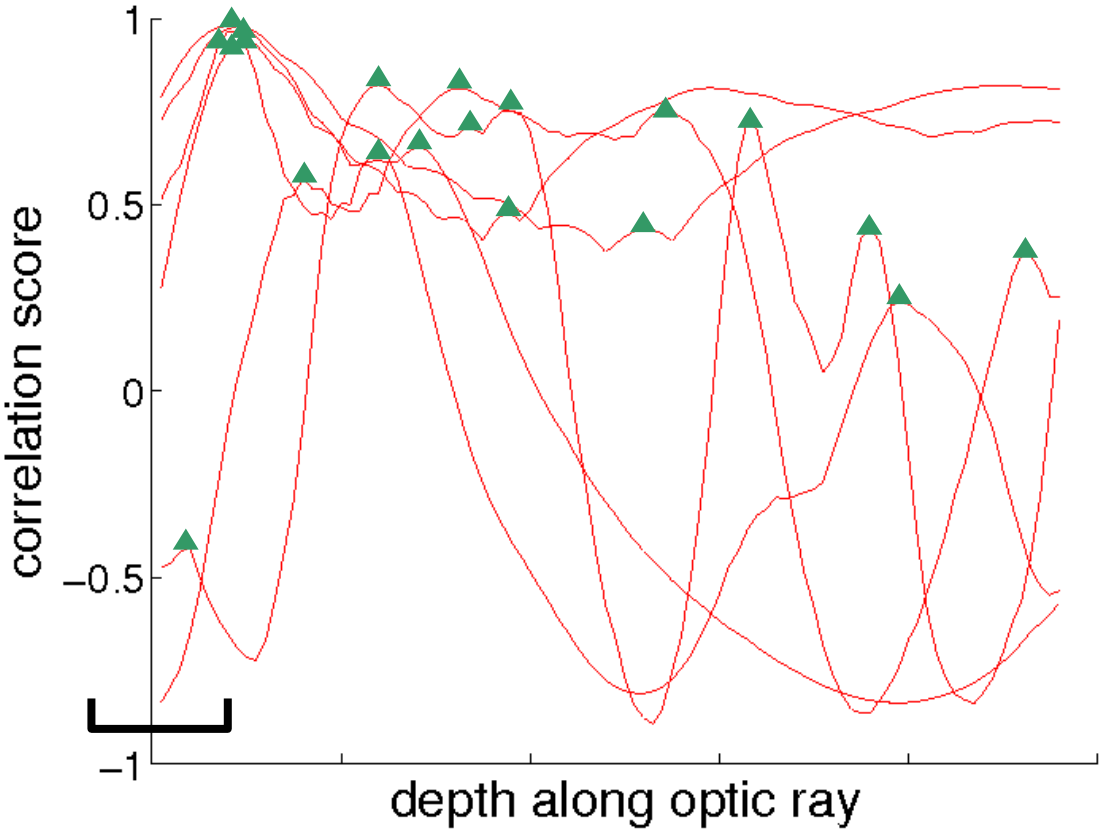
In 3d



Peaks of matching score

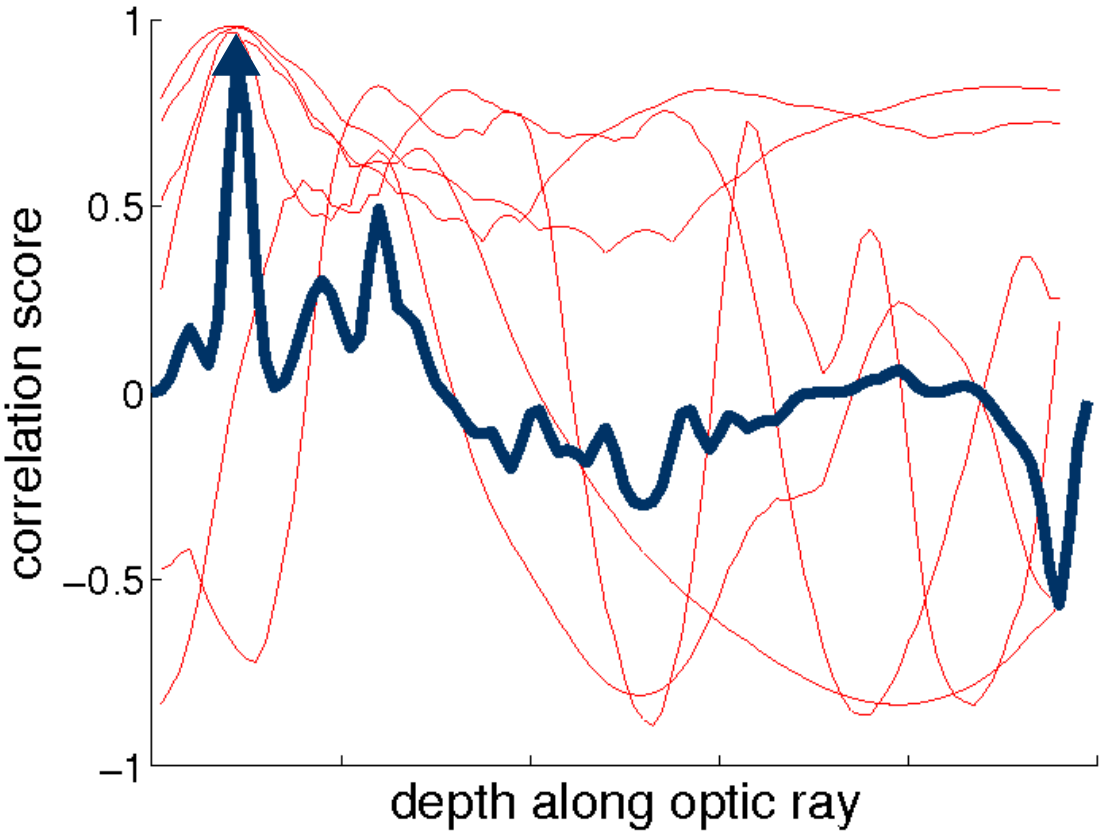
KEY assumption

the correct depth appears at a local maximum of matching score

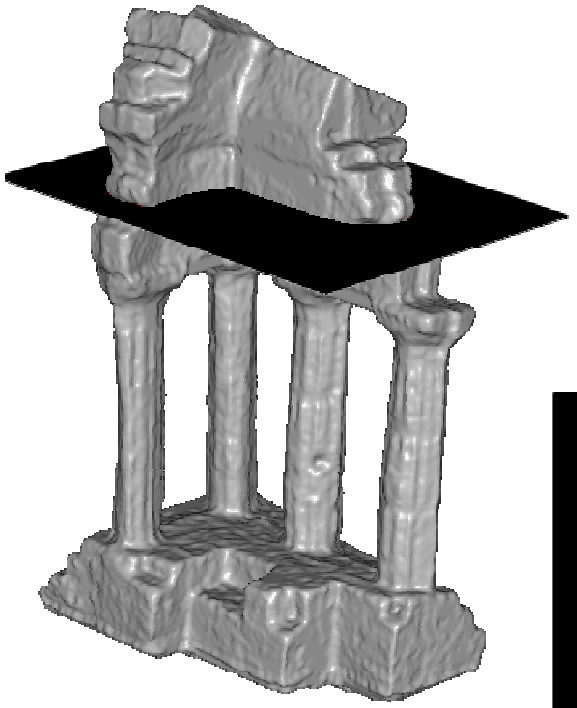


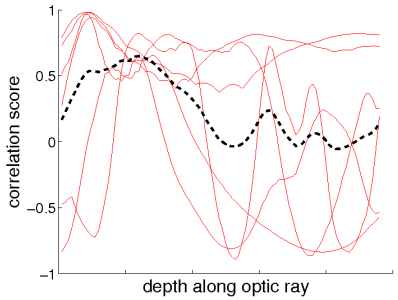
Peaks of matching score

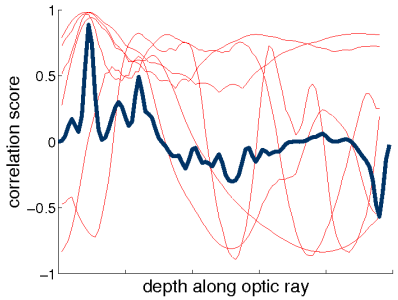
Parzen filter- sliding counter of local maxima



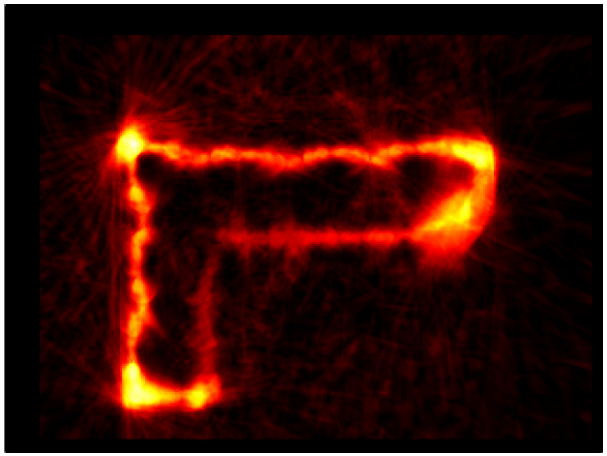
Max of Parzen Filter Output

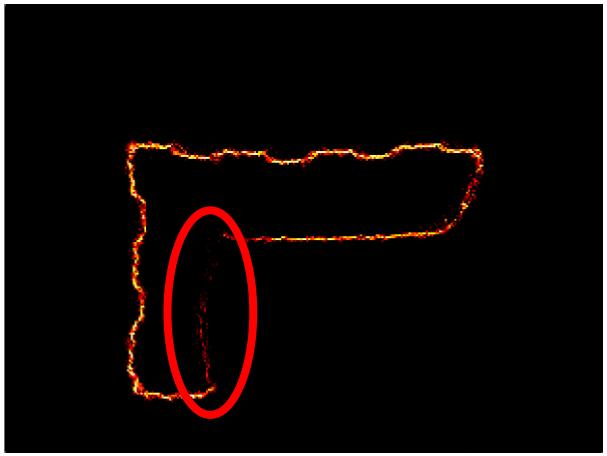






MATLAB

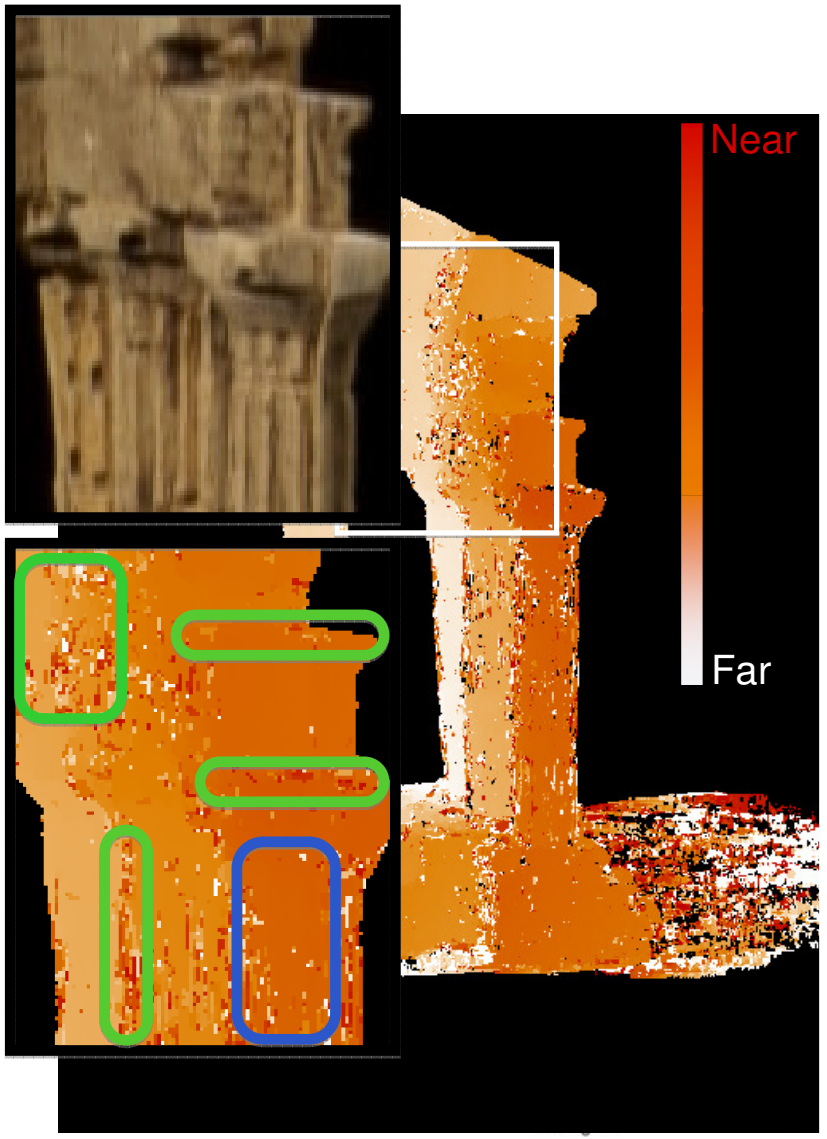




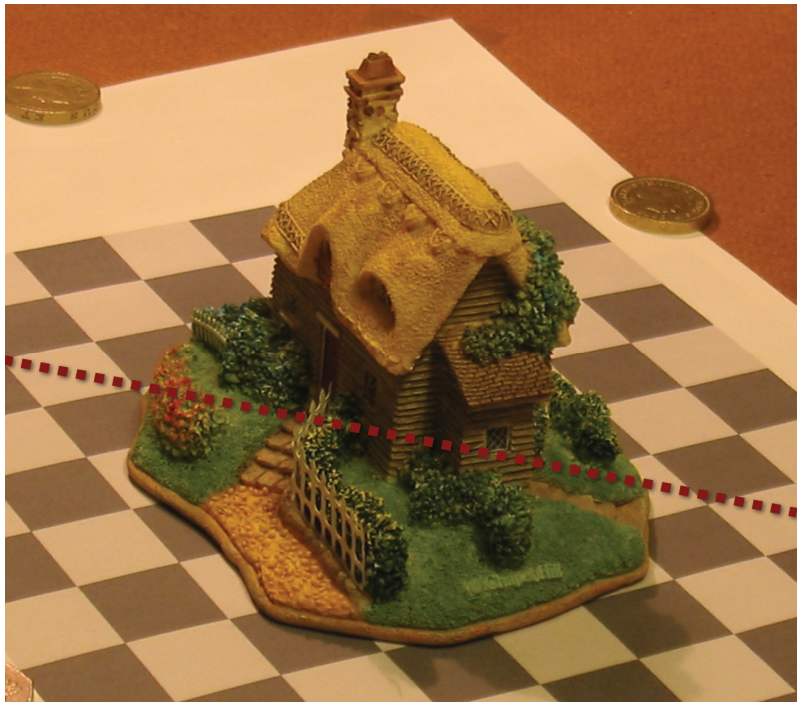
Failure cases

- Repeated Texture
(match in incorrect location)

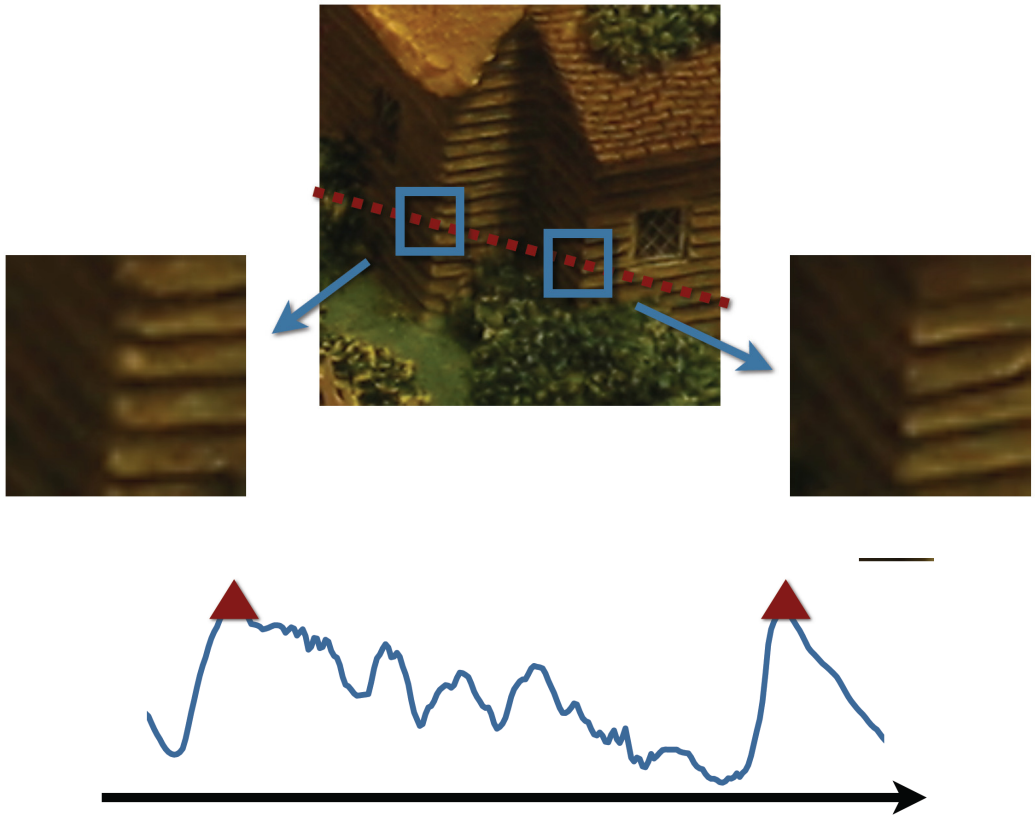
- Matching failure
(surface not visible in one view, lack of texture, image noise etc)



Repeated Texture



Repeated Texture



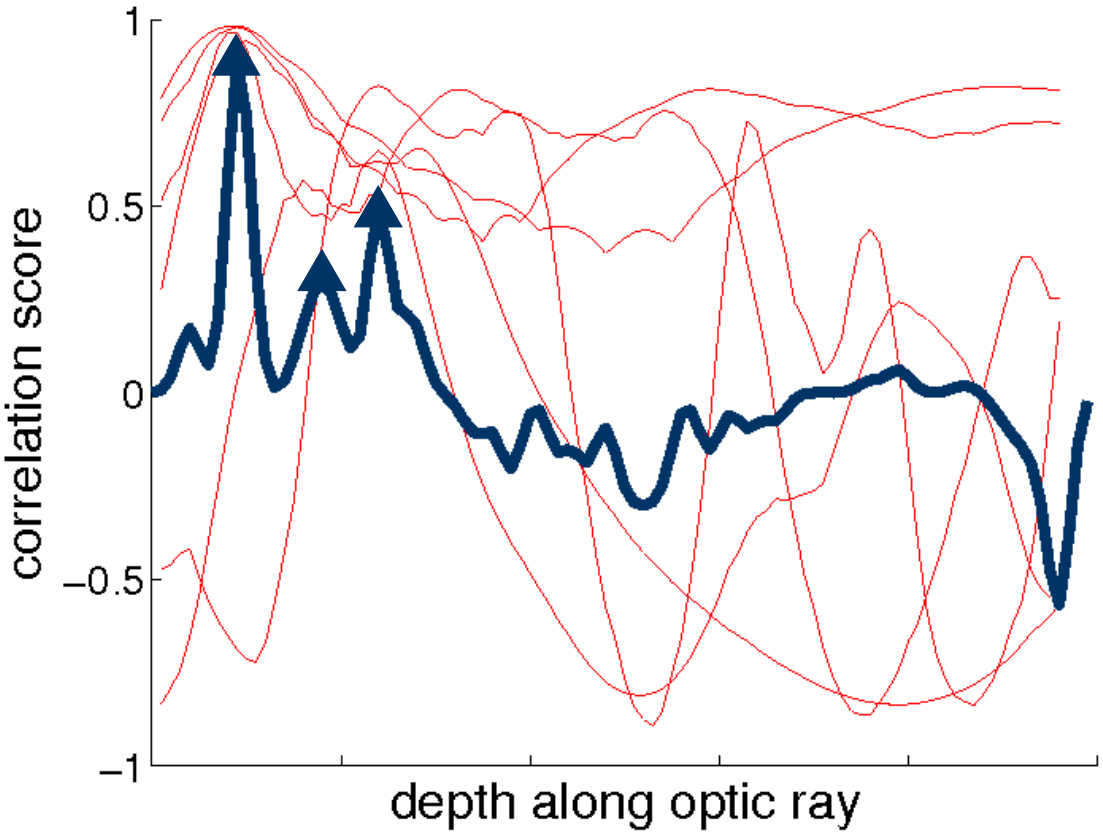
Correct depth can be found in **one** of the peaks

Matching failure

- Due to
  - Occlusion
  - Window distortion
  - Image noise
  - Lack of texture
- NCC score is essentially a **random signal**

Peaks of matching score

- All local maxima of Parzen filter output could be good hypotheses
- How do we pick right one?



Exploit redundancy

- Two types of redundancy:
  - Depth of **neighbouring** pixels in **same** image
    - Depth of neighbours can provide support for the correct depth hypothesis
    - Useful in sparse sequences
  - Depth of **same** pixel in **neighbouring** images
    - If viewpoints are sufficiently close, correct depth hypothesis persists, incorrect is unstable.
    - Useful in dense sequences

Exploit pixel neighborhoods

- Take the K top peaks of the filter output as depth hypotheses for each pixel. Additional hypothesis is "unknown depth"

$$k_i \in \{1 \dots K, \mathcal{U}\}$$

- Build a 2D discrete MRF to select the appropriate peak

$$\sum_i \phi(k_i) + \sum_{(i,j)} \psi(k_i, k_j)$$

1. Unary energy ϕ: bias depth towards highest peaks
2. Pair wise energy ψ: encourage neighbour pixels to have similar depths

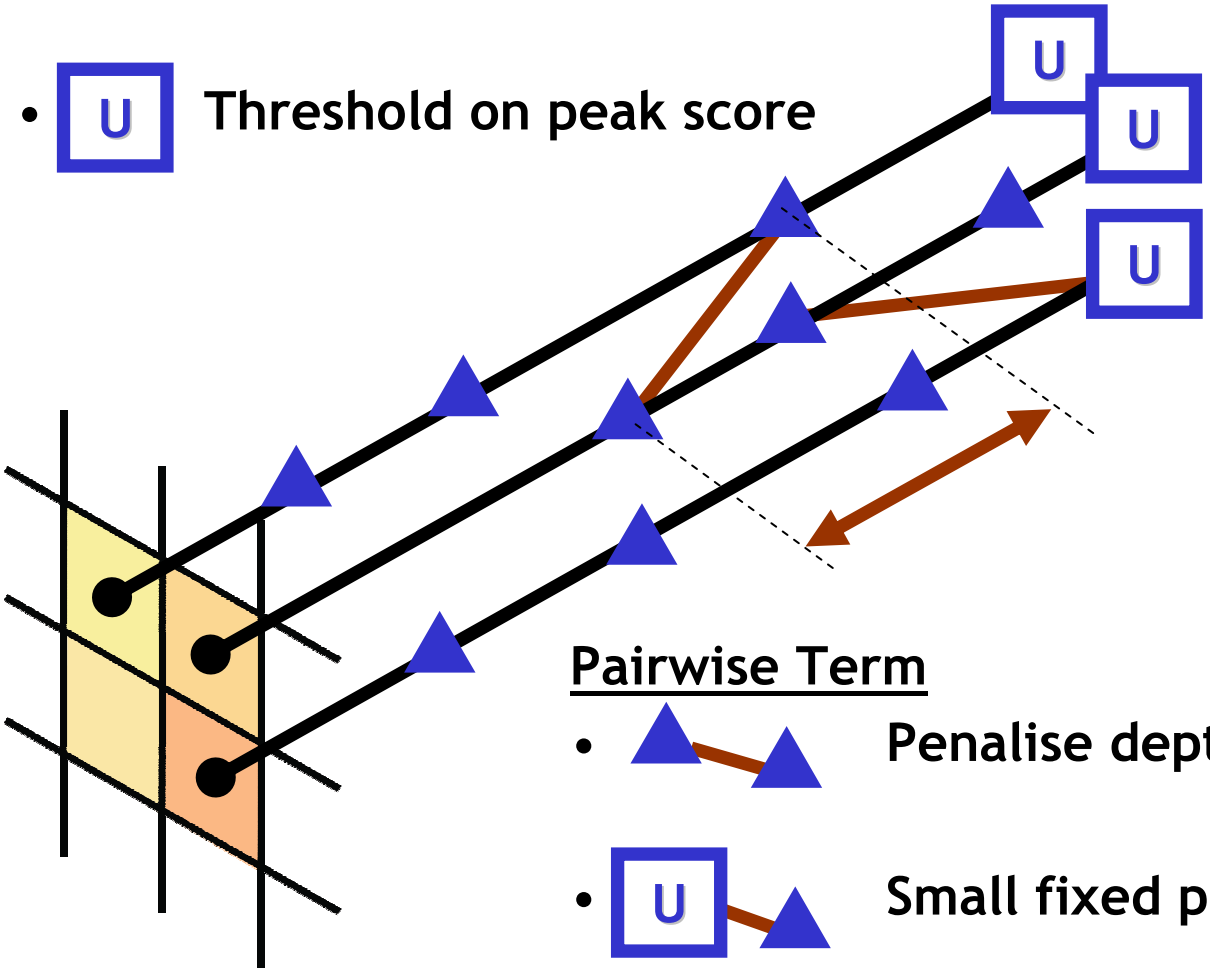
- Use "unknown depth" label for robustness against low scores or spatially inconsistent peaks
  - If one of the two neighbours is "unknown", pair wise term is constant
  - Small penalty cost for "unknown" labels

- MRF can be approximately solved using variety of solvers (TRW)

MRF

Unary Term

- Penalise peaks with low score
- U Threshold on peak score



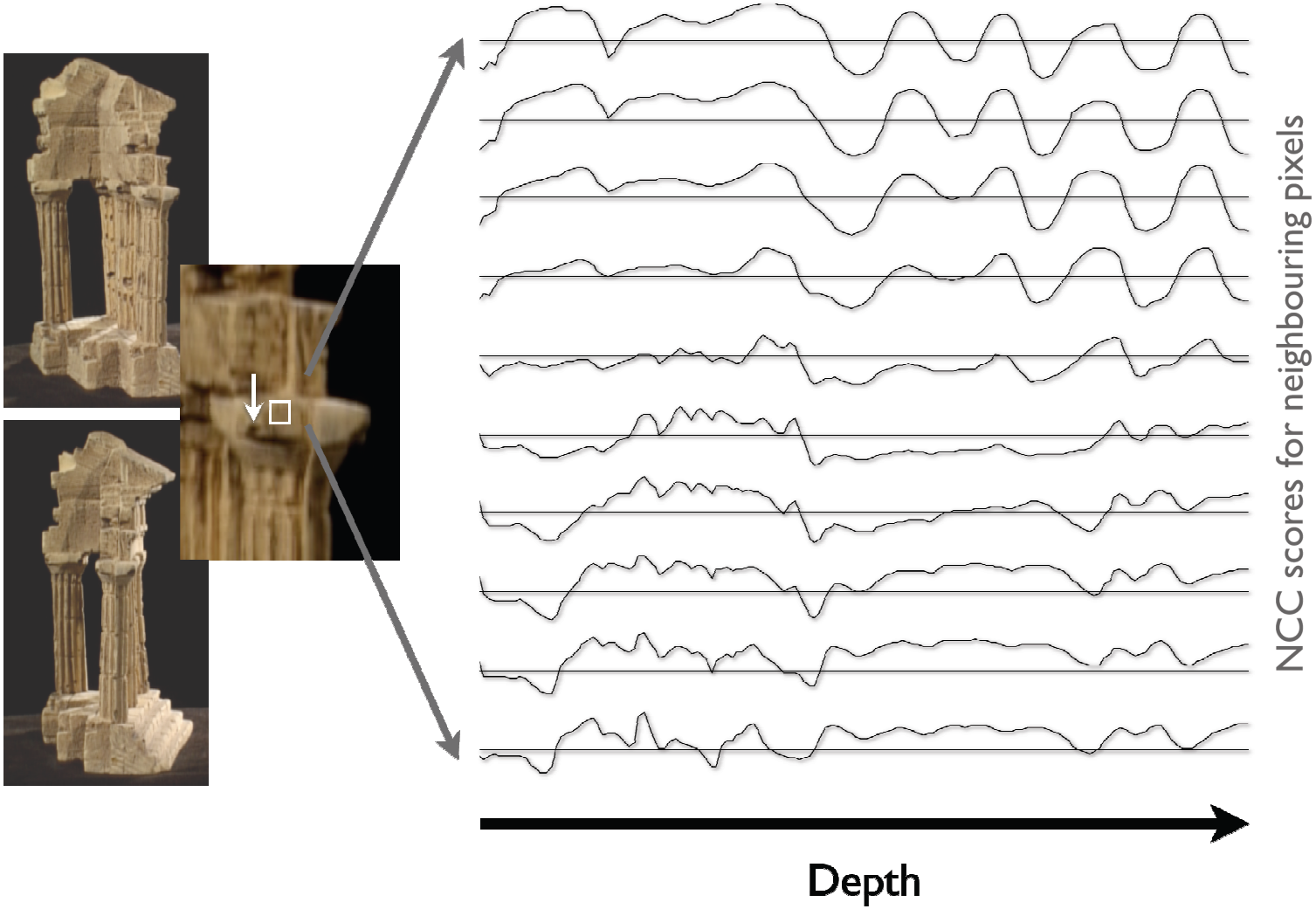
Pairwise Term

- Penalise depth disparity
- U Small fixed penalty

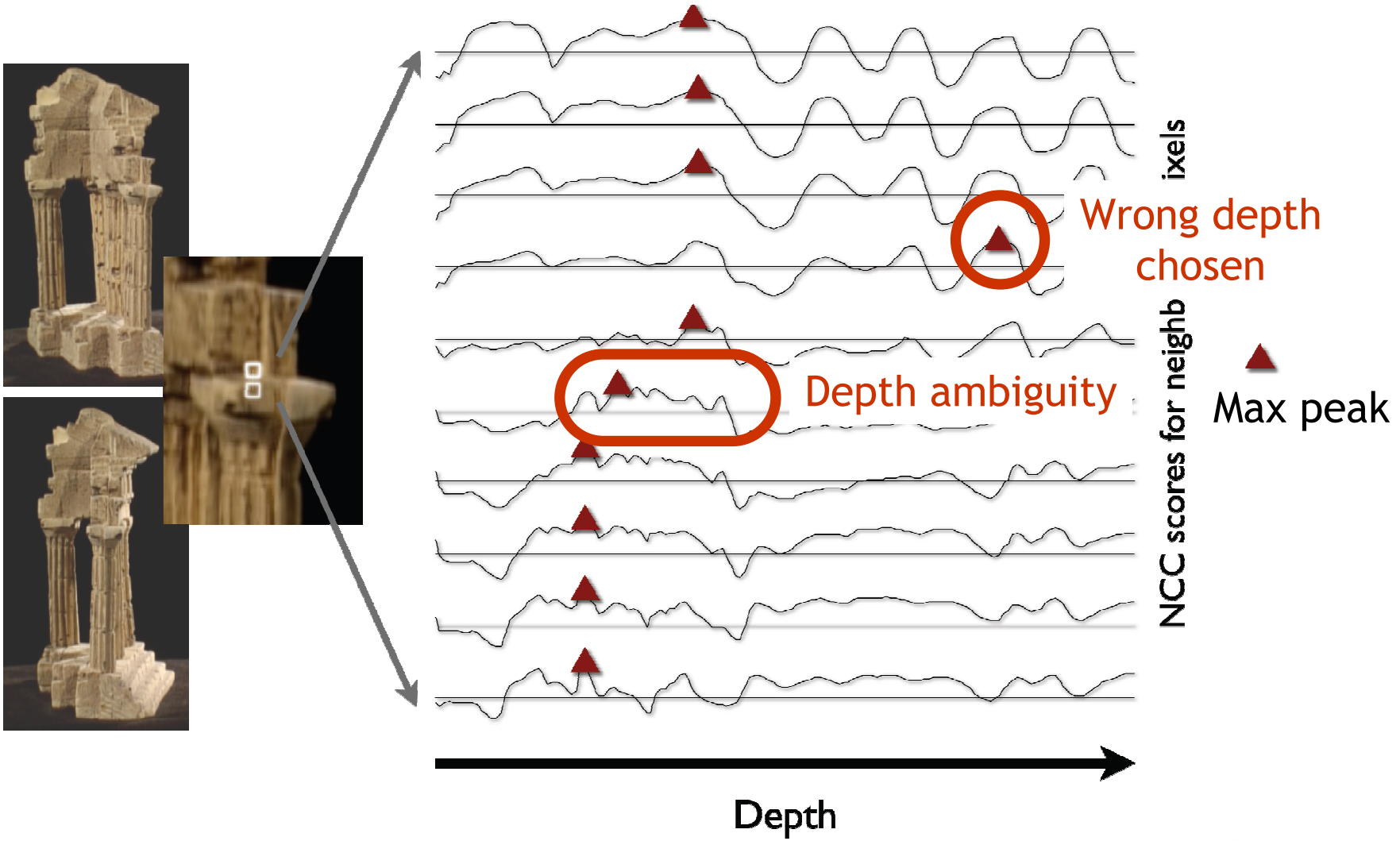
Isn't this early regularisation?

- Yes…
  - …but not so bad.
  - We merely filter out outlier measurements
  - If a pixel doesn't get support for any depth it becomes "unknown"
  - We are NOT filling in the data using a prior

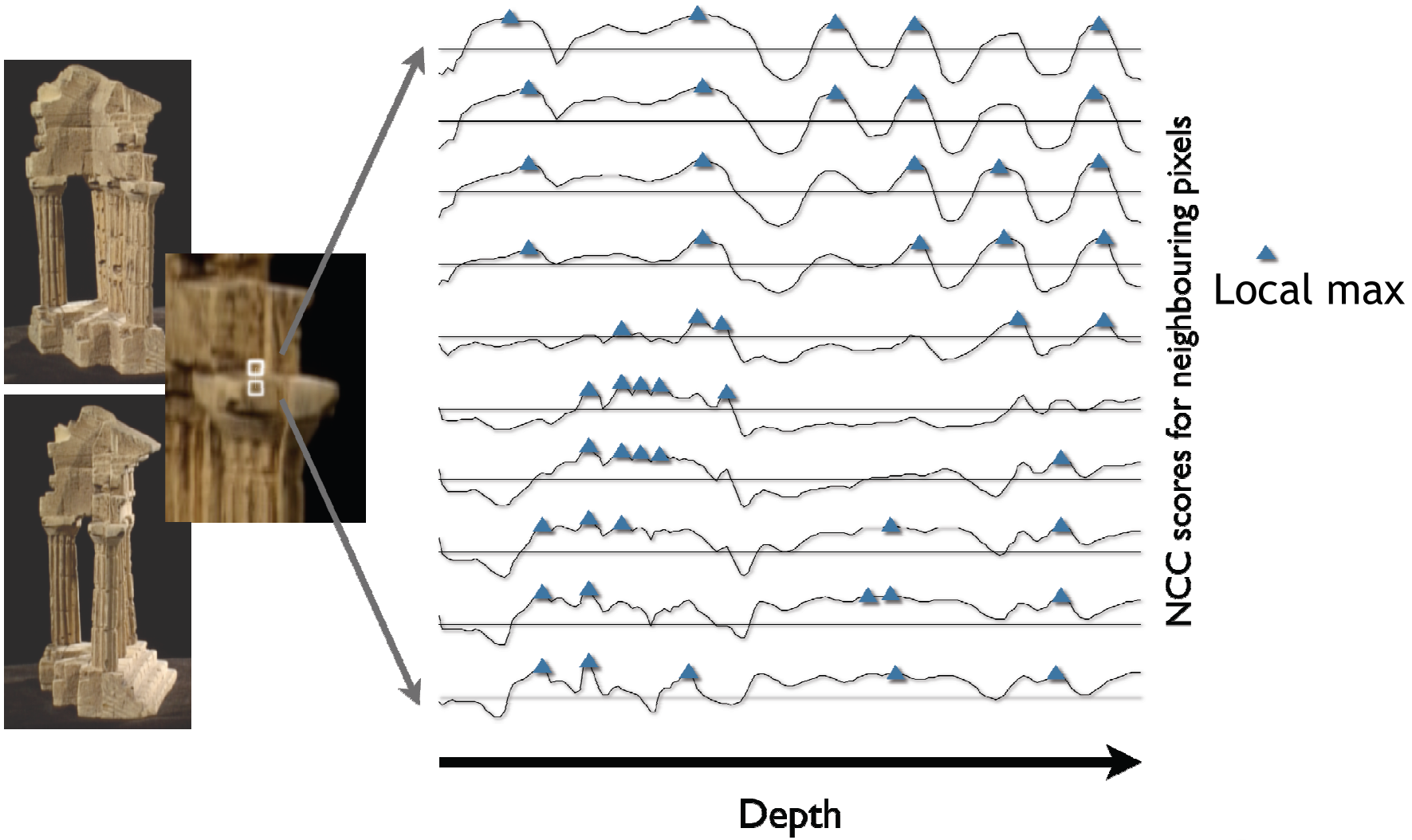
Depth estimation



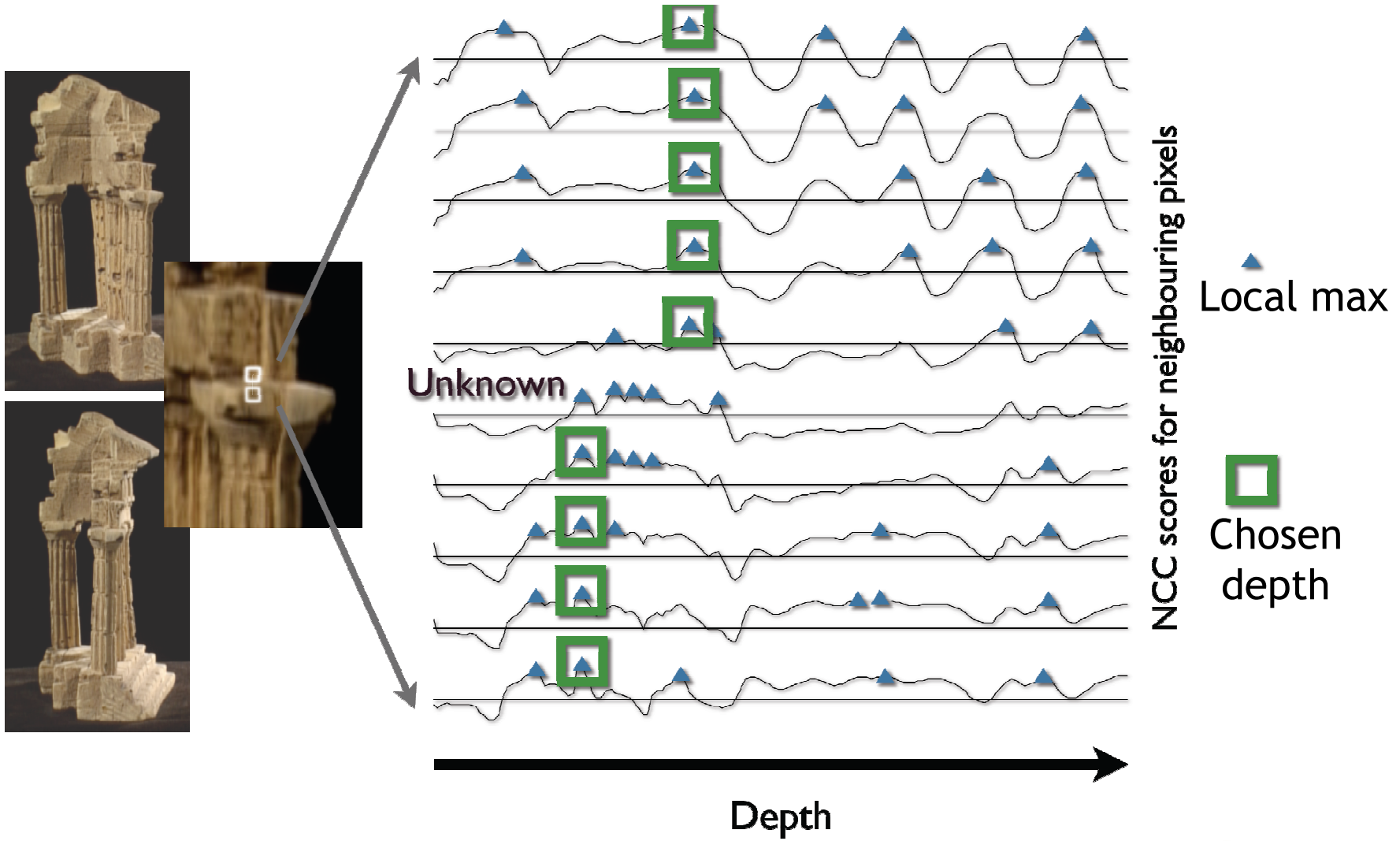
Max peak

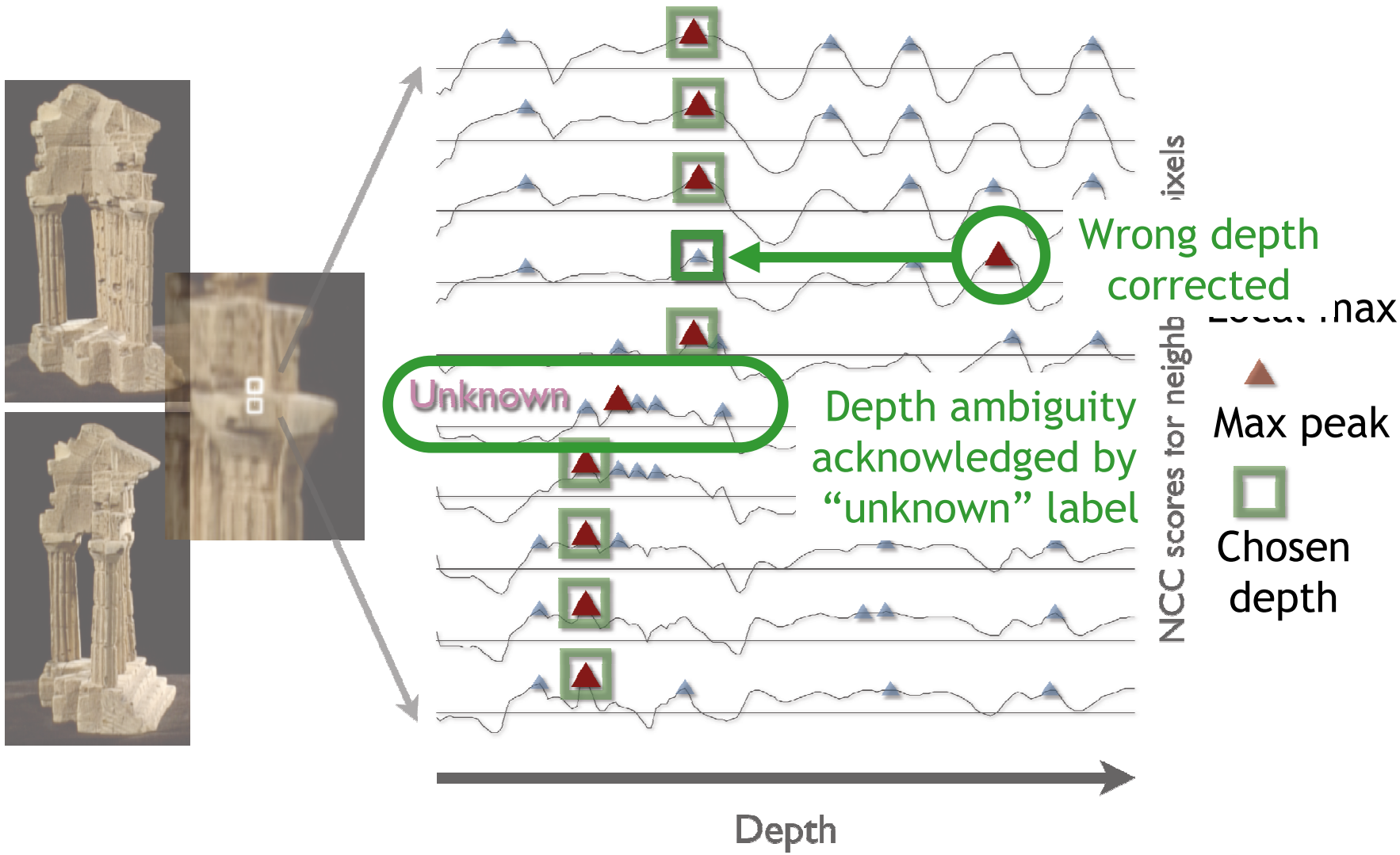

Wrong depth chosen
Depth ambiguity
Max peak
NCC scores for neighb ixels
Depth

K top peaks

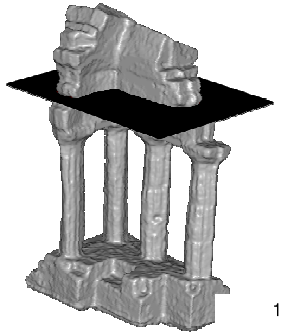


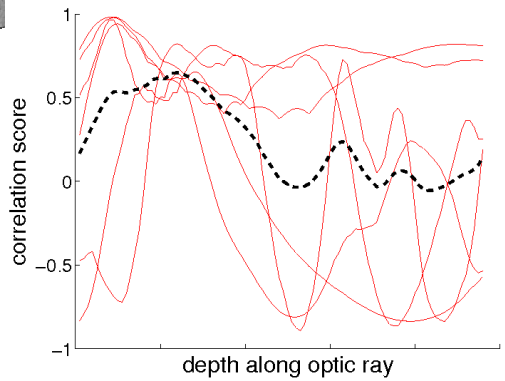
After MRF filtering

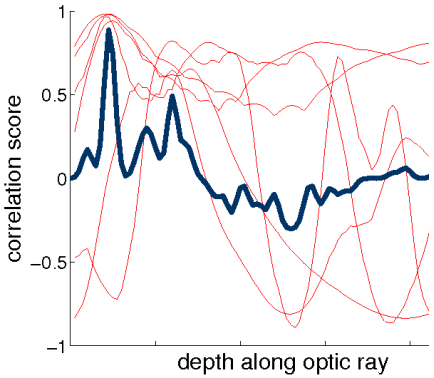


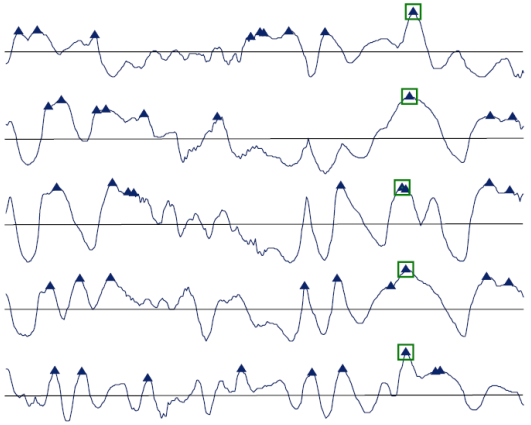

Wrong depth
corrected
Unknown
Depth ambiguity
acknowledged by
“unknown” label
Max peak
Chosen
depth
Depth

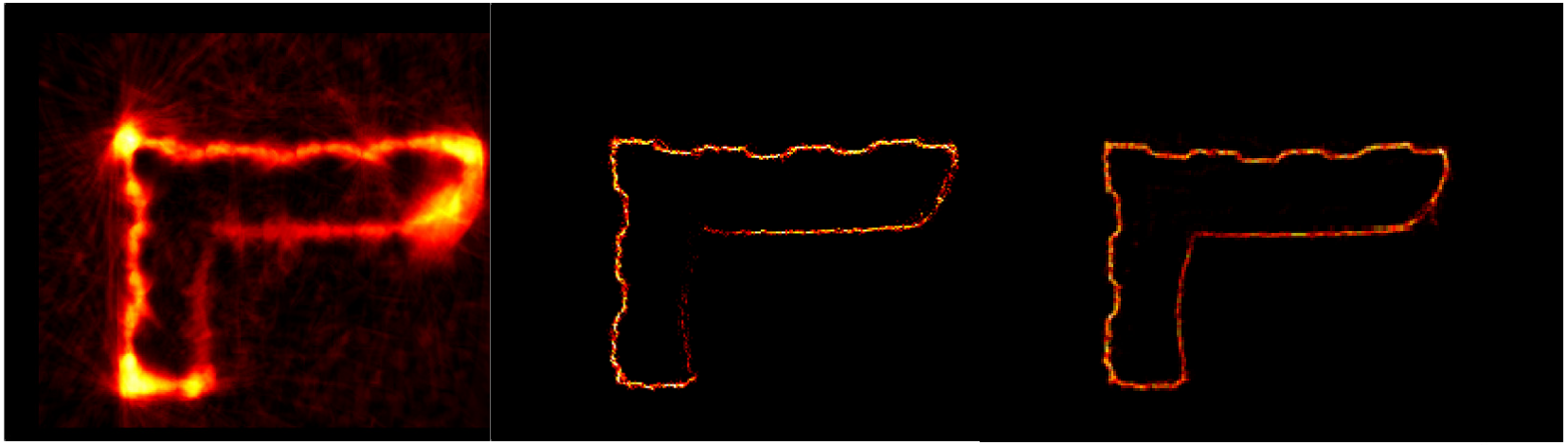
Multi-hypothesis MRF filtering







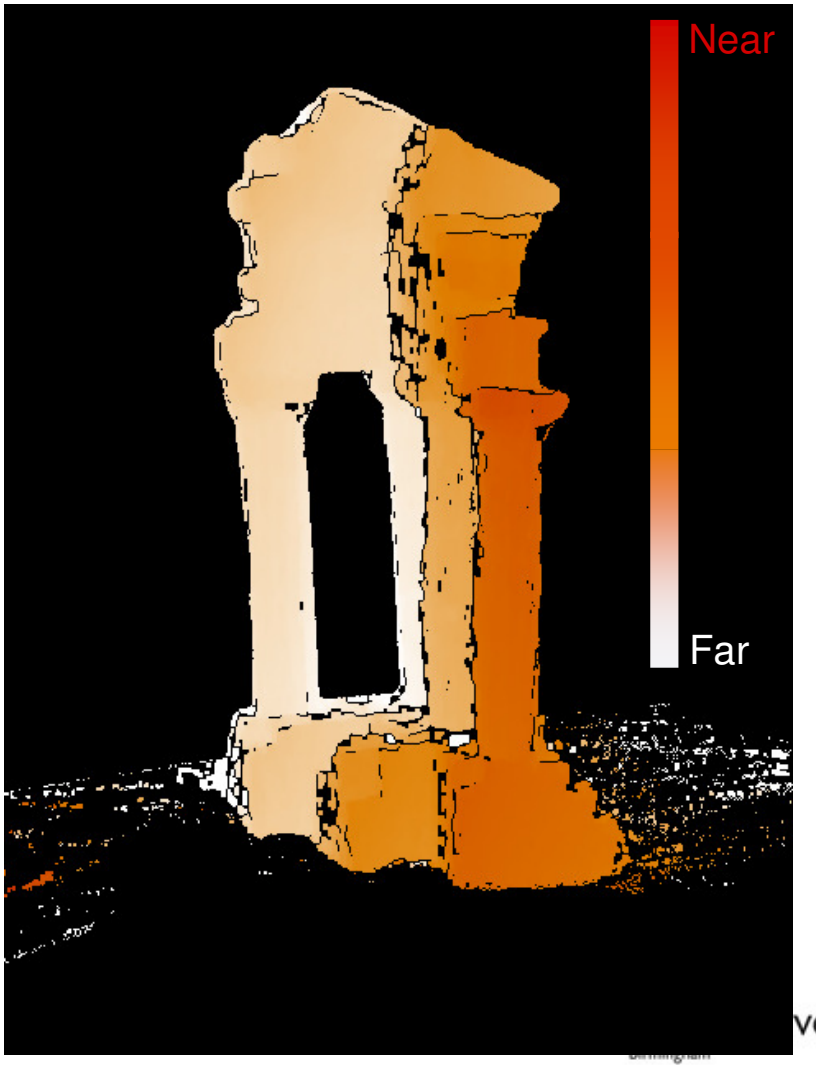




Depth-Map Results

Taking the maximum peak

Results of the MRF optimisation

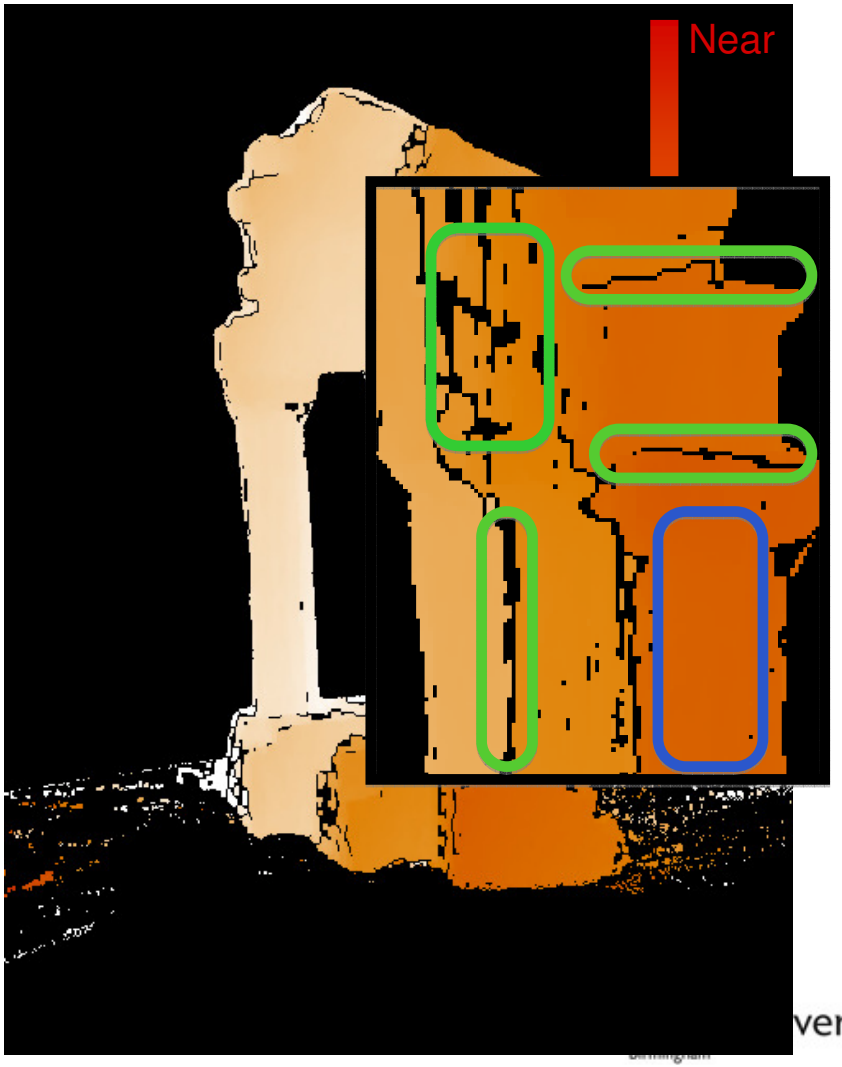


versity

Depth-Map Results

Taking the maximum peak

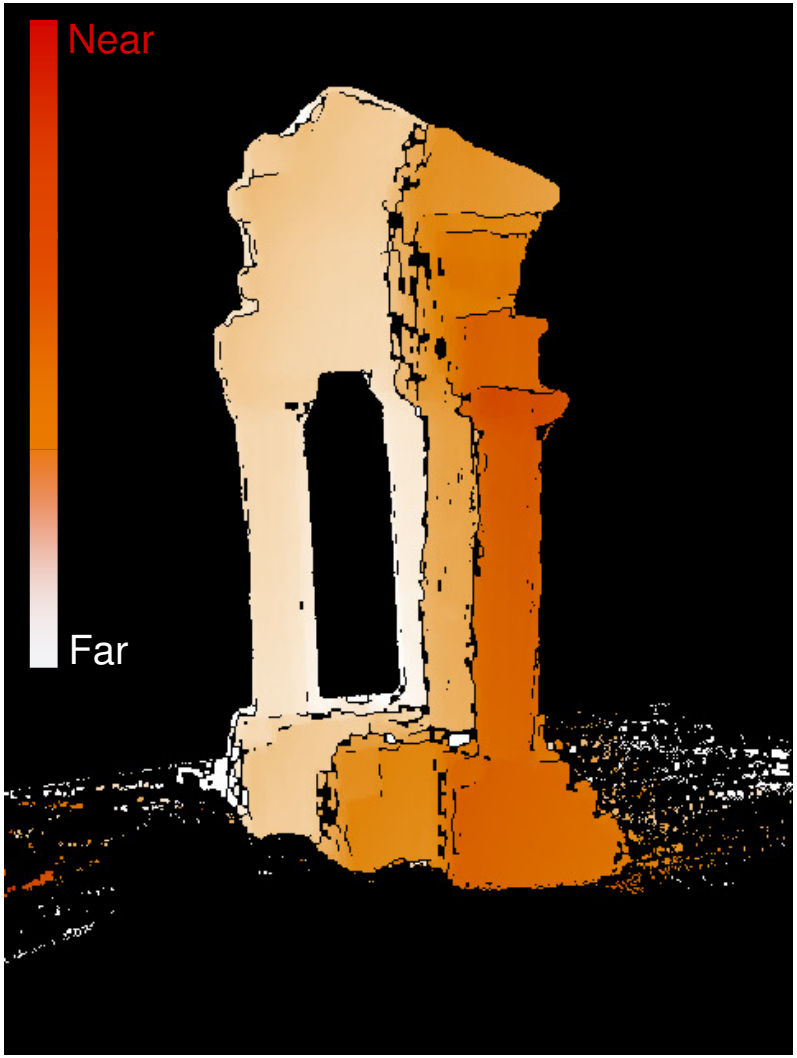
Results of the MRF optimisation



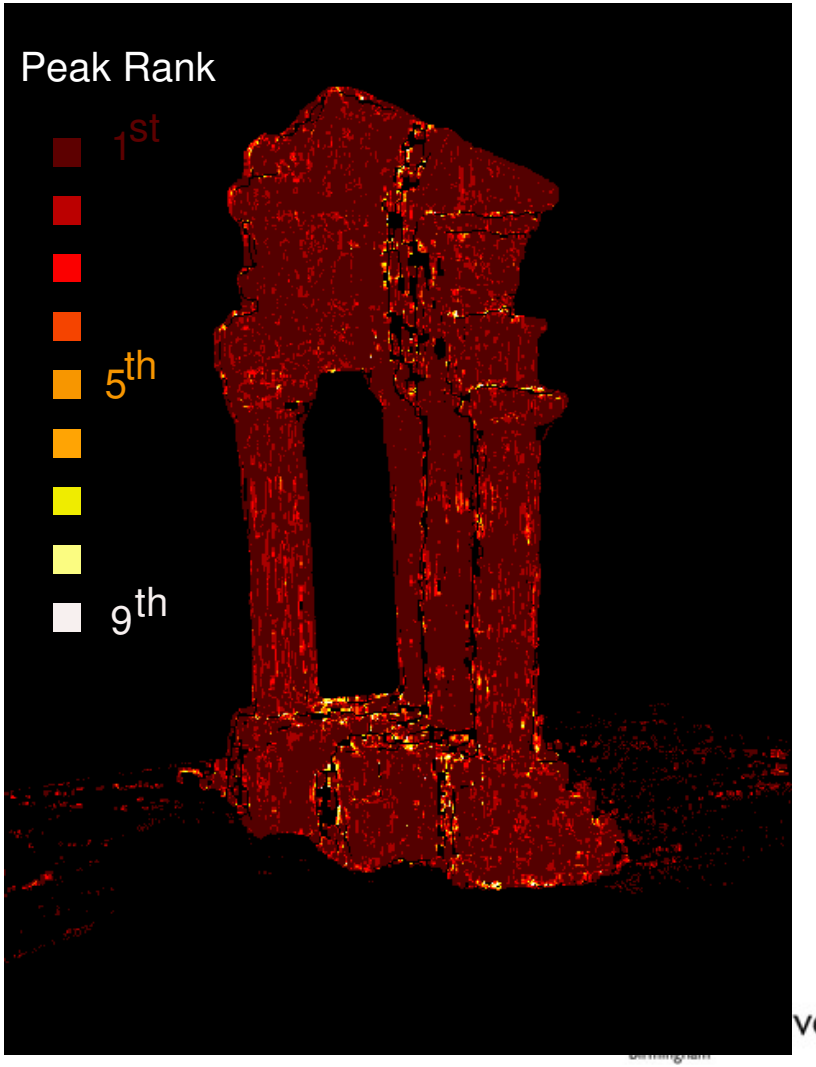
versity

Depth-Map Results

Results of the MRF optimisation



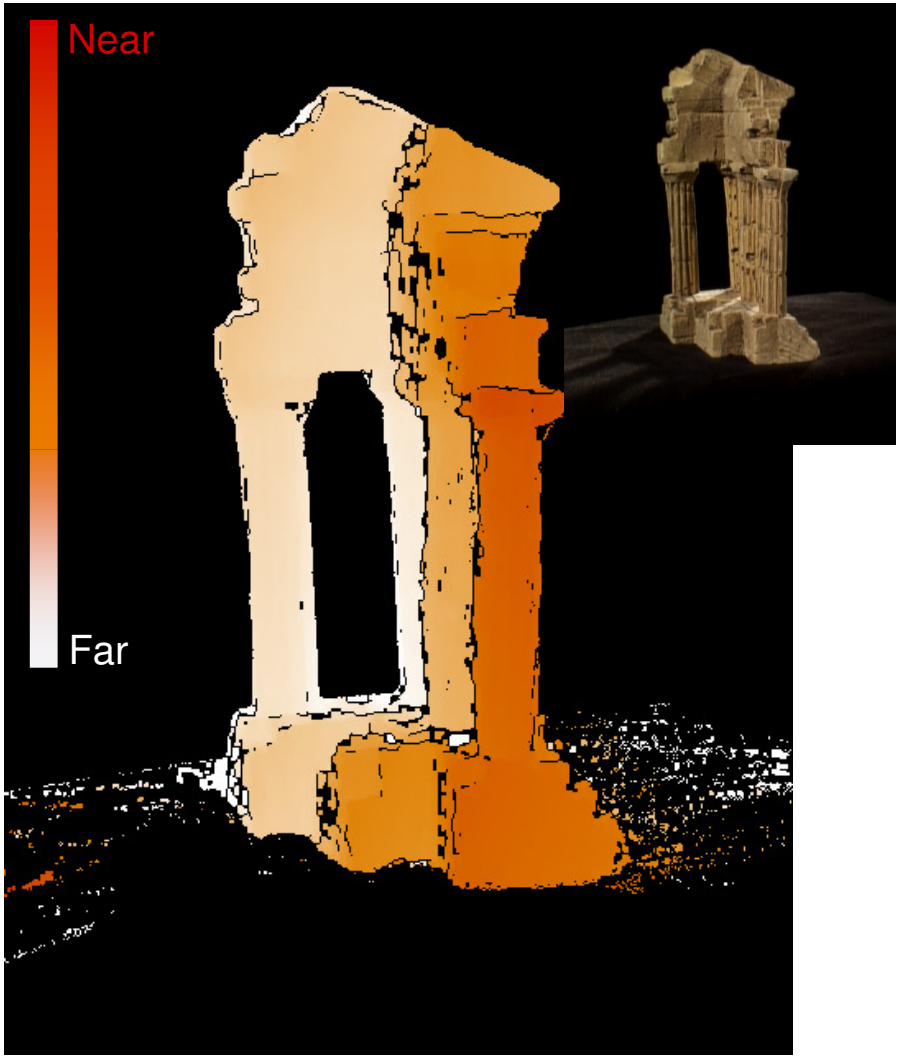
The ranking of the chosen peak

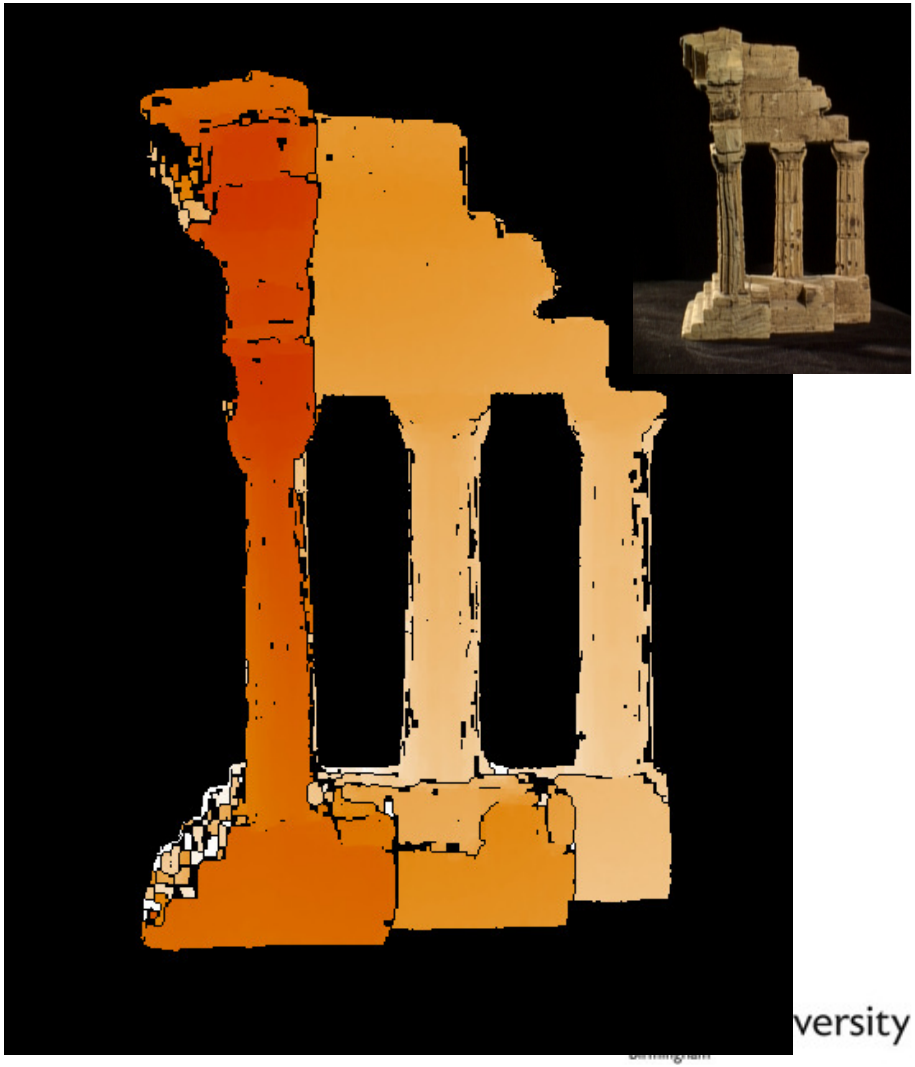


versity

Depth-Map Results

Results of the MRF optimisation for two neighbouring depth-maps

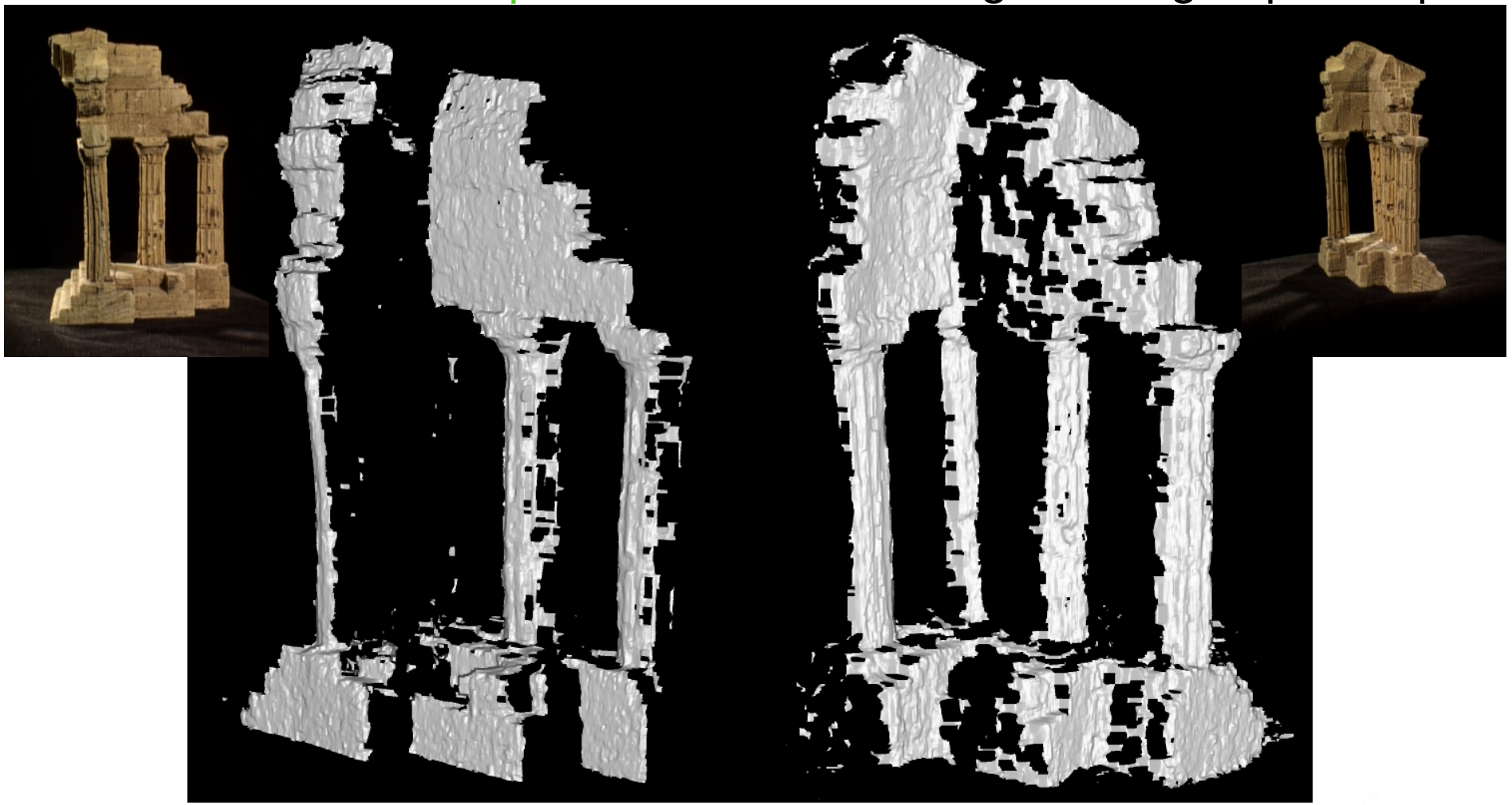




versity

Depth-Map Results

Results of the MRF optimisation for two neighbouring depth-maps



Exploit redundancy

- Two types of redundancy:
  - Depth of neighbouring pixels in same image
    - Depth of neighbours can provide support for the correct depth hypothesis
    - Useful in sparse sequences

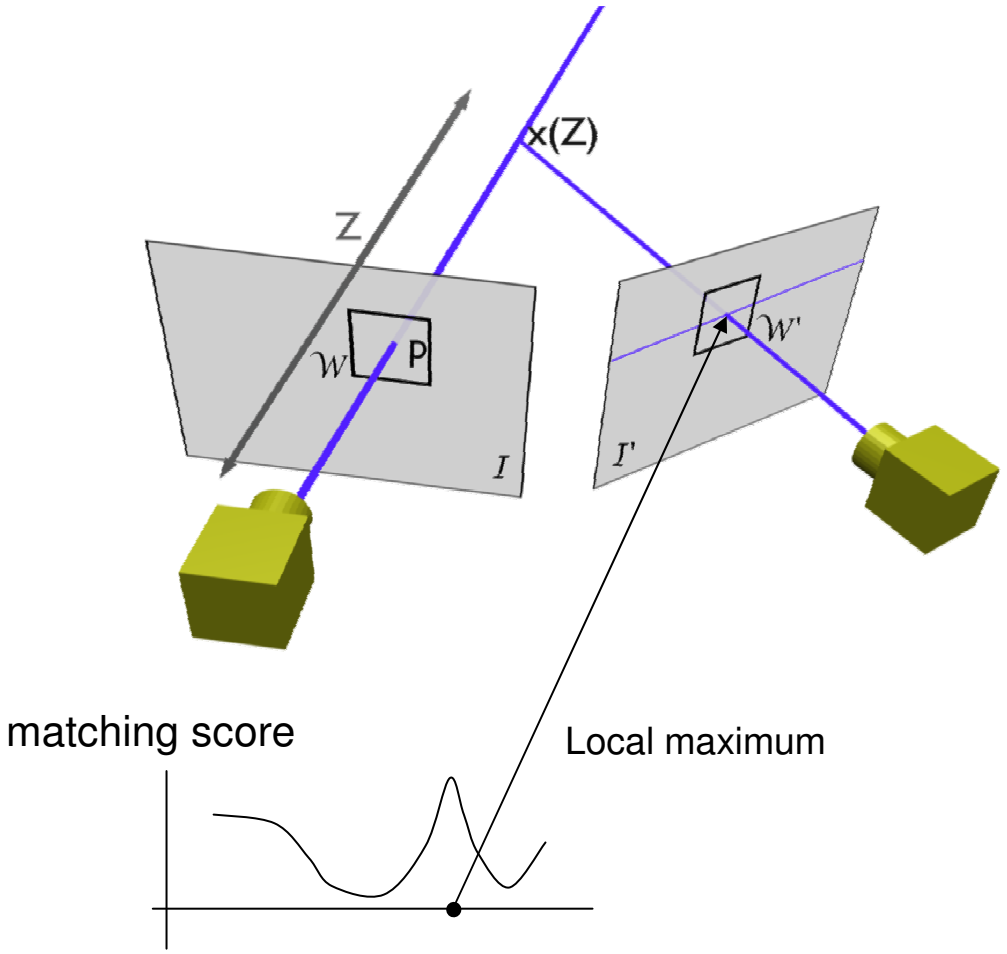
  - Depth of same pixel in neighbouring images
    - If viewpoints are sufficiently close, correct depth hypothesis persists, incorrect is unstable.
    - Useful in dense sequences

Video-based Multi-View Stereo

- The correct depth appears at a local maximum of matching score
- With enough images, all ambiguities get resolved

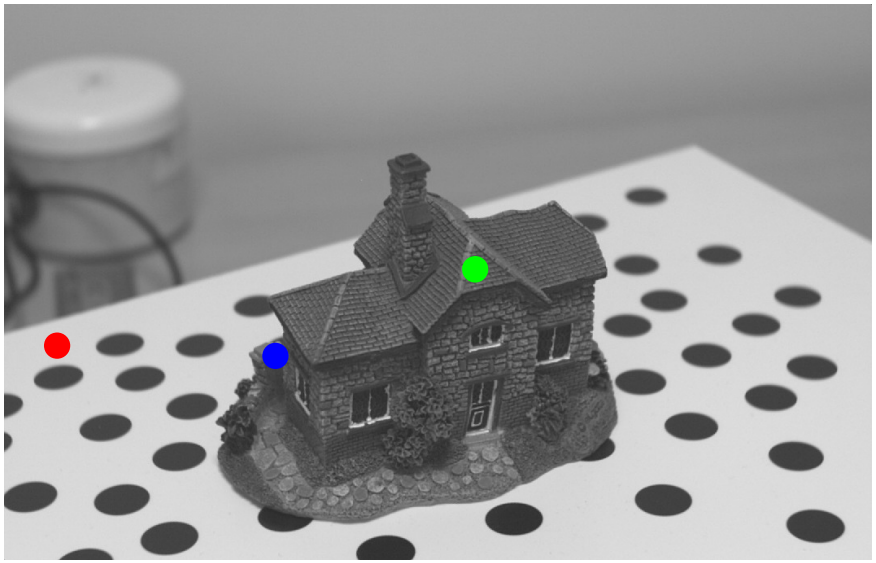
~600 frames

Pixel = depth sensor

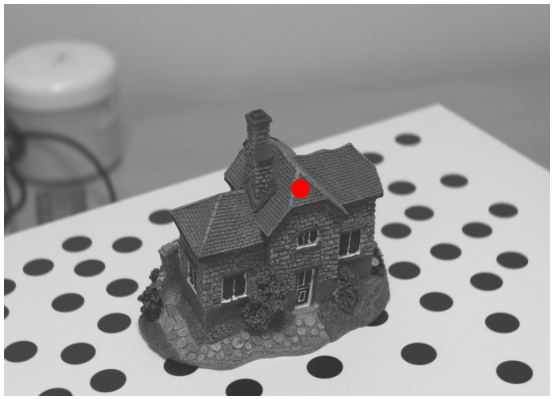


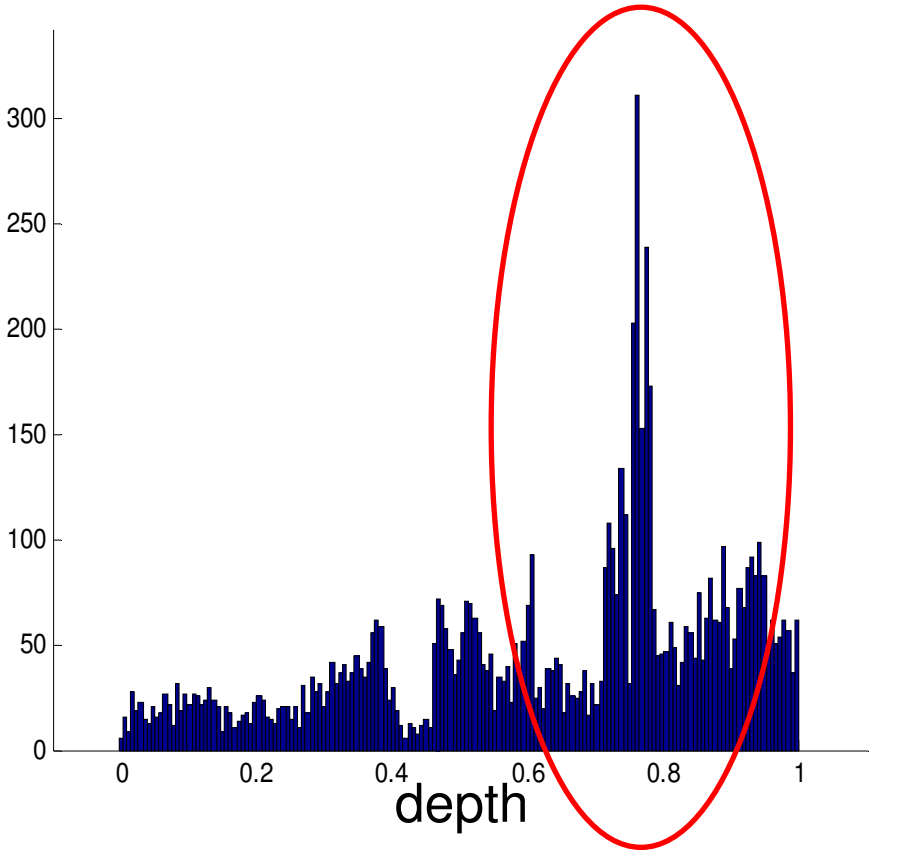
What are the measurements like?

- Well textured pixel
- Untextured point
- Occluded point



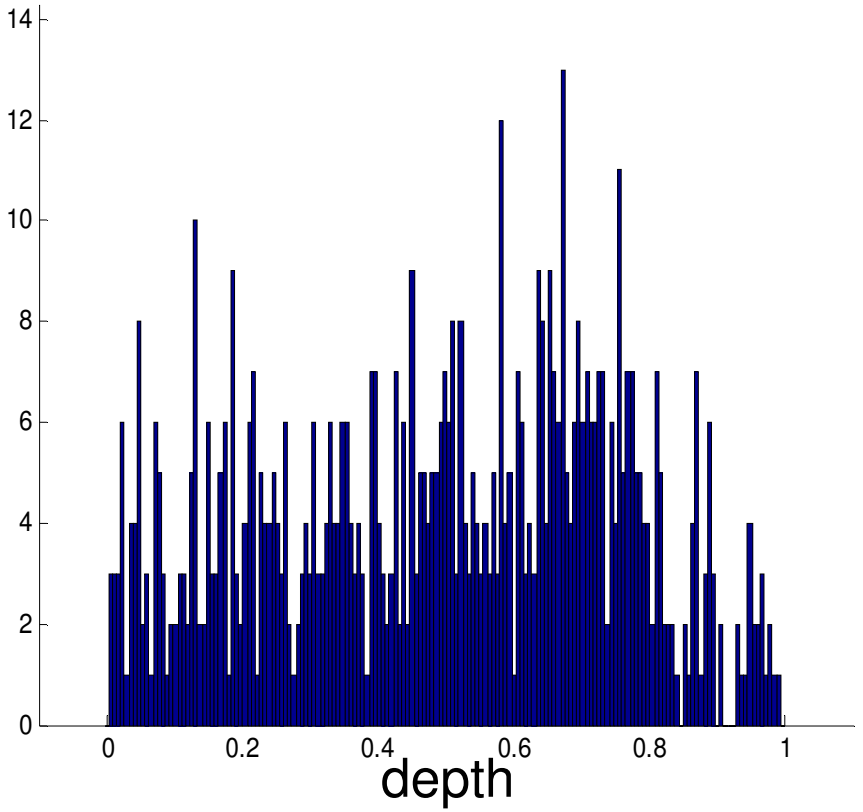
Textured point



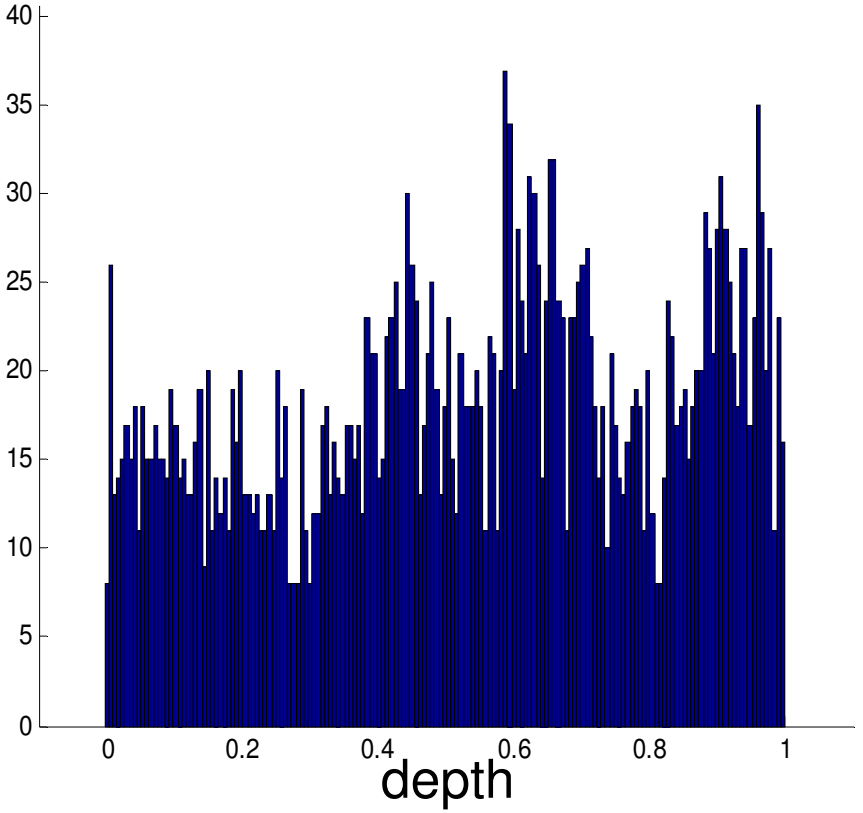


Clearly defined peak

Untextured point



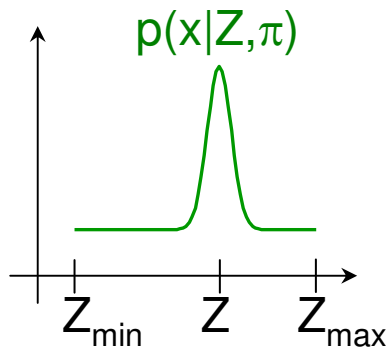
Occluded point



Strategy

- Model sensor probabilistically as a Gaussian+Uniform mixture

$$p(x|Z,\pi)=\pi\, N(x|Z,\tau^2) + (1-\pi)\, U(x,Z_{min},Z_{max})$$



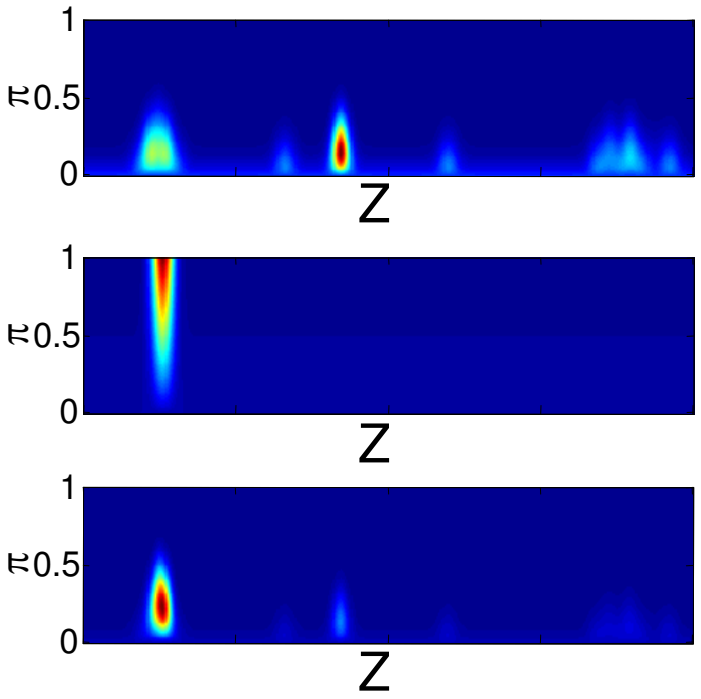
  - Z is the actual depth we are looking for
  - τ^2 is the variance of the sensor (1 pixel estimated from geometry of the problem)
  - Z_{min}, Z_{max} are the limits of the inference problem (bounding volume)
  - x is the measurement
  - π is the inlier ratio

Strategy

- Large quantity of depth measurements $x_1,\ldots,x_t$ available
- We could get max likelihood solution for Z and π with EM, but cannot afford EM for every pixel for every iteration
- Particle filtering is CPU intensive
- Histograms are memory intensive
- Need *sequential inference* of
  - Depth of pixel, Z
  - Inlier ratio for pixel, π

Sequential inference

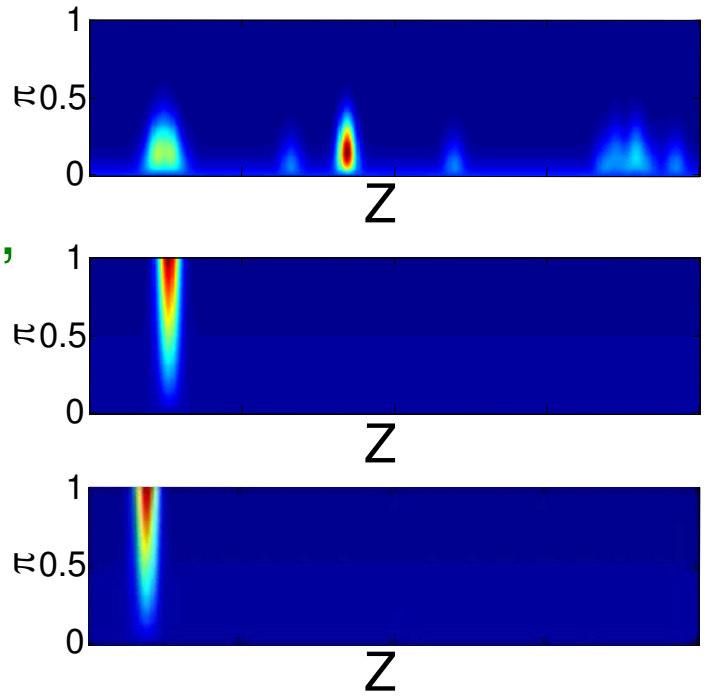
- Posterior at time t,
  $p_t(Z, \pi) := p(Z, \pi | x_1, \ldots, x_t)$
- Likelihood of measurement at t+1,
  $p(x_{t+1} | Z, \pi)$
- Posterior at time t+1,
  - $p_{t+1}(Z, \pi) \propto p(x_{t+1} | Z, \pi) \times p_t(Z, \pi)$



- What form can $p_t(Z, \pi)$ take?
  - Non-parametric 2d histogram is too expensive
  - Can we approximate with a parametric form?

Sequential inference

- **Posterior** at time t,
  $p_t(Z, \pi) := p(Z, \pi | x_1, \ldots, x_t)$
- **Likelihood** of measurement at t+1,
  $p(x_{t+1} | Z, \pi)$
- **Posterior** at time t+1,
  - $p_{t+1}(Z, \pi) \propto p(x_{t+1} | Z, \pi) \times p_t(Z, \pi)$



- What form can $p_t(Z, \pi)$ take?
  - Non-parametric 2d histogram is too expensive
  - Can we approximate with a parametric form?

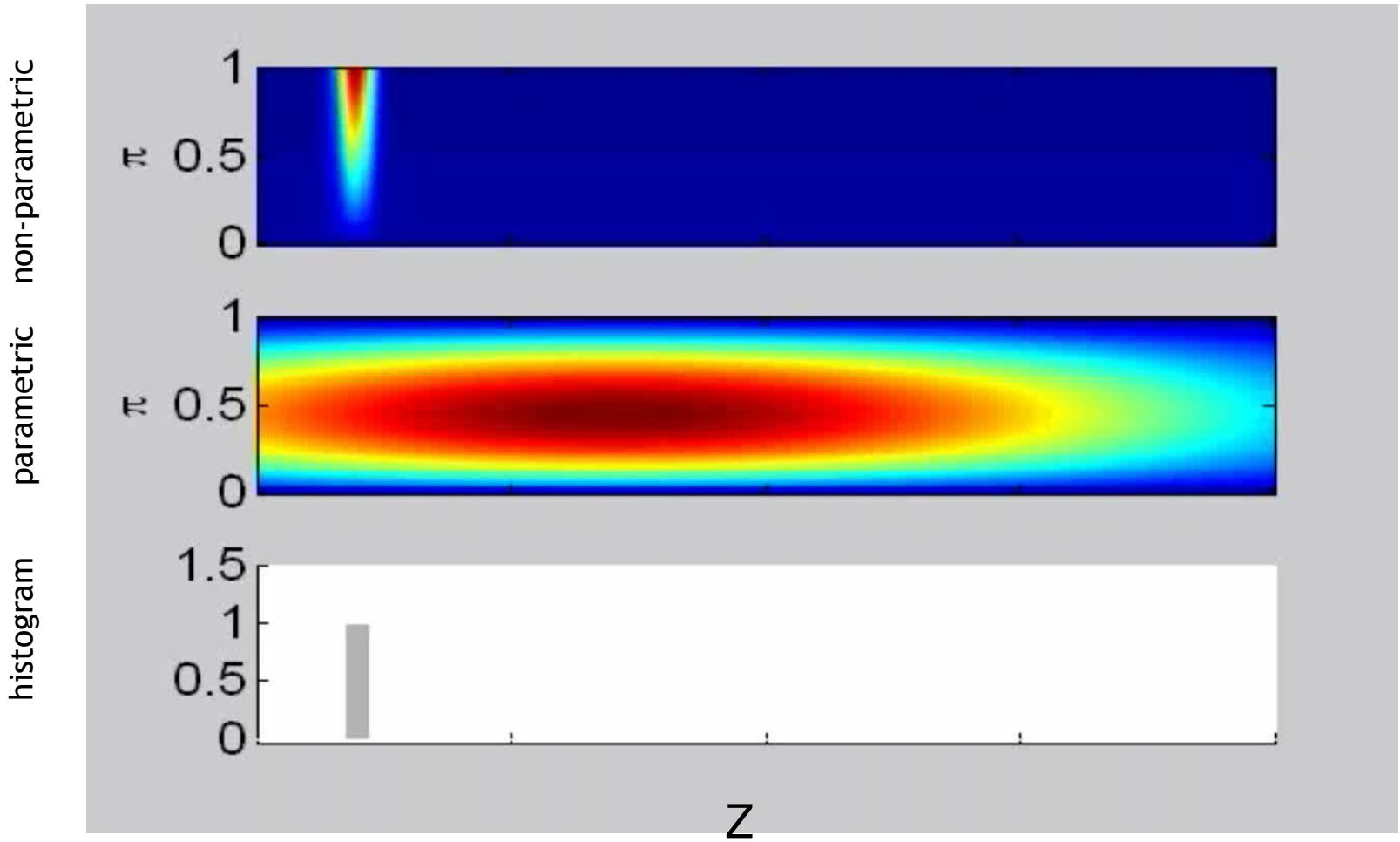
Parametric posterior

- Maintain a posterior of the form
  - $p_t(Z, \pi) = \text{Beta}(\pi | a_t, b_t) \times N(Z | \mu_t, \sigma_t)$
  - Unimodal
  - Beta models random variable in [0,1]

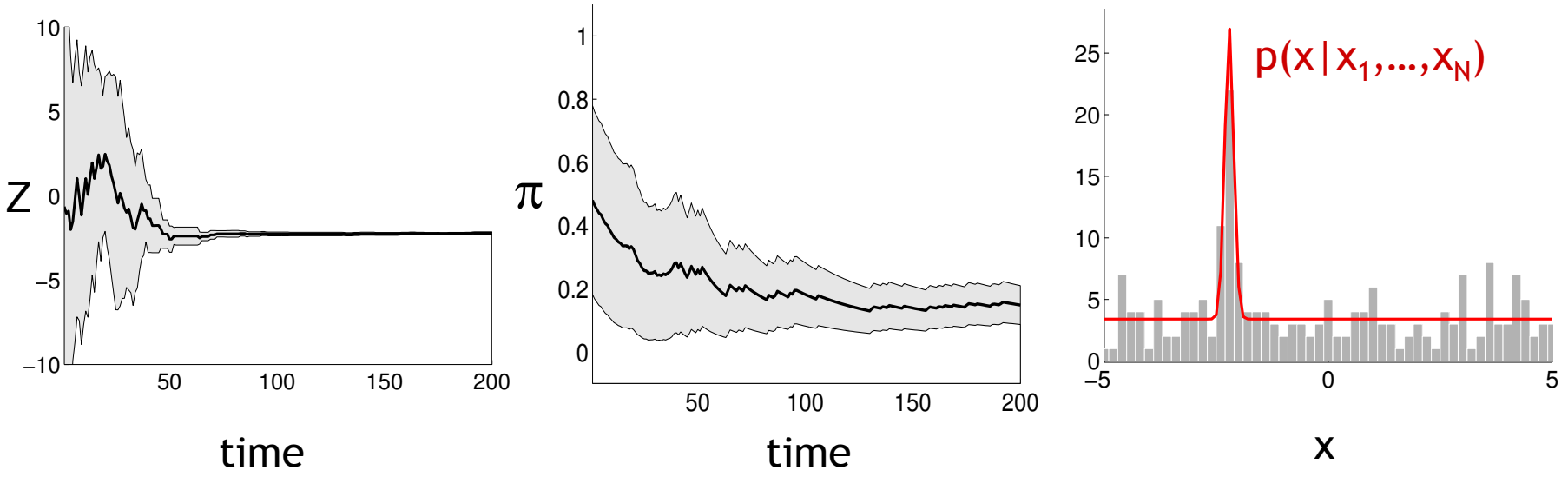
Posterior represented with 4 numbers

- Use moment matching to obtain $p_{t+1}(Z, \pi)$ from $p_t(Z, \pi)$ and $p(x_{t+1}|Z,\pi)$

How well does it work?

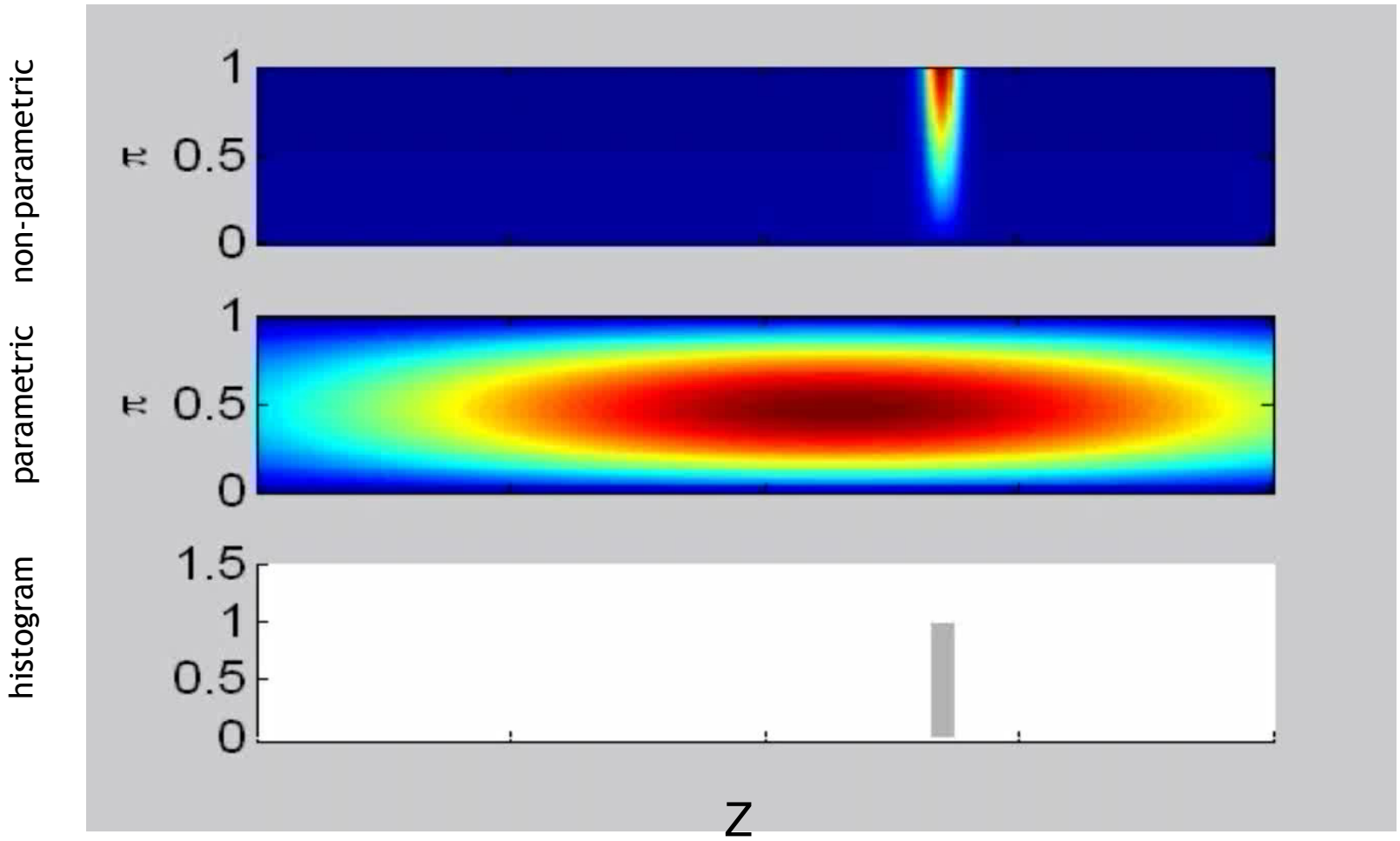


Evolution of estimates



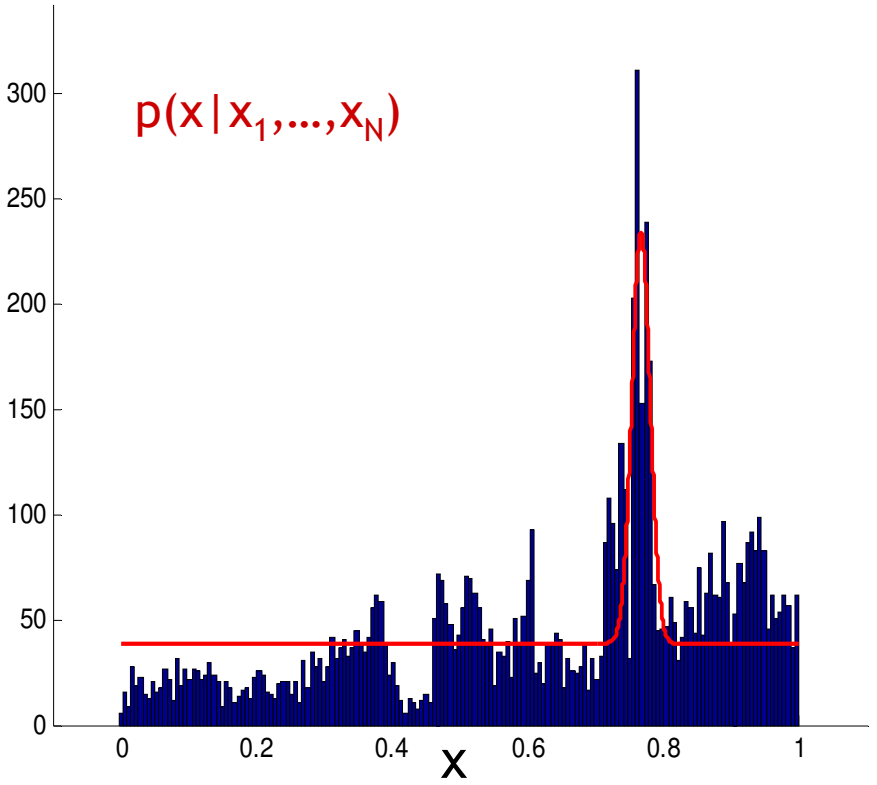
Intervals are two standard deviations either side away from the mean

Failure case of parametric posterior



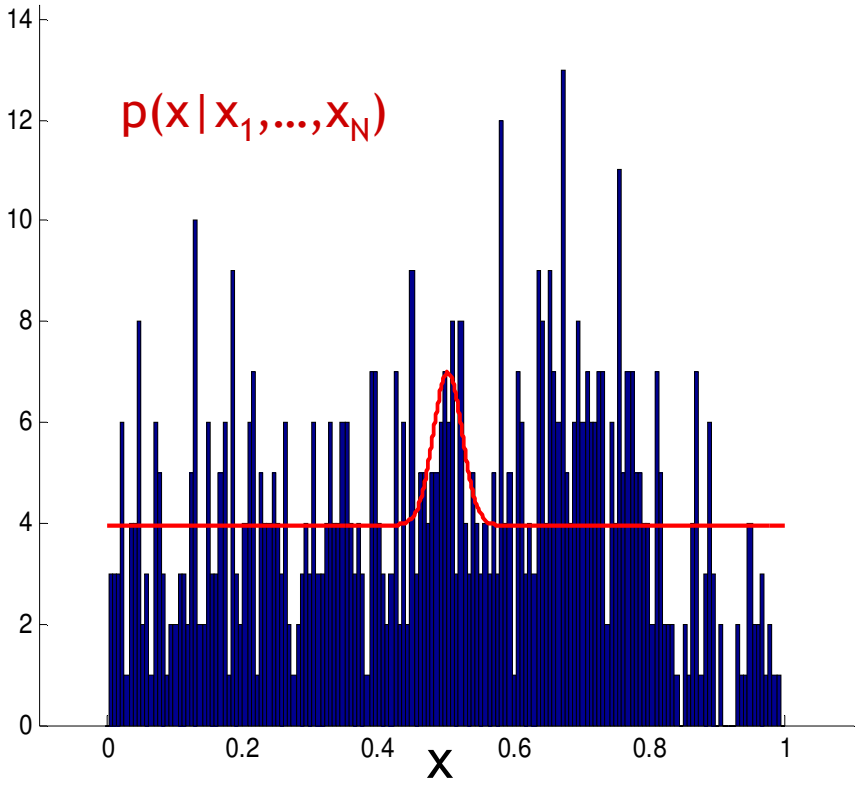
Textured point

π=14.5%



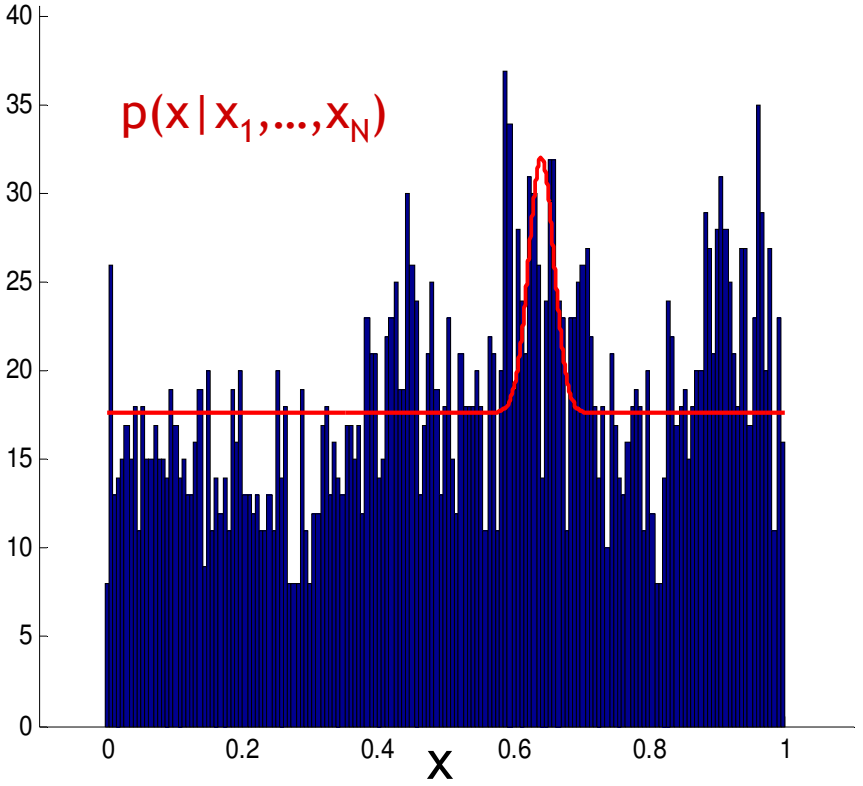
Untextured point

π=3.7%



Occluded point

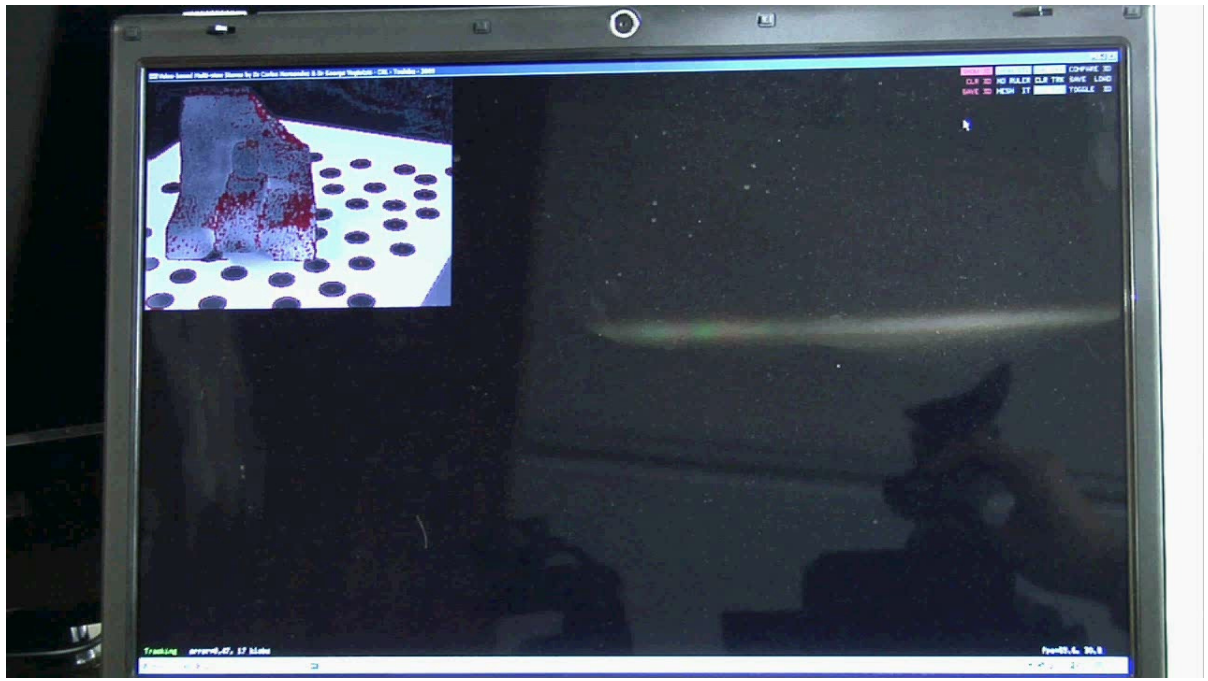
π=3.4%



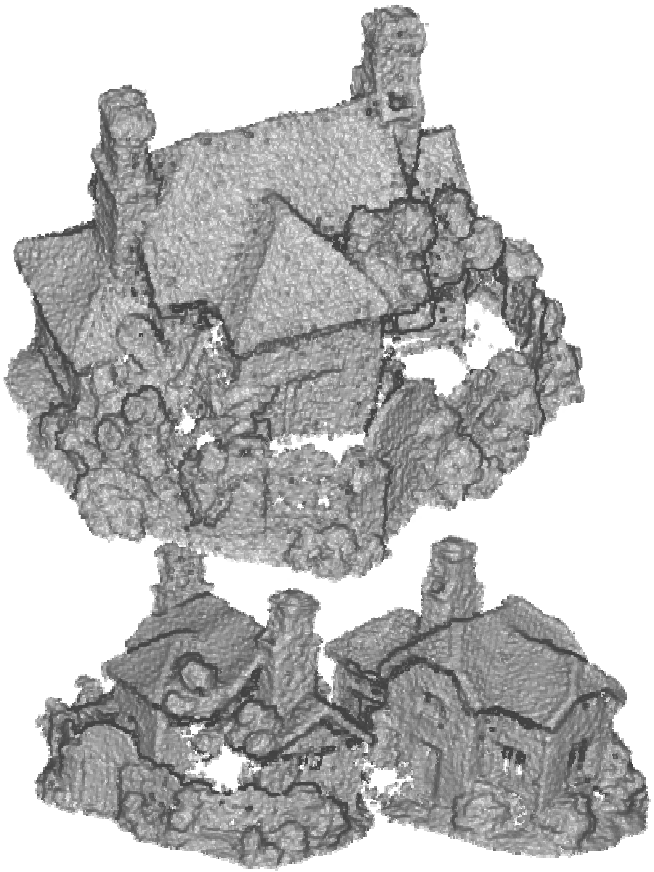
Interactive Multi-view stereo

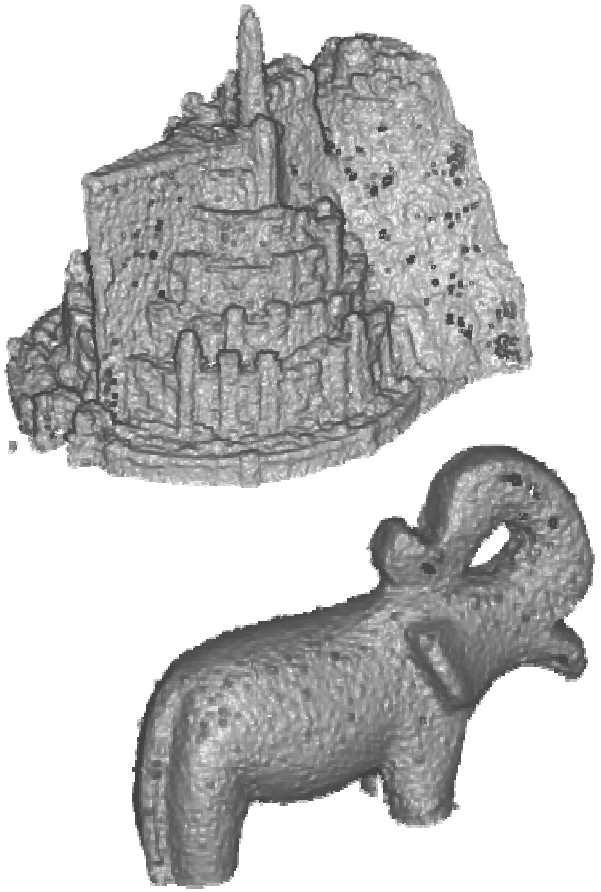
Benefits:

- Faster
- Feedback leads to better models
- Still passive & cheap

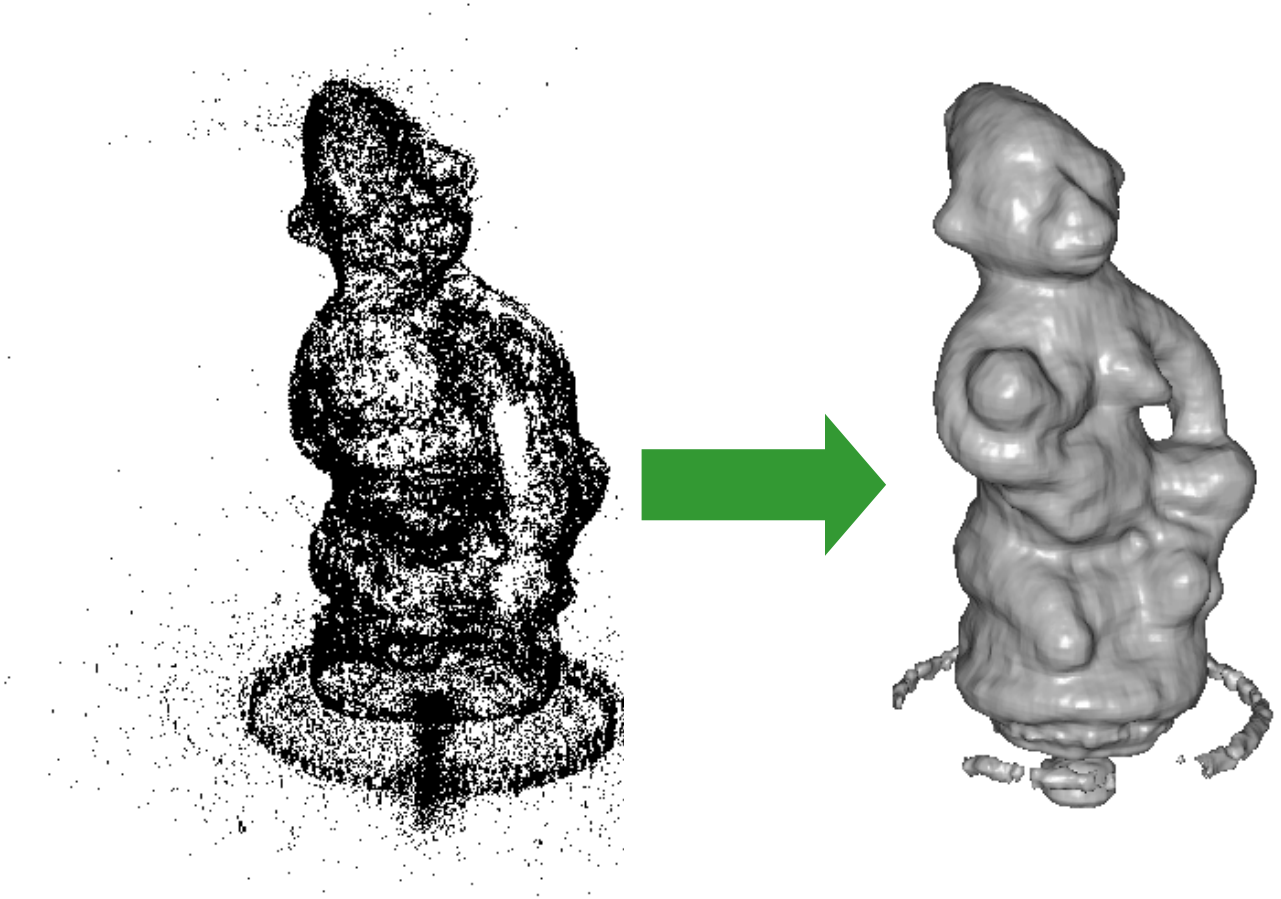


Some reconstructions





3. Extract surface

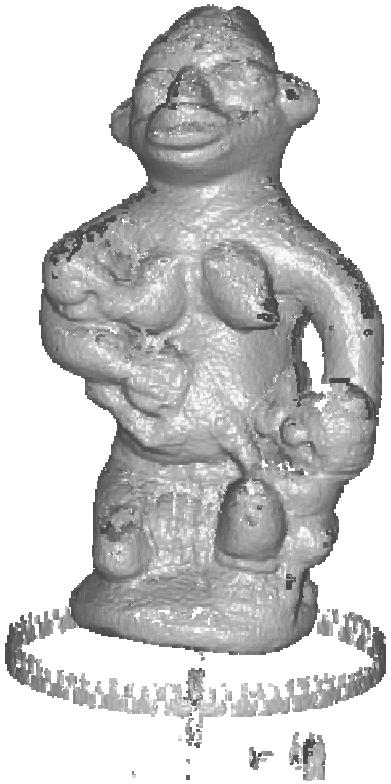


- 3d deformable meshes
- Graph-cuts
- Continuous convex optimisation

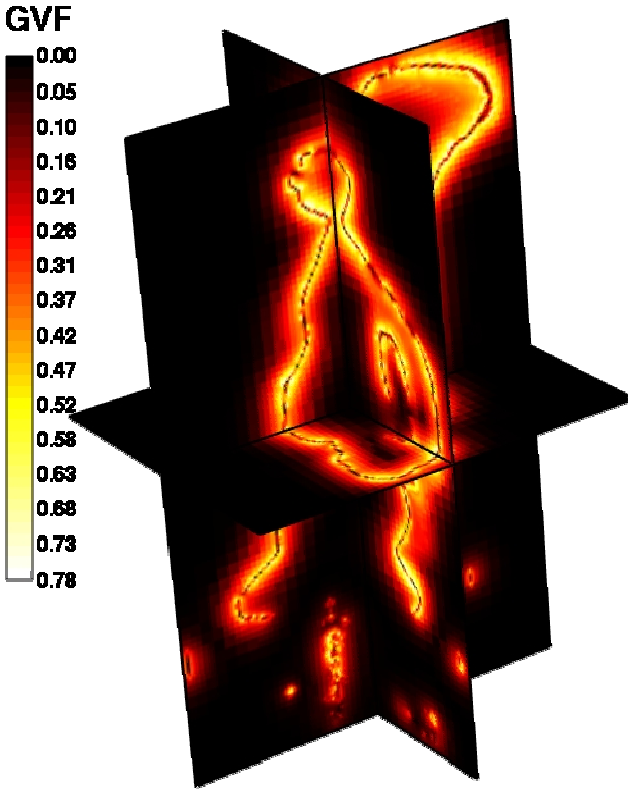
3D deformable meshes

- Effectively gradient descent
  - moving mesh towards photo-consistent locations
  - Smoothing at every iteration
    - $V_{t+1} := V_t + \alpha F_{data} + \beta F_{regularisation}$
- Advantages:
  - Fine control of the surface
  - First order (Laplacian) and second order (biharmonic) regularisation schemes available
  - Multiple clues can be fused easily, e.g. shading
- Cons:
  - Global convergence is **not** guaranteed
  - Difficult to cope with topology changes

Improving convergence

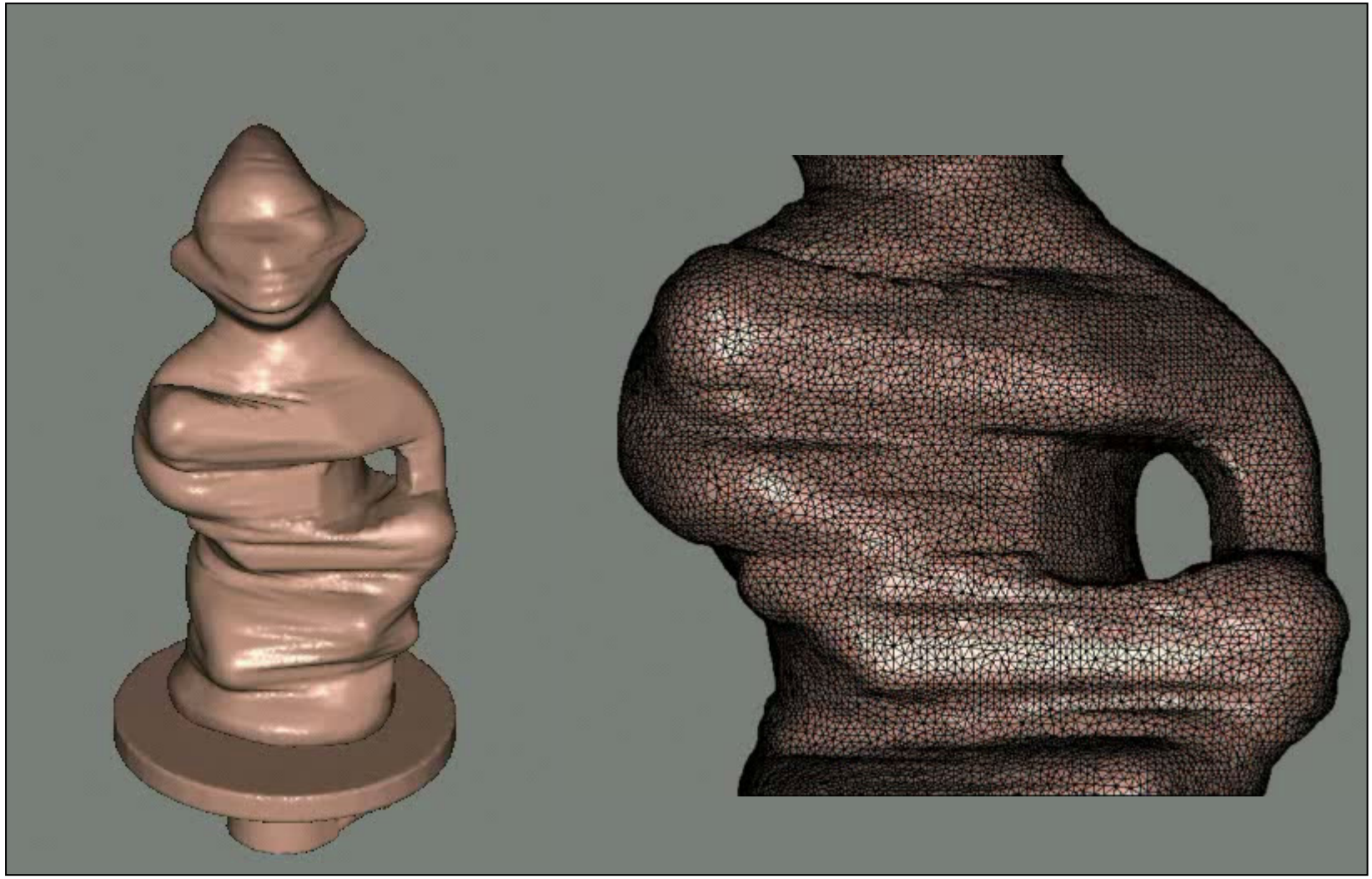


Robust NCC volume



Gradient Vector Flow

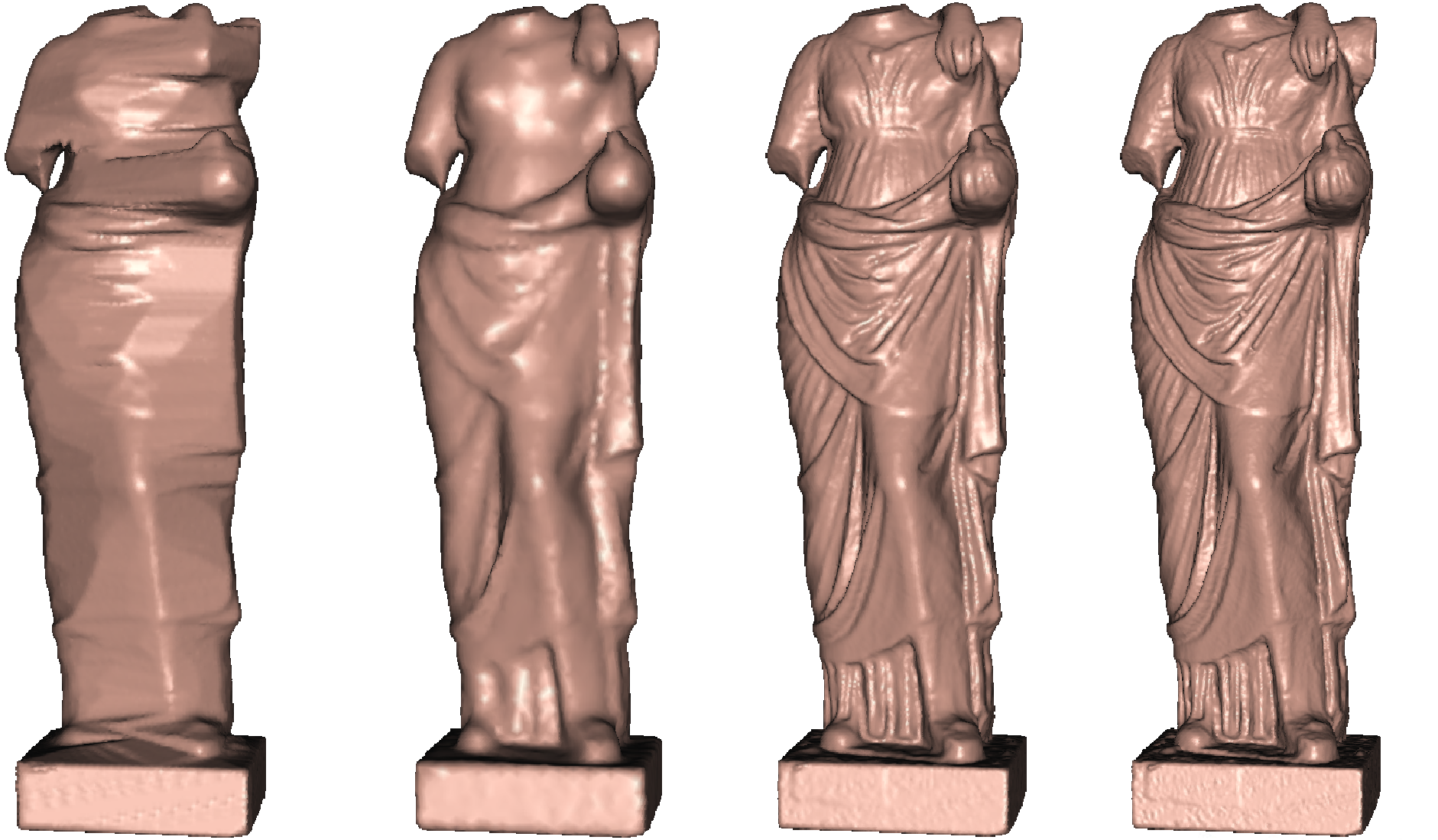
3D deformable meshes



Results

Results

Results



Aston University
Birmingham

3D deformable meshes

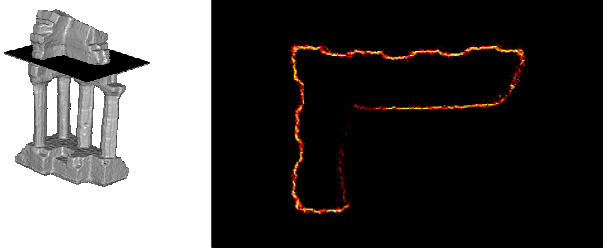
- Is local optimisation (gradient descent)
- Can get stuck in local minima
- Mesh can self-intersect
- Usually very slow
  - Typically hours of computation time
- Cannot change topology
  - That could be an advantage, more regularized
- Can we do better?

Extracting a surface from photo-consistency

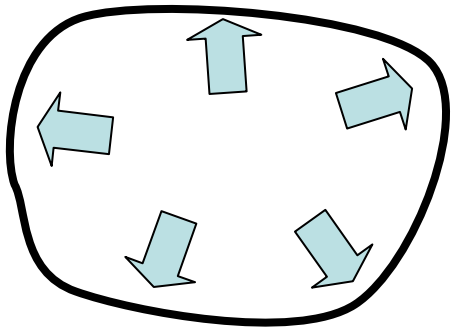
- Define a surface functional E[S] = some cost
- Minimize E
- Need *volume term* to avoid collapsing to a point

$$E[S] = \iint_S \rho(x) dS$$

ρ(x) = 1.0 - Photoconsistency



σ(x)= -1 (inflation)



Segmentation

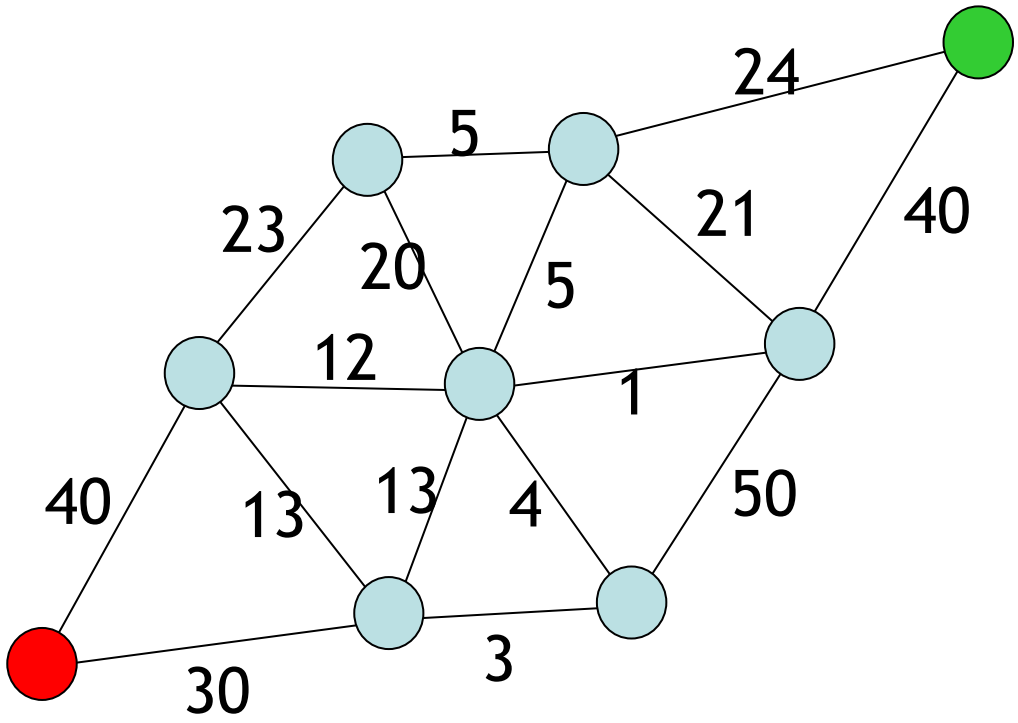
- In segmentation tasks we combine an edge term with a foreground/background term (e.g. ‘grab-cuts’ 2004)

$$E[S] = \iint_S \rho(x) dS + \iiint_{V(S)} \sigma(x) dV$$

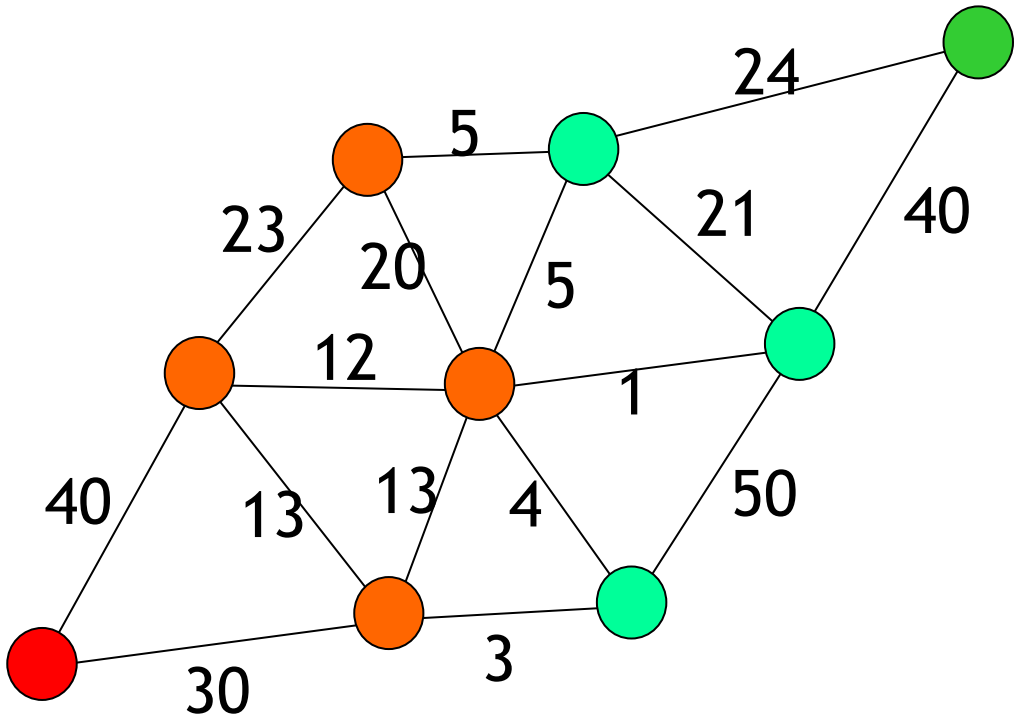
Edge cost

Foreground/background cost

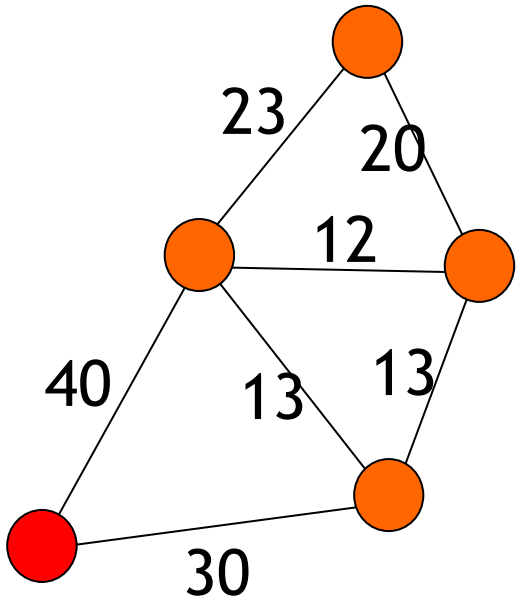
How to solve this?

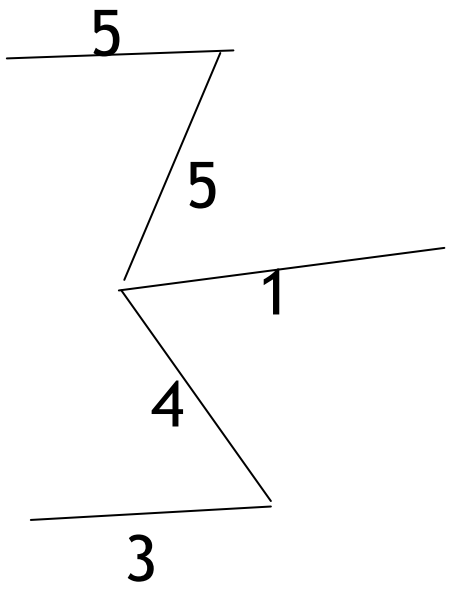


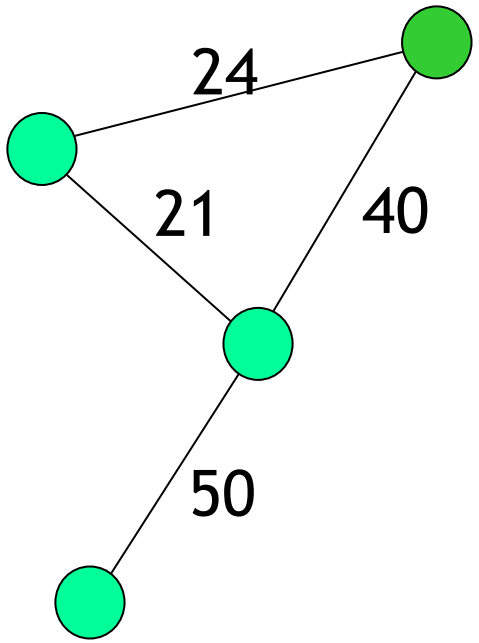
Graph *cut*



Minimum cut



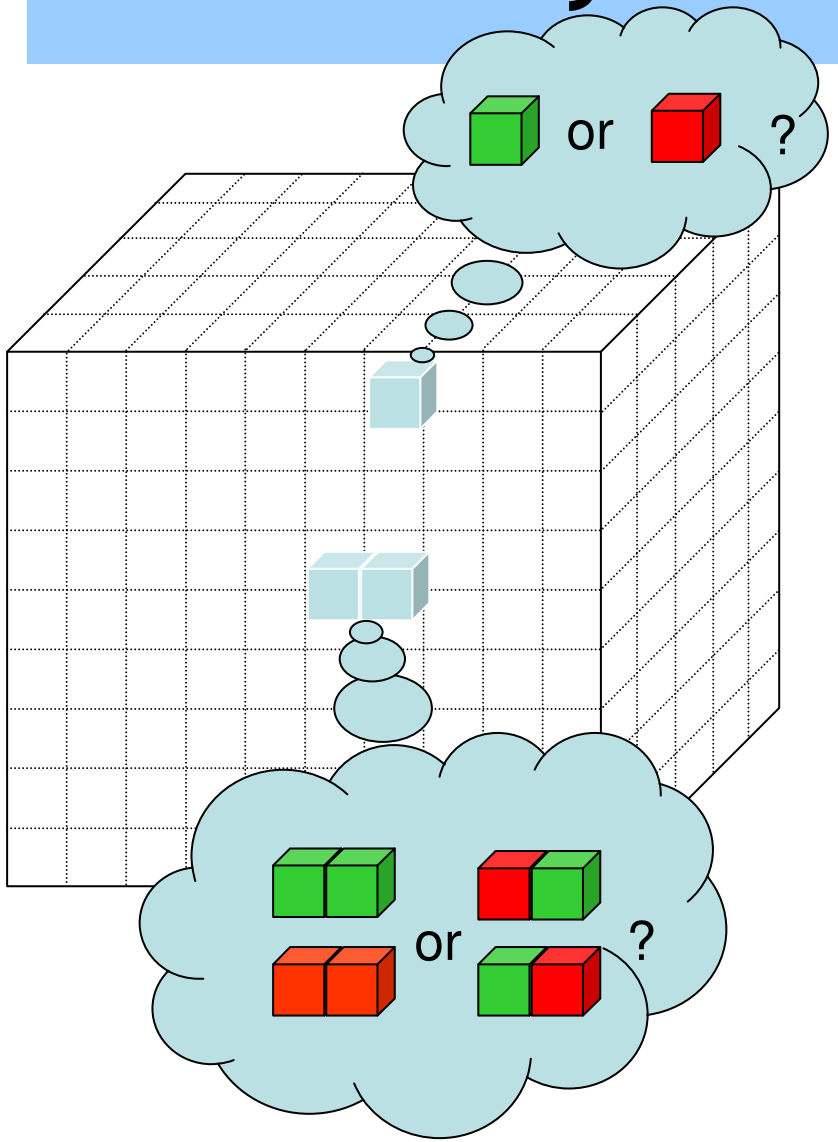




5+5+1+4+3=18

Can be computed
in polynomial time
with *Ford-Fulkerson (1956)*
algorithm

3D binary labelling problem

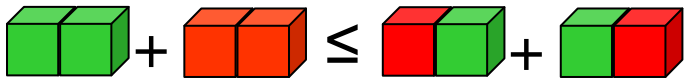


Labelling cost:

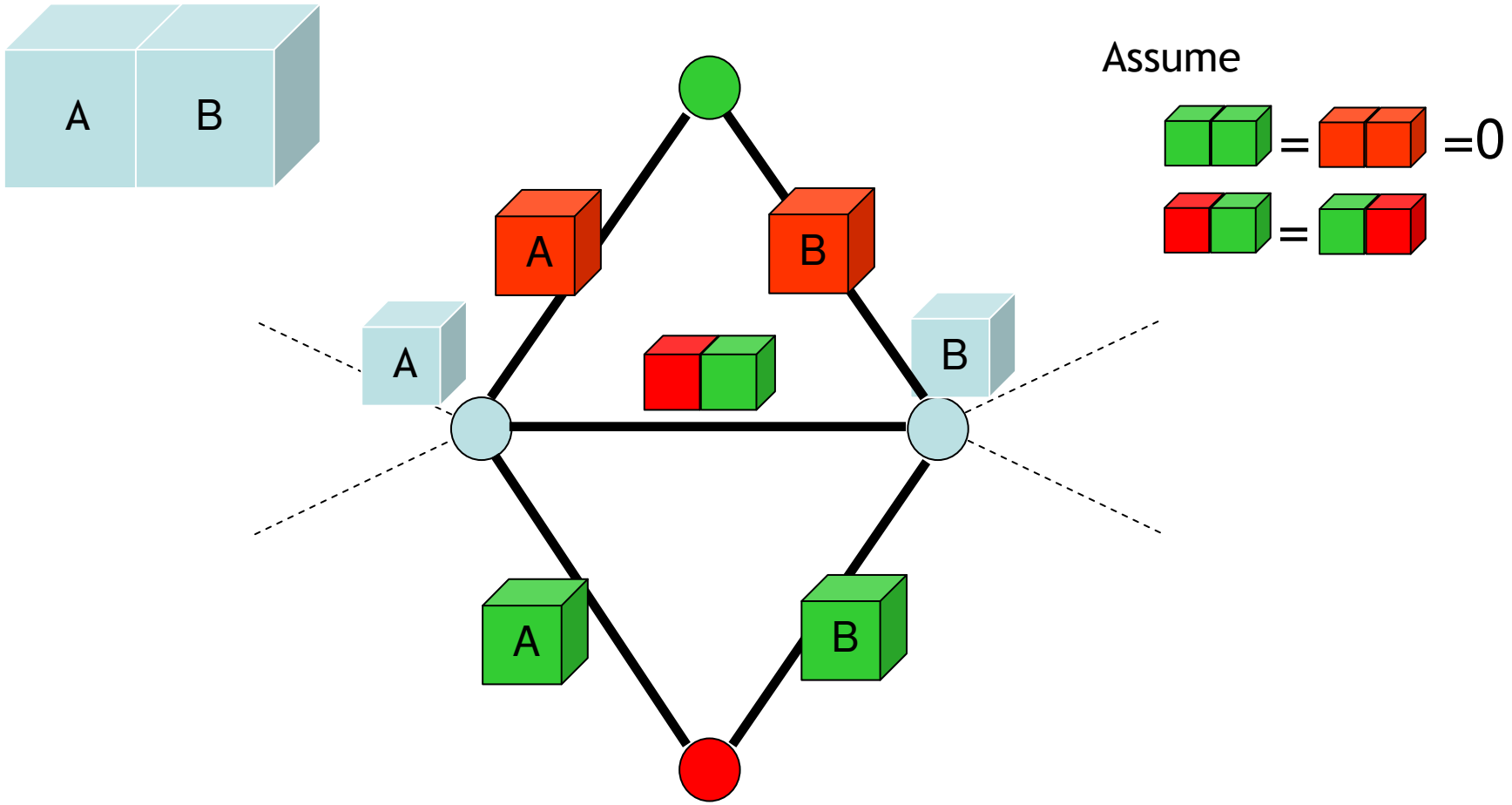
- Every voxel has a certain preference for being **foreground** or **background**

Compatibility cost:

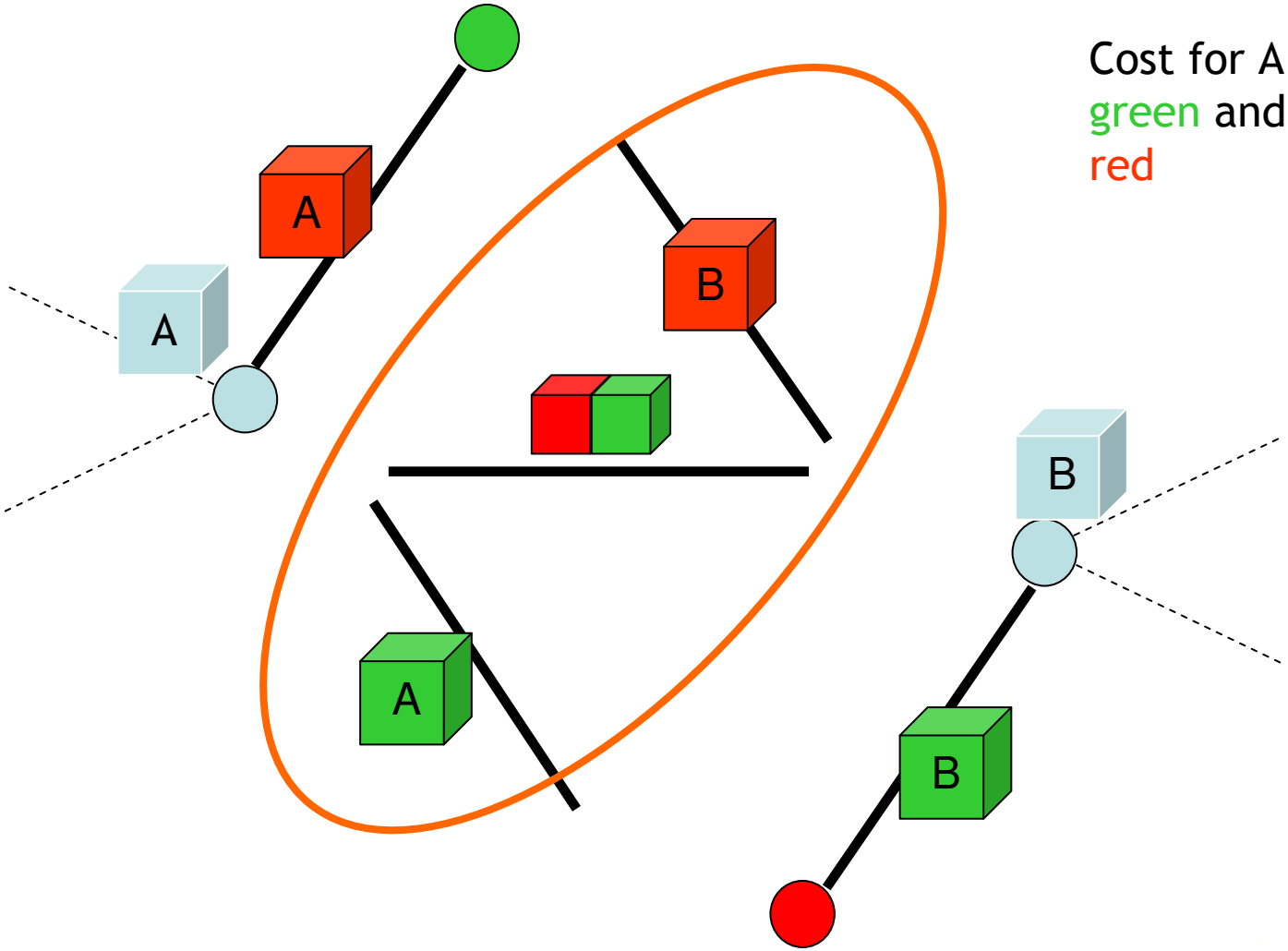
- Every pair of neighbour voxels has a certain preference for being given the *same* or *oposite* labels
- Cost for opposite labels is greater than for same label (sub-modular)



Graph-cuts for binary MRFs



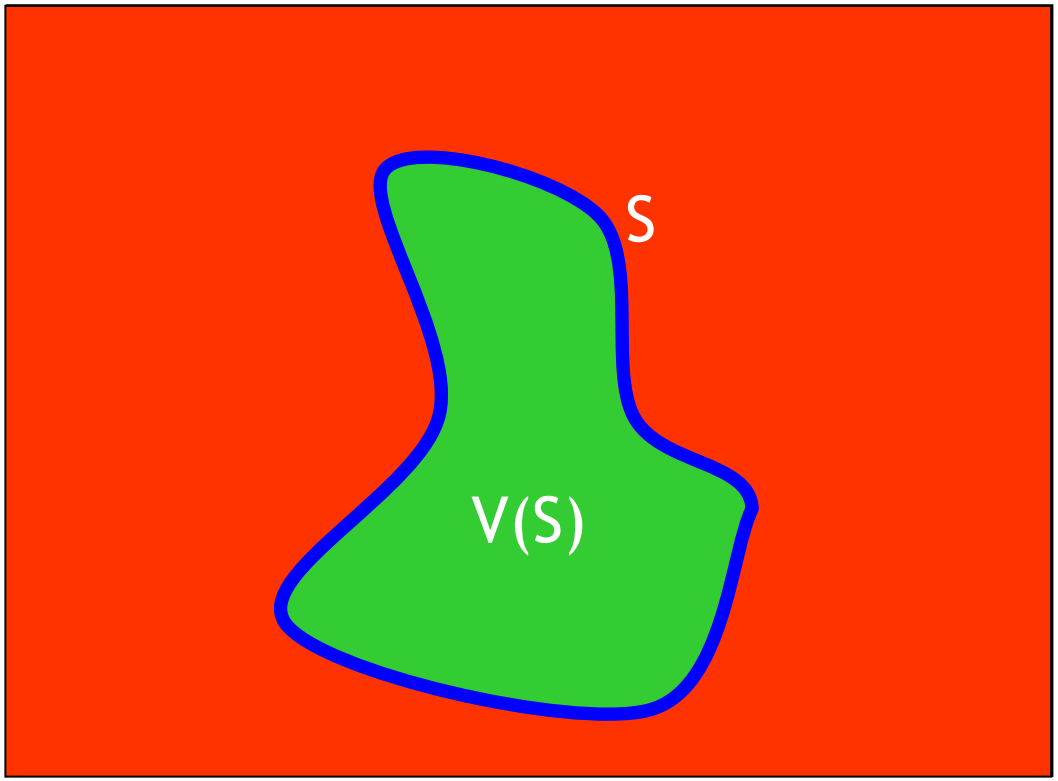
Graph-cuts for binary MRFs



Cost for A becoming green and B becoming red

Binary occupancy representation

Green = IN
Red = OUT

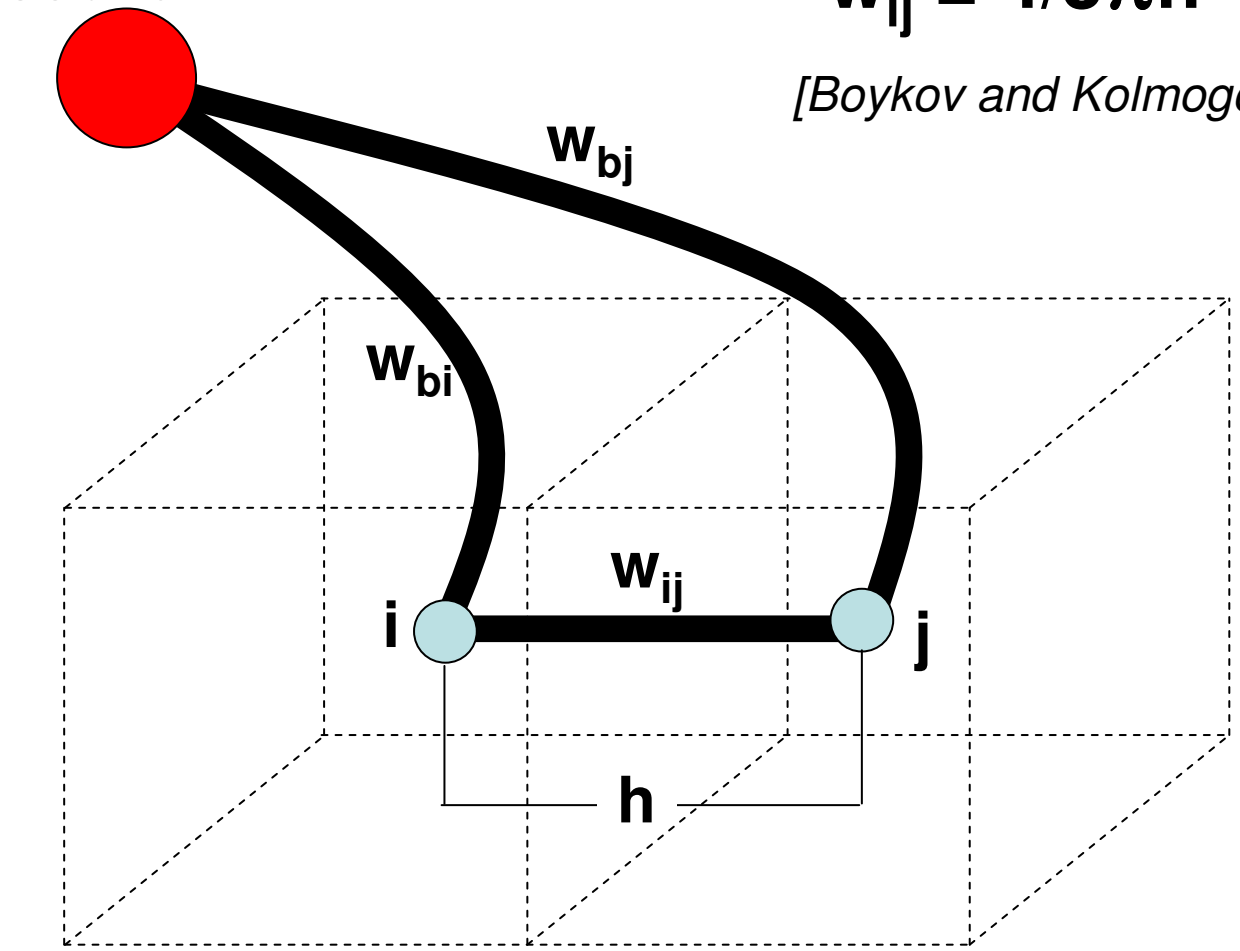


Continuous functional

- 3D binary MRFs can be seen as discrete approximation to surface + volume integral

$$E[S] = \iint_S \rho(x)dS + \iiint_{V(S)} \sigma(x)dV$$

Graph construction



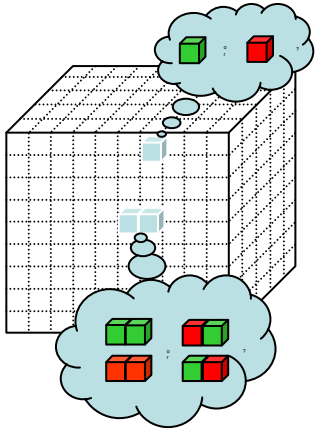
$$w_{ij} = 4/3\pi h^2 * (\rho_i + \rho_j)/2$$

[Boykov and Kolmogorov ICCV 2001]

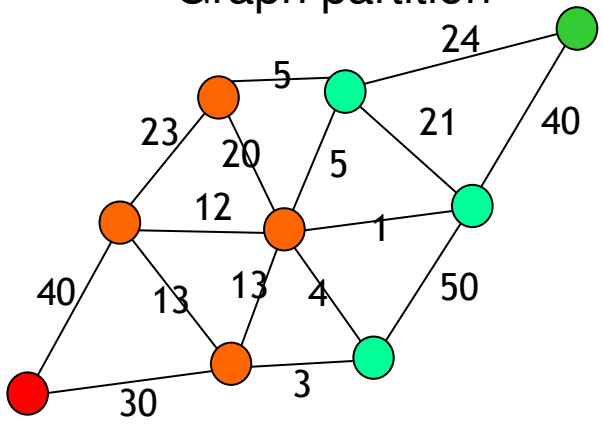
$$w_{bi} = \sigma_i h^3$$
$$w_{bj} = \sigma_j h^3$$

Three equivalent representations

Binary labelling



Graph partition

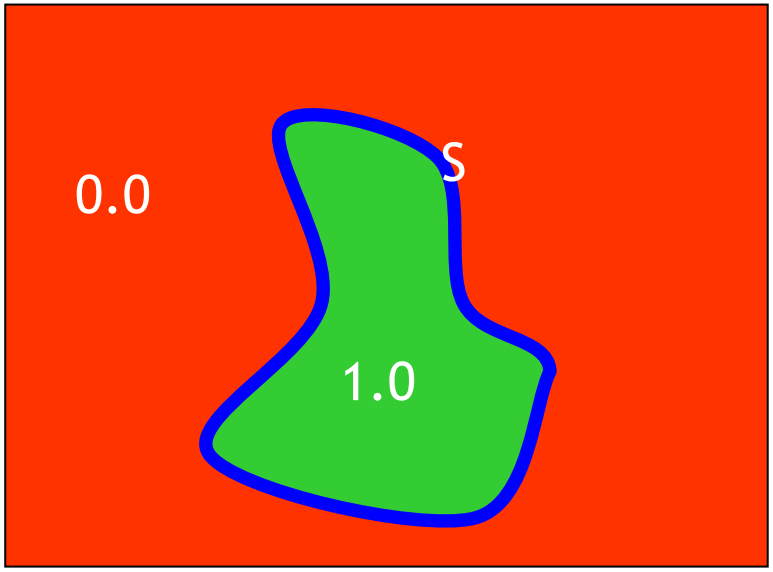


Continious functional

$$E[S] = \iint_S \rho(x)dS + \iiint_{V(S)} \sigma(x)dV$$

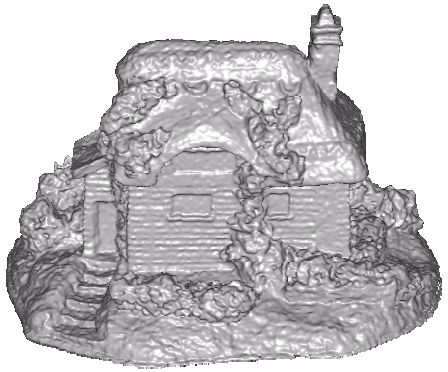
Aston University
Birmingham

Extracting surface from binary map

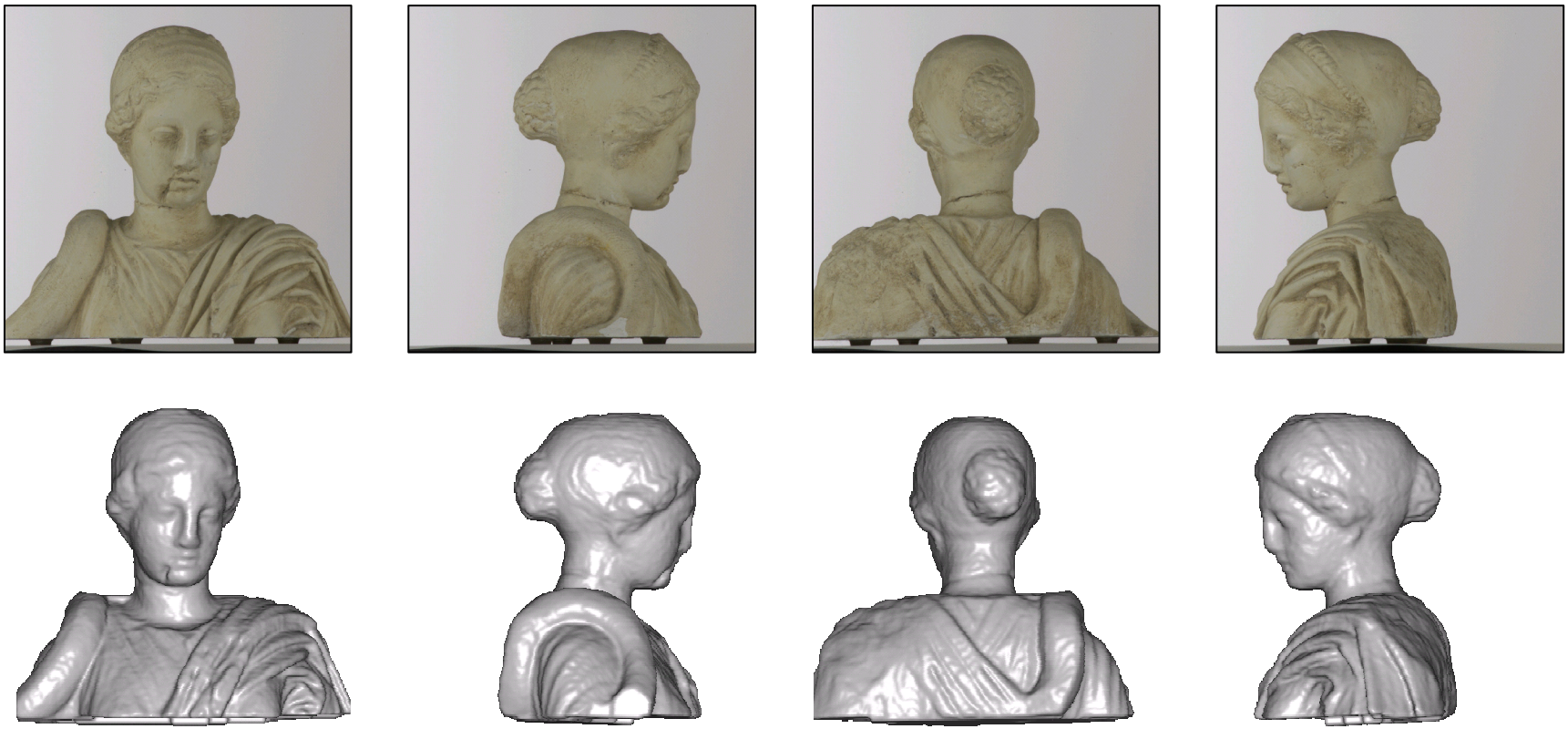


- Surface can be extracted by marching cubes algorithm
- Matlab: `[tri, pts] = isosurface(V)`
  - V is binary 3d volume of 0.0s and 1.0s
  - pts is a 3xN set of 3d points
  - tri is a 3xM set of vertex indices denoting mesh faces

Results

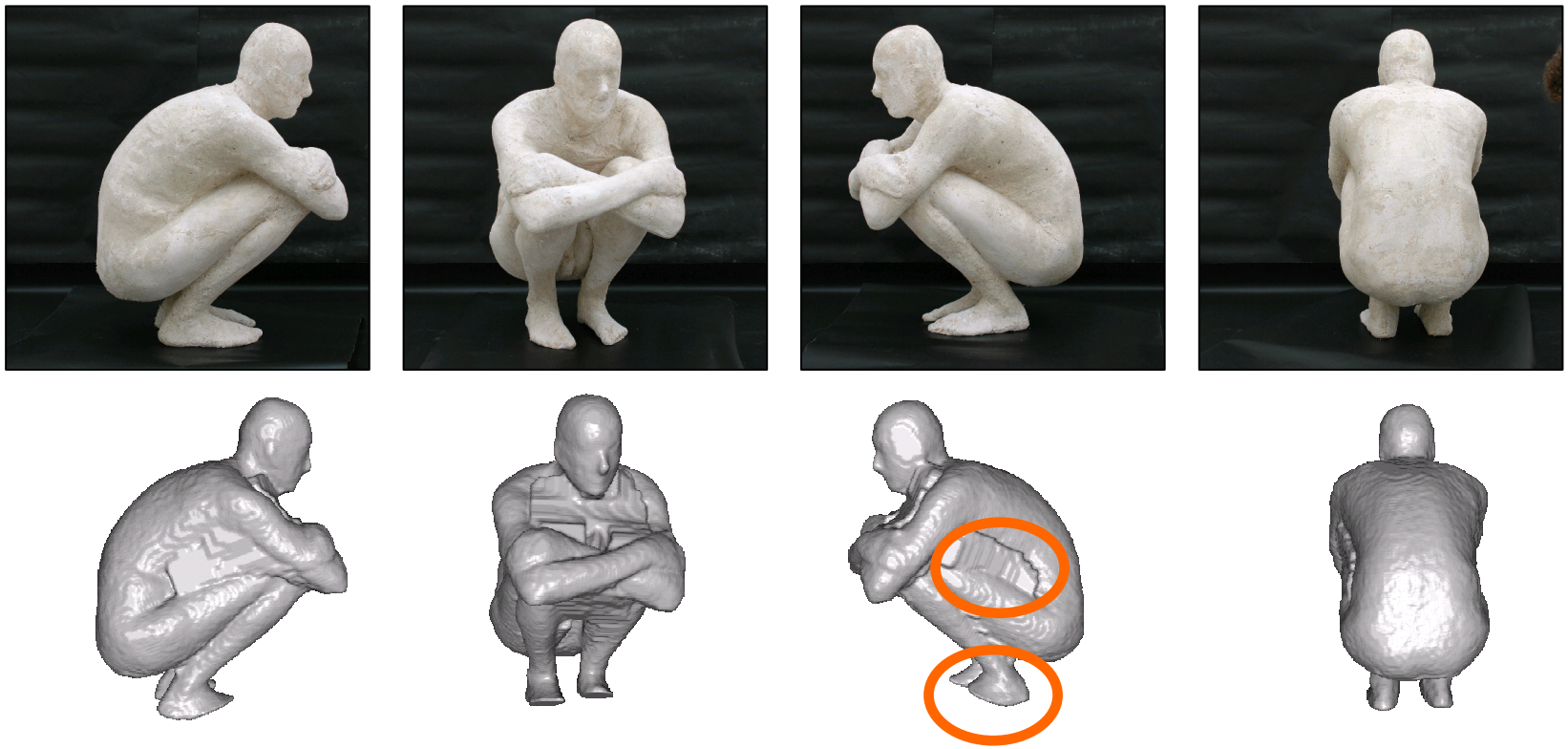


Results



gcut3d binary

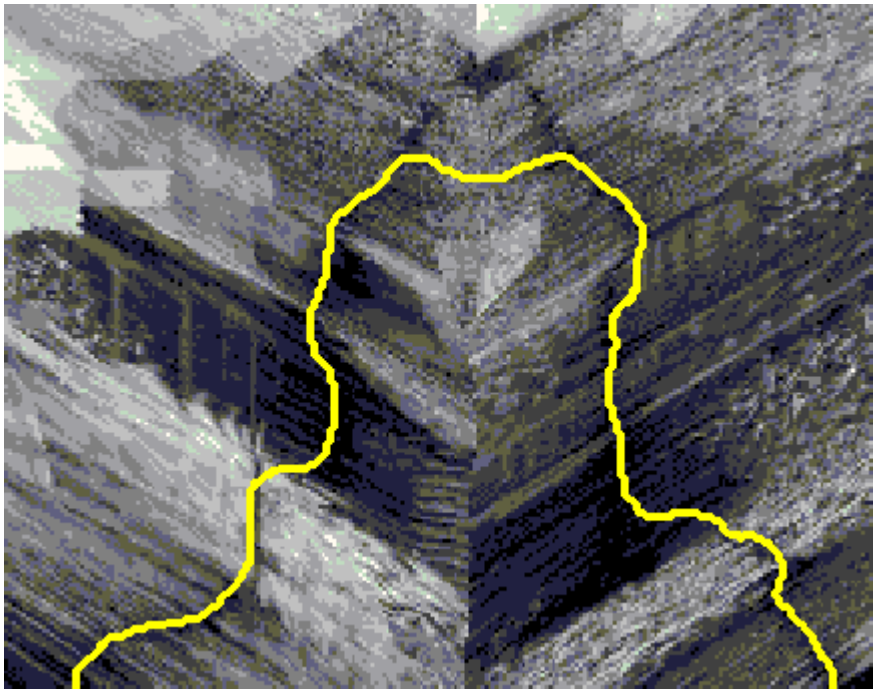
Results

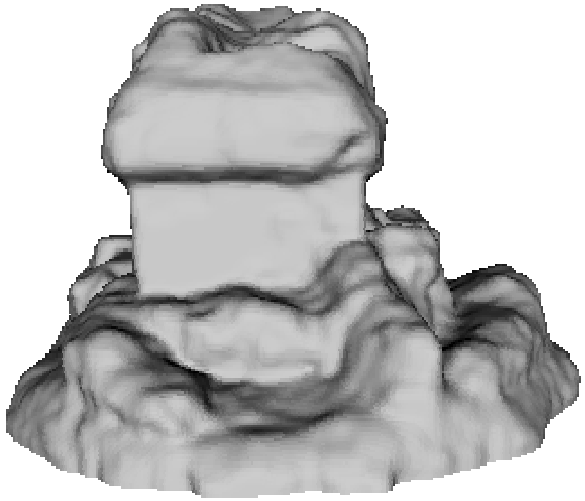


- Problem with concave regions and protrusions

Protrusion problem

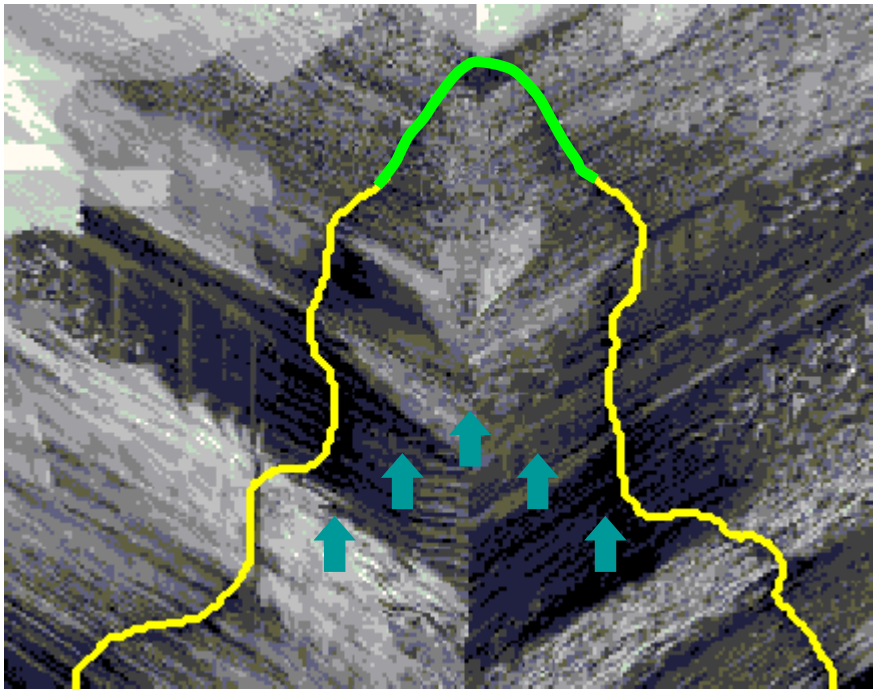


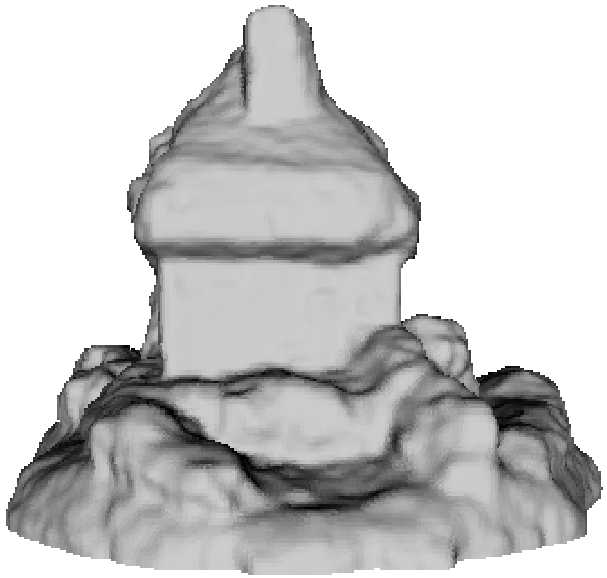




Protrusion problem

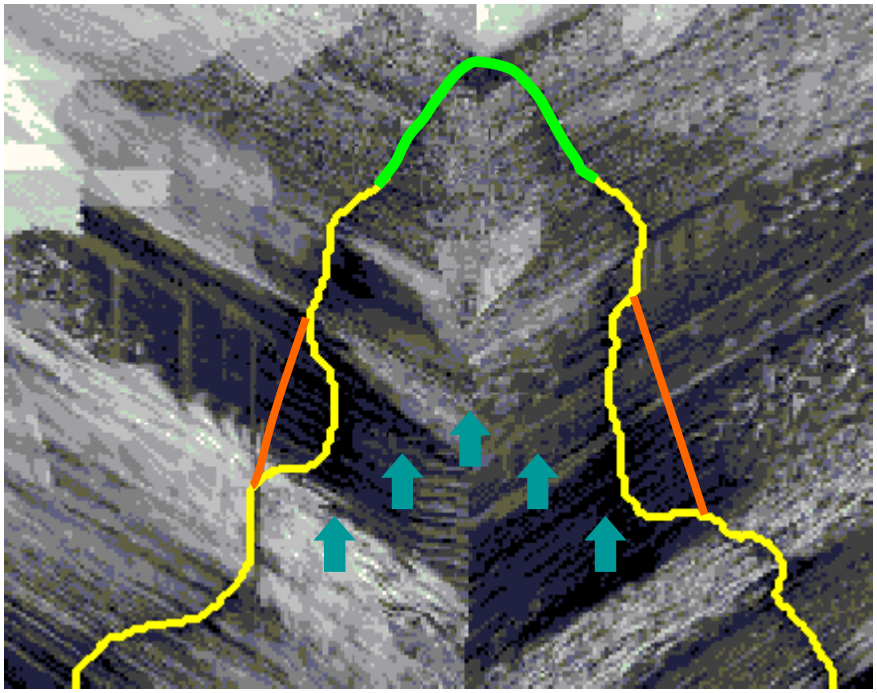


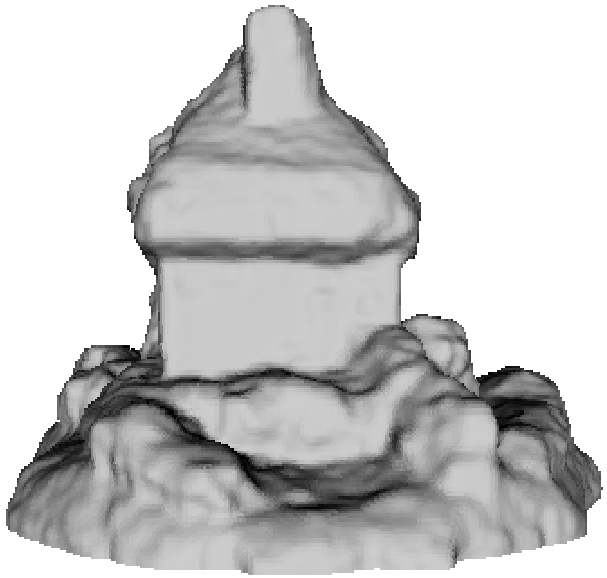




Protrusion problem

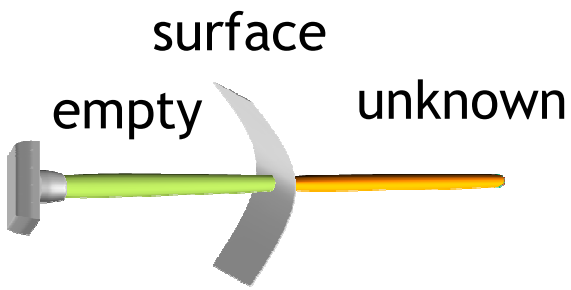


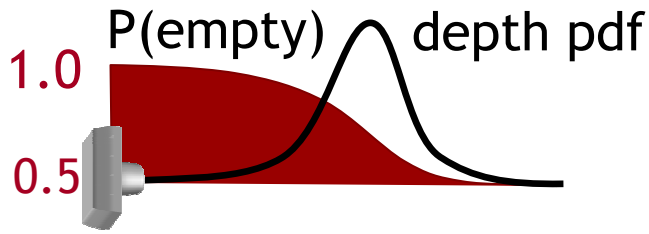




Data driven volume term

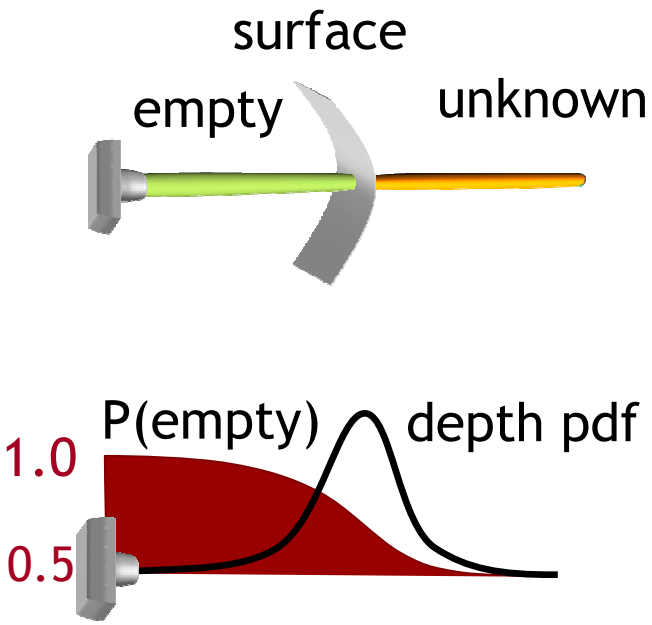
- When a camera makes a depth measurement, also gives info about empty space



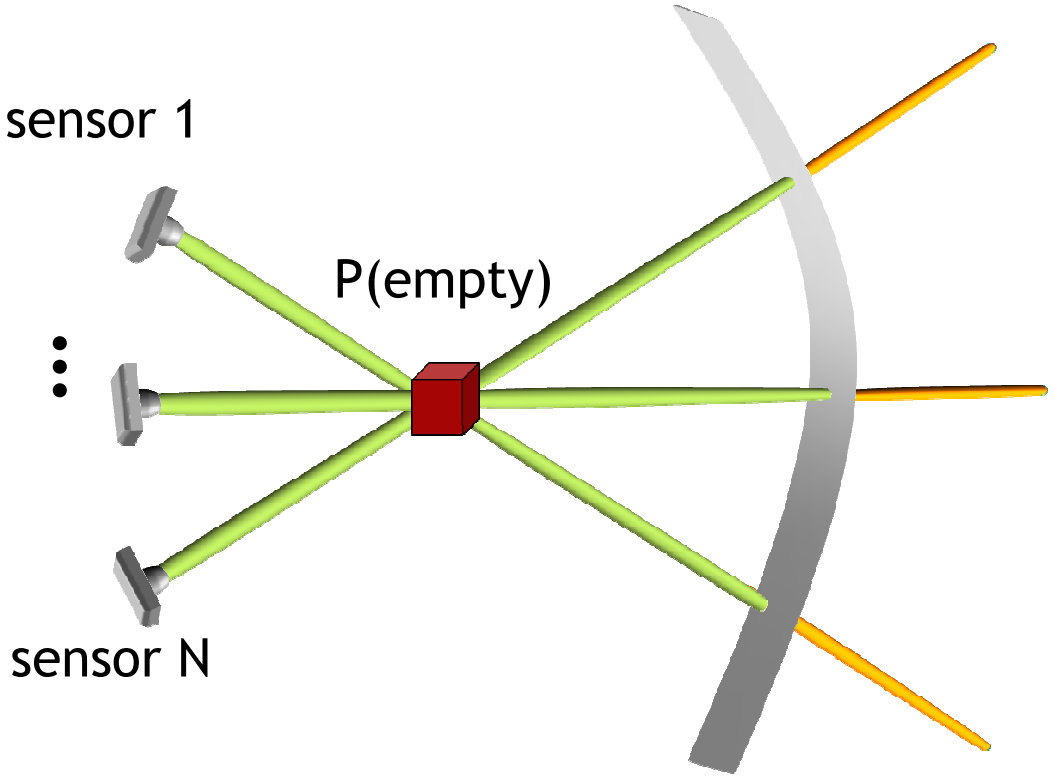


Data driven 3D MRF labelling cost

Probabilistic sensor model

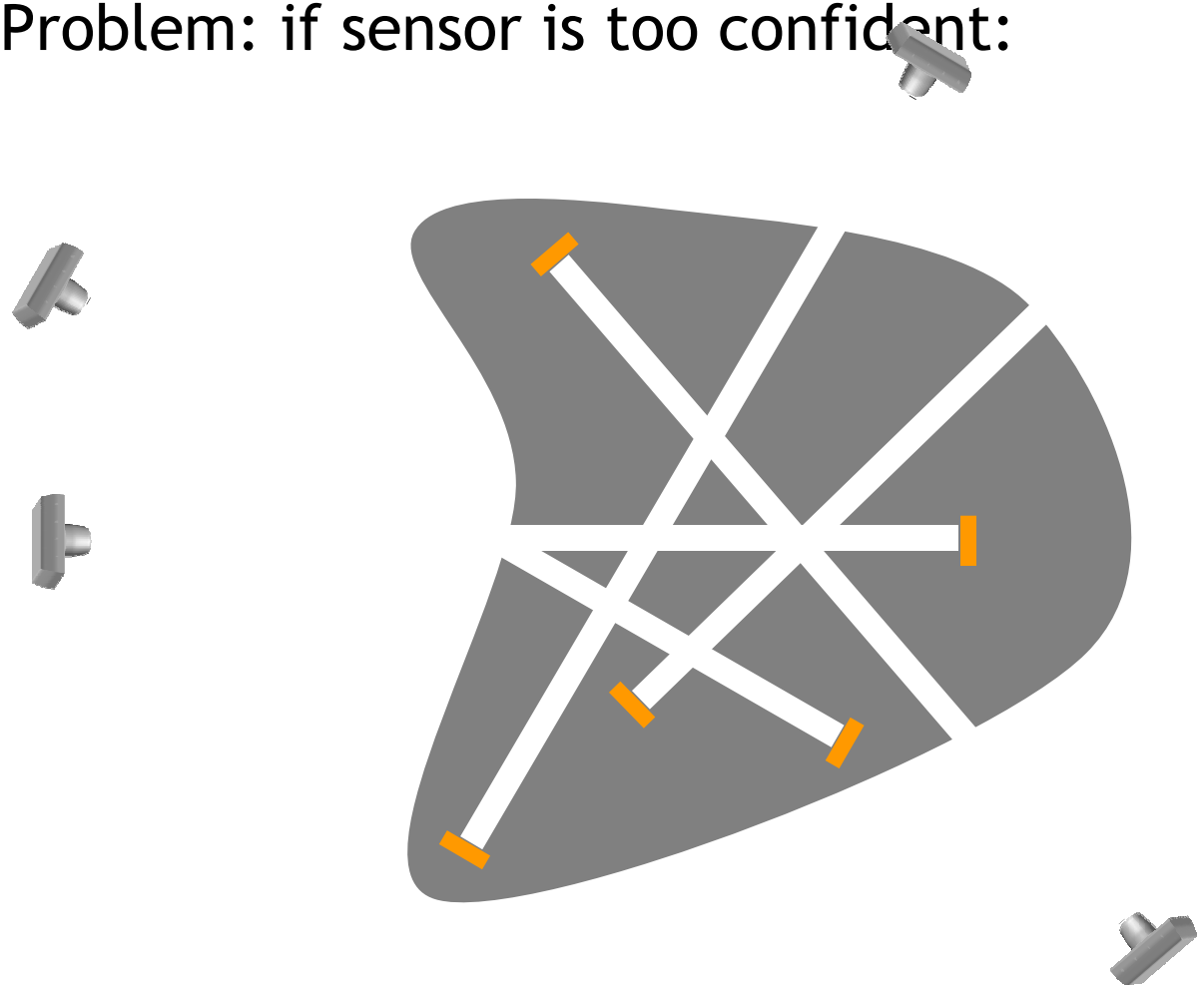


Probabilistic sensor fusion

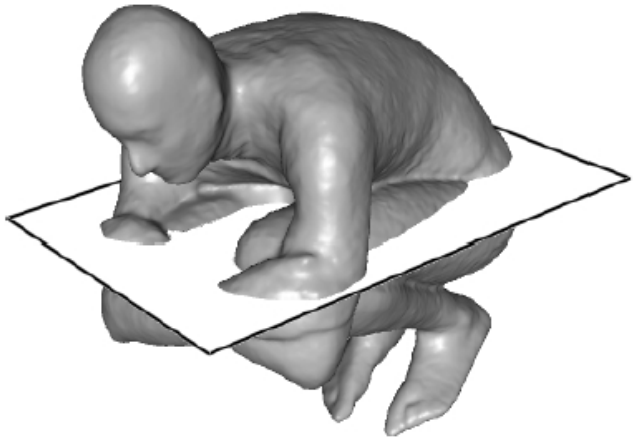


Data driven 3D MRF labelling cost

- Problem: if sensor is too confident:



Data driven 3D MRF labelling cost

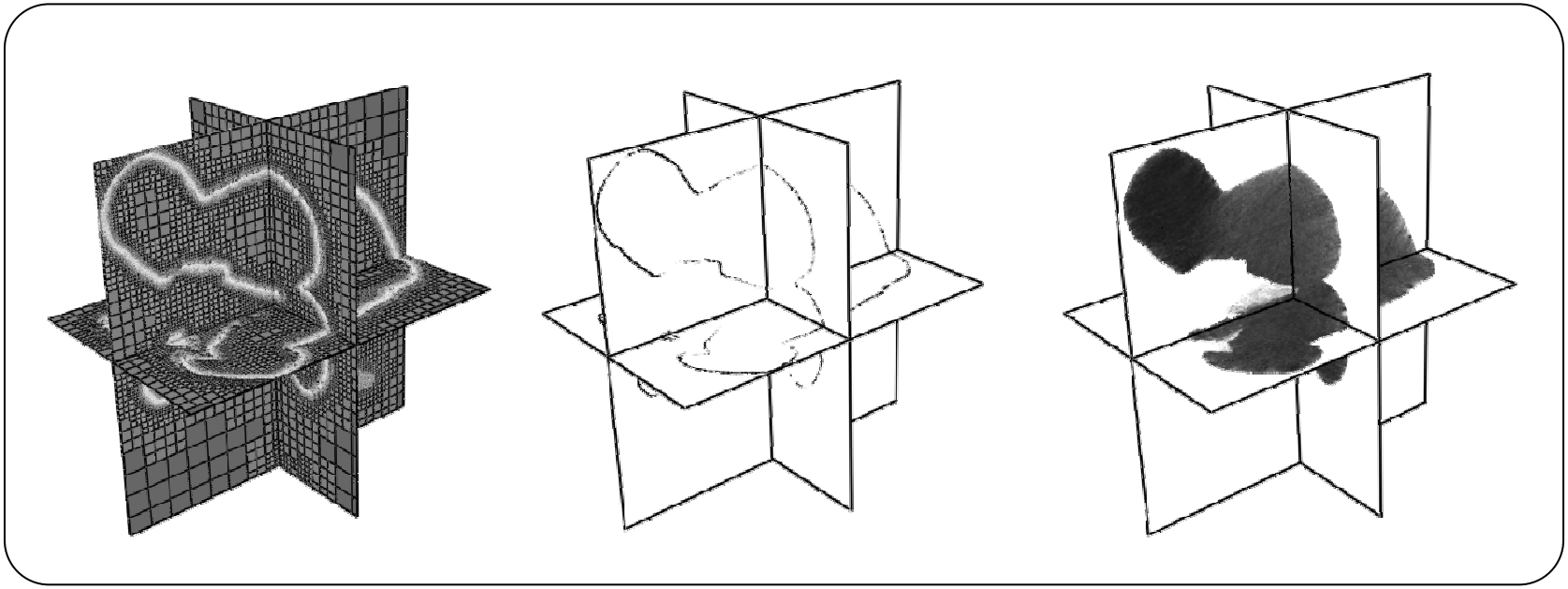


100% trusted sensor

When sensor gets depth wrong, we ‘drill hole’ inside the volume

Solution: sensor depth distribution = Gaussian + Outlier

3D MRF for 3D modelling



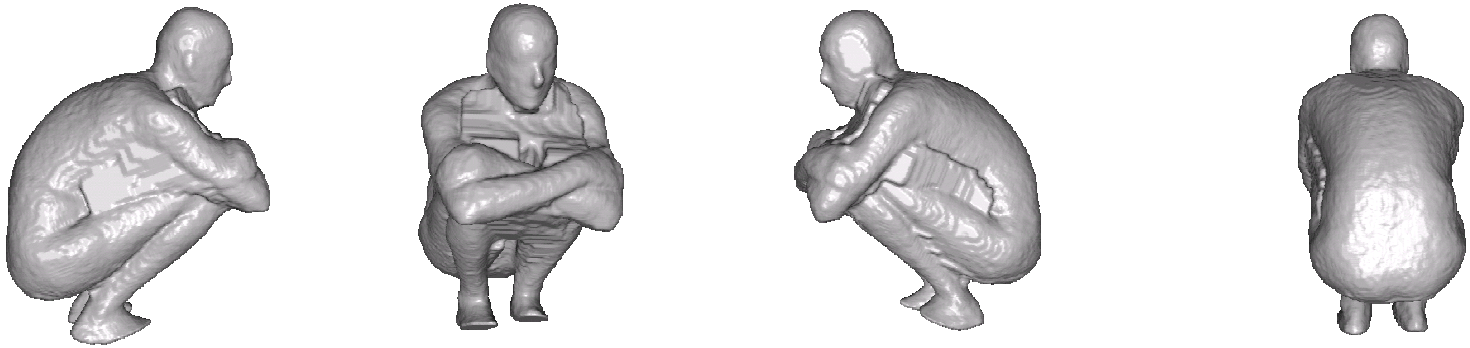
Multi-resolution grid

Edge cost

Foreground/ background cost

Results

constant ballooning



data-driven ballooning