Recovering Geometric Representations – Voxels and Density

D.A. Forsyth,

University of Illinois at Urbana-Champaign

Options

- Here:
  - Multiple depth maps
  - Point clouds
  - Voxel grids
  - Implicit functions
  - Density functions
- Others
  - CSG rep’ns
  - NURBS
  - etc.

Rep’n: voxel grid (++)

- Grid 3D into voxels, label
  - occupied, empty (maybe edge)

- Advantages
  - easy to construct
  - known how to pass to triangle mesh (if dense enough)
- Disadvantages
  - serious problems with resolution and scale
  - many geometric computations quite hard
    - volume, intersection, etc

Volumetric fusion, I

- Work in voxel space
- Depths from cameras
  - yield likely labels for SOME voxels (blue – empty; pink – occupied)
- Other voxels by optimization

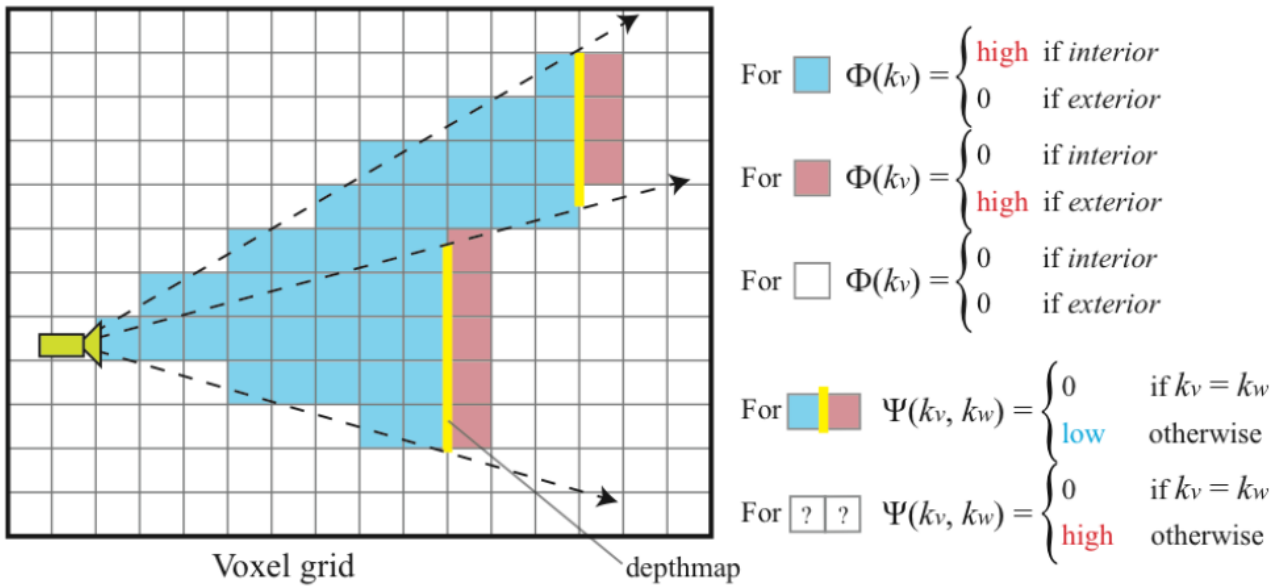
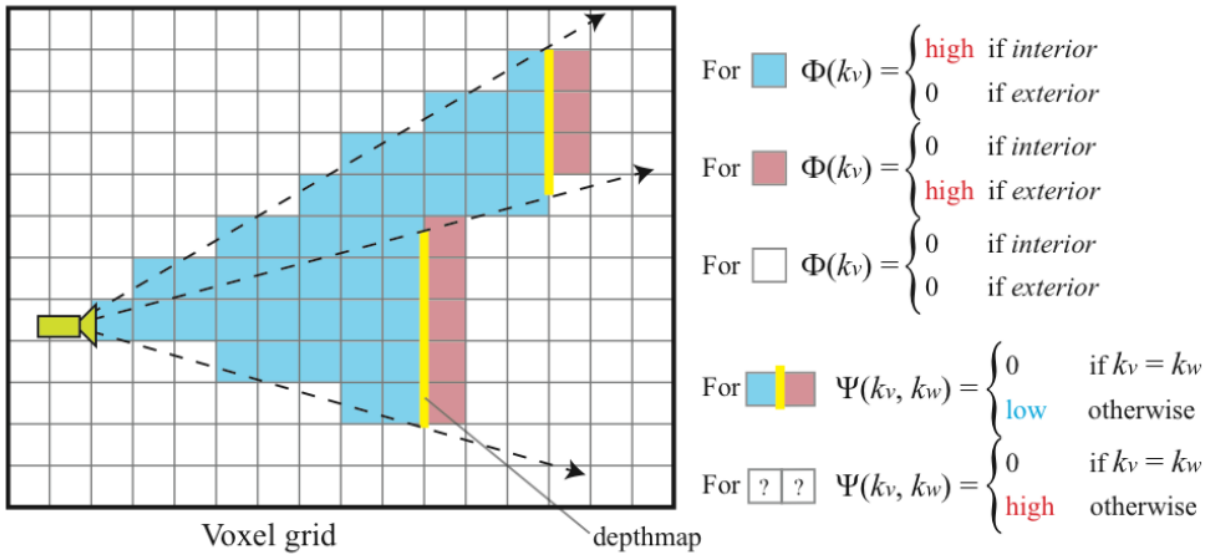


Figure 3.21: An example of how 3D MRF cost function should be set from a single depthmap.

Furukawa + Hernandez, 15, Multi-View Stereo: A tutorial

Volumetric fusion, II

- Other voxels:
  - ideally,
    - agree with original estimates;
    - agree with neighbors
  - This yields a cost function (rather like in stereo above)

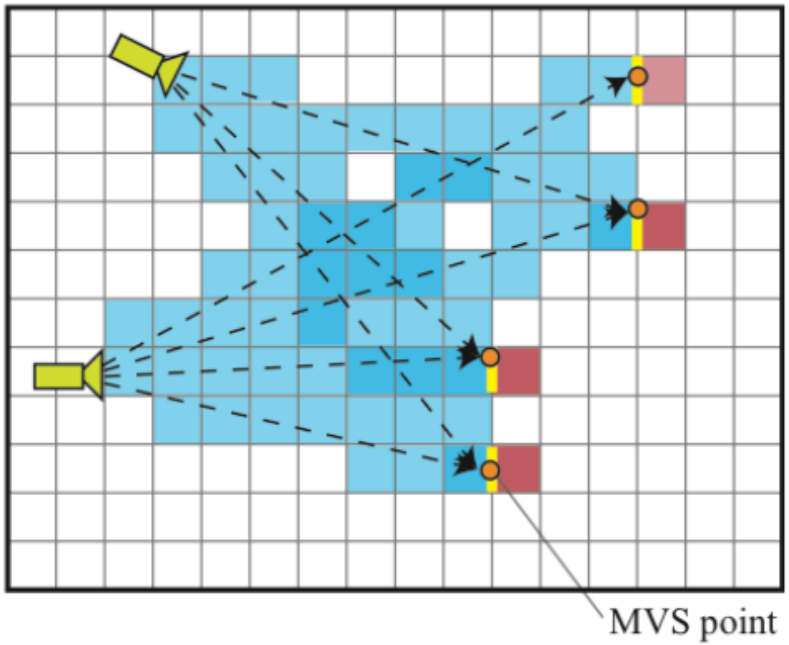


Furukawa + Hernandez, 15, Multi-View Stereo: A tutorial

Figure 3.21: An example of how 3D MRF cost function should be set from a single depthmap.

Volumetric fusion, III

- Deals with multiple cameras easily



Furukawa + Hernandez, 15, Multi-View Stereo: A tutorial

Volumetric fusion, IV

- You can use this to turn point clouds into volumes as well

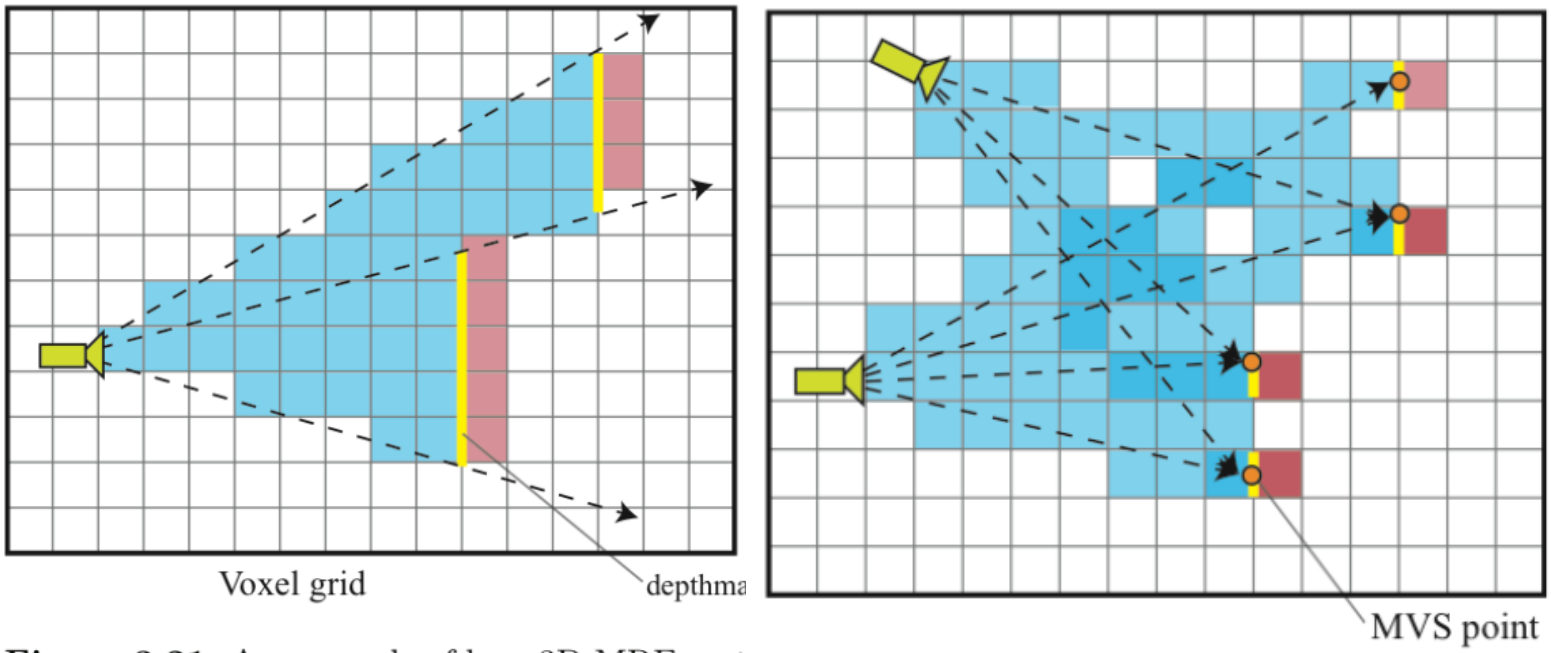


Figure 3.21: An example of how 3D MRF cost depthmap.

Furukawa + Hernandez, 15, Multi-View Stereo: A tutorial

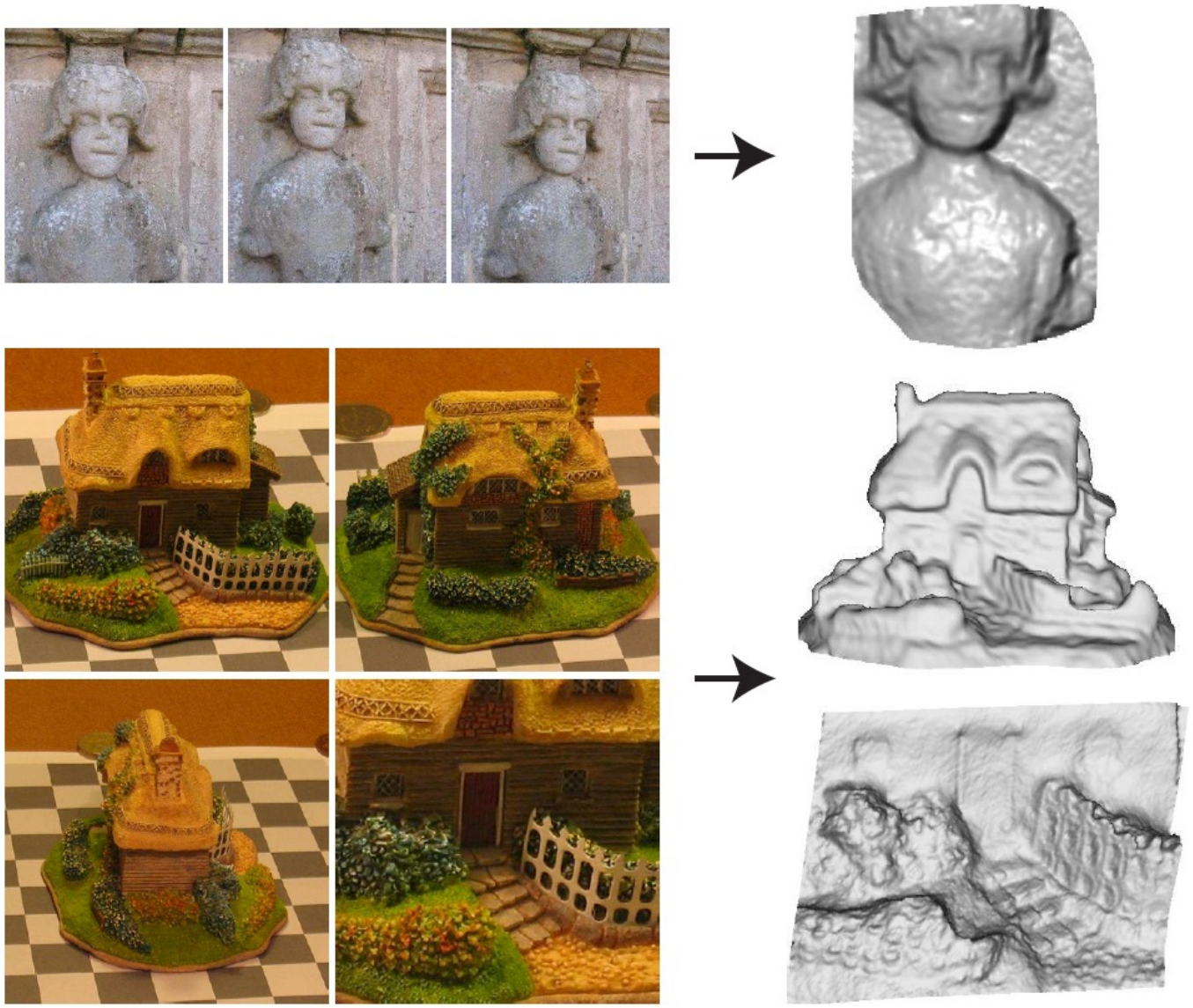


Figure 3.23: One of the earliest volume fusion techniques based on the volumetric graph-cuts by Vogiatzis, Torr and Cipolla [191]. (Figure courtesy of Vogiatzis et al.)

Furukawa + Hernandez, 15, Multi-View Stereo: A tutorial

Options

- Here:
  - Multiple depth maps
  - Point clouds
  - Voxel grids
  - Implicit functions
  - Density functions
  - Primitives
- Others
  - CSG rep’ns
  - NURBS
  - etc.

Rep’n: Implicit function

- Idea:
  - boundary between full/empty cells in space
    - like a continuous version of voxel field BUT
      - continuous field, not voxel grid
- Advantages
  - extremely rich range of field representations
  - good rendering and generalization properties
  - easy to pass to
    - mesh
- Disadvantages
  - many geometric properties hard to compute

Options

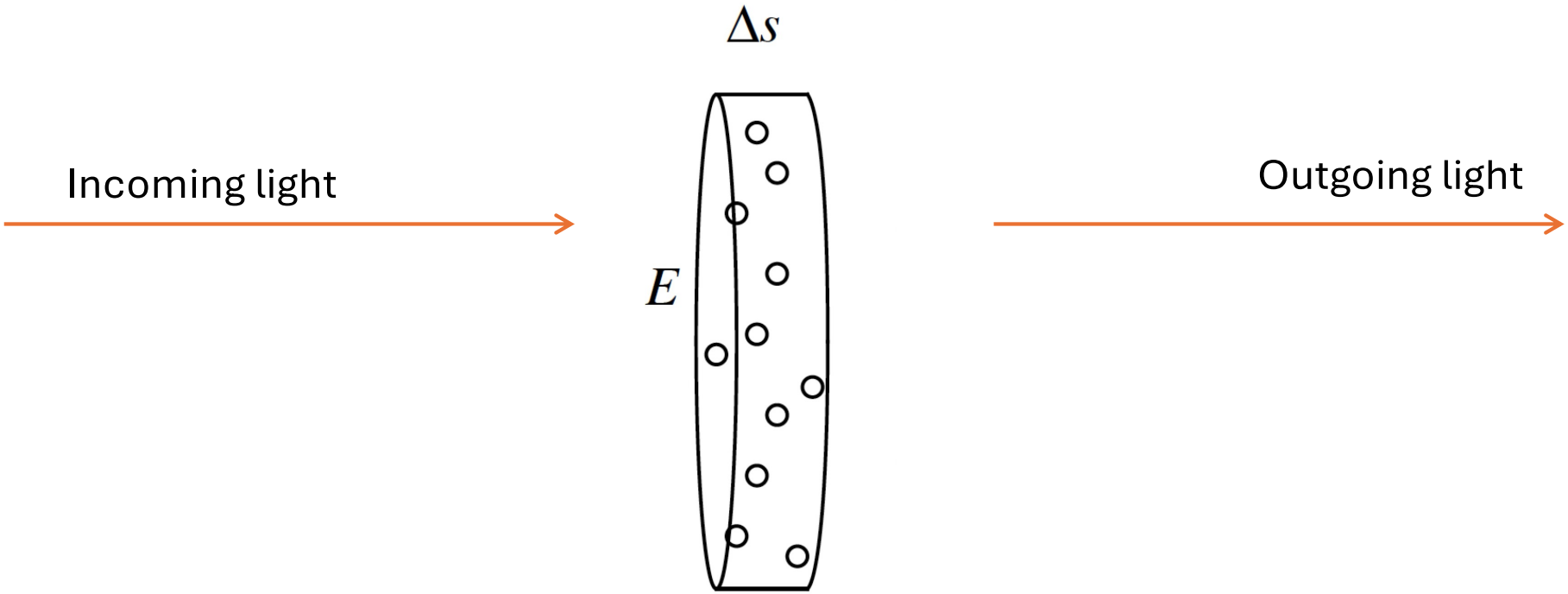
- Here:
  - Multiple depth maps
  - Point clouds
  - Voxel grids
  - Implicit functions
  - Density functions
  - Primitives
- Others
  - CSG rep’ns
  - NURBS
  - etc.

Rep’n: density fields

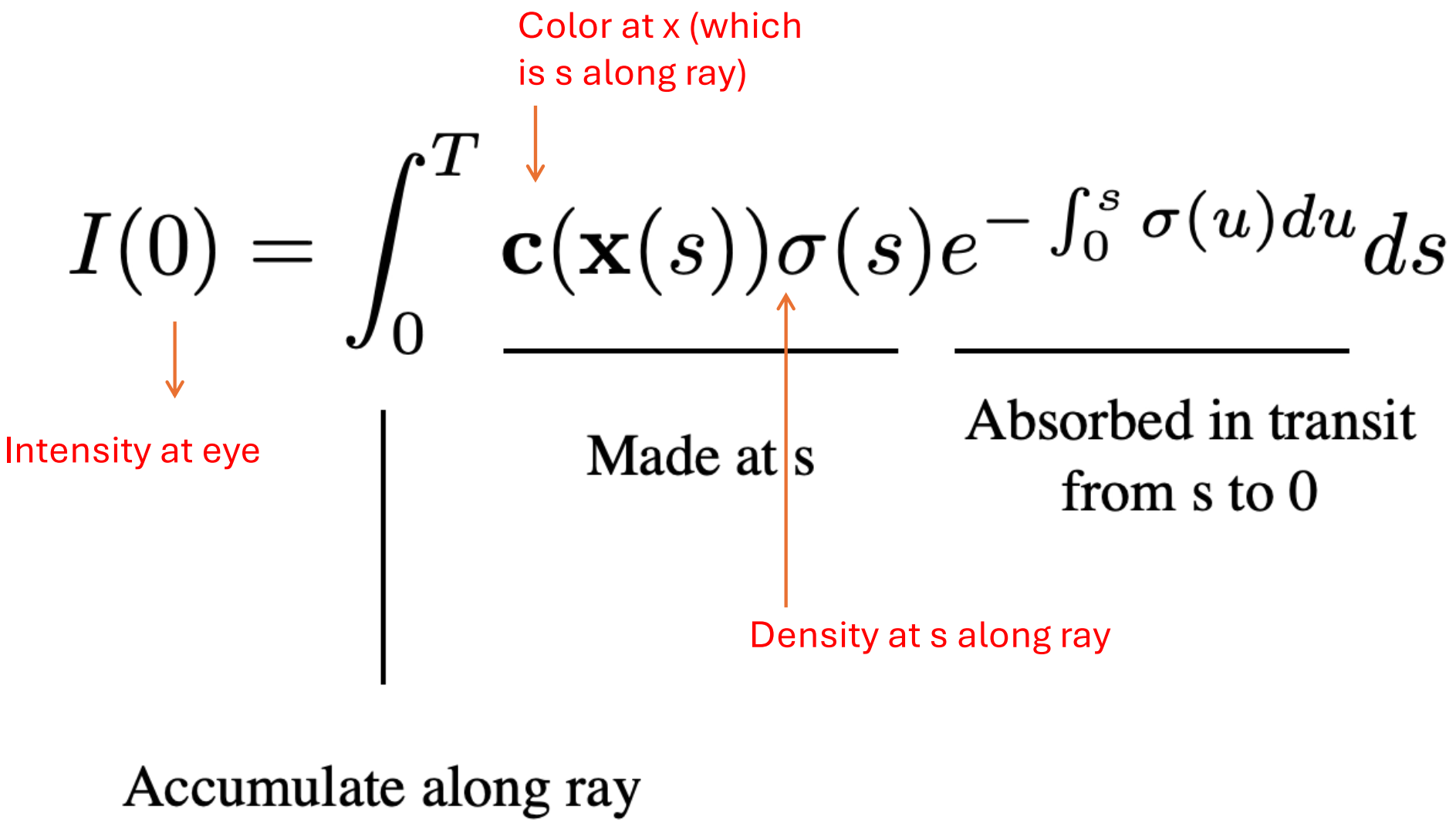
- Idea:
  - construct density field, color field in space
    - like a continuous version of voxel field BUT
      - continuous field, not voxel grid
      - density, not (full, empty)
- Advantages
  - extremely rich range of field representations
  - good rendering and generalization properties
  - easy to pass to
    - mesh, signed distance field
- Disadvantages
  - many geometric properties hard to compute

Model

- There is a field in space that
  - absorbs light
  - generates light
    - colored
- Accounting yields light at eye as a function of field



Accounting (!)


Color at x (which is s along ray)
$I(0) = \int_0^T \mathbf{c}(\mathbf{x}(s))\sigma(s)e^{-\int_0^s \sigma(u)du} ds$
Intensity at eye
Made at s
Absorbed in transit from s to 0
Density at s along ray
Accumulate along ray

Rendering

- Integration problem
  - walk back along ray from viewpoint, sampling
    - collect color at sample point
    - accumulate transmission
      - if weight is too small, stop walking
- This could be nasty…
  - variety of strategies, depending on what we know about c, sigma, eg
    - known in voxels
      - cut ray into segments (per voxel), compute integral per segment
    - parametric function
      - cut ray into uniform segments, one sample per segment
- but the integral is a differentiable function of c, sigma

Building a Nerf

- Build functional approximation to predict
  - c, sigma as functions of position, direction, parameters
- Render this object with a volume renderer
  - to make images
- Learn this object by
  - adjusting parameters (gradient descent etc)
    - so that images it produces, with renderer are the same as
      - known images
      - geometrically calibrated to one another

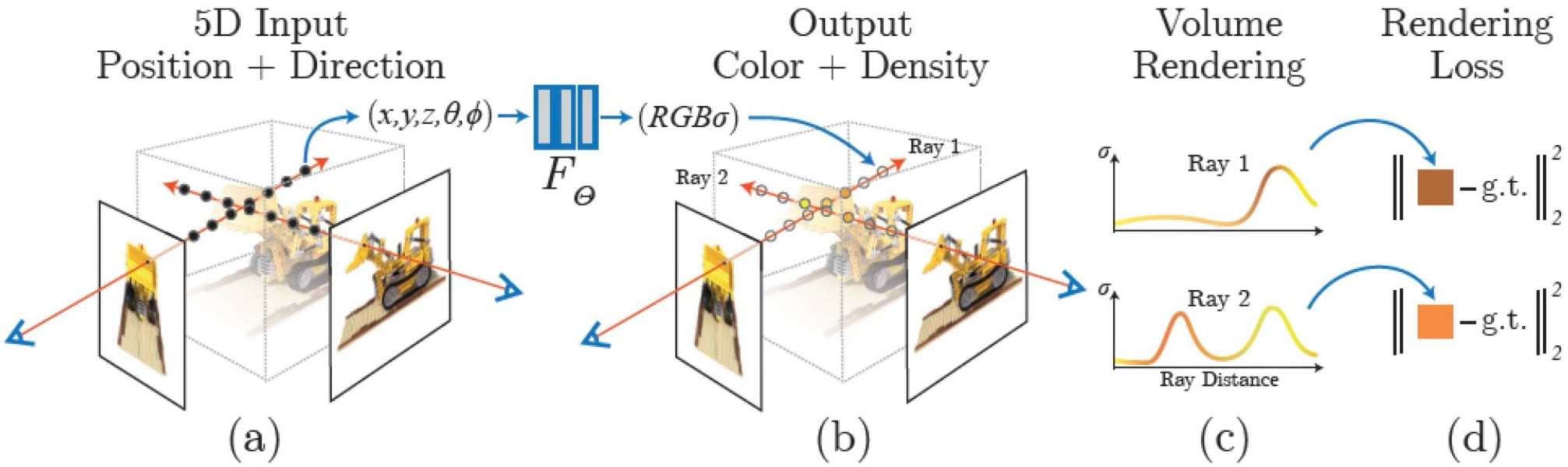
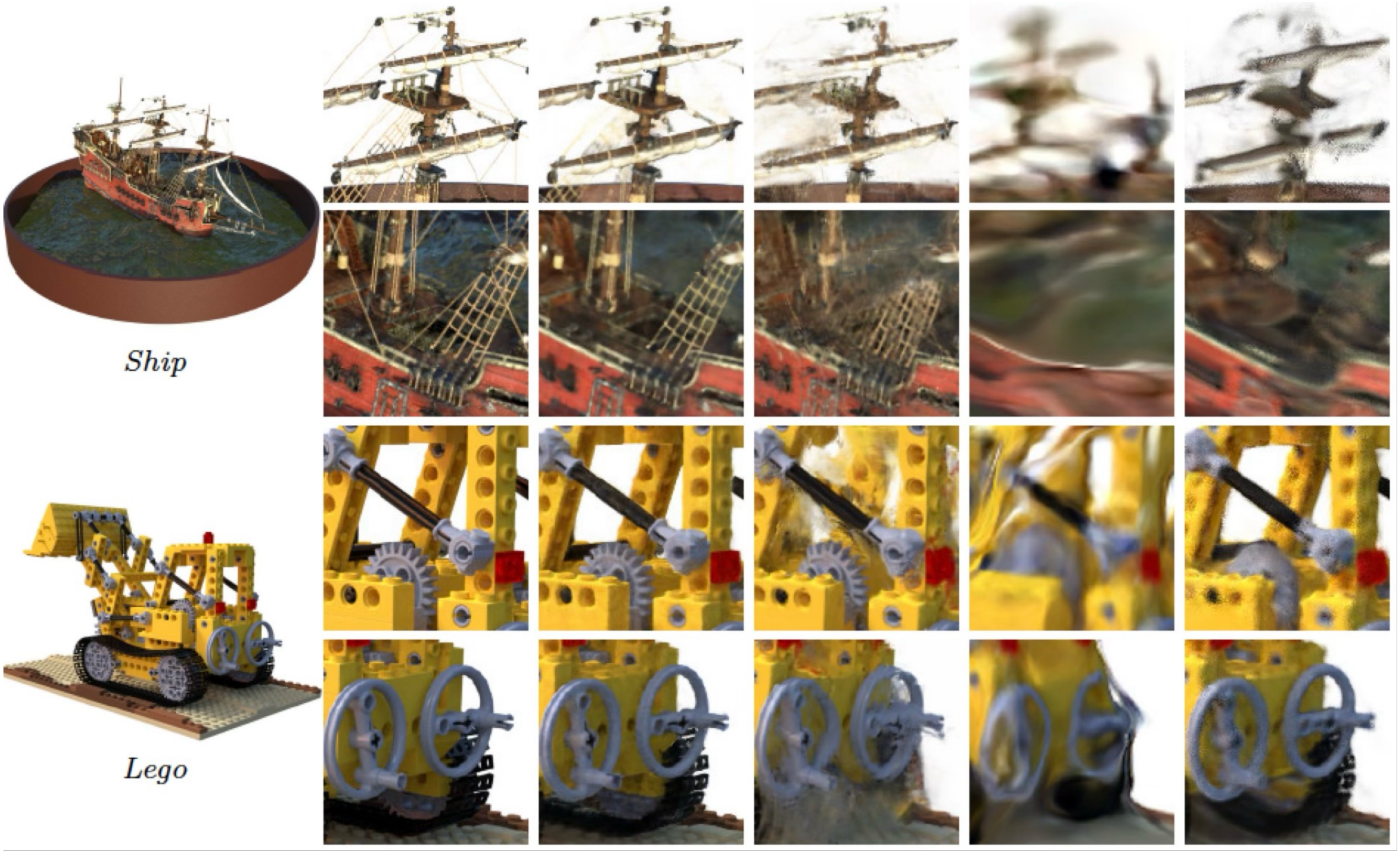


Fig. 2: An overview of our neural radiance field scene representation and differentiable rendering procedure. We synthesize images by sampling 5D coordinates (location and viewing direction) along camera rays (a), feeding those locations into an MLP to produce a color and volume density (b), and using volume rendering techniques to composite these values into an image (c). This rendering function is differentiable, so we can optimize our scene representation by minimizing the residual between synthesized and ground truth observed images (d).

Mildenhall et al, 20



Mildenhall et al, 20

Variants

- Very wide range of functional approximations
  - gaussian splats