3D Sensing and Sensor Fusion
http://cg.elte.hu/~sensing

Dmitry Chetverikov, Iván Eichhardt
csetverikov@sztaki.hu, eiiraai@inf.elte.hu

Eötvös Loránd University
Faculty of Informatics

2021 fall (2021-2022-1)

Introduction

Motivation (1/2)

(An anaglyph stereoscopic picture.)

Motivation (2/2)

Some uses of stereo depth estimates:

- View-interpolation
- View-prediction
- Segmentation
- Obtaining 3D data
- Dense SLAM
- Multiple-view reconstruction
- Tracking people, *etc.*

Single static calibrated camera 1/3

- Depth cannot be measured directly
  - at least two cameras are needed
- But surface normal vector can be estimated
  - normal vector integration $\longrightarrow$ surface
  - problems in case of drastic surface changes
- Intensity variation on smooth, weak-textured surface
  - **shape from shading**
  - intensity variation $\longrightarrow$ surface normal vector
  - less robust
  - ambiguity (multiple solutions) possible

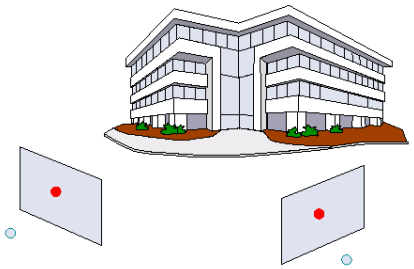
Single static calibrated camera 2/3

- Texture variation on smooth, well-textured surface
  - **shape from texture**
  - texture variation $\longrightarrow$ surface normal vector
  - less robust
- Multiple sources of illumination
  - **photometric stereo**
  - multiple measurements $\longrightarrow$ surface normal vector
  - more robust, but multiple solutions possible
  - good normal vectors, fine surface details
  - less precise positions

Single static calibrated camera 3/3

Using pr

- Spe
  -
  - $\rightarrow$
  -
- Lea
  -
  - Volumetric 3D reconstruction



Principles of stereo vision: video illustration

- Unambiguous calculation of 3D point
  - at least two different, calibrated cameras needed
  - point must be identified in camera images (correspondence)
- Procedure called **triangulation**

Approaches to stereo depth estimation

Possible approaches:

- match 'features' and interpolate
- match all pixels with windows
- use optimization:
  - iterative updating
  - dynamic programming
  - energy minimization (regularization, stochastic)
  - graph algorithms

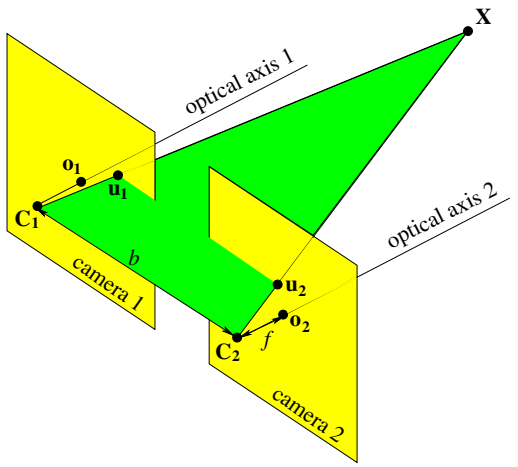
Standard stereo (1/2): Assumptions

- Two static, identical, calibrated cameras
- Parallel optical axes
- Joint image plane
- Relatively small, known distance between cameras
  - **narrow baseline**, NBL

Standard stereo (2/2): Principles of operation

- Principles of operation
  - correspondences between points of two images
  - depth calculation by triangulation
- For triangulation, we need
  - b baseline length
  - f focal length
  - d **disparity**
- Disparity: point displacement in two images

Geometry of standard stereo

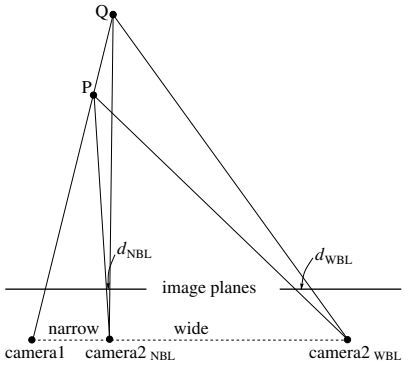


Wide-baseline stereo

- Two static, calibrated cameras
  - or two images by single camera from various veiwpoints
- Larger baseline
  - **wide baseline**, WBL
- Advantage over standard stereo
  - larger disparities
  - → more precise depth estimation
- Disadvantages
  - larger geometric distortion
  - more occlusions
  - → point correspondence more difficult

Advantage of wide baseline

- points P, Q are on same optical ray
  - → no change in camera 1
  - for simplicity
- $d_{\text{WBL}} \gg d_{\text{NBL}}$
  - → WBL yields more precise depth estimation
- d_{NBL} very small
  - few pixels
  - → rounding error
  - → 'layered' depth



Reconstruction from multiple images or video

- Three static, calibrated cameras
  - extension of standard two-camera stereo
  - certain technical advantages
- Multiple images by calibrated or uncalibrated camera
  - **multiview stereo**
- Reconstruction from one or more videos
  - redundancy
  - → autocalibrtion in case of varying camera parameters
  - dynamic 3D models
- Many calibrated snapshots or videos
  - approximate reconstruction without correspondence
  - more precise reconstruction with correspondence

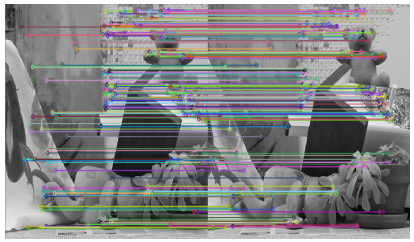
Feature-based stereo

1. Match features between images.
2. Scattered data interpolation.

Match features between images

First step: Feature-based vision approach.

- Features can be: points, lines, and regions.
- Features are matched by epipolar constraints. (*E.g.*, points along epipolar lines.)



Scattered data interpolation

Seconds step: Fill in the 'gaps'.

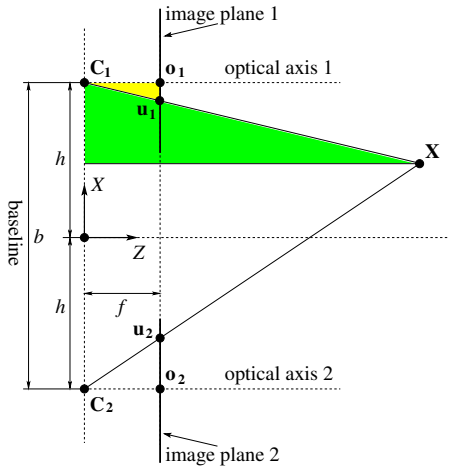
- 2D triangulation & fill
- Iterative local operations
- Fitting kernel functions
- Optimization

Geometry of standard stereo

$$\frac{u_1}{f} = \frac{h - X}{Z}$$
$$-\frac{u_2}{f} = \frac{h + X}{Z}$$
$$v_1 = v_2$$

$$Z = \frac{2hf}{u_1 - u_2} = \frac{bf}{d}$$
$$X = -\frac{b(u_1 + u_2)}{2d}$$
$$Y = \frac{bv_1}{d} = \frac{bv_2}{d}$$

$d \doteq u_1 - u_2$: disparity



Precision of depth estimation

- If $d \to 0$, $Z \to \infty$
  - disparity of distant points is small
- Relation between disparity error and depth error

$$\frac{|\Delta Z|}{Z} = \frac{|\Delta d|}{|d|}$$

  - as disparity grows, relative depth error decreases

  $\to$ depth precision increases
- Influence of baseline

$$d = \frac{bf}{Z}$$

  - for larger b same depth results in larger disparity

  $\to$ depth precision increases

  $\to$ more pixels $\longrightarrow$ growing disparity precision

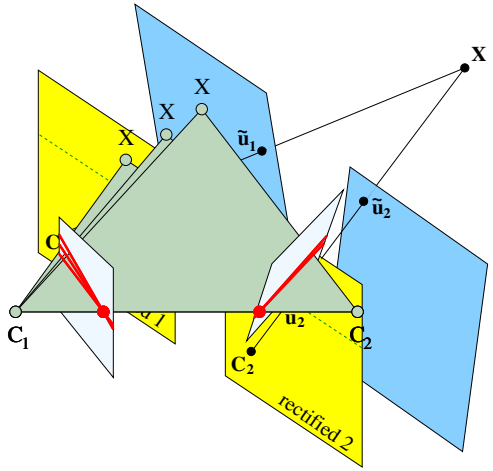
Goals and principles of rectification

- Input of rectification: **not standard** stereo image pair
- Goal of rectification: facilitate point correspondence
  - corresponding points will be in same row of two images
  - → standard stereo, 1D search
- Rectification is based on epipolar geometry
  - image pair transformed according to epipolar geometry
  - → corresponding epipolar lines go to same row
  - → epipoles tend to infinity
- Rectification needs fundamental matrix
  - → the matrix contains epipolar geometry

Rectification algorithms

- Only general principles of operation discussed here
  - rectification is complex procedure
  - **not compulsory step**, can have disadvantages
- Infinite number of plane homographies may rectify an image pair
  - → anisotropic scaling of rectified images gives rectified images
  - → valid for more complex, affine image distortions, as well
- Search for plane homography that
  - satisfies rectification conditions
  - results in minimal distortions or minimal information loss w.r.t. original images
- For calibrated cameras, much easier procedure

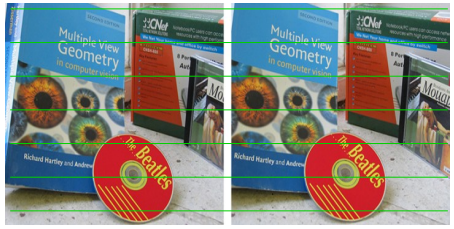
Rectification geometry



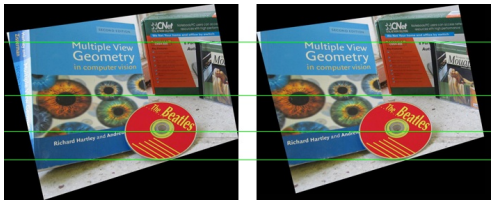
Rectification process: video

The two epipoles tend to infinity

Example of rectification



before rectification



after rectification

Advantages and practical conditions of rectification

- Results in data structure that (in principle) reduces correspondence to standard stereo
  - → numerous algorithms designed for standard stereo are applicable
- Shows the essence of epipolar geometry
- In practice, the geometry must be built with high precision
  - otherwise, rows in rectified images will be discrepant
  - → we will not find corresponding points

Disadvantages of rectification

- Wide-baseline stereo results in larger image distortion
  - rectification often further distorts images
  - → only pixel-based methods from standard stereo can be used
  - → window-based, e.g, correlation, methods cannot be used
- Size and shape of rectified image differ from those of original image
  - → finding correspondence is more difficult

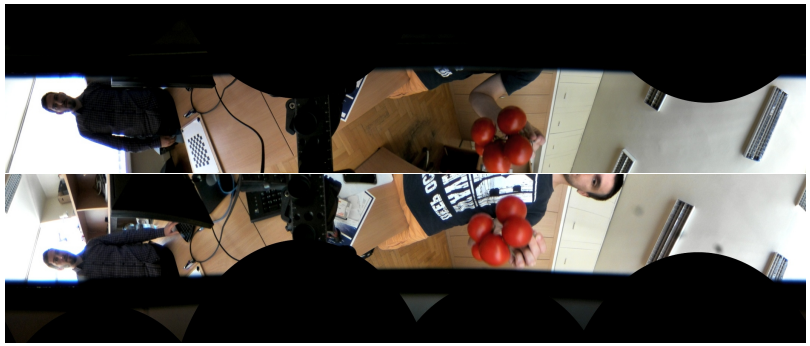
→ Not everyone agrees that rectification is necessary

- alternative opinion: search for correspondences in original images
- → taking into account epipolar constraint
- → and considering small neighbourhoods of points

Example of rectification: PAL lens (1/3)

A stereo pair shot using Panoramic Annular Lens.

Example of rectification: PAL lens (2/3)

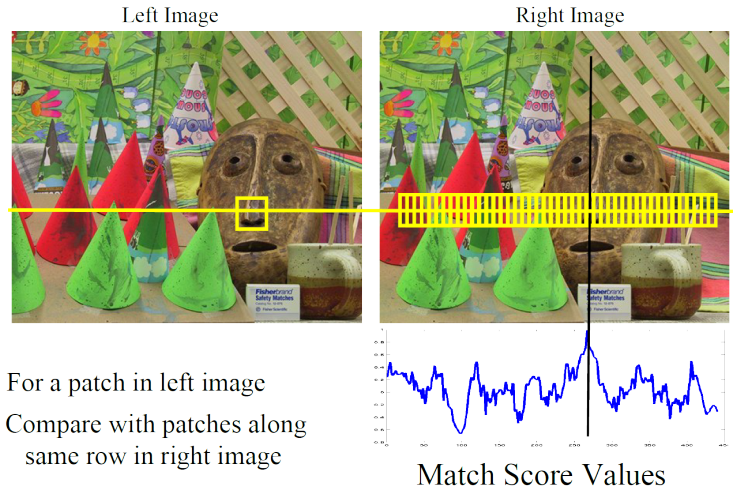


Cylindrical rectification:
a cylinder surface was used instead of a common image plane.

Example of rectification: PAL lens (3/3)

Rectification (top) compared to estimated disparity (bottom).

Matching score



Match score values

The purpose of matching score:

- Defines similarity of pixel locations.
- *E.g.*Sum of Squared Differences (SSD), Normalized SSD, (Normalized) Cross Correlation (NCC), *etc.*
- Aggregated in a window, thus, comparing *patches*.

Match score values

Let (x, y) be a location the Left image, and $(x + d, y)$ a location on the *Right image*, where d is the disparity.

*E.g.*a matching score, based on SSD, could be defined as follows:

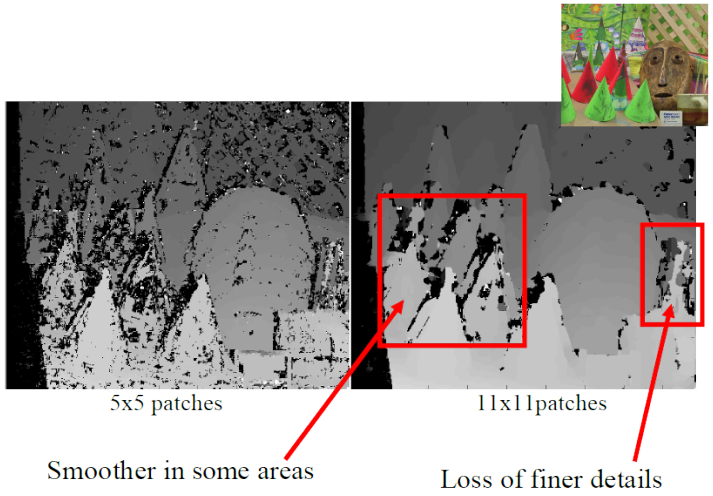
$$\mathrm{C}(x, y, d) = \sum_{(u,v)\in\Omega(x,y)} [I_L(u, v) - I_R(u + d, v)]^2 ,$$

where $\Omega(x, y)$ is a window of pixels around (x, y).

A trivial approach to obtain the disparity image $\mathbf{d}$:
solve for all pixel locations (x, y)

$$\mathbf{d}(x, y) = \arg\min_d \mathrm{C}(x, y, d)$$

The effect of patch/window size



Optimization: Formulation

The task is to minimize the following energy:

$$\mathrm{E}(\mathbf{d}) = \mathrm{E}_{data}(\mathbf{d}) + \lambda \mathrm{E}_{smoothness}(\mathbf{d})$$

- $\mathrm{E}_{data}(\mathbf{d})$ enforces data fidelity. *E.g.*:

$$\mathrm{E}_{data}(\mathbf{d}) = \sum_{(x,y)\in\mathrm{range}(\mathbf{d})} \mathrm{C}(x, y, \mathbf{d}(x, y))$$

- $\mathrm{E}_{smoothness}(\mathbf{d})$ enforces consistency in estimated parameters, *e.g.*, between neighbouring disparities in $\mathbf{d}$.
- λ is a weighting parameter.

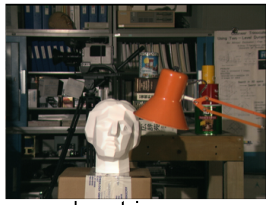
Optimization: Possible solutions

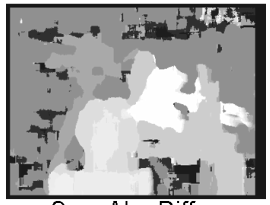
- Local steps: iteratively improve solution. Finds local optimum. Set first derivative to zero; iteratively solve and refine locally:

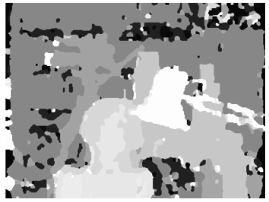
$$\nabla\mathrm{E}(\mathbf{d}) = \mathbf{0}$$

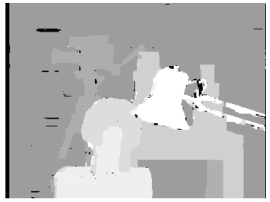
- Global optimization
  - Dynamic programming
  - Stochastic optimization
  - Variational methods, convex relaxation
  - Graph-based methods
  - *etc.*

Example comparison


Input image

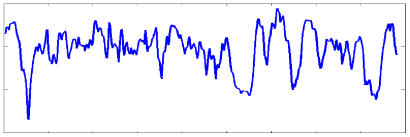

Sum Abs Diff


Mean field


Graph cuts

Why optimize?

Scanline:
matching score sampled for all disparity d, for a given pixel (x, y):



- So far, disparities of neighbouring locations are matched independently.
- This can lead to errors.

Ordering of projected objects

- In a *convex scene* the ordering of projected objects doesn't change across the left and right views.
- Ordering Constraint:

$$\text{if } x_1 < x_2 \text{ then } x_1 + \mathbf{d}(x_1, y) < x_2 + \mathbf{d}(x_2, y)$$

- I.e., consistency could be enforced between scanlines.
- **It's failure**: a *non-convex scene*:
  - Occlusion from the left?
  - Occlusion from the right?

1D optimization (1/2): Formulating energy

Let's consider row of $\mathbf{d}$ for fixed y_0.

$$\mathrm{E}^{y_0}(\mathbf{d}) = \mathrm{E}^{y_0}_{data}(\mathbf{d}) + \lambda \mathrm{E}^{y_0}_{smoothness}(\mathbf{d})$$

The data term is also slightly modified:

$$\mathrm{E}^{y_0}_{data}(\mathbf{d}) = \sum_{(x, y_0) \in \mathrm{range}(\mathbf{d})} \mathrm{C}(x, y_0, \mathbf{d}(x, y_0))$$

Next, we define $\mathrm{E}^{y_0}_{smoothness}(\mathbf{d})$ to handle occlusions.

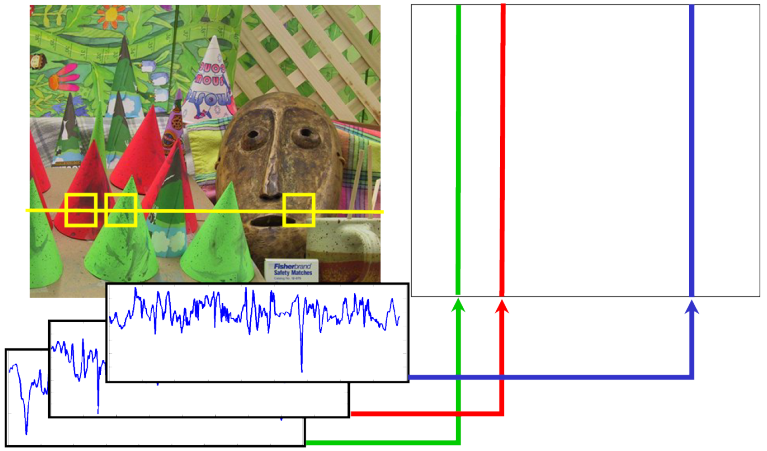
1D optimization (2/2): Formulating smoothness

Let's enforce consistency among scanlines in a row:

$$\mathrm{E}^{y_0}_{smoothness}(\mathbf{d}) = \sum_{(x,y_0),(x-1,y_0)\in \mathrm{range}(\mathbf{d})} |\mathbf{d}(x,y_0) - \mathbf{d}(x-1,y_0)|$$

This, or a similar term is often called a *discontinuity penalty*, penalizing substantial changes in disparity, weighted by λ.

Pairwise dissimilarity



Result: pairwise matching scores from left to right and back.

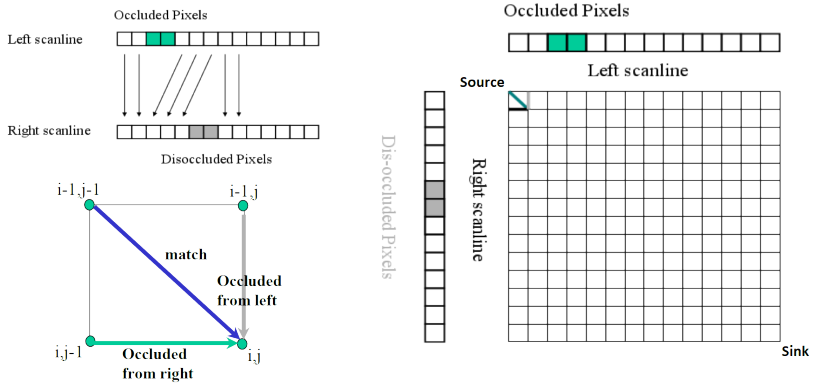
Solving 1D optimization

Observe: The *dissimilarity* image is a grid of matching scores.

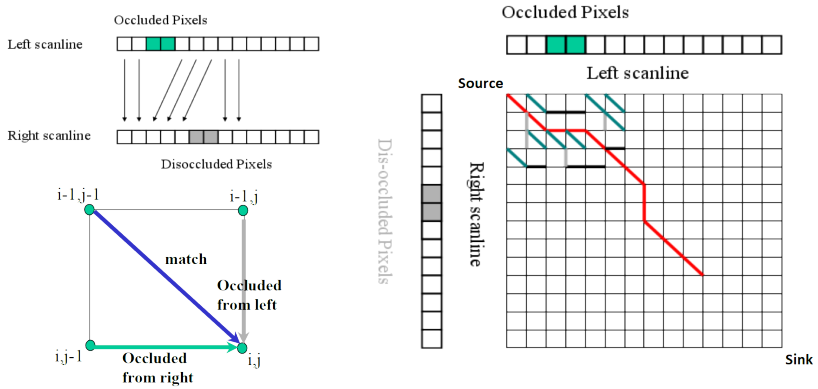
Find the optimal path:

- From *source* to *sink* in the grid
  - *source*: the beginning of the left scanline
  - *sink*: the very end of it
- Handle occlusions as edges in the path

Handling occlusion



Disparity: The optimal path



Dynamic Programming (DP) solution

Finding the optimal cost can be reformulated by evaluating the following at the *sink*.

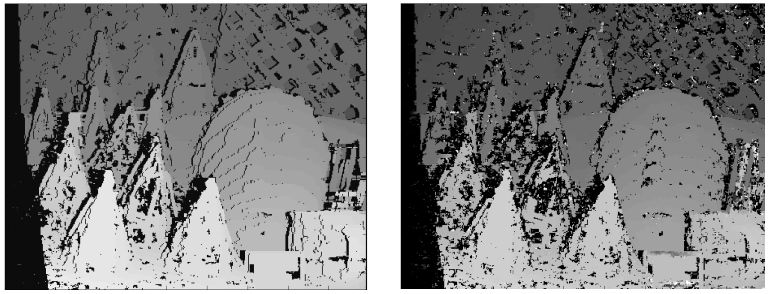
$$\mathrm{C}(i,j) = \min \begin{cases} \mathrm{C}(i-1,j-1) + \mathit{dissim}(i,j) & // \text{ match} \\ \mathrm{C}(i-1,j) + \lambda & // \text{ occluded from left} \\ \mathrm{C}(i,j-1) + \lambda & // \text{ occluded from right} \end{cases}$$

Solve it using DP – *i.e.*, find the minimum-cost path.

- Store the table of costs $\mathrm{C}(i,j)$.
- Store preceding nodes in $\mathrm{M}(i,j) \in \{\text{match}, \text{left}_{occl}, \text{right}_{occl}\}$.
- Mind the occlusions!

Reconstruct optimal path from *sink* to *source* $\rightarrow$ disparities.

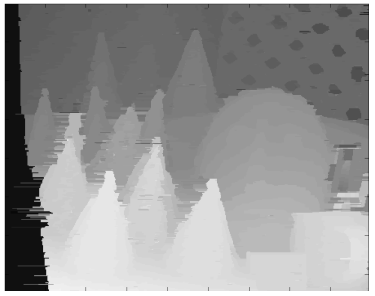
Comparison: DP vs independent pixels



using DP | naive approach

Black pixels are occluded regions.

Filling in occluded pixels

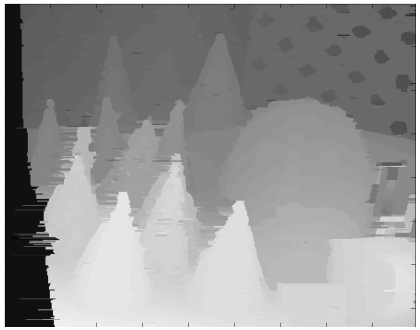


A simple trick to fill in the occluded pixels:

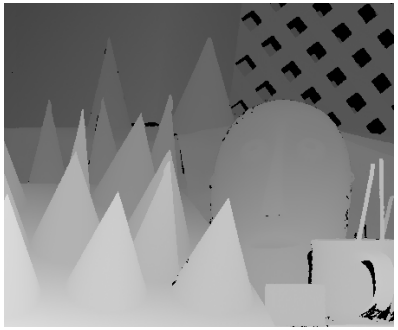
- if left occluded, look for the next valid disparity to the left.
- if right occluded, look for the next valid disparity to the right.

Comparison to Ground Truth (GT)

DP+fill



GT



Further considerations/optimizations?

Further scenes with ground truth:
`http://vision.middlebury.edu/stereo`

Bayesian formulation to derive cost functions

Given a state of a system S and a measurement M that has information on S:

$$P(S \mid M) = \frac{P(M \mid S)P(S)}{P(M)}$$

- prior probabilities: $P(S)$, $P(M)$
- posterior probability: $P(S \mid M)$
- likelihood function: $P(M \mid S)$

Bayesian formulation: ML vs MAP estimation

Maximum Likelihood estimation:

- solve $\max_S P(M \mid S)$.

To measurement M, assign S, for which the probability of M is maximal.

Maximum a Posteriori (MAP) estimation:

- solve $\max_S P(S \mid M) = \max_S \frac{P(M|S)P(S)}{P(M)}$.
- if prior $P(S)$, measurement model $P(M \mid S)$ are known.

Estimate state S which is most likely given a measurement M.

If P(S) is constant, then ML = MAP.

MAP formulation for depth cost function

$$E(\mathbf{d}) = -\log P(\mathbf{d} \mid I_L, I_R) =$$

$$= -\log \left(P\left(I_L, I_R \mid \mathbf{d}\right) P\left(\mathbf{d}\right) \right) =$$

$$= -\log P(I_L, I_R \mid \mathbf{d}) - \log P(\mathbf{d}).$$

The given cost then translates to one with data and smoothness terms. Then the task is to minimize $E(\mathbf{d})$.

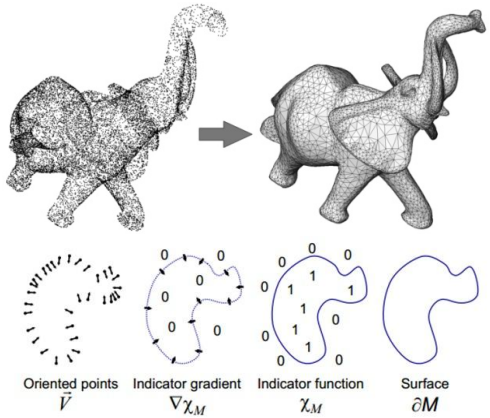
Multiple view stereo

Input: multiple images (+ maybe other data),
Output: volume / surface.

Ways to obtain that result:

- Estimate sparse reconstruction (*e.g.*, point cloud) then 'interpolate'.
- Merge multiple stereo images.
- Assume initial surface/volume, then perform optimization.

Poisson surface reconstruction



Kazhdan abd Hoppe '*Screened poisson surface reconstruction*' ToG (2013).

Multiple-View Stereo (MVS) algorithms

- Most algorithms start with an initial mesh (*e.g.*, Poisson reconstruction).
- Then obtain fine details and texture.

Some software performing MVS:

- OpenMVS – `cdcseacave.github.io/openMVS`
- 3D Novator – `3dnovator.com`
- *etc.*

Some nice 3D models: `https://sketchfab.com/cdc`

An example output of 3D Novator

Readings

Readings / video lectures:

- Multiple View Geometry (Prof. Daniel Cremers)
  https://www.youtube.com/watch?v=RDkwklFGMfo&list=PLTBdjV_4f-EJn6udZ34tht9EVIW7lbeo4
- Variational Methods for Computer Vision (Prof. Daniel Cremers)
  https://www.youtube.com/watch?v=fpw26tpHGr8&list=PLTBdjV_4f-EJ7A2iIH5L5ztqqrWYjP2RI