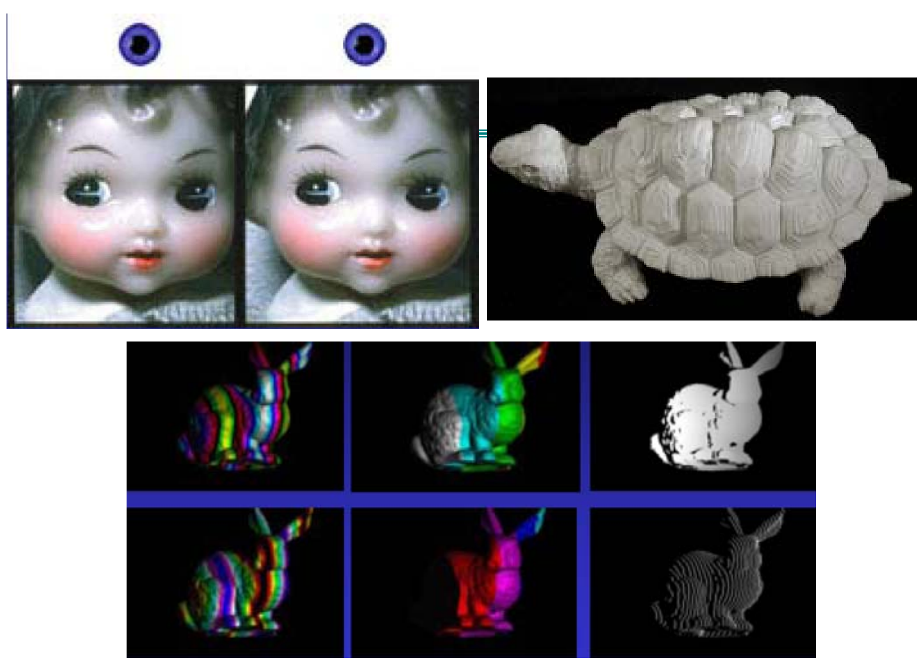
3D Surface Data Acquisition

For class CSE612

Presented by
Song Zhang

Department of Mechanical Engineering
SUNY at Stony Brook
Oct. 16, 2003



Picture from: Rusinkiewicz et al 02 and Luiz Velho – IMPA 02

Outline

- **Introduction**
  - Current 3D SDA methods and their applications
- **Stereovision**
  - Principle of stereovision
  - Problems with stereovision and solutions
- **Structured light**
  - Introduction of methods on 3D SDA based on structured light
  - Structured light - phase-shifting

Applications of 3D SDA

- **Industrial inspections**
  - Automotive, aircraft, shipbuilding, etc.
- **Reverse engineering**
  - Rapid prototyping, 3D fax & copy machines, etc.
- **Large infrastructure inspection**
  - Buildings, roads, bridges, tunnels, etc.
- **Robotic Vision**
  - Autonomous vehicle guidance, object recognition, etc.
- **3D microscopy**
  - Surface inspection, material testing, etc.
- **Medical imaging and diagnosis**
  - Anthropometry, plastic surgery, lung function study, etc.

3D SDA techniques

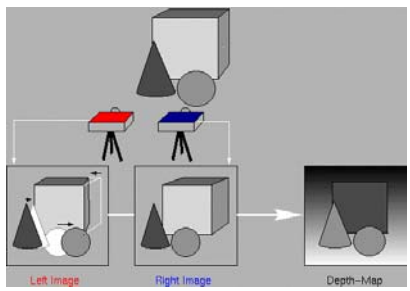
	Passive	Active
Monocular	shape-from-shading, texture, etc.	time-of-flight
Binocular	stereo	laser ranging, structure lighting
Multiple frames	shape-from-motion	computer tomography

Stereovision

This part is based on 3D vision lecture notes at:
http://www-edlab.cs.umass.edu/cs570/

Definition

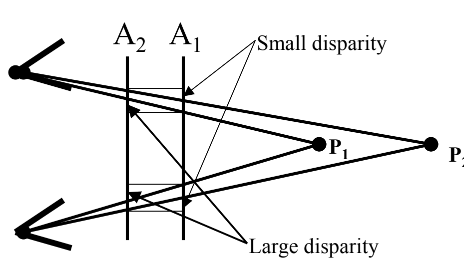
- **What is stereovision?**
  - The view from multiple cameras (or one moving camera) is used to compute the range or distance from the cameras. Binocular vision is used when two cameras are used.



Picture from: http://axon.physik.uni-bremen.de

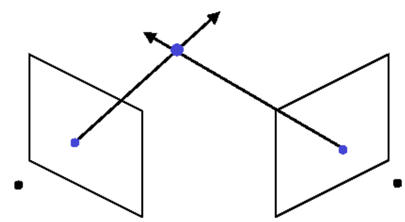
Principle

- **Points near an observer appear to move a lot when the viewpoint shifts, while points far away seem to move less.**
- **Knowing precise relationships between cameras(positions, focal length etc) depth information can be achieved.**



Basics

- **The stereo problem**
  – Reconstruct scene geometry from two or more *calibrated* images



- **Basic principle: triangulation**
  – Gives reconstruction as intersection of two rays
  – Requires *point correspondence*
    ➢ This is the hard part

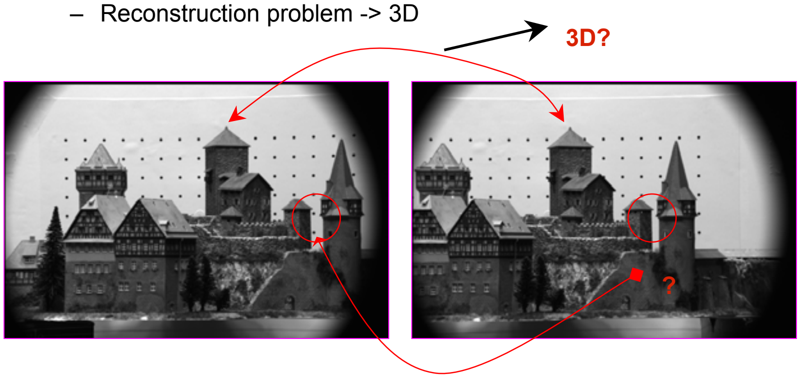
Problems

- **Problem**
  – Infer 3D structure of a scene from two or more images taken from different viewpoints

- **Two primary sub-problems**
  – Correspondence problem (stereo match) -> disparity map
    ➢ Similarity instead of identity
    ➢ Occlusion problem: some parts of the scene are visible in one eye only
  – Reconstruction problem -> 3D
    ➢ What we need to know about the cameras' parameters
    ➢ Often a stereo calibration problem

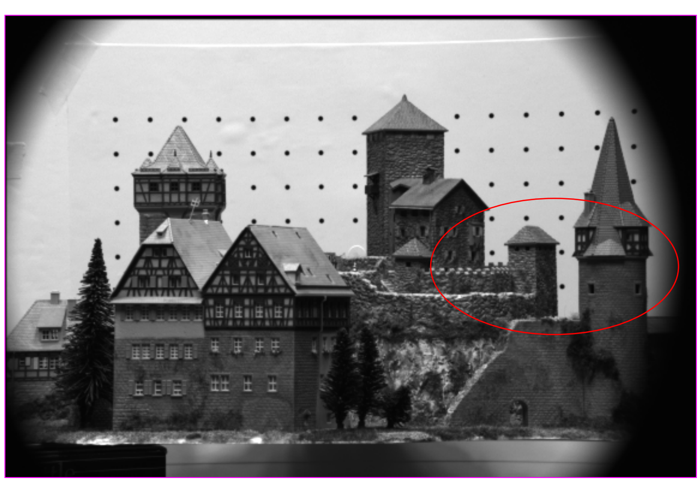
Problems(cont'd)

- **Problems**
  - Correspondence problem (stereo match) -> disparity map
  - Reconstruction problem -> 3D

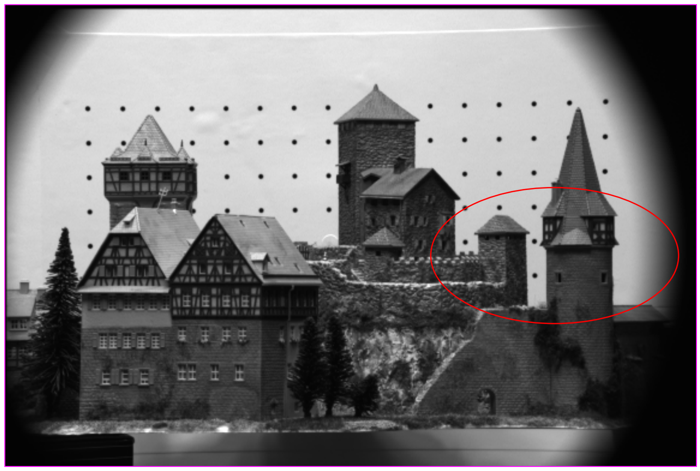


CMU CIL Stereo Dataset : Castle sequence
http://www-2.cs.cmu.edu/afs/cs/project/cil/ftp/html/cil-ster.html

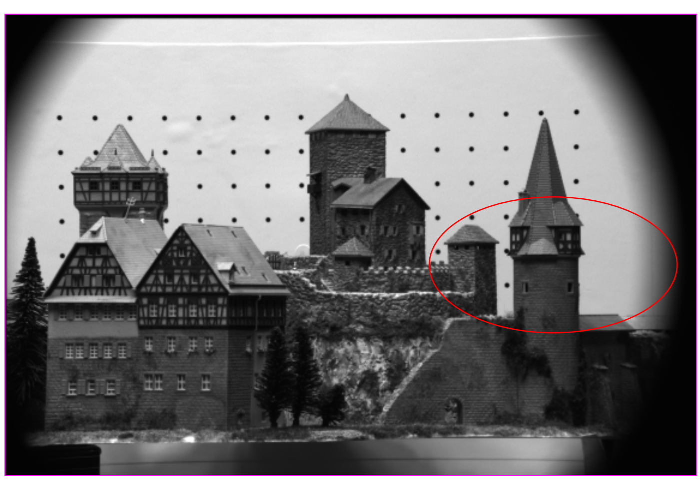
Images



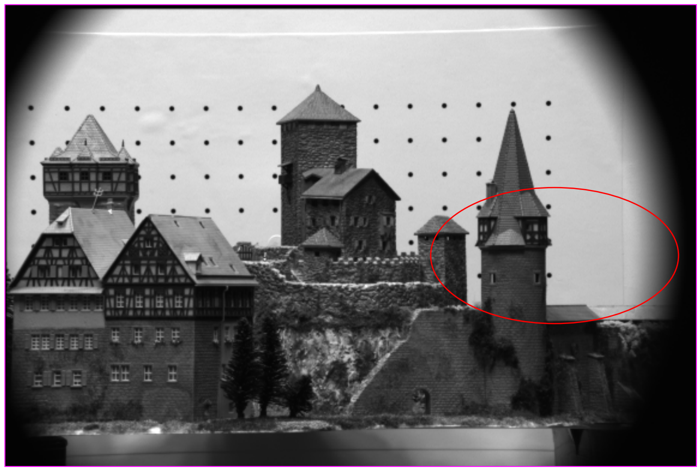
Images



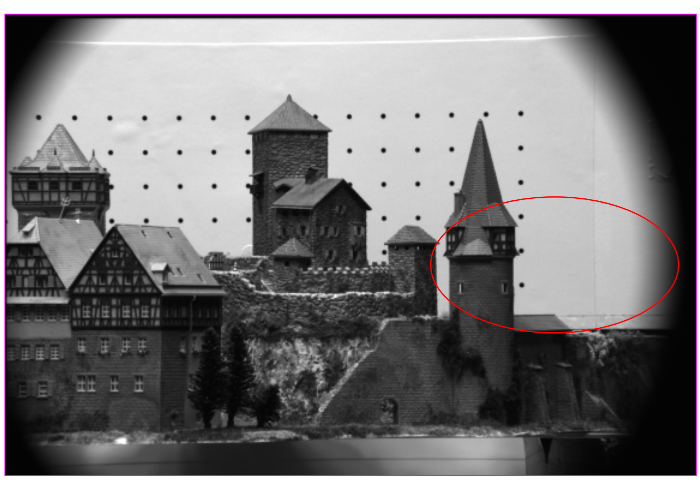
Images



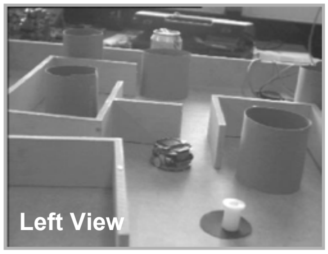
Images

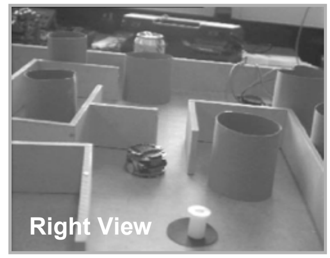


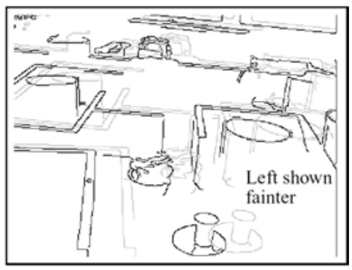
Images



Stereo disparity

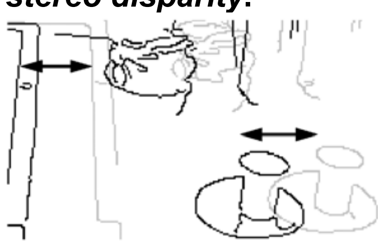






Stereo disparity(cont'd)

- **The separation between two matching objects is called the *stereo disparity*.**

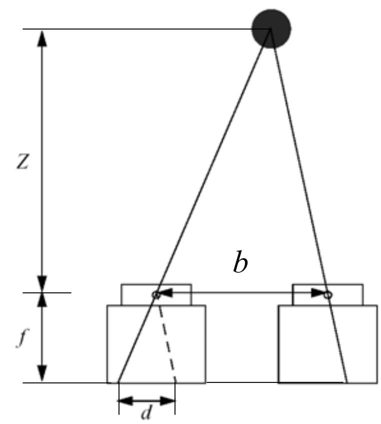


- **Disparity is measured in pixels and can be positive or negative (conventions differ). It will vary across the image.**

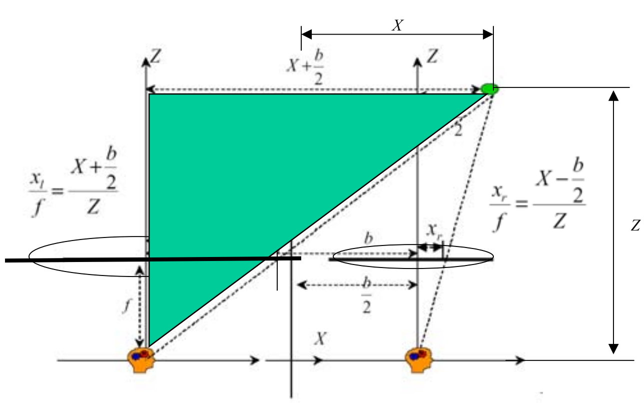
Depth extraction (1)

- **If the cameras are pointing in the same direction, the geometry is simple.**
  - *b* is the *baseline* of the camera system,
  - *Z* is the *depth* of the object,
  - *d* is the disparity (left *x* minus right *x*) and
  - *f* is the focal length of the cameras.
- **Then the unknown depth is given by**

$$Z = \frac{f\,\mathbf{b}}{d}$$



Depth extraction (2)



$$\frac{x_l - x_r}{f} = \frac{b}{Z}$$

$$Z = \frac{f\,b}{(x_l - x_r)}$$

Depth extraction (2) (cont'd)

- **Disparity is inversely proportional to depth—so stereo is most accurate for close objects.**
- **Once we have found depth, the other coordinates in 3-D follow easily — e.g. taking either one of the images:**

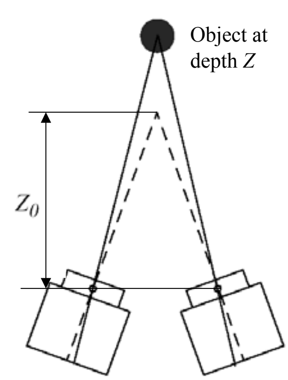
$$X = \frac{xZ}{f} = \frac{xB}{d}$$

where *x* is the image coordinate, and likewise for *y*.

Depth extraction (3)

- **This is the more realistic case.**
- **The depth at which the cameras converge, Z_0, is the depth at which objects have zero disparity.**
  - Finding Z_0 is part of stereo *calibration*.
- **Closer objects have convergent disparity (numerically positive) and further objects have divergent disparity (numerically negative).**

$$Z \approx \frac{f\,B}{d + \frac{fB}{Z_0}}$$



Correspondence problem

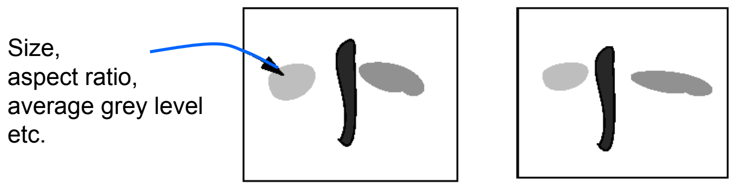
- **To measure disparity, we first have to find corresponding points in the two images.**
- **This turns out not to be easy.**
- **Our own visual systems can match at a low level, as shown by *random-dot stereograms*, in which the individual images have no structure above pixel scale, but which when fused show a clear 3-D shape.**

Stereo matching

- **Stereo matchers need to start from some assumptions.**
  - Corresponding image regions are similar.
  - A point in one image may only match a single point in the other image. (?)
  - If two matched features are close together in the images, then in most cases their disparities will be similar, because the environment is made of continuous surfaces separated by boundaries.
- **Many matching methods exist. The basic distinction is between**
  - feature-based methods which start from image structure extracted by preprocessing; and
  - correlation-based methods which start from individual grey-levels.

Feature based method

- **Extract feature descriptions: regions, lines, ….**



- **Pick a feature in the left image.**
- **Take each feature in the right image in turn (or just those close to the epipolar line), and measure how different it is from the original feature:**

$$S = \frac{1}{w_0(l_r - l_l)^2 + w_1(\theta_r - \theta_l)^2 + w_2(g_r - g_l)^2 + \dots}$$

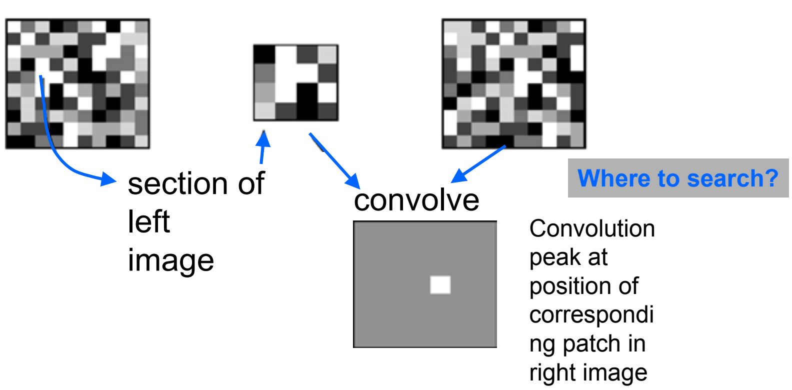
Feature based method(cont'd)

- **It is possible to use very simple features (just points, in effect) if the constraint that the disparity should vary smoothly is taken into account.**
- **Feature-based methods give a *sparse* set of disparities — disparities are only found at feature positions.**

Correlation-based method

- **Take a small patch of the left image as a mask and convolve it with the part of the right image close to the epipolar line.**
  - Over an area
- **The peak of the convolution output gives the position of the matching area of the right image, and hence the disparity of the best match.**

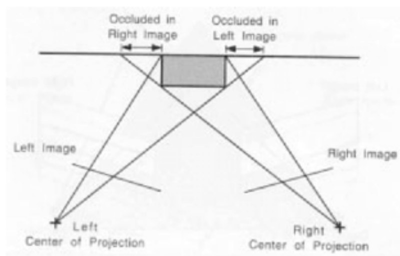
Correlation-based method(cont'd)



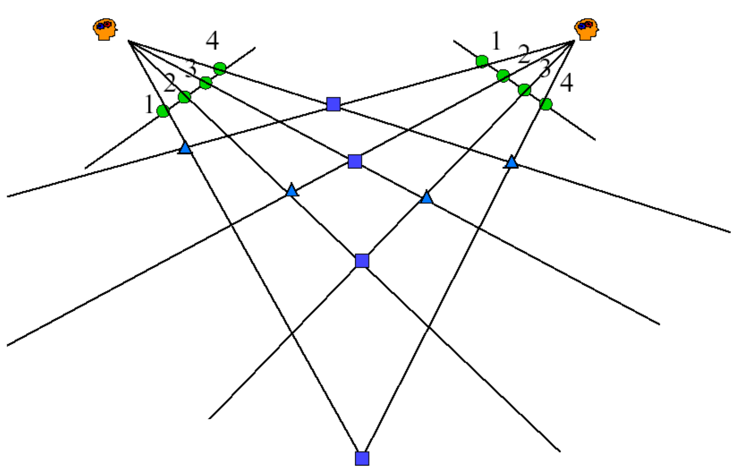
- **Convolution-based methods can give a *dense* set of disparities — disparities are found for every pixel.**
- **These methods can be very computationally intensive, but can be done efficiently on parallel hardware.**

Correspondence difficulties

- **Why is the correspondence problem difficult?**
  - Some points in each image will have no corresponding points in the other image.
    - (1) the cameras might have different fields of view.
    - (2) due to occlusion.
- **A stereo system must be able to determine the image parts that should not be matched.**

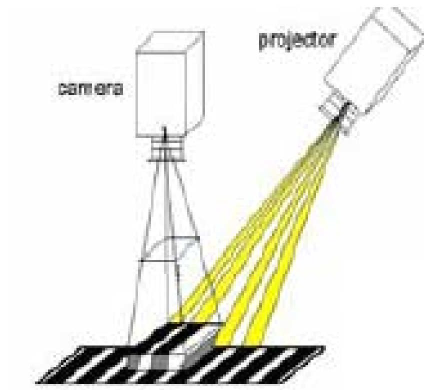


Correspondence difficulties(cont'd)



Solution to correspondence difficulties

- Replacing one camera of the stereovision by a projector who projects coded structured light pattern onto the measuring surface.



Structured Light

Principle

- Based on projecting a light pattern and imaging the illuminated scene from one or more points of view.
- The projected scene contains certain coded structured pattern that allows a set of pixels being easily distinguishable by means of a local coding strategy. (Correspondence problem is not a problem at all here!)
- 3D scene can be reconstructed from the decoded image points by applying triangulation.

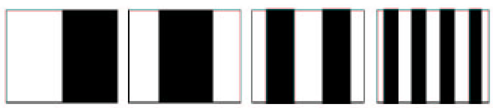
Coded structured light methods

- **Time – multiplexing**
  - A set of patterns are successively projected onto the measuring surface, codeword for a given pixel is formed by a sequence of patterns.

http://eia.udg.es/~jpages/

Time – multiplexing

- **Binary** **(Posdamer et al. ‘81, Inokuchi et al. ‘84, Minou et al. ‘81, Trobina, ‘95, Valkenburg and McIvor ‘98, Skocaj and Leonardis ‘00, Rocchini et al. ‘01)**
  - Only two illumination levels are commonly used, which are coded as 0 and 1. Every pixel of the pattern has its own codeword formed by the sequence of 0s and 1s corresponding to its value in every projected projected pattern.
    - ➢ Easy to segment the image patterns
    - ➢ Need a large number of patterns to be projected



Time – multiplexing

- **N-ary codes** **(Caspi et al. ‘98, Horn and Kiryati ‘99)**
  - Reducing the number of projected patterns by increasing the intensity levels used to encode the stripes.
    - ➢ Multilevel gray code based on color.
    - ➢ Multi grey levels instead of binary alphabet.

6 different colors

Mouth035

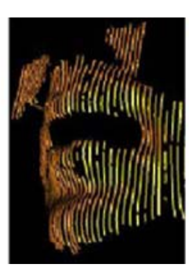
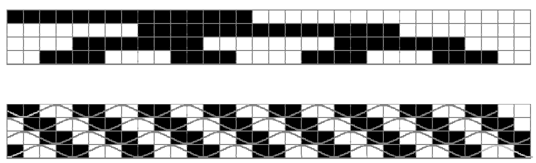


 Image from: Tim Monks, university of southampton

Time – multiplexing

- **Grey code + phase shifting** **(Bergmann '95, Sansoni et al. '97, Wiora, '00, Gühring '01)**
  - Grey code: (easy codification, low resolution)
  - Phase shifting: (high resolution, neighborhood ambiguity)
  - Gray code + phase shifting (robust codification, unambiguity and high resolution, but increase the number of projecting patterns).



Pattern Sequence: combination of Graycode (top) and phase shift (bottom).

http://eia.udg.es/~jpages/ Picture from: Gühring '01

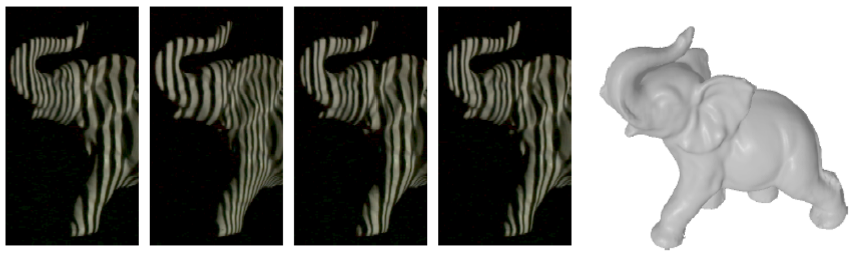
Time – multiplexing

- **Hybrid methods** **(Kosuke Sato, '96, Hall-Holt and Rusinkiewicz '01)**
  - Stripe boundary codes.
    - Divide 4 patterns into a total of 111 vertical stripes that were painted in white and black. Codification is located at the boundaries of each pair of stripes. The codeword of each boundary is formed by 8 bits. Every gives 2 of these bits, representing the value of the bounding stripes. (The author said they could do in real time)

http://eia.udg.es/~jpages/ picture from: Hall-Holt et al '01

Time – multiplexing

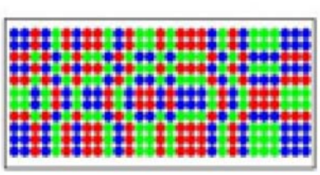
- **Stripe boundary codes (Hall-Holt and Rusinkiewicz '01).**



picture from: Hall-Holt et al '01

Coded structured light methods (cont'd)

- **Spatial Neighborhood**
  - Project a certain kind of spatial pattern so that a set of neighborhood points appears in the pattern only ones. Then the codeword that labels a certain point of the pattern is obtained from a neighborhood of the point around it.



Griffin - Narasimhan - Yee 1992

- Mathematical study to obtain the largest codification matrix from a fixed number of colours.
- Dot position coded by the colour of its four neighbours.

Both axis coded / Static / Colour / Absolute

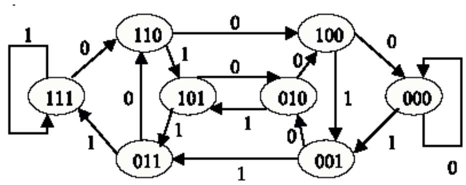
Davies '96 : Re-implementation.

Spatial Neighborhood

- **De bruijn sequence** **(Hügli and Maitre, 89, Monks et al. 93, Vuylsteke and Oosterlinck 90, Salvi et al. 98, Lavoie et al. 1999, Zhang et al. 02)**
  - De bruijn sequence of order m over an alphabet of n symbols is a circular string of length n^m that contains each substring of length m exactly once. This sort of sequence can be obtained by searching *Eulerian circuits* or *Hamiltonian circuits* over different kinds of *De Bruijn graphs.*

m = 4

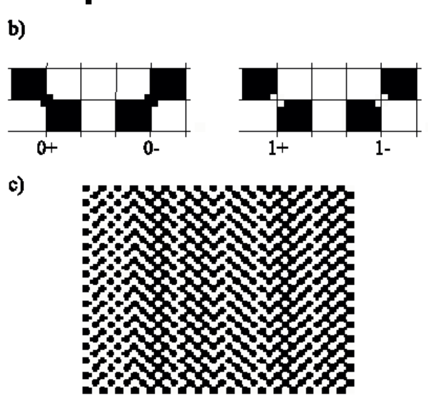
S = 1000010111101001



http://eia.udg.es/~jpages/

Spatial Neighborhood

- **De bruijn sequence**

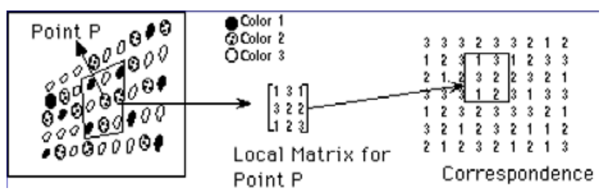


b) primitives proposed by Vuylsteke and Ooesterlick to represent the binary values; c) the resulting pattern using the primitives.

http://eia.udg.es/~jpages/

Spatial Neighborhood

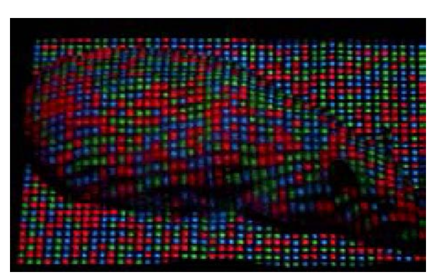
- **M-arrays** **(Morita et al. 88, Petriu et al. 92, Kiyasu et. al. 95 Spoelder et al. 00, Griffin and Yee 92, Davies et. al. 98, Morano et al. 98)**
  - In a r x v matrix M, if each different n x m submatrix except (0’s) only appears once. Then the matrix is M-array.



http://www.mri.jhu.edu/~cozturk/sl.html

Spatial Neighborhood

- **M-arrays** **(Morita et al. 88, Petriu et al. 92, Kiyasu et. al. 95 Spoelder et al. 00, Griffin and Yee 92, Davies et. al. 98, Morano et al. 98)**
  - In a r x v matrix M, if each different n x m submatrix except (0’s) only appears once. Then the matrix is M-array.



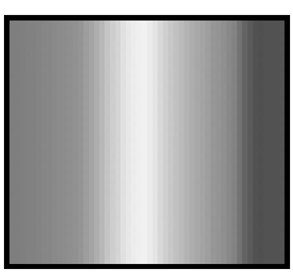
http://www.mri.jhu.edu/~cozturk/sl.html

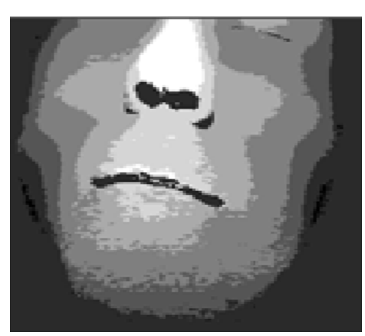
Coded structured light methods (cont'd)

- **Direct coding**
  - Creating a pattern so that every pixel can be labeled by the information represented on it. Thus, the entire codeword for a given point is contained in a unique pixel.
    - ➢ Sensitive to noise and color
    - ➢ Not suit for dynamic scenes (usually need to take one or more reference images) ?

Direct coding

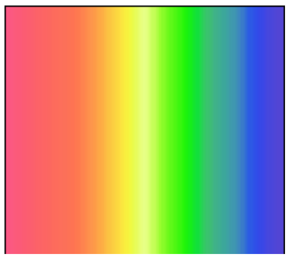
- **Grey levels** **(Carrihill and Hummel 1985, Chazan and Kiryati 1995, Hung 1993).**
  - Codification based on grey levels.: a spectrum of grey levels is used to encode the points of the pattern.





Direct coding

- **Color** (Tajima and Iwakawa 1990, Smutny and Pajdla, Geng 1996, Wust and Capson 1991, Tatsuo Sato 1999)
  - Codification based on color: use a large spectrum of colors.

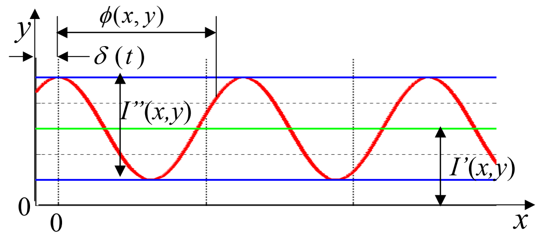


Rainbow pattern

Structured light - phase shifting

Phase-shifting algorithm

- **Fringe pattern**



$$I(x,y,t) = I'(x,y) + I''(x,y)\cos[\phi(x,y) + \delta(t)]$$

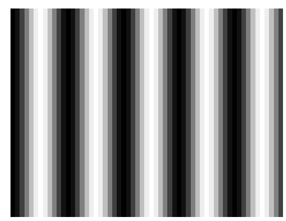
Where:

$I'(x,y)$: Signal bias

$I''(x,y)$: Intensity modulation

$\phi(x,y)$: Wavefront phase difference

$\delta(t)$: Time-varying phase shift



Phase-shifting algorithm (cont'd)

- **3-step method**

$$\delta_i = -\alpha, 0, \alpha$$

$i = -1, 0, 1$

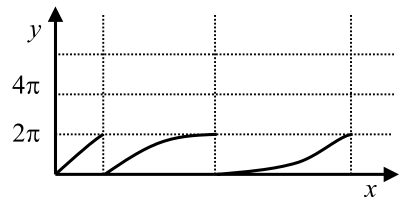
$$I_1 = I'(x,y) + I''(x,y)\cos[\phi(x,y) - \alpha]$$
$$I_2 = I'(x,y) + I''(x,y)\cos[\phi(x,y)]$$
$$I_3 = I'(x,y) + I''(x,y)\cos[\phi(x,y) + \alpha]$$

Phase-shifting algorithm (cont'd)

- **Phase wrapping**

$$\alpha = 120^0$$

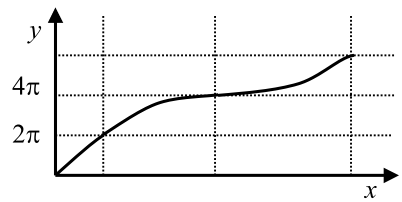
$$\phi(x,y) = \arctan\left(\sqrt{3}\,\frac{I_1 - I_3}{2I_2 - I_1 - I_3}\right)$$



1-D phase wrapping

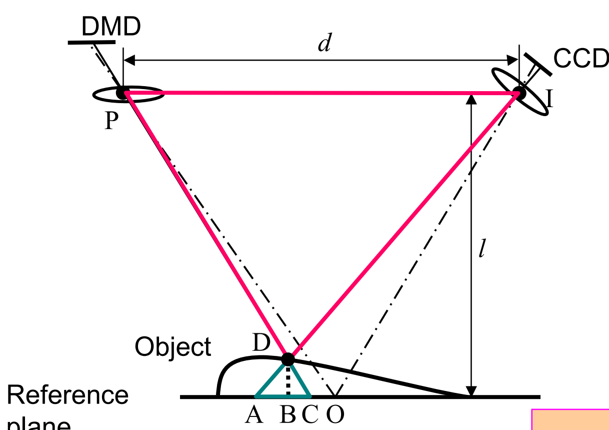
- **Phase unwrapping**

  - Remove the 2π discontinuity by adding or subtracting multiples of 2π



1-D phase unwrapping

Depth Z extraction



$$\Delta PDI \cong \Delta ADC$$

$$\overline{DB} = \frac{l - \overline{DB}}{d}\,\overline{AC}$$

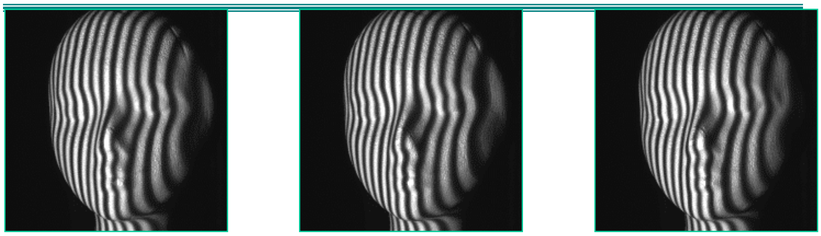
$$\approx \frac{l}{d}\,\overline{AC} = K\phi_{AC}$$

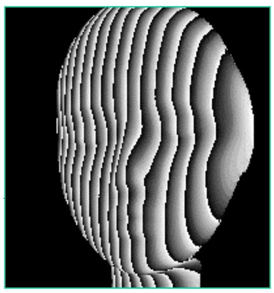
$$= K\left(\phi_A - \phi_C\right)$$

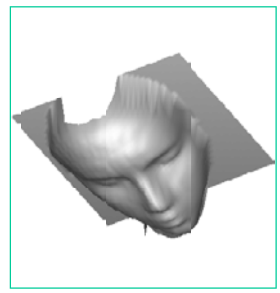
$$= K\left(\phi_D - \phi_C\right)$$

$$\boxed{Z = \overline{DB} \propto \left(\phi_D - \phi_C\right)}$$

Example

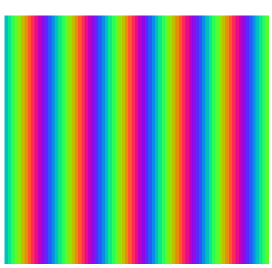






Color encoded phase-shifting methods (Huang et al. 1999)

- **3 phase-shifted patterns can be encoded as Red, Green and Blue channels of projector**



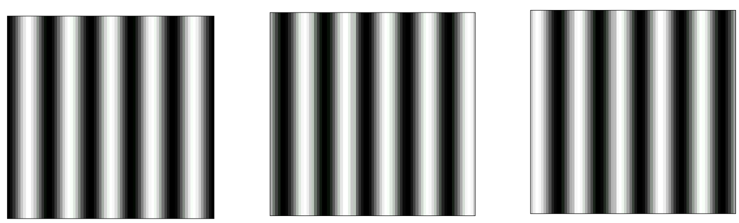
$$I_1 = I'(x,y) + I''(x,y)\cos[\phi(x,y) - \alpha]$$

$$I_2 = I'(x,y) + I''(x,y)\cos[\phi(x,y)]$$

$$I_3 = I'(x,y) + I''(x,y)\cos[\phi(x,y) + \alpha]$$

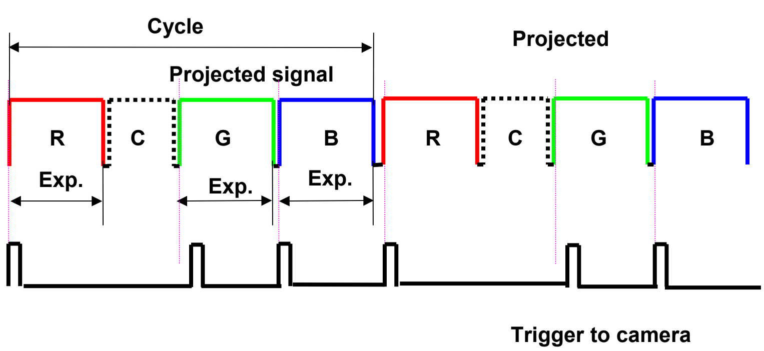
Color encoded phase-shifting methods (Huang et al. 1999)

- **A 3CCD color camera capture red green and blue channels of the projected patterns.**
  - Sensitive to color
  - Fail for color object



Color encoded phase-shifting methods (Huang et al. 02)

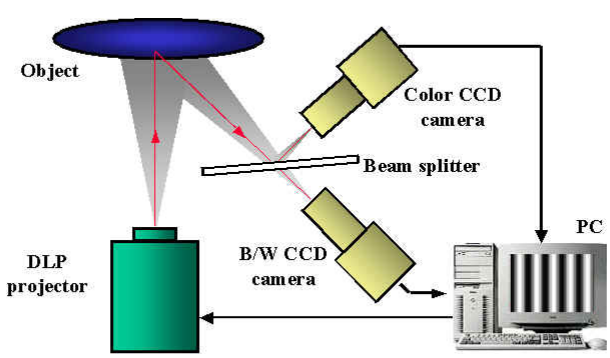
- **A B/W camera synchronized with projection (Single chip DLP projector) is used to capture 3 channels separately.**
  - DLP projector’s special projection mechanism



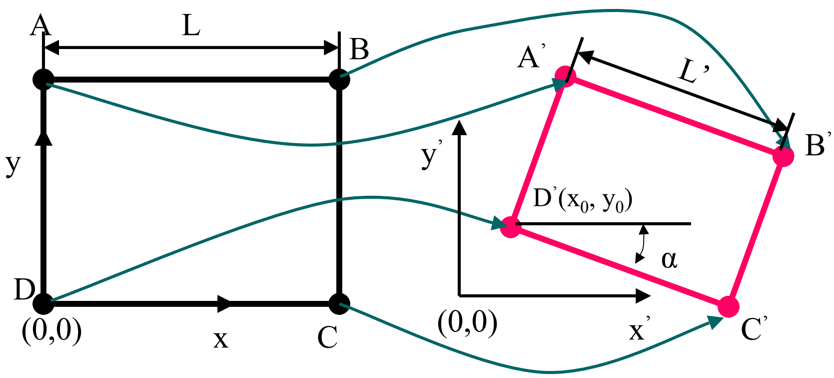
Color encoded phase-shifting methods (Huang et al. 02)

- **Advantages**
  - High spatial resolution
  - Fast.. How fast? (Projector's projection speed is the limit. 120Hz for single chip DLP projector, 3D speed can goes up to 360Hz)
  - It is possible to capture moving scenes accurately (RMS for Z, < 0.01mm)
  - Need not change the projected patterns during capturing images. (Our experience: changing projected patterns is very slow)
- **Black and white?**

Adding color texture



Matching 2 cameras



Matching 2 cameras (cont'd)

- **Translation**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

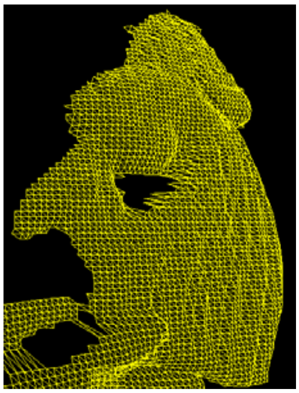
- **Rotation**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Scaling**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad s_x = s_y = \frac{L'}{L}$$

Results



3D image

2D color texture

3D image w texture

Why phase - shifting?

- **Reconstruction is very fast.** **(30ms is enough for PC Dell Dimension 4450 to do wrapping and unwrapping (image size 532 x 500)).**
- **So it is possible to do real-time grabbing, processing and displaying 3D scenes with high resolution.**

Problems

- **Ambiguity**
  - Periodic pattern, for sharp changing shape.
- **High-speed data transferring**
  - Over 60MB / sec data collecting speed
- **Real-time 3D grabbing, processing and displaying (30Hz)**
  - Possible?
    - Grabbing > 30Hz,
    - Processing > 30Hz
    - Rendering **?** 30Hz (Points: 532 x 500 @ 16bits)
  - How?
    - Parallel processing + high-speed rendering