# 3d shape reconstruction from photographs: a Multi-View Stereo approach

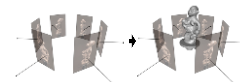
Carlos Hernández — Google
George Vogiatzis — Aston University
Yasutaka Furukawa — Google

http://carlos-hernandez.org/cvpr2010/index.html

# Talk plan

- Introduction

- Multi-View Stereo pipeline

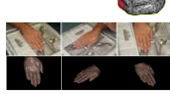


- Fusion of occlusion-robust depth-maps

- Region growing

# 3d model

- “Digital copy” of real object
- Allows us to
  - Inspect details of object
  - Measure properties
  - Reproduce in different material
- Many applications
  - Cultural heritage preservation
  - Computer games and movies
  - City modelling
  - E-commerce
  - 3d object recognition/scene analysis



# Applications: cultural heritage

**SCULPTEUR European project**

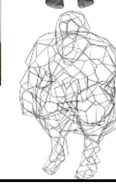
## Applications: art

Block Works Precipitate III 2004
*Mild steel blocks* 80 x 46 x 66 cm

Domain Series Domain VIII Crouching
1999 *Mild steel bar* 81 x 59 x 63 cm

## Applications: structure engineering

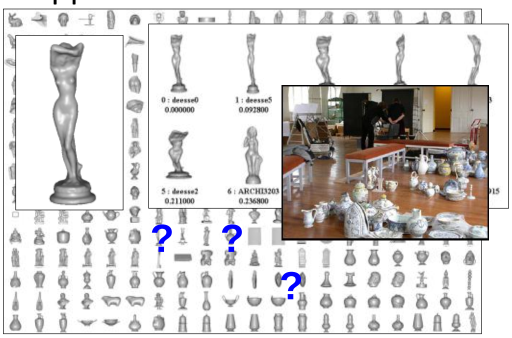
BODY / SPACE / FRAME, Antony Gormley, Lelystad, Holland



## Applications: art

## Applications: computer games

## Applications: 3D indexation

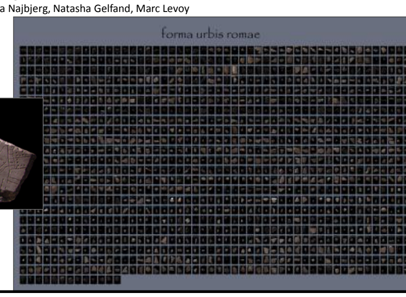


## Applications: archaeology

- “forma urbis romae” project

**Fragments of the City: Stanford's Digital Forma Urbis Romae Project**
David Koller, Jennifer Trimble, Tina Najbjerg, Natasha Gelfand, Marc Levoy
*Proc. Third Williams Symposium on Classical Architecture, Journal of Roman Archaeology supplement*, 2006.

1186 fragments

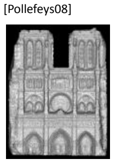



## Applications: large scale modelling

[Furukawa10]

[Pollefeys08]

[Cornelis08]



[Goesele07]

## Scanning technologies

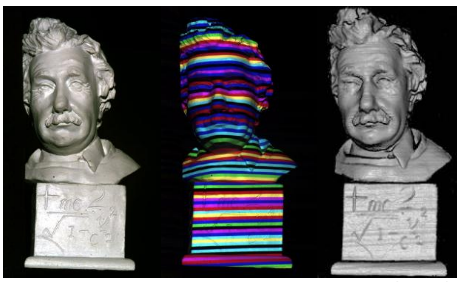
- Laser scanner, coordinate measuring machine
  - Very accurate
  - Very Expensive
  - Complicated to use

Minolta

Contura CMM

## Scanning technologies

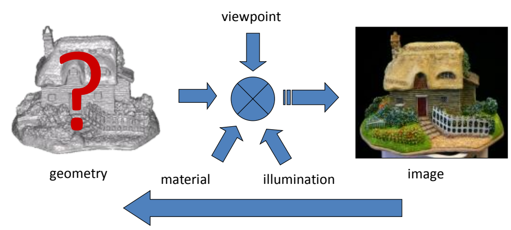
- Structured light



[Zhang02]

## 3d shape from photographs

*"Estimate a 3d shape that would generate the input photographs given the same material, viewpoints and illumination"*

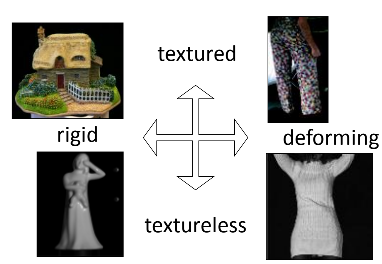



## 3d shape from photographs

*"Estimate a 3d shape that would generate the input photographs given the same material, viewpoints and illumination"*


## 3d shape from photographs

Appearance strongly depends on the material and lighting

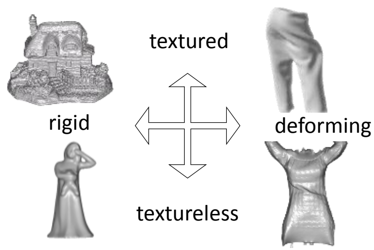


## 3d shape from photographs

Appearance strongly depends on the material and lighting

⇓

No single algorithm exists dealing with **any** type of scene



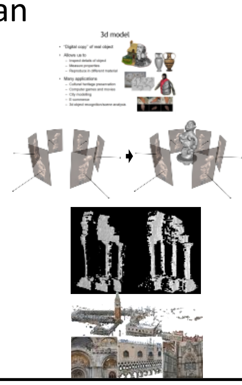

## 3d shape from photographs

Photograph based 3d reconstruction is:

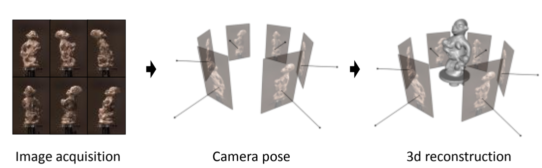
- ✓ practical
- ✓ fast
- ✓ non-intrusive
- ✓ low cost
- ✓ Easily deployable outdoors
- × “low” accuracy
- × Results depend on material properties

## Talk plan

- Introduction
- Multi-View Stereo pipeline
- Fusion of occlusion-robust depth-maps
- Region growing



## Multi-view stereo pipeline



## Image acquisition

- Studio conditions
  controlled environment

- Uncontrolled environment
  hand-held
  unknown illumination

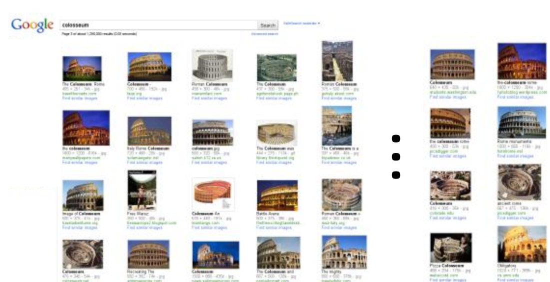
- Internet
  Unknown content

- Video
  small motion between frames
  huge amount of data

## Studio image acquisition

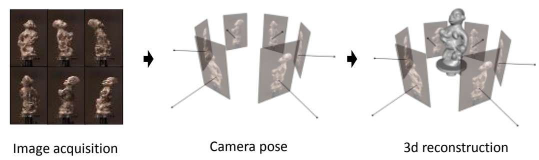
## Outdoor image acquisition

## Internet image acquisition



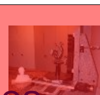
## Video image acquisition

## Multi-view stereo pipeline



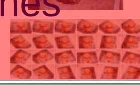

## Camera pose

- Robotic arm



Small Scenes

- Fiduciary markers



- Structure-from-Motion

Large scenes

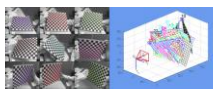
- SfM from unorganized photographs

## Robotic arm

## Fiduciary markers

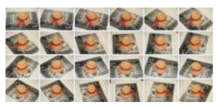
- ARToolkit

- Bouguet´s MATLAB Toolbox

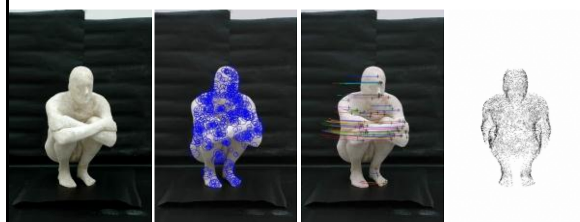


*www.vision.caltech.edu/**bouguet**j/calib_doc/*

- Robust planar patterns

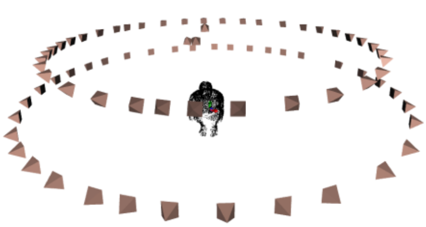


## Structure from motion



Input sequence | 2d features | 2d track | 3d points

## Motion estimation result



## Structure-from-Motion from unordered image collections

[Brown05, Snavely06, Agarwal09]

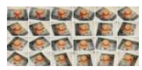
- Image clustering
- Pose initialization
- Bundle-adjustment

*phototour.cs.washington.edu/**bundler***

## Camera pose

- Robotic arm

- Fiducial markers



- Structure-from-Motion

- SfM from unorganized photographs

## Multi-view stereo pipeline

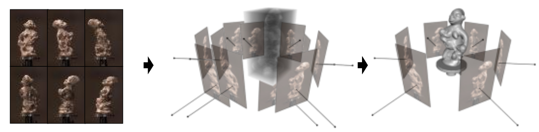


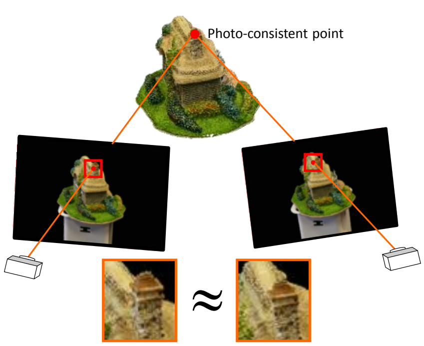
Image acquisition, camera pose

3d photo-consistency from images

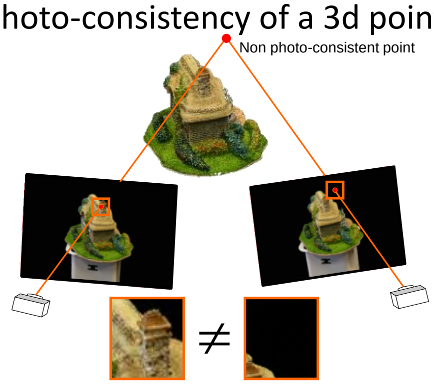
3d surface from 3d photo-consistency

**3d reconstruction = 3d segmentation**

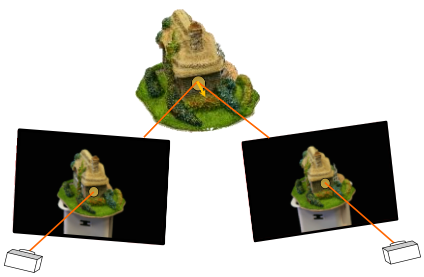
## Photo-consistency of a 3d point




## Photo-consistency of a 3d point

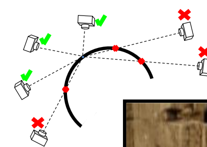


## Photo-consistency of a 3d patch



## Challenges of photo-consistency

- Camera visibility



- Failure of comparison metric
  - repeated texture
  - lack of texture
  - specularities

## Multi-view stereo algorithms

- Comparison and evaluation:
  - **A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms**, S. Seitz et al., CVPR 2006, vol. 1, pages 519-526.
- Quick history of algorithms:
  - **Representing stereo data with the Delaunay triangulation**, O. Faugeras et al., Artificial Intelligence, 44(1-2):41-87, 1990.
  - **A multiple-baseline stereo**, M. Okutomi and T. Kanade, TPAMI, 15(4):353-363, 1993.
  - **Object-centered surface reconstruction: Combining multi-image stereo and shading**, P. Fua, Y. Leclerc, International Journal of Computer Vision, vol. 16:35-56, 1995.
  - **A portable three-dimensional digitizer**, Y. Matsumoto et al., Int. Conf. on Recent Advances in 3D Imaging and Modeling, 197-205, 1997.
  - **Photorealistic Scene Reconstruction by Voxel Coloring**, S. M. Seitz and C. R. Dyer, CVPR., 1067-1073, 1997.
  - **Variational principles, surface evolution, PDE's, level set methods and the stereo problem**, O. Faugeras and R. Keriven, IEEE Trans. on Image Processing, 7(3):336-344, 1998.

## Multi-view stereo algorithms

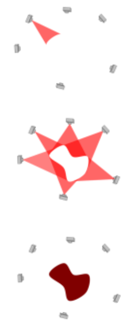
- Comparison and evaluation:
  - **A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms**, S. Seitz et al., CVPR 2006, vol. 1, pages 519-526.
  - http://vision.middlebury.edu/mview/
- Recently many new algorithms
- Very good accuracy & completeness
- Almost all deal with small number of images (~100) main exception [Pollefeys08]
- Offline algorithms, no feedback

## Best flexible algorithms

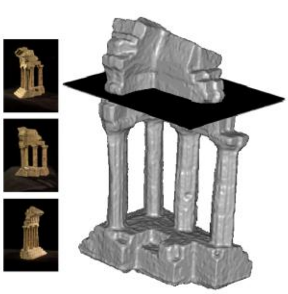
| | Region growing | Depth-map fusion |
|---|---|---|
| **summary** | Starts from a cloud of 3d points, and grows small flat patches maximizing photo-consistency | Fuses a set of depth-maps computed using occlusion-robust photo-consistency |
| **pros** | Provides best overall results due to a plane-based photo-consistency | Elegant pipeline<br>Plug-n-play blocks<br>Easily parallelizable |
| **cons** | Many tunable parameters, i.e., difficult to tune to get the optimal results | Photo-consistency metric is simple and not optimal. The metric suffers when images are not well textured or low resolution |

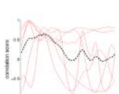
## Bird's-eye view: depth-map fusion

1. **Compute depth hypotheses**
2. **Volumetrically fuse depth-maps**
3. **Extract 3d surface**

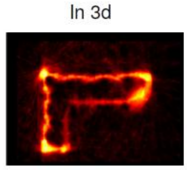


## Averaged NCC





In 3d



Aston University

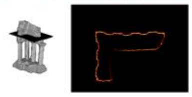
## Extracting a surface from photo-consistency

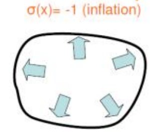
- Define a surface functional E[S] = some cost
- Minimize E
- Need *volume term* to avoid collapsing to a point

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

ρ(x) = 1.0 - Photoconsistency

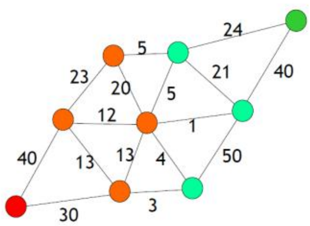
σ(x)= -1 (inflation)





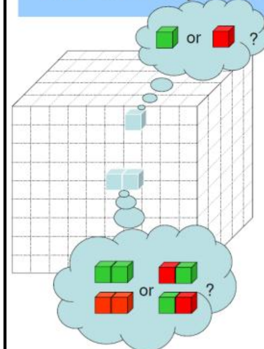
Aston University

## Graph *cut*




Aston University

## 3D binary labelling problem




Labelling cost:

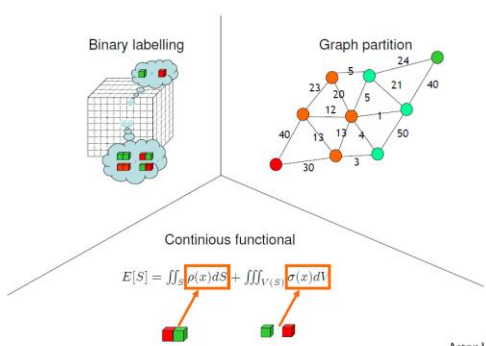
- Every voxel has a certain preference for being **foreground** or **background**

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)


Aston University

## Three equivalent representations

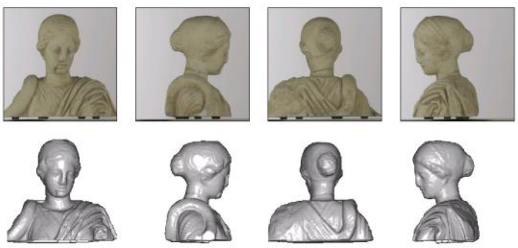



Aston University

## Results

Aston University

## Results

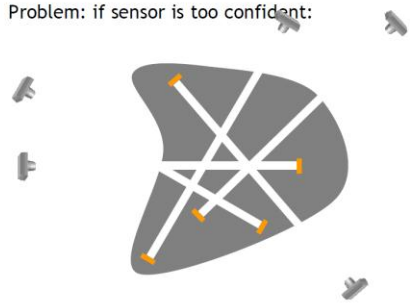


gcut3d binary

Aston University

## Data driven 3D MRF labelling cost

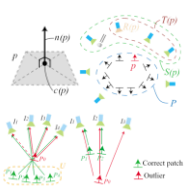
- Problem: if sensor is too confident:



Aston University

## Bird's-eye view: region growing

1. **Fitting step**
   A local surface patch is fitted, iterating visibility
2. **Filter step**
   Visibility is explicitly enforced



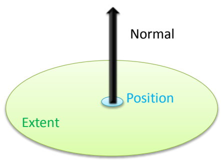

3. **Expand step**
   Successful patches are used to initialise active boundary

## Patch-based MVS and its Applications

- Why patches (oriented points)? [10 mins]
- Algorithmic details [30 mins]
- Applications [20 mins]

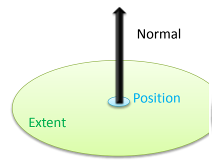
## What is a patch?

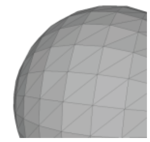
- Patch consists of
  - Position (*x, y, z*)
  - Normal (*nx, ny, nz*)
  - Extent (*radius*)
- Tangent plane approximation

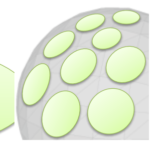



## What is a patch?

- Patch consists of
  - Position (*x, y, z*)
  - Normal (*nx, ny, nz*)
  - Extent (*radius*)
- Tangent plane approximation

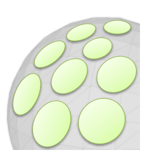




Mesh


Patch

## Why patches?

- Flexible



## Why patches?

- Flexible

## Why patches?

- Flexible ⟺ Hard to enforce regularization

## Why patches?

- Flexible ⟺ Hard to enforce regularization

Regularization not really necessary

because

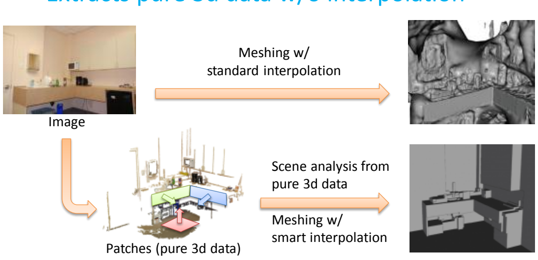
Local image patch is descriptive enough

[Goesele et al., CVPR06], [Furukawa et al., CVPR07]

9x9 pixels

## Why patches?

- Extracts pure 3d data w/o interpolation




[Sinha ICCV09, Furukawa ICCV09]

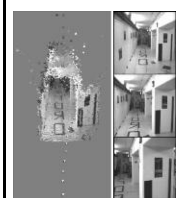
## Patches vs. multiple depthmaps (could be my biased-view…)

- Patches → Single global 3D model
  Depthmaps → Multiple redundant 3D models
- Patches → Clean 3D points
  Depthmaps → Noisy without merging
- Patches → Hard to make it fast (complex algo)
  Depthmaps → Easy to make it fast

## Patch-based MVS and its Applications

- Why patches (oriented points)? [10 mins]
- Algorithmic details [30 mins]
- Applications [20 mins]

## Patch-based MVS



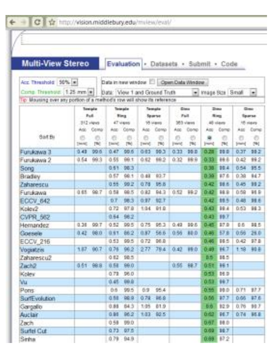
[Lhuillier and Quan, PAMI 05]

[Furukawa and Ponce, CVPR 07]

[Habbecke and Kobbelt, CVPR 07]

## Patch-based MVS

- Shown to work very well




Multi-view stereo evaluation
http://vision.middlebury.edu/mview
D. Scharstein et al.

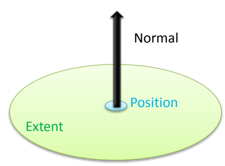
## Patch-based MVS [Furukawa and Ponce 07]

- Preliminaries
- Algorithm

Input image #1 #2 #3

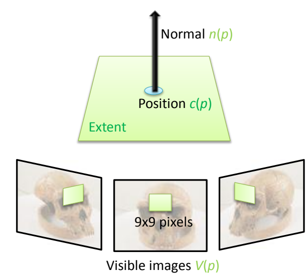
## Patch definition

- Position, normal, and extent




## Patch definition

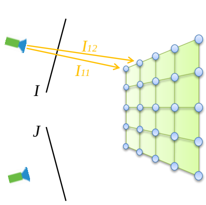
- Patch *p* is defined by
  - Position *c*(*p*)
  - Normal *n*(*p*)
  - Visible images *V*(*p*)
- Extent is set so that *p* is roughly 9x9 pixels in *V*(*p*)




## Photo-consistency

- Photo-consistency *N*(*I*, *J*, *p*) of *p* between two images *I* and *J*

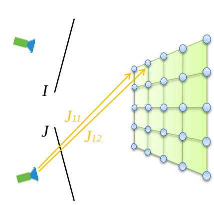
$I_{xy}$: pixel color in image $I$




## Photo-consistency

- Photo-consistency *N*(*I*, *J*, *p*) of *p* between two images *I* and *J*

$I_{xy}$: pixel color in image $I$
$J_{xy}$: pixel color in image $J$



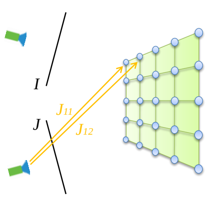

$$N(I,J,p)=\frac{\sum (I_{xy}-\overline{I_{xy}})\cdot(J_{xy}-\overline{J_{xy}})}{\sqrt{(I_{xy}-\overline{I_{xy}})^2}\sqrt{(J_{xy}-\overline{J_{xy}})^2}}$$

## Photo-consistency

- Photo-consistency $N(I, J, p)$ of $p$ between two images $I$ and $J$

$I_{xy}$: pixel color in image $I$
$J_{xy}$: pixel color in image $J$

Photo-consistency $N(p)$ of $p$ with



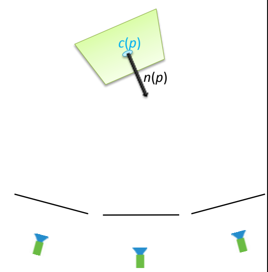

$$N(I,J,p)=\frac{\sum(I_{xy}-\overline{I_{xy}})\cdot(J_{xy}-\overline{J_{xy}})}{\sqrt{(I_{xy}-\overline{I_{xy}})^2}\sqrt{(J_{xy}-\overline{J_{xy}})^2}}$$

$$N(p)=\frac{\sum_{i=1}^{n}\sum_{j=i+1}^{n}N(I_i,I_j,p)}{(n+1)n/2}$$

## Reconstruct patch $p$

- Given initial estimates of
  - Position $c(p)$
  - Normal $n(p)$
  - Visible images $V(p)$
- Refine $c(p)$ and $n(p)$

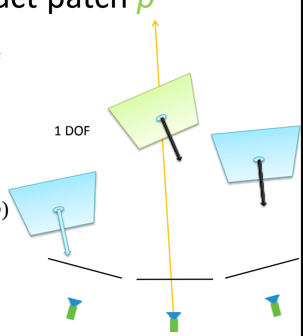
$$\{c(p),n(p)\}=\underset{\{c(p),n(p)\}}{\arg\max}\,N(p)$$




## Reconstruct patch $p$

- Given initial estimates of
  - Position $c(p)$
  - Normal $n(p)$
  - Visible images $V(p)$
- Refine $c(p)$ and $n(p)$

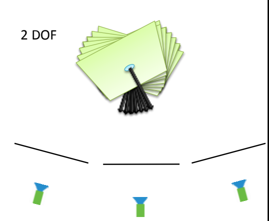
$$\{c(p),n(p)\}=\underset{\{c(p),n(p)\}}{\arg\max}\,N(p)$$




## Reconstruct patch $p$

- Given initial estimates of
  - Position $c(p)$
  - Normal $n(p)$
  - Visible images $V(p)$
- Refine $c(p)$ and $n(p)$

$$\{c(p),n(p)\}=\underset{\{c(p),n(p)\}}{\arg\max}\,N(p)$$



## Verify a patch

- Textures may match by accident

## Verify a patch

- Textures may match by accident
- Photo-consistency must be reasonably high

## Verify a patch

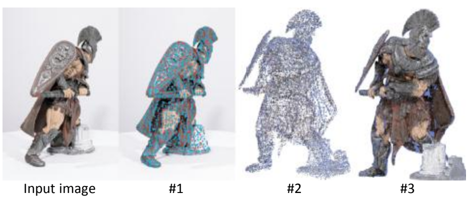
- Textures may match by accident
- Photo-consistency must be reasonably high
- Verification process
  - Keep only high photo-consistency images in $V(p)$

## Verify a patch

- Textures may match by accident
- Photo-consistency must be reasonably high
- Verification process
  - Keep only high photo-consistency images in $V(p)$
  - Accept if $|V(p)| \geq 3$

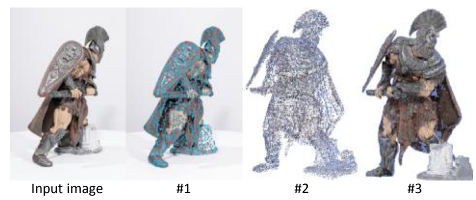
## Patch-based MVS [Furukawa and Ponce 07]

- Preliminaries
- Algorithm



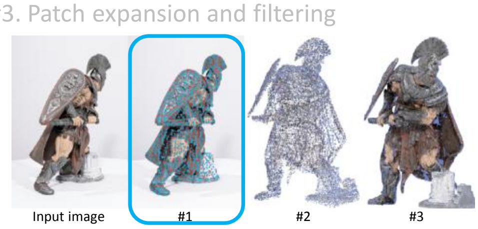

## Patch-based MVS [Furukawa and Ponce 07]

#1. Feature detection
#2. Initial feature matching
#3. Patch expansion and filtering



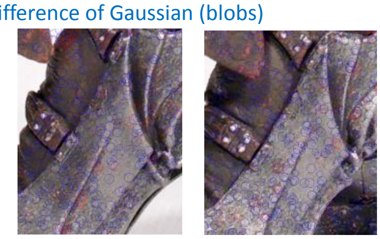

## Algorithm overview

#1. Feature detection
#2. Initial feature matching
#3. Patch expansion and filtering




## Feature detection

- Extract local maxima of
  - Harris corner detector (corners)
  - Difference of Gaussian (blobs)

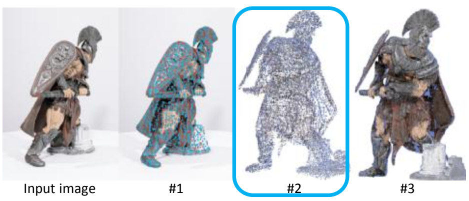


## Algorithm overview

#1. Feature detection

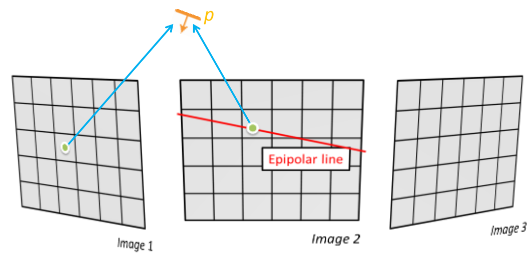
#2. Initial feature matching

#3. Patch expansion and filtering



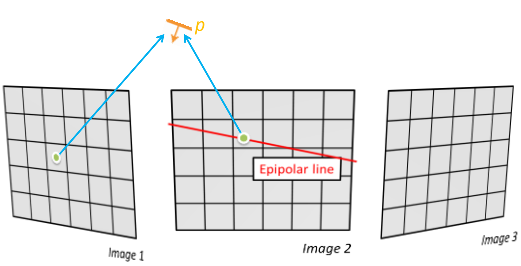

## Initial feature matching

*c*(*p*): triangulation
*n*(*p*):
*V*(*p*):



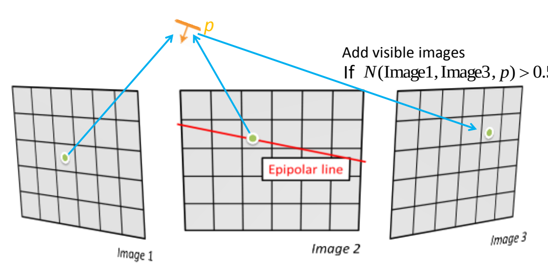

## Initial feature matching

*c*(*p*): triangulation
*n*(*p*): parallel to *Image1*
*V*(*p*): {*Image1, Image2*}



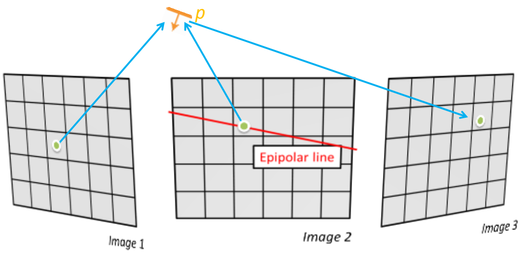

## Initial feature matching

*c*(*p*): triangulation
*n*(*p*): parallel to *Image1*
*V*(*p*): {*Image1, Image2*} *Image3*}



## Initial feature matching

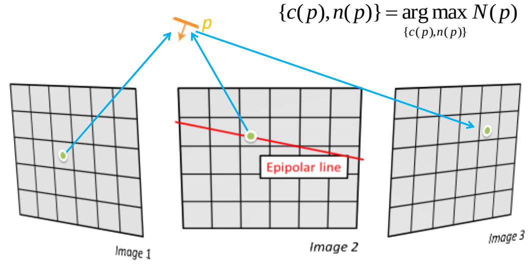
*c*(*p*): triangulation
*n*(*p*): parallel to *Image1*
*V*(*p*): {*Image1, Image2, Image3*}




## Initial feature matching

*c*(*p*): refine
*n*(*p*): refine
*V*(*p*): {*Image1, Image2, Image3*}

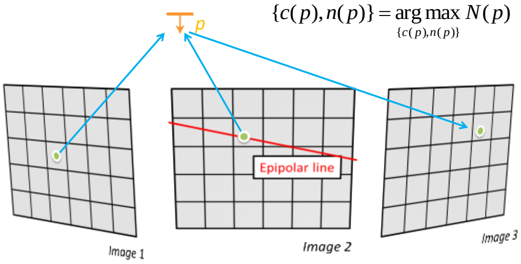
$$\{c(p), n(p)\} = \underset{\{c(p),n(p)\}}{\arg\max}\, N(p)$$




## Initial feature matching

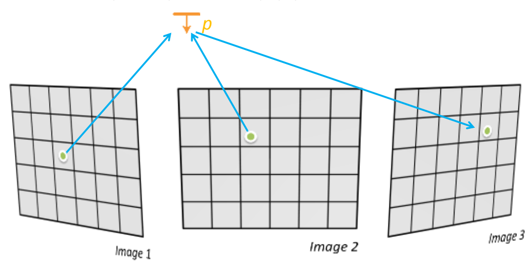
*c*(*p*): refine
*n*(*p*): refine
*V*(*p*): {*Image1, Image2, Image3*}

$$\{c(p), n(p)\} = \underset{\{c(p),n(p)\}}{\arg\max}\, N(p)$$

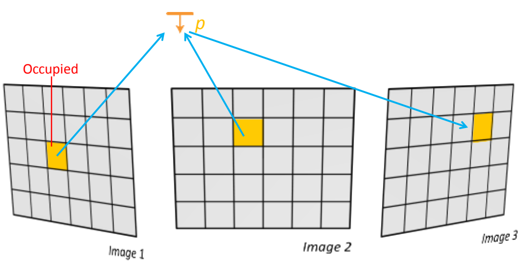



## Initial feature matching

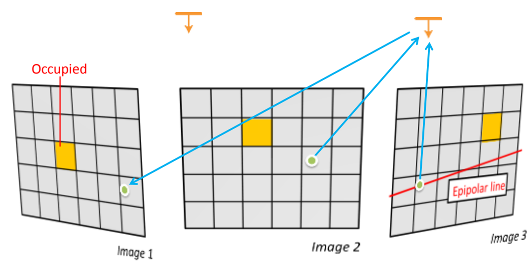
Verification
(update *V*(*p*) and check |*V*(*p*)|≥3)



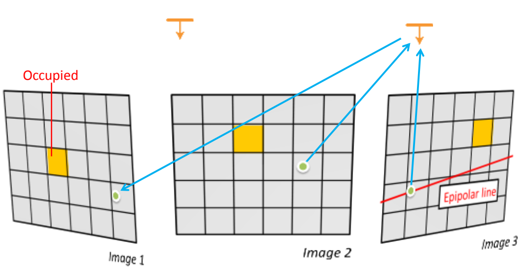
# Initial feature matching



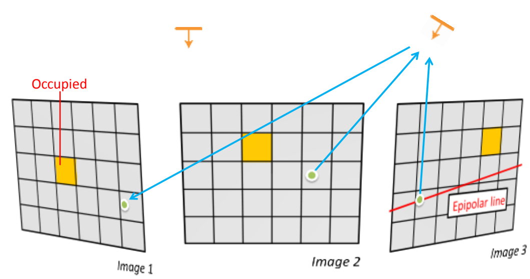

# Initial feature matching




# Initial feature matching

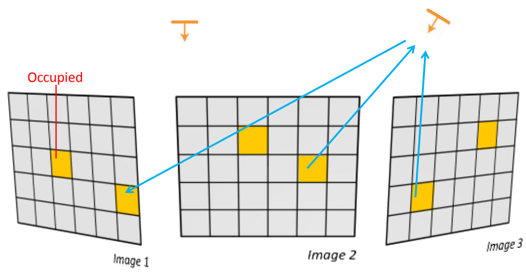



# Initial feature matching



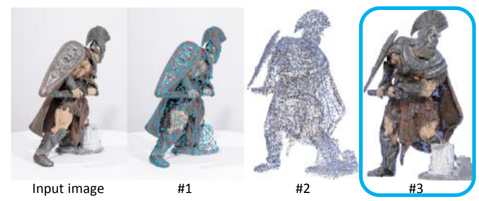
## Initial feature matching

- Repeat for all the image features

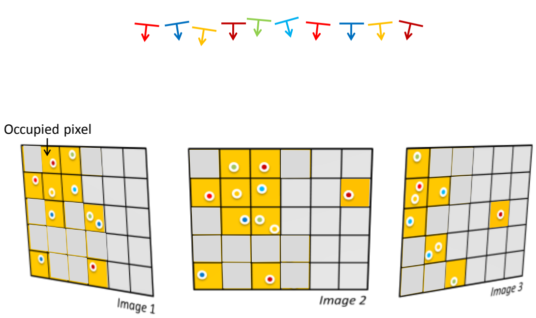



## Algorithm overview

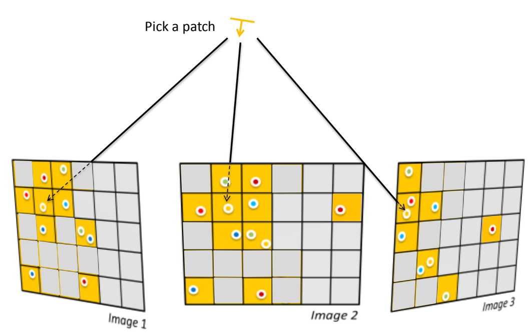
#1. Feature detection
#2. Initial feature matching
#3. Patch expansion and filtering



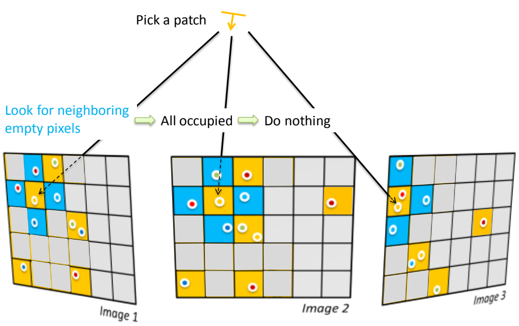

## Patch expansion



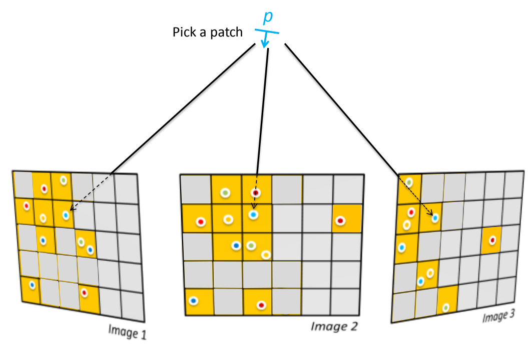

## Patch expansion



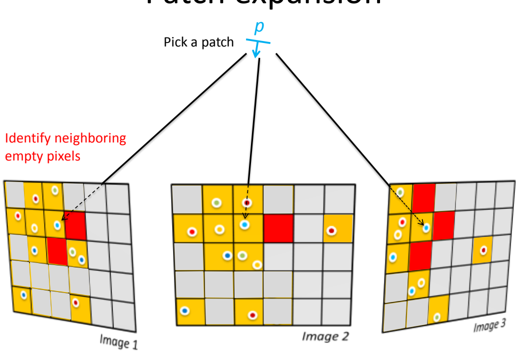
# Patch expansion



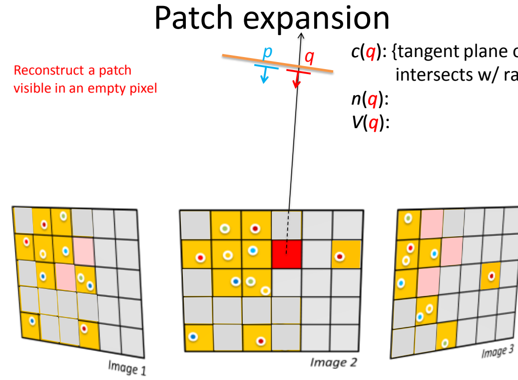

# Patch expansion




# Patch expansion



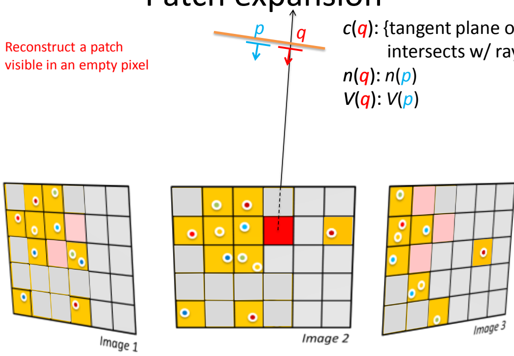

# Patch expansion




*c*(*q*): {tangent plane of *p* intersects w/ ray}
*n*(*q*):
*V*(*q*):

## Patch expansion

Reconstruct a patch visible in an empty pixel



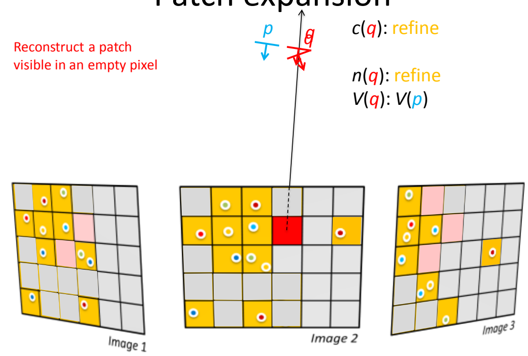

$c(q)$: {tangent plane of $p$ intersects w/ ray}

$n(q)$: $n(p)$

$V(q)$: $V(p)$

## Patch expansion

Reconstruct a patch visible in an empty pixel



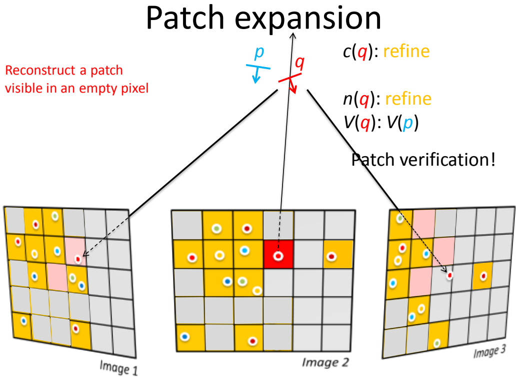

$c(q)$: refine

$n(q)$: refine

$V(q)$: $V(p)$

## Patch expansion

Reconstruct a patch visible in an empty pixel




$c(q)$: refine

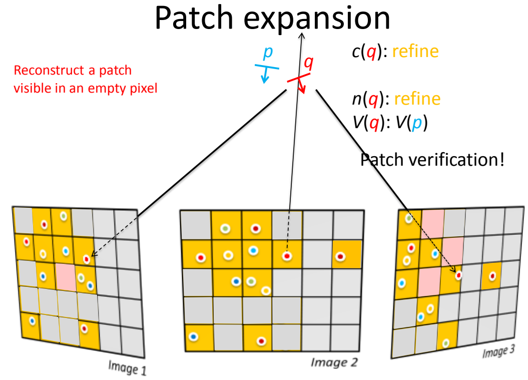
$n(q)$: refine

$V(q)$: $V(p)$

Patch verification!

## Patch expansion

Reconstruct a patch visible in an empty pixel



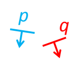

$c(q)$: refine

$n(q)$: refine

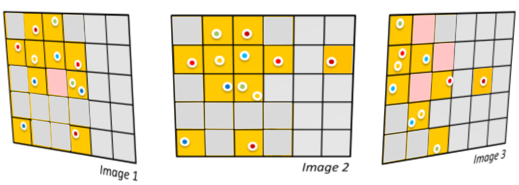
$V(q)$: $V(p)$

Patch verification!

## Patch expansion




Repeat
• for every patch
• for every neighboring empty pixel

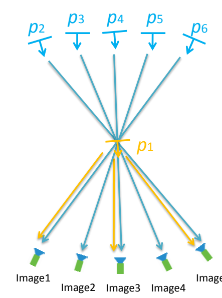



## Patch filtering

• Visibility consistency

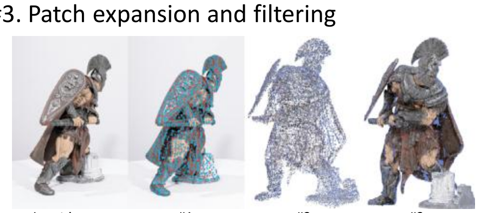
Filter out $p_1$ if

$$|V(p_1)|\,N(p_1) < \sum_{i=2}^{6} N(p_i)$$




## Patch-based MVS [Furukawa and Ponce 07]

#1. Feature detection
#2. Initial feature matching
#3. Patch expansion and filtering




Skull - 24 images
2000x2000 [pixels]

Face - 4 images
1400x2200 [pixels]
(Courtesy of Industrial Light & Magic)