Vision 3D artificielle
Disparity maps, correlation

Pascal Monasse pascal.monasse@enpc.fr

IMAGINE, École des Ponts ParisTech
http://imagine.enpc.fr/~monasse/Stereo/

Contents

Triangulation

Epipolar rectification

Disparity map

Contents

Triangulation

Epipolar rectification

Disparity map

Triangulation

- Let us write again the binocular formulae (in $\mathbb{P}^2$):

$$x = PX \quad x' = P'X$$

- We can write in homogeneous coordinates

$$[x]_\times PX = 0_3 \quad [x']_\times P'X = 0_3$$

- We can then recover X through SVD:

$$X \in \operatorname{Ker}\begin{pmatrix} [x]_\times P \\ [x']_\times P' \end{pmatrix}$$

Triangulation

- Let us write again the binocular formulae:

$$\lambda x = K(RX + T) \quad \lambda' x' = K' X$$

- Write $Y^\top = \begin{pmatrix} X^\top & 1 & \lambda & \lambda' \end{pmatrix}$:

$$\begin{pmatrix} KR & KT & -x & 0_3 \\ K' & 0_3 & 0_3 & -x' \end{pmatrix} Y = 0_6$$

(6 equations$\leftrightarrow$5 unknowns+1 epipolar constraint)

- We can then recover X.
- Special case: $R = Id$, $T = Be_1$
- We get:

$$z(x - KK'^{-1}x') = \begin{pmatrix} Bf & 0 & 0 \end{pmatrix}^\top$$

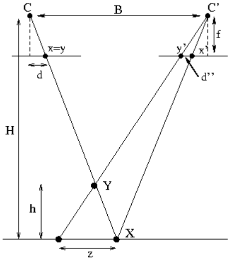
- If also $K = K'$,

$$z = fB/[(x - x') \cdot e_1] = fB/d$$

- d is the disparity

Triangulation

Fundamental principle of stereo vision



$$h \backsimeq \frac{z}{B/H}, \quad z = d'' \frac{H}{f}.$$

f focal length.

H distance optical center-ground.

B distance between optical centers (baseline).

Goal

Given two rectified images, point correspondences and computation of their apparent shift (disparity) gives information about relative depth of the scene.

Recovery of R and T

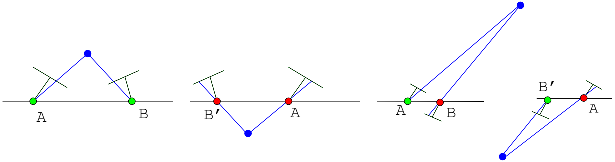
- Suppose we know K, K', and F or E. Recover R and T?
- From $E = [T]_\times R$,

$$E^\top E = -R^\top(TT^\top - \|T\|^2 I)R = -(R^\top T)(R^\top T)^\top + \|R^\top T\|^2 I$$

- If $\mathbf{x} = R^\top T$, $E^\top E\mathbf{x} = 0$ and if $\mathbf{y} \cdot \mathbf{x} = 0$, $E^\top E\mathbf{y} = \|T\|^2\mathbf{y}$.
- Therefore $\sigma_1 = \sigma_2 = \|T\|$ and $\sigma_3 = 0$.
- Inversely, from $E = Udiag(\sigma, \sigma, 0)V^\top$, we can write:

$$E = \sigma U \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^\top U \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} V^\top = \sigma[T]_\times R$$

- Actually, there are up to 4 solutions: $\begin{cases} T = \pm\sigma U[e_3]_\times U^\top \\ R = UR_z(\pm\frac{\pi}{2})V^\top \end{cases}$



What is possible without calibration?

- We can recover F, but not E.
- Actually, from

$$\mathbf{x} = P\mathbf{X} \quad \mathbf{x}' = P'\mathbf{X}$$

we see that we have also:

$$\mathbf{x} = (PH^{-1})(H\mathbf{X}) \quad \mathbf{x}' = (P'H^{-1})(H\mathbf{X})$$

- Interpretation: applying a space homography and transforming the projection matrices (this changes K, K', R and T), we get exactly the same projections.
- Consequence: in the uncalibrated case, reconstruction can only be done modulo a 3D space homography.

Contents

Triangulation

Epipolar rectification

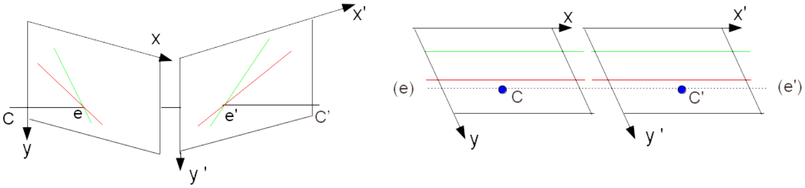
Disparity map

Epipolar rectification

- It is convenient to get to a situation where epipolar lines are parallel and at same ordinate in both images.
- As a consequence, epipoles are at horizontal infinity:

$$e = e' = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

- It is always possible to get to that situation by virtual rotation of cameras (application of homography)



- Image planes coincide and are parallel to baseline.

Epipolar rectification

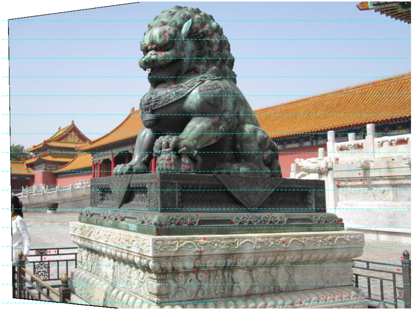
Image 1

Epipolar rectification

Image 2

Epipolar rectification

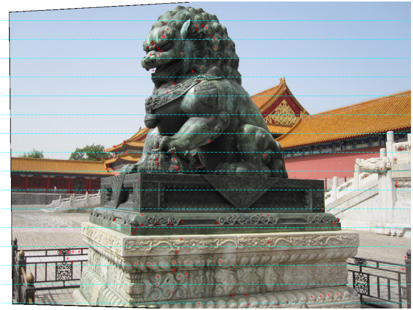
Image 1



Rectified image 1

Epipolar rectification

Image 2



Rectified image 2

Epipolar rectification

- Fundamental matrix can be written:

$$F = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}_\times = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \text{ thus } x^\top F x' = 0 \Leftrightarrow y - y' = 0$$

- Writing matrices $P = K \begin{pmatrix} I & 0 \end{pmatrix}$ and $P' = K' \begin{pmatrix} I & Be_1 \end{pmatrix}$:

$$K = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \quad K' = \begin{pmatrix} f'_x & s' & c'_x \\ 0 & f'_y & c'_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$F = BK^{-\top}[e_1]_\times K'^{-1} = \frac{B}{f_y f'_y} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -f_y \\ 0 & f'_y & c'_y f_y - c_y f'_y \end{pmatrix}$$

- We must have $f_y = f'_y$ and $c_y = c'_y$, that is identical second rows of K and K'

Epipolar rectification

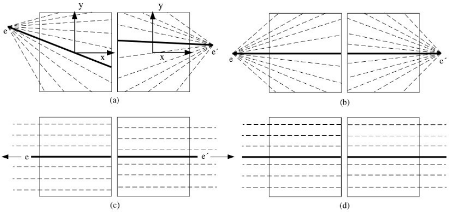
- We are looking for homographies H and H' to apply to images such that

$$F = H^\top [e_1]_\times H'$$

- That is 9 equations and 16 variables, 7 degrees of freedom remain: the first rows of K and K' and the rotation angle around baseline α
- Invariance through rotation around baseline:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}^\top \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} = [e_1]_\times$$

- Several methods exist, they try to distort as little as possible the image



Rectif. of Gluckman-Nayar (2001)

Epipolar rectification of Fusiello-Irsara (2008)

- We are looking for H and H' as rotations, supposing $K = K'$ known:

$$H = K_n R K^{-1} \text{ and } H' = K'_n R' K^{-1}$$

with K_n and K'_n of identical second row, R and R' rotation matrices parameterized by Euler angles and

$$K = \begin{pmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{pmatrix}$$

- Writing $R = R_x(\theta_x) R_y(\theta_y) R_z(\theta_z)$ we must have:

$$F = (K_n R K^{-1})^\top [e_1]_\times (K'_n R' K^{-1}) = K^{-\top} R_z^\top R_y^\top [e_1]_\times R' K^{-1}$$

- We minimize the sum of squares of points to their epipolar line according to the 6 parameters

$$(\theta_y, \theta_z, \theta'_x, \theta'_y, \theta'_z, f)$$

Ruins

$\|E_0\| = 3.21$ pixels.

$\|E_6\| = 0.12$ pixels.

Ruins

$\|E_0\| = 3.21$ pixels.

$\|E_6\| = 0.12$ pixels.

Cake

$\|E_0\| = 17.9$ pixels.

$\|E_{13}\| = 0.65$ pixels.

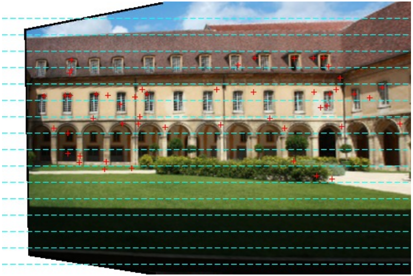
Cake

$\|E_0\| = 17.9$ pixels.

$\|E_{13}\| = 0.65$ pixels.

Cluny

$\|E_0\| = 4.87$ pixels.



$\|E_{14}\| = 0.26$ pixels.

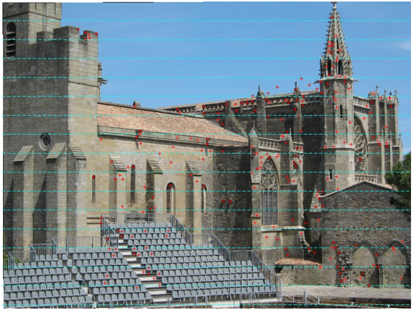
Cluny

$\|E_0\| = 4.87$ pixels.

$\|E_{14}\| = 0.26$ pixels.

Carcassonne

$\|E_0\| = 15.6$ pixels.



$\|E_4\| = 0.24$ pixels.

Carcassonne

$\|E_0\| = 15.6$ pixels.

$\|E_4\| = 0.24$ pixels.

Books

$\|E_0\| = 3.22$ pixels.

$\|E_{14}\| = 0.27$ pixels.

Books

$\|E_0\| = 3.22$ pixels.

$\|E_{14}\| = 0.27$ pixels.

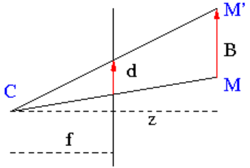
Contents

Triangulation

Epipolar rectification

Disparity map

Disparity map



$$z = \frac{fB}{d}$$

Depth z is inversely proportional to disparity d (apparent motion, in pixels).

- Disparity map: At each pixel, its apparent motion between left and right images.
- We already know disparity at feature points, this gives an idea about min and max motion, which makes the search for matching points less ambiguous and faster.

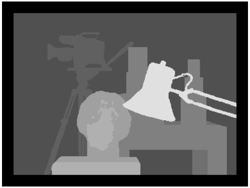
Stereo Matching

- Principle: invariance of something between corresponding pixels in left and right images (I_L, I_R)
- Example: color, x-derivative, census...
- Usage of a distance to capture this invariance, such as $\mathrm{AD}(p, q) = \|I_L(p) - I_R(q)\|_1$

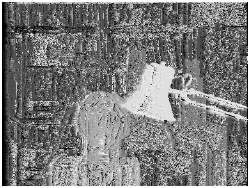
Stereo Matching

- Principle: invariance of something between corresponding pixels in left and right images (I_L, I_R)
- Example: color, x-derivative, census...
- Usage of a distance to capture this invariance, such as $\mathrm{AD}(p,q) = \|I_L(p) - I_R(q)\|_1$

Left image



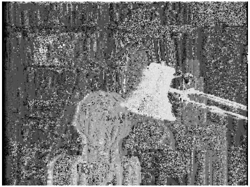
Ground truth

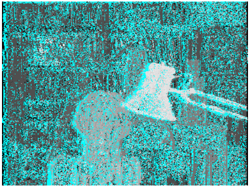


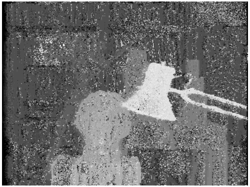
Min AD

Stereo Matching

- Post-processing helps a lot!
- Example: left-right consistency check, followed by simple constant interpolation, and median weighted by original image bilateral weights

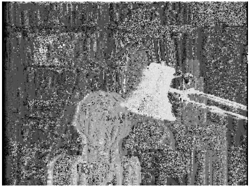
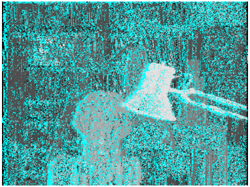
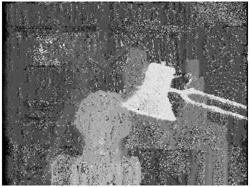

Min CG


Left-right test


Post-processed

Stereo Matching

- Post-processing helps a lot!
- Example: left-right consistency check, followed by simple constant interpolation, and median weighted by original image bilateral weights

Min CG | Left-right test | Post-processed

- Still, single pixel estimation not good enough
- Need to promote some regularity of the result

Global vs. local methods

- Global method: explicit smoothness term

$$\arg\min_d \sum_p E_{\text{data}}(p, p + d(p); I_L, I_R) + \sum_{p \sim p'} E_{\text{reg}}(d(p), d(p'); p, p', I_L, I_R)$$

- Examples: $E_{\text{reg}} = |d(p) - d(p')|^2$ (Horn-Schunk),
  $E_{\text{reg}} = \delta(d(p) = d(p'))$ (Potts),
  $E_{\text{reg}} = \exp(-(I_L(p) - I_L(p'))^2/\sigma^2)|d(p) - d(p')|$...

Global vs. local methods

- Global method: explicit smoothness term

$$\arg\min_d \sum_p E_{\text{data}}(p, p + d(p); I_L, I_R) + \sum_{p \sim p'} E_{\text{reg}}(d(p), d(p'); p, p', I_L, I_R)$$

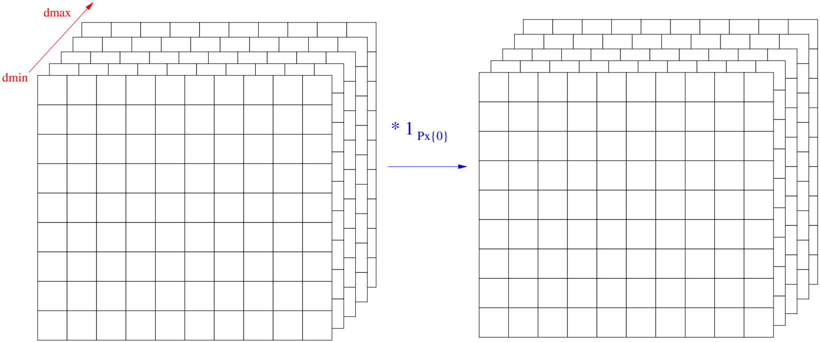
- Examples: $E_{\text{reg}} = |d(p) - d(p')|^2$ (Horn-Schunk), $E_{\text{reg}} = \delta(d(p) = d(p'))$ (Potts), $E_{\text{reg}} = \exp(-(I_L(p) - I_L(p'))^2/\sigma^2)|d(p) - d(p')|\ldots$
- Problem: NP-hard for almost all regularity terms except

$$E_{\text{reg}} = \lambda_{pp'}|d(p) - d(p')| \quad \text{(Ishikawa 2003)}$$

- Aternative: sub-optimal solution for submodular regularity (graph-cuts: Boykov, Kolmogorov, Zabih), loopy-belief propagation (no guarantee at all), semi-global matching (Hirschmüller)

Global vs. local methods

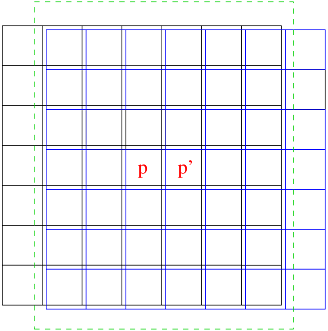
- Local method: Take a patch around p, aggregate costs E_{data} (Lucas-Kanade) $\Rightarrow$ No explicit regularity term
- Example: $\text{SAD}(p,q) = \sum_{r\in P} |I_L(p+r) - I_R(q+r)|$, $\text{SSD}(p,q) = \sum_{r\in P} |I_L(p+r) - I_R(q+r)|^2$, $\text{SCG}(p,q) = \sum_{r\in P} \text{CG}(p+r, q+r)$...
- Can be interpreted as a cost-volume filtering.



- Increasing patch size P promotes regularity.

Global vs. local methods

- Local method: Take a patch around p, aggregate costs E_{data} (Lucas-Kanade) $\Rightarrow$ No explicit regularity term
- Example: $\text{SAD}(p,q) = \sum_{r\in P} |I_L(p+r) - I_R(q+r)|$,
  $\text{SSD}(p,q) = \sum_{r\in P} |I_L(p+r) - I_R(q+r)|^2$,
  $\text{SCG}(p,q) = \sum_{r\in P} \text{CG}(p+r, q+r)$...
- Can be interpreted as a cost-volume filtering.
- Increasing patch size P promotes regularity.



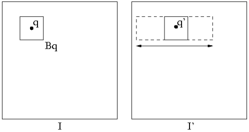
Proportion of common pixels between $p+P$ and $p'+P$:

$$1 - \frac{1}{n}$$

if P is $n \times n$

Local search

- At each pixel, we consider a context window and we look for the motion of this window.



- Distance between windows:

$$d(q) = \arg\min_{d} \sum_{p \in F} (I(q+p) - I'(q + de_1 + p))^2$$

- Variants to be more robust to illumination changes:
  1. Translate intensities by the mean over the window.

$$I(q+p) \to I(q+p) - \sum_{r \in F} I(q+r)/\#F$$

  2. Normalize by mean and variance over window.

Distance between patches

Several distances or similarity measures are popular:

- SAD: Sum of Absolute Differences

$$d(q) = \arg\min_d \sum_{p\in F} |I(q+p) - I'(q+de_1+p)|$$

- SSD: Sum of Squared Differences

$$d(q) = \arg\min_d \sum_{p\in F} (I(q+p) - I'(q+de_1+p))^2$$

- CSSD: Centered Sum of Squared Differences

$$d(q) = \arg\min_d \sum_{p\in F} (I(q+p) - \bar{I}_F - I'(q+de_1+p) + \bar{I}'_F)^2$$

- NCC: Normalized Cross-Correlation

$$d(q) = \arg\max_d \frac{\sum_{p\in F}(I(q+p) - \bar{I}_F)(I'(q+de_1+p) - \bar{I}'_F)}{\sqrt{\sum(I(q+p) - \bar{I}_F)^2}\sqrt{\sum(I'(q+de_1+p) - \bar{I}'_F)^2}}$$

Another distance

- The following distance is more and more popular in recent articles:

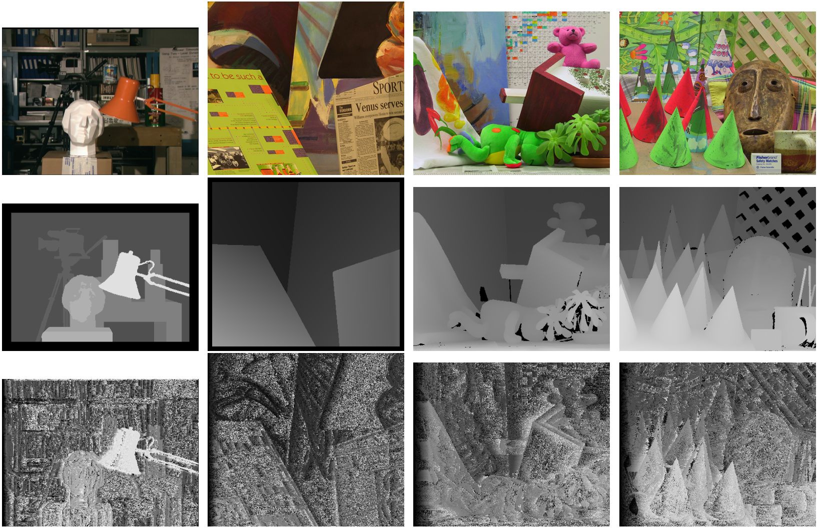
$$\epsilon(p,q) = (1-\alpha)\min\left(\|I(p)-I'(q)\|_1, \tau_{\text{col}}\right) + \alpha\min\left(|\frac{\partial I}{\partial x}(p) - \frac{\partial I'}{\partial x}(q)|, \tau_{\text{grad}}\right)$$

with

$$\|I(p)-I'(q)\|_1 = |I_r(p)-I_r(q)|+|I_g(p)-I_g(q)|+|I_b(p)-I_b(q)|$$

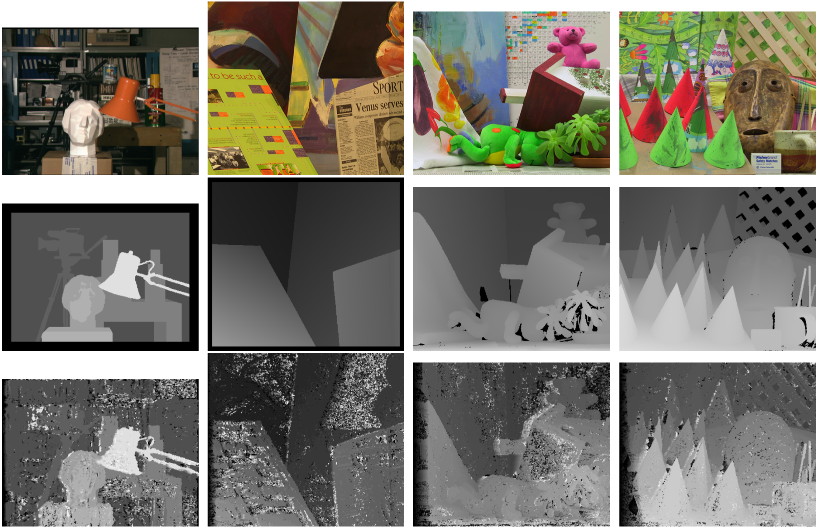
- Usual parameters:
  - $\alpha = 0.9$
  - $\tau_{\text{col}} = 30$ (not very sensitive if larger)
  - $\tau_{\text{grad}} = 2$ (not very sensitive if larger)
- Note that $\alpha = 0$ is similar to SAD.

Varying patch size



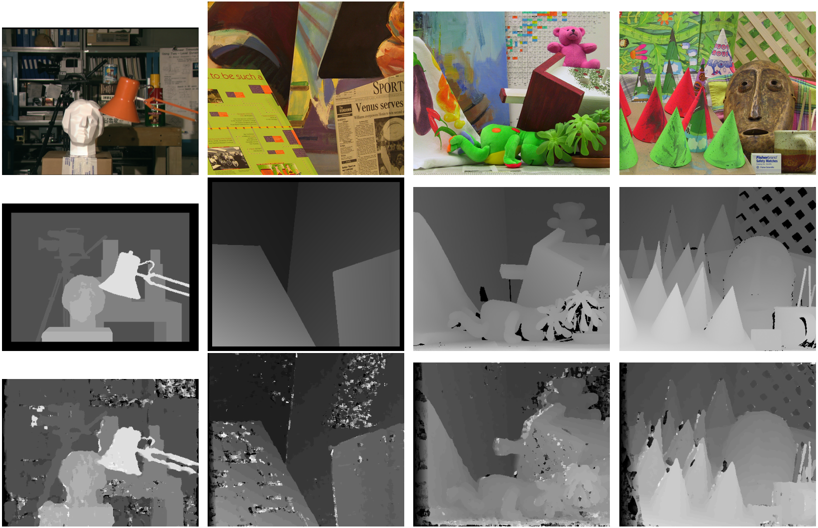
$$P = \{(0, 0)\}$$

Varying patch size



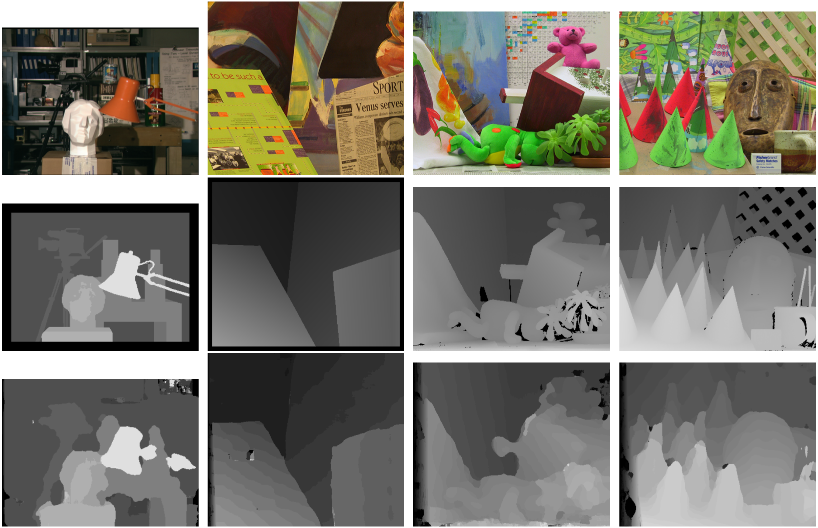
$$P = [-1, 1]^2$$

Varying patch size



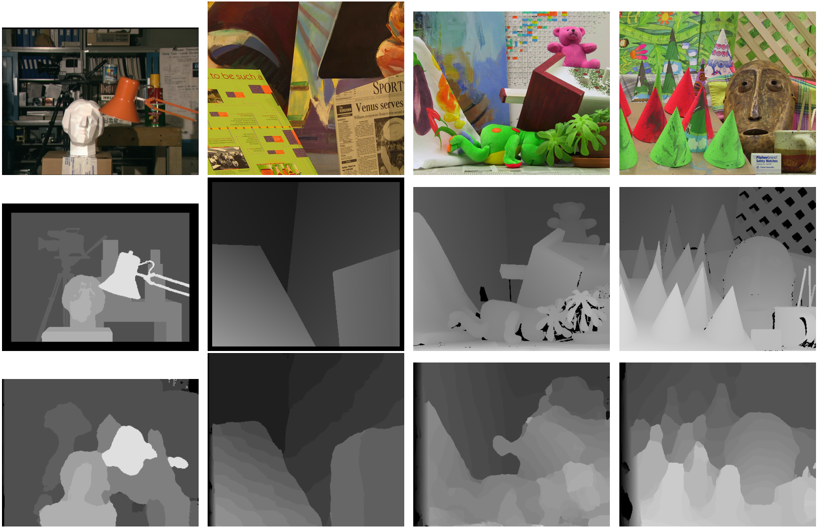
$$P = [-7, 7]^2$$

Varying patch size



$$P = [-21, 21]^2$$

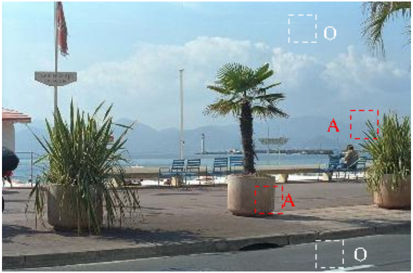
Varying patch size

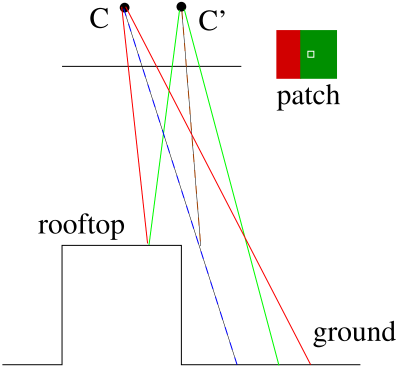


$$P = [-35, 35]^2$$

Problems of local methods

- Implicit hypothesis: all points of window move with same motion, that is they are in a fronto-parallel plane.
- aperture problem: the context can be too small in certain regions, lack of information.
- adherence problem: intensity discontinuities influence strongly the estimated disparity and if it corresponds with a depth discontinuity, we have a tendency to dilate the front object.





- O: aperture problem
- A: adherence problem

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Right image

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Left image

Example: seeds expansion

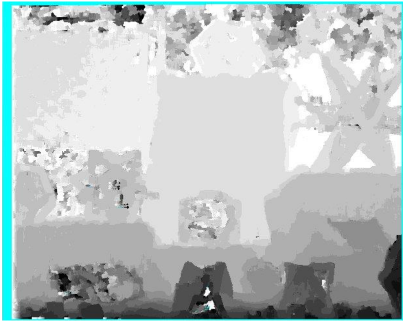
- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Seeds

Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Seeds expansion



Example: seeds expansion

- We rely on best found distances and we put them in a priority queue (seeds)
- We pop the best seed G from the queue, we compute for neighbors the best disparity between $d(G) - 1$, $d(G)$, and $d(G) + 1$ and we push them in the queue.

Left image

Adaptive neighborhoods

- To reduce adherence (aka fattening effect), an image patch should be on the same object, or even better at constant depth
- Heuristic inspired by bilateral filter [Yoon&Kweon 2006]:

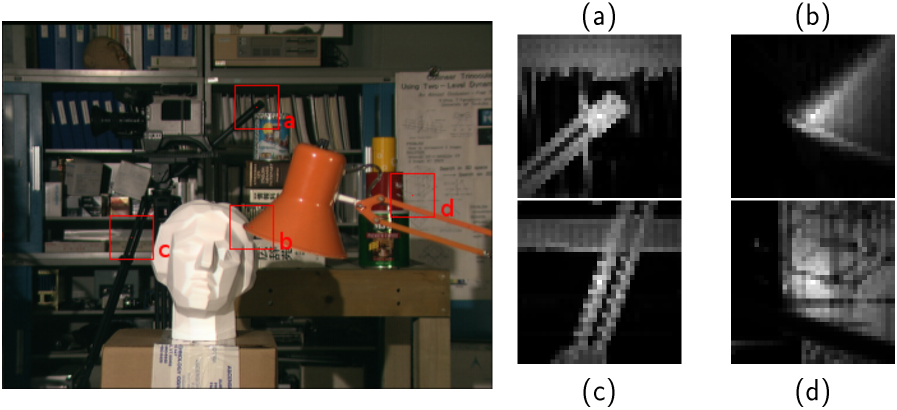
$$\omega_I(p,p') = \exp\left(-\frac{\|p-p'\|_2}{\gamma_{\text{pos}}}\right)\cdot \exp\left(-\frac{\|I(p)-I(p')\|_1}{\gamma_{\text{col}}}\right)$$

- Selected disparity:

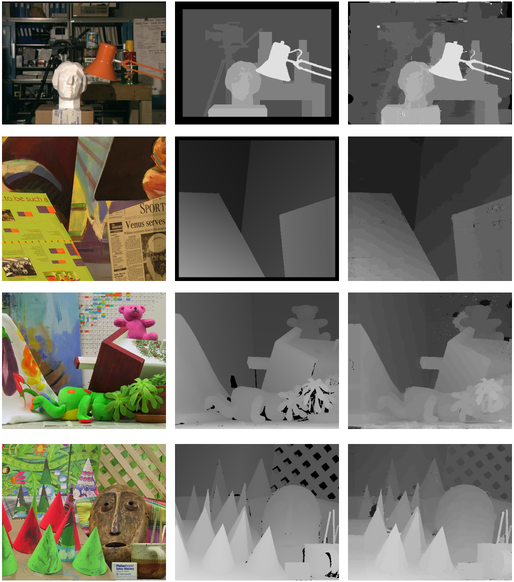
$$d(p) = \arg\min_{d=q-p} E(p,q) \text{ with}$$
$$E(p,q) = \frac{\sum_{r\in F}\omega_I(p,p+r)\,\omega_{I'}(q,q+r)\,\epsilon(p+r,q+r)}{\sum_{r\in F}\omega_I(p,p+r)\,\omega_{I'}(q,q+r)}$$

- We can take a large window F (e.g., 35×35)

Bilateral weights



Results



Tsukuba

Venus

Teddy

Cones

Left image Ground truth Results

What is the limit of adaptive neighborhoods?

- The best patch is $P_p(r) = 1(d(p+r) = d(p))$
- Suppose we have an oracle giving P_p
- Use ground-truth image to compute P_p
- Since GT is subpixel, use $P_p(r) = 1(|d(p+r) - d(p)| \leq 1/2)$

Test with oracle

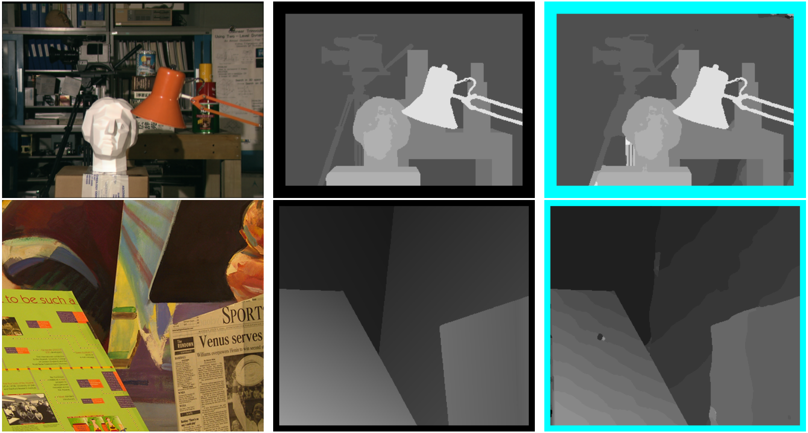
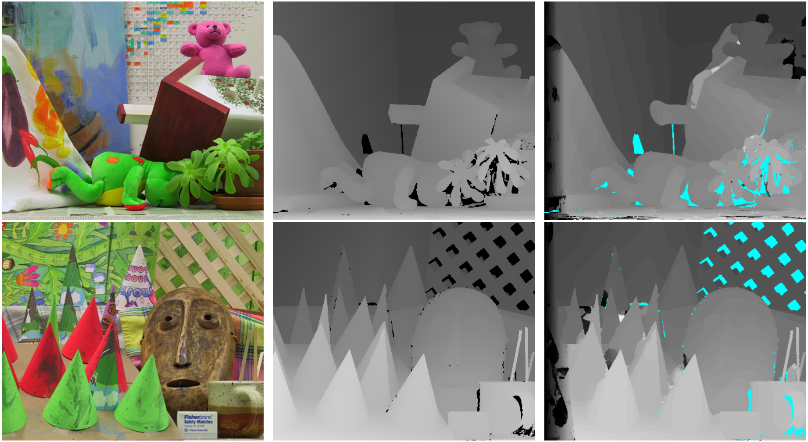


image ground truth oracle patches

Test with oracle



image

ground truth

oracle patches

Conclusion

- We can get back to the canonical situation by epipolar rectification. Limit: when epipoles are in the image, standard methods are not adapted.
- For disparity map computation, there are many choices:
  1. Size and shape of window?
  2. Which distance?
  3. Filtering of disparity map to reject uncertain disparities?
- You will see next session a *global* method for disparity computation
- Very active domain of research, >150 methods tested at `http://vision.middlebury.edu/stereo/`

Practical session: Disparity map computation by propagation of seeds

Objective: Compute the disparity map associated to a pair of images. We start from high confidence points (seeds), then expand by supposing that the disparity map is regular.

- Get initial program from the website.
- Compute disparity map from image 1 to 2 of all points by highest NCC score.
- Keep only disparity where NCC is sufficiently high (0.95), put them as seeds in a `std::priority_queue`.
- While queue is not empty:
  1. Pop P, the top of the queue.
  2. For each 4-neighbor Q of P having no valid disparity, set d_Q by highest NCC score among $d_P - 1$, d_P, and $d_P + 1$.
  3. Push Q in queue.

Hint: the program may be too slow in Debug mode for the full images. Use cropped images to find your bugs, then build in Release mode for original images.