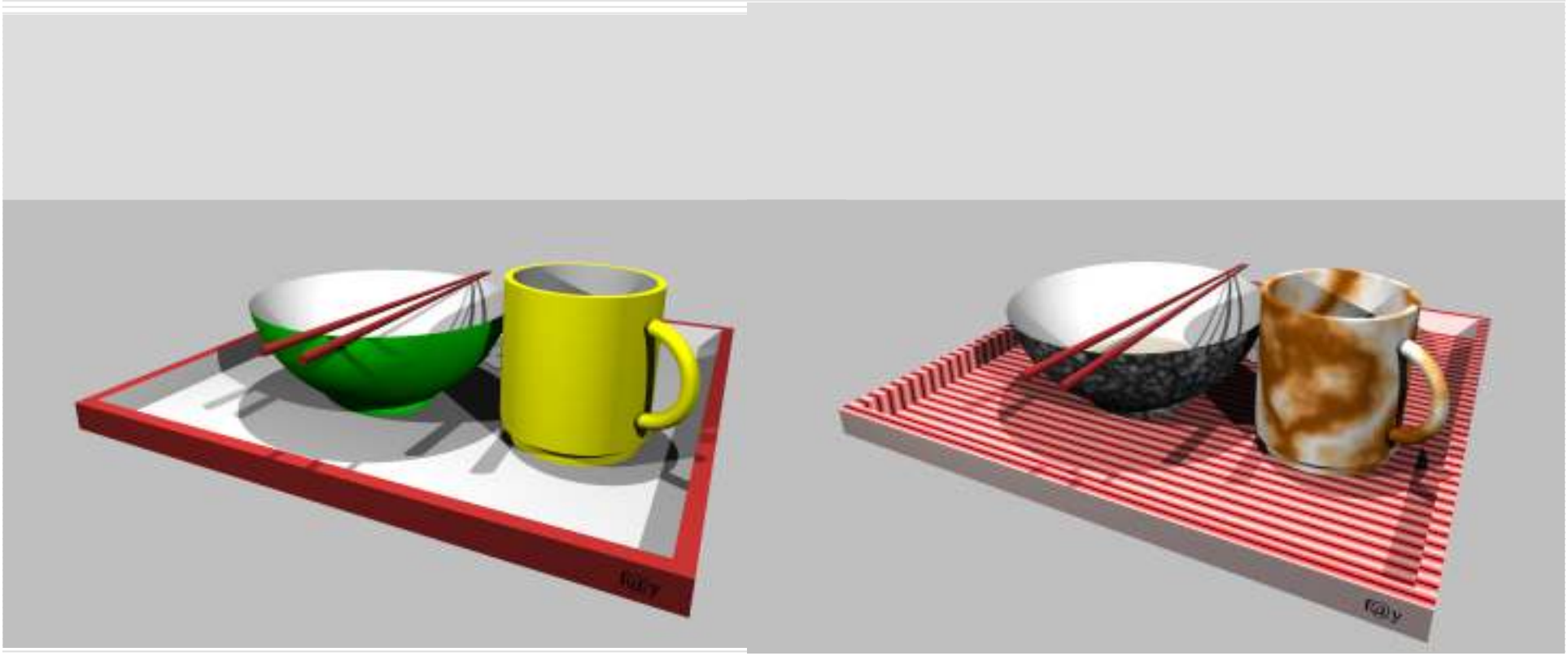
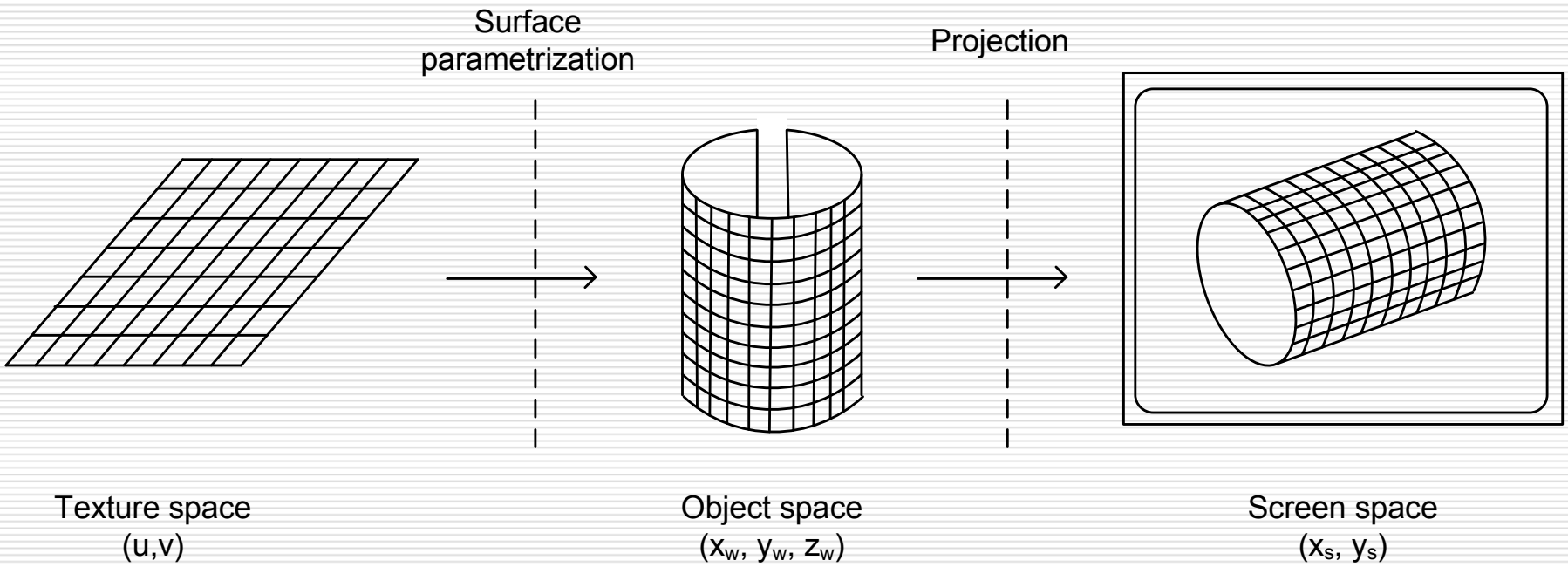
Contoh: tanpa & dengan texture mapping



Texture Mapping

- Memetakan peta tekstur 2D (2D *texture map*) ke permukaan objek kemudian memproyeksikannya ke bidang proyeksi *(projection plane)*
- Teknik:
  - Forward mapping (texture order alg)
  - Inverse mapping (screen order alg)

Forward mapping



Forward mapping (cont'd)

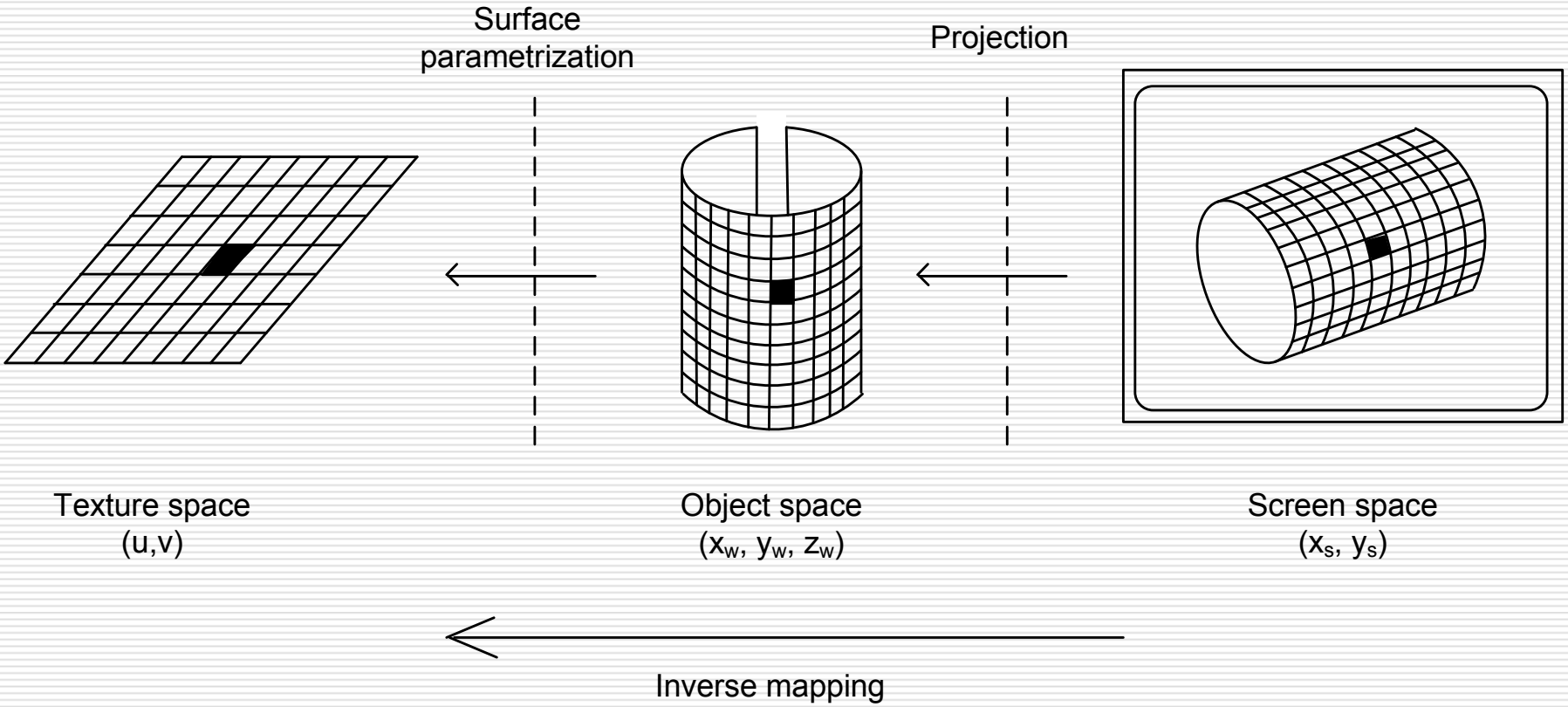
- Dispesifikasikan dengan fungsi linier parametrik:

$$x_s = f_{x_s}(u,v) = a_{x_s}u + b_{x_s}v + c_{x_s}$$

$$y_s = f_{y_s}(u,v) = a_{y_s}u + b_{y_s}v + c_{y_s}$$

- Object-to-image space mapping dilakukan dengan transformasi: viewing - projection
- Kekurangan: ukuran texture patch seringkali tidak sesuai dengan batas pixel, sehingga harus ada perhitungan untuk pemotongan

Inverse mapping



Inverse mapping (cont'd)

- Pada prakteknya, inverse mapping lebih sering digunakan
- Metoda:
  - Interpolasi bilinear
  - Memanfaatkan permukaan antara

Inverse mapping dgn interpolasi bilinear

- Dapat dibayangkan sebagai transformasi dari 2D screen space (x,y) ke 2D texture space (u,v)
- Operasi image warping, dimodelkan dengan:

$$x = \frac{au + bv + c}{gu + hu + i}; \qquad y = \frac{du + ev + f}{gu + hu + i}$$

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} u' \\ v' \\ q \end{bmatrix}$$

$$(x, y) = (x'/w, y'/w); \qquad (u, v) = (u'/q, v'/q)$$

The inverse transform

$$\begin{bmatrix} u' \\ v' \\ q \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} x' \\ y' \\ w \end{bmatrix}$$

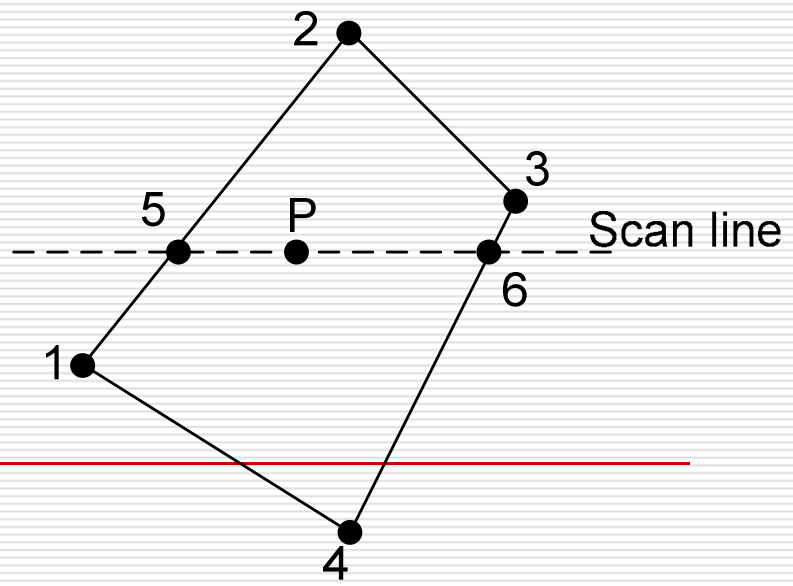
$$A^{-1} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}^{-1} = \frac{1}{|A|} \begin{bmatrix} ei - fh & ch - bi & bf - ce \\ fg - di & ai - cg & cd - af \\ dh - eg & bg - ah & ae - bd \end{bmatrix}$$

$$\begin{bmatrix} u' \\ v' \\ q \end{bmatrix} = \begin{bmatrix} ei - fh & ch - bi & bf - ce \\ fg - di & ai - cg & cd - af \\ dh - eg & bg - ah & ae - bd \end{bmatrix} \begin{bmatrix} x' \\ y' \\ w \end{bmatrix}$$

- Hubungan antara titik sudut poligon dengan koordinat pada texture map dispesifikasikan pada fase pemodelan $|A| = a(ei - fh) - b(di - fg) + c(dh - eg)$
  - Dengan empat titik sudut quadrilateral, bisa didapat 9 koefisien (a,b,c,d,e,f,g,h,i) → Gaussian elimination

Interpolasi bilinear pada screen space

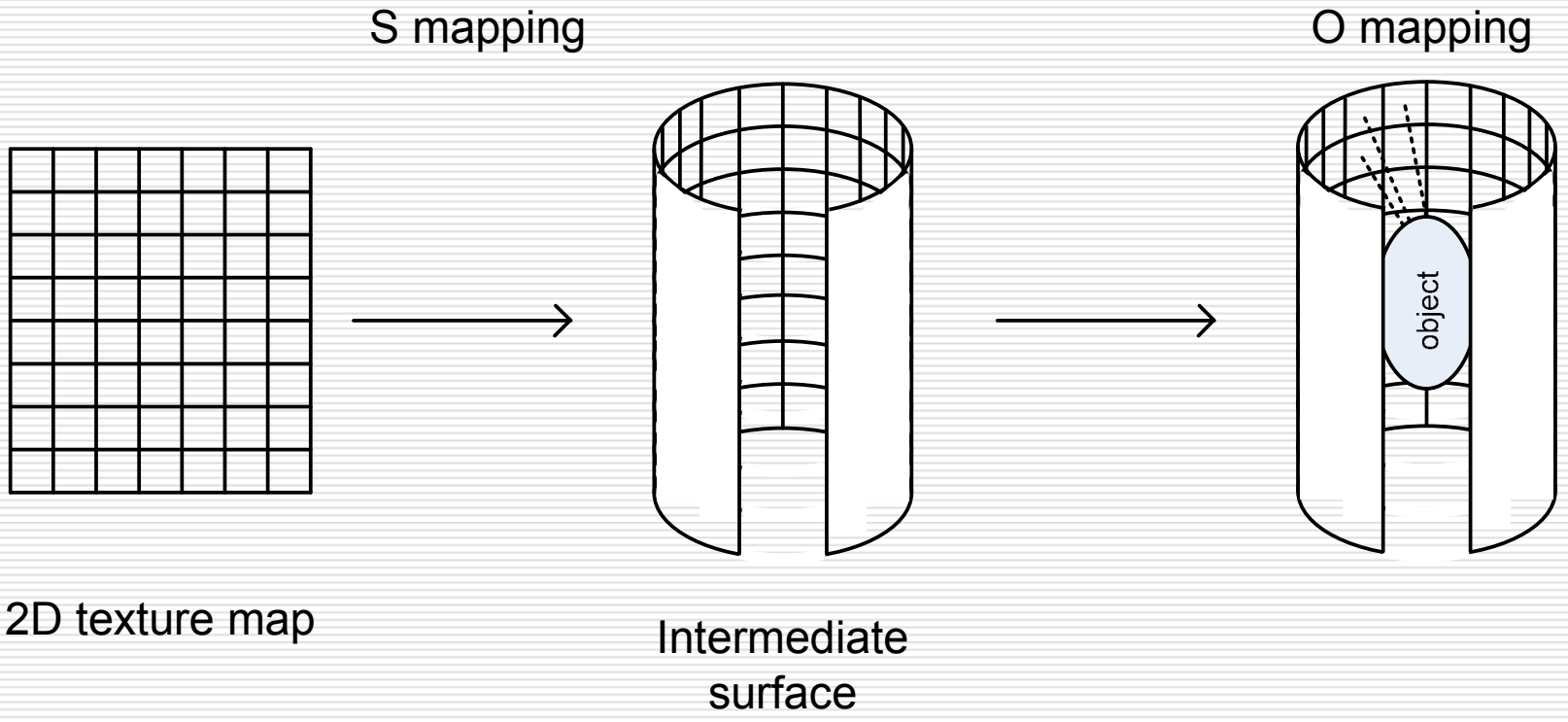
- Tiap koordinat vertex punya koordinat texture (u’,v’,q)
- Yang diinterpolasikan: (u’,v’,q)
  - (u,v) tidak berubah secara linear thd (x,y)
- (u,v) = (u’/q,v’/q)



Inverse mapping dengan penggunaan permukaan antara

- Bisa digunakan jika belum ada hubungan antara koordinat vertex dan texture
  - Digunakan untuk menentukan hubungan tsb
- Two-part mapping:
  - Texture dipetakan ke permukaan antara (biasanya non-planar)
  - Baru kemudian dipetakan ke objek (3D-to-3D mapping)

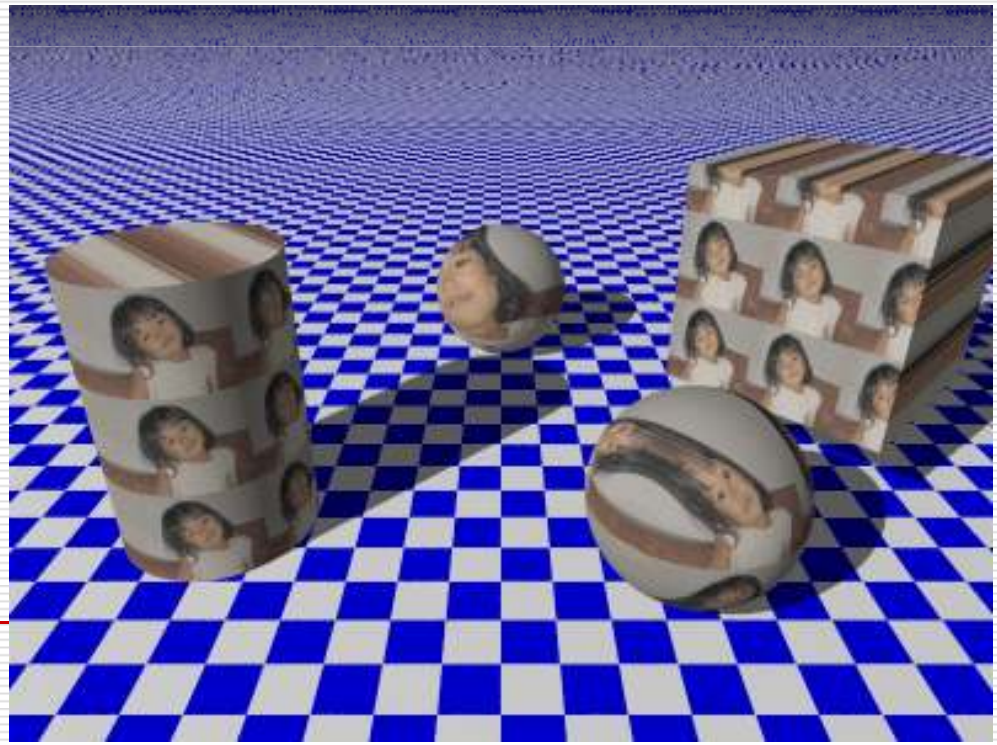
Two-part mapping



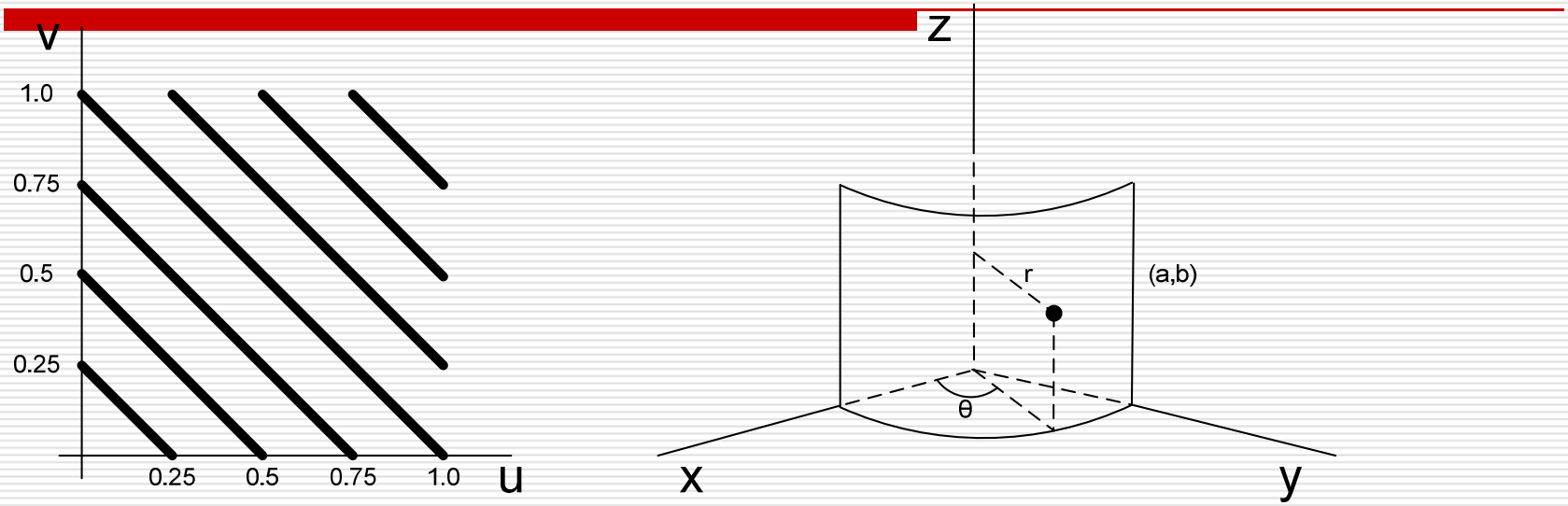
- S mapping: $T(u,v) \rightarrow T'(x_i,y_i,z_i)$
- O mapping: $T'(x_i,y_i,z_i) \rightarrow O(x_w,y_w,z_w)$

Jenis permukaan antara

- Bidang (dengan berbagai orientasi)
- Permukaan lengkung dari silinder
- Permukaan kubus
- Permukaan bola



Ilustrasi: transfer pola ke permukaan ¼ silinder

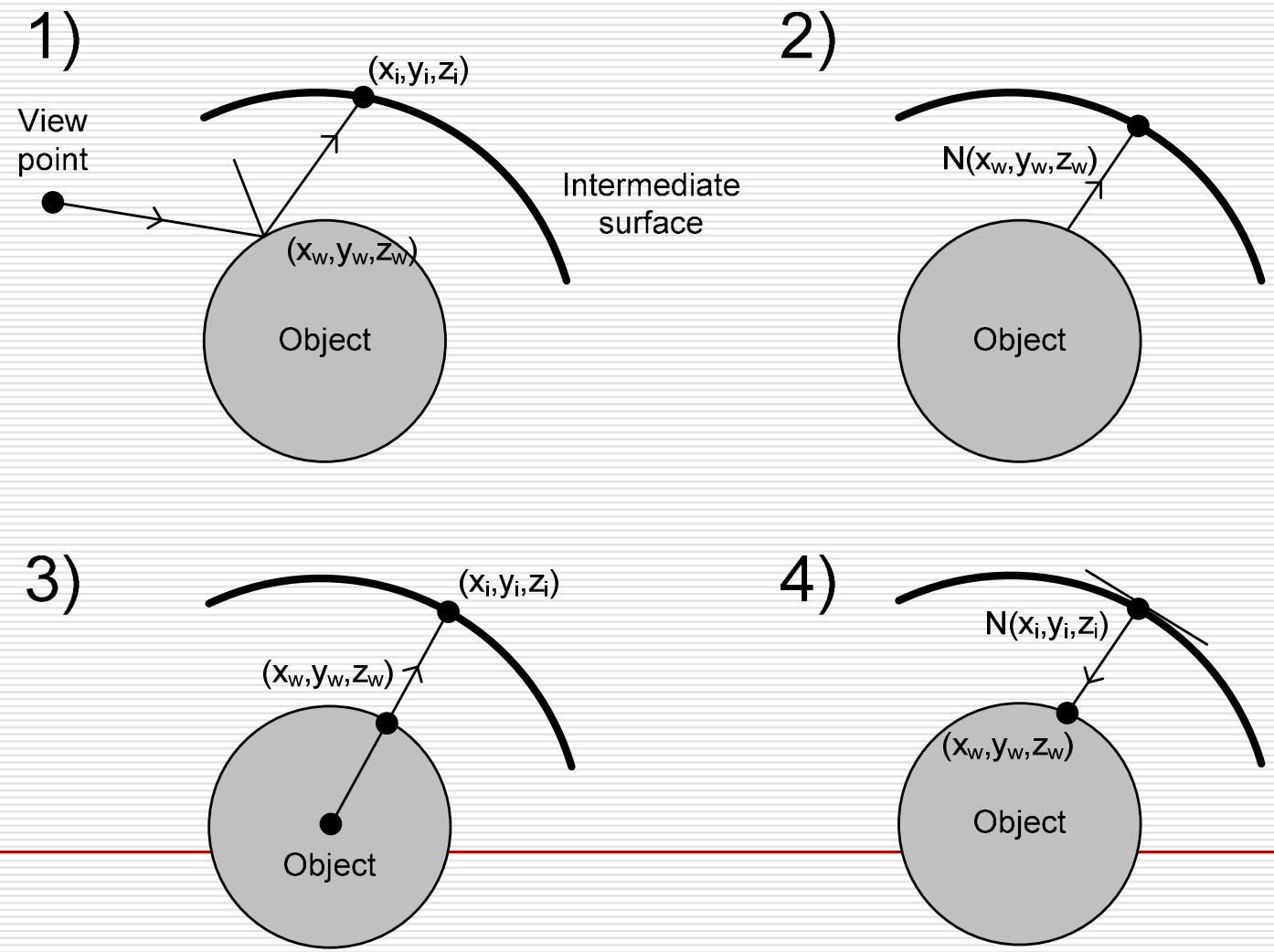


- Permukaan: $a = \theta$, $b = z$; $0 \le \theta \le \pi/2$, $0 \le z \le 1$
- $x_i = r \cos a$; $y_i = r \sin a$; $z_i = b$
- $a = u\pi/2$, $b = v$
- Inverse:
  - $a = \tan^{-1}(y_i/x_i)$, $b = z_i$
  - $u = 2a/\pi$, $v = b$

O mapping

- $T'(x_i,y_i,z_i) \rightarrow O(x_w,y_w,z_w)$
- Lebih mudah: sudut pandang ray tracing
- Empat kemungkinan:
  1. Perpotongan pantulan view ray dengan permukaan antara
  2. Perpotongan vektor normal dari permukaan pada (x_w,y_w,z_w) dengan permukaan antara
  3. Perpotongan garis yang menghubungkan pusat objek dan (x_w,y_w,z_w) dengan permukaan antara
  4. Perpotongan antara garis yang menghubungkan (x_w,y_w,z_w) dengan (x_i,y_i,z_i), yang orientasinya sama dengan vektor normal pada (x_i,y_i,z_i)

4 kemungkinan O mapping

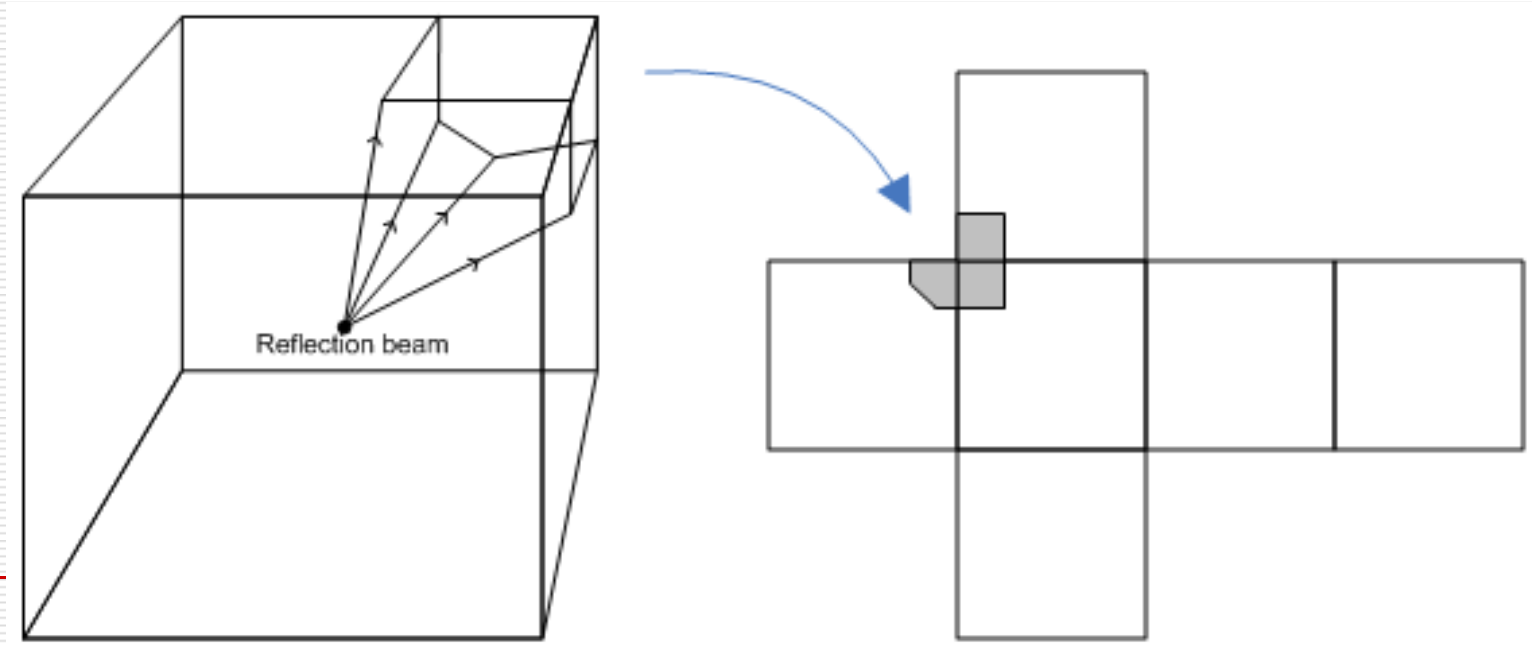


Environment mapping

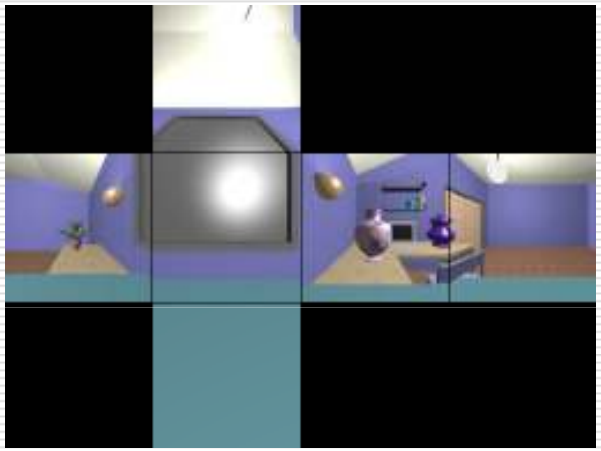
- Cara cepat untuk me-render objek mengkilap yang memantulkan cahaya dari sekelilingnya
- Batasan environment mapping:
  - Benar secara geometris jika objek relatif kecil dibanding lingkungannya
  - Objek hanya dapat memantulkan cahaya dari lingkungannya → tidak untuk objek concave
  - Setiap objek dalam scene membutuhkan environment map yang berbeda
  - Kadang diperlukan environment map yang baru jika sudut pandang berubah

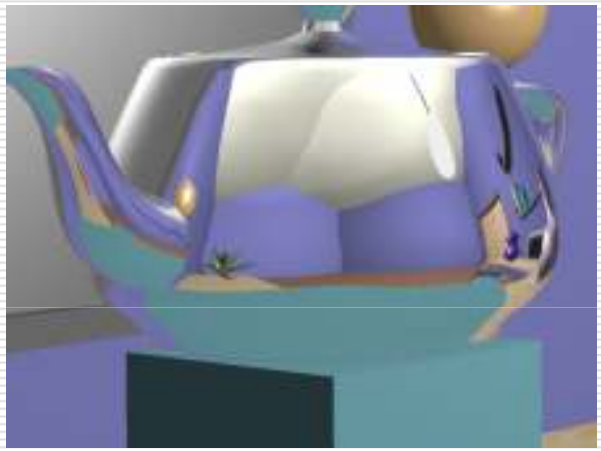
Cubic mapping

- Environment diwakili oleh 6 sisi kubus
- Titik pandang berada di pusat objek

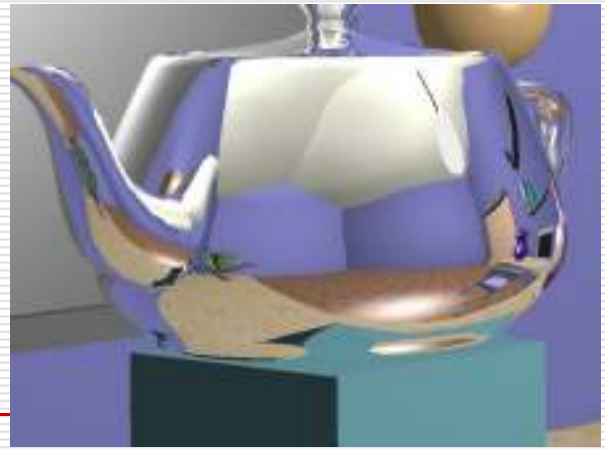


Contoh cubic mapping

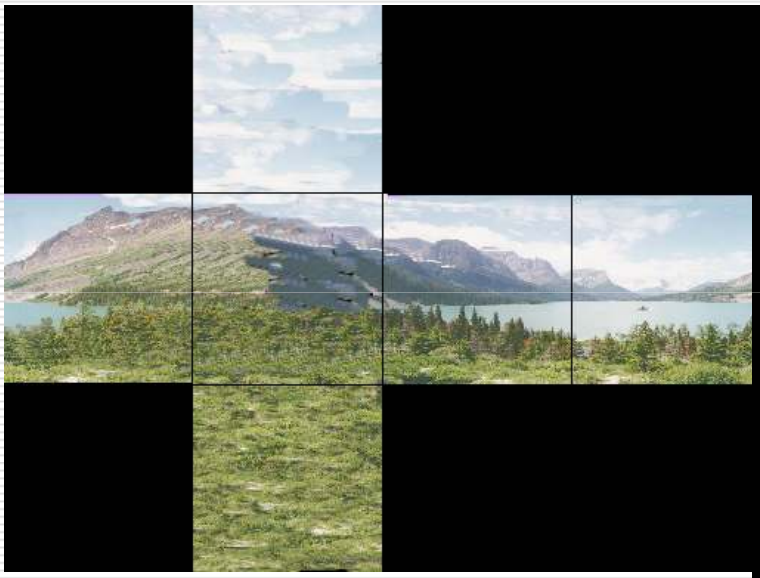




Bandingkan dengan
hasil Ray Tracing



Cubic mapping dengan real images



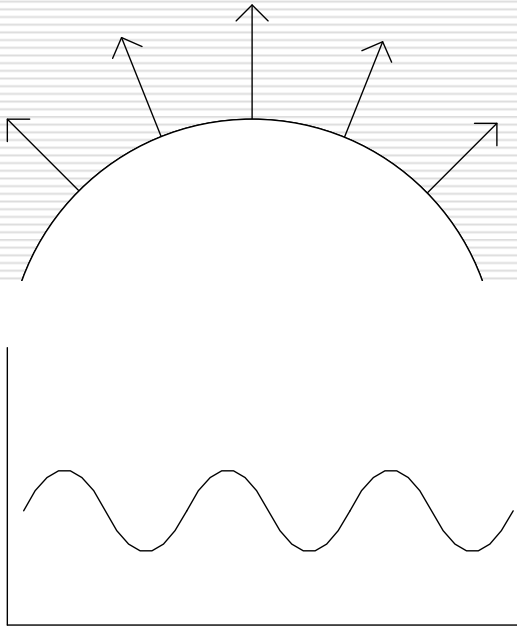
Environment map

Hasil mapping

Bump mapping

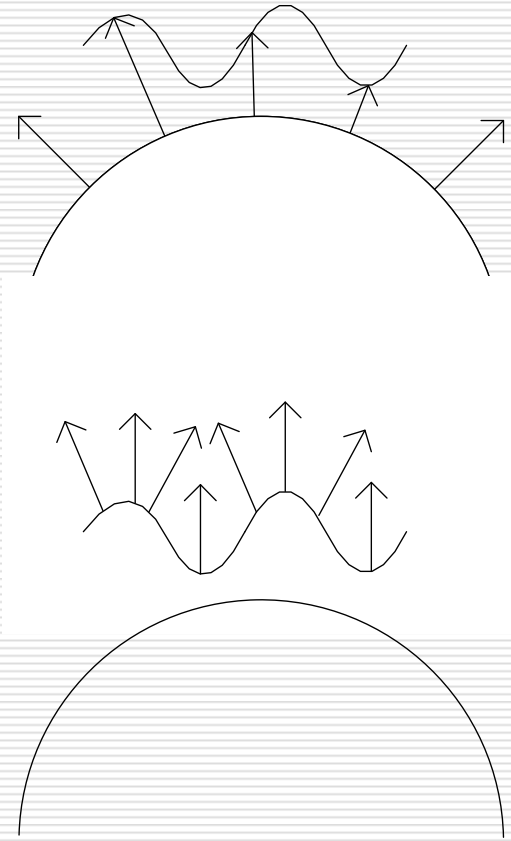
- Permukaan tampak berkerut tanpa harus memodelkannya secara geometris
- Vektor normal dari permukaan dimodifikasi secara angular → mempengaruhi model pantulan cahaya
- Kekurangan: garis siluet tetap mengikuti bentuk asli objek

Ilustrasi prosedur bump mapping



P(u):
Original
surface

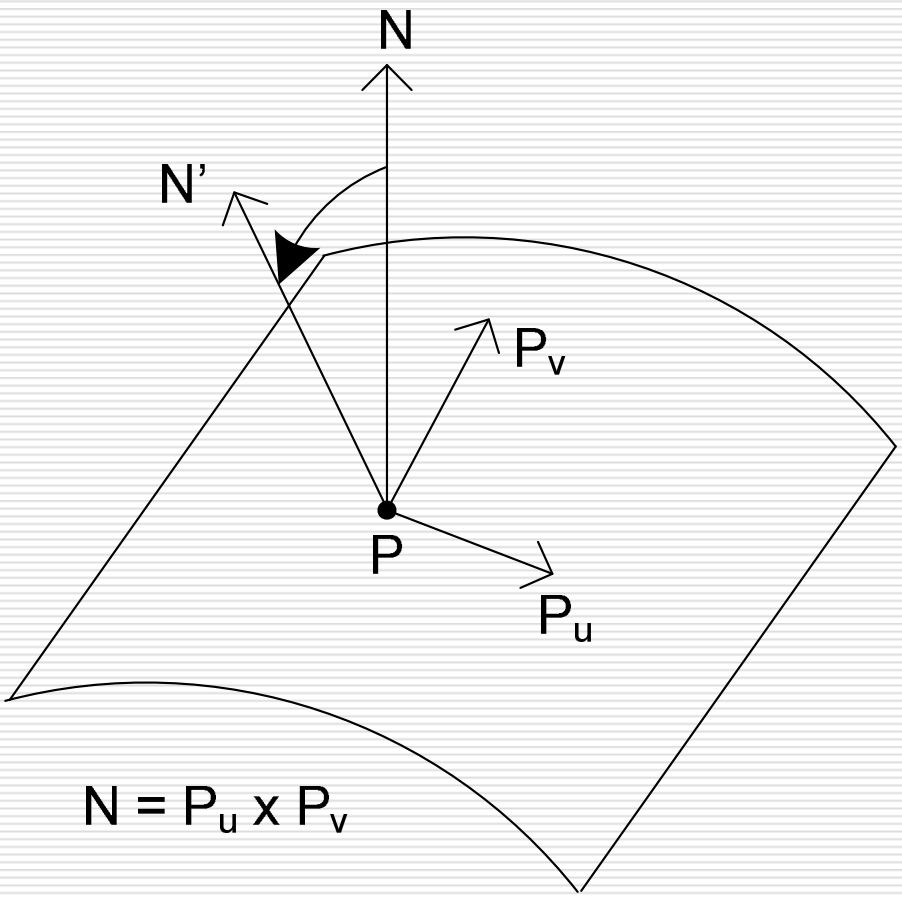
B(u):
Bump map

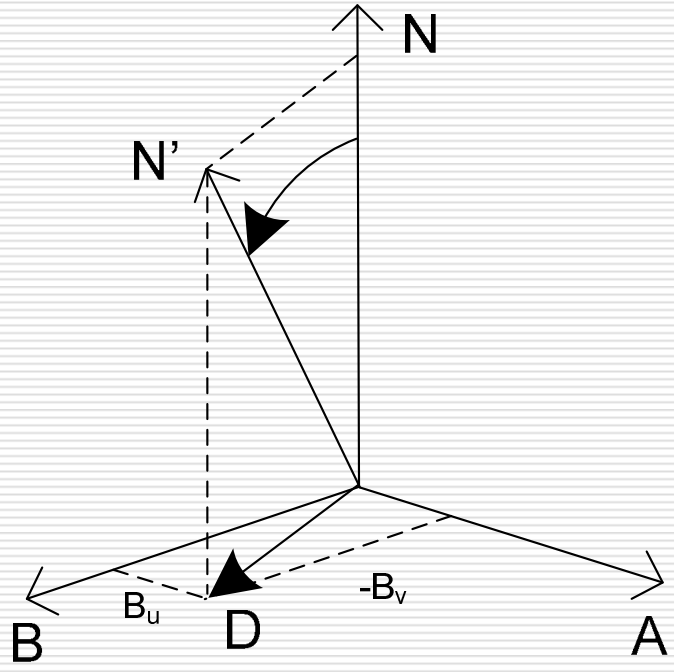


P’(u):
Modifying P(u)
with B(u)

N’(u): The
vectors to the
new ‘surface’

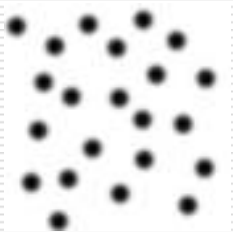
Interpretasi geometris bump mapping



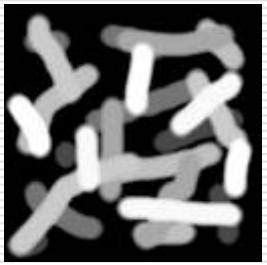


Contoh bump mapping

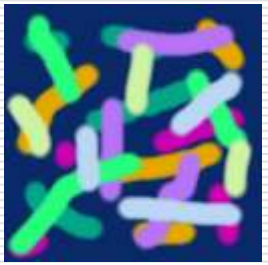
Contoh bump mapping (cont'd)



Bump map



Bump map

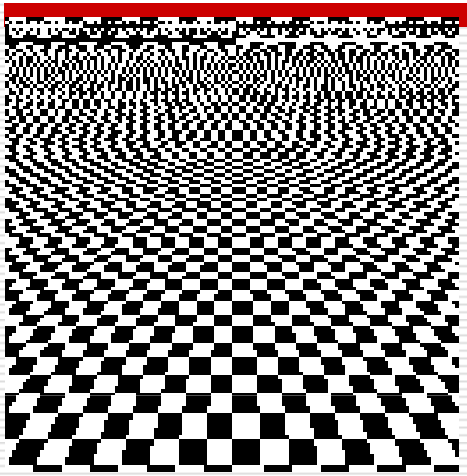


Texture map

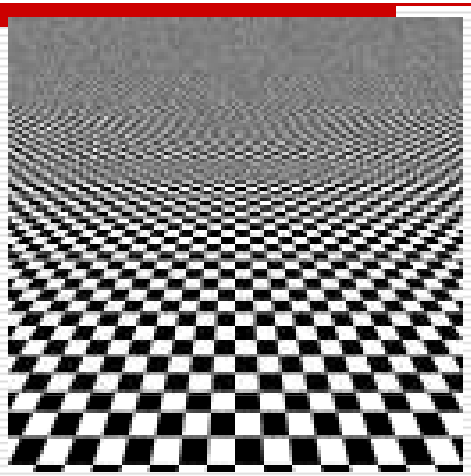
Anti-aliasing

- ‘Alias’: gejala pada image (paling terlihat pada texture map) dimana perulangan tekstur mencapai dimensi pixel
  - Under-sampling: menghitung satu warna untuk tiap pixel
- Anti-aliasing: mengurangi efek alias

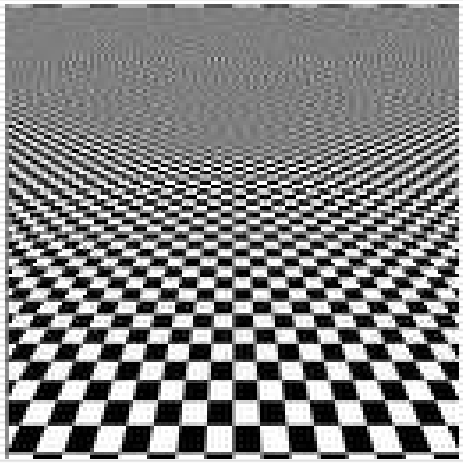
Contoh alias & antialiasing



a) Aliased
[Moiré effect]

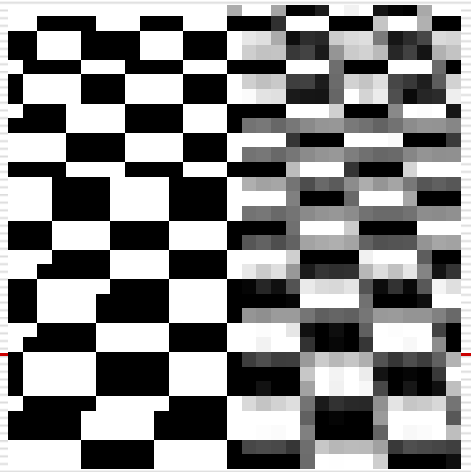


b) Anti-aliased



c) Anti-aliased-sinc

Zoom a)



Zoom c)

Prinsip pendekatan anti-aliasing

- Image ideal memiliki detil yang tak berhingga → direpresentasikan dengan fungsi *f*(x,y) dimana x dan y adalah bilangan real yang mendefinisikan koordinat.
- Untuk dapat menampilkan image *f(x,y)* dengan keterbatasan perangkat display, harus disederhanakan
  - Cara paling sederhana: pencuplikan image pada *f*(*i*,*j*) untuk tiap pixel (*i*,*j*) → Moiré effect
  - Cara yang lebih baik: untuk tiap pixel *(i,j)*, gunakan intensitas rata-rata dari area pada permukaan dalam scene yang memuat *f(i,j)*
  - Cara-cara lain yang lebih kompleks

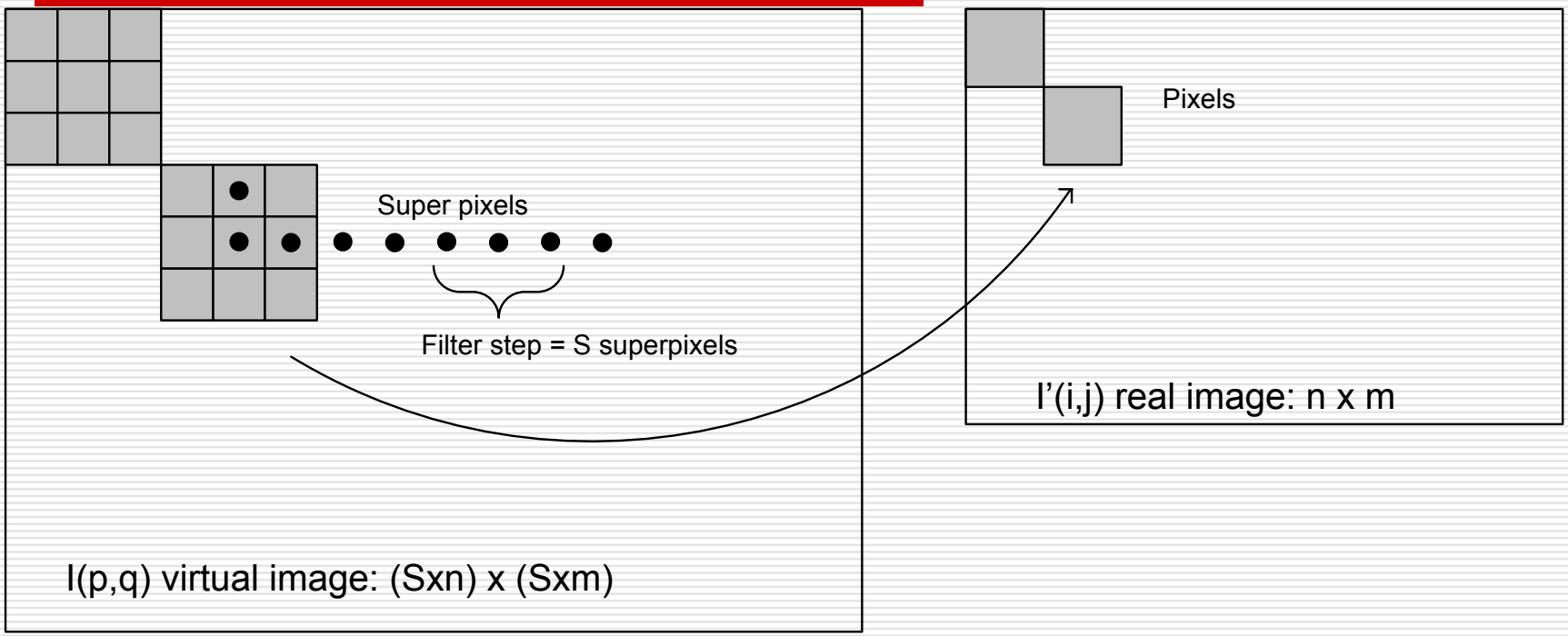
Anti-aliasing options

- Pre-filtering – ‘infinite’ samples per pixel
- No filtering – one sample per pixel
- Post-filtering – n uniform samples per pixel
- Post-filtering – stochastic samples

Post-filtering: supersampling

- Teknik yang paling sering digunakan untuk rendering polygon mesh
- Memperbesar frekuensi sampling
- Perhitungan melibatkan:
  - virtual image dengan resolusi spasial yang lebih besar daripada image yang akan ditampilkan
  - ‘down averaging’ image dengan resolusi tinggi ke resolusi yang akan ditampilkan

Supersampling & 'down averaging'



$$I'(i,j) = \sum_{p=Si-k}^{Si+k} \sum_{q=Sj-k}^{Sj+k} I(p,q)h(Si-p, Sj-q)$$

S: faktor skala; h: filter; k: dimensi filter

Contoh filter (h)

- Bartlett window (3 diantaranya)

3 x 3

1	2	1
2	4	2
1	2	1

5 x 5

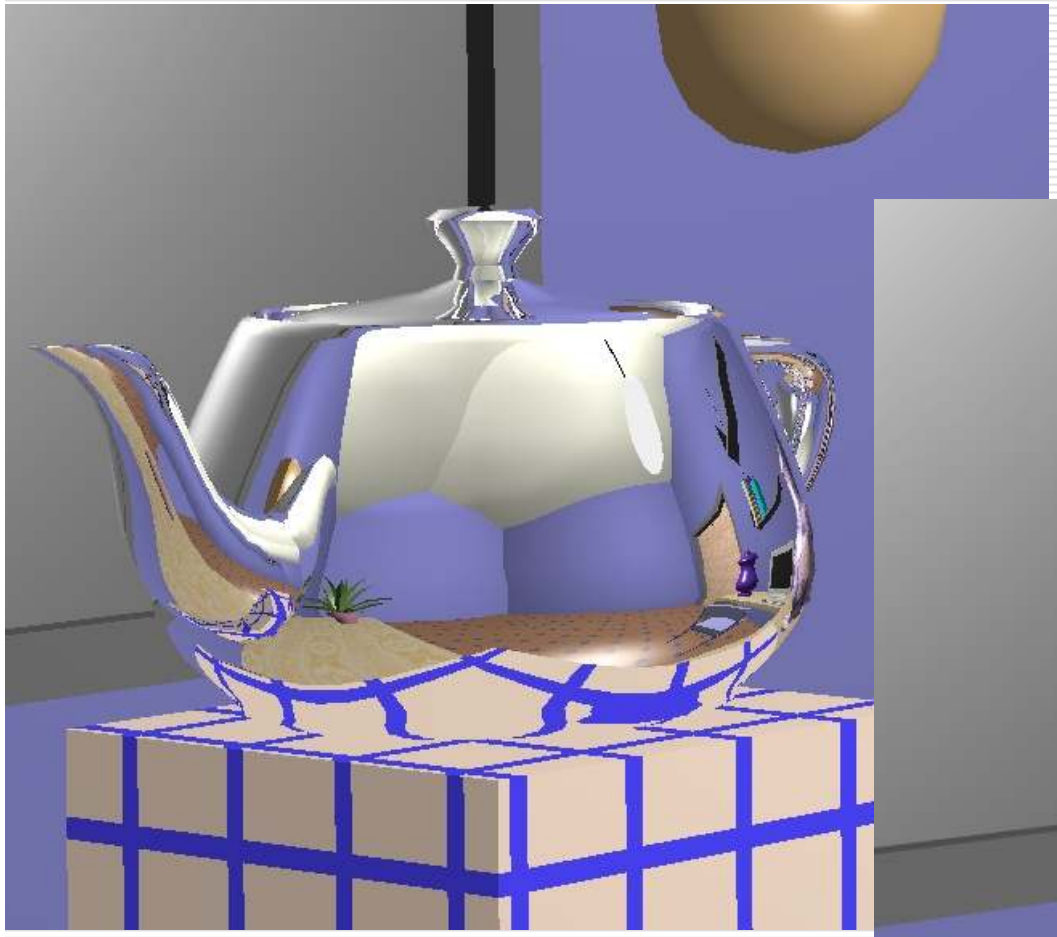
1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

7 x 7

1	2	3	4	3	2	1
2	4	6	8	6	4	2
3	6	9	12	9	6	3
4	8	12	16	12	8	4
3	6	9	12	9	6	3
2	4	6	8	6	4	2
1	2	3	4	3	2	1

- Semakin besar dimensi filter: semakin bagus menangani aliasing, semakin blur → trade-off

Contoh supersampling



Tanpa anti-aliasing

Anti-aliasing
virtual image 2x real image