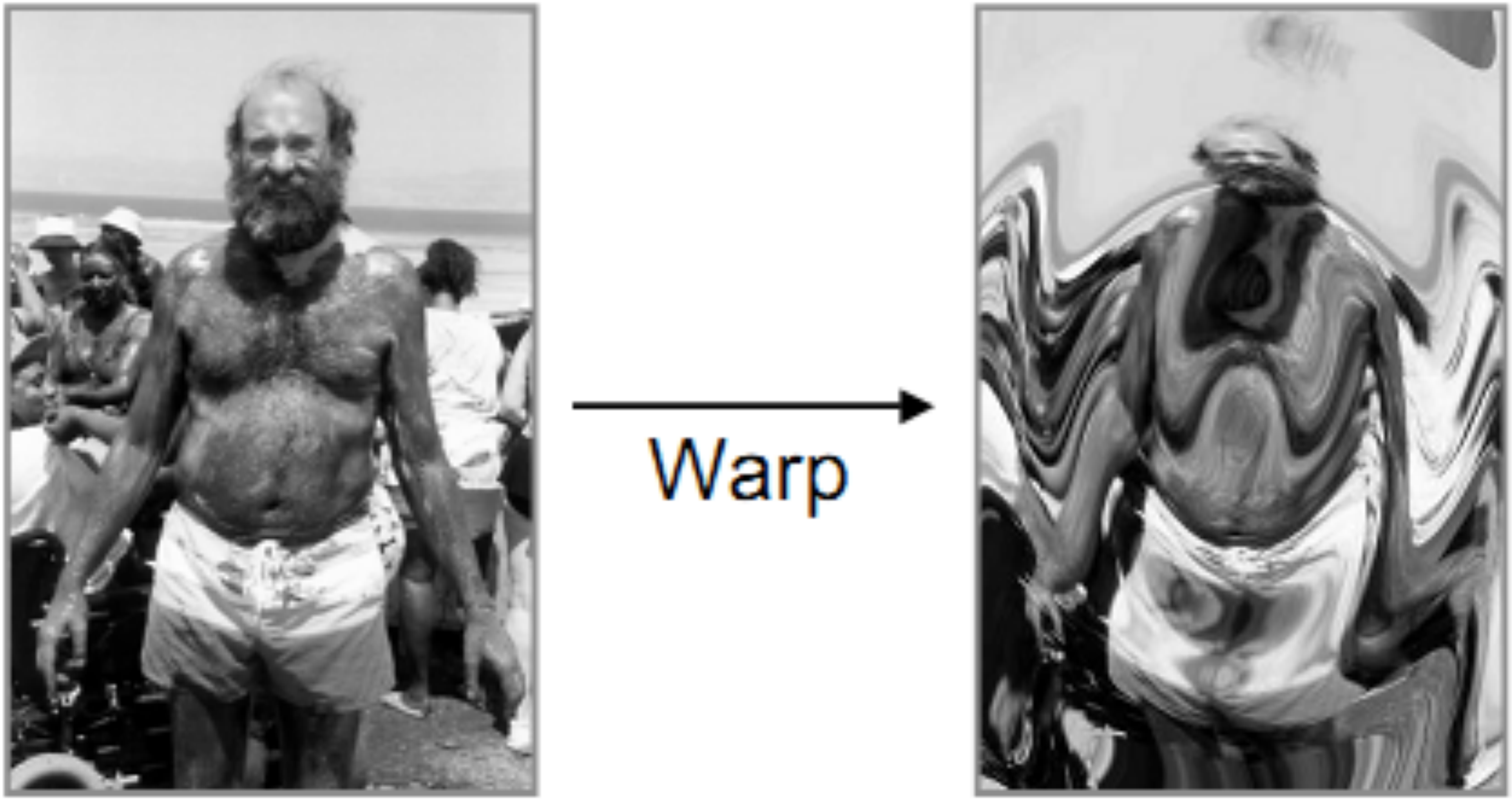
Image Warping



Source image

Destination image

Image Mapping

- Define transformation
  - Describe the destination (x,y) for every location (u,v) in the source (or vice-versa, if invertible)

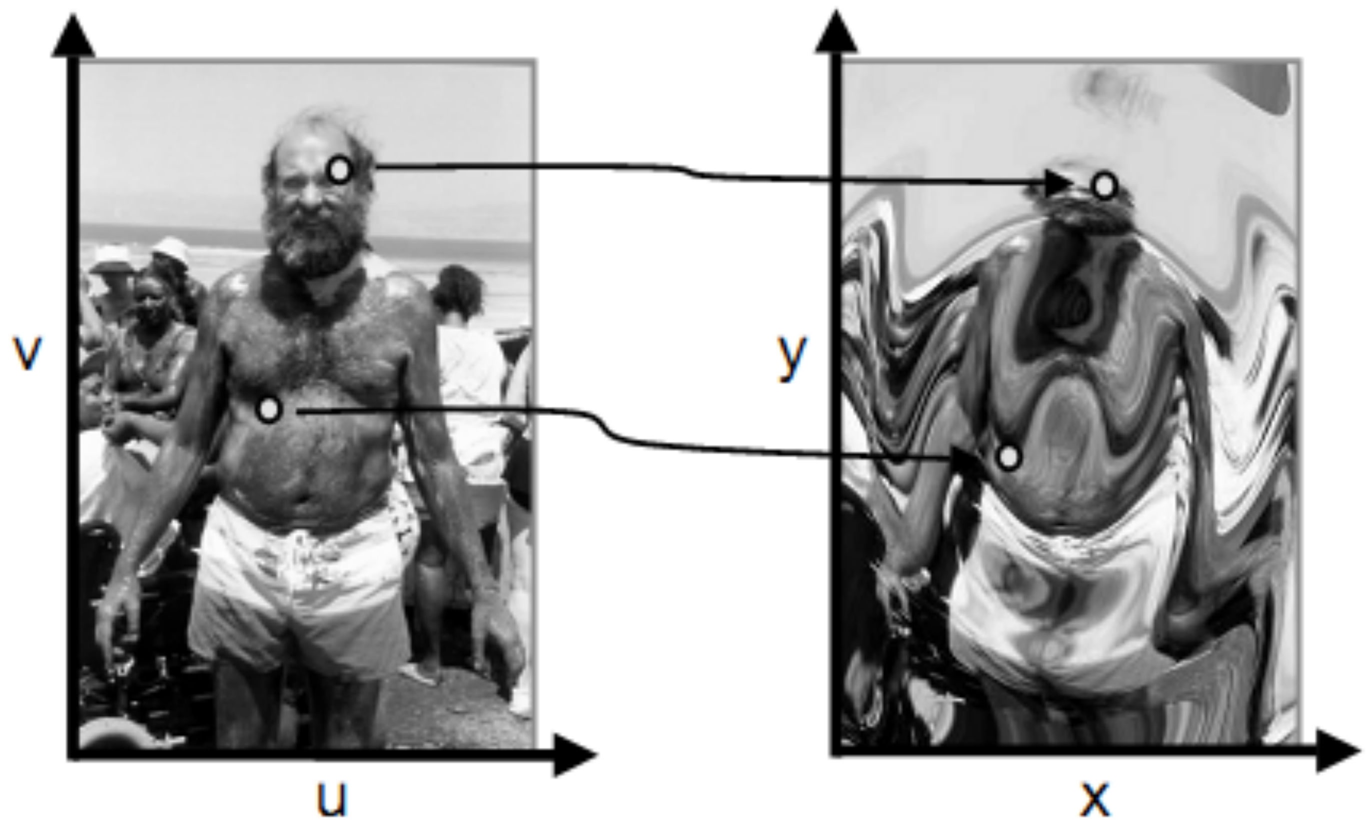
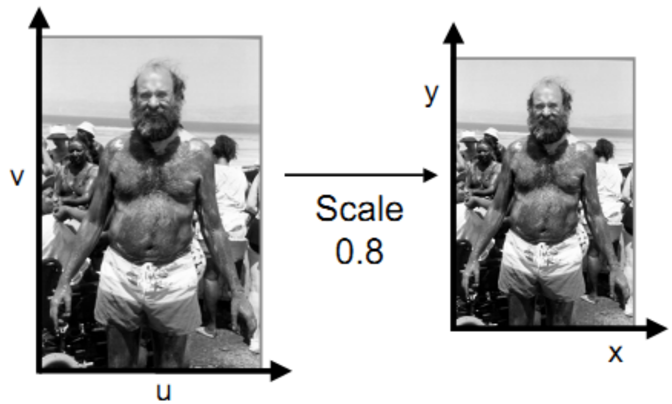
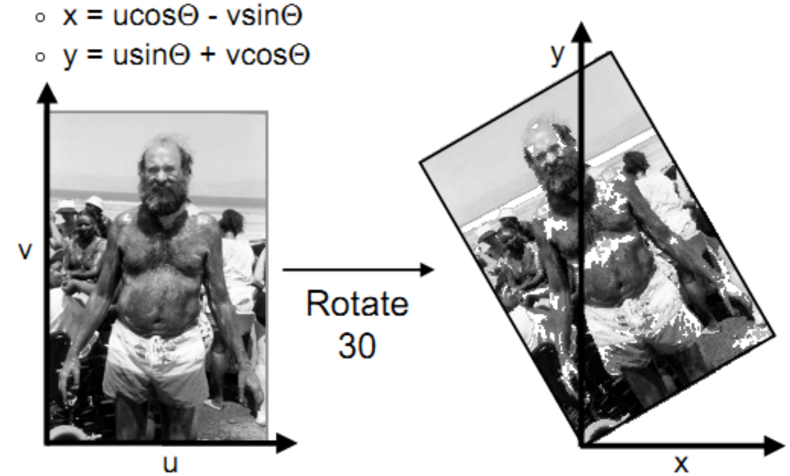


Image Mapping - Examples

- Scale by *factor*:
  - x = *factor* * u
  - y = *factor* * v



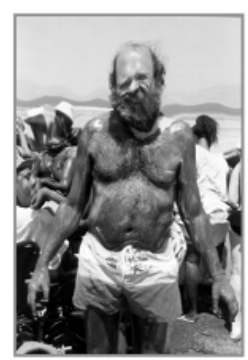
- Rotate by Θ degrees:
  - x = ucosΘ - vsinΘ
  - y = usinΘ + vcosΘ



- Any function of u and v:
  - $x = f_x(u,v)$
  - $y = f_y(u,v)$

Fish-eye

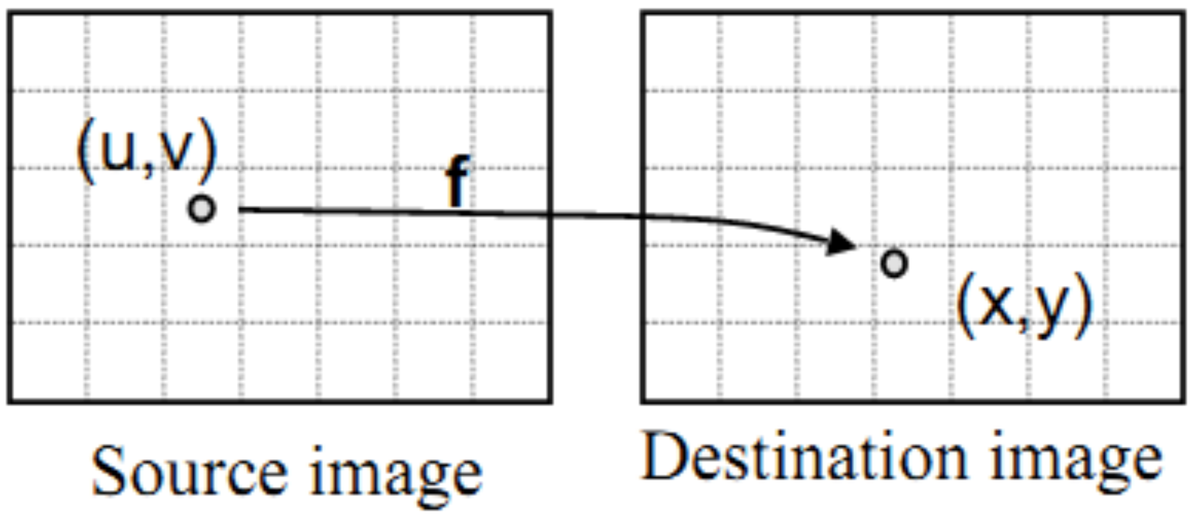
“Swirl”



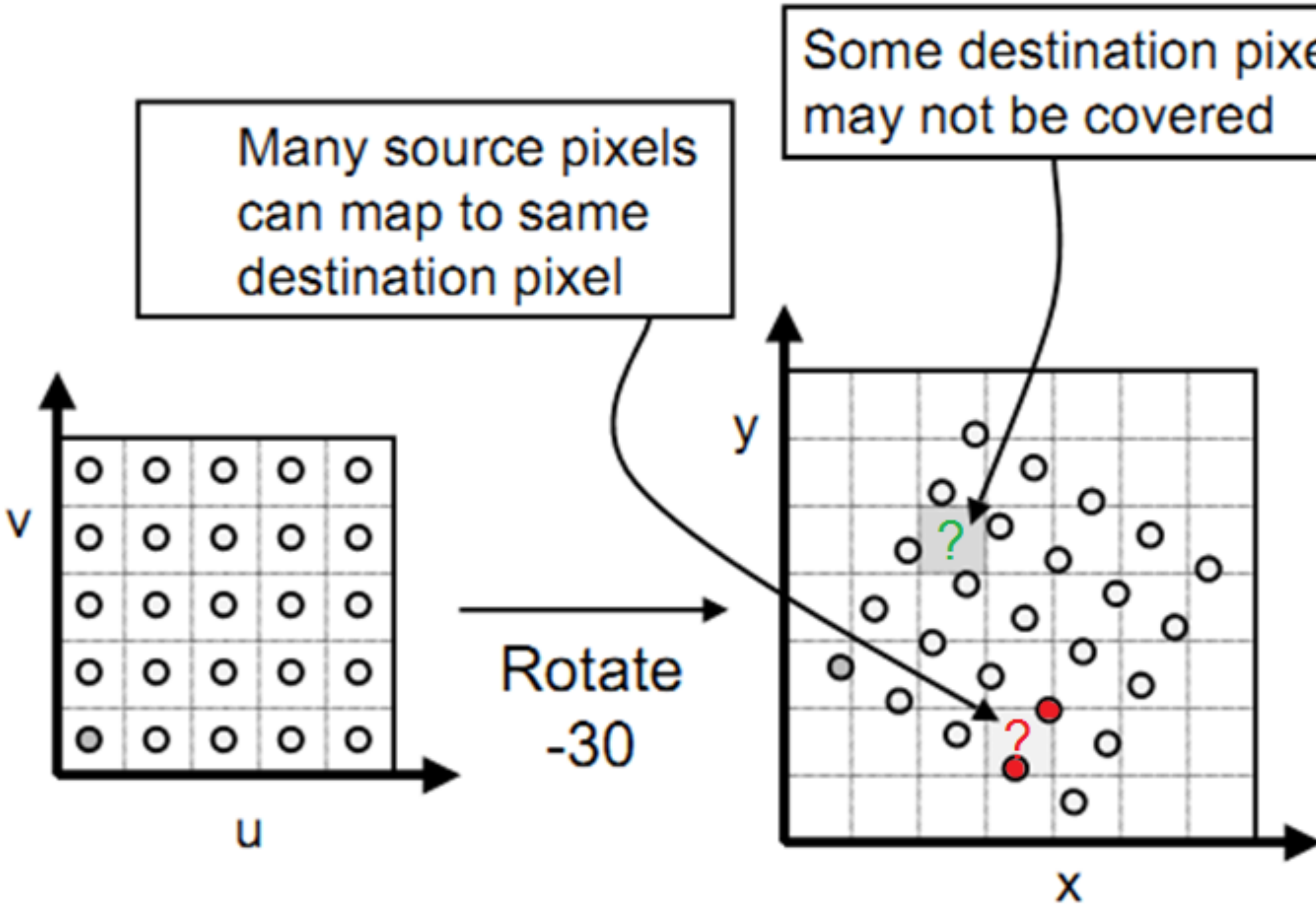
“Rain”

Forward Mapping

```
for (int u = 0; u < umax; u++) {
  for (int v = 0; v < vmax; v++) {
    float x = fx(u,v);
    float y = fy(u,v);
    dst(x,y) = src(u,v);
  }
}
```

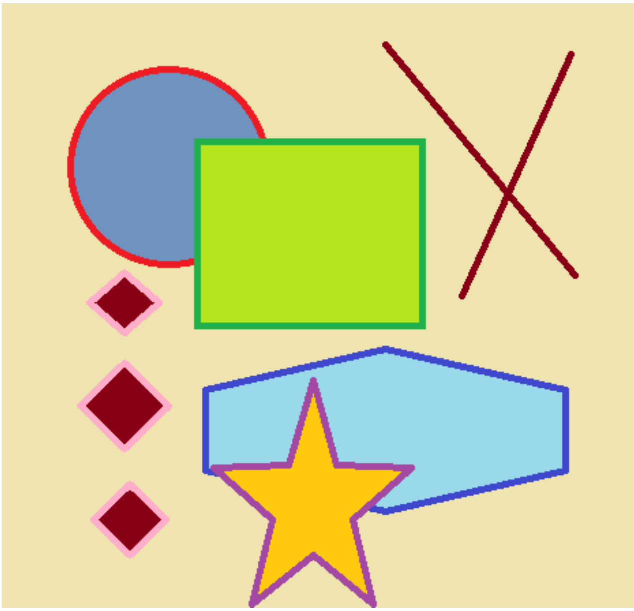


Forward Mapping - Disadvantages

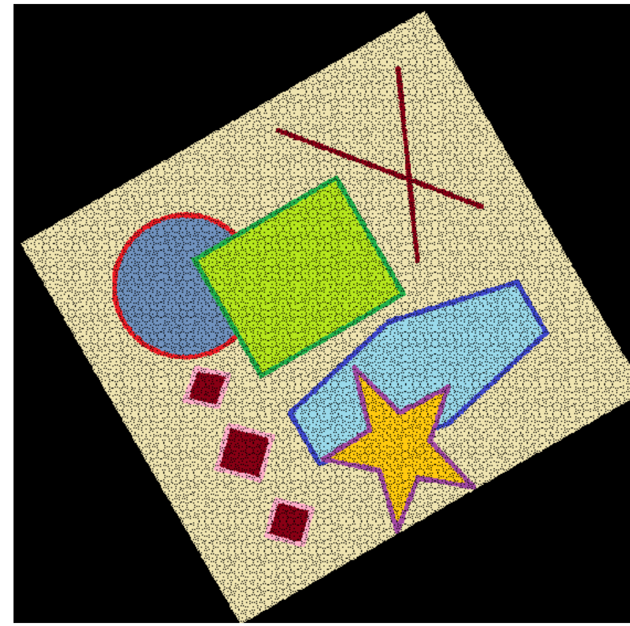


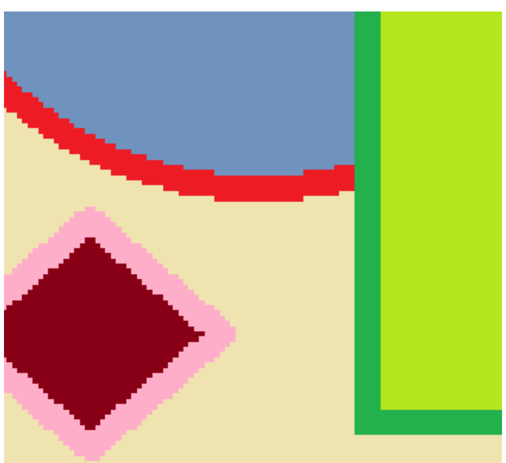
Example – Forward Mapping

Original

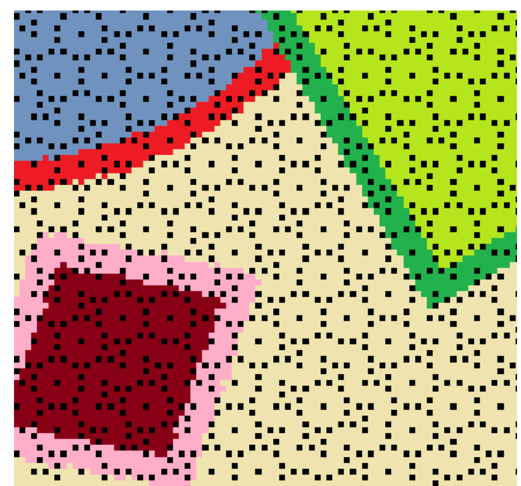


Rotated





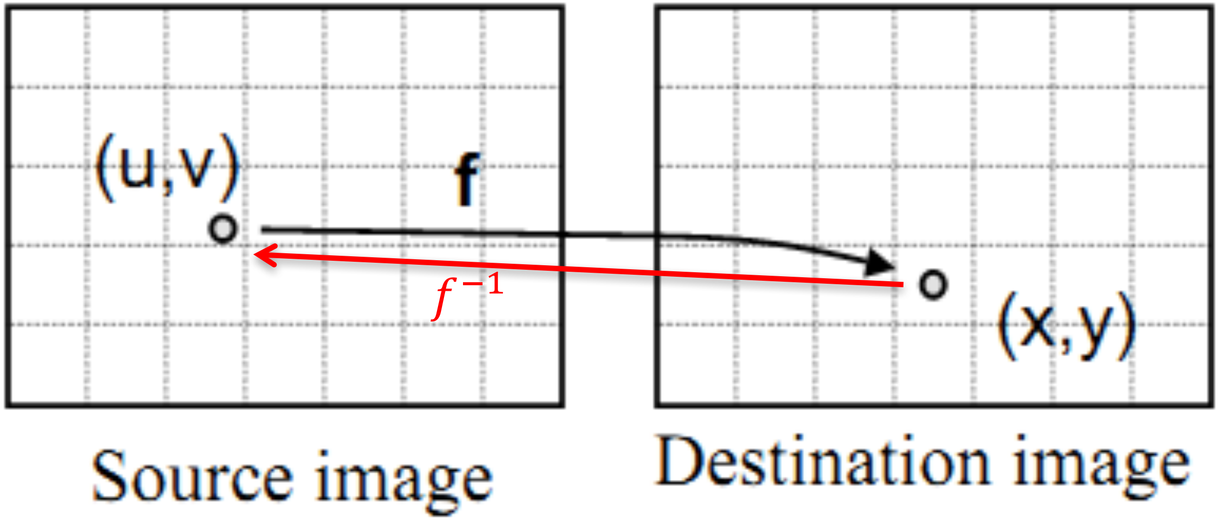
Zoom In

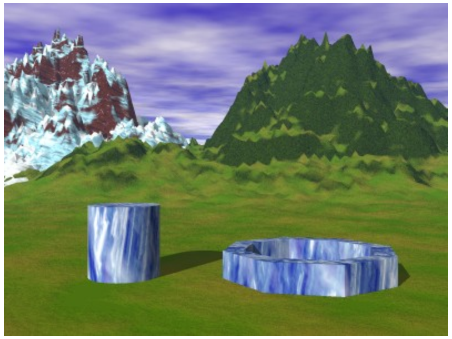
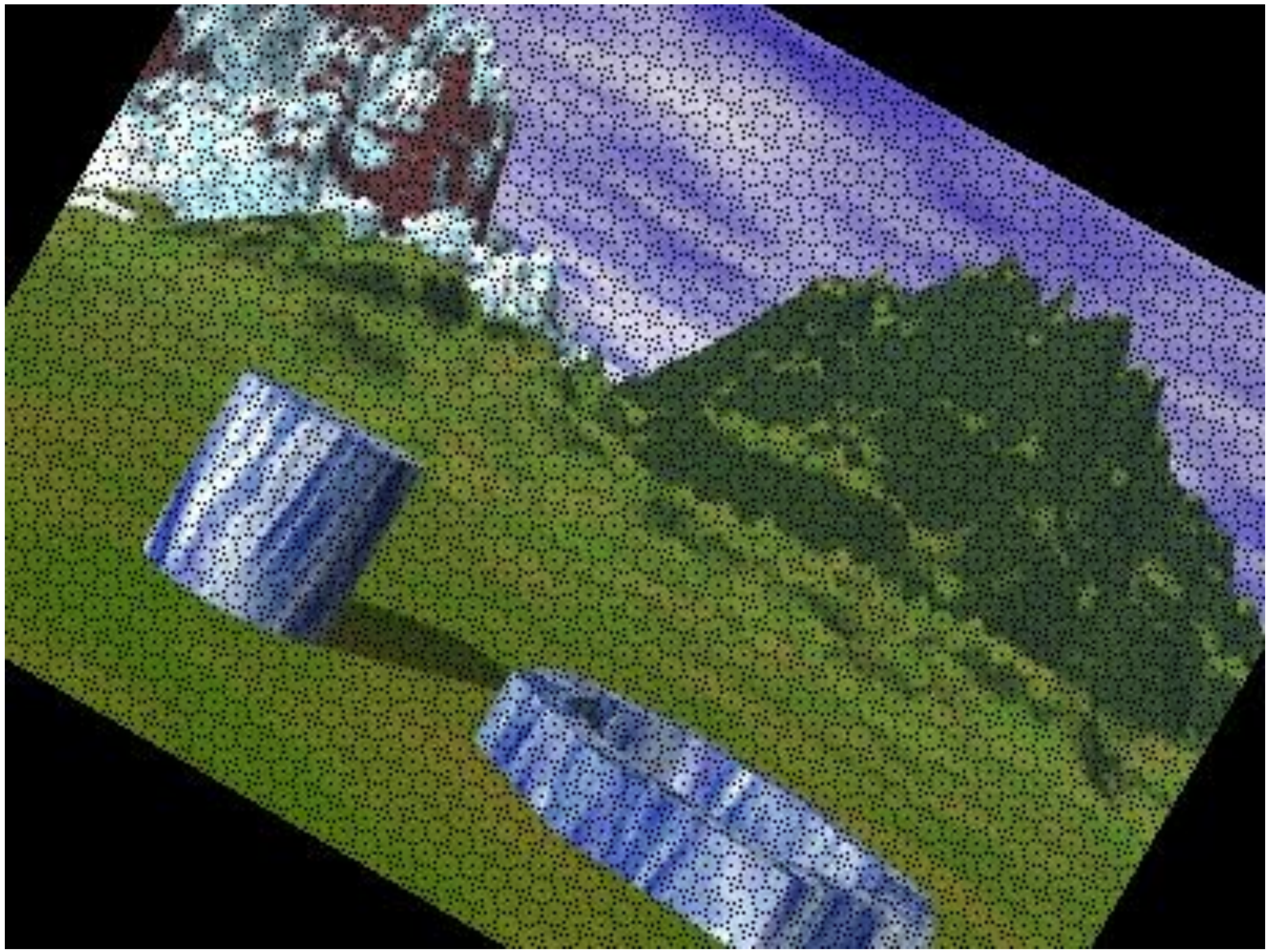


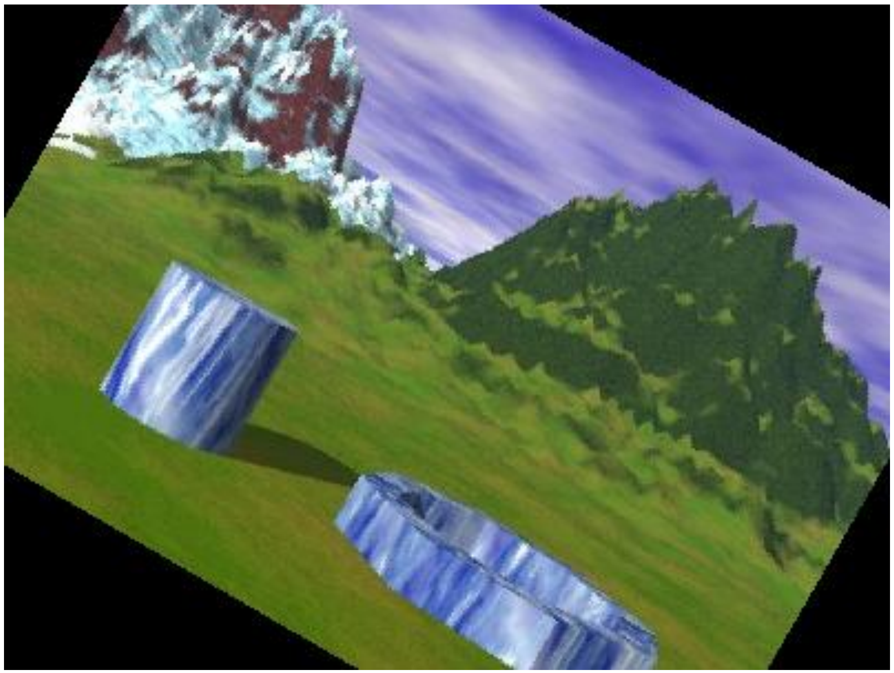
Backward Mapping

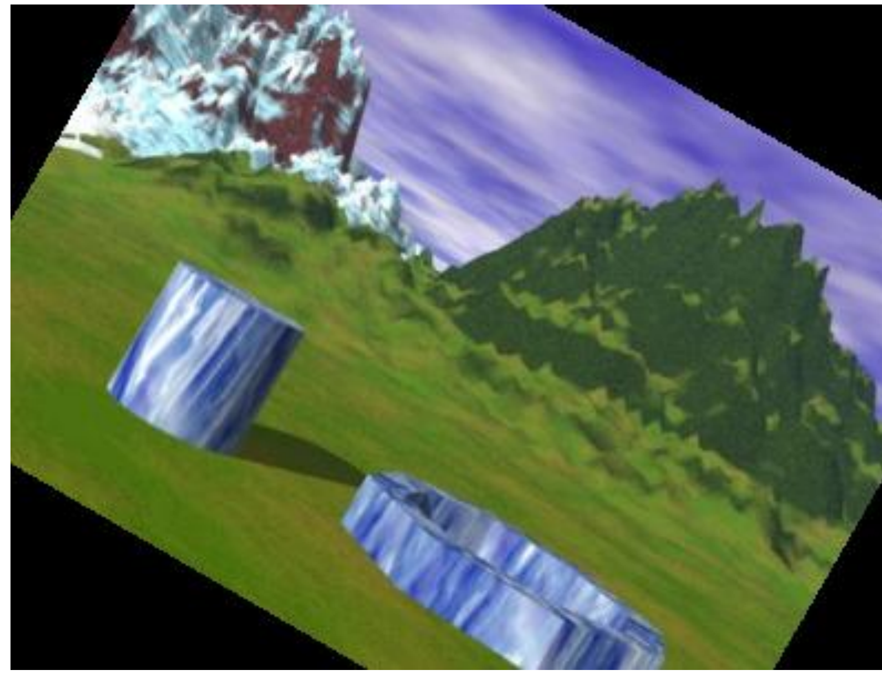
```
for (int x = 0; x < xmax; x++) {
  for (int y = 0; y < ymax; y++) {
    float u = fx^-1(x,y);
    float v = fy^-1(x,y);
    dst(x,y) = src(u,v);
  }
}
```

The Problem:
(u,v) are not integers!



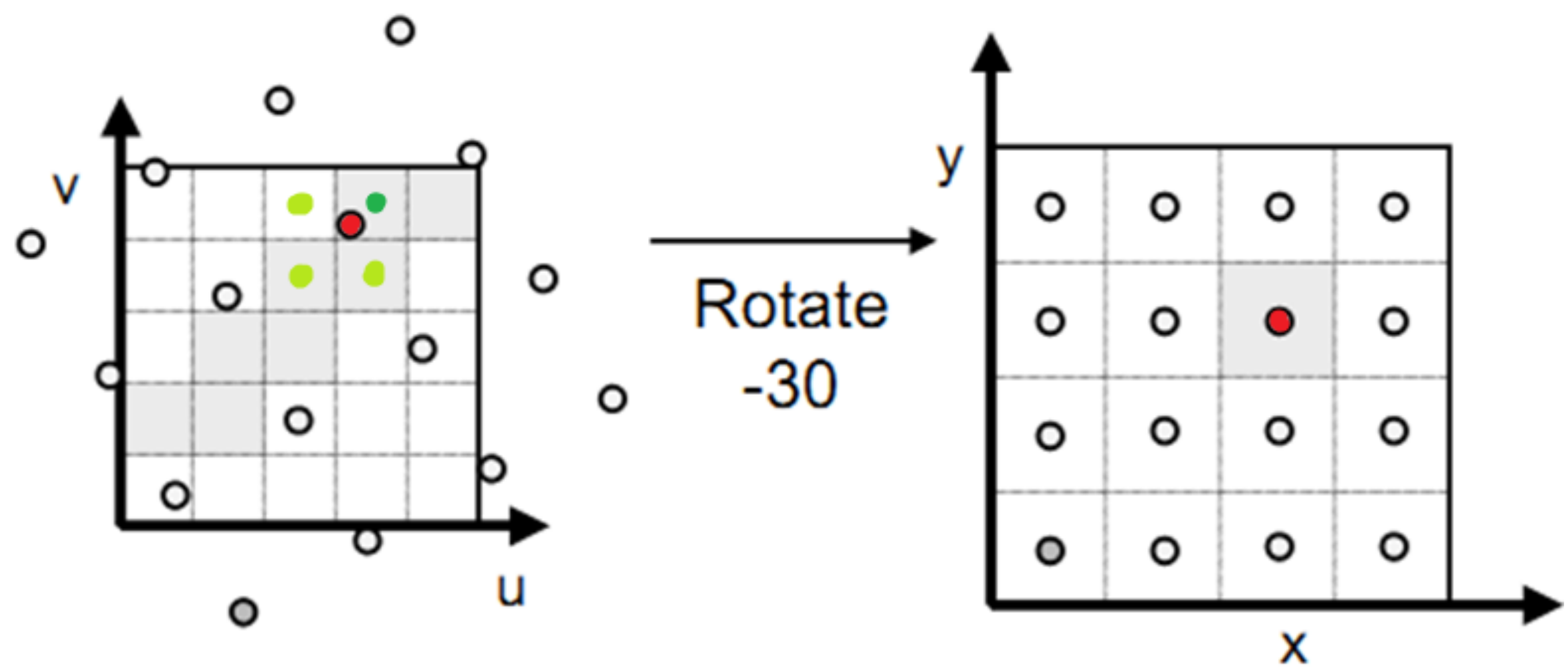


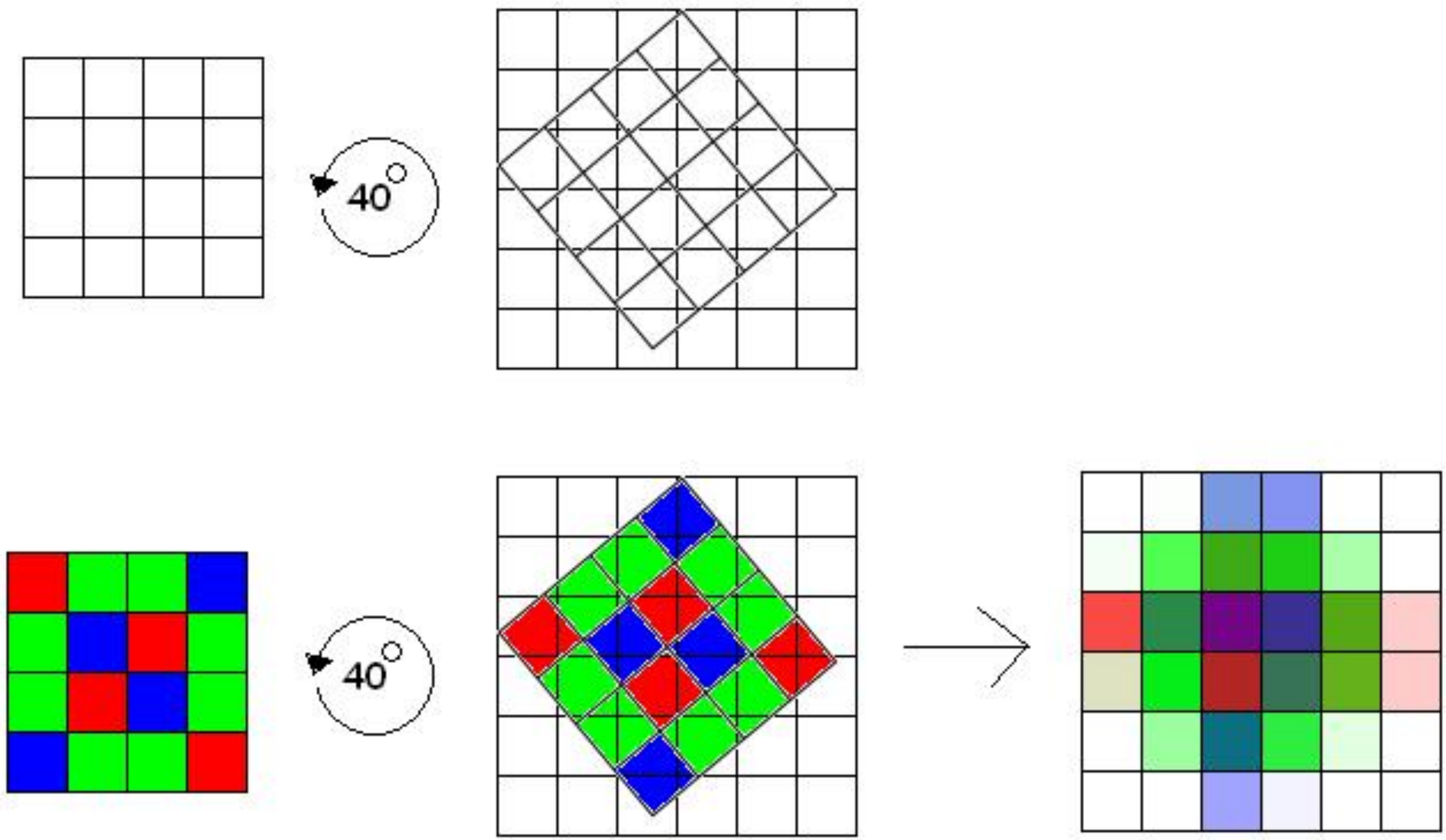


Nearest Neighbor

- Take value at closest pixel:
  - int iu = trunc(u+0.5);
  - int iv = trunc(v+0.5);
  - dst(x,y) = src(iu,iv);

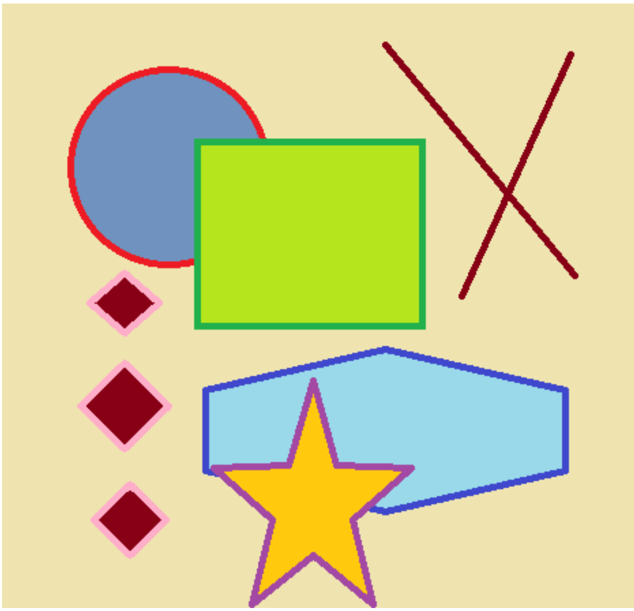
This method is simple, but it causes aliasing



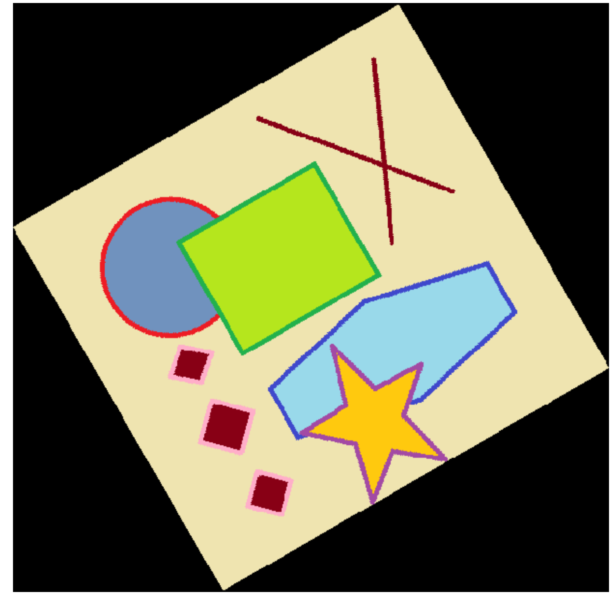

40°
40°

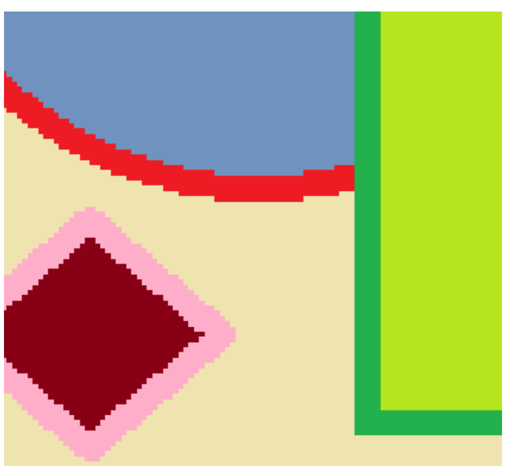
Example - Nearest Neighbor

Original

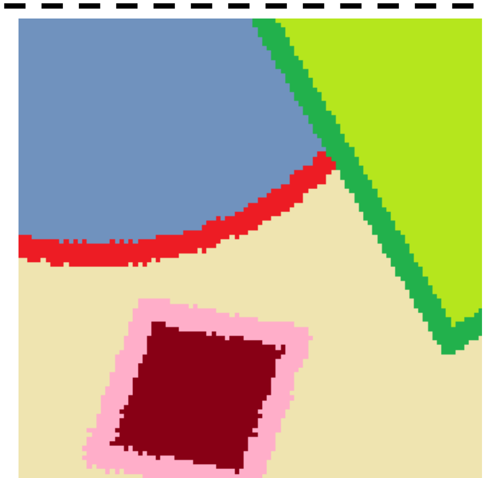


Rotated



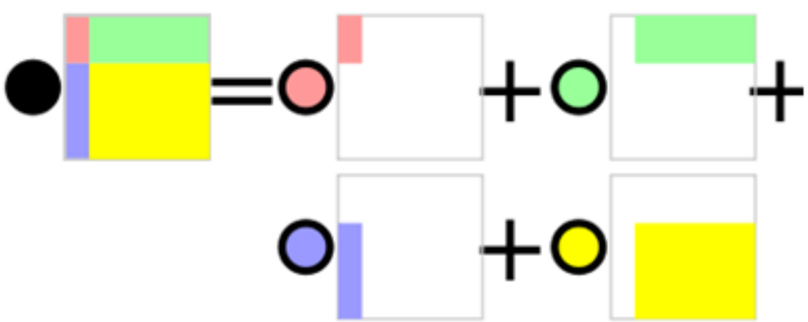


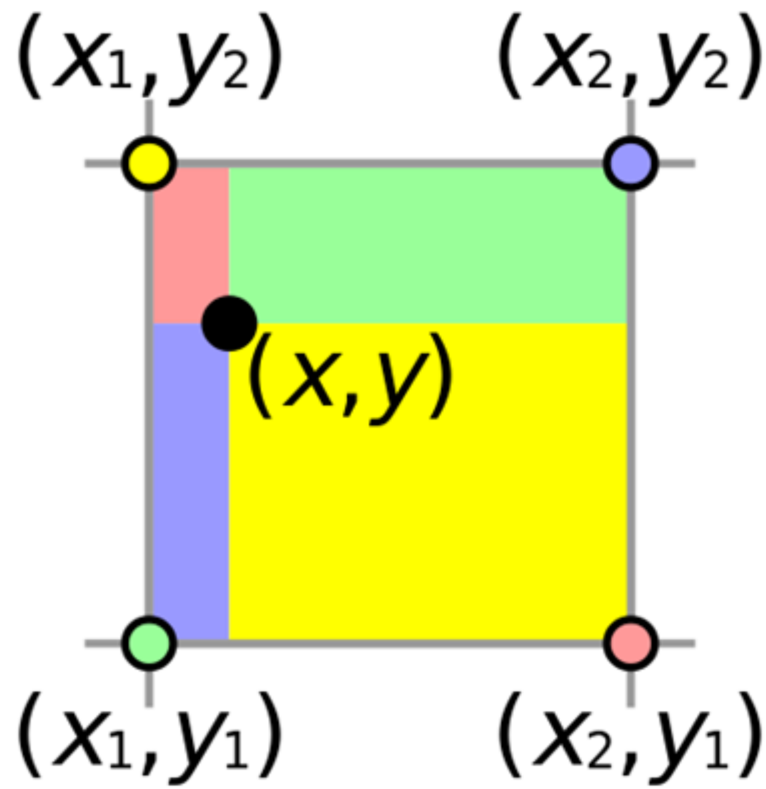
Zoom In



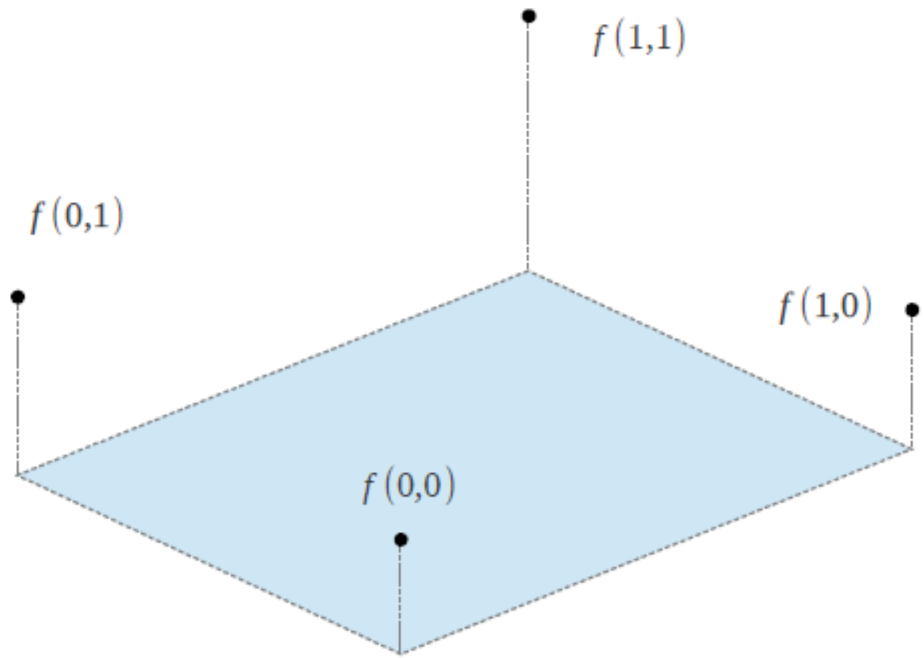
Bi-linear Interpolation

- Bi-linear interpolates four closest pixels.
- The weight for each pixel is proportional to its distance from the sampling point (x,y)

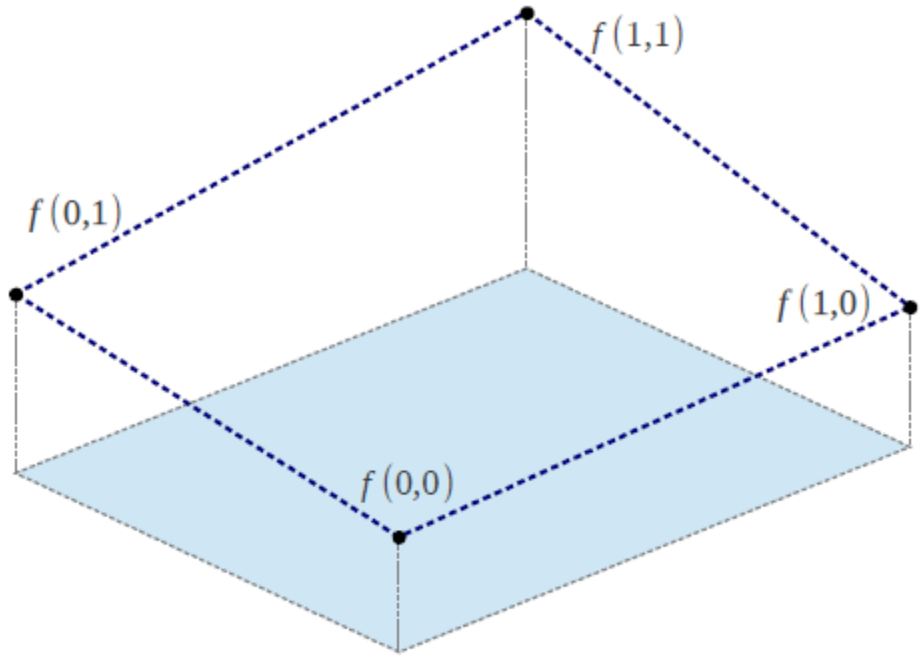




Bi-linear Interpolation

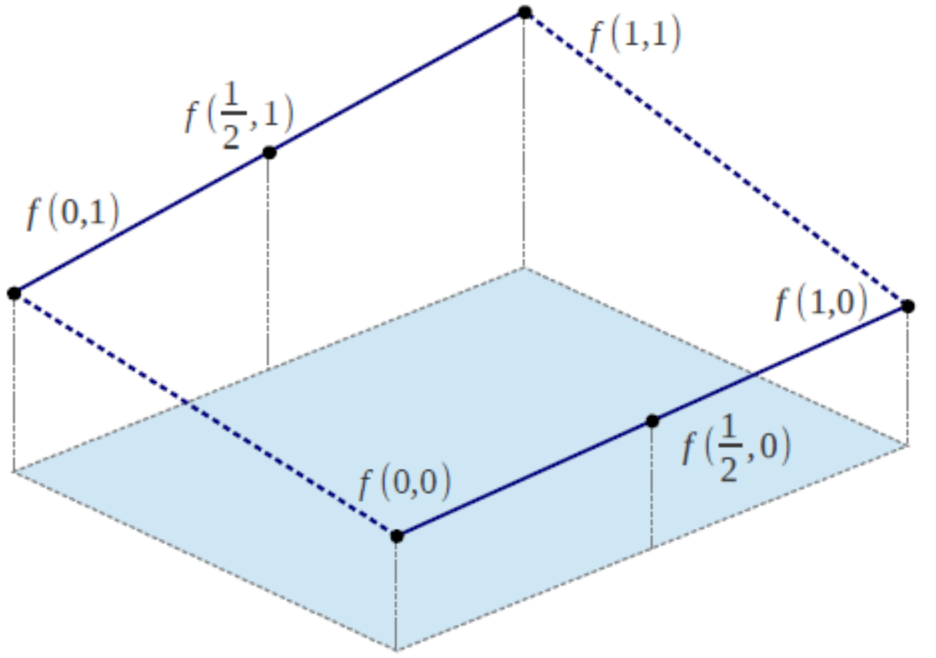


Bi-linear Interpolation



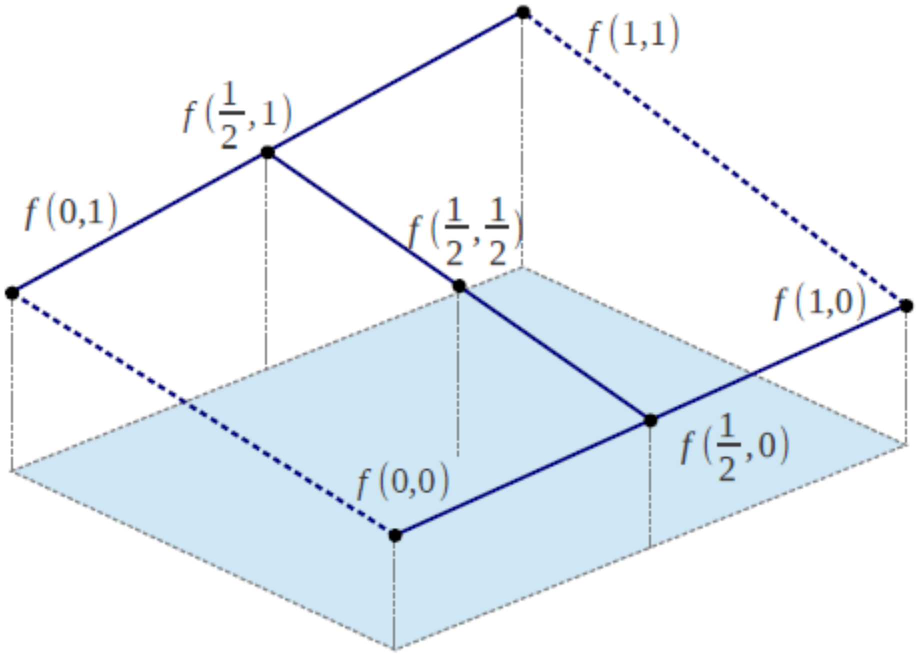
- Model $f(x, y)$ as a bilinear surface

Bi-linear Interpolation



- Model $f(x, y)$ as a bilinear surface
- Interpolate $f(\frac{1}{2}, 0)$ using $f(0, 0)$ and $f(1, 0)$
  Interpolate $f(\frac{1}{2}, 1)$ using $f(0, 1)$ and $f(1, 1)$

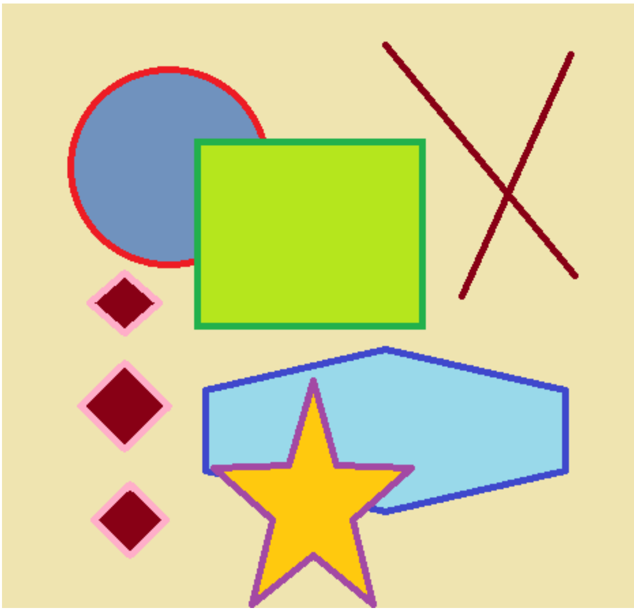
Bi-linear Interpolation



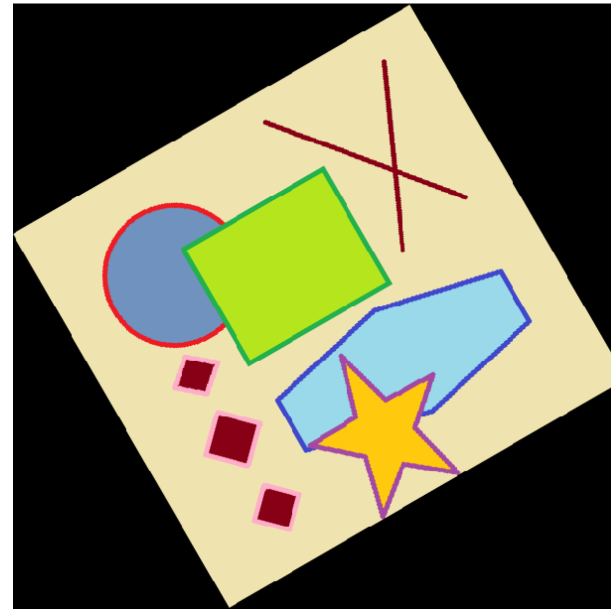
- Model $f(x, y)$ as a bilinear surface
- Interpolate $f(\frac{1}{2}, 0)$ using $f(0, 0)$ and $f(1, 0)$
  Interpolate $f(\frac{1}{2}, 1)$ using $f(0, 1)$ and $f(1, 1)$
- Interpolate $f(\frac{1}{2}, \frac{1}{2})$ using $f(\frac{1}{2}, 0)$ and $f(\frac{1}{2}, 1)$

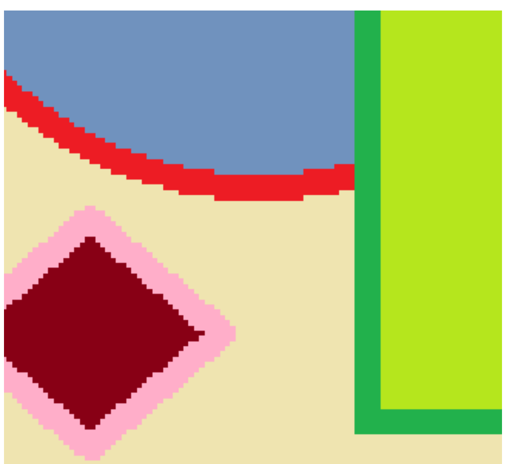
Example Bi-linear

Original

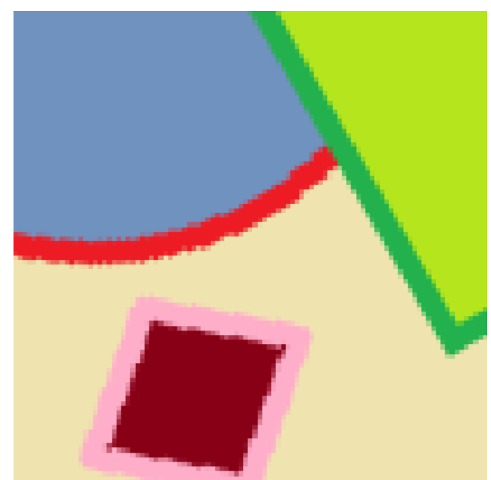


Rotated



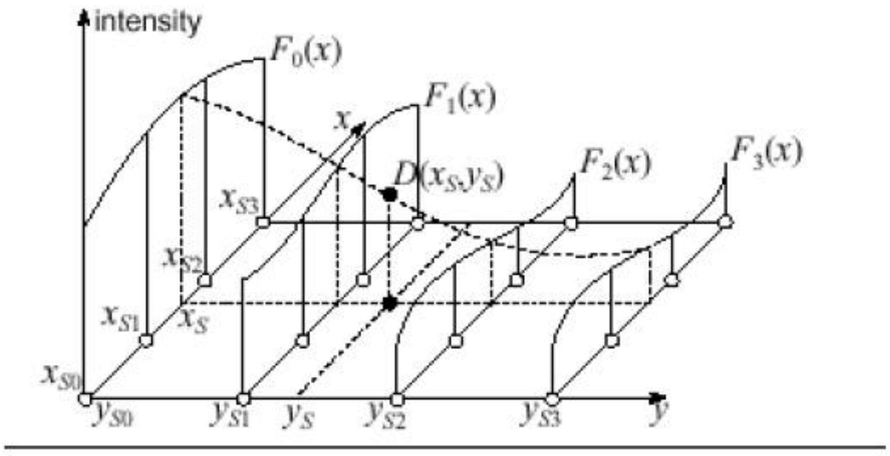


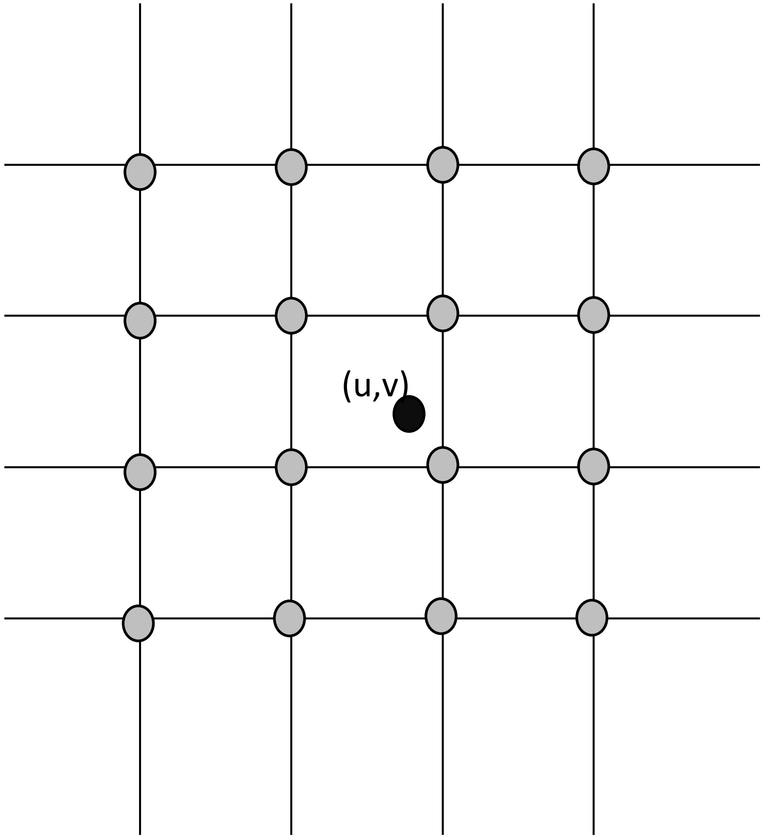
Zoom In



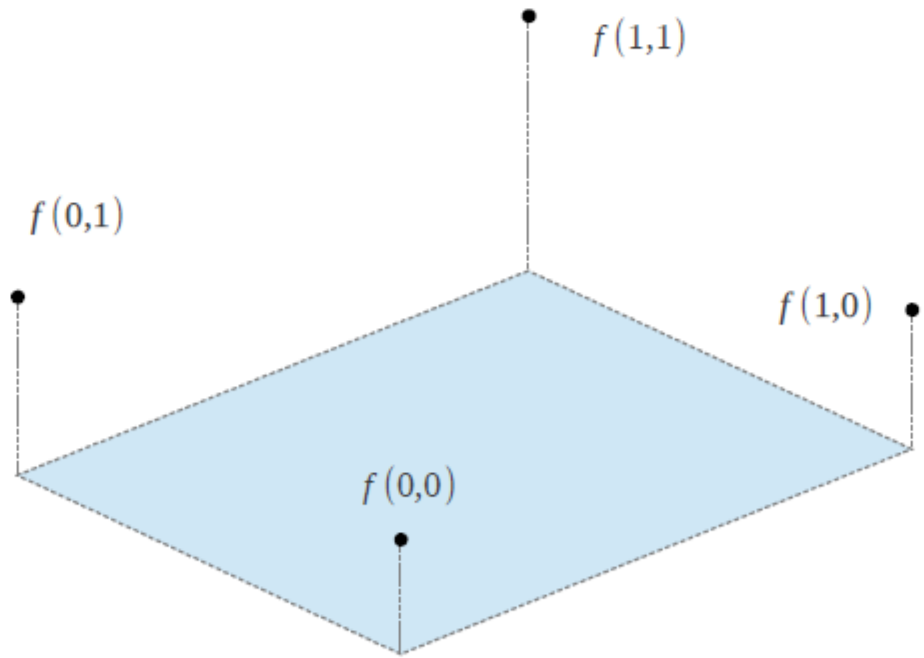
Bi-cubic

- Bicubic interpolates 16 closest neighbors (4x4 neighborhood)
  - The result is much more smooth

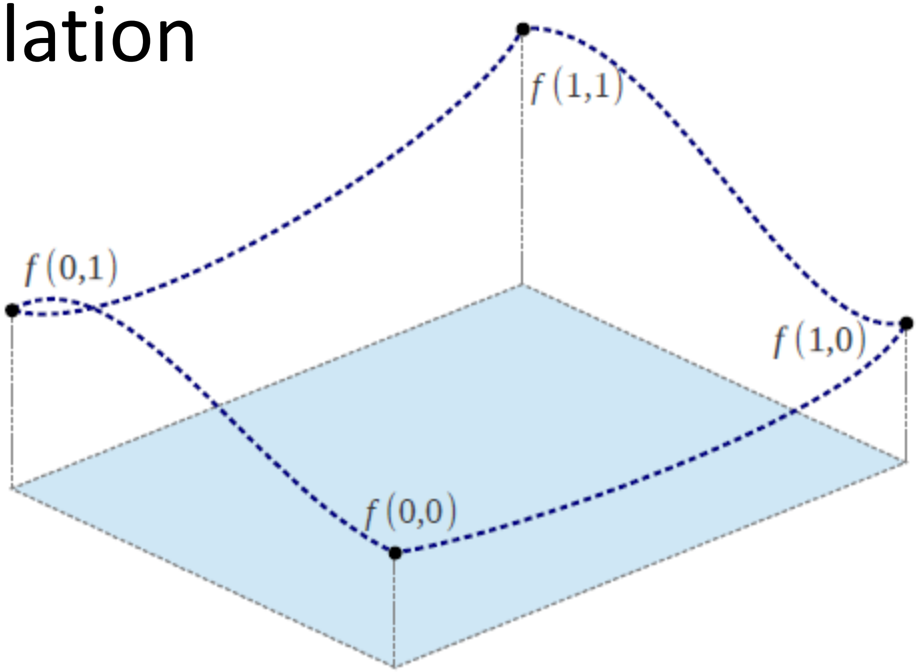




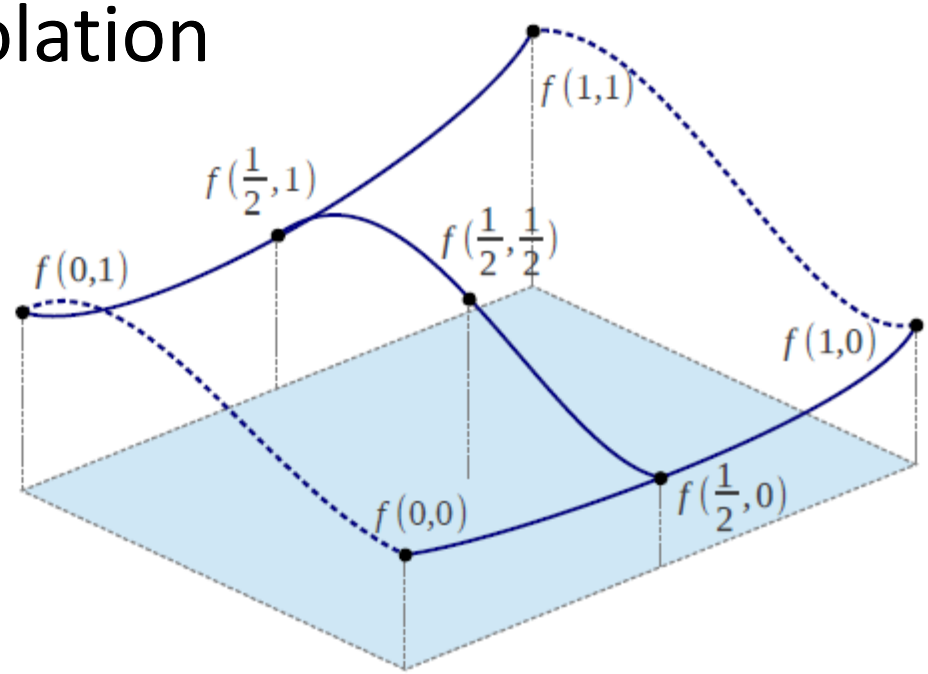
Bi-cubic Interpolation



Bi-cubic Interpolation



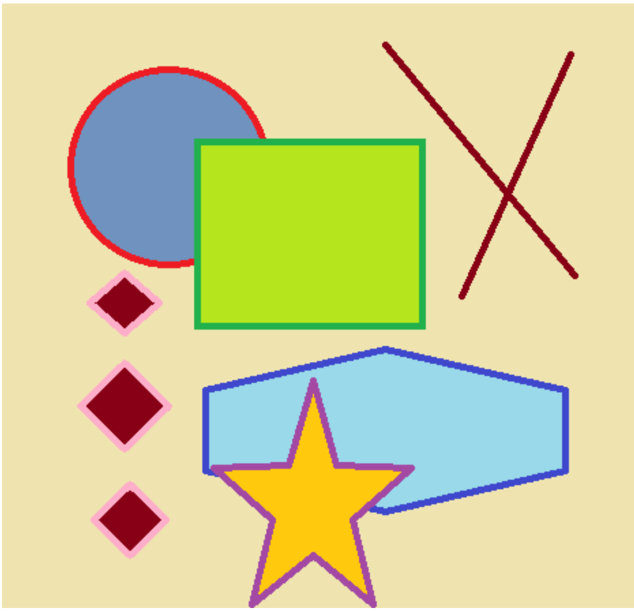
Bi-cubic Interpolation



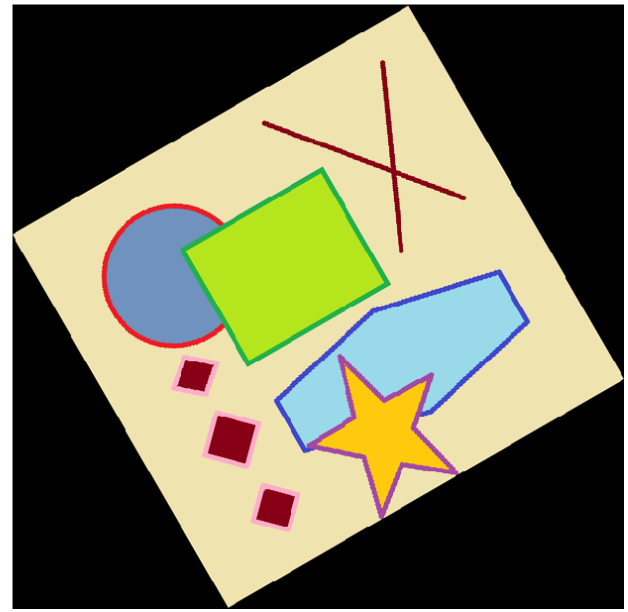
- Interpolate
  - $f(\frac{1}{2},0)$ using $f(0,0)$, $f(1,0)$, $\partial_x f(0,0)$ and $\partial_x f(1,0)$
  - $f(\frac{1}{2},1)$ using $f(0,1)$, $f(1,1)$, $\partial_x f(0,1)$ and $\partial_x f(1,1)$
  - $\partial_y f(\frac{1}{2},0)$ using $\partial_y f(0,0)$, $\partial_y f(1,0)$, $\partial_{xy} f(0,0)$ and $\partial_{xy} f(1,0)$
  - $\partial_y f(\frac{1}{2},1)$ using $\partial_y f(0,1)$, $\partial_y f(1,1)$, $\partial_{xy} f(0,1)$ and $\partial_{xy} f(1,1)$
- Interpolate $f(\frac{1}{2},\frac{1}{2})$ using $f(\frac{1}{2},0)$, $f(\frac{1}{2},1)$, $\partial_y f(\frac{1}{2},0)$ and $\partial_y f(\frac{1}{2},1)$

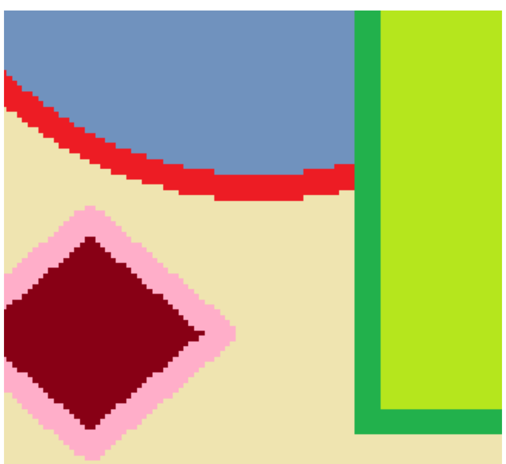
Example Bi-cubic

Original

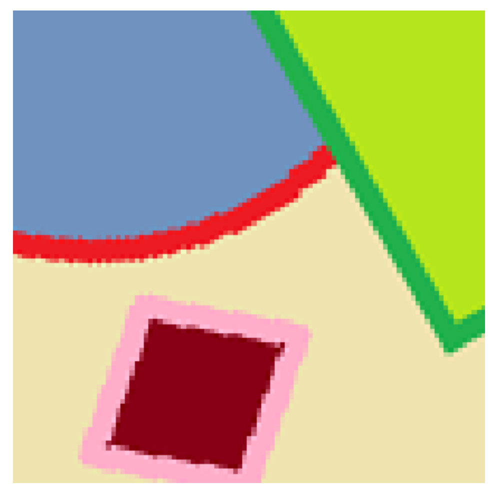


Rotated

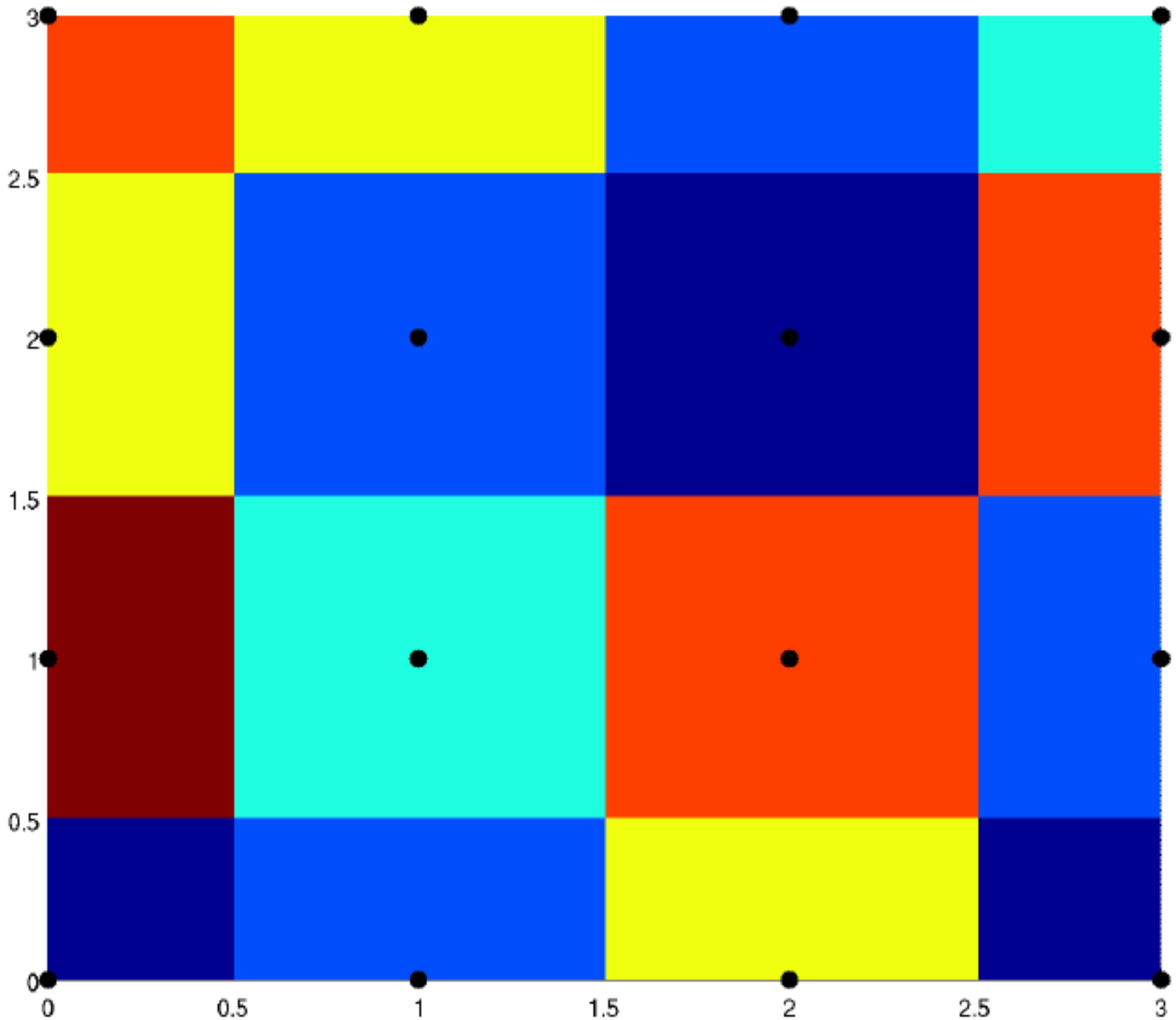




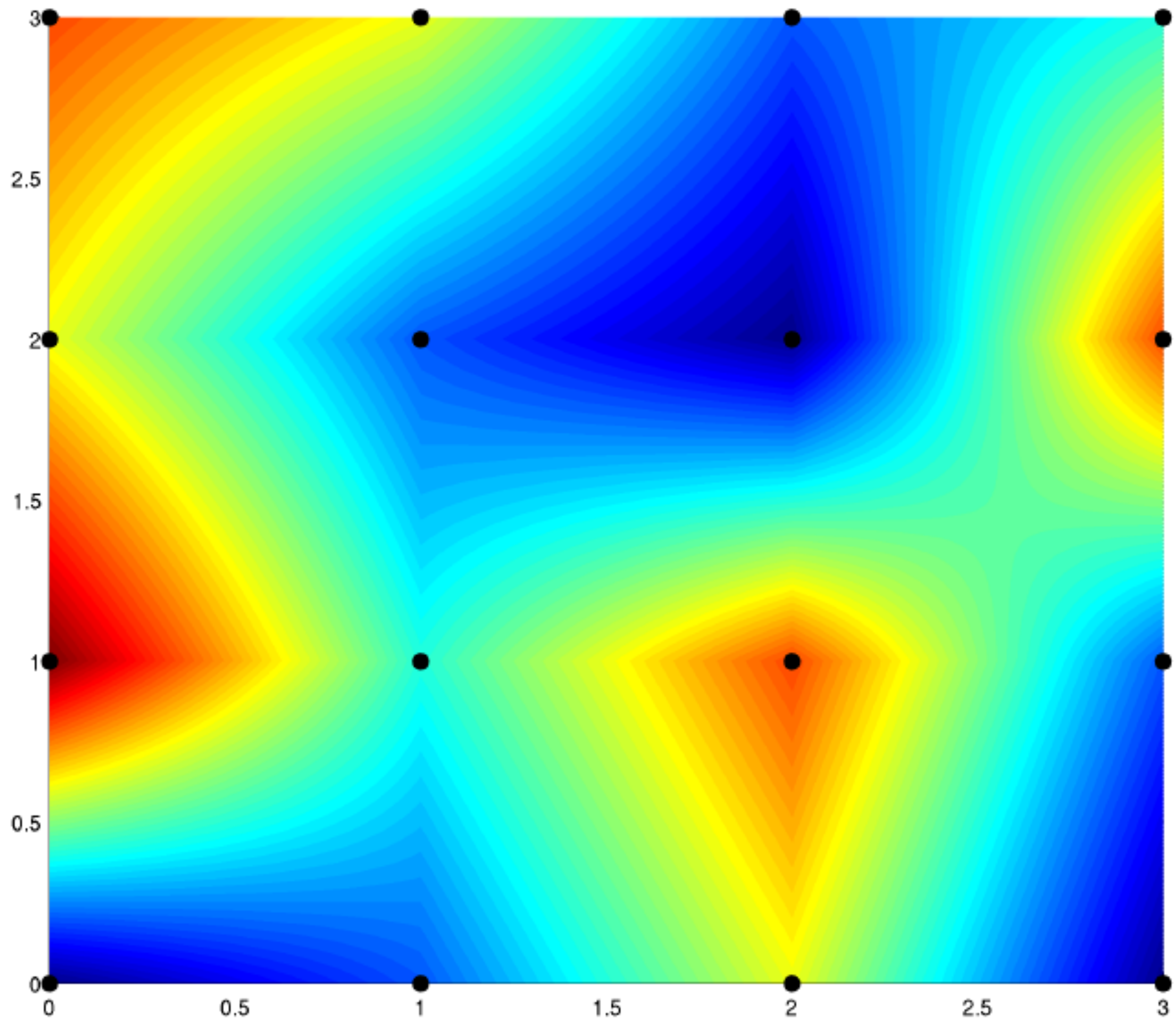
Zoom In



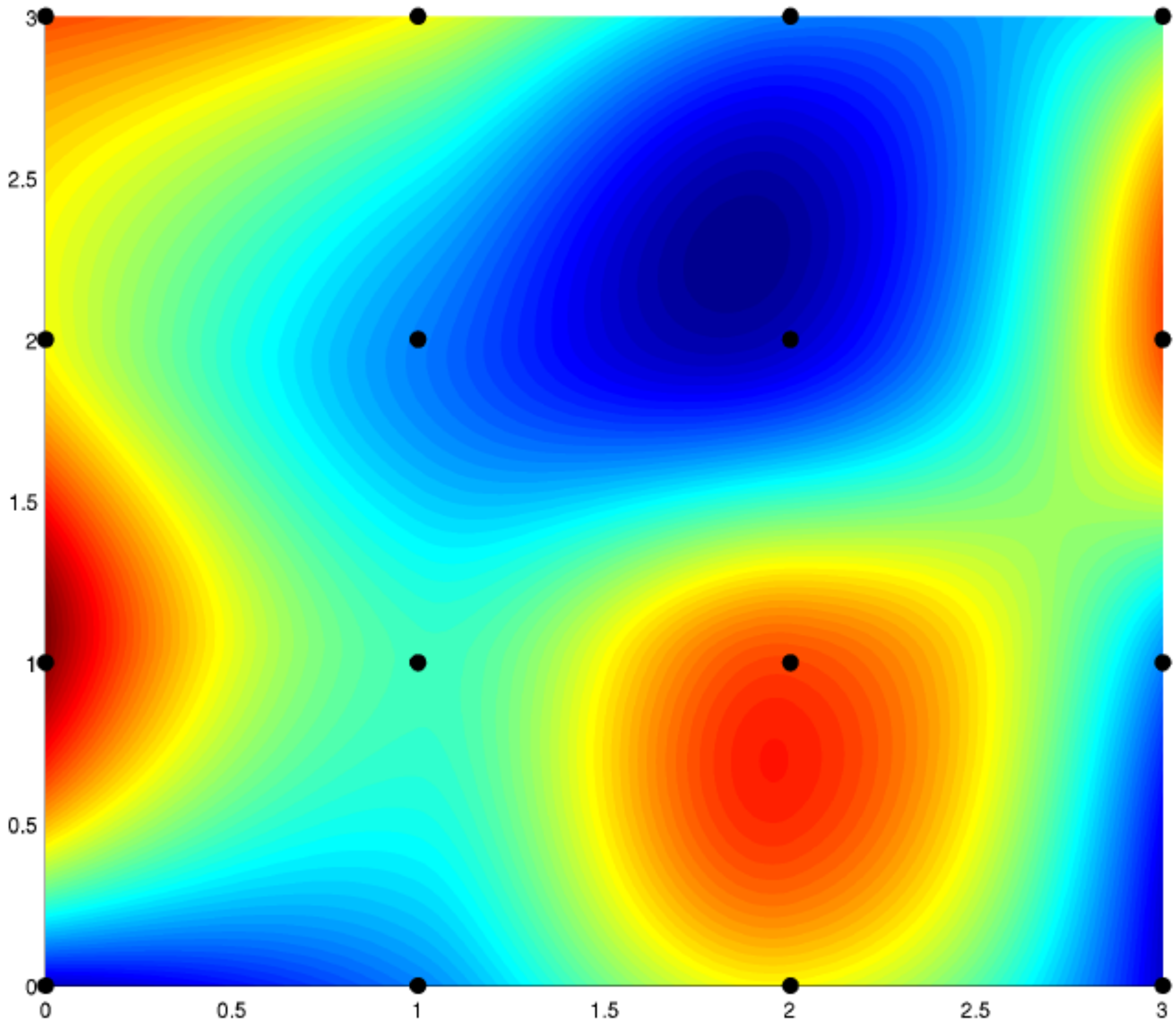
Nearest Neighbor



Bi-Linear Interpolation

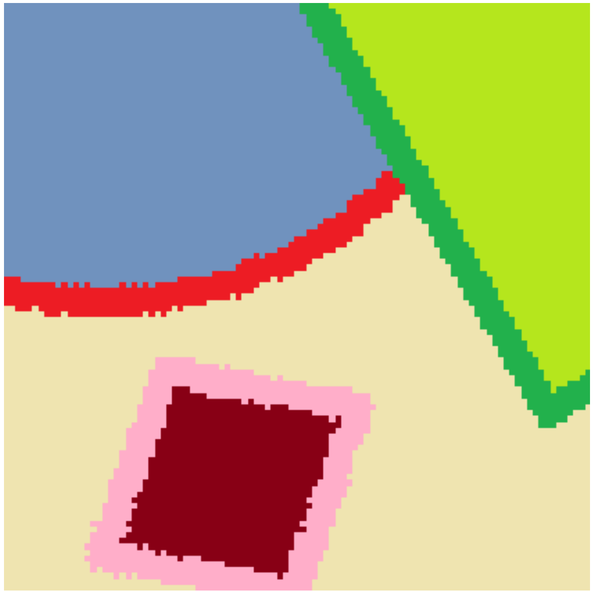


Bi-Cubic Interpolation

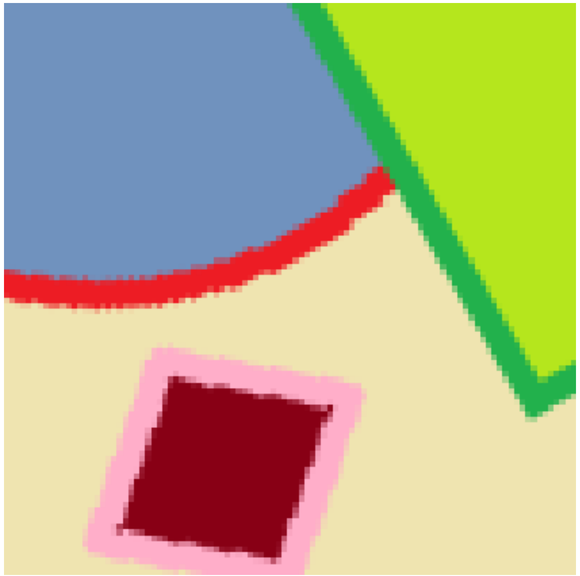


Comparison

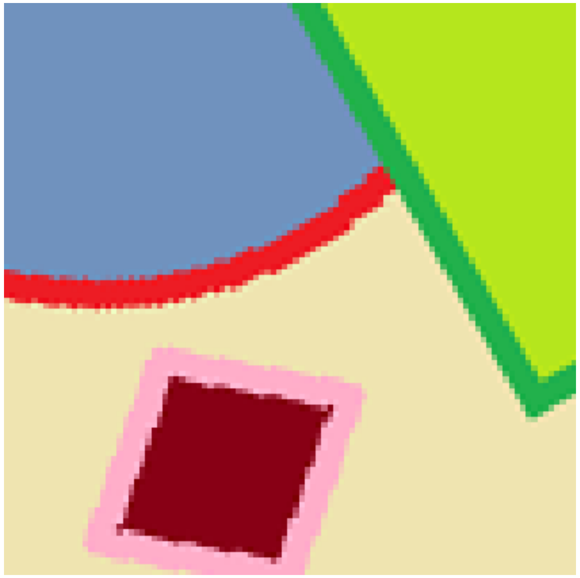
Nearest Neighbor



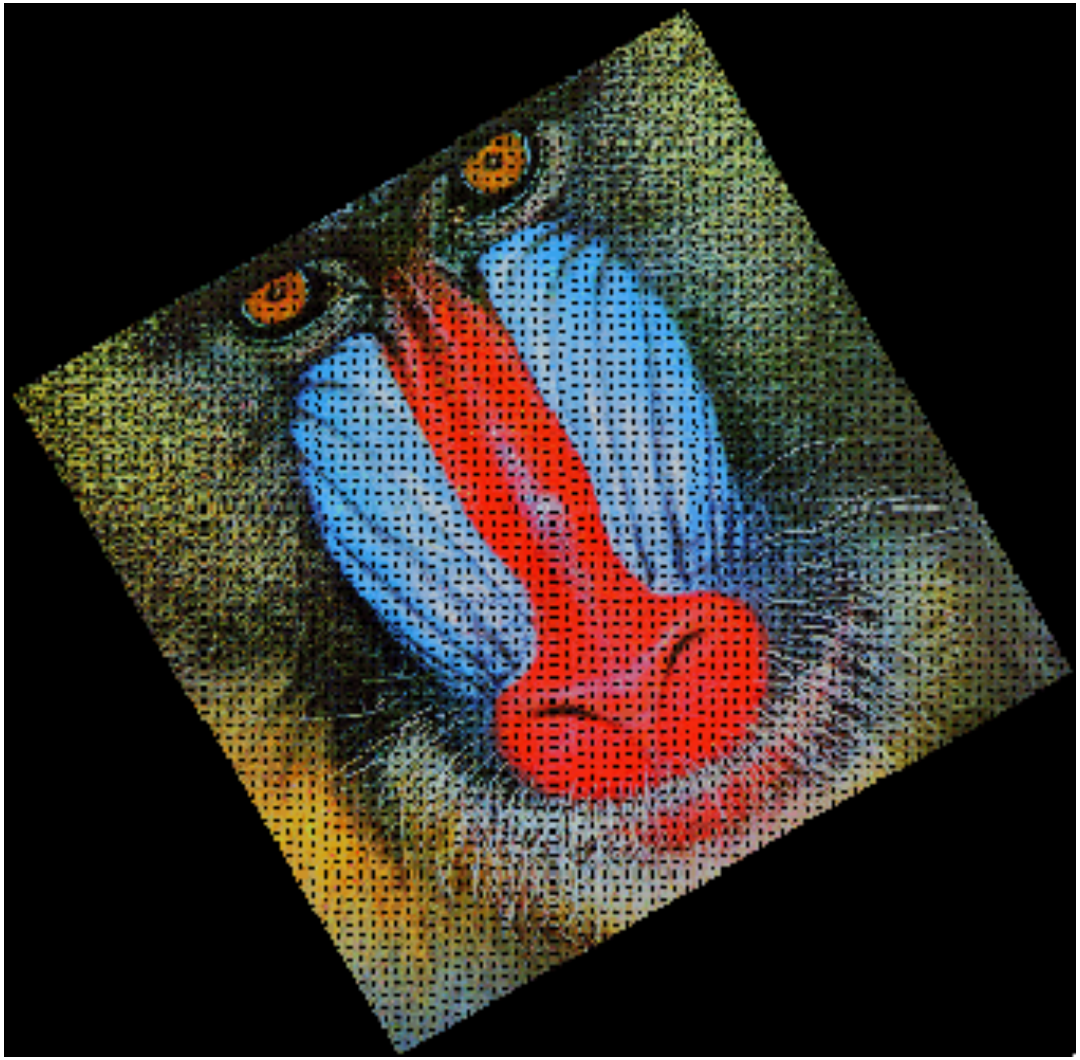
Bi-linear



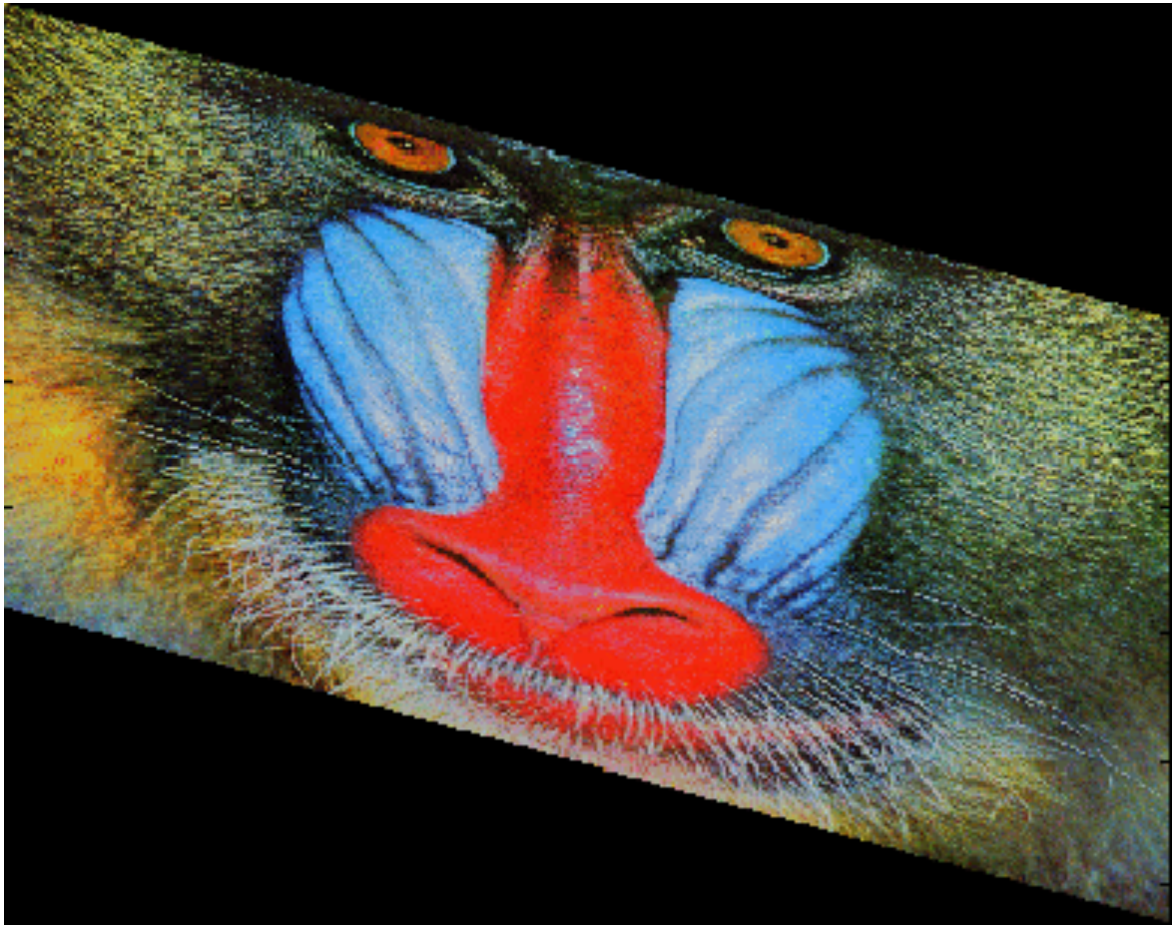
Bi-cubic



Direct Rotation



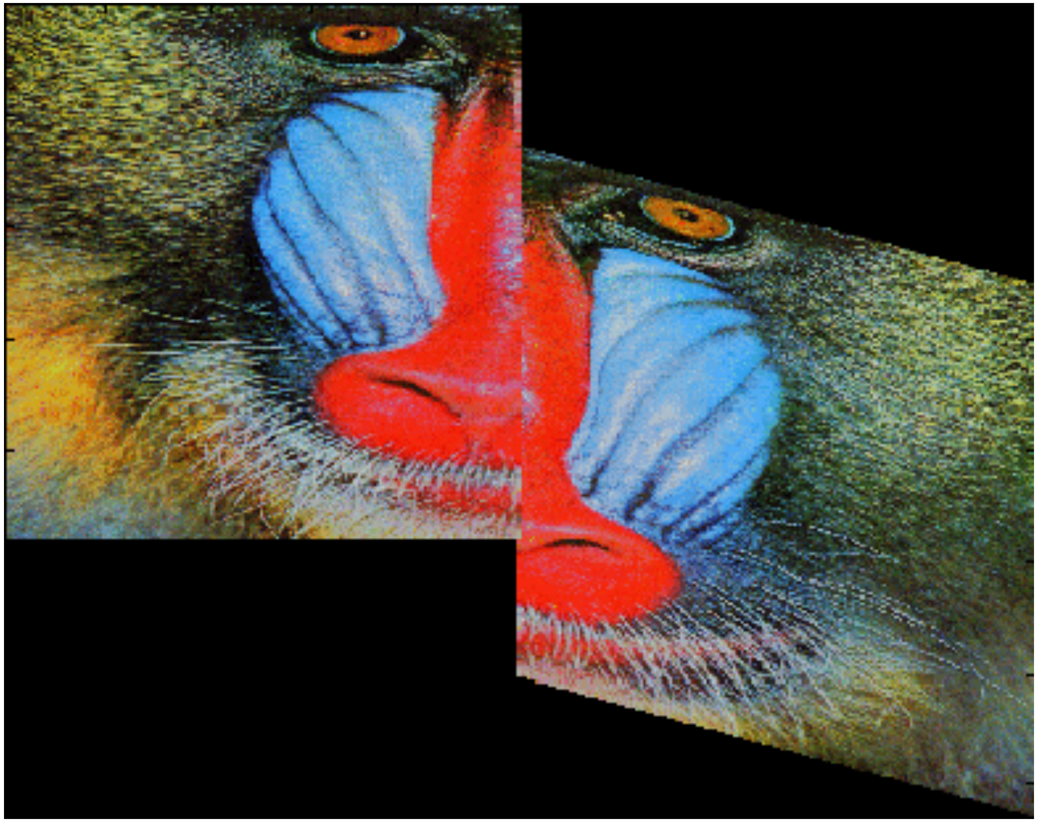
Shear



Shear

Shear and Scale

Operating line by line, faster and simpler filters



Shear

$$\begin{pmatrix} r \\ s \end{pmatrix} = T\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x\cos\alpha - y\sin\alpha \\ x\sin\alpha + y\cos\alpha \end{pmatrix}$$

$$\begin{pmatrix} r \\ s \end{pmatrix} = B\begin{pmatrix} u \\ v \end{pmatrix} = B\left(A\begin{pmatrix} x \\ y \end{pmatrix} \right) = T\begin{pmatrix} x \\ y \end{pmatrix}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = A\begin{pmatrix} x \\ y \end{pmatrix}$$

Shear

A preserve columns

$$\begin{pmatrix} u \\ v \end{pmatrix} = A\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ f(x, y) \end{pmatrix}$$

B preserve rows

$$\begin{pmatrix} r \\ s \end{pmatrix} = B\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} g(u, v) \\ v \end{pmatrix}$$

Shear

$$\begin{pmatrix} r \\ s \end{pmatrix} = T\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x\cos\alpha - y\sin\alpha \\ x\sin\alpha + y\cos\alpha \end{pmatrix}$$

$$\begin{pmatrix} r \\ s \end{pmatrix} = B\begin{pmatrix} u \\ v \end{pmatrix} = B\left(A\begin{pmatrix} x \\ y \end{pmatrix}\right) = B\begin{pmatrix} x \\ f(x,y) \end{pmatrix} = \begin{pmatrix} g(x, f(x,y)) \\ f(x,y) \end{pmatrix}$$

We get

$$f(x,y) = s = x\sin\alpha + y\cos\alpha$$

$$g(u,v) = x\cos\alpha - y\sin\alpha$$

Shear

$$g(u, v) = x\cos\alpha - y\sin\alpha$$

We need to express it in terms of u,v

We know that x=u, and

$$v = f(x, y) = x\sin\alpha + y\cos\alpha$$

We get

$$y = \frac{v - x\sin\alpha}{\cos\alpha} = \frac{v - u\sin\alpha}{\cos\alpha}$$

Shear

We put it all together and get

$$g(u, v) = u\cos\alpha - \frac{v - u\sin\alpha}{\cos\alpha}\sin\alpha = u\sec\alpha - v\tan\alpha$$

Shear

At last we get

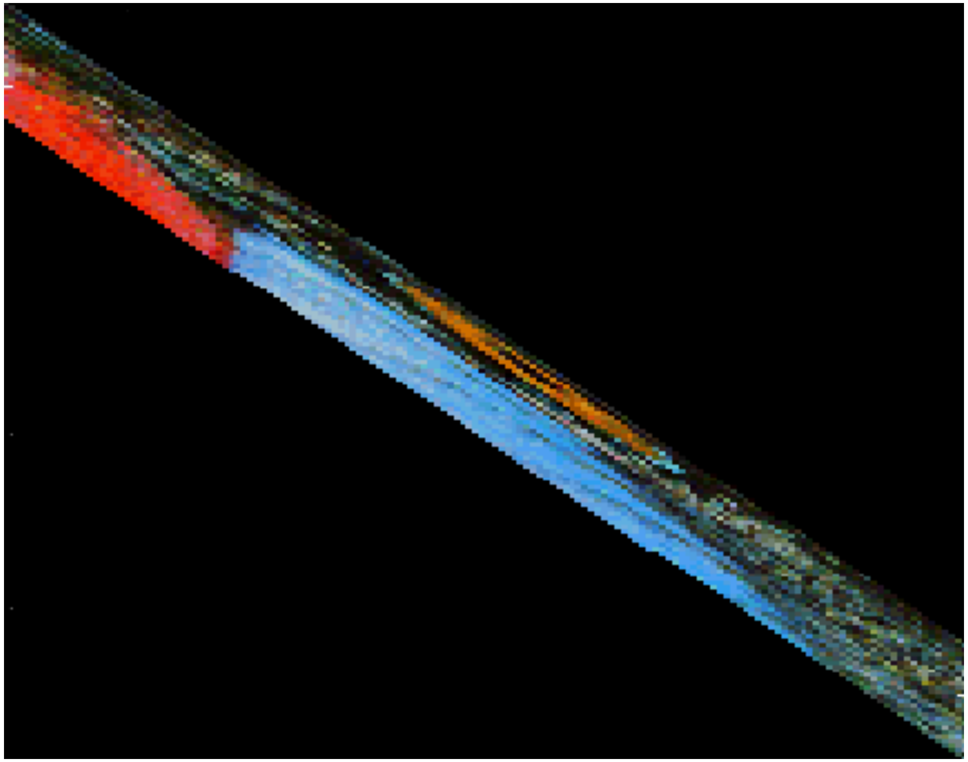
$$A\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ x\sin\alpha + y\cos\alpha \end{pmatrix}$$

$$B\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u\sec\alpha - v\tan\alpha \\ v \end{pmatrix}$$

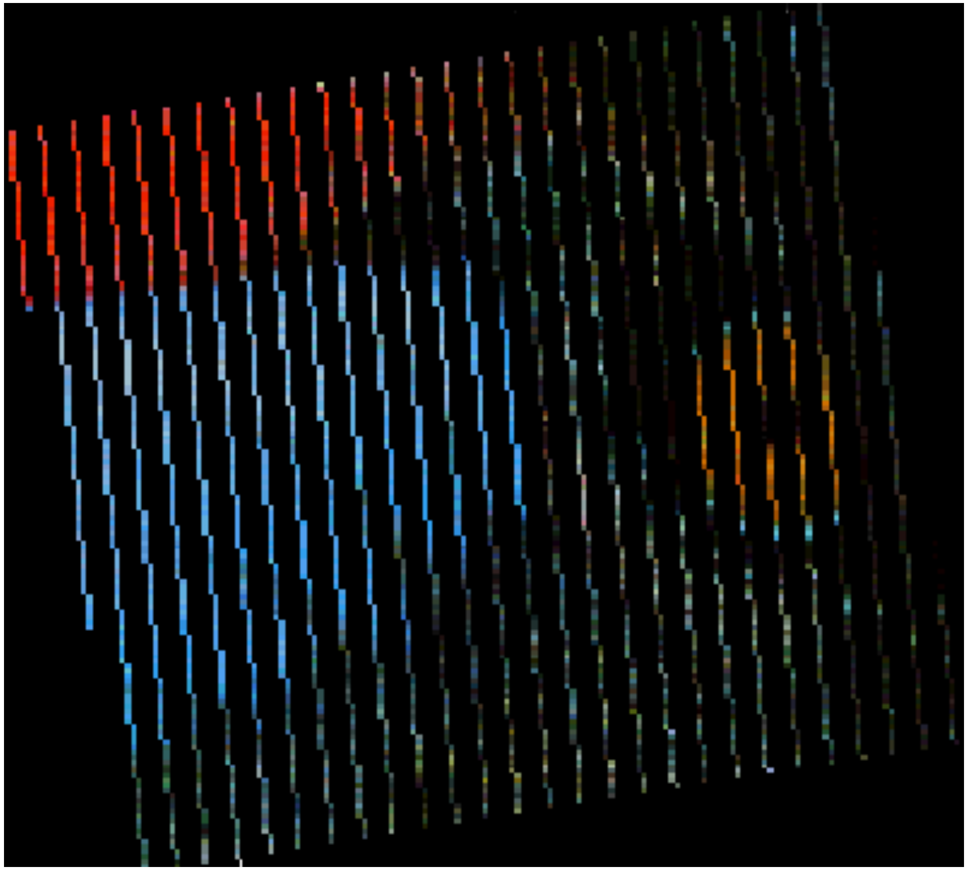
Shear

Using a large angle (80 degree)

Shear

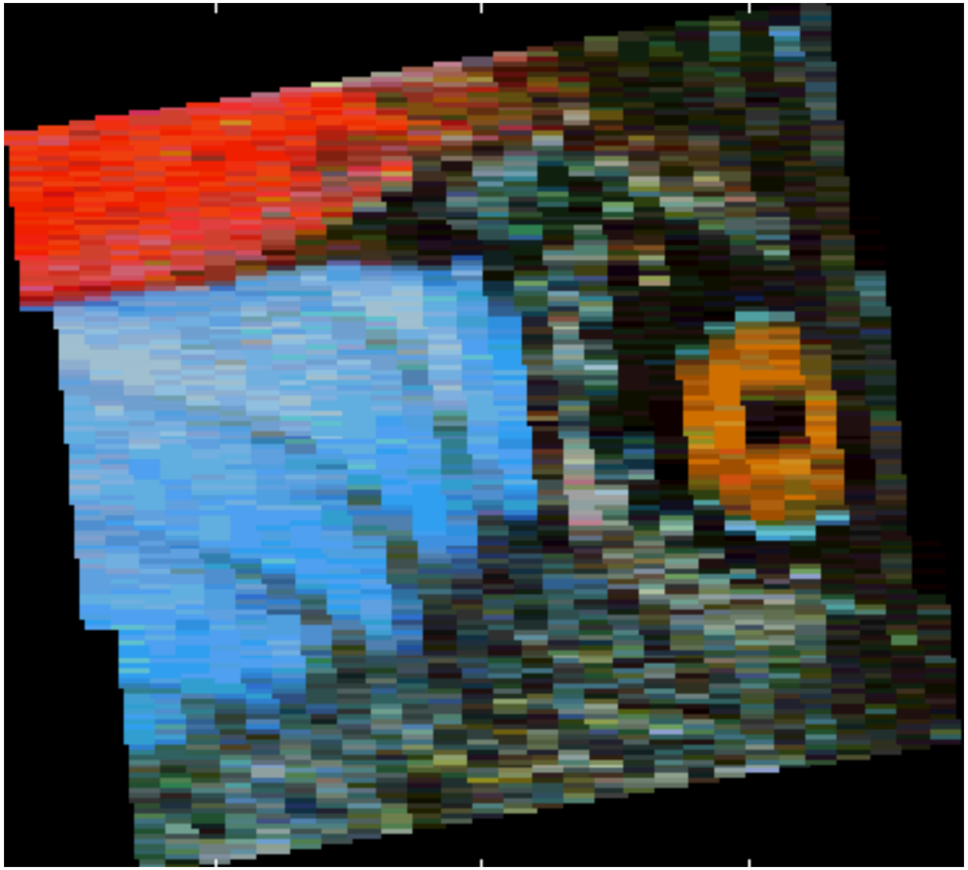


Shear



Shear

Second pass with Backward Mapping



Shear

- Rotate in 90 degree, then use shear with a small angle

- We still have a scale factor in the shear which create holes; one solution is to use filter

Shear

The other solution is by using three shear transformations

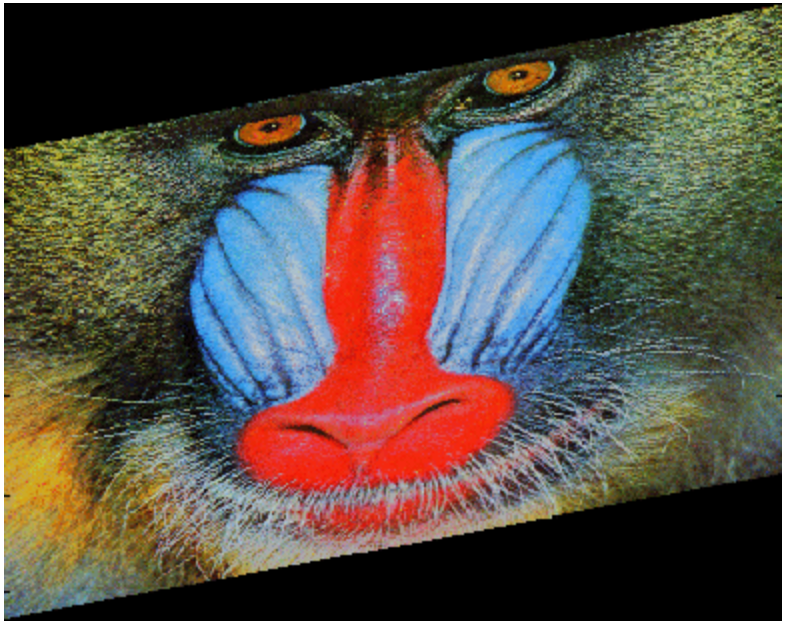
$$\begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} = \begin{pmatrix} 1 & -\tan\alpha/2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \sin\alpha & 1 \end{pmatrix} \begin{pmatrix} 1 & -\tan\alpha/2 \\ 0 & 1 \end{pmatrix}$$

We need Three passes instead of Two.

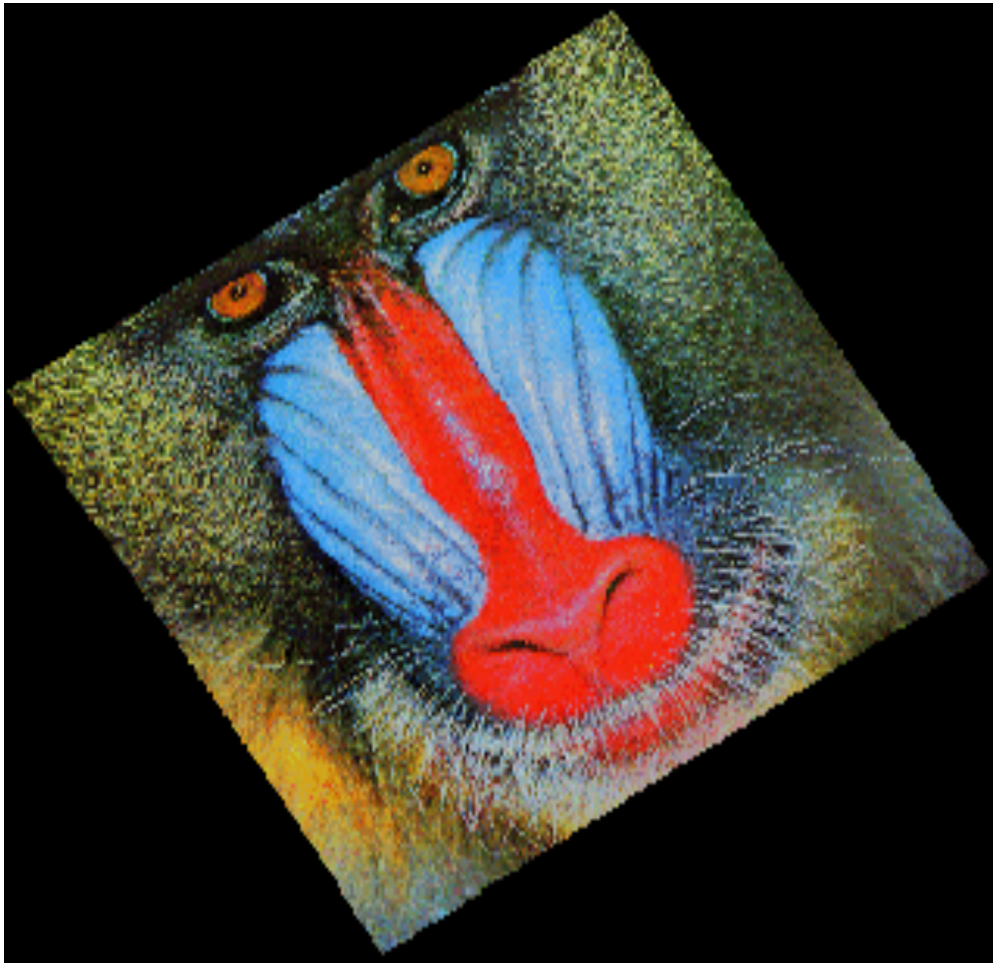
But no scale! Just shift lines!

Shear

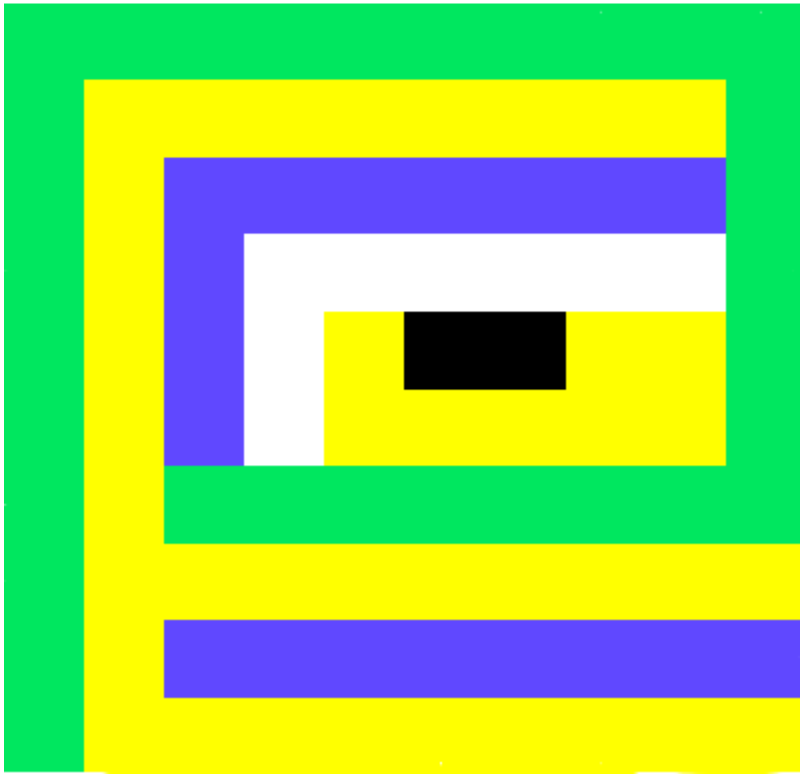
Two first shears



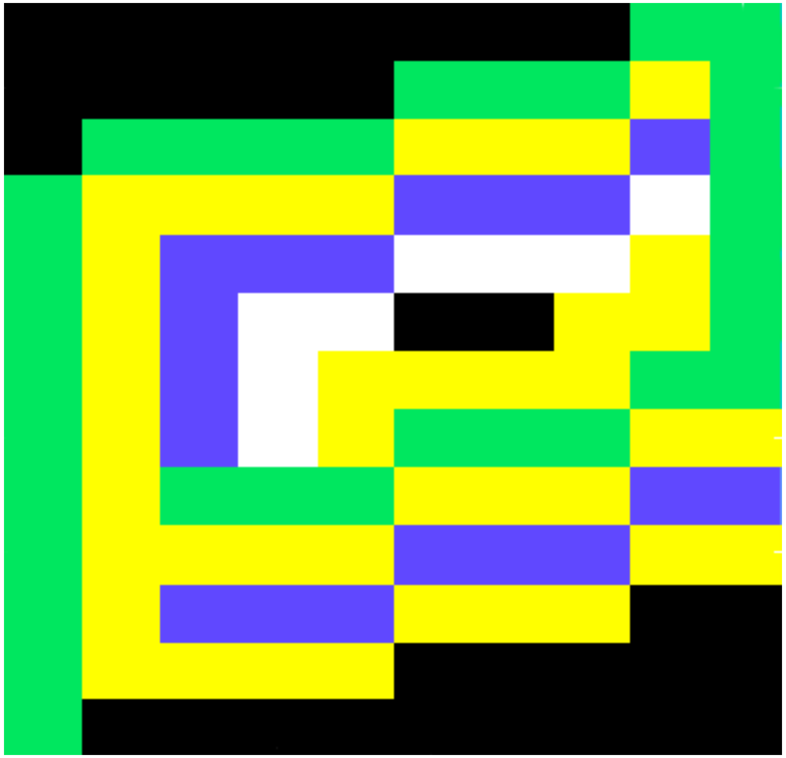
Shear



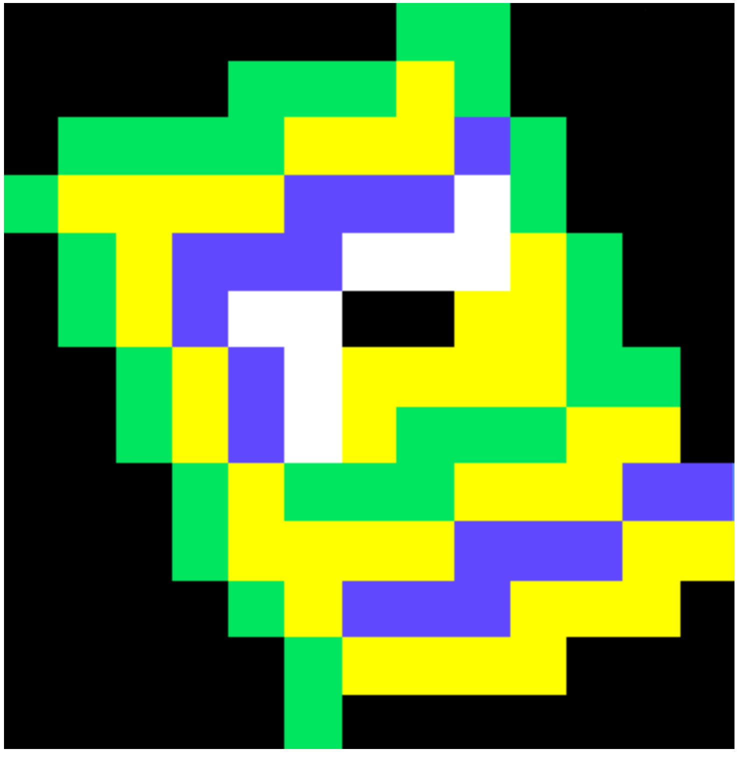
Shear



Shear



Shear



Shear