Generalized Re-Sampling

Using Little Square Model (math is less neat, but intuition are good)

\[ f(x, y) \rightarrow f^{-1}(x', y') \]

Scaling \( x', y' = (s_x x, s_y y) \)

Point samples \( \rightarrow \) point samples

Little squares (or other areas)
Square in \( I = \) square in \( I' \)

Scale up \( (s=3) \) \( I \) src square = 9 dst square
Scale down \( (s=3) \) \( I \) dst square = 9 src squares

Might not overlap perfectly \( \frac{1}{4} \) ths

Some Algorithms
Forward (splat) how to avoid holes?
how to deal with mapping to \( I' \)?

Reverse need \( f^{-1} \)
need interpolation
need sum over area (pre-filtering)
If separable - do it that way!
(can actually separate lots of things
Scale \(\Rightarrow\) Scale X, scale Y
affine transform (rotation) \(\Rightarrow\) shear X, shear Y)

Arbitrary \(f(x,y)\)

anisotropic - longer in 1 dimension than
the other
non-linear - may have a funny shape
spatially varying - different at different
parts of the image

Splatting (amount accumulation / normalization)

Area Averaging assume simple shape
circle / ellipse \(\leftarrow\) normal kernels
see that filter kernel is this
shape in source

Super Sampling lots of points
average together

equivalent \(\Rightarrow\) make big image and downsample
(but easy downsampling)
Splatting for Painting

Forward Splats vs Artistic Shape
Brush Stroke
Paint by Numbers

Easy improvements
Randomized Strokes / ~
randomized color
Edges on strokes (top is light / bottom is dark)

Harder improvements - Adapt to image
stay in lines (clip strokes to boundaries)
draw edges
easy version - edge ~ high frequencies
add small dots where there are HF
↑ don't do too much, or won't look painted.