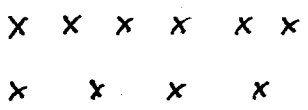
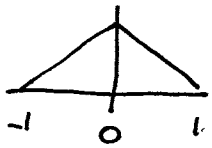
11/10

Recap: Resampling





in units of whichever is larger

source position

$w(t) = s\,t$

warp — scale

target position

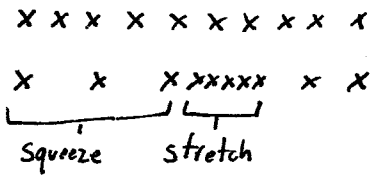
for each target position
  for each source position (in range)
    convert source pos → target pos
    lookup filter value *
    sum

* note: the filter width might need to be scaled

What about arbitrary warps?

$w(R) \Rightarrow R$ 1D

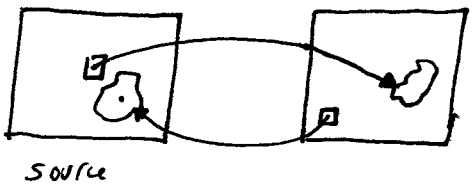
issues: kernel not constant



more general $\mathbb{R}^2 \Rightarrow \mathbb{R}^2$

- linear
- affine
- projective
- nonlinear
  - arbitrary functions
  - grid warps
  - scattered data interpolation
  - other controls

What about this wierd resampling



circles/squares map to odd shapes!

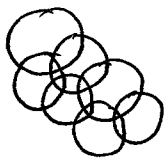
Pre-filter may have a wierd shape (circle/square in target)

Algorithm - forward

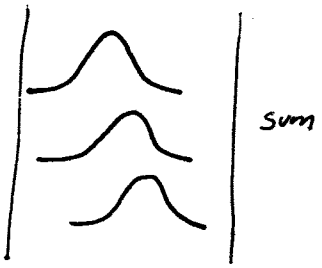
for each source pixel (x,y)

find destination position $w(x,y)$

splat



in 1D



problems: holes

hard to get filter shapes small

need to accumulate

kernel normalization at end (how many splats hit target)

Algorithm - reverse

for each target pixel (x',y')

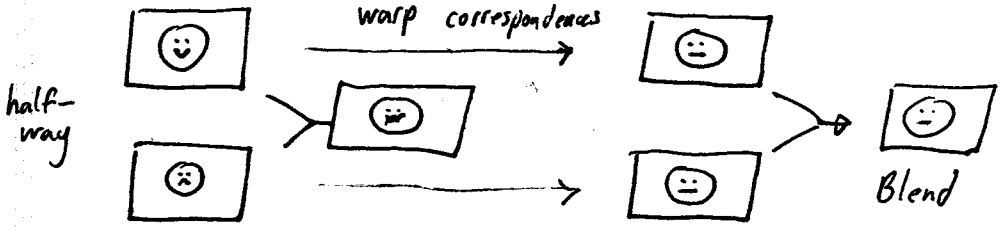
find source position $x,y = w^{-1}(x',y')$ ← need inverse

Sample source @ x,y

- reconstruct (interpolate)
- use larger reconstruction to account for prefilter

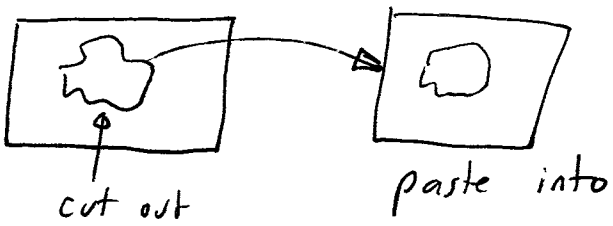
hard: account for prefilter (if funny shape), need inverse

Useful Case : Morph



warp to align correspondences
then blend works

Image Cloning



(1) just paste = hard edge, easily visible

(2) seam-cuts - find cut where images are the same

(3) feather - fade opacity as get near edges
But: doesn't adjust to fit
(white solid on solid blue background)

(4) match boundaries + interpolate