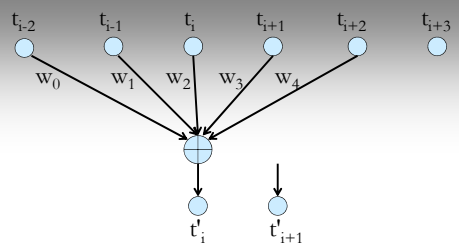
# Filters

## Motivating Example

- Consider we are tracking a fly
- Sensor reports the fly's position several times a second
- Some noise in the sensor
- Goal: reconstruct the fly's actual path
- Problem: can't rely on individual measurement due to noise
- How should we proceed?

## Tracking a fly, oh my!

- Note: there is coherence between the reported samples
- Looking at a few samples may give us a better picture
  - Noise may "cancel out"
- Instead of using a single sample, compute a weighted average of a couple of samples before and a couple after




- This construction is a type of *filter*
  - Moving weighted average
- Looks at multiple samples to adjust the output signal

## Moving Weighted Average Filter

- The weights define the behavior of a filter
- Weights must add to 1

## General Picture

## Demo

- Filtering noise with a simple box filter

## Iterated Filtering

- Can re-apply the filter
  - Take the output, and use it as the input to the filter
  - Called Iterated Filtering
- Applying a moving weighted average filter to itself multiple times will yield a filter with the shape of Gaussian Probability Distribution
- Demo
  - Iterated filtering on noisy sine wave
  - Iterated filtering of box filter

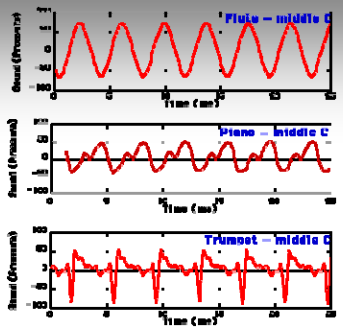
## Support

- The range or number of samples needed to compute the filter is referred to as the filter's *support*
  - Filter in example has support 5
- Generally want support to be *local*
  - i.e. not to need too many samples
  - Filter only reacts to local variation
  - Easier to compute

# More on Signals

## In Music…

- Lowest frequency is the *pitch*
  - Called fundamental frequency
- Additional harmonics will affect the sound
  - *Timbre* of the sound
  - Harmonic frequencies are an integer multiple of the fundamental frequency




Source: http://www.ux1.eiu.edu/~cfadd/1150/16Waves/char.html

## Frequency Bands

- Low pass filter
  - High frequency components are de-emphasized
  - Low frequency components kept the same
    - "passed"
  - Averaging filter is low pass
- High pass filter
  - Maintain high frequency, de-emphasize low
- Band pass
  - Filters can be tuned to any range of frequencies, or band
  - Pass that band and de-emphasize the other frequencies

# Working with Images

## Convolution (formal definition)

- Want to extend the idea of filters to 2D images
  - Many effects rely on using a pixel's neighbors to update its value
- (ADVANCED![1]) Convolution can be thought of as the integral of the effect of one function f (the filter) on a second function g (the image)
- In discrete representations, the filter and image are both grids
  - Do a summation instead of an integral
  - Easier to understand with an example

1. You are not responsible for this formal definition. It is included for completeness.

## Convolution (intuition)

- For every pixel
  - Replace pixel color with "average" of its neighbors
- Meaning of "average" can vary
  - In general, it is a "weighted average" where different pixels are given different importance, or weight
- Similar to applying a filter to a 2D image

## Convolution

- Convolution is done by replacing a pixel by the weighted sum of its neighbors
- e.g. a Sharpen filter can be defined as:

| | | |
|---|---|---|
| -1 | -1 | -1 |
| -1 | 9 | -1 |
| -1 | -1 | -1 |

  - Convoultion: pixel (i, j) = -pixel(i-1,j-1) -pixel(i-1,j) -pixel(i-1,j+1) -pixel(i,j-1) +9*pixel(i,j) -pixel(i,j+1) -pixel(i+1,j-1) -pixel(i+1,j) -pixel(i+1,j+1)
- Must be done for *every* pixel

## Example

- Left rect is intensity 200
- Right rect is intensity 100
- Apply unsharp mask

| | | |
|---|---|---|
| -1 | -1 | -1 |
| -1 | 9 | -1 |
| -1 | -1 | -1 |

## Convolution

- A Blur filter can be defined as:

| | | |
|---|---|---|
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |

- Filters can be any size
- The filter components must sum to 1
  - Avoids changing intensity

## Example

- Left rect is intensity 200
- Right rect is intensity 100
- Apply blur

| | | |
|---|---|---|
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |


## Blur

- What would I do if I wanted my image to be more blurred?
- Can increase the filter size
- Can apply it repeatedly

## Show some examples…

- Window filter
- Whole image

## Edge Detection Filter

- Edge detection:
  - Edges often marked by large differences in the value of adjacent pixels
  - In a copy image, store distance between adjacent pixels in the original image
  - Large differences often indicate an edge

## For every pixel, do:

```
// Pixel location and color
int loc = x + y*img.width;
color pix = img.pixels[loc];

// Pixel to the left location and color
int leftLoc = (x - 1) + y*img.width;
color leftPix = img.pixels[leftLoc];

// New color is difference between pixel and left neighbor
float diff = abs(brightness(pix) - brightness(leftPix));
destination.pixels[loc] = color(diff);
```

## Where will the previous code fail?

- Won't detect horizontal edges

## Example from Shiffman

## Processing Built in Filters

- Numerous built in image processing filters
- Command:
  - filter(<mode>);
  - filter(<mode>, <level>);
- <mode> :
- THRESHOLD, GRAY, INVERT, POSTERIZE, BLUR, OPAQUE, ERODE, DILATE

## Example Program

## Antialiasing

- Rasterizing an image or font creates aliases
  - Jagged borders that should be smooth
- Antialiasing creates a more visually appealing image by slightly blurring the edges
- Implemented in Processing
  - Command: smooth()