Sensors and Filtering

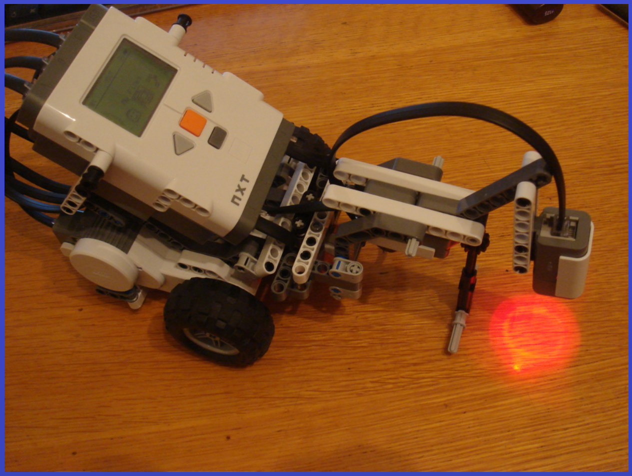
McGill

McGill

S&D - 1

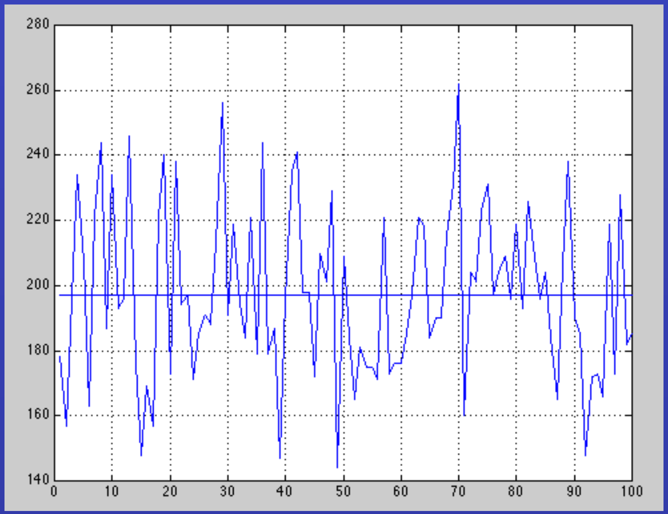
Sensors and Data Acquisition

- Measured signal = Signal + Noise
- *Knowledge* about the structure of noise can be used to design *filters* for noise removal.
- Simple (but effective) filters can often be derived from first and second order statistics computed from signal samples.



Experiment

- Write a program to drive tribot with light sensor across a carpet with uniform color. What does the signal look like?
- We'll show you how to log data c captured from the brick in the next few slides. This data can then be copied into an Excel spreadsheet with a couple of mouse clicks.



S&D - 2

```
+import lejos.hardware.Button;

// A simple demonstration of data acquisition with the EV3
// using the remote console utility to collect data.

public class DataAcquisition implements TimerListener{

// Class Constants
        public static final int SINTERVAL=100;  // Sampling interval (mS)
        public static final int NSAMPLES=100;  // Number of samples to acquire
        public static final int TIMEOUT=40000;  // Fail if no comms by this time (mS)
        public static final int FWDSPEED=15;     // Forward motion speed (deg/sec)
        public static final int SLEEPINT=500;     // Main thread sleeps for 500 (mS)

// Class variables (persistent)
        public static int numSamples;
        public static int currentSample;

// Set up instances of the Text Display, left and right Motors

        static TextLCD t = LocalEV3.get().getTextLCD();
        static RegulatedMotor leftMotor = Motor.A;
        static RegulatedMotor rightMotor = Motor.D;
```

S&D - 3

McGill

```
// Allocate ports for the Color and Touch sensors.

        static Port portColor = LocalEV3.get().getPort("S1");
        static Port portTouch = LocalEV3.get().getPort("S2");

// Attach instances of Color and Touch sensors to specified ports.

        static SensorModes myColor = new EV3ColorSensor(portColor);
        static SensorModes myTouch = new EV3TouchSensor(portTouch);

// Get an instance of a sample provider for each sensor.  Operating
// modes are specified in the constructor.  Note that the color
// sensor is set to return the intensity of the reflected light.

        static SampleProvider myColorSample = myColor.getMode("Red");
        static SampleProvider myTouchStatus = myTouch.getMode(0);

// Need to allocate buffers for each sensor

        static float[] sampleColor = new float[myColor.sampleSize()];
        static float[] sampleTouch = new float[myTouchStatus.sampleSize()];
```

McGill

S&D - 4

```
// Entry point
        public static void main(String[] args) throws InterruptedException {
                boolean rolling = true;             // cart moves while true
                int status;
                numSamples=0;

// Set up display area
                t.clear();
                t.drawString("Data Acquisition Demo",0,0,false);
                t.drawString("Remote stream...",0,2,false);
                t.drawString("# Samples ",0,4,false);
                t.drawString("Last Val. ",0,5,false);

// Set up timer interrupts
                Timer myTimer = new Timer(SINTERVAL,new DataAcquisition());

// Start cart rolling...
                leftMotor.setSpeed(FWDSPEED);
                rightMotor.setSpeed(FWDSPEED);
                leftMotor.forward();
                rightMotor.forward();

// Enable exception handler
                myTimer.start();
```

S&D - 5

```
// The Main thread continually updates the display
// and checks for stop
                    while(rolling) {
                      status=Button.readButtons();
                      if ((status==Button.ID_ENTER)||(numSamples>=NSAMPLES)) {
                              System.exit(0);
                      }
                    t.drawInt(numSamples,4,11,4);      // show current count
                    t.drawInt(currentSample,4,11,5);   // and current value read
                    Thread.sleep(SLEEPINT);            // sleep 'till next cycle
                              }
                    }

          @Override
          public void timedOut() {

// Acquire sample and write immediately to remote console.

                    myColorSample.fetchSample(sampleColor,0);
                    numSamples++;
                    System.out.println(numSamples + ", " + sampleColor[0]*1000);

          }
}
```

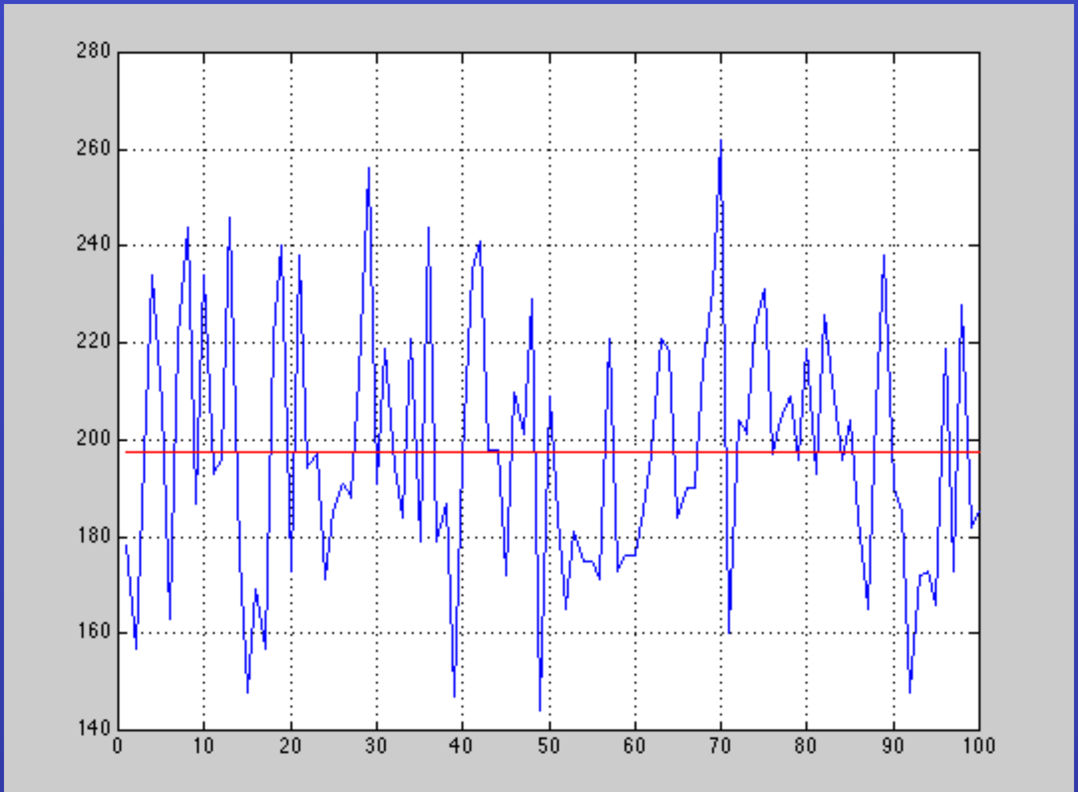
McGill

S&D - 6

Sensors and Data Acquisition, cont.

• Let's look at the data acquired by the Tribot in detail.

• Excel can do plots as well as provide Descriptive Statistics



Column1	
Mean	197.927835
Standard Erro	2.66513248
Median	196
Mode	221
Standard Devi	26.2485108
Sample Varian	688.984321
Kurtosis	-0.51042855
Skewness	0.2380261
Range	118
Minimum	144
Maximum	262
Sum	19199
Count	97

Basic Statistics and their Usage

- The formulae for the *mean* and *standard deviation* are respectively

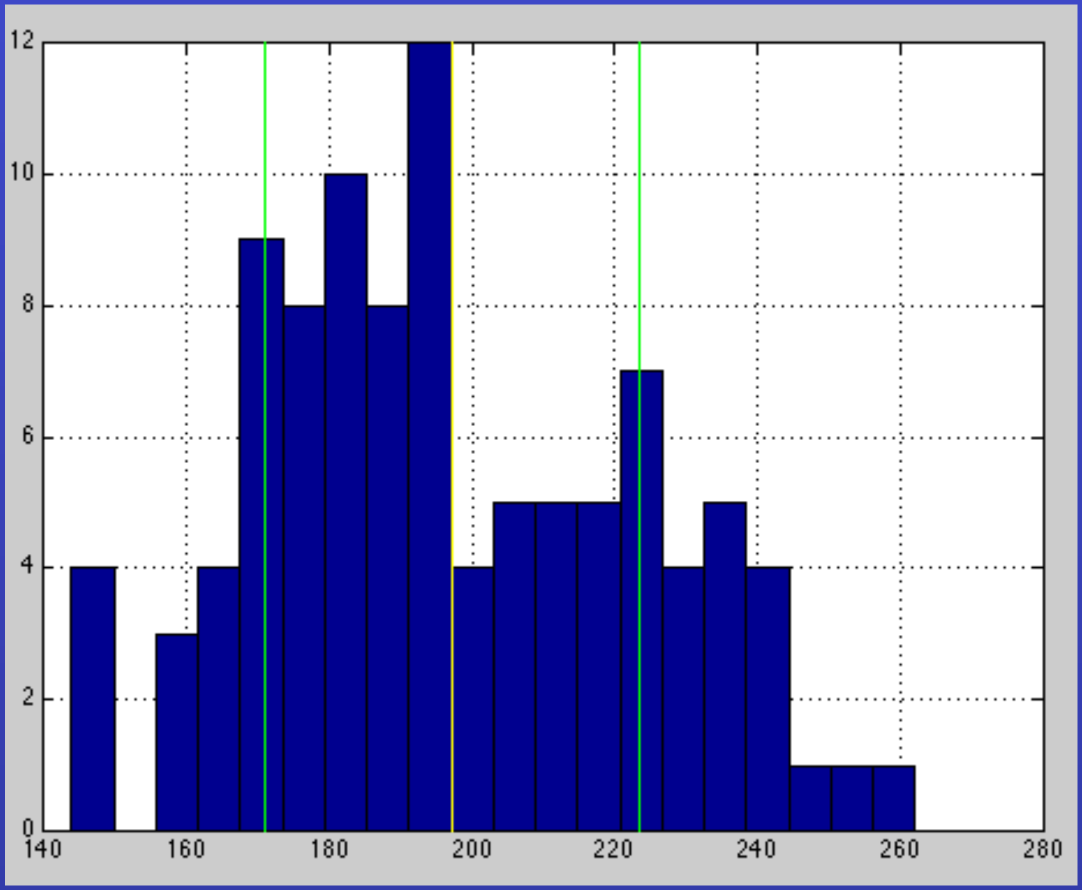
$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}{n-1}}$$

- In the presence of noise, the mean can often serve as a good estimate of the uncorrupted signal (provided there is no systematic error or bias in the measurements).
- The standard deviation is an indicator of how the error is dispersed about the mean. It is often useful in rejecting measurements that fall significantly out of the expected error distribution. More about this later.

Basic Statistics and their Usage, cont. 1

• Another way of viewing the variation of data is by means of a *histogram.*



- Assume y = f (x)
- Divide y into n evenly spaced bins; set counts to 0.
- for each x_i
    1. determine y_i = f (x_i)
    2. find bin;
       increment count
- Plot counts vs. bins
- Height = # values in each bin
- Yellow - mean(x)
- Green - mean +/- standard deviation

Basic Statistics and their Usage, cont. 2

- A Gaussian (normal) distribution is often used to represent the characteristics of real (physical) noise. The equation which describes this function is shown below.

$$N(x, \bar{x}, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left(\frac{(x-\bar{x})^2}{2\sigma^2}\right)}$$

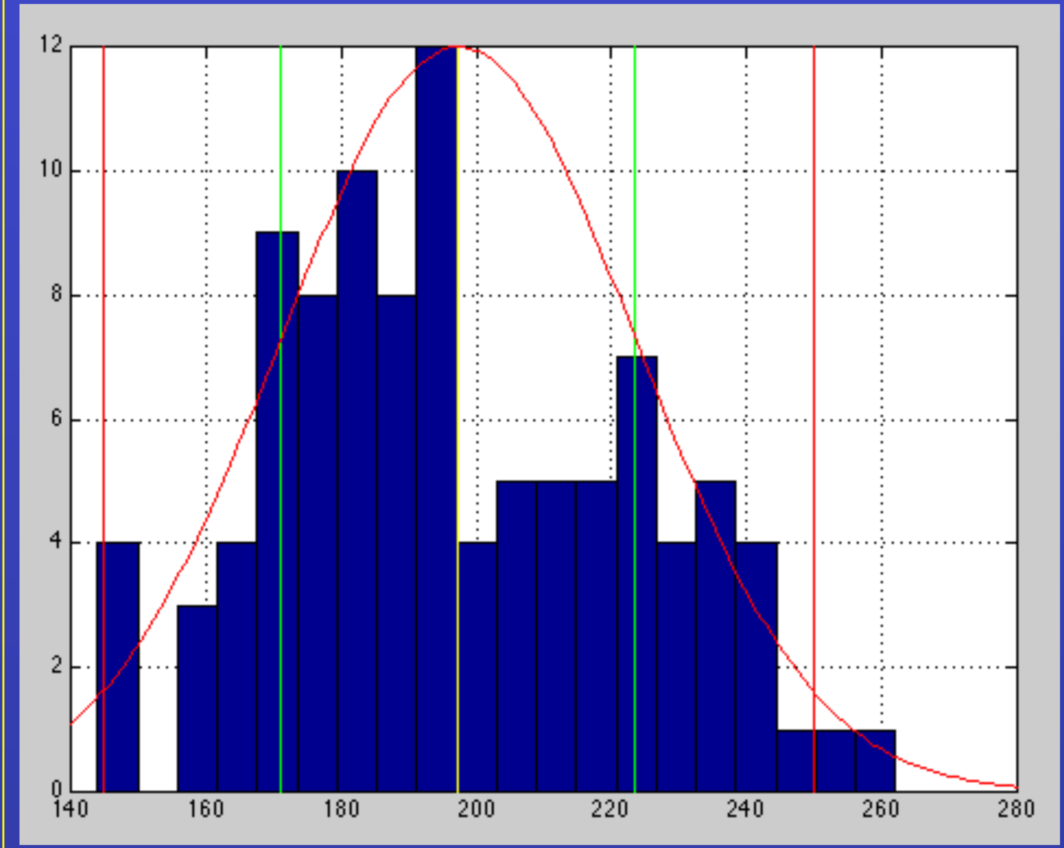
- The function is completely described by two parameters, the mean and the standard deviation.
- To see whether the Gaussian is an adequate representation of the signal we encountered earlier, we will plot this function against the data histogram, using the mean and standard deviation computed from the data as the function's parameters.
- We will also adjust the constant multiplying the exponential so that the function has the same maximum value as the histogram.

McGill

S&D - 10

Basic Statistics and their Usage, cont. 3

- The histogram with Gaussian approximation super-imposed



- Decent but slightly skewed fit
- The mean is used to approximate the true value of the measurement
- Outlier = data that do not fall in the interval [mean-2*std,mean+2*std]

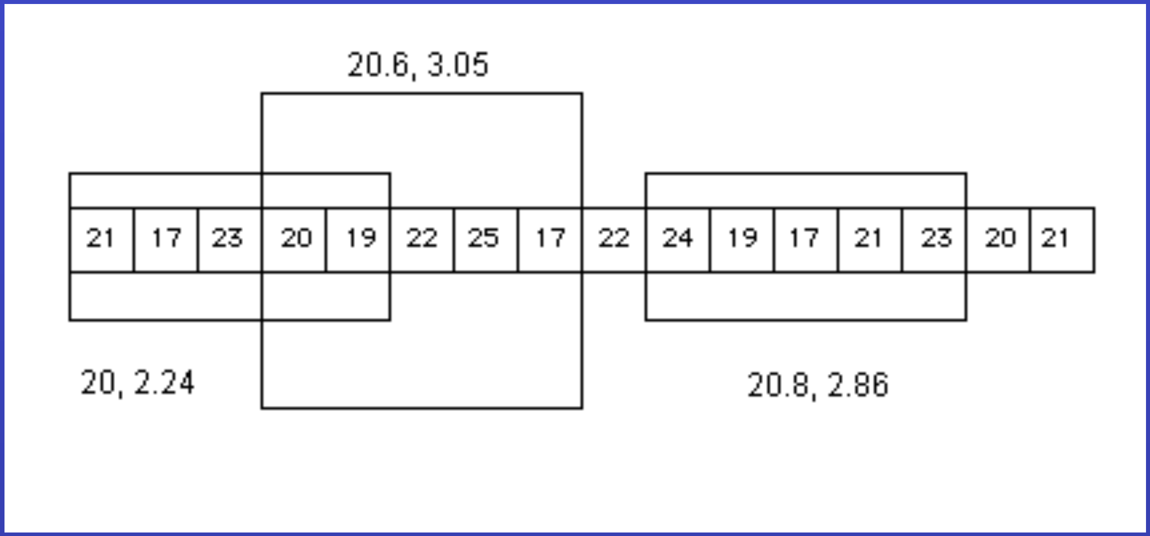
$$N(x, \bar{x}, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left(\frac{(x-\bar{x})^2}{2\sigma^2}\right)}$$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}{n-1}}$$

Computing on the Fly

- The preceding discussion assumes that all of the data are available when computing statistics. In robotics and control applications, signals can vary in space and time. Statistics are computed in a *moving window*.



- In the example shown earlier, averages were taken over space
- Window size is a tradeoff. Too big smooths out the structure of the signal, but too small makes measurement more sensitive to noise.
- Solution: for the purposes of this course, window size can be determined empirically.

S&D - 12

Recursive Technique for estimating a Moving Average, M. Tham, Newcastle Upon Tyne

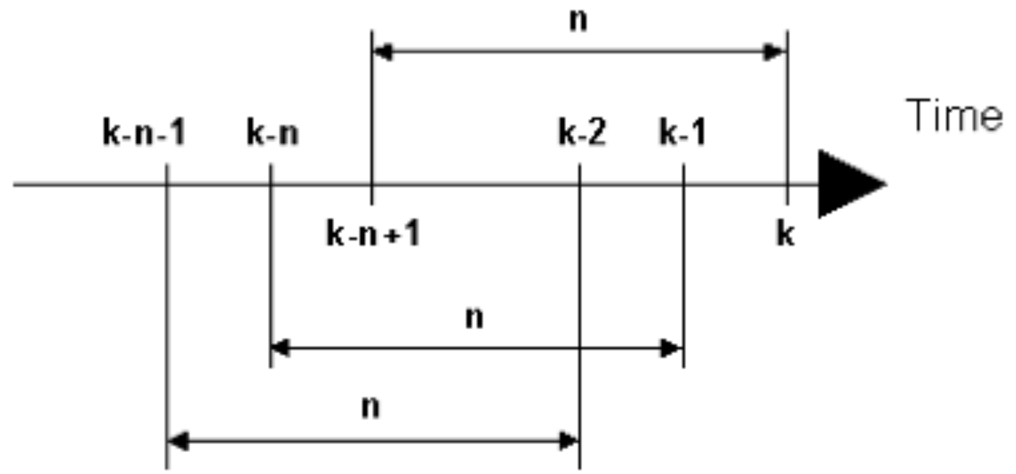


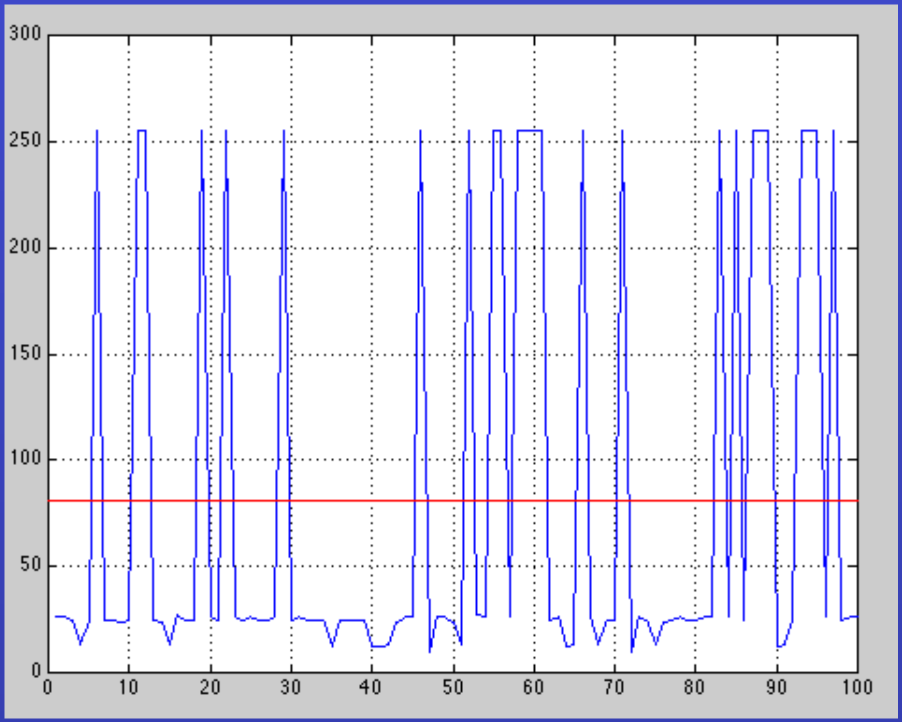
Figure 1 Moving Window of *n* Data Points

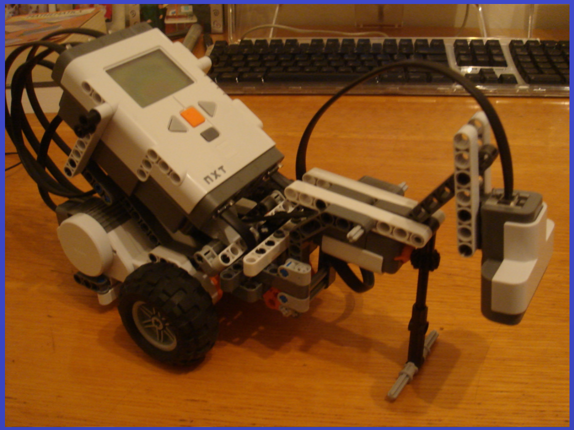
$$\bar{x}_k = \bar{x}_{k-1} + \frac{1}{n}\left[x_k - x_{k-n}\right]$$

McGill

S&D - 13

Ultrasonic Sensor and Outliers - another experiment



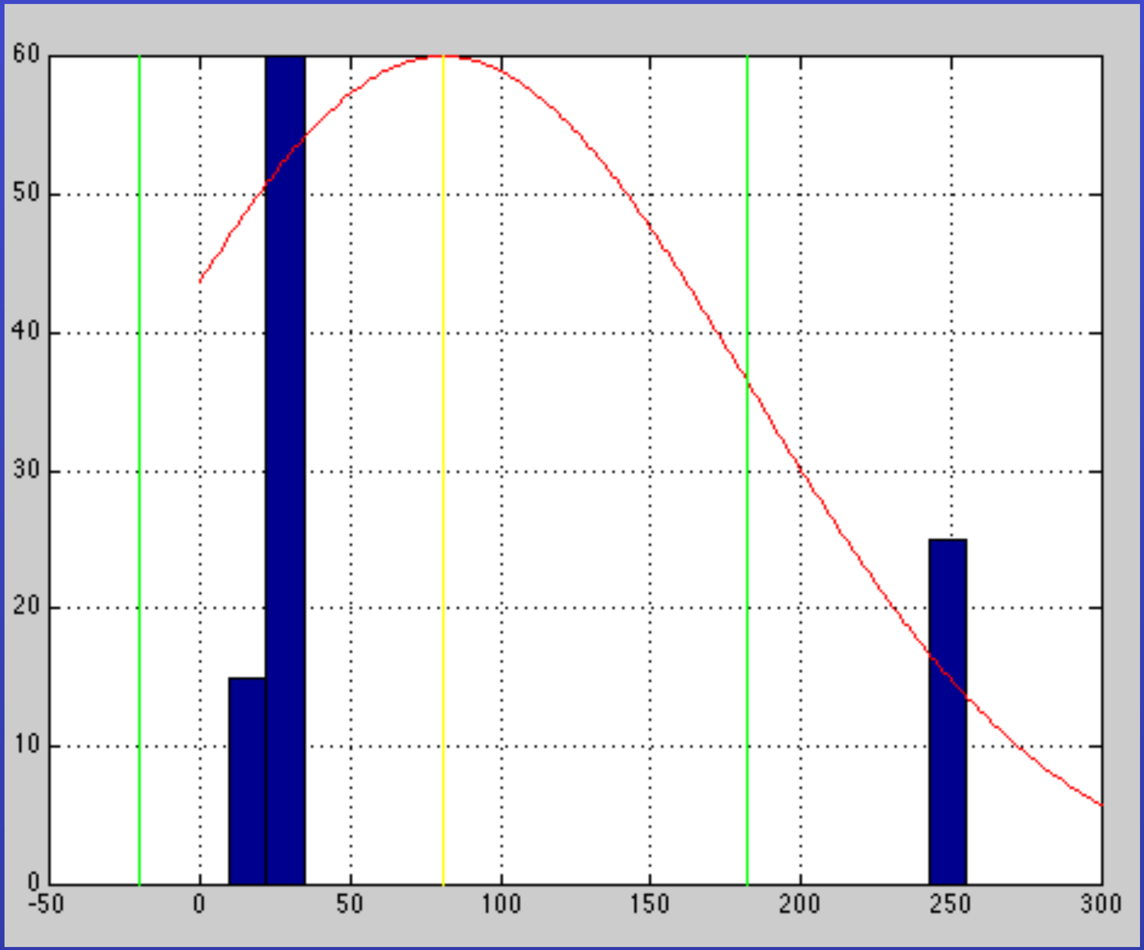


- Drop outs in the return signal read is maximum distance 255
- Cannot use the mean to estimate the true signal - noise no longer follows a Gaussian distribution.

Actual distance on the order of 25!

McGill

Outliers in the Ultrasonic Data

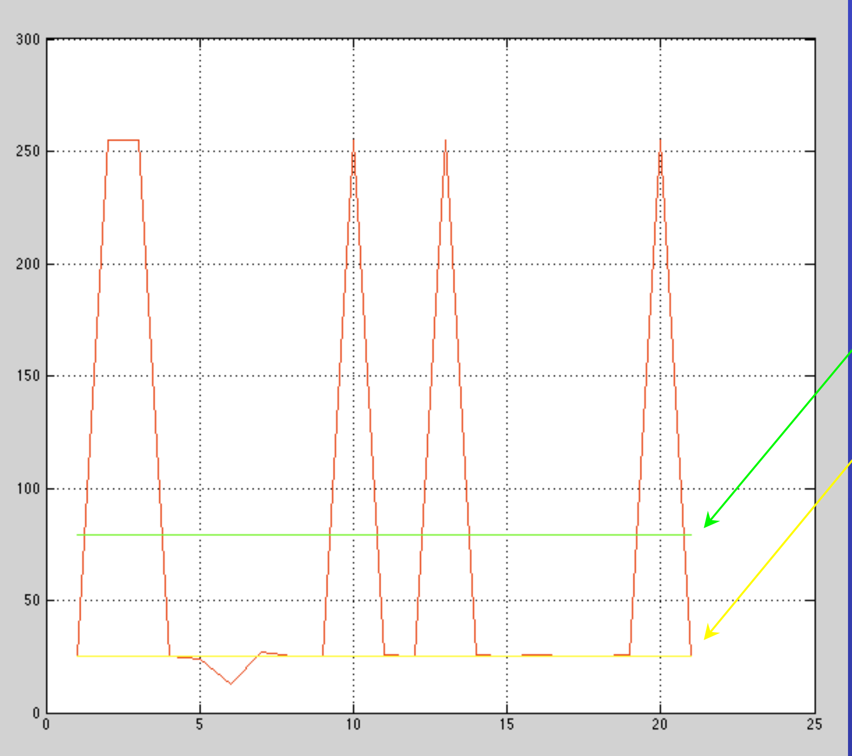


- There are essentially 2 peaks in the histogram.
- In this case, a Gaussian is a poor fit to the data
- In general, need to remove outliers *before* computing statistics
- Need a different filter model

Dealing with Outliers: Non-Linear Filtering

- Consider the following sequence of numbers from the US data set

  {25 255 255 25 24 13 27 25 25 255 26 25 255 26 25 26 25 25 26 255 25}



Average = 79.43

Median = 25

Which is a better characterization of the signal?

What condition must hold over the data for this scheme to work?

The Median Filter

- Consider a moving window, 5 samples wide
- Calculate the median of the window values
- If the sample is > the median, replace its value by the median

25 255 255 25 24 13 27 25 25 255 26 25 255 26 25 26 25 25 26 255 25

24 25 25 255 255	13 24 25 255 255	13 24 25 27 255
(25)	(25)	(25)
25 255 255 25 24	25 25 255 25 24	25 25 25 25 24

Median ⇒ The value of the center position of an ordered list (odd)

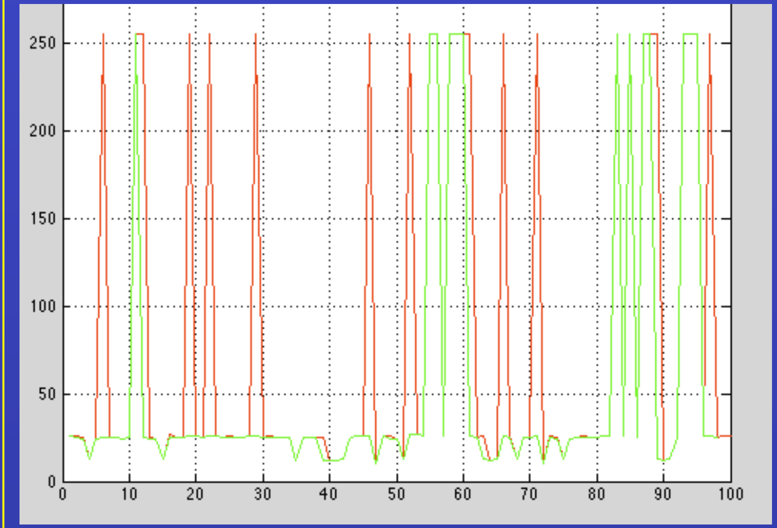
The average of the 2 center elements of an ordered list (even)

McGill

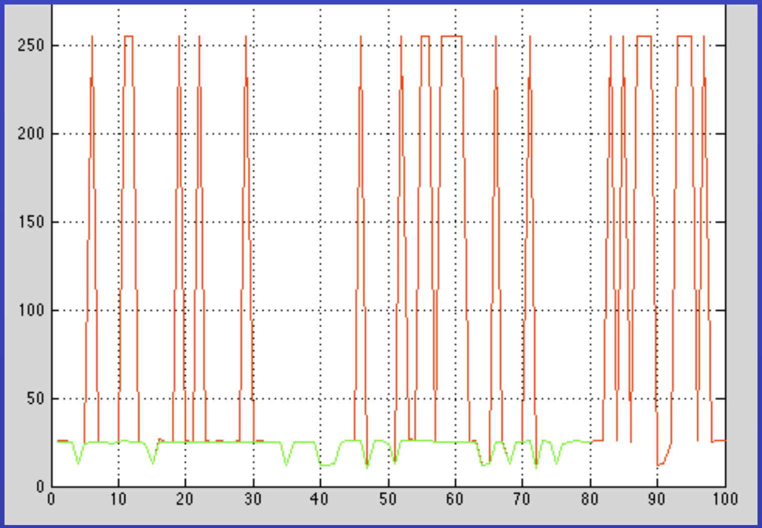
The Median Filter

- Single parameter - width of moving sample window, W
- Need to know noise characteristics in order to choose
- Specifically, # Outliers < = int(W/2) for W Odd, else < int(W/2) for W even

Examples:



Filter Width = 3

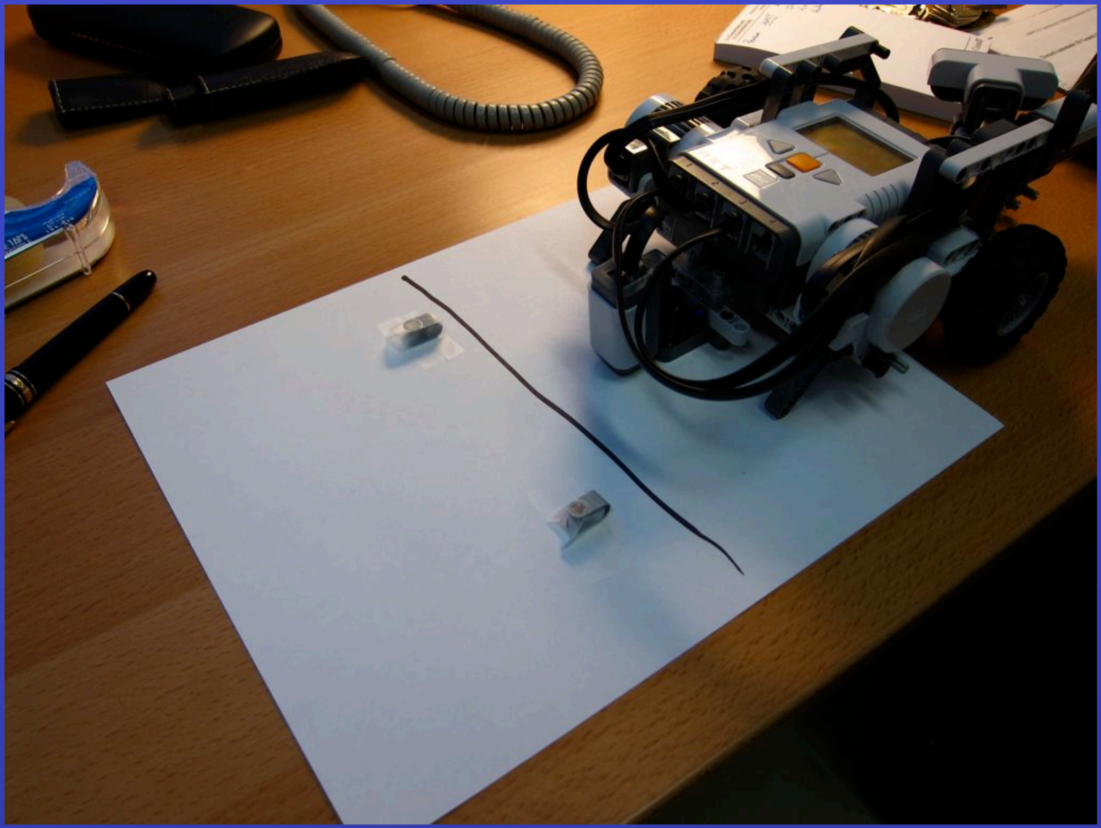


Filter Width = 21

A Simple Median Filter (Matlab)

```
function fv = median_fiter(iv,fw)
% Applies a median filter of length fw on input vector iv.
% Returns filtered output.
vlen=length(iv);
for i=1:vlen-fw+1
    sv=sort(iv(i:i+fw-1));
    med_val=sv(int32(fw/2));
    if (iv(i)>med_val)
        fv(i)=med_val;
    else
        fv(i)=iv(i);
    end
end
```

- Limitation: The filter width, fw, must be an odd number for this to work.

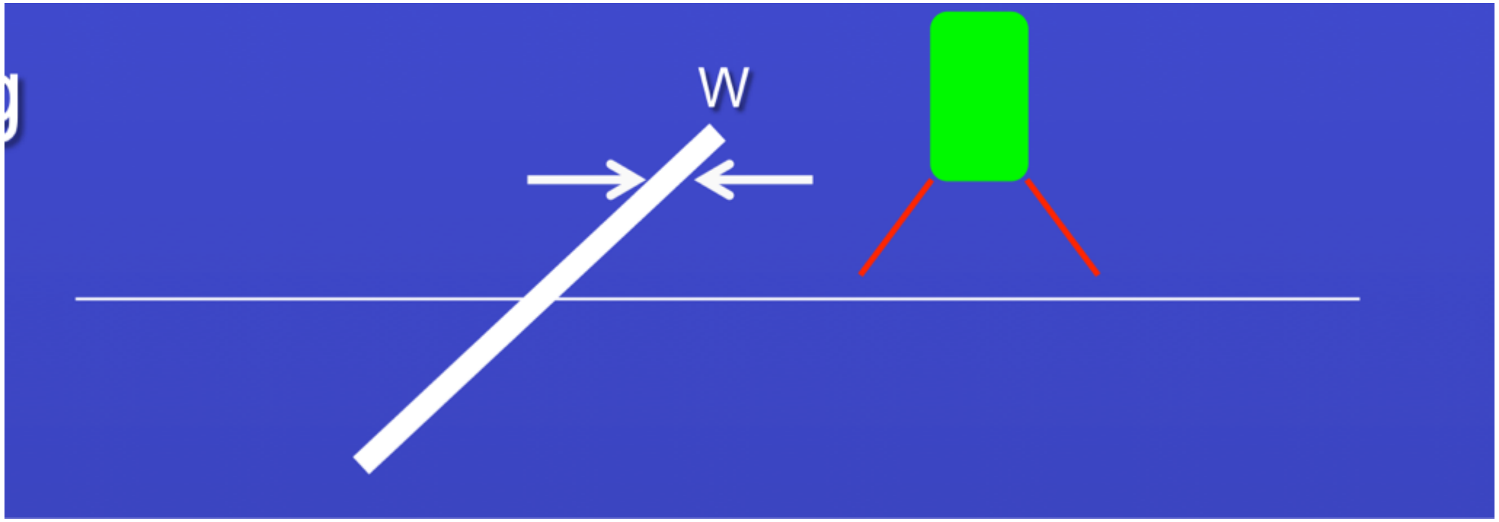


Problem

Determine the precise instant when the robot crosses the black line.

Two main issues:

- sampling – how many samples acquired at what rate are sufficient for detecting the line.
- detection – given a set of samples, determine if a line is present and at what location.



$V \triangleq$ velocity of robot

$T \triangleq$ time to traverse line

$w \triangleq$ line width.

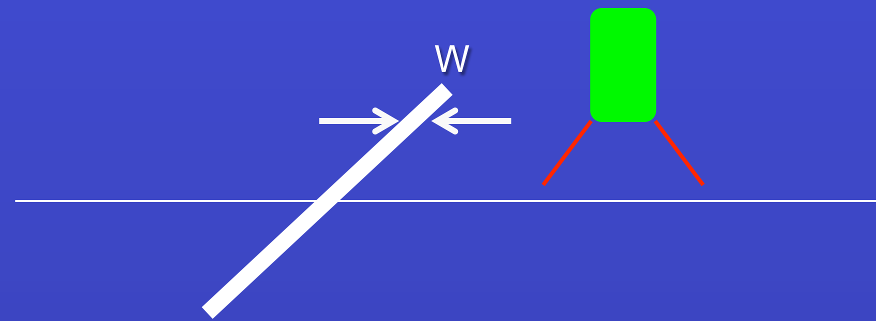
$$W_{cm} = V_{cm/sec} \, T_{sec}$$

$$V_{cart} = \frac{2\pi R_w \Omega^\circ/sec}{360^\circ} = \frac{\pi R_w \Omega}{180} \; cm/sec$$

$$w = \frac{\pi R_w \Omega}{180} T \qquad T = \frac{180 \times w}{\pi R_w \Omega}$$

$$f = \frac{1}{T} = \frac{\pi R_w \Omega}{180 \, w}$$

Sampling



Problem: Detect the presence of a Line of width W perpendicular to the direction of motion.

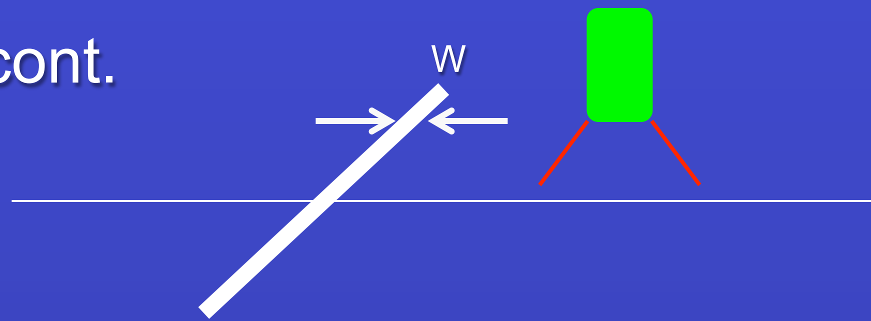
Assuming that the line contrasts with the background, one must take at least one sample every W units along the direction of travel. The sampling rate is a function of both W and the velocity of travel, V, i.e., the sampling period $T = W / V$, or equivalently, the sampling frequency $F = V / W$.

For our two-wheeled robot: $V = (2 \pi R \Omega) / 360$, where R is the radius of the wheels, and Ω the wheel velocity in degrees/second. Thus $F = (\pi R \Omega) / (180 W)$, provided that both R and W are expressed in the same units.

For a two-wheeled robot with R = 27 mm, moving with a velocity of 360°/sec, to detect a line of 6.35 mm, F must be at least $(\pi \times 27 \times 360) / (180 \times 6.35) = 26.72$ Hz.

McGill

Sampling cont.

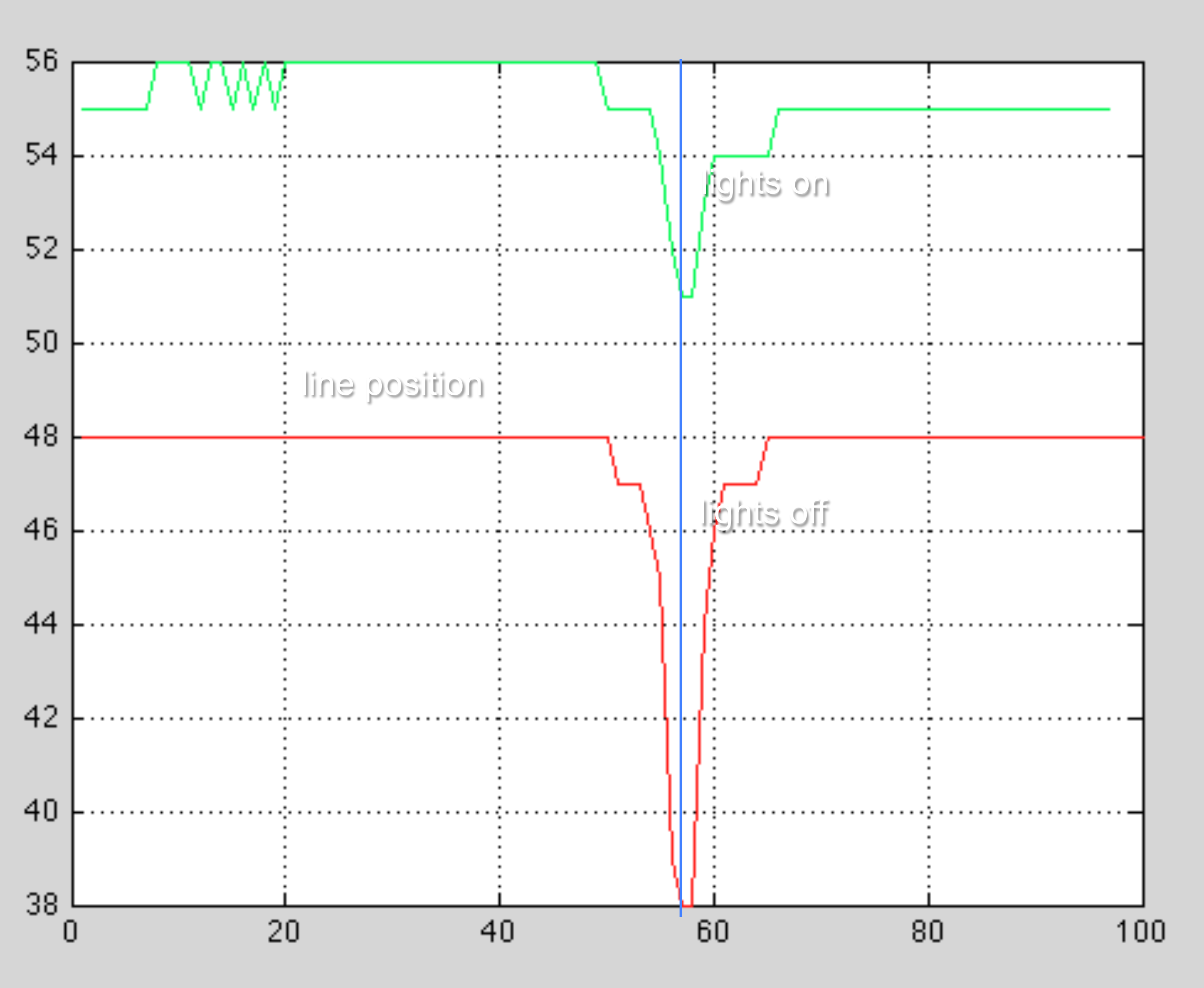


Observations:

1. A line is detected by observing a transition from bright to dark to bright (it is sufficient to detect a bright to dark transition). For the conditions of the previous slide and a sampling rate of 26.72 Hz, this means two samples per line crossing. In practice one would desire at least 6 or 8 samples to ensure adequate detection.
2. Upping the sampling rate by a factor of 3 to 4, e.g., 80.15 Hz or 106.0 Hz would provide an adequate sample, but severely load the processor (unless one was very careful).
3. Alternately, one could slow down the travel velocity or else combination of upping the sampling rate and reducing velocity.

Although this is a simple example, it shows that key performance characteristics (such as maximum velocity) are constrained by fundamental physical limits.

S&D - 22



Two sample runs: one with room lights off, other with lights on.

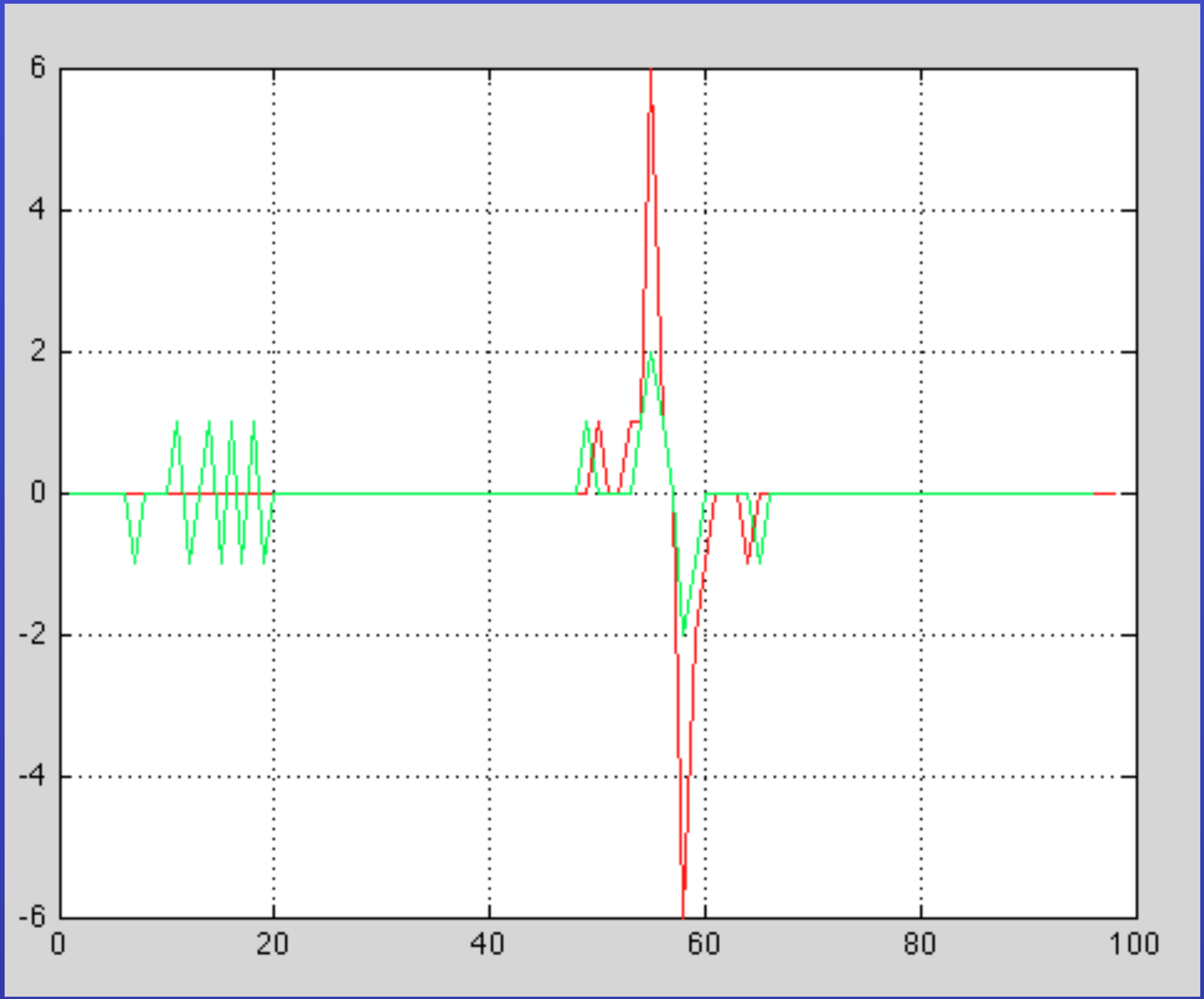
Detection

Based only on numerical values, detect the black line from the data.

Problem:

Cannot rely on absolute values!

Method of finite differences (a.k.a. differentiation)



Discrete Derivative

$d_1 = 0$

for i = 2 to N

$d_i = x_i - x_{i+1}$

Notice that the influence of ambient light is factored out.

Magnitude of difference used to identify presence of line.

Discrete Derivative

$X = \{x_0, x_1, \ldots, x_n\}$ (sampled data)

Continuous function $\Big\}$ $$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Discrete function $\Big\}$ $$f'(x_i) = \lim_{\Delta\, min} \frac{f(x_i + \Delta) - f(x_i)}{\Delta}$$

$$\approx f(x_{i+1}) - f(x_i)$$ discrete difference

Discrete Derivative: cont. $f'(x_i) \approx f(x_{i+1}) - f(x_i)$

Assumptions:

1. $\{X\}$ are samples from a continuous function
2. x_{i+1} and x_i are a constant distance apart.

$\Rightarrow$ 1. Apply a filter to $\{X\}$ to remove outliers (e.g. smooth via averaging)

2. Sample at constant frequency

30 40 40 41 39

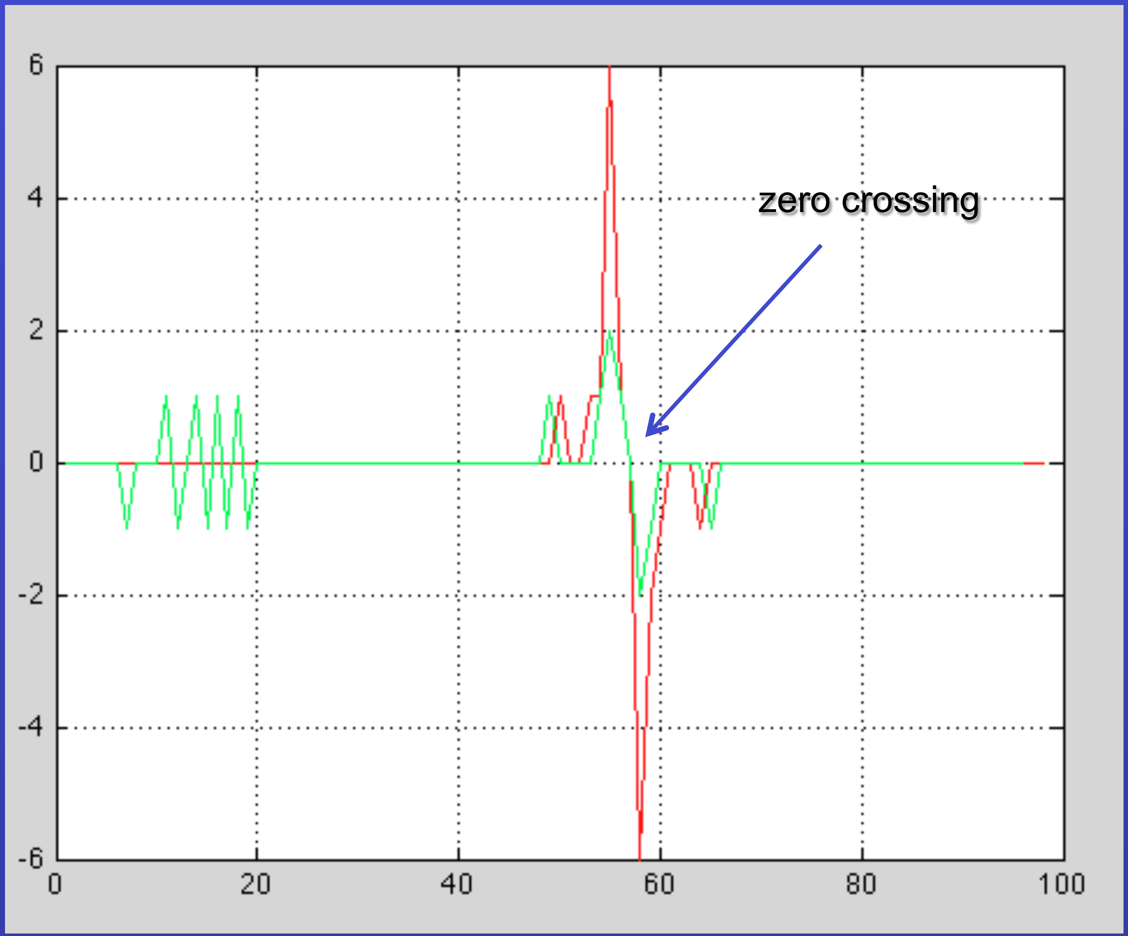
.25 .5 .25

$-\frac{1}{4}$ $\frac{1}{2}$ $\frac{1}{4}$

S&D - 24

McGill

Where, exactly, is the precise edge location?



Key Observations

Locations where the derivative goes through 0, i.e., zero crossings, indicate the presence of a line:

\+ 0 - b lack line on white

\- 0 + white line on black

The magnitude of the zero crossing indicates the strength of the line response.

This is an example of PATTERN RECOGNITION