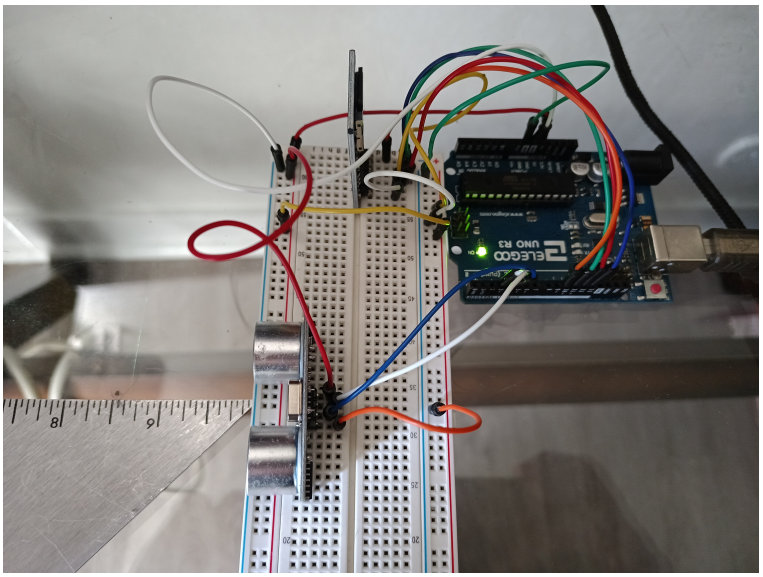
Sensitivity Analysis of a Kalman Filter for an Ultrasonic Sensor in ArduinoC

Steven Sharp

July 29, 2023

AERO 465 - Project 1

In this project, a basic Kalman filter was implemented in ArduinoC to improve the distance measurements of an HC-SR04 ultrasonic sensor and investigate how changing the filter parameters affects the accuracy and response time of the measurements.

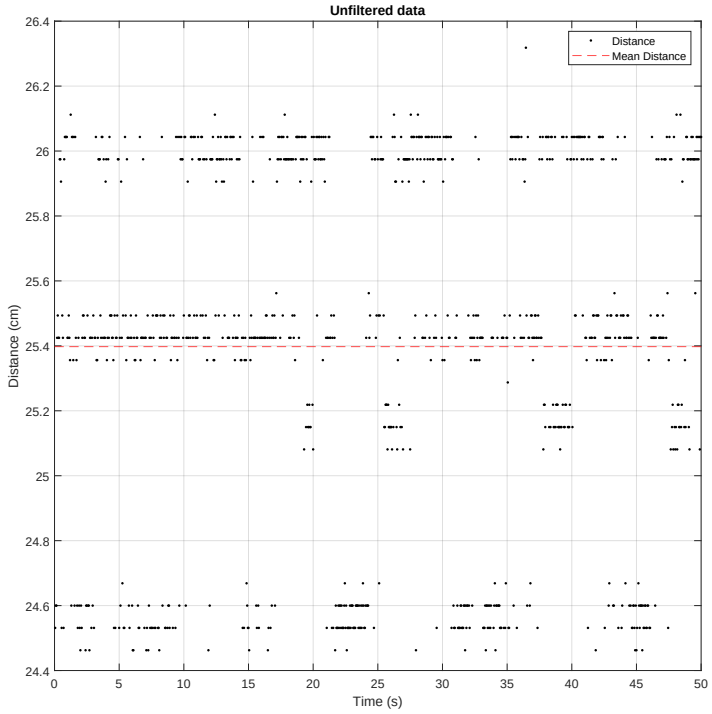


Initial Data

The ultrasonic sensor was placed $10''$ or 25.4 cm away from the wall and perpendicular to its surface. The initial measured distance before calibration had a mean of 25.8 cm, so an offset of -0.4 cm was applied to all sensor measurements.

The unfiltered set consisted of 1000 measurements, taken every 50 ms for a total duration of 50 s, with no modification to the data. The unfiltered set had the following parameters:

$$\mu = 25.3975\,\text{cm}$$
$$R = 0.3023\,\text{cm}^2$$
$$\sigma = 0.5498\,\text{cm}$$



Control Set

The Kalman filter was constructed using the following parameters:

R	0.3023
P_{k0}^{-}	0.1
$\hat{x}_{k0}^{-}$	25.4 cm
Φ	1
H	1
w	0
Q	0 s

The Kalman filter was then applied to a new set of measurements using the following code:

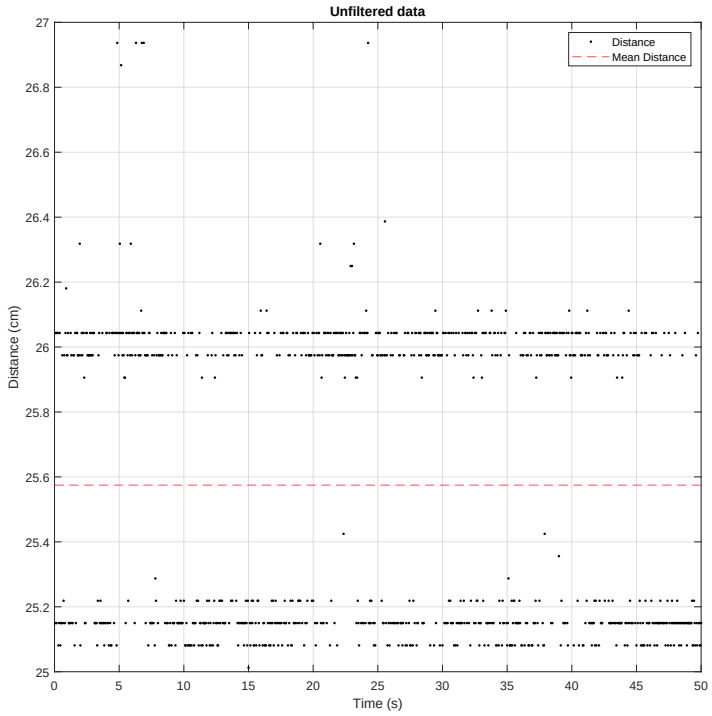
```
double distance = sonar.ping() / (2 * 29.1) + calibration;

K_k = P_k0 * H_k / (H_k * P_k0 * H_k + R);
x_k = x_k0 + K_k * (distance - H_k * x_k0);
P_k = (1 - K_k * H_k) * P_k0;
x_k0 = st_k * x_k;
P_k0 = st_k * P_k0 * st_k + Q_k;
```

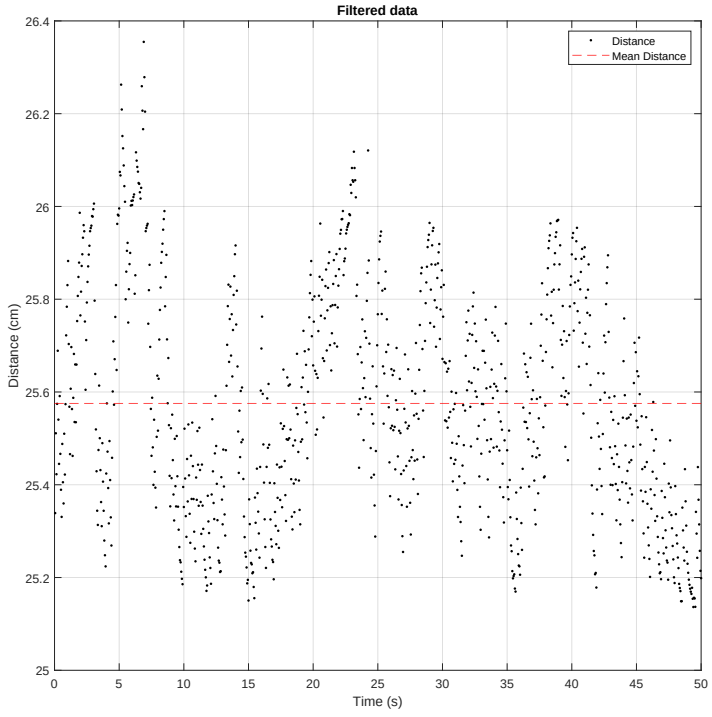
where st_k = Φ. Note that the variable 'distance' is halved to account for the travel time back to the sensor and converted from µs to cm using an approximate ratio from the speed of sound.

This results in the two data sets plotted on the subsequent page with the following parameters:

	Unfiltered	Filtered
μ	25.5746 cm	25.5753 cm
R	0.2047 cm^2	0.0589 cm^2
σ	0.4524 cm	0.2427 cm

This Kalman filter was able to smooth the bimodal distribution of the measured distance, and improved the standard deviation of the distribution by 46%.


Unfiltered data
Distance
Mean Distance
Distance (cm)
Time (s)
27
26.8
26.6
26.4
26.2
26
25.8
25.6
25.4
25.2
25
0
5
10
15
20
25
30
35
40
45
50


Filtered data
Distance
Mean Distance
Distance (cm)
Time (s)
26.4
26.2
26
25.8
25.6
25.4
25.2
25
0
5
10
15
20
25
30
35
40
45
50

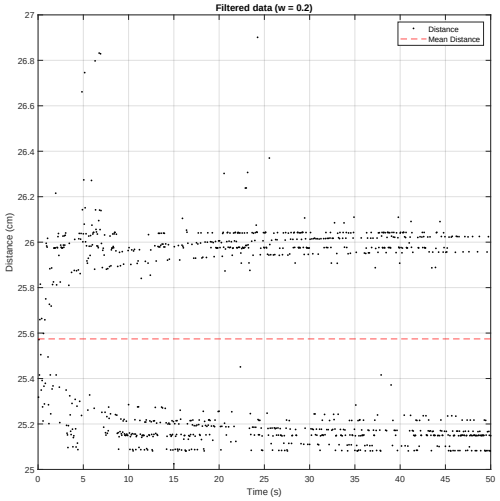
Sensitivity Analysis We begin by using the same control set of unfiltered data and constructing the same Kalman filter with the following parameters:

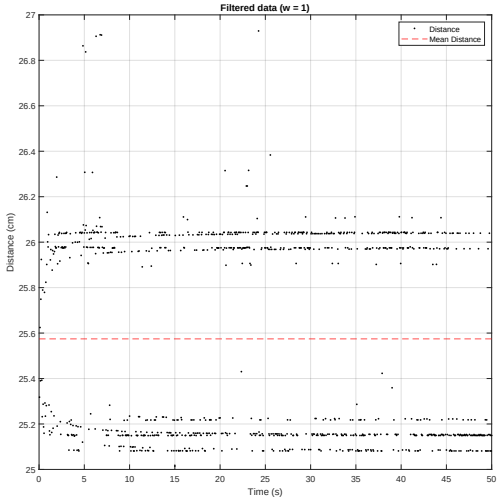
R	0.3023
P_{k0}^{-}	0.1
$\hat{x}_{k0}^{-}$	25.4 cm
Φ	1
H	1
w	0
Q	0 s

Parameters are then changed to extreme values one-by-one to observe the effects on the filter data.

First, increasing the power of the model noise massively reduces the performance of the filter on the bimodal distribution, and even small values of w mean that the filter barely improves the standard deviation by just 5.9%:

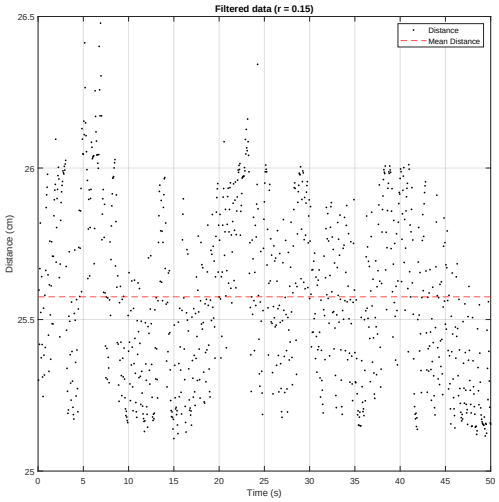
	Unfiltered	$w = 0$	$w = 0.2$	$w = 1$
Q	N/A	0 s	0.01 s	0.05 s
μ	25.5746 cm	25.5753 cm	25.5743 cm	25.5745 cm
R	0.2047 cm^2	0.0589 cm^2	0.1838 cm^2	0.1981 cm^2
σ	0.4524 cm	0.2427 cm	0.4257 cm	0.4450 cm

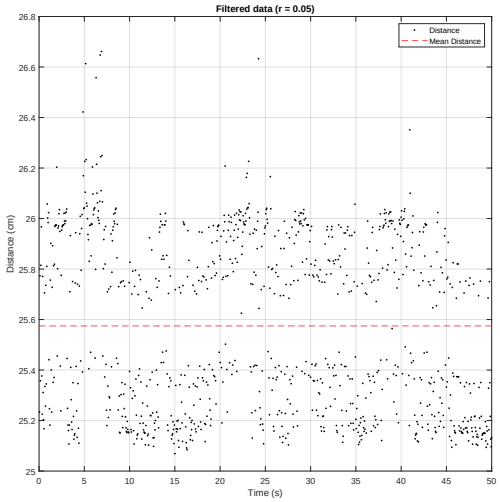




Decreasing the variance has a similar effect, but it is much less sensitive than the model noise:

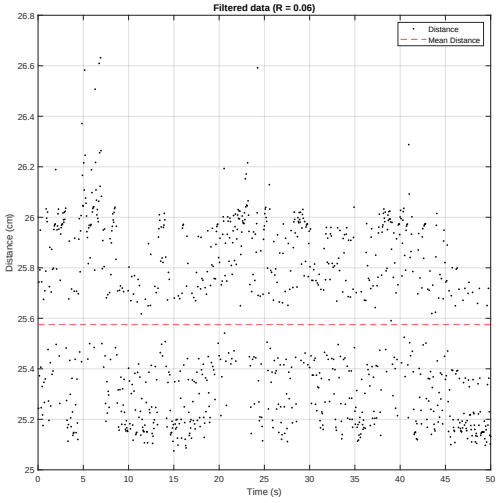
	Unfiltered	$R = 0.3023$	$R = 0.15$	$R = 0.05$
μ	25.5746 cm	25.5753 cm	25.5750 cm	25.5748 cm
R	0.2047 cm^2	0.0589 cm^2	0.0784 cm^2	0.1209 cm^2
σ	0.4524 cm	0.2427 cm	0.2799 cm	0.3477 cm

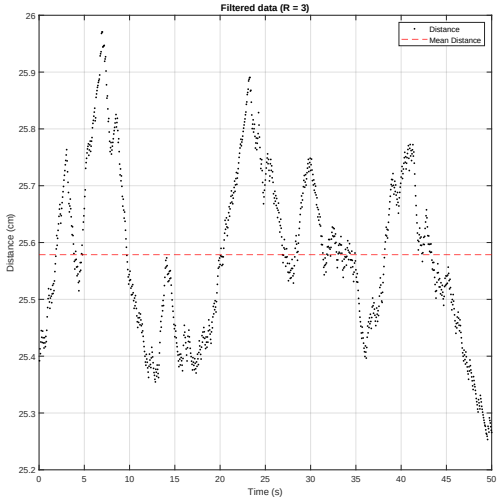




Conversely, we may improve the performance of the filter by increasing the variance:

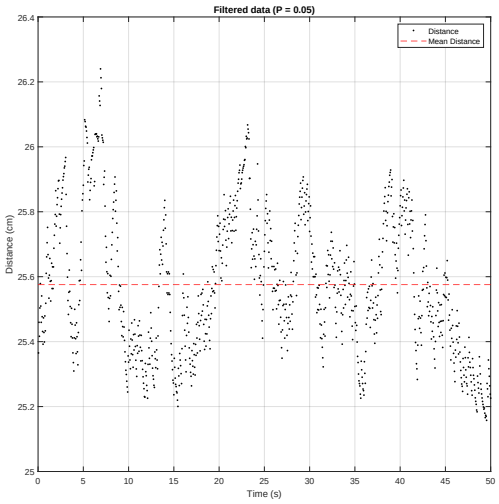
	Unfiltered	$R = 0.3023$	$R = 0.6$	$R = 3$
μ	25.5746 cm	25.5753 cm	25.5757 cm	25.5787 cm
R	0.2047 cm^2	0.0589 cm^2	0.0462 cm^2	0.0215 cm^2
σ	0.4524 cm	0.2427 cm	0.2150 cm	0.1466 cm

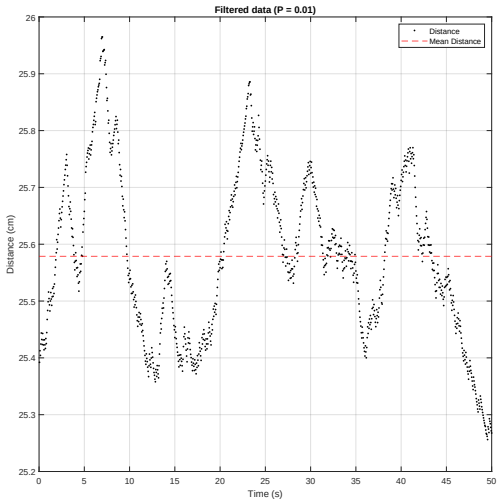




Since the actual distance is very close to the initial guess, decreasing the error improves the performance of the filter:

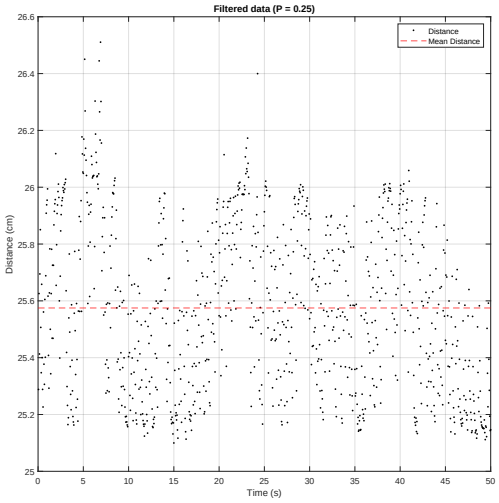
	Unfiltered	$P_{k0}^{-} = 0.1$	$P_{k0}^{-} = 0.05$	$P_{k0}^{-} = 0.01$
μ	25.5746 cm	25.5753 cm	25.5757 cm	25.5748 cm
R	0.2047 cm^2	0.0589 cm^2	0.0457 cm^2	0.0210 cm^2
σ	0.4524 cm	0.2427 cm	0.2138 cm	0.1450 cm

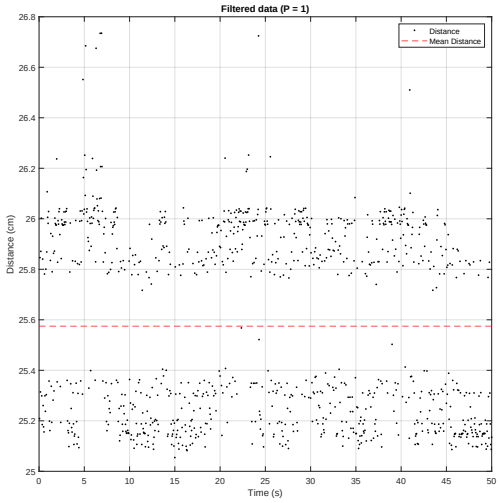




Conversely, increasing the error reduces the performance of the filter:

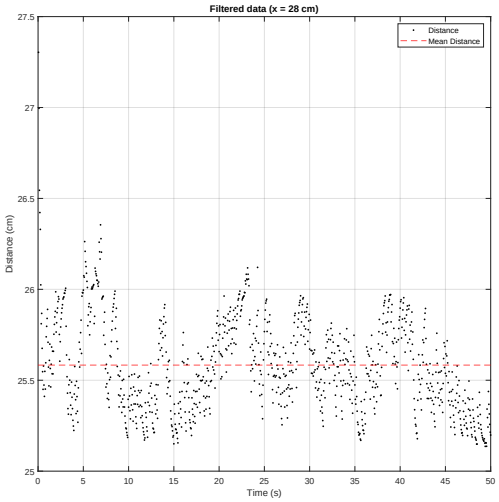
	Unfiltered	$P_{k0}^{-} = 0.1$	$P_{k0}^{-} = 0.25$	$P_{k0}^{-} = 1$
μ	25.5746 cm	25.5753 cm	25.5769 cm	25.5744 cm
R	0.2047 cm^2	0.0589 cm^2	0.0848 cm^2	0.0210 cm^2
σ	0.4524 cm	0.2427 cm	0.2913 cm	0.1450 cm

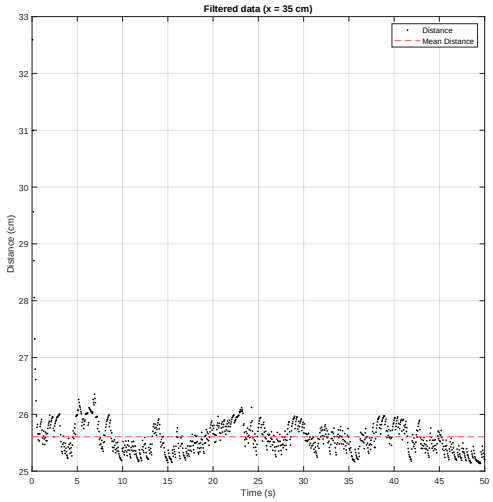




Surprisingly, giving a bad initial guess doesn't affect the filter as much as variance since it quickly returns to the actual mean:

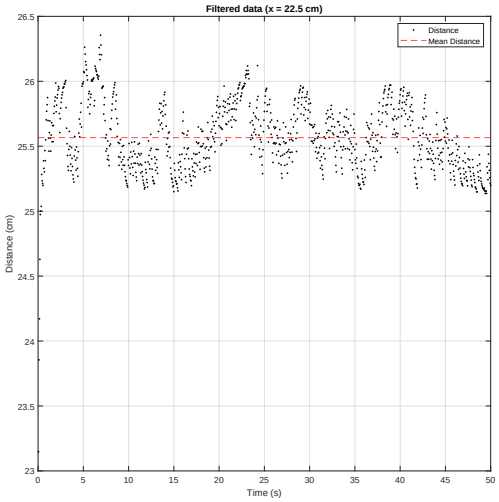
	Unfiltered	$\hat{x}_{k0}^{-} = 25.4\,\text{cm}$	$\hat{x}_{k0}^{-} = 28\,\text{cm}$	$\hat{x}_{k0}^{-} = 35\,\text{cm}$
μ	$25.5746\,\text{cm}$	$25.5753\,\text{cm}$	$25.5833\,\text{cm}$	$25.6050\,\text{cm}$
R	$0.2047\,\text{cm}^2$	$0.0589\,\text{cm}^2$	$0.0662\,\text{cm}^2$	$0.1746\,\text{cm}^2$
σ	$0.4524\,\text{cm}$	$0.2427\,\text{cm}$	$0.2572\,\text{cm}$	$0.4178\,\text{cm}$

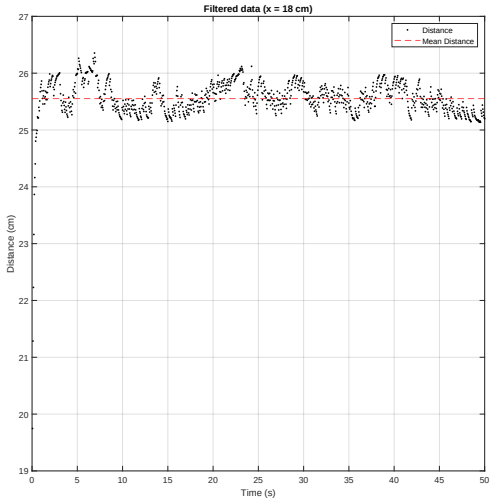




Unsurprisingly, a positive or negative error in the initial guess will both reduce the performance of the filter:

	Unfiltered	$\hat{x}_{k0}^{-} = 25.4\,\text{cm}$	$\hat{x}_{k0}^{-} = 22.5\,\text{cm}$	$\hat{x}_{k0}^{-} = 15\,\text{cm}$
μ	$25.5746\,\text{cm}$	$25.5753\,\text{cm}$	$25.5663\,\text{cm}$	$25.5748\,\text{cm}$
R	$0.2047\,\text{cm}^2$	$0.0589\,\text{cm}^2$	$0.0718\,\text{cm}^2$	$0.1360\,\text{cm}^2$
σ	$0.4524\,\text{cm}$	$0.2427\,\text{cm}$	$0.2680\,\text{cm}$	$0.3687\,\text{cm}$

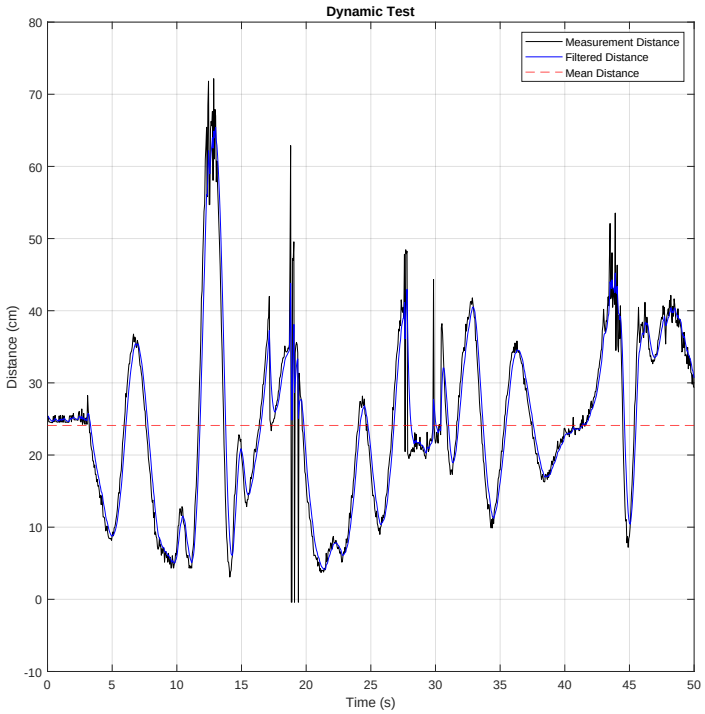




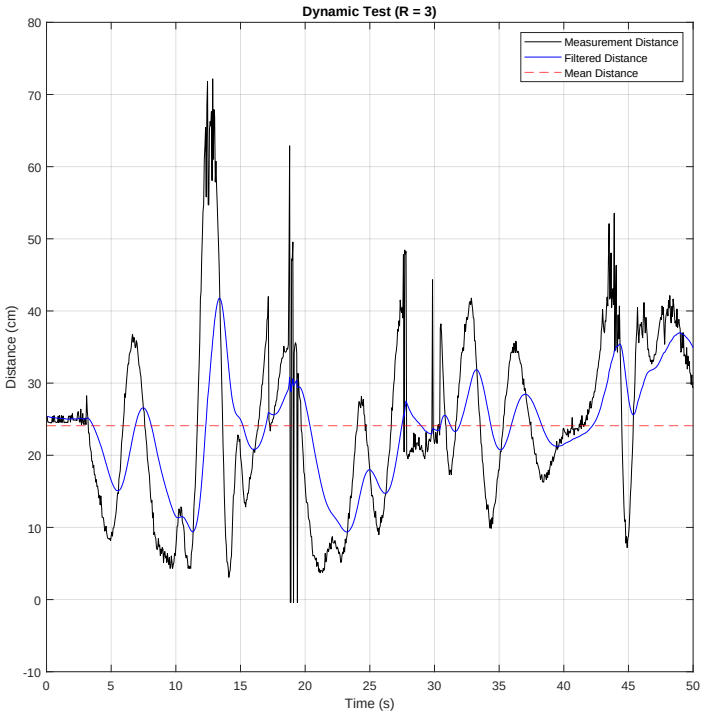
From these results, it appears to be the case that increasing the variance and decreasing the error will give a much more performant filter with a reduced standard deviation. This is indeed true for a static ultrasonic sensor at a fixed distance. However, changing the parameters in this way also has the cost of increasing the reaction time of the filter, so that even though the standard deviation may be lower for the static measurement, the model lags behind the dynamic measurement significantly.

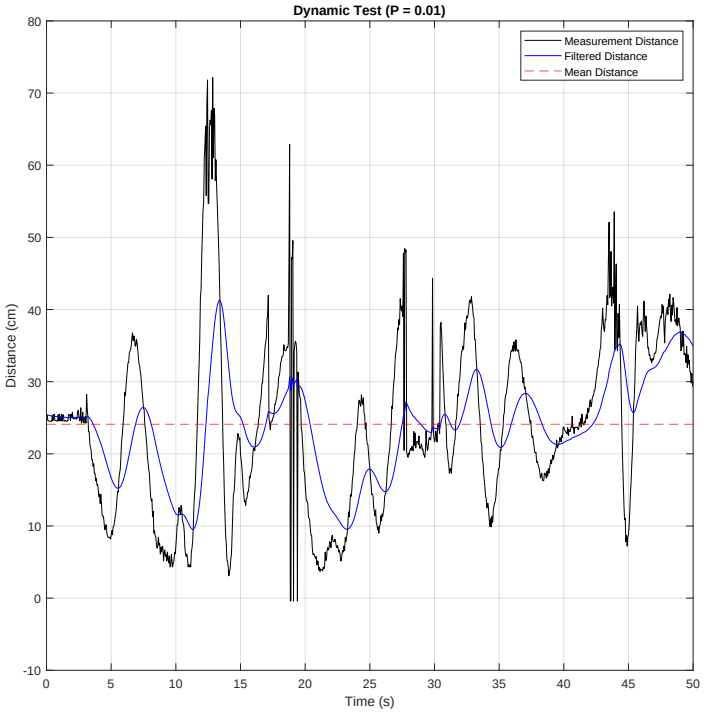
To illustrate this, another set of 1000 measurements was taken every 50 ms for a total duration of 50 s, but this time the sensor was moved around the room, sensing the objects at various distances.

The original Kalman filter performs well at keeping up with the measurement data:



Neither the initial guess nor the model noise power influence the response rate of the filter. However, increasing the variance and decreasing the error both introduce a large time delay, as shown on the following page.


Dynamic Test (R = 3)
Measurement Distance
Filtered Distance
Mean Distance
Distance (cm)
Time (s)


Dynamic Test (P = 0.01)
Measurement Distance
Filtered Distance
Mean Distance
Distance (cm)
Time (s)

Appendix

```
#include <SPI.h>
#include <SD.h>
#include <NewPing.h>

#define TRIGGER_PIN 3
#define ECHO_PIN 4
#define MAX_DISTANCE 200

const int chipSelect = 10;
const int maxIter = 1000;
int n = 0;

const int H_k = 1;
double K_k;
double x_k;
double P_k;
const int st_k = 1;
const double R = 0.3023;
const double calibration = -0.4;

double x_k0 = 25.4;
double P_k0 = 0.1;

int dt = 50;
double Q_k = 1 * dt / 1000;

int trigPin = 3;
int echoPin = 4;

File dataFile;

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
  Serial.begin(9600);
  while (!Serial) {;}

  SD.begin(chipSelect);
  dataFile = SD.open("kalman_run_1.txt", FILE_WRITE);
}
```

```
void loop() {
  if (n < maxIter)
  {
    delay(dt);
    double distance = sonar.ping() / (2 * 29.1) + calibration;

    K_k = P_k0 * H_k / (H_k * P_k0 * H_k + R);
    x_k = x_k0 + K_k * (distance - H_k * x_k0);
    P_k = (1 - K_k * H_k) * P_k0;
    x_k0 = st_k * x_k;
    P_k0 = st_k * P_k0 * st_k + Q_k;

    dataFile.print("Measurement ");
    dataFile.print(n);
    dataFile.print(" at ");
    dataFile.print((double) n/20);
    dataFile.println(" s: ");
    dataFile.print("Unfiltered: ");
    dataFile.print(distance, 4);
    dataFile.println(" cm");
    dataFile.print("Filtered: ");
    dataFile.print(x_k, 4);
    dataFile.println(" cm");

    n++;
  }

  dataFile.close();
}
```