Number: 2012-1
FEN VE TEKNOLOJİ BİLGİ PAYLAŞIMI
SCIENCE AND TECHNOLOGY INFORMATION SHARING
Article Web Page: www.ibrahimcayiroglu.com

Kalman Filtresi ve Programlama

Kalman Filter and Programming

İbrahim ÇAYIROĞLU

Karabük Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği, 78050, Karabük, icayiroglu@yahoo.com

Anahtar Kelimeler:
Kalman Filtresi

Özet: Bu makalede Kalman Filtresinin temelden başlayarak tüm aşamalarının uygumalı olarak anlatılması hedeflenmiştir. Kalman filtresi 20 yüzyılın en önemli buluşlarından biridir. Bu filtre, gürültülü ölçüm yapılan bir sistemi tahmin etmek için oldukça güçlü ve yetenekli bir algoritmadır. Konular örnekli olarak anlatılmaya çalışılmıştır. Verilen örnekler direk uygulanıp denenebilir. Kavramların daha iyi anlaşılması için hem Türkçe alternatif anlatımları hem de İngilizce karşılıkları verilmiştir.

Keywords:
Kalman Filter

Abstract: In this paper aims to describe all phases of the Kalman Filter from base to top as practically. Kalman filter is one of the most important discoveries of the 20th century. This filter is a very strong and talented algorithm to predict a system has noise measurements. The topics were described in the sample. The examples given are applied directly. Alternative explanations for better understanding of concepts in both Turkish and English equivalents are given.

1. Giriş

Modelin önceki bilgileriyle birlikte giriş ve çıkış bilgilerinden sistemin durumlarını tahmin edilebilen filtredir. Eğer sistemin stokastik veya rasgele gürültülü yönü hesaba katılırsa minimum varyans tahmini veya Kalman Filtresi çok uygun olmaktadır. Kalman Filtresi, geleneksel tahmin edicilerde olduğu gibi filtreleme özelliğine rağmen, sistemin ölçülemeyen durumlarını tahmin etmek için çok güçlü ve yeteneklidir.

Algoritma, gürültülü veriler üzerinde özyinelemeli gerçek zamanlı çalışarak hataları, enaz-kareler eğriye sığdırma yöntemi ile filitre eder ve sistemin fiziksel karakteristiklerinin modellenmesi ile üretilen gelecek durumun matematiksel tahminine göre optimize eder.

Model tahmini, gözlem ile karşılaştırılır. Elde edilen fark, Kalman kazancı {Kalman gain} olarak bilinen bir çarpan ile ölçeklendirilir. Daha sonra sıradaki tahminleri iyileştirmek için modele bir girdi olarak geri besleme uygulanır {feedback}. Kazanç performansı iyileştirmek için ayarlanabilir yapılır. Yüksek bir kazanç ile, filtre gözlemleri daha yakın olarak takip edilir. Düşük bir kazanç ile, filtre model tahminlerini daha yakın olarak takip edilir. Yöntem, gerçek bilinmeyen değerlere, model tahminlerine dayanarak elde edilebilecek tahminlerden daha yakın tahminler üretmeye çalışır.

Her bir zaman adımında, Kalman Filitresi, gerçek bilinmeyen değerlerin tahminlerini belirsizlikleriyle {uncertainty} beraber üretir. Sıradaki ölçümün sonucu gözlendiğinde, bu tahminler, belirsizliği düşük tahminlere daha fazla ağırlık vererek, ağırlıklı ortalama ile güncellenir.

Kalman filtresi sensör füzyonu ve veri füzyonu için kullanılır. Tipik olarak, gerçek zamanlı sistemler bir sistemin durumunu elde etmek için tek bir ölçüm yapmak yerine bir çok ardışık ölçüm üretir. Bu bir çok ölçüm daha sonra o zaman anında sistemin durumunu üretmek için matematiksel olarak birleştirilir.

Örnek bir uygulama olarak, bir aracın yerini hassas olarak belirleme problemini düşünelim. Aracın pozisyonu bir kaç metre hata ile belirleyebilen bir GPS aleti ile ölçelim. GPS tahminleri gürültülüdür; okumalar, her zaman gerçek pozisyonun birkaç metre yakınında olmasına rağmen, hızlıca etrafta zıplayabilir. Aracın pozisyonu, direksiyon dönüşleri ve direksiyonun açısını izleyerek, hızı ve yönü zamana göre entegre ederek de tahmin edilebilir. Bu teknik parakete hesabı olarak bilinir. Tipik olarak, parakete hesabı aracın yeri hakkında çok

yumuşak bir tahmin sağlayacaktır, ancak küçük hatalar biriktikçe sapacaktır. Ayrıca, aracın fizik kurallarını takip etmesi de beklenir, yani pozisyonunun hızıyla orantılı olarak değişmesi beklenir.

Bu örnekte, Kalman filitresinin iki ayrı fazda çalıştığı düşünülebilir: tahmin et ve güncelle. Tahmin etme fazında, aracın yeri Newton'un hareket kurallarına göre, gaz pedalının durumuna göre ve direksiyonun açısına göre hesaplanacaktır. Sadece bir pozisyon tahmini hesaplanmayacak, ancak yeni bir kovaryans da (ortak sapma miktarı) hesaplanacaktır. Kovaryans aracın hızı ile orantılıdır; Yüksek hızlarda parakete hesabının hassaslığı daha düşük, düşük hızlarda daha yüksektir. Daha sonra, güncelleme fazında, aracın pozisyonu GPS biriminden alınır. Bu ölçümle beraber bir miktar belirsizlik daha gelir ve bunun kovaryansının (ortak sapma miktarı) önceki fazdan gelen tahminin kovaryansına oranı, yeni ölçümün güncellenen tahminini ne kadar etkileyeceğini belirler. Gerçek pozisyondan uzaklaşıldıkça, bu hesaplama yöntemi, tahminleri gerçek pozisyona doğru hızlıca ve gürültülü olmayacak şekilde çeker.

Kalman filtresi pek çok farklı alanda sistemin durumunu, değerlerini kestirebilen (tahmin edebilen) bir yöntemdir. Matematiksel olarak doğrusal sistemlerin (lineer sistemlerin-hareket denklemleri birinci derece olan denklemler) durumlarını tahmin eder. Pratikte çok faydalı bir filtredir. Ayrıca teorik yönüde güçlüdür. Çünkü mevcut filtreler içinde kestirim hatasını (tahmini) minimize eden (gittikçe azaltan) tek filtredir.

Kalman filtresi doğrusal sistemler için optimal bir kestiricidir (tahmin edicidir). Ancak gerçek dünyada çok az sayıda doğrusal sistem vardır. Bu problemin üstesinden gelmenin yaygın bir yaklaşımı, Kalman filtresini kullanmadan önce sistemi doğrusal hale getirmektir ve bu yaklaşım "Genişletilmiş Kalman Filtresi" olarak bilinir. Bununla birlikte, doğrusallaştırma kararsız kestirimler (kontrol edilemeyen tahminler) gibi bazı problemlerin çıkmasına da yol açabilir.

1.1. Bazı Önemli Noktalar

a) Kalman filtresi her ne kadar filtre olarak geçsede bir filtre değil daha çok bir tahmin edicidir {estimator}.

b) 20 yüzyılda yapılan en önemli buluşlardan biri olduğu söylenebilir.

c) Ona tamamen hakim olmak zor olsa da bir miktar matematik alt yapı ile onunla oynanabilir.

d) Bilgisayar algoritmalarına uygulamaya çok elverişlidir.

e) Tekrarlı {recursive} bir metottur. Yani bir önceki adımın çıktısı bir sonraki adımda girdi olarak kullanılabilir.

f) Denklemleri çok karmaşık {complicated} ve matrisleri çok gizemli {mysterious} olsa da, çoğu zaman onları, karmaşık gerçek bilimsel faaliyetler yapmıyorsanız ihmal edebilir {omit} yada görmezden gelebilirsiniz {ignore}.

1.2. Problemin Tanımı

Her hangi bir sinyal düşünün. Ses sinyali olabilir, radar sinyali olabilir hatta sayısallaştırılmış {digitized} resim olabilir. Ve tabiki ortamda bir miktar gürültü {noise} olacaktır. Bu gürültüden nasıl kurtulabiliriz. Elimizdeki sinyali içinde gürültü olmadan nasıl kullanabiliriz. Aklımıza ilk gelen örneklerin ortalamasını almaktır. Bu basit yaklaşım çoğu gerçek problemlerde çalışmayacaktır. Bizim daha gelişmiş {sophisticated} bir yaklaşıma ihtiyacımız vardır.

Dijital sinyal işleme bilginleri {scholars} yıllardır bu problemle ilgileniyorlar ve çok sayıda teknik geliştirilmiştir. Kalman filtresi bunların en güçlü olanlarından biridir.

2. Kalman Filtresi Formülleri

Daha önceden de bahsedildiği gibi Kalman filtresinin tanımları ve denklemlerinin tüm yönleriyle başlangıçta anlaşılması zordur. Çoğu durumda durum matrisleri {state matrices} atılırsa aşağıdaki denklem elde edilir. Bununla başlamak daha kolaydır.

$$\boxed{\hat{X}_k = K_k.Z_k + (1 - K_k).\hat{X}_{k-1}}$$

Burada alt indis olarak kullanılan k harfi durumları/evreleri {states} gösterir. Biz onu ayrık/kesikli zaman aralıkları {discrete time intervals} olarak varsayabiliriz. Şöyle ki, k=1 olduğunda 1 ms (mili saniye), k=2 olduğunda ise 2 ms olarak kabul edilebilir.

Bizim amacımız x sinyalinin tahmini olan $\widehat{\boldsymbol{X}}_{\boldsymbol{k}}$ değerini bulmaktır. Biz her bir k için bu değeri bulmaya çalışırız. Aynı zamanda burada $\boldsymbol{Z}_{\boldsymbol{k}}$ ölçülen değerdir {measured value}. Şunu aklımızda bulunmalıyız. Biz bu ölçülen değerin doğruluğundan kesin olarak emin değiliz. Diğer türlü bütün bunları yapmamıza zaten gerek kalmazdı. Burada $\boldsymbol{K}_{\boldsymbol{k}}$ Kalman Kazancı ki en önemli parametredir {kalman gain} ve $\widehat{\boldsymbol{X}}_{\boldsymbol{k-1}}$ değeri ise sinyalin önceki durumdaki {previous state} tahminidir.

Bu denklemdeki bilinmeyen tek eleman Kalman kazancıdır (K_k). Çünkü biz zaten ölçülen değere (Z_k) ve öndeki tahmin edilen sinyale ($\hat{X}_{k-1}$) sahibiz, bunları biliyoruz. Her bir durum için buradaki Kalman kazancını hesaplamalıyız. Bu doğal olarak kolay bir iş değildir. Bunu bulmak için tüm araçlara sahibiz.

Örnek olarak Kalman kazancını (K_k) 0,5 gibi bir değer alalım. O zaman förmülümüz basit bir

ortalamayı bulan fonksiyona dönüşecektir. Oysa kalman kazancı bundan daha ileri bir yeteneğe sahiptir. Her bir adımda bu parametrenin değerini hesaplamalıyız. Böylece kalman filtresinin esas yeteneği ortaya çıkmış olsun. Bu filtre en optimum ortalama faktörünü her bir adım için bulur ve bir yere kadar geçmiş durumlarıda hatırlar.

2.1. Adım Adım Formüllerin Anlatım

Kalman filtreye hızlı bir başlangıç için aşağıda basit bir anlatım verilmiştir.

2.1.1. Adım 1. Modeli Oluştur

Tüm adımlardan en önemlisi Kalman filtrenin şartlarını {Kalman filtrering conditions} probleminize uydurmaktır.

Kalman filtrenin iki denklemi aşağıdaki şekildedir.

$x_k = Ax_{k-1} + Bu_k + w_{k-1}$

$Z_k = Hx_k + v_k$

Sinyalimizin her bir x_k değeri birinci denklem kullanılarak bulunur. (doğrusal-rastsal denklem {linear stochastik equation}). Herhangi bir x_k değeri, önceki değerinin (x_{k-1}) üzerine kontrol sinyali (u_k) ve önceki işlem gürültüsü (w_{k-1}) eklenerek oluşturulan lineer bir kombinasyonla bulunur. Çoğu zaman kontrol sinyali u_k pek kullanılmaz.

İkinci denklem bize şunu söyler. Herhangi bir ölçüm değeri (z_k) (ki biz onun doğruluğundan emin değiliz) sinyalin değeri (x_k) ile ölçümün gürültüsünün (v_k) lineer bir kombinasyonundan oluşur (doğrusal olarak birleşiminden yani birinci derece bir denklem olarak, kare yada küp kök bir denklem halinde değil). Her ikisi de (w_{k-1},v_k) Gaussian olarak ele alınır (Gaussian ileride açıklanacak, çan eğrisi şeklinde bir grafiği vardır).

İşlem/süreç gürültüsü {process noise}(w_{k-1}) ve ölçüm gürültüsü {measurement noise}(v_k) istatistiksel olarak birbirinden bağımsızdır.

A, B ve H elemanları matrislerin genel gösterimidir. Çoğu sinyal problemlerimizde bunlar sadece nümerik bir değerdir (bir sayıdır). Ayrıca ek bir kolaylık olsun diye, çoğu zaman, her bir aşamada bu değerler değişebilmesine rağmen, bu değerleri sabit varsayabiliriz.

Eğer biz sistemimizden oldukça eminsek, (çoğu zaman bu şekilde yapılır), geriye sadece gürültü fonksiyonlarının (w_{k-1},v_k) aritmetik ortalamasını {mean} ve stantart sapmasını {standart deviation} tahmin etmek kalır. Şunu biliyoruz ki gerçek dünyada hiç bir sinyal Gaussian değildir. Fakat biz onu bir miktar yaklaşık alma ile {approximation} böyle kabul edebiliriz. Bu büyük bir problem değildir. Çünkü biz Kalman Filtresinin doğru tahminlere doğru yakınsamaya {converge} çalıştığını göreceğiz. Hatta Gaussian gürültü parametreleri kötü bir şekilde tahmin edilmiş olsa bile bu mümküdür.

Şunu unutmamalıyız ki; Gürültü parametreleri için yapmış olduğumuz iyi tahminler, daha iyi tahmin sonuçları elde etmemizi sağlayacaktır.

2.1.2. Adım 2. İşleme Başla

Eğer modelinizi Kalman Filtresine uydurmayı başardınız ise, bundan sonraki adım gerekli parametreleri ve başlangıç değerlerini belirlemektir.

İki ayrı denklem takımız vardır. Zaman güncelleme/Tahmin {Time Update/Prediction}ve Ölçümü Güncelleme/Düzeltme {Measurement Update/Correction}. Her iki denklem takımı da k. durumda (her bir zaman diliminde) işleme konulur.

Tahmin / Zamanı güncelleme {Prediction / Time Update }	**Düzeltme** / Ölçümü Güncelleme {Correction / Measurement Update}
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$ $P_k^- = AP_{k-1}A^T + Q$	$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$ $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$ $P_k = (1 - K_k H)P_k^-$

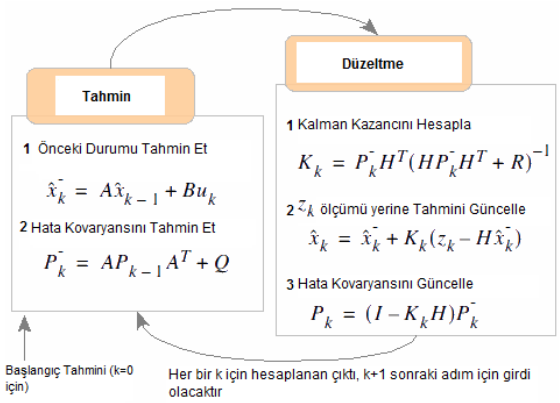
Biz modellemeyi Adım 1'de yaptık. Bu yüzden A, B, ve H matrislerini oradan biliyoruz. Çoğunlukla bu matrisler sabit sayı olabilir. Hatta belki de çoğunlukla 1 gibi sabit bir değer alabilir. Bu denklemleri (adım 1 deki denklemleri) karşınıza alıp bunları nasıl sadeleştirebileceğinizi düşünmenizi tavsiye ederiz.

Adım 2'deki denklemlerde ise en zor belirlenecek katsayılar R ve Q dur. R'yi bulmak daha kolaydır. Çünkü biz genellikle etraftaki gürültüden çoğunlakla eminizdir. Fakat Q değerini ise ortaya çıkarmak o kadar kolay değildir. Ve bu aşamada özel bir metod veremiyoruz.

2.1.3. Adım 3: İterasyon (Döngüsel Hesaplamalar)

İhtiyacımız olan tüm bilgileri topladıktan sonra işleme başlayabiliriz. Şunu unutmayın, önceki tahminler mevcut durum için girdi olacaktır. Yani her önceki tahmin bir sonraki hesaplamanın girdisi olacaktır.

Burada $\hat{x}_k^-$ önceki tahmindir. Yani ölçümü düzeltme güncellemesi yapılmadan önceki ham tahmindir. Aynı zamanda P_k^- da önceki hata Kovaryansıdır (hatanın ortak değişme miktarı). Bu iki değeri bir sonraki aşama olan Ölçümleri Güncelleme/Düzeltme aşamasında "önceki" {prior}değerler olarak kullanıyoruz. Yani bir sonraki aşamaya girdi olarak veriyoruz.



Ölçüm güncelleme aşamasındaki denklemlerde biz gerçek $\hat{x}_k$ değerini buluyoruz. Bu değer x'in k anındaki değeridir. Yani esas bulmak istediğimiz değerdir. Aynı zamanda biz P_k değerini de buluyoruz. Bu iki değeri bir sonraki gelecek (k+1) aşaması için hesaplıyoruz. K_k Kalman kazancı bir sonraki aşama için gerekli olduğundan hesaplamıyoruz. Bu değer gizli, gizemli olmakla birlikte, bu denklemlerin en önemli parçadır.

Bu ikinci aşamadaki (ölçüm güncelleme aşaması) değerler "sonraki" {posterior}değerlerdir.

3. Basit Bir Örnek

Şimdi bir kaynaktan elde edilen voltaj değerlerini okuma üzerine bir çalışma yapalım. Kaynağın sabit bir Voltaj (V) değerine sahip olduğunu farz edeceğiz. Fakat tabiki berlli bir volt aşağı ve yukarı gürültülü okuma olacaktır. Buna ölçüm gürültüsünün standart sapmasını {standart deviation} 0.1 V olarak kabul edelim.

Şimdi modelimizi oluşturalım. Daha öncede söylediğimiz gibi, denklemleri çok basit formlara düşürelim.

$x_k = Ax_{k-1} + Bu_k + w_k \Rightarrow x_k = x_{k-1} + w_k$

$z_k = Hx_k + v_k \Rightarrow z_k = x_k + v_k$

Her şeyden önce, biz 1 boyutlu bir sinyal problemine sahibiz. Böylece modelimizdeki her bir eleman sayısal değere sahiptir. Yani matris formatında değildir.

Biz hiç bir kontrol sinyaline (u_k) sahip değiliz. Böylece bu değer 0 olur.

Sinyalimiz sabit bir değere sahip olduğundan A katsayısı 1 değerini alır. Çünkü biz bir sonraki gelen değeri biliyoruz. Buda öncekiler gibi aynı değeri alacaktır. Buradaki örnek basittir, fakat başka linear örneklerde de bu A değerini işlemleri basitleştirmek için 1 olarak alabiliriz.

H değeri 1 eşittir. Çünkü biz biliyoruz ki ölçüm, durum değeri (ölçülen esas değer){state value} ile bir miktar gürültünün birleşiminden ibarettir. Gerçek dünyada H değerinin 1 den farklı olduğu durumlarla nadiren karşılaşılır.

Sonuç olarak aşağıdaki ölçüm değerlerine sahip olduğumuzu varsayalım.

Zaman (Time) *(ms)*	1	2	3	4	5	6	7	8	9	10
Değer (Value) *(V)*	0.39	0.50	0.48	0.29	0.25	0.32	0.34	0.48	0.41	0.45

k=0 zamanından başlayalım. Bazı başlangıç durumlarını ya bulmalıyız yada tahmin etmeliyiz. Burada biz bazı başlangıç değerlerini atıyoruz. X_0=0 (başlangıçtaki ölçüm değeri 0) ve P_0=1 (Başlangıçtaki hata kovaryansı) değerlerini varsayalım. Burada P_0=0 değerini niye almadık. Cevabı basittir. Eğer biz bu şekilde seçseydik, bu durum ortmada hiç bir gürültünün olmadığı anlamına gelirdi. Bu varsayım sonradan gelecek tüm $\hat{x}_k$ değerlerinin (tahmin değerlerinin) sıfır olmasına yol açacaktı. Bu nedenle P_0 değerini sıfırdan farklı bir değer seçiyoruz.

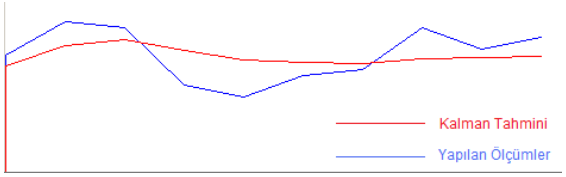
Bu örnek için Kalman Filtresi Algoritmasında kullanacağımız Zaman Güncelleme {Time Update, prediction} ve Ölçüm Güncelleme {Measurement Update, Correction} denklemlerini yazalım.

Zaman Güncelleme {Time Update, prediction}	Ölçüm Güncelleme {Measurement Update, Correction}
$\hat{x}_k^- = \hat{x}_{k-1}$ $P_k^- = P_{k-1}$	$K_k = \frac{P_k^-}{P_k^- + R}$ $\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{x}_k^-)$ $P_k = (1 - K_k)P_k^-$

Şimdi her bir iterasyon (zaman dilimi) için $\hat{x}_k$ durum tahmin değerlerini hesaplayalım.

Burada ilk iki iterasyon detaylı bir şekilde verilmiştir. Diğerleri de aynı şekilde hesaplanır. Algoritmanın bilgisayar programı yazıldığında, Kalman Filtrenin uygulamasının çok daha kolay olduğu görülecektir.

Grafik incelendiğinde (Şekil 1) Kalman Filtrenin gerçek voltaj değerlerine nasıl yakınsadığı {Converge} daha iyi görülecektir. Burada 10 iterasyon gösterilmiştir. Eğer 50 yada daha fazla iterasyonlar yapılmış olsa çok daha iyi değerler elde edilebilir.



Şekil 1. Kalman Filtresi hesabı

k	1	2	3	4	5	6	7	8	9	10
z_k (Ölçümler)	0,390	0,500	0,480	0,290	0,250	0,320	0,340	0,480	0,410	0,450
$\hat{x}_{k-1}$	0	0,355	0,424	0,442	0,405	0,375	0,365	0,362	0,377	0,380
P_k^-	1	0,091	0,048	0,032	0,024	0,020	0,016	0,014	0,012	0,011
Zaman Güncelleme {Time Update}	$\hat{x}_k^- = \hat{x}_{k-1}$ =0 $P_k^- = P_{k-1}$ =1	$\hat{x}_k^-$ = 0,355 P_k^- = 0,091	~	~	~	~	~	~	~	~
Ölçüm Güncelleme {Measurment Update}	$K_k = \frac{P_k^-}{P_k^- + R}$ =1/(1+0,1) =0,909 $\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{x}_k^-)$ =0+0,909(0,390-0) = 0,355 $P_k = (1-K_k)P_k^-$ =(1-0,909).1 =0,091	K_k=0,091/(0,091+0,1) =0,476 $\hat{x}_k$=0,355+0,476(0,500-0,355) =0,424 P_k=(1-0,476)0,091 =0,048	~	~	~	~	~	~	~	~
$\hat{x}_k$ (Tahminler)	**0,355**	**0,424**	**0,442**	**0,405**	**0,375**	**0,365**	**0,362**	**0,377**	**0,380**	**0,387**
P_k	0,091	0,048	0,032	0,024	0,020	0,016	0,014	0,012	0,011	0,010

Bir kaç adımda yakınsamayı {convergence} daha iyi elde edebilmek için şunları yapınız.

- Sistemi daha zarif bir şekilde {elegantly} modelleyin.

- Gürültüyü daha hassas {precisely} olarak hesaplayın.

4. Kaynaklar

1. Greg Welch, Gary Bishop, "An Introduction to the Kalman Filter", *University of North Carolina at Chapel Hill Department of Computer Science*, 2001.

2. M.S. Grewal, A.P. Andrews, "Kalman Filtering - Theory and Practice Using MATLAB", *Wiley*, 2001.

Ek-1. Kalman Filtresi Algoritma Kodları

```
//Kalman Filtresi global tanımlar
double X_k_KalmanTahminEski = 0; //Prior Estimate
double Pk_HataKovaryansiEski = 1; //Error
Covariance
//************************************
//Kalman Hesabı için tıklanan butonun kodları
private void btnKalmanFiltresi_Click(object
sender, EventArgs e)
{
//Tanımlamalar
double[] Zk_OlcumDegerleri = new
double[10]{0.39,0.5,0.48,0.29,0.25,0.32,0.34,0.48
,0.41,0.45};
double[] Xk_HesaplananDegerler = new double[10];
double R_HataMiktari = 0.1;

//Ölçüm değerlerini tek tek Kalman Filtresine
gönderiyor.
for (int i = 0; i < 10; i++)
{
  Xk_HesaplananDegerler[i] =
  KalmanFiltresiHesapla(Zk_OlcumDegerleri[i],
  R_HataMiktari);
}
//Grafiği Çiziyor
double X_OlculenOncekiDeger = 0;
double Y_OlculenOncekiDeger = 0;
double X_HesaplananOncekiDeger = 0;
double Y_HesaplananOncekiDeger = 0;
for (int j = 0; j < 10; j++)
{
  //Ölçülen değerleri gösteriyor.
  GrafikAlani.DrawLine(KalemRengiMavi,(int)
  X_OlculenOncekiDeger,(int)
  Y_OlculenOncekiDeger, (j * 50),
  (int)(Zk_OlcumDegerleri[j] * 250));
  //Hesaplanan değerleri gösteriyor.
  GrafikAlani.DrawLine(KalemRengiKirmizi, (int)
  X_HesaplananOncekiDeger,
  (int)Y_HesaplananOncekiDeger, (j * 50),
  (int)(Xk_HesaplananDegerler[j] * 250));
  X_OlculenOncekiDeger = (j * 50);
  Y_OlculenOncekiDeger = (Zk_OlcumDegerleri[j] *
  250);
  X_HesaplananOncekiDeger = (j * 50);
  Y_HesaplananOncekiDeger =
  (Xk_HesaplananDegerler[j] * 250);
}
}
//************************************
public double KalmanFiltresiHesapla(double
Zk_OlculenDeger, double R_HataMiktari)
{
//Güncelleme--Eski değerleri yeni değerler içine
atıyor.
double X_k_KalmanTahminYeni =
X_k_KalmanTahminEski;
double Pk_HataKovaryansiYeni =
Pk_HataKovaryansiEski;

//Ölçümleri Düzeltme--
double Kk_KalmanKazanci = Pk_HataKovaryansiYeni /
(Pk_HataKovaryansiYeni + R_HataMiktari);
double Xk_KalmanHesaplanan = X_k_KalmanTahminYeni
+ Kk_KalmanKazanci * (Zk_OlculenDeger -
X_k_KalmanTahminYeni);

Pk_HataKovaryansiYeni = (1 - Kk_KalmanKazanci) *
Pk_HataKovaryansiEski;

//Eski Değerleri Atama--bu değişkenler
Global tanımlandı. bu procedure her
geldiğinde bunları kaybetmemelidir.
Pk_HataKovaryansiEski =
Pk_HataKovaryansiYeni;
X_k_KalmanTahminEski = Xk_KalmanHesaplanan;
//bulunan sonuç bir sonraki adım için eski
tahmin olacak.
return Xk_KalmanHesaplanan;
}
```

Ek-2. Rudolf Emil Kalman

Rudolf Kalman Budapeşte / Macaristan da doğmuştur. 1953 yılında Lisans derecesini, 1954 MIT'de elektrik mühendisi olarak Master derecesini elde etmiştir. 1957'de Kolombiya Üniversitesinden doktorasını almıştır.

Kalman, elektrik mühendisliği üzerine eğitim almış ve geliştirdiği Kalman Filtresi ile ünlü olmuştur. Bu teknik kontrol sistemlerinde ve havacılık uygulamalarında genişçe kullanılan matematiksel bir yöntemdir. Yöntem sayesinde gürültülü ölçümlerden elde edilen sinyali filtreleyip, doğruya yakın sinyali çıkarmak mümkün olmuştur. 1960'lı yıllarda Kalman Filtresi Apollo Uzay programında kullanılmaya başlanmıştır.

_____ The Author _____

Ibrahim Cayiroglu *is an insructor in Mechatronic Engineering at Karabuk University, Turkey. He received his B.Sc. in Mechanical Engineering from Istanbul Technical University in 1991. He received his M.Sc. and Ph.D. in Computer Aided Design and Manufacturing from Kirikkale University, in 1996 and 2002, respectively. His research interests include CAD-CAM, Software and Mechatronic Systems.*