Modelli Probabilistici per le Decisioni, Antonio Vivace

Kalman Filter

A background study and an
interactive, real time simulation

Overview

1. Introduction to the Kalman Filter
    a. Assumptions
    b. Model
2. Applications
    a. Historical Relevance
3. Demo
    a. Software stack, design choices
    b. Implementation and quirks
    c. Analysis of the results
4. Conclusions

Kalman Filter: Model and Assumptions

- Thiele-Swerling developed optimal estimation of orbits of satellites and trajectories of missiles, anticipating the development of the Kalman filter.
- Recursive algorithm (*estimator*) for dynamic systems (*Markov* and *Stationary* process) *minimum mean-square error* estimator

  Filtering Task: $P(X_t|e_{1-t})$
  Prediction Task: $P(X_{t+k}|e_{1:t})$ per un $k > 0$

- Model matching the real system?

- Gaussian Noises Q, R, independent and known

- *Markov Assumption*:
  Previous state knowledge (ONLY the previous state!)
  Transition model: $P(X_t \mid X_{0:t-1}) = P(X_t \mid X_{t-1})$
  Sensor model: $P(E_t \mid X_{0:t-1}, E_{0:t-1}) = P(E_t \mid X_t)$

- Practical implementation of the Kalman Filter is often difficult due to the difficulty of getting a good estimate of the noise covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$

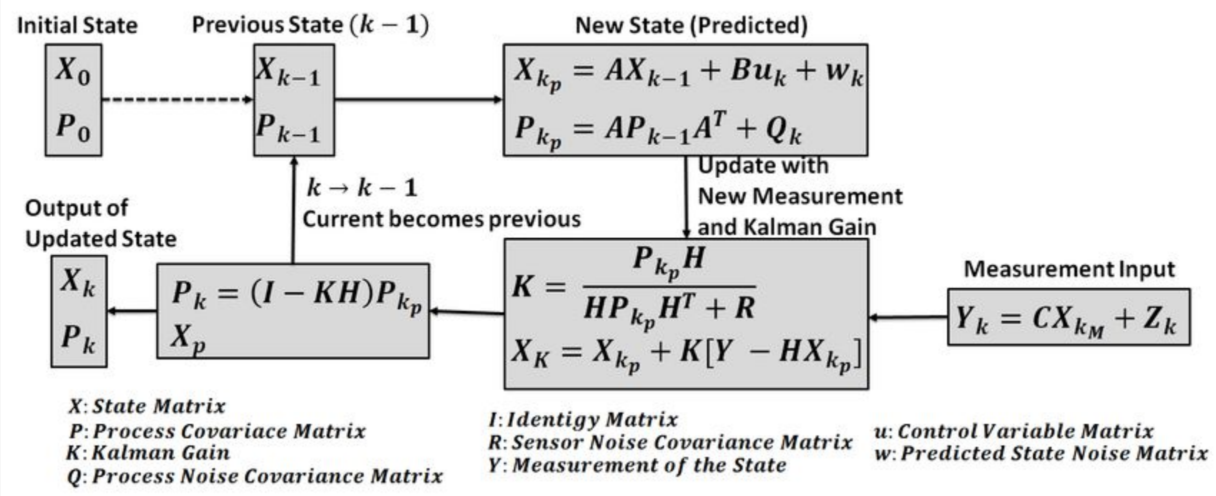
Introduction to the Kalman Filter

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1},$$

$$z_k = Hx_k + v_k.$$

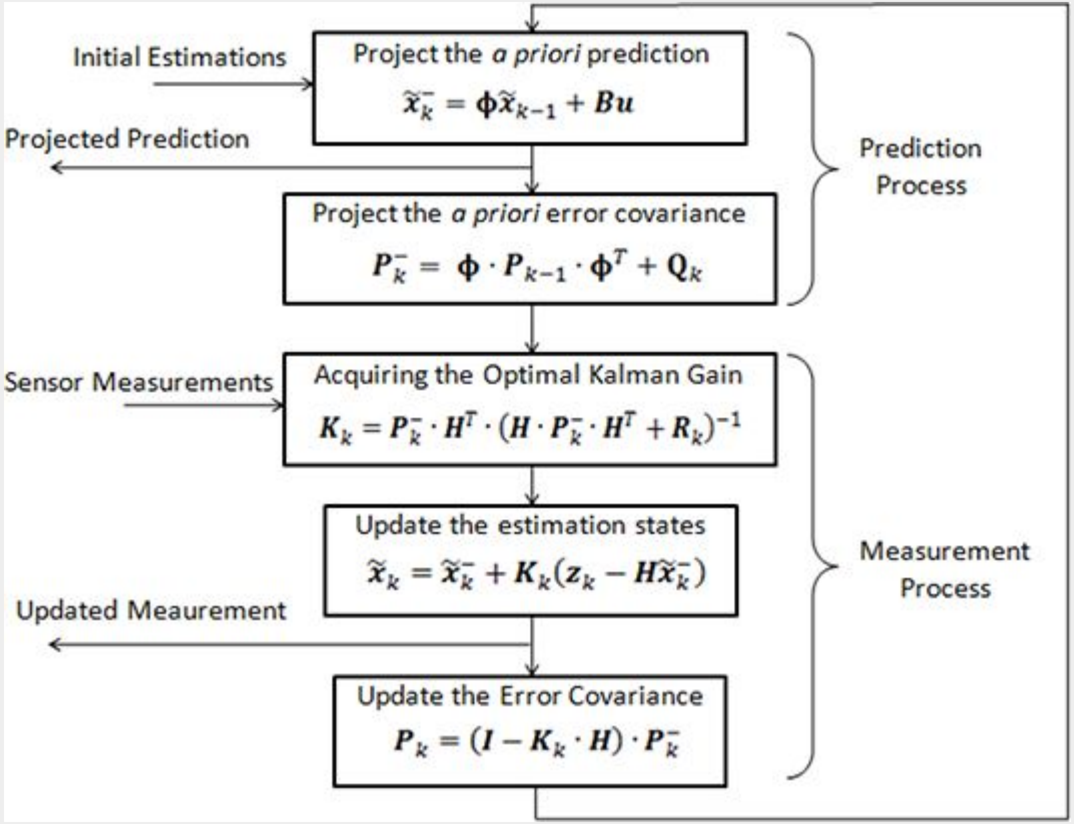
$$p(w) \sim N(0, Q),$$
$$p(v) \sim N(0, R).$$

Q: Process Noise, usually treated as a *tuning* parameter to adjust the gain of the Kalman filter to smooth either more or less the data
A (n x n): State Transition
B (n x l): Input Control
c: control vector, which would contain estimated changes from direct action commands
H (n x m): Measurement (stato/misurazione)
x in R^n, z in R^l, c in R^m.

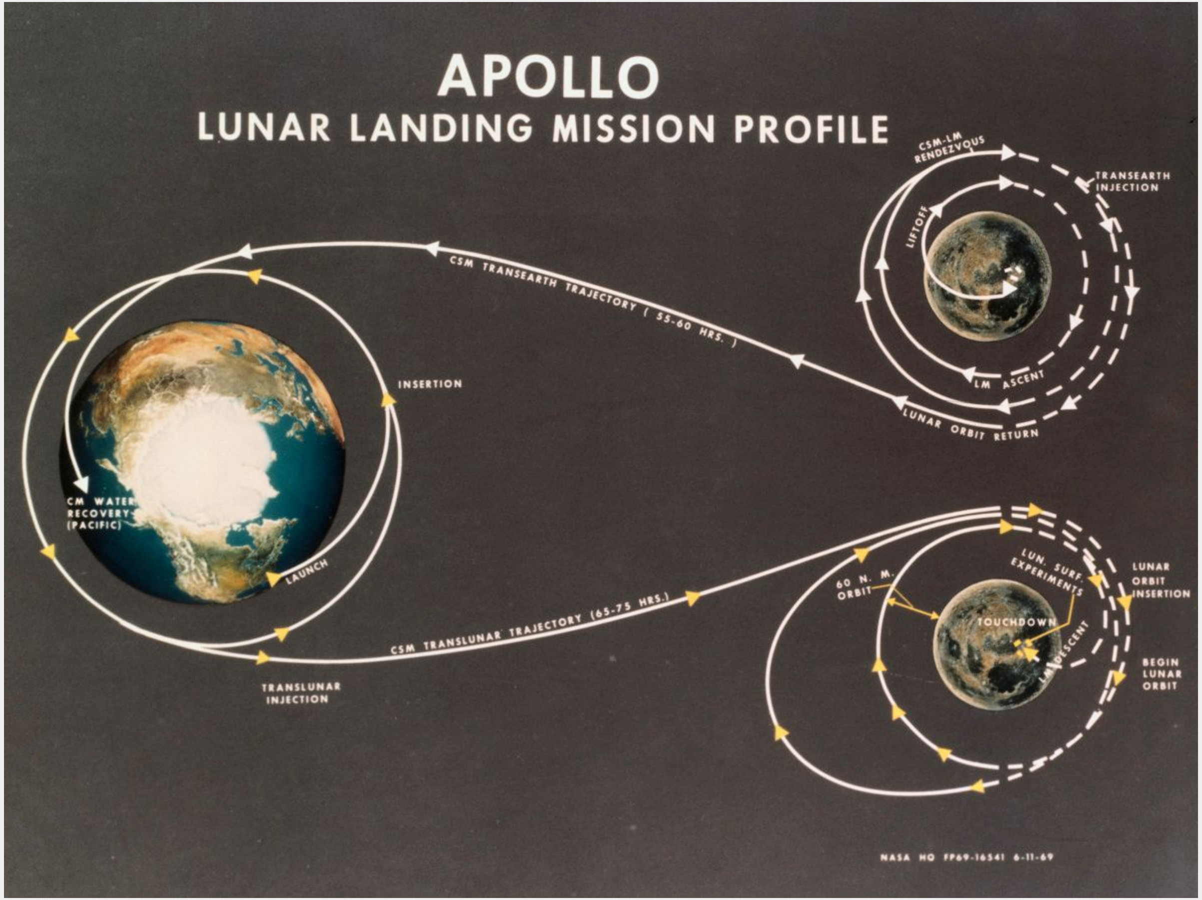


Introduction to the Kalman Filter

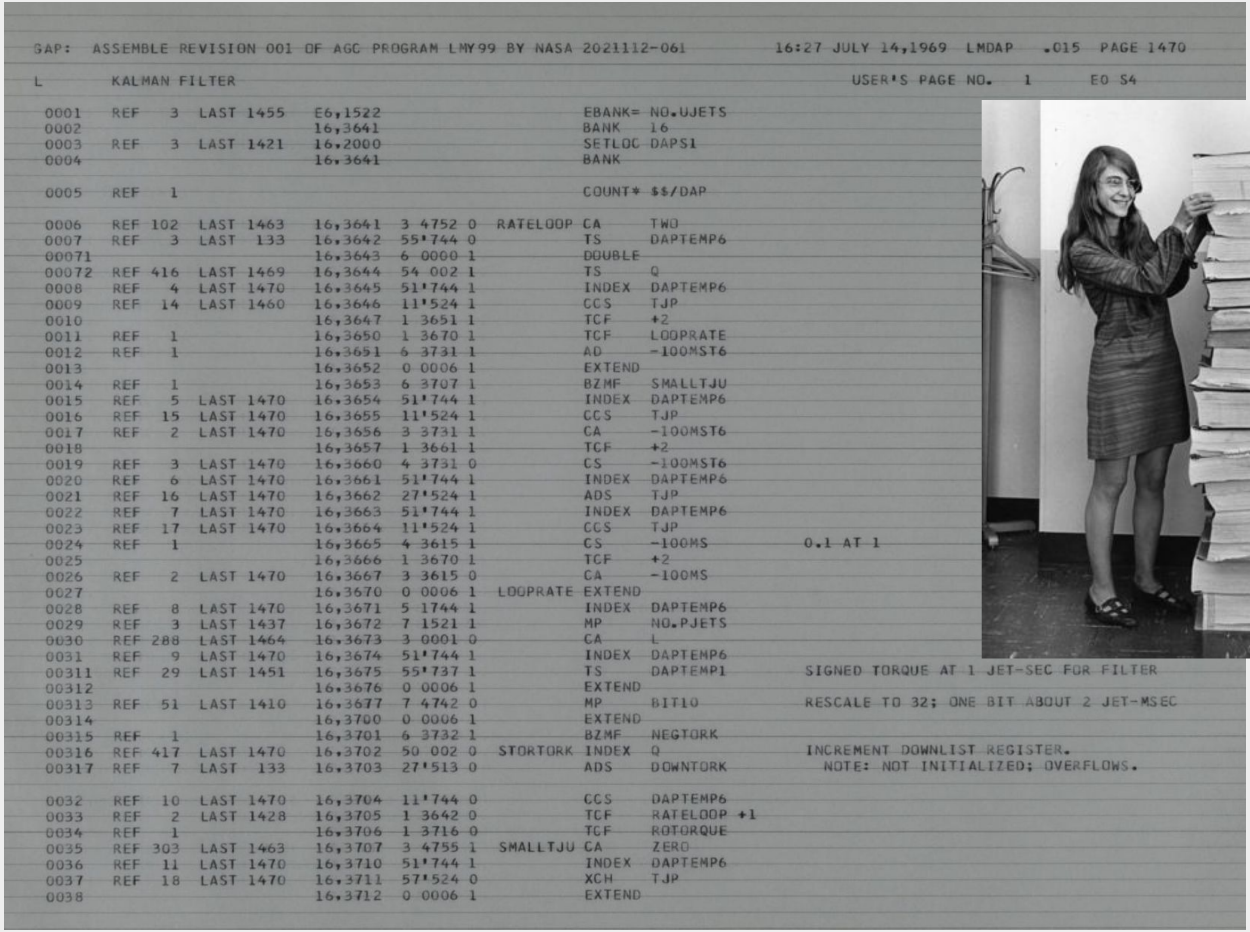
The Kalman gain is the **relative weight** given to the measurements and current state estimate, and can be "tuned" to achieve particular performance. With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely. At the extremes, a high gain close to one will result in a more jumpy estimated trajectory, while low gain close to zero will smooth out noise but decrease the responsiveness.



Historical Relevance

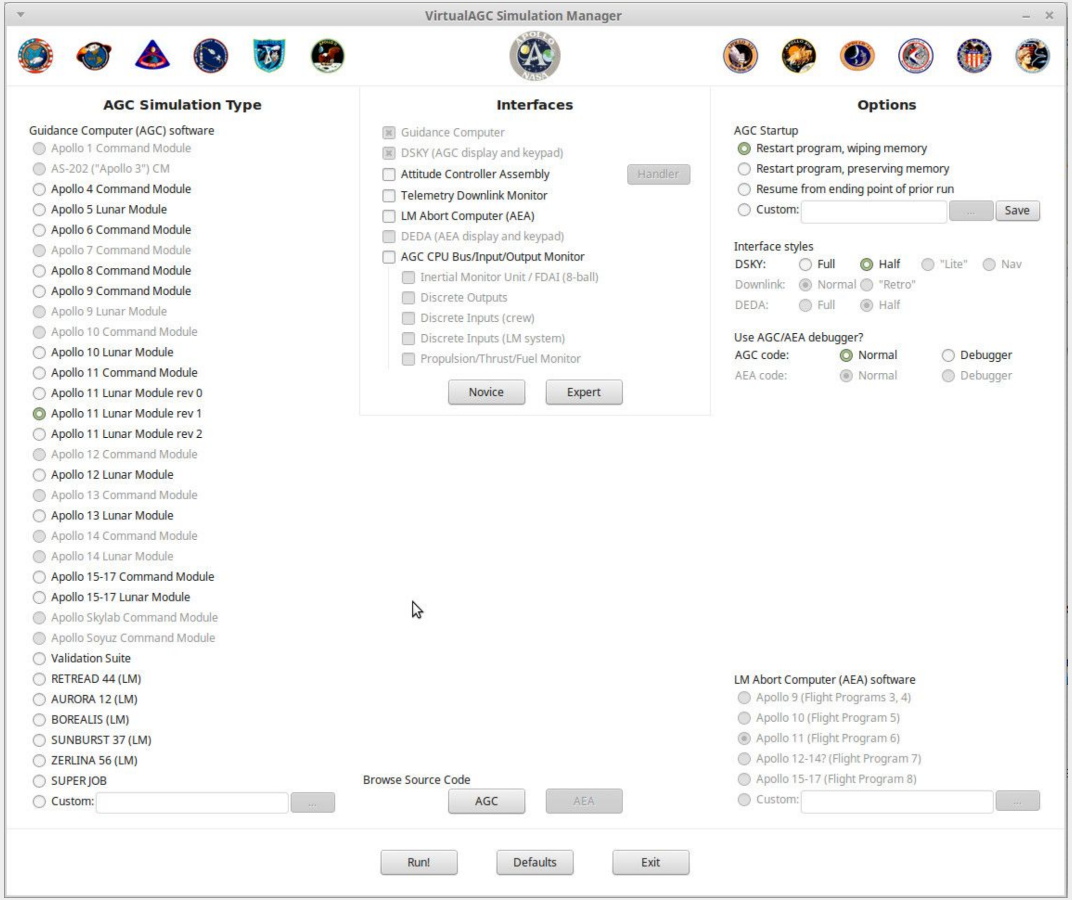


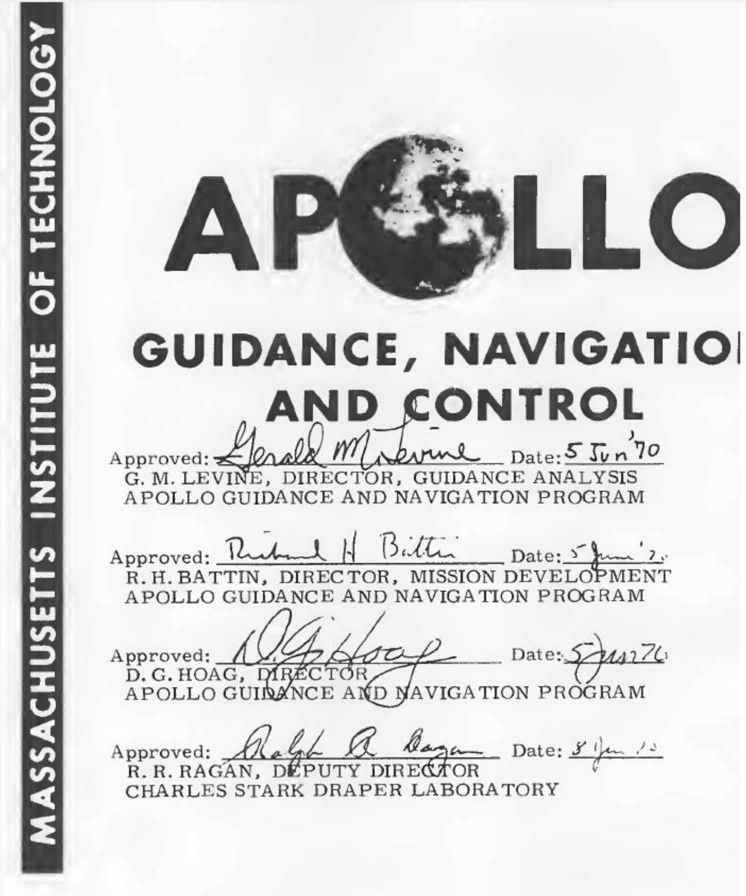
Historical Relevance



```
GAP:  ASSEMBLE REVISION 001 OF AGC PROGRAM LMY99 BY NASA 2021112-061          16:27 JULY 14,1969  LMDAP   .015  PAGE 1470

L        KALMAN FILTER                                                                  USER'S PAGE NO.   1      E0 S4

 0001    REF   3  LAST 1455   E6,1522                          EBANK= NO.UJETS
 0002                         16,3641                          BANK   16
 0003    REF   3  LAST 1421   16,2000                          SETLOC DAPS1
 0004                         16,3641                          BANK

 0005    REF   1                                               COUNT* $$/DAP

 0006    REF 102  LAST 1463   16,3641  3 4752 0  RATELOOP CA     TWO
 0007    REF   3  LAST  133   16,3642  55'744 0           TS     DAPTEMP6
 00071                        16,3643  6 0000 1           DOUBLE
 00072   REF 416  LAST 1469   16,3644  54 002 1           TS     Q
 0008    REF   4  LAST 1470   16,3645  51'744 1           INDEX  DAPTEMP6
 0009    REF  14  LAST 1460   16,3646  11'524 1           CCS    TJP
 0010                         16,3647  1 3651 1           TCF    +2
 0011    REF   1              16,3650  1 3670 1           TCF    LOOPRATE
 0012    REF   1              16,3651  6 3731 1           AD     -100MST6
 0013                         16,3652  0 0006 1           EXTEND
 0014    REF   1              16,3653  6 3707 1           BZMF   SMALLTJU
 0015    REF   5  LAST 1470   16,3654  51'744 1           INDEX  DAPTEMP6
 0016    REF  15  LAST 1470   16,3655  11'524 1           CCS    TJP
 0017    REF   2  LAST 1470   16,3656  3 3731 1           CA     -100MST6
 0018                         16,3657  1 3661 1           TCF    +2
 0019    REF   3  LAST 1470   16,3660  4 3731 0           CS     -100MST6
 0020    REF   6  LAST 1470   16,3661  51'744 1           INDEX  DAPTEMP6
 0021    REF  16  LAST 1470   16,3662  27'524 1           ADS    TJP
 0022    REF   7  LAST 1470   16,3663  51'744 1           INDEX  DAPTEMP6
 0023    REF  17  LAST 1470   16,3664  11'524 1           CCS    TJP
 0024    REF   1              16,3665  4 3615 1           CS     -100MS           0.1 AT 1
 0025                         16,3666  1 3670 1           TCF    +2
 0026    REF   2  LAST 1470   16,3667  3 3615 0           CA     -100MS
 0027                         16,3670  0 0006 1  LOOPRATE EXTEND
 0028    REF   8  LAST 1470   16,3671  5 1744 1           INDEX  DAPTEMP6
 0029    REF   3  LAST 1437   16,3672  7 1521 1           MP     NO.PJETS
 0030    REF 288  LAST 1464   16,3673  3 0001 0           CA     L
 0031    REF   9  LAST 1470   16,3674  51'744 1           INDEX  DAPTEMP6
 00311   REF  29  LAST 1451   16,3675  55'737 1           TS     DAPTEMP1         SIGNED TORQUE AT 1 JET-SEC FOR FILTER
 00312                        16,3676  0 0006 1           EXTEND
 00313   REF  51  LAST 1410   16,3677  7 4742 0           MP     BIT10            RESCALE TO 32; ONE BIT ABOUT 2 JET-MSEC
 00314                        16,3700  0 0006 1           EXTEND
 00315   REF   1              16,3701  6 3732 1           BZMF   NEGTORK
 00316   REF 417  LAST 1470   16,3702  50 002 0  STORTORK INDEX  Q                INCREMENT DOWNLIST REGISTER.
 00317   REF   7  LAST  133   16,3703  27'513 0           ADS    DOWNTORK           NOTE: NOT INITIALIZED; OVERFLOWS.

 0032    REF  10  LAST 1470   16,3704  11'744 0           CCS    DAPTEMP6
 0033    REF   2  LAST 1428   16,3705  1 3642 0           TCF    RATELOOP +1
 0034    REF   1              16,3706  1 3716 0           TCF    ROTORQUE
 0035    REF 303  LAST 1463   16,3707  3 4755 1  SMALLTJU CA     ZERO
 0036    REF  11  LAST 1470   16,3710  51'744 1           INDEX  DAPTEMP6
 0037    REF  18  LAST 1470   16,3711  57'524 0           XCH    TJP
 0038                         16,3712  0 0006 1           EXTEND
```

Historical Relevance




MASSACHUSETTS INSTITUTE OF TECHNOLOGY

APOLLO

GUIDANCE, NAVIGATIO[N]

AND CONTROL

Approved: Gerald M Levine Date: 5 Jun '70
G. M. LEVINE, DIRECTOR, GUIDANCE ANALYSIS
APOLLO GUIDANCE AND NAVIGATION PROGRAM

Approved: Richard H Battin Date: 5 Jun '7.
R. H. BATTIN, DIRECTOR, MISSION DEVELOPMENT
APOLLO GUIDANCE AND NAVIGATION PROGRAM

Approved: D.G. Hoag Date: 5 Jun 70
D. G. HOAG, DIRECTOR
APOLLO GUIDANCE AND NAVIGATION PROGRAM

Approved: Ralph R. Ragan Date: 8 Jun 70
R. R. RAGAN, DEPUTY DIRECTOR
CHARLES STARK DRAPER LABORATORY

THE APOLLO RENDEZVOUS NAVIGATION FILTER THEORY, DESCRIPTION AND PERFORMANCE

Historical Relevance

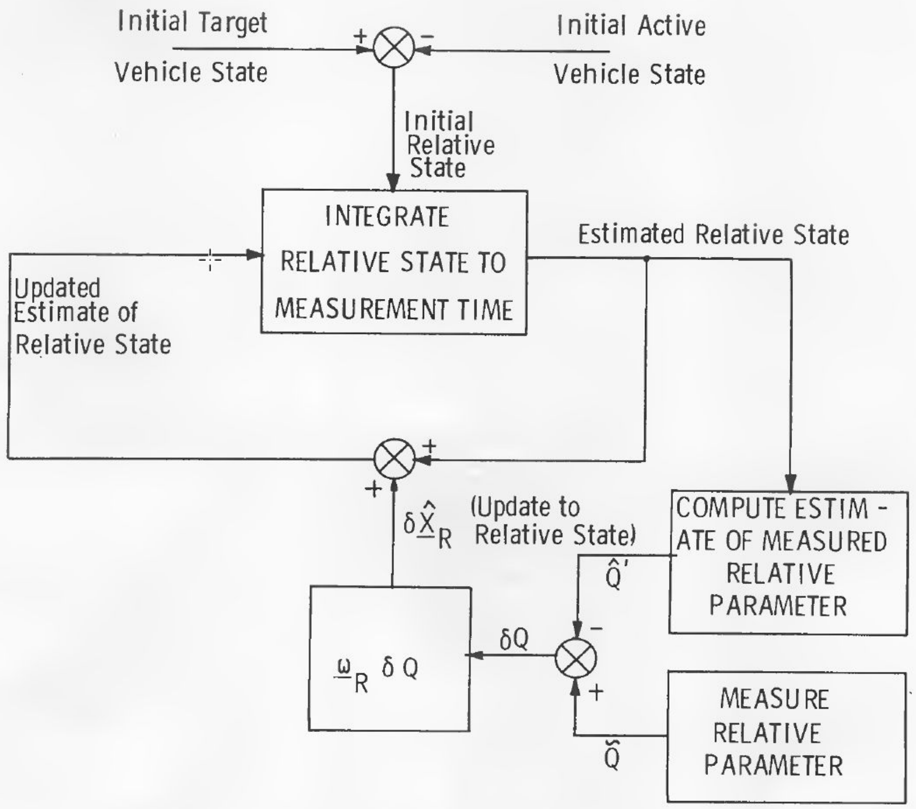


Fig. II-1 Rendezvous Navigation System Using Recursive Kalman Filter

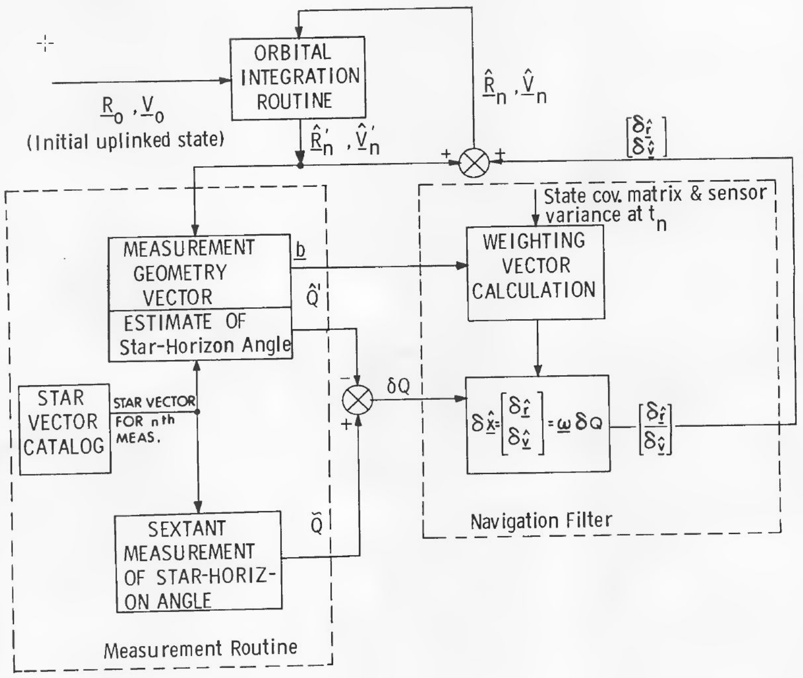


Fig. I-1 Cislunar Recursive Navigation System

Applications

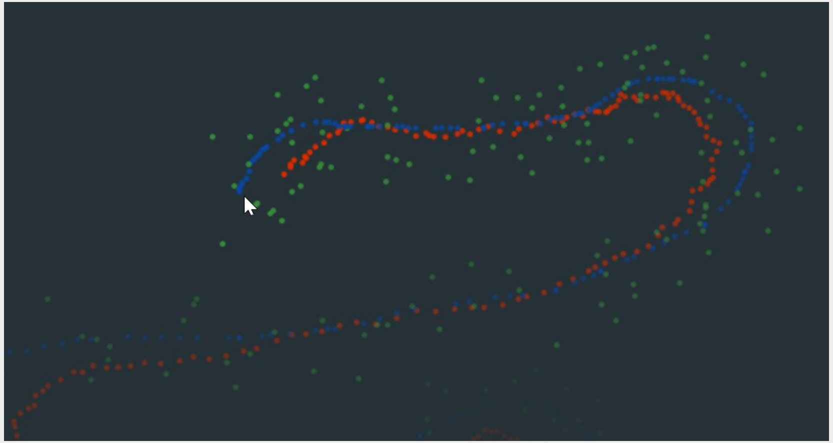
1. Navigation (Apollo 11!)
2. Seismology
3. Autopilot
4. Vehicle control (Sensorless control of AC motor variable-frequency drives)
5. many more!

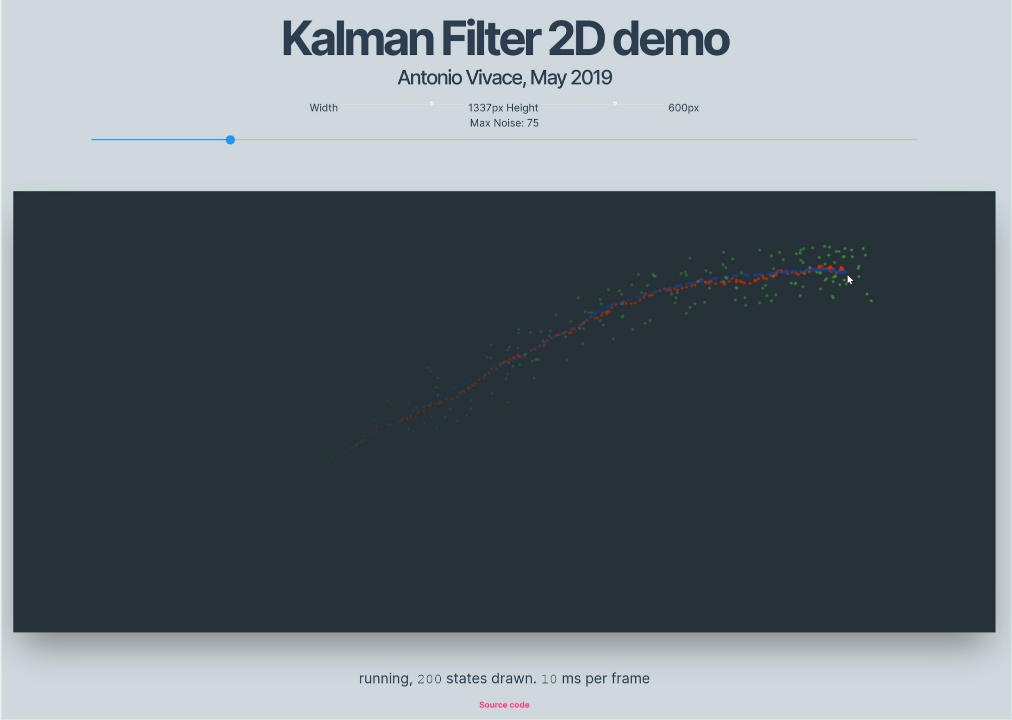
- Attitude and heading reference systems
- Autopilot
- Battery state of charge (SoC) estimation[57][58]
- Brain-computer interface
- Chaotic signals
- Tracking and vertex fitting of charged particles in particle detectors[59]
- Tracking of objects in computer vision
- Dynamic positioning
- Economics, in particular macroeconomics, time series analysis, and econometrics[60]
- Inertial guidance system
- Nuclear medicine – single photon emission computed tomography image restoration[61]
- Orbit Determination
- Power system state estimation
- Radar tracker
- Satellite navigation systems
- Seismology[62]
- Sensorless control of AC motor variable-frequency drives
- Simultaneous localization and mapping
- Speech enhancement
- Visual odometry
- Weather forecasting
- Navigation system
- 3D modeling
- Structural health monitoring
- Human sensorimotor processing[63

Demo (1)

Software stack:

- Javascript
- VueJS
- **SylvesterJS**
- MuseUI
- **CanvasRenderingContext2D web API**





Why:

- Cross platform
- Reactive: realtime editable parameters
- Presets
- Paths
- Customisable canvas and stable performance
- Working on touchscreen devices
- Fully client side

Demo (2)

m = [measuredX, measuredY, xVelocity, yVelocity] ← *measurement vector*

c = [0, 0, 0, 0] ← *control vector*

Prediction Step:

x = (A * x) + (B * c)

P = (A * P * A^T) + Q ← *A^T is the matrix transpose of A*

Correction Step:

S = (H * P * H^T) + R ← *H^T is the matrix transpose of H*

K = P * H^T * S^{-1} ← *S^{-1} is the matrix inverse of S*

y = m - (H * x)

x = x + (K * y)

P = (**I** - (K * H)) * P ← ***I** is the Identity matrix*

predX = x

*predX = (A * predX) + (B * c)*

x ← [xPos, yPos, xVel, yVel]

Demo (3)

Drawing

```
for (let i = this.states.length - 1; i > 0; --i) {
      let state = this.states[i];
      state.display();
      state.update();
      if (state.dead) {
            this.states.splice(i, 1);
      }
}
```

Keeping the simulation framerate costant:

```
// See ya in 1000/desiredFramerate milliseconds
this.lastCalledTime = performance.now();
let ms = performance.now() - start;
this.ms = ms
setTimeout(this.frame, 1000 / this.framerate - ms);
```

Parameter Setting

```
// State Transition
this.A = m([
        [1, 0, 0.2, 0],
        [0, 1, 0, 0.2],
        [0, 0, 1, 0],
        [0, 0, 0, 1]
]);
```

```
this.H = m([
        [1, 0, 0, 0],
        [0, 1, 0, 0],
        [0, 0, 1, 0],
        [0, 0, 0, 1]
]);
```

```
this.Q = m([
        [0.001, 0, 0, 0],
        [0, 0.001, 0, 0],
        [0, 0, 0, 0],
        [0, 0, 0, 0]
]);
```

```
this.R = m([
        [0.1, 0, 0, 0],
        [0, 0.1, 0, 0],
        [0, 0, 0.1, 0],
        [0, 0, 0, 0.1]
]);
```

Demo (4)

Kalman Filter implementation (simple, ignoring c and B)

```
// Adding artificial and gaussian noise
// m = [noisyX, noisyY, deltaX, deltaY]
let deltaX = noisyX - this.lastPoint.elements[0];
let deltaY = noisyY - this.lastPoint.elements[1];
let measurement = v([noisyX, noisyY, deltaX, deltaY]);

// PREDICTION step
// x = (A * x) + (B * c)
var x = this.A.multiply(this.lastPoint)
// P = (A * P * AT) + Q
var P = ((this.A.multiply(this.last)).multiply(this.A.transpose())).add(this.Q);

// CORRECTION step
// S = (H * P * HT) + R
var S = ((this.H.multiply(P)).multiply(this.H.transpose())).add(this.R);
// K = P * HT * S-1
var K = (P.multiply(this.H.transpose())).multiply(S.inverse());
// y = m - (H * x)
var y = measurement.subtract(this.H.multiply(x));
// x = x + (K * y)

//  this is the final filtered point for this iteration
this.lastPoint = x.add(K.multiply(y));
// P = (I - (K * H)) * P
this.last = ((window.Matrix.I(4)).subtract(K.multiply(this.H))).multiply(P);
```

Demo: experimentation and conclusions

1. Presets
2. Paths
3. Mouse input
4. Random Path
5. Changing R during the simulation
6. H and c
7. Q and R estimation
   Autocovariance Least-Squares (ALS) Package