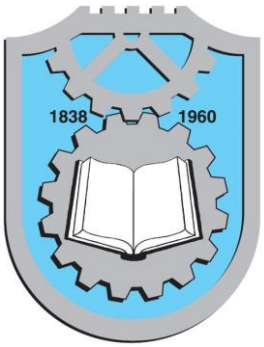
Универзитет у Крагујевцу
Факултет Инжењерских наука



Извештај лабораторијског задатка на тему

„Калманов филтар“

Студент:
Саво Вуковић 578/2016

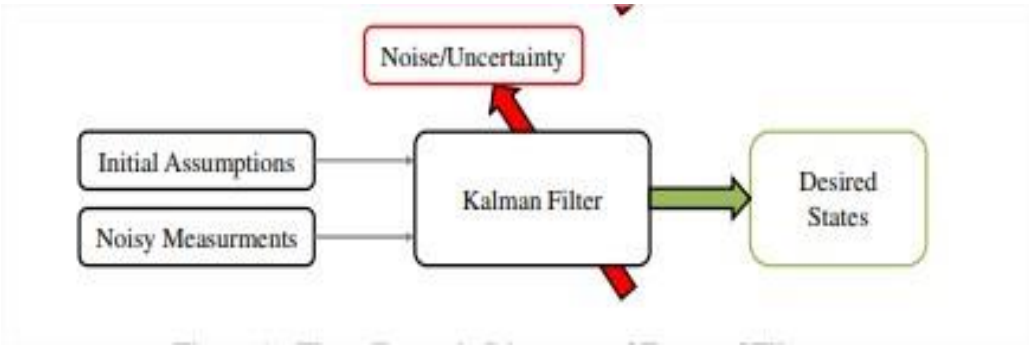
Предметни професор:
Милан Матијевић

Крагујевац 2018.

Садржај

1. Увод

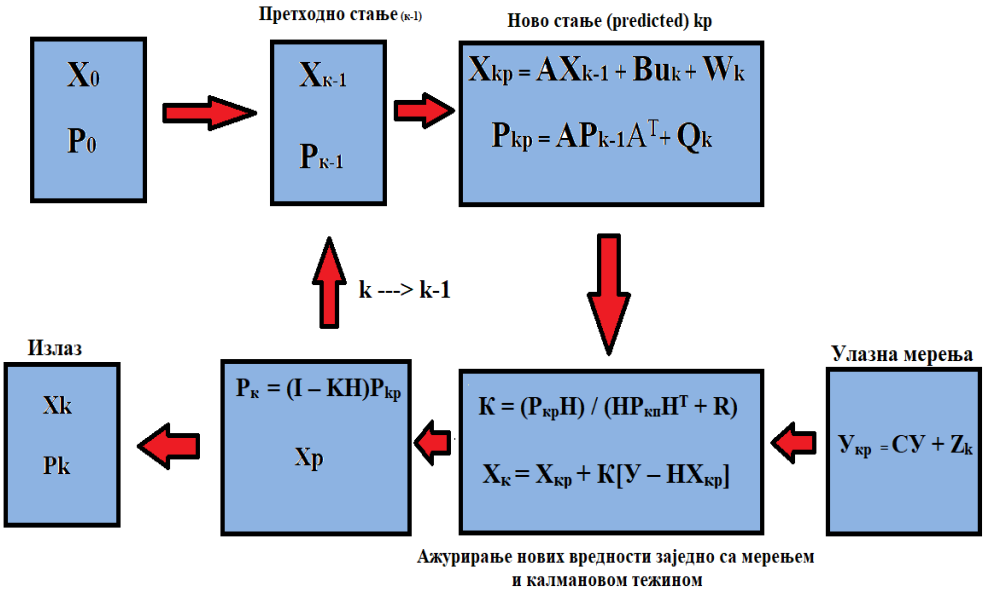
Калманов филтар представља веома корисну алатку за много разноврсних ситуација. Сам филтар је први развио Рудолф Калман и он представља алгоритам који на основу претходно добијених мерења заједно са шумом или сметњама претвара у процене које су далеко боље од самих мерења. Калманов филтер има бројне примене у технологији. Заједничка апликација је за вођење, навигацију и контролу возила, посебно ваздухоплова и свемирских летелица. Штавише, Калман филтар је широко примењен концепт у анализи временских серија који се користе у областима као што су обрада сигнала и економетрија. Калманови филтери су такође једна од главних тема у области роботског планирања и контроле покрета, а понекад су укључена у оптимизацију путање. Калман филтер такође ради на моделовању контроле кретања централног нервног система. Због временског кашњења између издавања моторичких команди и примања сензорних повратних информација, коришћење Калман филтера подржава реалан модел за процену тренутног стања моторног система и издавање ажурираних команди. Калман филтар прима мерења која поседују одређени шум или сметње и циљ је да услед тих сметњи издвоји сигнал који је нама од користи.



Слика 1. Пример функционисања Калман филтера

2. Калман филтар - Алгоритам

Калман филтар користи предикцију коју прати корекција како би се утврдило стање филтера. Ова техника се понекад зове „**predictor-corrector**“ или „**prediction update**”. Главна идеја је да се користи информација о динамици стања и система како би филтер предвидео које ће следеће стање бити. Једноставан пример овог алгоритма би био кретање по правој линији одређеном брзином тако да на основу претходног стања и кретања, пешак може предвидети где ће бити у одређеном временском интервалу. Корекција или „**update**” део укључује поређење мерења са претпоставком филтера у односу на наша предвиђења стања. На следећој слици дато је једноставно функционисање алгоритма.



Слика 2. Пример функционисања алгоритма код Калмановог филтра

У почетку као што се може видети са слике 2 узимају се одговарајуће вредности претходног стања, где $X_{к-1}$ представља „State matrix”, матрица стања или преходна процена, а $P_{к-1}$ представља претходну „State error covariance matrix” или уколико је у питању прва итерација ”Initial state error covariance matrix”. Уколико имамо неки сензор или више њих који прате одговарајуће стање објекта они припадају матрици стања док матрица грешке коваријанце представља грешку у самом процесу. У следећем кораку можемо видети једначину:

$$X_{кр} = AX_{к-1} + Bu_k + W_к$$

Где $\mathbf{X_{кр}}$ представља следеће предвиђено стање (нпр. следећи корак који ће наш пешак узети) . Матрице **A** и **B** у једначини се зову „State matrix" и као што име каже представљају стање или динамику самог система. Члан $\mathbf{X_{к-1}}$ представља претходно стање које смо узели у претходном кораку, **u** улазни вектор а $\mathbf{W_к}$ представља унутрашње сметње процеса (нпр. уколико меримо одређено кретање или слетање авиона ова променљива би представљала турбуленцију) .

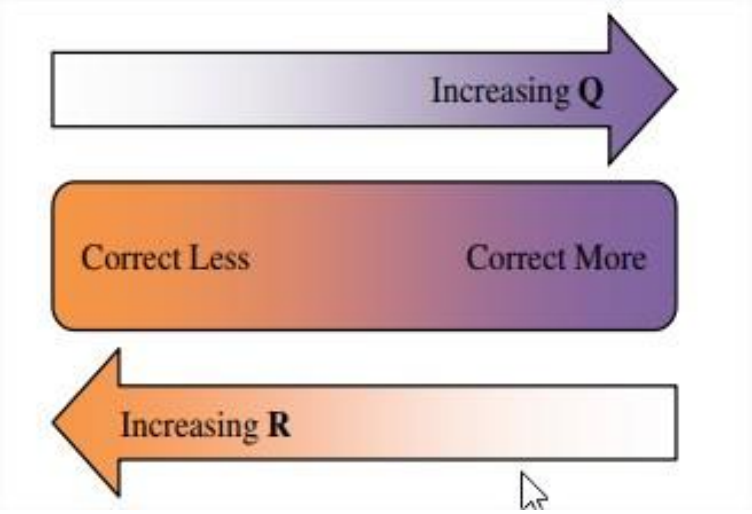
$$\mathbf{P_{кр} = AP_{к-1}A^T + Q_k}$$

Претходна једначина служи за предвиђање следеће вредности „State error covariance" матрице где **Qk** представља грешку у овом процесу.

Следећи корак представља једну од најважнијих целина овог алгоритма а то је израчунавање „Kalman gain" или калманове тежине. Као што можемо видети главна једна од главних једначина у овом кораку је.

$$\mathbf{K = (P_{кр}H) / (HP_{кп}H^T + R)}$$

K као што се може и претпоставити представља калманову тежину, матрица **H** представља матрицу која нам омогућава да модификујемо цео израз тако добијемо матрицу жељених димензија потребне за калманову тежину, као што се може приметити у овом изразу се користи и $\mathbf{P_{кр}}$ или предвиђена матрица стања ковариансе дефинисана у претходном кораку и величина **R** која представља грешку у мерењу (нпр имамо одређени сензор и он има одређене грешке при мерењу напона у неком колу).



Слика 3. Граф функционисања калмановог гејна

На овом графу је веома лепо објашњено функционисање саме Калманове тежине (нпр замислите да имате одређену вагу на којој имају одређене тежине, да би вага остала избалансирана морамо ,са једне и са друге стране, да узмемо одређене тежине. Уколико је R променљива велика то значи да треба мање тежине да узмемо са оне стране ваге где се налазе наша мерења а више тежине са оне стране где се налазе наше предвиђене вредности и обрнуто). Овај претходан пример објашњава практично срж функционисања самог филтера и то је представљено следећом једначином.

$$\mathbf{X_к = X_{кр} + K[У - HX_{кр}]}$$

Променљива $X_к$ представља тренутни излаз из самог филтера тј тренутну „State" матрицу која је једнака збиру $X_{кп}$ која представља предвиђену вредност и производу калманове тежине и члана који садржи променљиву **У** коју раније нисмо дефинисали и која представља вредност мерења самог система и описана је једначином.

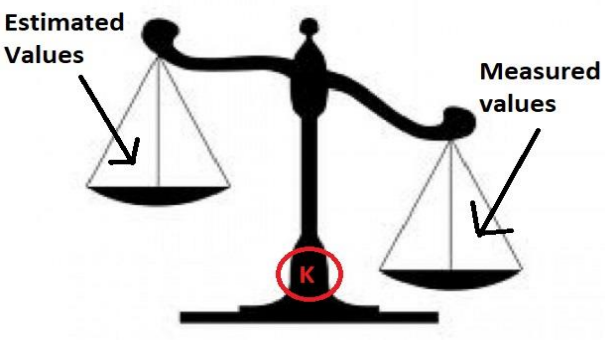
$$\mathbf{У_{кр} = СУ + Z_k}$$

Као и код претходних корака матрица **С** представља одређену матрицу за представљање диманике самог система, а Z_k представља одређени шум у мерењу.Практично матрица С омогућава да променљива $У_{кр}$ има исти матрични облик као и матрица стања Х.

У претпоследњој једначини можемо видети да што је К мање то ће се више „тежине" ставити на страну где су предвиђене вредности и обрнуто, као што је већ и описано. У последњем кораку уносе се нове вредности матрице стања коваријанце на основу величине калманове тежине:

$$\mathbf{P_к = (I - KH)P_{kp}}$$

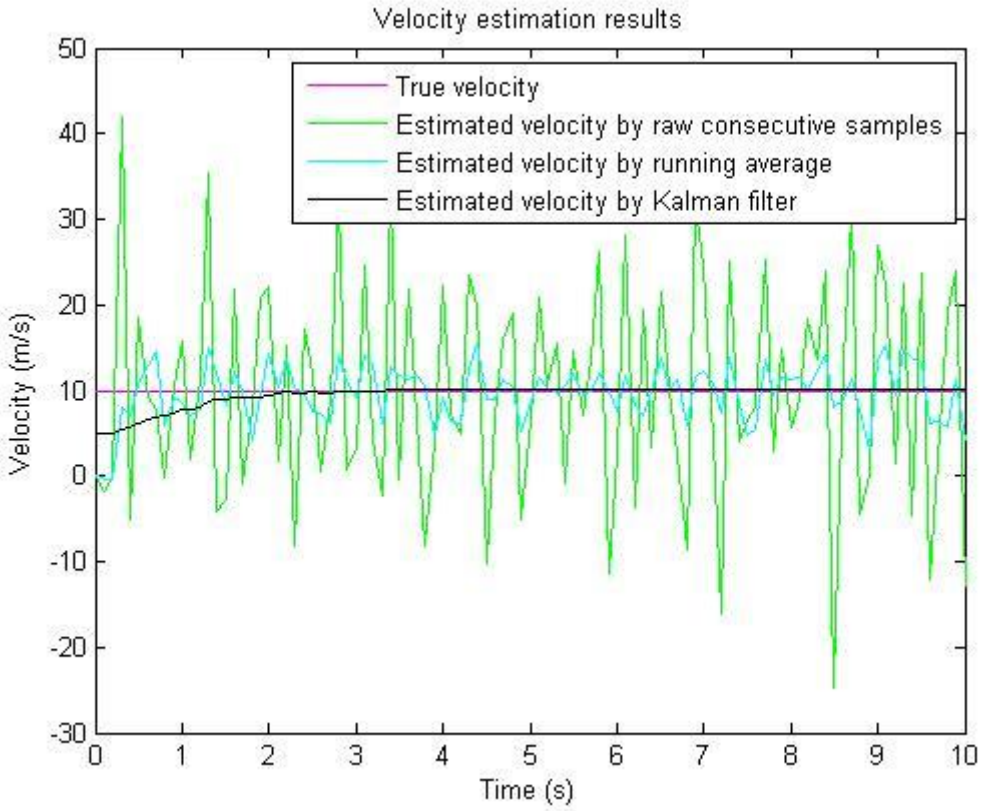
Овде можемо видети да за рачунање нове матрице стања коваријанце велику улогу опет има променљива К као и предвиђено стање $Р_{кр}$. У следећој итерацији „тренутне" вредности $\mathbf{P_к}$ и $\mathbf{X_к}$ постају претходне променљиве у следећој циклусу и овај процес се понавља за све вреености које улазе у дати филтар.



Слика 4. Илустрација рада Калмановог филтра

3. Реализација - Калман филтар

Ради илустрација рада самог филтра написао сам програм у програмском језику **Python,** који уноси одређене улазне податке нпр узлетање авиона, мерење температуре, ниво водостаја на реци, мерење напона у неком колу итд. С обзиром да ова мерења поседују одређени шум услед кога не можемо да лепо да видимо одређену вредност ових мерења искористићемо одговарајући Калман филтар.



Слика 5. Функционисање самог Калман филтра

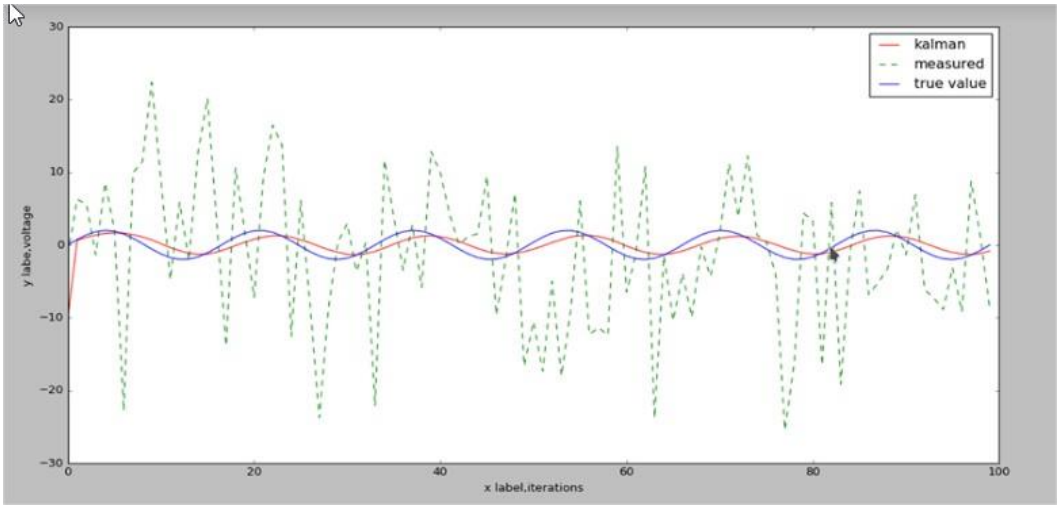
Као што се може видети на слици љубичастом бојом је приказана права вредност коју желимо да видимо, зеленом одређени сигнал који меримо а тамно црном бојом је приказана вредност на излазу Калман филтера које је веома близу правој вредности и која нам омогућава да прочитамо одговарајућу вредност. На овај начин приказано је основно функционисање Калман филтра који на улазу прима одређени сигнал са шумом и из којег узимамо одређене приближне вредности. На следећој слици је приказана реализација филтра у програмском језику Python.

```python
import numpy as np
import matplotlib.pyplot as plt

plt.rcParams['figure.figsize'] = (10, 8)

# intial parameters
n_iter = 50
sz = (n_iter,) # size of array
x = -0.37727 # truth value (typo in example at top of p. 13 calls this z)
z = np.random.normal(x,0.1,size=sz) # observations (normal about x, sigma=0.1)

Q = 1e-5 # process variance

# allocate space for arrays
xhat=np.zeros(sz)      # a posteri estimate of x
P=np.zeros(sz)         # a posteri error estimate
xhatminus=np.zeros(sz) # a priori estimate of x
Pminus=np.zeros(sz)    # a priori error estimate
K=np.zeros(sz)         # gain or blending factor

R = 0.1**2 # estimate of measurement variance, change to see effect

# intial guesses
xhat[0] = 0.0
P[0] = 1.0

for k in range(1,n_iter):
    # time update
    xhatminus[k] = xhat[k-1]
    Pminus[k] = P[k-1]+Q

    # measurement update
    K[k] = Pminus[k]/( Pminus[k]+R )
    xhat[k] = xhatminus[k]+K[k]*(z[k]-xhatminus[k])
    P[k] = (1-K[k])*Pminus[k]

plt.figure()
plt.plot(z,'k+',label='noisy measurements')
plt.plot(xhat,'b-',label='a posteri estimate')
plt.axhline(x,color='g',label='truth value')
plt.legend()
plt.title('Estimate vs. iteration step', fontweight='bold')
plt.xlabel('Iteration')
plt.ylabel('Voltage')
```

Слика 7. Реализација задатка у пајтону

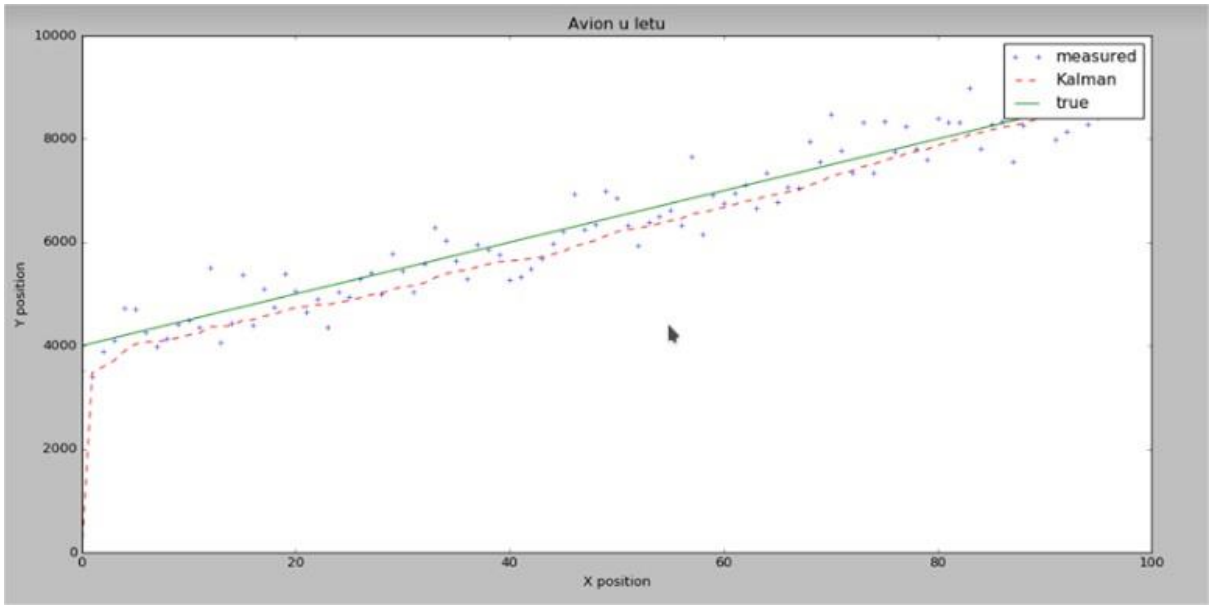
На слици 7 је приказана реализација самог филтра са једном променљивом где практично можемо да изоставимо одређене матрице стања и на овај начин можемо практично да видимо сам скелет овог алгоритма, који је као што се може видети реализован практично преко једне петље неколико променљивих и наравно библиотека као што су **matplotlib, numpy, math** без којих реализација пројекта не би била могућа

На следећој слици је дат пример мерења неизменичног напона у одређеном електронском колу, због присуства одређеног шума као што се може видети не можемо лепо да видимо како изгледа тај излазни напон нити да измеримо одређене караткеристике које су нам потребне у овом колу. Измерени напон је описан зеленом бојом, филтриран сигнал помоћу Калмановог филтра је дат црвеном бојом а права вредност тј. вредност која не поседује одређени шум је дата плавом бојом. Може се видети да, уз помоћ Калмановог филтра, добијамо сигнал који је готово идентичан правој вредности.



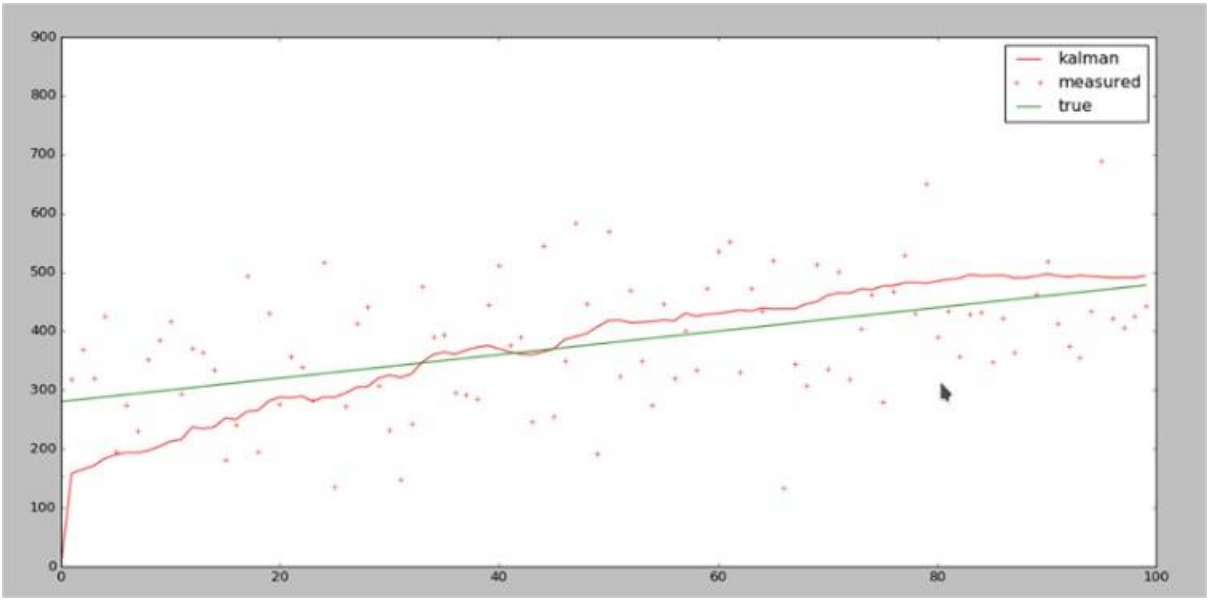
Слика 8. Примена Калмановог филтра

Замислите следећу ситуацију где меримо висину и брзину одређеног авиона при његовом лету. Због присуства одређених сметњи нпр. због временских неприлика или неких других спољашњих околности, на сензору добијамо одређени шум који треба да „подесимо“ како бисмо добили вредност коју можемо да користимо. На следећим сликама је приказана примена Калмановог филтра на датом примеру:



Слика 9. Примена Калмановог филтра за мерење висине авиона.

Овде можемо видети алгоритам присутан код Калмановог филтра који добија готово идеалну вредност коју можемо искористити приликом мерења висине авиона. Наравно код данашњих радара и сензора који су базирани на примени Калмановог филтра сама имплементација је далеко сложенија док овде посматрамо само висину авиона и његову брзину и дато убрзање.



Слика 10. Примена Калмановог филтера за добијање брзине авиона

Слика 10 приказује примену Калмановог филтера за мерење брзине авиона где претпостављамо да промена брзине авиона нема никакве везе са самом висином авиона.

4. Закључак

Као што смо видели у претходним примерима Калман филтер представља веома корисну алатку за разне проблеме које могу да се нађу на нашем путу. Пре свега за сваку итерацију је потребно само претходно стање последње итерације да памтимо чиме знатно олакшавамо рад самог програма јер не морамо да бринемо о самој меморији система. Веома је користан за комбинацију са разним сензорима и може се рећи да је сам филтар реализован као нека врста вештачке интелигенције с обзиром да на основу претходних стања доноси одлуке о следећим могућим стањима система што га чини знатно интересантнијим за само имплементирање

5. Референце

- https://www.youtube.com/watch?v=CaCcOwJPytQ&list=PLX2gX-ftPVXU3oUFNATxGXY90AULiqnWT
- https://en.wikipedia.org/wiki/Kalman_filter
- http://scipy-cookbook.readthedocs.io/index.html
  https://home.wlu.edu/~levys/kalman_tutorial/