Experiments planning

Ricardo Westhauser

28/10/2019

Doing some configuration

```
nRuns <- 30  #Number of runs for each set of parameters
gtFileName = "UFRGS_VET_100M_GT_390_613.txt"
escalaPxM = 0.10
divMultExcalaZ = 135
zEscalaMetros = 1.083333333
altitudeFotoRefEscala = 101.09999847412
imgCount = 224 #Trajectory size in image count
vecColor <- c("red", "black", "darkgreen", "cadetblue", "cyan", "azure3", "blue2",
"brown2", "darkgoldenrod1", "darkolivegreen2", "darkorange1", "dodgerblue", "darkor
chid1", "deeppink2", "gold1", "darkred", "deeppink4", "burlywood2", "bisque1", "bei
ge", "aquamarine", "green", "darkslategray", "gold4", "chocolate1", "cornsilk3" )
```

Creating a Full Factorial Design of the experiment: We're currently working with two factors, each one with 11 levels.

```
library(DoE.base)
```

```
## Loading required package: grid
```

```
## Loading required package: conf.design
```

```
## Registered S3 method overwritten by 'DoE.base':
##   method           from
##   factorize.factor conf.design
```

```
##
## Attaching package: 'DoE.base'
```

```
## The following objects are masked from 'package:stats':
##
##     aov, lm
```

```
## The following object is masked from 'package:graphics':
##
##     plot.design
```

```
## The following object is masked from 'package:base':
##
##     lengths
```

```
fullFac <- fac.design(nfactors= 2, replications= 1, repeat.only= FALSE, blocks= 1,
randomize= FALSE,seed= 17366,
                nlevels = c(11, 11), factor.names=list( Alfa=c("0","0.1","0.2","0.
3","0.4","0.5","0.6","0.7","0.8","0.9","1"), Beta=c("0","0.1","0.2","0.3","0.4","0.
5","0.6","0.7","0.8","0.9","1")));
```

```
## creating full factorial with 121 runs ...
```

As we can see, a Full Factorial will have 121 different factor level combinations. I will reduce it to 25 combinations:

```
   ## a full quadratic model with constraint in three quantitative factors
library(DoE.wrapper)
```

```
## Loading required package: FrF2
```

```
## Loading required package: rsm
```

```
   plan <- Dopt.design(25,factor.names=list(Alfa=c(0,1), Beta=c(0,1)),
                         nlevels=c(10,10),
                         formula=~quad(.))
```

```
## creating full factorial with 100 runs ...
```

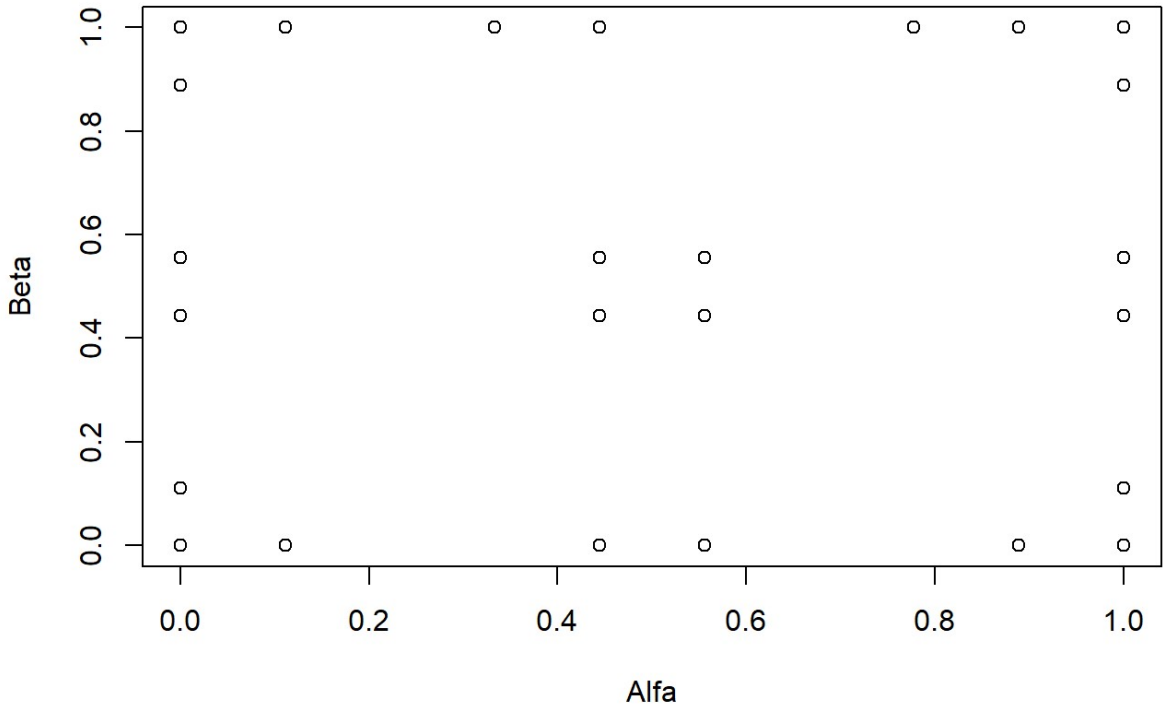
```
   plan
```

```
##         Alfa      Beta
## 1  0.3333333 1.0000000
## 2  0.1111111 1.0000000
## 3  0.4444444 0.0000000
## 4  1.0000000 0.1111111
## 5  1.0000000 0.8888889
## 6  0.4444444 1.0000000
## 7  0.0000000 0.4444444
## 8  1.0000000 1.0000000
## 9  0.0000000 0.8888889
## 10 0.1111111 0.0000000
## 11 0.0000000 0.0000000
## 12 0.8888889 0.0000000
## 13 0.5555556 0.5555556
## 14 0.0000000 0.1111111
## 15 0.0000000 0.5555556
## 16 0.0000000 1.0000000
## 17 0.7777778 1.0000000
## 18 0.5555556 0.0000000
## 19 1.0000000 0.0000000
## 20 0.5555556 0.4444444
## 21 1.0000000 0.4444444
## 22 0.8888889 1.0000000
## 23 1.0000000 0.5555556
## 24 0.4444444 0.4444444
## 25 0.4444444 0.5555556
## class=design, type= Dopt
```

```
   cor(plan)
```

```
##             Alfa        Beta
## Alfa 1.000000000 0.007017793
## Beta 0.007017793 1.000000000
```

```
   y <- rnorm(25)
   r.plan <- add.response(plan, y)
   plot(plan)
```


Beta
0.0
0.2
0.4
0.6
0.8
1.0
Alfa
0.0
0.2
0.4
0.6
0.8
1.0

```
   configSize = nrow(plan)

   c <- 1
   vecConfigs <- vector(mode = "list", length = configSize)
   configPath <- vector(mode = "list", length = configSize)

   #while (c <= configSize)
   #{
   #   vecConfigs[c] <- paste("alfa_", format(round(plan[c,1],7), nsmall = 0), "__b
eta_", format(round(plan[c,2],7), nsmall = 0), sep="")
   #   configPath[c] <- paste("UFRGS_VET_V3_11_08_2011/traj/", vecConfigs[c], sep
="")
   #   c = c+1
   #}


vecConfigs[1] = "alfa_0.4444444__beta_0.4444444"
vecConfigs[2] = "alfa_0__beta_0.5555556"
vecConfigs[3] = "alfa_0.2222222__beta_1"
vecConfigs[4] = "alfa_0.8888889__beta_1"
vecConfigs[5] = "alfa_0.4444444__beta_0"
vecConfigs[6] = "alfa_0.5555556__beta_0"
vecConfigs[7] = "alfa_1__beta_0.8888889"
vecConfigs[8] = "alfa_0.5555556__beta_0.5555556"
vecConfigs[9] = "alfa_0__beta_0"
vecConfigs[10] = "alfa_0__beta_0.1111111"
vecConfigs[11] = "alfa_0.8888889__beta_0"
vecConfigs[12] = "alfa_0.4444444__beta_0.5555556"
vecConfigs[13] = "alfa_0.5555556__beta_0.4444444"
vecConfigs[14] = "alfa_0.1111111__beta_1"
vecConfigs[15] = "alfa_0.1111111__beta_0"
vecConfigs[16] = "alfa_1__beta_1"
vecConfigs[17] = "alfa_1__beta_0.1111111"
vecConfigs[18] = "alfa_0__beta_0.8888889"
vecConfigs[19] = "alfa_0.6666667__beta_1"
vecConfigs[20] = "alfa_0.5555556__beta_1"
vecConfigs[21] = "alfa_0__beta_0.4444444"
vecConfigs[22] = "alfa_1__beta_0.4444444"
vecConfigs[23] = "alfa_1__beta_0.5555556"
vecConfigs[24] = "alfa_0__beta_1"
vecConfigs[25] = "alfa_1__beta_0"



#vecConfigs[26] = "alfa_0.2222222__beta_0"
#vecConfigs[27] = "alfa_0.4444444__beta_1"
#vecConfigs[28] = "alfa_0.6666667__beta_0"

while (c <= configSize)
   {
      configPath[c] <- paste("UFRGS_VET_V3_11_08_2011/traj/", vecConfigs[c], sep
="")
```

```
      c = c+1
   }
```

Reading the ground truth file:

```
groundTruth <- read.table("UFRGS_VET_100M_GT_390_613.txt",
                 header = FALSE)
colnames(groundTruth) [1:4] <- c("X (px)","Y (px)","Z (m)","Yaw (deg)")
```

Reading the experiment files:

```
library(plyr)
```

```
##
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:conf.design':
##
##     join
```

```
library(readr)

experiments <- vector(mode = "list", length = configSize) #Experiments list (one fo
r each config)

c <- 1
while (c <= configSize)
{
   expDir <- paste("UFRGS_VET_V3_11_08_2011/traj/", vecConfigs[c], sep="")
   myfiles = list.files(path=expDir, pattern="*txt", full.names=TRUE)

   r <- 1
   while(r <= nRuns)
   {
      experiments[[c]][[r]] = read.table(myfiles[r], header=TRUE, sep=" ", dec=".")
      r = r+1
   }
   c = c+1
}
```

Calculating the mean for each position in the trajectory:

```
vecImgMeanError <- vector(mode = "list", length = configSize) #Experiments list (on
e for each config)
vecImgMeanX <- vector(mode = "list", length = configSize) #Experiments list (one fo
r each config)
vecImgMeanY <- vector(mode = "list", length = configSize) #Experiments list (one fo
r each config)

c <- 1
while (c <= configSize)#Configs
{
   i <- 1
   while(i <= imgCount)
   {
      xyzError = 0
      sumX = 0
      sumY = 0
      r <- 1
      while(r <= nRuns)#Runs
      {
         xErrorMeters = 0
         yErrorMeters = 0
         zErrorMeters = 0

         sumX = sumX + experiments[[c]][[r]][["meanPX"]][i]
         sumY = sumY + experiments[[c]][[r]][["meanPY"]][i]
         xErrorMeters = ((experiments[[c]][[r]][["meanPX"]][i] * escalaPxM) - (grou
ndTruth[["X (px)"]][i] * escalaPxM))^2
         yErrorMeters = ((experiments[[c]][[r]][["meanPY"]][i] * escalaPxM) - (grou
ndTruth[["Y (px)"]][i] * escalaPxM))^2
         zErrorMeters = ((experiments[[c]][[r]][["meanPZ"]][i] * (altitudeFotoRefEs
cala/zEscalaMetros)) - groundTruth[["Z (m)"]][i])^2

         xyzError = xyzError + sqrt(xErrorMeters + yErrorMeters + zErrorMeters)
         r = r+1
      }
      vecImgMeanError[[c]][[i]] = xyzError/nRuns
      vecImgMeanX[[c]][[i]] = sumX/nRuns
      vecImgMeanY[[c]][[i]] = sumY/nRuns
      i = i+1
   }
   c = c+1
}
```

Ploting all configurations: There are configurations that have an error bigger than 100 meters on some or all pictures/positions.

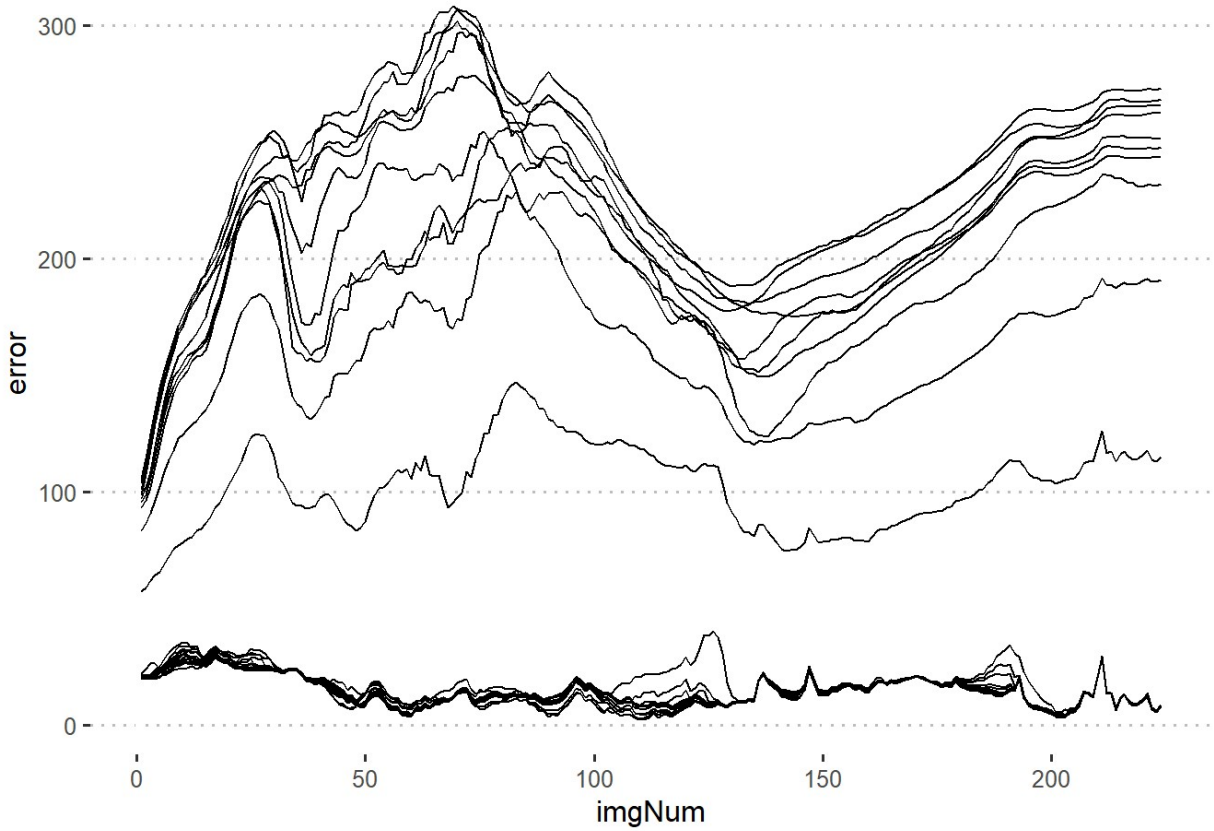
```
library(ggplot2)
library(ggpubr)
```

```
## Loading required package: magrittr
```

```
##
## Attaching package: 'ggpubr'
```

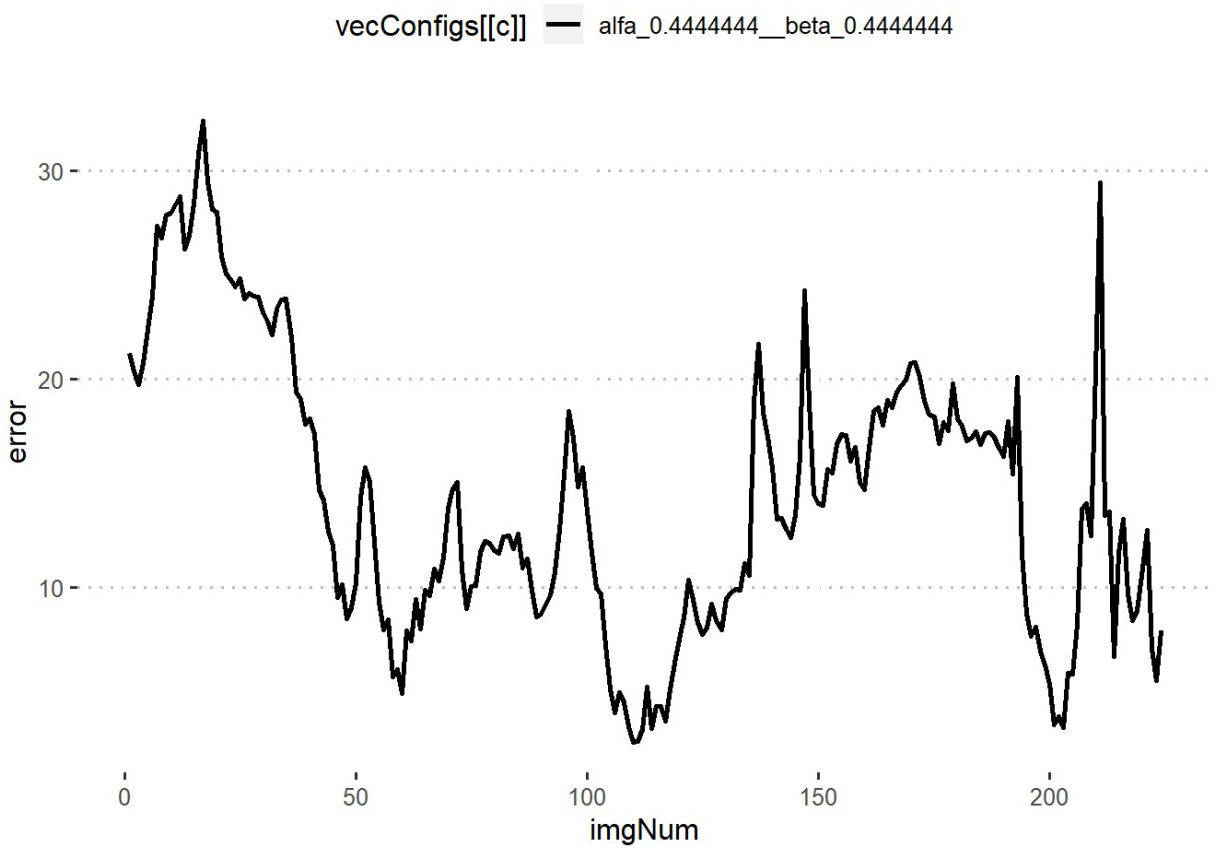
```
## The following object is masked from 'package:plyr':
##
##     mutate
```

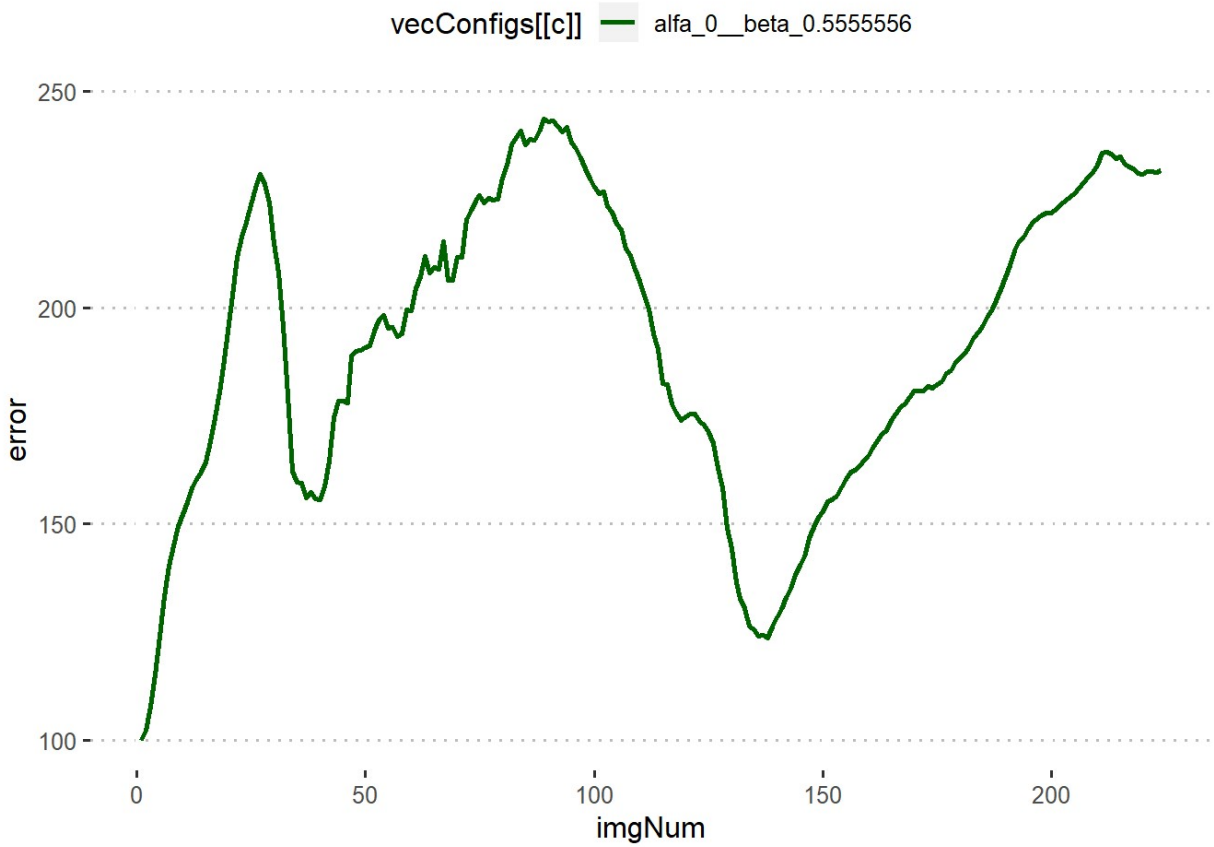
```
library(gridExtra )

vecDf <- vector(mode = "list", length = configSize)

c <- 1
while(c <= configSize)
{
   vecDf[[c]] <- data.frame( imgNum = 1:imgCount, cfName = c, error = matrix(unlist
(vecImgMeanError[[c]]), nrow=length(vecImgMeanError[[c]]), byrow=T))
   c = c+1
}

p <- ggplot(vecDf[[1]], aes(imgNum, error)) + geom_line() + theme_pubclean()

c <- 2
while(c <= configSize)
{
   p <- p + geom_line(data = vecDf[[c]])
   c = c+1
}

p
```

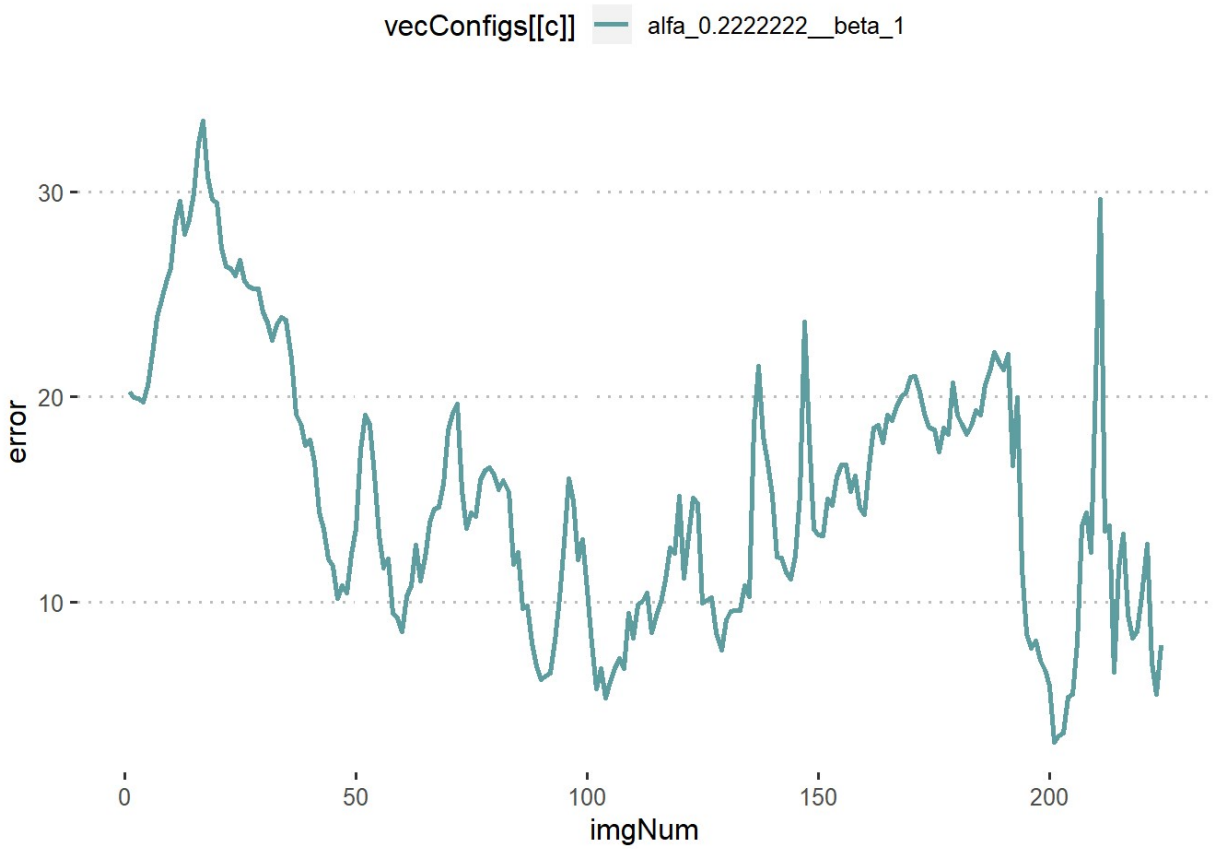


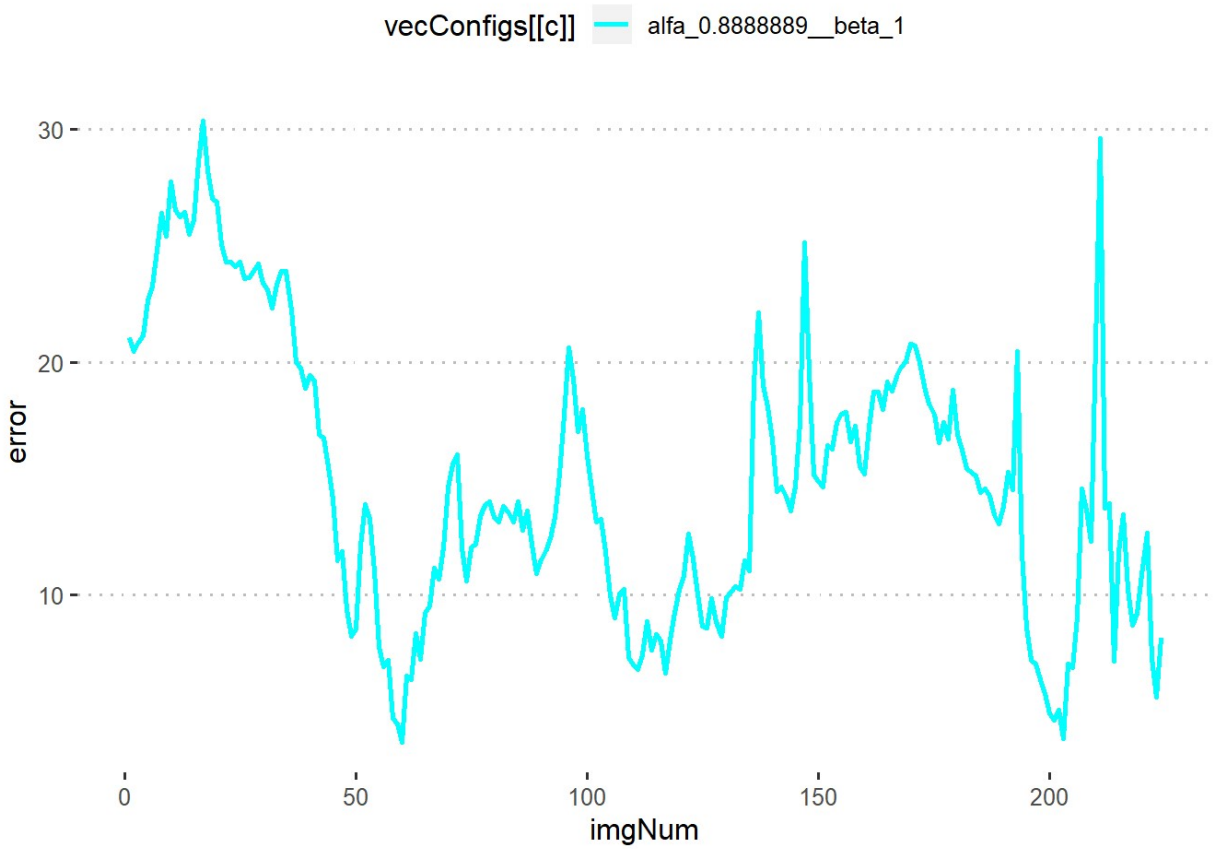
Individual plots showing the mean error: Plotting the mean error on each image of the drone flight. The X axis are the image taken on each moment, while the Y axis represemnt the mean error in meters, of the estimation algorithm.

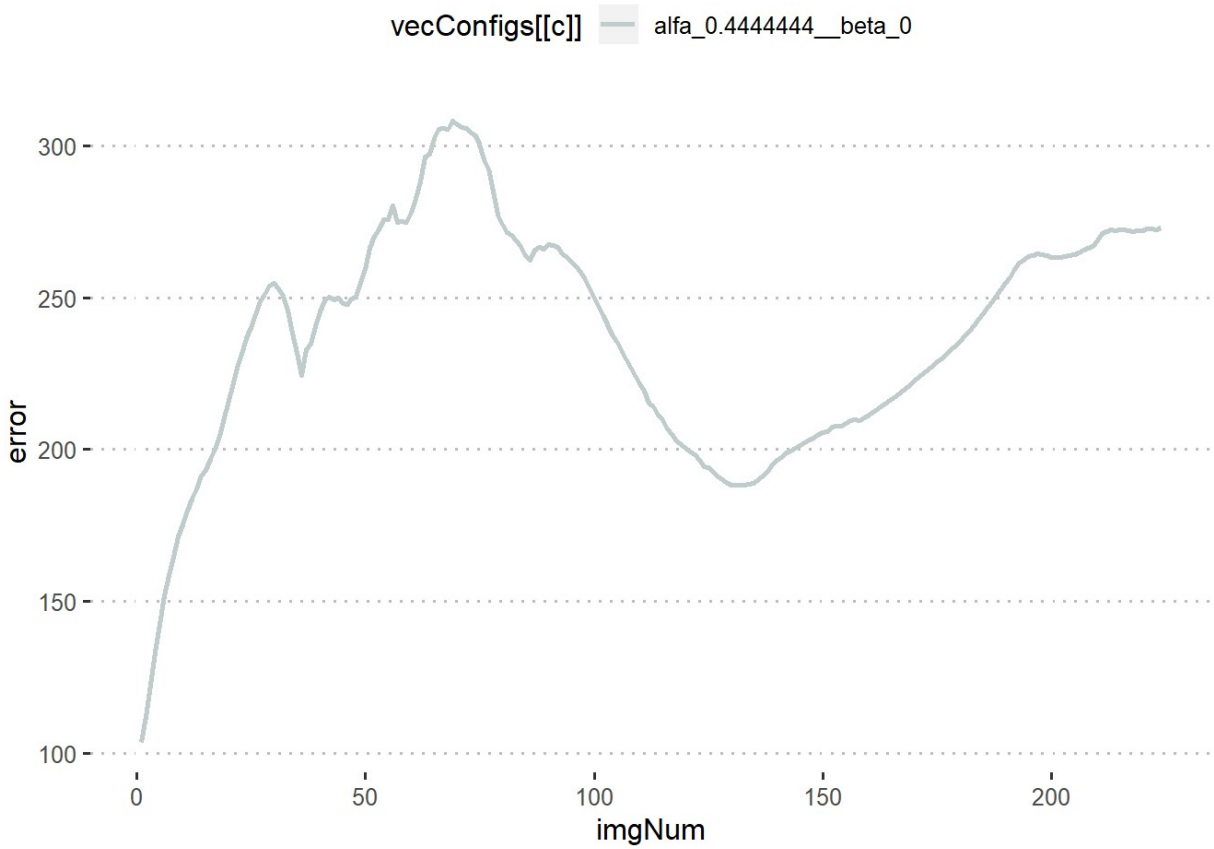
```
c  <- 1
while(c <= configSize)
{
   print(ggplot(vecDf[[c]], aes(imgNum, error)) + geom_line(aes(color = vecConfigs
[[c]]), size=1) + scale_color_manual(values = c(vecColor[c+1])) + theme_pubclean())
   c = c+1
}
```

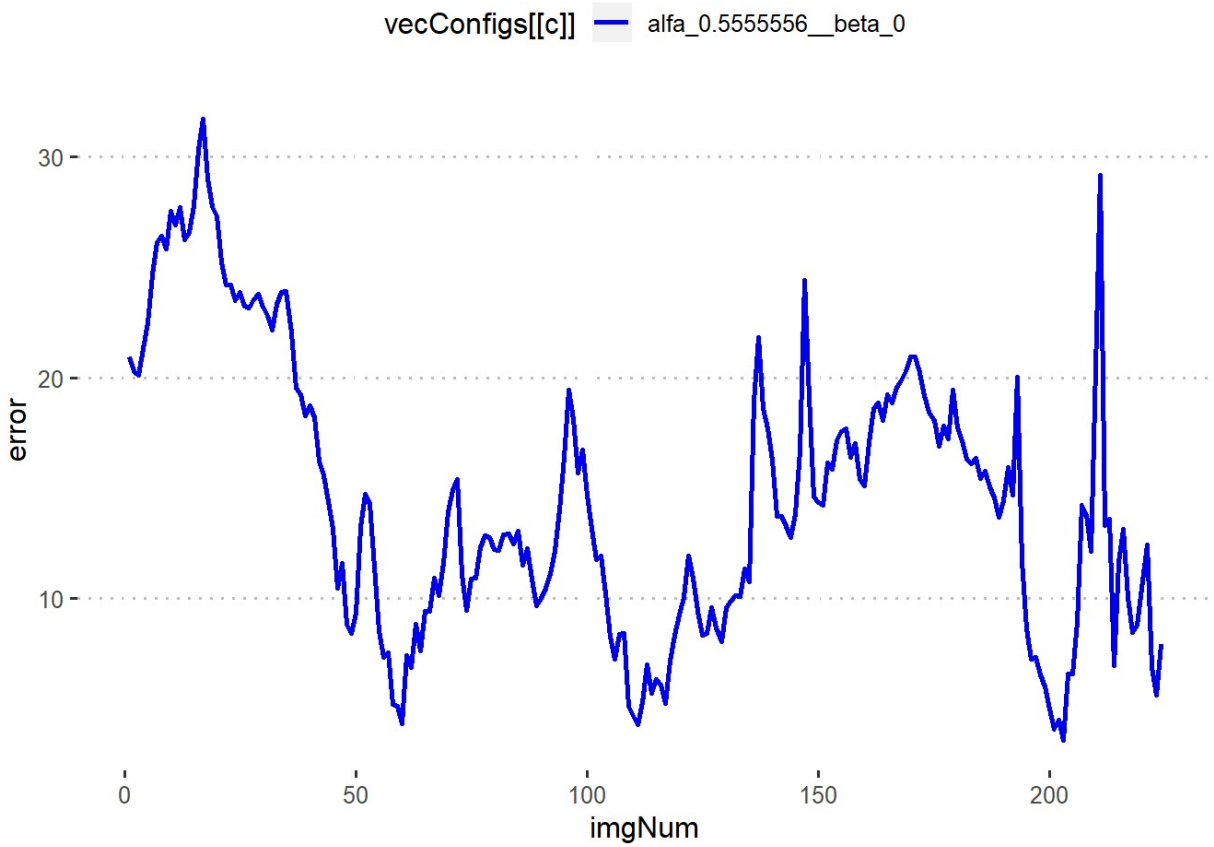

vecConfigs[[c]]
alfa_0.4444444__beta_0.4444444
30
20
10
error
0
50
100
150
200
imgNum

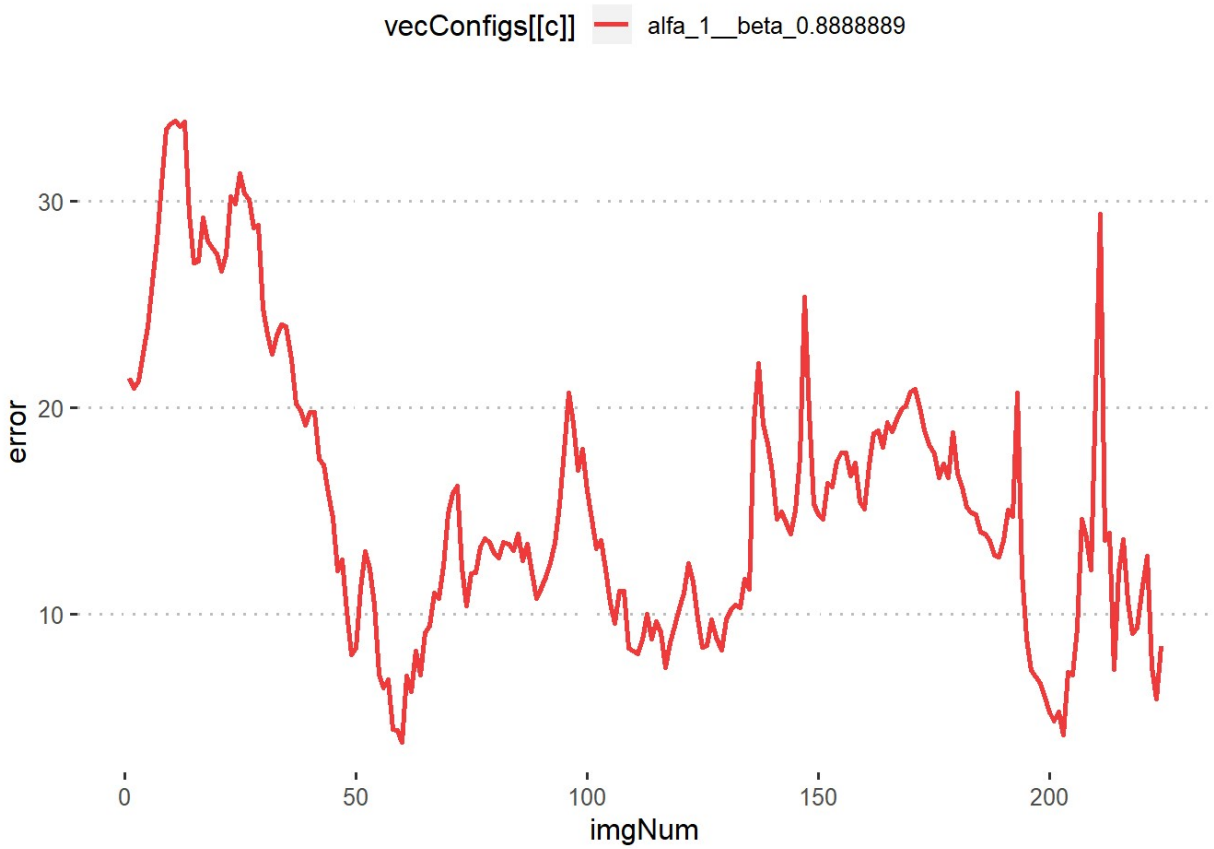

vecConfigs[[c]]
alfa_0__beta_0.5555556
250
200
150
100
error
0
50
100
150
200
imgNum

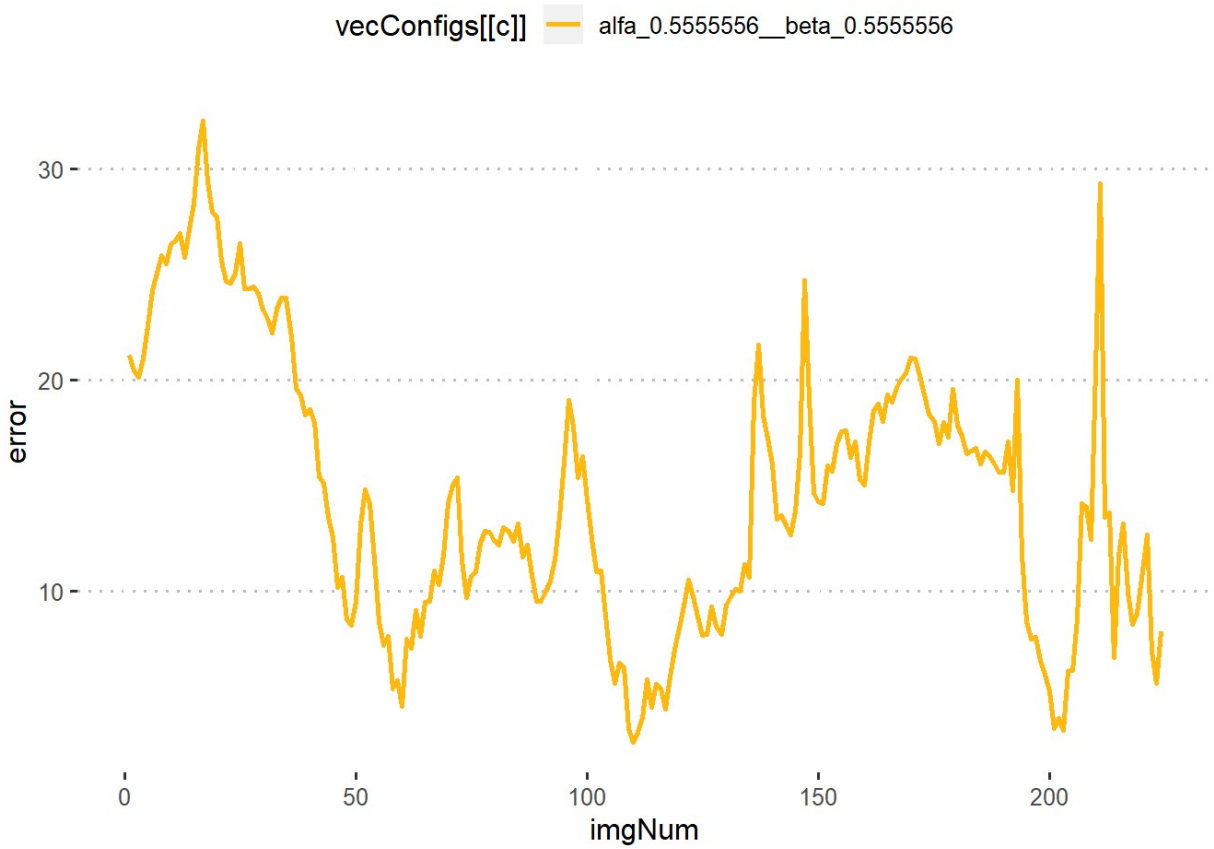

vecConfigs[[c]]
alfa_0.2222222__beta_1
30
20
10
error
0
50
100
150
200
imgNum

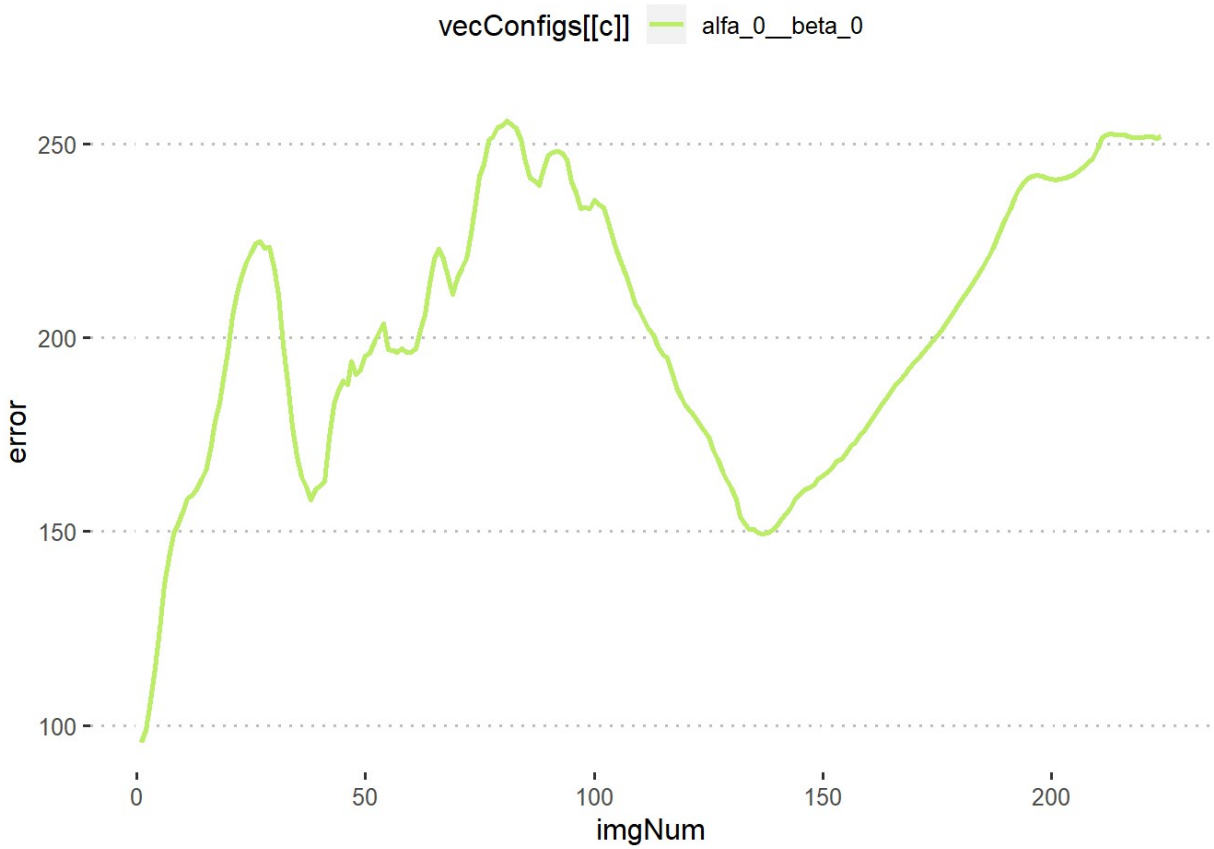

vecConfigs[[c]]
alfa_0.8888889__beta_1
30
20
10
error
0
50
100
150
200
imgNum

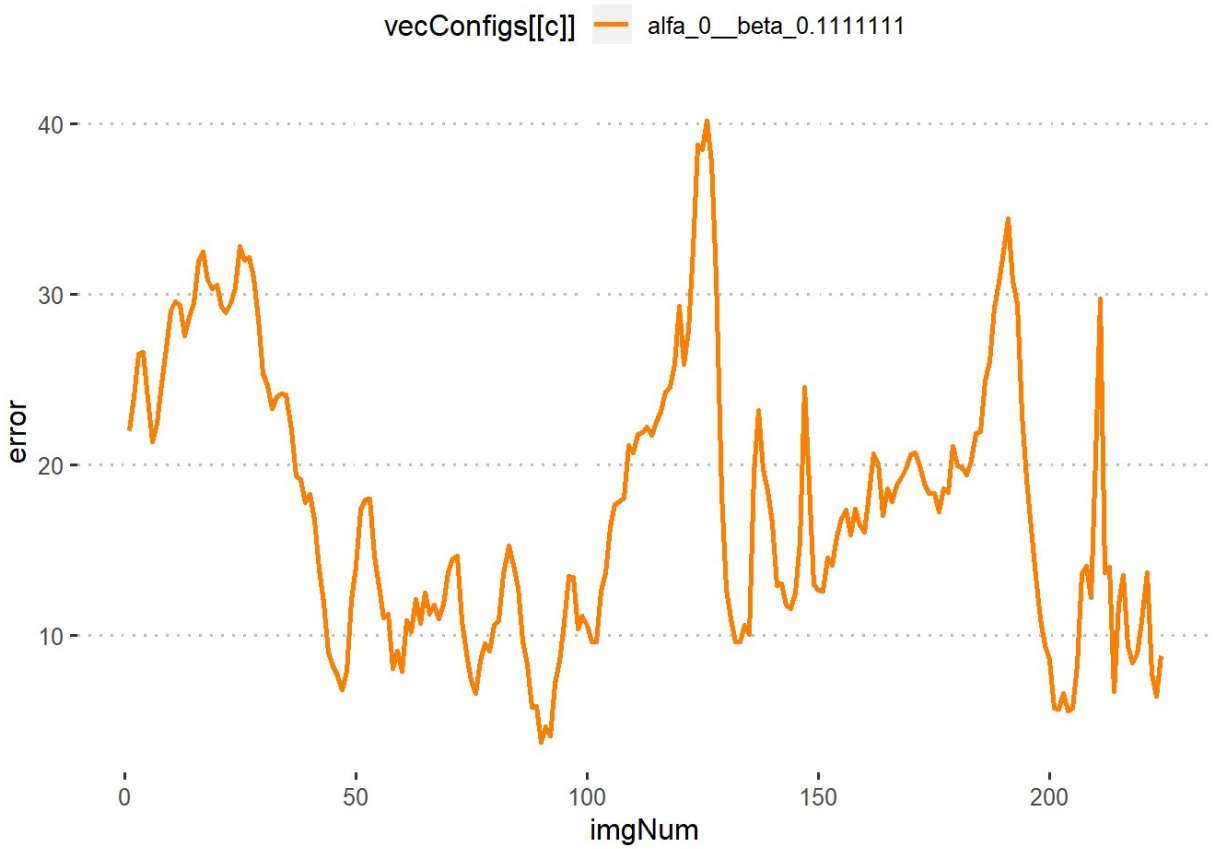

vecConfigs[[c]]
alfa_0.4444444__beta_0
300
250
200
150
100
error
0
50
100
150
200
imgNum

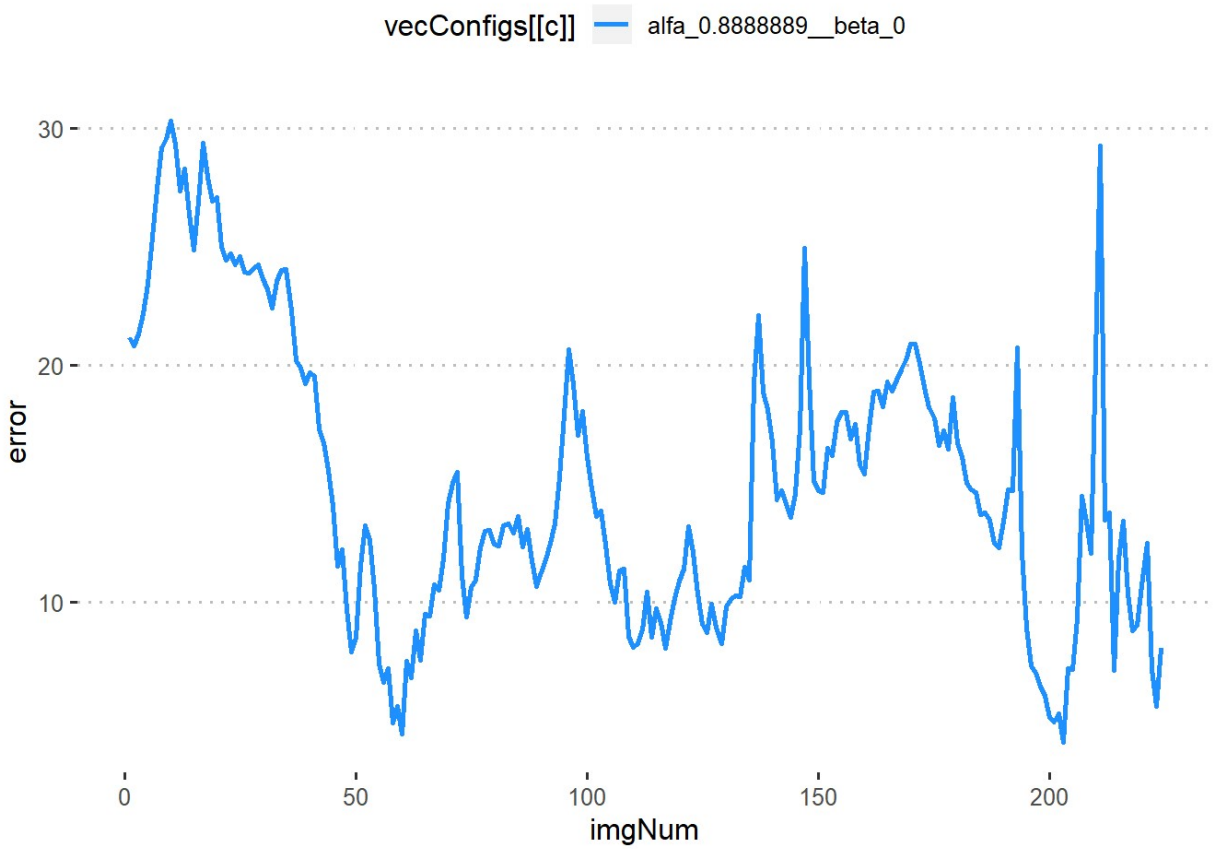

vecConfigs[[c]]
alfa_0.5555556__beta_0
30
20
10
error
0
50
100
150
200
imgNum

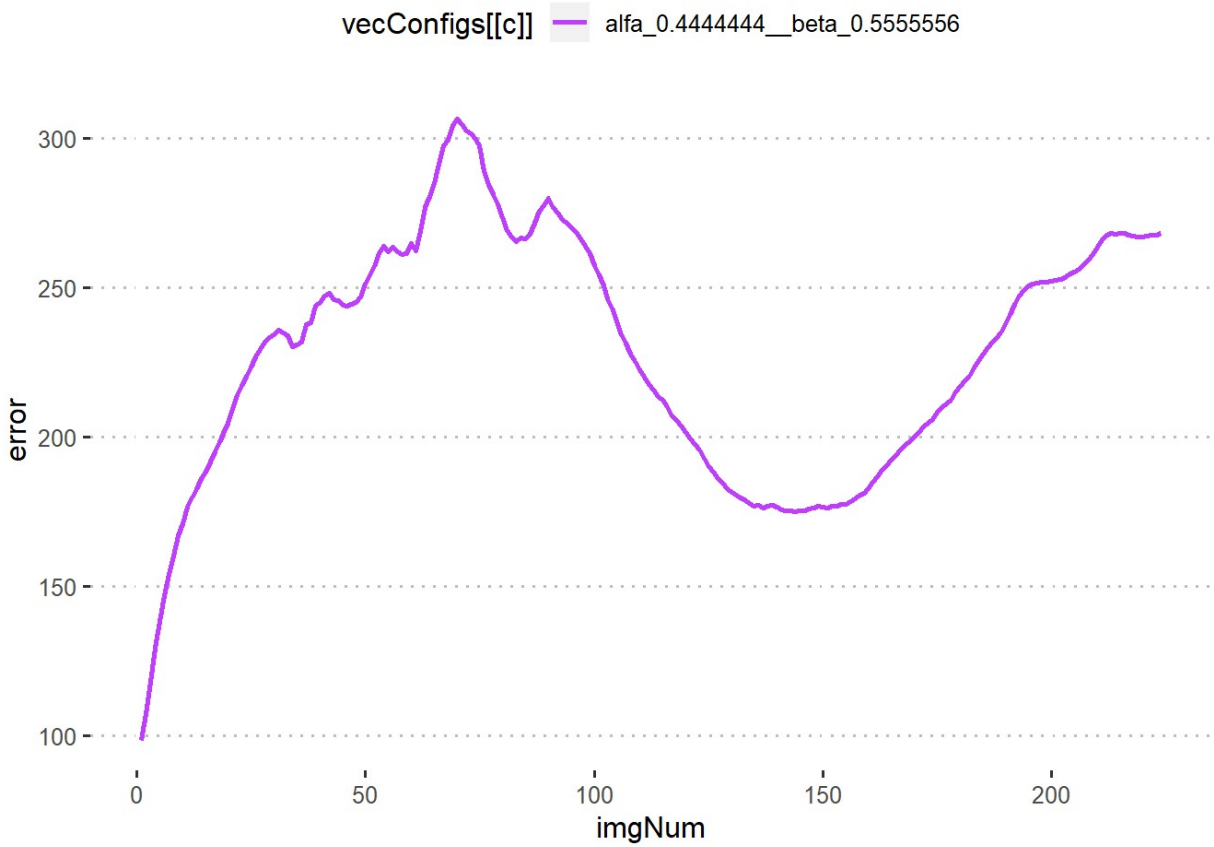

vecConfigs[[c]]
alfa_1__beta_0.8888889
error
30
20
10
0
50
100
150
200
imgNum

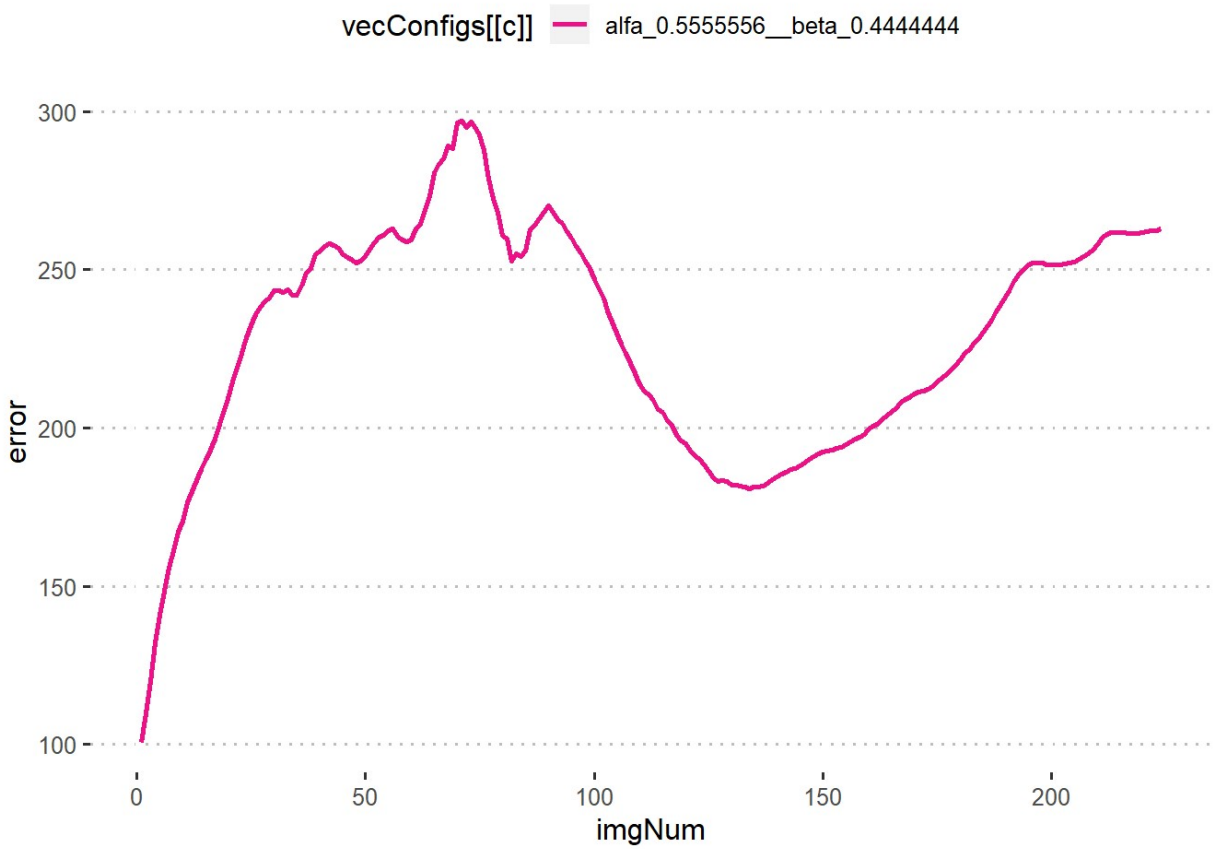

vecConfigs[[c]]
alfa_0.5555556__beta_0.5555556
error
30
20
10
0
50
100
150
200
imgNum

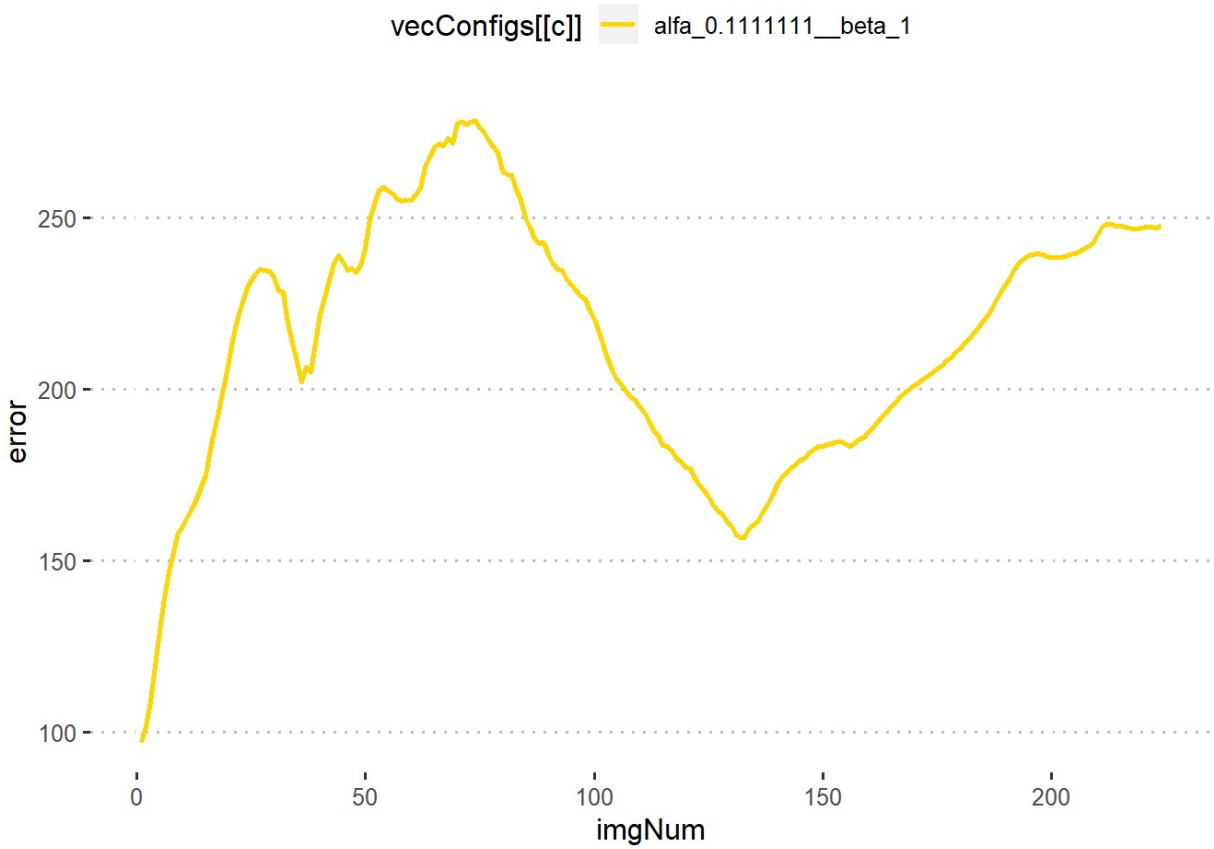

vecConfigs[[c]]
alfa_0__beta_0
250
200
150
100
error
0
50
100
150
200
imgNum

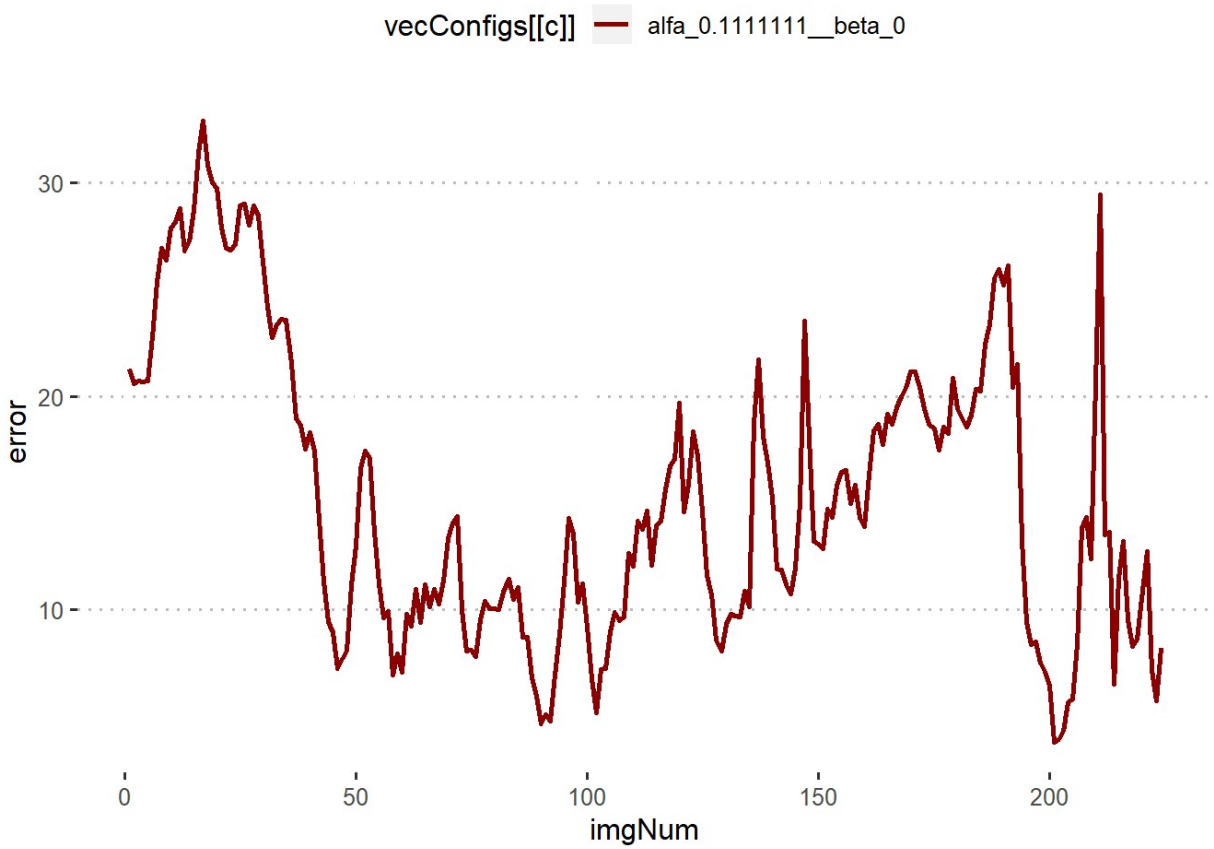

vecConfigs[[c]]
alfa_0__beta_0.1111111
40
30
20
10
error
0
50
100
150
200
imgNum

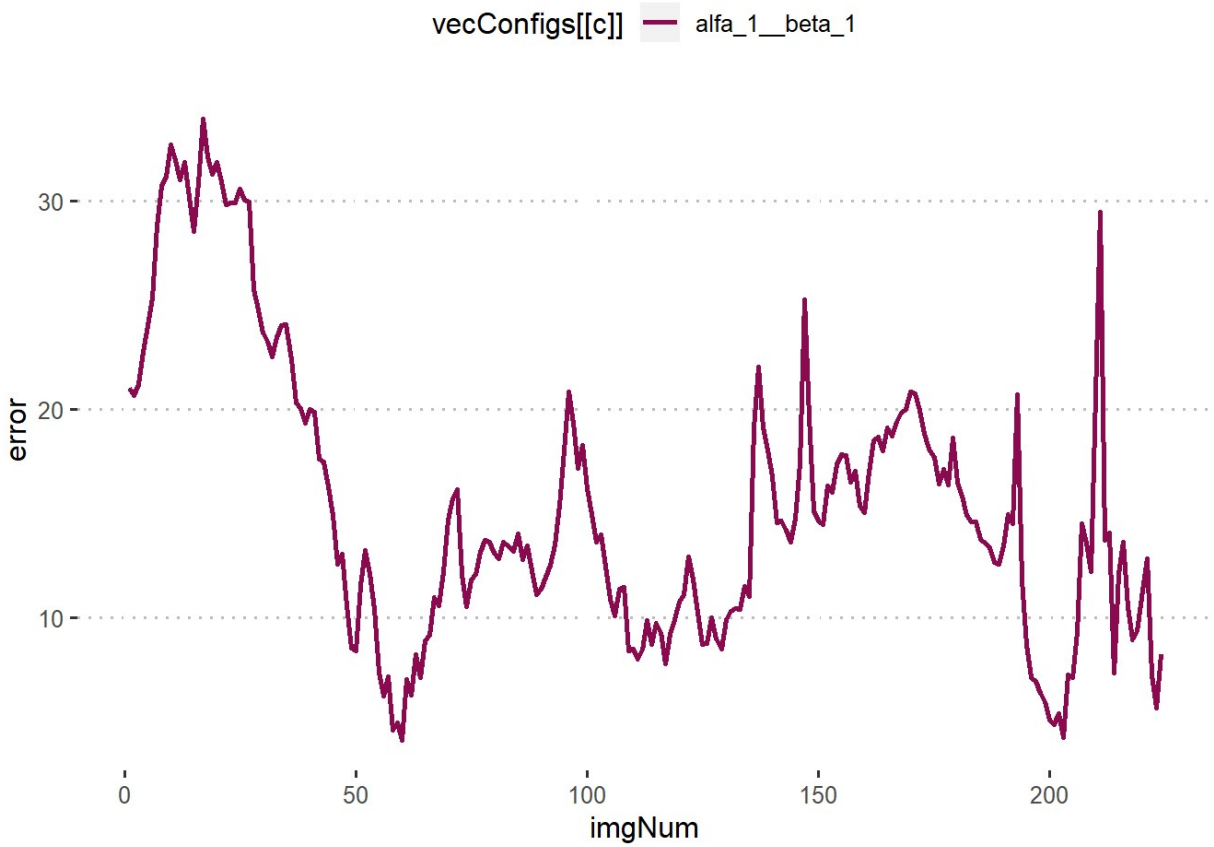

vecConfigs[[c]]
alfa_0.8888889__beta_0
30
20
10
error
0
50
100
150
200
imgNum

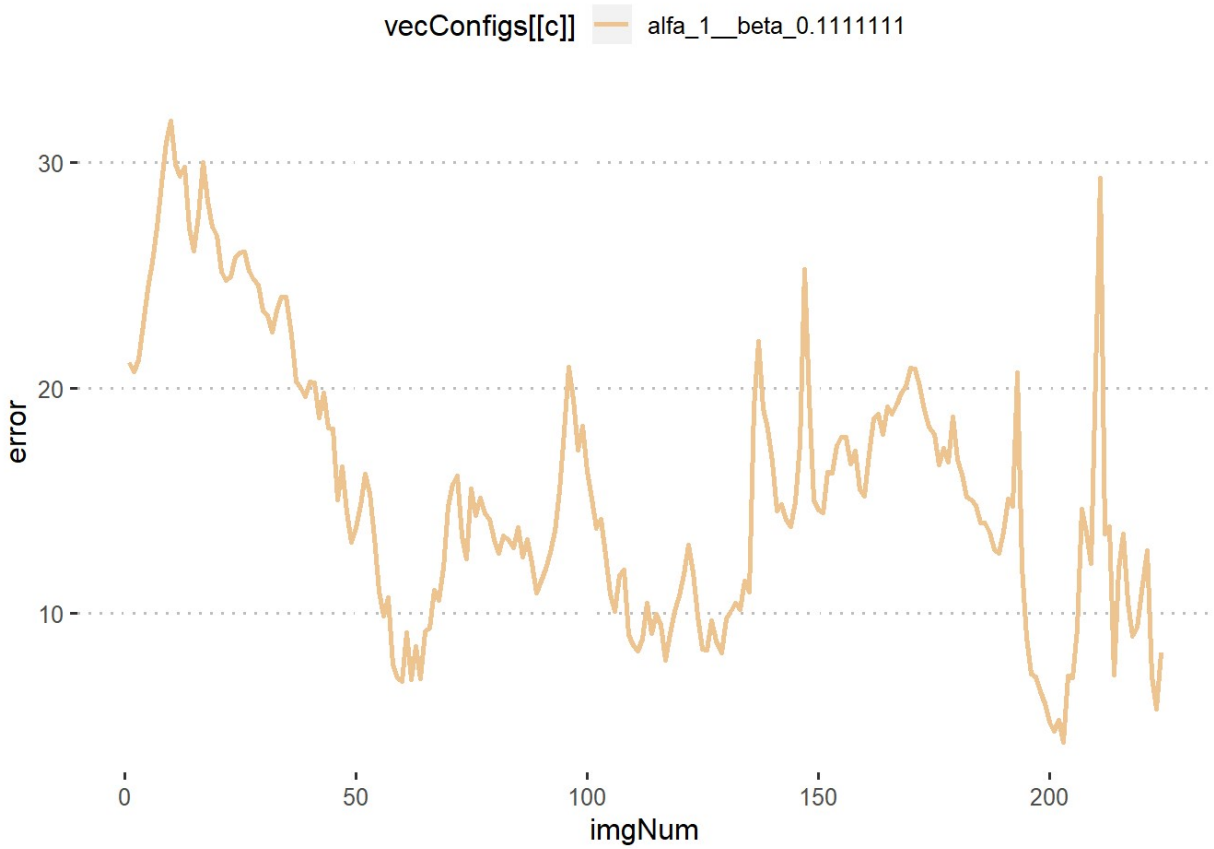

vecConfigs[[c]]
alfa_0.4444444__beta_0.5555556
300
250
200
150
100
error
0
50
100
150
200
imgNum

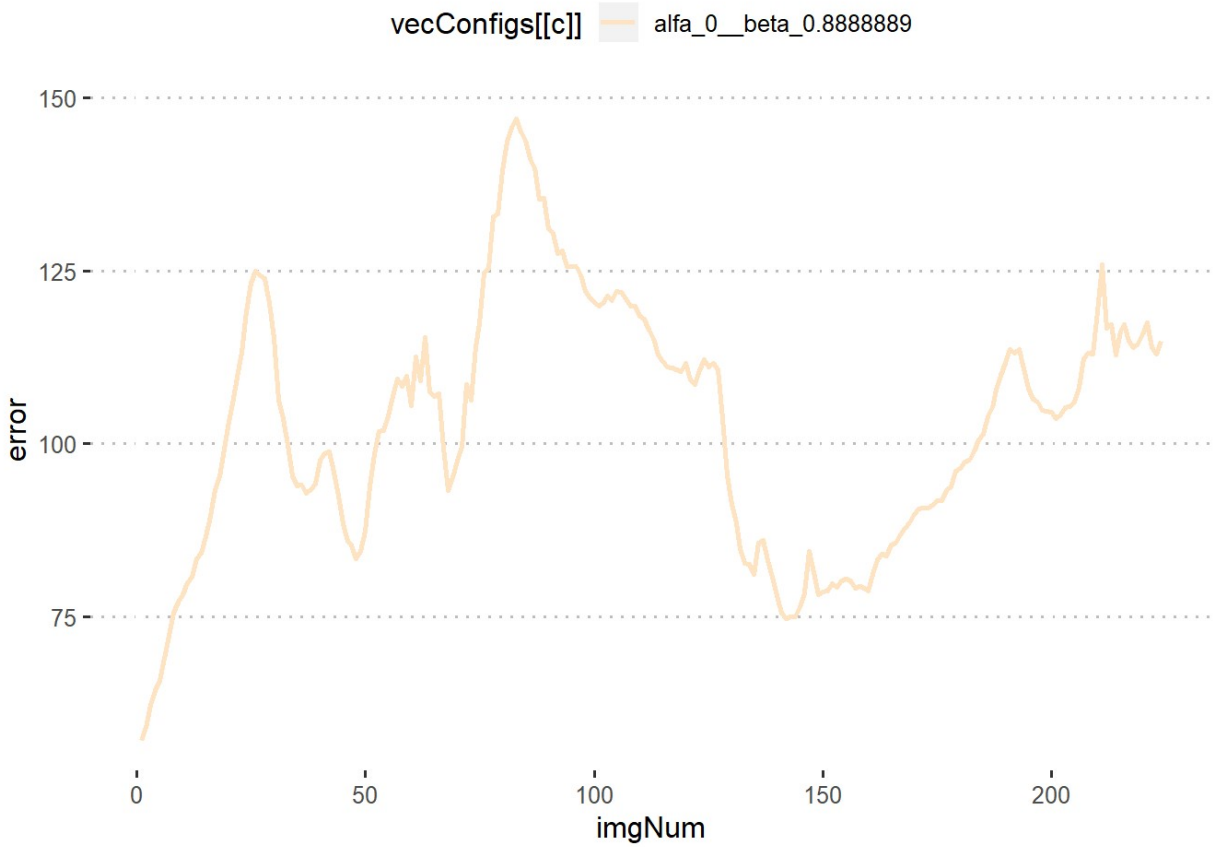

vecConfigs[[c]]
alfa_0.5555556__beta_0.4444444
300
250
200
150
100
error
0
50
100
150
200
imgNum

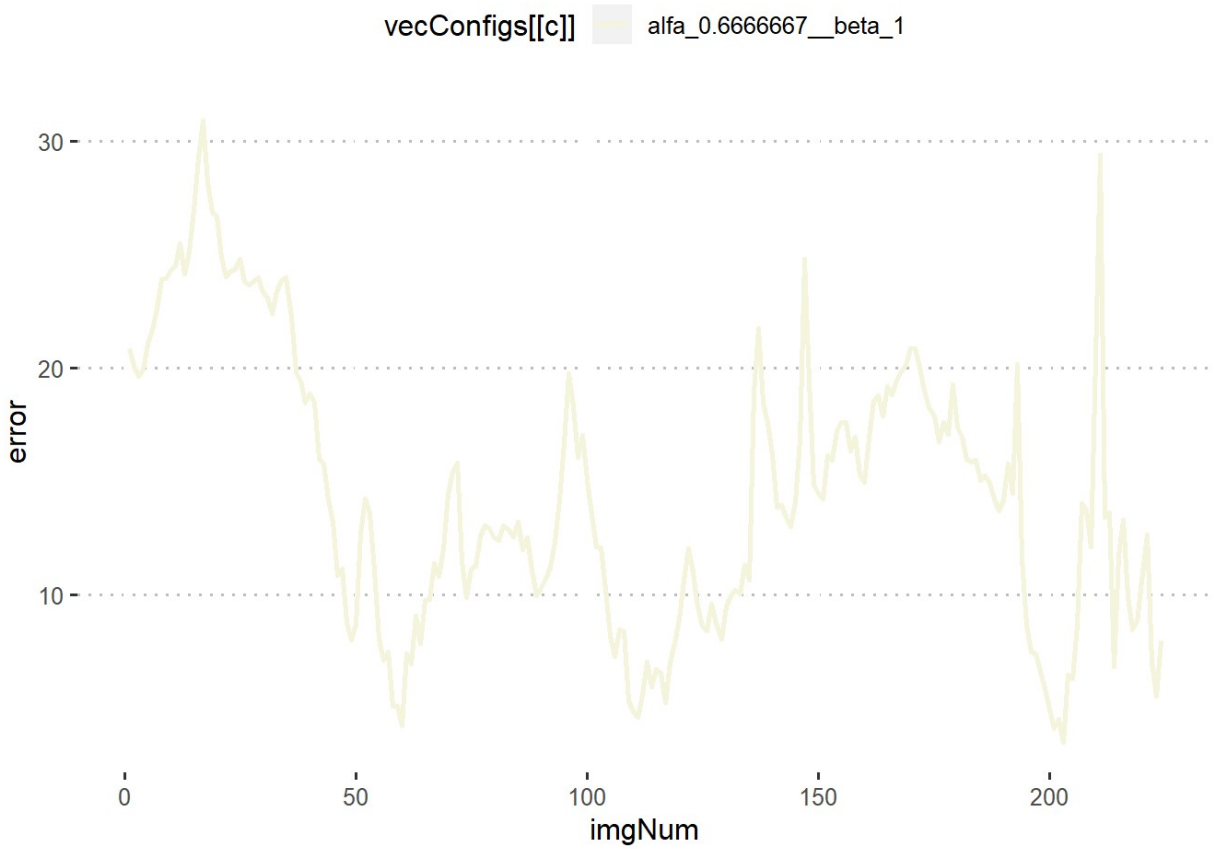

vecConfigs[[c]]
alfa_0.1111111__beta_1
250
200
150
100
error
0
50
100
150
200
imgNum

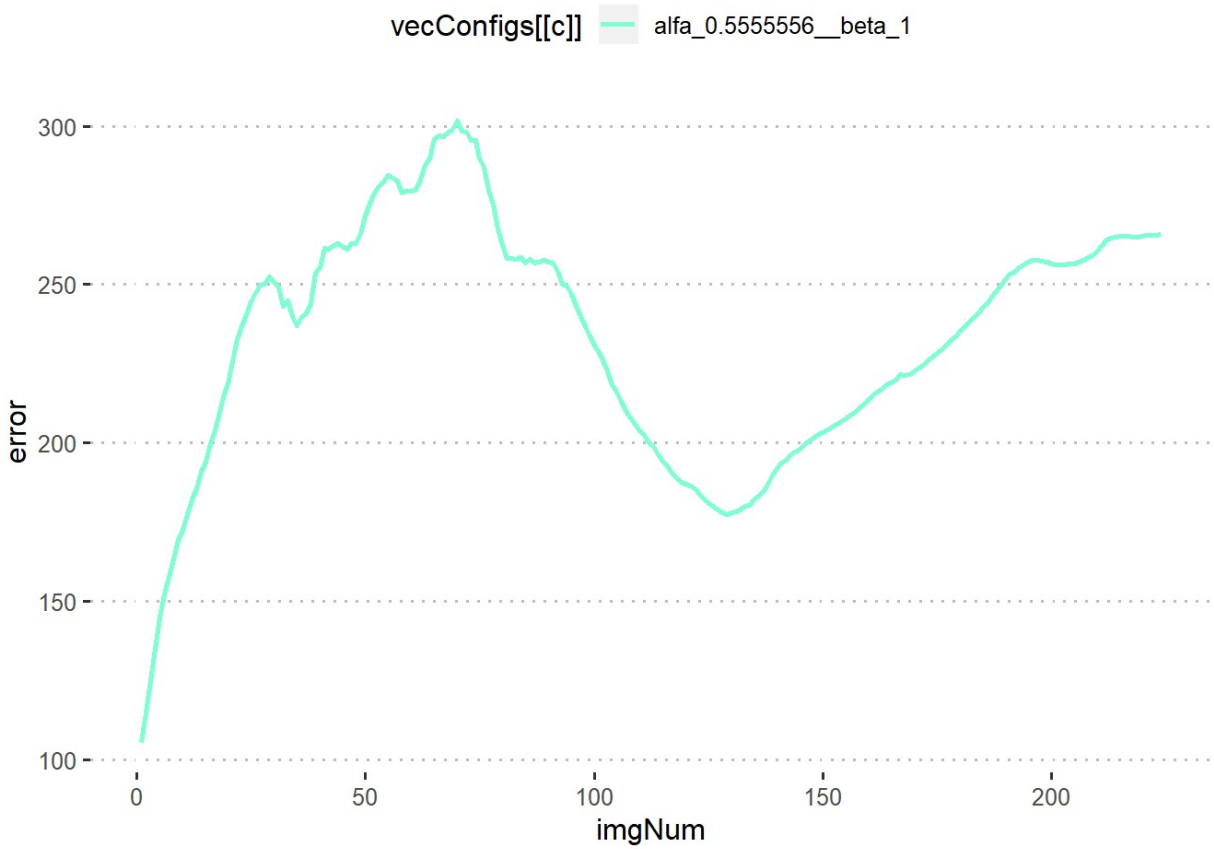

vecConfigs[[c]]
alfa_0.1111111__beta_0
30
20
10
error
0
50
100
150
200
imgNum

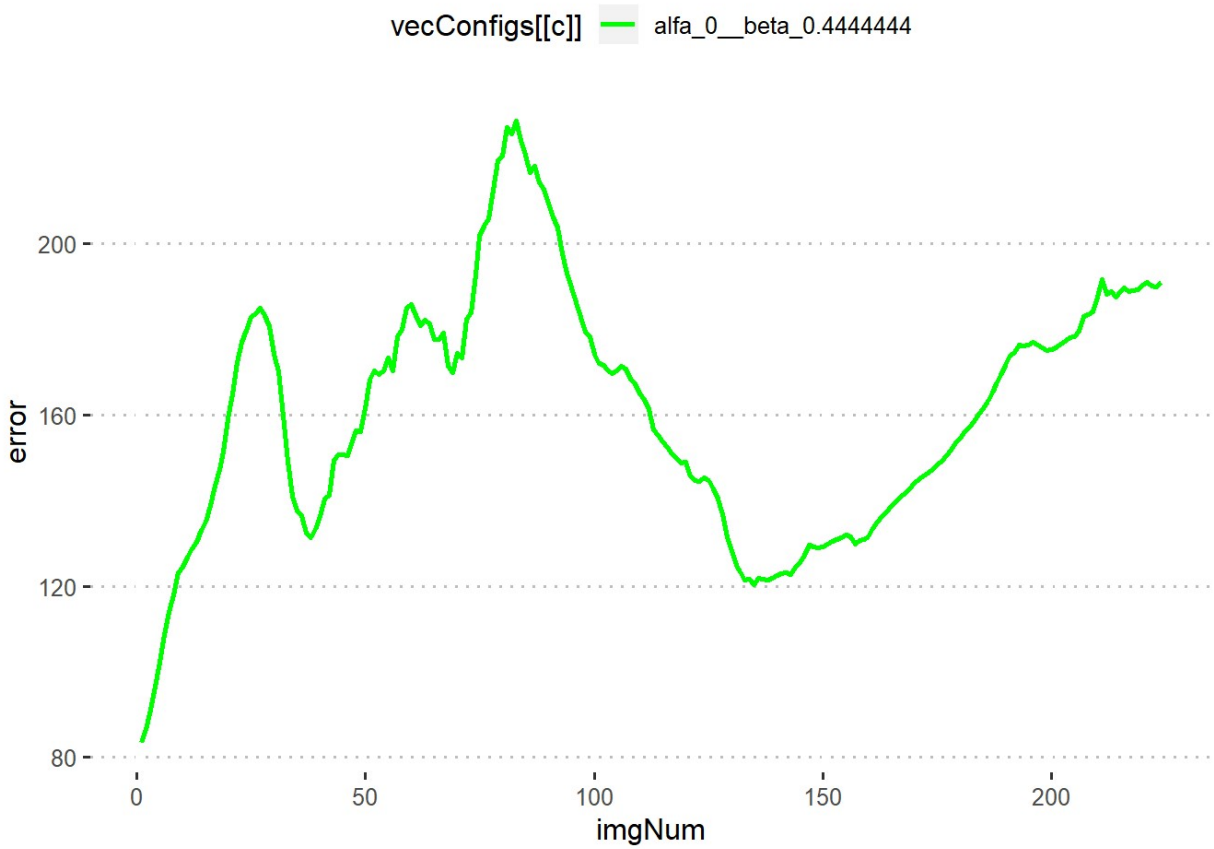

vecConfigs[[c]]
alfa_1__beta_1
30
20
10
error
0
50
100
150
200
imgNum

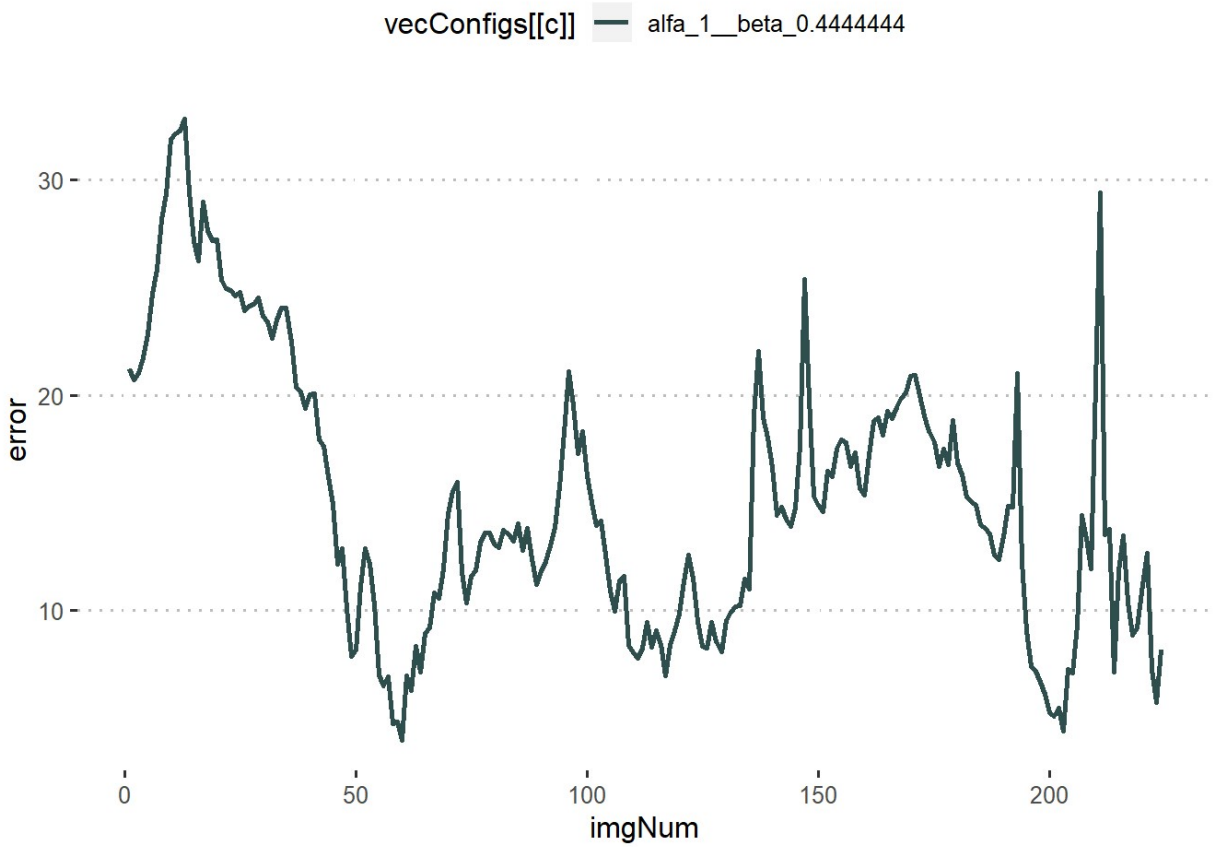

vecConfigs[[c]]
alfa_1__beta_0.1111111
30
20
10
error
0
50
100
150
200
imgNum

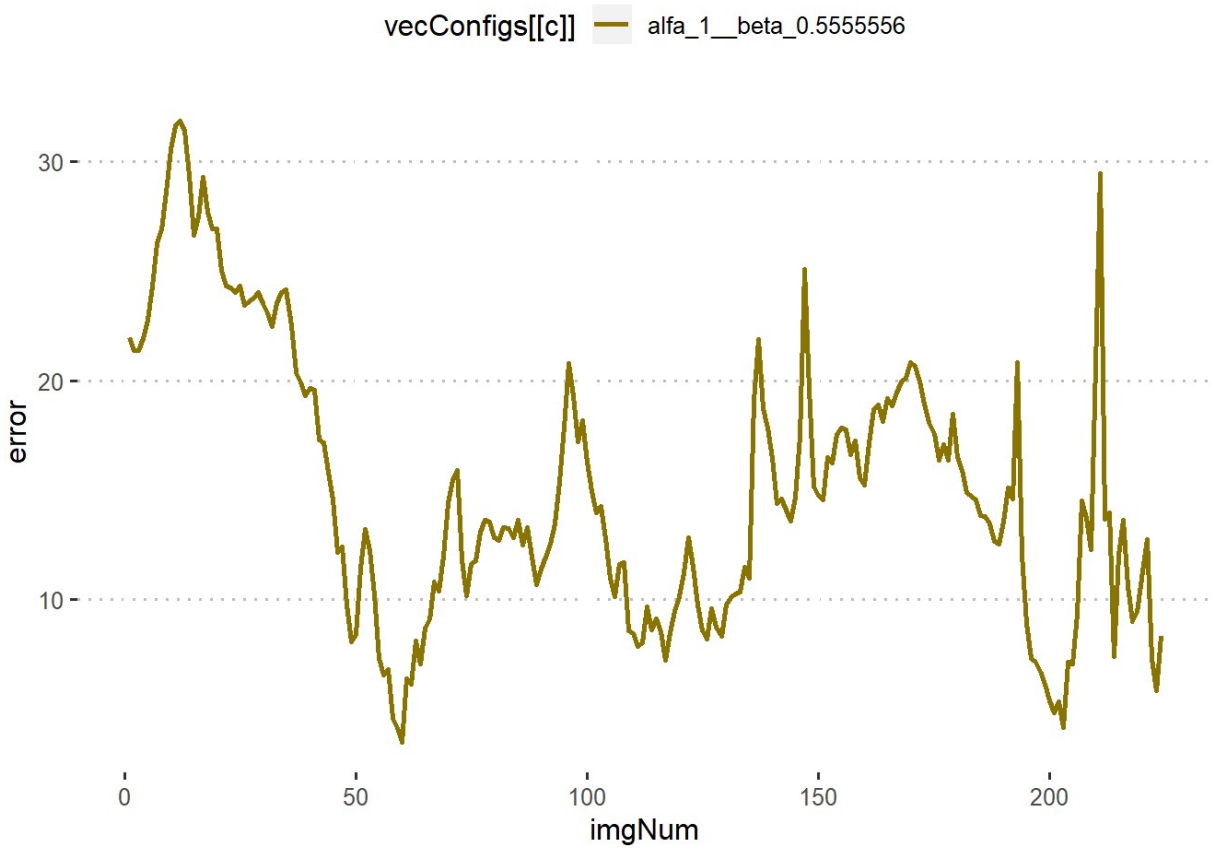

vecConfigs[[c]]
alfa_0__beta_0.8888889
150
125
100
75
error
0
50
100
150
200
imgNum

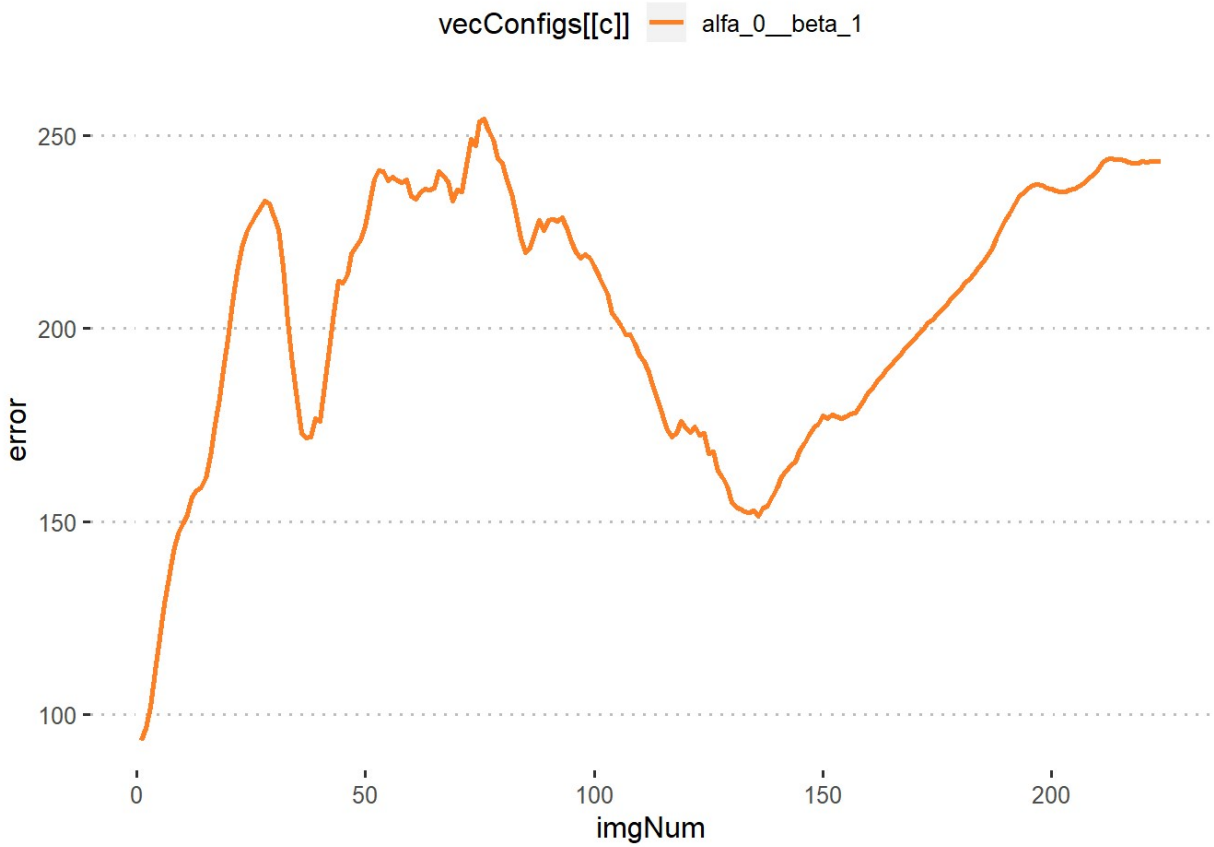

vecConfigs[[c]]
alfa_0.6666667__beta_1
30
20
10
error
0
50
100
150
200
imgNum

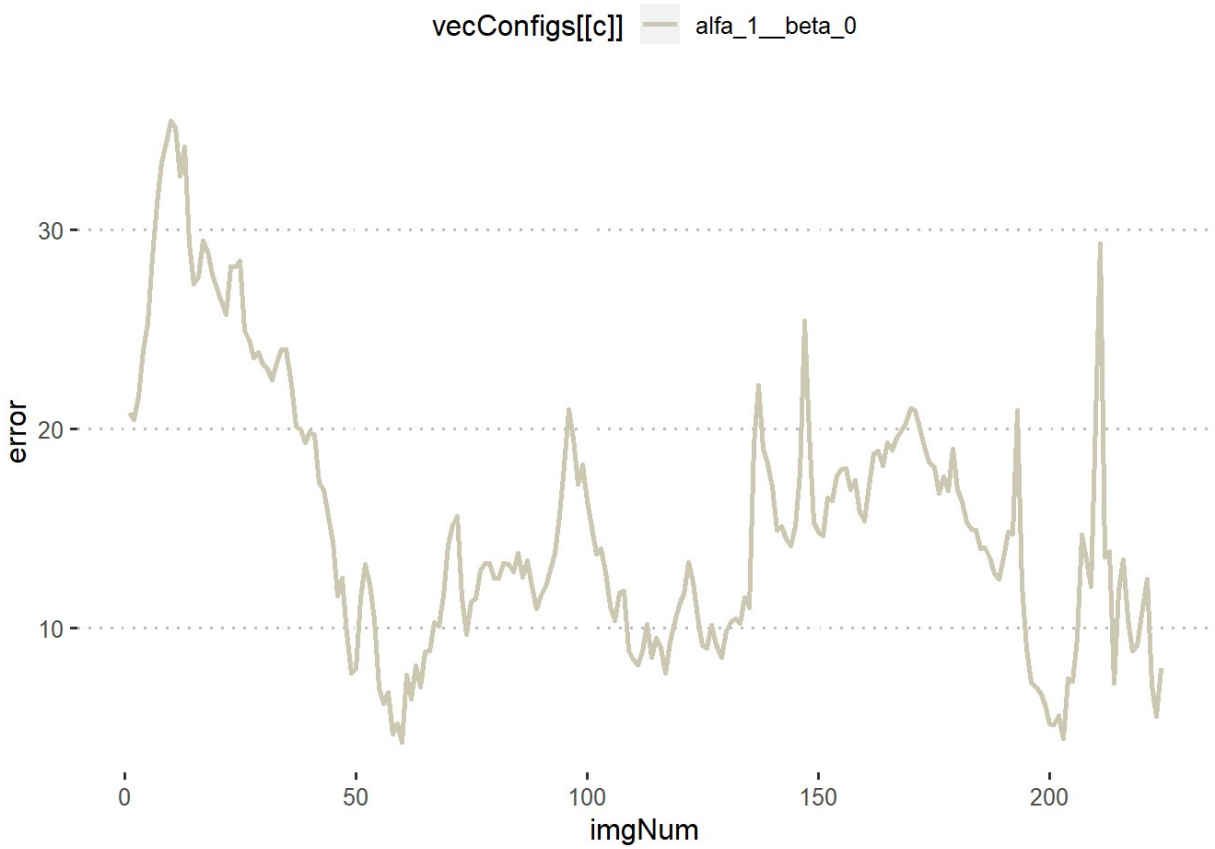

vecConfigs[[c]]
alfa_0.5555556__beta_1
300
250
200
150
100
error
0
50
100
150
200
imgNum


vecConfigs[[c]]
alfa_0__beta_0.4444444
200
160
120
80
error
0
50
100
150
200
imgNum


vecConfigs[[c]]
alfa_1__beta_0.4444444
30
20
10
error
0
50
100
150
200
imgNum


vecConfigs[[c]]
alfa_1__beta_0.5555556
30
20
10
error
0
50
100
150
200
imgNum


vecConfigs[[c]]
alfa_0__beta_1
250
200
150
100
error
0
50
100
150
200
imgNum



Ploting the error for all configurations:

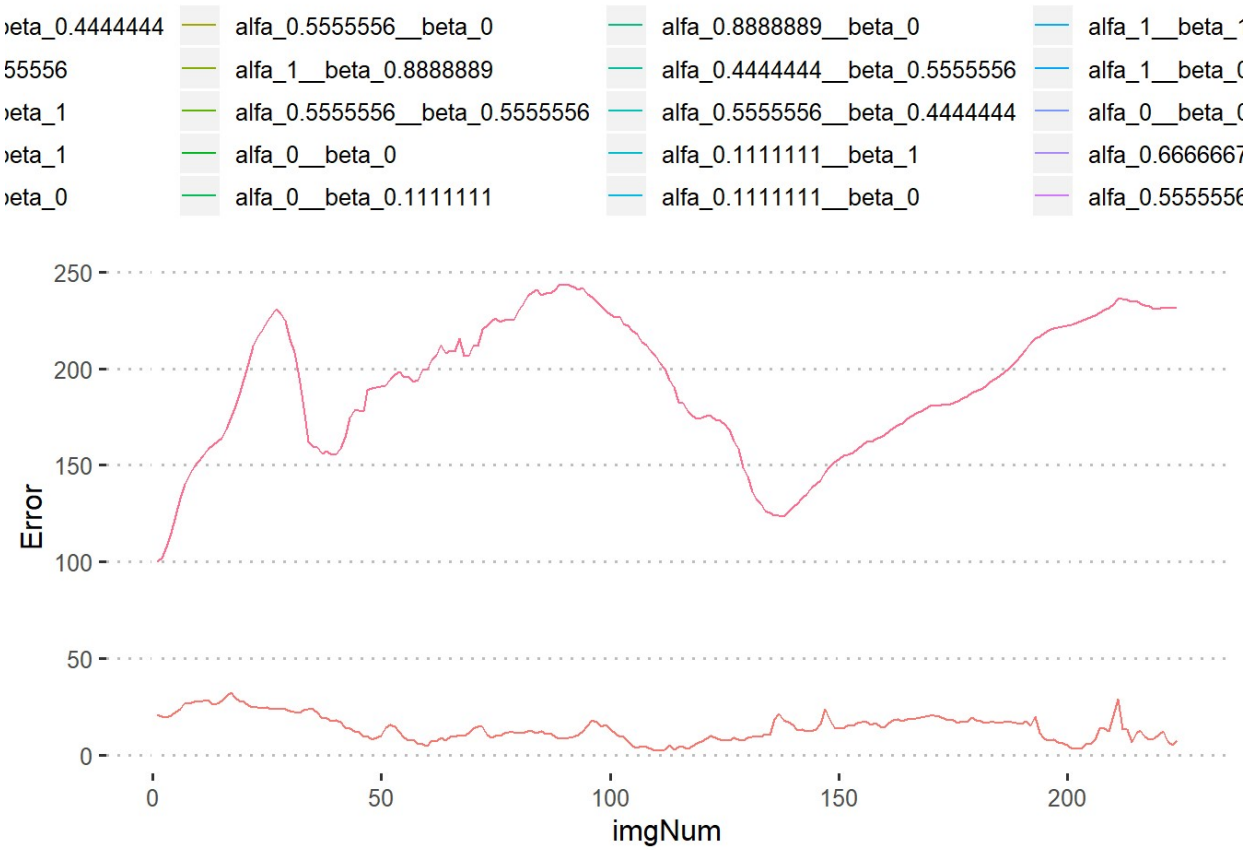
```
library(ggplot2)
library(ggpubr)
library(gridExtra )
library(reshape2)

df <- data.frame( imgNum = 1:imgCount , error1 = matrix(unlist(vecImgMeanError
[[1]]), nrow=length(vecImgMeanError[[1]]), byrow=T))
names(df)[2]<-paste(vecConfigs[1])

c <- 2
while (c <= configSize)
{
  df[[ vecConfigs[[c]] ]] <- matrix(unlist(vecImgMeanError[[2]]), nrow=length(vecIm
gMeanError[[2]]), byrow=T)
  c = c+1
}


df.long<-melt(df,id.vars="imgNum")
```

```
## Warning: attributes are not identical across measure variables; they will
## be dropped
```

```
head(df.long)
```

```
##   imgNum                       variable    value
## 1      1 alfa_0.4444444__beta_0.4444444 21.23227
## 2      2 alfa_0.4444444__beta_0.4444444 20.37498
## 3      3 alfa_0.4444444__beta_0.4444444 19.72728
## 4      4 alfa_0.4444444__beta_0.4444444 20.73931
## 5      5 alfa_0.4444444__beta_0.4444444 22.39243
## 6      6 alfa_0.4444444__beta_0.4444444 23.98365
```

```
names(df.long) <- c("imgNum", "config", "Error")

ggplot(df.long, aes(imgNum,Error,color=config)) + geom_line() + theme_pubclean()
```



Ploting the mean in the map:

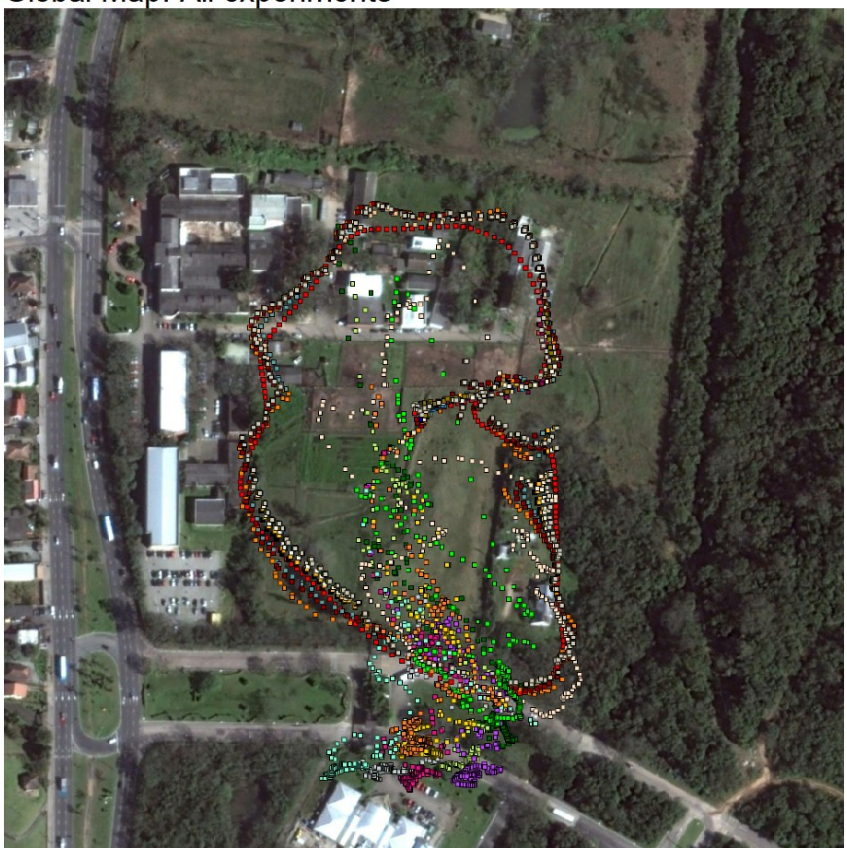
```
library(magick)
```

```
## Linking to ImageMagick 6.9.9.14
## Enabled features: cairo, freetype, fftw, ghostscript, lcms, pango, rsvg, webp
## Disabled features: fontconfig, x11
```

```
library(ggplot2)

globalMap <- image_read("UFRGS_VET_V3_11_08_2011.jpg")
scaleDiv <- 5

scaledGlobalMap <- image_scale(globalMap, 4800/scaleDiv)
img <- image_draw(scaledGlobalMap)

trajPos <- 1
gtSize = nrow(groundTruth)

while (trajPos <= gtSize)
  {
    symbols(groundTruth[trajPos,1]/scaleDiv, groundTruth[trajPos,2]/scaleDiv, squar
es = 5, add = TRUE, inches = FALSE, fg = "black", bg=vecColor[1])
    trajPos = trajPos+1
}

c <- 1
while (c <= configSize)
{
   trajPos <- 1
   while (trajPos <= gtSize)
     {
       symbols(vecImgMeanX[[c]][trajPos]/scaleDiv, vecImgMeanY[[c]][trajPos]/scaleD
iv, squares = 5, add = TRUE, inches = FALSE, fg = "black", bg=vecColor[c+1])

       trajPos = trajPos+1
   }
   c = c+1
}
```

```
myplot <- image_ggplot(img)
myplot + ggtitle("Global Map: All experiments")
```

Global Map: All experiments



Configuration x Ground Truth

```
library(magick)
library(ggplot2)

vecMaps <- vector(mode = "list", length = configSize)
globalMap <- image_read("UFRGS_VET_V3_11_08_2011.jpg")
scaleDiv <- 5

scaledGlobalMap <- image_scale(globalMap, 4800/scaleDiv)
img2 <- image_draw(scaledGlobalMap)

c <- 1
while (c <= configSize)
{
   #Drawing the Ground Truth
   trajPos <- 1
   gtSize = nrow(groundTruth)

   while (trajPos <= gtSize)
     {
       symbols(groundTruth[trajPos,1]/scaleDiv, groundTruth[trajPos,2]/scaleDiv, sq
uares = 5, add = TRUE, inches = FALSE, fg = "black", bg=vecColor[1])
       trajPos = trajPos+1
   }

   trajPos <- 1
   while (trajPos <= gtSize)
     {
       symbols(vecImgMeanX[[c]][trajPos]/scaleDiv, vecImgMeanY[[c]][trajPos]/scaleD
iv, squares = 5, add = TRUE, inches = FALSE, fg = "black", bg=vecColor[c+1])

       trajPos = trajPos+1
   }

   vecMaps[[c]] = img2

   c = c+1
   img2 <- image_draw(scaledGlobalMap)
}

#print(img2)
#myplot2 <- image_ggplot(img2)
#myplot2 + ggtitle("Global Map: All experiments")
```

```
c <- 1
while (c <= configSize)
{
   print(myplot <- image_ggplot(vecMaps[[c]]) + ggtitle(vecConfigs[c]))
   c = c+1
}
```

alfa_0.4444444__beta_0.4444444

alfa_0__beta_0.5555556

alfa_0.2222222__beta_1

alfa_0.8888889__beta_1

alfa_0.4444444__beta_0

alfa_0.5555556__beta_0

alfa_1__beta_0.8888889

alfa_0.5555556__beta_0.5555556

alfa_0__beta_0

alfa_0__beta_0.1111111

alfa_0.8888889__beta_0

alfa_0.4444444__beta_0.5555556

alfa_0.5555556__beta_0.4444444

alfa_0.1111111__beta_1

alfa_0.1111111__beta_0

alfa_1__beta_1

alfa_1__beta_0.1111111

alfa_0__beta_0.8888889

alfa_0.6666667__beta_1

alfa_0.5555556__beta_1

alfa_0__beta_0.4444444

alfa_1__beta_0.4444444

alfa_1__beta_0.5555556

alfa_0__beta_1

alfa_1__beta_0