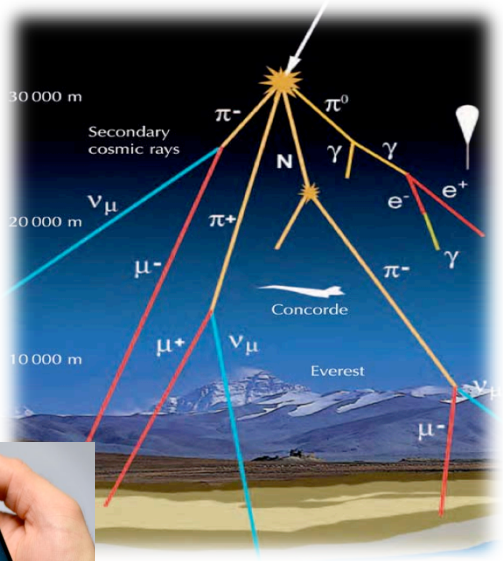
CREDO

THE QUEST FOR THE UNEXPECTED

Simple analysis of smartphones images

D. Góra for the CREDO Collaboration
Institute of Nucler Physics PAS, Cracow



Outline:
- Introduction: Cosmic rays, preshower effect
- How to select muon like events from smartphones images
- Summary

Visegrad Fund

Cosmic - Rays

Cosmic-rays: *energetic nuclei propagating in space at speeds close to light velocity.*
Discovered by Victor Hess in early 20th century.

> Power-law flux over many order of magnitude

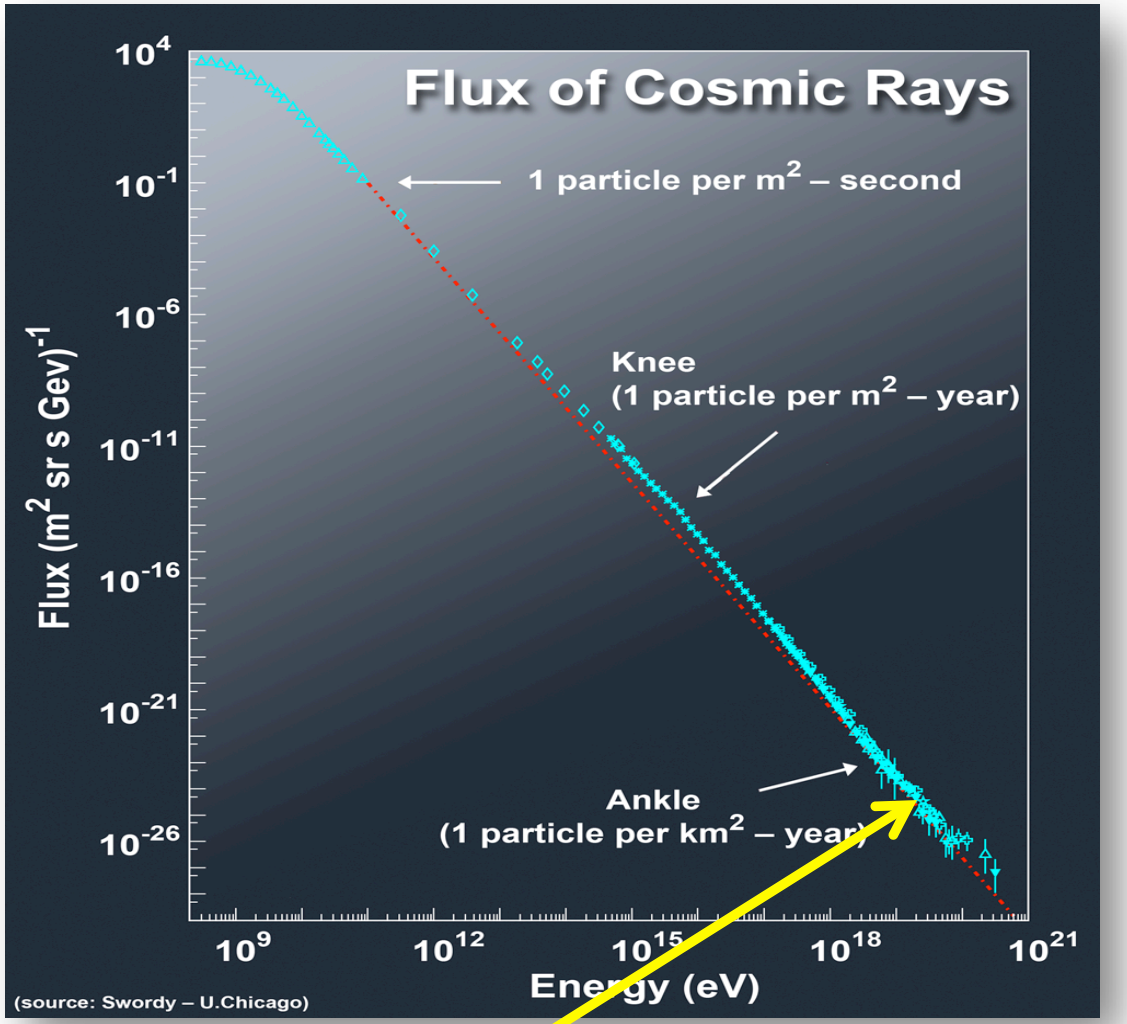
> Two features: knee and ankle

> end of spectrum ?

> Direct measured only below 10^{14}eV

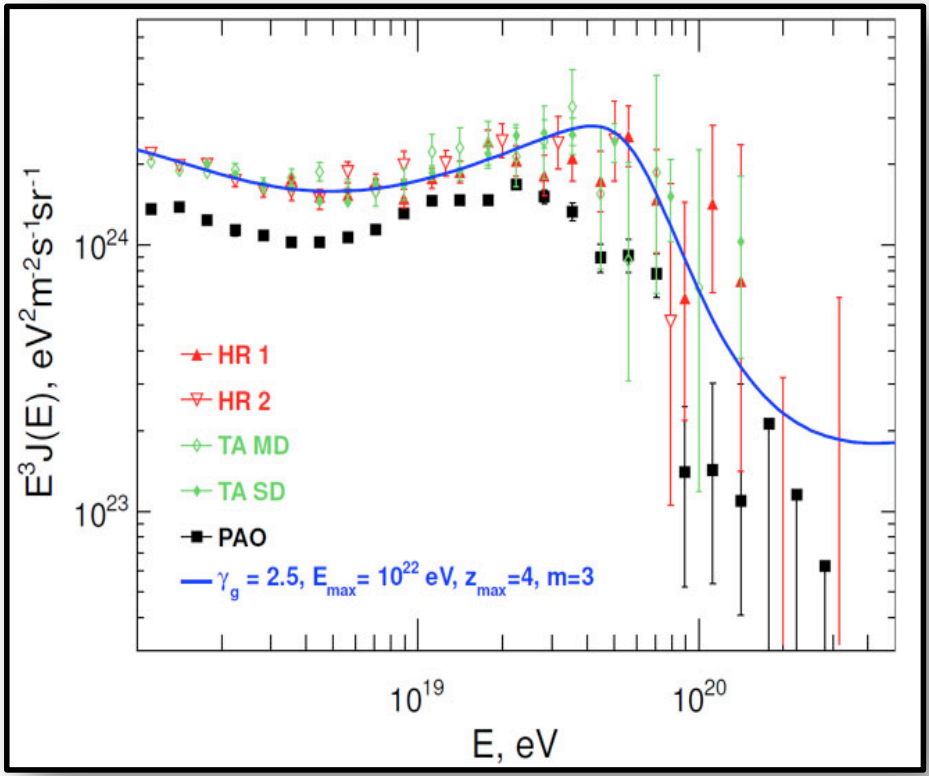
> Measurement of air showers at higher energies

Cosmic ray spectrum (credit: HAP / A. Chantelauze)



Ultra High Energy Cosmic Rays (UHCRs), E > 10^{17} eV

Cosmic-Ray mystery



Still open questions:

> What's their composition?

> Where do they come from?

⟶ *anisotropies weakly correlated to known possible sources: active galactic nuclei, gamma-ray burst,…*

> How do they reach such tremendous energies?

(past the GZK cut-off !)

Greisen-Zatsepin-Kuzmin (1966) – *cosmic ray absorption in Cosmic Mirowave Background CMB (1965):*

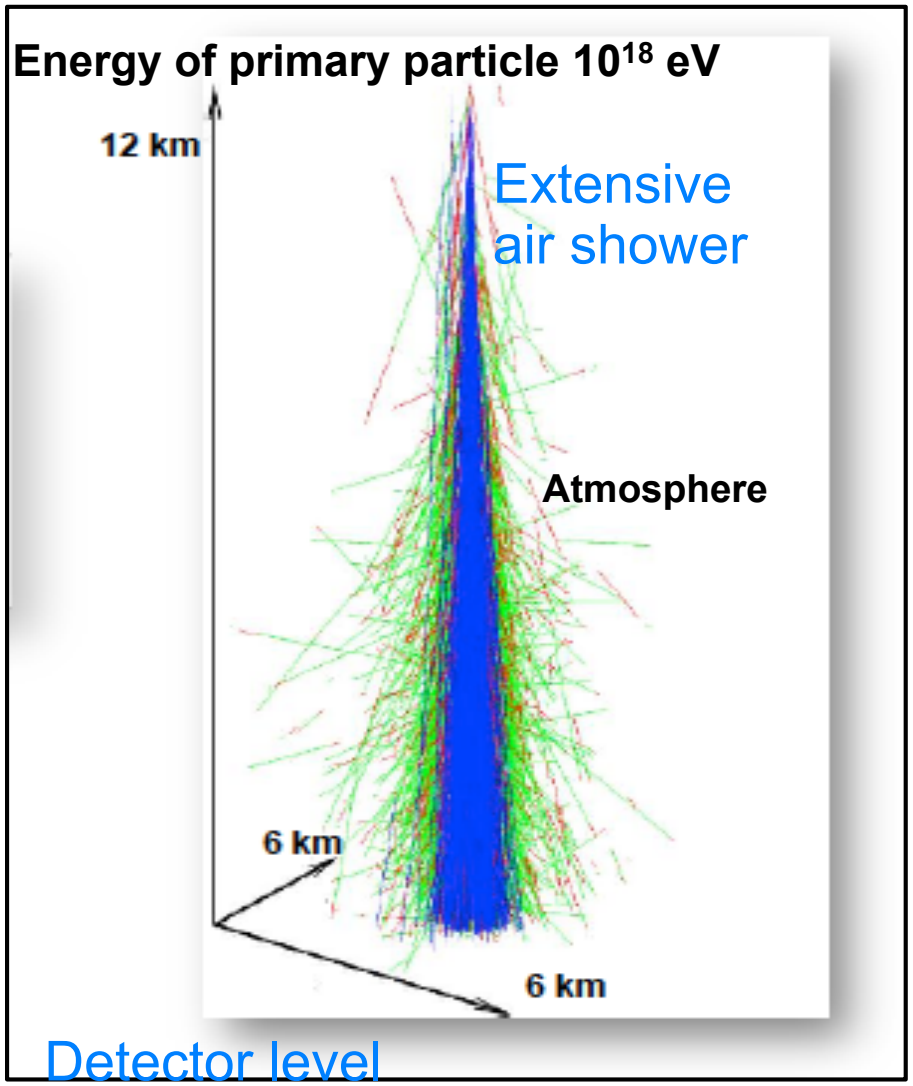
$$p + \gamma_{cmb} \longrightarrow \Delta(1232) \longrightarrow p + \pi^0 \quad \text{or} \quad n + \pi^+$$

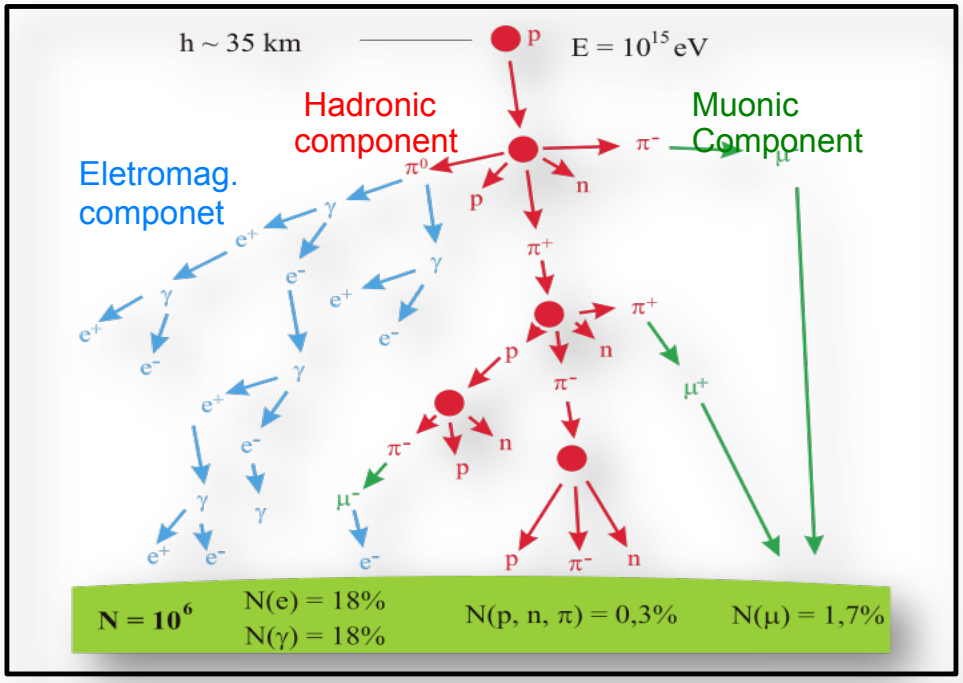
suppresion of cosmic ray flux above energy of 4 x10^{19} eV (GZK-cut-off), maximum source distance of 50-100 Mpc

Do photons of very high energies (above 10^{15} eV) exist ?

Extensive air shower

Extensive air shower – *collision of primary particle in air produce a shower of relativistic secondary particles*



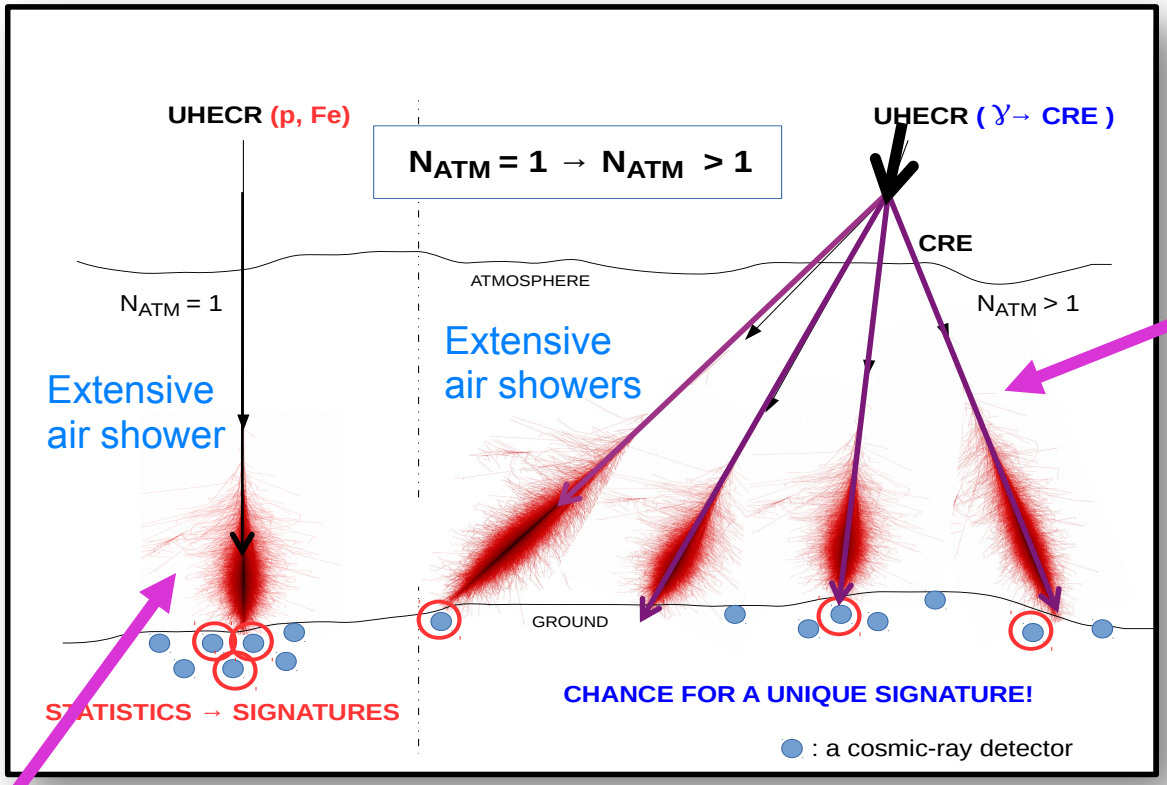


Muons are one of the EAS component

Atmospheric muons:

~ 1 particle/ cm^2 /minute
energy of muons ~$4*10^9$ eV (4 GeV)
at sea level

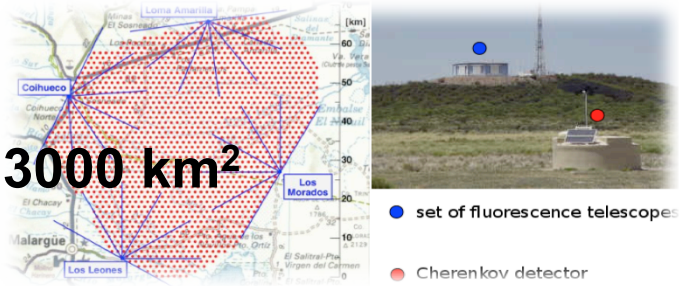
Motivation: looking for Cosmic Ray Ensambles (CRE)

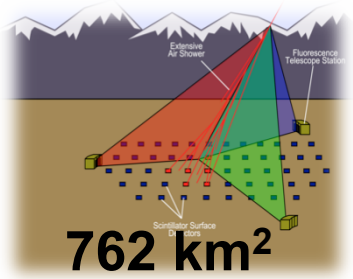


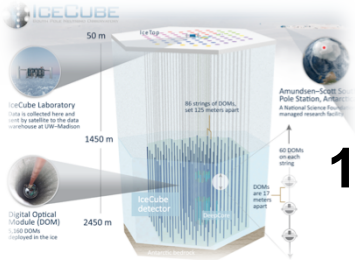
New strategy:
Looking for multiple air showers correlated in time

CREDO

Typical strategy: looking for ONE shower i.e.
Pierre Auger Observatory, Telescope Array, IceCube ....



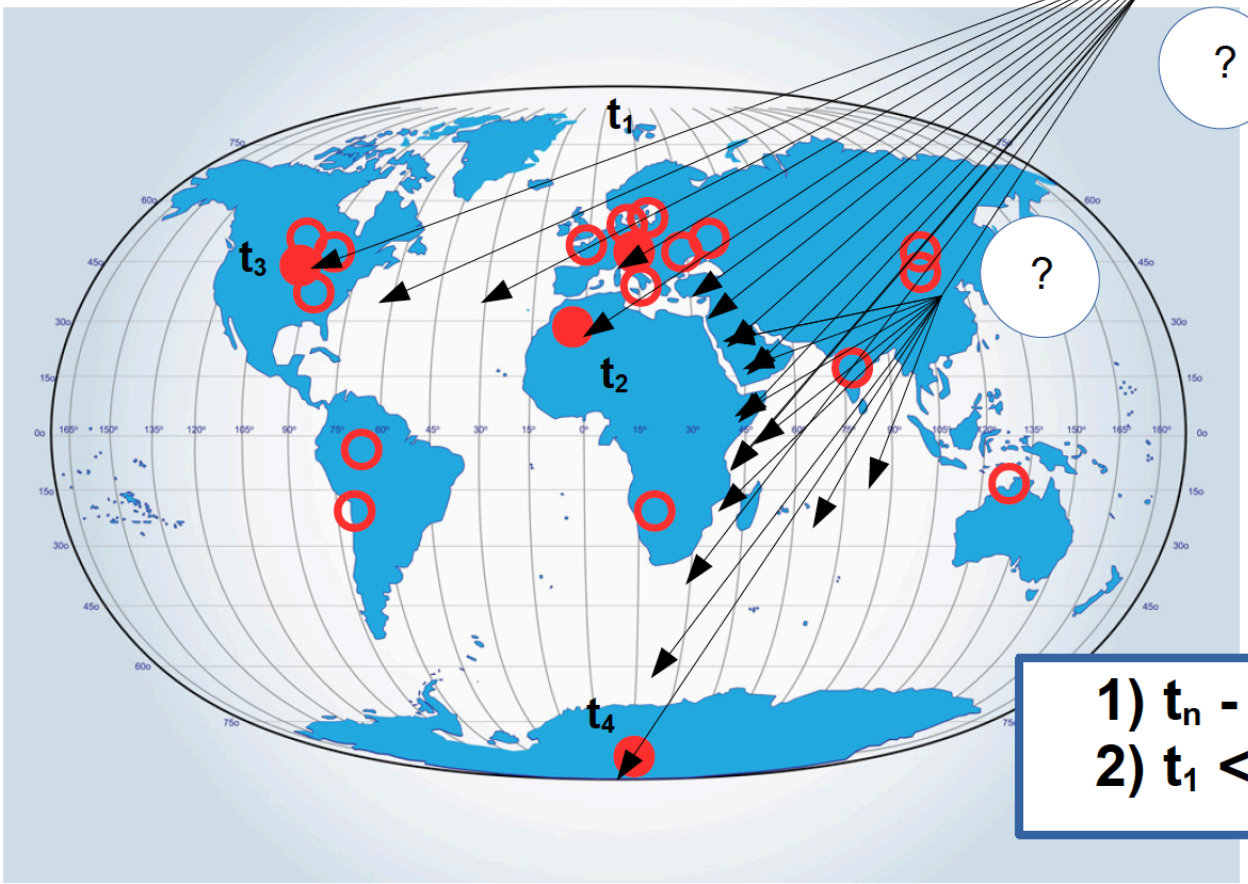




Cosmic-ray Extremaly Distributed Observatory

CREDO
THE QUEST FOR UNEXPECTED

Citizens strenghten trigger capabilities of the educational arrays with **smartphone networks**



Citizens browse the data looking for „improbable **time-space coincidences**

1) $t_n - t_1 < \sim 1\ \mu s$
2) $t_1 < \ldots < t_n$,

→ **indirect search for New Physics manifestations!**
→ **verification of „classic" QED predictions (preshower @ Sun)**

Mobile application

Smartphone application developed by CREDO collaboration

http://credo.science
https://play.google.com/store/
apps/details?id=
science.credo.credomobiledetektor

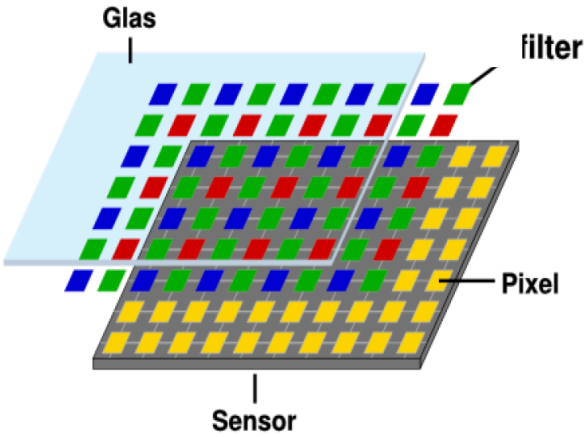
CREDO
THE QUEST FOR UNEXPECTED

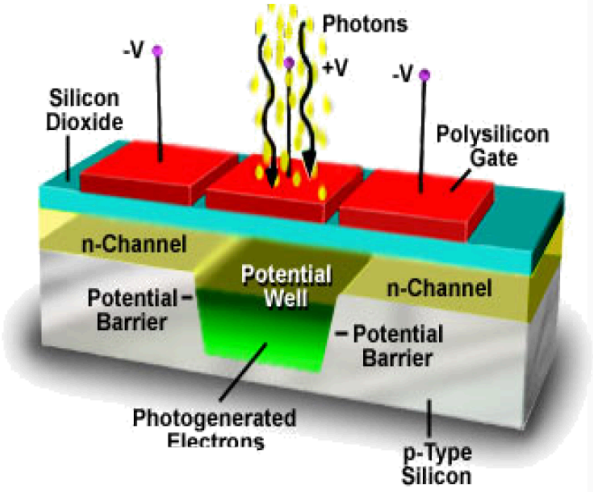
CREDO detector

Code of application is public on GitHub

Level 2:
data acquisition

Charged-coupled devices (CCD)





Smartphones have a detector of typical 4 x 4 mm²
→ can detect cosmic rays
D. Groom, Cosmic rays and other nonsense in astro-nomical CCD imagers, Experimental Astronomy 14 (1) (2002) 45–55.

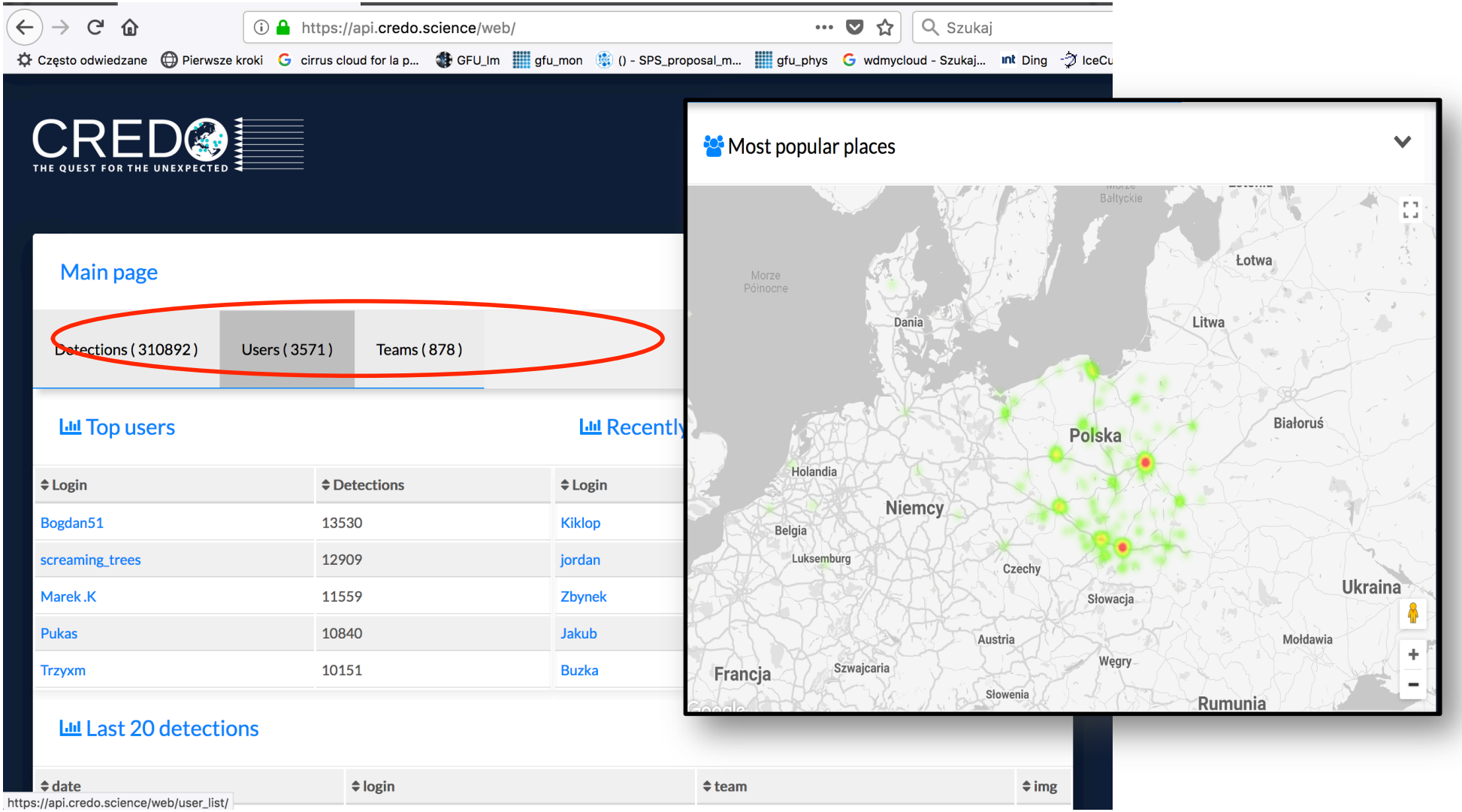
→ large number required (10^6)
(astro-ph 1410.2895, astro-ph 1505.04777)

DECO
Distributed Electronic Cosmic-ray Observatory

CRAYFIS

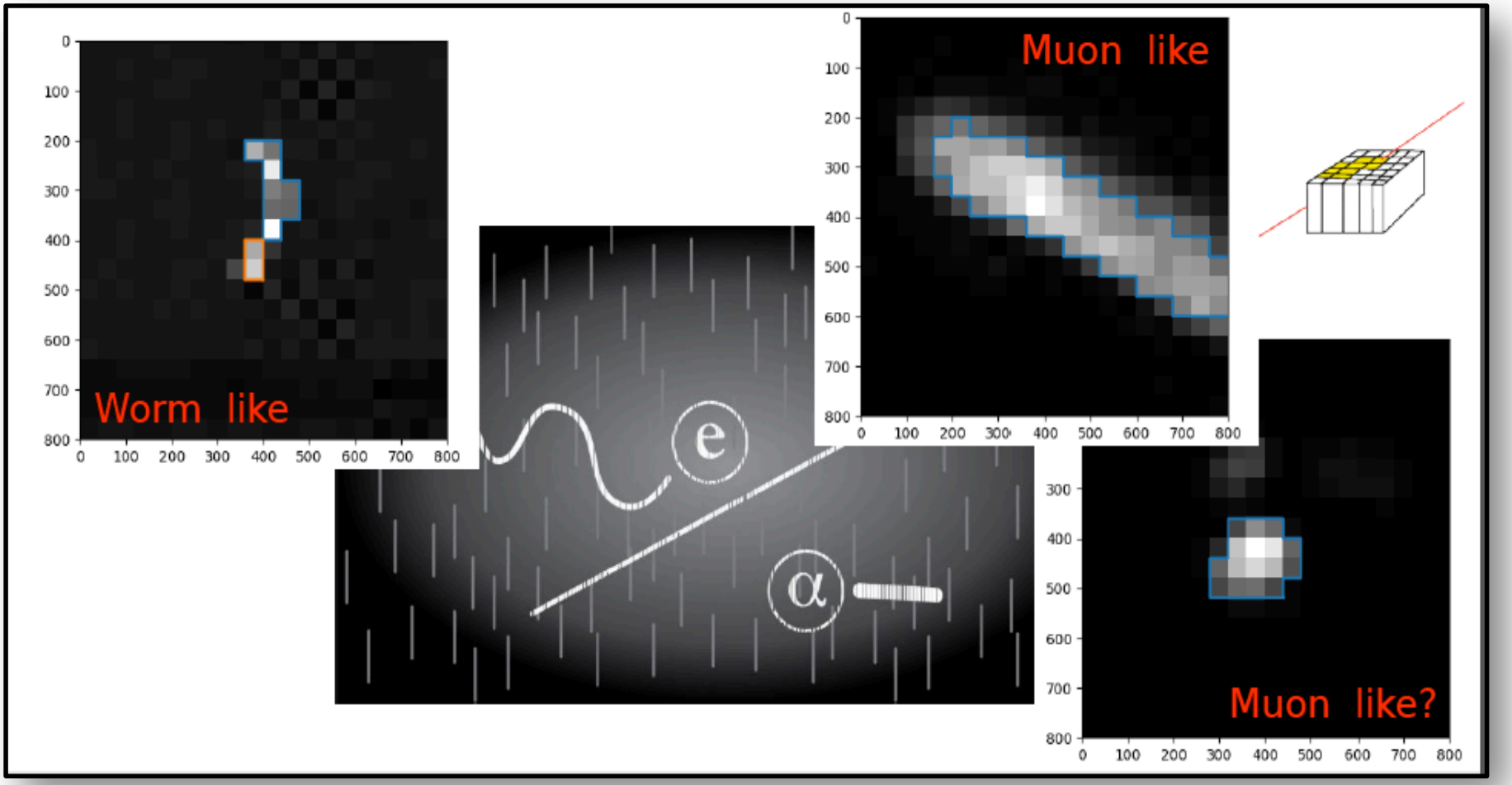
Michael Unger, Glennys Farrar

api.credo.science



Typical classes of events from smartphones

…. from api.credo.science



How to extract signal-like events (muon like) ?

Method: the simple filter to extract muon-like events

1) Convertion the RGB pixel values to a single gray-scale amplitude

compute luminance of an RGB image with formula:

Y = 0.2125 R + 0.7154 G + 0.0721 B

http://scikit-image.org/docs/dev/api/skimage.color.html#skimage.color.rgb2grey

2) Calculation of a contour using the “marching squares” algorithm

(implemented in python module: *http://scikit-image.org/docs/stable/user_guide.html*)

to delimit the pattern of pixels that detected ionization above a particular threshold.

3) Calculation of several metrics (observables) for each event:

- the total luminance above threshold (the SIZE parameter)
- the length and width parameter from Principal Component Analysis (PCA)

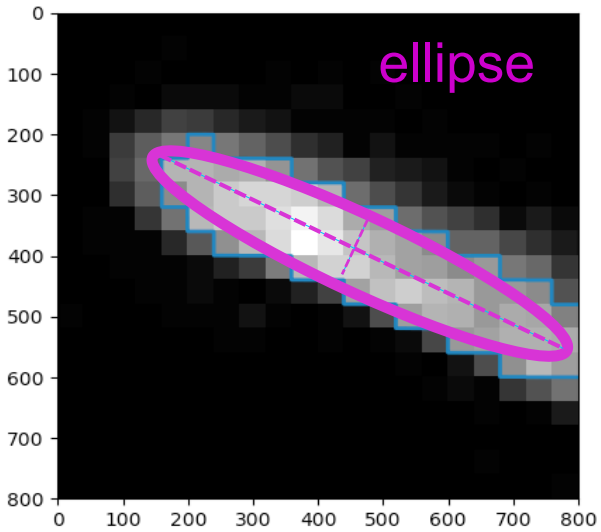
https://en.wikipedia.org/wiki/Principal_component_analysis[1]

Discrimination parameters:

LENGTH/WIDTH: *the ratio of major to minor axis of ellipse*

SIZE: *sum of illuminace in all pixels above certain defined threshold*

Listing of analasis script was attached to this presentation.



Public engagement as a scientific tool

The match: “IFJ” vs. “Team Rzezawa”

Discipline: **catching secondary cosmic ray particles with mobile devices with CREDO Detector**

When? 16.11.2017, 11:00 – 12:00

Where? IFJ PAN, Gimanzjum Publiczne Rzezawa, Poland, world

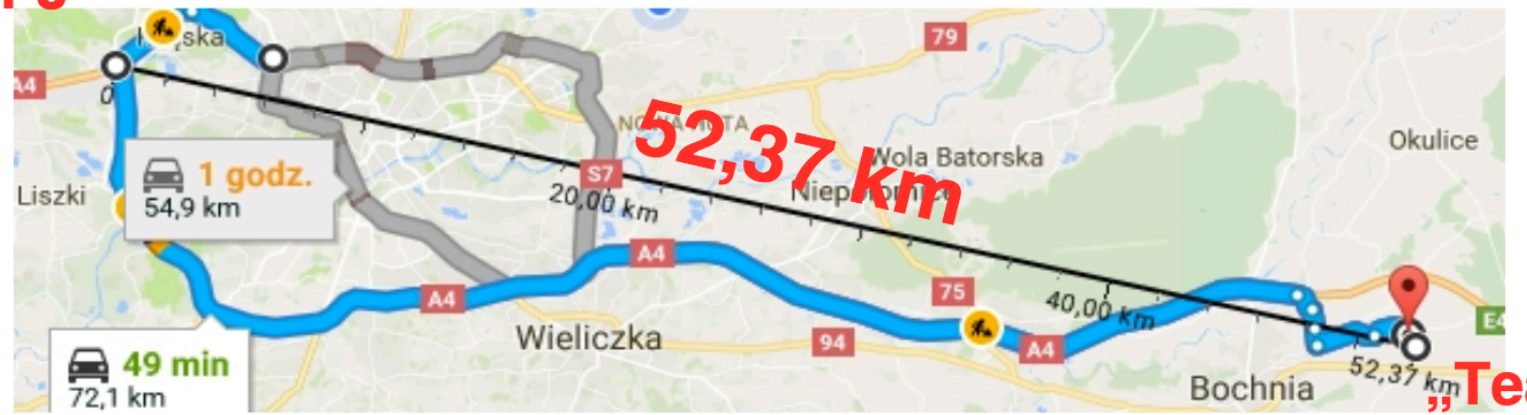
Transmitted live: CREDO YouTube Channel

Number of registered players: 32:30

Number of catched particles: 12:4

Final score: -135 do -257

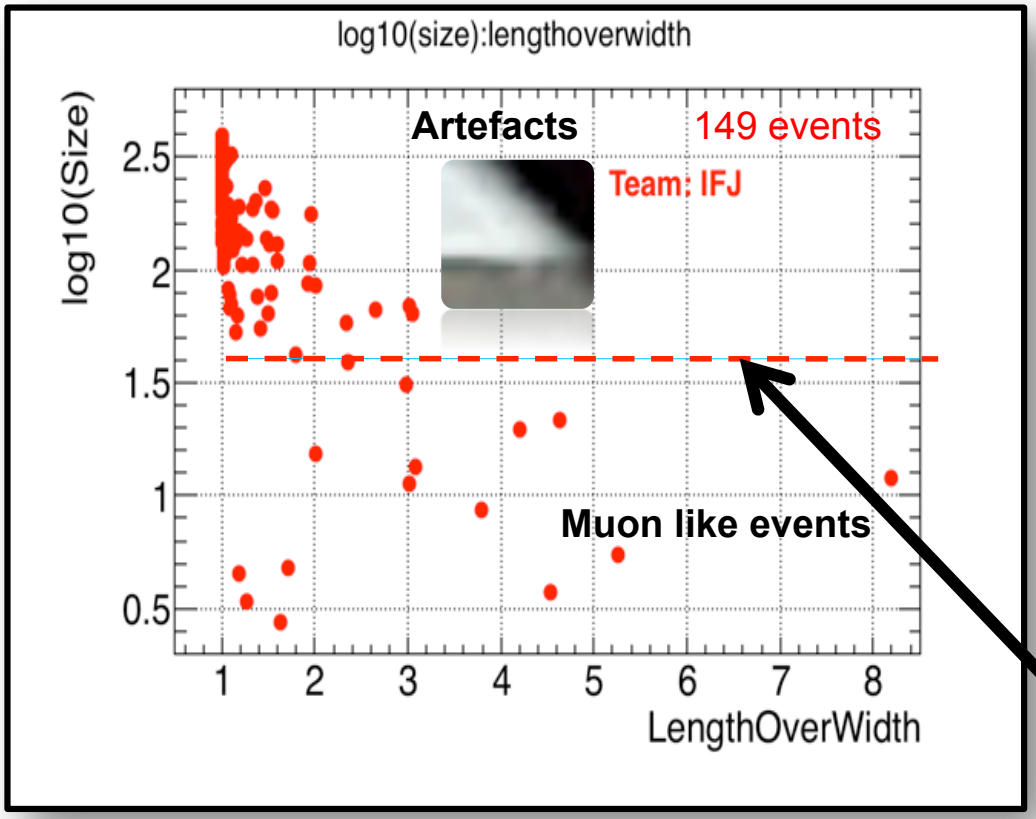
„IFJ”



„Team Rzezawa”

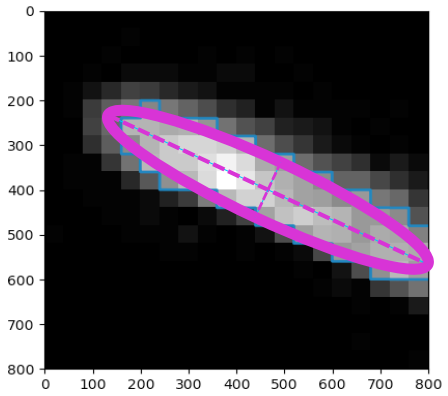
Level MAX: fun and emotions

The match: results ”IFJ”



Discrimination parameters:

SIZE: *sum of illuminace in all pixels above certain defined threshold*

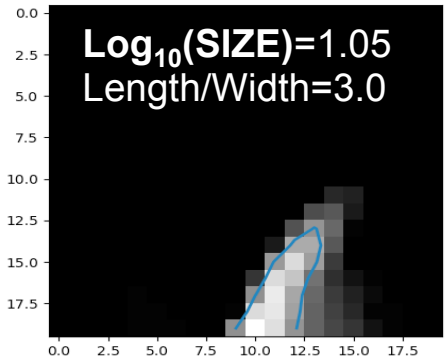


ellipse

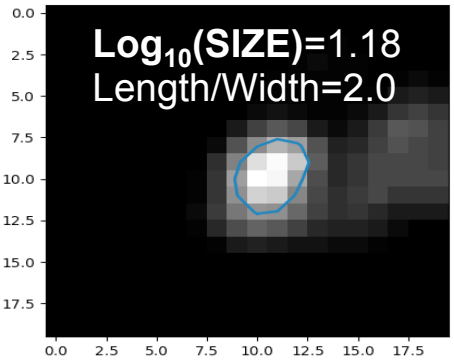
LENGTH/WIDTH: *the ratio of major to minor axis of ellipse*

Selection criterion: $\log_{10}(\text{SIZE}) < 1.7$

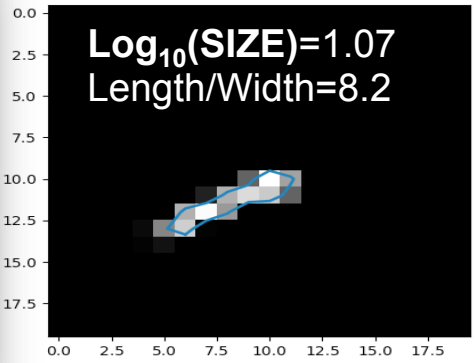
Muon like event



Muon like event

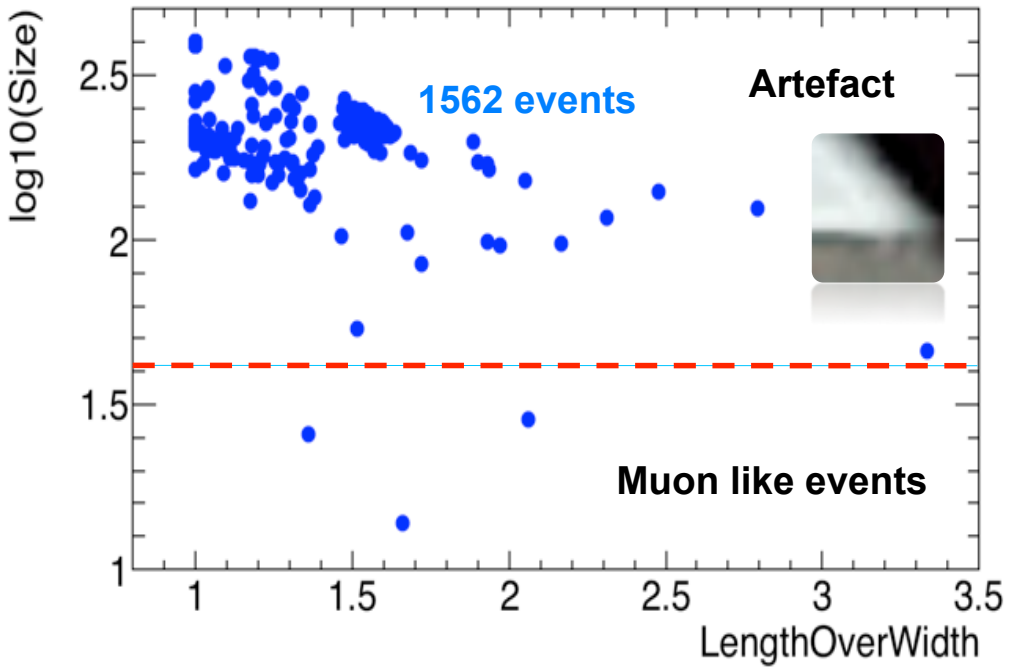


Muon like event



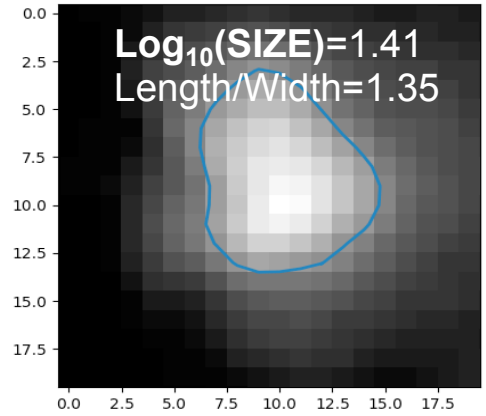
The match: results ‘’RZEZAWA’’

TEAM: „RZEZAWA”

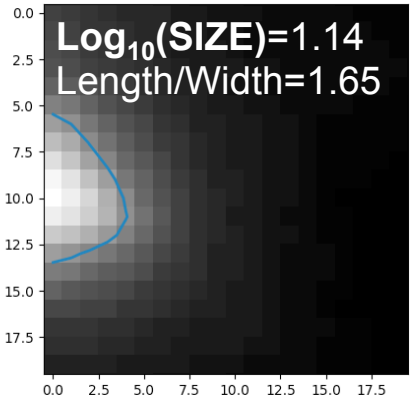


Eaxample of images:

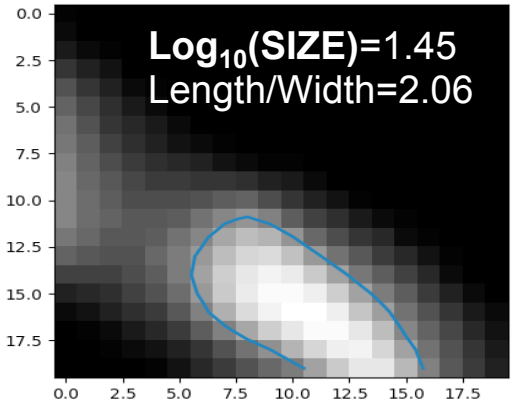
Muon like event



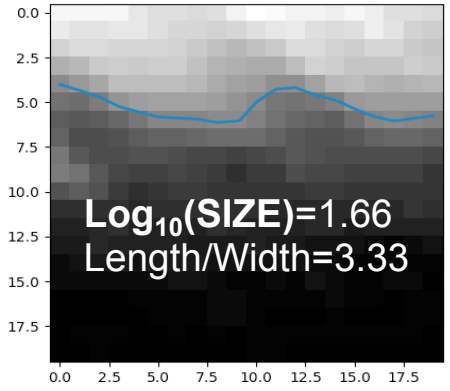
Muon like event



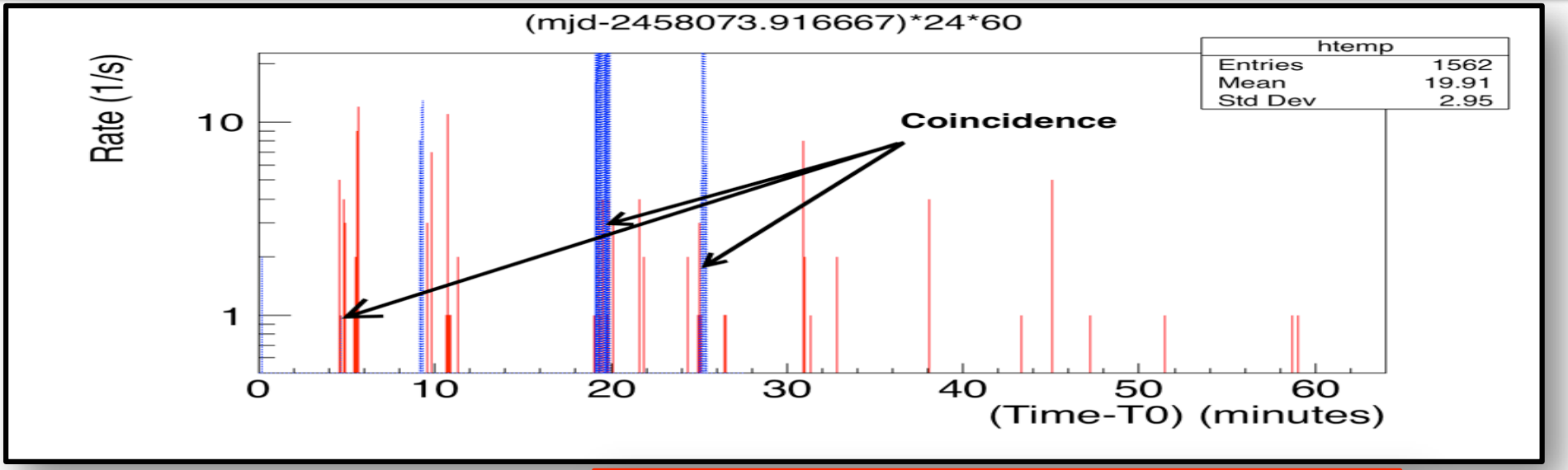
Muon like event



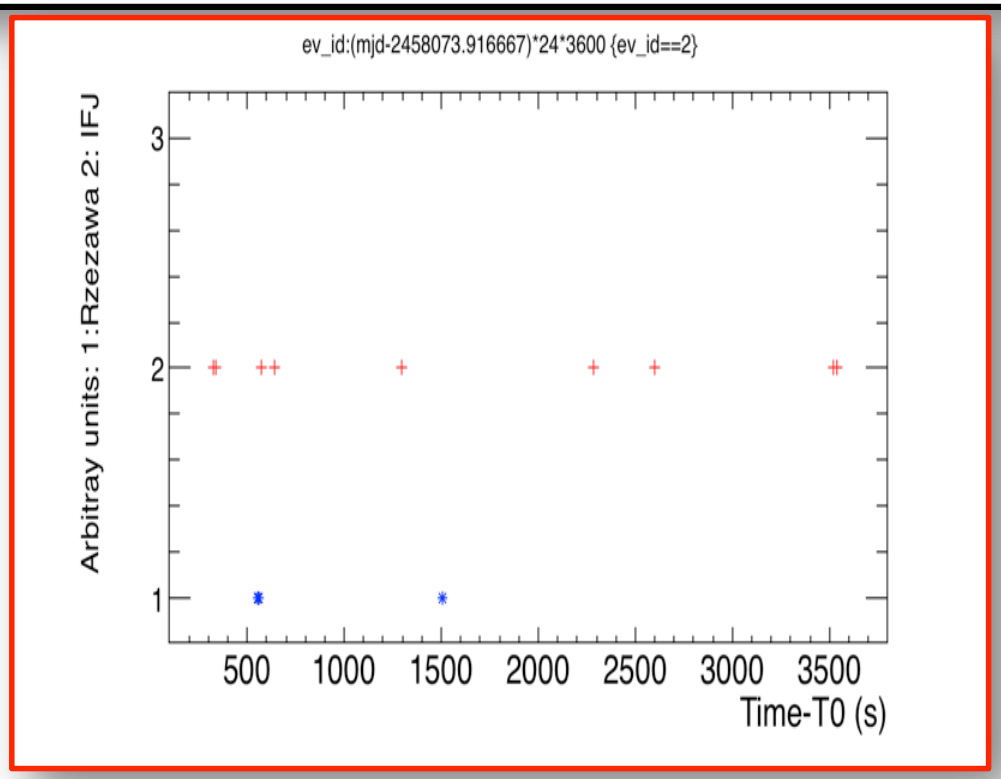
Artefact



The match: search for coincidence events



No coincidence events has been found



Summary

> CREDO: a unifying, global cosmic-ray project:

> Smartphones can be used to search for Cosmic-rays,
Proof of concept: the IFJ - Rzezawa match for catching cosmic-ray particles with the mobile device

> We invite the entire physics community to contribute to CREDO efforts!

Thank you for your attention

Analysis Script

This script runs over DB entres (json file) and

```
- extract entries for defined team/user
 - performed PCA component analysis
 - extract images after defined selection cut
 - and plot event rate.
```

As result you obtain the data file in the format:
MJD EventId time Size LengthoverWidth Length Width username

which can be use for future analysis

Remark: Dump of data base
https://api.credo.science/export/dump-all.jsonl

This is just example, you are welcome to modify
script and play.

```
import io
import json
import ijson
import base64
import numpy as np
import skimage
from skimage import io
from skimage import measure
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import math
from skimage.color import rgb2gray
from skimage import img_as_float
from skimage import io   #see link below for more information about imaga manipulations
#http://scikit-image.org/download.htmlfrom skimage.color import rgb2gray
from StringIO import StringIO
import sys
reload(sys)
sys.setdefaultencoding('utf8')

R_USER='IFJ'
ff=open('DATA_IFJ.txt', 'w')

data1= []
timeperiod=[]
k=0
ip=0
pv=0
pv1=0
with open('CREDO-data-dump-1510927369292.jsonl', 'r') as f:
 for line in f:
    data = json.loads(line)
    k=k+1
    print k
#read data
    x1=data["id"]
    x2=data["json"]["detection"]["width"]
    x3=data["json"]["detection"]["height"]
```

```
    x4=data["json"]["detection"]["accuracy"]
    x5=data["json"]["detection"]["altitude"]
    x6=data["json"]["detection"]["latitude"]
    x7=data["json"]["detection"]["longitude"]
    x8=data["json"]["detection"]["timestamp"]
    x9=data["json"]["user_info"]["name"]
    x9a=data["json"]["user_info"]["team"]
    x10=data["json"]["device_info"]["deviceModel"]

#Conversion from inis miliseconds to MJD minus MJD for first entry
    x11=((x8/(86400.0*1000))+2440587.5)-2458028.00723
#    timeperiod.append(x11)
###HERE YOUR SELECTION CUT is defined

    required_cut=R_USER
    container=str(x9a)
    frame=data["json"]["detection"]["frame_content"]
    if  str(frame)=='FRAME_CONTENT':
            continue

######################
#http://aa.usno.navy.mil/data/docs/JulianDate.php
#CET and UT +1 hrs diiference for Cracow: UT=CET-1
# 16.11.2017 10:00 UTC-11:00 UTC
    mjdday=((x8/(86400.0*1000))+2440587.5)
    if  (container==required_cut) and (mjdday>2458073.916667 and mjdday<2458073.958333) :

#    if  k>30000 and k<=35000:
        data1.append(x11)
        timeperiod.append(x11)

        ip=ip+1
                # read image from DB and decode

        a=data["json"]["detection"]["frame_content"]
        png_recovered = base64.decodestring(a)

        moon=skimage.io.imread(StringIO(png_recovered))
        moon.shape

        # string=str(x1)+".png"
        #f = open(string, "w")
        #f.write(png_recovered)
        #f.close()
        #png_recovered.shape
        #image.shape

#Conversion of image to black and white scale
#http://scikit-image.org/docs/dev/api/skimage.color.html#skimage.color.rgb2grey
# compute luminance of an RGB image with formula:
#Y = 0.2125 R + 0.7154 G + 0.0721 B

        img_gray = rgb2gray(moon)
        luminance_threshold=0.5*img_gray.max()
        contours = measure.find_contours(img_gray, luminance_threshold)
        nrows, ncols = img_gray.shape

        data_vector_x=[]
        data_vector_y=[]
        data2_vector_x=[]
        data2_vector_y=[]
        data2_vector_z=[]
        data=[]
        data2=[]

        size=0
        for i in xrange(nrows):
```

```
            for j in xrange(ncols):
                if img_gray[i, j]>luminance_threshold:
                   data_vector_x.append(j)
                   data_vector_y.append(-i)

                   data2_vector_x.append(j)
                   data2_vector_y.append(-i)
                   data2_vector_z.append(img_gray[i, j])
                   size=size+img_gray[i, j]

        for i in range(0, len(data_vector_x)):
                  data.append([data_vector_x[i],data_vector_y[i]])
                  data2.append([data2_vector_x[i],data2_vector_y[i],data2_vector_z[i]])

#np.savetxt('data_above_threshold.txt',data2)

#PCA analysis
# in my opinion the best tutorial about PCA
# http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf
# and for python implemetation, see for example this link
# https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html
        if len(data)<4:
            continue

        pca = PCA(n_components=2)
        pca.fit(data)
#the squared-length of the vector lambda1=math.sqrt
        if   len( pca.explained_variance_ ) < 2:
              print 'PCA fails, ebent skipped'
              continue

        lambda1,lambda2=pca.explained_variance_
        if lambda2   ==0:
            print ' Engenvalule zero: event skipped'
            continue
        pv=pv+1
        #with open(str(x9)+str(x1)+'.txt', 'w') as  f:
           #print >> f,  x1, '  ', x9
           #print >> f,  'Maximal luminance value in the image',img_gray.max()
           #print >>f, '  '
           #print >>f, 'Results for the following luminance threshold:',luminance_threshold
           #print >>f, 'Sum  of luminance for all pixels present  above the threshold: Size=',size

#print  >>f,  'Length=',math.sqrt(lambda1),' ','Width=',math.sqrt(lambda2), ' Length/Width=',math.
sqrt(lambda1)/math.sqrt(lambda2)
        a_eff=math.sqrt(lambda1)*math.sqrt(lambda2)*math.pi
           #print >>f,  'Effective area of elippse A_eff=Length*Width*pi:', a_eff
           #print  >>f, 'Size/A_eff', size/a_eff
        print  >>ff, mjdday, '  ',str(x1), '   ',str(x11),'  ', size, '  '
, math.sqrt(lambda1)/math.sqrt(lambda2), '   ', math.sqrt(lambda1), '   ', math.sqrt(lambda2), '
   ', str(x9)
         #  f.close()

        #   data_pca = pca.transform(data)
#np.savetxt('data_in_PCA_cordinate_system',data_pca)

#printin data after PCA i.e. in coorsinate system of data
#for i in range(0, len(data_pca)):
#     print data_pca[i][0],' ',data_pca[i][1]

        def draw_vector(v0, v1, ax=None):
             ax = ax or plt.gca()
             arrowprops=dict(arrowstyle='->',
                    linewidth=2,
                    shrinkA=0, shrinkB=0)
             ax.annotate('', v1, v0, arrowprops=arrowprops)
```

```
        fig, ax = plt.subplots(1, 2, figsize=(16, 6))
        fig.subplots_adjust(left=0.0625, right=0.95, wspace=0.1)
        ax[0].scatter(data_vector_x,data_vector_y, alpha=0.2)
        for length, vector in zip(pca.explained_variance_, pca.components_):
             v = vector * np.sqrt(length)
            #print v
             draw_vector(pca.mean_, pca.mean_ + v, ax=ax[0])

        ax[0].axis('equal');
        ax[0].set(xlabel='x', ylabel='y', title='input')
        X_pca = pca.transform(data)
        ax[1].scatter(X_pca[:, 0], X_pca[:, 1], alpha=0.4)

        draw_vector([0, 0], [0, 400], ax=ax[1])
        draw_vector([0, 0], [400, 0], ax=ax[1])

        ax[1].axis('equal')
        ax[1].set(xlabel='component 1', ylabel='component 2',
        title='principal components',
        xlim=(-800, 800), ylim=(-800, 800))

#plt.show()
 ###       fig.savefig('PCA-rotation_'+str(x1)+'.png')

 #plotting  luminance countour

        fig, ax = plt.subplots()
        ax.imshow(img_gray, interpolation='nearest', cmap=plt.cm.gray)
        for n, contour in enumerate(contours):
           ax.plot(contour[:, 1], contour[:, 0], linewidth=2)

        if     (math.log10(size)<1.7):
        # and (math.log10(size)>0.7):
          pv1=pv1+1
          fig.savefig('BW_countour_'+str(x1)+'.png')
  ###      fig.savefig('data_grayacle_with_countour_'+str(x1)+'.png')
      #  plt.show()

print "USER entris", len(data1), '    Ntot',   k,   '    usr.entr   ',ip, '   anlyzed events '
, pv, 'muon like:   ',pv1
print "Time period of data", timeperiod[0], "   ", timeperiod[-1],"    ", timeperiod[-1
]- timeperiod[0]
ff.close()
ntimebins=10000
deltaT=((timeperiod[-1]- timeperiod[0])*24*3600)/ntimebins

fig, ax = plt.subplots()
hist, bin_edges = np.histogram(data1, bins=10000)
hist_neg_cumulative =hist/deltaT
# [np.sum(hist[i:]) for i in range(len(hist))]
bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2.
plt.step(bin_centers, hist_neg_cumulative)
#
plt.title('Rate of '+R_USER+'  images in DB')
plt.xlabel(" T_i-T_0 (days) ")
plt.ylabel("Rate (1/s)")
plt.yscale('log')
plt.axis([0,50, 1e-3, 100])
plt.savefig('A_Rate_'+R_USER+'.png')
exit()
```