assignment05_2

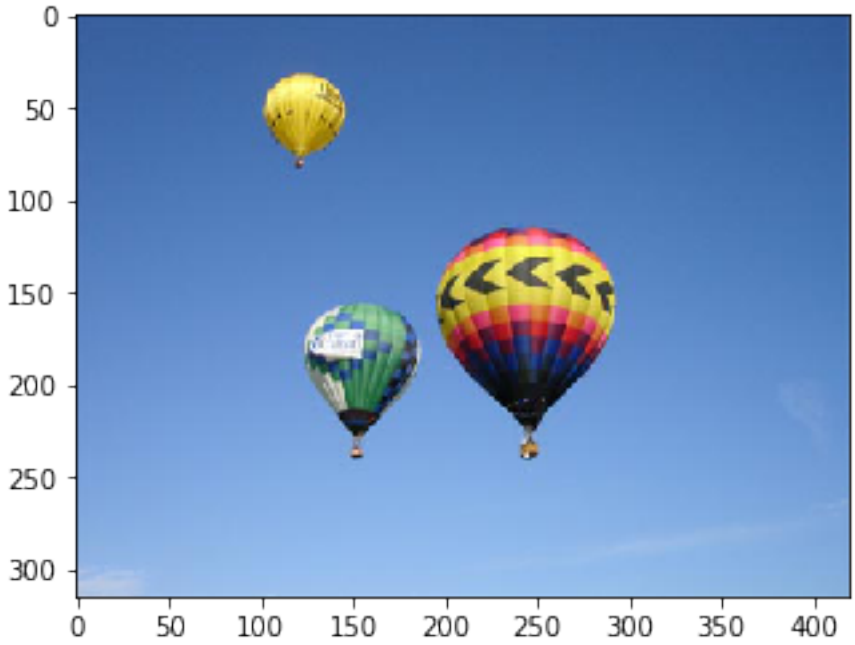
June 11, 2019

1 Input color image

In [1]:
```
import PIL.Image as pilimg
import numpy as np
import matplotlib.pyplot as plt
import random
import copy

#read my image(pixel)
image = pilimg.open("C:\\Users\\recognize_data\\image.jpg")
#image = pilimg.open("C:\\Users\\recognize_data\\color.png")

#image pixel data to array
#315*420
#image_pixel[0][0] = [74 112 175]
image_pixel = np.array(image)
#image size
size_col = image.size[0]    # width of the image 420
size_row = image.size[1]    # height of the image 315
image_size = size_col * size_row
```

In [2]:
```
plt.imshow(image_pixel)
```

Out[2]:
```
<matplotlib.image.AxesImage at 0x204f5cb2908>
```



In [3]:
```
class AllProcess:

    def _init_(self, k, image_label, output_image, centroid, count, energy, store_dista

        self.k = k
        self.image_label = image_label
        self.output_image = output_image

        self.centroid = centroid
        self.count = count

        self.energy = energy
        self.store_distance = store_distance

        self.t = t

    def setdata(self,k):
        self.k = k
        self.centroid = np.empty((k, 3), dtype = int)
        self.count = np.empty(k, dtype = int)
        self.store_distance = np.empty(k)

        self.image_label = np.empty((size_row, size_col), dtype = int)
        self.output_image = np.empty((size_row, size_col,3), dtype = int)
        self.energy = np.empty(1, dtype = float)
        self.t = 0
```

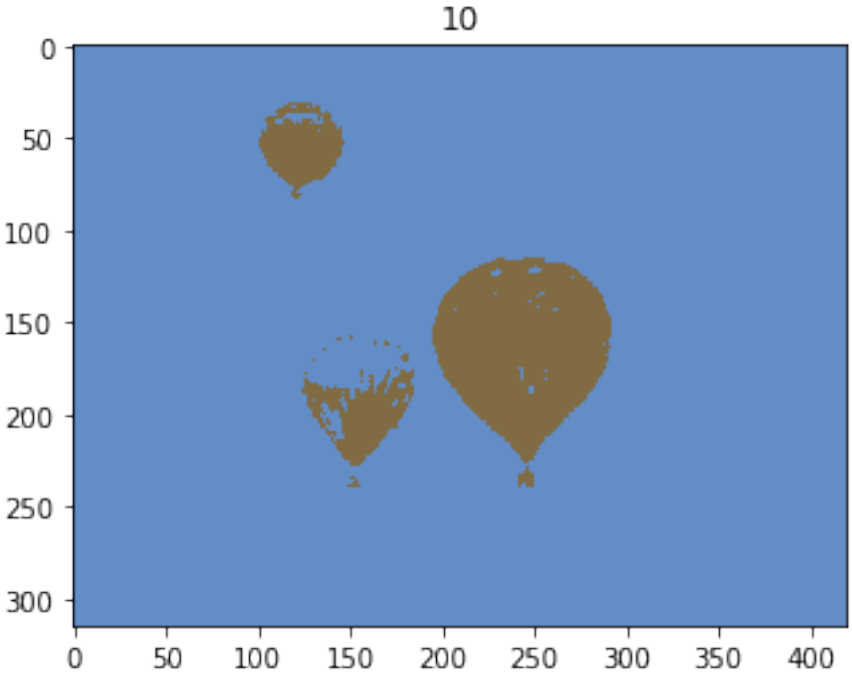
```
def random_labeling(self):
    for i in range(0,size_row):
        for j in range(0,size_col):
            self.image_label[i][j] = random.randrange(0, self.k)

def update_centroid(self):
    #initialize the centroid
    for i in range(0,self.k):
        self.centroid[i][0] = 0
        self.centroid[i][1] = 0
        self.centroid[i][2] = 0
        self.count[i] = 0

    for i in range(0,size_row):
        for j in range(0,size_col):
            a = self.image_label[i][j]
            self.centroid[a][0] += image_pixel[i][j][0]#R
            self.centroid[a][1] += image_pixel[i][j][1]#G
            self.centroid[a][2] += image_pixel[i][j][2]#B
            self.count[a] += 1

        # average center
    for i in range(0,self.k):
        if(self.count[i]==0):
            continue
            #count[i] = 1
        self.centroid[i][0] = self.centroid[i][0]/self.count[i]
        self.centroid[i][1] = self.centroid[i][1]/self.count[i]
        self.centroid[i][2] = self.centroid[i][2]/self.count[i]

def energy_function(self, t):
    global energy
    sum_value = 0
    for i in range(0,size_row):
        for j in range(0,size_col):
            a = self.image_label[i][j]
            sum_value += sum((image_pixel[i][j]-self.centroid[a]) ** 2)

    sum_value = np.sqrt(sum_value)/image_size
    if t == 0:
        self.energy = sum_value
    else:
        self.energy = np.append(self.energy, sum_value)

def print_output(self, t):
    for i in range(0,size_row):
        for j in range(0,size_col):
```

```
                    a = self.image_label[i][j]
                    self.output_image[i][j][0] = self.centroid[a][0]
                    self.output_image[i][j][1] = self.centroid[a][1]
                    self.output_image[i][j][2] = self.centroid[a][2]
            plt.title(t)
            plt.imshow(self.output_image)

        #distance
        def distance(self,x,y):
            d = sum((x-y) ** 2)
            s = np.sqrt(d)
            return(s)

        def find_nearest(self, a):
            for i in range(0, self.k):
                self.store_distance[i] = self.distance(a, self.centroid[i])
            r = self.store_distance.argmin()
            return r
        #update label
        def update_label(self):
            for i in range(0,size_row):
                for j in range(0,size_col):
                    self.image_label[i][j] = self.find_nearest(image_pixel[i][j])

        def plot_energy(self):
            print(self.energy)
            plt.plot(self.energy,'ro')#red(r) dot(o)
            plt.xlabel('time')
            plt.ylabel('energy')
            plt.show()
```

2 k = 2

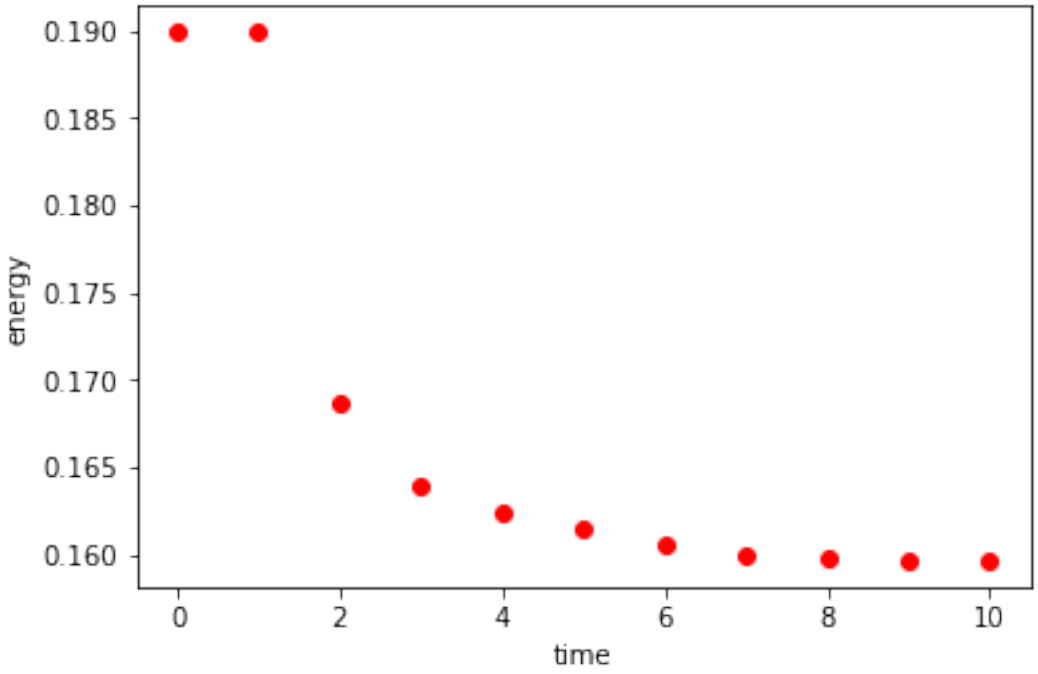
In [4]:
```
p1 = AllProcess()
# k = 2
p1.setdata(2)
p1.random_labeling()
p1.update_centroid()
p1.energy_function(p1.t)

while True:
    p1.update_label()
    old_centroid = copy.deepcopy(p1.centroid)
    p1.update_centroid()
    p1.t += 1
    p1.energy_function(p1.t)
    if np.array_equal(old_centroid, p1.centroid):
```

```
p1.print_output(p1.t)
break
```



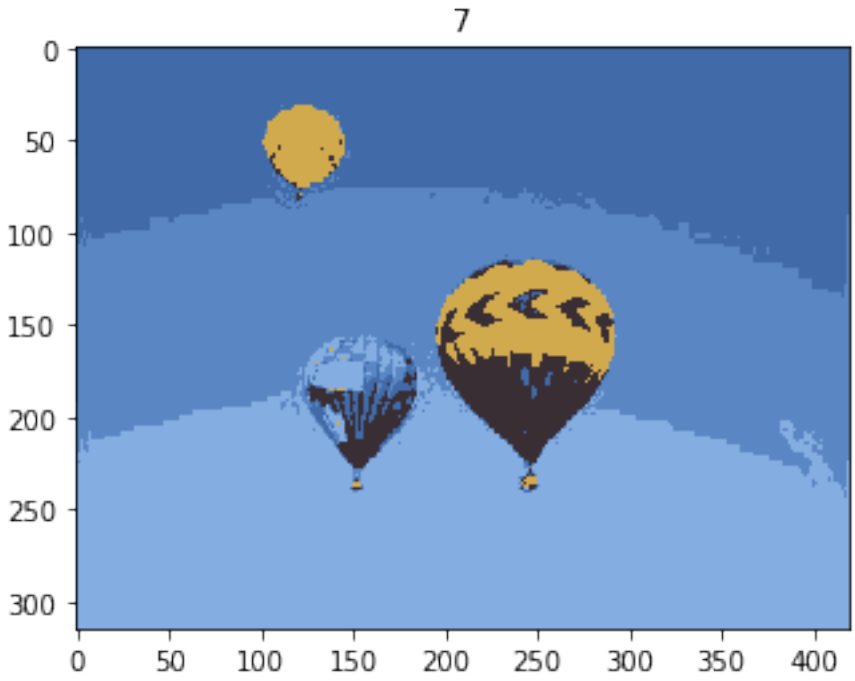
In [5]: `p1.plot_energy()`

```
[0.18990147 0.18990147 0.16864041 0.16398045 0.16245398 0.16148212
 0.16054345 0.15991095 0.1597088  0.15964442 0.15964189]
```



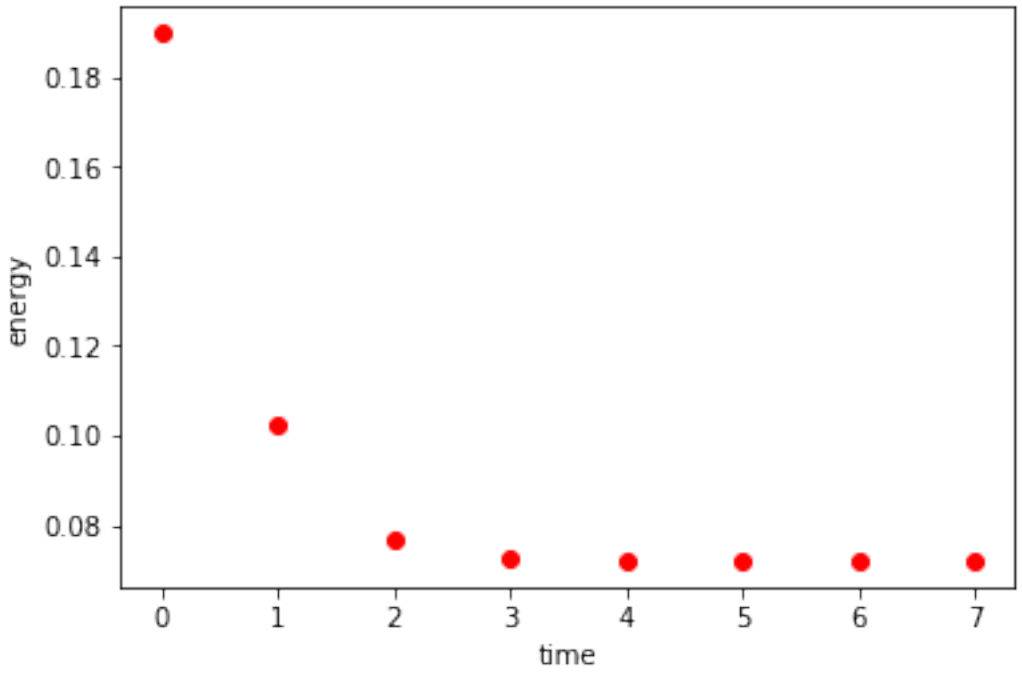
3 k = 5

```
In [6]: p4 = AllProcess()
        # k = 5
        p4.setdata(5)
        p4.random_labeling()
        p4.update_centroid()
        p4.energy_function(p4.t)

        while True:
            p4.update_label()
            old_centroid = copy.deepcopy(p4.centroid)
            p4.update_centroid()
            p4.t += 1
            p4.energy_function(p4.t)
            if np.array_equal(old_centroid, p4.centroid):
                p4.print_output(p4.t)
                break
```



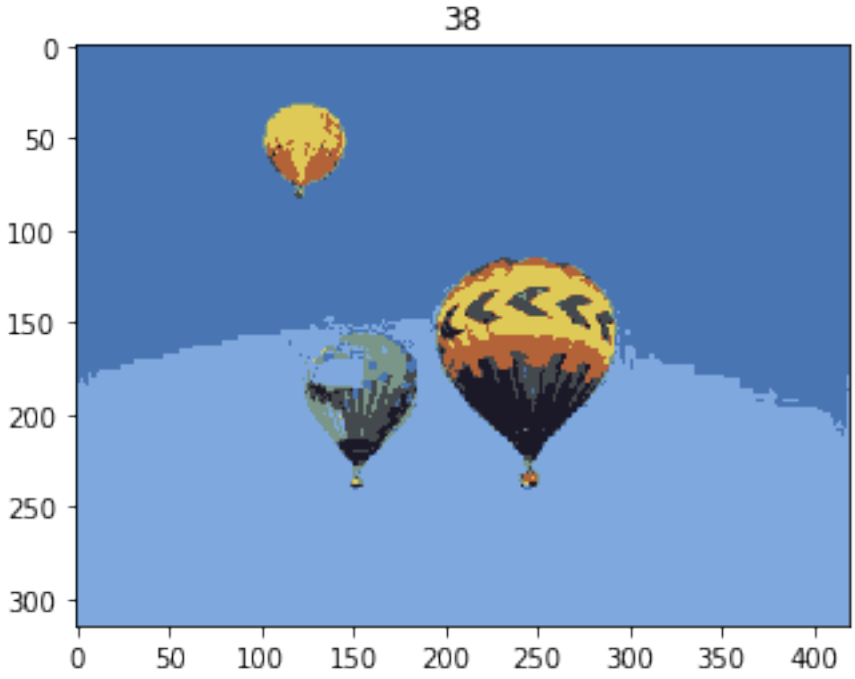
In [7]: p4.plot_energy()

```
[0.18989786 0.10262613 0.07654553 0.07284614 0.07223356 0.07205918
 0.07201633 0.07201384]
```



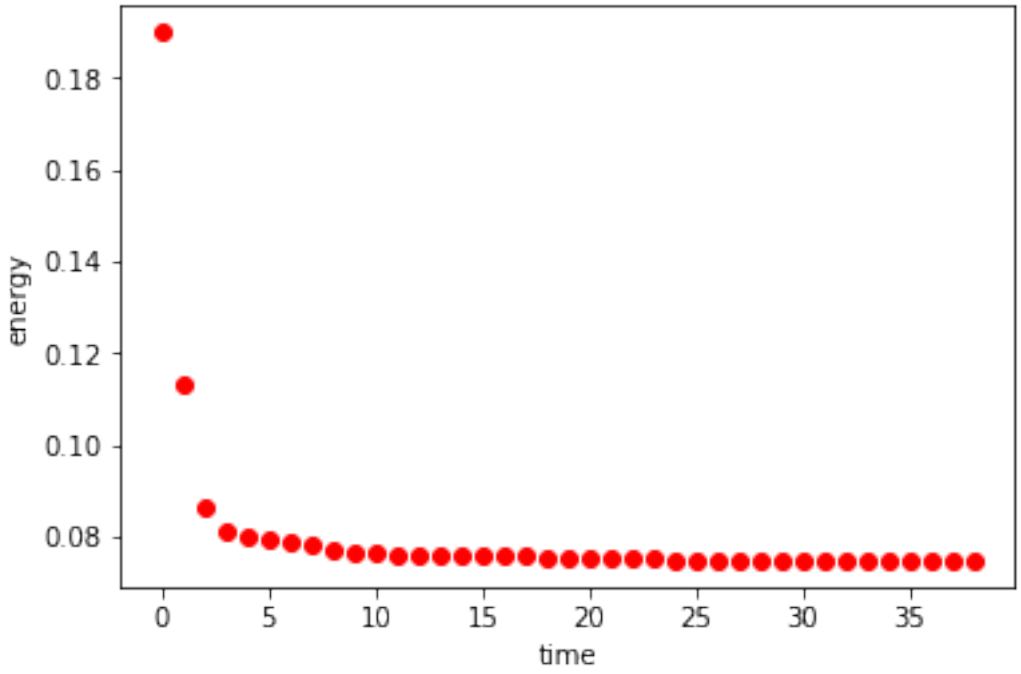
4 k = 7

In [8]:
```
p2 = AllProcess()
# k = 7
p2.setdata(7)
p2.random_labeling()
p2.update_centroid()
p2.energy_function(p2.t)

while True:
    p2.update_label()
    old_centroid = copy.deepcopy(p2.centroid)
    p2.update_centroid()
    p2.t += 1
    p2.energy_function(p2.t)
    if np.array_equal(old_centroid, p2.centroid):
        p2.print_output(p2.t)
        break
```



In [9]:
```
p2.plot_energy()
```

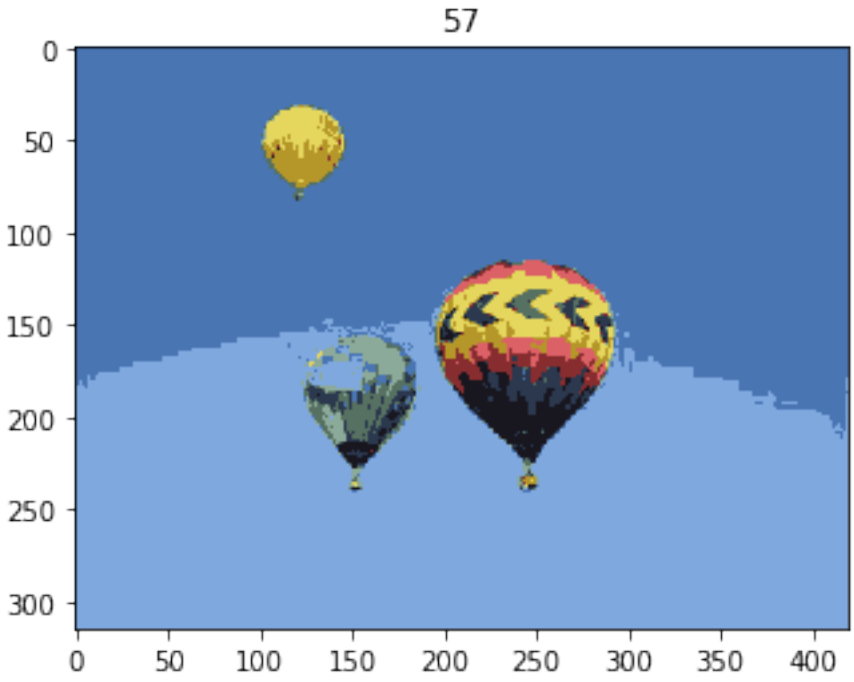
```
[0.18989893 0.11296979 0.08637688 0.0814129  0.08015638 0.07943739
 0.07874134 0.07802938 0.07720512 0.07653088 0.07620618 0.07605134
 0.0759259  0.07585064 0.07579517 0.07575386 0.07571115 0.07565061
 0.07556709 0.07546632 0.07537426 0.07524793 0.07514429 0.07506239
 0.07499591 0.07494549 0.07490963 0.07487707 0.0748538  0.07483104
 0.07481083 0.07479641 0.07478543 0.07477934 0.07477655 0.074773
 0.07477145 0.07479224 0.0747908 ]
```



5 k = 10

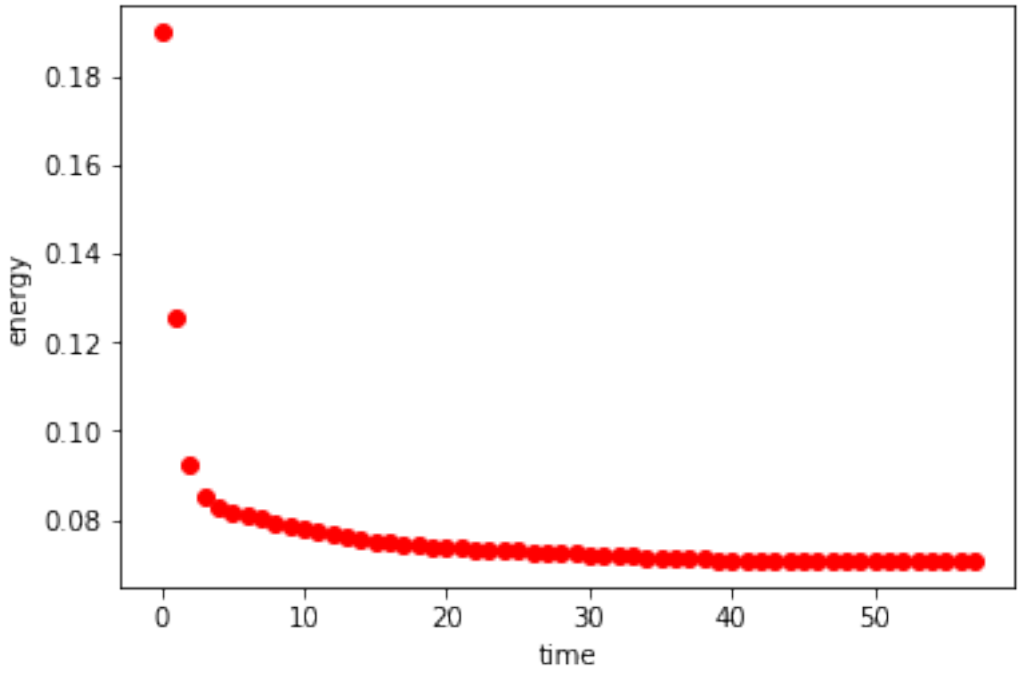
```
In [10]: p3 = AllProcess()
         # k = 10
         p3.setdata(10)
         p3.random_labeling()
         p3.update_centroid()
         p3.energy_function(p3.t)

         while True:
             p3.update_label()
             old_centroid = copy.deepcopy(p3.centroid)
             p3.update_centroid()
             p3.t += 1
             p3.energy_function(p3.t)
```

```
if np.array_equal(old_centroid, p3.centroid):
    p3.print_output(p3.t)
    break
```



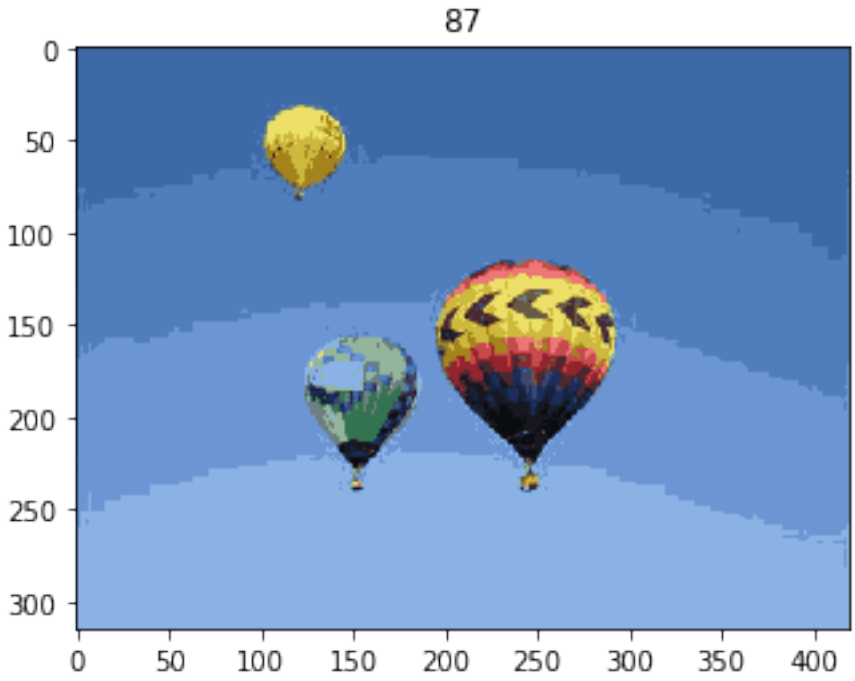
In [11]: p3.plot_energy()

```
[0.18989918 0.12535864 0.09255066 0.08506829 0.08267119 0.08154609
 0.08085999 0.08016063 0.07938706 0.0785976  0.07773333 0.07707196
 0.07663423 0.07622729 0.07577646 0.07524495 0.07479171 0.0744773
 0.07421632 0.07396288 0.07375721 0.07352914 0.0733414  0.07319183
 0.07304415 0.07289818 0.07276602 0.07262771 0.07248105 0.07229498
 0.07209685 0.0719276  0.07179402 0.07167913 0.07154921 0.07135519
 0.07121952 0.07113446 0.07107717 0.07102486 0.07097248 0.07091722
 0.07086984 0.07084067 0.07082346 0.07081136 0.0708028  0.07079735
 0.07079326 0.07078931 0.07078472 0.07078312 0.07078193 0.07078209
 0.07078211 0.07078077 0.07078118 0.0707809 ]
```

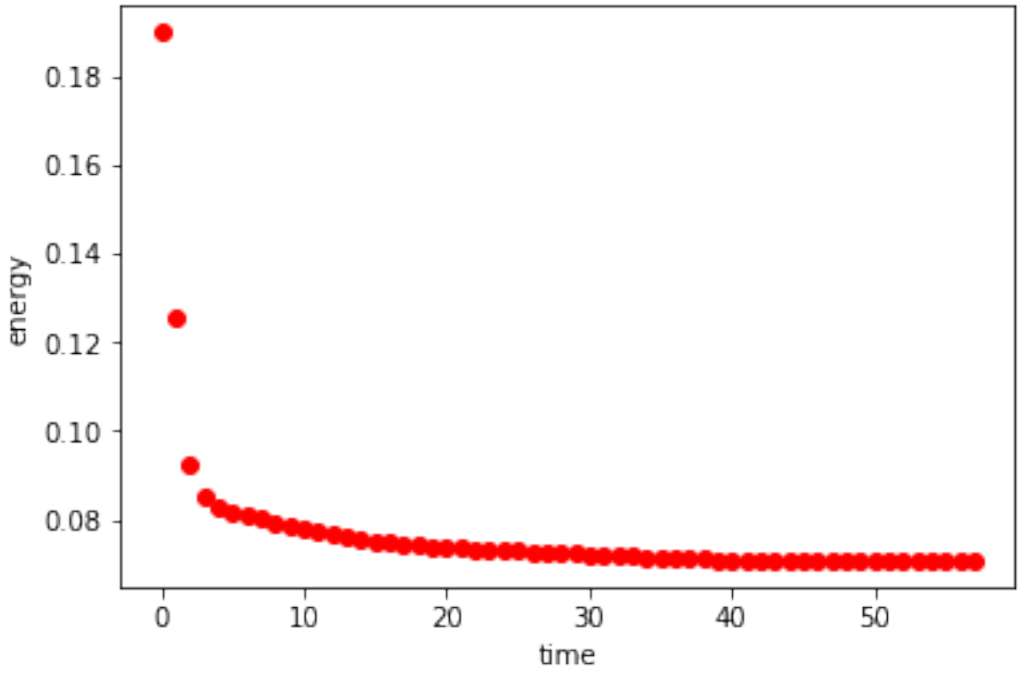


6 k = 20

```
In [12]: p5 = AllProcess()
         # k = 20
         p5.setdata(20)
         p5.random_labeling()
         p5.update_centroid()
         p5.energy_function(p5.t)

         while True:
             p5.update_label()
             old_centroid = copy.deepcopy(p5.centroid)
             p5.update_centroid()
             p5.t += 1
             p5.energy_function(p5.t)
             if np.array_equal(old_centroid, p5.centroid):
                 p5.print_output(p5.t)
                 break
```




```
In [13]: p3.plot_energy()

[0.18989918 0.12535864 0.09255066 0.08506829 0.08267119 0.08154609
 0.08085999 0.08016063 0.07938706 0.0785976  0.07773333 0.07707196
 0.07663423 0.07622729 0.07577646 0.07524495 0.07479171 0.0744773
 0.07421632 0.07396288 0.07375721 0.07352914 0.0733414  0.07319183
 0.07304415 0.07289818 0.07276602 0.07262771 0.07248105 0.07229498
 0.07209685 0.0719276  0.07179402 0.07167913 0.07154921 0.07135519
 0.07121952 0.07113446 0.07107717 0.07102486 0.07097248 0.07091722
 0.07086984 0.07084067 0.07082346 0.07081136 0.0708028  0.07079735
 0.07079326 0.07078931 0.07078472 0.07078312 0.07078193 0.07078209
 0.07078211 0.07078077 0.07078118 0.0707809 ]
```



In [ ]: