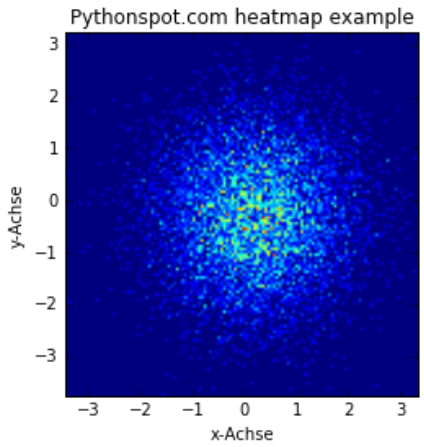
In [ ]:
```
#Grobe Ideen zur Programmierung der Heatmap

#man müsste den Tisch / die Videoaufnahme in ein Rastergitter einteilen
und dann für jeden Abschnitt des Gitters einen
#Counter erstellen, der immer 1 hinzufügt, wenn sich ein Puck im Feld be
findet.
#dies könnte man wahrscheinlich mit einer if-Schleife lösen, bei der man
die Koordinaten abfragt und überprüft
#z.B if getPos(Puck) == posFeld:
#         counterF1 =+ 1
#Problem: Position Puck ist ein Punkt (Mittelpunkt) (oder 4 Punkte mit d
er Box von opencv)
#und das Feld ist eine Fläche dx*dy -> mathematisch ausgedrückt: koordPu
ck1 ist Element der Menge Feld1


#eine Geschwindigkeits- und Positionstabelle muss importiert werden, dam
it die Daten ausgewertet werden können.
#Tabelle im Stil:  Zeit | x-Koordinate | y-Koordinate | Geschwindigkeit(
Pixel/frame)
#                     0        1              2             0
#                     1        2              3             1
#                     2        4              2            2.24
#usw. und das für jeden Puck. daraus kann das ermittelt werden, wann ein
Puck eines der Felder betritt, wann er es
#verlässt und welche Geschwindigkeit er währenddessen hat. (Position für
Aufenthaltswahrscheinlichkeits-Heatmap und
#Geschwindigkeit für Geschwindigkeitsheatmap)

#desweiteren muss dann der Counter für jedes Feld einer Farbe zugeordnet
werden, was wahrscheinlich mit einem dictionary
#zu lösen wäre. z.B.  d = {0 = 'white'; 1 = 'lightblue'; 2 = 'blue'
#                         3 = 'green; 4 = 'yellow'; 5 = 'orange'; 6 = '
red'}    o.Ä.
#danach müssten die Farben mittels Farbwerte in eine Grafik gezeichnet w
erden.
#-> je kleiner man die Felder macht, desto genauer wär die Heatmap und d
esto mehr Rechenarbeit hätte man. -> Mittelmaß
```

In [12]:
```
import numpy as np
import numpy.random
import matplotlib.pyplot as plt

# Create data
x = np.random.randn(8192)     #zufällige Punkte mittels numpy.random, An
zahl der Versuche
y = np.random.randn(8192)     # -> hier müssten dann die Wertetabellen e
ingefügt werden.

# Create heatmap
heatmap, xedges, yedges = np.histogram2d(x, y, bins=(128,128))  #Ausmaße
/ Genauigkeit der Heatmap
extent = [xedges[0], xedges[-1], yedges[0], yedges[-1]]

# Plot heatmap
plt.clf()
plt.title('Pythonspot.com heatmap example')
plt.ylabel('y-Achse')
plt.xlabel('x-Achse')
plt.imshow(heatmap, extent=extent)
plt.show()
```



In [ ]: