Script Python + R

June 6, 2022

1 Automated tracking using OpenCV

1.1 Instructions

The following code was designed in Python lenguage and works by automatically detecting objects in order to track the location of a single animal across the course of a single video session using adaptive thresholding based on their contrast with the environment. The animal, in x,y coordinates, is recorded, as well as the distance in pixels that the animal moves from one frame to the next. Here we describe the most revelvant commands in order to the user can undertand and create your own traking routing. Here we describe the most revelvant commands in order to the user can undertand and create your own animal tracking routine.

El siguiente código se ha diseñado en lenguaje Python y funciona detectando automáticamente los objetos para rastrear la ubicación de un solo animal a lo largo de una única sesión de vídeo utilizando un umbral adaptativo basado en su contraste con el entorno. Se registra el animal, en coordenadas x,y, así como la distancia en píxeles a la que se mueve el animal de un fotograma a otro. Aquí describimos los comandos más reveladores para que el usuario pueda entender y crear su propia rutina de seguimiento de animales.

2 1. Load Packages

The following code loads neccessary Python packages.

Los siguientes códigos son para importar los paquetes de Python necesarios

[1]:
```
import cv2
import numpy as np
from datetime import datetime, timedelta
import time
from numpy import shape
import sys
```

---

3 2. Set directory and setup video dimension

If the video file you are reading is in the same folder as your pyton script, simply specify the correct file name. Else, you would have to specify the complete path to the video file.

Si el archivo de vídeo que está leyendo está en la misma carpeta que le script de python, simplemente especifique el nombre correcto del archivo. De lo contrario, tendrás que especificar la ruta completa del archivo de vídeo.

`video_in` : The directory path of the folder containing the video to be analysed including the file extension (e.g., /Users/Videos/Crab.avi').

`video_in` : La ruta del directorio de la carpeta la cual contiene el vídeo a analizar incluyendo la extensión del mismo (e.g., /Users/Videos/Crab.avi').

`data_file` : The directory path of the folder containing the dataframe ouput including the file extension (e.g., /Users/Videos/Crab.csv').

`data_file` : La ruta del directorio de la carpeta que contiene la salida de los datos, incluyendo la extensión del archivo (por ejemplo, /Usuarios/Videos/Crab.csv').

`video_start_datetime` : The date and datetime start of the video on which to begin processing.

`video_start_datetime` : La fecha y hora de inicio del vídeo en la que se debe comenzar a procesar.

`video_end_datetime` : The date and datetime start of the video on which to begin processing.

`video_end_datetime` : La fecha y hora de inicio del vídeo en la que se debe comenzar a procesar

`video_number` : ***(Optional)*** Number of analyzed video only is usefull to have a reference of the analyzed videos.

`video_number` : ***(Opcional)*** Número de video analizado sólo sirve para tener una referencia de los vídeos analizados.

`tank` : Allows the user to alter the tracking arena using differents projection coordinates. This is useful when videos have irregular dimensions due to the camera position. By defoult use `tank_1` (this one is not corrected)

`tank` : Permite al usuario alterar el escenario de seguimiento utilizando diferentes coordenadas de proyección. Esto es útil cuando los vídeos tienen dimensiones irregulares debido a la posición de la cámara. Por defecto utiliza `tank_1` (este no está corregido)

`video_out` : The directory path of the folder containing the video ouput analysed including the file extension (e.g., /Users/Videos/video_out.avi').

`video_out` : La ruta del directorio de la carpeta que contiene la salida del video analizado, incluyendo la extensión del archivo (por ejemplo, /Usuarios/Videos/video_out.avi')

Warning! Note that if you are using a Windows system in all the cases (e.i. `video_in`, `data_file` and `video_out`) the path with backslashes, place an 'r' in front of the directory path (e.g., r'.csv').

Tenga en cuenta que si utiliza una ruta de Windows las barras estan invertidas por lo que en todos los casos (ei `video_in`, `data_file` y `video_out`) hay que colocar una 'r' delante de la ruta del directorio (por ejemplo, r'.csv').

Remember! The user should reduce the frame rates of video with ffmpeg library (v2.8.14), or other similar program, to 1 frames/second (this example) or by the frames that are convenient (according to animal movements or video length). This will make the routine have a better tracking accurate and will shorten the computational time of the process.

reducir la velocidad de los fotogramas del vídeo con la biblioteca ffmpeg (v2.8.14), u otro programa similar, a 1 fotograma/segundo (este ejemplo) o por los frames que les sea conveniente (según movimientos del animal o longitud del video). Esto hara que la rutina tenga una mejor presision en el seguimiento y acortará los tiempos computacionales del proceso.

[2]:
```
video_in = '/run/media/kechu/kECHU/PC_lab_peque/opencv/OpenCV_actualizado/jose/
 ↪cam6azul.avi'
data_file = '/run/media/kechu/kECHU/PC_lab_peque/opencv/OpenCV_actualizado/jose/
 ↪cam6azul.csv'
video_start_datetime = datetime.strptime("2/05/2018 10:20:00", "%d/%m/%Y %H:%M:
 ↪%S")
print(video_start_datetime)

video_end_datetime = datetime.strptime("2/05/2018 10:40:27", "%d/%m/%Y %H:%M:%S")
print(video_end_datetime)

video_number = 1
tank=1
video_out = '/run/media/kechu/kECHU/PC_lab_peque/opencv/OpenCV_actualizado/jose/
 ↪video_out.avi'
print(video_out)
```

```
2018-05-02 10:20:00
2018-05-02 10:40:27
/run/media/kechu/kECHU/PC_lab_peque/opencv/OpenCV_actualizado/jose/video_out.avi
```

---

4 3. Parameter settings to improve tracking accuracy under different conditions

`fourcc` : is a 4-byte code used to specify the video codec (an identifier for the video codec, compression format, and color/pixel format in video files). The list of available codes can be found at fourcc.org. There are many FOURCC codes available, but in this post, we will work only with XVID.

`fourcc` : es un código de 4 bytes utilizado para especificar el códec de vídeo (identificador para el códec de vídeo, el formato de compresión y el formato de color/píxel en los archivos de vídeo). La lista de códigos disponibles se puede encontrar en fourcc.org. Hay muchos códigos FOURCC disponibles, pero en este post, trabajaremos sólo con XVID.

`video_writer` : To save a video in OpenCV `cv.VideoWriter()` method is used. This use four parameters cv2.VideoWriter(filename, fourcc, fps, frame_size), filename: is the name for the output video, fourcc: was defined above, fps: is the frame rate (video speed) of the output video, must be an integer and may also be slower depending upon computer speed. frame_size: is the size of the output video frames (width, height). If the user wants the output to be larger or smaller, or they want the aspect ratio to be different, resize can be supplied as in the following example: 'resize' : '(100,200)' Here, the first number corresponds to the adjusted width of the frame, whereas the

second number corresponds to the adjusted height. Both numbers reflect pixel units and should be integers. Set resize equal to None if no resizing is to be done.

`video_writer` : Para guardar un video en OpenCV se utiliza el método `cv.VideoWriter()`. Esto utiliza cuatro parámetros cv2.VideoWriter(filename, fourcc, fps, frame_size), filename: es el nombre del video de salida, fourcc: fue definido anteriormente, fps: es la velocidad de los cuadros del video de salida y frame_size: es el tamaño de los cuadros del video de salida (ancho, alto).

`cap` : Create a VideoCapture object

`cap` : Crea un objeto de videocaptura

`kernel` : The `cv2.getStructuringElement` function requires two arguments: the first is the type of structuring element we want, and the second is the size of the structuring element.

`kernel` : to dilate and filter image

`fgbg` : In brackets the firts parameter is the length of the history (`1000`). The second parameter is the threshold on the squared Mahalanobis distance between the pixel and the model to decide whether a pixel is well described by the background model (`45`). The trhird paramater is use to detect the shadows, if true, the algorithm will detect its and mark them. So if you do not need this feature, set the parameter to false (`False`).

`fgbg` : Entre paréntesis: el primer parámetro es la longitud del historial (`1000`). El segundo parámetro es el umbral de la distancia de Mahalanobis al cuadrado entre el píxel y el modelo para decidir si un píxel está bien descrito por el modelo de fondo (`45`). El tercer parámetro, se utiliza para detectar la sombra, si es verdadero, el algoritmo la detectará la seguirá. Si no necesita esta función, setee el parámetro en falso (`False`).

Warning! Only a few of the FourCC codecs will work on your system based on the availability of the codecs on your system. XVID is a safe choice.

Sólo algunos de los codecs de FourCC funcionarán en su sistema en función de la disponibilidad de los codecs en su sistema. XVID es una opción segura

[3]:
```
fourcc = cv2.VideoWriter_fourcc(*'XVID')
video_writer = cv2.VideoWriter(video_out, fourcc, 1.0, (1280,720))

cap = cv2.VideoCapture(video_in)
frames_number = cap.get(cv2.CAP_PROP_FRAME_COUNT)
print('frames_number: '+str(frames_number))

kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(9,9))
fgbg = cv2.createBackgroundSubtractorMOG2(1000,45,False)
f = open(data_file,'a')

def time_calculation(frame_number):
    time_transcurred = timedelta(seconds=(frame_number * seconds_by_frame))
    frame_time = video_start_datetime + time_transcurred
    return frame_time
```

```
frames_number: 1227.0
```

5 4. (Optional) Projection coordinates

In case the camera is not positioned precisely (i.e. exactly at a right angle horizontally or perpendicular to the experimental arena). The following code is used to adjust the coordinates of the position of the animals to be tracked according to the position of the camera with respect to the arena. `tank == 1` by default and it is exactly perpendicular, `tank == 2` and `tank == 3` are examples of different settings where the cameras are in different angular positions with respect to the arena. Here the user with the camera position data can modify the tank to be used to record an accurate position.

En caso en que la cámara no este colocada de manera precisa (e.i. exactamente en ángulo recto horizontal o perpendicular a la arena de experimentación). El código es utilizado para ajustar las coordenadas de la posición de los animales a seguir según la posición de la cámara respecto a la arena. `tank == 1` es por omisión y es exactamente perpendicular, los `tank == 2` y `tank == 3` son ejemplos de diferentes ajustes dónde las cámaras están en diferentes posiciones angulares respecto a la arena. Aquí el usuario con los datos de la posición de la cámara puede modificar el tanque a usar para registrar una posición precisa.

`calculate_movement`: Transforme pixels to mm.

`calculate_movement`: Transforma pixeles a mm.

[4]:
```
def project_XY(center):
    if tank == 1:
        center_prj = ((center[0]),center[1])
    elif tank == 2:
        center_prj = (center[0],(0.0017*center[1]**2+5.0233*center[1]-2456.7))
    elif tank == 3:
        center_prj = ((0.877*center[0]+79.953),center[1])
    else:
        center_prj = ((-0.0019*center[0]**2+3.5582*center[0]-866.06),center[1])
    return center_prj

def calculate_movement(center1,center2):
    if center1[0] == -1 or center2[0] == -1:
        millimeters = 0.0
    else:
        pixels = ((center2[0]-center1[0])**2+(center2[1]-center1[1])**2)**0.5
        if tank == 1:
            millimeters = pixels*(50/40)
        else:
            millimeters = pixels*(50/20)
    return millimeters
```

---

6 5. Initialize variables

`path` : The directory path of the folder containing the initial paramaters attached togheter this routine. It is convenient to place it in the same directory where the video to be analyzed is located

(e.g., /Users/Videos/'). The name of the archive is 'last_roi1_center.npy'

`path` : La ruta del directorio de la carpeta que contiene los parámetros iniciales adjuntos a esta rutina. Es conveniente colocarla en el mismo directorio donde se encuentra el vídeo a analizar (ej, /Usuarios/Videos/'). El nombre del archivo es 'last_roi1_center.npy'

Warning! Note that if you are using a Windows system in all the cases (e.i. `video_in`, `data_file` and `video_out`) the path with backslashes, place an 'r' in front of the directory path (e.g., r").

[5]:
```
path='/run/media/kechu/kECHU/PC_lab_peque/opencv/OpenCV_actualizado/jose/'
if video_number == 1:
    roi1_center=(-1,-1)
    last_roi1_center = (-1,-1)
    roi1_center_prj=(-1,-1)
else:
    last_roi1_center = np.load(path+'last_roi1_center.npy',mmap_mode=None)
    roi1_center=last_roi1_center
    roi1_center_prj=project_XY(roi1_center)

video_duration = video_end_datetime - video_start_datetime
print(video_duration)

seconds = video_duration.seconds
print(seconds, frames_number)

seconds_by_frame = round(seconds / frames_number, 1)
print(seconds_by_frame)
img_num = 0
img_time = video_start_datetime
start_time = time.time()
```

```
0:20:27
1227 1227.0
1.0
```

---

7 6. Define size of arena (Tank ROI's)

The following code is used to exclude internal portion/s of the field of view from the analysis. The user can draw a rectangle that defines the minimum region of your arena that contains your whole animal modifing the parameters in the brackets.

If the user do not use a camera positioned precisely (i.e. exactly at a right angle horizontally or perpendicular to the experimental arena), the user here have to select the corresponding tank which was setting using modified correspondig coordenates in 4 section.

El siguiente código se utiliza para excluir del análisis la/s porción/es interna/s del campo de visión. El usuario puede dibujar un rectángulo que defina la región mínima de su campo que contenga su animal completo modificando los parámetros en los paréntesis.

Si el usuario no utiliza una cámara posicionada con precisión (es decir, exactamente en ángulo recto horizontalmente o perpendicularmente a la arena experimental), el usuario aquí tiene que seleccionar el tanque correspondiente que fue establecido usando coordenadas correspondientes modificadas en la sección 4.

[6]:
```
# Define Tank ROI's
if tank == 1:
    roi1 = np.array([[200,900],[1000,1000],[1000,200],[200,200]], np.int32)
    roi1 = roi1.reshape((-1,1,2))
```

---

8 7. Ouput results in .csv format

The ouput dataframe should contein five raws:

"Prfmc_Msr_1a" The variable "Prfmc_Msr_1a" is difficult to type and you may have to remember that "Prfmc_Msr" was an abbreviation for Performance Measure.

"PerfMeasure1a." The variable "Prfmc_Msr_1a" is difficult to type and you may have to remember that "Prfmc_Msr" was an abbreviation for Performance Measure.

[ ]:
```
while(cap.isOpened()):
    ret, frame = cap.read()
    if frame is None:
        print('fin')
        break
    img_num += 1
    print('img_number: '+str(img_num))
    img_time = time_calculation(img_num)
    print('img_time: '+str(img_time))
    gray_img = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    print('image size: '+ str(shape(gray_img)))
    # Apply frame to Background Subtractor Object
    fgmask = fgbg.apply(gray_img, learningRate=1.0/90) # Specific Learning rate␣
↪to Geryon longipes sp.
    #cv2.imshow('fgmask', fgmask)
    print('image size2: '+ str(shape(fgmask)))
    fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, kernel)
    fgmask = cv2.dilate(fgmask,kernel,iterations = 3)
    print('image size3: '+ str(shape(fgmask)))
    # Paint roi lines
    poly1 = cv2.polylines(frame,[roi1],True,(0,255,255),3)


    # Find contours into a frame
    contours, hierarchy = cv2.findContours(fgmask,cv2.RETR_TREE,cv2.
↪CHAIN_APPROX_SIMPLE)
```

```
    print('Found contours: ' + str(len(contours)))

    # Check each detected contour conditions
    for i, cnt in enumerate(contours):
        (x,y),radius = cv2.minEnclosingCircle(cnt)
        #center = (int(x),int(y))
        radius = int(radius)
        area = cv2.contourArea(cnt, True)
        hull = cv2.convexHull(cnt, True)
        hull_area = cv2.contourArea(hull)
        x,y,w,h = cv2.boundingRect(cnt)
        roi_aspect_ratio = float (w/h)
        gray_level_mean = int(np.mean(gray_img[y:y+h, x:x+w]))
        gray_level_median = int(np.median(gray_img[y:y+h, x:x+w]))

        if abs(area) > 200.0 and abs(area) < 90000.0:
            solidity = float(area)/hull_area
            if radius > 10: #and solidity > -4.0 and roi_aspect_ratio > 0.9 and␣
↪roi_aspect_ratio < 2.9:
                M = cv2.moments(cnt)
                if M['m00'] > 0.0:
                    center = (int(M['m10']/M['m00']),int(M['m01']/M['m00']))
                    if cv2.pointPolygonTest(roi1, center, False)==+1 or cv2.
↪pointPolygonTest(roi1, center, False)==0:
                        cv2.circle(frame,center, 10, 255, -1)
                        rect = cv2.rectangle(frame,(x,y),(x+w,y+h),255,3)
                        roi1_center=center
                        roi1_center_prj = project_XY(center)
                    else:
                        print("Warning: point out of ROI's")
                        # Show frame analysis results
    res_frame = cv2.resize(frame,(0,0),fx=0.5,fy=0.5)
    cv2.imshow('Original Frame',res_frame)

     # Out video
    # Write the flipped frame
    video_writer.write(frame)

# Write Data
    value0 = ('video_number_'+str(video_number)+'\t'+str(img_num)+'\t'+str(tank)+
              ␣
↪'\t'+str(roi1_center[0])+'\t'+str(roi1_center[1])+'\t'+str(roi1_center_prj[0])+'\t'+str(roi1_

    f.write(value0)

    # Save last centers values
    np.save(path+'last_roi1_center.npy',roi1_center)
```

```
    #Swap center values
    last_roi1_center=roi1_center

    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

# To check if video analysis is OK
print ("Analized Frames Number:",img_num, "Video Frames Number:",frames_number,␣
 ↪"Seconds by Frame Assigned:",seconds_by_frame, "Analysis OK:
 ↪",img_num==frames_number)
print ("--- %s seconds ---" % (time.time() - start_time))
# release system resources
cap.release()
video_writer.release()
```

9 Analysing and presenting data

9.1 Here we have to changes the interpretate, Python to R

The following code loads neccessary R packages.

Los siguientes códigos son para importar los paquetes de R necesarios

[ ]:
```
library(tidyverse)
library(ggplot2)
library(cowplot)
library(grid)
library(gridExtra)
```

10 8. Sets directory and setup video dimension

`setwd` : The directory path of the folder containing the data to be analysed (e.g., /Users/Videos/').

La ruta del directorio de la carpeta la cual contiene la data a analizar (e.g., /Users/Videos/').

setwd("~/Users/Videos/")

datos<- read.delim("data_5min.csv",header=FALSE)

head(datos)

11 9. Selecting the rows of interest to be analyzed

[ ]:
```
dt <-datos%>%
  select(V3 ,V6, V7)
names(dt)=c("number","x","y")
```

12 10. Calculing parameters

Whit the follow comands the user can calculate, cumulative distance, velocity over time and absolute and relative bearings across frames (e.g. “distance”, “abs.angle”, “rel.angle”, “velocity”)

[ ]:
```
parametros = function(ruta, fps = 1, box = 1, jitter.damp = 0.9) {
  time = seq(0, length.out = length(xpos), by = 1/fps)
  distance = c()
  abs.angle = c()
  rel.angle = c()
  velocity = c()
  count = 1
  for (j in 2:length(xpos)) {
    A = (xpos[j] - xpos[j - 1])
    B = (ypos[j] - ypos[j - 1])
    distance[count] = sqrt((A^2) + (B^2))
    abs.angle[count] = ifelse(distance[count] != 0 | count == 1, (atan2(A, B) *␣
↪(180/pi)) %% 360, abs.angle[count - 1])
    rel.angle[count] = ((((abs.angle[count] - abs.angle[count - 1]) %% 360) +␣
↪540) %% 360) - 180
    velocity[count] = distance[count]/(1/fps)
    count = count + 1
  }


  movement = matrix(ncol = 5, nrow = count)
  colnames(movement) = c("distance", "abs.angle", "rel.angle", "velocity",␣
↪"time")
  movement[, 1] = c(0, distance)
  movement[, 2] = c(0, abs.angle)
  movement[, 3] = c(0, abs.angle[1], rel.angle[2:length(rel.angle)])
  movement[, 4] = c(0, velocity)
  movement[, 5] = c(time)
  total.distance = round(sum(movement[,1], na.rm = TRUE))
  mean.velocity = mean(movement[,4], na.rm = TRUE)
  total.duration = movement[nrow(movement),5]
```

```
  return(list(position = cbind(xpos, ypos), fps = fps, movement = movement,␣
↪total.distance = total.distance, mean.velocity = mean.velocity, total.duration␣
↪= total.duration))

}


valores<-parametros(dt)

dat<-as.data.frame(valores)
```

13 11. Plotting the data of general movement patterns

Plot the animal's path across the arena with density clouds representing the relative time spent in a given location.

Traza el recorrido del animal por la arena con nubes de densidad que representan el tiempo relativo que pasa en un lugar determinado.

[ ]:
```
ggplot(aes(xpos, ypos), data = dat) +
  stat_density2d(aes(fill = ..density.., alpha = ..density..), geom = "tile",␣
↪contour = FALSE) +
  scale_fill_gradientn(colours = viridis::viridis(256)) +
  geom_path(na.rm = TRUE) +
  geom_point(aes(x = dat[1,1], y = dat[1,2]), size = 3, color = "blue") +
  geom_point(aes(x = dat[nrow(dat),1], y = dat[nrow(dat),2]), size = 3, color =␣
↪"red") +
  coord_fixed() +
  xlab("Distance (m)") +
  ylab("Distance (m)") +
  theme_bw() +
  theme(legend.position = "none", axis.title = element_text(size = 14, face =␣
↪"bold"), panel.grid.major = element_blank(), panel.grid.minor =␣
↪element_blank())
```

14 12. Plotting the data of the parameters

Four-panelled plot summarising the the animal's path, i.e. cumulative distance and velocity over time, and absolute and relative bearings across frames

Gráfico de cuatro paneles que resume la trayectoria del animal, es decir, la distancia y la velocidad acumuladas a lo largo del tiempo, y la orientación absoluta y relativa entre los fotogramas

[ ]:
```
dat$cumDistance<-cumDistance

plot1 = ggplot(dat, aes(x = time, y = cumDistance)) +
  geom_line(color = "grey25") +
```

```
  theme_bw() +
  xlab("") +
  ylab("Distancia (m)") +
  ggtitle("Distancia acumulativa")

plot2 = ggplot(dat, aes(x = time, y = velocity)) +
  geom_line(color = "grey85") +
  geom_line(aes(x = time, y = movingAverage(velocity, n = 13, centered = TRUE)),␣
 ↪color = "grey25") +
  theme_bw() +
  xlab("Tiempo (s)") +
  ylab("Velocidad (m/s)") +
  ggtitle("Velocidad")

dat = subset(dat, distance != 0)

plot3 = ggplot(dat, aes(x = abs.angle)) +
  geom_histogram(aes(y = ..count../sum(..count..)), breaks = seq(0, 360, 15)) +
  coord_polar(start = 0) +
  theme_bw() +
  scale_fill_brewer() +
  ylab("Proporción") +
  ggtitle("Cambios de dirección absoluta") +
  scale_x_continuous("", limits = c(0, 360), breaks = seq(0, 359, 45), labels =␣
 ↪seq(0, 359, 45))

plot4 = ggplot(dat, aes(x = rel.angle)) +
  geom_histogram(aes(y = ..count../sum(..count..)), breaks = seq(-180, 180, 15))␣
 ↪+
  coord_polar(start = pi) +
  theme_bw() +
  scale_fill_brewer() +
  ylab("Proporción") +
  ggtitle("Cambios de dirección relativa") +
  scale_x_continuous("", limits = c(-180, 180), breaks = seq(-180, 180, 45),␣
 ↪labels = seq(-180, 180, 45))

grid.arrange(plot1, plot3, plot2, plot4, ncol = 2)
```

[ ]: