



東南大學  
SOUTHEAST UNIVERSITY

人工智能学院

# 专业技能实训

授课教师：王洪松

邮箱：hongsongwang@seu.edu.cn



東南大學  
SOUTHEAST UNIVERSITY

人工智能学院

# 视频处理

- imageio: 处理图像的接口, 包括动画图像

```
import imageio.v2 as imageio
import glob
filenames = glob.glob('*.png' ) # 静态图片存放路径
save_name_gif = "generate.gif" # 转化的GIF图片名称
# fps 就是图片切换的频率, 越大越快。
fps = 1
# 播放次数, 0表示循环播放, 1表示播放1次, 2表示播放2次, 以此类推
loop = 0
# 存放图片的列表
pics_list = [imageio.imread(image_name) for image_name in filenames]
# 生成gif
imageio.mimsave(save_name_gif, pics_list, 'GIF', fps=fps, loop=loop)
```











# 视频处理：FFMPEG



- FFMPEG：专门用于处理音视频的开源库
- 下载编译好的FFMPEG可执行文件

<https://github.com/BtbN/FFmpeg-Builds/releases>

选择对应的电脑系统版本，对于windows系统，可以选

 ffmpeg-master-latest-linux64-gpl-shared.tar.xz	41.9 MB
 ffmpeg-master-latest-linux64-gpl.tar.xz	96.2 MB
 ffmpeg-master-latest-linux64-lgpl-shared.tar.xz	37 MB
 ffmpeg-master-latest-linux64-lgpl.tar.xz	86.9 MB
 ffmpeg-master-latest-linuxarm64-gpl-shared.tar.xz	34.2 MB
 ffmpeg-master-latest-linuxarm64-gpl.tar.xz	77.7 MB
 ffmpeg-master-latest-linuxarm64-lgpl-shared.tar.xz	30.9 MB
 ffmpeg-master-latest-linuxarm64-lgpl.tar.xz	71.4 MB
 ffmpeg-master-latest-win64-gpl-shared.zip	51 MB
 ffmpeg-master-latest-win64-gpl.zip	128 MB

将 “ffmpeg-master-latest-win64-gpl-shared\bin” 添加到电脑环境目录中（或把以下三个拷贝到程序运行目录）。

三个可执行文件：

ffmpeg、ffprobe、ffplay

# 视频处理：FFMPEG



- 转化格式
  - 抽取画面中的音频
  - 音频+视频合成
  - 视频分离成图片  
`ffmpeg -i video.mp4 -r 1 -f image2 out_%04d.jpg`  
(-r 帧率每秒钟转化1张)
  - 图片合成视频  
`ffmpeg -ss 15 -t 25 -i video.mp4 -c:v libx264 -c:a aac`
  - 截取视频/音频  
`-strict experimental clip.mp4`
  - 图片生成gif动图  
`ffmpeg -i input_image_%03d.png -r 5 output_test.gif`
- ss 15 : 从第15秒开始剪切视频。  
-t 25 : 指定剪切后的视频时长为25秒。  
-c:v libx264: 指定视频的编码格式为libx264格式。  
-c:a aac : 指定音频的编码格式为aac格式。  
-strict experimental : 安全处理。

# 生成视频文件



```
img1=cv2.imread('BaboonRGB.bmp')
img2=cv2.imread('LenaRGB.bmp')
rows1, cols1 = img1.shape[:2]
rows2, cols2 = img2.shape[:2]
print(img1.shape[:2], img2.shape[:2]))
# 根据小图像的大小, 在大图像上创建感兴趣区域
roi = img1[0:rows2, 0:cols2]
fourcc = cv2.VideoWriter_fourcc(*'XVID' )
out = cv2.VideoWriter('result.avi', fourcc, 80,
(cols2, rows2), True) # (width, height)
for a in range(0,1000,1):
    b=a/1000
    dst = cv2.addWeighted(roi, b, img2, 1-b, 0)
    out.write(dst)
out.release()
```



# 生成动图文件



```
import cv2
import imageio
```

```
img1 = cv2.imread('BaboonRGB.bmp')
img2 = cv2.imread('LenaRGB.bmp')
rows2, cols2 = img2.shape[:2]
roi = img1[0:rows2, 0:cols2]
```

```
duration = 5 # 动画长度为5秒
```

```
fps = 5 # 每秒5帧
```

```
images = []
```

```
for i in range(duration*fps):
```

```
    b = i/(duration*fps - 1)
```

```
    dst = cv2.addWeighted(roi, b, img2, 1 - b, 0)
```

```
    b, g, r = cv2.split(dst)
```

```
    images.append(cv2.merge((r, g, b)) )
```

```
imageio.mimsave( "res.gif" ,images, fps=fps, loop=1) # 播放1次, loop=0循环播放
```

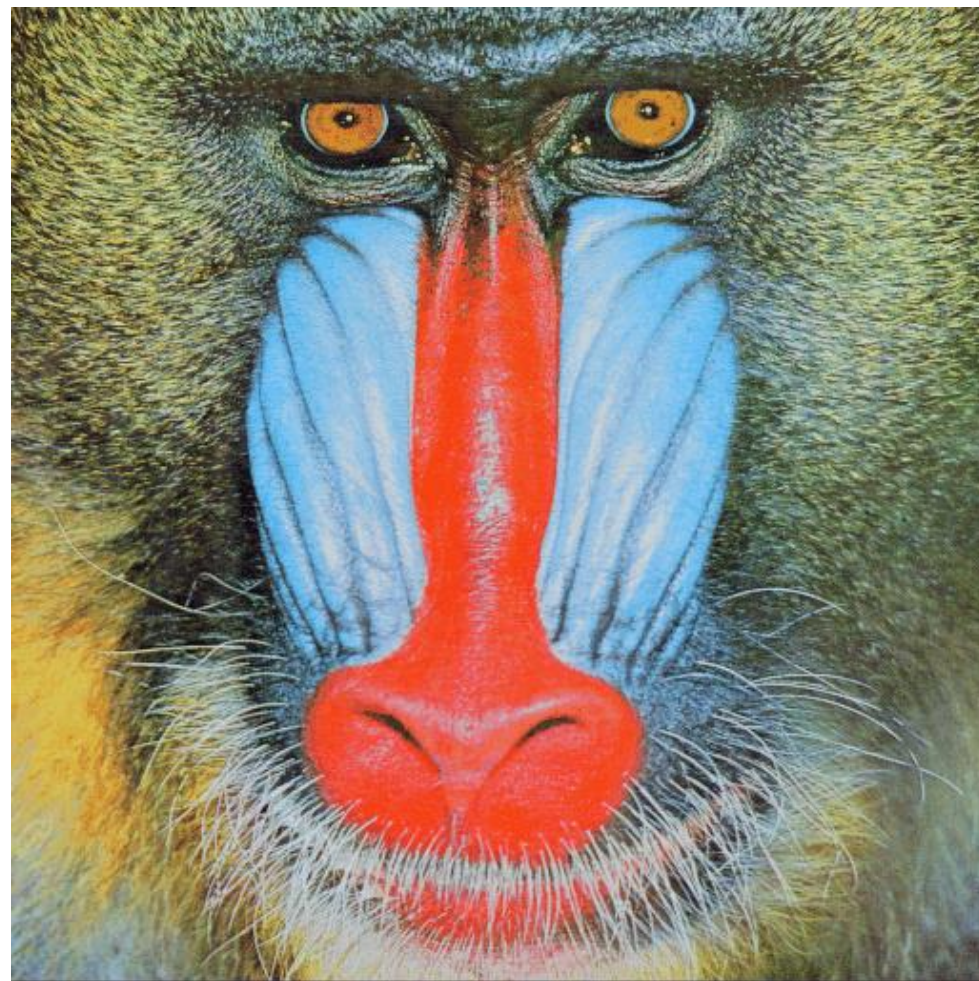




# 转场动画：平移（仿射变换）



```
import cv2
import numpy as np
import imageio
img1 = cv2.imread('BaboonRGB.bmp')
img2 = cv2.imread('LenaRGB.bmp')
img = np.hstack([img1, img2])
rows, cols = img1.shape[:2]
duration = 2 # 动画长度为5秒
fps = 5 # 每秒5帧
images = []
for i in range(duration*fps):
    percent = i/(duration*fps - 1)
    x = int(percent * cols)
    M = np.float32([[1, 0, -x], [0, 1, 0]])
    dst = cv2.warpAffine(img, M, (rows, cols))
    b, g, r = cv2.split(dst)
    images.append(cv2.merge((r, g, b)) )
imageio.mimsave("res.gif", images, fps=fps, loop=1)
```

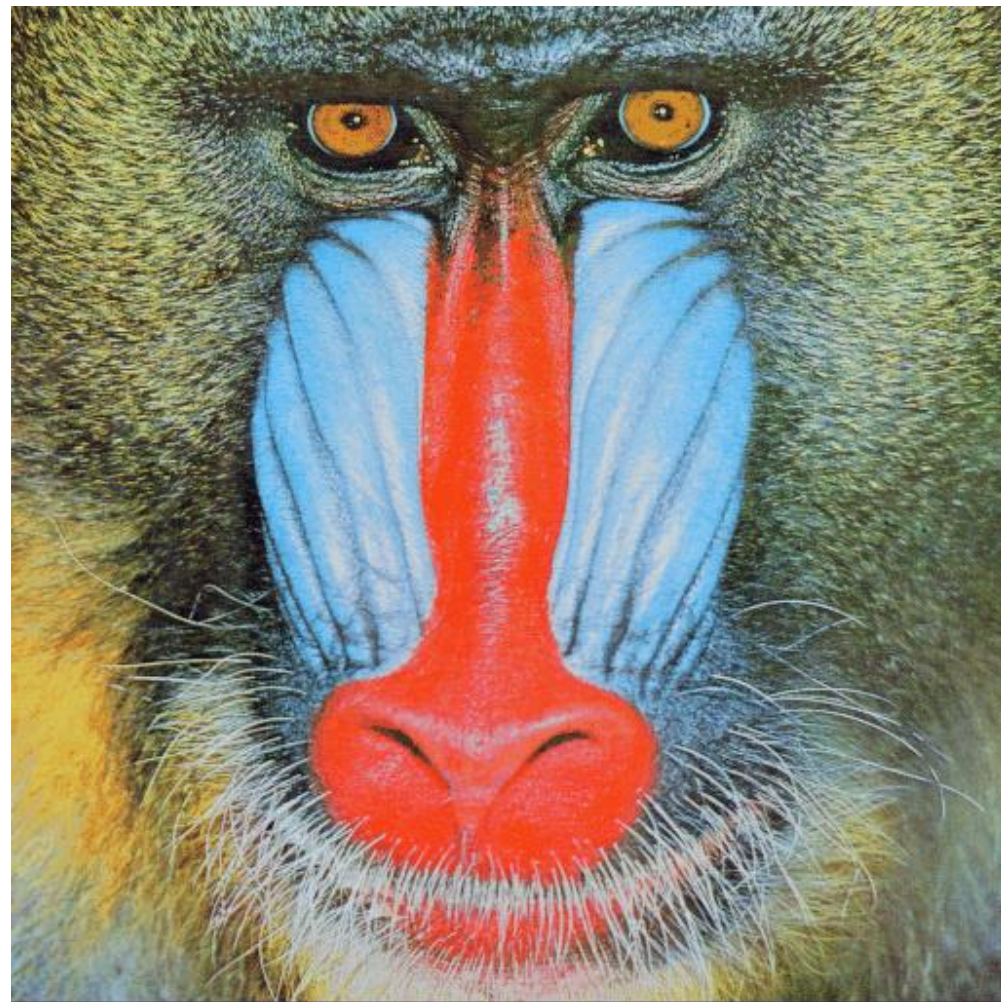




# 转场动画：控制变化速率



```
import cv2
import numpy as np
import imageio
img1 = cv2.imread('BaboonRGB.bmp')
img2 = cv2.imread('LenaRGB.bmp')
img = np.hstack([img1, img2])
rows, cols = img1.shape[:2]
duration = 2 # 动画长度为5秒
fps = 5
images = []
for i in range(duration*fps):
    n = 2
    percent = (i/(duration*fps - 1))**n
    x = int(percent * cols)
    M = np.float32([[1, 0, -x], [0, 1, 0]])
    dst = cv2.warpAffine(img, M, (rows, cols))
    b, g, r = cv2.split(dst)
    images.append(cv2.merge((r, g, b)) )
imageio.mimsave("res.gif", images, fps=fps, loop=1)
```



- 实现图像转场动画
  - 转场类型包括：闪黑、上移/下移/左移/右移、向上擦除/向下擦除/向左擦除/向右擦除、横向拉幕/竖向拉幕、旋转等。
  - 用多个滚动条控制转场类型、持续时间等，结果保存为动图。
- 实现滚动字幕（滚动文字动画）
  - 输入一张图像和一段文字作为该图像的字幕。
  - 把文字插入图像中，并在图像的下侧从左像向右滚动，一直到最后的一行文字出现并消失。
  - 采用变量分别控制文字的大小、滚动速率和滚动的方向（例如，从下往上滚动，即一行文字从图像的下侧居中出现，然后往上滚动到上侧并消失）。

# 专业技能实训：练习题



- 截取视频片段
  - 用滚动条分别控制开始时间、结束时间和输出帧率（FPS）。
- 把视频转化为动画图像
  - 基于采样的帧率（FPS）把视频转化成图像序列。
  - 选择感兴趣的多张图像制作动画图像。
- 基于单张图像制作动画图像
  - 输入包含前景目标和背景的一张图像。
  - 制作前景目标在背景中移动的动画图像。
- 基于单张图像制作局部运动的动画图像
  - 输入包含前景目标和背景的一张图像，假设前景是一个人。
  - 保持人的上身不动，制作人的部件的运动图像，比如挥手的动作。