1. 将图片保存在内存中，进行下一步的存取 2022.3.1
   1. 之前我们相机采集到的图片数据，先存储于硬盘中，然后在从硬盘中读取上传到mongoDB数据库中。数据流程为：相机——内存——硬盘——内存——mongoDB。
   2. 因为I/O一直是计算机工作的瓶颈，硬盘读写。DDR4内存大概是60GB/s的水平，nvme固态硬盘大概是2000MB/s的水，SATA 固态硬盘大概是450MB/s的水平  
      机械硬盘大概是100～150MB/s的水平。DDR4内存寻址时间：6ns左右，机械硬盘寻址时间：1/7200 ≈0.14ms。总而言之，内存读写速度比机械硬盘快大约4个数量级，比固态硬盘快约30倍。
   3. 我们使用python 的io.BytesIO模块，该模块可以在内存上实现类似硬盘上存取文件的功能，将采集到的数据直接保存在内存中，再上传至MongoDB。数据流程为：相机——内存——mongoDB
   4. Demo代码如下：

from io import BytesIO

import base64

import pymongo

from bson.objectid import ObjectId

import scipy.io as scio

from PIL import Image

import numpy as np

import matplotlib.pyplot as plt

client = pymongo.MongoClient()

db = client.test1

mydb = db.andor1

## 最终成果

load\_mat = scio.loadmat('./HandwrittenDigits.mat')

pic\_arraydata = load\_mat['X'][:,1]

pic\_arraydata = pic\_arraydata.reshape(16, 16)

# plt.imshow(pic\_arraydata)

image\_Andor = Image.fromarray(pic\_arraydata\*256) #可传输灰度图和彩色图

image\_Andor = image\_Andor.convert('L')

# 将Image对象存在内存BytesIO中

pic\_in\_memory = BytesIO()

image\_Andor.save(pic\_in\_memory, format='TIFF')

# 将BytesIO中的图片文件二进制编码后存入mongoDB

mydb.insert\_one({'file\_name': 'test\_pic.tiff', 'pic\_test1': base64.encodebytes( pic\_in\_memory.getvalue() )})

# 从mongoDB中读取文件

img = mydb.find\_one({'file\_name': 'test\_pic.tiff'})['pic\_test1']

img = base64.decodebytes(img) # 二进制解码

# 方式1：保存在本地

with open('test\_pic.tiff', 'wb') as pic\_file:

    pic\_file.write(img)

# 方式2：或者保存到内存，然后显示

pic\_in\_memory2 = BytesIO()

pic\_in\_memory2.write(img)

img\_from\_memory = Image.open(pic\_in\_memory2)

img\_from\_memory.show()

1. 测试
2. 在本机的linux虚拟机上跑IOC，运行程序需要的几个PV，包括 状态位PV（13ANDOR1:cam1:NumImagesCounter\_RBV）、数据PV（13ANDOR1:image1:ArrayData）、（IT:PSB1:GetCurrent）