```
#!/usr/bin/env python
# coding:utf-8
#···程序中注释掉的语句为利用cv提供的gui界面显示摄像头画面
import cv2
import rospy
from std_msgs.msg import Header
from sensor_msgs.msg import Image
from cv_bridge import CvBridge, CvBridgeError
import time
if __name__=="__main__":
    capture = cv2.VideoCapture(2) # 定义摄像头
    rospy.init_node('pub_camera_node', anonymous=True) #定义节点
    image_pub=rospy.Publisher('camera/image', Image, queue_size = 1) #定义话题
    rate = rospy.Rate(30) # 设置频率为30hz
    bridge = CvBridge()

    while not rospy.is_shutdown():    # Ctrl C正常退出，如果异常退出会报错device busy！
        start = time.time()
        ret, frame = capture.read()  #读取摄像头数据，capture.read()会返回两个值，一个是是否读取成功，一个是摄像头数据
        if ret: # 如果有画面再执行
            #frame = cv2.flip(frame,0)   #垂直镜像操作
            #frame = cv2.flip(frame,1)   #水平镜像操作
            #cv2.imshow("view", frame) #在窗口中显示摄像头数据
            #cv2.waitKey(1)          #等待，这个函数只有在cv的gui窗口下才有效
            ros_frame = Image()     #ROS消息类型实例化
            header = Header(stamp = rospy.Time.now())
            header.frame_id = "Camera"
            ros_frame.header=header
            ros_frame.width = 640
            ros_frame.height = 480
            ros_frame.encoding = "bgr8"
            ros_frame.step = 1920
            try:
                ros_frame = bridge.cv2_to_imgmsg(frame, encoding="bgr8") #图片格式转换，imgmsg_to_cv2可以将ROS图像转换为CV类型
            except CvBridgeError as e:
                print(e)
            image_pub.publish(ros_frame) #发布消息
            end = time.time()
            print("cost time:", end-start ) # 看一下每一帧的执行时间，从而确定合适的rate
            rate.sleep()
    capture.release()  #释放摄像头
    #cv2.destroyAllWindows()  #摧毁窗口
    print("quit successfully!")
```