摘　　要

近年来，随着计算机的飞速发展，其应用渗透在社会的各个领域，不断地深入日常生活，给寻常人带来了意想不到的便利。这些年大火的 AR 测距，凭借其测量精准，数据可靠，消耗资源少等显著特点，广泛地应用在生活，工程测量等众多领域。通过相关装置对物体(特别是对于非静止物体而言)进行测量可代替人工实现较多复杂测量或是对较大待测物体的测量，以此减轻人力物力成本。

本次大赛完成对动态非全貌物体的全数据测量。测得物体的长度精度较高，误差范围仅仅为在 0.5~1.5cm 之间,运动方向同拍摄方向夹角精度在 5° 范围内，但同时也存在一定的波动性。

关键词：opencv；树莓派 4B；互联网交换机；测距测角度

目录

基于互联网的摄像测量系统（D 题）
【本科组】

一、设计方案工作原理

1、预期实现目标定位

本系统设计之初用于完成赛题任务的基本功能，利用两个摄像节点上传实时的画面，画面中不出现系统全貌，并有红色方框对运动激光笔进行实时框选，鱼线悬挂的激光笔在水平方向上做来回摆幅小于 10cm 的往复圆弧运动，通过摄像头捕捉的画面，利用 OpenCV 对视频画面进行分析，计算出悬挂的鱼线长度，并在计算结果反馈到显示器的同时进行声光播报提示。

竞赛后期在完成基本功能之后，对发挥部分进行研究，30 秒内完成当运动水平夹角为$\theta = 0^\circ$和 $\theta = 90^\circ$时鱼线长度的测量和运动角度θ的测量。

2、技术方案分析比较

方案一：采用 STM32 单片机和以太网。

在初始阶段发现 STM32 的核心 MCU 适合做一些运算比较简单的中小型程序，不适合复杂的运算或是跑系统，处理缓慢甚至死机，优势在于实时性高，无需等待操作系统启动，一上电就能开始跑程序。

方案二：采用树莓派和交换机组成系统。

采用树莓派做系统，树莓派是目前非常流行的一款小型计算机系统，一款开源硬件的成品开发板，上面有 CPU、内存，以分立的芯片的形式存在，且性能远超单片机。将电脑机箱里的大部分东西都集成到了一块电路板上的微型电脑，接上显示器鼠标键盘等东西和电脑几乎没有实质上的区别，操作系统不一样是基于 Linux 的系统。同时支持本地上编程、编译、运行，在向原有程序增删功能时，或切换新任务时，不用像单片机一样烧写程序。其强大的芯片更适合于做复杂的运算，比如图像采集，处理，深度学习和识别等。

综合以上两种方案，选择方案二。

3、系统结构工作原理

MT-2602U 摄像头捕捉物体运动画面后，摄像节点与树莓派相连上传串流，树莓派 4B 上跑 OpenCV 对画面进行处理调节灰度或是长曝光处理，红框框定运

动物体，建立出坐标系用于测量相关位置点坐标进行后期处理可以求出鱼线长度 l 和水平夹角 θ 。树莓派通过网线与交换机相连，在显示器上实时显示拍摄画面。终端节点相连的显示器会将测量所得结果显示出来。

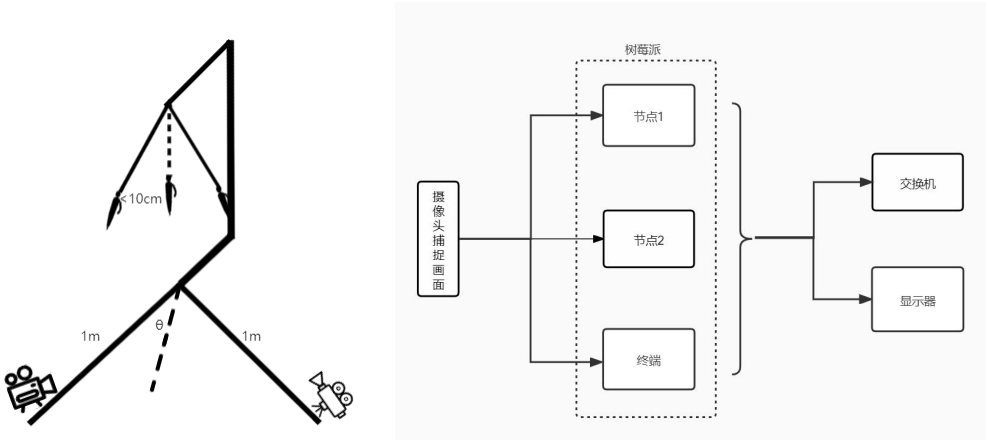


图 1.1 系统总体框图

4、功能指标实现方法

系统在树莓派 GPIO 口上引出外接开关用以对系统整体进行控制，起到一键启动功能的作用。摄像头拍摄画面通过树莓派全局分析处理计算出各个任务点所需要的结果。交换机起到网络互联的作用，网线网口连接后，形成两个拍摄节点显示和一个终端节点显示激光笔的运动状态。OpenCV 对画面处理框选激光笔，经树莓派处理后将相关测量数据反馈到终端显示器上。

5、测量控制分析处理

透明鱼线长度的测量：在求其长度时，由于题目要求摄像头不可以直接拍摄全部透明鱼线和激光笔轨迹路线，所以无法直接通过挂线处和激光笔头测得之间的差值。但通过摄像头可以锁定激光笔的运动路径，激光笔在运动过程中是一条弧形，对画面进行灰度值调节，使激光笔更易被观测到。

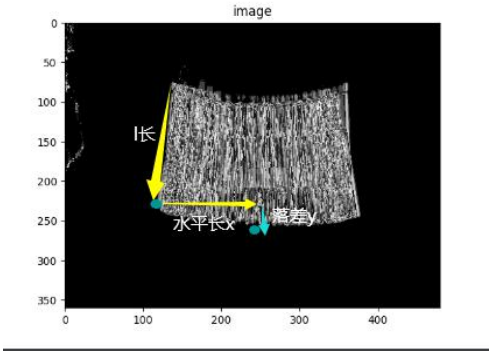


图 1.2 激光笔运动灰度图

从初始落点运动到最低点之间存在着一个高度差，摄像画面捕捉之后可以得到这个落差值记为y，而水平方向上的距离是人为拉取可以记为x。设要求的鱼线长度为l，通过勾股可以求得l：

$$l = \frac{1}{2} \bullet (\frac{x^2}{y} + y)$$

水平夹角的测量：利用扇形面积与角度占比进行求解。

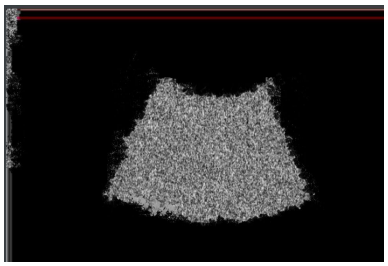


图 1.3 扇形运动面的面积

在测试时，将激光笔拉起，与地面正方形夹角成$\theta = 0^\circ$和 $\theta = 90^\circ$两种情况，分别计算出拍摄所得到的激光笔运动形成的扇形面积S_1和S_2。由于底面夹角为直角，所以可计算出一度的面积为：

$$\text{设系数}：k = \frac{S_1 + S_2}{\pi / 2}$$

接下来只需任意移动激光笔(允许范围内)使其运动，摄像捕捉画面计算来回摆动时的激光笔形成的面积记为S，可以得到此时激光笔运动时的夹角为：

$$\theta = \frac{S}{k}$$

二、核心部件电路设计

1、关键器件性能分析

摄像头模块：高清红外防水，最新的 DSP 技术图像稳定，自动降噪。

图 2.3 摄像头模块

树莓派 4B：内芯强大，区别于单片机，可用于复杂运算，性能较强，CPU 速率高，外接 GPIO 口可对功能进行拓展。整体功耗小，重量轻，可板载系统并进行在线编辑，十分适用于图像采集处理和识别。

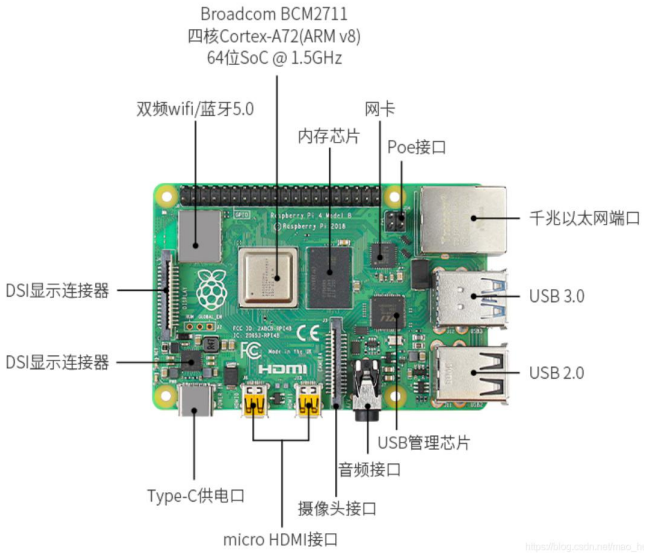


图 2.1 树莓派实物图

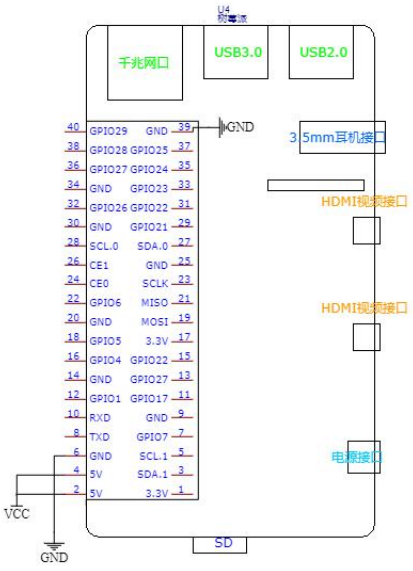


图 2.2 树莓派原理结构图

2、电路设计仿真

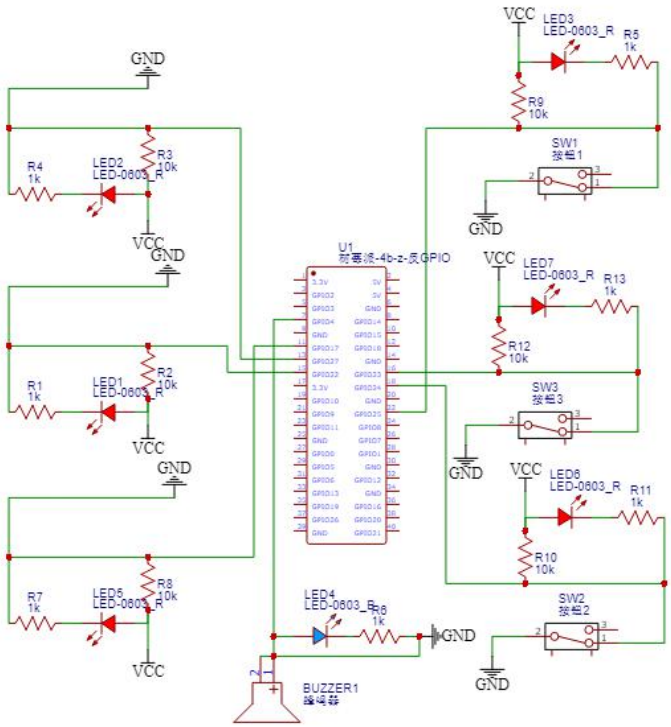


图 2.4 树莓派外接电路结构图

3、电路实现调试测试

上述电路图较为简单，在树莓派外利用一块面包板，按电路图搭建电路于树莓派 GPIO 口相连，进行上电测试。系统要求一键启动功能，设计按键功能，在按下对应的按键后，相应的功能将会按序执行，完成所有操作。蜂鸣器和 LED 灯在数据处理完成后会发生电平跳变，LED 灯亮蜂鸣器鸣，提示操作完成。

4、关键电路驱动接口

Micro HDMI 接口，USB2.0，USB3.0 接口，Type-C 接口，千兆网口，SD 卡接口。

三、系统软件设计

1、系统总体工作流程

首先，通过两个摄像头与终端之间的节点串流。进行数据交换，同时调用几个画面，然后锁定对应的测量目标，对其进行曝光，显示在直角坐标系中，最后，通过相应的算法来求鱼线的长度 l 以及角度 θ 。

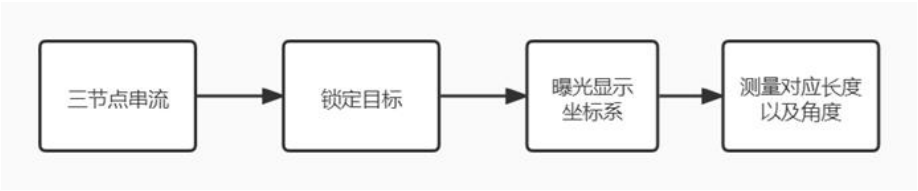


图 3.1　系统处理流程

2、主要模块程序设计

利用 python/OpenCV 实现摄影中的图像长曝光；提取特征区域，进行图片通道分离；查找图像的轮廓，返回图像的所有轮廓，从而找到所有大的联通区域；取出最大的解锁边缘，根据最大的轮廓来读取外包圆。

3、关键模块程序清单

见附件。

四、竞赛工作环境条件

1、设计分析软件环境

OpenCV	跨平台计算机视觉和机器学习软件库，同时提供 python, Matlab 等

	语言接口，可以实现图像处理和计算机视觉方面的很多通用算法。
Pycharm	一种 Python IDE 集成开发环境，用于支持 Django 框架下的专业 web 开发。可以支持编码协助、项目代码导航、代码分析、集成的单元测试和自定义可扩展功能。
VNC viewer	一款远程控制的软件，一般用于远程解决电脑故障或软件调试。体积小巧轻便，功能丰富齐全，使用途中不会出现任何卡顿、黑屏等情况，连接速度快而稳定，远程控制体验很好。

2、仪器设备硬件平台

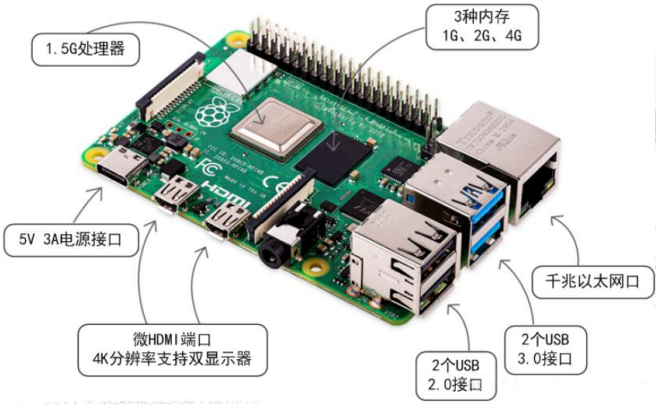


图 4.1 树莓派结构图

树莓派 4B： 实质上是一台迷你的嵌入式计算机，树莓派使用 TF 卡作为“硬盘”。利用树莓派可以编辑文档、浏览网页、玩游戏、播放视频、播放音频等，还可以利用树莓派制作智能小车、示波器、电子相框、家庭影院、相机等。同时支持很多种操作系统 Raspbian、Ubuntu Mate、Windows 10 IoT 等。树莓派 4B 比以前的树莓派运行速率要快得多，拥有多个不同速率的 USB 接口，能够以 60 Hz 输出 4K 视频并双显示器支持。

图 4.2 交换机图

交换机：用于连接多台设备，让其具备网络互通的条件，是一个扩大网络的器材，能为子网络中提供更多的连接端口具有性能价格比高、高度灵活、相对简

单、易于实现等特点。

3、配套加工安装条件

图片详见附件。

1）实验室具有钢材用于搭建系统骨架；

2）热熔胶固定大小物体；

3）亚克力板板载处理部分；

4）显示器、网线、HDMI 转换线等；

5）树莓派、交换机、摄像头、3D 打印机等。

4、前期设计使用模块

树莓派 4B、MT-2602U 摄像模块、激光笔、交换机和外接蜂鸣器 LED 电路。

五、作品成效总结分析

1、系统测试性能指标

表 5-1 鱼线长度测量值

测量次数	系统测量值/cm	鱼线实际值/cm	高度坐标 Y	误差 α /cm
第一次	133.56	132.20	483	1.36
第二次	114.25	113.80	412	0.45
第二次	91.48	90.20	332	1.28
第三次	81.45	79.90	281	1.55
第四次	**69.60**	**68.80**	**210**	**0.8**
第五次	58.64	57.30	151	1.34
第六次	**47.70**	**45.60**	**95**	**2.1**

2、成效得失对比分析

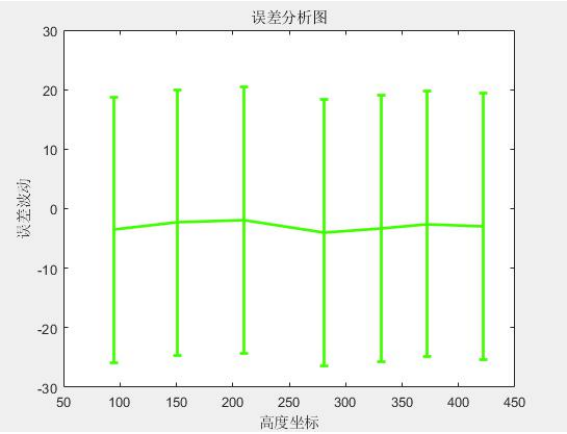


图 5.2 数据误差分析图

利用 Matlab 进行误差分析，观察数据误差整合，比较得出系统测得的数据普遍偏高，基本在误差允许范围内可调。

3、创新特色总结展望

该赛题作品能够较为精确的测量被测物体的长度，测量时长可自定义。

忙碌而又充实的四天三夜的电赛马上拉上帷幕，通过这次比赛，我们受益匪浅。这次比赛不仅考验了我们的理论知识，更加考验了我们的动手能力，我们也懂得了理论与实践结合的重要性。光具备实践与理论知识还远远不够，这次比赛还需要团队成员之间的明确分工，默契配合。这次比赛也充分考验了我们的毅力。我们还需要具备一颗持之以恒的心，从选题到方案论证再到器件的购买，到最后的调试，都是每个成员共同努力的成果。在大家的共同努力下，我们最终完成了本次比赛。

六、参考资料及文献

[1] 王艳红.基于 OpenCV 的运动目标检测与跟踪算法的研究[D].杭州电子科技大学. 2013-12-01

[2] 熊倩.基于 OpenCV 的运动目标检测与跟踪算法研究[D].武汉理工大学.2014-04-01

[3] 韩宇，张磊，吴泽民，胡磊. 基于嵌入式树莓派和 OpenCV 的运动检测与跟踪系统[D].电视技术.2017-02-17

[4] 陈凯悦.基于图像处理的机器视觉测距系统的设计与实现[D].河北科技大学.2017-12-01

[5] 刘东东.基于 USB 摄像头的单目视觉测距技术的研究[D].大连理工大学.2014-05-15

[6] 岳兵.基于 OpenCV 的目标检测与跟踪算法的研究与实现[D].安徽大学.2016-05-51

七、附件

关键模块程序清单

1、Analysis 程序段

```
GPIO.setmode(GPIO.BOARD)
GPIO.setup(37, GPIO.IN)
GPIO.setup(35, GPIO.IN)
GPIO.setup(33, GPIO.IN)

ser = serial.Serial('/dev/ttyS0', 9600,timeout=10)
coefficient =
[0.3097,0.3241,0.3365,0.3517,0.3645,0.3825,0.3954,0.4093,0.4268,0.4457,0.4695,0.4877,0.5077,
0.5287,0.5287,0.5287,0.5287,0.5287,0.5287,0.5287,0.5287,0.5287,0.5287,0.5287,0.5287]

tmp = 0
Distance = 0.0
decade =0
X_angle=0
Y_angle=0

def distance():
    global tmp
    all = 0.0
    global Distance
    global decade
    ser.flushInput()
    tmp_start='0,0,0,0,1'
    ser.write(tmp_start.encode())
    data = ser.read(90)
    time.sleep(2)
    ser.flushInput()
    #print(data)
    str_num=(re.findall(r"\d+\.?\d",data.decode('utf-8')))
    #print(str_num)
    long = len(str_num)
    for num in range(0,long):
        if(float(str_num[num])>150 and float(str_num[num])<350):
            all += float(str_num[num])
            tmp+=1
            #print(tmp,all)
    Distance=all/tmp
    #print("距离"+"%.2f"%(Distance))
```

```
    flag=0
    if (Distance>200):
        flag=Distance-200
    if (Distance<200):
        flag=200-Distance
    decade = int(flag /10)
    #print(decade)
    tmp = 0
    dis_relay()
    Distance =0.0

def distance_ball():
    global tmp
    all = 0.0
    global Distance
    global decade
    ser.flushInput()
    tmp_start='0,0,0,0,1'
    ser.write(tmp_start.encode())
    data = ser.read(90)
    time.sleep(2)
    ser.flushInput()
    #print(data)
    str_num=(re.findall(r"\d+\.?\d",data.decode('utf-8')))
    #print(str_num)
    long = len(str_num)
    for num in range(0,long):
        if(float(str_num[num])>150 and float(str_num[num])<350):
            all += float(str_num[num])
            tmp+=1
            #print(tmp,all)
    Distance=all/tmp
    #print("距离"+"%.2f"%(Distance))
    flag=0
    if (Distance>200):
        flag=Distance-200
    if (Distance<200):
        flag=200-Distance
    decade = int(flag /10)
    tmp = 0
    dis_relay()
    Distance =0.0
```

```
class ShapeAnalysis:

    def __init__(self):
        self.shapes = {'triangle': 0, 'rectangle': 0, 'polygons': 0, 'circles': 0}

    def analysis(self, frame):
        h, w, ch = frame.shape
        global coefficient
        global Distance
        result = np.zeros((h, w, ch), dtype=np.uint8)
        #print("start ")
        gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
        ret, binary = cv.threshold(gray, 0, 255, cv.THRESH_BINARY_INV | cv.THRESH_OTSU)

        distance()
        #print("距离"+"%.2f"%(distance_relay+5))

        contours, hierarchy = cv.findContours(binary, cv.RETR_EXTERNAL,
cv.CHAIN_APPROX_SIMPLE)
        for cnt in range(len(contours)):
            cv.drawContours(result, contours, cnt, (0, 255, 0), 2)
            epsilon = 0.01 * cv.arcLength(contours[cnt], True)
            approx = cv.approxPolyDP(contours[cnt], epsilon, True)

            corners = len(approx)
            shape_type = ""
            if corners == 3:
                count = self.shapes['triangle']
                count = count+1
                self.shapes['triangle'] = count
                shape_type = "三角形"
            if corners == 4:
                count = self.shapes['rectangle']
                count = count + 1
                self.shapes['rectangle'] = count
                shape_type = "正方形"
            if corners >= 10:
                count = self.shapes['circles']
                count = count + 1
                self.shapes['circles'] = count
                shape_type = "圆形"
            if 4 < corners < 10:
                count = self.shapes['polygons']
```

```
                count = count + 1
                self.shapes['polygons'] = count
                shape_type = "正方形"


        global X_angle
        global Y_angle

        p = cv.arcLength(contours[cnt], True)
        area = cv.contourArea(contours[cnt])

        if(p>200):
            mm = cv.moments(contours[cnt])
            cx = int(mm['m10'] / mm['m00'])
            cy = int(mm['m01'] / mm['m00'])
            X_angle=cx
            Y_angle=cy
            cv.circle(result, (cx, cy), 3, (0, 0, 255), -1)

        if (p > 200 ):
            if (shape_type == "圆形"):
                if((math.sqrt(4*area/math.pi)*coefficient[decade])<28):
                    continue
                else:
                    print(" 直 径 : %.3f,  中 心 点 : %s  形 状 : %s " %
( math.sqrt(4*area/math.pi)*coefficient[decade], (cx,cy), shape_type))
            if (shape_type == "正方形"):
                if(((math.sqrt(area))*coefficient[decade])<26                    and
((math.sqrt(area))*coefficient[decade]) > 15):
                    print(" 边 长 : %.3f,    中 心 点 : %s  形 状 :  三 角 形   " %
((math.sqrt(area / (math.sqrt(3) / 4)))*coefficient[decade], (cx,cy)))
                else:
                    print(" 边 长 : %.3f,     中 心 点 : %s  形 状 : %s " %
((math.sqrt(area))*coefficient[decade], (cx,cy), shape_type))
            if (shape_type == "三角形"):
                if(((math.sqrt(area / (math.sqrt(3) / 4)))*coefficient[decade])<20):
                    continue
                print(" 边 长 : %.3f,    中 心 点 : %s  形 状 : %s " % ((math.sqrt(area /
(math.sqrt(3) / 4)))*coefficient[decade],(cx,cy), shape_type))

        Distance = 0.0

        return self.shapes
```

```
    def analysis_ball(self, frame):
        h, w, ch = frame.shape
        global coefficient
        global Distance
        result = np.zeros((h, w, ch), dtype=np.uint8)
        #print("start\n")
        gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
        ret, binary = cv.threshold(gray, 0, 255, cv.THRESH_BINARY_INV | cv.THRESH_OTSU)

        distance_ball()
        football =0
        basketball = 0
        volleyball = 0
        contours, hierarchy = cv.findContours(binary, cv.RETR_EXTERNAL,
cv.CHAIN_APPROX_SIMPLE)
        #print("start")
        for cnt in range(len(contours)):
            cv.drawContours(result, contours, cnt, (0, 255, 0), 2)

            global X_angle
            global Y_angle

            p = cv.arcLength(contours[cnt], True)
            area = cv.contourArea(contours[cnt])
            if(p>30):
                mm = cv.moments(contours[cnt])
                cx = int(mm['m10'] / mm['m00'])
                cy = int(mm['m01'] / mm['m00'])
                X_angle=cx
                Y_angle=cy
                cv.circle(result, (cx, cy), 3, (0, 0, 255), -1)
                color = frame[cy][cx]
                #print(cx, cy)
                color_str = "(" + str(color[0]) + ", " + str(color[1]) + ", " + str(color[2]) + ")"
                #print(color_str)
                b,g,r =np.transpose(color)[0],np.transpose(color)[1],np.transpose(color)[2]
                b,g,r =int(b),int(g),int(r)
                #print(r,g,b)

                if(r+g+b<255 or r+g+b>550):
                    football +=1
                elif(r>=(g+b)/2):
```

```
                    basketball +=1
                else:
                    volleyball +=1

        if(basketball>football and basketball > volleyball):
            print("篮球")
        elif(football>basketball and football >=volleyball):
            print("足球")
        else:
            print("排球")
        #print(basketball,football,volleyball)
        football =0
        basketball = 0
        volleyball = 0
        gougu=math.sqrt((cx-140)**2+(cy-140)**2)
        dis_ball= math.sqrt((gougu*coefficient[decade])**2+(distance_relay)**2)
        print("距离 : "+"%.2f" % (dis_ball-11))

        Distance = 0.0
        return self.shapes

def angle_send_ball():
    global distance_relay
    #print(distance_relay)
    x=angle.getangle_x(X_angle-120,distance_relay)
    y=angle.getangle_y(Y_angle-120,distance_relay)
    tmp2=str(y)+','+str(x)+',1,1,0'
    #print(tmp2)
    ser.write(tmp2.encode())

while 1:
    while GPIO.input(35) == GPIO.LOW:
        if __name__ == "__main__":
            print("识别开始：\n")
            cut.cut_img()
            sharp.sharp_img()
            src = cv.imread("result.png")
            ld = ShapeAnalysis()
            ld.analysis(src)
            angle_send()
            print("结束")
            time.sleep(5)
            ser.flushInput()
```

```
            print("\n")

    while GPIO.input(37) == GPIO.LOW:
        if __name__ == "__main__":
            print("识别开始：\n")
            cut.cut_img()
            sharp.sharp_img()
            src = cv.imread("result.png")
            ld = ShapeAnalysis()
            ld.analysis(src)
            ser.flushInput()
            print("距离 : "+"%.2f" % (distance_relay+4))
            print("结束")
            finish_light()
            print("\n")

    while GPIO.input(33) == GPIO.LOW:
        if __name__ == "__main__":
            print("识别开始：\n")
            cut.cut_ball()
            sharp.sharp_ball()
            src = cv.imread("result.png")
            ld = ShapeAnalysis()
            ld.analysis_ball(src)
            angle_send_ball()
            print("结束")
            time.sleep(5)
            print("\n")
```

2、color locking

```
import cv2
import numpy as np

def nothing(value):
    pass
hsv_l = np.array([104,81,60])
hsv_u = np.array([255,255,255])
# def setup_trackbars():
#     cv2.namedWindow("Trackbars",0)
#
#     for i in ["MIN","MAX"]:
#         v =0 if i =="MIN" else 255
```

```
#
#               for j in 'HSV':
#                       cv2.createTrackbar("%s_%s" % (j,i),"Trackbars",v,255,nothing)
#
# def get_trackbar_values():
#       values =[]
#
#       for i in ["MIN","MAX"]:
#               for j in 'HSV':
#                       v =cv2.getTrackbarPos("%s_%s" % (j,i),"Trackbars")
#                       values.append(v)
#       return values

RTSP_URL = 'http://192.168.2.145:8080/?action=stream'    # your camera's rtsp url

#初期化 USB 摄像头
cap = cv2.VideoCapture(RTSP_URL)
while True:
    ret,frame =cap.read()
    if not ret:
        break
    frame =cv2.resize(frame,(480,360))
    img_hsv    =cv2.cvtColor(frame , cv2.COLOR_BGR2HSV)
    # hl,sl,vl,hu,su,vu =get_trackbar_values()
    mask = cv2.inRange(img_hsv,hsv_l,hsv_u)
    #先复制一份
    mask_morph =mask.copy()
    # 函数的第一个参数表示内核的形状
    # 矩形：MORPH_RECT;
    # 交叉形：MORPH_CROSS;
    # 椭圆形：MORPH_ELLIPSE;
    # ,内核的尺寸以及锚点的位置
    kernel =cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
    # 开运算
    mask_morph =cv2.morphologyEx(mask_morph,cv2.MORPH_OPEN,kernel)
    # 闭运算
    mask_morph =cv2.morphologyEx(mask_morph,cv2.MORPH_CLOSE,kernel)
    output =cv2.bitwise_and(frame,frame,mask=mask_morph)


    # 查找图像的轮廓，返回图像的所有轮廓，从而找到所有大的联通区域，-2 是取方法返回中的第二个参数
    cnts=cv2.findContours(mask_morph,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)[-2]
```

```
    if cnts:
        #   取出最大的解锁边缘，解锁条件 key 是面积
        c =max(cnts,key=cv2.contourArea)
        # 根据最大的轮廓来读取外包圆
        ((x,y),raduis) = cv2.minEnclosingCircle(c)
        # 计算轮廓的矩
        M = cv2.moments(c)
        # 计算轮廓的重心
        center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))
        # # 计算坐标
        # cx = int(M['m10'] / M['m00'])    # 求 x 坐标
        # cy = int(M['m01'] / M['m00'])    # 求 y 坐标
        # 矩形
        w, h =    25, 25
        # 只处理尺寸足够大的轮廓
        if raduis > 5:
            # # 画出最小外接圆
            # cv2.circle(frame, (int(x), int(y)), int(raduis), (0, 255, 255), 2)
            # 矩形
            cv2.rectangle(frame,                                    (int(x)-int(raduis),
int(y)-int(raduis),int(2*raduis),int(2*raduis)), color=(0,0, 255), thickness=1)    # BGR
            # 画出重心
            # cv2.circle(frame, center, 5, (0, 0, 255), -1)

        # 如果满足条件，就画出圆，画图函数，frame，中心，半径，颜色，厚度
        # if raduis>10:
        #         cv2.circle(frame,(int(x),int(y)),int(raduis),(0,255,0),6)
    # print("重心坐标是", cx, ",", cy, ")")
    cv2.imshow("original",frame)



    # cv2.imshow("mask",mask)
    # cv2.imshow("mask_morph",mask_morph)
    # cv2.imshow("output",output)

    if cv2.waitKey(300) & 0xFF is ord("q"):
        break


cap.release()
cv2.destroyAllWindows()
```

3、数据测量

```
hsv_l = np.array([104,81,60])
hsv_u = np.array([255,255,255])
# def setup_trackbars():
#         cv2.namedWindow("Trackbars",0)
#
#         for i in ["MIN","MAX"]:
#                 v =0 if i =="MIN" else 255
#
#                 for j in 'HSV':
#                         cv2.createTrackbar("%s_%s" % (j,i),"Trackbars",v,255,nothing)
#
# def get_trackbar_values():
#         values =[]
#
#         for i in ["MIN","MAX"]:
#                 for j in 'HSV':
#                         v =cv2.getTrackbarPos("%s_%s" % (j,i),"Trackbars")
#                         values.append(v)
#         return values

RTSP_URL = 'http://192.168.2.68:8080/?action=stream'    # your camera's rtsp url

#初期化 USB 摄像头
cap = cv2.VideoCapture(RTSP_URL)
if( cap.isOpened() ):
  print("cap.isOpened() ")
# setup_trackbars()
Y_min=[]
for i in range(0,120):
        ret, frame = cap.read()
        if not ret:
                break
                print("2")
        frame = cv2.resize(frame, (480, 360))
        img_hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
        # hl,sl,vl,hu,su,vu =get_trackbar_values()
        mask = cv2.inRange(img_hsv, hsv_l, hsv_u)
        # 先复制一份
        mask_morph = mask.copy()
        # 函数的第一个参数表示内核的形状
        # 矩形：MORPH_RECT;
        # 交叉形：MORPH_CROSS;
```

```
    # 椭圆形：MORPH_ELLIPSE;
    #,内核的尺寸以及锚点的位置
    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5))
    # 开运算
    mask_morph = cv2.morphologyEx(mask_morph, cv2.MORPH_OPEN, kernel)
    # 闭运算
    mask_morph = cv2.morphologyEx(mask_morph, cv2.MORPH_CLOSE, kernel)
    output = cv2.bitwise_and(frame, frame, mask=mask_morph)

    # 查找图像的轮廓，返回图像的所有轮廓，从而找到所有大的联通区域，-2 是取方法返回中的第二个参数
    cnts = cv2.findContours(mask_morph, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)[-2]
    if cnts:
        #  取出最大的解锁边缘，解锁条件 key 是面积
        c = max(cnts, key=cv2.contourArea)
        # 根据最大的轮廓来读取外包圆
        ((x, y), raduis) = cv2.minEnclosingCircle(c)
        # 计算轮廓的矩
        M = cv2.moments(c)
        # 计算轮廓的重心
        center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))
        # 计算坐标
        cx = int(M['m10'] / M['m00'])  # 求 x 坐标
        cy = int(M['m01'] / M['m00'])  # 求 y 坐标
        # 矩形
        w, h = 25, 25
        # 只处理尺寸足够大的轮廓
        if raduis > 5:
            ## 画出最小外接圆
            # cv2.circle(frame, (int(x), int(y)), int(raduis), (0, 255, 255), 2)
            # 矩形
            cv2.rectangle(frame, (int(x) - int(raduis), int(y) - int(raduis), int(2 * raduis), int(2 * raduis)),
                          color=(0, 0, 255), thickness=1)  # BGR
            # 画出重心
            cv2.circle(frame, center, 5, (0, 0, 255), -1)

        # 如果满足条件，就画出圆，画图函数，frame，中心，半径，颜色，厚度
        # if raduis>10:
        #       cv2.circle(frame,(int(x),int(y)),int(raduis),(0,255,0),6)
    cv2.imshow("original", frame)
```

```
        # print("重心坐标是(", cx, ",", cy, ")")
        Y_min.append(cy)
        # cv2.imshow("mask",mask)
        # cv2.imshow("mask_morph",mask_morph)
        # cv2.imshow("output",output)

        if cv2.waitKey(300) & 0xFF is ord("q"):
            break
Y_min.sort(reverse=True)
L=(Y_min[0]-479)*21/92 + 142.3087-3
if L>2:
        L=L-2.5
        if L>0:
            print(L,"cm")
else:
        print(L, "cm")
cap.release()
cv2.destroyAllWindows()
```

4、树莓派开机自启

cd /home/pi/.config
mkdir autostart
cd autostart
vi my.desktop

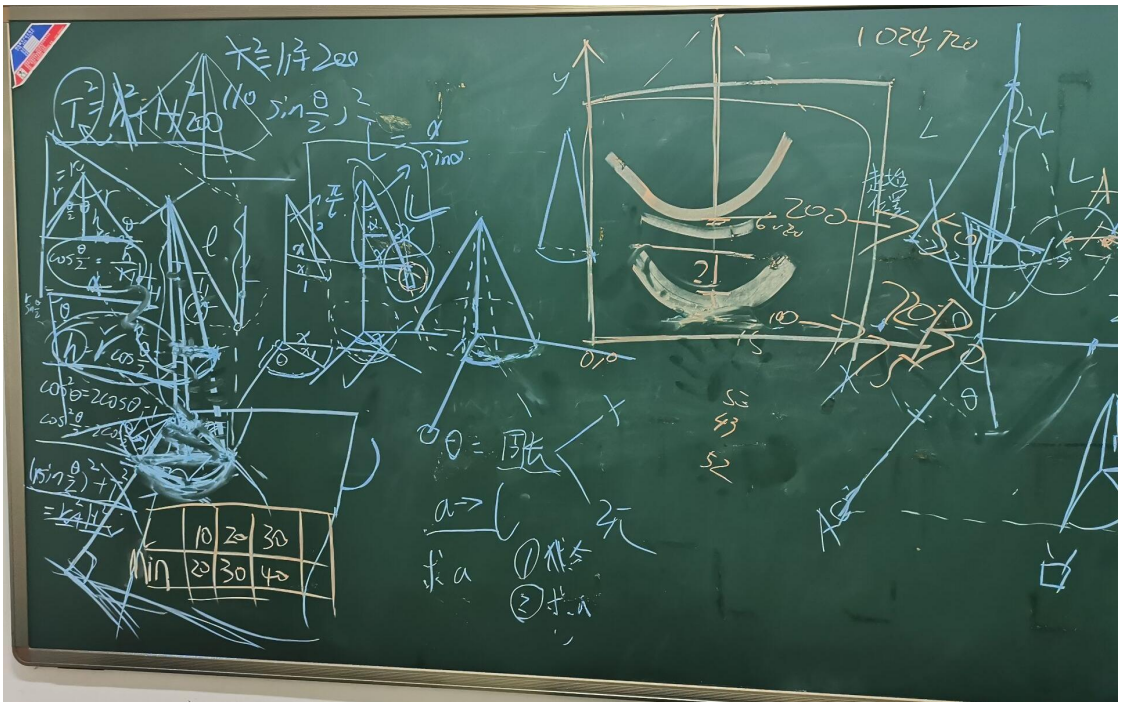
[Desktop Entry]
Type=Application
Exec=chromium-browser
--disable-popup-blocking
--no-first-run --disable-desktop-notifications --kiosk "域名地址"

配套加工安装条件

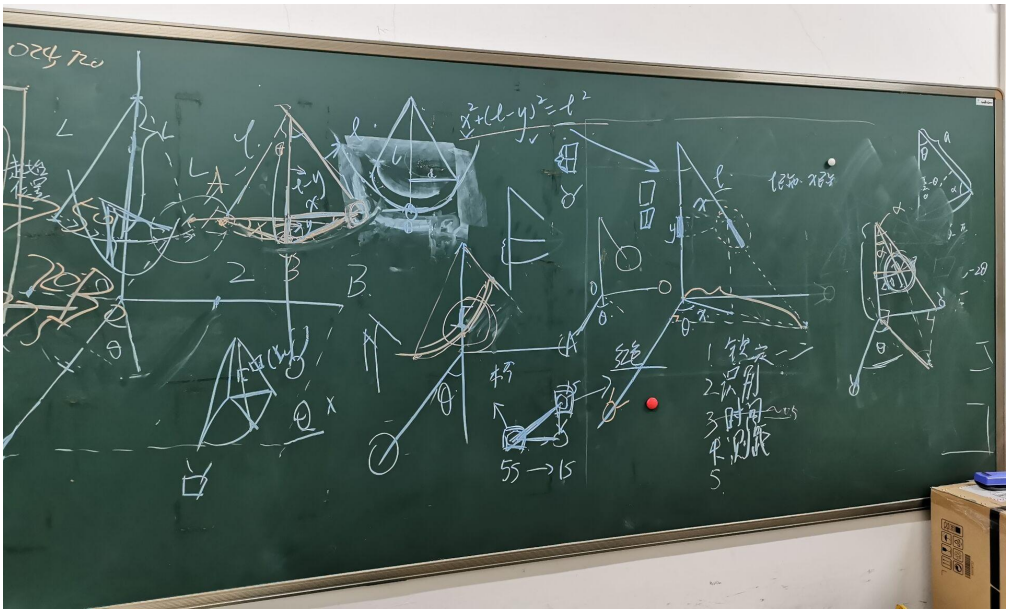
附图 1 3D 打印机

附图 2 热熔枪

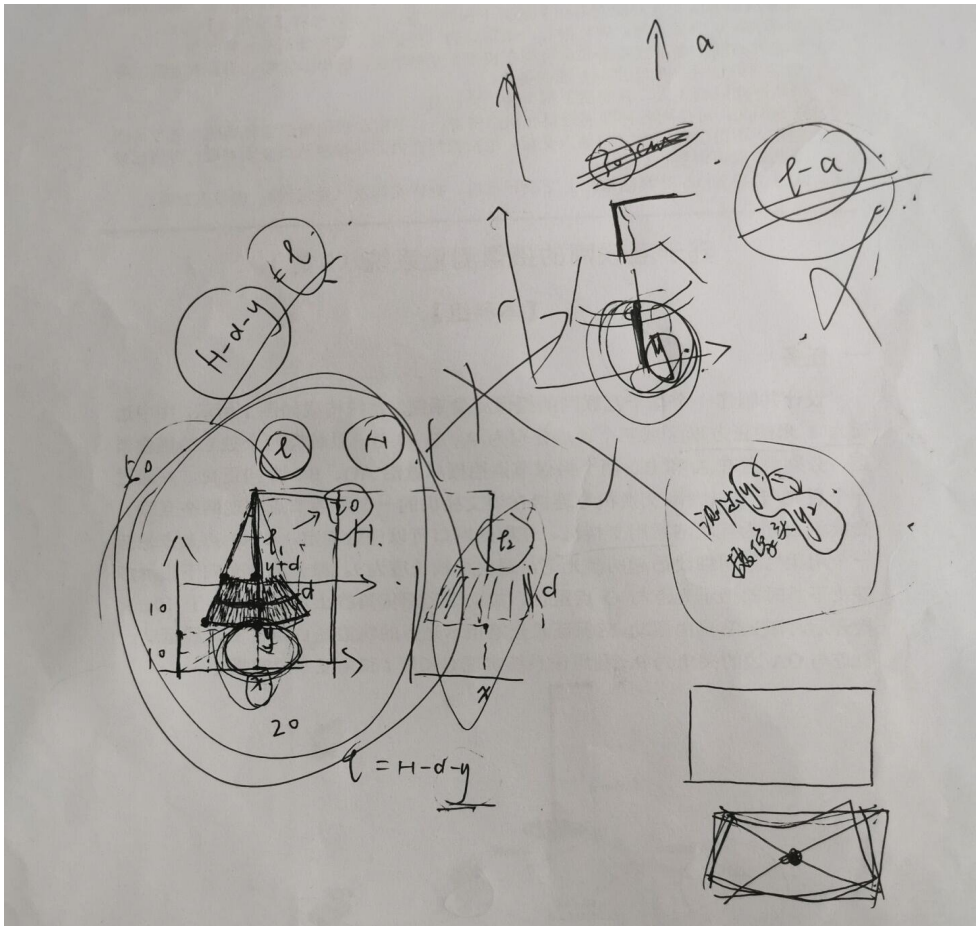
f



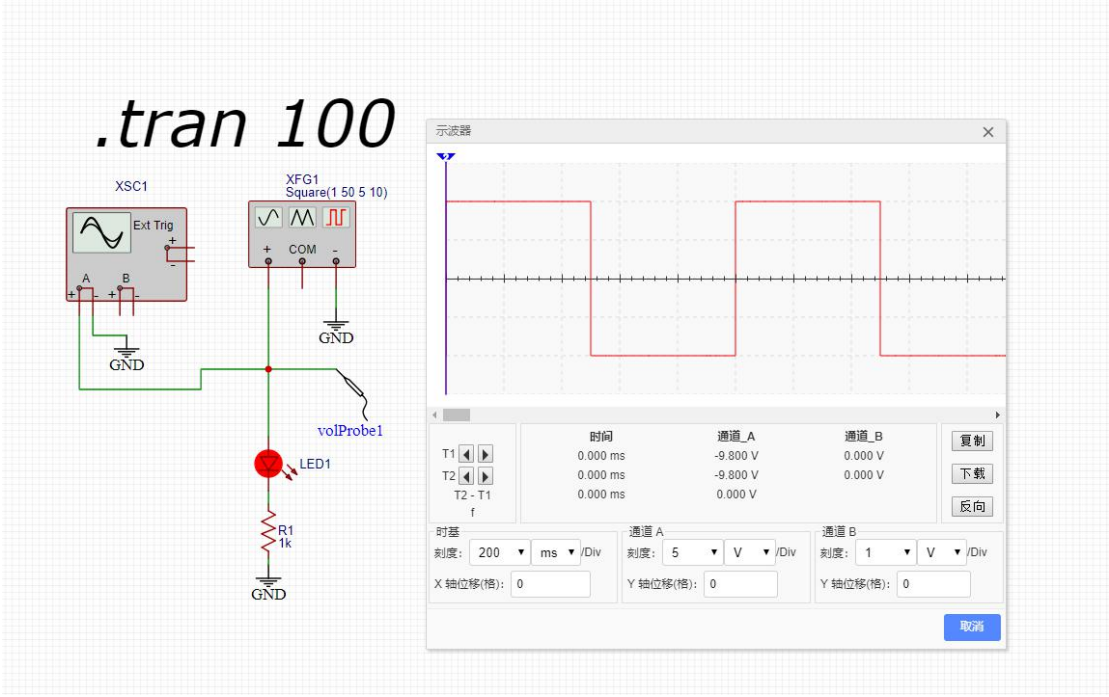
附图 3 草稿图 1



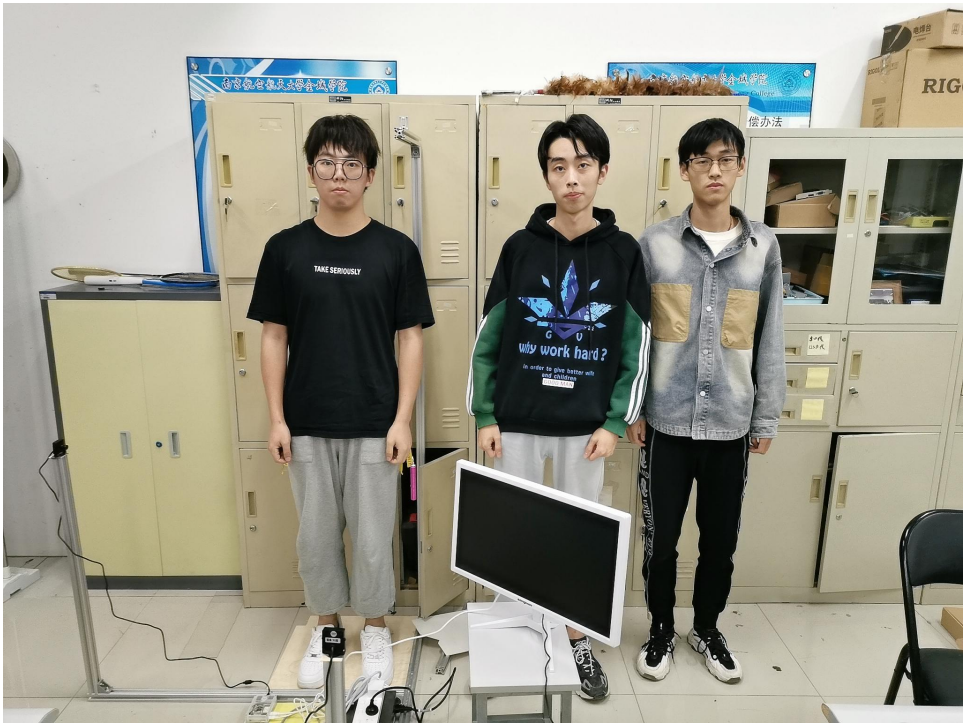
附图 4 草稿图 2



附图 5 草稿图 3


.tran 100
XSC1
Ext Trig
A
B
XFG1
Square(1 50 5 10)
COM
GND
GND
volProbe1
LED1
R1
1k
GND
示波器
时间
通道_A
通道_B
T1
0.000 ms
-9.800 V
0.000 V
T2
0.000 ms
-9.800 V
0.000 V
T2 - T1
0.000 ms
0.000 V
f
复制
下载
反向
时基
刻度: 200 ms /Div
X 轴位移(格): 0
通道 A
刻度: 5 V /Div
Y 轴位移(格): 0
通道 B
刻度: 1 V /Div
Y 轴位移(格): 0
取消

附图 6 电路仿真图



附图 7 三人与作品合影

附图 8　作品图