智能快递柜现状：

国内自动货柜技术及其应用已经特别成熟。彭香玲等[5]总结了我国目前自动仓储设施的特点，并详细介绍了新型联体提升货柜的运作原理、技术问题及结构创新。梁新强等[6]总结了升级优化控制方案的优点，并阐述了新型联体提升货柜的结构特点、电气控制及技术参数。徐俊愉,张鹏等[7]将RFID技术与智能快递箱结合起来，提出了新的硬件架构，其工作原理如图1所示：快递员将快件送达指定地点后，只需将其存入智能快递箱并用控制锁锁上即可。利用非接触式且加密程度高的射频RFID作为快递的标签，快递柜能够将物品进行识别然后利用MSP43O单片机控制TC35模块系统给用户发送一条短信，包括取件地址和验证码，用户在方便的时间到达该终端前输入验证码或者刷校园卡，即可打开智能控制箱的控制锁，取出快件。

MSP430单片机

TC35模块

TRF7960模块

控制器

扫描仪

图1 RFID快递箱工作原理图

李浩然，李朵然等[8]设计了基于无线传输技术的远程监控系统。下位机通过按键验证身份后进入温湿度采集界面，并通过Zigbee将环境情况传给上位机，同时下位机要检测上位机是否发来指令并进行相对应的操作。上位机接收Zigbee模块传送的数据并加以解析显示，通过与阈值的比较来进行相应指令的操作。其系统原理图如下：

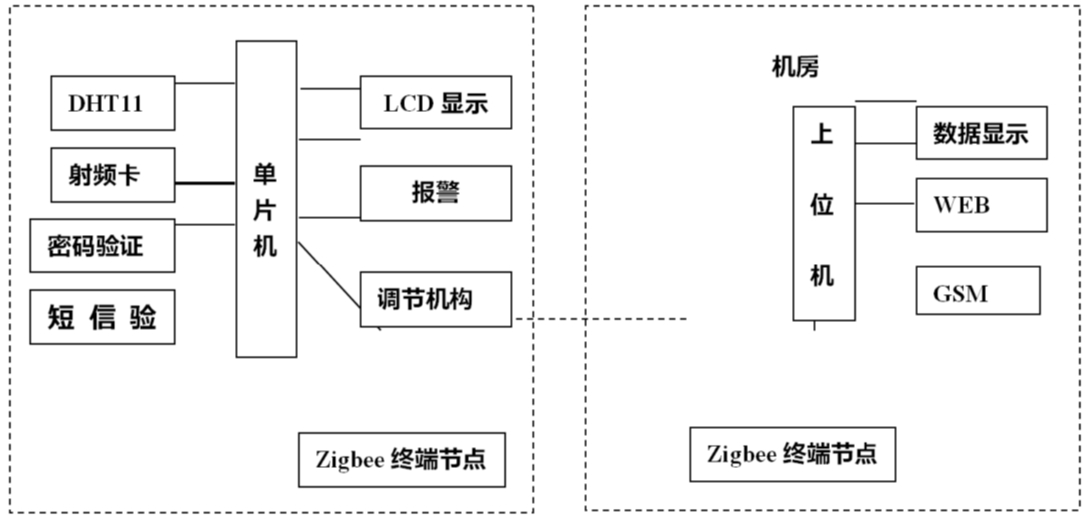


图2 智能快递柜系统原理图

传统快递柜大多体型巨大，移动性差



图3传统快递柜概念图

车载快递柜设计：

如图4所示，快递柜由柜体与由树莓派硬件电路控制的电磁锁两部分组成，通过扫描快递上的物流码与用户微信小程序端的身份二维码完成对快递的存取功能。

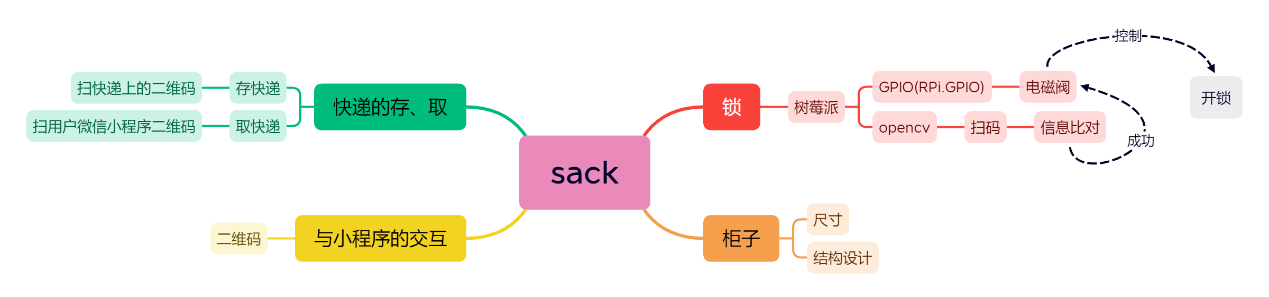


图4快递柜设计原理图

尺寸设计：

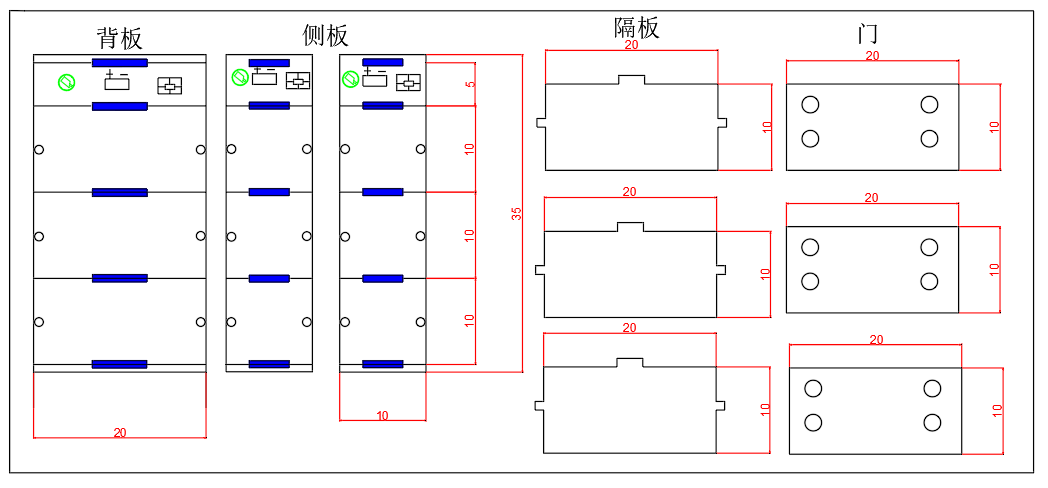


图5快递柜尺寸示意图

如图5所示，快递柜整体长、宽、高为20cm、10cm、35cm，分为三层，每层高10cm，用于存放快递，顶部5cm高的空间用于存放硬件电路设备。

控制电路设计：

采用树莓派作为主控，通过其GPIO接口连接继电器以实现对电磁锁的控制，采用5V电源给电磁锁供电，仿真中用充电宝给树莓派供电作为主控，。硬件连线图示意图如图

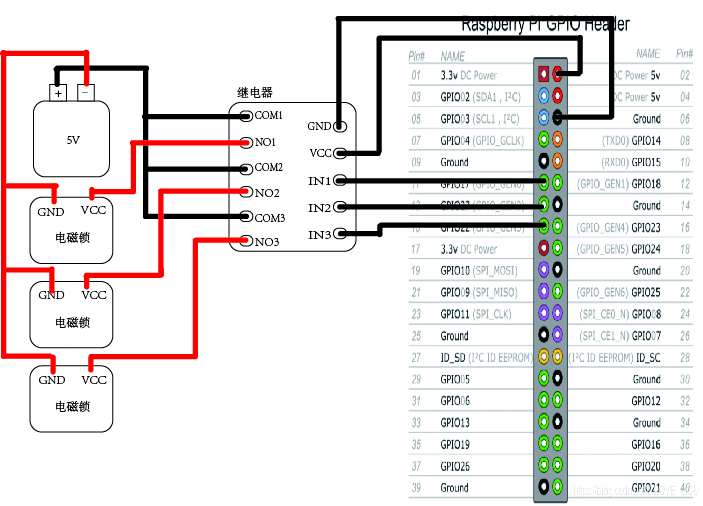


图6硬件电路连线示意图

扫码程序设计：

图7程序框图

具体实现：

柜体的制作：

在模拟仿真中，采用木板来制作柜体，其重量轻，扩展性好，且易于制作，是较好的制作材料。制作成品如图7所示：



图8快递柜柜体

硬件电路选材与实现：

选用树莓派4B作为主控，将其GPIO17、GPIO27、GPIO22三个GPIO接口与继电器相连接以实现对电磁锁的控制。电磁锁的控制选用１２Ｖ可充电锂电池，使电磁锁具有更大的弹力。使用树莓派的摄像头以实现后续对二维码的识别。

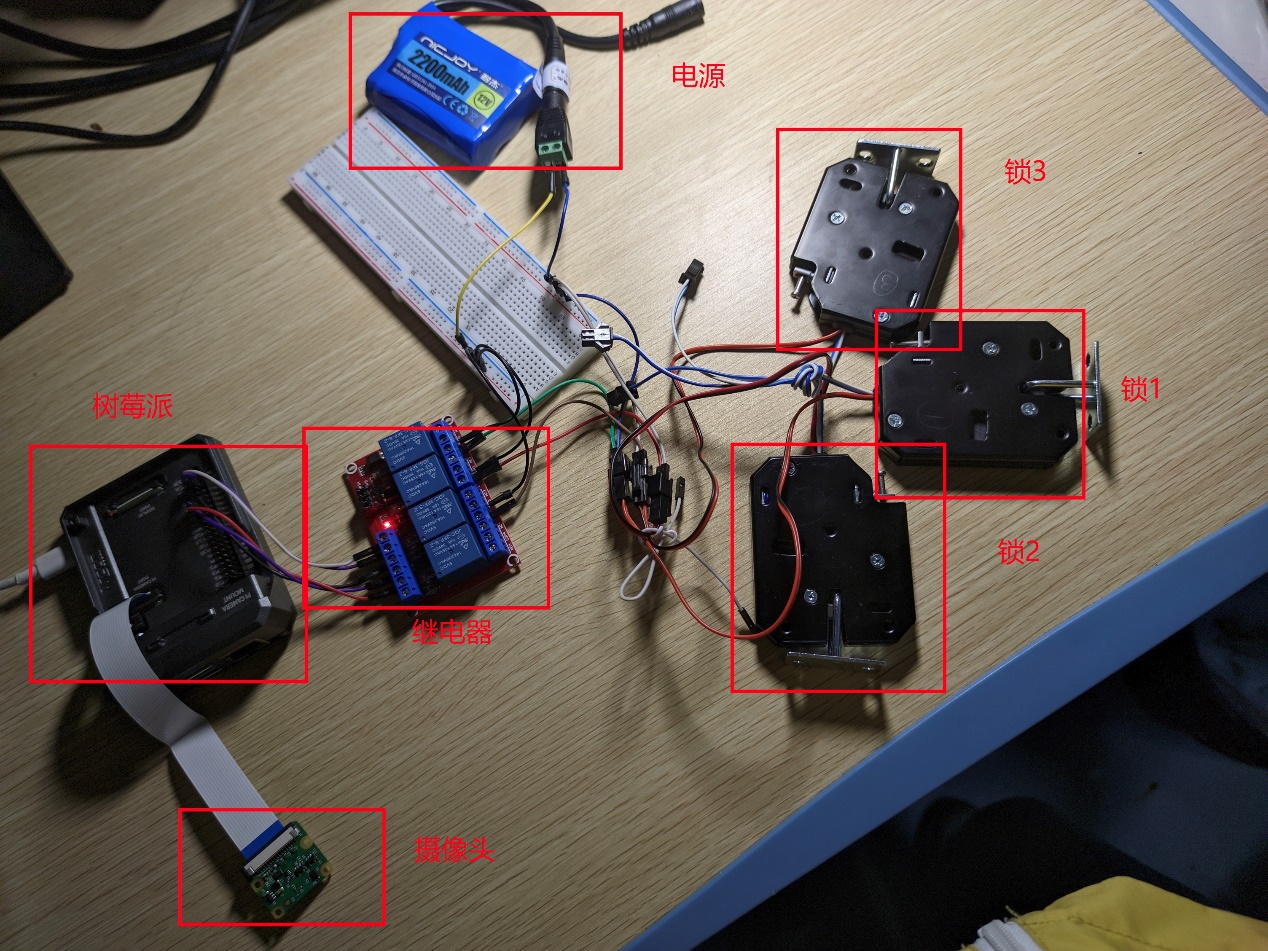


图9硬件电路连线图

软件功能的实现:

二维码识别程序

1. # 动态识别二维码，并打印结果

2. import sys

3. import numpy as np

4. import cv2

5.

6. cap = cv2.VideoCapture(0) # 设备号为0

7. cap.set(cv2.CAP\_PROP\_FOURCC, cv2.VideoWriter\_fourcc('M', 'J', 'P', 'G'))

8. cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH, 640)

9. cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 480)

10.

11. while (True):

12. if cap.isOpened() == False:

13. print('can not open camera')

14. break

15. else:

16. ret, frame = cap.read() # 读取图像

17. inputImage = frame

18.

19. #使用函数cv2.QRCodeDetector().detectAndDecode（）获取二维码的内容data并截取展示二维码的部分rectifiedImage

20. data, bbox, rectifiedImage = cv2.QRCodeDetector().detectAndDecode(inputImage)

21. if len(data) > 0:

22. print("Decoded Data : {}".format(data)) #打印识别到的数据

23. rectifiedImage = np.uint8(rectifiedImage)

24. cv2.namedWindow("Results", cv2.WINDOW\_AUTOSIZE)

25. rectifiedImage = cv2.resize(rectifiedImage, (300, 300))

26. cv2.imshow("Results", rectifiedImage)

27. else:

28. print("QR Code not detected")

29. cv2.namedWindow('Rectified QRCode')

30. cv2.imshow("Rectified QRCode", inputImage)

31. mykey = cv2.waitKey(1)

32. if mykey & 0xFF == ord('q'):

33. #释放内存、关闭窗口、退出进程

34. cap.release()

35. cv2.destroyAllWindows()

36. sys.exit(0)

37.

38. if ret == False: # 图像读取失败则直接进入下一次循环

39. continue

40.

电磁阀控制程序实现

1. import RPi.GPIO as GPIO

2. import time

3.

4. GPIO.setmode(GPIO.BCM)

5. GPIO.setup(17, GPIO.OUT) #锁1

6. GPIO.setup(27, GPIO.OUT) #锁2

7. GPIO.setup(22, GPIO.OUT) #锁3

8. GPIO.output(17, GPIO.HIGH)

9. time.sleep(0.1)

10. GPIO.output(17, GPIO.LOW)

11. GPIO.cleanup(17)

12. 存快递程序的实现

13. #QR\_read\_save

14. # 动态识别二维码，并打印结果

15. import sys

16. import numpy as np

17. import cv2

18. cap = cv2.VideoCapture(0) # 设备号为0

19. cap.set(cv2.CAP\_PROP\_FOURCC, cv2.VideoWriter\_fourcc('M', 'J', 'P', 'G'))

20. cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH, 640)

21. cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 480)

22. while (True):

23. if cap.isOpened() == False:

24. print('can not open camera')

25. break

26. else:

27. ret, frame = cap.read() # 读取图像

28. inputImage = frame

29. #使用函数cv2.QRCodeDetector().detectAndDecode（）获取二维码的内容data并截取展示二维码的部分rectifiedImage

30. data, bbox, rectifiedImage = cv2.QRCodeDetector().detectAndDecode(inputImage)

31. if len(data) > 0:

32. print("Decoded Data : {}".format(data)) #打印识别到的数据

33. rectifiedImage = np.uint8(rectifiedImage)

34. cv2.namedWindow("Results", cv2.WINDOW\_AUTOSIZE)

35. rectifiedImage = cv2.resize(rectifiedImage, (300, 300))

36. cv2.imshow("Results", rectifiedImage)

37. else:

38. print("QR Code not detected")

39. cv2.namedWindow('Rectified QRCode')

40. cv2.imshow("Rectified QRCode", inputImage)

41. mykey = cv2.waitKey(1)

42. if mykey & 0xFF == ord('q'):

43. #释放内存、关闭窗口、退出进程

44. cap.release()

45. cv2.destroyAllWindows()

46. sys.exit(0)

47. if ret == False: # 图像读取失败则直接进入下一次循环

48. continue

49.

取快递程序的实现

1. #package\_get.py

2. # 动态识别二维码，并打印结果

3. import sys

4. import numpy as np

5. import cv2

6. #import RPi.GPIO as GPIO

7. import time

8. #GPIO.setmode(GPIO.BCM)

9. #GPIO.setup(17, GPIO.OUT) #锁1

10.

11. cap = cv2.VideoCapture(0) # 设备号为0

12. cap.set(cv2.CAP\_PROP\_FOURCC, cv2.VideoWriter\_fourcc('M', 'J', 'P', 'G'))

13. cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH, 640)

14. cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 480)

15. cap.set(cv2.CAP\_PROP\_FPS, 10)

16. def compare(data):

17. filenew = open(r"user.txt", "w", encoding="utf8")

18. filenew.write(data)

19. file = open(r"user1.txt", "r", encoding="utf8")

20. filenew = open(r"user.txt", "r", encoding="utf8")

21. message=file.read() # 将识别到的信息存入

22. messagenew=filenew.read()

23. if messagenew==message

24. GPIO.output(17, GPIO.HIGH)

25. time.sleep(0.1)

26. GPIO.output(17, GPIO.LOW)

27. GPIO.cleanup(17)

28.

29. while (True):

30. if cap.isOpened() == False:

31. print('can not open camera')

32. break

33. else:

34. ret, frame = cap.read() # 读取图像

35. inputImage = frame

36.

37. #使用函数cv2.QRCodeDetector().detectAndDecode（）获取二维码的内容data并截取展示二维码的部分rectifiedImage

38. data, bbox, rectifiedImage = cv2.QRCodeDetector().detectAndDecode(inputImage)

39. if len(data) > 0:

40. print("Decoded Data : {}".format(data))

41. compare(data) #将读入的信息存入txt中

42. rectifiedImage = np.uint8(rectifiedImage)

43. cv2.namedWindow("Results", cv2.WINDOW\_AUTOSIZE)

44. rectifiedImage = cv2.resize(rectifiedImage, (300, 300))

45. cv2.imshow("Results", rectifiedImage)

46. else:

47. print("QR Code not detected")

48. cv2.namedWindow('Rectified QRCode')

49. cv2.imshow("Rectified QRCode", inputImage)

50.

51. mykey = cv2.waitKey(1)

52. if mykey & 0xFF == ord('q'):

53. #释放内存、关闭窗口、退出进程

54. cap.release()

55. cv2.destroyAllWindows()

56. sys.exit(0)

57.

58. if ret == False: # 图像读取失败则直接进入下一次循环

59. continue

60.



图10成品图

[5] 彭香玲, 刘茂生, 郭志钢. 全自动联体提升货柜的研制及其应用[J]. 装备制造技术, 2010(09):155-157.

[6] 梁新强, 邓转龙. 新型联体提升货柜的研制[J]. 物流工程与管理, 2010(08):187-190.

[7] 徐俊愉, 张鹏, 周恒飞. 基于RFID的智能快递箱[J]. 山东工业技术, 2016(01):151.

[8] 李浩然, 李朵然, 王蕾, 等. 智能快递柜的设计与实现[J]. 电子制作, 2015(07):60-61.