



*VISUAL IMPAIRED HELPER*

# A QUICK RECAP

## **Problem**

Many visually impaired people are using older methods for guidance and do not leverage the technology available today.

## **Solution**

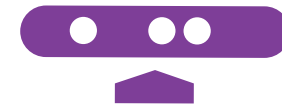
My proposed solution is to use a system that a user wears i.e a headband that guides the visually impaired person

# PROJECT IMPLEMENTATION

Implementation Changes

User Interface Specification

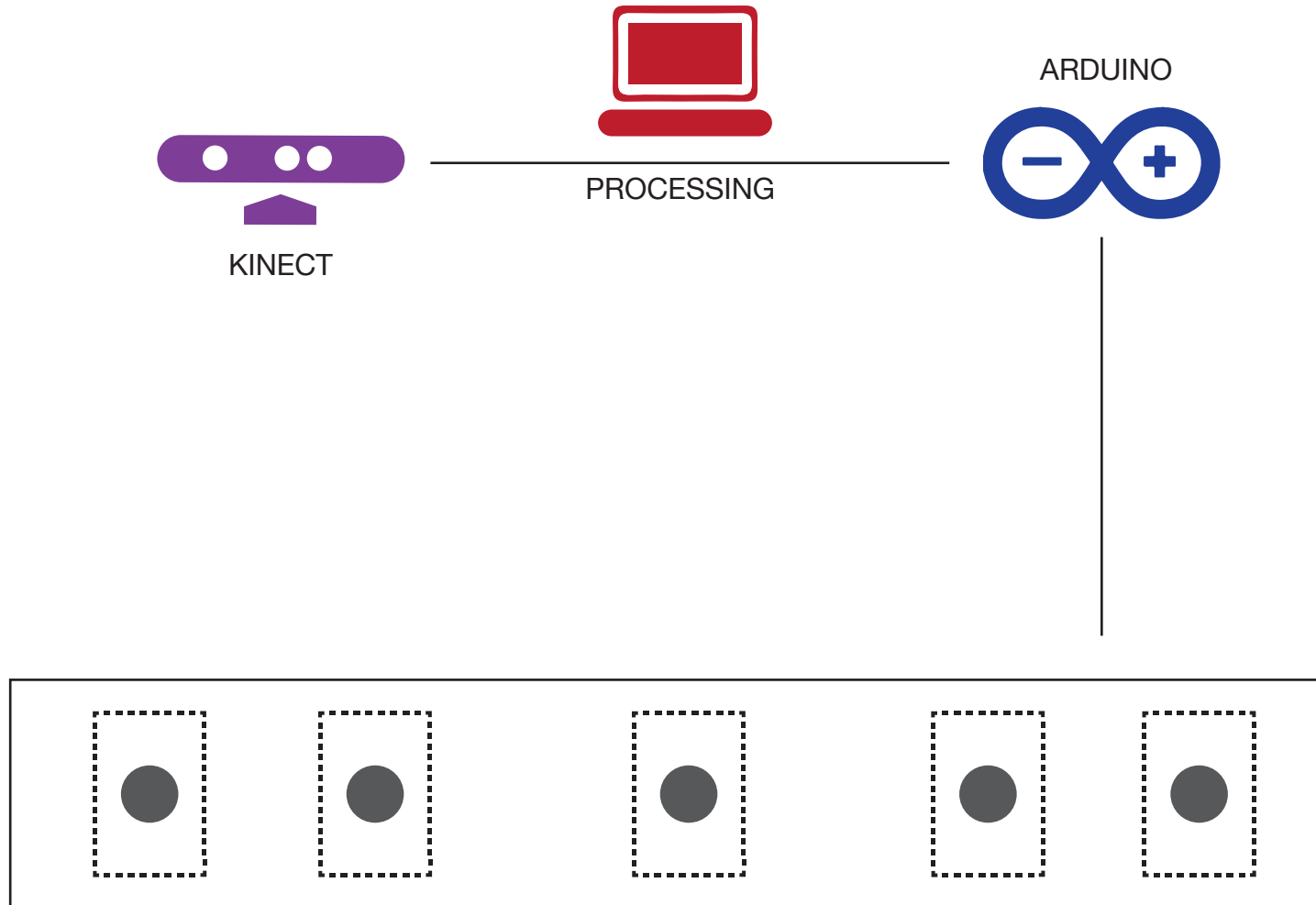
Low Level Design Computer Vision



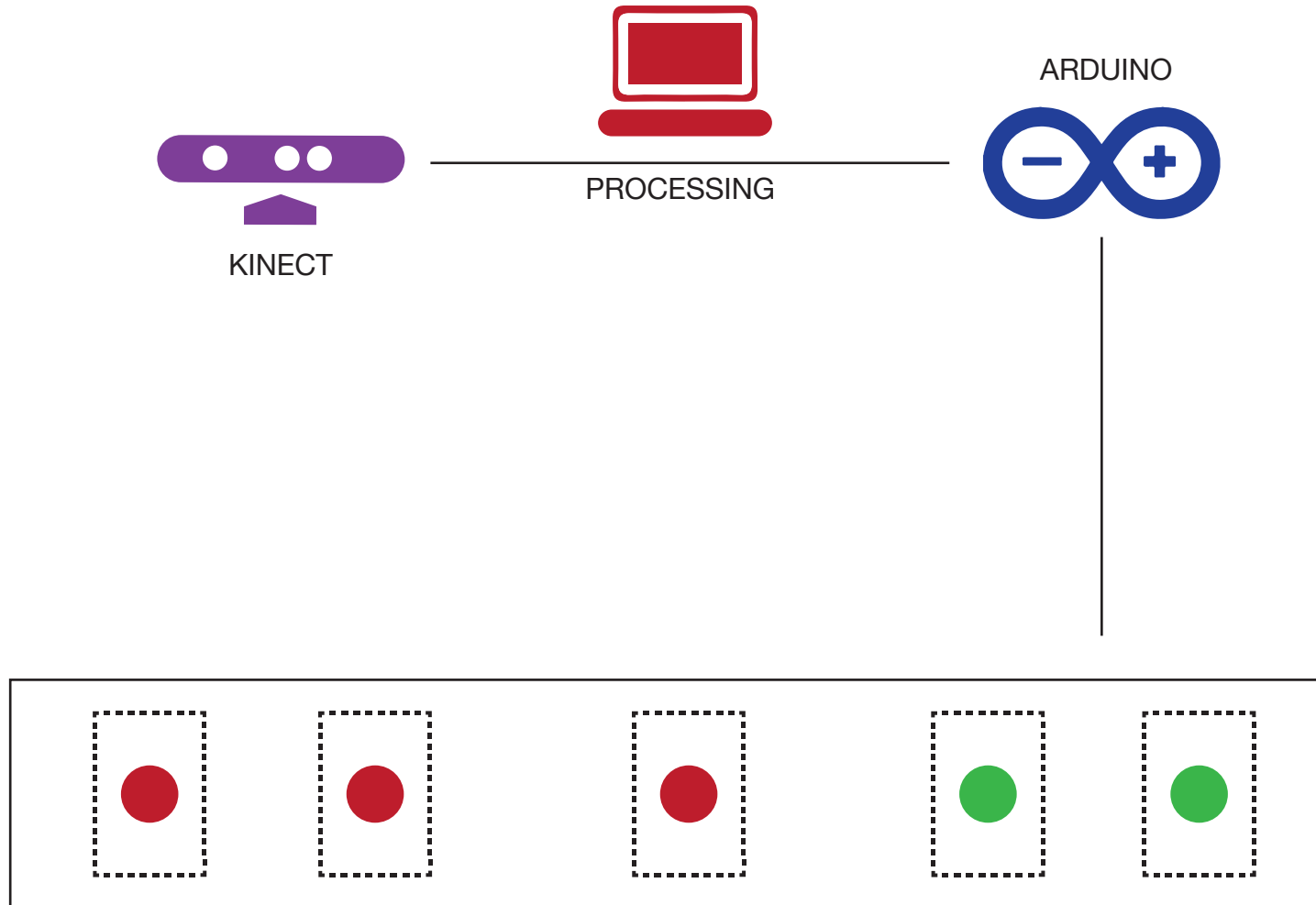
Low Level Design Engineering



# USER INTERFACE SPECIFICATION



# USER INTERFACE SPECIFICATION



# LOW LEVEL DESIGN (VISION)

1

## INPUT

START KINECT SENSOR

ENABLE COLOR & DEPTH STREAMS

2

## CAPTURE IMAGE DATA

CAPTURE FRAMES USING AN EVENT HANDLER (THIS MODEL KEEPS GETTING FRAMES UNTIL PROGRAM HALTS)

3

## BLOB DETECTION

CONVERT KINECT DATA STREAM INTO AN OPENCV IMAGE

CONVERT OPENCV IMAGE TO GRAYSCALE

DETECT OUTERMOST CONTOURS

DRAW MINIMUM BOUNDING BOX AROUND THE CONTOUR

DETECT CLOSEST PART OF BLOB AND OBTAIN THE DISTANCE USING KINECT API

4

## PATH RECOGNITION

CHECK WHICH BOUNDING BOXES COLLIDE WITH SEGMENTS

BUILD ENCODED STRING BASED ON COLLISIONS

5

## OUTPUT

SENT BUILT STRING TO ARDUINO MICROCONTROLLER USING SERIAL CONNECTION

# LOW LEVEL DESIGN (VISION)

## 1 INPUT

START KINECT SENSOR  
ENABLE COLOR & DEPTH STREAMS

## 2 CAPTURE IMAGE DATA

CAPTURE FRAMES USING AN EVENT  
HANDLER (THIS MODEL KEEPS GETTING  
FRAMES UNTIL PROGRAM HALTS)  
CAPTURE AT 3 FPS (10 SEC INTERVALS)

## 3 BLOB DETECTION

CONVERT KINECT DATA STREAM  
INTO AN OPENCV IMAGE  
CONVERT OPENCV IMAGE TO  
GRAYSCALE  
DETECT OUTERMOST CONTOURS  
DRAW MINIMUM BOUNDING BOX  
AROUND THE CONTOUR  
DETECT CLOSEST PART OF BLOB  
BY OBTAINING THE DISTANCE  
FROM KINECT API

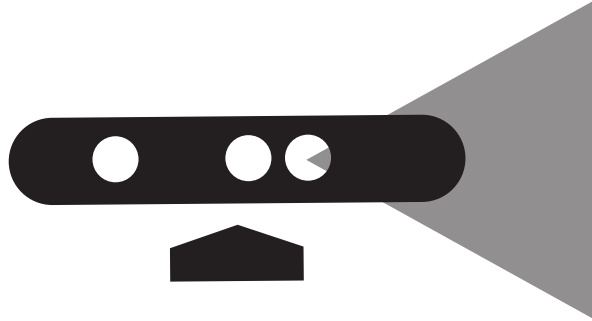
## 4 PATH RECOGNITION

CHECK WHICH BOUNDING BOXES  
COLLIDE WITH SEGMENTS AND THE  
DEPTH FOR COLLIDING OBJECTS  
BUILD ENCODED STRING BASED ON  
COLLISIONS AND DEPTH FROM  
KINECT API

## 5 OUTPUT

SENT BUILT STRING TO ARDUINO  
MICROCONTROLLER USING SERIAL  
CONNECTION

# GRAPHICAL REPRESENTATION (VISION)



1

INPUT



2

CAPTURE IMAGE DATA



3

BLOB DETECTION



4

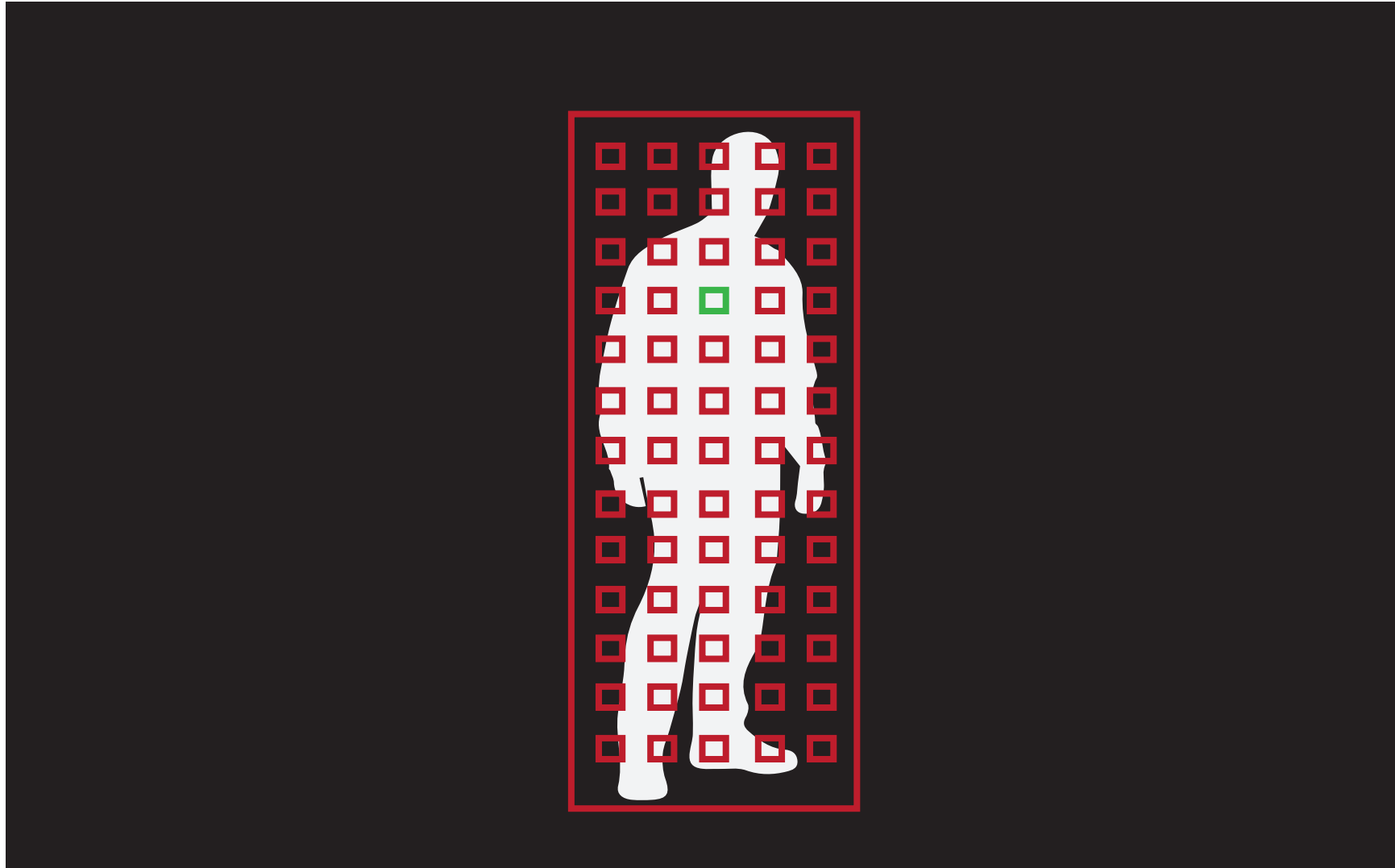
PATH RECOGNITION



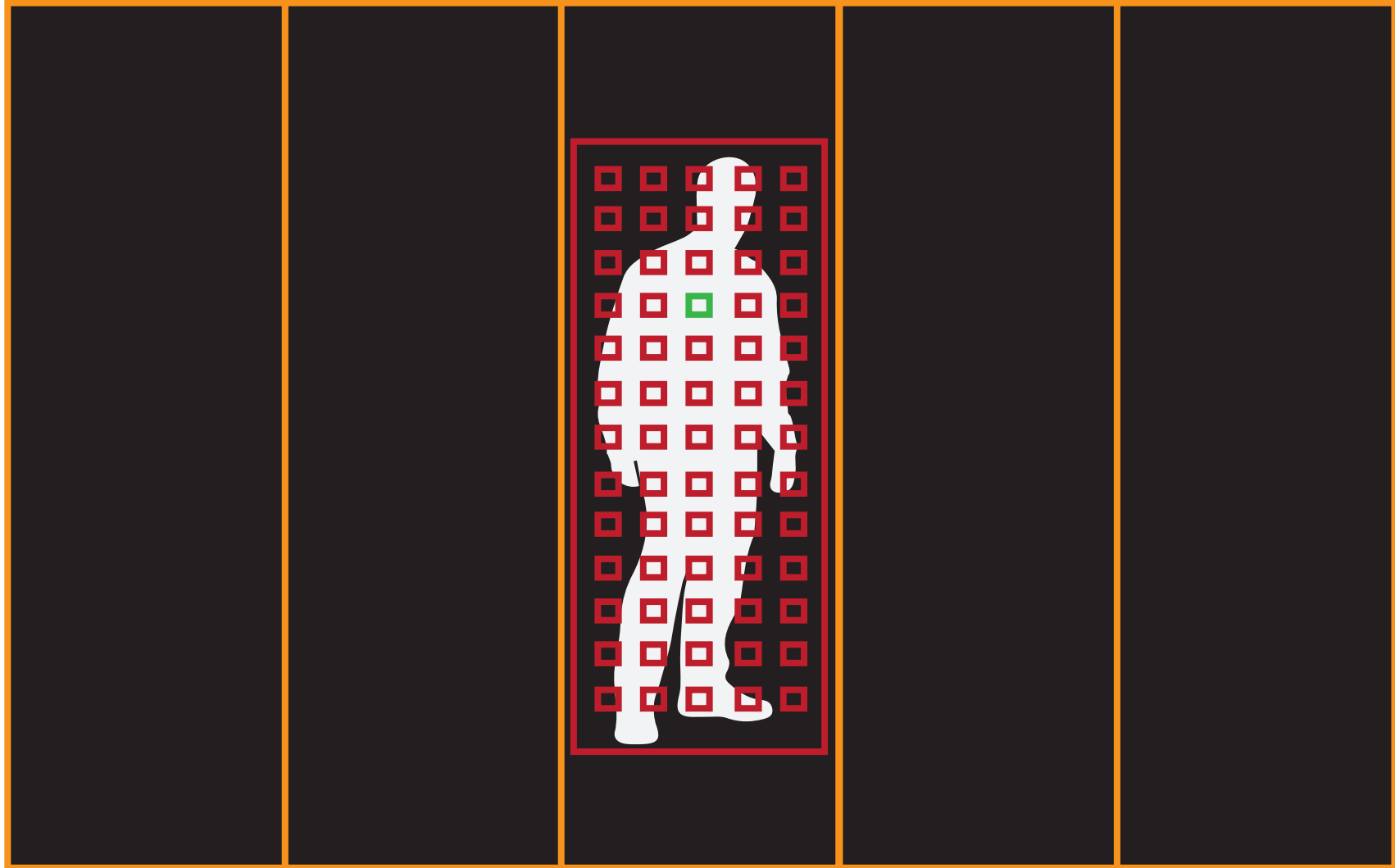
# IMPLEMENTATION SAMPLE



# IMPLEMENTATION SAMPLE



# IMPLEMENTATION SAMPLE



0,0,255,0,0

# LOW LEVEL DESIGN (ENGINEERING)

1

## INPUT

LISTEN ON SERIAL PORT USING  
SERIAL.BEGIN(9600) AND CHECK FOR  
SERIAL INPUT IN MAINLOOP USING  
SERIAL.AVAILABLE()

2

## PATH RECOGNITION

CUSTOM METHOD TO SPLIT INPUT  
STRING, SPLITANDSET(CHAR []  
INPUT)

3

## OUTPUT

VIBRATE MOTORS USING  
DIGITALWRITE(OBJECT, HIGH)

# LOW LEVEL DESIGN (ENGINEERING)

1

## INPUT

LISTEN ON SERIAL PORT USING  
SERIAL.BEGIN(9600) AND CHECK FOR  
SERIAL INPUT IN MAINLOOP USING  
SERIAL.AVAILABLE()

2

## PATH RECOGNITION

STORE DEGREE OF VIBRATION IN  
INTEGER VARIABLES BY PARSING  
THEM AS A STRING.

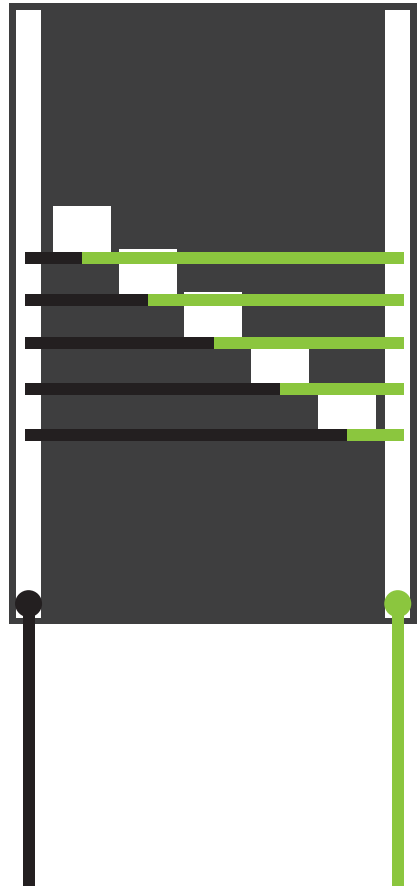
3

## OUTPUT

VIBRATE MOTORS USING  
ANALOGWRITE(OBJECT, HIGH)

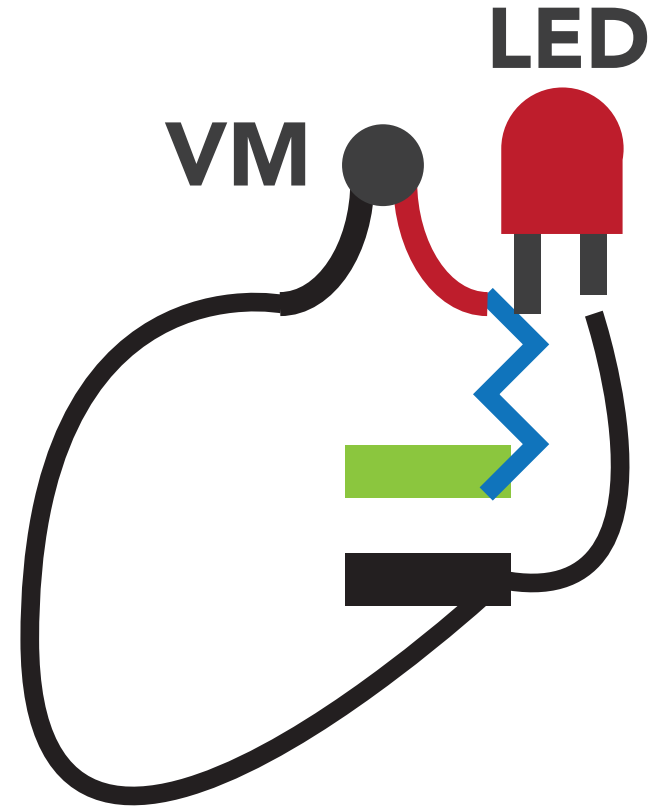
# CIRCUIT IMPLEMENTATION (ENGINEERING)

FINAL DESIGN



DISTRIBUTION BOARD

1X



VIBRATION CIRCUIT

5X

# TOOLS USED

