Hummingbird Python Package Description

Description of Files

hummingbird.py – Contains the Hummingbird API functions for controlling Hummingbird

hummingbirdconnection.py – Contains low level functions for sending and receiving data over USB, used by hummingbird.py

hidapi32/64.dll, libhidapi.dylib, libhidapi32/64.so, libhidapipi.so – compiled C libraries used by hummingbirdconnection.py to scan for USB devices (specifically, the Hummingbird) and open a connection. **dll** files are for Windows, **dylib** is for OSX, and **so** is for Linux.

cricketexample.py, driverexample.py, LEDexample.py, hummingbirdTester.py – Example Python programs, open them for more information on what each one does.

Hummingbird API

General

close – closes the connection to the Hummingbird

Usage: call this when your program is exiting to cleanly close the Hummingbird connection. This will shut off the motors/servos/LEDs and then send the Hummingbird back to idle mode (green LED color fades)

halt – convenience function to shut off the Hummingbird LEDs and hummingbird motors/vibration motors

Usage: call whenever you wish the Hummingbird to stop moving and turn off its LEDs with one function call.

Outputs

set_tricolor_led – controls the Hummingbird tri-color LEDs

Usage: The first argument is the port (1 or 2), the second is the color to set the LED to as either a hex triplet string or three 0-255 RGB values where the first value indicates the red intensity, second is green, third is blue.

```
hummingbird.set_tricolor_led(1, '#00FF00') # sets LED on port 1 to green
hummingbird.set_tricolor_led(2, 127, 127, 127) # set LED on port 2 to white at 50% power
hummingbird.set_tricolor_led(1, 0, 0, 0) # turns of LED on port 1
hummingbird.set_tricolor_led(2, '#8000FF') # sets LED 2 to 50% red on, green off , 100% blue on
```

set_single_led – Sets the single color LEDs

Usage: First argument is the port of the LED (1-4), second is intensity from 0 to 255.

```
hummingbird.set_single_led(1, 255) # Turns LED on port one fully on
hummingbird.set_single_led(2, 0) # Turns LED on port two off
hummingbird.set_single_led(3, 51) # Turns LED on port three on with 20% power
```

set_motor – Controls one of the DC gear motor ports

Usage: First argument is the motor port (1 or 2) and the second is the speed from -1.0 (full throttle reverse) to 1.0 (full throttle forward)

```
hummingbird.set_motor(1, 1.0) # full forward on port one
hummingbird.set_motor(2, -1.0) # full reverse on port two
hummingbird. set_motor (1, 0.0) # turn motor port one off
```

set_all_motors – Controls both DC gear motor ports simultaneously

Usage: First argument is the power to motor port one, second is the power to motor port two. Speeds range from -1.0 (full throttle reverse) to 1.0 (full throttle forward)

```
hummingbird.set_all_motors(1.0, 1.0) # full forward on both ports
hummingbird.set_ all_motors(0.3, -1.0) # 30% forward on port one, full reverse on port two
hummingbird. set_ all_motors (0.0, 0.0) # turn both motor ports  off
```

set_vibration_motor - Controls the vibration motor ports

Usage: First argument is the motor port (1 or 2) and the second is the intensity of vibration, from 0 to 255.

```
hummingbird.set_vibration_motor(1, 255) # Turn vibration motor 1 to full power
hummingbird.set_vibration_motor(2, 127) # Turn vibration motor 2 to 50% power
hummingbird.set_vibration_motor(1, 0) # Turn vibration motor 1 off
```

set_servo - Controls the servo motor ports

Usage: First argument is the servo port (1 to 4) and the second is the angle, from 0 to 180.

```
hummingbird.set_servo(1, 180) # Set servo 1 to 180 degrees
hummingbird. set_servo (2, 90) # Set servo 2 to 90 degrees
hummingbird. set_servo (3, 0) # Set servo 3 to 0 degrees
```

Sensors

get_raw_sensor_value – returns a value corresponding to a voltage at a specified sensor port

Usage: Takes one parameter, the sensor port number (1 to 4). Returns a float between 0 and 255 corresponding to the voltage at the port. 0V reads 0, 5V reads as 255, and values in between scale linearly.

```
port_one_sensor = hummingbird. get_raw_sensor_value(1) # reads value on sensor port one
```

get_light_sensor – returns a value corresponding to the light sensor value

Usage: Takes one parameter, the sensor port number (1 to 4). Returns a float between 0 and 255 corresponding to the amount of light seen by the sensor at the port, with 0 being complete darkness and 255 being direct sunlight.

```
port_two_lightsensor = hummingbird. get_light_sensor(2) # reads value on sensor port two
```

get_knob_value– returns a value corresponding to the knob's orientation

Usage: Takes one parameter, the sensor port number (1 to 4). Returns a float between 0 and 255 corresponding to the orientation of the knob.

```
knob = hummingbird. get_knob_value(3) # reads value on sensor port three
```

get_sound_sensor – returns a value corresponding to the sound sensor value

Usage: Takes one parameter, the sensor port number (1 to 4). Returns a float between 0 and 255 corresponding to the amount of ambient sound heard at the port, lower values indicate less sound.

```
ambient_sound = hummingbird. get_sound_sensor(2) # reads value on sensor port two
```

get_temperature – returns a value corresponding to the temperature in celsius

Usage: Takes one parameter, the sensor port number (1 to 4). Returns the temperature in Celsius as a float between -28 and 78.

```
temp_celsius = hummingbird. get_temperature(4) # reads temperature on port 4
```

get_distance – returns a value corresponding to the distance to an object in centimeters (cm)

Usage: Takes one parameter, the sensor port number (1 to 4). Returns the distance to an object in cm, the range of the sensor is 8 to 80 cm.

```
distance = hummingbird. get_distance(2) # reads distance on sensor port two
```

get_all_sensors – returns all four raw sensor values

Usage: Returns all four sensor values in raw format. Each value is a float between 0 and 255 corresponding to the voltage at the port. 0V reads 0, 5V reads as 255, and values in between scale linearly.

```
sensor_one, sensor_two, sensor_three, sensor_four = hummingbird. get_all_sensors()
```

are_motors_powered– returns if motor power is plugged in

Usage: Returns a Boolean flag – True if motor power is plugged in, false otherwise.

```
motors_powered = hummingbird.are_motors_powered()
```