Artículo **Revista de Tecnología e Innovación**
Septiembre 2017 Vol.4 No.12, 29-39

Sistema de oxigenación de granjas de cultivo acuícola por medio de energía sustentable

HERNÁNDEZ-SÁNCHEZ, Adán †*, GARCÍA-NAVARRO, Josefina, ZUMAYA-QUIÑONES, Rocío y BAUTISTA-VARGAS, María Esther

Universidad Politécnica de Altamira

Recibido Febrero 15, 2017; Aceptado Agosto 29, 2017

Resumen

Según el Anuario Estadístico de Acuacultura y Pesca 2013 Tamaulipas generó 9,192 toneladas de camarón usando granjas camaronicolas. Sin embargo, se reportó, que un virus dañó la producción de especies, esto debido a que el método de oxigenación incorpora agua proveniente de un río, garantizando agua oxigenada, pero, teniendo la posibilidad de que ingrese algún virus que afecte a la población en estas granjas. El objetivo de esta investigación es publicar el diseño de un sistema de oxigenación de granjas de cultivo acuícola por medio de energía sustentable. Esto se logró haciendo uso de un sensor de oxígeno disuelto conectado a una Raspberry pi 3, la cual tiene la función de controlar un aspersor de agua en el estanque según el nivel de oxígeno disuelto en el agua. Además, la raspberry pi 3 es capaz de transmitir los valores de oxígeno disuelto en el estanque a un servidor remoto para estadística. Este sistema funciona con energía solar por medio de un sistema fotovoltaico. La aportación de este proyecto es realizar la oxigenación del agua de cualquier cultivo acuícola sin la posibilidad de introducir virus al sistema, que puedan perjudicar a la población, dado que esto sucede el método convencional

Sistemas de oxigenación, energía solar, cultivo acuícola, sistema fotovoltaico

Abstract

According to the Statistical Yearbook of Aquaculture and Fisheries 2013 Tamaulipas produced 9,192 ton of shrimp using shrimp farms. However, it was reported that a virus damaged the production of species, this because the oxygenation method incorporates water from a river, guaranteeing oxygenated water, but, having the possibility of entering a virus that affects the population in these farms. This research shows the design of oxygenation systems of aquaculture farms using sustainable energy. Making use of a dissolved oxygen sensor connected to a Raspberry pi 3, wich can controller a nozzle pressure when level of dissolved oxygen in the water is low. In adition the Raspberry pi 3 can transmit the datums to remote server for statistics. This system works with solar energy through a photovoltaic system. The contribution of this project is to make the oxygenation of the water of any aquaculture crop without the possibility of introducing viruses into the system of said crop that can harm the population, since this happens the conventional method. This system can be used without put in risk any aquaculture farm by viruses

Oxygenation system, solar energy, Aquaculture, solar panel

Citación: HERNÁNDEZ-SÁNCHEZ, Adán, GARCÍA-NAVARRO, Josefina, ZUMAYA-QUIÑONES, Rocío y BAUTISTA-VARGAS, María Esther. Sistema de oxigenación de granjas de cultivo acuícola por medio de energía sustentable. Revista de Tecnología e Innovación 2017, 4-12: 29-39.

* Correspondencia al Autor (Correo electrónico: adan.hernandez@upalt.edu.mx)
† Investigador contribuyendo como primer autor.

Introducción

Los inversionistas acuícolas buscan zonas libres de enfermedades patógenas para que no afecten el cultivo de camarón y Tamaulipas representa un campo fértil en este concepto por la excelente inocuidad de sus aguas (zonas lejanas a la mancha urbana y/o rural). Sin embargo, en 2014 se reportó, que un virus dañó la producción de camarón en el municipio de Matamoros. Es no es algo que solo afecte a Tamaulipas, por ejemplo, la producción de camarón del Estado de Sonora ha reportado descensos del 39.3% y del 50% respectivamente, en proporción al año 2009, donde la producción fue de 81,422.8 t.

Este declive se debió a la presencia del virus de la mancha blanca (WSSV). De hecho, esto sucede debido al método de oxigenación del agua de estas granjas camaronícolas, que implica la apertura de compuertas para que ingrese agua de esteros, lagunas costeras o ríos a la presa donde se cultiva el camarón.

Esto no solo implica la posibilidad de infectar a la población de crustáceos de la granja con algún virus, sino también la posibilidad de la fuga del camarón hacia el estero, lagunas costeras o ríos, ya que, para mantener la concentración de agua en la presa, se requiere abrir otra compuerta que permita la salida del agua con baja concentración de oxígeno disuelto. Existen otros métodos de oxigenación que no requieren exponer la población de biomasa a virus, o permitir que estos salgan del estanque, sin embargo, requieren de un tendido eléctrico, y por la lejanía no cuentan con este o consumo de combustible para generadores eléctricos portátiles.

Es por esto, que se hace cada vez más necesaria la implementación de técnicas y formas de manejo del cultivo que contribuyan a reducir los impactos ambientales y ayuden a sostener la base natural de recurso.

Por lo que en esta publicación se muestra el diseño de un sistema de detección, monitoreo y cambio de concentración de oxígeno disuelto de agua para granjas de camarones utilizando una fuente de energía renovable como lo es, la energía solar. Esto se logra mediante la extracción de agua del fondo del estanque (donde se encuentra menos oxigenada) y esparcir el agua en forma de gotas en el aire (para que se oxigene) y se incorpore nuevamente al cuerpo de agua, pero oxigenada.

La extracción de agua con bajos niveles de oxígeno disuelto continuara hasta que se llegue al nivel deseado, y se reactivará la extracción cuando baje nuevamente, todo esto funciona gracias a la energía solar. Con este sistema se logra la reducción de efluentes de los estanques, la disminución de la entrada de agua proveniente del estero lo cual reduce el riesgo de introducción de predadores, camarón silvestre y la posible diseminación de enfermedades y se evita la pérdida de la productividad natural que se produce dentro de los estanques.

El sistema de oxigenación de granjas de cultivo acuícola por medio de energía sustentable requiere para su implementación el material de la tabla 1.

Elemento	Función
Raspberry pi 3	Procesador de datos
Kit extracción y aspersión de agua	Extrae el agua menos oxigenada y la esparce en el aire para oxigenarla
Kit de Generación de Energía sustentable	Provee de electricidad a la bomba, raspberry y sensores mediante paneles solares
Módulo de expansión del raspberry pi 3	Acoplador entre raspberry pi 3 y la bomba
Sensor de Oxígeno Disuelto	Proporciona los datos de Oxígeno Disuelto
Banda Ancha Móvil	Provee de internet al sistema

Tabla 1 Material necesario para la implementación del sistema de oxigenación

ISSN-2410-3993

Raspberry pi 3

El sistema de oxigenación utiliza un raspberry pi 3 como unidad central de procesamiento, por lo que conviene tener idea de sus capacidades y funcionamiento.El raspberry pi es una computadora del tamaño de una tarjeta de crédito, ver Figura 1, tiene puertos USB, Ethernet, Wireless, HDMI, bocinas, micrófono, video analógico, un puerto con entradas y salidas de propósito general, y utiliza una memoria microSD como disco duro, y lo mejor de este que solo requiere una fuente de alimentación de 5Vcd.

A esta micro computadora se le puede instalar una gran variedad de sistemas operativos, la mayoría de ellos base Linux, sin embargo, también puede funcionar con Windows 10.

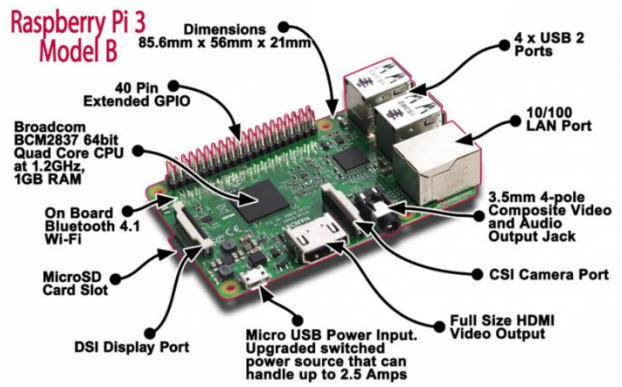


Figura 1 Imagen que muestra físicamente a la raspberry pi 3

El raspberry 3 utiliza Raspbian como sistema operativo (OS por sus siglas en inglés), ya que, trae preinstalado el compilador Python con el que se utiliza en el puerto de entradas y salidas generales. La guía de instalación de Raspbian OS se verá en el apartado "Guia de Instalación de Raspbian".

Elemento de extracción y aspersión

Este elemento es necesario para oxigenar el agua, consta de una bomba sumergible que funciona con 12Vcd, 60psi de presión, una boquilla aspersor, 3m de manguera de ½", filtro y cable (aprox. 5m). Se requiere además de una base flotante de poliestireno expandido para la boquilla aspersora, y una base metálica para la bomba, la cual puede o no estar sumergida en el agua.

Elemento de energía sustentable

Es el que provee de electricidad al sistema de oxigenación. Este elemento consta de un panel solar de 100W, una batería recargable de 12V (al menos 20AH), base para el panel, cables (calibre 14AWG), controlador de carga (15A) y un regulador de voltaje (LM2596, 12V a 5V). La configuración eléctrica de este se puede ver en la figura 2.

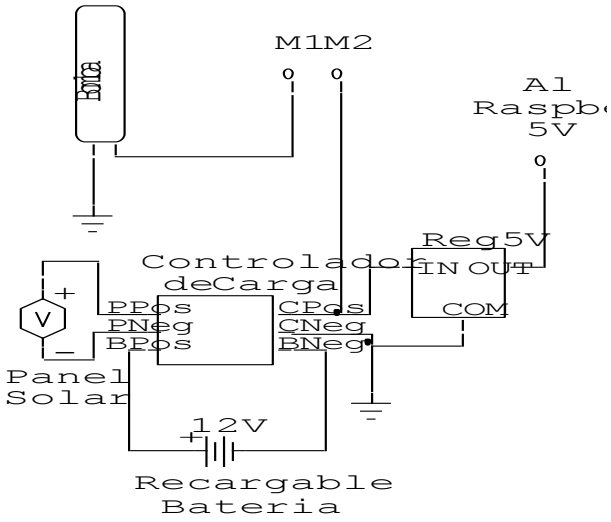


Figura 2 Diagrama eléctrico de conexión del sistema de oxigenación. M1 y M2 se conectan al módulo de expansión

Módulo de expansión para Raspberry pi

Debido a que el raspberry pi trabaja con 5Vcd, y la bomba funciona con 12Vcd, se hace necesario un acoplador. Este módulo expansión permite al raspberry activar o desactivar equipos de hasta 250VAC/5A, 30VDC/5A. Se conecta directamente en el puerto de entradas y salidas generales.

Sensor de oxígeno disuelto

Módulo Tentacle T3 con sensor de Oxígeno disuelto (D.O. por sus siglas en inglés), de Altas Scienfic es el elemento que determina el funcionamiento de la bomba (cuando el agua se encuentra fuera de los niveles de oxígeno disuelto adecuados), Es capaza de medir en rangos de 0.01 to +35.99 mg/L y se conecta directamente al raspberry 3.

Módulo de banda ancha móvil

Es un módulo USB que permite la transmisión de datos de manera remota, de tal manera que sea posible monitorear el nivel de oxígeno disuelto de manera remota, y de ser necesario activar o desactivar la bomba desde cualquier lugar con acceso a internet.

Metodología a desarrollar

Una vez instalado el OS en la raspberry pi 3 (ver apartado 3), se montan el módulo de expansión y el sensor de oxígeno disuelto (esto con el raspberry apagado). Al iniciar Raspbian, se busca el compilador Python, si es necesario se instala (Ver apartado 5). Se activa el bus I^2C (ver apartado 6).En un editor de textos (Word, block de notas o editor de textos de Python) se copia el código fuente del apartado 7 (es importante copiar exactamente igual ya que de otra forma pudiera generar errores), se guarda con el nombre *oxigenador.py*.

Se requiere que este programa se ejecute de manera automática al iniciar la raspberry pi 3, por lo que se deberá seguir las instrucciones del apartado 8.

En este punto el proceso ya está automatizado, ahora se debe instalar el elemento de energía sustentable como lo indica la figura 2. Lo más cercano al estanque ya que la bomba tiene que estar conectado a este, y deberá estar sumergida en el estanque (3 mts. max, utilizar cable cal. 8AWG). El raspberry pi 3 deberá estar montado es su base fuera del agua y lo más cercano al elemento de energía sustentable. Al montar correctamente el sistema de oxigenación se debe formar una fuente de agua. Esta fuente deberá activarse o desactivarse según las necesidades del estanque. Si la batería no tiene carga, deben bastar un par de horas de sol para que empiece a funcionar el sistema

Este equipo se probó en conjunto a 2 estanque más durante 3 meses. Uno patrón donde se utilizó el método de oxigenación convencional (abrir y cerrar compuertas para la entrada y salida de agua), y otro en donde el sistema de oxigenación de agua fue un aireador de paletas que necesita de un tendido eléctrico para funcionar. En estos dos casos se requirió de personal para monitorear el nivel de oxígeno disuelto en el estanque, esta medida se realizó con un medidor portátil do veces por día, mañana y tarde como lo recomienda el manual de buenas prácticas de manejo para el cultivo de camarón. La medición de oxígeno disuelto con el sistema de oxigenación fue instantánea.

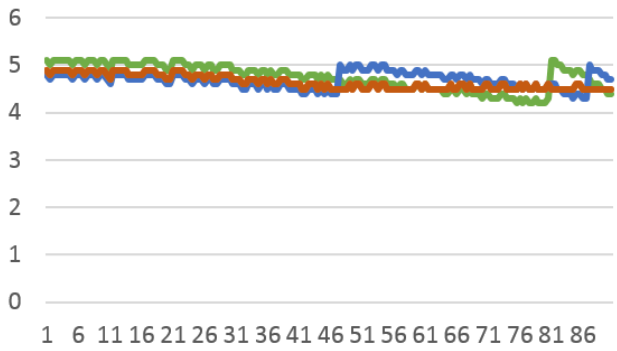
Se utilizó tilapia para hacer prueba, ya que la inversión es mucho menor comparada con la inversión del camarón, además de haber veda de camarón entre el 1 de mayo y el 15 de agosto de 2017. La tilapia crece y se desarrolla en parámetros que coinciden con los del camarón blanco.

ISSN-2410-3993

Antes de iniciar se vaciaron los estanques y se limpiaron, quedando libre de fitoplancton que pudieran cambiar el nivel de oxígeno disuelto del estanque.
La tasa de siembra en este sistema fue de 3,000 peces (alevín de aprox. 5 gramos).

Resultados

La gráfica 1 muestra los niveles de DO monitoreados en los 3 estanques. Línea verde DO en estanque patrón. Línea Azul DO en estanque con aireador. Línea guinda DO con sistema de oxigenación.



Gráfica 1 Niveles de DO en los estanques de prueba

En la gráfica 1 se puede observar que el patrón requirió únicamente de un recambio de agua que se dio cuando el nivel de DO llegó a los 4.4mg/l en el día 81. El estanque que utilizó aireadores de paleta tuvo que oxigenarse en dos ocasiones, cuando el nivel de DO llegó a los 4.4mg/l en el día 47, y a 4.3 en el día 86.

El estanque donde se probó el sistema de oxigenación, muestra al inicio el mismo comportamiento de consumo de DO pero una vez que llega a los 4.5mg/l de DO ya no permite que baje más (si subió), haciendo de este sistema el más estable en cuanto a niveles de DO.

Se observa que el nivel de DO se ve afectado más rápidamente conforme crece la tilapia, es decir se requieren recambios de agua más frecuentemente o poner en funcionamiento el aireador de paletas. En cuanto al sistema de oxigenación se hace evidente su constante funcionamiento.

Anexos

Apartado 1 Requerimientos de funcionamiento en granjas camaronícolas

El cultivo se lleva a cabo en estanques rústicos de tierra o forrados con geomembrana de alta densidad, conocida como liner, cuyas dimensiones pueden variar entre 0.2 hasta 10 ha. Tanques circulares de geomembrana. Estos están ubicados en lugares alejados de manchas urbanas o rurales que es donde se encuentra el agua con mejor calidad, por lo que no cuentan con línea eléctrica.

El Promedio de flujo de agua para el cultivo o la "Tasa de recambio de agua" (TRA, en %) depende del sistema utilizado: extensivo, 5 - 10%; semi-intensivo, 10 - 20%; intensivo, >20%.

La densidad de siembra para un sistema extensivo: 4 - 10 PL/m²; y para un sistema semi-intensivo: 10 - 30 PL/m²; intensivo: 60 - 300 PL/m²; híper-intensivo: 300 - 450 PL/m².El porcentaje de sobrevivencia para un sistema Extensivo: 50%; para un sistema semi-intensivo e intensivo es de 75%, y para un sistema hiper-intensivo es de 80 a 85%.

Parámetro	Min	Max	Promedio
Temperatura (°C)	20	35	28
Salinidad (ups)	5	40	35
Oxígeno disuelto (mg/l)	4	10	6

Tabla 2 Parámetros Físicos del agua de cultivo de camarón

ISSN-2410-3993

Parámetro	Rango
PH	7-9
Nitrito	<0.1 mg/l
Nitrato	0.4 0.8 mg/l
Amonio	0.1 a 1 mg/l
Turbidez	35 a 45 cm
Alcalinidad	100 a 140 mg/l

Tabla 3 Parámetros Químicos del agua de cultivo de camarón

Apartado 2 Principales enfermedades que afectan al camarón Blanco

Las enfermedades reportadas: Síndrome de Taura (TSV); Virus de la mancha blanca (WSSV); Virus de la cabeza amarilla (YHV); Baculovirosis tetraédrica (Baculovirus penaei BP); Virus de la necrosis hipodérmica y hematopoyética infecciosa (IHHNV); Litopenaeus vannamei nodavirus (LvNv); virus de la necrosis de la glándula digestiva (BMN); enfermedad viral del órgano linfoide del tipo parvovirus (LPVD) y enfermedad de la vacuolización del órgano linfoide (LOVD).

En el 2009, se detectó la presencia de la Mancha Blanca en el Estado de Tabasco, lo cual deja el precedente de la presencia del virus en el Golfo de México. Actualmente, el virus de la mancha blanca (WSSV), se presenta en los Estados de Sonora, Nayarit y Sinaloa, lo cual ha causado grandes pérdidas en la producción de camarón, como es el caso de Sonora en 2010 y 2011.

Apartado 3 Instalación del Raspbian OS en raspberry pi 3

Lo primero es descargar la imagen del OS requerido, en este caso Raspbian Jessie, Release 2017-03-02, se recomienda se realice esta descarga desde la página oficial www.raspberrypi.org, en la pestaña de DOWNLOADS. Se requiere una memoria microSD de al menos 4G de capacidad (libres).

La imagen se descarga en formato ZIP por lo que se recomienda el uso la aplicación 7-Zip (Windows), The unarchiver (Mac) o Unzip (Linux). Una vez descargada se extrae el archivo con extensión *img*.

Se requiere de un programa para instalar la imagen del OS en la memoria micro SD. El fabricante recomienda utilizar *Etcher* (funciona en win, mac y Linux). Este programa se puede descargar de manera gratuita desde https://etcher.io/.

Una vez descargado, se procede con la instalación de este último programa. Se inserta la memoria micro SD en la computadora donde se instaló el programa *Etcher*. Para instalar la imagen de ejecuta *Etcher* y desde esta aplicación se selecciona la imagen descargada con extensión *.img* y se selecciona la microSD que se insertó. Se da clic en "flash", al finalizar la instalación se tiene listo el OS en la micro SD.

Ahora esta memoria se inserta en la raspberry pi 3 y se conectan los periféricos y por último la fuente. En este momento en el monitor conectado en la raspberry pi 3 se visualizará el sistema operativo raspbian.

Apartado 4 Activación del SSH en Raspbian

SSH™ (o Secure SHell) es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente.

A diferencia de otros protocolos de comunicación remota tales como FTP o Telnet, SSH encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas.Para su activación ejecute una terminal y siga los siguientes pasos:
sudo apt-get install ssh

ISSN-2410-3993

Esto solo es necesario si no está instalado el servicio Inicie el servicio con el siguiente comando:

sudo /etc/init.d/ssh start
sudo /etc/init.d/ssh start

Y para que se ejecute automáticamente al iniciar el Raspberry Pi, ingrese el siguiente código:

sudo update-rc.d ssh defaults

Lo siguiente es conocer la dirección IP del raspberrypi. En una terminal se escribe lo siguiente:

sudo ifconfig

Ahora aparece una serie datos, en el campo wlan0 (en caso de tener una red inalámbrica) o eth0 (en caso de ser por cable Ethernet), veremos que aparece un campo llamado inet addr:XXX.XXX.X.X, siendo X dígitos numéricos. Por ejemplo, 192.168.1.3 sería una hipotética dirección IP. Anótala puesto que te hará falta para pasos posteriores.

IP Config
Una vez instalada, en la parte inferior de la aplicación aparece un campo así:
pi@raspberrypi:20
Luego pedirá el password o contraseña de nuestra sesión de Raspbian
Login: pi
Pass: raspberry
Listo

Apartado 5 Instalar Python en Rasbian

Este paso solo será necesario si no se utiliza raspbian OS o raspbian no trae instalado Python instalado o le falta alguna librería.

sudo apt-get update
sudo apt-get install python3-picamera

Si se requiere desinstalar, utilicé el siguiente comando:

sudo apt-get remove python3-picamera ó
sudo apt-get purge python3-picamera

Apartado 6.- Bus I²C

El I²C-bus es un popular y potente bus utilizado para la comunicación entre un maestro (o varios maestros) y uno o varios dispositivos esclavos. La figura 3 ilustra cómo diferentes periféricos pueden compartir un bus que está conectado a un procesador a través de sólo 2 hilos, que es uno de los mayores beneficios que el bus I²C puede dar cuando se compara con otras interfaces.

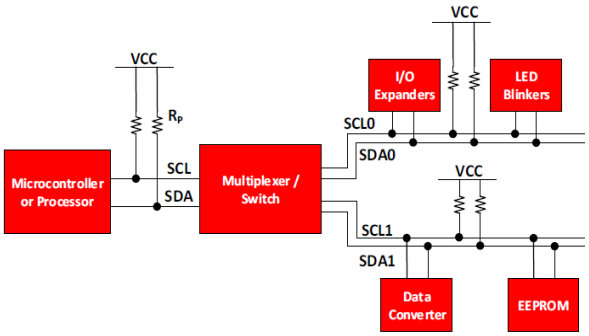


Figura 3 Ejemplo de conexión Bus I²C

Para activar el I²C en el raspberry pi 3 se ejecuta en la terminal el siguiente comando

sudo raspi-config

Lo cual abrirá un menú en el que se debe seleccionar la opción:

P5 I2C Enable/Disable automatic loading

Se reinicia el raspberry pi 3 y en la terminal se ejecutan los siguientes comandos:

sudo apt-get update
sudo apt-get upgrade

Y es necesario instalar I2C tools con el siguiente comando desde la terminal:

```
sudo apt-get install python-smbus
sudo apt-get install i2c-tools
```

Y se reinicia nuevamente la raspberry pi 3 Para verificar que está funcionando correctamente el bus I^2C se ejecuta en la terminal el siguiente comando (con el sensor DO conectado):

```
sudo i2cdetect -y 1
```

Si funciona correctamente, en la pantalla se visualizará una tabla con todos los dispositivos I^2C conectados a la raspberry pi 3 (solo uno en este caso).

Apartado 7 Código fuente

El código fuente son las instrucciones que el raspberry pi 3 debe realizar. En este caso las instrucciones son leer el sensor de oxígeno disuelto (bus I^2C), comparar el dato obtenido con los valores de 4 a 10 mg/l que requiere el camarón (tabla 2), sin embargo para no poner en riesgo la biomasa, se dejará un rango de 4.5 a 10 mg/l. Además, con la activación de SSH, se puede conectar de manera remota desde cual quiere Smartphone y monitorear y controlar de manera remota el sistema.

El código fuentes es:

```
#!/usr/bin/python

import io        # used to create file streams
import fcntl     # used to access I2C parameters like addresses

import time      # used for sleep delay and timestamps
import string    # helps parse strings


class AtlasI2C:
        long_timeout = 1.5      # the timeout needed to query readings and calibrations
        short_timeout = .5      # timeout for regular commands
        default_bus = 1         # the default bus for I2C on the newer Raspberry Pis, certain older boards use bus 0
        default_address = 98    # the default address for the sensor
        current_addr = default_address

        def __init__(self, address=default_address, bus=default_bus):
                # open two file streams, one for reading and one for writing
                # the specific I2C channel is selected with bus
                # it is usually 1, except for older revisions where its 0
                # wb and rb indicate binary read and write
                self.file_read = io.open("/dev/i2c-"+str(bus), "rb", buffering=0)
                self.file_write = io.open("/dev/i2c-"+str(bus), "wb", buffering=0)

                # initializes I2C to either a user specified or default address
                self.set_i2c_address(address)

        def set_i2c_address(self, addr):
                # set the I2C communications to the slave specified by the address
                # The commands for I2C dev using the ioctl functions are specified in
                # the i2c-dev.h file from i2c-tools
                I2C_SLAVE = 0x703
                fcntl.ioctl(self.file_read, I2C_SLAVE, addr)
                fcntl.ioctl(self.file_write, I2C_SLAVE, addr)
                self.current_addr = addr

        def write(self, cmd):
                # appends the null character and sends the string over I2C
                cmd += "\00"
                self.file_write.write(cmd)

        def read(self, num_of_bytes=31):
```

```
            # reads a specified number of bytes from I2C, then parses and displays the result
            res = self.file_read.read(num_of_bytes) # read from the board
            response = filter(lambda x: x != '\x00', res)  # remove the null characters to get the response
            if ord(response[0]) == 1:             # if the response isn't an error
                # change MSB to 0 for all received characters except the first and get a list of characters
                char_list = map(lambda x: chr(ord(x) & ~0x80), list(response[1:]))
                # NOTE: having to change the MSB to 0 is a glitch in the raspberry pi, and you shouldn't have to do this!
                return "Command succeeded " + ".join(char_list)     # convert the char list to a string and returns it
            else:
                return "Error " + str(ord(response[0]))

        def query(self, string):
            # write a command to the board, wait the correct timeout, and read the response
            self.write(string)

            # the read and calibration commands require a longer timeout
            if((string.upper().startswith("R")) or

        (string.upper().startswith("CAL"))):
                time.sleep(self.long_timeout)
            elif string.upper().startswith("SLEEP"):
                return "sleep mode"
            else:
                time.sleep(self.short_timeout)

            return self.read()

        def close(self):
            self.file_read.close()
            self.file_write.close()

        def list_i2c_devices(self):
            prev_addr = self.current_addr # save the current address so we can restore it after
            i2c_devices = []
            for i in range (0,128):
                try:
                    self.set_i2c_address(i)
                    self.read()
                    i2c_devices.append(i)
                except IOError:
                    pass
            self.set_i2c_address(prev_addr)        # restore the address we were using
            return i2c_devices
def main():
        device = AtlasI2C()        # creates the I2C port object, specify the address or bus if necessary

        print(">> Atlas Scientific sample code")
        print(">> Any commands entered are passed to the board via I2C except:")
        print(">>    List_addr lists the available I2C addresses.")
        print(">>    Address,xx changes the I2C address the Raspberry Pi communicates with.")
        print(">>    Poll,xx.x command continuously polls the board every xx.x seconds")
        print(" where xx.x is longer than the %0.2f second timeout." % AtlasI2C.long_timeout)
        print(">> Pressing ctrl-c will stop the polling")

        # main loop
        while True:
            input = raw_input("Enter command: ")

            if
input.upper().startswith("LIST_ADDR"):
                devices                    =
device.list_i2c_devices()
                for i in range(len (devices)):
                    print devices[i]

            # address command lets you change which address the Raspberry Pi will poll
            elif
input.upper().startswith("ADDRESS"):
                addr    =    int(string.split(input, ',')[1])
                device.set_i2c_address(addr)
                print("I2C address set to " + str(addr))

            # continuous polling command automatically polls the board
            elif input.upper().startswith("POLL"):
```

```
                delaytime = float(string.split(input, ',')[1])

                # check for polling time being too short, change it to the minimum timeout if too short
                if delaytime < AtlasI2C.long_timeout:
                        print("Polling time is shorter than timeout, setting polling time to %0.2f" % AtlasI2C.long_timeout)
                        delaytime = AtlasI2C.long_timeout

                # get the information of the board you're polling
                info = string.split(device.query("I"), ",")[1]
                print("Polling %s sensor every %0.2f seconds, press ctrl-c to stop polling" % (info, delaytime))
                try:
                        while True:

        print(device.query("R"))

        time.sleep(delaytime - AtlasI2C.long_timeout)
                except KeyboardInterrupt:
        # catches the ctrl-c command, which breaks the loop above
                        print("Continuous polling stopped")

            # if not a special keyword, pass commands straight to board
            else:
                if len(input) == 0:
                        print "Please input valid command."
                else:
                        try:

        print(device.query(input))
                        except IOError:
                                print("Query failed \n - Address may be invalid, use List_addr command to see available addresses")

if __name__ == '__main__':
        main()
```

Apartado 8 Como ejecutar un programa desde el inicio des raspbian OS en Raspberry Pi 3

Hay que otorgar permisos de ejecución 0755 o rwxr-x al programa *oxigenador.py* desde la terminal.Hay que guardar el archivo *oxigenador.py* en /etc/init.d/. Nuevamente se otorgan los mismos permisos: 0755 ó rwxr-x desde la terminal.

Se ejecuta el siguiente comando desde la terminal

sudo update-rc.d /etc/init.d/oxigenador.py defaults

En caso de ser necesario se puede desinstalar con el siguiente comando:

sudo update-rc.d -f /etc/init.d/oxigenador.py remove

Se reinicia la raspberry pi 3 y deberá ejecutarse el programa de manera automática.

Agradecimiento

Al Programa para el Desarrollo Profesional Docente, para el tipo Superior, ya que financió el desarrollo de este proyecto mediante: Fortalecimiento de Cuerpos Académicos Convocatoria 2015, donde se aprobó el proyecto Sistema de Oxigenación de Granjas de Cultivo de Camarón Mediante la Electrificación Por Medio de Paneles Solares En Estanqueria Rustica O Forrados Con Liner En el Estado de Tamaulipas con IDCA 21018, y clave UPALT-CA-7.

Conclusiones

Con base en los resultados, se puede concluir que el sistema de oxigenación proporciona niveles de DO estables aun cuando la biomasa en el estanque consume más rápidamente el DO en el agua.

Quedan algunas incógnitas como área máxima de oxigenación, tiempos de oxigenación por unidad de área, y un sistema de control con amortiguamiento, ya que el sistema se activaba y desactivaba continuamente. Esto debido a que estas incógnitas surgen a partir del desarrollo de este proyecto.

Referencias

Atlas Scientific. (7/25/17). Dissolved Oxygen EZO circuit. Environmental Robotics, 3.7, 68

Atlas scientific. (12/07/2017). Dissolved Oxigen probe. Environmental Robotics, 2.5, 7.

Chávez, M. & Higareda, I. (2003). Manual de Buenas Prácticas de Producción Acuícola de Camarón para la Inocuidad Alimentaria. Mazatlán, Sinaloa, México: Centro de Investigación en Alimentación y Desarrollo, A.C.

Comisión Nacional de Acuacultura y Pesca. (31 de diciembre de 2015). Anuario Estadístico de Acuacultura y Pesca. Anuario Estadístico de Acuacultura y Pesca 2013, 1, 299. 28 de julio de 2017, De conapesca Base de datos.

García, A & Calvario, O. (2008). Manual de Buenas Prácticas de Producción Acuícola de Tilapia para la Inocuidad Alimentaria. Mazatlán, Sinaloa, México: Centro de Investigación en Alimentación y Desarrollo, A.C.

Greg Loyse. (17/07/2017). Release. raspberry-pi Documentation, 0.0, 37.

James Adams. (4/04/2016). Reduced Schematics. Raspberry Pi 3 Model B, 1.2, 1.

Órgano del Gobierno Constitucional de los Estados Unidos Mexicanos. (9/09/2013). Actualización de la carta nacional acuícola. Diario Oficial de La federación, 68.

Rojas, A., Haws, M. & Cabanillas, J. (2005). Buenas Prácticas de Manejo Para el Cultivo de Camarón. The David and Lucile Packard Foundation. United States Agency for International Development (Cooperative Agreement No. PCE-A-00-95- 0030-05).

Raspberry Foundation. (2012). INSTALLATION. 9/01&2017, de Raspberry Foundation Sitio web: https://www.raspberrypi.org/documentation/installation/

Romero M. (2010). ENERGIA SOLAR FOTOVOLTAICA. Málaga, España: CEAC.

Saavedra M. (2006). Manejo del cultivo de tilapia. Managua, Nicaragua: university of hawaii Hilo

The Raspberry Pi Foundation. (5/07/2017). RASPBIAN JESSIE WITH DESKTOP. 10/07/2017, de Oracle Binary Code Sitio web: https://www.raspberrypi.org/downloads/raspbian/

Valdez J & Becker J. (2015). Understanding the I2C Bus. Dallas, Texas: Texas Instruments Incorporated.

Whiteboxes labs. (2017). Tentacle Documentation. 30/06/2017, de Whitebox Labs Sitio web: https://www.whiteboxes.ch/tentacle/#tentacle-t3

ISSN-2410-3993