Sistema autónomo de video conferencia motorizado - Implementación sobre Raspberry PI

Daniel Giulianelli, Pablo Vera, Rocío Rodríguez, Gabriela Vallés, Mariano Dogliotti

Universidad Nacional de La Matanza
Departamento de Ingeniería e Investigaciones Tecnológicas
San Justo, Buenos Aires, Argentina
{dgiulian, pvera, rocio.rodriguez, gvalles, mdogliotti}@unlam.edu.ar

Resumen. En el presente artículo se plantea la importancia de desarrollar una solución para Video Conferencia que ofrezca un sistema integral con un hardware dedicado de bajo costo, el cual a través de distintos componentes permita la motorización de una cámara que seguirá al orador cuando este se desplaza por su entorno. Permitiendo de este modo poder realizar video conferencias interactivas con manipulación de objetos en donde siempre el foco de atención esté dirigido al orador. Se presenta la estrategia para la construcción del sistema y las selecciones de hardware que resultan necesarias para llevar a cabo el mismo.

Palabras Clave: Video Conferencia, Hardware Dedicado, Raspberry Pi, Sensores, Motorización

1 Introducción

Los sistemas de videoconferencia son cada vez más populares, hoy existen herramientas de uso masivo como por ejemplo Skype de Microsoft, Hangout de Google, que permiten entablar videoconferencias rápidamente incluso con varias personas en simultáneo. Sin embargo, muchas veces estos sistemas se ven afectados por una pobre performance y falta de calidad al momento de realizar una videoconferencia, principalmente cuando la misma es de una duración considerable. Existen empresas como Cisco [1] o Polycom [2] que se dedican a realizar software y hardware específico brindando soluciones de gran calidad pero con el inconveniente de los altos precios para poder acceder a estos equipos. Es por eso que se busca crear un equipo de videoconferencia dedicado que tenga características profesionales y que además pueda ser implementado con hardware de bajo costo.

Los requerimientos del sistema son: (a) Solución autónoma que permita conectarla directamente a un televisor o monitor para uso; (b) Inicio automático de la aplicación con solo encender el equipo; (c) Control por medio de hardware sencillo como un teclado para smart tv o un control remoto; (d) Fácil configuración inicial; (e) Cámara que permita mediante motorización cambiar el ángulo de visión o seguir al orador automáticamente mientras este se mueve; (f) Conexión directa por IP sin necesidad de un servidor central; (g) Posibilidad de utilización de IPv6 para mejor

performance; (h) Posibilidad de realización de videoconferencia entre varias personas al mismo tiempo; (i) Video fluido con alta calidad de imagen.

Esto trae aparejado el desafío de planificar la utilización de hardware dedicado que permita implementar el sistema autónomo de video-conferencia (VC) motorizado, considerando Cámara, Sensores, Servomotores y Hardware específico donde poder ejecutar la solución.

El desarrollo del sistema de video conferencia en un dispositivo dedicado de tamaño reducido permite crear una solución de fácil implementación ya que con solo conectarla a un televisor o monitor ya podrá utilizarse sin necesidad de poseer conocimientos adicionales.

Al tener un sistema dedicado en el hardware el software debe contemplar los siguientes puntos: (a) El sistema debe iniciar automáticamente al iniciar el equipo; (b) No debe permitir salir del sistema; (c) En caso que por alguna razón como un error desconocido el sistema se cerrará, se deberá disponer de un proceso automático que intente volver a iniciarlo y si no es posible reiniciar el equipo; (d) Se debe diseñar una interfaz amigable para cualquier tipo de usuario tomando en cuenta que puede tener pocos o nulos conocimientos en el manejo de tecnología; (e) La interfaz debe estar diseñada para ser manejada por dispositivos sencillos (inicialmente se desarrolla una interfaz para los teclados de smart-tv y una segunda versión incorporará el manejo desde un control remoto tradicional); (f) Al desconocer en qué tipo de pantalla se utilizará, se debe diseñar una interfaz de usuario que se visualice correctamente en cualquier resolución y tamaño; (g) La configuración inicial para el uso de la aplicación debe realizarse de la forma más sencilla posible, guiando al usuario en cada paso (por ejemplo al configurar la red wifi).

2 Selección del Hardware

Se analizaron distintas plataformas en las cuales podría desarrollarse la solución planteada y se opta por Raspberry Pi debido a su mayor poder de cómputo sumado a la posibilidad de utilizar una cámara integrada de alta resolución. La raspberry además cuenta con puertos especializados para la comunicación directa con hardware externo lo que facilita las conexiones necesarias para poder realizar la automatización de la cámara.

La Raspberry Pi 2 Model B [3] tiene las características en cuanto a hardware que se presentan en la tabla 1.

Tabla 1. Características Raspberry PI 2 Model B

RAM	1 GB
Almacenamiento	Micro SD
Procesador	ARM A7
Velocidad	900 Mhz
Ethernet	Sí
Salida HDMI (audio y video)	Sí
Nº puertos USB	4

Soporta instalación de S.O.	Sí (Linux)
Cantidad de GPIO (puertos útiles)	40
Lenguajes de programación soportados	Python, C, C++, Java, Perl, Ruby, etc
Alimentación	5v
Tamaño	85 x 56 x 17mm

En una memoria SD es posible instalar un sistema operativo como Linux y luego un programa que permita realizar algo en particular, en el caso del presente proyecto una aplicación de video-conferencia, desarrollada específicamente para este hardware. Cabe destacar que actualmente ya existe una versión superior en el mercado de la raspberry pi que incluye mejoras en hardware y conectividad wifi incorporada, dicha versión fue lanzada con posterioridad a la adquisición del hardware para realizar el proyecto. De todos modos la versión adquirida tiene las prestaciones necesarias para poder implementar la solución.

2.1 Cámara

Cada uno de los nodos está implementado con una solución de hardware que incluye una Raspberry Pi 2 Model B. Cada una de ellas se conectará por HDMI a un monitor o televisor, se le agregó a la solución una cámara interna usando el puerto SCI de la Raspberry Pi (ver figura 1).

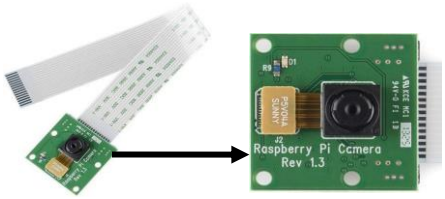

Fig. 1. Módulo de Cámara – Puerto SCI

La ventaja de la cámara integrada mediante el puerto SCI es la velocidad y la calidad de captura de imagen y video. El gabinete de la Raspberry trae un zócalo en donde se inserta la placa correspondiente al módulo de la cámara (ver figura 2). Los modos de videos soportados por la cámara integrada son: 1080p 30fps, 720p 60fps y 640x480p 60/90fps

Fig. 2. Módulo de Cámara conectado a la Raspberry Pi 2 Model B

2.2 Sensores

Se incorporan diversos módulos y elementos adicionales a la Raspberry Pi: Módulo de WIFI por el cual se establecerá la conexión a internet; Sensores (sonido y

ultrasonido – ver figura 3), los cuales se conectan a un puerto denominado GPIO (General Purpose input/out) especialmente diseñado para operaciones de entrada/salida. Este puerto tiene distintos pines que pueden utilizarse para controlar fácilmente tanto sensores como actuadores.

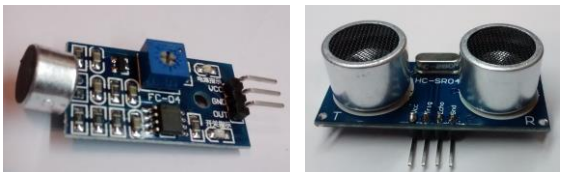

Fig. 3. Sensor sonido y ultrasonido

El sensor ultrasónico HC-SR04 para Raspberry Pi es un sensor que permite medir la distancia entre un objeto y el sensor. Tiene dos cilindros uno de ellos transmite un pulso de ultrasonido (de alta frecuencia, no audible por el oído humano), el cual rebotará contra el objeto que está frente al sensor y será recibido (dicho rebote) por el otro cilindro. Cuanto más cerca esté el objeto, el tiempo de recepción de dicho pulso será menor y por medio de ese tiempo se puede calcular la distancia con el objeto en cuestión, en un área de 3cm a 3m (con una precisión de 3 mm).

2.3 Motorización

Los sensores permitirán detectar la posición del orador dentro de una sala y motorizar la cámara para que pueda moverse siguiendo al orador. Para mover la cámara se agregan dos servomotores (vertical y horizontal) los cuales moverán la cámara (ver figura 4).

Fig. 4. Base de montura para la Raspberry PI con la cámara integrada – Sensores en dos Ejes

Los servos son motores de corriente continua (CC), pero no pueden lograr un giro continuo de 360º, están preparados para moverse a un ángulo fijo en respuesta a una señal de control, y mantenerse fijos en dicha posición. “Un servo principalmente está formado por un conjunto reductor (engranajes), un motor de CC y por último por un circuito de control, aunque en la práctica se comporta como un bloque funcional que posiciona su eje en un ángulo preciso en función de la señal de control” [4].

Normalmente estos pequeños servos funcionan con una tensión de alimentación de 5V y a su vez la señal digital de control puede ser de un nivel de tensión entre 3V y 7,2V, pero el valor de voltaje recomendable para evitar errores de detección del "1 lógico" es de 5V. El control se realiza mediante una señal modulada por ancho de pulso, en la que el ancho el pulso indica el ángulo que se desea que adopte el eje. La modulación por ancho de pulso, PWM (Pulse Width Modulation), es uno de los sistemas más empleados para el control de motores. Este sistema consiste en generar una onda cuadrada en la que se varía el tiempo que el pulso está en el nivel alto, manteniendo el mismo período (normalmente), con el objetivo de modificar la posición del servo según se desee (ver figura 5).

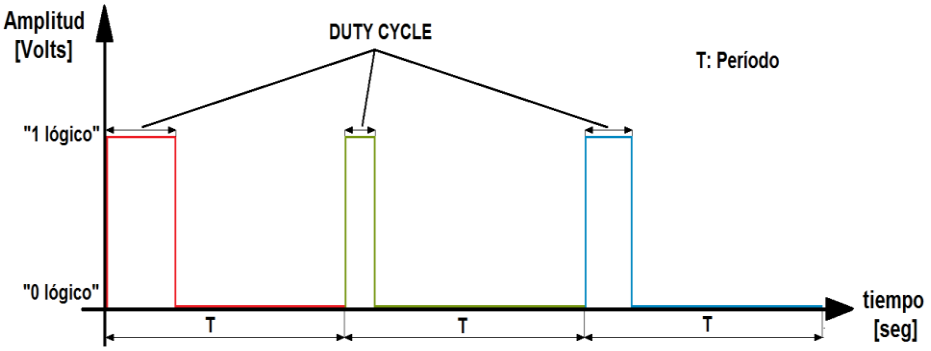

Fig. 5. Ejemplo de modulación por ancho de pulso (PWM)

Por lo tanto, se enviará un tren de pulsos por la entrada de control del servomotor tal que irá variando la duración (anchura) del pulso activo en función al ángulo de giro deseado del motor. Los pulsos donde la señal de PWM se mantiene en "1 lógico" se denomina "Duty Cycle".

Se necesitan valores específicos de frecuencia en la señal de control. Los valores más generales se corresponden con períodos de la señal cada 20 ms, y pulsos en nivel alto de entre 1 ms y 2 ms de anchura. En otras palabras, se trataría de una señal de 50Hz de frecuencia con un duty cycle entre el 5% y el 10% del período.

Habitualmente cada servo tiene sus márgenes de operación, que se corresponden con el ancho del pulso máximo y mínimo con el que el mismo funciona. Mayormente pueden moverse entre un ángulo de 0º y 180º, aunque existen algunos modelos comerciales que permiten un giro de 360º. En el caso del modelo SG90 permite un ángulo de giro de 180º, con lo cual con un duty cycle de 1.5 ms indicaría la posición central o neutra (90º), con otro de 2 ms giraría 180º, con otro de 1 ms se movería a la posición de 0º, y mientras que otros valores del pulso lo dejarían en posiciones intermedias. En la figura 6, se demuestra gráficamente lo explicado.

Estos valores suelen ser los recomendados, sin embargo, es posible emplear pulsos menores de 1 ms o mayores de 2 ms, para poder conseguir ángulos mayores de 180°. Pero en la práctica esto no se corresponde con la teoría, ya que existen límites mecánicos propios del servomotor, que si se sobrepasan el servo comenzará a emitir un zumbido ya que estará ejerciendo una fuerza para intentar girar un ángulo mayor a 180º que el factor limitante mecánico no se lo permitirá. Es por ello que se debe evitar enviar señales de control de estas características para evitar problemas tales como:

pérdida de precisión de giro o desgaste mecánico producido más rápidamente de lo estimado.

La duración entre pulso y pulso en que la señal de PWM se encuentra en nivel bajo ("0 lógico") no es crítica, e incluso puede ser distinto entre uno y otro período de la señal. Generalmente se suelen emplear valores de período de la señal de aproximadamente 20 ms (entre 10 ms y 30 ms). Si el tiempo de duración del pulso en nivel bajo es inferior al mínimo, puede interferir con la temporización interna del servo, causando un zumbido, y la vibración del eje de salida. Si es mayor que el máximo, entonces el servo pasará a estado dormido entre pulsos. Esto provoca que se mueva con intervalos pequeños.

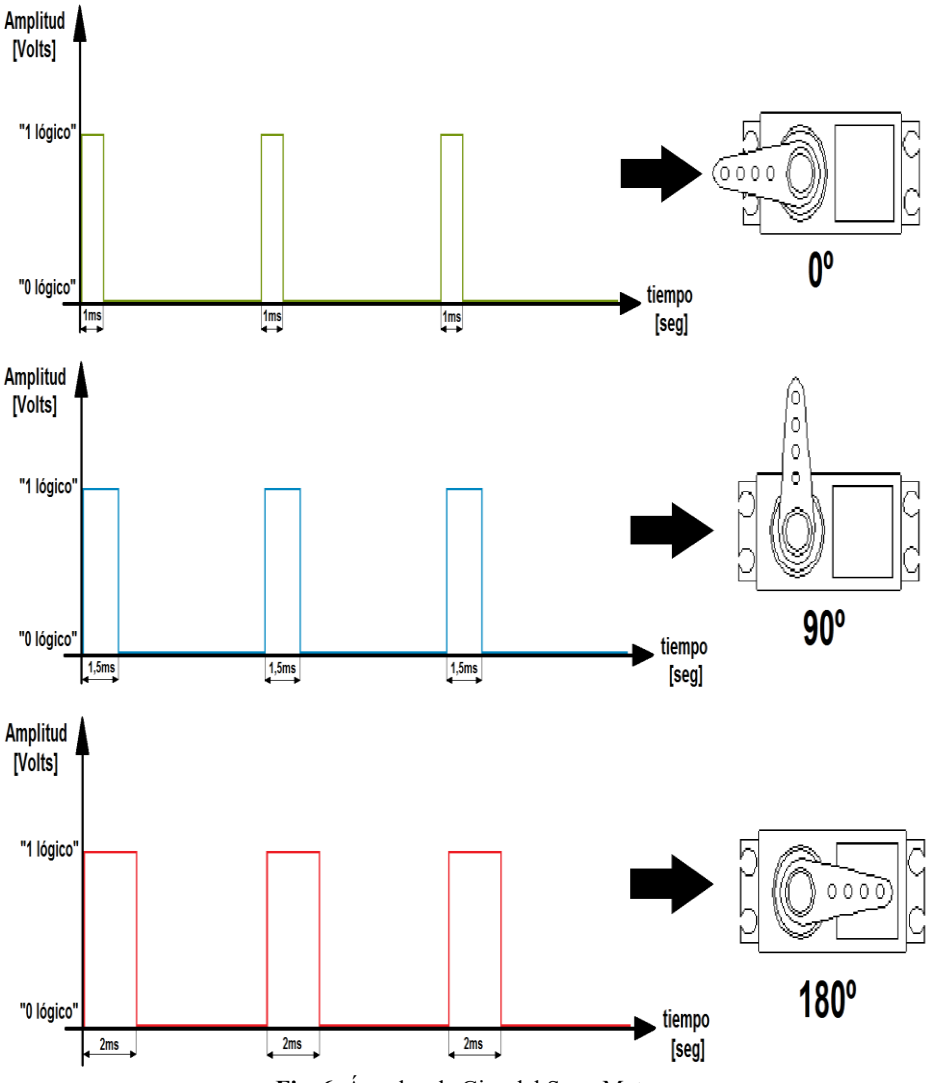

Fig. 6. Ángulos de Giro del ServoMotor

Es importante destacar que para que un servo se mantenga en la misma posición durante un cierto tiempo, es necesario enviarle continuamente el pulso correspondiente. De este modo, si existe alguna fuerza que le obligue a abandonar dicha posición, este intentará resistirse. De igual manera, si se deja de enviar pulsos (o el intervalo entre pulsos es mayor que el máximo) entonces el servo perderá fuerza y dejará de intentar mantener su posición, de modo que cualquier fuerza externa podría desplazarlo.

Un servo tiene un conector de 3 hilos, Alimentación de 5V (rojo), GND (negro o marrón) y el otro Control (amarillo o blanco). Además como se mencionó anteriormente funcionan con niveles de tensión de la señal de control entre 3V y 7,2V, por lo tanto esto nos permite utilizarlos fácilmente en placas de Arduino o Raspberry, ya que la primera trabaja con niveles de tensión de 5V y la segunda con 3,3V.

A continuación se presenta un fragmento de código en phyton que permite mover los servomotores cambiando la posición de la cámara.

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setup(21,GPIO.OUT)
p = GPIO.PWM(21,50)
p.start(7.5)
try:
    while True:

        p.ChangeDutyCycle(4.5)
        time.sleep(0.5)
        p.ChangeDutyCycle(10.5)
        time.sleep(0.5)
        p.ChangeDutyCycle(7.5)
        time.sleep(0.5)
except KeyboardInterrupt:
    p.stop()
GPIO.cleanup()
```

El servomotor utilizado es el modelo SG90, en la hoja de especificación de datos del fabricante puede observarse sus características, algunas de ellas se resumen en la tabla 2.

Tabla 2. Características del ServoMotor seleccionado (SG90)

Peso	9 g
Dimensiones	22.2 x 11.8 x 31 mm aprox.
Longitud del cable de Conector	24.5cm
Torque	1.8 kg / cm (a un voltaje de operación de 4.8V)
Velocidad de Operación	0.1 seg / 60º (a un voltaje de operación de 4.8V)
Voltaje de Operación	3.0V ~ 7.2V (valor recomendado 5V)
"Dead Band Width"	10 µs

Rango de Temperatura	-30ºC ~ 60ºC
Ángulo de Rotación	180º
Ancho de pulso	500-2400 µs

"Dead Band Width" significa Ancho de Banda Muerto y es el tiempo máximo hasta el cual el servomotor no detecta si hay una señal activa en su entrada de control. Por ejemplo, si se enviase una señal de control con un pulso activo de 1,5ms, entonces el servo se movería a su posición central (90º). Si ahora el pulso activo fuese de 1,5ms + 5µs, el servo seguiría en la misma posición que antes, es decir que no se moverá hasta que el pulso activo sea de 1,5ms ± 10µs.

Dead Band Width es un parámetro utilizado para evitar que el servo detecte señales de muy bajo duty cycle, ya que las mismas podrían tratarse de ruido y esto provocaría que el mismo se moviese sin haber enviado ninguna señal de control.

3 Desarrollo de Software de Video Conferencia

La raspberry pi tiene una aplicación de consola que permite capturar fácilmente el video de la cámara incorporada. Esta aplicación se denomina raspivid [5] y soporta numerosos parámetros de configuración entre los cuales se encuentra la calidad del video a capturar, los cuadros por segundo y la posibilidad de enviar el resultado a un archivo o a la salida estándar. Este último parámetro de configuración es justamente el que permite capturar el video desde una aplicación al enviarlo a la salida estándar.

Por ejemplo desde Java se puede utilizar la clase Process y asignarle al mismo la ejecución del comando raspivid con los parámetros deseados:

```
Process procesoRaspivid = Runtime.
Runtime.getRuntime().exec("raspivid -n -b 15000000 -w
1920 -h 1080 -fps 30 -t 0 -o -"");
```

Donde se invoca al comando raspivid con los parámetros presentados en la tabla 3.

Tabla 3. Parámetros del comando raspivid

-w 1920	Para establecer el ancho del video en 1920 pixel
-h 1080	Para establecer el alto del video en 1080 pixel
-b 15000000	Para establecer el bit rate a 15Mbit/s
-n	Evita que muestre una ventana con el video capturado
-fps 30	Establece la captura a 30 cuadros por segundo
-t 0	Captura el video indefinidamente
-o -	Envía el resultado a la salida estándar

Luego se puede capturar el flujo de video generado por el proceso mediante el comando:

```
InputStream flujoVideo;
flujoVideo = procesoRaspivid.getInputStream();
```

Al flujo de video capturado se lo divide en paquetes de información para que puedan ser enviados por la red:

```
byte[] datosVideo = new byte[4096];
while (flujoVideo.read(datosVideo) != -1)
{
    //envío de los paquetes por la red
}
```

El tamaño de los paquetes a enviar va a depender de la resolución del video y la cantidad de cuadros por segundo. Existen numerosos autores que trabajan sobre ese tema, entre ellos podemos nombrar [6], [7], [8]. Además puede adaptarse el tamaño de los paquetes de acuerdo al tiempo que tarden los paquetes en llegar a destino según la conexión particular de cada caso.

Para el manejo de las comunicaciones pueden utilizarse las bibliotecas estándar de java o puede facilitarse el proceso incorporando bibliotecas externas por ejemplo Eneter Messaging Framework [9] que es un framework para comunicación entre procesos multiplataforma gratuito para uso no comercial.

En la figura 7, se puede ver la solución en funcionamiento conectada a un monitor con HDMI.

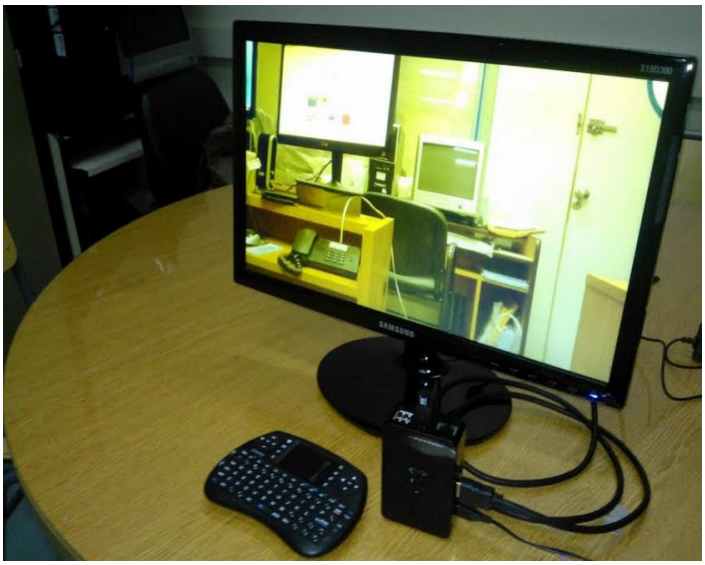

Fig. 7. Conectividad entre dos nodos en laboratorios de la Universidad

4 Conclusiones

Los beneficios de esta solución residen principalmente en la autonomía, que puede ser conectada por HDMI a una pantalla y manejada en forma remota con un teclado de smart tv. Crear esta solución requiere considerar diversos aspectos, que integran no sólo cuestiones de hardware o desarrollo de la aplicación, sino también considerar un

diseño de la interfaz general que le permita al usuario de forma amigable manejarse con el sistema. Actualmente se cuenta con tres nodos idénticos implementados, sobre los cuales en una memoria SD se ejecuta una versión puntual del sistema de video conferencias para IPv4 y se está trabajando en otra versión que incorporará soporte a IPv6. Aún queda por delante trabajar en poder manejar el sonido según lo lejos o cerca que se encuentra el orador, permitiendo automáticamente variar el volumen para obtener un volumen final adecuado, también se continúa trabajando en mejorar el seguimiento del orador por medio de los valores arrojados por el sensor de ultrasonido.

Referencias

1. Cisco, “Cisco TelePresence SX20 Quick Set” http://www.cisco.com/c/en/us/products/collaboration-endpoints/telepresence-sx20-quick-set/index.html
2. Polycom “HD Video Conferencia” http://www.polycom.com/hd-video-conferencing.html
3. Raspberry. (2016) “Raspberry Pi 2 Model B”. https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
4. del Campo García Mariano, “Programar posiciones en un Micro Servo Tower Pro SG90 9G”. (2016) http://miarduinounotieneunblog.blogspot.com.ar/2016/01/programar-posiciones-en-un-micro-servo.html
5. Raspberry. Raspivid. https://www.raspberrypi.org/documentation/usage/camera/raspicam/raspivid.md
6. B. Q. Zhao, J. C. S. Lui and D. M. Chiu, "Exploring the optimal chunk selection policy for data-driven P2P streaming systems," 2009 IEEE Ninth International Conference on Peer-to-Peer Computing, Seattle, WA, (2009), pp. 271-280. doi: 10.1109/P2P.2009.5284548 http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5284548&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D5284548
7. N. Ramzan *et al*., "Peer-to-peer streaming of scalable video in future Internet applications," in *IEEE Communications Magazine*, vol. 49, no. 3, pp. 128-135, March (2011). doi: 10.1109/MCOM.2011.5723810 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5723810&isnumber=5723785
8. JOHN Samson Mwela, OYEKANLU Emmanuel Adebomi. Impact of Packet Loss on the Quality of Video Stream Transmission. (2010) http://www.bth.se/com/mscee.nsf/attachments/6182_Thesis_Report_pdf/$file/6182_Thesis_Report.pdf
9. ENETER Messaging Framework. Cross-Plataform Framework for Interprocess Communication http://www.eneter.net/