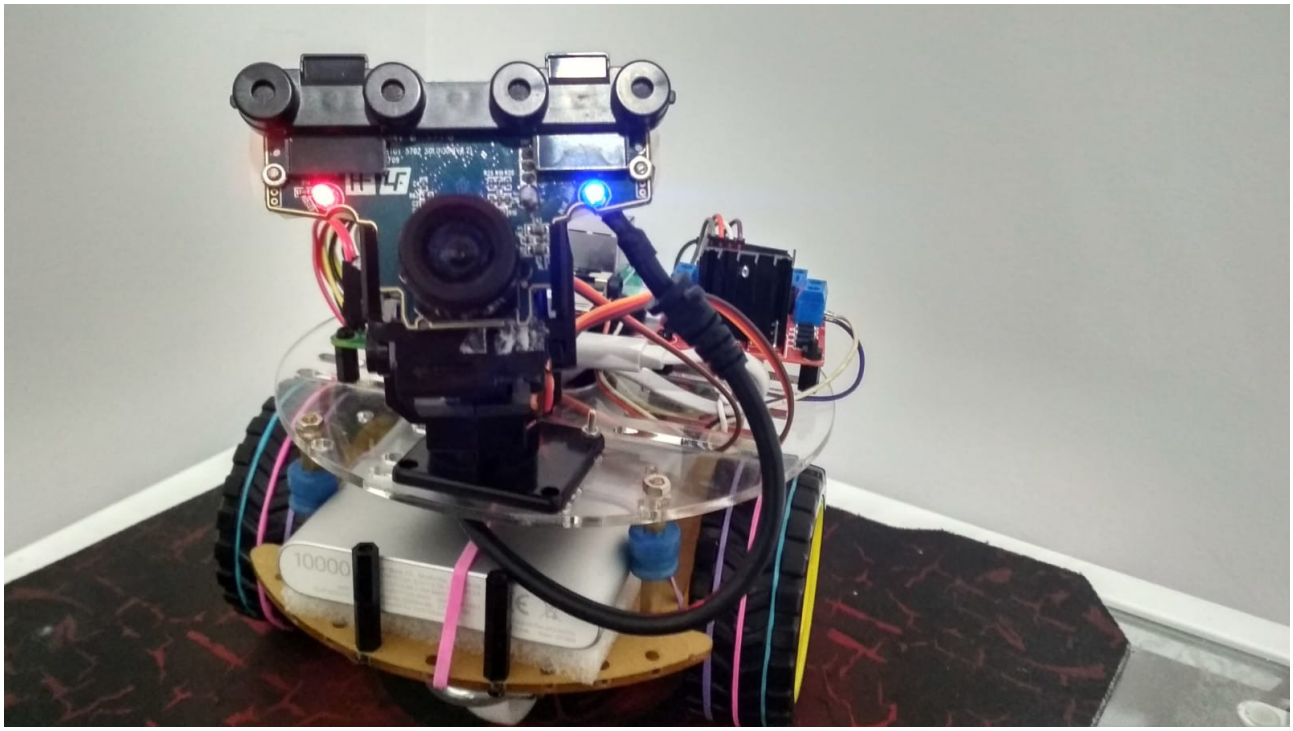
Proyecto IOT con React.js, Node y OpenCv sobre RaspberryPi



Jose Alberto Vilches Prada

Conceptos básicos del proyecto

¿Que es IOT?

Es la conexión a Internet de objetos cotidianos o no con Internet para realizar una funcionalidad.

La capacidad de conectar dispositivos embebidos con capacidades limitadas de CPU, memoria y energía significa que IoT puede tener aplicaciones en casi cualquier área, como la domotica, exploraciones, seguridad...

¿Que es una RaspberryPi?

Es un ordenador de placa reducida, de bajo coste desarrollado en el Reino Unido por la Fundación Raspberry Pi.

En este caso empleamos una RaspberryPi 2B antigua que tenia para pruebas y una 3B+ para el concepto final.

La RaspberryPi esta formada por su propio procesador, ram, puertos usb, wifi y bluetooth en algunos modelos y lo mas importante, GPIO.

El GPIO es un pin genérico de entrada/salida de propósito general que puede ser programado/controlado en el tiempo de ejecución por parte del usuario.

La RaspberryPi monta una imagen en su micro-sd de Raspbian Stretch Lite, una imagen mínima de una distro de Debian, para no malgastar los recursos de la placa, ya que esta es algo limitada.

Componentes mas característicos

Aparte de la RaspberryPi nos encontramos con un L298N (Doble Puente H)

Un Puente en H es un circuito electrónico que se usa para permitir a un motor eléctrico DC girar en ambos sentidos, *avance* y *retroceso*.

Un puente H se construye con 4 interruptores (mecánicos o mediante transistores). Cuando los interruptores S1 y S4 (ver primera figura) están cerrados (y S2 y S3 abiertos) se aplica una tensión positiva en el motor, haciéndolo girar en un sentido. Abriendo los interruptores S1 y S4 (y cerrando S2 y S3), el voltaje se invierte, permitiendo el giro en sentido inverso del motor.

En nuestro caso el componente monta dos puentes controlado por un chip.

Un Ps3 Sony EyeToy, que es una webcam a la que se le ha aligerado quitandole la carcasa y volviendo a soldar el cable mas corto.

Un PowerBank Xiaomi de 10.000mAh con doble salida usb, que permite alimentar la RaspberryPi, la webcam y demás componentes junto al L298N y los dos motores. Permitiendo una autonomía superior 4 horas.

El chasis que conforma a Tau donde se monta todo, proveniente de China que incluía los motores.

¿Que es Node.js?

Node.js es un entorno Javascript del lado del servidor, basado en eventos. Aprovechando el motor V8 permite a Node proporciona un entorno de ejecución del lado del servidor que compila y ejecuta Javascript a velocidades increíbles.

¿Que es un WebSocket?

Es una tecnología que proporciona un canal de comunicación bidireccional y full-duplex sobre un único socket TCP. Está diseñada para ser implementada en navegadores y servidores web, pero puede utilizarse por cualquier aplicación cliente/servidor.

Esto permite el intercambio en tiempo real, sin perdidas de paquetes entre el cliente y el servidor, algo muy adecuado para este proyecto ya que el servidor podría mandar infinidad de datos de sensores por ejemplo, al cliente.

¿Que es OpenCv?

OpenCv es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X, Windows y Android. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estérea y visión robótica.

El proyecto pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado realizando su programación en código C y C++.

En el proyecto se ha utilizado opencv4nodejs, un paquete que permite el uso de Javascript en el fantástico mundo de OpenCv, al principio TAU si era capar de reconocer caras y objetos, pero fue desechada esa idea tras comprobar que necesitaría o un cluster de RaspberryPi o una cámara mas idónea y cara para ello, por lo que esta primera versión de TAU, no incluye reconocimiento, pero OpenCv es utilizado en el para mostrar un streaming de su cámara web al cliente.

¿Que es React.js?

React es una librería Javascript focalizada en el desarrollo de interfaces de usuario. Esa es su principal área de trabajo, pero lo cierto es que con todo el ecosistema de aplicaciones y herramientas y componentes, con React encontramos un excelente aliado para hacer todo tipo de aplicaciones web, SPA (Single Page Application) o incluso aplicaciones para móviles.

Es por tanto una base sobre la cual se puede construir casi cualquier cosa con Javascript y que nos facilita mucho el desarrollo, ya que nos ofrece muchas cosas ya listas, en las que no necesitamos invertir tiempo para desarrollar.

Fue desarrollada por Facebook.

¿Como comenzar?

Lo primero es disponer de un ordenador, una RaspberryPi con una imagen de Raspbian Stretch Litle (donde tenemos configurado ya la localización, usuario, gpio remoto...) una conexión a Internet.

Conectamos todo a la misma red local, y mediante ssh podemos controlar la RaspberryPi por terminal.

Comprobar que todo esta configurado

sudo raspi-config

Actualizar las aplicaciones de la RaspberryPi.

$sudo apt-get update
$sudo apt-get upgrade

Instalaremos git y clonaremos el repositorio

$sudo apt-get install git

```
pi@TAU:~ $ sudo apt-get install git
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  git-man liberror-perl
Paquetes sugeridos:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-arch git-cvs git-mediawiki
  git-svn
Se instalarán los siguientes paquetes NUEVOS:
  git git-man liberror-perl
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 15 no actualizados.
Se necesita descargar 4.849 kB de archivos.
Se utilizarán 26,4 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

```
pi@TAU: ~
pi@TAU:~ $ git clone https://HysteriaShoot@bitbucket.org/HysteriaShoot/tauwebsocketopencv.git
```

Instalaremos Node.js y npm

$curl -slL hhtps://deb.nodesource.com/setup_11.x | sudo -E bash -
$sudo apt-get install -y nodejs

```
pi@TAU:~ $ curl -sL https://deb.nodesource.com/setup_11.x | sudo -E bash -
```

```
+ apt-get update
Des:1 https://deb.nodesource.com/node_11.x stretch InRelease [4.612 B]
Des:2 https://deb.nodesource.com/node_11.x stretch/main armhf Packages [764 B]
Obj:3 http://raspbian.raspberrypi.org/raspbian stretch InRelease
Obj:4 http://archive.raspberrypi.org/debian stretch InRelease
Descargados 5.376 B en 1s (3.242 B/s)
Leyendo lista de paquetes... Hecho

## Run `sudo apt-get install -y nodejs` to install Node.js 11.x and npm
## You may also need development tools to build native addons:
     sudo apt-get install gcc g++ make
## To install the Yarn package manager, run:
     curl -sL https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -
     echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /etc/apt/sources.list.d/yarn.list
     sudo apt-get update && sudo apt-get install yarn
```

Instalaremos cmake, necesario para buildear Opencv4nodejs

$sudo apt-get install cmake

```
pi@TAU: ~/tauwebsocketopencv/server-socket
pi@TAU:~/tauwebsocketopencv/server-socket $ sudo apt-get install cmake
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  gyp javascript-common libc-ares2 libhttp-parser2.8 libjs-inherits libjs-jquery libjs-node-uuid libjs-underscore
  libssl-dev libssl-doc libuv1-dev node-abbrev node-ansi node-ansi-color-table node-archy node-async
  node-balanced-match node-block-stream node-brace-expansion node-builtin-modules node-combined-stream
  node-concat-map node-cookie-jar node-delayed-stream node-forever-agent node-form-data node-fs.realpath
  node-fstream node-fstream-ignore node-github-url-from-git node-glob node-graceful-fs node-gyp
  node-hosted-git-info node-inflight node-inherits node-ini node-is-builtin-module node-isexe
  node-json-stringify-safe node-lockfile node-lru-cache node-mime node-minimatch node-mkdirp node-mute-stream
  node-node-uuid node-nopt node-normalize-package-data node-npmlog node-once node-osenv node-path-is-absolute
  node-pseudomap node-qs node-read node-read-package-json node-request node-retry node-rimraf node-semver node-sha
  node-slide node-spdx-correct node-spdx-expression-parse node-spdx-license-ids node-tar node-tunnel-agent
  node-underscore node-validate-npm-package-license node-which node-wrappy node-yallist nodejs-doc
  python-pkg-resources
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  cmake-data libarchive13 libjsoncpp1 liblzo2-2
Paquetes sugeridos:
  codeblocks eclipse ninja-build lrzip
Se instalarán los siguientes paquetes NUEVOS:
  cmake cmake-data libarchive13 libjsoncpp1 liblzo2-2
0 actualizados, 5 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 3.986 kB de archivos.
Se utilizarán 19,8 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

Navegaremos hasta el directorio principal del proyecto

$cd tauwebsocketopencv/

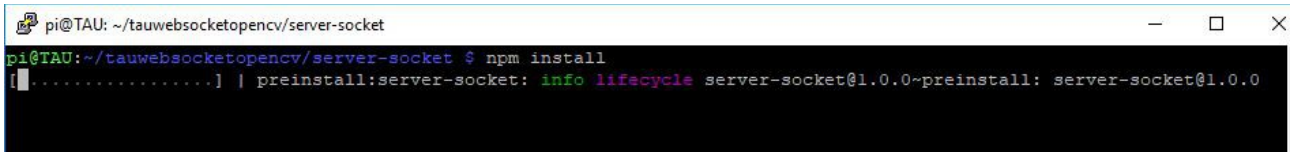
En su interior primero iremos al servidor

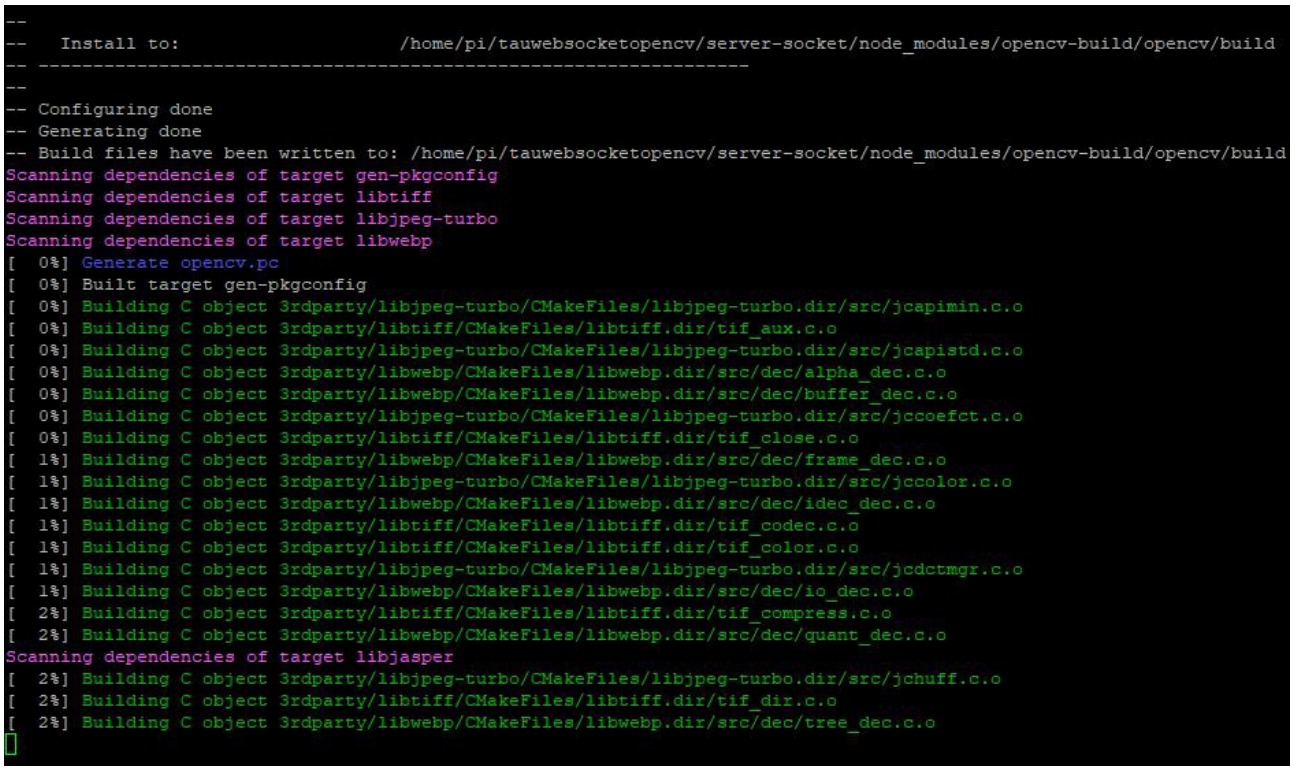
$cd server-socket/

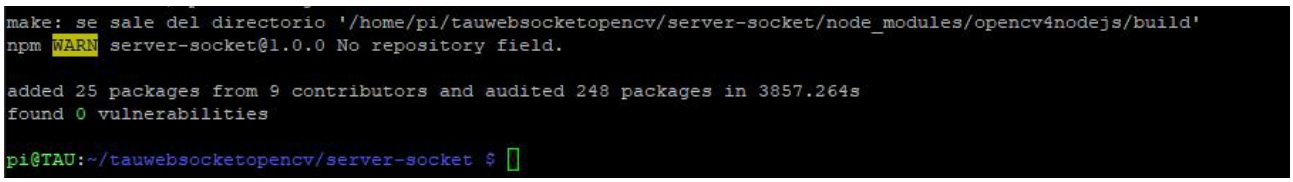
Realizaremos

$npm install

Esta tarea puede llevar horas para una RaspberryPi 2B, una 3B+ aprovechando todos sus núcleos puede llevar como máximo una hora, dependiendo también de la temperatura ambiente.





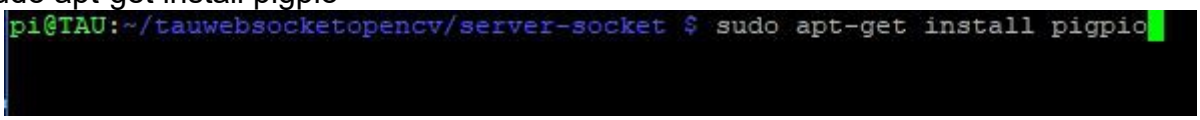


Instalaremos forever

$sudo npm install forever -g

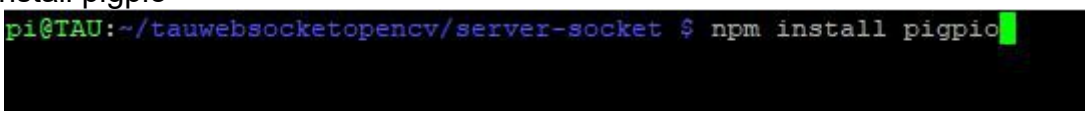
Instalaremos pigpio (normalmente ya esta instalado, pero para verificar)

$sudo apt-get install pigpio



Ahora el paquete de pigpio por si no ha bindeado con pigpio anterior.

$npm install pigpio

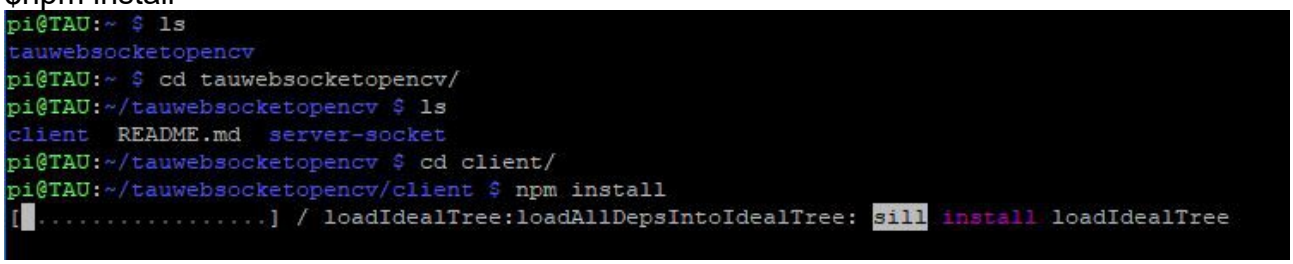


Abrimos otro terminal, e iremos al client

$cd tauwebsocketopencv/client/

Realizaremos una instalación mas

$npm install



Lanzando la app de control de TAU

Nota:
La aplicación esta montada para ser lanzada por una RaspberryPi 3B+, por su potencia extra comparada con otros modelos, ¿es posible su despliegue en otro modelo? Por supuesto, ha sido desarrollada y probada en una 2B con una antena wifi por usb, pero era demasiado lenta compilando en ocasiones, ademas de subir la temperatura notablemente, por lo que para ahorrar tiempos y prescindir de disipadores y ventiladores se decidió realizarlo en una 3B+.

La RaspberryPi se ha configurado como un punto de acceso wifi, con una IP 192.168.4.1 que es donde los websockets esperan los lanzamientos de eventos para realizar su función, en el archivo:

pilotMode.js dentro de client/src/components/views encontrara un state con dos atributos; endpointOpenCv y endpointGpio cuyo valor son Ip's, las cual puede modificar para la configuración de red que usted tenga, teniendo en cuenta que si pone la RaspberryPi como dispositivo esclavo de un router wifi (conexión normal de cualquier dispositivo a un router) y le asigna una dirección Ip estática, y modifica lo anterior citado con esa ip, no

necesitaría en ningún momento la creación de un punto wifi en la RaspberryPi que generalmente drena recursos.

Conectaremos la RaspberryPi al PowerBank y el L298N al PowerBank y esperamos unos segundos a que se inicie todo.

Nos conectamos a la red creada por TAU o en su defecto a la red local que este conectada la RaspberryPi

Iniciamos SSH mediante un ordenador o smartphone en la red deseada apuntando a la RaspberryPi

Lanzando client:

$cd tauwebsocketopencv/client/
$npm start

```
pi@TAU:~/tauwebsocketopencv $ cd client/
pi@TAU:~/tauwebsocketopencv/client $ npm start

> pruebas_tau@0.1.0 start /home/pi/tauwebsocketopencv/client
> react-scripts start
```

Si el servidor ya ha sido buildeado:
$serve -s build

```
The project was built assuming it is hosted at the server root.
You can control this with the homepage field in your package.json.
For example, add this to build it for GitHub Pages:

  "homepage" : "http://myname.github.io/myapp",

The build folder is ready to be deployed.
You may serve it with a static server:

  npm install -g serve
  serve -s build

Find out more about deployment here:

  http://bit.ly/CRA-deploy

pi@TAU:~/tauwebsocketopencv/client $
```

```
pi@TAU:~ $ cd tauwebsocketopencv/
pi@TAU:~/tauwebsocketopencv $ cd client/
pi@TAU:~/tauwebsocketopencv/client $ serve -s build
WARNING: Checking for updates failed (use `--debug` to see full error)
ERROR: Cannot copy to clipboard: Couldn't find the required `xsel` binary. On De
bian/Ubuntu you can install it with: sudo apt install xsel

   Serving!

   - Local:            http://localhost:5000
   - On Your Network:  http://127.0.1.1:5000
```

Lanzado server-socket (Nueva ventana de ssh)

$cd tauwebsocketopencv/server-socket/
$forever start servidorOpenCv.js
$sudo node servidorGpio.js

```
pi@TAU:~/tauwebsocketopencv/server-socket $ ls
index.html  node_modules  package-lock.json  servidorOpenCv.js
index.js    package.json  servidorGpio.js
pi@TAU:~/tauwebsocketopencv/server-socket $ forever start servidorOpenCv.js
warn:    --minUptime not set. Defaulting to: 1000ms
warn:    --spinSleepTime not set. Your script will exit if it does not stay up f
or at least 1000ms
info:    Forever processing file: servidorOpenCv.js
```

¿Por que forever?

Forever permite mantener un servidor de Node vivo continuamente, si el servidor entra en un fallo y se cierra, algo que ocurre en ocasiones con OpenCv y WebSockets, este lo volverá a reiniciar.

¿Por que sudo node?

Es necesario lanzar el servidor de control Gpio como usuario administrador por construcción de las directivas de permisos y seguridad en cuanto al uso de los Gpio, ya que en ellos podemos modificar muchas cosas sobre la RaspberryPi, y una mala conexión o programación de los Gpio puede corromper la memoria, quemar el procesador... (como ya ha ocurrido en una ocasión con este proyecto)

Usando la aplicación

Entrando en un explorador de Internet vamos a la dirección ip de la RaspberryPi.

Esperamos que se complete la carga de la barra, esta asegura que todos los servidores websockets estén activados y esperando su lanzamiento.

Nos encontramos dos botones, uno con instrucciones básicas sobre el uso, y otro que nos permite el uso de TAU.

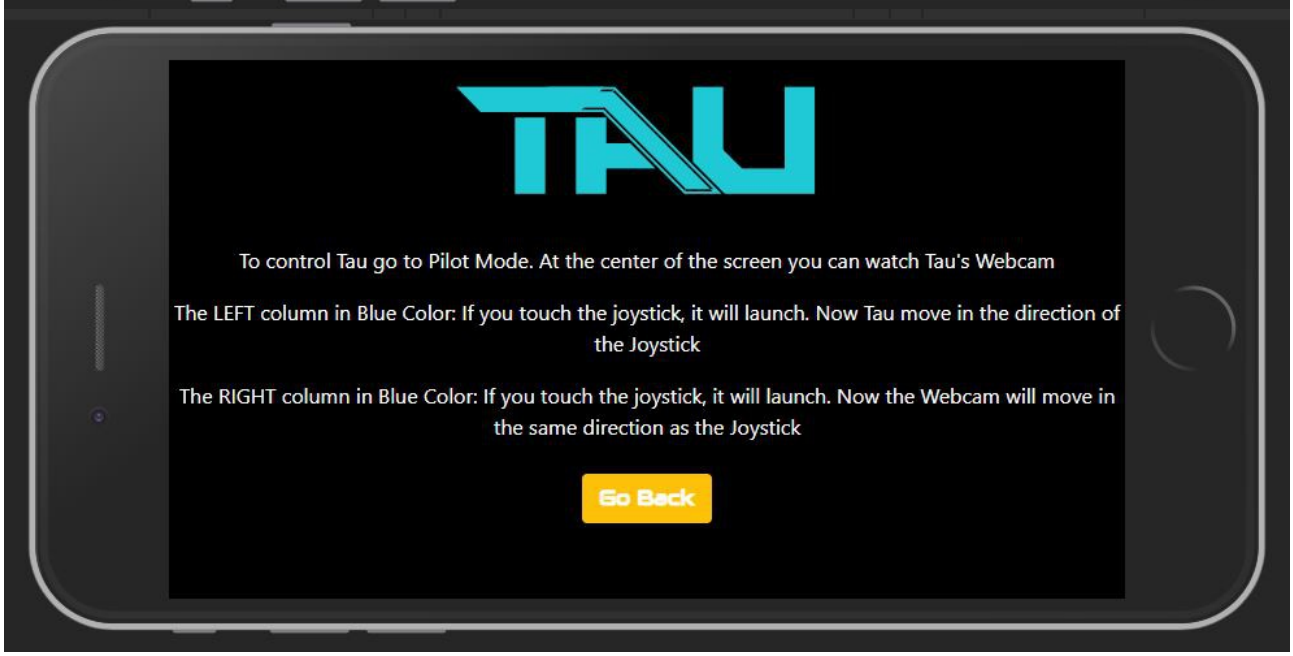
Controlando a TAU

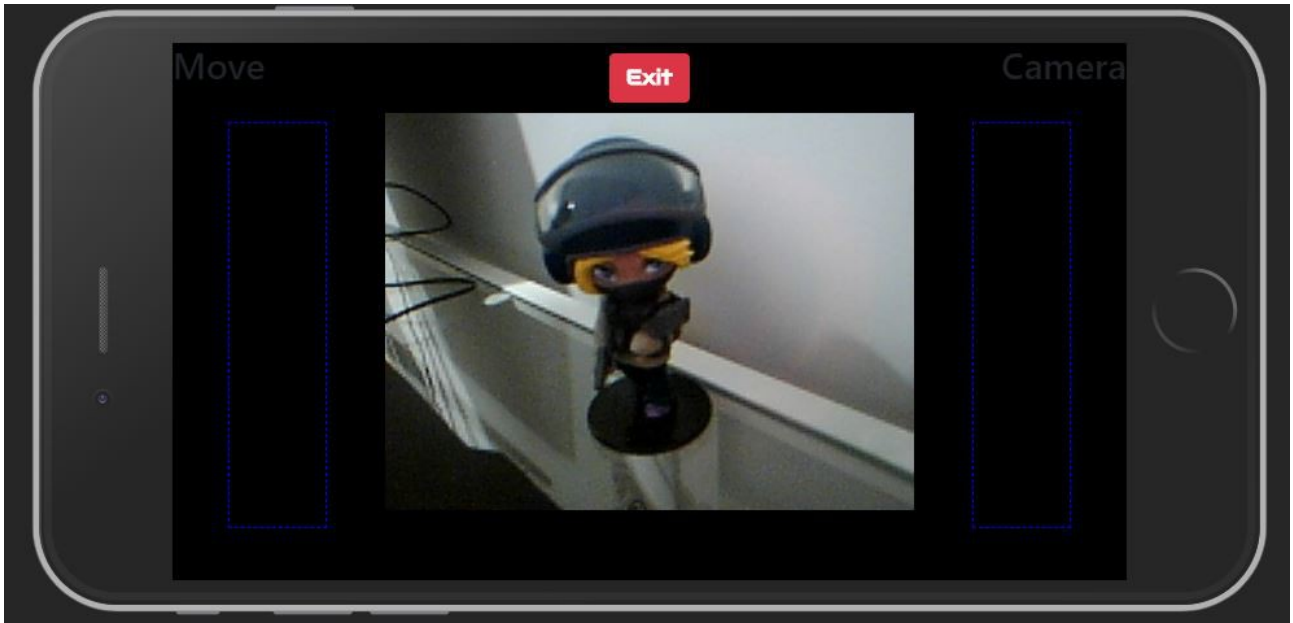
Al entrar en el modo piloto, la cámara de TAU se desplegara, para indicar que esta activado y esperando.

Nos encontraremos dos columnas azules, si presionamos con el dedo se activara el gesto, que lanzara un Joystick virtual que nos permite controlar la sección de los motores o los servos de la cámara, depende de la columna. Al ser multitouch, permite el lanzamiento de ambos y la realización de dos eventos distintos para cada uno, por lo que no harán interferencias entre ellos.

Si dejamos de pulsar el Joystick de los motores, TAU se detendrá en su posición actual, si es el de los servos, se quedaran en su posición actual, para centrar la cámara bastara con una doble pulsación en la zona para reiniciar los servos a su estado inicial.

Si salimos, la cámara volverá a replegarse indicando que esta en espera.





Paquetes, librerías y aplicaciones usadas para desarrollar el proyecto

Putty (SSH)
React.js
Node.js
Pigpio
Nipple.js (Joysticks multitouch)
Socket.io
Opencv4nodejs
Bootstrap
Cmake (Buildeo de Opencv4nodejs)
Samba (Gestion de archivos desde Pc hacia la RaspberryPi)
Visual Studio Code
Dnsmasq hostapd (sudo apt-get install dnsmasq hostapd) (Creación de punto acceso wifi en RaspberryPi)
Chrome para Windows y Android
Win32DiskImager (Grabación de la imagen en la microsd y realización de copias de seguridad)

Fuentes de información:

Wikipedia: https://es.wikipedia.org/
StackOverFLow: https://stackoverflow.com/
Raspberry.org: https://www.raspberrypi.org/
NPM: https://www.npmjs.com/

Agradecimientos:

A mis amigos y mi novia por su apoyo y horas probando a TAU durante el tiempo de desarrollo.

A mi familia por facilitarme los componentes necesarios, al igual que mis amigos, profesores y compañeros de Avirato.SL por sus consejos, apoyo y material para finalizar a TAU.