Sistema de distribución del cambio de estado de semáforos hacia vehículos a través de una red oportunista

Cristian García Alonso

Resum– Actualmente, una gran cantidad de marcas automovilísticas y centros de investigación trabajan en el diseño y la fabricación de vehículos autónomos. Estos vehículos toman decisiones a partir de cámaras y sensores. En este trabajo se plantea crear un entorno que, mediante la utilización de redes tipo *Delay and Disruption Tolerant Networking*, proporcione información a los vehículos sobre el estado de los semáforos que se van a encontrar.
Para ello, los semáforos envían la información solicitando que sea propagada por los vehículos que puedan estar dirigiéndose hacia los semáforos.
Palabras clave– Redes tolerantes a interrupciones y retrasos (DTN)(ADTN), Conducción Autónoma, Semáforos

Abstract– Nowadays, many automobile brands and research centers work in the design and manufacture of autonomous vehicles. These vehicles make decisions from cameras and sensors. This article proposes to create an environment that provides information to vehicles using delay tolerant networks (DTN). The main idea is to transmit the information of the change of state of the traffic lights to vehicles. In addition, it is suggested that vehicles can communicate with each other using DTN.

Keywords– Delay-tolerant networking (DTN), Autonomous vehicles, Traffic lights

1 Introducción

La conducción autónoma está en auge. Multitud de empresas y centros de investigación alrededor del mundo trabajan en el diseño y fabricación de vehículos autónomos. En paralelo, están emergiendo nuevas tecnologías de comunicación que no requieren el uso de una infraestructura de telecomunicaciones. En este artículo se plantea la utilización de dichas tecnologías de comunicación para transmitir información desde los semáforos que regulan el tráfico hacia los vehículos autónomos y no autónomos.

Este trabajo de final de grado nace a partir de la unión de los resultados obtenidos en proyectos de investigación del Centro de Visión por Computador (CVC) [1] y el grupo Security of Networks and Distributed Applications (SENDA) [2], ambos pertenecientes a la Universidad Autónoma de Barcelona (UAB).

El CVC es un centro de investigación el que se lleva a cabo investigación de vanguardia de gran impacto internacional en el campo de la visión por computador. Uno de los grupos de investigación es Advanced Driver Assistense Systems (ADAS) [3]. Su proyecto llamado Elektra [4], consiste en el diseño y la fabricación de un vehículo autónomo.

SENDA es un grupo de investigación consolidado perteneciente al Departamento de Ingeniería de la Información y las Comunicaciones (DEIC) de la UAB. Su objetivo es investigar en las áreas de seguridad de red aplicada y aplicaciones distribuidas seguras. Uno de los proyectos de SENDA consiste en investigar la tecnología DTN. En concreto, han desarrollado una arquitectura de comunicación DTN en la que los mensajes llevan consigo parte del código de encaminamiento.

Lo que plantea es la creación de semáforos que a través de técnicas de visión por computador analicen el entorno y decidan de forma inteligente la programación del cambio de

• E-mail de contacte: Cristian.GarcíaA@e-campus.uab.cat
• Menció realitzada: Enginyeria de Tecnologies de la Informació
• Treball tutoritzat per: Àngela Fàbregues Vinent (DEIC)
• Curs 2016/17

estado de estos. Es decir, cuando se van a poner en verde, en rojo, etc. Además, estos semáforos transmitirán la programación actual del semáforo hacia los vehículos. De esta manera el conductor o el sistema inteligente del vehículo podrían saber con cierta antelación en qué estado se van a encontrar el semáforo (rojo, ámbar, verde...) cuando lleguen a él, y por lo tanto, adaptar la conducción.

El resto de este artículo está organizado de la siguiente manera: en la Sección número 2 se describe cual es la motivación que lleva a tener los objetivos de este trabajo. En la sección 3 se explica cuál es el estado actual de las tecnologías DTN y en qué proyectos ha sido utilizada. En la sección 4 se indica el procedimiento utilizado para realizar este trabajo. En la sección 5 se detalla el proceso de desarrollo del software que se implementa en este trabajo. En la sección 6 se explican los experimentos más relevantes realizados en este TFG. Finalmente, en la sección 7 se exponen las conclusiones de ese trabajo.

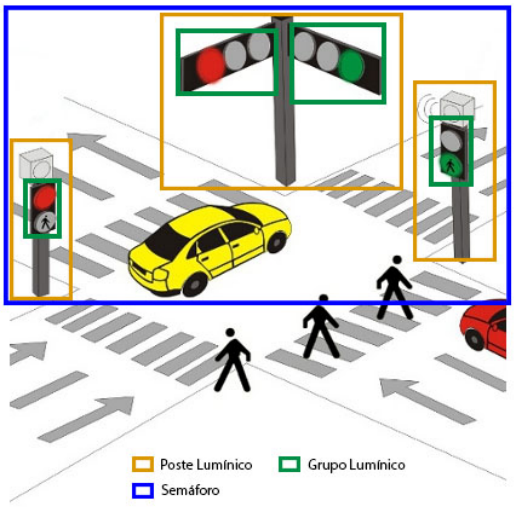

Fig. 1: Concepto de semáforo, grupo lumínico y poste lumínico.

2 Objetivos

Los objetivos principales de este proyecto son el control de semáforo y la propagación de la información.

2.1 Control del semáforo

Se entiende por *semáforo* el conjunto de uno o más aparatos eléctricos de señales luminosas utilizado para regular la circulación. A cada uno de estos aparatos eléctricos, durante el ámbito de este trabajo, lo llamaremos *grupo lumínico*. Es decir, un grupo lumínico será un conjunto de bombillas de diferentes colores. Por ejemplo: un grupo lumínico para peatones tendrá una bombilla roja y otra verde. Para los vehículos habitualmente tendrá una bombilla roja, una ámbar y otra verde. En muchas ocasiones podemos encontrar varios grupos lumínicos anclados al mismo poste, a esto lo llamaremos *poste lumínico*.

Otro concepto importante es el de *estado del semáforo*. Este concepto indica qué bombilla está encendida en cada uno de los grupos lumínicos en cada instante de tiempo. Por ejemplo, cuando en el grupo lumínico de los peatones está encendida la bombilla de color rojo, en el grupo lumínico de los vehículos está encendida la bombilla de color verde. Esto es en si un estado.

En la figura 1 se ilustran los conceptos introducidos en esta sección para una mejor comprensión de estos.

El objetivo de controlar el semáforo se divide en dos subobjetivos:

1. **Control del estado de un poste lumínico**: Instalar un dispositivo electrónico en el poste que controle la secuencia de cambio y pueda modificarla.

2. **Control del estado del semáforo**: Extender las capacidades de dicho dispositivo para realizar el control de varios postes que formen un único semáforo de manera coordinada.

2.2 Propagación de la información

Los dispositivos que controlan los postes disponen de información muy interesante para la navegación de los vehículos que se aproximan a ellos. Estos dispositivos deben enviar dicha información hacia los vehículos utilizando la tecnología ADTN. Esta tecnología permite que el algoritmo de encaminamiento del mensaje viaje junto al mensaje. Por lo tanto, un objetivo es diseñar un algoritmo de encaminamiento que haga que la información del semáforo llegue a todos los vehículos posiblemente interesados en esta.

3 Estado del arte

En esta sección se hace un repaso del estado actual de las tecnologías de las cuales se habla en este documento.

3.1 Control de semáforos

En la actualidad existen diferentes formas de controlar el estado de los semáforos de una población. Una de ellas consiste en mantener un control centralizado del estado de todos los semáforos, tal y como ofrece la empresa SCT [5]. Estos sistemas esencialmente constan de una central de datos que a través de diversos protocolos de comunicación mandan órdenes a los semáforos. Hay ocasiones en que, por diversas causas, estos sistemas centralizados no pueden ser utilizados. En estos casos, una opción, es programar los semáforos de forma estática. Es decir, un operario introduce de forma manual la secuencia que el semáforo deberá reproducir y este la reproduce de forma iterativa durante un periodo indefinido de tiempo.

3.2 Delay-tolerant networking

Originalmente las redes entre ordenadores fueron pensadas para que un conjunto de nodos se enviaran paquetes de unos a otros. Estos paquetes viajarían a través de una estructura de telecomunicaciones estable. En esta estructura el encargado de definir qué camino debe tomar un paquete para llegar de un destino a otro es el router.

Para evitar interrupciones del servicio a causa de la caída de un elemento, como por ejemplo un router, se pensó en replicar la estructura de esta manera que si un router estuviera fuera de servicio este paquete pudiera tomar un camino alternativo para llegar a destino con éxito.

Por lo tanto, en esta arquitectura clásica, tenemos dos actores principales: los encargados de enviar y recibir paquetes, y los encargados de encaminar estos paquetes.

En la figura 2 se muestra un esquema básico de la arquitectura tradicional de red.

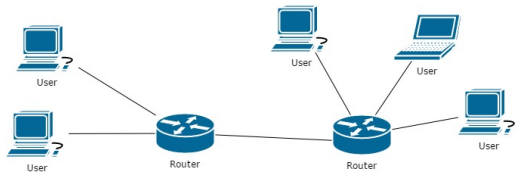

Fig. 2: Arquitectura tradicional de red

Delay-Tolerant Networking Architecture (DTN) [6], cuyo RFC fue publicado en 2007, propone una arquitectura de red alternativa la cual es tolerante a fallos por interrupciones y retrasos. Es decir, permite el correcto envió y recepción de mansajes independientemente de si el receptor está conectado en el momento del envió o de si se producen caídas en la red. Además, permite la movilidad geográfica de todos los elementos que formen la red.

Ahora ya no existen unos actores llamados router. Tampoco existe un camino predefinido para ir de un origen a un destino. Ahora lo que existen son unos actores llamados nodos DTN. Estos nodos tendrán la función de enviar, recibir, transportar y reenviar mensaje utilizando el protocolo bundle [7].

El proceso que se siguen los nodos DTN para transmitir un bundle desde un origen hacia uno o varios destinos es el siguiente: los nodos guardan y se transfieren bundles entre si hasta que el bundle es recibido por el nodo o los nodos destinatarios. El tiempo que cada nodo mantiene un bundle guardado y la decisión de reenviar o no un bundle irá en función de un algoritmo indefinido. Es decir, hay múltiples estrategias posibles para realizar el encaminamiento entre nodos.

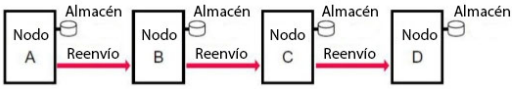

Fig. 3: Arquitectura de nodos DTN

A pesar de que esta tecnología de comunicación no es muy conocida, la National Aeronautics and Space Administration (NASA) lleva trabajando en ella desde el año 2002 [8]. Persiguen el objetivo de crear una Internet interplanetaria. El 10 de Julio de 2009 la NASA realizo una prueba real descargando imágenes desde un satélite [9].

3.3 Active delay-tolerant networking

Active delay-tolerant networking (ADTN) [10] [11], es una extensión del bundle utilizado por DTN diseñada y desarrollada [12] por el grupo SENDA. Esta extensión permite enviar el algoritmo de encaminamiento junto al mensaje. De esta manera cuando los nodos ADTN reciben el mensaje ejecutan el algoritmo que este lleva consigo. Este algoritmo define diversos aspectos como: de que manera debe ser reenviado el mensaje, cuando este debe ser eliminado, etc. Este hecho permite que cada mensaje se pueda propagar de forma diferente.

Esta tecnología has sido utilizada en dos trabajos de final de grado anteriores [13] [14].

4 Metodología

La metodología de trabajo utilizada es la metodología Agile, basada en ciclos de trabajo cortos e iterativos. Para aplicar esta metodología se han dividido los objetivos en tareas que se puedan realizar en un plazo de una o varias semanas de tiempo. En general, estas tareas incluyen diseño, despligue, implementación, y pruebas.

Dado que esta metodología se basa en ciclos cortos de trabajo es posible en cada ciclo corregir los errores cometidos o introducir mejoras respecto al ciclo anterior. Para poder detectar estos posibles errores o mejoras, antes de iniciar cada ciclo, se realiza una revisión de todo el trabajo realizado anteriormente teniendo en cuenta el objetivo del siguiente ciclo. Este procedimiento es muy útil para detectar de forma prematura errores que si no fueran detectados podrían ser críticos en fases futuras del proyecto.

5 Desarrollo

El desarrollo de este TFG se ha basado en diseñar e implementar cada uno de los objetivos de manera que al final de la realización de estos se obtenga un sistema funcional. Para cada una de las partes del desarrollo se han hecho experimentos para analizar el funcionamiento y detectar errores.

5.1 Componentes físicos

Para poder llevar a cabo el objetivo de controlar un semáforo fue necesario analizar el funcionamiento lógico y físico de un semáforo real. El CVC llego a un acuerdo con la empresa de semáforos *Traffic Futura* [15] situada en Valencia la cual cedió 4 grupos lumínicos, dos destinados a peatones y dos destinados a vehículos. Estos lumínicos constan de entre 2 y 3 bombillas led con input de 230VAC - 0.024A a 50Hz cada una. Cada uno de ellos consta de un cable de alimentación y una toma tierra.

Para el control de las luces ha optado por un dispositivo electrónico que es un ordenador de tamaño reducido que tuviera salidas y entradas de propósito general (GPIO). Entre todas las opciones del mercado se decidió escoger la Raspberry Pi 3 modelo B.

Dado que la GPIO de la Raspberry Pi 3 [16] no tiene la suficiente potencia como para encender el grupo lumínico, se decidió acoplar a esta un relé de una potencia adecuada.

En la figura 4, se muestra una ilustración de una Raspberry Pi 3. En la ilustración se destacan con un cuadro negro los pins GPIO utilizados para enviar las señales de encendido y apagado de los grupos lumínicos.

Para solucionar el problema de la alimentación eléctrica se decidió que las pruebas se harían en el interior del edificio del CVC conectando todos los elementos del semáforo a diversas tomas eléctricas. En el caso de realizar una prueba en el exterior se utilizarían baterías de alta capacidad.

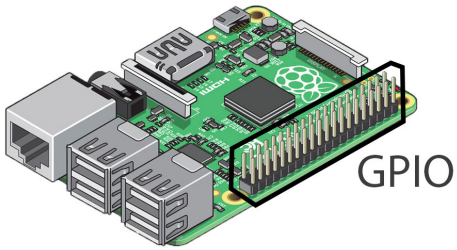

Fig. 4: Ilustración de una Raspberry Pi 3. Destacado el GPIO.

5.2 Controlador del semáforo

Dado que el hardware en cual se ejecuta el controlador tiene poca potencia de computo, se decidió los idiomas de programación a utilizar serian C y C++. Además, se decidió utilizar el mínimo número de dependencias posibles por tal de facilitar la portabilidad del software.

5.2.1 Control del GPIO

El primer problema a resolver fue cómo controlar la GPIO. Existen dos maneras de realizar dicho control:

- Realizando llamadas al sistema el cual tiene la capacidad de modificar tanto el modo de funcionamiento de cada uno de los pins que componen la GPIO como el estado. El sistema al realizar estas operaciones tarda más tiempo del deseado, es decir, existe un retraso desde la llamada hasta la ejecución
- Modificando los registros tal y como se explica detalladamente en la guía *CM2835 ARM Peripherals* [17]. Realizar operaciones con la GPIO modificando los registros es instantáneo. Existe una librería llamada WiringPi [18] que implementa una interfaz para poder modificar estos registros de forma sencilla.

De estas dos posibles opciones fue escogida la segunda, ya que WiringPi es una librería muy ligera, rápida y fácilmente portable.

5.2.2 Diseño de la arquitectura del software

A partir de los conocimientos obtenidos en la carrera se diseñó una arquitectura basada en patrones de diseño. Esta arquitectura fue pensada para que el funcionamiento de esta fuera altamente configurable. De manera que gran parte del comportamiento del software pudiera ser modificado a partir de un fichero de configuración externo. Como la cantidad parámetros de configuración incrementaría a medida que se fueran ampliando las funcionalidades del software, se utilizó el patrón de configuración por defecto. Este patrón indica que para cada una de las opciones que tiene un software debe haber definida una opción por defecto. De esta manera se evita que un usuario deba necesariamente conocer todos los detalles del software.

El software desarrollado tiene las siguientes características:

- **Carga dinámica de la secuencia de cambio de estado**: unas de las configuraciones se permiten realizar desde el fichero de configuración es el hecho de definir cada cuanto tiempo que grupo lumínico realizara un cambio de estado. Por ejemplo, de rojo a verde.
- **Cambio de la secuencia de cambio de estado en función de la hora del día**: habitualmente la situación del tráfico cambia a medida que pasa el día. Por esa razón es interesante permitir la configuración de diversas secuencias de cambio de estados. Cada una es configurada para que se ejecute a partir de una determinada hora.
- **Escalable**: habitualmente un semáforo está compuesto de varios postes lumínicos. Entre ellos están separados varios metros de distancia. A cada poste lumínico va unidas almenas una Raspberry, que a través del GPIO controla el encendido y apagado de los leds. El semáforo debe mantener un único estado en cada instante de tiempo. Para que esto sea posible este estado debe estar compartido entre todas las Raspberry que controlan el semáforo. Para conseguir que esto sea posible, es necesario crear una red formada por todas las Raspberry que forman el semáforo. El funcionamiento de esta red se ha pensado para facilitar adición o sustracción de Raspberry al semáforo.
- **Secuencias modificables en tiempo de ejecución**: el software está pensado para poder recibir secuencias de cambio de estado proporcionadas por un agente externo. Este agente externo pude ser un sistema de visión por computador que analiza el tráfico y decide la mejor secuencia para cada momento. El software cuando recibe una nueva secuencia la guarda y cuando sea la hora de inicio de la nueva secuencia, la reproduce.
- **Envió del estado del semáforo a través de DTN**: el software utiliza la interfaz proporcionada por la librería aDTN Plus desarrollada por el grupo senda. A través de esta, envía la información del cambio de estado hacia un software que simula ser un vehículo. Los detalles de esta parte esta del software están explicados en la sección 5.3.

5.2.3 Arquitectura maestro-esclavo

Tal y como se explica en la sección 5.2.2 el software puede funcionar de forma distribuida, es decir, puede ser ejecutado en varias máquinas y entre todas las instancias de este mantener un único estado del semáforo para cada instante de tiempo.

Para hacer esto posible se utiliza una arquitectura de maestro-esclavo. En este tipo de arquitectura el maestro que envía tareas a ejecutar a uno o más esclavos.

En el caso del software controlador del semáforo, el maestro se encarga de mantener el estado lógico del semáforo y de mandar señales a los esclavos cuando hay un cambio de estado. El esclavo recibe la señal y cambia estado físico del grupo lumínico que controla. Es decir, apaga una luz y enciende otra.

El maestro también puede controlar estar conectado a un grupo lumínico a través de la GPIO y controlar físicamente el cambio de estado. Además, este está preparado para poder recibir en cualquier instante una nueva programación

de cambios de estado que ejecutar. Esta programación puede ser enviada, por ejemplo, por un sistema de visión por computador que analice el tráfico y genere nuevas programaciones.

5.2.4 Limitaciones del hardware

Para el correcto funcionamiento del software del semáforo y que la información enviada hacia los vehículos sea la correcta es importante mantener la hora. Una de las limitaciones de trabajar con una Raspberry Pi es el hecho de que no lleva integrado ningún mecanismo para guardar la hora cuando esta no está conectada a una fuente eléctrica. Para solucionar este problema se encontraron dos soluciones:

- **Conectar un GPS**: de esta manera la hora siempre se obtendría desde el GPS.
- **Agreagar una extensión hardware**: de manera que gracias a esta se pudiera conservar la hora. Esta solución cosiste en un reloj hardware que se conecta a la GPIO. Este está alimentado por una pila.

De estas opciones se escogió utilizar el reloj hardware por la sencillez y la economicidad de la solución.

Otro de los problemas de trabajar con este ordenador de tamaño reducido es la falta de potencia de computo. Esto hace que el tiempo de compilación de un programa en C++ sea de minutos. La consecuencia es que se pierde mucho tiempo esperando a que termine dicha compilación. La solución a este problema fue utilizar un compilador cruzado. Es decir, un compilador que permite compilar software para una arquitectura diferente a la que está siendo usada para compilar. Gracias a este se pudo trabajar sobre un ordenador de mayor potencia reduciendo los tiempos de compilación a segundos.

En la fase final del proyecto, cuando se integró la librería ADTN-Plus, no se pudo utilizar el compilador cruzado ya que este daba unos errores que no se supieron solucionar. Esto ralentizo esta última fase.

5.2.5 Control del GPIO

5.3 Transmisión de la información con ADTN Plus

ADTN Plus [12] es una implementación en C++ realizada por el grupo senda del protocolo de comunicación llamado Active-DTN [10]. A pesar de que la implementación es muy clara y configurable el hecho de que la librería no disponga de con documentación hace complicado entender su funcionamiento. Para comprender el funcionamiento de esta librería, el cual se explica a continuación, se han realizado diversos experimentos.

5.3.1 Funcionamiento de aDTN Plus

La librería consta de diversos ejecutables el principal es el llamado *BundleAgent*. Este ejecutable lo que hace es iniciar un nodo aDTN a partir de la configuración que se indique en un fichero de configuración *.ini* el cual se le tiene que pasar como primer parámetro a *BundleAgent*. Este fichero está dividido en las siguientes secciones:

- **Node**: en esta sección se indica la ip y el puerto en el cual *BundleAgent* está esperando recibir conexiones TCP [19] por parte de otros nodos.
- **NeighbourDiscovery**: en esta sección se indica la dirección del grupo multicast a través de la cual *BundleAgent* va a enviar mensajes de descubrimiento llamados beacons.
- **AppListener**: aquí se indica la ip y el puerto por donde *BundleAgent* va a esperar conexiones de las diversas aplicaciones que utilicen adtnPlus.

Se puede entender a *BundleAgent* como un router que realiza principalmente las siguientes funcionalidades:

- **Descubrir nodos vecinos**: a través de un grupo multicast busca la existencia de otros nodos.
- **Guardar y enviar mensajes**: cuando BundleAgent recibe un mensaje de una aplicación lo guarda hasta que detecta otro nodo y en función de un algoritmo que puede llevar el mensaje reenvía el mensaje al nodo vecino.
- **Escuchar a las aplicaciones**: en adtn plus se entiende como una aplicación cualquier aplicación que utilice la interfaz adtnSocket tanto para recibir como para enviar datos a través de adtn. Esta interfaz lo que hace es conectarse a *BundleAgent* por TCP [19] y recibir o dejar un mensaje a este.

5.3.2 Encaminamiento de mensajes de ADTN Plus

Una de las características que diferencia Active-DTN de otras arquitecturas DTN es el hecho de que el encaminamiento de los mensajes se realiza en función de un algoritmo puede llevar integrado el propio mensaje. En la implementación de aDTN Plus no es necesario definir un algoritmo en cada mensaje. Esto es posible porque la librería lleva integrados unos algoritmos por defecto. Estos algoritmos no solo definen el encaminamiento, sino que también definen el tiempo de vida, que se hace en la creación de cada mensaje, etc. Para cambiar la configuración por defecto basta con modificar un fichero de configuración llamado NodeState.json.in.

5.3.3 Algoritmo de encaminamiento

De cara a la propagación de la información del semáforo se ha diseñado un algoritmo de ADTN. El objetivo de dicho algoritmo es que los vehículos mantengan solo la información que es útil para cada vehículo en particular y no toda. Se considera que la información útil para un vehículo es la de los grupos lumínicos que se va a encontrar a continuación. Por lo tanto, el algoritmo descarta la información de los grupos lumínicos por los cuales el vehículo ya ha pasado. Además, cada vehículo mantiene la información de los grupos lumínicos del sentido contrario al que circula el vehículo. Dicha información es reenviada al resto de vehículos.

Para poder realizar el algoritmo 1 es necesario saber la posición geográfica de cada uno de los postes lumínicos y la orientación en grados respecto al norte de cada uno de los

grupos lumínicos. A partir de esta información y utilizando la ecuación de la recta [20] se traza una recta imaginaria. Si el vehículo ha superado esta recta [21] imaginaria quiere decir que ha sobrepasado el grupo lumínico y por lo tanto puede descartar la información de este.

En la figura 5 se muestra como el vehículo amarillo elimina el bundle que contiene información sobre el semáforo que aparece en la imagen. El vehículo azul conserva el bundle porque aún no ha superado el grupo lumínico.

Algorithm 1 Determina si un vehículo ha superado un grupo lumínico o no.

Precondition: $P1$ y $P1$ coordinadas geográficas del poste lumínico
Precondition: $Angle$ angulo en grados respecto al norte de un grupo lumínico
Precondition: $Q1$ y $Q2$ coordenadas geográficas de un vehículo

1: **function** $(P1, P2, Angle, Q1, Q2)$
2: $m \leftarrow tan(Angle)$ ▷ Pendiente
3: $B \leftarrow m * (-1 * P1) + P2$
4: $MX \leftarrow m * Q1$
5: $Y \leftarrow MX + B$ ▷ Ecuación de la recta
6: **if** $Y - Q2 > 0$ **then**
7: **return** $True$ ▷ Superado
8: **else**
9: **return** $False$ ▷ No superado
10: **end if**
11: **end function**

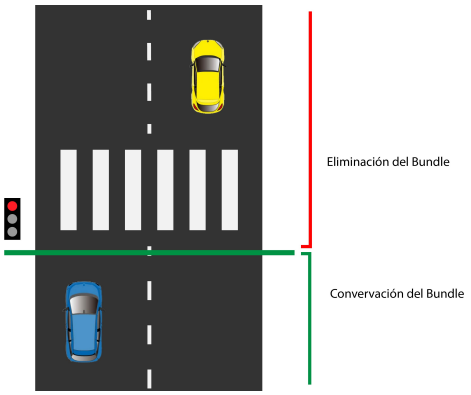

Fig. 5: Ilustración del comportamiento

Además de tener en cuenta si el vehículo ya supero el grupo lumínico, también se tienen en cuenta el tiempo de expiración del bundle. Aunque el vehículo no haya superado el grupo lumínico si el tiempo de expiración ha pasado el bundle es eliminado.

En la implementación de aDTN también hay que definir el criterio para decidir si enviar un bundle a un nodo vecino o no. En este caso como los tiempos de expiración son reducidos se ha decidido que todos los bundles serán enviados a todos los nodos vecinos. Otro algoritmo que hay que definir es a que aplicaciones conectadas al nodo se les enviara el mensaje contenido en el bundle y a cuáles no. En este caso se ha decidido enviar el mensaje a las aplicaciones con una ID especifica. Esta ID será la que tienen que tener todas las aplicaciones interesadas en la información de los semáforos.

Información del bundle

Para que sea posible utilizar el algoritmo descrito anteriormente cada bundle debe tener la información de la posición geográfica del poste lumínico y el ángulo de un grupo lumínico. El mensaje también tendrá la información de un grupo lumínico.

Por lo tanto, un semáforo envía tantos bundles como grupos lumínicos diferentes. Los vehículos cuando reciben los bundle ejecutan el algoritmo que estos llevan integrados agregando la geolocalización del vehículo. De esta manera el vehículo solo mantiene los bundles que aportan información útil para el vehículo.

6 Experimentación

Durante el desarrollo de este trabajo se han realizado diversos experimentos con el objetivo de experimentar el funcionamiento del controlador de semáforo desarrollado y deducir funcionamiento de ADTN Plus. En esta sección del documento se exponen de forma detallada los experimentos más relevantes.

6.1 Experimentos con el controlador del semáforo

En esta sección se explican los experimentos hechos con el objetivo de comprobar el correcto funcionamiento del conrolador del semaforo.

6.1.1 Experimento ctrl-1: con un semáforo real

Este experimento fue realizado en el Centro de Visión por computador (CVC). El objetivo es ver la estabilidad del controlador y de la Raspberry Pi cuando estos se están ejecutando durante días en un entorno parecido al real.

Preparación

Para realizar este experimento se han anclado dos grupos lumínicos oficiales, proveídos por la empresa *Traffic Futura* [15], en un poste. En el mismo poste se instaló una Raspberry. Mediante diversos cables se conectaron los pins GPIO de la Raspberry hacia un relé. Este relé a su vez estaba conectado a las bombillas led de los 2 grupos lumínicos. Para proveer de energía eléctrica al semáforo y a la Raspberry pi ambos fueron conectados a la red eléctrica. Previamente al montaje, en la Raspberry se instaló el software controlador del semáforo y se configuro para que reprodujera una secuencia durante un tiempo indefinido.

Resultado

El poste lumínico estuvo funcionando por más de una semana sin presentar problema alguno.

Conclusión

El software controlador del semáforo cuando se ejecuta con una sola Raspberry es muy estable y no causa ningún tipo de sobrecalentamiento. El relé tampoco presentaba ningún tipo de problema.

6.1.2 Experimento ctrl-2: prueba del sistema distribuido

Preparación

Para realizar este segundo experimento se instaló el software controlador del semáforo en 2 Raspberry PI y en un ordenador.

Los 3 fueron conectados a la misma red LAN. Para poder controlar las 2 Raspberry Pi desde el ordenador se habilito el acceso SSH [22] a ambas.

Cada una de las Raspberry Pi y el ordenador representan un nodo. De estos 3 nodos uno de ellos fue configurado como MASTER y los otros dos como SLAVE. Estos 3 nodos se configuraron para que entre los 3 reprodujeran una secuencia de cambios de estados durante un tiempo indefinido.

Resultado

El sistema se estuvo ejecutando durante varios minutos sin presentar problemas. Pasados estos minutos se decidió desconectar un nodo SLAVE. A consecuencia de esto el nodo MASTER se cerró y el otro nodo SLAVE continúo ejecutándose. El comportamiento esperado era que el nodo MASTER debía detectar la caída del SLAVE y espera a la reconexión de este.

Conclusión

Se detectó un problema que hace que si un nodo SLAVE cae el nodo MASTER termine la ejecución en vez simplemente de detectar la caída.

6.1.3 Experimento ctrl-3: prueba del sistema distribuido y simulando una cámara

Preparación

Para realizar este segundo experimento se instaló el software controlador del semáforo en 3 Raspberry PI y en un ordenador. Los 4 fueron conectados a la misma red LAN. Para poder controlar las 3 Raspberry Pi desde el ordenador se habilito el acceso SSH [22] a ambas. Cada una de las Rasperry Pi representan un poste lumínico y controlan diversos grupos lumínicos. El ordenador simula ser una cámara que una vez ha analizado el tráfico ha generado una nueva secuencia de cambios de estado y se la envía al nodo MASTER del semáforo.

Resultado

El sistema se estuvo ejecutando durante varios minutos utilizando las secuencias de tiempo de la configuración. Pasados estos minutos desde el ordenador se envió una nueva secuencia de cambios de estado que debía ejecutarse dentro de 1 minuto. El nodo MASTER recibió correctamente esta secuencia y pasado un minuto ordeno la ejecución de esta. Para confirmar el buen funcionamiento pasados varios minutos de realizo el mismo procedimiento con resultado idéntico.

Conclusión

El sistema puede recibir nuevas secuencias de cambios de estado y ejecutarlas a la hora programada sin presentar problemas.

6.2 Experimentos con aDTN plus

6.2.1 Algoritmo de routing

Preparación

Previamente a la realización de este experimento se han implementado los algoritmos de enrutamiento indicados en la sección 5.3.3.

Para realizar el experimento se han utilizado 3 Raspberry y un ordenador conectados a la misma red vía ethernet. En todas ellas se ha habilitado el acceso SSH para poder acceder a ellas desde el ordenador situado en la misma red. Previamente en todas ellas y en el ordenador se ha instalado la librería Adtn Plus con unas pequeñas variaciones y el software controlador del semáforo.

En cada una de las Raspberry (A, B y C) y el ordenador (D) se inicio un nodo aDTN. En A se inició el software controlador del semaforo el cual controlaba un único grupo lumínico. Un B fue configurado para simular que tenia una posición geografía situada por encima de A. Los nodos situados en C y D fueron configurados para estar en una posición geográfica inferior a la de A.

En B, C y D se inició el software adtnPlus-recv, el cual sirve pare recepcionar bundles destinados a una ID, todos ellos con la misma ID de aplicación. Esta misma ID es la que utiliza el controlador del semáforo situado en A para indicar el destino de los bundle.

Resultado

La aplicación controladora del semáforo empezó a generar bundles cada cierto instante de tiempo. Estos bundles fueron enviados por el nodo aDTN A al cual está registrado la aplicación hacia un nodo vecino. Esto sucedió porque este nodo aDTN solo detectaba un nodo como vecino el B. El nodo B si que detecto todos los demás nodos y envió el bundles hacia los otros nodos A,C,D.

El nodo B descarto el bundle porque detecto que estaba en una posición geográfica superior a la altura del grupo lumínico. Los nodos C y D no descartaron el bundle y pasaron el mensaje a la aplicación. Segundos después, cuando expiro el tiempo los bundle fueron eliminados de los nodos C y D.

Conclusión

El algoritmo que determina si conservar un bundle o no funciono de la manera esperada. Fue sorprendente el hecho de que algunos nodos detectaban a todos los vecinos y otros no.

El tiempo de propagación de la información del semáforo hacia los diferentes nodos fue de segundos. Esto hace pensar que en una situación real en la que hubiera mucho tráfico los vehículos recibirían esta información en un tiempo reducido.

6.3 Experimentos simulando sistema completo

En esta sección se explican los experimentos hechos con el objetivo de comprobar el correcto funcionamiento del controlador del semáforo cuando funciona con más de un nodo y envía mensajes por DTN.

6.3.1 Experimento ctrl-dtn-1: simulación del sistema final

Preparación

Para realizar siguiente experimento se han utilizado 3 Raspberry conectadas a la misma red vía ethernet. En todas ellas se ha habilitado el acceso SSH para poder acceder a ellas desde un ordenador situado en la misma red. Previamente en todas ellas se ha instalado la librería Adtn Plus y el software controlador del semáforo.

Una de las Raspberry fue configurada como MASTER y otra como SLAVE. Amabas fueron configuradas para ejecutar de forma conjunta una secuencia de cambios de estado durante que cambiaba a partir de una determinada hora.

La tercera Raspberry se habilito para funcionar en modo CAR. En este modo de funcionamiento el software funciona simulando el funcionamiento del software que estaría funcionando en un coche. Es decir, renciona bundles a través de DTN.

Resultado

El sistema estuvo funcionando durante más de 10 minutos sin mostrar ningún tipo de problema. En este tiempo la Rasberry que hacía de coche recibió todos los bundles que fueron enviados por el MASTER. Este era el único que enviaba bundles.

Conclusión

El sistema es capaz de funcionar con más de un nodo al mismo tiempo que envía información a través de DTN.

7 Conclusiones

Durante la primea etapa de este proyecto se ha desarrollado un programa capaz de controlar un semáforo. Sobre este programa se han hecho diversos experimentos, algunos de ellos mencionados en este documento. La mayoría de estos experimentos han tenido los resultados esperados.

En la segunda etapa se integró la tecnología DTN. En la actualidad la tecnología DTN y su extensión ADTN son poco conocidas. Ambas tecnologías están en fase de experimentación y son utilizadas en proyectos de investigación. Hay un número reducido de implementaciones de estas tecnologías. Además, la información que se puede encontrar sobre estas tecnologías habitualmente es teórica. Estos hechos hacen que sea difícil que un proyecto de ámbito comercial decida utilizar estas tecnologías.

DTN y ADTN no substituyen a las tecnologías clásicas, sino que resuelven problemas diferentes. En este trabajo se puede ver como se ha utilizado ADTN para enviar información desde un origen hacia un número indefinido de destinos por un camino indeterminado. En una tecnología de red clásica es necesario determinar un destino y que haya un camino predefinido. Por lo tanto, estas tecnologías pueden ser muy útiles en casos de uso en los cuales no hay un destino fijado.

En los experimentos realizados con más de un nodo ADTN se ha visto que el envió de un mensaje de un nodo a otro es instantáneo cuando ambos nodos se detectan como vecinos. Esto hace que en situaciones donde hay muchos nodos unos cercanos a otros los mensajes se puedan propagar rápidamente.

Realizar estos experimentos con ADTN es un trabajo tedioso y lento. No se ha podido realizar un experimento con un vehículo real en un entorno que simulara la realidad dado que es un experimento que necesita mucho tiempo de preparación y excedería el tiempo que hay para realizar un trabajo final de grado.

Por ultimo indicar que pese a que la Raspberry Pi es un buen hardware sobre el que trabajar presenta el problema de que no tiene integrado ningún sistema para mantener la hora cuando esta se apaga. Para solucionar este problema se acoplo un reloj hardware alimentado por una pila.

7.1 Trabajo futuro

En el proyecto no se aplica ningún tipo de seguridad en las comunicaciones. Por lo tanto, un siguiente paso sería agregar seguridad a todas las comunicaciones. Otro posible avance podría ser mejorar la aplicación ADTN que va situada en el vehículo. Esta podría ofrecer una interfaz para que cualquier aplicación de navegación pudiera acceder a los datos de los semáforos de forma transparente. Además, también se podría conectar un sistema de visión por computador que analizara el paso de los vehículos con la aplicación que controla el semáforo para que este funcionara de forma completamente autónoma.

Agradecimientos

Este trabajo de final de grado no habría sido posible sin la colaboración del CVC y SENDA. El CVC me ha ayudado a conseguir todos los materiales necesarios para poder realizar este trabajo y también me ha dado un espacio de trabajo. En especial agradezco mucho el trabajo del Dr. David Vázquez ya que es un gran motivador y consiguió los semáforos. Por parte del grupo SENDA agradecer en primer lugar a Àngela Fàbregas por soportar todos mis correos pidiendo ayuda y ayudarme a solucionar los problemas relacionados con la librería aDTN. Por último, y no menos importante, agradecer a Sergi Robles y Carlos Borrego el permitirme realizar este trabajo de final de grado. Ellos son dos de los mejores profesores que he tenido.

Referències

[1] *Computer Vision Center*, `http://www.cvc.uab.es/`, (cons. 22-06-2017).

[2] *SeNDA*, `http://senda.uab.es/`, (Accessed on 06/22/2017).

[3] *Advanced Driver Assistance Systems*, `http://adas.cvc.uab.es/`, (cons. 22-06-2017).

[4] *Elektra – Autonomous Vehicle developed by CVC & UAB & UPC*, `http://adas.cvc.uab.es/elektra/`, (cons. 22-06-2017).

[5] S. S. de control de transito. (2017). ¿Cómo funciona?, adr.: `http://www.sctvial.com/como-funciona` (cons. 10-06-2017).

[6] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall i H. Weiss, "Delay-Tolerant Networking Architecture", RFC Editor, RFC 4838, 2007. adr.: http://www.rfc-editor.org/rfc/rfc4838.txt (cons. 13-06-2017).

[7] K. Scott i S. Burleigh, "Bundle Protocol Specification", RFC Editor, RFC 5050, 2007. adr.: http://www.rfc-editor.org/rfc/rfc5050.txt (cons. 13-06-2017).

[8] NASA. (2017). NASA - DTN, adr.: https://www.nasa.gov/directorates/heo/scan/engineering/technology/txt_dtn.html (cons. 11-06-2017).

[9] B. S.T.U.o.C.B.C.U. S. Kevin Gifford Ph.D. (2015). NASA - DTN, adr.: https://www.nasa.gov/mission_pages/station/research/experiments/730.html (cons. 11-06-2017).

[10] A. F.A.S.-C. Carlos Borrego Sergi Robles, "A mobile code bundle extension for application-defined routing in delay and disruption tolerant networking", Article, 2015. adr.: http://www.sciencedirect.com/science/article/pii/S1389128615001942 (cons. 12-06-2017).

[11] A. F.A. S. Carlos Borrego Sergi Robles. (2017). SENDA - ADTN, adr.: http://senda.uab.es/Active-DTN (cons. 11-06-2017).

[12] S. c. Angela Fabregues Blackcatn13. (). GitHub - aDTN PLUS, adr.: https://github.com/SeNDA-UAB/aDTNPlus (cons. 11-06-2017).

[13] C. G. Rodríguez, "Red Oportunista para la Regulación del Tráfico", TFG, 2016. adr.: https://ddd.uab.cat/pub/tfg/2016/tfg_49411/Articulo.pdf (cons. 13-06-2017).

[14] D. C. Roca, "Desenvolupament d'una aplicació de reportatge d'incidències sobre DTN", TFG, 2015. adr.: https://ddd.uab.cat/pub/tfg/2015/tfg_27447/ARTICLE_TFG_David_Cara_Roca.pdf (cons. 13-06-2017).

[15] *Señalización, seguridad vial y movilidad - Industrias Saludes*, http://www.industriassaludes.es/, (cons. 22-06-2017).

[16] R. Pi. (2017). Raspberry Pi GPIO Usage, adr.: https://www.raspberrypi.org/documentation/usage/gpio/ (cons. 11-06-2017).

[17] B. E. Ltd, "BCM2835 ARM Peripherals", GUIDE, 2012. adr.: https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2835/BCM2835-ARM-Peripherals.pdf (cons. 13-06-2017).

[18] WiringPi. (2017). WiringPi Website, adr.: http://wiringpi.com/ (cons. 11-06-2017).

[19] J. Postel, "Transmission Control Protocol", RFC Editor, STD 7, 1981. adr.: http://www.rfc-editor.org/rfc/rfc793.txt (cons. 13-06-2017).

[20] Vitutor. (2017). Ecuación de la recta, adr.: http://www.vitutor.com/geo/rec/d_6.html (cons. 13-06-2017).

[21] *Linear algebra - Calculate if a point lies above or below (or right to left) of a line - Mathematics Stack Exchange*, https://math.stackexchange.com/questions/1435779/calculate-if-a-point-lies-above-or-below-or-right-to-left-of-a-line, (cons. 22-06-2017).

[22] T. Ylonen i C. Lonvick, "The Secure Shell (SSH) Protocol Architecture", RFC Editor, RFC 4251, 2006. adr.: http://www.rfc-editor.org/rfc/rfc4251.txt (cons. 13-06-2017).