Herramienta de monitorización de misiones

para robots móviles

Resumen del Proyecto de Fin de Carrera

Ingeniería en Informática

Junio 2010

Autor

Víctor Teniente Mateos

Tutoras

Belén Curto Diego

Ángeles Mª Moreno Montero

VNiVERSiDAD
D SALAMANCA

1. Introducción

El trabajo que se propone en este proyecto se enmarca en el sector productivo/industrial, formando parte de un proyecto mayor que consiste en la automatización de una carretilla industrial con el fin de conseguir un sistema independiente capaz de realizar tareas de almacenaje sin intervención del usuario.

El sistema global diseñado integrará la teleoperación con la automatización y ciertas capacidades autónomas, donde la seguridad será un aspecto crucial. En aquellas actividades donde se necesite cierta destreza (como la carga y manipulación de diferentes materiales), o presenten un alto grado de complejidad o riesgo, el vehículo podrá ser manejado por un operador desde una estación local. La finalidad es obtener un sistema autónomo para el transporte de mercancías.

El despliegue se realizará sobre la carretilla industrial de la Figura 1 con la finalidad de obtener un sistema autónomo para el transporte de mercancías. Todo este sistema será gestionado a través de una arquitectura de control definida.

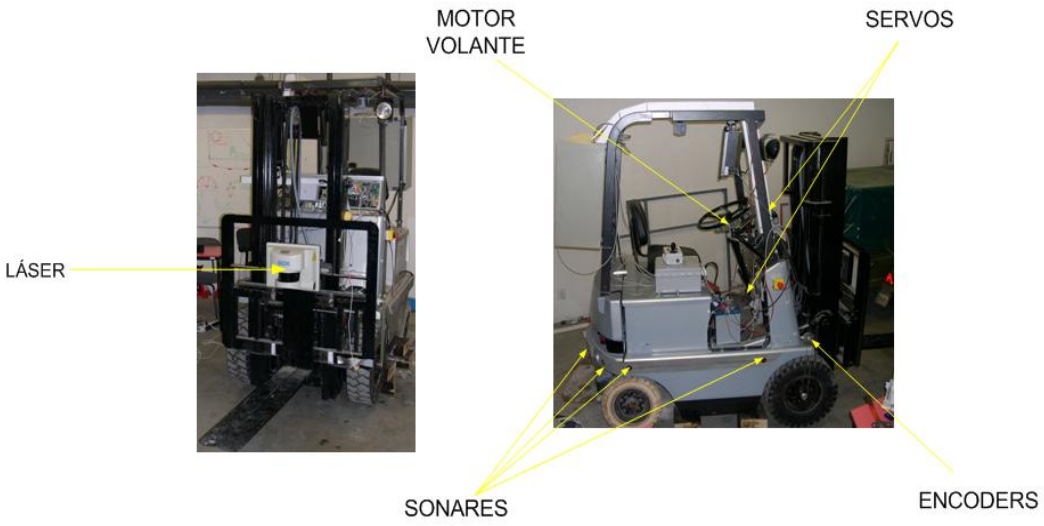

Figura 1. Carretilla industrial utilizada en el proyecto y localización de los sensores

El objetivo principal de este proyecto de fin de carrera es crear un nuevo componente e integrarlo en la capa de control y visualización propia de la arquitectura de control adaptado a nuestras necesidades y que pueda coexistir junto con los componentes propios de la

arquitectura. Dicho componente debe ser totalmente independiente e integrarse dentro de la arquitectura de control de una manera transparente. Por otro lado, se diseñara e integrará un *driver* específicamente creado para la carretilla, que permita probar y analizar el correcto funcionamiento de todo el sistema global, tanto en modo real como en simulación.

La herramienta de visualización y control permitirá configurar el entorno de ejecución, así como el posterior lanzamiento de las tareas a ejecutar por la carretilla. Asimismo, una vez en ejecución y mediante un protocolo de comunicación preestablecido, se solicita la información sensorial y la referente al estado actual del sistema. La teleoperación y guiado de forma automática será otra de las funcionalidades del sistema.

En la Figura 2, puede observarse un esquema del sistema global y en color rojo los componentes desarrollados y su integración dentro de la plataforma de control.

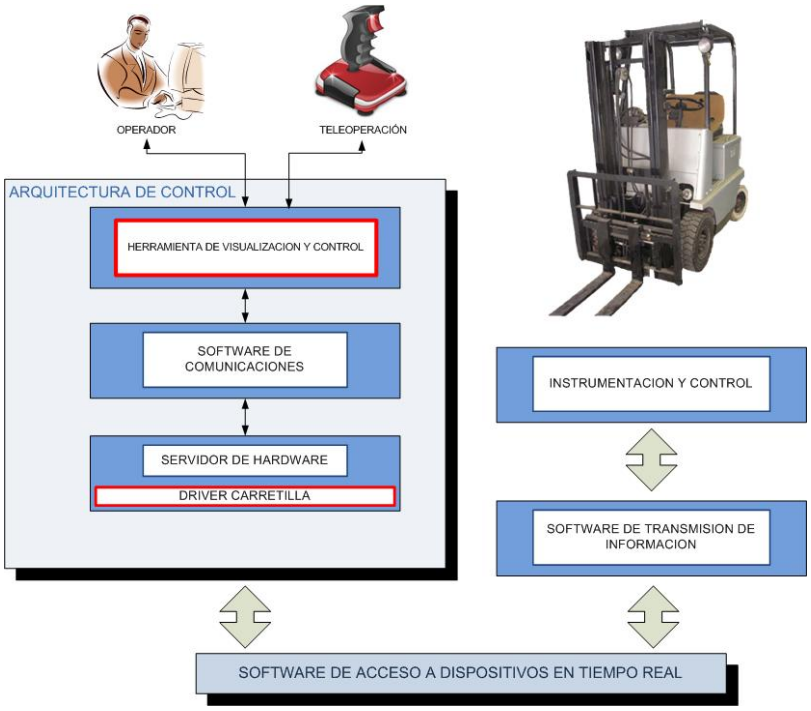

Figura 2. Esquema general del sistema

2. Objetivos del Proyecto

El objetivo principal del proyecto es desarrollar una interfaz multisensorial hombre-máquina que permita monitorizar en tiempo real el estado y los datos de posicionamiento de la carretilla industrial en diversos entornos, de manera que el operador pueda examinar la correcta conducta de la tarea asociada al robot e interactuar si fuese necesario de una manera sencilla y eficaz.

En concreto los objetivos que tiene que cumplir el sistema son:

- Integrar la aplicación con la arquitectura de control *MissionLab*.
- Diseño e implementación de un *driver* de acceso a los sensores y actuadores instalados en la carretilla.
- Realizar una monitorización completa de todos y cada uno de los sensores y actuadores instalados en la carretilla.
- Visualizar de forma gráfica el desarrollo y estado de una tarea en ejecución por parte de un robot móvil.
- Posibilitar tareas de teleoperación y supervisión de manera directa y sencilla sobre el robot.
- Integración del analizador léxico y sintáctico de *MissionLab* en la aplicación desarrollada con el fin de posibilitar la carga de archivos de mapas de manera transparente.
- La aplicación estará dotada de características multimodales[1].
- Diseño, desarrollo e implantación del sistema sobre un ordenador con pantalla táctil.

3. Aspectos relevantes

En este apartado del resumen se explicarán brevemente todos los aspectos relevantes del proyecto desarrollado.

3.1. Arquitectura de control y funcionamiento del sistema

La arquitectura de control elegida para el despliegue del sistema ha sido *MissionLab.*

[1] Se entiende por interfaz multimodal aquella que es capaz de procesar múltiples entradas (habla, tacto, etc.) y responder a ellas de la misma forma.

MissionLab, es una arquitectura de control híbrida donde la parte reactiva está basada en la teoría de motor-esquema que consta de un conjunto de herramientas y comportamientos reactivos predefinidos que facilitan el desarrollo y prueba de una tarea por parte de un robot. Permite crear misiones[2] a alto nivel y ejecutarlas en robots individuales o equipos de robots reales o simulados. Presenta una arquitectura distribuida, por lo que es posible ejecutar la misión en un ordenador mientras que el controlador o controladores de robots se encuentren en distintos nodos de una red. El acceso al hardware específico de la carretilla se realiza mediante el servidor *HServer* a través del *driver* diseñado y creado en este proyecto. Para la comunicación a través de redes TCP/IP se utiliza *IPTServer*, herramienta que proporciona una interfaz de comunicación entre los elementos de una red.

Como ya se ha comentado, el propósito de este proyecto es definir un nuevo componente dentro del subsistema ejecutivo de *MissionLab*, adaptado a las necesidades que supone la automatización de la carretilla industrial y que sea capaz de coexistir con el propio componente de la arquitectura (*mlab*). De esta manera, proporcionaremos una interfaz de alto nivel para acceder y controlar la ejecución de una misión. En la Figura 3, podemos observar las partes desarrolladas en este proyecto resaltadas con círculos rojos, y su integración dentro de la arquitectura de control.

Por un lado tenemos el *Visualizador de misiones*, encargado de configurar el entorno de ejecución y del lanzamiento de la misión, de solicitar periódicamente a los datos de los dispositivos y de permitir al operador realizar tareas de teleoperación y guiado de forma automática. Por otro lado, la creación del *driver de acceso al hardware* para la carretilla permitirá acceder a los sensores y actuadores instalados.

[2] Se entiende por misión la tarea a ejecutar por parte de un robot móvil

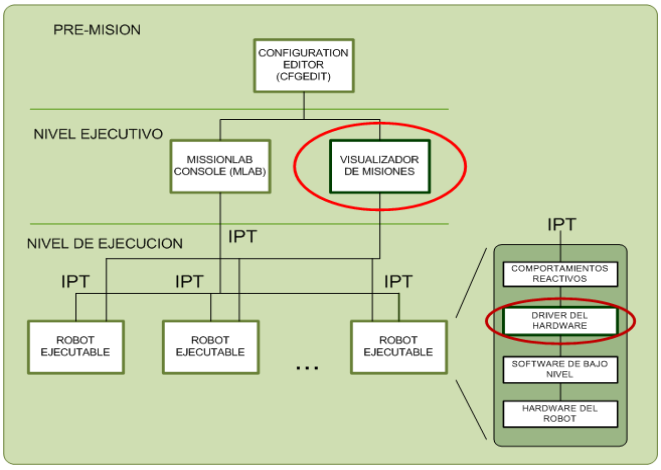

Figura 3. Componentes software del sistema global

El esquema de funcionamiento del sistema se puede observar en la Figura 4.

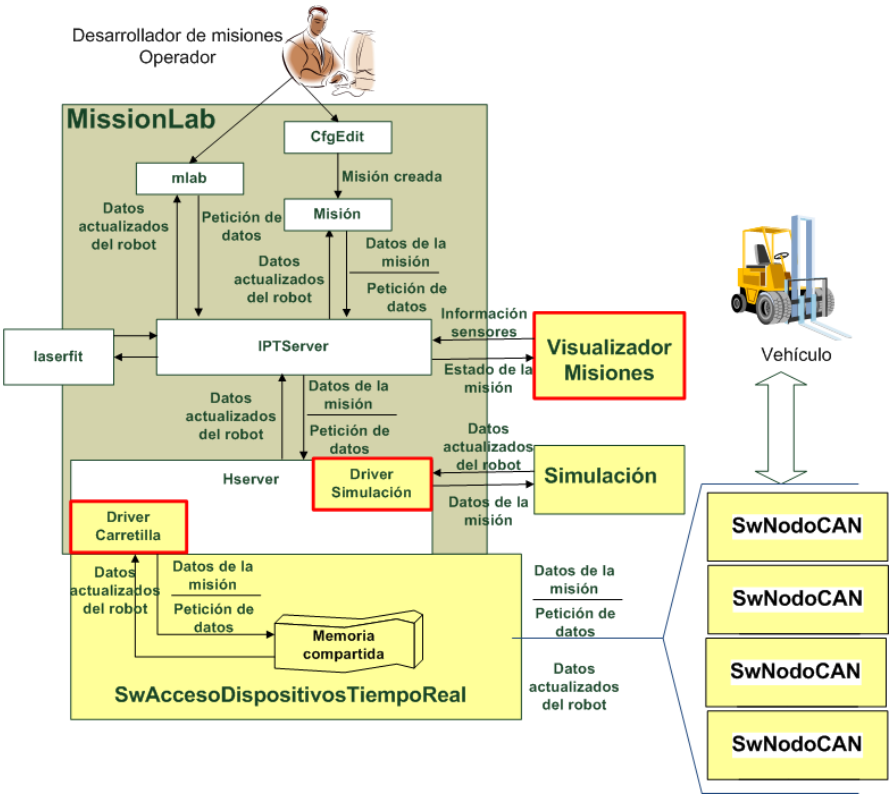

Figura 4. Esquema de funcionamiento del sistema global

En un primer momento el operador creará una misión a partir de unos comportamientos reactivos predefinidos mediante la herramienta *Cfgedit.* Una vez creada la misión se realizará el despliegue de la misma mediante el *Visualizador de Misiones*, el cual se encargará de lanzarla y

realizar una monitorización de la misma. Dicha misión, se comunicará con *Hserver* y con el *driver* de la carretilla para conocer el estado de sus sensores y actuar en consecuencia según los comportamientos definidos. Asimismo, las órdenes de control enviadas por la *Misión* de alto nivel al *robot físico* (en nuestro caso la carretilla industrial), son encapsuladas en mensajes por el *SistemaSwAccesoDispositivosTiempoReal* , encargado de establecer las comunicaciones con los sensores y actuadores instalados en la carretilla y de realizar el envío y actualización periódica de las mediciones en el robot físico

3.2. Integración del componente desarrollado en MissionLab

Para conseguir la integración del componente (*Visualizador de misiones)* dentro de *MissionLab* ha sido necesario recurrir a la ingeniería inversa para tener un conocimiento sobre esta plataforma, ya que la documentación técnica sobre MissionLab es prácticamente inexistente De este modo, se ha realizado un gran esfuerzo en revisar y estudiar su código fuente, lo que ha supuesto un considerable trabajo inicial en su aprendizaje en las primeras fases de la realización de este proyecto.

Debido a la necesidad de establecer comunicaciones entre el componente creado con los componentes propios de *MissionLab,* se ha estudiado como se manejan internamente esas comunicaciones para aplicarlo a nuestro caso y ser capaces de conseguir su integración con la arquitectura de control. Una vez realizado, podemos decir que las comunicaciones se establecen mediante *IPTserver* y los diversos canales de comunicación establecidos que se recogen en la Figura 5 . Los principales, son los establecidos para el envío de la información de los sensores/estado misión, para la realización de la teleoperación y para la simulación.

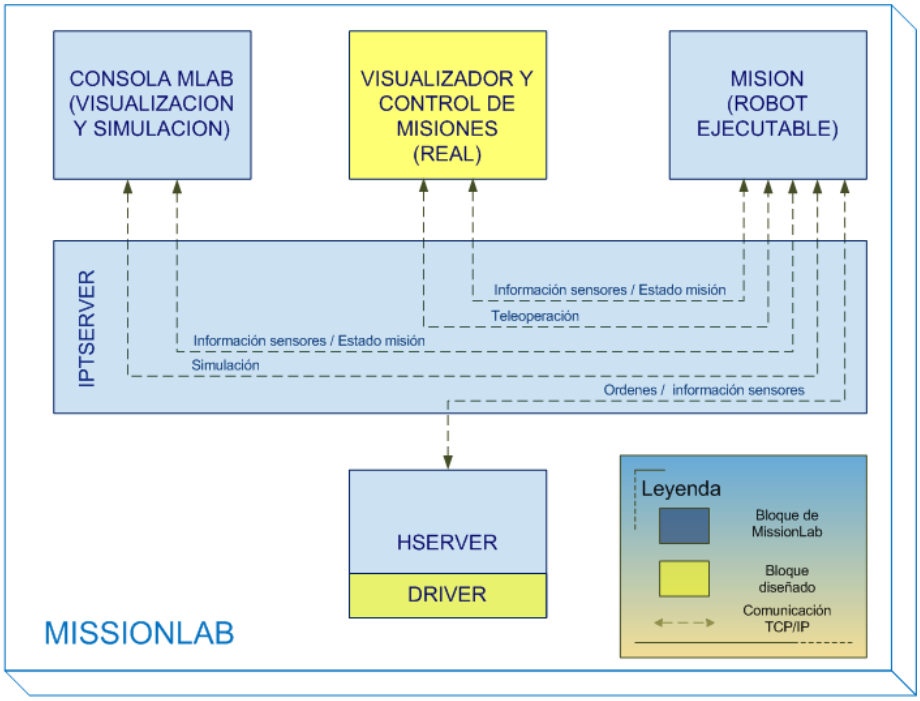

Figura 5. Canales de comunicación establecidos

Una vez definidos los canales de comunicación, la integración del componente se realizará mediante paso de mensajes registrados en cada uno de los módulos que intervienen en la comunicación. Para ello ha sido necesario definir una serie de nuevos mensajes siguiendo el formato de los existentes en la arquitectura, para poder establecer dichas comunicaciones. Posteriormente, se definirá un protocolo de comunicación haciendo uso de dichos mensajes. Un ejemplo de ello es el mensaje *MSG_ObtenerSensores*, el cual solicita de forma conjunta todos los valores de los sensores instalados en la carretilla.

3.3. Desarrollo e integración de un driver para la carretilla

Se ha desarrollado y creado un *driver* con el fin de integrarlo dentro de la arquitectura de control, más concretamente en *HServer*. Este *driver* permitirá simular el comportamiento de la carretilla sin necesidad de estar conectado físicamente a la misma y con la posibilidad de realizar pruebas controladas y observar resultados. También permitirá acceder a los sensores y actuadores de la carretilla en modo real.

Para que las misiones puedan recibir el estado completo de la carretilla, incluidos los sensores no definidos de forma predeterminada, se ha creado un nuevo tipo de estructuras de datos en las que se almacenan los datos de todos los sensores específicos de la carretilla. De esta forma pueden ser enviados desde las capas de bajo nivel (*driver*) a las capas de alto nivel (Visualizador de misiones).

Ha supuesto una tarea complicada debido a la necesidad de utilizar conceptos de bajo nivel.

3.4. Adaptación del parser de MissionLab para la generación de mapas

Debido a la necesidad de mostrar los archivos de mapas propios de *MissionLab,* ha sido de nuevo necesario examinar e investigar el proceso de creación e interpretación de este tipo de archivos.

MissionLab define un lenguaje propio denominado ODL (*Overlay Description Language*), que se utiliza para describir el entorno de las misiones tanto para *mlab* como para la herramienta de *Visualización de Misiones* creada en este proyecto.

Estos ficheros se estructuran en dos partes: por un lado información relativa al escenario y por otro la descripción de todas las medidas de control. Para interpretar este tipo de ficheros, *MissionLab* define una gramática mediante un archivo escrito en Yacc, junto con un analizador léxico escrito en Lex.

En este momento surge la necesidad de aislar el parser de *MissionLab* y adaptarlo para que funciones en nuestra aplicación. Para ello se ha realizado una adaptación de los ficheros que definen la gramática, de manera que a la hora de analizar este tipo de ficheros de entrada se invoquen las funciones necesarias para realizar el dibujado del mapa en nuestra aplicación.

Por este motivo, se han definido una serie de clases que encapsulan las funciones primitivas de dibujado bajo Qt, como pueden ser polilinea, arco, línea, etc. En la Figura 6, puede observarse la aplicación en ejecución con un mapa cargado

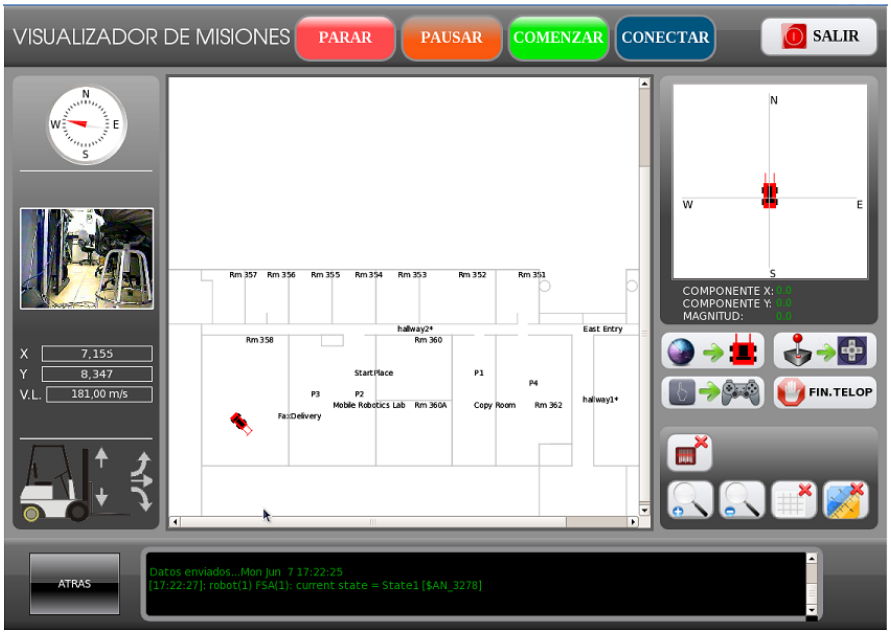

Figura 6. Carga de fichero de mapas en la aplicación

3.5. Estudio del mecanismo de teleoperación de MissionLab

Uno de los puntos principales de la aplicación desarrollada es la posibilidad de realizar tareas de teleoperación de manera sencilla por parte del operador. Por este motivo, es necesario realizar un estudio acerca de cómo *MissionLab* gestiona la teleoperación para poder así introducirlo y adaptarlo a nuestras necesidades, de manera que combinado con los mecanismos de teleoperación definidos en la aplicación visual, seamos capaces de realizar este tipo de tareas.

Es por ello que se han integrado dentro de la aplicación lar órdenes de teleoperación propias de *MissionLab*.

3.6. Desarrollo de una interfaz multimodal táctil

Una interfaz multimodal proporcionará al operador diferentes maneras de control, como son por ejemplo, componentes con realimentación gráfica y visual. Las interfaces de este tipo toman información de varios sensores y la combinan en una aplicación gráfica de manera que el operador pueda interactuar y conocer el estado de una manera más sencilla.

Para conseguir una interfaz de este tipo en nuestro proyecto, se han integrado una serie de elementos para mejorar significativamente la interacción con el operador. Una muestra de ello son los componentes gráficos creados para visualizar los datos procedentes de los sensores y que pueden verse en la Figura 7.

El primero de ellos muestra una barra de desplazamiento adaptada para representar valores entre dos límites, permite personalizar colores, tipo de numeración, etc. El segundo representa un widget de cinco botones combinados en forma de círculo para representar de forma grafica un joystick. El último representa una barra de progreso, en la cual se representan valores entre un límite superior e inferior con un gradiente de color entre ambos valores.

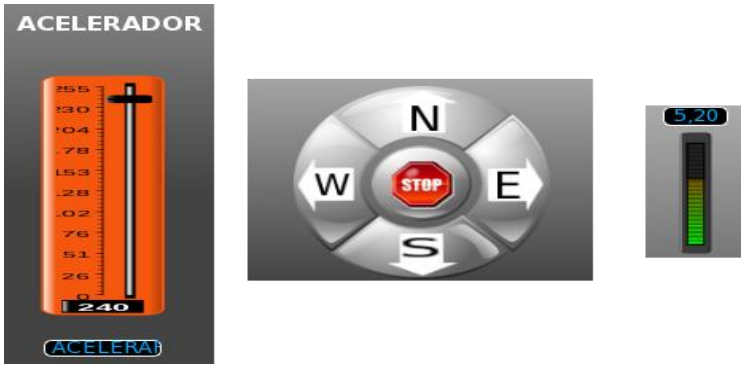

Figura 7. Componentes gráficos de la aplicación

Otra característica que otorga multimodal a la aplicación, es la integración de un sintetizador de voz, permitiendo así el acceso a la información generada no mediante una interfaz visual pura, sino empleando las capacidades de salida de audio que son comunes en casi cualquier tipo de terminal.

Además del sintetizador, se ha incluido en la aplicación un visor para mostrar las imágenes de una cámara web conectada también a la carretilla de manera que el operador pueda obtener una percepción más completa y real del entorno.

El despliegue de la aplicación es un terminal táctil, y por tanto el desarrollo de la misma ha estado influido por las líneas de desarrollo propias de este tipo de interfaces.

En las aplicaciones con pantalla táctil, surgen una serie de desafíos debidos a las características específicas de este tipo de interfaces.

En la Tabla 1 se detallarán algunos de los principales desafíos y como se han solucionado en el presente proyecto.

PROBLEMA	SOLUCION
Organización de los componentes	Se han identificado y dividido la aplicación en grupos funcionales de elementos situados de forma conjunta y organizados en áreas funcionales.
Ausencia de teclado y ratón	Se ha evitado la entrada de datos por teclado de modo que la interacción se realice mediante elementos con valores prefijados o haciendo clic repetidamente sobre ciertos componentes gráficos.
Tamaño de los botones y menús y realimentación de su estado	Se ha definido un tamaño de botón suficientemente grande apto para cualquier tipo de dedo. Además se indica el estado del botón mediante diferentes iconos. Definición de menús mediante botones
Interacción con la aplicación	Interfaz simple con botones adecuados. La jerarquía de la aplicación en dos niveles: pantalla principal con las diversas opciones y pantallas secundarias para cada una de ellas. De este modo se consigue que el usuario no se pierda en la aplicación y que el uso de la misma sea intuitivo.

Tabla 1. Limitación y soluciones propuestas para el diseño de la interfaz táctil

3.7. Teleoperación con joystick de Playstation

Con el fin de facilitar la teleoperación en determinados casos, se ha introducido el uso de un joystick real para permitir una teleoperación manual de forma sencilla. Esto es posible gracias al módulo del kernel para joysticks.

3.8. Arquitectura del sistema

La arquitectura global de la aplicación desarrollada ha sido realizada desde el punto de vista del patrón *layers*, con la restricción de que los componentes de una misma capa sólo pueden hacer referencia a componentes de la misma capa o de capas inferiores. En la Figura 8 puede observarse el resultado final de la arquitectura del sistema.

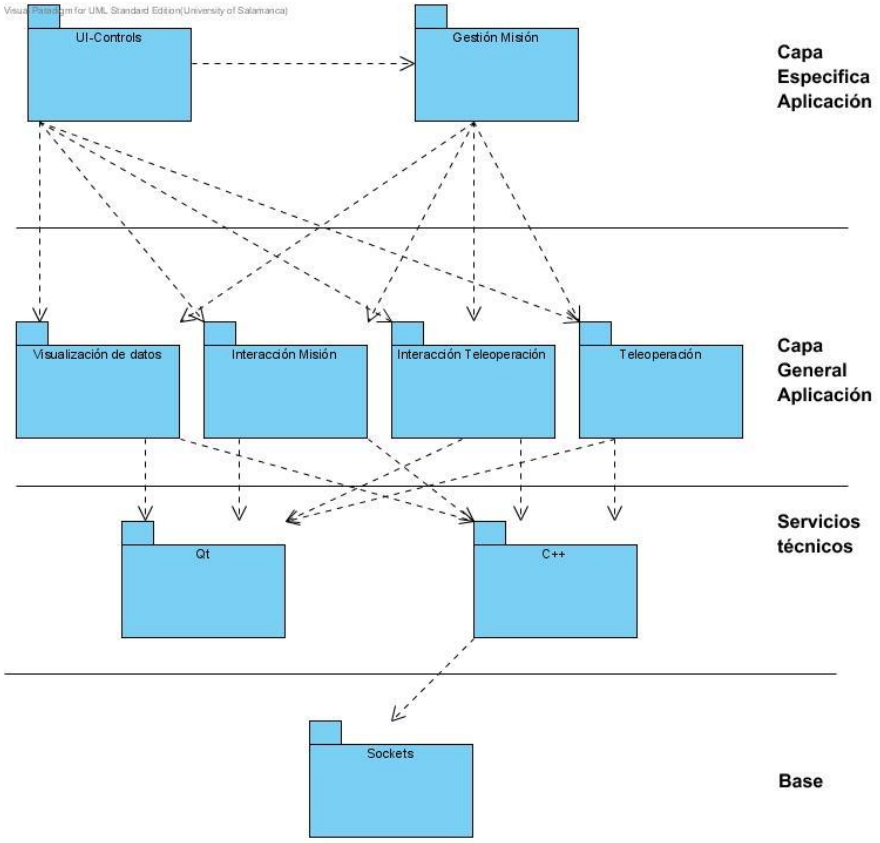

Figura 8. Dependencias entre los paquetes de la aplicación

3.9. Descripción del producto final

La aplicación *Visualizador de Misiones* ofrece una interfaz multimodal hombre máquina que proporciona al operador una herramienta de monitorización y control de misiones en ejecución, combinada con la posibilidad de ejecutar tareas de teleoperación de manera sencilla para el operador.

Dispone de diferentes maneras de control sobre la carretilla, con el fin de ayudar al operador a mantener una supervisión global y adecuada de todo el sistema.

Las principales características se describen en los siguientes puntos.

- Visualización de datos

Permite llevar a cabo un control y monitorización de los valores obtenidos desde la carretilla de manera síncrona. Navegando por las diversas secciones de la aplicación (Figura 9) podemos realizar un seguimiento de los sonares, servos, encoders, y de todos los demás sensores instalados en la carretilla.

Figura 9. Ventanas de la aplicación

- Generación de gráficas

La posibilidad de crear gráficas de manera síncrona con la recepción de los datos procedentes de la carretilla, ofrece al operador una nueva forma de control y monitorización con el fin de observar anomalías en los sensores (Figura 10).

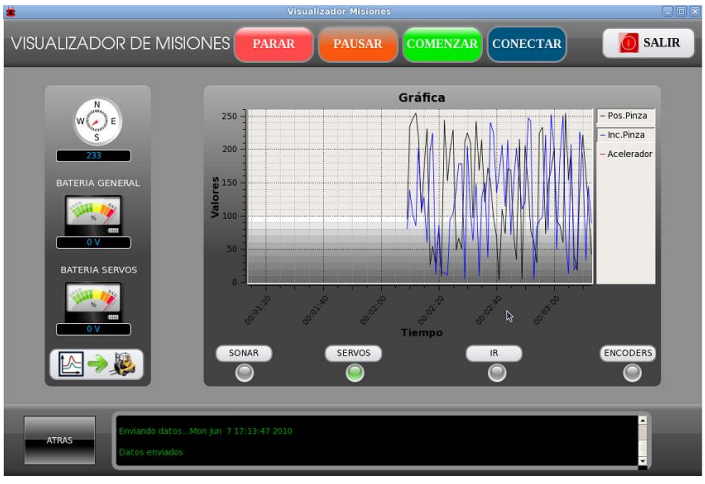

Figura 10. Generación de gráficas en la aplicación

- Monitorización de la misión en ejecución

La aplicación permite realizar un control y monitorización completo de la ejecución de la misión. En la parte central se observa el desarrollo de la misión, con la posibilidad de visualizar de manera actualizada los valores referentes a los sonares (triángulos dorados) y al láser (puntos rojos y verdes). Además, se incluye un visor para mostrar las imágenes de la cámara.

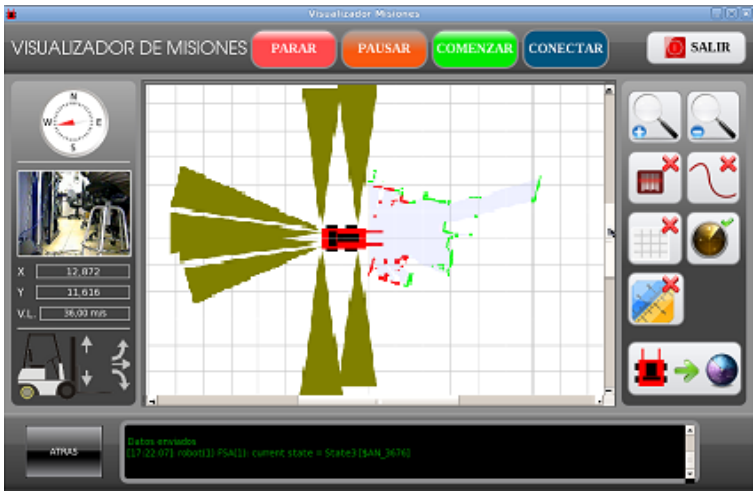

Figura 11. Ventana de monitorización de la misión

- Teleoperación sobre la misión en ejecución

Otro de los puntos importantes, es la posibilidad de ejecutar operaciones de teleoperación de manera remota mediante el uso de controles gráficos diseñados e integrados en la aplicación (Figura 12). Se han creado dos tipos de controles, el izquierdo de ajuste más preciso y el derecho de ajuste más grueso.

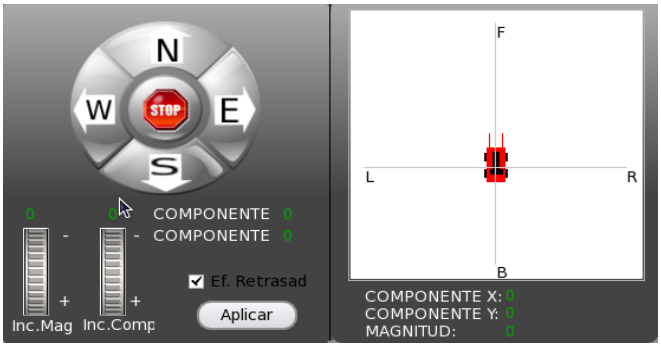

Figura 12. Controles gráficos de teleoperación

En la Figura 13 se puede observar la ventana de teleoperación en la que se ha seguido el mismo diseño que en la ventana de navegación.

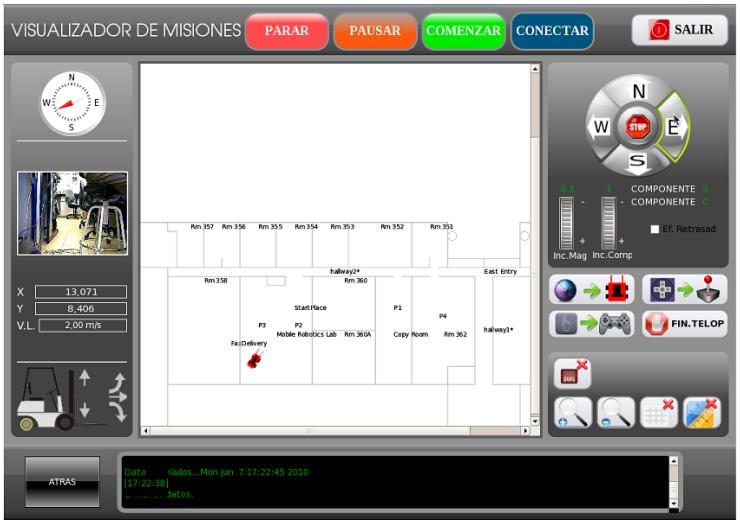

Figura 13. Ventana de teleoperación

Para completar las operaciones de teleoperación, cabe destacar la integración y posibilidad de uso de joystick de *Playstation* para realizar estas tareas de una manera más sencilla por parte del operador.

4. Entorno de desarrollo y explotación

El desarrollo del sistema se ha realizado utilizando el lenguaje de programación C++ combinado con el *framework* de interfaces gráficas *Qt*.

La explotación se realiza sobre un PC con una distribución *Fedora Core 11* o superior, mediante la instalación del archivo *RPM* proporcionado.

5. Conclusiones

Tras finalizar este proyecto podemos concluir que se han alcanzado todos los objetivos propuestos al inicio del mismo.

El desarrollo de una interfaz multisensorial hombre-máquina ha supuesto un gran logro, ya que se ha diseñado siguiendo las directrices de diseño de este tipo de interfaces haciendo hincapié en aspectos críticos como son el tipo de usuarios y la interacción mediante los dedos de las manos.

La integración de la aplicación dentro de *MissionLab* supuso un hito muy importante, debido a la falta de documentación por parte de la arquitectura de control. En muchas ocasiones, la ingeniería inversa ha sido el único modo de comprensión de determinados aspectos. Ha requerido un esfuerzo considerable que se ha visto recompensado en los resultados obtenidos.

El acceso a los sensores y actuadores de la carretilla ha sido posible mediante la definición de nuevos mensajes integrados en la plataforma que permiten acceder de manera sincronizada a los valores generados. Dichos datos obtenidos, combinados y representados mediante *widgets* visuales creados específicamente para el proyecto, permiten llevar un control óptimo del estado de la carretilla.

Gracias a la integración de la teleoperación dentro de la aplicación desarrollada, podemos dirigir y teleoperar la carretilla de forma segura y controlada dentro del entorno de la misma. Además, la posibilidad de utilizar un mando de *PlayStation* en la ejecución de tareas de teleoperación dota de mayor sencillez y libertad al operador.

El formar parte de un proyecto ambicioso y mayor como es la automatización de una carretilla industrial ha supuesto un aliciente importante y una motivación extra para la consecución del proyecto.

Para finalizar, podemos decir que se ha adquirido gran experiencia en el desarrollo de programas adecuados a pantallas táctiles y se han podido afrontar nuevos retos de manera satisfactoria gracias a las capacidades obtenidas durante la carrera y al afán de investigación por parte de todos los componentes del Grupo de Robótica.