(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 800 321**

(51) Int. CI.:

H04L 9/32 (2006.01)
B64C 39/02 (2006.01)
G05B 19/418 (2006.01)
G01C 21/20 (2006.01)
G05D 1/00 (2006.01)
G05D 1/02 (2010.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **05.01.2017 PCT/US2017/012321**

(87) Fecha y número de publicación internacional: **14.09.2017 WO17155598**

(96) Fecha de presentación y número de la solicitud europea: **05.01.2017 E 17763681 (8)**

(97) Fecha y número de publicación de la concesión europea: **01.04.2020 EP 3400676**

(54) Título: **Arquitectura de seguridad para vehículos autónomos**

(30) Prioridad:

05.01.2016 US 201662387804 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
29.12.2020

(73) Titular/es:

CARNEGIE MELLON UNIVERSITY (100.0%)
5000 Forbes Avenue
Pittsburgh, PA 15213, US

(72) Inventor/es:

WAGNER, MICHAEL D.;
RAY, JUSTIN;
KANE, AARON y
KOOPMAN, PHILIP

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Arquitectura de seguridad para vehículos autónomos

ANTECEDENTES

La complejidad del software para vehículos no tripulados supera la de las técnicas de ingeniería de seguridad de software disponibles en la actualidad. Las normas de seguridad de software definen los procesos que se deben emplear al crear y validar software. Si bien son necesarios, los procesos prescritos por las normas vigentes pueden no ser suficientes para garantizar la seguridad de software autónomo en vehículos de conducción autónoma. En algunos casos, los procedimientos para una autonomía avanzada, tales como el aprendizaje automático, no se pueden validar fácilmente utilizando procedimientos de prueba de software tradicionales. Como resultado, se han utilizado componentes de supervisión independientes invariables en tiempo de ejecución para acotar la criticidad de la seguridad en un pequeño subconjunto de la arquitectura, separando así las técnicas de ingeniería de seguridad de software intensivas en recursos del software de autonomía complejo y centrándolas en componentes de supervisión mucho más simples. Sin embargo, hasta ahora, dichas técnicas se han implantado con más éxito en vehículos no tripulados controlados a distancia o teledirigidos. Hasta la fecha, no ha quedado claro cómo se podrían utilizar más eficazmente los componentes de supervisión invariables en tiempo de ejecución para mitigar los riesgos de seguridad que plantean funciones autónomas tales como la planificación y el control.

El documento US 2008/0234861 A1 describe un sistema de control para controlar los movimientos de una pluralidad de unidades mecánicas. El sistema de control incluye un programa que incluye una pluralidad de programas de unidades mecánicas. Cada programa incluye instrucciones de movimiento para al menos una de las unidades mecánicas. El sistema de control también incluye una pluralidad de planificadores de rutas. Al menos uno de los planificadores de rutas está adaptado para recibir instrucciones desde más de uno de los programas de unidades mecánicas y, en función de las mismas, determinar cómo deben moverse las unidades mecánicas para sincronizar sus movimientos. El sistema de control incluye además conmutadores adaptados para cambiar un programa de unidades mecánicas de un planificador de rutas a otro, por lo que los movimientos de las unidades mecánicas se sincronizan cuando sus programas de unidades mecánicas están conectados al mismo planificador de rutas y los movimientos de las unidades mecánicas son independientes cuando sus programas de unidades mecánicas están conectados a diferentes planificadores de rutas.

El documento US 2009/0292422 A1 describe un procedimiento para desechar un dispositivo de seguridad pirotécnica que incluye proporcionar una unidad de control electrónico que tiene una unidad de control primaria y una unidad de control auxiliar. La unidad de control auxiliar incluye un modo de protección y un modo de desguace, y funciona en el modo de protección en un estado inicial. La unidad de control auxiliar pasa del modo de protección al modo de desguace cuando la unidad de control primaria envía una primera señal predeterminada. La unidad de control auxiliar se activa solo si recibe una segunda señal predeterminada desde la unidad de control primaria mientras la unidad de control auxiliar está funcionando en el modo de desguace. La unidad de control primaria envía la primera y segunda señales predeterminadas a la unidad de control auxiliar en función de las señales que la unidad de control primaria recibe desde una fuente externa. Después, la unidad de control primaria se deshace de un dispositivo de seguridad pirotécnica (PSD) mediante el envío de una señal de implantación al PSD en función de otra señal recibida desde la fuente externa.

RESUMEN

La presente divulgación describe una arquitectura para vehículos autónomos que incorpora algoritmos de autonomía arbitrarios en un sistema que cumple con estrictos requisitos de seguridad. En esta arquitectura, se permite que componentes de autonomía fallen arbitrariamente, incluso maliciosamente, mientras que componentes de "puerta de seguridad" de mayor integridad (por ejemplo, mayor nivel de integridad de seguridad), que podrían crearse sin necesidad de técnicas de autonomía, cumplen los requisitos de seguridad. Se crea un conjunto de fases arquitectónicas basadas en un patrón arquitectónico reutilizable para cartografiar, planificar y ejecutar trayectorias seguras. Cada fase incluye un par "realizador/comprobador" primario y un par "realizador/comprobador" secundario opcional para proporcionar un modo degradado de funcionamiento en caso de que el par primario falle. En esta divulgación, "realizar" significa llevar a cabo un control autónomo, mientras que "comprobar" significa confirmar que las señales de control son seguras de ejecutar. Si se aplica con éxito, este principio de realizador/comprobador puede ser una opción adecuada para su adopción por normas de seguridad para la creación de sistemas fiables. En arquitecturas conocidas que utilizan el patrón arquitectónico realizador/comprobador, si el realizador no actúa correctamente, el comprobador inhabilita toda la función (ambos módulos), lo que da como resultado un sistema silencioso ante fallos (es decir, cualquier fallo da como resultado un componente silencioso, también denominado en ocasiones componente de detención ante fallos o componente a prueba de fallos en casos apropiados). Esto puede plantear un desafío en sistemas autónomos en los que es habitual requerir un comportamiento de sistema operativo ante fallos (por ejemplo, una aeronave debe seguir volando incluso si hay un fallo de autonomía). La arquitectura descrita en esta divulgación aborda esta cuestión usando un enfoque multicanal para garantizar el funcionamiento continuo a pesar de uno o, posiblemente, múltiples fallos de componentes.

En un aspecto, un sistema de arquitectura de seguridad incluye una primera fase que comprende: una unidad primaria que genera datos primarios para llevar a cabo la funcionalidad normal del sistema; una unidad secundaria que genera datos secundarios para llevar a cabo una funcionalidad alternativa del sistema; una puerta de seguridad primaria acoplada a la unidad primaria, donde la puerta de seguridad primaria proporciona los datos primarios como una salida primaria en respuesta a una determinación de validez de los datos primarios; y una puerta de seguridad secundaria acoplada a la unidad secundaria, donde la puerta de seguridad secundaria proporciona los datos secundarios como una salida secundaria en respuesta a una determinación de validez de los datos secundarios. El sistema también incluye un selector de salida que está acoplado tanto a la puerta de seguridad primaria como a la puerta de seguridad secundaria de la primera fase, donde el selector de salida proporciona una salida de sistema en respuesta a las determinaciones de la validez de los datos primarios y los datos secundarios.

Las implementaciones de la divulgación pueden incluir una o más de las siguientes características. La puerta de seguridad primaria puede determinar la validez de los datos primarios en respuesta a una envolvente permisiva proporcionada por la puerta de seguridad secundaria. El sistema puede incluir una o más fases adicionales que comprenden una segunda fase, donde una salida de datos primarios de la segunda fase proporciona una entrada a la unidad primaria de la primera fase, y una salida de datos secundarios de la segunda fase proporciona una entrada a la unidad secundaria de la primera fase. La puerta de seguridad secundaria puede determinar si los datos secundarios se recibieron dentro de una ventana de tiempo predefinida para determinar si los datos secundarios son válidos. La puerta de seguridad secundaria puede incluir una memoria intermedia que almacena los datos secundarios en respuesta a la determinación de la validez de los datos secundarios. La unidad secundaria puede proporcionar datos secundarios previamente almacenados como salida secundaria en respuesta a una determinación de no validez de los datos secundarios. La salida del sistema puede incluir datos de control para hacer funcionar un vehículo.

En otro aspecto, un procedimiento incluye generar, mediante uno o más procesadores, datos primarios para llevar a cabo la funcionalidad normal del sistema; generar, mediante el uno o más procesadores, datos secundarios para llevar a cabo una funcionalidad alternativa del sistema; proporcionar, mediante el uno o más procesadores, los datos primarios como una salida primaria de una primera fase en respuesta a la determinación de validez de los datos primarios; proporcionar, mediante el uno o más procesadores, los datos secundarios como salida secundaria de la primera fase en respuesta a la determinación de validez de los datos secundarios; y proporcionar, mediante el uno o más procesadores, una salida de sistema en respuesta a la determinación de validez de los datos primarios y los datos secundarios.

Las implementaciones de la divulgación pueden incluir una o más de las siguientes características. La determinación de la validez de los datos primarios puede responder a una envolvente permisiva. La generación de los datos primarios puede incluir recibir una entrada primaria a través de una salida de datos primarios de una segunda fase; y generar los datos secundarios puede incluir recibir una entrada secundaria a través de una salida de datos secundarios de una segunda fase. La determinación de la validez de los datos secundarios puede incluir determinar que los datos secundarios se generaron dentro de una ventana de tiempo predefinida. El procedimiento puede incluir almacenar los datos secundarios en respuesta a la determinación de la validez de los datos secundarios. El procedimiento puede incluir proporcionar datos secundarios previamente almacenados como salida secundaria de la primera fase en respuesta a la determinación de no validez de los datos secundarios. La salida del sistema puede incluir datos de control para hacer funcionar un vehículo.

En aún otro aspecto, un sistema incluye una unidad de planificación primaria que genera datos de ruta primaria para mover un dispositivo desde una primera ubicación a una segunda ubicación; una unidad de planificación de protección que genera datos de ruta de protección para mover el dispositivo en presencia de una o más condiciones adversas durante el movimiento del dispositivo de acuerdo con los datos de ruta primaria; una puerta de seguridad de planificación primaria que recibe los datos de ruta primaria desde la unidad de planificación primaria, determina si los datos de ruta primaria proporcionan el movimiento del dispositivo de acuerdo con los datos de ruta primaria de una manera segura, y proporciona los datos de ruta primaria como una salida de ruta primaria verificada en respuesta a una determinación de que los datos de ruta primaria proporcionan el movimiento del dispositivo de acuerdo con los datos de trayectoria primaria de la manera segura; una puerta de seguridad de planificación de protección que recibe los datos de ruta de protección desde la unidad de planificación de protección, determina si los datos de ruta de protección proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas, y proporciona los datos de ruta de protección como una salida de ruta de protección verificada en respuesta a una determinación de que los datos de ruta de protección proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas; una unidad de ejecución de trayectoria primaria que recibe la salida de ruta primaria verificada y genera datos de trayectoria primaria a partir de un punto de recorrido actual del dispositivo en función de la salida de ruta primaria verificada; una unidad de ejecución de trayectoria de protección que recibe la salida de ruta de protección verificada y genera datos de trayectoria de protección a partir de un punto de recorrido actual del dispositivo en función de la salida de ruta de protección verificada; una puerta de seguridad de trayectoria primaria que recibe los datos de trayectoria primaria desde la unidad de ejecución de trayectoria primaria, determina si los datos de trayectoria primaria son coherentes con un estado actual del dispositivo y proporciona los datos de trayectoria primaria como una salida de trayectoria primaria verificada en respuesta a una determinación de que los datos de trayectoria primaria son coherentes con el estado actual del dispositivo; una puerta de seguridad de trayectoria de protección que recibe los datos de trayectoria de protección desde la unidad de ejecución de trayectoria de protección, determina si los datos

de trayectoria de protección son coherentes con el estado actual del dispositivo y proporciona los datos de trayectoria de protección como una salida de trayectoria de protección verificada en respuesta a una determinación de que los datos de trayectoria de protección son coherentes con el estado actual del dispositivo; y un selector de prioridad que está acoplado a la puerta de seguridad de trayectoria primaria para recibir la salida de trayectoria primaria verificada, a la puerta de seguridad de trayectoria de protección para recibir la salida de trayectoria de protección verificada, y a un controlador para proporcionar datos de control, donde el selector de prioridad proporciona como datos de control uno de: la salida de trayectoria primaria verificada si se recibe la salida de trayectoria primaria verificada, la salida de trayectoria de protección verificada si solo se recibe la salida de trayectoria de protección verificada, o una salida predeterminada si no se recibe ni la salida de trayectoria primaria verificada ni la salida de trayectoria de protección verificada.

Las implementaciones de la divulgación pueden incluir una o más de las siguientes características. La unidad de ejecución de trayectoria de protección puede generar una envolvente permisiva que especifica una velocidad de aceleración mínima, una velocidad de aceleración máxima, una velocidad de desaceleración mínima, una velocidad de desaceleración máxima, una velocidad de cambio de curvatura mínima y una velocidad de cambio de curvatura máxima; y la puerta de seguridad de trayectoria primaria puede determinar si los datos de trayectoria primaria se encuentran dentro de los valores especificados por la envolvente permisiva para determinar si los datos de trayectoria primaria son coherentes con un estado actual del dispositivo, y puede proporcionar la salida de trayectoria primaria verificada en respuesta a una determinación de que los datos de trayectoria primaria se encuentran dentro de los valores especificados por la envolvente permisiva. La puerta de seguridad de trayectoria de protección puede determinar si los datos de trayectoria de protección se recibieron dentro de una ventana de tiempo predefinida para determinar si los datos de trayectoria de protección son coherentes con el estado actual del dispositivo. La puerta de seguridad de trayectoria de protección puede incluir una memoria intermedia que almacena los datos de trayectoria de protección en respuesta a una determinación de que los datos de trayectoria de protección son coherentes con el estado actual del dispositivo. La puerta de seguridad de trayectoria de protección, en respuesta a una determinación de que los datos de ruta de protección no son coherentes con el estado actual del dispositivo, puede acceder a datos de trayectoria de protección previamente almacenados, puede determinar si los datos de trayectoria de protección previamente almacenados son coherentes con el estado actual del dispositivo, y puede proporcionar los datos de trayectoria de protección previamente almacenados como salida de trayectoria de protección verificada en respuesta a una determinación de que los datos de trayectoria de protección previamente almacenados son coherentes con el estado actual del dispositivo. El dispositivo puede incluir un vehículo y la una o más condiciones adversas pueden incluir al menos una de una condición que prohíbe el movimiento del vehículo de acuerdo con los datos de ruta primaria o un fallo de un componente del vehículo que impide que pueda lograrse el movimiento del vehículo de acuerdo con los datos de trayectoria primaria.

Todo o parte de lo anterior se puede implementar como un producto de programa informático que incluye instrucciones que están almacenadas en uno o más medios de almacenamiento no transitorios legibles por máquina y que se pueden ejecutar en uno o más dispositivos de procesamiento. Todo o parte de lo anterior puede implementarse como un aparato, procedimiento o sistema electrónico que puede incluir uno o más dispositivos de procesamiento y memoria para almacenar instrucciones ejecutables para implementar las funciones indicadas.

La materia objeto descrita en esta memoria descriptiva puede implementarse para obtener una o más de las siguientes ventajas potenciales. La brecha entre los complejos algoritmos de autonomía y los sistemas de software fiables se puede cerrar al permitir que componentes de autonomía se integren en un marco de alta fiabilidad. Dicho marco puede garantizar el funcionamiento seguro del sistema incluso cuando componentes individuales, tales como los componentes de autonomía, fallan de una manera arbitrariamente perjudicial (por ejemplo, tanto fallos accidentales como un comportamiento maliciosamente no seguro de un componente individual). Además, el comportamiento a nivel de sistema con funcionamiento ante fallos se puede proporcionar incluso cuando los componentes individuales fallan o se deben desactivar debido a comportamientos no seguros a nivel de componente. Se pueden proporcionar módulos funcionales heterogéneos para reducir la posibilidad de fallos en modo común, y se puede proporcionar un comportamiento en modo degradado para, por ejemplo, llevar a cabo una misión de protección cuando falla la funcionalidad primaria.

Los detalles de una o más implementaciones se exponen en los dibujos adjuntos y en la siguiente descripción. Si bien se describen implementaciones específicas, existen otras implementaciones que incluyen operaciones y componentes diferentes a los ilustrados y descritos a continuación. Otras características, objetos y ventajas serán evidentes a partir de la descripción, los dibujos y las reivindicaciones.

BREVE DESCRIPCIÓN DE LAS FIGURAS

La FIG. 1 muestra un diagrama de flujo de una descripción multifase de muy alto nivel de capacidades autónomas para un vehículo terrestre autónomo.

La FIG. 2 muestra un diagrama de bloques de un sistema para evitar la necesidad de desarrollar software de autonomía de funcionalidad completa de acuerdo con normas críticas de seguridad.
La FIG. 3 muestra un mapa de costes binario basado en cuadrícula.

La FIG. 4 muestra un diagrama de bloques de un ejemplo generalizado de una versión de dos canales de la arquitectura de autonomía segura.

La FIG. 5 muestra un diagrama de bloques de un ejemplo de una instanciación de sistema de una arquitectura de autonomía segura.

La FIG. 6 muestra un diagrama de un ejemplo de una cuadrícula de ocupación para la planificación de movimiento.

La FIG. 7 muestra un diagrama de un ejemplo de una cuadrícula de ocupación para la planificación de movimiento como reacción a un árbol que cae.

La FIG. 8 muestra un diagrama de flujo de un ejemplo de un proceso llevado a cabo por un sistema de arquitectura de seguridad de dispositivo autónomo.

DESCRIPCIÓN DETALLADA

La presente divulgación describe una arquitectura de propósito general que permite que los componentes de autonomía con modos de fallo arbitrariamente perjudiciales se integren en un marco de alta fiabilidad. En esta arquitectura, se permite que los componentes de autonomía fallen, mientras que los componentes de "puerta de seguridad" cumplen los requisitos de seguridad. Si bien esta divulgación describe la arquitectura en el contexto de un vehículo terrestre autónomo (AGV), la arquitectura es de propósito general para su aplicación en cualquier sistema autónomo que incluye, sin limitación, vehículos terrestres totalmente autónomos, vehículos terrestres semiautónomos, vehículos aéreos y otros sistemas robóticos con autonomía total o parcial.

A modo de ejemplo, la FIG. 1 es un diagrama de flujo 100 que resume, a muy alto nivel, las capacidades que deben implementarse de forma fiable en un AGV. En 102, el AGV genera un modelo del entorno circundante. El modelo describe, en cierto grado, la información necesaria para detectar peligros contra la seguridad, incluidos vehículos tripulados, otro tráfico, peatones, objetos en la carretera, etc. Los modelos pueden generarse a partir de datos de múltiples sensores y también pueden utilizar mapas previos.

En 104, el AGV planifica una trayectoria en el entorno que cumpla los requisitos de seguridad. Esto se puede lograr utilizando algoritmos de planificación de rutas. Por ejemplo, los planificadores de rutas pueden buscar trayectorias que eviten obstáculos y mantengan la estabilidad. En 106, el AGV ejecuta esta trayectoria. Cada una de estas capacidades debe implementarse de manera fiable en la arquitectura de software del AGV, y debe estar respaldada por alegaciones en su caso de seguridad.

Esta descomposición de comportamientos autónomos da como resultado algoritmos, donde cada uno aborda una fase de una pluralidad de fases, incluyendo modelos que fusionan lecturas de sensor en mapas, planificadores probabilísticos de hoja de ruta que encuentran una ruta hacia un destino en esos mapas y algoritmos de seguimiento de ruta que ejecutan la ruta. Debido a que la arquitectura implica "comprobaciones simples" de las salidas de control, la comprobación puede ser más sencilla si se hace dentro del alcance limitado de un solo algoritmo de control. Por lo tanto, la comprobación puede simplificarse mediante la creación de una comprobación para cada fase del sistema de autonomía.

La presente divulgación describe un patrón arquitectónico que puede instanciarse dentro de cualquier fase de procesamiento de autonomía para dar garantías acerca de las salidas enviadas a fases posteriores. Este patrón es adecuado en toda la categoría general de funcionalidad de autonomía y en otras estructuras y funciones de sistema similares, y permite sustituir diferentes algoritmos de autonomía en una fase sin interrumpir el funcionamiento de otras fases. Un sistema de acuerdo con esta arquitectura puede tener una o más fases. Como ejemplo de este patrón, esta divulgación describe técnicas adecuadas para las fases de planificación y ejecución. El patrón arquitectónico puede aplicarse adicionalmente a aplicaciones no relacionadas con la autonomía que tienen una mezcla de componentes menos fiables (por ejemplo, componentes de software disponibles en el mercado) y componentes más fiables (por ejemplo, componentes cruciales para la seguridad),

Aplicación de patrones de fiabilidad

Las técnicas implican aplicar patrones de fiabilidad probados para planificar y ejecutar trayectorias seguras de forma fiable. Proporcionar redundancia ejecutando el mismo software en dos ordenadores diferentes puede ser ineficaz a la hora de mitigar los fallos de diseño de software porque existe la posibilidad de que ambas copias de software fallen al mismo tiempo a partir del mismo defecto de software si se activa dicho defecto. Tal uso de software diverso (también conocido como programación de múltiples versiones) requiere que cada copia del software de autonomía sea crucial para la seguridad, duplicando así (o más) el coste de desarrollar software en comparación con una sola copia. En lo que respecta al software de autonomía, es posible que no se sepa cómo crear incluso una copia de software de alta integridad con la funcionalidad requerida, lo que puede hacer que dicho enfoque sea inviable.

La FIG. 2 muestra un diagrama de bloques de un sistema 200 para evitar la necesidad de desarrollar software de autonomía de funcionalidad completa de acuerdo con normas cruciales de seguridad. El sistema 200 ilustrado en la FIG. 2 se conoce como arquitectura simple. La arquitectura simple incluye dos componentes de control diferenciados: el subsistema complejo 202 y el subsistema de seguridad 204. El subsistema complejo 202 puede ser un algoritmo de control sofisticado que es difícil de desarrollar hasta un nivel suficiente de integridad. El subsistema de seguridad 204 puede proporcionar características de control similares, pero simplificadas, a las del subsistema complejo 202, pero lo hace usando una implementación de alta integridad. La implementación de alta integridad puede ser mucho más simple y, por lo tanto, menos optimizada, que la implementación compleja. El subsistema de seguridad 204 puede ser una capacidad de respaldo fiable si el subsistema complejo 202 sufre un fallo. La arquitectura simple se puede utilizar para integrar de manera segura una tecnología de alto rendimiento pero menos probada. En la arquitectura simple, la lógica de decisión se encarga de desconectar el subsistema complejo 202 de una planta 206 si sus salidas pueden dar lugar a un estado de sistema no seguro. Cuando se predice una condición no segura, el subsistema de seguridad 204 toma el control para evitar un accidente.

La correcta implementación de la arquitectura simple puede proporcionar tanto rendimiento como fiabilidad en la misma arquitectura, lo que puede ser muy útil. El rendimiento nominal se determina por el subsistema de control complejo de alto rendimiento 202, mientras que el rendimiento en el caso más desfavorable está delimitado por el subsistema de seguridad 204. El alcance de la costosa verificación y validación se centra en el subsistema de seguridad 204 y la lógica de decisión 208, que, si se diseñan correctamente, son componentes relativamente simples. Sin embargo, el logro de estos beneficios puede requerir un análisis cuidadoso del diseño y el estricto cumplimiento de los requisitos. En la arquitectura simple hay dos realizadores (subsistema de seguridad 204, subsistema complejo 202) y un comprobador (lógica de decisión 208).

La arquitectura simple tiene cierta utilidad en la planificación y ejecución fiables de trayectorias para un AGV. En un AGV, el subsistema complejo 202 podría utilizar un algoritmo robótico tradicional de planificación de rutas. El subsistema de seguridad 204 podría ser un subsistema de control de apagado seguro (por ejemplo, detener el vehículo de forma controlada). Sin embargo, el diseño sigue planteando desafíos.

Un desafío restante en cuanto al diseño incluye determinar si un "planificador de seguridad" que cumple con los requisitos del subsistema de seguridad 204 puede implementarse de manera viable. Un ejemplo para un planificador de pares líder/seguidor incluye un sistema de planificación que utiliza una biblioteca de maniobras de emergencia para garantizar que siempre haya una trayectoria segura disponible.

Otro desafío restante en cuanto al diseño incluye diseñar una lógica que pueda determinar cuándo habilitar el subsistema de seguridad 204 (por ejemplo, la lógica de decisión 208). Esto puede ser difícil porque requiere la capacidad de evaluar la seguridad de las trayectorias generadas por el subsistema complejo 202. Además, la lógica de decisión 208 debe ser un componente de alta integridad, ya que tiene la capacidad de habilitar/inhabilitar el control del vehículo por parte del subsistema de seguridad.

La lógica de decisión 208 incluye un componente de "evaluación de trayectoria" que evalúa las trayectorias producidas por el subsistema complejo 202. Si el componente determina que una trayectoria no es segura, inhibe la salida del subsistema complejo 202. El componente envuelve el subsistema complejo 202 dentro de una *arquitectura de puerta de seguridad* para hacer que falle de forma silenciosa tras dicha determinación. La viabilidad de implementar dicho componente implica si un algoritmo de evaluación de trayectoria puede implementarse de una manera que sea factible de verificar. Para ello, puede utilizarse el concepto de que *la evaluación es más sencilla que la planificación*. La planificación de rutas es un problema de búsqueda en el espacio de control del vehículo, que puede ser tan grande que se deban emplear algoritmos complejos y aleatorizados para encontrar soluciones prácticas. En comparación, evaluar la seguridad de una ruta individual a través de este espacio de control es un ejercicio relativamente simple de (i) alinear la trayectoria indicada sobre un mapa de costes binario basado en cuadrícula (que se muestra como el mapa 300 en la FIG. 3), (ii) simular el recorrido de la ruta (por ejemplo, la ruta 302) a través de celdas de mapa de costes, y (iii) si una celda no es transitable (por ejemplo, interseca las celdas oscuras 304 del mapa 300 en la FIG. 3), rechazar entonces la ruta y, de lo contrario, aceptarla.

En este análisis, se supone que el modelo del entorno contiene información suficiente para representar todos los obstáculos. Un ejemplo de una representación de modelo incluye un mapa de costes, que es una cuadrícula regular que codifica el "coste" de atravesar una unidad de espacio discreta dada delante del vehículo. Una métrica de costes simple puede ser la altura de los objetos en una celda por encima del plano terrestre nominal, lo que puede ser relevante para la navegación en carretera ya que la superficie de carretera generalmente plana hace que la altura de los objetos sea un componente intuitivo de la transitabilidad. Cuando se viaja sobre superficies muy complejas, tales como la navegación agresiva fuera de la carretera, la noción de un "plano terrestre nominal" puede volverse menos útil, y la simulación directa del movimiento del vehículo se puede utilizar para evaluar características de transitabilidad aparentemente simples, tales como la pendiente efectiva. Sin embargo, incluso en condiciones benignas en carretera, fuentes de error, tales como la calibración de sensores, pueden confundir al análisis de transitabilidad simple.

Si el subsistema complejo 202 falla de forma silenciosa (a través de la lógica de decisión 208 desconectando el subsistema complejo 202 cuando genera trayectorias no seguras), entonces los requisitos de la lógica de decisión 208

se simplifican en el sentido de que el subsistema de seguridad 204 toma el control del vehículo si no se recibe un comando del subsistema complejo 202 dentro de una ventana de tiempo especificada. Forzar que las trayectorias del planificador de seguridad terminen en un estado seguro y detenido puede garantizar que las trayectorias se generen dentro de un horizonte de planificación limitado. Al limitar el horizonte de planificación, la generación de trayectorias seguras puede ser computacionalmente factible y puede existir completamente dentro de la región de espacio de configuración *conocido* (CSPACE) local al vehículo, eliminando así la posibilidad de encontrar un obstáculo más allá del alcance de sensor mientras se ejecutan los controles de trayectoria de emergencia. Terminar una trayectoria de planificador de seguridad con un estado detenido puede ser necesario para evitar circunstancias en las que *después* de ejecutar una maniobra de seguridad el vehículo termina en un *estado de colisión inevitable*, un estado en el que sin importar qué acción de control se tome en el futuro, se producirá una colisión.

Por ejemplo, supóngase que se genera una maniobra de seguridad para girar pasado un árbol, por lo que al final de la maniobra el vehículo ha regresado a su velocidad original. Aunque la maniobra de seguridad puede haber evitado con éxito el árbol, el estado final de la maniobra puede hacer que el vehículo no pueda detenerse a tiempo para evitar una roca que estaba fuera del radio de sensor local del vehículo cuando comenzó la maniobra. Una trayectoria que no termina a velocidad cero puede provocar que el vehículo colisione con la roca no detectada previamente.

En la arquitectura simple, la lógica de decisión 208 tiene al menos un componente operativo ante fallos porque el subsistema complejo 202 proporciona un comportamiento optimizado, pero no es fiable. La lógica de decisión 208 está lista en cualquier momento para conmutar el funcionamiento al subsistema de seguridad 204. Debido a que la lógica de decisión 208 puede no detectar si el subsistema de seguridad 204 no es seguro, el subsistema de seguridad 204 (que es un "realizador" y no un "comprobador") funciona en caso de fallos con alta integridad, y la lógica de decisión 208 también es de alta integridad. La lógica de decisión 208 puede funcionar ante fallos o ser a prueba de fallos con un comportamiento "seguro" que da como resultado una conmutación al subsistema de seguridad 204. Por el contrario, la arquitectura descrita en esta divulgación no requiere que ninguno de los realizadores funcione ante fallos, ni requiere que ninguno de los realizadores sea de alta integridad.

Descripción general de la arquitectura

La arquitectura de seguridad descrita en esta divulgación incluye un patrón de diseño reutilizable que puede servir como base para el control seguro de cualquier sistema robótico u otro sistema autónomo o semiautónomo, así como de cualquier sistema que deba construirse como una composición de componentes de alta fiabilidad y baja fiabilidad. Ese patrón proporciona, entre otras cosas, un comportamiento a nivel de sistema con funcionamiento ante fallos sin requerir ningún bloque de componente operativo ante fallos y sin requerir ningún bloque de autonomía realizador de alta integridad. Al definir las relaciones entre clases de objetos en un sistema de control, se puede entender el problema general de la navegación segura de vehículos y los requisitos de información y dependencias necesarias para implementar estos conceptos en el contexto de un sistema de autonomía general más amplio. Esta estructura formalizada no sólo facilita un mayor entendimiento del problema en cuestión, sino que también proporciona beneficios tales como la comunicación inequívoca de los requisitos y una mejor mantenibilidad a través de la modularización.

La FIG. 4 es un diagrama de bloques que muestra un ejemplo generalizado de una versión de dos canales de la arquitectura de autonomía segura 400. La parte izquierda de la arquitectura 400 incluye una unidad primaria 402, una puerta de seguridad primaria 404, una unidad de protección 406 y una puerta de seguridad y protección y memoria intermedia 408. La parte izquierda de la arquitectura 400 puede repetirse una o más veces en cascada. La parte derecha de la arquitectura 400 incluye un selector de prioridad opcional 410 para el enfoque de resolución de accionamiento final.

La arquitectura 400 incluye un canal "primario" y un canal "de protección" que están encadenados entre sí a través de capas de un sistema, hasta que en algún momento se proporciona una única salida de comando, tal como un comando de control de motor. El selector de prioridad 410 arbitra entre los canales. Si las salidas están disponibles desde el canal primario y el canal de protección, el selector de prioridad 410 transmite la salida de canal primario. Si solo hay disponible una salida de canal de protección, el selector de prioridad 410 transmite la salida de canal de protección. En cualquier otro caso, el selector de prioridad 410 transmite un comando de parada de movimiento ("MSTOP"), que es un medio de respaldo de bajo nivel para detener el vehículo, tal como activar los frenos e interrumpir el funcionamiento del acelerador, o desplegar un paracaídas para una aeronave.

Un beneficio de la arquitectura 400 se deriva del hecho de que las unidades 402, 406 utilizadas para *generar* salidas en *ambos* canales (es decir, los realizadores) pueden tener bajos niveles de integridad y, de hecho, cada una puede fallar arbitrariamente. Si se produce un fallo no tratado en la unidad primaria 402 o en la unidad de protección 406, una arquitectura debidamente instanciada permanecerá operativa y seguirá cumpliendo los requisitos de seguridad. Si fallan tanto la unidad primaria como la secundaria, el sistema seguirá siendo seguro, pero las fases posteriores tendrán la tarea de llevar a cabo una recuperación del sistema (por ejemplo, ejecutando un comando MSTOP). Esto evita la necesidad de certificar o garantizar completamente la seguridad de las unidades realizadoras. En otras palabras, ni la unidad primaria 402 ni la unidad de protección 406 son cruciales para la seguridad, porque la seguridad está garantizada por los comprobadores correspondientes. Sin embargo, es importante tener en cuenta que si la unidad primaria o la de protección no son fiables, el vehículo podría sufrir problemas de *disponibilidad*; la

arquitectura 400 detendrá o reducirá el rendimiento del vehículo conmutando al canal de protección con más frecuencia de la deseable.

Los dos componentes de "puerta de seguridad" 404, 408 en la arquitectura 400 se encargan de comprobar las salidas de la unidad primaria y de la unidad de protección 402, 406 y de fallar de forma silenciosa si estas salidas no son seguras. Estos son componentes de alta integridad, pero pueden ser silenciosos antes fallos. La instanciación de puertas de seguridad para una aplicación en particular puede requerir un trabajo cuidadoso de diseño; sin embargo, en la mayoría de los casos previstos, esto requerirá muchos menos recursos (particularmente en cuanto a la verificación y validación) que el desarrollo de la unidad primaria y la unidad de protección 402, 406 a un alto nivel de rigor e integridad. Por lo tanto, este enfoque relaja los requisitos de integridad de los realizadores y permite el uso de comprobadores de detención ante fallos al tiempo que proporciona una arquitectura general de funcionamiento ante fallos.

La arquitectura 400 impone requisitos más estrictos al selector de prioridad 410. El selector de prioridad 410 debe seguir funcionando en presencia de fallos para suministrar un comando primario o de protección. El selector de prioridad 410 puede ser silencioso ante fallos siempre y cuando ese fallo desencadene un comando MSTOP. Este componente es más sencillo que las puertas de seguridad, y puede dedicarse un gran esfuerzo a su verificación para lograr el alto nivel de integridad requerido.

En algunas implementaciones, la arquitectura 400 se activa en el tiempo. En una arquitectura activada en el tiempo, los fallos se detectan a través de tiempos de espera. No se permite que los componentes posteriores utilicen valores "obsoletos" pasados algunos múltiplos del período de mensaje (para que sean robustos ante fallos transitorios). La excepción es la puerta de seguridad y protección y memoria intermedia 408, que puede almacenar una planificación de protección, pero vuelve a comprobar la planificación de protección almacenada para ver que es aceptable en cada etapa de tiempo. También es posible un enfoque activado por evento, que incluye, pero sin limitarse a, un enfoque activado por evento que emula un enfoque activado en el tiempo mediante una generación periódica de eventos.

Un tercer canal opcional de la arquitectura 400, denominado *superposición*, que incluye la unidad de superposición 412, permite incorporar otros equipos en la arquitectura de seguridad 400. Esto puede incluir componentes temporales de "verdad-terreno" que, por ejemplo, notifican la posición del personal en un emplazamiento de prueba para que la arquitectura 400 pueda detener el vehículo si se acercan demasiado. Estos componentes pueden incluir sensores adecuados tales como un sensor de radiobaliza. También pueden incluir otros equipos (tal vez temporales) que transmiten comandos MSTOP de forma inalámbrica.

La FIG. 5 es diagrama de bloques que muestra un ejemplo de una instanciación de sistema AGV 500 de la arquitectura de autonomía segura. La arquitectura ha sido instanciada para dos fases de control: la fase de planificación 504 y la fase de ejecución de trayectoria 506. Cada una de estas dos fases incluye un canal primario y un canal de protección. En la fase de control de vehículo 508, un selector de prioridad 530 se encarga de conmutar entre estos canales. Opcionalmente, la arquitectura puede incluir un canal de superposición que permite incorporar otros equipos, tales como superposición de planificador 534 y superposición de trayectoria 536, en la arquitectura. La arquitectura también puede incluir conceptos de fiabilidad para la fase de percepción 502, que genera los mapas sobre los que funciona la fase de planificación 504. En algunas implementaciones relevantes para AGV, la comunicación entre los componentes se logra usando el sistema operativo de robot (ROS), como se describe en el documento de Quigley, Morgan, et al., "ROS: an open source Robot Operating System", taller de ICRA sobre software de código abierto, Vol. 3, n.º 3.2, 2009, cuyo contenido completo se incorpora en el presente documento como referencia. Se pueden utilizar otras redes de comunicación, incluidas redes de área de controlador (ISO 11898), Ethernet activada por tiempo (SAE AS6802) o FlexRay (ISO 17458), para integrar la arquitectura con características específicas de aplicación.

Planificador primario

El planificador primario 512 planifica la trayectoria primaria que sigue el vehículo terrestre en condiciones normales de funcionamiento. Cada trayectoria producida está libre de colisiones y dentro de los límites cinemáticos del vehículo. Esta representación es extensible a varias limitaciones realistas en la dirección, aceleración, frenado y curvatura para que el marco presentado pueda evaluarse en el futuro con una complejidad dinámica adicional. Algunas implementaciones relevantes para AGV de esta arquitectura utilizan la biblioteca abierta de planificación de movimiento (OMPL) que utiliza el sistema operativo de robot (ROS). OMPL se describe en el documento de Şucan et al., "The Open Motion Planning Library" (PDF), revista de robótica y automatización del IEEE (diciembre de 2011), cuyo contenido completo se incorpora en el presente documento como referencia. En la OMPL se utiliza RRT* como algoritmo de planificación. RRT* se describe en el documento de LaValle, Steven M., "Rapidly-exploring random trees: A new tool for path planning,", Informe Técnico (Departamento de Informática, Universidad del Estado de Iowa) (TR 98-11) (octubre de 1998), cuyo contenido completo se incorpora en la presente documento como referencia. El planificador primario 512 recibe una cuadrícula de ocupación desde la fase de percepción 502 y planifica rutas a través de este mapa de cuadrícula. La planificación de movimiento OMPL también puede utilizarse en aplicaciones de vehículos aéreos no tripulados.

Tratamiento de fallos arbitrarios en el planificador primario

La salida del planificador primario 512, que en algunas implementaciones relevantes para AGV es una trayectoria que consiste en una secuencia de puntos de recorrido, es comprobada por la puerta de seguridad de planificador primario (PPSG) 514. El PPSG 514 comprueba si la salida del planificador primario 512 es válida utilizando una comprobación específica de aplicación, y comprueba además que la salida esté dentro de la envolvente permisiva (PE) proporcionada por la puerta de seguridad de planificador de protección (SPSG) 518. Si la salida del planificador primario 512 falla en cualquiera de estas comprobaciones, el PPSG 514 simplemente inhibe la salida. En base a la definición de arquitectura, las fases posteriores responden inhibiendo sus salidas de canal primario y, finalmente, el selector de prioridad 530 pasa el control al canal de protección.

Planificador primario en implementaciones AGV

Mientras que los puntos de inicio y de destino en un escenario operativo se definen por una descripción de misión, algunas configuraciones de implementación relevantes para AGV definen los destinos en el espacio de estados SE2, que representa cada posible estado de vehículo de acuerdo con su ubicación (x,y) en el entorno 2D (especificado por la cuadrícula de ocupación), así como su rumbo (θ). El planificador primario 512 en las implementaciones relevantes para AGV genera una ruta cinemáticamente viable entre dos puntos de modo que la ruta pueda ser seguida de manera realista por un vehículo terrestre similar a un automóvil.

La viabilidad cinemática de cualquier ruta generada puede ser crucial, ya que la salida final del control de vehículo 532 no es una ruta o trayectoria geométrica, sino una secuencia de controles que se pueden aplicar a un vehículo terrestre concreto para hacer que siga la trayectoria solución. Para resolver este problema, el sistema 500 divide el problema en dos fases separadas: *generación de trayectoria* (crear la ruta que seguirá el vehículo) y *generación de control* (generar la secuencia de controles que harán que el vehículo siga la trayectoria generada).

Durante la fase de generación de trayectoria, las rutas creadas son capaces de hacer que un vehículo de pase de una posición inicial a otra final sin violar las *restricciones de colisión* (impuestas por la cuadrícula de ocupación) o las *restricciones cinemáticas* (impuestas por el modelo de vehículo). Al considerar estas limitaciones durante la fase de generación, se generan controles para el vehículo de manera que sea capaz de seguir la trayectoria solución. Para generar los controles, se utiliza una implementación de espacio de estados denominada "coche de Dubin", que es un modelo cinemático simple de un coche o camión que solo permite que el vehículo se mueva de las maneras posibles para un vehículo en carretera. Este modelo limita el movimiento del vehículo a solo tres formas posibles: girar en arco a la derecha, girar en arco a la izquierda e ir recto. Al crear trayectorias utilizando secuencias de estas primitivas de movimiento, se generan controles para seguir la trayectoria durante la fase de *generación de control* dada una trayectoria que satisfaga estas limitaciones (suponiendo que el radio de giro coincide con el vehículo físico). Al instanciar la OMPL con el comprobador de validez basado en cuadrícula de ocupación y la restricción de movimiento de Dubin, se pueden generar trayectorias que sean viables y libres de colisiones.

La FIG. 6 es un diagrama que muestra un ejemplo de una cuadrícula de ocupación 600 para la planificación de movimiento con un modelo cinético de un coche de Dubin usando OMPL. En la FIG. 6, un vehículo 602 trata de llegar a un estado objetivo deseado 604 desde su ubicación actual 606. El vehículo 602 debe lograr el estado objetivo deseado 604 al tiempo que evita un obstáculo 608. La presencia del obstáculo 608 provoca que las casillas pertinentes de la cuadrícula de ocupación 600 se marquen como "ocupadas", evitando así que el vehículo 602 se desplace a través de estas casillas. Como se muestra en la FIG. 6, el vehículo 602 gira a la izquierda en el obstáculo 608 en lugar de a la derecha, aunque la ubicación objetivo 604 está más cerca del lado derecho del obstáculo 608. Si el vehículo 602 girara a la derecha y luego a la izquierda, estaría circulando en una dirección incorrecta y, por lo tanto, no satisfaría el rumbo deseado descrito por el objetivo. Dado que dar la vuelta después de girar a la derecha en el obstáculo 608 es más costoso que girar a la izquierda inicialmente, el vehículo 602 planifica la trayectoria 610 hacia la izquierda. La trayectoria solución 610 no puede interrumpirse cerca de las casillas ocupadas por el obstáculo 608 debido a que las limitaciones de radio de giro en el vehículo 602 evitan la generación de arcos que son demasiado estrechos. Dado que este radio de giro es un parámetro configurable, el procedimiento de generación de trayectorias subyacente puede probarse con diferentes tipos de vehículos. Como se analizó anteriormente, la biblioteca de planificación OMPL se puede utilizar para generar planificaciones de movimiento. La OMPL incluye una representación del espacio de estados de Dubin que se puede utilizar para inicializar la clase planificador OMPL junto con un "comprobador de validez" personalizado.

Planificador de protección

Con referencia nuevamente a la FIG. 5, dado el entorno existente y el estado del vehículo, el planificador de protección 516 en implementaciones relevantes para AGV crea trayectorias viables diseñadas para permitir que el vehículo se detenga de forma rápida y segura cuando se producen problemas. El planificador de protección 516 proporciona una opción de emergencia para el vehículo y reevalúa continuamente las planificaciones a medida que el vehículo se desplaza por el entorno y se encuentra con un obstáculo dinámico o estático.

Tratamiento de fallos arbitrarios en el planificador de protección

Aunque los requisitos anteriores son deseables para el planificador de protección 516, el sistema no *depende* del planificador de protección 516 para garantizar la seguridad del vehículo. El planificador de protección 516 está marcado como un bloque "arbitrario ante fallos", al igual que el planificador primario 512. El sistema logra esto con las siguientes características:

1. Las salidas del planificador de protección 516 son evaluadas por la puerta de seguridad de planificador de protección 518, que, por ejemplo, inhibe su salida si la planificación de protección choca con un obstáculo.
2. El planificador de protección 516 también produce una envolvente permisiva (PE) contra la cual se comprueban las salidas del planificador primario 512 (lo que se describe con más detalle posteriormente). La puerta de seguridad de planificador de protección 514 también evalúa si esta envolvente es apropiada en sí misma.
3. Después de que el planificador primario 512 falle, el planificador de protección 516 toma el control; sin embargo, la puerta de seguridad de planificador de protección 518 proporciona al planificador de protección 516 una ventana de tiempo limitada en la que detener el vehículo. Después de que expire esa ventana de tiempo, la puerta de seguridad de planificador de protección 518 inhibe sus salidas. En respuesta a una falta de salidas del canal primario o de protección, el selector de prioridad 530 activa un comando MSTOP.

Planificador de protección en implementaciones AGV

Tal como se mencionó anteriormente con respecto al planificador primario 512, puede haber algunas circunstancias o condiciones adversas en las que un vehículo terrestre puede no ser capaz de lograr su ruta planificada deseada. Esto podría deberse a una especificación de misión no válida, un fallo de hardware que hace que la ruta planificada no pueda lograrse, o a otras diversas circunstancias dinámicas, tales como un obstáculo, un retraso debido al tráfico y un retraso en la construcción no considerado en la planificación original. Para proporcionar un funcionamiento seguro del vehículo durante estos eventos, el sistema incluye un componente de planificación de ruta aparte (por ejemplo, el planificador de protección 516) para generar rutas que dirigen el vehículo desde su estado actual hasta un estado objetivo seguro en el que el vehículo se detiene. Al forzar que el vehículo se detenga, el vehículo fallido permanece en su estado objetivo de forma segura durante un período de tiempo indefinido. Una operación análoga para una aeronave sería una desviación hacia la zona de aterrizaje más cercana. Dado que el objetivo del planificador de protección 516 es llevar a cabo una detención de forma segura, el planificador de protección 516 considera múltiples configuraciones objetivo diferentes para encontrar una ruta de detención segura. El Planificador de protección 516 encuentra una ruta de detención segura de la siguiente manera:

1. Planificar una trayectoria para aplicar el máximo frenado, deteniendo el vehículo a un lado de la carretera.
2. Si el vehículo no puede detenerse a un lado de la carretera, intentar llegar un estado objetivo seguro con la cantidad mínima de desviación desde la trayectoria original.
3. Si el vehículo sigue sin poder lograr una trayectoria segura, aumentar la cantidad permisible de desviación de rumbo e intentar una replanificación.
4. Repetir hasta que se alcance el máximo ángulo de dirección o se haya generado una trayectoria segura.

Dado que el planificador de protección 516 siempre debe estar preparado para un posible fallo, cada vez que se actualiza la información de mapa o de estado de vehículo, el planificador de protección 516 inicia una nueva búsqueda de una trayectoria de detención segura. En algunas implementaciones AGV, el planificador de protección 516 lleva a cabo esta búsqueda en un conjunto fijo de trayectorias, lo que se logra de nuevo con el sistema operativo de robot (ROS). El planificador de protección 516 recibe una cuadrícula de ocupación de la fase de percepción 502 y planifica rutas a través de este mapa de cuadrícula. Si el planificador de protección 516 no puede generar una nueva planificación segura, indica al sistema que inicie la última trayectoria segura generada ya que viajar sin una opción de emergencia es intrínsecamente no seguro.

La FIG. 7 es un diagrama que muestra un ejemplo de una cuadrícula de ocupación 700 para la planificación de movimiento como reacción a un árbol que cae. Supóngase que el obstáculo 608 en la FIG. 6 es un árbol grande. Después de crear la trayectoria 610 como se muestra en la FIG. 6, el árbol cae en la ruta prevista. Este cambio actualiza el estado del mapa como se indica en la cuadrícula 700 de la FIG. 7, y por lo tanto, hace que el planificador de protección 516 (en la FIG. 5) genere tres trayectorias de emergencia (t0, t1 y t2). Dado que la trayectoria de detención segura t0 no colisiona con el obstáculo 708, se selecciona la trayectoria t0 y se aplica inmediatamente. Sin embargo, si el vehículo 702 es demasiado grande para detenerse en el tiempo calculado para la trayectoria t0, la trayectoria t2 queda invalidada por el obstáculo 708 y, por lo tanto, no se considera. En esta situación, el vehículo 702 selecciona, en cambio, la trayectoria alternativa segura t1. Debido a que el vehículo 702 ya estaba comenzando su giro hacia la izquierda, puede ser capaz de efectuar un giro más pronunciado lejos del obstáculo 708 a la izquierda en lugar de a la derecha. Por lo tanto, la consideración del estado del vehículo terrestre 702 puede ser crucial en la generación de estas rutas de protección.

Envolturas permisivas

Con referencia nuevamente a la FIG. 5, se utiliza una *envoltura permisiva* PE1 para confirmar que la salida del planificador de protección 516 aún se podrá lograr si el vehículo comienza a ejecutar las salidas del planificador primario 512. Esto es importante ya que si se produjera un fallo, el sistema 500 pasaría de la salida principal a la salida

de protección. La PE1 es generada por el planificador de protección 516 junto con el plan de protección. La seguridad de PE1 es comprobada por la puerta de seguridad de planificador de protección 518. Si la PE1 pasa esta comprobación, se pasa a la puerta de seguridad de planificador primario 514. La puerta de seguridad de planificador primario 514 utiliza la PE1 como parte de los criterios para aceptar o rechazar la salida del planificador primario 512. Las envolturas permisivas, tanto en general como en el contexto específico de un ejemplo de una fase de ejecución de trayectoria AGV, se describen adicionalmente más adelante.

Puerta de seguridad primaria

Con referencia a la FIG. 4, el propósito de la puerta de seguridad primaria (PSG) 404 es inhibir las entradas no seguras de la unidad primaria 402. Como se describe más adelante, la salida de la unidad primaria 402 se comprueba para verificar la seguridad inherente (por ejemplo, en el contexto de la fase de planificación AGV 504 de la FIG. 5, si la planificación choca con obstáculos o viola las restricciones dinámicas), y, posteriormente, también se comprueba su compatibilidad con la envoltura permisiva generada por la unidad de protección 406. Si el PSG 404 inhibe las salidas de la unidad primaria 402, el selector de prioridad activado en el tiempo 410 pasa a enviar salidas desde la unidad de protección 406 a los accionadores de vehículo.

Con referencia a la FIG. 5, en el contexto de la fase de planificación AGV 504, la puerta de seguridad de planificador primario 514 se implementa como un nodo en el sistema operativo de robot, que recibe una cuadrícula de ocupación de la fase de percepción 502, así como la salida del planificador primario 512 y una envoltura permisiva. La puerta de seguridad de planificador primario 514 itera a través de los puntos de recorrido especificados en la salida del planificador primario 512 y determina si la salida haría que el vehículo colisione con cualquier obstáculo en la cuadrícula, o si no se ajusta a la envolvente permisiva.

Puerta de seguridad y protección

Con referencia a la FIG. 4, la puerta de seguridad y protección (SSG) 408 lleva a cabo comprobaciones similares en las salidas de la unidad de protección 406, a la que se le permite fallar arbitrariamente. Si la salida de la unidad de protección 406 choca con un obstáculo o no se ajusta a las restricciones dinámicas, entonces esa salida se inhibe. La SSG 408 mantiene una memoria intermedia que contiene la última salida de la unidad de protección 406 que pasa estas comprobaciones. Cuando una planificación entrante es segura, la SSG 408 escribe la planificación entrante en la memoria intermedia. Pero si la planificación entrante no es segura, la SSG 408 descarta la planificación entrante y sigue ejecutando la planificación almacenada en memoria intermedia. La SSG 408 inhibe sus salidas dentro de una ventana de tiempo, a menos que se reciba una nueva planificación segura desde la unidad de protección 406. Si la SSG 408 inhibe sus salidas, el selector de prioridad activado en el tiempo 410 activa un comando MSTOP.

Con referencia a la FIG. 5 en el contexto de la fase de planificación AGV 504, la puerta de seguridad de planificador de protección 518 se implementa como un nodo en el sistema operativo de robot, que recibe una cuadrícula de ocupación desde la fase de percepción 502, así como la salida del planificador de protección 516. La puerta de seguridad de planificador de protección 518 itera a través de los puntos de recorrido especificados en la salida del planificador de protección 516 y determina si la salida haría que el vehículo choque con cualquier obstáculo en la cuadrícula.

Selector de prioridad

Con referencia a la FIG. 4, un selector de prioridad 410 elige entre la salida primaria y la salida de protección, típicamente en la fase final de la arquitectura. Por ejemplo, el selector de prioridad 410 puede decidir qué comandos de trayectoria enviar a los accionadores de vehículo. La lógica del selector de prioridad 410 es tal que si se inhibe la salida primaria, entonces se transmite la salida de protección. Si se inhibe la salida de protección, se activa un comando MSTOP. El selector de prioridad 410 supone que las puertas de seguridad 404, 408 fallan silenciosamente.

Con referencia a la FIG. 5 en el contexto de implementaciones de sistema AGV, el selector de prioridad 530 se instancia en la salida de la fase de ejecución de trayectoria 506. El selector de prioridad 530 se implementa como un nodo en el sistema operativo de robot (ROS) y recibe las salidas tanto de la puerta de seguridad de ejecutor de trayectoria primaria 524 como de la puerta de seguridad de ejecutor de trayectoria de protección 528 dentro de la fase de ejecución de trayectoria 506. Las salidas del selector de prioridad 530 incluyen comandos de velocidad y curvatura del vehículo, que se convierten por otro nodo ROS, conocido como el controlador de vehículo 532, en comandos para los accionadores de frenado, dirección y aceleración del vehículo.

Fase de ejecución de trayectoria

Un *punto de recorrido* P_i es una posición bidimensional con un rumbo, es decir, $\{x_i, y_i, theta_i\}$. Una *trayectoria* $\{V_i, C_i\}$ es un par de velocidad-curvatura con una duración implícita basada en el período activado en el tiempo. El estado de sistema actual (por ejemplo, posición del vehículo, etc.) es S_i. En la fase de ejecución de trayectoria 506, la envolvente permisiva PE2 es un límite de intervalo para la aceleración y guiñada con la trayectoria de protección. PE2 se

proporciona como una lista de aceleraciones mínimas y máximas y velocidades de guiñada {ΔV^{min}_i, ΔV^{max}_i, ΔC^{min}_i, ΔC^{max}_i}.

El ejecutor de trayectoria primaria 522 y el ejecutor de trayectoria de protección 526 son los nodos que ejecutan los algoritmos de trayectoria primaria (Algoritmo 1) pta y los algoritmos de trayectoria de protección (Algoritmo 2) sta, respectivamente. Ambos toman un punto de recorrido P_i y el estado del vehículo actual S_i y proporcionan una trayectoria desde la posición actual del punto de recorrido.

Algoritmo 1: Ejecutor de trayectoria primaria

Entrada: Un punto de recorrido $PW_i : (x_i, y_i, \theta_i)$
El estado de sistema S_i
Salida: Una trayectoria $\{V_i, C_i\}$

$\{V_i, C_i\} \leftarrow \mathbf{pta}(PW_i, S_i)$;

Algoritmo 2: Ejecutor de trayectoria de protección

Entrada: Un punto de recorrido $SW_i : (x_i, y_i, \theta_i)$
El estado de sistema S_i
Salida: Una trayectoria $\{V_i^s, C_i^s\}$
La envolvente de protección $\{\Delta V_i^{min}, \Delta V_i^{max}, \Delta C_i^{min}, \Delta C_i^{max}\}$

$\{V_i^s, C_i^s\} \leftarrow \mathbf{sta}(SW_i, S_i)$;
$\Delta V_i^{min} \leftarrow V_0^s + \Delta V_r^{min}$;
$\Delta V_i^{max} \leftarrow V_0^s + \Delta V_r^{max}$;
$\Delta C_i^{min} \leftarrow C_0^s + \Delta C_r^{min}$;
$\Delta C_i^{max} \leftarrow C_0^s + \Delta C_r^{max}$;

La puerta de seguridad de ejecutor de trayectoria primaria 524 ejecuta el Algoritmo 3, que comprueba la trayectoria primaria contra el estado actual y la PE2 creada por la puerta de seguridad de ejecutor de trayectoria de protección 528. Si la trayectoria es coherente con el estado actual y está dentro de los límites de la PE2, la trayectoria primaria se transfiere a través de la puerta de seguridad de ejecutor de trayectoria primaria 524.

Algoritmo 3: Puerta de seguridad primaria

Entrada: La trayectoria primaria $\{V_i, C_i\}$
El estado de sistema S_i
La envolvente de protección $\{\Delta V_i^{min}, \Delta V_i^{max}, \Delta C_i^{min}, \Delta C_i^{max}\}$
Salida: La trayectoria primaria $\{V_i, C_i\}$

pte_ok $\leftarrow$ **psg_check**(V_i, C_i, S_i) ;
if pte_ok *and* $\Delta V_i^{min} \leq V_i \leq \Delta V_i^{max}$ *and* $\Delta C_i^{min} \leq C_i \leq \Delta C_i^{max}$ **then**
  pass $\{V_i, C_i\}$
end

La puerta de seguridad de ejecutor de trayectoria de protección 528 es un nodo que ejecuta el Algoritmo 4 para comprobar la trayectoria del ejecutor de trayectoria de protección 526 y transfiere la nueva trayectoria de protección entrante o una anterior trayectoria de protección almacenada en memoria intermedia. La puerta de seguridad de ejecutor de trayectoria de protección 528 también crea la PE2 que la puerta de seguridad de ejecutor de trayectoria primaria 524 utiliza para garantizar que la trayectoria primaria sea coherente con la trayectoria de seguridad actual. La PE2 se genera mediante la creación de una "envolvente" de aceleraciones permitidas y cambios de curvatura en función de los valores de trayectoria de protección actuales y un conjunto de límites de recuperabilidad constantes {ΔV^{min}_r, ΔV^{max}_r, ΔC^{min}_r, ΔC^{max}_r}.

Algoritmo 4: Puerta de seguridad y protección

Entrada: La trayectoria de protección $\{V_i^s, C_i^s\}$
El estado de sistema S_i
La envolvente de protección $\{\Delta V_i^{min}, \Delta V_i^{max}, \Delta C_i^{min}, \Delta C_i^{max}\}$
Salida: La trayectoria de protección $\{V_i^s, C_i^s\}$
La envolvente de protección $\{\Delta V_i^{min}, \Delta V_i^{max}, \Delta C_i^{min}, \Delta C_i^{max}\}$

```
if ssg_check(V_i^s, C_i^s, S_i) then
    V_b^s ← V_i^s;
    C_b^s ← C_i^s ;
    E_b ← {ΔV_i^min, ΔV_i^max, ΔC_i^min, ΔC_i^max} ;
end
if ssg_check(V_b^s, C_b^s, S_i) then
    pass {V_b^s, C_b^s} ;
    pass E_b ;
end
```

Algoritmos generalizados

Los algoritmos 1 a 4 se expresan a continuación de manera generalizada en lugar de en el contexto de la ejecución de trayectorias. El algoritmo primario (Algoritmo 5) produce alguna salida O_i, cuyos detalles son específicos de cada capa. No se imponen requisitos en la exactitud del algoritmo primario; puede fallar de manera arbitraria.

Algoritmo 5: Algoritmo Primario

Entrada: Una entrada I_i
El estado de sistema S_i
Salida: Una salida O_i

```
O_i ← primAlg(I_i, S_i) ;
```

El algoritmo de protección (Algoritmo 6) se encarga de su propia salida O^{S_i} junto con una envolvente permisiva E_i.

Algoritmo 6: Algoritmo de protección

Entrada: Una entrada I_i^s
El estado de sistema $\mathcal{S}_i$
Salida: Una salida O_i^s ;
La envolvente de protección $\mathcal{E}_i$

```
O_i^s ← safeAlg(I_i^s, S_i) ;
E_i ← genEnv(O_i^s)
```

La puerta de seguridad primaria (Algoritmo 7) se encarga de dos tipos de comprobaciones. En primer lugar, ejecuta una función específica de la aplicación, psg_check, que garantiza que la salida del algoritmo primario es aceptable dado el estado actual de sistema S_i. La PSG también se encarga de garantizar que la salida del algoritmo primario esté dentro de la envolvente permisiva.

Algoritmo 7: Puerta de seguridad primaria

Entrada: La salida primaria O_i
El estado de sistema $\mathcal{S}_i$
La envolvente de protección $\mathcal{E}_i$
Salida: La salida primaria O_i

```
pte_ok ← psg_check(O_i, S_i) ;
if pte_ok and O_i ∈ E_i then
    pass O_i
end
```

La puerta de seguridad de protección se muestra en el Algoritmo 8, que define la lógica de almacenamiento en memoria intermedia descrita anteriormente en la sección "Puerta de seguridad y protección". Cabe señalar que la

función ssg_cheek(...) también comprueba la antigüedad de la trayectoria de protección y de la envoltura permisiva; si cualquiera de las mismas es anterior a un período de tiempo de espera especificado, entonces ssg_check(...) devuelve *false*.

Algoritmo 8: Puerta de seguridad y protección

Entrada: La trayectoria de protección O_i^s
La envolvente de protección $\mathcal{E}_i$
El estado de sistema S_i

Salida: La trayectoria de protección O_i^s
La envolvente de protección $\mathcal{E}_i$

```
if ssg_check(O_i^s, E_i, S_i) then
    O_s^s ← O_i^s;
    E_s ← E_i ;
end
if ssg_check(O_s^s, E_s, S_i) then
    pass O_s^s ;
    send E_s ;
end
```

El caso de seguridad de una implementación AGV se basa en puertas de seguridad que evalúan trayectorias de vehículo para detectar colisiones en un mapa de obstáculos. Dado un mapa y una trayectoria, una puerta de seguridad informa si la trayectoria dada es segura o no segura en el mapa dado. La puerta de seguridad comprueba una trayectoria de dimensionalidad moderadamente alta en un mapa de dimensionalidad moderadamente alta utilizando dinámicas continuas y trascendentales. Una trayectoria se convierte en una ruta (serie de posiciones) en función de un conjunto de ecuaciones dinámicas, y estas posiciones se comprueban en el mapa para decidir si la trayectoria es segura o no. La definición de una trayectoria segura para los propósitos de la presente divulgación es una que no se interseca con un obstáculo en el mapa usando ecuaciones cinemáticas. Una puerta de seguridad calcula posiciones con cierta precisión y resolución basándose en las trayectorias e informa de un problema de seguridad cuando una de estas posiciones se superpone a un obstáculo.

El sistema descrito en esta divulgación incluye un modelo híbrido de toda la arquitectura que evita colisiones. El sistema híbrido sigue las trayectorias primarias o una trayectoria de protección del proveedor, y proporciona una ejecución segura. Las partes de "comprobador" de puerta de seguridad de la arquitectura se pueden crear como componentes de alta integridad siguiendo normas de seguridad de software conocidas en la técnica, tales como ISO 26262, IEC 61508, MIL-STD 882E y otras normas relevantes. Se puede utilizar un mecanismo de tiempo de espera para comprobar que la planificación de protección detiene el vehículo dentro del período de tiempo requerido, ya que la unidad de protección podría fallar arbitrariamente.

La arquitectura puede incluir además una o más fases particulares en la invención que necesitan retroalimentación de fases inferiores. Por ejemplo, si falla una función de ejecución de trayectoria, se informa a la fase de planificación. Además, aunque la arquitectura de seguridad descrita incluye diferentes fases apropiadas para dicha arquitectura, muchas arquitecturas robóticas segregan la "detección", el "pensamiento" y la "acción" por fases, y la arquitectura de seguridad adopta un enfoque similar.

La siguiente tabla enumera ejemplos de condiciones y comportamientos del sistema que pueden satisfacer el requisito de que el vehículo no choque con obstáculos.

	Condición	Comportamiento
1	No se procesa una colisión	El vehículo evita adecuadamente los obstáculos simulados.
	Fase de planificación	
2	Funcionamiento incorrecto del planificador primario	El vehículo comienza a ejecutar planificaciones de protección, deteniéndose a un lado de la carretera.
3	Funcionamiento incorrecto del planificador de protección	El vehículo ejecuta el último plan de protección adecuado recibido por la puerta de seguridad de planificador de protección, deteniéndose a un lado de la carretera.
4	La puerta de seguridad de planificador primario (PPSG) se bloquea	Mismo comportamiento que la condición 2.

5	La puerta de seguridad de planificador de protección (SPSG) se bloquea	Desactivar la SPSG inhibe tanto la planificación primaria como la de protección. La fase de trayectoria no obtiene datos de entrada, y por lo tanto, no envía salidas, lo que hace que la fase de ejecución de trayectoria ejecute una trayectoria de protección.
	Fase de ejecución de trayectoria	
6	Funcionamiento incorrecto del ejecutor de trayectoria primaria	El vehículo comienza a ejecutar planes del planificador de protección. En esta implementación de ejemplo, el planificador de protección no es notificado del fallo en la fase de trayectoria, por lo que continúa enviando planificaciones. Al final de un período de tiempo de espera, la puerta de seguridad de trayectoria de protección deja de transmitir comandos. En este punto, ni el canal primario ni el de protección en la fase de trayectoria transmiten comandos. En respuesta, el selector de prioridad ejecuta un comando MSTOP para el vehículo. Si bien esto cumple los requisitos de seguridad, la ejecución del comando MSTOP no hubiera sido necesaria con una forma de realización alternativa en la que se notifica al planificador de protección acerca de un mal funcionamiento del ejecutor de trayectoria primaria para que el canal de protección pueda detener el vehículo de forma controlada.
7	Funcionamiento incorrecto del ejecutor de trayectoria de protección	La capa de trayectoria no se ocupa de los comandos de larga duración, sino que calcula comandos de accionador de vehículo basándose en la planificación más larga. Por lo tanto, cuando el ejecutor de trayectoria de protección falla, la puerta de seguridad de trayectoria de protección ejecuta la última trayectoria de protección válida en la memoria intermedia. Si no hay disponible ninguna trayectoria de protección válida, se agota el tiempo de espera de la puerta de seguridad de trayectoria de protección, lo que provoca que tanto el canal primario como el canal de protección estén en silencio. A su vez, esto hace que el selector de prioridad ejecute un comando MSTOP.
8	La puerta de seguridad de trayectoria primaria (PTSG) se bloquea	Cuando la PTSG se bloquea, la STSG también inhibe las salidas. En respuesta, el selector de prioridad ejecuta un comando MSTOP para el vehículo. Si bien esto cumple los requisitos de seguridad, el comando MSTOP no es necesario porque el canal de protección sigue siendo válido. La STSG hace que el selector de prioridad ejecute un comando MSTOP para el vehículo después de algún periodo de tiempo de espera, durante el cual los canales de protección en las fases de planificación y trayectoria detienen el vehículo.
9	La puerta de seguridad de trayectoria de protección (STSG) se bloquea	Al bloquearse la STSG se inhiben ambas salidas, lo que provoca que el selector de prioridad ejecute un comando MSTOP.

La FIG. 8 es un diagrama de flujo de un ejemplo de un proceso 800 llevado a cabo por un sistema de arquitectura de seguridad de dispositivo autónomo. El proceso 800 puede llevarse a cabo mediante un sistema de uno o más ordenadores. El proceso 800 puede incluir detalles que se han analizado anteriormente.

El sistema recibe datos primarios para mover un dispositivo en una ruta planificada (802). El sistema puede incluir una puerta de seguridad primaria que recibe los datos primarios de una unidad primaria. El sistema también recibe datos secundarios para mover el dispositivo en presencia de una o más condiciones adversas durante el movimiento del dispositivo en la ruta planificada (804). El sistema puede incluir una puerta de seguridad secundaria que recibe los datos secundarios de una unidad secundaria.

El sistema valida los datos primarios (806) determinando si los datos primarios proporcionan el movimiento del dispositivo en la ruta planificada de manera segura. La puerta de seguridad primaria puede ejecutar un algoritmo que toma esta determinación. El sistema también valida los datos secundarios (808) determinando si los datos secundarios proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas. La puerta de seguridad secundaria puede ejecutar un algoritmo que toma esta determinación. Si el sistema determina que los datos secundarios no proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas, el sistema accede a datos previamente almacenados para el movimiento del dispositivo en presencia de la una o más condiciones

adversas (810) y valida los datos previamente almacenados determinando si los datos previamente almacenados proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas (812).

El sistema selecciona los datos primarios, los datos secundarios, los datos previamente almacenados o datos predeterminados (814) basándose en la evaluación de datos lógicos. El sistema puede incluir un selector de prioridad que evalúa los datos lógicos para llevar a cabo una selección. Los datos lógicos especifican reglas que definen qué datos se seleccionan en condiciones específicas. Los datos primarios se seleccionan después de determinar que los datos primarios proporcionan el movimiento del dispositivo en la ruta planificada de manera segura. Los datos secundarios se seleccionan después de determinar que (i) los datos primarios no proporcionan el movimiento del dispositivo en la ruta planificada de manera segura y (ii) los datos secundarios proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas. Los datos previamente almacenados se seleccionan después de determinar que (i) los datos primarios no proporcionan el movimiento del dispositivo en la ruta planificada de manera segura, (ii) los datos secundarios no proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas, y (ii) los datos previamente almacenados proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas. Los datos predeterminados especifican una acción predeterminada que debe ser llevada a cabo por el dispositivo. Los datos predeterminados se seleccionan después de determinarse que (i) los datos primarios no proporcionan el movimiento del dispositivo en la ruta planificada de manera segura, (ii) los datos secundarios no proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas, (iii) y los datos previamente almacenados no proporcionan el movimiento del dispositivo para evitar la una o más condiciones adversas.

El sistema proporciona los datos primarios, secundarios, previamente almacenados o predeterminados seleccionados a un controlador que controla el movimiento del dispositivo (816).

El uso de esta arquitectura puede simplificar y hacer posible la implementación de un sistema de autonomía operativo ante fallos. En lugar de requerir una o más versiones de algoritmos de autonomía de alta integridad, se pueden utilizar algoritmos de autonomía de baja integridad (por ejemplo, los módulos "realizadores" no tienen que funcionar perfectamente para lograr la seguridad). Los módulos "comprobadores" de puerta de seguridad se utilizan para garantizar el comportamiento silencioso ante fallos de cada par realizador/comprobador. Las compuertas de seguridad tienen que desarrollarse con alta integridad, pero en general son más simples y minimizan o eliminan algoritmos avanzados de autonomía difíciles de validar, haciéndolos más fáciles de validar. Además, los propios comprobadores pueden ser silenciosos ante fallos. El comportamiento del sistema con funcionamiento ante fallos se consigue al tener dos (o más) conjuntos diversos de bloques funcionales de par realizador/comprobador silenciosos ante fallos en cada fase arquitectónica. Ningún componente individual debe ser operativo ante fallos, y solo los comprobadores deben ser de alta integridad.

Las formas de realización se pueden implementar en circuitos electrónicos digitales o en hardware informático, firmware, software o en combinaciones de los mismos. Un aparato se puede implementar en un producto de programa informático materializado de forma tangible o almacenado en un dispositivo de almacenamiento legible por máquina para su ejecución por un procesador programable; y las acciones de procedimiento se pueden llevar a cabo por un procesador programable que ejecuta un programa de instrucciones para llevar a cabo funciones al operar en datos de entrada y generar datos de salida. Las formas de realización descritas en el presente documento, y otras formas de realización de la invención, se pueden implementar de forma ventajosa en uno o más programas informáticos que se pueden ejecutar en un sistema programable que incluye al menos un procesador programable acoplado para recibir datos e instrucciones desde, y transmitir datos e instrucciones a, un sistema de almacenamiento de datos, al menos un dispositivo de entrada y al menos un dispositivo de salida. Cada programa informático puede implementarse en un lenguaje de programación de alto nivel orientado a objetos o procedimientos, o en lenguaje ensamblador o máquina si se desea; y en cualquier caso, el lenguaje puede ser un lenguaje compilado o interpretado.

Procesadores adecuados para la ejecución de un programa informático incluyen, a modo de ejemplo, microprocesadores de propósito general y especial, y uno o más procesadores de cualquier tipo de ordenador digital. Generalmente, un procesador recibirá instrucciones y datos de una memoria de solo lectura o una memoria de acceso aleatorio o ambas. Los elementos esenciales de un ordenador son un procesador para ejecutar instrucciones y uno o más dispositivos de memoria para almacenar instrucciones y datos. Generalmente, un ordenador también incluirá, o estará acoplado de forma operativa para recibir datos desde o transferir datos a, o ambas cosas, uno o más dispositivos de almacenamiento masivo para almacenar datos, por ejemplo, discos magnéticos, magneto-ópticos o discos ópticos. Los medios legibles por ordenador para incorporar instrucciones y datos de programas informáticos incluyen todas las formas de memoria no volátil, lo que incluye, a modo de ejemplo, dispositivos de memoria semiconductores, por ejemplo, EPROM, EEPROM y dispositivos de memoria flash; discos magnéticos, por ejemplo, discos duros internos o discos extraíbles; discos magneto-ópticos; y discos CD-ROM y DVD-ROM. El procesador y la memoria se pueden complementar por o incorporar en circuitos lógicos de propósito especial. Cualquiera de los anteriores puede complementarse por o incorporarse en ASIC (circuitos integrados específicos de la aplicación).

Para proporcionar interacción con un usuario, las formas de realización se pueden implementar en un ordenador que tiene un dispositivo de visualización, por ejemplo, un monitor LCD (pantalla de cristal líquido), para mostrar datos al usuario, y un teclado y un dispositivo de puntero, por ejemplo, un ratón o una bola de seguimiento, mediante los cuales

el usuario puede proporcionar datos de entrada al ordenador. También se pueden utilizar otros tipos de dispositivos para proporcionar interacción con un usuario; por ejemplo, la retroalimentación proporcionada al usuario puede ser cualquier forma de retroalimentación sensorial, por ejemplo, retroalimentación visual, retroalimentación auditiva o retroalimentación táctil; y los datos de entrada del usuario se pueden recibir en cualquier forma, incluidos datos de entrada acústicos, verbales o táctiles.

Otras formas de realización están dentro del alcance y el espíritu de las reivindicaciones de la descripción. Además, debido a la naturaleza del software, las funciones descritas anteriormente se pueden implementar utilizando software, hardware, firmware, cableado o combinaciones de cualquiera de estos. Las funciones de implementación de características también pueden estar ubicadas físicamente en diversas posiciones, lo que incluye estar distribuidas de manera que partes de las funciones se implementen en diferentes ubicaciones físicas. El uso del término "un/una" en el presente documento y a lo largo de la solicitud no se utiliza de manera limitativa y, por lo tanto, no pretende excluir un significado múltiple o un significado de "uno o más" para el término "un/una". Además, en la medida en que se reivindica prioridad a una solicitud de patente provisional, debe entenderse que la solicitud de patente provisional no es limitativa, sino que incluye ejemplos de cómo se pueden implementar las técnicas descritas en el presente documento.

REIVINDICACIONES

1. Un sistema de arquitectura de seguridad (400, 500) para vehículos autónomos, que comprende:

una primera fase, que comprende:
una unidad primaria (402) que genera datos primarios para llevar a cabo la funcionalidad normal de sistema;
una unidad secundaria que genera datos secundarios para llevar a cabo una funcionalidad alternativa de sistema;
una puerta de seguridad primaria (404) acoplada a la unidad primaria (402), donde la puerta de seguridad primaria (404) proporciona los datos primarios como una salida primaria en respuesta a una determinación de validez de los datos primarios; y
una puerta de seguridad secundaria acoplada a la unidad secundaria, donde la puerta de seguridad secundaria proporciona los datos secundarios como una salida secundaria en respuesta a una determinación de validez de los datos secundarios; estando el sistema caracterizado por que el sistema comprende además:
un selector de salida (410) que está acoplado tanto a la puerta de seguridad primaria (404) como a la puerta de seguridad secundaria de la primera fase, donde el selector de salida proporciona una salida de sistema en respuesta a las determinaciones de la validez de los datos primarios y los datos secundarios.

2. El sistema según la reivindicación 1, en el que la puerta de seguridad primaria (404) determina la validez de los datos primarios en respuesta a una envolvente permisiva proporcionada por la puerta de seguridad secundaria.

3. El sistema según la reivindicación 1, que comprende además:

una o más fases adicionales que comprenden una segunda fase, donde una salida de datos primarios de la segunda fase proporciona una entrada a la unidad primaria de la primera fase, y una salida de datos secundarios de la segunda fase proporciona una entrada a la unidad secundaria de la primera fase.

4. El sistema según la reivindicación 1, en el que la puerta de seguridad secundaria determina si los datos secundarios se recibieron dentro de una ventana de tiempo predefinida para determinar si los datos secundarios son válidos.

5. El sistema según la reivindicación 1, en el que la puerta de seguridad secundaria comprende una memoria intermedia (408) que almacena los datos secundarios en respuesta a la determinación de la validez de los datos secundarios.

6. El sistema según la reivindicación 1, en el que la unidad secundaria proporciona datos secundarios previamente almacenados como salida secundaria en respuesta a una determinación de no validez de los datos secundarios.

7. El sistema según la reivindicación 1, en el que la salida de sistema comprende datos de control para hacer funcionar un vehículo.

8. Un procedimiento implementado por ordenador para hacer cumplir estrictos requisitos de seguridad en una arquitectura de seguridad (400, 500) para vehículos autónomos, donde el procedimiento es ejecutado por uno o más procesadores y comprende:

generar datos primarios para llevar a cabo la funcionalidad normal de sistema;
generar datos secundarios para llevar a cabo una funcionalidad alternativa de sistema;
proporcionar los datos primarios como una salida primaria de una primera fase en respuesta a la determinación de validez de los datos primarios;
proporcionar los datos secundarios como una salida secundaria de la primera fase en respuesta a la determinación de validez de los datos secundarios; estando el procedimiento caracterizado por que el procedimiento comprende además:
proporcionar una salida de sistema en respuesta a la determinación de validez de los datos primarios y los datos secundarios, donde la salida de sistema es proporcionada por un selector de salida (410) que está acoplado tanto a una puerta de seguridad primaria (404) que proporciona los datos primarios como a una puerta de seguridad secundaria que proporciona los datos secundarios, donde la puerta de seguridad primaria y la puerta de seguridad secundaria son de la primera fase.

9. El procedimiento según la reivindicación 8, en el que la determinación de la validez de los datos primarios es sensible a una envolvente permisiva.

10. El procedimiento según la reivindicación 8, en el que:

generar los datos primarios comprende recibir una entrada primaria a través de una salida de datos primarios de una segunda fase; y
generar los datos secundarios comprende recibir una entrada secundaria a través de una salida de datos secundarios de una segunda fase.

11. El procedimiento según la reivindicación 8, en el que la determinación de la validez de los datos secundarios comprende determinar que los datos secundarios se generaron dentro de una ventana de tiempo predefinida.

12. El procedimiento según la reivindicación 8, que comprende además almacenar los datos secundarios en respuesta a la determinación de la validez de los datos secundarios.

13. El procedimiento según la reivindicación 8, que comprende además:

proporcionar datos secundarios previamente almacenados como salida secundaria de la primera fase en respuesta a la determinación de no validez de los datos secundarios.

14. El procedimiento según la reivindicación 8, en el que la salida de sistema comprende datos de control para hacer funcionar un vehículo.

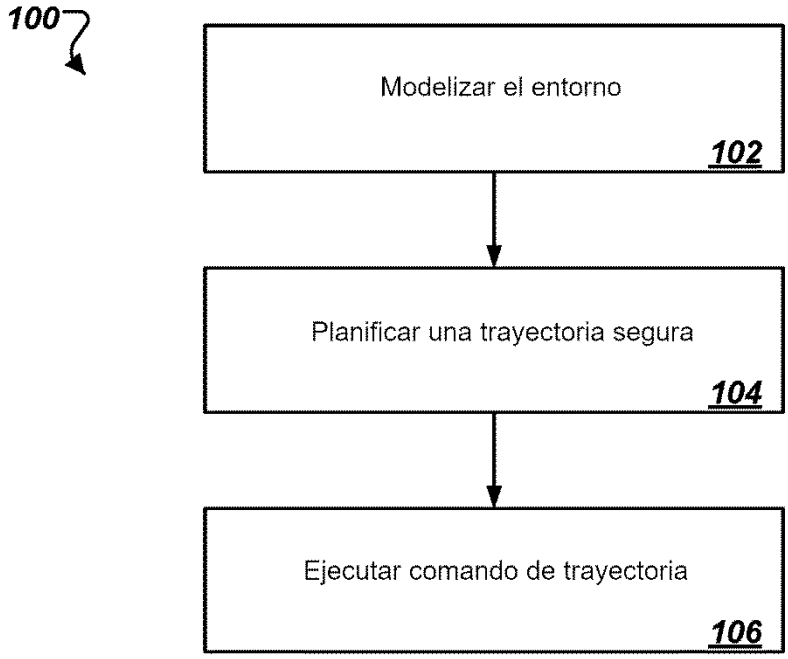
100
Modelizar el entorno
102
Planificar una trayectoria segura
104
Ejecutar comando de trayectoria
106

FIG. 1

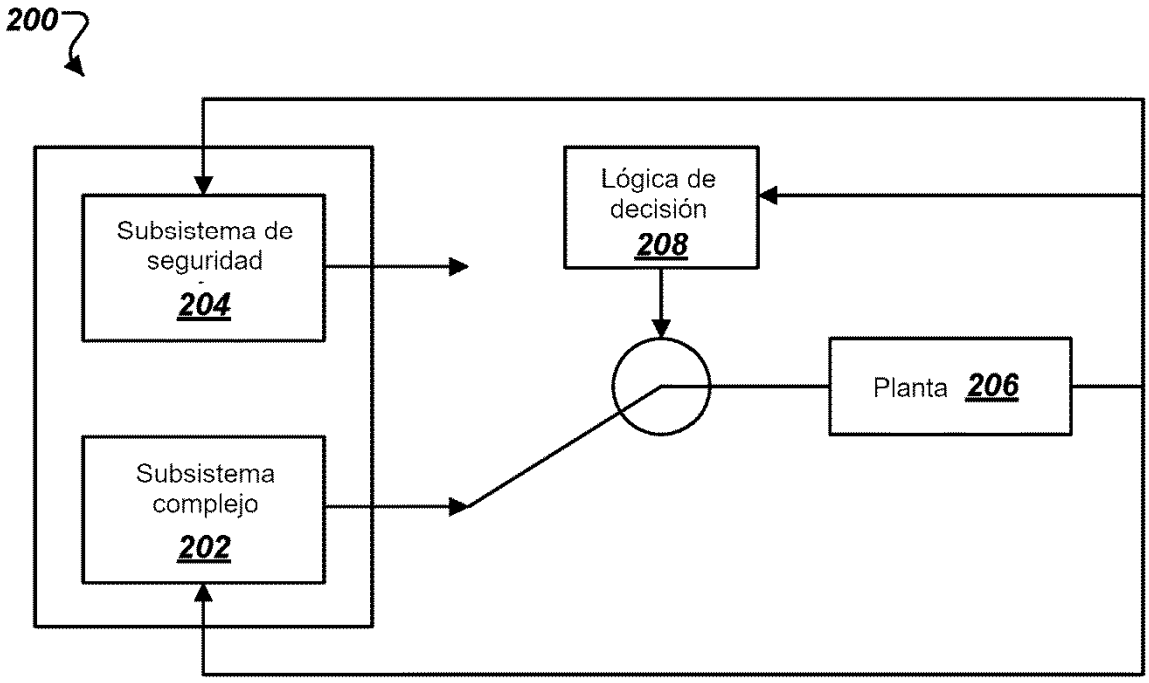
200
Subsistema de seguridad
204
Subsistema complejo
202
Lógica de decisión
208
Planta 206

FIG. 2

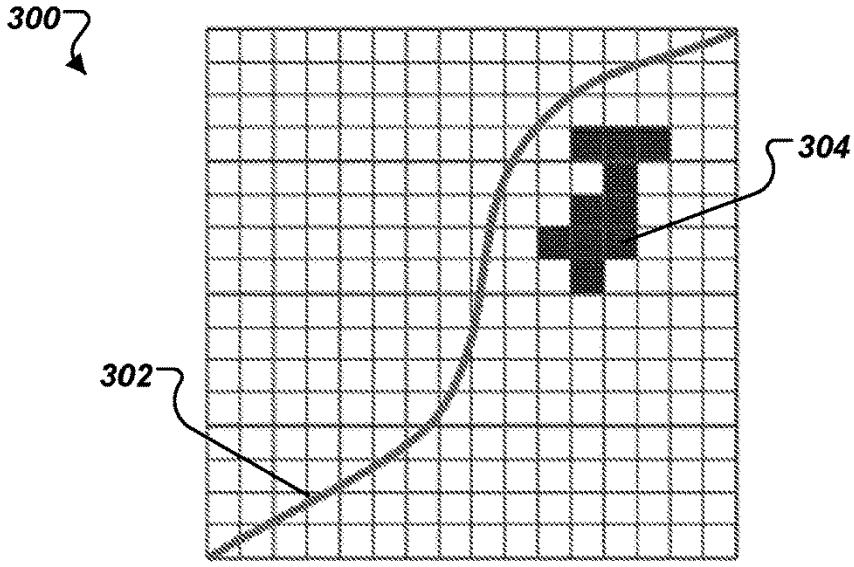

FIG. 3

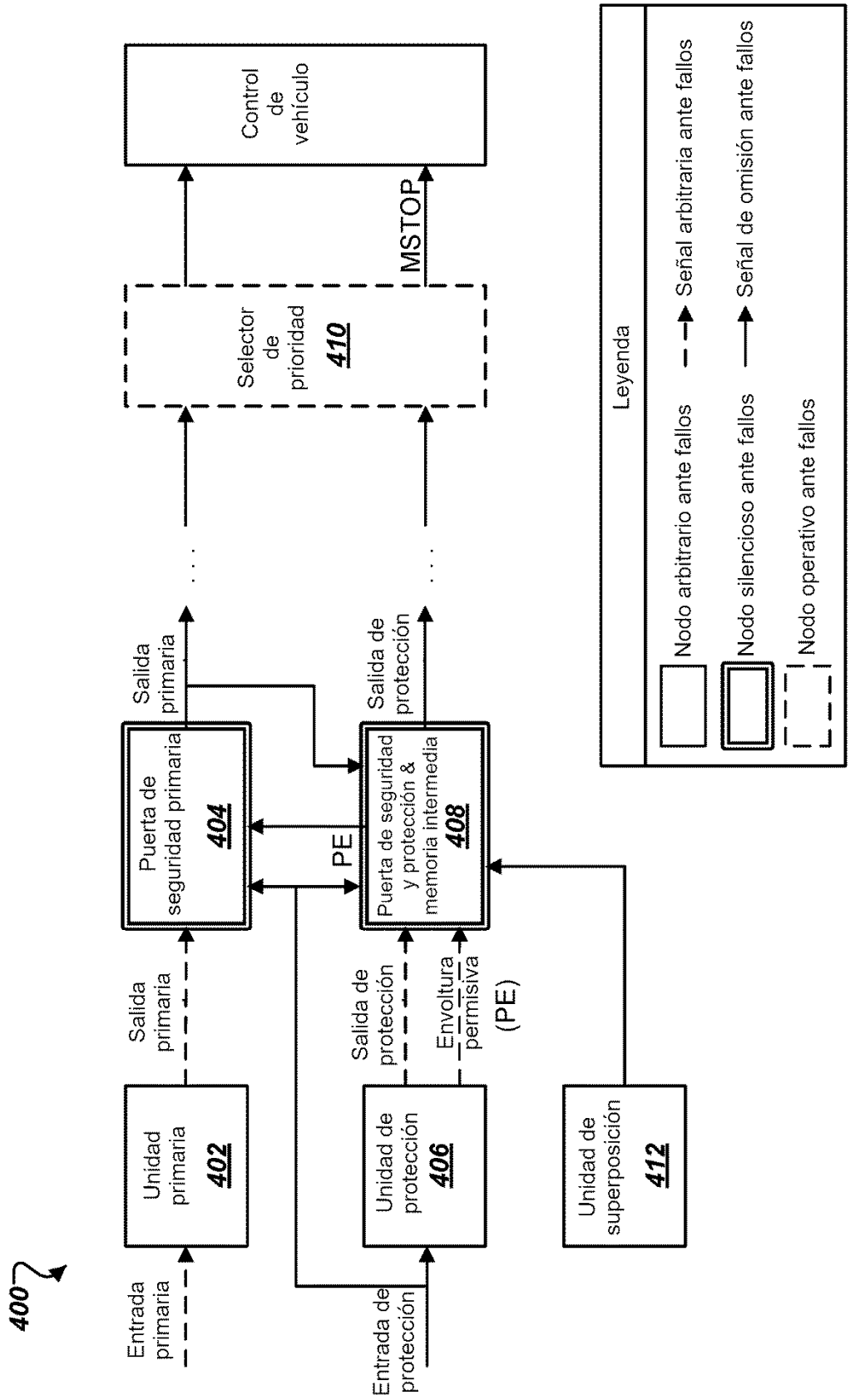
400
Entrada primaria
Entrada de protección
Unidad primaria 402
Unidad de protección 406
Unidad de superposición 412
Salida primaria
Salida de protección
Envoltura permisiva (PE)
Puerta de seguridad primaria 404
PE
Puerta de seguridad y protección & memoria intermedia 408
Salida primaria
Salida de protección
Selector de prioridad 410
MSTOP
Control de vehículo
Leyenda
Nodo arbitrario ante fallos
Señal arbitraria ante fallos
Nodo silencioso ante fallos
Señal de omisión ante fallos
Nodo operativo ante fallos

FIG. 4

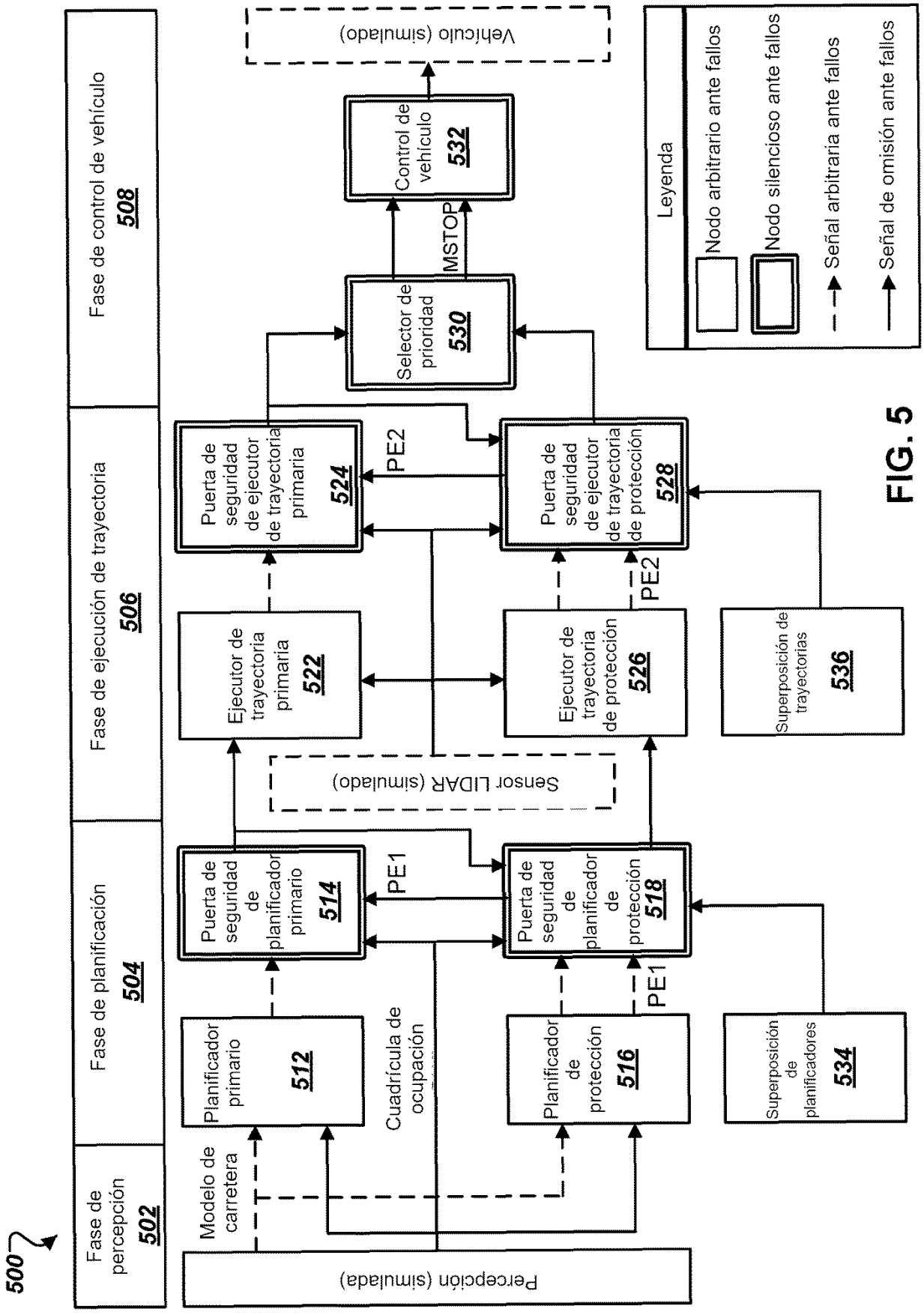
500
Fase de percepción 502
Fase de planificación 504
Fase de ejecución de trayectoria 506
Fase de control de vehículo 508
Percepción (simulada)
Modelo de carretera
Cuadrícula de ocupación
Planificador primario 512
Planificador de protección 516
Superposición de planificadores 534
Puerta de seguridad de planificador primario 514
Puerta de seguridad de planificador de protección 518
PE1
PE1
Sensor LIDAR (simulado)
Ejecutor de trayectoria primaria 522
Ejecutor de trayectoria de protección 526
Superposición de trayectorias 536
Puerta de seguridad de ejecutor de trayectoria primaria 524
Puerta de seguridad de ejecutor de trayectoria de protección 528
PE2
PE2
Selector de prioridad 530
MSTOP
Control de vehículo 532
Vehículo (simulado)
Leyenda
Nodo arbitrario ante fallos
Nodo silencioso ante fallos
Señal arbitraria ante fallos
Señal de omisión ante fallos

FIG. 5

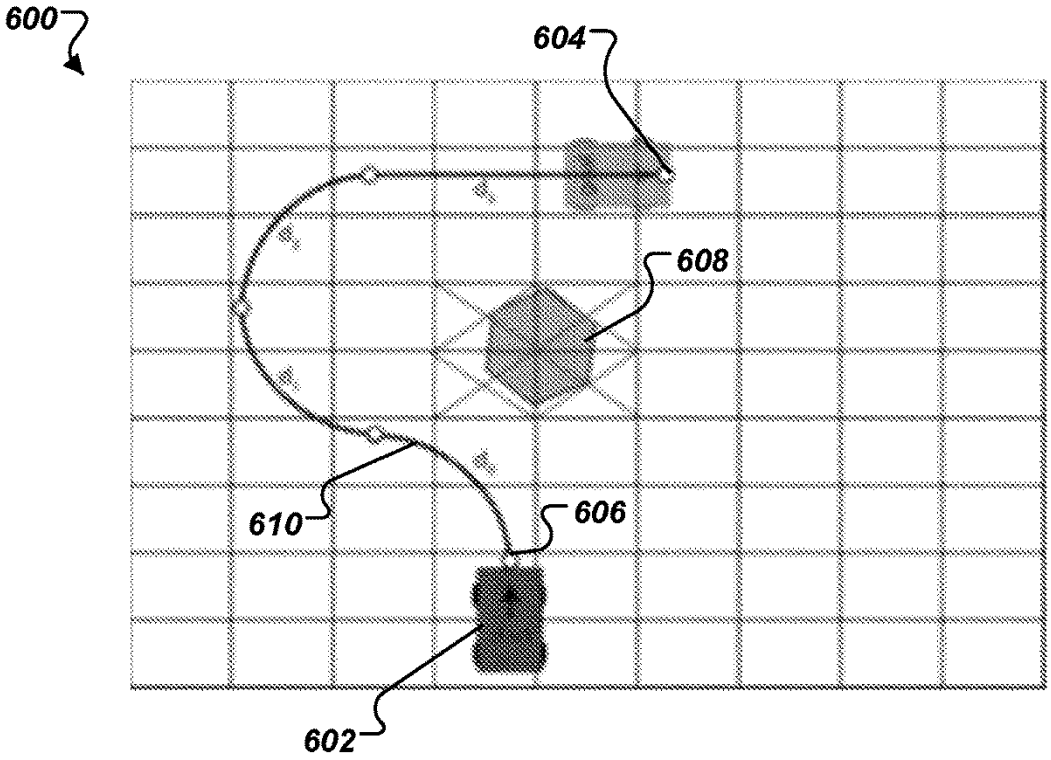

FIG. 6

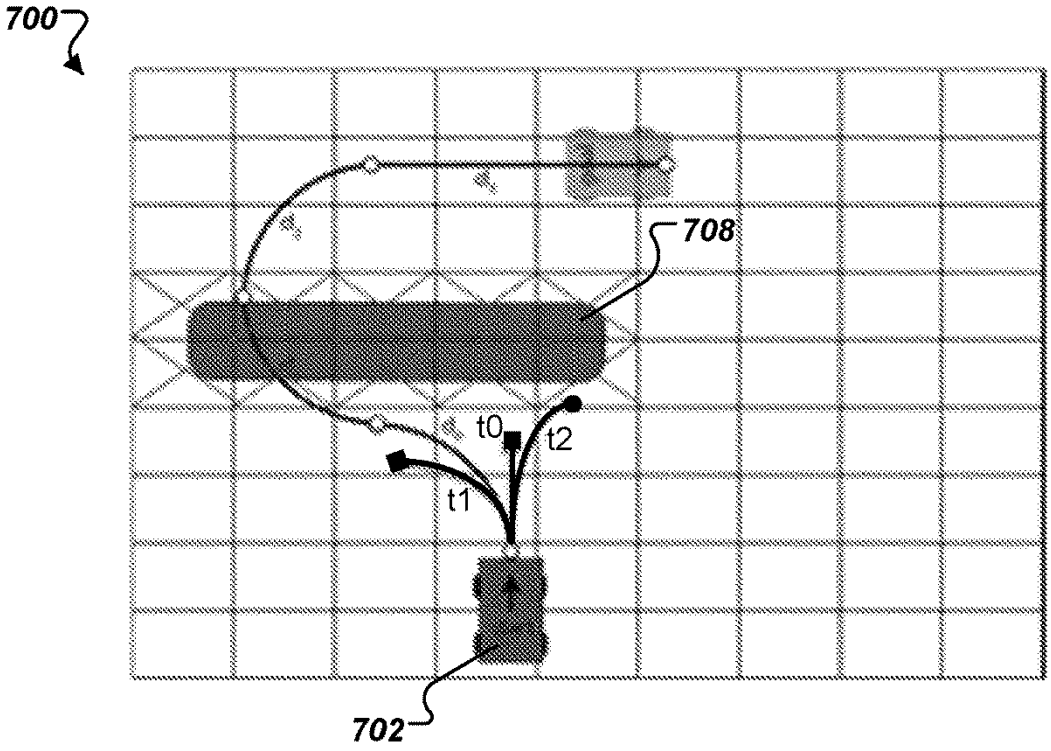

FIG. 7

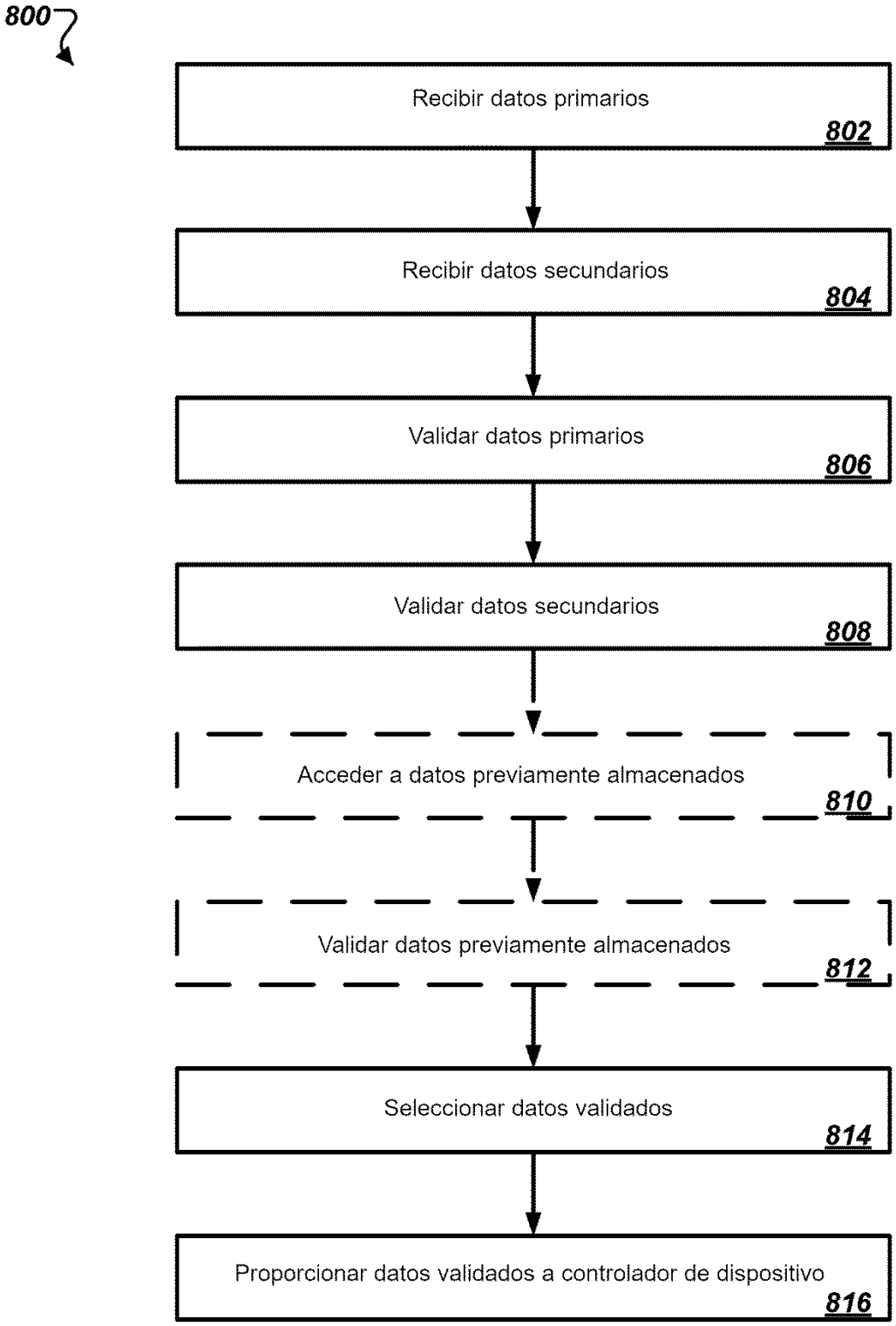
800
Recibir datos primarios
802
Recibir datos secundarios
804
Validar datos primarios
806
Validar datos secundarios
808
Acceder a datos previamente almacenados
810
Validar datos previamente almacenados
812
Seleccionar datos validados
814
Proporcionar datos validados a controlador de dispositivo
816

FIG. 8