Proyecto de Grado

Presentado ante la ilustre Universidad de Los Andes como requisito parcial para obtener el Título de Ingeniero de Sistemas

Estudio e Implementación de una Plataforma de Software que permita generar Mapas para la Navegación de un Robot Móvil

Por

Br. Carlos Eduardo Paparoni Bruzual

Tutor: Prof. Dr. Rafael Rivas Estrada

Junio 2015

Estudio e Implementación de una Plataforma de Software que permita generar Mapas para la Navegación de un Robot Móvil

Br. Carlos Eduardo Paparoni Bruzual

Proyecto de Grado — Sistemas Computacionales, 39 páginas

Resumen: Un reto substancial en el desarrollo y masificación de los robots móviles autónomos es el proceso de reconocimiento o modelado de su entorno.
Existen múltiples soluciones de software que permiten el control de bases robóticas, planificación de rutas, localización y creación de mapas de entorno para navegación y ubicación de uno o varios robots en éste.
En este trabajo se presenta la investigación, documentación e implementación de una plataforma de software adecuada para el manejo del hardware robótico disponible en el Laboratorio de Sistemas Discretos, Automatización e Integración (LaSDAI), con el fin de soportar y utilizar el dispositivo Kinect®, desarrollado por la compañía Microsoft®, específicamente mediante el uso del emisor láser y de la cámara infrarroja del mismo, para realizar medidas y generar mapas del entorno que puedan ser utilizados para la localización y navegación de un robot móvil.
Se desarrollará utilizando UML 2.0 (Unified Modeling Language) para el modelado y como guía en su desarrollo, el método PXP que forma parte de los métodos Ágiles de desarrollo.

Palabras clave: Robots Móviles, Construcción de Mapas, Visión por Computador, Kinect, ROS

Este trabajo fue procesado en LaTeX.

Índice general

Índice de cuadros

Capítulo 1

Introduccion

En este capítulo se presenta una descripcion del Laboratorio de Sistemas Discretos, Automatizacion e Integracion (LaSDAI), en donde se llevó a cabo la elaboracion del proyecto. Se definen los antecedentes que son la base para la presentacion del problema así como también el planteamiento de este último, la justificacion del proyecto de grado, los objetivos y la metodología que encaminaron el desarrollo de la solucion del mismo.

1.1. Descripcion de LaSDAI

El Laboratorio de Sistemas Discretos, Automatizacion e Integracion (LaSDAI), se funda en el año 1993 bajo la direccion del Dr. Edgar Chacon con el apoyo del Programa de Nuevas Tecnologías del CONICIT (Consejo Nacional de Investigaciones Científicas y Tecnológicas), bajo el proyecto N°I-22. LaSDAI está adscrito a la Escuela de Ingeniería de Sistemas de la Facultad de Ingeniería de la Universidad de Los Andes. A partir del año 2007 actualiza sus líneas de investigacion y es re-estructurado. Actualmente está conformado por personal docente, estudiantes y colaboradores de la Universidad de Los Andes.[1]

1.1.1. Personal

Miembros

- Dr. Eladio Dapena G. (Coordinador)

- Dr. Rafael Rivas Estrada
- Ing. Jose G. Gonzalez
- Dr. Edgar Chacón

Colaboradores

- PhD. Addison Rios
- Dr. José Aguilar
- Dr. José María Armingol (UC3M)
- Dr. Arturo de La Escalera (UC3M)
- Dra. Ana Corrales (UC3M)
- Ing. David Godoy
- Lic. Nadia González
- MSc. Asdrúbal Fernández

1.1.2. Mision

El Laboratorio de Sistemas Discretos, Automatizacion e Integracion, LaSDAI, es un espacio para la docencia, la investigacion y el desarrollo de productos, en las áreas de robótica, automatizacion industrial y vision por computador, con el objeto de coadyuvar en el desarrollo tecnológico del país. LaSDAI, tiene como meta difundir sus resultados y vincularse con el sector productivo nacional, con la consiga de I+D+I Investigacion, Desarrollo e Innovacion. Sus actividades de soporte a la docencia tanto en pregrado como postgrado, junto con el desarrollo de proyectos de investigacion y las labores de extension, constituyen un complemento que integra diferentes aristas para el desarrollo. Las actividades de extension, mediante asesorías y cursos, colaboran a lograr el autofinanciamiento como estrategia de consolidacion de nuestras actividades.

1.1.3. Vision

LaSDAI será un referente nacional, como un Instituto de Docencia, Investigacion y Desarrollo, en las áreas de robótica y automatizacion, de carácter autonomo y autofinanciado.

1.1.4. Objetivos

Objetivo General

El Laboratorio de Sistemas Discretos, Automatizacion e Integracion fue creado con el fin de desarrollar conceptos y herramientas que soporten la construccion de sistemas automáticos en ambientes de produccion continua, tal como son los de la industria de procesos, mediante una generalizacion de las técnicas de los ambientes de manufactura.

La automatizacion en la industria de Procesos Continuos es un proceso complejo, por la cantidad de elementos involucrados estén un proceso de mejoramiento, donde la automatizacion juega un papel esencial. Los paradigmas para la implantacion de una automatizacion integral en una industria compleja no han sido totalmente definidos, pues el mayor énfasis en la comunidad científica a nivel internacional ha sido dado a la industria de manufactura; de aquí la importancia que tiene para el grupo el desarrollo de trabajos de investigacion en el área, así como la formacion de personal.

Objetivos Específicos

- Realizar actividades de Investigacion, Desarrollo e Innovacion (I+D+I), en las áreas de Robótica, Automatizacion Industrial y Vision por Computador.
- Realizar asesorías a la Industria nacional e internacional.
- Divulgacion del conocimiento mediante la docencia, organizacion de eventos, publicacion de resultados, etc.
- Formacion de personal en las áreas a fines a nivel de pregrado, postgrado, doctorado en diversas universidades.
- Desarrollar proyectos en cooperacion con otros centros y laboratorios de investigacion.
- Formacion de personal mediante cursos de extension en empresas.
- Establecer relaciones de cooperacion con el sector productivo nacional mediante el desarrollo de proyectos.
- Participar en eventos científicos nacionales e internacionales.

1.1.5. Antecedentes

El filósofo Immanuel Kant propuso a través de su teoría de la percepcion, que nuestro conocimiento del mundo exterior depende de nuestras formas de percepcion. Así como el cuerpo humano posee, en general, cinco sentidos universalmente conocidos que le ayudan a percibir el entorno que lo rodea, el estudio de los mismos lo ha llevado a investigar y desarrollar maneras de emular estos sentidos de forma artificial, con múltiples propósitos; entre ellos, el de proveer a entidades hechas por el hombre de la habilidad de reconocer el mundo a su alrededor, y en consecuencia, la capacidad de actuar en él.

Nuestra condicion humana nos permite percibir la estructura en tres dimensiones del mundo a nuestro alrededor con aparente facilidad. Por ejemplo, con sólo ver alrededor en una habitacion llena de cosas, usted podría contar e inclusive nombrar a cada uno de los objetos que le rodea; inclusive, podría adivinar la textura de los mismos sin necesidad de hacer uso del sentido del tacto. Así mismo, la percepcion en tres dimensiones le permite juzgar con gran precision la distancia desde su ubicacion actual hasta cada objeto de interés, permitiéndole tocarlo, tomarlo o manipularlo si así lo desea. Esta percepcion, que nosotros como seres humanos llamamos sentido de la vista, se efectúa a través de células especializadas que tienen receptores que reaccionan a estímulos específicos (en este caso, ondas de radiacion electromagnética de longitudes específicas, que se registran como la sensacion de la luz), ubicadas en nuestros ojos.

Si bien la descripcion del sentido de la vista es -o parece ser- sencilla, se trata de un sentido sumamente complejo y de hecho, podría decirse que es uno de los sentidos más importantes para el ser humano, así como el más perfecto y evolucionado.

¿Por qué se habla de complejidad? Szeliski nos explica que, "en parte, es porque la vision es un problema inverso, donde buscamos encontrar variables desconocidas dada informacion insuficiente para especificar totalmente la solucion. Por tanto, debemos recurrir a modelos físicos y probabilísticos para discerner entre soluciones potenciales. Sin embargo, modelar el mundo visual en toda su complejidad es mucho más difícil que, por ejemplo, modelar el tracto vocal que produce sonidos hablados."[2]

Esta complejidad lo hace convertirse en un campo de estudio de gran importancia, cuya denominacion, a los fines de la emulacion mencionada anteriormente, es de la

Vision Artificial, también conocida como Vision por Computador.

El inicio de la vision artificial, desde el punto de vista práctico, fue marcado por Larry Roberts, el cual, en 1961 creó un programa que podía "ver" una estructura de bloques, analizar su contenido y reproducirla desde otra perspectiva, demostrando así a los espectadores que esa informacion visual que había sido mandada al ordenador por una cámara, había sido procesada adecuadamente por él.[3]

1.2. Definicion del Problema

LaSDAI, en sus áreas de Vision por Computador y Robótica, desea estudiar alternativas de plataformas de software para poder utilizar robots autonomos, que provean soporte para sistemas de medicion láser, infrarrojo o una combinacion de ambos, mediante los cuales se pueda realizar medidas de distancias y así, poder generar mapas del entorno a través de dichas medidas.

Si bien se cuenta ya con dos plataformas robóticas (denominados "LR1" y "LR2") se carece de una plataforma programática común, con amplio soporte de la comunidad de investigacion en robótica y de conocimiento en LaSDAI. Esta condicion limita sustancialmente la investigacion, el uso y la difusion de tecnologías afines a la robótica y la vision por computadora, dejando de lado este campo de investigacion.

1.3. Justificacion

LaSDAI posee y utiliza dos plataformas robóticas, los cuales cuentan cada uno con interfaces programáticas desarrolladas por separado. Esto, si bien es adecuado para el uso específico de cada plataforma, supone problemas de intercomunicacion e interoperacion, sin mencionar el costo en mantenimiento de dichas interfaces a nivel de código. Por ende, establecer una plataforma de software común para ambos, reduce a lo mínimo necesario la codificacion personalizada para cada plataforma robótica, provee soporte al involucrar a un mayor número de personas y facilita el desarrollo de otras plataformas robóticas derivadas de esta.

1.4. Objetivos

1.4.1. Objetivo General

Investigar y desarrollar documentacion adecuada que permita establecer una plataforma común de software para el manejo y navegacion de robots móviles, que provea soporte a sensores tales como Microsoft Kinect, para obtener datos y realizar mediciones de entorno.

1.4.2. Objetivos Específicos

1. Analizar las alternativas en plataformas de software disponibles para control robótico.
2. Analizar el software disponible para elaboracion de mapas de entorno.
3. Analizar los requerimientos del módulo de creacion de mapas.
4. Generar un mapa de entorno mediante el software.
5. Realizar documentacion de la estructura de los mapas generados mediante el software.
6. Realizar documentacion adecuada y actualizada para la difusion y posterior uso del software.

1.5. Metodología Utilizada

Con la finalidad de llevar a cabo el desarrollo del proyecto de grado de forma eficiente y a la vez incorporar metodologías actuales enfocadas al desarrollo por parte de individuos (como es normalmente el caso en cuanto a proyectos de grado), se estudió el uso del método PSP [4] (Personal Software Process) mejorado con prácticas tomadas de los métodos de programacion Ágiles, en particular, el método Extreme Programming, orientado a una sola persona, es decir, PXP [5] (Personal eXtreme Programming).

Esto se llevó a cabo mediante las siguientes actividades realizadas:

- Recoleccion de requerimientos.

- Planificacion.
- Inicializacion de iteracion
  - Diseño.
  - Implementacion.
    - Pruebas unitarias (aplicacion de Desarrollo Orientado por Pruebas).
    - Codificacion.
    - Refactorizacion.
  - Pruebas de sistema.
  - Retrospectiva, análisis de resultados.
- Finalizacion de iteracion

Capítulo 2

Marco Teórico

En este capítulo, se describen los fundamentos teóricos necesarios para el entendimiento y comprensión del proyecto. Se da una definición de los conceptos de robótica, robots, visión por computadora y la definición de las tareas realizadas por un robot autónomo para la navegación y ubicación de si mismo en un entorno. Se especifican las características técnicas del sistema Kinect y por último, se define la metodología Ágil para desarrollo de proyectos y los métodos PXP y Kanban para el desarrollo de aplicaciones, creando con esto una base teórica con la finalidad de tener una introducción del hardware y software utilizado, las herramientas de diseño y permitir al lector tener una idea de la naturaleza del contenido del resto del documento.

2.1. Robótica

La Robótica es aquella rama dentro de la Ingeniería que se ocupa de la aplicación de la informática al diseño y al uso de máquinas con el objetivo que de lo que de esto resulte pueda de alguna manera sustituir a las personas en la realización de determinadas funciones o tareas. En palabras más simples, la robótica es la ciencia y la tecnología de los robots, porque básicamente se ocupa del diseño, manufactura y aplicaciones de los robots que crea. En la Robótica se combinan varias disciplinas al mismo tiempo, como la mecánica, la electrónica, la inteligencia artificial, la informática y la ingeniería de control, en tanto, también, por los quehaceres que desempeña, resulta fundamental el

aporte que recibe y extrae de campos tales como el álgebra, los autómatas programables y las máquinas de estados.

2.2. Robots

El término robot alcanza su primera repercusión en la tercera década del siglo pasado, a instancias de R.U.R (robots Universal Rossum), una obra teatral de ciencia ficción escrita por el autor checo Karel Čapek, en la cual por primera vez se hace alusión al concepto de robot, extraído del término checo "robota", que significaba "trabajos forzados".

A su vez, el término "robótica" es acuñado por Isaac Asimov, definiendo a la ciencia que estudia a los robots. Asimov creó también las Tres Leyes de la Robótica, definidas de esta manera:

1. Un robot no puede actuar contra un ser humano o, mediante la inacción, permitir que un ser humano sufra daños.
2. Un robot debe obedecer las órdenes dadas por los seres humanos, salvo que estén en conflictos con la primera ley.
3. Un robot debe proteger su propia existencia, a no ser que esté en conflicto con las dos primeras leyes.

Desde sus comienzos como disciplina y como parte fundamental de la Ingeniería, la Robótica ha estado incansablemente buscando construir artefactos que materialicen el deseo humano de crear seres a su semejanza a quienes poder delegarles tareas, trabajos o actividades por demás pesadas y desagradables de llevar a cabo. Pero y aunque muchos ni se lo esperen, desde tiempos inmemoriales, muy, muy lejos de las computadoras, hubo unas cuantas expresiones de la robótica. Porque por ejemplo, los antiguos egipcios unieron brazos mecánicos a las estatuas de sus dioses y esgrimían que el movimiento de los miembros se llevaba a cabo por obra y gracias de estos, inclusive los griegos construyeron estatuas que operaban con sistemas hidráulicos, los cuales eran utilizados para fascinar a los adoradores de los templos.

Y también, aproximadamente entre los siglos XVII y XVIII, en Europa, se construyeron muñecos mecánicos muy ingeniosos que ostentaban algunas características como las que presentan los robots de la actualidad. En un constante e incansable ensayo a través de los siglos y cuando ya era un hecho la entrada en el nuevo milenio (2000), la empresa Honda Motor Co. Ltda. concretó a Asimo, el primer robot humanoide capaz de desplazarse de forma bípeda e interactuar con las personas.

La historia de la robótica va unida a la construcción de "artefactos", que trataban de materializar el deseo humano de crear seres a su semejanza y que lo descargasen del trabajo. El ingeniero español Leonardo Torres Quevedo (que construyó el primer mando a distancia para su automóvil mediante telegrafía sin hilo, el ajedrecista automático, el primer transbordador aéreo y otros muchos ingenios) acuñó el término "automática" en relación con la teoría de la automatización de tareas tradicionalmente asociadas.

2.2.1. Clasificación de los robots

En términos generales, un robot se clasifica por sus capacidades, así como también su área de operación, sus grados de autonomía o el fin con el que han sido construidos. Sin embargo, también pueden clasificarse en términos de la era en la que fue implementado, según su forma de construcción o la manera como son controlados. A continuación se exponen algunas formas comunes de clasificación:

Según su cronología

La que a continuación se presenta es la clasificación más común:

1ª Generación. Manipuladores. Son sistemas mecánicos multifuncionales con un sencillo sistema de control, bien manual, de secuencia fija o de secuencia variable.

2ª Generación. Robots de aprendizaje. Repiten una secuencia de movimientos que ha sido ejecutada previamente por un operador humano. El modo de hacerlo es a través de un dispositivo mecánico. El operador realiza los movimientos requeridos mientras el robot le sigue y los memoriza.

3ª Generación. Robots con control sensorizado. El controlador es una computadora que ejecuta las órdenes de un programa y las envía al manipulador para que realice los movimientos necesarios.

4ª Generación. Robots inteligentes. Son similares a los anteriores, pero además poseen sensores que envían información a la computadora de control sobre el estado del proceso. Esto permite una toma inteligente de decisiones y el control del proceso en tiempo real.

Según su arquitectura

La arquitectura, es definida por el tipo de configuración general del robot, puede ser metamórfica. El concepto de metamorfismo, de reciente aparición, se ha introducido para incrementar la flexibilidad funcional de un robot a través del cambio de su configuración por el propio robot. El metamorfismo admite diversos niveles, desde los más elementales (cambio de herramienta o de efecto terminal), hasta los más complejos como el cambio o alteración de algunos de sus elementos o subsistemas estructurales.

Los dispositivos y mecanismos que pueden agruparse bajo la denominación genérica del robot, tal como se ha indicado, son muy diversos y es por tanto difícil establecer una clasificación coherente de los mismos que resista un análisis crítico y riguroso. La subdivisión de los robots, con base en su arquitectura, se hace en los siguientes grupos: poliarticulados, móviles, androides, zoomórficos e híbridos.

1. Poliarticulados En este grupo se encuentran los robots de muy diversa forma y configuración, cuya característica común es la de ser básicamente sedentarios (aunque excepcionalmente pueden ser guiados para efectuar desplazamientos limitados) y estar estructurados para mover sus elementos terminales en un determinado espacio de trabajo según uno o más sistemas de coordenadas, y con un número limitado de grados de libertad. En este grupo, se encuentran los manipuladores, los robots industriales, los robots cartesianos y se emplean cuando es preciso abarcar una zona de trabajo relativamente amplia o alargada, actuar sobre objetos con un plano de simetría vertical o reducir el espacio ocupado en el suelo.

2. Móviles Son robots basados en carros o plataformas, dotados de un sistema locomotor de tipo rodante y con gran capacidad de desplazamiento. Siguen su camino por telemando o guiándose por la información recibida de su entorno a través de sus sensores. Estos robots aseguran el transporte de piezas de un punto a otro de una cadena de fabricación. Guiados mediante pistas materializadas a través de la radiación electromagnética de circuitos empotrados en el suelo, o a través de bandas detectadas fotoeléctricamente, pueden incluso llegar a sortear obstáculos y están dotados de un nivel relativamente elevado de inteligencia.
3. Androides Son robots que intentan reproducir total o parcialmente la forma y el comportamiento cinemática del ser humano. Actualmente, los androides son todavía dispositivos muy poco evolucionados y sin utilidad práctica, y destinados, fundamentalmente, al estudio y experimentación. Uno de los aspectos más complejos de estos robots, y sobre el que se centra la mayoría de los trabajos, es el de la locomoción bípeda. En este caso, el principal problema es controlar dinámica y coordinadamente en el tiempo real el proceso y mantener simultáneamente el equilibrio del robot.
4. Zoomórficos Los robots zoomórficos, que considerados en sentido no restrictivo podrían incluir también a los androides, constituyen una clase caracterizada principalmente por sus sistemas de locomoción que imitan a los diversos seres vivos. A pesar de la disparidad morfológica de sus posibles sistemas de locomoción es conveniente agrupar a los robots zoomórficos en dos categorías principales: caminadores y no caminadores. El grupo de los robots zoomórficos no caminadores está muy poco evolucionado. Los experimentos efectuados en Japón basados en segmentos cilíndricos biselados acoplados axialmente entre sí y dotados de un movimiento relativo de rotación. Los robots zoomórficos caminadores multípedos son muy numerosos y están siendo objeto de experimentos en diversos laboratorios con vistas al desarrollo posterior de verdaderos vehículos terrenos, piloteados o autónomos, capaces de evolucionar en superficies muy accidentadas. Las aplicaciones de estos robots serán interesantes en el campo de la exploración espacial y en el estudio de los volcanes.
5. Híbridos Corresponden a aquellos de difícil clasificación, cuya estructura se sitúa

en combinación con alguna de las anteriores ya expuestas, bien sea por conjunción o por yuxtaposición. Por ejemplo, un dispositivo segmentado articulado y con ruedas, es al mismo tiempo, uno de los atributos de los robots móviles y de los robots zoomórficos.

Según su modalidad de control

La modalidad de control se refiere a la dependencia –o no– de un operador humano que instruya órdenes al robot; por tanto, se subdivide en dos grandes grupos:

1. Teledirigidos Se define como robots teledirigidos a aquellos que necesitan la intervención de un operador humano, ya sea en forma parcial o total, por ejemplo los utilizados en la desactivación de explosivos.
2. Autónomos Se les llama autónomos a aquellos robots que son capaces de tomar sus propias decisiones basados en la comprensión del entorno en que se encuentren. Existen numerosos tipos de robots de variadas configuraciones que se encuadran en esta categoría, como es el caso de un brazo robot con visión artificial sin asistencia humana realizando tareas de clasificación de objetos, o de los robots móviles del tipo vehículo.

2.2.2. Robots Autónomos

Un robot autónomo es un robot que realiza comportamientos o tareas con un alto grado de autonomía, que es particularmente deseable en campos tales como la exploración del espacio, la limpieza de suelos, cortar el césped, y el tratamiento de aguas residuales.

Algunos robots de fábricas modernas son "autónomos" dentro de los límites estrictos de su entorno directo. Puede que no sea la existencia de todos los grados de libertad en su entorno, pero el lugar de trabajo del robot de la fábrica es un reto y, a menudo puede contener, variables caóticas e impredecibles. La orientación exacta y la posición del siguiente objeto de trabajo e incluso (en las fábricas más avanzadas) el tipo de objeto y la tarea requerida debe ser determinado. Esto puede variar de manera impredecible (por lo menos desde el punto de vista del robot).

Un área importante de la investigación robótica es permitir que el robot pueda hacer frente a su entorno ya sea en tierra, bajo el agua, en el aire, bajo tierra o en el espacio.

Un robot completamente autónomo puede:

- Obtener información sobre el medio ambiente (Regla #1)
- Trabajar por un período prolongado sin intervención humana (Regla #2)
- Mover todo o parte de sí mismo a través de su entorno operativo sin ayuda humana (Regla #3)
- Evitar situaciones que son perjudiciales para las personas, los bienes, o sí mismo, si esos son parte de sus especificaciones de diseño (Regla #4)

Un robot autónomo también puede aprender o adquirir nuevos conocimientos como ajustarse a nuevos métodos para llevar a cabo sus tareas o adaptarse a un entorno cambiante.

Al igual que otras máquinas, los robots autónomos todavía requieren de un mantenimiento regular.

Ejemplos:

Automantenimiento

El primer requisito para la autonomía física completa es la capacidad de un robot para cuidar de sí mismo. Muchos de los robots que funcionan con baterías en el mercado hoy en día pueden encontrar y conectarse a una estación de carga, y algunos juguetes como Aibo de Sony son capaces de realizar auto-acoplamiento para cargar sus baterías.

El mantenimiento realizado se basa en la "propiocepción", o la capacidad de sentir el propio estado interno. En el ejemplo de carga de la batería, el robot puede decir propioceptivamente que sus baterías están bajas y en consecuencia, buscar el cargador. Otro sensor propioceptivo es común para la supervisión de calor. El aumento de la propiocepción se requerirá para los robots para trabajar de forma autónoma, cerca de la gente y en ambientes hostiles. Las propiocepciones comunes incluyen sensores de detección térmica, óptica y háptica, así como el efecto Hall (eléctrica).

Sintiendo el medio ambiente

La exterocepción es la detección de información del medio ambiente. Los robots autónomos deben tener una gama de sensores ambientales para llevar a cabo su tarea y no meterse en problemas.

Los sensores exteroceptivos comunes incluyen el espectro electromagnético, el sonido, el tacto, química (olor), la temperatura, la distancia hacia múltiples objetos, y la altitud. Algunas cortadoras de césped robóticas adaptarán su programación mediante la detección de la velocidad en la que la hierba crece a medida que sea necesario para mantener un césped perfectamente cortado, y algunos robots de limpieza por aspiración toemem detectores de tierra que detectan la cantidad de suciedad que se está recogido y utilizan esta información para decirles que deben permanecer en un área durante más tiempo.

Desempeño de tareas

El siguiente paso en el comportamiento autónomo es para llevar a cabo realmente una tarea física. Una nueva área que muestra promesa comercial es la de los robots domésticos, con una avalancha de pequeños robots aspiradora que comienzan con iRobot y Electrolux en 2002 Mientras que el nivel de inteligencia no es muy alta en estos sistemas, que navegan en zonas extensas y conducen en situaciones estrechas alrededor de los hogares utilizando sensores de contacto y sin contacto. Ambos robots utilizan algoritmos propietarios para aumentar la cobertura por encima del simple rebote al azar.

El siguiente nivel de ejecución de la tarea autónoma requiere que un robot pueda realizar tareas condicionales. Por ejemplo, los robots de seguridad se pueden programar para detectar intrusos y responder de una manera particular, dependiendo de donde esté ubicado el intruso.

Navegación interior

Para que un robot pueda asociar comportamientos con un lugar (localización) requiere saber dónde está y ser capaz de navegar de punto a punto. Tal navegación

comenzó mediante guía cableada en la década de 1970 y progresó en la década de 2000 a la triangulación mediante balizas. Los robots autónomos comerciales actuales navegan basándose en la detección de caraterísticas naturales.

Los primeros robots comerciales en lograrlo fueron el robot de hospital HelpMate de Pyxus y el robot guardia CyberMotion, ambos diseñados por pioneros en robótica en la década de 1980. Estos robots utilizaban originalmente planos de piso CAD creados manualmente, detección mediante sonar y variaciones de seguimiento de paredes para navegar a través de edificios. La próxima generación, tales como PatrolBot y la silla de ruadas autónoma de MobileRobots, ambos introducidos en 2004, tienen la capacidad de crear sus propios mapas basados en láser de un edificio y navegar por zonas abiertas, así como corredores. Su sistema de control cambia su ruta sobre la marcha si algo bloquea el camino.

Inicialmente, la navegación autónoma se basaba en sensores planares, como los telémetros láser, que sólo pueden realizar detecciones en un plano o nivel. Los sistemas más avanzados ahora fusionan la información de diversos sensores, tanto para la localización (posición) y la navegación. Los sistemas tales como Motivity pueden contar con diferentes sensores en diferentes áreas, dependiendo de lo que proporcione los datos más fiables en el momento, y pueden volver a generar mapas de entorno de forma autónoma.

En lugar de subir escaleras, lo que requiere hardware altamente especializado, la mayoría de los robots de interior navegan en áreas accesibles para minusválidos, controlando ascensores y puertas electrónicas. Con este tipo de interfaces de control de acceso, los robots ya pueden navegar libremente en el interior. Subir o bajar escaleras de forma autónoma y abrir puertas de forma manual, son temas actuales de investigación.

A medida que estas técnicas en interiores se siguen desarrollando, los robots aspiradora adquirirán la capacidad de limpiar una habitación específica designada por el usuario o una planta entera. Los robots de seguridad podrán rodear intrusos de forma cooperativa e incluso cortarles las salidas. Estos avances también traen protecciones asociadas: los mapas internos de los robots permiten típicamente la definición de “zonas prohibidas” para evitar que los mismos entren de forma autónoma en ciertas regiones.

Navegación en exteriores La autonomía al aire libre se logra más fácilmente en el aire, ya que los obstáculos son raros. Los misiles de crucero son robots altamente autónomas y bastante peligrosos. Los aviones no tripulados se utilizan cada vez más para el reconocimiento. Algunos de estos vehículos aéreos no tripulados (UAV) son capaces de volar toda su misión sin ninguna interacción humana en absoluto, excepto posiblemente para el aterrizaje cuando una persona interviene mediante control remoto por radio. Sin embargo, algunos aviones son capaces de realizar aterrizajes automáticos y seguros.

La autonomía al aire libre es más difícil para los vehículos de tierra, debido a:

- La tridimensionalidad del terreno,
- Grandes disparidades en la densidad de superficie
- Exigencias climáticas
- Inestabilidad del medio ambiente detectado

En Estados Unidos, el proyecto MDARS (Mobile Detection Assessment and Response System), que definió y construyó un robot prototipo de vigilancia exterior en la década de 1990, se está llevando a producción y será implementado en 2006. El robot MDARS de General Dynamics puede navegar de forma semi-autónoma y detectar intrusos, utilizando la arquitectura de software MRHA (Multiple Robot Host Architecture) planeada para todos los vehículos militares no tripulados. El robot Seekur fue el primer robot comercial para demostrar las capacidades similares a MDARS para uso general por los aeropuertos, plantas de servicios públicos, instalaciones correccionales y Seguridad Nacional.

Los rovers MER-A y MER-B (conocidos actualmente como los rovers Spirit y Opportunity) pueden encontrar la posición del sol y navegar sus propias rutas a destinos sobre la marcha a través de:

- Cartografía de la superficie mediante visión en 3D
- Cálculo de zonas seguras e inseguras en la superficie dentro de ese campo de visión
- Cálculo de rutas óptimas en toda la zona segura hacia el destino deseado

- Conducción a lo largo de la ruta calculada
- La repetición de este ciclo hasta que el destino se alcanza, o no haya ninguna ruta conocida hacia el destino

El Rover de ESA en planificación, ExoMars Rover, es capaz de localización relativa basada en visión y localización absoluta para navegar de forma autónoma a trayectorias seguras y eficaces a objetivos a través de:

- La reconstrucción de modelos 3D del terreno que rodea al Rover con un par de cámaras estéreo
- La determinación de las zonas seguras e inseguras del terreno y la "dificultad" general para el Rover para navegar por el terreno
- Cálculo de caminos eficientes a través de la zona de seguridad hacia el destino deseado
- Conducir el Rover a lo largo del camino planeado
- La creación de un mapa de navegación de todos los datos de navegación anterior

El DARPA Grand Challenge y DARPA Urban Challenge han alentado el desarrollo de capacidades aún más autónomas para vehículos de tierra, mientras que este ha sido el objetivo demostrado por robots aéreos desde 1990 como parte de la AUVSI Internacional Aerial Robotics Competition.

2.3. SLAM

Para un robot autónomo, capaz de navegar por si mismo en un entorno, una de las tareas más desafiantes que puede realizar es la de estimar su posición y orientación en el ambiente en el cual navega, especialmente si no se cuenta con información previa sobre ese ambiente. De esto se trata SLAM, acrónimo en inglés para Simultaneous Localization And Mapping, o Localización y Mapeo Simultáneos.

2.3.1. Introducción

El problema de construcción de mapas y localización simultáneos pregunta si es posible que un vehículo autónomo comience en una ubicación desconocida dentro de un entorno desconocido y que construya de forma incremental un mapa de este entorno mientras que usa este mapa de forma simultánea para calcular de forma absoluta la localización o ubicación del vehículo.

La solución para este problema es, en múltiples aspectos, el "Santo Grial" de la comunidad de investigación de vehículos autónomos. Por ello, la principal ventaja de SLAM es que elimina la necesidad de infraestructuras artificiales o de poseer conocimiento topográfico a priori del entorno.

2.3.2. Historia

El problema general de SLAM ha sido objeto de considerable investigación desde el inicio de una comunidad de investigación de la robótica y de hecho antes de este en áreas como sistemas de navegación de vehículos tripulados y estudios geofísicos. Se han propuesto varios enfoques para abordar tanto el problema de SLAM y también problemas de navegación más simplificados donde se hacen disponible informaciones adicionales de mapa o de ubicación del vehículo.

En términos generales, estos enfoques adoptan una de tres filosofías principales. La más popular de ellas es la aproximación estimación-teoría o basada en el filtro de Kalman. La popularidad de este enfoque se debe a dos factores principales. En primer lugar, proporciona directamente tanto una solución recursiva para el problema de navegación y de una manera de computar estimaciones consistentes para la incertidumbre en ubicaciones de vehículos y referencias de mapas sobre la base de modelos estadísticos para el movimiento del vehículo y observaciones relativas de referencias. En segundo lugar, un corpus sustancial del método y la experiencia ha sido desarrollado en el sector aeroespacial, marítimo y otras aplicaciones de navegación, de las que la comunidad autónoma de vehículos puede extraer.

Una segunda filosofía es la de evitar la necesidad de estimaciones de posición absoluta y de medidas precisas de la incertidumbre y en lugar de ello, emplear

conocimiento más cualitativo de la ubicación relativa de las referencias y el vehículo para construir mapas y guiar el movimiento. El enfoque cualitativo para la navegación y el problema general de SLAM tiene muchas ventajas potenciales sobre la metodología de estimación-teoría en términos de limitar la necesidad de modelos precisos y los requisitos computacionales resultantes, y en su significativo "atractivo antropomórfico".

La tercera y muy amplia filosofía, elimina el filtro de Kalman o el riguroso formalismo estadístico al tiempo que conserva un enfoque esencialmente numérico o computacional para el problema de navegación y SLAM. Tales enfoques incluyen el uso de pareos de lugares de interés icónicos, el registro de un mapa global, regiones delimitadas y otras medidas para describir la incertidumbre. Un trabajo notable en esta filosofía se ha realizado mediante el uso de un enfoque bayesiano para mapeo de edificios que no asume las distribuciones de probabilidad de Gauss como es requerido por el filtro de Kalman. Esta técnica, aunque muy eficaz para la localización con respecto a los mapas, no se presta para proporcionar una solución gradual a SLAM, donde un mapa se construye gradualmente a medida que se recibe información de los sensores. [6]

2.4. Visión por Computadora

Es el campo de la Inteligencia Artificial enfocado a que las computadoras puedan extraer información a partir de imágenes, ofreciendo soluciones a problemas del mundo real. La visión para los humanos no es ningún problema, pero para las máquinas es un campo muy complicado. Influyen texturas, luminosidad, sombras, objetos complejos, etc.

El propósito de la visión artificial o por computadora, es programar un computador para que "entienda" una escena o las características de una imagen.

Los objetivos típicos de la visión artificial incluyen:

- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo, caras humanas).
- La evaluación de los resultados (por ejemplo, segmentación, registro).

- Registro de diferentes imágenes de una misma escena u objeto, es decir, hacer concordar un mismo objeto en diversas imágenes.
- Seguimiento de un objeto en una secuencia de imágenes.
- Mapeo de una escena para generar un modelo tridimensional de la escena; este modelo podría ser usado por un robot para navegar por la escena.
- Estimación de las posturas tridimensionales de humanos.
- Búsqueda de imágenes digitales por su contenido.

Estos objetivos se consiguen por medio de reconocimiento de patrones, aprendizaje estadístico, geometría de proyección, procesamiento de imágenes, teoría de grafos y otros campos. La visión artificial cognitiva está muy relacionada con la psicología cognitiva y la computación biológica.

2.5. Microsoft Kinect

Kinect para Xbox 360, o simplemente Kinect (originalmente conocido por el nombre en clave "Project Natal"), es un controlador de juego libre y entretenimiento creado por Alex Kipman, desarrollado por Microsoft para las videoconsolas Xbox 360 y Xbox One, y desde junio del 2011 para PC a través de Windows 7 y Windows 8.

Kinect permite a los usuarios controlar e interactuar con la consola sin necesidad de tener contacto físico con un controlador de videojuegos tradicional, mediante una interfaz natural de usuario que reconoce gestos, comandos de voz, y objetos e imágenes. El dispositivo tiene como objetivo primordial aumentar el uso de la Xbox 360, más allá de la base de jugadores que posee en la actualidad.

En sí, Kinect compite con los sistemas Wiimote con Wii MotionPlus y PlayStation Move, que también controlan el movimiento para las consolas Wii y PlayStation 3, respectivamente.

El nombre en clave "Proyecto Natal" responde a la tradición de Microsoft de utilizar ciudades como nombres en clave. Alex Kipman, director de Microsoft, quien incubó el proyecto, decidió ponerle el nombre de la ciudad brasileña Natal como un homenaje a su país de origen y porque la palabra natal significa "de o en relación al nacimiento",

lo que refleja la opinión de Microsoft en el proyecto como “el nacimiento de la próxima generación de entretenimiento en el hogar”. Poco antes de la E3 2010 varios weblogs tropezaron con un anuncio que supuestamente se filtró en el sitio italiano de Microsoft de que sugirió el título “Kinect”, que confirmó más tarde.

2.5.1. Características

El sensor de Kinect es una barra horizontal de aproximadamente 23 cm (9 pulgadas) conectada a una pequeña base circular con un eje de articulación de rótula, y está diseñado para ser colocado longitudinalmente por encima o por debajo de la pantalla de vídeo.

El dispositivo cuenta con una cámara RGB, un sensor de profundidad, un micrófono de múltiples matrices y un procesador personalizado que ejecuta el software patentado, que proporciona captura de movimiento de todo el cuerpo en 3D, reconocimiento facial y capacidades de reconocimiento de voz.

El sensor contiene un mecanismo de inclinación motorizado y en caso de usar una PC o un modelo original de Xbox 360, tiene que ser conectado a una toma de corriente, ya que la corriente que puede proveerle el cable USB es insuficiente; para el caso del modelo de Xbox 360 S esto no es necesario ya que esta consola cuenta con una toma especialmente diseñada para conectar el Kinect y esto permite proporcionar la corriente necesaria que requiere el dispositivo para funcionar correctamente.

El sensor de profundidad es un proyector de infrarrojos combinado con un sensor CMOS monocromo que permite a Kinect ver la habitación en 3D en cualquier condición de luz ambiental. El rango de detección de la profundidad del sensor es ajustable gracias al software de Kinect capaz de calibrar automáticamente el sensor.

2.5.2. Especificaciones Técnicas

Sus especificaciones más relevantes son:

- Cámara / sensor infrarrojo: Microsoft / X853750001 / VCA379C7130 / MT9M001

- Rango efectivo (aproximado): 1,2m – 3,5m (puede ser menor o mayor, dependiendo de las condiciones ambientales)
- Resolución: 640x480 píxeles @ 30 Hz (profundidad de 11 bits: 2048 niveles de sensibilidad)
- Cámara RGB: VNA38209015 / MT9M112 / MT9v112
- Resolución: 640x480 píxeles @ 30 Hz (VGA de 8 bits)
- Emisor / proyector infrarrojo: OG12 / 0956 / D306 / JG05A
- Proyector láser de 830 nm.
- Potencia de salida: 60m̃W

2.5.3. Requerimientos para uso en Robótica

Llenar.

2.6. Desarrollo Ágil de Proyectos

En todo proyecto, el objetivo deseado es (o debería ser) el de maximizar la eficiencia y disminuir el costo, ya sea monetario o de tiempo. Para lograr este objetivo se han propuesto diferentes formas de administrar o gerenciar los proyectos, con infinidad de niveles de control, desde prácticamente ningún control en lo absoluto, hasta la microgerencia de las tareas más mínimas posibles.

El desarrollo de un trabajo especial de grado, tesis, o proyecto de grado puede verse, tal como se denominó por último, como un proyecto en si mismo, donde el objetivo es cumplir con los objetivos generales y específicos dentro de un marco de tiempo determinado, por lo que puede aplicarse lo mencionado al inicio para cumplir con los objetivos de la mejor manera posible.

Así pues, en el área de la ingeniería de sistemas y el desarrollo de software, se han empleado igualmente innumerables métodos para llevar a cabo la gerencia de un proyecto; sin que esto pretenda convertirse en un estudio a profundidad de dichos métodos, se nombra a continuación uno de los más populares en la actualidad, junto algunos métodos que emplean su filosofía y cuyo fin no es otro que el de organizar el

desarrollo del proyecto y así poder rendir cuentas del mismo.

Las metodologías Ágiles de desarrollo, fueron propuestas en el año 2001 por diversos representantes de múltiples metodologías de desarrollo, tales como SCRUM, Programación Extrema, DSOM, Desarrollo de Software Adaptativo, Crystal y otros, con el fin de encontrar un terreno común que involucrara una alternativa al desarrollo de software pesado y orientado a la documentación.

Su idea consistió en enfocar el proceso de desarrollo en las personas en vez de al proceso en sí, reduciendo la burocracia y los procesos de planificación, documentación y modelado al mínimo posible, a través de principios documentados en un "manifiesto".

2.6.1. Principios del Desarrollo Ágil

El manifiesto para el Desarrollo Ágil de Software, indica el enfoque a seguir por esta metodología. Sus valores principales son los siguientes:

- Individuos e interacciones sobre procesos y herramientas
- Software funcionando sobre documentación extensiva
- Colaboración con el cliente sobre negociación contractual
- Respuesta ante el cambio sobre seguir un plan

En las propias palabras del manifiesto: "Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda".[7]

Los principios producto de estos valores son:[8]

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

- Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.
- Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

2.7. Personal Extreme Programming

La programación extrema o *eXtreme Programming* (XP) es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia, *Extreme Programming Explained: Embrace Change* (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.

Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

2.7.1. Características

Las características fundamentales del método son:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación. Véase, por ejemplo, las herramientas de prueba JUnit orientada a Java, DUnit orientada a Delphi, NUnit para la plataforma.NET o PHPUnit para PHP. Estas tres últimas inspiradas en JUnit, la cual, a su vez, se inspiró en SUnit, el primer framework orientado a realizar tests, realizado para el lenguaje de programación Smalltalk.
- Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. La mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata. En este sentido, la modalidad personal difiere por razones obvias; sin embargo, las demás características son perfectamente aplicables.
- Frecuente integración del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve

el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

- Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Cuanto más simple es el sistema, menos tendrá que comunicar sobre éste, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

2.8. Kanban

Kanban, del japonés (かんばん(看板)), que es literalmente "letrero" o "cartelera", es un sistema de programación para la producción *Lean* y justo-a-tiempo.

Kanban se basa en una idea muy simple: el trabajo en curso (Work In Progress, WIP) debería limitarse, y sólo deberíamos empezar con algo nuevo cuando un bloque de trabajo anterior haya sido entregado o ha pasado a otra función posterior de la cadena. El Kanban (o tarjeta señalizadora) implica que se genera una señal visual para indicar que hay nuevos bloques de trabajo que pueden ser comenzados porque el trabajo en curso actual no alcanza el máximo acordado.

Kanban usa un mecanismo de control visual para hacer seguimiento del trabajo conforme este viaja a través del flujo de valor. Típicamente, se usa un panel o pizarra con notas adhesivas o un panel electrónico de tarjetas. Las mejores prácticas apuntan probablemente al uso de ambos.

Las metodologías Ágiles han obtenido buenos resultados proporcionando transparencia respecto al trabajo en curso y completado, así como en el reporte de métricas como la velocidad (cantidad de trabajo realizada en una iteración). Kanban

sin embargo va un paso más allá y proporciona transparencia al proceso y su flujo. Kanban expone los cuellos de botella, colas, variabilidad y desperdicios. Todas las cosas que impactan al rendimiento de la organización en términos de la cantidad de trabajo entregado y el ciclo de tiempo requerido para entregarlo. Kanban proporciona a los miembros del equipo y a las partes interesadas visibilidad sobre los efectos de sus acciones (o falta de acción). De esta forma, los casos de estudios preliminares están demostrando que Kanban cambia el comportamiento y motiva a una mayor colaboración en el trabajo. La visibilidad de los cuellos de botella, desperdicios y variabilidades y su impacto también promueve la discusión sobre la posibles mejoras, y los equipos comienzan rápidamente a implementar mejoras en su proceso.[9]

2.8.1. Trello

Trello es una plataforma de software que utiliza el paradigma Kanban para el manejo de proyectos. Iniciado en el año 2011 por la compañía Fog Creek Software y caracterizado como software de productividad, este utiliza un sistema de tableros, listas y tarjetas para llevar el control de múltiples tipos de proyecto, sin estar limitado a proyectos de desarrollo de software, por lo cual es altamente versatil. Las tarjetas aceptan comentarios, archivos adjuntos, votos, fechas de entrega y listas de verificación.

Para este proyecto, se abrió un tablero en Trello accesible desde la dirección `https://trello.com/b/yIMdcTCR/tesis-ros-kinect`, cuyas listas representan un híbrido entre las disponibles en PXP y Kanban, de la siguiente manera:

- Pendientes: Tareas que están por ser ejecutadas.
- Actuales: Tareas que están siendo ejecutadas actualmente.
- Entregadas: Tareas que están marcadas como realizadas, pero que no han sido verificadas aún.
- Rechazadas: Tareas que, habiendo sido entregadas, presentan alguna falla o requieren algún cambio, por lo cual se colocan en esta lista para ser revisadas.
- Listas: Tareas que han sido verificadas y aceptadas.
- Ideas: Tarjetas que representan ideas aplicar, o tareas que no pueden pasar a la lista “Pendientes” por no haber recursos disponibles para ellas.

—-Insertar imagen del tablero de Trello del proyecto de grado—-

Uso

Cada tarea a realizar se coloca como una tarjeta en la lista correspondiente a “Pendientes” o “Ideas” según sea el caso, y una vez estén disponibles los recursos para tomarla (siendo estos recursos tiempo, o disponibilidad de la o las personas involucradas) se pasa dicha tarjeta, mediante arrastrar y colocar, a la lista de “Actuales”. Esto permite ver el flujo de trabajo, tal como se mencionaba anteriormente, y permite saber el estado actual del proyecto si la tabla se mantiene actualizada.

Capítulo 3

Software de Control Robótico

Este capítulo pretende ahondar en las características de las plataformas o software de control robótico, comenzando por su definición, la exposición de las características más comunes, los criterios bajo los cuales se selecciona una lista de posibles plataformas para el desarrollo del proyecto y finalmente, la justificación de la plataforma de software escogida.

3.1. Definición

Llenar.

3.2. Características

Llenar.

3.3. Criterios de Selección

Llenar.

3.4. Plataformas de Software Consideradas

Llenar.

3.5. Plataforma de Software Seleccionada

Llenar.

3.6. Justificación

Llenar.

Capítulo 4

ROS

En este capítulo se desea profundizar en la definición de ROS (Robot Operating System) como plataforma de software seleccionada para el desarrollo de este proyecto de grado. Asimismo, se describen sus características principales, la arquitectura de software que utiliza, la instalación del mismo en el entorno de desarrollo a utilizar, y los módulos en los cuales se apoya este proyecto para llevar a cabo la generación de mapas de entorno.

4.1. Definición

Llenar.

4.2. Arquitectura

Llenar.

4.3. Características Principales

Llenar.

4.4. Requisitos de Instalación

Llenar.

4.5. Procedimiento de Instalación

Llenar.

4.6. Módulos Disponibles para SLAM

Llenar.

Capítulo 5

Generación de Mapas a través de RTAB-Map

En este capítulo se describe el software RTAB-Map, sus características, su proceso de funcionamiento, su instalación y uso en el entorno de desarrollo en sus presentaciones como software independiente o como módulo integrado a ROS y por último, la generación de un mapa de entorno a través del mismo.

5.1. Definición

Llenar.

5.2. Características

Llenar.

5.3. Funcionamiento

Llenar.

5.4. Instalación

Llenar.

5.4.1. Como software independiente

Llenar.

5.4.2. Como módulo de ROS

Llenar.

5.5. Generación de Mapas de Entorno

Llenar.

5.5.1. Requerimientos

Llenar.

5.5.2. Procedimiento

Llenar.

5.5.3. Pruebas

Llenar.

Generación de mapas 3D

Llenar.

Generación de mapas 2D

Llenar.

Selección de altura mínima y máxima para generación del mapa 2D

Capítulo 6

Conclusión y Recomendaciones

6.1. Conclusión

Llenar.

6.2. Recomendaciones

Llenar.

Bibliografía

[1] "Lasdai," noviembre, 2013. [Online]. Disponible: http://www.ing.ula.ve/lasdai/

[2] R. Szeliski, "Computer vision: Algorithms and applications," 7 septiembre, 2009. [Online]. Disponible: http://mesh.brown.edu/engn1610/szeliski/01-Introduction.pdf

[3] L. Roberts, "Machine perception of 3-d solids," en *OE-OIP65*, 1965, páginas 159–197.

[4] W. S. Humphrey. (2005, 3) Psp(sm): A self-improvement process for software engineers. [Online]. Disponible: http://amazon.com/o/ASIN/B001EWOG8A/

[5] Y. Dzhurov, I. Krasteva, y S. Ilieva, "Personal extreme programming - an agile process for autonomous developers," en *Proceedings of International Conference on SOFTWARE, SERVICES & SEMANTIC TECHNOLOGIES*, Octubre 2009, página 252.

[6] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, y M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *Robotics and Automation, IEEE Transactions on*, tomo 17, nº 3, páginas 229–241, Jun 2001.

[7] Manifiesto por el desarrollo ágil de software. [Online]. Disponible: http://agilemanifesto.org/iso/es/

[8] Principios del manifiesto ágil. [Online]. Disponible: http://agilemanifesto.org/iso/es/principles.html

[9] H. Kniberg, *Kanban and Scrum - making the most of both (Enterprise Software Development)*. lulu.com, 3 2010. [Online]. Disponible: http://amazon.com/o/ASIN/0557138329/

[10] "Hokuyo laser scanning range finder pbs-03jn — robots in search," octubre 2013. [Online]. Disponible: http://www.robotsinsearch.com/hokuyo-pbs03jn-scanning-infrared-obstacle-detection-sensor-p-882.html

[11] "Lms-200-30106 by sick optic electronic," Octubre 2013. [Online]. Disponible: http://www.plccenter.com/en-US/Buy/SICK%20OPTIC%20ELECTRONIC/LMS20030106

[12] M. Labbe y F. Michaud, "Online global loop closure detection for large-scale multi-session graph-based slam," en *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, páginas 2661–2666.

[13] ——, "Appearance-based loop closure detection for online large-scale and long-term operation," *IEEE Transactions on Robotics*, tomo 29, nº 3, páginas 734–745, 2013.