Sistema de localización para robots móviles de bajo costo utilizando marcas de referencia artificiales en ambientes de interiores

Rafael Ignacio Zurita[1], Alejandro Mora[1], Candelaria Alvarez[1], y Miriam Lechner[1]
[1] Departamento de Ingeniería de Computadoras,
Facultad de Informática, Universidad Nacional del Comahue
{rafa, alejandro.mora, candelaria.alvarez, mtl}@fi.uncoma.edu.ar

Resumen. La localización es un requisito de la navegación con mapas de robots móviles. Usualmente se implementa con múltiples sensores y gran poder de cómputo, por lo que no suele ser una capacidad de los robot móviles de bajo costo. Se propone el diseño e implementación de un sistema prototipo que reporta su localización utilizando un hardware de mínimas prestaciones y marcas artificiales en el terreno. Se realizaron pruebas en un ambiente de interior, obteniendo desde el prototipo repetidas localizaciones, para evaluar la precisión y estimar la exactitud. En base a los resultados se concluye que el prototipo propuesto es capaz de determinar ubicaciones con una precisión menor a 10cm, y a una frecuencia de 2Hz; lo que permite plantear la posibilidad, en trabajos futuros, de incluir navegación con mapas a una clase de robots que hasta hoy no suele utilizar esta característica.

Palabras claves: Robot móvil, localización, navegación, sistema embebido, AprilTag.

1 Introducción

La localización es uno de los componentes requeridos para la navegación con mapas de los robots móviles, y requiere que el robot sea capaz de determinar su ubicación y orientación exacta en el ambiente en el cual se encuentre. Con esta información y un marco de referencia virtual que represente el ambiente real, el robot puede planificar una ruta y desplazarse para cumplir con sus objetivos [1].

En la actualidad, los robots móviles capaces de realizar navegación en interiores son diseñados y construidos para tareas complejas. Suelen trabajar en ambientes industriales, para desplazar grandes cargas dentro de galpones o edificaciones. También en ambientes públicos, por ejemplo en hoteles, donde los robots suelen navegar desde la recepción a diferentes habitaciones, realizando la entrega de suministros solicitados por huéspedes [2,3]. En estas clases de robots el costo y/o consumo energético usualmente no es una restricción primordial, por lo que suelen contar con sensores láser (LIDAR), visuales (cámaras) y de orientación y velocidad (unidades de medición inercial) para recolectar datos del ambiente; y con

computadoras capaces de procesar grandes cantidades de información en tiempo real, lo que en conjunto permite conocer la ubicación del robot y navegar en sus ambientes de trabajo [4].

Por el contrario, los robots móviles de bajo costo utilizados en interiores poseen un microcontrolador o microprocesador de bajas prestaciones como sistema central de control. Estos robots realizan tareas sencillas, tales como aspirar el polvo en un living (domésticos), o realizar movimientos simples (adelante, izquierda, etc) al ejecutar programas desarrollados por estudiantes (robótica educativa). Para ello utilizan sensores que permiten una interacción básica con el entorno, por ejemplo para evitar el choque con obstáculos o moverse dentro de un cerco invisible. Los sensores en esta clase de robots pueden detectar el objeto a evitar cuando se está a una distancia muy corta, devolviendo información básica que puede ser discretizada para indicar si existe o no un objeto frente al sensor. Por tal motivo estos robots realizan únicamente navegación reactiva (sin utilización de mapas), ya que no son capaces de determinar su ubicación [5].

En este artículo se presenta la arquitectura de un sistema embebido prototipo, que reporta su localización utilizando un microprocesador de bajas prestaciones y marcas artificiales en el terreno. Está diseñado para ser integrado fácilmente a robots móviles de bajo costo, con el fin de que esta categoría de robots pueda determinar su ubicación en ambientes de interiores. El sistema está basado en la adquisición de señales a través de una cámara de video, y la detección de marcas artificiales a través del software AprilTag. El dispositivo cuenta con una computadora de arquitectura MIPS de bajo consumo que controla el sensor de captura, procesa la imágenes con AprilTag, y calcula la localización. Por lo que las contribuciones principales de este trabajo son dos:

- Una aplicación para sistemas embebidos portable (desarrollada en C sin ninguna dependencia de bibliotecas externas) que captura imágenes de una cámara, y reporta la localización utilizando AprilTag.
- La validación de la precisión del sistema a través de la experimentación del mismo en un hardware de mínimas prestaciones.

El resto de este trabajo está estructurado de la siguiente manera: en la sección 2 se presentan trabajos relacionados; en la sección 3 se presenta la arquitectura del sistema propuesto, con énfasis en el algoritmo de localización; en la sección 4 se exponen los resultados obtenidos de la evaluación del sistema, al experimentar con un robot de bajo costo para la obtención de mediciones de localización; finalmente, en la sección 5 se detallan las conclusiones y trabajos futuros.

2 Trabajos Relacionados

Una solución a la localización es a través de marcas artificiales en el terreno (del inglés artificial landmark), que son colocadas para que puedan ser unívocamente identificadas de manera fácil y precisa. Si bien con este método se puede evitar el uso de varios de los sensores que habitualmente se utilizan en localización, aún requiere

visión por computadora para reconocer la ubicación y orientación de la marca artificial. Los algoritmos de visión por computadora para la detección de estas marcas fiduciales suelen implementarse y verificarse en computadoras con recursos suficientes para procesar un gran número de imágenes por segundo, y permitir el uso de un amplio rango de bibliotecas y aplicaciones al mismo tiempo. Por ejemplo, opencv para la captura y filtrado de imágenes, y el entorno matlab para programación. Esto posibilita el prototipado rápido y la validación de los algoritmos de visión en la misma computadora. En [6] se presenta el diseño de AprilTag para su uso en robótica y realidad aumentada. La validación se realizó en una computadora con una CPU Intel Xeon E5-2640 de 2.5GHz. Se menciona que es posible lograr una buen rendimiento del sistema en un equipo iPhone, sin especificar un modelo particular. En [7] se implementa un aterrizaje autónomo de un drone a una estación en movimiento, utilizando una computadora con un Nvidia Tegra K1 SoC, CPU ARM A-15 de 2.3GHz, y 8GB de DDR3 RAM para la detección de marcas de referencias con AprilTag. En [8] se presenta el diseño de un sistema para localización en interiores, utilizando también AprilTag. La experimentación se llevó a cabo mediante un robot móvil Pioneer 3-DX, que contó con una notebook empotrada con procesador Intel Core 2 Due 2.0 GHz y 2G de RAM. En [9] se presenta un sistema capaz de ser utilizado en el control de acercamiento de dos barcos para el traspaso de cargas entre embarcaciones. Se utilizó para el procesamiento de las marcas de referencia con AprilTag una notebook Lenovo W540 con un procesador Intel Quad-Core i7-4900MQ de 2.80GHz CPU y una NVIDIA Quadro K2100M para la rectificación de las imágenes. En [10] se analizan diferentes sistemas de detección de marcas de referencia y se propone uno mixto. Los experimentos se realizaron utilizando una computadora con procesador i7-7600U a 2.80GHz para el procesamiento de las imágenes. En [11] se controlan brazos manipuladores industriales basando sus movimientos a la posición de los objetos a manipular. A estos últimos se les añadieron marcas de referencia de AprilTag para estimar su posición. La ejecución de este sistema en los experimentos se realizó en una PC embebida con un procesador -i7 quad core, y 8GB de RAM.

A diferencia de los casos expuestos nuestro trabajo se focaliza en la implementación y validación de un sistema embebido que pueda reportar la localización de robots móviles de bajo costo, utilizando un hardware de mínimas prestaciones y el software AprilTag 3 para la detección de marcas de referencia. No hemos encontrado otros trabajos que evalúen el rendimiento o la precisión de AprilTag 3 en sistemas mínimos.

3 Arquitectura

3.1 Arquitectura de Hardware

La arquitectura de hardware del prototipo presenta dos componentes principales:

- El dispositivo de cámara rotativa;
- El hardware de procesamiento embebido

El dispositivo de cámara rotativa consta de una cámara USB genérica montada sobre un servo motor, el cual permite rotar a la cámara de 0 a 360º, con una resolución de giro de un grado. La señal de control para girar la cámara es una señal individual de modulación por ancho de pulsos (PWM), con una frecuencia estándar de 50hz para este dispositivo. La cámara tiene una interfaz USB 2.0 y es un dispositivo de clase UVC. Ambas interfaces están conectadas al hardware embebido.

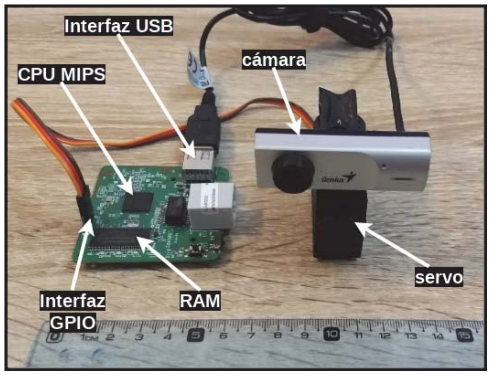

Fig. 1. Dispositivo de cámara rotativa y hardware de procesamiento embebido.

El hardware embebido es la computadora en donde se ejecuta el sistema de localización. Contiene un System On Chip (SOC) Atheros AR9331, con una CPU MIPS de 400Mhz. La memoria principal (RAM) es de 32MB. También cuenta con puertos de E/S (GPIO), un puerto USB 2.0, una interfaz UART y una interfaz Wireless WiFi 802.11n/g/b. El tamaño del módulo, que puede observarse en la Fig. 1, es de aproximadamente 6x6x9cm, y tiene un peso total de 120g. Los costos de los componentes se enumeran a continuación:

1. Computadora MIPS: USD 32.85
2. Servo motor DS04-NFC: USD 4.54
3. Camara USB genérica: USD 5

3.2 Arquitectura de Software

El hardware está controlado por un sistema operativo Linux construído con buildroot[1], destinado a sistemas embebidos. Los controladores de hardware que se utilizan en el dispositivo presentado son el universal video class (UVC) para la captura de imágenes desde la cámara, y el de un puerto de E/S general (GPIO). Adicionalmente se integró también un driver en espacio de usuario de modulación por ancho de pulsos (PWM) (que utiliza a bajo nivel el driver GPIO). Este driver controla un servo motor para rotar la cámara. Cuando se requiere una localización la aplicación busca primero la marca de referencia. Para esto va rotando la cámara 45 grados y capturando imágenes, analizando cada una para detectar la marca de referencia. Cuando se detecta la marca

[1] https://buildroot.org/

de referencia en una imagen, el sistema detiene la rotación de la cámara, y se procede a realizar el proceso de localización, el cual consta de tres etapas, presentadas en la Fig. 2.

En la primera etapa del proceso la aplicación embebida captura una imagen desde la cámara, utilizando la interfaz en espacio de usuario del driver UVC del sistema Linux. El formato de la imagen digital recibida es JPEG y la resolución es de 320x240 píxeles.

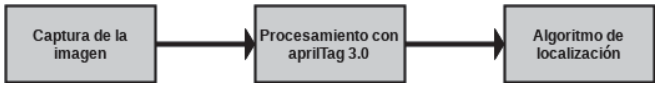

Fig. 2. Etapas del proceso de localización.

En la segunda etapa el sistema procesa la señal obtenida (imagen JPEG) con el software AprilTag. Este software es un sistema visual fiducial, útil para una amplia variedad de tareas, incluida la realidad aumentada, la robótica y la calibración de la cámara [10]. Es capaz de reconocer marcas de referencia artificiales llamadas etiquetas (tags), las cuales son similares a códigos QR pero con menor cantidad de bits, lo que permite una detección más rápida que los sistemas QR tradicionales. Otra ventaja de AprilTag es que presenta un bajo porcentaje de falsas detecciones (falsos positivos) aún con cambios en la iluminación del ambiente, lo cual es una característica deseable en los sistemas de visión por computadora, que suelen presentar inconvenientes ante dicho factor.

En esta etapa AprilTag analiza la imagen, y reporta la posición y orientación de las etiquetas identificadas. Como en este trabajo se experimenta con una única marca de referencia en el ambiente, se obtiene como salida una única matriz de rotación y un vector de traslación de la marca detectada, ubicados en el sistema de coordenadas de la cámara, con origen el centro de la lente focal.

La última etapa, Algoritmo de localización, realiza el cálculo de la posición de la cámara en el sistema de coordenadas del mundo real (sistema de coordenadas de la marca de referencia) y consta de tres sub-etapas:

1. Se obtiene el ángulo euler pitch a partir de la matriz de rotación obtenida desde AprilTag. Para su cálculo se utilizan funciones de la biblioteca eigen[2], las cuales se incorporaron como parte del código de la aplicación. En la Fig. 3 (a) 1 se puede observar que el ángulo euler pitch obtenido α (alfa) es el ángulo formado por el eje *X Cam* del sistema de coordenadas de la cámara y el eje *X tag* del sistema de coordenadas del tag.
2. Luego, a partir del punto (*Xc*,*Yc*) provisto por AprilTag (Fig. 3 (a)) se obtiene el punto *Xc'* e *Yc'* en el sistema de coordenadas del mundo real (Fig. 3 (b) 2), donde *Xc'*=*Xc* y *Yc'*=*Yc* (sus valores algebraicos son equivalentes).
3. Finalmente, se realiza una rotación del punto (*Xc'*, *Yc'*), la cantidad de grados α obtenido en el paso 1. Para esto se define un vector columna a partir de *Xc'* e *Yc',* y se multiplica por una matriz de rotación calculada a partir del ángulo alfa α:

[2] `http://eigen.tuxfamily.org`

$$\begin{bmatrix} Xt \\ Yt \end{bmatrix} = \begin{bmatrix} cos(\alpha) & -sen(\alpha) \\ sen(\alpha) & cos(\alpha) \end{bmatrix} \begin{bmatrix} Xc' \\ Yc' \end{bmatrix}$$

Se obtiene entonces el punto *(Xt, Yt)* (Fig. 3 (b) 3), a partir de la multiplicación recién descrita:

$$Xt = cos(\alpha).Xc' - sen(\alpha).Yc'$$
$$Yt = sen(\alpha).Xc' + cos(\alpha).Yc'$$

El punto *(Xt, Yt)* (Fig. 3 (b) 3) indica la posición de la cámara en el sistema de coordenadas de la marca de referencia, es decir, la posición del dispositivo portátil en el escenario o mundo real.

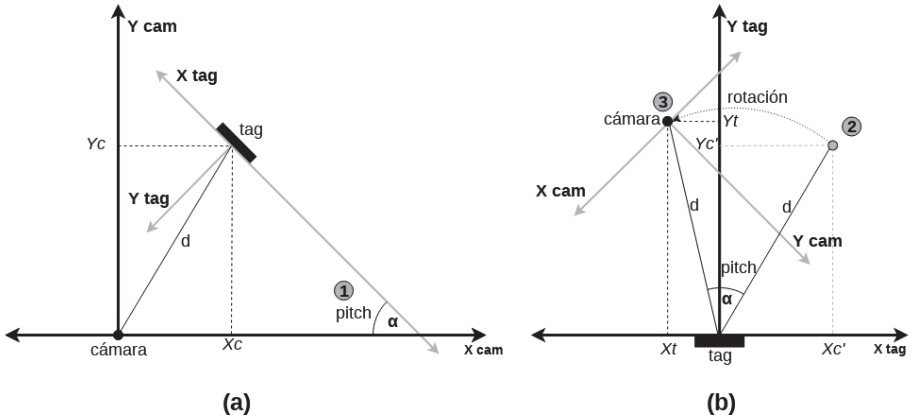

Fig. 3. (a) posición de la marca de referencia en el sistema de coordenadas de la cámara. (b) posición de la cámara en el sistema de coordenadas del mundo real (la marca de referencia).

4 Resultados

Para validar el prototipo se evaluó el tiempo, y precisión de las localizaciones reportadas por el sistema propuesto.

4.1 Tiempo de ejecución

En la etapa de obtención de requerimientos de este trabajo se tomó como referencia los robots móviles de bajo costo utilizados por la Facultad de Informática en proyectos de robótica educativa [12]. El robot Frankestito se desplaza a una velocidad máxima de 12cm/seg, y se estimó que, para futuras funcionalidades de navegación el robot educativo, debería indicar su ubicación aproximadamente cada 20 centímetros. Esto requiere una localización cada 1,6 segundos en este robot, por lo que una frecuencia de 1Hz sería suficiente si el error de la localización es +/- 12cm.

Se utilizó la función de la biblioteca `libc gettimeofday()` para evaluar el tiempo transcurrido de cada etapa del sistema propuesto (tiempo de E/S y tiempo de ejecución) y corroborar que el tiempo de localización está dentro de los requerimientos. En una prueba de 415 mediciones utilizando el hardware propuesto se obtuvo (como se observa en la Tabla 1) una frecuencia general de localizaciones de 2Hz, y una frecuencia para el caso más desfavorable de 1Hz.

Tabla 1. Tiempos promedios de E/S y CPU en el dispositivo prototipo en 415 localizaciones.

Etapa	Tipo	Tiempo (media aritmética)	Tiempo máx. (peor caso)
Captura	Tiempo de E/S	74 (ms)	95 (ms)
Detección con AprilTag	Tiempo de ejecución	292 (ms)	750 (ms)
Algoritmo de localización	Tiempo de ejecución	41 (ms)	46 (ms)
TOTAL	Tiempo de E/S + CPU	407 (ms)	891 (ms)

4.2 Precisión

Para las pruebas se delimitó un escenario real de navegación de 2mts x 2mts, con una cuadrícula en el terreno compuesta de mosaicos de 33cm x 33cm. Se colocó una marca de referencia (tag en la terminología de AprilTag) en este ambiente. El dispositivo prototipo fue entonces montado sobre un robot móvil de bajo costo, y se calibró la altura de la lente de la cámara para que se encuentre a la misma altura que el centro del tag, como se observa en la Fig. 4. Para la comunicación entre la CPU del prototipo y el microcontrolador del robot móvil se interconectó la interfaz serial UART de ambos.

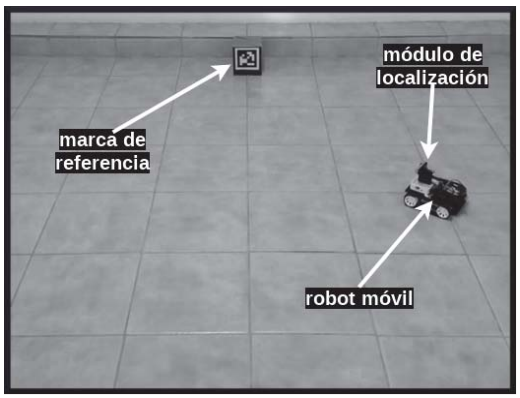

Fig. 4. Escenario real en donde se realizaron las mediciones con el dispositivo propuesto.

Durante el experimento se utilizó una marca de referencia (tag) de 20cm x 20cm, y se fue situando al robot móvil en cada esquina de la cuadrícula del ambiente, cuyas localizaciones exactas son conocidas a priori. En cada ubicación se realizaron aproximadamente 10 mediciones desde el dispositivo prototipo y se almacenaron los resultados para su análisis. El conjunto fue graficado en una cuadrícula que representa

el escenario real (Fig. 5). La marca de referencia (tag) se encontraba en X=99 e Y=0 de esta representación. Cada punto representa una localización reportada por el sistema.

Se observa que hasta una distancia en Y de 125cm el robot reporta localizaciones muy cercanas al valor real de la ubicación. El valor exacto en cada medición (la ubicación real) está representada en la Figura por cada intersección de la cuadrícula. Las mediciones son también muy precisas (las 10 mediciones en cada ubicación están concentradas). Luego, a más de 175cm de distancia (Y=175) la precisión decae en el eje x mucho más que en el eje y (puede notarse en la figura que los puntos de cada medición tienen una dispersión mayormente horizontal). En esta zona del escenario la distancia máxima (diferencia) entre mediciones de una misma ubicación es de hasta aproximadamente 20cm en el eje X, y de 10cm en el eje Y. Finalmente, a 2mts de distancia, las mediciones ya no se concentran alrededor de la ubicación real, por lo que estas mediciones sólo podrían utilizarse como un cálculo estimativo de la zona donde se encuentre el robot, y ya no como una localización exacta.

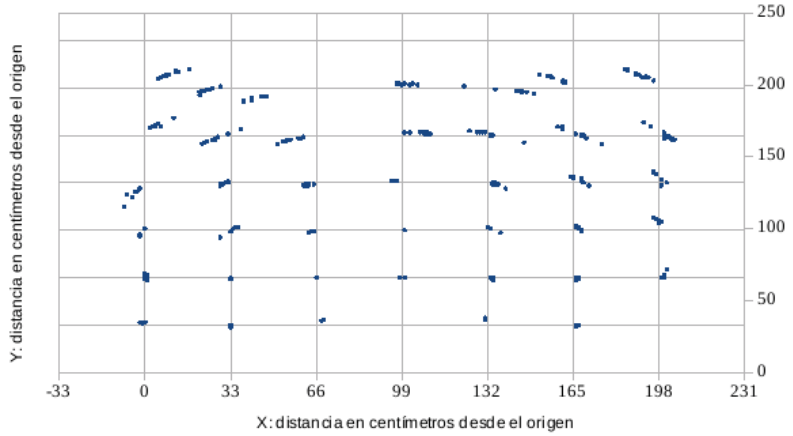

Fig. 5. Relación entre las localizaciones reportadas por el dispositivo prototipo (puntos) y la ubicación real (intersecciones de la cuadrícula).

Cada localización es una ubicación en un plano 2D con ejes (X, Y). En la Figura 6 se presentan dos histogramas (uno para el eje X y otro para Y) de las distancias de las localizaciones con respecto al valor de la ubicación real.

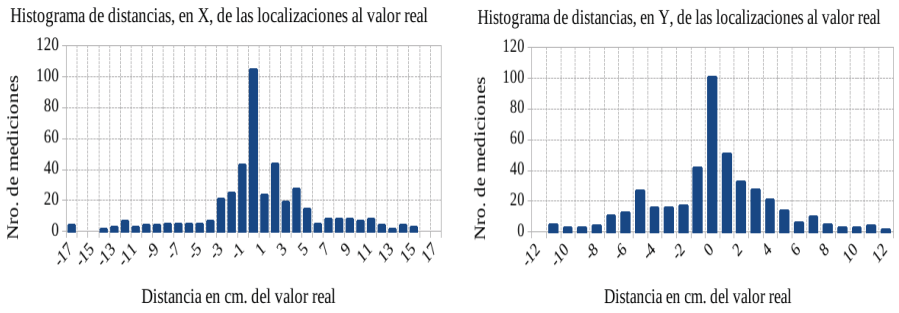

Fig. 6. Histograma de las distancias en centímetros de las localizaciones con respecto al valor real.

El valor x=0 en ambos es la distancia mínima con respecto valor estimativo real (ubicaciones conocidas previo al experimento), y representa las mediciones más exactas. Se observa que al menos unas 100 mediciones coincidieron con la ubicación real (de un total de 415), y también, que un gran número de mediciones están comprendidas en el rango [-5,5] cm. Su justificación puede apreciarse en las estadísticas de la Tabla 2.

Tabla 2. Estadística de las distancias de localizaciones con respecto a su valor estimativo real.

Estadística	Valor
Cantidad de mediciones	415
Media del error (distancia media de las localizaciones al valor real de la ubicación)	4,971cm
Varianza	28,985
Desvío estándar	5,383
Error estándar del promedio	0,264
Resultado numérico de las mediciones (distancia al valor estimativo real)	(4,971 +/- 0,264) cm
Porcentaje de mediciones del experimento dentro del desvío estándar	65%

5 Conclusiones y Trabajo Futuro

Se presentó en este trabajo la arquitectura de un dispositivo prototipo para reportar localizaciones de robots móviles de bajo costo en ambientes interiores, y utilizando marcas de referencia en el terreno. Se experimentos para medir la frecuencia, precisión y exactitud, cuando se lo utiliza en un hardware de mínimas prestaciones.

En base a los resultados se concluye que el hardware y software propuesto es capaz de localizar un robot de bajo costo a 2hz, y con una precisión de (4,971 +/-

0,264) cm el 65% de los casos, si el robot navega en los límites impuestos por: tamaño de la marca de referencia en el terreno y la resolución de la cámara utilizada. Los robots de bajo costo suelen trabajar en ambientes pequeños (por ej. una habitación), y si bien se experimentó con pequeñas marcas de referencia se estima (y se plantea para validación futura) que con un tamaño de tag de 40cm x 40cm sería suficiente en la mayoría de estos casos (por ej. para ambientes de hasta 16mt²).

Como trabajo futuro se espera continuar, también, con la implementación y validación de los componentes restantes de la navegación con mapas en robots móviles de bajo costo. Algunos ejemplos de estos son: representación de mapas, planificación de rutas, historial de trayectorias, y estimación de los errores de localización. También se pretende incrementar el rendimiento del módulo de localización, realizando principalmente mejoras en el software (por ejemplo, solapando la E/S con el tiempo de CPU).

Referencias

1. Roland Siegwart, Illah R. Nourbakhsh.: Introduction to Autonomous Mobile Robots. Massachusetss Institute of Technology. ISBN: 9780262195027. MIT Press (2004)
2. Melonee Wise, Michael Ferguson, Derek King, Eric Diehr and David Dymesich.: Fetch & Freight: Standard Platforms for Service Robot Applications. Fetch Robotics Inc. press (2018)
3. Michael Jae-Yoon Chung, Justin Huang, Leila Takayama, Tessa Lau, Maya Cakmak.: Iterative Design of a System for Programming Socially Interactive Service Robots. International Conference on Social Robotics. (2016)
4. Luc Jaulin.: Mobile Robotics. Book ENSTA-Bretagne, France. ISBN: 9781785480485. Elsevier (2015)
5. Peter Corke.: Robotics, Vision and Control. Fundamental Algorithms in MATLAB. Book. ISBN: 9783642201448. Spinger (2011)
6. John Wang, Edwin Olson.: AprilTag 2: Efficient and robust fiducial detection. Proceedings of the International Conference on Intelligent Robots and Systems (2016)
7. Alexandre Borowczyk, Duc-Tien Nguyen, André Phu-Van Nguyen, Dang Quang Nguyen,David Saussié, Jerome Le Ny.: -Autonomous Landing of a Multirotor Micro Air Vehicle on a High Velocity Ground Vehicle. IFAC-PapersOnLine, Volume 50, Issue 1, pp 10488--10494. 2017
8. Xu Zhong, Yu Zhou, Hanyu Liu.: Design and recognition of artificial landmarks for reliable indoor self-localization of mobile robots. International Journal of Advanced Robotic Systems. pp 1--13. (2017)
9. Joshua G. Mangelson, Ryan W. Wolcott, Paul Ozog, Ryan M. Eustice.: Robust Visual Fiducials for Skin-to-Skin Relative Ship Pose Estimation. OCEANS 2016 MTS/IEEE Monterey (2016)
10. Maximilian Krogius, Acshi Haggenmiller, Edwin Olson.: Flexible Layouts for Fiducial Tags. Proceedings of the {IEEE/RSJ} International Conference on Intelligent Robots and Systems (2019)
11. Christian Nissler, Stefan Büttner, Zoltan-Csaba Marton, Laura Beckmann, Ulrike Thomas.: Evaluation and Improvement of Global Pose Estimation with Multiple AprilTags for Industrial Manipulators. IEEE 21st International Conference on ETFA. (2016)
12. J. Rodriguez, G. Grosso, R. Zurita, L. Cecchi.: Intervención de la Facultad de Informática en la enseñanza de Ciencias de la Computación en la escuela media basada en robótica educativa. XI Congreso de TE&ET pp221--231 (2016)