Ambientes de aprendizaje en robots de servicio

[1]Apolinar Ramírez S., [2]L. Enrique Sucar S., [3]Eduardo Morales M., [4]Efraín Damián L.

Coordinación de Ciencias Computacionales
Instituto Nacional de Astrofisica, Óptica y Electrónica (INAOE)
Tonantzintla, Puebla, México
Contacto: {[1]apolinar_r,[2]esucar,[3]emorales,[4]eldamian}@inaoep.mx
Paper received on 04/08/08, accepted on 11/09/08.

Resumen. El advenimiento de los robots de servicio en ambientes poco estructurados y con usuarios no especializados en su programación, hace necesario desarrollar formas de programación acordes a las formas de comunicación usuales entre humanos. Una de estas formas es la programación por demostración (PbD), donde la programación de robots puede considerarse como el resultado de la interacción entre el robot y el usuario en un ambiente multimodal, con el usuario en el papel de profesor. Bajo este paradigma, el robot observa pasivamente la tarea realizada por el usuario, genera una representación abstracta de la demostración, generaliza obteniendo la información relevante y reproduce la tarea. Los enfoques usados en programación por demostración generalmente utilizan sensores especiales adheridos al cuerpo del usuario y requieren de ambientes controlados. Por otro lado, una vez que se aprende una tarea no hay forma de modificarla. Finalmente, las tareas aprendidas pueden ser sub-óptimas en cuanto a las posibles acciones del robot. Nosotros proponemos un ambiente de aprendizaje donde el robot aprende los movimientos de la tarea empleando únicamente sensores dispuestos sobre el robot, pero con un papel más activo tanto del robot como del usuario. Por el lado del robot, estableciendo una estructura jerárquica en las habilidades aprendidas y por el lado del usuario al optimizar la realización de la tarea reproducida por el robot mediante la retroalimentación. Se muestran resultados para mostrar la viabilidad de lo propuesto en una tarea simple de seguimiento.

1 Introducción

Los robots de servicio personal, de acuerdo a la ONU[1] [11], son robots autónomos de uso personal que asisten o entretienen a personas en actividades domésticas o recreativas y que típicamente realizan tareas tales como vigilar, manipular objetos y mensajes (tomar, traer, llevar), limpiar y guiar, entre otras. Este tipo de robots van siendo más comunes y los encontramos en la forma de guías en un museo, de vigilantes en áreas interiores [6], de asistentes en hospitales y oficinas [10] o de limpiadores en hogares [13]. Los robots de servicio operan en entornos no estructurados y dinámicos a diferencia de los robots industriales que operan en ambientes conocidos y que es posible mantener sin cambios.

[1] Organización de las Naciones Unidas

©M. G. Medina Barrera, J. F. Ramírez Cruz, J. H. Sossa Azuela. (Eds.)
Advances in Intelligent and Information Technologies.
Research in Computing Science 38, 2008, pp. 55-64.

Los usuarios de los robots de servicio son generalmente personas sin entrenamiento en la programación de robots, con diferentes requerimientos en las tareas a realizar por el robot, lo que motiva al desarrollo de sistemas de programación automática, donde la generación de código dependa de la información que el usuario proporcione en línea. Uno de los paradigmas utilizado para la programación de robots ha sido el de Programación por Demostración *(Programming by Demonstration - PbD)*, que consiste básicamente en mostrar varias veces una misma tarea y que a partir de las demostraciones el robot aprenda a hacer la tarea al imitar a la persona que la realiza.

De acuerdo a Dillmann [1] la programación, cooperación e interacción con robots de servicio tiene un carácter multimodal, entendiendo por esto que el usuario los programe mediante el habla, gestos o mostrando la tarea a realizar; el robot observa, interpreta y entonces trata de realizar la tarea mostrada por el usuario. La programación multimodal permite que usuarios inexpertos en la programación, pero que conozcan la forma de realizar la tarea, sean capaces de realizar la programación de los robots de servicio.

En la última década, el paradigma *PbD*, -- también conocido como Aprendizaje por imitación --, ha sido uno de los métodos aplicado y desarrollado de manera notable para el aprendizaje de tareas en robots de servicio. La importancia de *PbD* se debe a su mayor flexibilidad en el uso de dispositivos, la economía de la programación y su adaptabilidad a usuarios no especialistas en la programación de robots.

Los enfoques usados en *PbD* generalmente utilizan sensores especiales adheridos al cuerpo del usuario y requieren de ambientes controlados. Por otro lado, una vez que se aprende una tarea no hay forma de modificarla si hay necesidad de mejorarla o adecuarla a otro contexto. Finalmente, las tareas aprendidas pueden ser subóptimas en cuanto a las posibles acciones el robot. Este artículo aborda estos problemas con un sistema de aprendizaje por demostración de tareas en un robot de servicio cosiderando únicamente sensores dispuestos sobre el robot, optimizando la realización de la tarea mediante aprendizaje por refuerzo (RL) y utilizando retroalimentación del usuario para su modificación.

2 Trabajos relacionados

Dos son los problemas relevantes que han merecido mayor atención en el paradigma *PbD*: (a) la manera como se aprende y (b) las interfaces usuario-robot que se emplean. Para Ehrenman [3] el problema fundamental en el aprendizaje de tareas complejas reside en interpretar y generar una descripción abstracta que refleje las acciones e intenciones del usuario por lo que se requiere de un entorno aislado de ruidos (cambios de iluminación, otros actores, otros sonidos) y dotado de sensores que permitan una observación lo más completa posible, con sistemas de visión con más de dos cámaras, guantes electromagnéticos, sensores de movimiento y marcas. Independientemente de los resultados, tales condiciones controladas se alejan de un ambiente natural de operación de los robots de servicio.

Otras técnicas se orientan a un solo tipo de tarea, como es el caso de aquellas donde las observaciones del usuario corresponden a los puntos de recorrido del robot. En [2] se aborda el problema de programación de un robot manipulador median-

te *PbD*, donde con la ayuda de un guante magnético, visión y gesto coverbal[2] se guía al manipulador a una serie de puntos en el espacio de trabajo. Aquí, las *observaciones del usuario* son los *estados* (*waypoints* en el original) del programa del robot, que se representan como vectores reales discretos en el tiempo. Las desventajas de esta técnica son (i) su orientación a un solo tipo de tareas que dependen de la precisión en la determinación de los *waypoints*, por lo que no puede generalizarse a otros tipos de tareas requeridas en un robot de servicio y (ii) una vez aprendida una tarea no hay manera de reprogramarla. Una ventaja de este trabajo es el guíado del robot mediante un guante magnético, lo que reduce el problema de mapeo de la dinámica humana a la dinámica del robot, ya que la información que toma el robot corresponde a sus atributos internos. En particular, versiones de esta técnica se emplean en robots aspiradores con resultados que han permitido su aplicación comercial exitosa.

Otro enfoque dentro de *PbD* lo presenta el trabajo de Katsuki [8], que consiste en interpretar órdenes verbales en lugar de determinar por observación visual las intenciones del usuario. Las frases están formadas por solo dos componentes, *Task* y *Target*, donde el primero debe ser un verbo y el segundo un sustantivo, que representan la tarea y el sujeto, respectivamente. Los verbos y sustantivos utilizados tienen asociados un conjunto de reglas que describen su tipo de movimiento que puede ser *lineal, circular*, *punto-a-punto* o *imposible*. Para asociar a cada palabra su tipo de movimiento, se definen relaciones de concordancia $A_m(w)$ y de factor de confianza $C_m(w)$ entre la palabra w y el tipo de movimiento m. Por ejemplo, $A_{lineal}(puerta)=1$, $C_{lineal}(puerta)=0.5$ significan respectivamente que ``hay una puerta que se mueve con movimiento lineal" pero ``se tiene sólo un 50% de certeza de que el movimiento sea lineal". Las palabras y sus relaciones conforman una base de datos empleando WordNet [12] lo que permite relacionar sinónimos y enriquecer el vocabulario a emplear para dar instrucciones al robot. El razonamiento lo definen como el proceso de selección del tipo de movimiento a partir de una frase, que en caso de no tener asignados sus valores de relaciones de concordancia y de factor de confianza, estos se obtienen mediante reglas de propagación heurística entre los árboles del WordNet.

Entre las desventajas de este enfoque tenemos que: (i) el uso de un diccionario no resuelve el problema semántico de múltiples significados de una misma palabra, (ii) el usuario no retroalimenta al robot para mejorar la comprensión de las frases, (iii) las reglas aplicadas para la propagación de las relaciones no tiene un sustento formal, son de índole heurístico, y (iv) un rendimiento no mayor al 55% en la comprensión de frases.

En contraste a estos enfoques, se propone un esquema donde el robot solo cuente con sensores montados en el mismo y utilice la retroalimentación del usuario para mejorar la tarea una vez aprendida.

---

[2] De acuerdo a [9] (pp.1), es un gesto acompañado de voz.

3 Metodología

El esquema propuesto consiste principalmente de una representación jerarquizada de las tareas, un módulo de aprendizaje de políticas y de mejora por retroalimentación del usuario, como se muestra en la Figura 1, donde las demostraciones de la tarea junto con la interacción del usuario se presentan al módulo de aprendizaje en términos de estados, acciones y refuerzos. Estos datos son la entrada a un método de RL para obtener la política óptima inicial de la tarea, la que se pone a prueba en el ambiente real para recibir retroalimentación del usuario y mejorarla. Finalmente la tarea se registra en la base de datos.

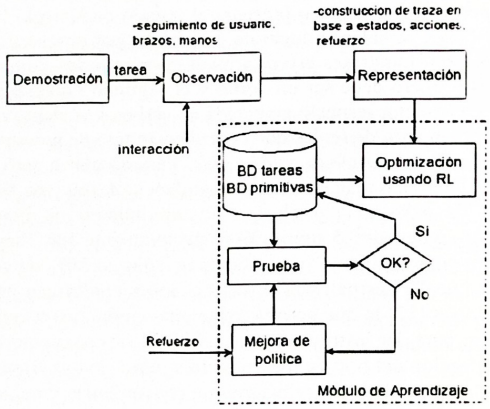

Fig. 1. Esquema general del aprendizaje de tareas. En la primera etapa, se muestra la tarea a realizar generando una representación en términos de espacio de estados, primitivas y refuerzos. Se establece una política inicial empleando algoritmo de aprendizaje por refuerzo. En la segunda etapa se mejora la política empleando retroalimentación directa del usuario.

3.1 Representación de tareas

Para la representación de las tareas se estableció una estructura de tres niveles. En el primer nivel se encuentran las primitivas básicas propias del robot que interactúan directamente con los actuadores y sensores. En un segundo nivel, se encuentran las *primitivas simples*, consistentes en soluciones a pequeñas partes de una tarea que pueden combinarse para formar una tarea. Estas primitivas simples están basadas en las primitivas básicas y pueden ser proporcionadas o aprendidas. En un tercer nivel se encuentran las tareas construidas con base en las dos anteriores. Se tienen definidas en principio un conjunto de primitivas simples para las tareas de un robot de servicio [7] clasificadas en primitivas de tarea y de navegación, además de expresiones para la retroalimentación por el usuario (ver Tabla 1).

3.2 Módulo de aprendizaje

Se considera que el robot posee un conjunto de capacidades (primitivas) en términos de las cuales se puede definir la tarea a realizar, por ejemplo: *avanzar, girar, tomar, soltar,* etc. Para la tarea que se desea que aprenda el robot, se presentan varias demostraciones de la misma en condiciones del ambiente que pueden variar: por ejemplo, si se quiere que aprenda la tarea *"seguir al usuario"* consistente en que el robot se mantenga a una distancia determinada del usuario, se pueden elegir diferentes puntos de inicio. Las demostraciones de la tarea se mapean en acciones primitivas del robot y condiciones (estados) en las que se realizan. Estas representaciones obtenidas se aplican al aprendizaje de la tarea. Posteriormente se prueba la tarea aprendida y se retroalimenta al robot tanto en resultados positivos como negativos. La flexibilidad del proceso consiste en permitir al usuario después de una primera serie de demostraciones y aprendizaje, retroalimentar al robot para mejorar la tarea. Así, en las etapas del proceso se realizan las siguientes actividades:

Tabla 1. Tabla de primitivas simples para tareas de un robot de servicio. Otras primitivas consideradas serían de manipulación de objetos, que no aparecen en esta tabla.

Primitiva simple	**Descripción**
Primitivas de tarea	
Robot	prepara ejecución de una primitiva o tarea
Aprende	aprendizaje de una tarea
Inicio	inicio de tarea/aprendizaje
Fin	fin de tarea/aprendizaje
Primitivas de navegación	
Avanza	avance al frente a velocidad normal
Alto	detener marcha
gira derecha	giro a la derecha
gira izquierda	giro a la izquierda
+ velocidad	aumenta velocidad de avance
Expresiones de retroalimentación	
Bien	se da recompensa
Mal	se da penalización

1. **Demostración.** El usuario especifica el espacio de estados y el conjunto de primitivas a usar, con el fin de delimitar el dominio de la tarea. Por ejemplo, si la tarea es *``seguir al usuario''*, el usuario lo indicaría con la frase: *``robot, aprende sígueme''*. El robot almacena la palabra *``sígueme''* y la va a relacionar con la tarea a aprender. El usuario indicaría enseguida *``robot buscar persona''* con lo que el robot puede establecer que *``buscar''* se refiere al uso de su sistema de visión y *``persona''* el objeto de su interés. Esto le sirve para definir que su representación de estados va a estar en términos de la salida de *``buscar persona''* que corresponde a las variables de *``distancia''* y *``ángulo''* entre la cámara y la persona (Tabla 2).

2. **Observación.** Secuencia de captura de las demostraciones por el robot empleando sus sensores y recibiendo retroalimentación del usuario.
3. **Representación.** En base a las secuencias/trazas de la demostración, se construye un conjunto de estados, acciones relacionadas y refuerzos dados por el usuario. Los datos percibidos a través del sistema de visión conforman el estado del ambiente, las primitivas corresponden a las acciones y las frases de retroalimentación corresponden a los refuerzos del usuario.
4. **Optimización.** Las trazas sólo corresponden a algunos de los estados y acciones del dominio. Dado que la secuencia dada por el usuario no es necesariamente la mejor se utiliza aprendizaje por refuerzo (RL) para determinar una política inicial. Para emplear aprendizaje por refuerzo se necesita especificar estados, acciones y refuerzos, por lo que se asume: i) dado el problema se especifica el espacio de estados, ii) el robot tiene una serie de primitivas que puede usar, iii) sólo las acciones dentro de las trazas se van a considerar y iv) los refuerzos son los adecuados.
5. **Mejora.** La mejora de política corresponde a la prueba de la tarea aprendida por el robot en condiciones de poder recibir retroalimentación por el usuario para incluir cambios en las acciones o modificar refuerzos. Estos refuerzos y acciones se emplean para la corrección de la tarea mediante RL. La tarea mejorada se registra en la base de datos. Se considera que esta actividad puede realizarse cuando se requiera modificar una tarea ya aprendida.

4 Resultados experimentales

En este experimento, mediante simulación en Player/Stage [4], se aplica aprendizaje por demostración para aprender la tarea de seguimiento a un usuario, buscando como meta mantener al robot de servicio a una distancia de no más de un metro del usuario. El usuario a seguir se simula mediante un robot Pioneer que deambula en un ambiente conocido: este ``robot-usuario" utiliza un juego de sonares para eludir los obstáculos en el mapa. El robot de servicio se simula con un Pioneer dotado de un sensor láser para ubicar su posición y seguir la posición del usuario en su desplazamiento. Hay también un ``experto" que muestra los ejemplos al robot y lo retroalimenta mediante un *teach-pendant virtual* que contiene las primitivas a emplear.

En la segunda fila se tienen primitivas de visión (que no se emplean en este experimento). La tercer fila contiene las primitivas de navegación: AVANZA para que el robot avance a una velocidad constante en línea recta, ALTO para detenerlo, DERECHA para que dé un giro a la derecha, IZQUIERDA para que gire a la izquierda, + (*signo más*) para indicar que aumente su velocidad al doble, - (*signo menos*) para indicar que reduzca su velocidad a la mitad.

Por último en la cuarta fila se encuentran expresiones de BIEN y MAL para indicar respectivamente refuerzos positivos y negativos durante la retroalimentación en las pruebas de las tareas.

El proceso de aprendizaje ocurre en dos etapas: en la primera, se indica al robot APRENDE, con lo que se indica que se trata del punto de registro de la tarea y su parámetro es el nombre del archivo donde se grabarán los estados y las acciones de

las demostraciones realizadas. Se indica INICIO de la tarea y el robot comienza ubicando su posición global en el mapa mediante marcas naturales del mapa de acuerdo a [5]. El ``usuario-robot'' se desplaza en el entorno mientras el ``experto'' le da indicaciones al robot sobre las acciones que debe realizar para lograr el objetivo de seguimiento al usuario mediante comandos a través del *teach-pendant*. La demostración termina cuando el robot logra aproximarse a un metro o menos del usuario y se indica con la primitiva FIN, lo que da un refuerzo positivo además que el robot identifica la meta de la tarea.

Con el conjunto de demostraciones obtenidas, se aplicó el algoritmo RL Q-Learning [14] a los atributos y resultados para establecer una política inicial. Se aplicó recompensa sólo sobre el estado meta . En las pruebas obtenidas en esta primer etapa se observó un 50% de eficacia en la tarea.

En la segunda etapa, se busca perfeccionar el aprendizaje de la tarea. Para esto, se prueba la política obtenida y se retroalimenta con refuerzos del usuario, generando nuevas demostraciones, retroalimentando negativamente al robot cuando la acción no es la óptima mediante el *teach-pendant* virtual en cuyo caso se le indica cuál debería ser la acción para dicho estado. Las recompensas obtenidas -- positivas y negativas -- sirven de retroalimentación al aplicar nuevamente Q-learning, considerando los resultados de la primera etapa como previos.

Tabla 2. Primitivas simples de acuerdo a los dispositivos empleados. Las primitivas definidas se asocian a variables de estado que forman parte del espacio de estados de la tarea a realizar.

Dispositivo	**Primitiva simple**	**Variables (var)**	**Descripción**
robot	*avanza*	$r=(x,y), \theta, r', \theta'$	posición robot
	gira derecha	$r=(x,y), \theta, r', \theta'$	"
	gira izquierda	$r=(x,y), \theta, r', \theta'$	"
	+ velocidad	$r=(x,y), \theta, r', \theta'$	"
	alto	$r=(x,y), \theta, r', \theta'$	"
cámara	*buscar persona*	z_i, α_i	distancia, ángulo
	registrar persona		
	identificar persona	z_i, α_i	distancia, ángulo
	registrar torso	z_i, α_i	"
	registrar escenario		
	registrar objeto		
	identificar objeto	z_i, α_i	distancia, ángulo
	buscar objeto	z_i, α_i	"
	mirar izquierda	*áng. pan*	áng.pan<-áng.pan -5°
	mirar derecha	*áng. pan*	áng.pan<-áng.pan +5°
	mirar arriba	*áng. tilt*	áng.tilt<-áng.tilt +5°
	mirar abajo	*áng. tilt*	áng.tilt<-áng .tilt-5°
láser	*localizarse globalmente*	$r=(x,y), \theta$	posición robot
	localizar obstáculo	z_i, α_i	distancia, ángulo

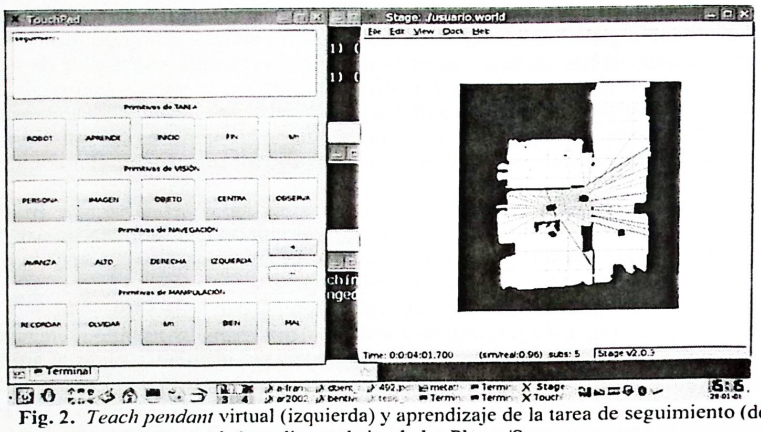

Fig. 2. *Teach pendant* virtual (izquierda) y aprendizaje de la tarea de seguimiento (derecha) mediante el simulador Player/Stage.

Al realizar la aplicación de las políticas obtenidas en las dos etapas, se muestra una eficacia mayor (83%) después del proceso de retroalimentación por el usuario, considerando un ambiente sin obstáculos en doce pruebas realizadas con diferentes puntos de inicio y posiciones del usuario y del robot (Tabla 3).

Tabla 3. Eficacia obtenida en tarea de seguimiento, considerando el número de veces que logra llegar a la meta como un episodio completo. La meta establecida fué mantener el robot a una distancia de un metro del usuario en su recorrido.

	% de episodios completos
1a. etapa	50
2a. etapa	83

Un ejemplo de la aplicación de la política mejorada por la segunda etapa del proceso de aprendizaje se muestra en la Figura 3.

4 Conclusiones y trabajo futuro

Los resultados muestran la factibilidad de utilizar retroalimentación del usuario para mejorar la eficiencia del aprendizaje que se obtiene en las demostraciones. La importancia de la segunda etapa de aprendizaje es que existe más información por la retroalimentación (refuerzo y acciones correctivas) del usuario y el sistema puede aprender a hacer una relación más apropiada entre estados y primitivas.

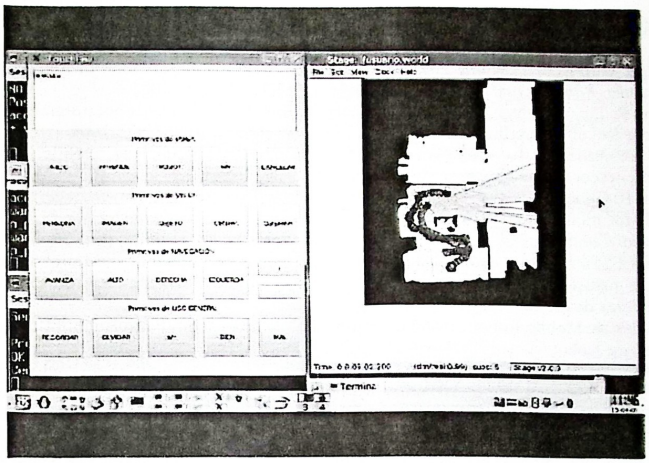

Fig. 3. Robot aplicando la tarea de seguimiento del usuario mejorada por retroalimentación del usuario. El robot se mantiene detrás del usuario. Prueba mediante el simulador Player/Stage.

El problema de la discretización del espacio de estados se debe analizar de acuerdo al tipo de problema. En otras tareas de un robot de servicio, como las tareas de manipulación, se requerirá de una mayor discretización del espacio de estados, con lo que aumentará el tamaño del espacio de estados, lo que afecta la eficiencia del algoritmo.

Con este experimento se muestra que es posible que el robot aprenda del usuario con pocos ejemplos. Igualmente, la retroalimentación ayuda a mejorar la tarea y evita un gran número de demostraciones para cubrir todo el espacio de estados. Aunque este experimento es un caso ``sencillo" por estar basado sólo en primitivas y no en otras tareas, se plantea resolver problemas más complejos como la manipulación de objetos por el robot.

El aprendizaje en robots de servicio es un dominio de investigación con retos propios. Para encontrar una política apropiada entre situaciones del mundo real y las acciones del robot, se requiere contar con ejemplos representativos, es decir contar con un modelo, lo cual no puede obtenerse la mayoría de las veces por las características dinámicas y desconocidas del ambiente natural en que debe desenvolverse un robot de servicio. Así, el uso de métodos de aprendizaje por refuerzo en aprendizaje de tareas para los robots ha sido un camino natural, a pesar de sus limitantes en cuanto al número de ejemplos que puede obtener un robot antes de que se agoten sus baterías -- con lo que el teorema de convergencia no se aplica -- y la necesaria discretización de su espacio de estados. Se considera que con el uso de primitivas, la jerarquía de tareas y los métodos de RL, se logrará el aprendizaje de tareas de una manera más simple y natural.

Referencias

1. Rüdger Dillmann. Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems*, 47(2-3):109-116, 2005.
2. Kevin R. Dixon, John M Dolan, and Pradeep K. Khosla. Predictive robot programming: Theoretical and experimental analysis. *I.J. Robotic. Res*, 12(9):955-973,2004.
3. M. Ehrenmann, T. Liitticke, and R. Dillmann. Directing a mobile robot. http://citeseer.ist.psu.edu/543651.html, July 30, 2001.
4. Brian P. Gerkey, Richard T. Vaughan, and Andrew Howard. The player/stage project: Tools for multi-robot and distributed sensor systems, in *Proceedings of the 11th International Conference on Advanced Robotics (icar2003)*, pages 317-323, june 30, 2003.
5. Sergio F. Hernández. Navegación de un robot móvil en ambientes interiores usando marcas naturales del ambiente. *Master's Thesis*, Instituto Tecnológico y de Estudios Superiores de Monterrey, 2005.
6. Patrolbot de Mobile Robots, inc., http://www.mobile robots.com/PatrolBot.html, 2006
7. A Service Robot Named Markovito. LARS. 2007
8. Rie Katsuki, Roland Siegwart, Jun Ota, and Tamio Arai. Reasoning of abstract motion of a target object through task order with natural language – prenowledge of object-handling-task programming for a service robot. *Advanced Robotics*, 20(4):391-412,2006.
9. Isabella Poggi. How to decide which gesture to make according to our goals and our contextual knowledge, http://citeseer.ist.psu.edu/501075.html, October 17, 2001.
10. Nicholas Roy, Gregory Baltus, Dieter Fox, Francine Gemperle, Jennifer Goetz, Tad Hirsch, Dimitris Margaritis, Mike Montemerlo, Joelle Pineau, Jamie Schulte, and Sebastian Theun. Towards personal service robots for the elderly, *Proc. of the Workshop on Interactive Robotics and Entertainment*, September 12, 2000.
11. UN. *United Nations and The International Federation of Robotics: World Robotics 2000*. New York and Geneva, 2002 edition.
12. Wornet 2.0 . Princeton University, 1985.
13. Irobot, Aspiradoras roomba, www.irobot.com .
14. Watkins C.J.C.H., Dayan P.; Technical Note: Q-Learning; *Machine Learning*, Vol. 8, pages 55-68, 1992.