Localización probabilística del humanoide Nao en el campo de la Robocup

Francisco J. Rdez. Lera, Juan F. García,Camino Fdez. Llamas, Vicente Matellán

Grupo de Robótica - Escuela de Ingenierías Industrial e Informática

Campus de Vegazana. Universidad de León. 24071 León (Spain)

E-mail:{fjrodl@unileon.es,jfgars@unileon.es,camino.fernandez@unileon.es, vicente.matellan@unileon.es}

Web: http://robotica.unileon.es

Resumen

Una de las habilidades fundamentales que debe poseer un robot para definirse como autónomo es la de auto-localizarse en un entorno dado. En este artículo abordaremos el problema de la localización aplicando técnicas probabilísticas combinado representaciones topológicas y métricas sobre un terreno acotado como es el campo de fútbol de la Robocup. El artículo presenta también el funcionamiento y resultados obtenidos sobre un conjunto de pruebas simuladas y reales utilizando el humanoide Nao.

1. Introducción

La Robocup promueve varias competiciones que son el punto de encuentro de gran cantidad de instituciones educativas. En ellas es posible comprobar los avances que se han logrado en diferentes ámbitos de la robótica.

En nuestro caso participamos en la categoría de fútbol *Standar Platform League*, grupo en el cual todos los equipos concurren con la misma plataforma hardware, el Nao de Aldebaran Robotics.

En este artículo mostramos una implementación de un sistema de localización basada en técnicas probabilísticas [12]. En esta primera aproximación no vamos a tener en cuenta nada más que una portería, excluyendo la utilización de ambas porterías simultáneamente, utilización de las líneas del campo y posibles oclusiones. En la siguiente sección realizaremos un pequeño repaso a la localización probabilística, para seguidamente mostrar como se ha realizado la implementación sobre el entorno de la RoboCup. Finalmente se mostrarán los experimentos realizados para la prueba del software tanto sobre simulador como a nivel real y los resultados obtenidos.

2. Localización Probabilística

La localización probabilística es un conjunto de algoritmos, basados en el teorema de Bayes, que nos permitirán estimar la posición del robot sobre un mapa dado, a lo largo de un intervalo de tiempo utilizando un modelo a priori de acciones y observaciones. Una de las primeras utilizaciones de estas técnicas en robot móvil fue la de Simmons y Koening [10].

La idea básica de funcionamiento consiste en mantener una distribución de probabilidad sobre todas las posibles localizaciones. Este sistema transforma la creencia de estar en un estado s en el instante $t-1$ en una nueva creencia de estar en el estado s' en el instante t tras efectuar una acción a o una observación o. De este modo la distribución se ira actualizando tras cada actualización.

A diferencia de los trabajos de Simmons, que utilizan sensores de rango (ultrasónicos o láser), en este trabajo utilizaremos la cámara del robot Nao, la razón de seleccionar este sensor como fuente de entrada es el buen resultado que ofrece en este tipo de entornos

[9], [15].

También, a diferencia de muchos trabajos anteriores, en este planteamiento utilizaremos una aproximación mixta de localización topológica (similar a [5] o [6]) y métrica.

Finalmente, indicar que los métodos basados en este tipo de localización están ampliamente difundidos [2], [11], [14], pero se sigue trabajando sobre ellos para proporcionar robustez a este tipo de sistemas de localización sobre diferentes plataformas, como es el caso del trabajo descrito en este artículo.

3. Inicialización de los estados de creencia

La implementación de nuestro sistema parte de los Procesos de Decisión Parcialmente Observables de Markov [10], donde definimos la localización del robot como una distribución de probabilidad sobre un conjunto de estados $S = \{s_0, s_1, s_2, ...\}$. Definimos la creencia $Bel_t(S = s)$ de estar en un estado s en un instante dado t como el valor de probabilidad de la distribución para dicho estado.

Inicialmente podemos partir de un estado conocido, en ese caso la distribución de probabilidad estaría concentrada sobre ese estado. También podemos iniciar desde un estado inicial desconocido, que implicaría que la distribución de probabilidad sería uniforme sobre todos los estados.

La creencia $Bel_t(s)$ para cada estado se actualizará mediante dos etapas, la primera corresponderá al paso de predicción, asociada con el desplazamiento del robot. La segunda etapa de estimación corresponde con la observación obtenida por la cámara.

3.1. Etapa de predicción

En esta etapa definimos la probabilidad de estar en un estado s' después de ejecutar una acción a, como la agregación de las probabilidades de acceder a dicho estado desde cualquier otro estado s (incluido el mismo). Definimos entonces $P(s' \mid s, a)$ como la probabilidad de terminar en el estado s', partiendo del estado s y ejecutando la acción a. Esta probabilidad se puede calcular a priori.

Podemos calcular entonces la probabilidad de estar en s' en el instante t como la acumulación $\forall s$ de $P(s' \mid s, a)$ ponderada por la probabilidad que tenía asignada en el estado anterior $Bel_{t-1}(s)$.

$$Bel_t(s') = \int_s P(s'|s, a) * Bel_{t-1}(s)$$

Es interesante hacer notar que el resultado de este proceso aumenta la incertidumbre sobre la posición del robot, puesto que contempla todas las posibles transiciones para llegar desde un estado a otro incluso las que no se han producido.

3.2. Etapa de estimación

Una vez hemos ejecutado la acción, el robot se encontrará situado en un nuevo estado s. Procedemos entonces al paso de estimación de la posición a partir de las observaciones. En nuestro caso hemos decidido esta relación acción - observación, pero se pueden establecer otras relaciones (con más acciones por observación o a la inversa).

En esta etapa actualizamos la creencia a partir de la probabilidad, que también se puede calcular a priori, de obtener cierta observación o en el estado s, $P(o \mid s)$. Actualizaremos la creencia que cada estado poseía en el instante anterior en función de dicha probabilidad y la que tenía el estado en el instante anterior:

$$\forall s \in S, Bel_t(s) = p(o|s) * Bel_{t-1}(s)$$

Una vez calculado $Bel_t(s)$ $\forall s$ normalizamos la distribución para que $\sum_0^s Bel_t(s) = 1$, para lo que se calcula el factor de normalización η.

$$Bel_t^*(s) = \eta * Bel_t(s)$$

Nos serviremos de este paso para reducir la incertidumbre de la distribución obtenida en el paso de la predicción de forma que decrementamos la verosimilitud de los estados en los que la probabilidad de realizar la observación obtenida es pequeña.

4. Aplicación a la Robocup

4.1. Modelado de las acciones

En esta fase modelamos el sistema de acciones que implementará el robot. En primer lugar generamos la rejilla. Hemos optado por una rejilla de 12 teselas de largo por 8 de ancho de similar tamaño (500 mm x 500 mm). Sobre cada una de estas tendremos cuatro estados posibles para nuestro robot definidos por cuatro orientaciones básicas y sobre los cuales se repartirá la distribución de probabilidad.

Igualmente definimos las acciones a realizar por nuestro robot. Hemos decidido para evitar un gran coste computacional acotarlas a tres: avanzar, girar noventa grados en sentido horario y antihorario.

A continuación definimos la matriz de transiciones (cuadro 1). Esta tabla representa los estados s' que alcanzaríamos si realizásemos cierta acción a desde un estado s y la ejecución de la acción fuese perfecta.

	Nuevo Estado s'		
Estado s	Avanzar	Girar 90°	Girar -90°
0	32	1	3
1	1	2	0
2	2	3	1
3	7	0	2
...	...	...	...

Cuadro 1: Modelado de transiciones

En la figura 1 podemos observar como se distribuyen los estados asignados a cada nodo. Como ejemplo, ejecutando la acción "Avanzar" desde el estado 0 (córner derecho), idealmente se alcanzará el estado 32.

En la tercera etapa modelaremos la incertidumbre asociada a la ejecución real de cada acción. De forma experimental realizamos cada acción en 50 ocasiones obteniendo los datos que se resumen en el cuadro 2.

- N = No tiene efecto (el robot permanece en la misma tesela).
- A, AA, AAA = Avanza en una, dos o tres teselas.

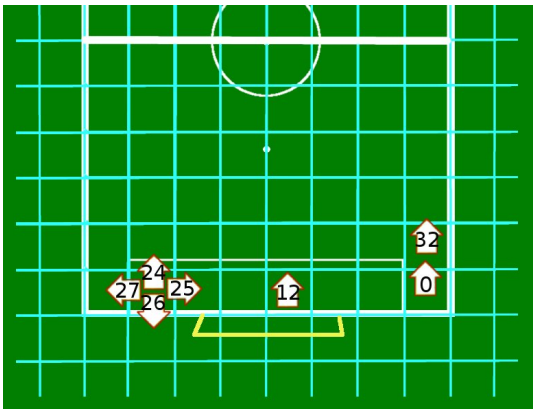

Figura 1: Representación parcial del campo

- G, GG = Gira una o dos veces

	Resultado de la acción			
Acción	N	A,G	AA,GG	AAA
Avanzar	5 %	80 %	10 %	5 %
Girar 90	2 %	90 %	8 %	-
Girar -90	2 %	93 %	5 %	-

Cuadro 2: Modelo de la incertidumbre en la ejecución de las acciones

Con ese modelo de incertidumbre se genera el modelo de las acciones. En el cuadro 3 podemos observar la matriz que obtenemos para la acción avanzar.

Avanzar	Nuevo Estado s '					
s	0	1	2	...	32	...
0	5	0	0	...	80	...
1	0	100	0	...	0	...
2	0	0	100	...	0	...
3	0	0	0	...	0	...
...	...	...	...	...	...	...

Cuadro 3: Modelado con incertidumbre de la acción avanzar

Esta matriz nos representa las probabilidades de alcanzar un estado dado a partir de su estado anterior. De forma similar,

obtendremos las tablas que modelan el resto de las acciones.

4.2. Modelado de las Observaciones

En esta primera aproximación no utilizaremos las líneas del campo ni las dos porterías simultaneamente. Solamente trabajaremos sobre una de las porterías dependiendo de la mitad del campo que nos corresponda. Esto nos permite controlar el gasto computacional pero no nos ayuda a disminuir la incertidumbre sobre el sistema.

4.2.1. Tamaño de portería observada

Para el modelado de las observaciones visuales vamos a utilizar como elementos significativos las porterías, que detectamos en un proceso de filtrado por color reflejado en la figura 2. En primer lugar utilizamos el tamaño de la portería observada.

Para calcular el modelo a priori de incertidumbre asociado a este parámetro, definimos en primer lugar unos intervalos (cuadro 4) discriminantes en función del número de píxeles (φ) que teóricamente debería tener el blob de portería que detecte el sistema de visión. Estos intervalos se calculan en función de la distancia y orientación de la tesela con respecto a la portería observada.

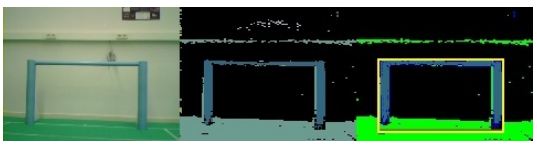

Figura 2: Observación, segmentación y blob de la imagen

A continuación, sobre cada uno de los estados tomamos 50 capturas de la portería y calculamos el porcentaje de veces que φ corresponde a cada intervalo. Con los valores reales obtenidos generamos el modelo de incertidumbre de la observación. El cuadro 5 presenta una porción del modelo a priori.

Observación del blob	Caso
φ <200	0
200 <φ <400	1
400 <φ <700	2
700 <φ <1000	3
1000 <φ <1300	4
1300 <φ <1500	5
1500 <φ <1700	6
1700 <φ <2000	7
φ >2000	8

Cuadro 4: Clasificación de las observaciones según el número de píxeles φ del blob

	% de veces que se percibe el caso					
estado	0	1	2	3	...	8
0	1	99	0	0	...	0
1	100	0	0	0	...	0
2	100	0	0	0	...	0
...	...	...	...	...	...	....
16	0	1	99	0	...	0
...	...	...	...	...	...	...

Cuadro 5: Modelo de incertidumbre de las observaciones de tamaño de portería

4.2.2. Observación visual del centroide

En este caso queremos modelar la orientación de la portería respecto a la posición del robot en cada estado, es decir, como debería ver orientada la portería si estuviese en ese estado. Para modelarla utilizaremos como observación la posición del centroide del mismo blob anterior, el que detecta la portería, sobre el eje X. Para optimizar los recursos computacionales, clasificaremos la posición que ocupa el centroide sobre el eje X definiendo un conjunto de casos $\{c_0, c_1, ...c_8\}$ donde c_0 corresponde al caso de no ver la portería y los otros ocho casos corresponden a desviaciones de aproximadamente 8 grados, según muestra la figura 3.

Para obtener los valores a priori de esta observación se realizaron 50 capturas desde cada estado, creando el modelo de incertidumbre que se muestra en el cuadro 6.

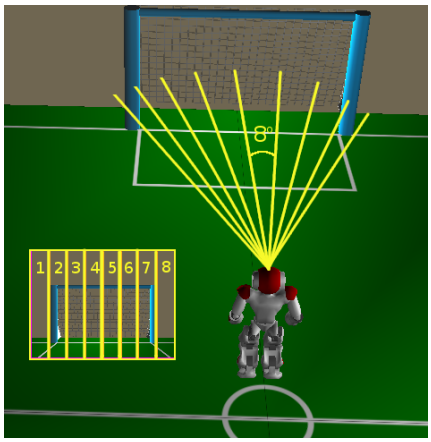

Figura 3: Posiciones posibles del centroide

4.3. Inicialización y funcionamiento

Con el modelado de acciones y observaciones a priori tenemos los componentes estáticos de nuestro sistema. Para completar su definición debemos establecer la inicialización de los estados de creencia y la función de explotación de la distribución de probabilidad $Bel(S)$.

Para la inicialización hemos realizado experimentos en los que el robot no conoce su estado inicial, en cuyo caso inicializamos $Bel_0(s) = \frac{1}{N_{TotalEstados}}, \forall s \in S$ y experimentos en los que sí la conoce, donde inicializamos $Bel_0(s_0) = 0,8$ para el estado conocido $s_0 = 0$ del que parte y $Bel_0(s) = \frac{0,2}{N_{TotalEstados}-1}$ para el resto.

Para la explotación, hemos decidido que para que que nuestro sistema designe la situación del robot como *localizado*, la creencia sobre el estado s que el sistema de localización designará como estado actual del robot, debe de acumular el 50 % de la probabilidad, es decir $Bel_t(s) > 0,5$ y establecemos como posición métrica del robot el centro de la tesela correspondiente a ese estado.

	% de veces que se percibe el caso					
estado	0	1	2	3	...	8
0	1	10	89	0	...	0
1	100	0	0	0	...	0
2	100	0	0	0	...	0
3	100	0	0	0	...	0
...	...	...	...	...	...	...

Cuadro 6: Porción de la matriz de incertidumbre de la posición del centroide

5. Experimentos

Para mostrar la validez del modelo planteado hemos realizado dos tipos de experimentos. En primer lugar hemos desarrollado pruebas con acciones y observaciones simuladas. De esta forma pudimos depurar el algoritmo y obtener rápidamente unos resultados iniciales acerca de como se comporta el sistema. En una segunda fase hemos realizado los mismos experimentos sobre el robot real con la única restricción de que las capturas no se realicen en movimiento. Es decir, suponemos que el robot se detiene después de ejecutar una acción y toma la imagen en parado. El objetivo es aislar el sistema de localización en su fase de desarrollo de los problemas asociados a las imágenes en movimiento (borrosidad, oclusiones, etc.). Igualmente hemos supuesto en los experimentos que la cabeza es solidaria con el cuerpo, es decir, no usamos las articulaciones del cuello.

Los objetivos de los experimentos realizados se pueden resumir entonces en:

- Validar si el sistema desarrollado funciona.
- Analizar el error con el que nos localizamos.
- Comprobar el coste computacional que supone sobre el robot.

5.1. Imágenes sintéticas

En este primer acercamiento realizamos una prueba del sistema de localización utilizando un conjunto de imágenes sintéticas.

Acotamos el problema a un supuesto en el que el robot se desplaza de forma ideal por el borde derecho del campo, definiendo como portería observable la azul y partiendo

desde un estado inicial conocido $s_0 = 0$. En la figura 4 se puede aprecia la distribución de probabilidad inicial en el eje de las Y y el conjunto de estados sobre el eje X.

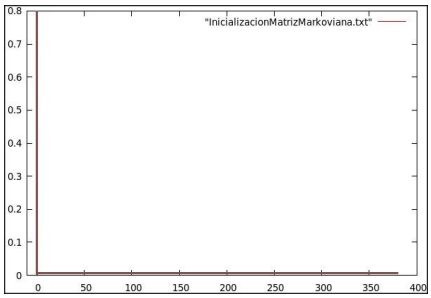

Figura 4: Estado de la distribución instante t=0

A partir del instante $t = 0$ se realizan de forma iterativa primero una acción y una observación después, para cada observación se realizan las dos estimaciones anteriores (tamaño y orientación del centroide del blob de la portería). En las figuras figuras 5 y 6, correspondientes a los instantes $t = 6$ y $t = 11$ (final) se puede apreciar como la distribución de probabilidad va actualizándose correctamente en cada instante a lo largo del recorrido.

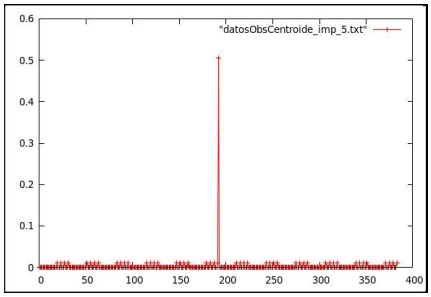

Figura 5: Estado de la distribución instante t=6

5.2. Imágenes reales

Una vez comprobada la validez del sistema con imágenes sintéticas, realizamos experimentos con imágenes reales, incorporando además el problema de la localización global, es decir,

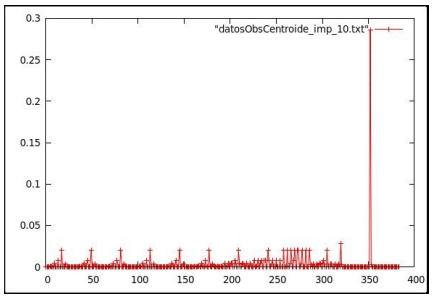

Figura 6: Estado de la distribución instante t=11

partiendo de un estado inicial desconocido para el sistema. Sobre el campo el robot comenzará en la posición correspondiente al estado s_0 representado en la figura 1 y realizará la trayectoria que le conducirá al córner contrario del mismo lateral $(s_0, s_{32}, s_{64}, s_{128}, s_{160}, ..., s_{352})$.

Realizamos la acción avanzar (definida dentro de la arquitectura diseñada del robot portero para la SPL [4]) y que equivale al desplazamiento de una tesela (500 mm) y posteriormente realizamos una observación. La figura 7 muestra la imagen real capturada y procesada por el código del portero en el estado inicial. A partir de esa segmentación se obtiene el blob de la portería y se calcula el número de píxeles azules del blob rectangular que la contiene y su centroide. Tras las dos estimaciones, la distribución de probabilidad $Bel_1(S)$ es la mostrada en la figura 8.

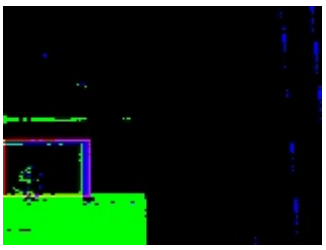

Figura 7: Captura del robot en el estado s_{32}

En este momento ningún estado acumula más del 50 % de la distribución de probabilidad, por lo que el sistema de localización

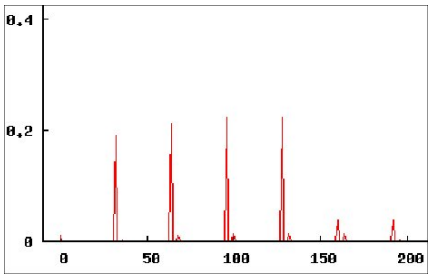

Figura 8: Distribución de probabilidad en el instante $t = 1$

retornaría el valor de "No localizado".

Cuando el robot está en el instante $t = 4$ ya hemos acumulado en un solo estado más del 50 % de la distribución $Bel_t(s_{128}) > 50\,\%$ (figura 9). Es a partir de este estado en el que empezamos a considerar que ya estamos localizados. Desde ese instante, la distribución de probabilidad acumula en el estado correspondiente a la posición real del robot la mayor parte de la probabilidad, es decir, el sistema de localización funciona correctamente.

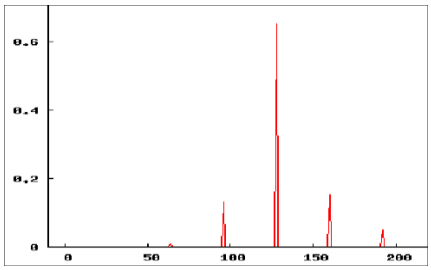

Figura 9: Distribución de la creencia en $t = 4$

6. Discusión y conclusiones

Analizando los resultados de ambos experimentos, simulado y real, podemos concluir que nuestro sistema de localización ofrece los resultados planteados teóricamente. Además, para evaluar este sistema de localización se añadió como un módulo más al portero preexistente [4].

La inclusión de este módulo permite al portero partir desde posiciones aleatorias en el lateral del campo y desplazarse hacia la portería. Esta situación es especialmente relevante porque el árbitro humano puede sancionar al portero si realiza alguna acción antirreglamentaria y la sanción es desplazarlo manualmente a un lateral del centro del campo.

La estrategia implementada en el portero ha sido hacerle caminar por el lateral derecho hasta el final del campo (estado s_{352}). Posteriormente, girar a la izquierda 90 grados (estado s_{355}), avanzar hasta la portería (estado s_{371}), girar de nuevo 90 grados en sentido contrario a las agujas del reloj para acabar en el centro de la portería (estado s_{370}). Este recorrido se muestra en la figura 10.

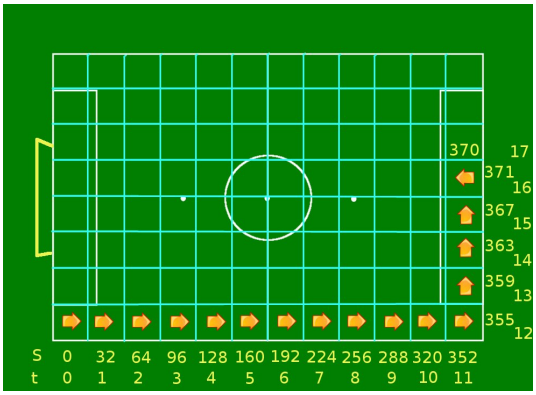

Figura 10: Recorrido para alcanzar la posición de portero

En las pruebas realizadas observamos que el robot acabó en la posición deseada con una diferencia a la posición ideal de entre 150 mm y 350 mm, es decir, dentro de la tesela destino.

Como trabajo en curso, estamos valorando el gasto computacional que supondría aumentar el número de teselas, el conjunto de estados y la resolución de las acciones (giros de $\pi/4$ p.e.). Si el incremento computacional fuese aceptable podríamos entrar a valorar otros tipos de observaciones como las líneas del campo [1].

Indicar finalmente, que un problema

detectado ha sido que si se obtienen muchas observaciones en ausencia de portería (el robot está mirando fuera del campo p.e.) el sistema se degrada mucho, ya que ven aumentada su creencia muchos estados, ralentizando la recuperación.

7. Agradecimientos

Los autores reconocen y agradecen el apoyo del Ministerio de Ciencia e Innovación a través del proyecto COCOGROM (DPI2007-66556-C03-01).

Referencias

[1] Bais, A., Sablatnig, R. and Novak, G., *Line-based landmark recognition for self-localization of soccer robots*, IEEE International Conference on Emerging Technologies, pp. 132:137, 2005.

[2] Cassandra,A., Kaelbling,L. and Kurien,J., *Acting under uncertainty: Discrete bayesian models for mobile robot navigation*, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 963:972, 1996.

[3] Enderle, S., Ritter, M., Fox, D, Sablatnög, S. , Kraetzschmar, G. and Palm, G, *Soccer robot localization using sporadic visual features*, International Conference on Intelligent Autonomous Systems, pp. 959:966, 2000.

[4] García,J., Rodríguez, F., Fernández, C, and Matellán, V., *Designing a minimal reactive goalie for the RoboCup SPL*, X Workshop de Agentes Físicos, 2009.

[5] Kosecka,J. and Li,F., *Vision based topological Markov localization*, IEEE Int. Conf. on Robotics and Automation, pp. 1481:1486, 2004.

[6] López,E., Barea,R., Bergasa,L.M., and Soledad Escudero,M., *Aplicación del método de Markov a la localización de un robot móvil en entornos interiores*, Proceedings of the "Seminario Anual de Automática, Electrónica Industrial e Instrumentación", pp. 339:342, 2002.

[7] Lowe., D., *Distinctive image features from scale-invariant keypoints*, International Journal of Computer Vision, pp. 91:110, 2004,

[8] Miura, J. and Yamamoto, K., *Robust View Matching-Based Markov Localization in Outdoor Environments*, International Conference on Intelligent Robots and Systems, pp. 2970:2976, 2008.

[9] Röfer T., and Jöngel M., *Vision based fast and reactive monte-carlo localization*. IEEE International Conference on. Robotics and Automation, pp. 856:861, 2003.

[10] Simmons, R.G. and Koening, S., *Probablistic robot navigation in partially observable enviroments*, International Joint Conference on Artificial Intelligence, pp. 1080:1087, 1995.

[11] Sridharan,M., Kuhlmann, G., and Stone, P., *Practical vision- based monte carlo localization on a legged robot*, IEEE International Conference on Robotics and Automation, 2005.

[12] Thrun,S., Burgard,W., and Fox, D., *Probablistic Robotics*, The MIT Press, 2005

[13] Thrun,S., Burgard,W., and Fox, D., *Markov localization for mobile robots in dynamic enviroments*, Journal of Artificial Intelligence Research 11, pp. 391-427, 1999.

[14] Yuen, D. and MacDonald,B., *A comparison between extended kalman filtering and sequential monte carlo technique for simultaneous localisation and map-building*, Australian Conference on Robotics and Automation, 2002.

[15] Wolf, J., Burgard, W., and Burkhardt, H., *Robust vision-based localization by combining an image retrieval system with monte carlo localization*, IEEE Transactions on Robotics, 21 (2), pp. 208:216, 2005.