A Dynamic and Efficient Active Vision System for Humanoid Soccer Robots

Matías Mattamala(✉), Constanza Villegas, José Miguel Yáñez, Pablo Cano, and Javier Ruiz-del-Solar

Department of Electrical Enginering, Advanced Mining Technology Center (AMTC), Universidad de Chile, Av. Tupper 2007, Santiago, Chile
{mmattamala,jruizd}@ing.uchile.cl

Abstract. This paper presents an efficient active vision system which controls the head of a humanoid soccer robot. The system explicitly separates static information obtained offline from the map, and dynamic information from mobile objects, such as the ball and other players. Both types of information are mapped and handled in a simplified structure called *action space*, which assigns scores to each possible action of the robot's head. Scores also consider the movement constraints of the robot's head. Due to its simplicity and efficient information handling, the proposed active vision system is able to run in real-time in less than 1 ms. The performance of the system in a robot soccer environment is tested via simulation and real experiments.

Keywords: Active vision · Camera control · Robot soccer · Self-localization · RoboCup · Standard Platform League

1 Introduction

Achieving a goal and staying self-localized is a natural task for a human, but not for a robot. This task is even more complex if the robot is in a highly dynamic environment -as a soccer match-, where several landmarks must be considered in order to have a good performance in both self-localization and playing during the game. In this context, it is not recommended to use passive self-localization systems based on predefined routines or heuristics because these are not related with the real world; in fact, the number of possible configurations of the ball and players on the field is huge. Then, it is necessary to use active localization methodologies to cope with the environment dynamics. Active localization systems are frequently divided into two types: active navigation and active sensing. Active navigation involves robot displacement in order to reduce self-localization uncertainties; on the other hand, active sensing refers to the manipulation of the robot's sensor in order to get more information from the world. In particular, an active vision system uses a camera as main sensor.

This work describes an active vision system for a humanoid soccer robot which aims to choose a head pose that maximizes a score in the so-called *action* *space*. The space is updated during the game using *a priori* information from the map, and dynamic information obtained from the ball and other players' positions. The proposed system also takes into account the movement constraints of the robot's head. This approach allows not only to control the robot's head but also to calculate the action to be performed using minimum computational resources in real-time.

The paper is organized as follows: relevant related work is presented in Sect. 2. The active vision system proposed is described in detail in Sect. 3. Finally, Sect. 4 presents the experimental results obtained, and Sect. 5 ends with the main conclusions.

2 Related Work

Active vision is a field mainly related to computer vision which origins date back to late 80's under the name of *active perception* [2], *animate vision* [3] and *active vision* [1]. These paradigms, despite having different backgrounds and motivations, are based on the same principle: in order to get better and most valuable information from the environment it is necessary to control the camera gaze, such as humans do with their vision system. This idea establishes a direct relation between perception and action, reason for which not only robotics have been involved in its development, but also neurosciences and cognitive sciences.

In this work we are concerned about applications of active vision systems in mobile robotics. Active vision has developed a strong relation with the problem of robot self-localization, currently one of the most important topics in robotics. For instance, Burgard [4] introduced an entropy minimization criteria to actively explore the environment. Davison [6] presented the first active vision approach to the SLAM problem using a stereo head, then Vidal-Calleja extended it to the monocular case [14]. Seara [10] introduced an intelligent gaze control system for a walking robot.

In the RoboCup context, several works have been published for the Aibo quadruped robots, such as Fukase [7], Mitsunaga [9], Stronger [12] and Guerrero [8]. Regarding NAO humanoid robots, currently used in the Standard Platform League, Seekircher et al. [11] presented an entropy minimization approach which handles both self-localization and ball uncertainty as well as the costs of choosing among them by using a camera control policy. Czarnetzki et al. [5] proposed another entropy minimization approach using a particle filter and testing the method in static and dynamic environments. However, neither the ball nor other mobiles objects are considered in that systems. Both approaches rely on an expensive modified particle filter to determine the target to gaze, requiring several optimizations that reduce the accuracy of the system. These modifications allow both systems to run in 5 ms in average. In contrast, our approach not only avoids online state estimations but it also considers the information of the ball and other players, achieving self-localization improvements and running in less than 1 ms.

3 Active Vision

3.1 General View

This work describes an efficient active vision system for robot soccer applications. It is based on an explicit division between the *a priori* information that can be obtained off-game from the known-map using its inherent landmarks, and the dynamic information provided by the moving ball and other robots in play. Our approach maps all information sources into an *action space* which modify certain scores associated to each possible head action. Then, the next target to be seen is selected as the action with highest score.

3.2 Action Space

Before describing the active vision system it is necessary to introduce a major concept for this work: the action space. The action space A of an end-effector is defined as follows:

$$A := \{(a, \omega) : a \in \mathbb{R}^N, \omega \in [0, 1]\} \tag{1}$$

where N is the number of *degrees of freedom* given by the effector's joints, a is an *action* which denotes a possible state of the effector, and ω is the *score* of the action a. Then, every possible state of the effector has a score assigned according to a criteria that depends on the application. The idea is to perform decision making over the next effector's actions just by selecting the highest score.

In this work we are concerned about controlling the robot's head, so the set of actions a will be reduced to reachable head poses $a = (\phi_{pan}, \phi_{tilt}) \in \Phi_{pan} \times \Phi_{tilt}$, where Φ_{pan} and Φ_{tilt} denotes the set of allowed angles for each joint. Since the robot will move its head in order to improve its self-localization, each action a is associated with real world observations that affect the robot's localization system. Hence, the criteria that defines ω will be to quantify the accuracy of the self-localization given a determined action a.

The next sections present a methodology to assign scores to each possible action, using a discrete representation of the action space. In a first formulation only prior information obtained from the known map is considered. Later, the static model is improved by adding some robot's physical constraints as well as dynamic information from the environment.

3.3 Obtaining a Priori Information

Most of the active vision systems presented in the Sect. 2 modify a state estimator using simulated perceptions in order to find the optimal action to be performed by the robot's head. Guerrero [8] proposed a probabilistic framework to infer which landmarks could be gazed depending on the current head pose. On the other hand, Seekircher [11] used a learned model for each perceptor (goal posts, lines, corners) to perform a Monte Carlo Exploration [13].

In order to avoid expensive online calculations using simulated perceptions, we rather propose to exploit the prior information of the map by sampling a measure of self-localization improvement while executing different actions, i.e. observing different landmarks. This process avoids calculating accurate sensor models because it fuses their localization improvements. Then, the collected data can be stored in a look-up-table, facilitating access during execution.

We develop an offline routine which uses the particle filter self-localization, and samples the particles' weighting sum for each possible robot pose (x, y, θ) and head pose $(\phi_{pan}, \phi_{tilt})$. In order to make the data acquisition tractable, we run the algorithm in a predefined discrete grid for the x, y, θ and ϕ_{tilt} dimensions, using the grid pose as input of the particle filter estimation. We did not consider the ϕ_{pan} dimension into the sampling process because this information can be approximated from poses with the same orientation θ while the active vision system runs.

Algorithm 1 presents the main procedure of the sampling process, which is run via simulation. We create a high dimensional table according to the sampling setup (the init_table function). Then, the system iterates moving the robot over the field, changing its pose and head tilt. The robot executes the particle filter localization system but we modified it to use the ground truth pose as the resulting motion update, resetting the previous frame estimation. Afterwards, the sensor update is executed using the exact observations expected for the current pose, which should provide the best possible response of the observation models. We also modified the output of the particle filter in order to extract the particles' weighting sum ω as a measure of self-localization accuracy; this is the score that will be assigned to the action performed.

Algorithm 1. Algorithm to sample the Particle Filter result over the field.

```
Generate_Field_Table()
init_table($T$)
for all $\bar{x}$ in $x_{grid}$ do
    for all $\bar{y}$ in $y_{grid}$ do
        for all $\bar{\theta}$ in $\theta_{grid}$ do
            move_to_pose($\bar{x}$,$\bar{y}$,$\bar{\theta}$)
            for all $\bar{\phi}$ in $\phi_{pgrid}$ do
                set_head_tilt($\bar{\phi}$)
                $\omega \leftarrow$ sensor_update($\bar{x}, \bar{y}, \bar{\theta}$)
                save_in_table($T, \bar{x}, \bar{y}, \bar{\theta}, \bar{\phi}, \omega$)
            end for
        end for
    end for
end for
```

The results obtained for the sampling process are shown in Fig. 1. Please notice that every position on field has a white ring assigned which represents the pan-tilt action space for each pose.

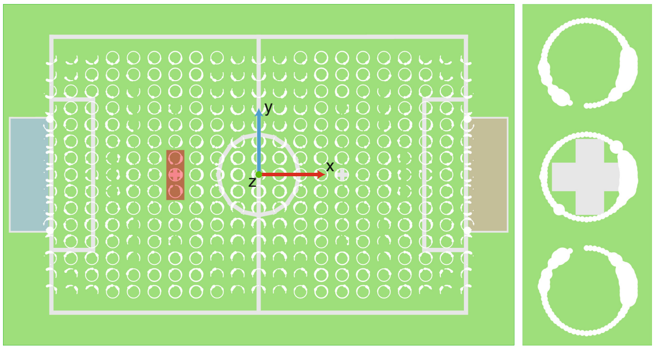


Fig. 1. Left: Example of field scores obtained for each grid position (x, y) after sampling. Each white ring denotes a set of pan-tilt actions. Right: A zoom of field area, marked in red. White rings are formed by small circles which denote the possible actions. The radius of each circle indicates the score assigned to the action (only the highest score is shown). In this case, we can notice that if we are close to the right penalty cross, the best action to perform in order to improve self-localization is to point at the center circle (Color figure online).

Using this approach, the action space for a determined pose in the field is obtained directly from the look-up-table. The action space changes while the robot navigates, allowing it to perform different head movements. Nevertheless, it works only in an empty field, because no other objects are being considered; that is why we call it a *static action space*. Next section will cover how to solve this problem by adjusting the scores dynamically.

3.4 Dynamic Information

In the previous section we sampled data from the field to determine a set of possible targets to gaze for each field position, each one having an assigned score. However, this information is not enough to cope with the environment dynamics because it only considers the map information. We have to improve the action space in order to deal with this issue, using dynamic models to modify the data. This section presents three adjustments to the *static action space* using the variable information of the field during a game. These are modeled by 2-dimensional functions that weight the scores ω.

Head Limits Constraints. A first improvement consist of adding the head limits information to the model. A humanoid robot has a limited range for the pan and tilt head angles, so this information must be considered. Denoting by Φ_{limit} the set of actions inside the constrained range, we can define a head-limiting function as:

$$f_{limits}(\phi_{pan}, \phi_{tilt}) = \mathbf{1}_{\Phi_{limit}}(\phi_{pan}, \phi_{tilt}) \tag{2}$$

Where $\mathbf{1}_{\Phi_{limit}}$ denotes the *indicator* or *characteristic function*. Then, every score of actions out of the pan or tilt ranges defined by the robot head is set to zero.

Head Movement Penalization. In general, we are concerned about choosing the best action in the next frame of execution, so we are interested in choosing an action that provides the most information at the lower *angular cost* for the head. According to this fact, it is necessary to consider a penalization function which reduces the score of the actions further away from the actual pose of the head. Assuming independence between coordinates in the action space, we can model the penalization function as follows:

$$f_{head}(\phi_{pan}, \phi_{tilt}) = v^h_{\mu,\kappa}(\phi_{pan}) g^h_{\mu,\sigma}(\phi_{tilt}) \tag{3}$$

where $v^h_{\mu,\kappa}(\phi)$ corresponds to the *von Mises function* or *circular gaussian function* (see (4)), whereas $g^h_{\mu,\sigma}(\phi)$ denotes a *gaussian function* (see (5)). The election of von Mises function is related with the periodicity of the action space with respect to pan angle as shown in Fig. 1.

$$v^h_{\mu,\kappa}(\phi) = V e^{\kappa \cos{(\phi - \mu)}} \tag{4}$$

$$g^h_{\mu,\sigma}(\phi) = G e^{-\frac{(\phi-\mu)^2}{2\sigma^2}} \tag{5}$$

In this case, the means of both functions are set at the current head pose. The dispersion κ and variance σ are parameters set by-hand. The constants V and G are chosen so as to normalize the peak of each function to 1.

Avoiding Obstacles in the Field of View. Obstacles occlude the information obtained from the map and affect the observations expected by the robot, so we would like to avoid gazing in those directions. This does not affect the obstacle detection, because the robot keeps moving the head during the active vision execution. In addition, obstacle detections are shared among the player of the same team. We model each obstacle using the same one-dimensional functions as before but with different parameters:

$$f_{obstacle}(\phi_{pan}, \phi_{tilt}, r) = k(r)(1 - v^o_{\mu,\kappa}(\phi_{pan})) g^o_{\mu,\sigma}(\phi_{tilt}) \tag{6}$$

The function shown in Eq. 6 further reduces the score of the actions closer to an obstacle on the action space. However, it also includes a correction function $k(r)$ which weights the score reduction depending on the obstacle distance, denoted by r. This function is estimated using real data. The means (μ_{pan}, μ_{tilt}) associated to the von Mises function $v^o_{\mu,\kappa}$ and the Gaussian function $g^o_{\mu,\sigma}$, respectively, are estimated by projecting the obstacle center onto the action space, using the inverse kinematics of the robot camera (Fig. 2 left). The von Mises' dispersion κ and the Gaussian variance σ are estimated in a similar way, by projecting the vertices of the obstacle bounding box onto the action space, and calculating the midpoint for each dimension p_{pan} and q_{tilt}. Using these points and the previously estimated mean, it is possible to estimate κ and σ as follows:

$$\hat{\kappa}_{pan} = \frac{1}{\kappa_0 |p_{pan} - \mu_{pan}|} \tag{7}$$

$$\hat{\sigma}^2_{tilt} = \sigma_0^2 |q_{tilt} - \mu_{tilt}|^2 \tag{8}$$

where κ_0 and σ_0 are compensation factors to adjust the parameter estimation empirically.

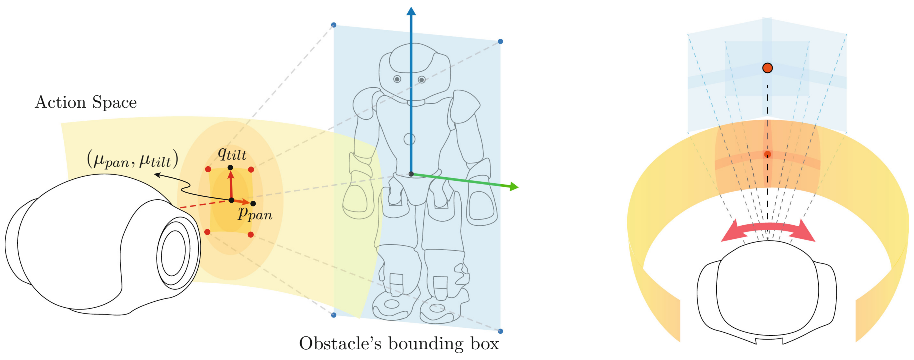


Fig. 2. Left: $f_{obstacle}$ parameter estimation using the bounding box vertices projected onto the action space. Right: Actions in which the ball appears in the Field of View define a region in the *ball action space* (Color figure online).

Updating Static Information. Finally, the *localization action space* can be updated using the information described anteriorly. The previous functions are evaluated in each action $a = (\phi_{pan}, \phi_{tilt})$ of the current static action space, and used to weight and, therefore, update its scores:

$$\omega_{loc}(a) = \omega(a) \cdot f_{limits}(a) \cdot f_{head}(a) \cdot f_{obstacle}(a) \tag{9}$$

3.5 Adding the Ball to the Model

The model presented shows a clear strategy to cope with both static and dynamic field information. This allows the robot to keep self-localized despite having several obstacles and to choose the best target under the principles previously exposed. This strategy, however, cannot handle the most important element in the game: the ball.

In order to have a good game performance, all players must maintain self-localized and track the ball frequently. This can be achieve in a simple form by using the same principles used to update the action space. Let Φ_{ball} be a set of actions where the ball appears inside the robot's field of view. This can be calculated by using inverse kinematics as we did for the obstacles. We define a complementary *ball action space* as an action space which handles the ball gazing scores. The scores ω_{ball} are set to 1 if the action is in Φ_{ball}, and 0 if not. This defines a subset of actions where the ball can be seen (Fig. 2 right). This information will be included into the active vision model.

3.6 Action Space-Based Decision Making

Finally, we merge the localization information handled in the *localization action space* with the ball information (Eq. 10). This is done by defining a cost function which provides a new score for each action. In order to cope with the localization and ball gazing *trade-off*, a factor α is introduced to determine ball importance. This factor is modified by the robot's running behaviors depending on the player's role or the game state. A graphical 3D visualization of the final action space is shown in Fig. 3.

$$\Omega(\phi_{pan}, \phi_{tilt}) = \omega_{loc}(\phi_{pan}, \phi_{tilt}) + \alpha\omega_{ball}(\phi_{pan}, \phi_{tilt}) \tag{10}$$

Afterwards, the best action is selected as the action with the maximum fused score associated, as is shown in (11).

$$\phi^*_{pan}, \phi^*_{tilt} = \underset{\phi_{pan}, \phi_{tilt}}{\arg\max}\, \Omega(\phi_{pan}, \phi_{tilt}) \tag{11}$$

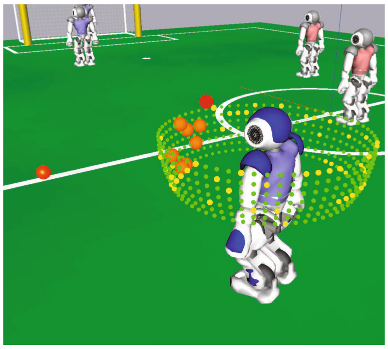


Fig. 3. Resulting action space after information fusion. A red circle denotes the selected target. Orange circles show actions with highest score for each pan angle where the ball can be observed (Color figure online).

4 Experimental Results

In order to test the proposed methodology, we prepared two different experimental settings: the first one is a simulated setup considering ten robots on a RoboCup SPL field, whereas the second one is a real experiment using the robot in a reduced field.

4.1 Simulated Full Field Experiment

The use of simulations allow to carry out repeatable experiments and to explore the use of different sets of parameters. For this reason, the simulated setup was used to analyze the proposed active vision system. Besides the robot under analysis, i.e. the one that uses the robot's vision system, nine other robots were added on the field to represent a real game setup. The robots were located arbitrarily following a common team formation. The ball was located at position (0, 1900) as shown in Fig. 4 left.

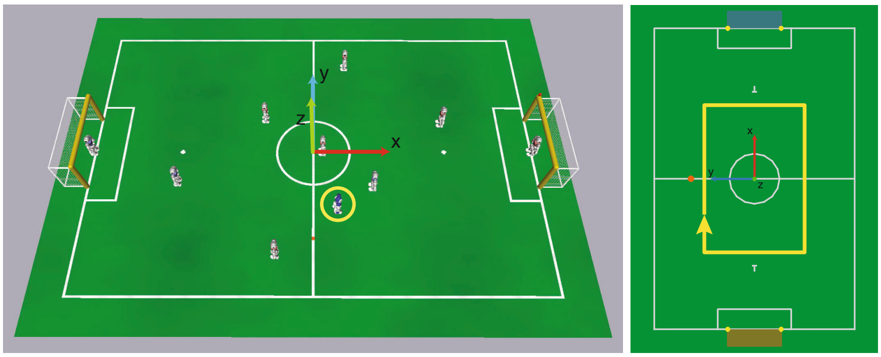


Fig. 4. Left: Simulated experiment configuration. 9 robots were added to represent a real game. The ball was located at position (0, 1900). The robot using the active vision system, indicated with a yellow circle, had to walk 10 times over the shown in the right side. Right: Path followed by the robot in the experiment (Color figure online).

This experiment consisted of following a rectangular path over the full field (Fig. 4 right). The proposed active vision system was compared with a passive one. When using the passive vision system the robot followed a predefined head control routine. It pointed the camera at the estimated ball position for 3 s, and to the other landmarks such as goals and corners for 1 s.

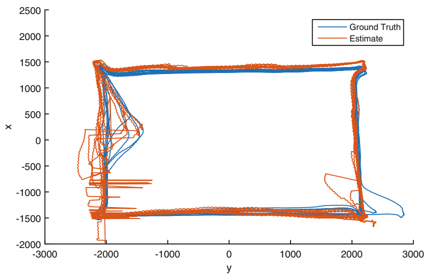
The active vision system used the targets calculated using the scores. The look-up table that stores the results of the particle filters used 40 and 24 bins for the x and y dimensions, whereas 64 and 6 bins where used for the pan and tilt angles respectively. Head pan penalization parameter κ was set to 2 [rads], whereas tilt penalization parameter σ was fixed to 5 [rads]. Obstacle compensation factors κ_0 and σ_0 were set to 1 and 0.05 [rads] respectively, according to experimental observations. The ball importance factor α was varied in several tests.

Translational and rotational errors as well as ball seen percentage were measured for both systems. Main results obtained for several ball importance factor values are shown in Table 1. A graphical comparison of the paths walked using both approaches is shown in Fig. 5.

Table 1. Self-localization errors and ball seen percentage for the simulated setup

	Self-localization			Ball
Head control system	Error (mean, mm)	Error (std, mm)	Error (rad)	Seen (%)
Passive	96.8	139.2	0.18	36.8
Active, $\alpha = 0.5$	44.927	23.926	0.08	22.9
Active, $\alpha = 1$	57.2	35.5	0.11	31.9
Active, $\alpha = 2$	59.4	40.3	0.11	32.8

This quantitative results show that the active vision systems outperforms the passive one with respect to translational and rotational error, in both mean and standard deviation. The proposed active vision method reduced the pose estimation errors in 40 % in average, with respect to the passive system's errors. However, the ball seen percentage depends strongly of the ball importance factor α chosen, being slightly reduced in a 10 % from 36.8 % in the passive case, to 32.8 % for the active system with $\alpha = 2$. Translational and rotational errors are also affected by the ball importance as is expected.



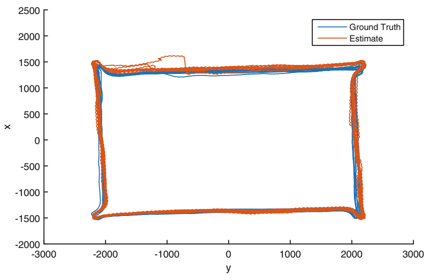


Fig. 5. Left: Comparison of the trajectory followed by the robot while using the passive vision system (in red) and the estimated ground truth position (shown in blue). Erratic behavior is mainly caused by field side ambiguities as well as by obstacle occlusion. Right: Same path using the active system with $\alpha = 2$ (Color figure online).

Most of the errors before changing the robot path direction. Passive vision systems assume full landmark observation and do not consider neither occlusions nor current robot position. Several errors are caused by wrong observations that affect the state estimator, inducing systematic errors in the system. The proposed active vision system can cope with the environment dynamics, choosing a target strongly related with the current robot position. Nevertheless, ball gaze frequency is reduced at the same time, according to the ball importance factor selected.

4.2 Real Robot Experiment

A second test was performed in a half field using a real robot, by utilizing the ground truth system of the RoboCup Small Size League, just as in [11]. The robot was configured with the same parameters as the simulated experiment, using a ball importance factor of 2, and had to follow a straight path between the points $(1000, -1000)$ and $(1000, 1000)$. The ball was positioned at the center of the field. The aim of this experiment was to test the system in both self-localization and ball gazing, as well as to measure the time required for its execution. Main results are shown in Table 2.

Time execution measurements show that the system uses minimal computational resources to run, requiring less than 1 ms. These results were obtained while executing other modules required for a soccer robot, such as perception, self-localization and decision making.

Table 2. Self-localization errors and ball seen percentage for the real setup

	Self-localization			Ball
Head control system	Error (mean, mm)	Error (std, mm)	Error (rad)	Seen (%)
Passive	151.4	86.9	0.58	52.2
Active, $\alpha = 2$	86.6	56.1	0.32	51.9

5 Conclusions and Future Work

In this paper we present a dynamic and efficient approach to the active vision problem focused on robot soccer applications. The method explicitly separates static and dynamic information from the environment, allowing the system to run in real-time by calculating offline expensive procedures, such as simulations of the resulting state estimation for each action candidate. In addition, information representation by the *action space* allows us to choose a head pose considering different information sources such as the position of other robots and the ball, as well as movement constraints of the robot's head. The proposed system could considerably improve the accuracy of the self-localization estimation process by reducing both translation and rotational errors, using low computational resources.

Further developments include an extension of the action space to the real case, which would require a replacement of the particle filter discrete look-up-table used for the *static action space* by an interpolated model or learned model. Since we are coping with the particle filter results, this should not increment the computational cost considerably during the game. In addition, parameter estimation could also be improved through learning algorithms, such as reinforcement learning.

Acknowledgments. The authors thank Pablo Saavedra for his assistance to prepare the ground truth system and the UChile Robotics Team for their general support. We also thank the BHuman SPL Team for sharing their coderelease, contributing the development of the Standard Platform League. This work was partially funded by FONDECYT Project 1130153.

References

1. Aloimonos, J.: Purposive and qualitative active vision. In: Proceedings of the 10th International Conference on Pattern Recognition, 1990, vol. 1, pp. 346–360. IEEE (1990)
2. Bajcsy, R.: Active perception. Proc. IEEE **76**(8), 966–1005 (1988)
3. Ballard, D.H.: Animate vision. Artif. Intell. **48**(1), 57–86 (1991)
4. Burgard, W., Fox, D., Thrun, S.: Active mobile robot localization by entropy minimization. In: Proceedings of the 2nd EUROMICRO Workshop on Advanced Mobile Robots, 1997, pp. 155–162, October 1997
5. Czarnetzki, S., Kerner, S., Kruse, M.: Real-time active vision by entropy minimization applied to localization. In: Ruiz-del-Solar, J. (ed.) RoboCup 2010. LNCS, vol. 6556, pp. 266–277. Springer, Heidelberg (2010)
6. Davison, A.J., Murray, D.W.: Simultaneous localization and map-building using active vision. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 865–880 (2002)
7. Fukase, T., Yokoi, M., Kobayashi, Y., Ueda, R., Yuasa, H., Arai, T.: Quadruped robot navigation considering the observational cost. In: Birk, A., Coradeschi, S., Tadokoro, S. (eds.) RoboCup 2001. LNCS (LNAI), vol. 2377, pp. 350–355. Springer, Heidelberg (2002)
8. Guerrero, P., Ruiz-del-Solar, J., Romero, M.: Explicitly task oriented probabilistic active vision for a mobile robot. In: Iocchi, L., Matsubara, H., Weitzenfeld, A., Zhou, C. (eds.) RoboCup 2008. LNCS, vol. 5399, pp. 85–96. Springer, Heidelberg (2009)
9. Mitsunaga, N., Asada, M.: Visual Attention Control by Sensor Space Segmentation for a Small Quadruped Robot Based on Information Criterion. In: Birk, A., Coradeschi, S., Tadokoro, S. (eds.) RoboCup 2001. LNCS (LNAI), vol. 2377, pp. 154–163. Springer, Heidelberg (2002)
10. Seara, J.F., Schmidt, G.: Intelligent gaze control for vision-guided humanoid walking. Rob. Auton. Syst. **48**(4), 231–248 (2004)
11. Seekircher, A., Laue, T., Röfer, T.: Entropy-based active vision for a humanoid soccer robot. In: Ruiz-del-Solar, J., Chown, E., Plöger, P.G. (eds.) Robocup 2010. LNCS, vol. 6556, pp. 1–12. Springer, Heidelberg (2011)
12. Stronger, D., Stone, P.: Selective visual attention for object detection on a legged robot. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 158–170. Springer, Heidelberg (2007)
13. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press, Cambridge (2005)
14. Vidal-Calleja, T., Davison, A., Andrade-Cetto, J., Murray, D.: Active control for single camera SLAM. In: IEEE International Conference on Robotics and Automation, Orlando, May 2006