Fuzzy Self-Localization using Natural Features on an AIBO Robot

D. Herrero Pérez **H. Martínez Barberá**
Dept. Information and Communication Engineering
University of Murcia
30100 Murcia, Spain
dherrero@dif.um.es humberto@um.es

A. Saffiotti
Dept. of Technology
Örebro University
70218 Örebro, Sweden
asaffio@aass.oru.se

Abstract

In current RoboCup four-legged league the robots rely only on coloured landmarks for localisation. It is intended to remove these landmarks in a few years, as happened in other leagues. The odometry of four-legged robots is extremely unreliable and there is a high uncertainty in robot motion. Therefore, sensor-based localization is necessary. We propose to extend our previous technique for fuzzy self-localization based on landmars, by including observations of the corners between the field lines. We show experimental tests in a simulator using only the field lines and the two nets.

Keywords: Autonomous robots, fuzzy logic, image processing, localization, state estimation.

1 Introduction

RoboCup is an international competition whose final goal is to develop a team of autonomous robots able to compete with the human world champion in 2050. The current soccer field (Fig. 1) in the Four-Legged Robot League has a size of approximately $4,5m \cdot 3m$, and the only allowed robot is the SONY AIBO [10] . The exteroceptive sensor of the robot is a camera, which can detect objects on the field. Objects are color coded: there are six uniquely colored beacons, two goal nets of different color, the ball is orange, and the robots wear colored uniforms.

However, in a real soccer field there are not characteristic colored landmarks and nets neither orange ball. The rules of RoboCup are gradually changed year after year in order to push progress toward the final goal. Removal of the artificial colored beacons will be the next step in this direction. Accordingly, some preliminary development has been done by some teams in this league to allow the robot to self-localize without using the artificial beacons (e.g., [4]).

This paper describes the process of self-localization using the field lines as a source of features. The technique proposed is based on [5], which use fuzzy logic to account for errors and imprecision in visual recognition of landmarks and nets, and for uncertainly estimate of robot's displacement. The technique allows for large odometric errors and inaccurate exteroceptive detections with excellent results. Experiments run on a simulator are included in the paper. Experiments on real robots are underway.

2 Perception

The AIBO robots [10] use a CCD camera as the only exteroceptive sensor. The perception process is in charge of extracting convenient features of the environment from the images provided by the camera. As the robot will localize relying on the extracted features, both the amount of features detected and their quality will clearly affect the process. Cur-

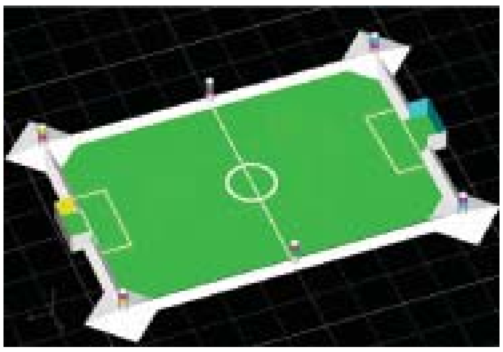
Figure 1: RoboCup 2003 official field for AIBO robots.

rently the robots rely only on coloured landmarks for localization, and thus the perception process is based on color segmentation for detecting the different landmarks. In a few years the landmarks will be not used anymore, as happened in other leagues. For this reason, our short term goal is augmenting the current landmark based localization with information obtained from the field lines, and our long term goal is relying only on the field lines.

Because of the League rules all the processing must be done on board and for practical reasons it has to be performed in real time, which prevent use from using time consuming processing algorithms. A typical approach for detecting straight lines in digital images is the Hough Transform and its numerous variants. The various variants have been developed to try to overcome the major drawbacks of the standard method, namely, its high time complexity and large memory requirements. Common to all these methods is that either they may yield to erroneous solutions or they have a high computational load.

Instead of using the field lines as references for the self localization, we decided to use the corners produced by the intersection of the field lines (which are white). The two main reasons for using corners is that they can be labeled (depending on the type of intersection) and they can be tracked more appropriately given the small field of view of the camera. In addition, detecting corners can be more computationally efficient than detecting lines. There are several approaches, as reported in the literature, for detecting corners. They can be broadly divided into two groups:

- Extracting edges and then searching for the corners. An edge extraction algorithm is pipelined with a curvature calculator. The points with maximum curvature (partial derivate over the image points) are selected as corners.
- Working directly on a grey-level image. A matrix of cornerness of the points is computed (product of gradient magnitude and the rate of change of gradient direction), and then points with a value over a given threshold are selected as corners.

Because of performance reasons, we have evaluated two algorithms that work on grey-level images and detect corners depending on the brightness of the pixels, either by minimizing regions of similar brightness (SUSAN) [8] or by measuring the variance of the directions of the gradient of brightness [9]. These two methods produce corners, without taking into account the color of the regions. As we are interested in detecting field lines corners, the detected corners are filtered depending on whether they come from a white line segment or not.

In the first method (SUSAN) [8] , non-linear filtering is used to define which parts of the image are closely related to each individual pixel; each pixel has associated with it a local image region which is of similar brightness to that pixel. The detector is based on the minimization of this local image region. The local area contains much information about the structure of the image. From the size, centroid and second moments of this local area corners and edges can be detected. This approach has many differences to the well-known methods, the most obvious being that no image derivatives are used and that no noise reduction is needed. The integrating effect of the principle, together with its non-linear response, give strong noise rejection. The local area is at a maximum when

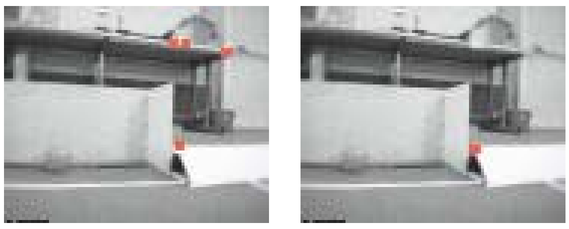

Figure 2: Brightness similarity based detector. (a) without color based filtering (b) with color based filtering.

the nucleus lies in a flat region of the image surface, it falls to half of this maximum very near a straight edge, and falls even further when inside a corner. It is this property which is used as the main determinant of the presence of edge and corner features. The output of this method is shown in Fig. 2.

The second method [9] is based on measuring the variance of the directions of the gradient of brightness. The probability of the event that a point belongs to the approximation of a straight segment of the isoline of brightness passing through the point being tested is computed using the technique of Bayesian estimations and used as a weight. Along every isoline, the size of the gradient of brightness remains constant. The two half-lines that form the isoline along which the size of the gradient of brightness is maximal are edges. Along each half-line that is a part of an isoline, the direction of the gradient of brightness remains constant. The directions of the gradient are different at the points lying to the left and to the right of the corner axis. The image points at which the size of the gradient of brightness is greater than a predefined threshold are considered to be candidates for corners. The candidates are then examined, the candidate at which the value of corner measure exhibits its local maximum and at which the values of angle of break, and corner appearance are greater than chosen thresholds is a corner. The output of this method is shown in Fig. 3.

The gradient based method is more parameterizable, produces more candidate points and

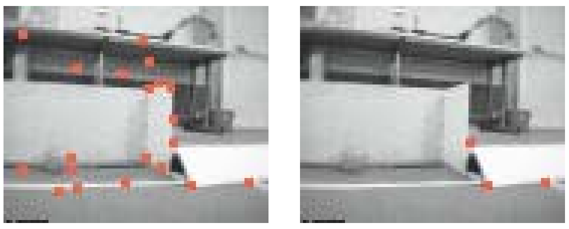

Figure 3: Brightness gradient based detector. (a) without color based filtering (b) with color based filtering.

requires similar processing capabilities, and thus it is the one implemented in the robots for corner detection. As mentioned above, the corners are filtered so that we keep only the relevant ones, that is, those produced by the white field lines. These are then labeled according to the amount of field (carpet) and line-band (white), to produce the following categories:

Open Corner Pixel with a lot of carpet around it, and line or band over a threshold.

Closed Corner Pixel with a lot of line or band around it, and carpet over a threshold.

Net Closed Corner Pixel with a lot of line or band around it, carpet and a net over a threshold.

In order to classify the detected corners, it is only required to look to the pixels in the neighbourhood. After the detected corners have been classified, the following features are then obtained:

Type C An **Open corner** nearby of a **closed corner**. This feature can be detected in the goal keeper area of field.

Type T of field Two **closed corner**. Produced by the intersection of the walls and the inner field lines.

Type T of net A **closed corner** nearby of a **net closed corner**. Produced by the intersection of goal field lines and the net.

These features, together with the landmarks and the nets, are used for localizing a robot in the field. The next sections present both the representation of the uncertainty associated to each feature and the way they are combined for obtaining an estimate of the robot position.

3 Uncertainty Representation

3.1 Fuzzy locations

Location information may be affected by different types of uncertainty, including vagueness, imprecision, ambiguity, unreliability, and random noise. An uncertainty representation formalism to represent locational information, then, should be able to represent all of these types of uncertainty and to account for the differences between them. Fuzzy logic techniques are attractive in this respect [6]. We can represent information about the location of an object by a fuzzy subset μ of the set X of all possible positions [11, 12]. For instance, X can be a 6-D space encoding the (x, y, z) position coordinates of an object and its (θ, ϕ, η) orientation angles. For any $x \in X$, we read the value of $\mu(x)$ as the degree of possibility that the object is located at x given the available information.

Fig. 4 shows an example of a fuzzy location, taken in one dimension for graphical clarity. This can be read as "the object is believed to be approximately at θ, but this belief might be wrong". Note that the unreliability in belief is represented by a uniform bias b in the distribution, indicating that the object might be located at any other location. Total ignorance in particular can be represented by the fuzzy location $\mu(x) = 1$ for all $x \in X$.

3.2 Representing the robot's pose

Fuzzy locations can be represented in a discretized format in a position grid: a tessellation of the space in which each cell is associated with a number in $[0, 1]$ representing the degree of possibility that the object is in that cell. In our case, we use a 3D grid to represent the robot's belief about its own pose,

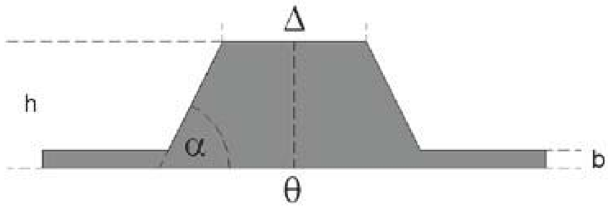

Figure 4: Fuzzy set representation of an angle measurement θ.

that is, its (x, y) position plus its orientation θ. A similar approach, based on probabilities instead of fuzzy sets, was proposed in [3].

This 3D representation has the problem of having a high computation complexity, both in time and space. To reduce complexity, we adopt the Approach proposed by [5]. Instead of representing all possible orientations in the grid, we use a 2D grid to represent the (x, y) position, and associate each cell with a trapezoidal fuzzy set $\mu_{x,y} = (\theta, \Delta, \alpha, h, b)$ that represents the uncertainty in the robot's orientation. Fig. 4 shows this fuzzy set. The θ parameter is the center, Δ is the width of the core, α is the slope, h is the height and b is the bias. The latter parameter is used to encode the unreliability of our belief.

For any given cell (x, y), $\mu_{x,y}$ can be seen as a compact representation of a possibility distribution over the cells $(x, y, \theta)|\theta \in [-\pi, \pi]$ of a full 3D grid. The reduction in complexity is about two orders of magnitude with respect to a full 3D representation (assuming a angular resolution of one degree). The price to pay is the inability to handle multiple orientation hypotheses on the same (x, y) position — but we can still represent multiple hypotheses on different positions. In our domain, this restriction is acceptable.

3.3 Representing the observations

An important aspect of our approach is the way to represent the uncertainty of observations. Suppose that the robot observes a given feature at time t. The observed range and bearing to the feature is represented by a vector $\overrightarrow{r}$. Knowing the position of the feature in the map, this observation induces in the robot a belief about its own position in the environment. This belief will be affected by uncertainty, since there is uncertainty in the observation.

In our domain, we consider three main facets of uncertainty. First, *imprecision* in the measurement, i.e., the dispersion of the estimated values inside an interval that includes the true value. Imprecision cannot be avoided since we start from discretized data (the camera image) with limited resolution. Second, *unreliability*, that is, the possibility of outliers. False measurements can originate from a false identification of the feature, or from a mislabelling. Third, *ambiguity*, that is, the inability to assign a unique identity to the observed feature since features (e.g., corners) are not unique. Ambiguity in observation leads to a multi-modal distribution for the robot's position.

All these facets of uncertainty can be represented using fuzzy locations. For every type of feature, we represent the belief induced a time t by an observation $\overrightarrow{r}$ by a possibility distribution $S_t(x, y, \theta|\overrightarrow{r})$ that gives, for any pose (x, y, θ), the degree of possibility that the robot is at (x, y, θ) given the observation $\overrightarrow{r}$. This distribution constitutes our *sensor model* for that specific feature.

The shape of the $S_t(x, y, \theta|\overrightarrow{r})$ distribution depends on the type of feature. In the case of net observations, this distribution is a circle of radius $|\overrightarrow{r}|$ in the (x, y) plane, blurred according to the amount of uncertainty in the range estimate. Fig. 5 and 6 show an example of this case. In the figure, darker cells indicate higher levels of possbility. We only show the (x, y) projection of the possibility distributions for graphical clarity.

Note that the circle has a roughly trapezoidal section. The top of trapezoid (core) identifies those values which are fully possible. Any one of these values could equally be the real one given the inherent imprecision of the observation. The base of the trapezoid (support) identifies the area where we could still possibly have meaningful values, i.e., values outside this area are impossible given the observation. In order to account for unreliability, then, we include a small uniform bias, representing the degree of possibility that the robot is "somewhere else" with respect to the measurement.

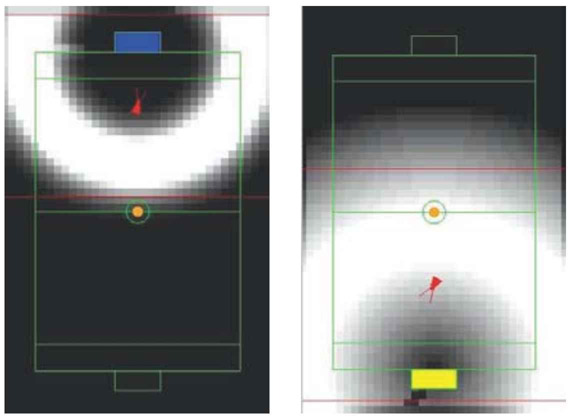
(a) (b)

Figure 5: Belief induced by the observation of a blue net (a) and a yellow net (b). The arrow marks the center of gravity of the grid map, indicating the most likely robot localization.

The $S_t(x, y, \theta|\overrightarrow{r})$ distribution induced by a corner observations is the union of several circles, each centered around a corner in the map, since our simple corner detector does not give us a unique ID for corners. Fig. 6 shows an example of this. It should be noted that the ability to handle ambiguity in a simple way is a distinct advantage of our representation. This means that we can do not need to deal separately with the data association problem, but this is automatically incorporated in the fusion process (see below). Data association is one of the unsolved problems in most current self-localization techniques, and one of the most current reasons for failures.

4 Fuzzy Self-Localization

Our approach to feature-based self-localization extends the one proposed by Buschka *et al* in [5], who relied on unique

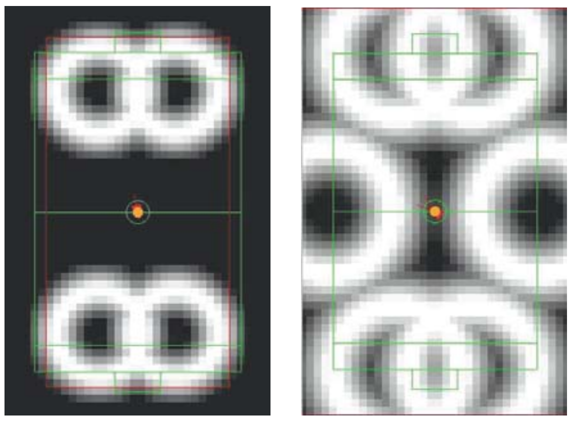

Figure 6: Belief induced by the observation of a feature type Cs (a) and Ts of field (b). Due to symmetry of the field, the center of gravity is close to the middle of the field.

artificial landmarks. Buschka's approach combines the Markov localization technique proposed by Burgard in [3] with the fuzzy landmark-based localization technique proposed by Saffioti and Wesley in [7].

The robot's belief about its own pose is represented by a distribution G_t on a $2\frac{1}{2}$D possibility grid as described in the previous section. This representation allows us to represent, and track, multiple possible positions where the robot might be. When the robot is first placed on the field, G_0 is set to 1 everywhere to represent total ignorance. This belief is then updated according to the typical predict-observe-update cycle of recursive state estimators as follows.

Predict When the robot moves, the belief state G_{t-1} is updated to G_t using a model of the robot's motion. This model performs a translation and rotation of the G_{t-1} distribution according to the amount of motion, followed by a uniform blurring to account for uncertainty in the estimate of the actual motion.

Observe The observation of a feature at time t is converted to a possibility distribution on the $2\frac{1}{2}$ grid using the sensor model discussed above. For each pose (x, y, θ), this distribution measures the possibility of the robot being at that pose given the observation.

Update The possibility distribution generated by each observation at time t is used to update the belief state G_t by performing a fuzzy intersection with the current distribution in the grid at time t. The resulting distribution is then normalized.

If the robot needs to know the most likely position estimate at time t, it does so by computing the center of gravity of the distribution G_t. A reliability value for this estimate is also computed, based on the area of the region of G_t with highest possibility and on the minimum bias in the grid cells. This reliability value is used, for instance, to decide to engage in an active relocalization behavior.

In practice, the predict phase is performed using tools from fuzzy image processing, like fuzzy mathematical morphology, to translate, rotate and blur the possibility distribution in the grid [1, 2]. The intuition behind this is to see the fuzzy position grid as a grayscale image.

For the update phase, we update the position grid by performing pointwise intersecftion of the current state G_t with the observation possibility distribution $S_t(\cdot|r)$ at each cell (x, y) of the position grid. For each cell, this intersection is performed by intersecting the trapezoid in that cell with the corresponding trapezoid generated for that cell by the observation. This process is repeated for all available observations. Intersection between trapezoids, however, is not necessarily a trapezoid. For this reason, in our implementation we actually compute the outer trapezoidal envelope of the intersection. This is a conservative approximation, in that it may over-estimate the uncertainty but it does not incur the risk of ruling out true possibilities.

There are several choices for the intersection operator used in the update phase, depending on the independence assumptions that we can make about the items being combined. In our case, since the observations are independent, we use the product operator which reinforces the effect of consonant observations.

Our self-localization technique has nice computational properties. Updating, translating, blurring, and computing the center of gravity (CoG) of the fuzzy grid are all linear in the number of cells. In the RoboCup domain we use a grid of 36×54, corresponding to a resolution of 10 cm (angular resolution is unlimited since the angle is not discretized). All computations can be done in real time using the limited computational resources available on-board the AIBO robot.

5 Experimental results

To show how the self-localisation process works, we will present an example generated from a simulator. Let's suppose that the robot starts in a position around the mid-field, more or less facing the yellow net. At this moment the robot has a belief distributed along the full field – it does not know its own location. Then the robot starts scanning its surroundings by moving its head from left to right. As soon a feature is perceived, it is incorporated into the localization process.

When scanning, the robot first detects two corners (Fig. 7). The outlined triangle represents the real location of the robot, while the filled one represents the current estimate. Then the beliefs associated to the detection are fused (Fig. 8). Due to the high symmetry of the environment, there are two candidate positions, each one in a different side of the field.

In order to cope with the natural symmetry of the field, we use unique features, like the nets are. When the robot happens to detect the yellow net (Fig. 9), it helps to identify in which part of the field the robot is currently in, and the fusion with the previous corner based location (Fig. 9) gives a fairly good estimate of the robot position.

6 Conclusions

The fuzzy position grid, which has shown to work in real matches using colored landmarks and the nets [5], provides a viable solution to

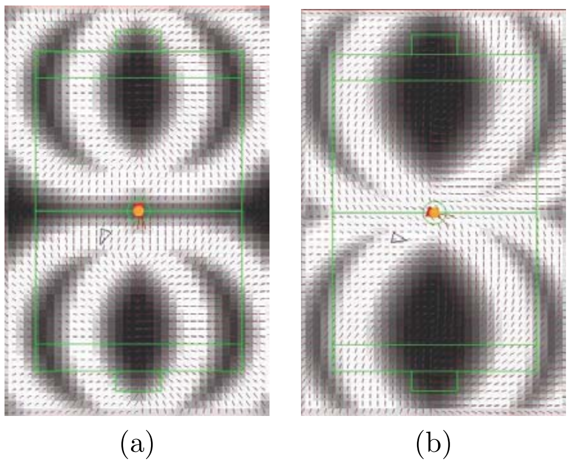
Figure 7: Belief induced by the observation of (a) the first corner, and (b) the second corner.

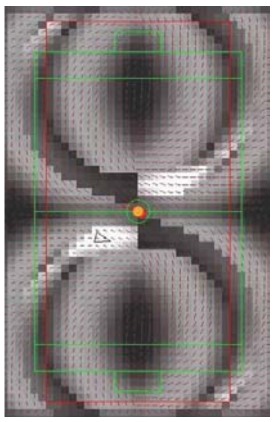
Figure 8: Fusion of the belief associate to the two corners.

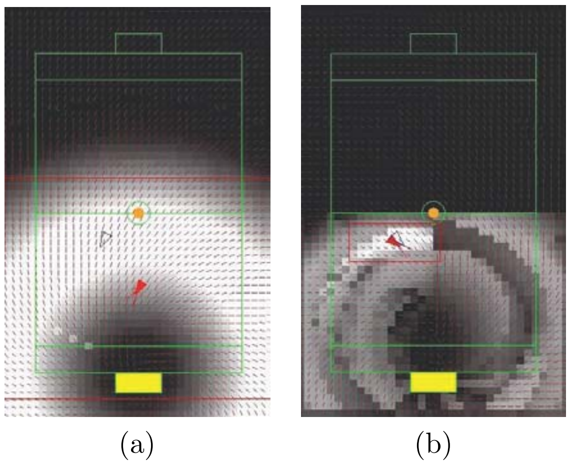
Figure 9: (a) Belief induced by the observation of the yellow net, and (b) fusion of the previous information.

the problem of localization of a legged robot in the Robocup domain. In this environment motion estimates are highly unreliable, observations are uncertain, accurate sensor models are not available, and real time operation is of essence. In this paper we extend and use the position grid to work with corners. We use gray-level image processing for detecting corners and color based filtering to reject corners that do not come from field lines.

The main advantage of this approach, given the current RoboCup rules, is the ability of having more references for guidance, and thus the amount of time spent for looking for the colored landmarks is reduced because when the robot is playing for the ball or aiming at a net, at the same time it can detect corners. In addition, in the near future the rules will eliminate the colored landmarks, and then these kind of techniques will be of paramount importance. In this paper we show experimental results of the technique, while real world experiments show quite promising results.

Acknowledgements

This work has been supported by the CICYT project TIC 2001-0245-C02-01, Spanish Ministry of Science and Technology, and by the Swedish KK Fundation. Our thanks to Angel Gil García for implementing the vision routines.

References

[1] I. Bloch and H. Maître. Fuzzy mathematical morphologies: a comparative study. *Pattern Recognition*, 28(9):1341–1387, 1995.

[2] I. Bloch and A. Saffiotti. Why robots should use fuzzy mathematical morphology. In *Proc. of the 1st Int. ICSC-NAISO Congress on Neuro-Fuzzy Technologies*, La Havana, Cuba, 2002. Online at http://www.aass.oru.se/˜asaffio/.

[3] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probabilitygrids. In *Proc. of the National Conferenceon Artificial Intelligence*, 1996.

[4] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Fast and robust edge-based localization in the sony four-legged robot league. In *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Padova, Italy, 2003.

[5] P. Buschka, A. Saffiotti, and Z. Wasik. Fuzzy landmark-based localization for a legged robot. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1205–1210, Takamatsu, Japan, 2000. Online at http://www.aass.oru.se/˜asaffio/.

[6] A. Saffiotti. The uses of fuzzy logic in autonomous robot navigation. *Soft Computing*, 1(4):180–197, 1997. Online at http://www.aass.oru.se/˜asaffio/.

[7] A. Saffiotti and L. P. Wesley. Perception-based self-localization using fuzzy locations. In *Proc. of the 1st Workshop on Reasoning with Uncertainty in Robotics*, pages 368–385, Amsterdam, NL, 1996.

[8] S.M. Smith and J.M. Brady. Susan - a new approach to low level image processing. *International Journal of Computer Vision*, 1(23):45–78, 1997.

[9] E. Sojka. A new and efficient algorithm for detecting the corners in digital images. In *Proc. 24th DAGM Symposium*, pages 125–132, Springer, LNCS 2449, Berlin, NY, 2002.

[10] Sony. Sony aibo robots. Online at http://www.aibo.com.

[11] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[12] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.