Self-Localization in Dynamic Environments based on Laser and Vision Data

Erik Schulenburg, Thilo Weigel, Alexander Kleiner
Institut für Informatik
Universität Freiburg
79110 Freiburg, Germany
{*schulenb, weigel, kleiner*}*@informatik.uni-freiburg.de*

Abstract

For a robot situated in a dynamic real world environment the knowledge of its position and orientation is very advantageous and sometimes essential for carrying out a given task. Particularly, one would appreciate a *robust*, *accurate* and *efficient* self-localization method which allows a global localization of the robot. In certain polygonal environments a laser based localization method is capable of combining all these properties by correlating observed lines with an *a priori* line model of the environment [5]. However, often line features can rather be detected by a vision system than by a laser range finder. For this reason we propose an extension of the laser based approach for the simultaneous use with lines detected by an omni-directional camera. The approach is evaluated in the RoboCup domain and experimental evidence is given for its robustness, accuracy and efficiency, as well as for its capability of global localization.

1 Introduction

For a robot, the knowledge of its position and orientation is very advantageous and sometimes essential for carrying out a given task. In general, this can be achieved by a self-localization method that integrates sensor and odometry data over time. The challenge here is to estimate the robot's current pose accurately and robustly despite of sensor noise and ambiguous observations.

In dynamic environments, such as robotic soccer in the RoboCup F2000 league, an additional challenge is to carry out this process within a short amount of time. When robots move faster than a meter per second decisions have to be made within a few milliseconds. In such dynamic environments, one has to find a good trade-off between *robustness*, *accuracy* and *efficiency*.

In this paper we contribute a robust and accurate self-localization method that determines the robot's pose by fusing features detected by a laser range finder (LRF) and an omni-directional vision system. The method exploits the polygonal structure of the environment by matching straight lines extracted from the sensor data with an *a priori* line model. We adapted and enhanced the efficient *LineMatch* algorithm that was originally utilized for matching lines extracted from LRF scans [5]. The proposed *Extended LineMatch* algorithm localizes the robot with comparable efficiency and can be carried out with lines from either one or both sensors simultaneously. Furthermore, by incorporating vision data, the *Extended LineMatch* algorithm allows for a unique global localization of the robot in the RoboCup domain. The line extraction carried out by the omni-directional vision system is additionally supported by a specific mirror design as proposed by Marchese *et al.* [10].

Previous research on vision-based localization in the context of office-like environments focused particularly on robustness [11; 16; 4]. The proposed methods are mainly appearence based and carried out by e.g. applying Principal Component Analysis (PCA) on images taken at particular positions in the environment. The global position of the robot is then determined by the correspondence between images and their topological representation in an eigenspace created by the PCA. Arras *et al.* proposed a robust and accurate localization method that combines extracted lines from a camera and an LRF. The method does not support global localization in their test domain [1].

Previous research in the RoboCup domain focused particularly on efficiency. Iocchi *et al.* and Lima *et al.* favor the use of the *Hough transform* for correlating image lines and model lines in the hough domain. Their method is efficient, but relies on an already good estimate of the position and the orientation of the robot and can't recover from heavy localization errors [8; 9]. Restelli *et al.* utilize evidence accumulation based on a fixed discretization of the hypotheses space [12]. Their results indicate a strong correlation between accuracy and efficiency. Hanek *et al.* match line segments with a 3D model of the field. But the use of only a single frontal camera makes their approach vulnerable to not perceiving the necessary number of features [7].

The proposed method has been evaluated regarding efficiency, accuracy and robustness in the realistic scenario of the RoboCup F2000 league. Experimental results show that our method represents a good trade-off between the three properties. Furthermore, results demonstrate that the method is capable of solving the *wake-up problem* where global localization is carried out without prior knowledge about the initial pose.

The remainder of this paper is structured as follows. In the next section we briefly summarize the omni-directional camera system we use. Section 3 shows the filtering of lines extracted from the sensor data. In Section 4 we describe the *Enhanced LineMatch* algorithm for the simultaneous use of vision- and laser range finder data. Section 5 reports results from a series of experiments evaluating our approach. In Section 6 we conclude and give an outlook on future work.

2 The Omni-Vision System

When designing a vision system, one usually has to find an appropriate trade-off between accuracy and the available field of view. One advantage of omni-directional vision systems is that this trade-off can be biased towards the own requirements by an adequate mirror design. Following the suggestions from Marchese and colleagues [10], we designed our omni-vision system (see Figure 1(a)) according to the following requirements of the RoboCup-domain: Firstly, objects in the near vicinity of the robot have to be recognized with high accuracy in order to facilitate obstacle avoidance and handling of the ball. Secondly, far-off or tall

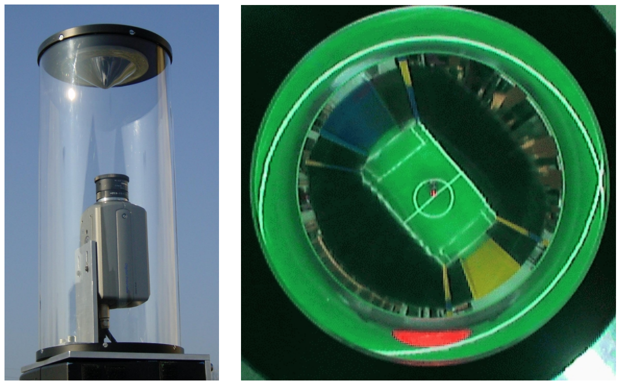
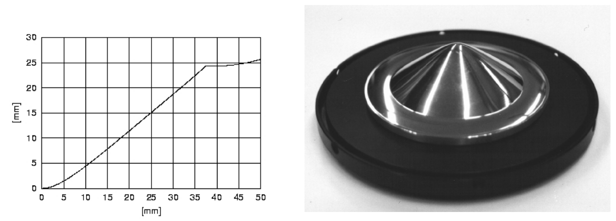

(a) (b)

Figure 1: (a) The omni-directional camera system and (b) an image taken close to the center circle.

objects, e.g. other robots, have to be recognized, whereas accuracy plays not an important role. Thirdly, features providing evidence of the robot's location, e.g. field lines, have to be recognized with a constant distance error in order to simplify the self-localization process.

Based on these requirements we calculated a mirror profile which is composed of three parts [10]. The first part is *isometric* and shaped such that it removes distortion due to the mirror projection. This part allows a linear mapping from objects on the field to the camera image up to a distance of 6 meters as shown around the center in Figure 1(b). The second part is designed with *constant curvature* and does not remove the distortion of the image, but allows a reliable detection of high and far-off objects with a maximum height of 0.6 meters and up to 10 meters away. The third part, designed with curvature as well, allows an accurate detection of objects within a range of $0.2m$ and $0.8m$ as shown for the ball in the outer region of Figure 1(b). Figure 2 shows the calculated profile and a prototype of the mirror. From a camera image such as the one shown in Figure 1(b) relevant features useful for self-localization, namely lines and regions of pre-defined colors (e.g. the blue and yellow goals in the Robocup-domain) are extracted. This is done by firstly classifying the image pixels utilizing the *CMVision* library [2], which also provides sets of regions (blobs) representing clustered pixels of the same color.



(a) (b)

Figure 2: (a) The calculated profile and (b) the manufactured mirror.

The extraction of the field lines is carried out by, firstly detecting the color transitions in the classified image and secondly, by building lines from these transitions. In order to allow the use of a fast divide and conquer line extraction method that was originally developed for LRFs [3; 6], transitions are sorted by their angle, similar as range measurements are provided by a LRF. For this reason, transitions are detected by raytracing with a 2° resolution from the image center to the end of the isometric projection of the mirror as shown by Algorithm 1 [13].

Algorithm 1 *OmniScan($\mathcal{B}$)*

```
Input: Color segmented image $\mathcal{B}$
Output: Lists of transitions $\mathcal{T}$ in polar coordinates
  $\mathcal{T}_i := empty, \forall i \in \{1, 2, .., max\}$
  for all $\phi$ do
    $layer := 0;$
    $pixel := CenterOfImage(\mathcal{B})$
    while $pixel \in IsometricPart(\mathcal{B})$ do
      if $isTransition(pixel)$ then
        $layer := layer + 1$
        $\mathcal{T}_{layer} := \mathcal{T}_{layer} \cup (d(pixel), \phi)$
      endif
      $pixel := NextPixel(pixel, \phi)$
    endwhile
  endfor
  return $\mathcal{T}$
```

The function $isTransition(pixel)$ detects if there is a color change to a certain color (e.g. white for field lines) at the current pixel. The function $NextPixel(pixel, \phi)$ returns the next pixel in direction of ϕ. For pixels outside the isometric part of the mirror, $pixel \in IsometricPart(\mathcal{B})$ returns $false$. Both functions are efficiently realized as pre-computed lookup tables.

Found transitions are stored by their polar coordinates in a transition list $\mathcal{T}_1$. After a transition has been found, the search continues along the same ray and subsequently found transitions are stored in the transition lists $\mathcal{T}_2, \mathcal{T}_3 ... \mathcal{T}_{max}$ respectively. We name the nth transition list the nth layer of the omni-scan. Figure 3 shows the four layers of an omni-scan that has been generated from the image shown in Figure 1(b). Due to the isometric shape of the mirror, pixel distances are easily mapped to real world distances by multiplication with a constant value.

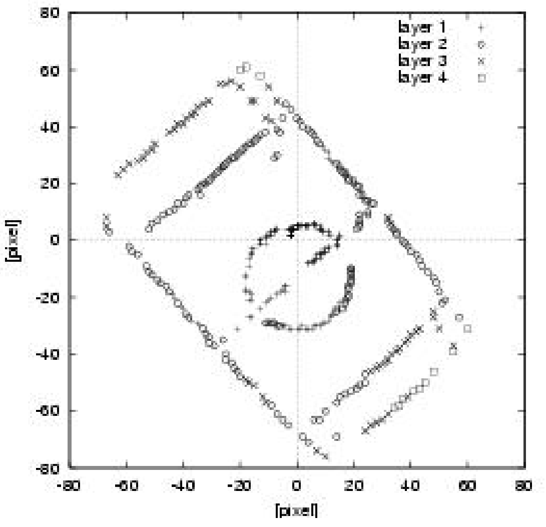


Figure 3: Four layers of transitions detected from field lines.

3 Line Filtering

From the laser range measurements and the layers of transitions found in the camera image, line segments are extracted in time $O(n \log n)$ where n is the total number of data points [3; 5; 6]. We apply a series of filters to these line segments for obtaining line data that is more suitable for our localization method.

LRF line filtering: It is assumed that valid line segments provided by the LRF sensor can be associated with polygonal objects of a specific color. For example, line segments generated from the two goals on the soccer field can either be associated with the color blue or yellow. Based on this assumption, line segments which cannot be associated with one of the pre-defined colors are filtered out. Moreover, line segments that are significantly shorter or longer than the given model lines, are filtered out as well.

Vision line filtering: As shown in Figure 4(a), the result of the vision line extraction algorithm can also be improved. Sensor noise but also the center circle caused non-perpendicular lines. Furthermore, lines were partially split into smaller segments. In order to improve the quality of the vision line set, we apply an orthogonality filter and merge, if appropriate, segments to lines. Of course, this can only be done if all model lines are either parallel or perpendicular. The orthogonality filter generates a histogram above all angles modulo $90°$ which accumulates the length of each line at the position of its angle and angles adjacent within a tolerance of $10°$. By this, long lines, whose angle estimates are usually more accurate, get a stronger weight than short lines. The angle at the maximum of the histogram then represents the principal axis of the lines and consequently all lines which deviate too much from this direction can be removed. The result of applying the orthogonality filter to the line set from Figure 4(a) is shown in Figure 4(b). In the following, co-linear lines (within a certain tolerance) are merged if their distance is within a certain range. The resulting line set is shown in Figure 4(c).

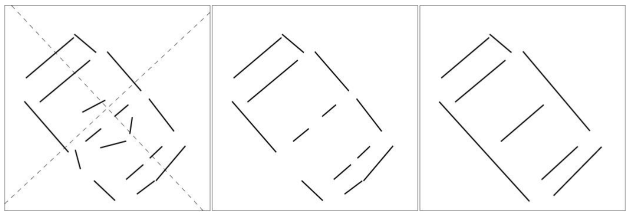


Figure 4: Lines extracted from the transition layers of Figure 3. (a) without filtering (dashed lines indicate the principal axes), (b) after applying the orthogonality filter and (c) after merging segments located on the same straight line.

4 The Extended LineMatch Algorithm

The *LineMatch* algorithm determines a set of position hypotheses by correlating lines extracted from the sensor data with a model containing the location of significant lines in the environment. For determining the currently best pose estimate the most plausible hypothesis is taken and fused with odometry using an extended Kalman filter [5].

With field lines extracted from the camera image and preprocessed as described in the previous section, the original algorithm [5] can be applied without modifications. Recursively all pairings between observed field lines and *a priori* model lines are examined:

```
Algorithm 2 LineMatch(M, L, P)
Input: model lines $M$, detected lines $L$, pairs $P$
Output: set of positions hypotheses $H$
  if $|P| = |L|$ then
    $H := \{P\}$
  else
    $H := \emptyset$
    $l := SelectLine(L, P)$
    for all $m \in M$ do
      if $VerifyMatch(M, L, P \cup \{(m, l)\})$ then
        $H := H \cup LineMatch(M, L, P \cup \{(m, l)\})$
      endif
    endfor
  endif
  return $H$
```

SelectLine selects the next line that should be matched and *VerifyMatch* verifies that the new (m, l) pairing is compatible with the set of pairings P already accepted.

However, we now allow that a fixed number of observed lines can be *outliers*[1]. Since such a line is ignored by *VerifyMatch*, the algorithm can better cope with false line readings and inacurracies in the line extraction process. To incorporate the outlier-match we simply augment the set of model lines by an extra ”outlier line”: $M := M \cup \{outlier\}$.

When called as $LineMatch(M, L, \{\})$ the algorithm returns possible mappings between observed lines L and

[1]In our experiments we allowed up to two outlier lines

model lines M. As such a mapping implies a position where the image possibly has been taken, each mapping represents a hypothesis of where the robot was located at this time. Each hypothesis can be evaluated in terms of the deviations between

- the input lines and the matched model lines
- the observed and the estimated angles to prominent color regions (e.g. goals)
- the estimated pose and the current pose estimate from the Kalman filter

The most plausible hypothesis is now taken as the one which maximizes the weighted average between these influence factors. Doing so always ensures reliable position tracking once the robot is globally localized. However, in the RoboCup environment, at many locations enough field lines and at least one goal is visible so that even the robot's global position on the field can uniquely be determined.

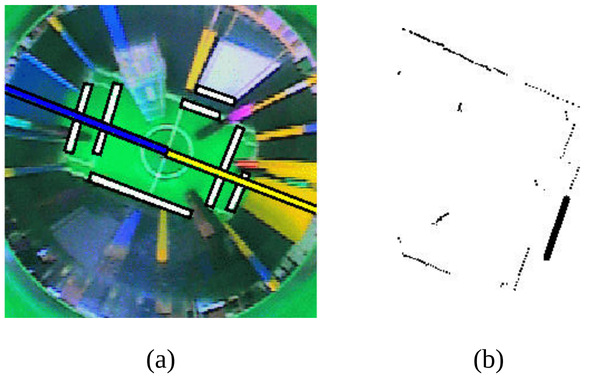
We further enhanced the basic algorithm to deal with field lines extracted from the camera image and lines extracted from a laser scan at the same time[2]. For example, in the RoboCup F2000 league, goal boxes or sponsor logos can be detected by a laser range finder with high precision. The extension is accomplished by introducing different types for lines provided by the camera system and the laser range finder, respectively. Now, *VerifyMatch* additionally checks the line types and discards a set of pairings P as soon as it contains a pairing with differing line types.

Basically, *VerifyMatch* rates a match based on the sum of the deviations between input lines and their corresponding model lines. This allows to account for the different precisions of the sensors by simply weighting these deviations according to the involved line type.

Depending on the available features in the environment either laser range finder data only or vision data only or a combination of both can be utilized for the self-localization process. The only thing that has to be changed is the hand crafted line model of the environment. Figure 5(a) shows a typical scene, including extracted features, in the RoboCup environment as perceived by the omni-directional vision system. Figure 5(b) shows the same scene from the point of view of the laser range finder, inluding one extracted line of a goal box. Figure 5(c) shows the resulting pose estimate based on the scan- and vision lines. Note, that the color of the top line indicates a line considered to be an outlier.

Even though at first sight the algorithm's complexity is $O((|M||L|)^{|L|})$ (*VerifyMatch* requires $O(|L|)$ time in general) it can be shown that its upper bound can be assessed as $O(|M|^3|L|^2)$ if no outlier lines are allowed [5]. The rationale behind this is that only in the first two recursive steps all possible pairings have to be considered. After that, all degrees of freedom for rotating and translating the input lines are fixed and further recursive levels only verify that match. However, if outliers are admitted, the degrees

[2]Camera and laser range finder are synchronized by projecting their reference positions to the same point in time based on information from odometry



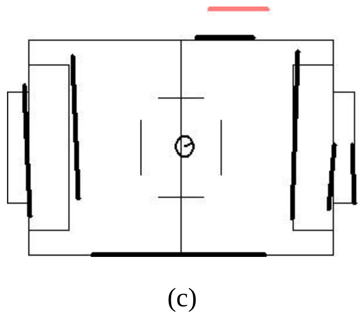


Figure 5: (a) Lines and goal orientations extracted from the camera image, (b) a line extracted from a laser scan taken in the same scene and (c) the resulting pose estimate with an outlier at the top.

of freedom might not be fixed until all outliers are determined. For this, with a maximum of k outliers, $|M|^{k+2}$ recursive calls might be necessary. Subsequently, for each pairing another $|M||L|$ mappings have to be considered for verifying the match. Since for the first $k+2$ recursive calls *VerifyMatch* requires only $O(k+2)$ time, the worst case upper bound can be assessed as $O(((k+2)|M|)^{k+3}|L|^2)$.

5 Empirical Results

In order to verify the efficiency, accuracy and robustness of the *Enhanced LineMatch* approach we conducted a series of experiments with one of our CS Freiburg players [15] (Figure 6(a)) in a typical RoboCup scenario (Figure 6(b)). The player is a *Pioneer I* mobile robot equipped with a

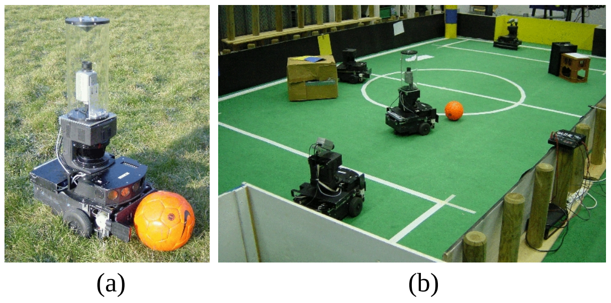


Figure 6: (a) A CS Freiburg player, (b) the experimental setup.

Sony Vaio PCG-C1VE notebook (600 MHz, 112MB), a custom-made kicking device, a *SICK LMS200* laser range finder and a custom made omni-directional vision system based on a *Sony DFW-V500* digital camera providing YUV-images at a 640x480 pixel resolution with 30 frames per second. The laser range finder provides depth information for a $180°$ field of view with an angular resolution of $0.5°$ and an accuracy of 1cm. We collected vision and laser range finder data in in a realistic game scenario with stationary and moving obstacles. For obtaining reference positions we set up walls around the field and carried out the original laser-based *LineMatch*-localization method while collecting the data. As the walls may filter out noise which usually arises from objects outside the field we randomly put a couple of colored labels along the walls for confusing the vision system. By adding stationary obstacles close to the walls we ensured that only rather short lines can be extracted from the laser scans. This way, in conjunction with the color labels, noise is also generated for the laser based detection of the goal boxes.

From the data we computed the average run time of the algorithm and added different levels of Gaussian noise to the odometry data for determining the accuracy and robustness of our approach. Figure 7 shows the vision and LRF data our robot collected during the experiment. Note, that

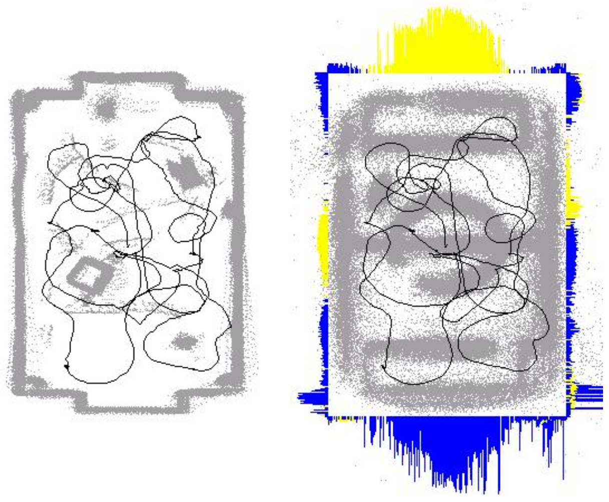


Figure 7: Collected laser (a) and vision data (b) plotted with respect to the reference position. Grey dots indicate detetected line points. The colored bars at the field boundaries indicate the direction and size of detected blobs of the goal colors. The robot's trajectory during the test run is plottet black.

from the LRF data only the goal lines are extracted and matched. During the run the robot moved a total distance of approximately 68 meters and turned about a total of 8000 degrees.

In the experiments we distinguish between three different modes of the *Extended LineMatch* algorithm. Firstly, we ran tests with only lines extracted from the laser scans as input for the algorithm. Secondly, only lines extracted from the camera image were taken as input. Finally we ran tests with both, laser range finder and vision data.

The Gaussian noise $\langle \Delta_\delta(\delta), \Delta_\alpha(\alpha), \Delta_\alpha(\delta) \rangle$ was added to the (already naturally noisy) odometry readings as $\delta \leftarrow \delta + \Delta_\delta(\delta)$ and $\alpha \leftarrow \alpha + \Delta_\alpha(\alpha) + \Delta_\alpha(\delta)$ when the robot moves a certain distance δ and turns a certain angle α [5].

Table 1 shows the average run times for the different modes of the algorithm. Obviously the most time is spent

	Only LRF	Only Vision	Combined
Preprocessing	$1.2ms$	$25.2ms$	$26.4ms$
LineMatch	$1.0ms$	$5.6ms$	$7.2ms$
Total	$2.2ms$	$30.8ms$	$33.6ms$

Table 1: Run time results on a 600 MHz Laptop.

for the extraction of the relevant features from the camera image. Also not surprisingly, the matching procedure needed the more time, the more lines it had as input. On the one hand, the time measured for the combined version reflects the fact that the algorithm doesn't scale linearily. But on the other hand, a total run time of less than 35ms clearly shows the algorithm's efficiency.

Figure 8 shows the distance error (a) and the angle error (b) to the reference position for the different modes of the algorithm under different levels of Gaussian noise. It can

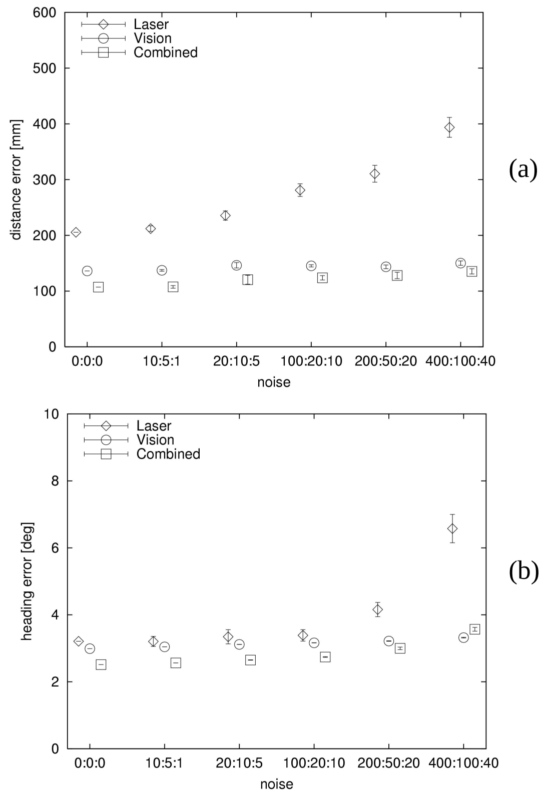


Figure 8: (a) Distance and (b) angle error to the reference position for different levels of Gaussian noise.

be seen, that only the LRF-mode suffered from the increasing noise levels whereas the combined mode and the vision mode showed to be very robust even against high extra odometry noise. Due to the higher accuracy of the LRF the combined mode performed slightly better than the vision mode. Quite noticeable is the high angular accuracy for all modes. Especially the combined mode and the vision mode showed only little distance deviations to the reference position. Please note, that the combined algorithm requires the synchronization between LRF and vision system. Since in our current implementation this is done by projecting the measurements to the same point in time based on the odometry readings, the combined algorithm suffers additionally from odometry noise. Nevertheless, a precision of up to 10cm and $2.5°$ demonstrates the algorithm's accuracy.

In order to verify the robustness of our approach we counted the number of times where the robot was considered to be lost because its pose deviated from the reference pose by more than 0.5m or more than $30°$. Figure 9 shows the times for the different modes of the algorithm under different levels of Gaussian noise. With little noise

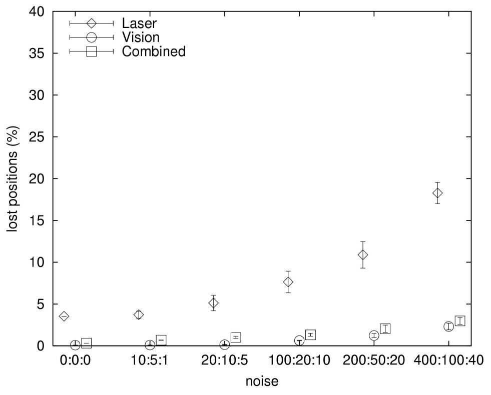


Figure 9: Number of times (in %) where the position error was above 0.5m or above $30°$.

in odometry all three modes showed a very robust performance. Using the combined mode and the vison mode, the robot was lost in less than 4% of the cases even under high noise levels. However, too little features were available for the LRF-mode to compensate for highly noisy odometry readings and the robot got lost too frequently.

In the above experiments we were able to perfectly adjust the YUV-bounds for the color classification of the camera image. However, quite often this is not feasible because the environment isn't iluminated uniformly or the lighting conditions change over time. Therefore, we conducted a second experiment in order to evaluate the algorithm's performance when color bounds are only adjusted poorly.

In the same environment as before we let the robot move around the field in a similar way as before. This time, however, the color classificator for the white field lines was adjusted more conservatively such that fewer lines could be detected in the camera images. We also made the LRF-based recogniton of the goal lines more difficult by moving the bottom obstacle in Figure 7(a) closer in front of the blue goal.

Figure 10 shows the distance error (a) and the angle error (b) to the reference position for the different modes of the algorithm in the second experiment. As can be seen, especially the distance accuracy of the purely vision-based and purely LRF-based method suffered from the harder experimental conditions. At the same time, the combined processing of LRF lines and vision lines lead to a significantly higher accuracy than processing either only LRF lines or only vision lines. Figure 11 shows the corresponding num-

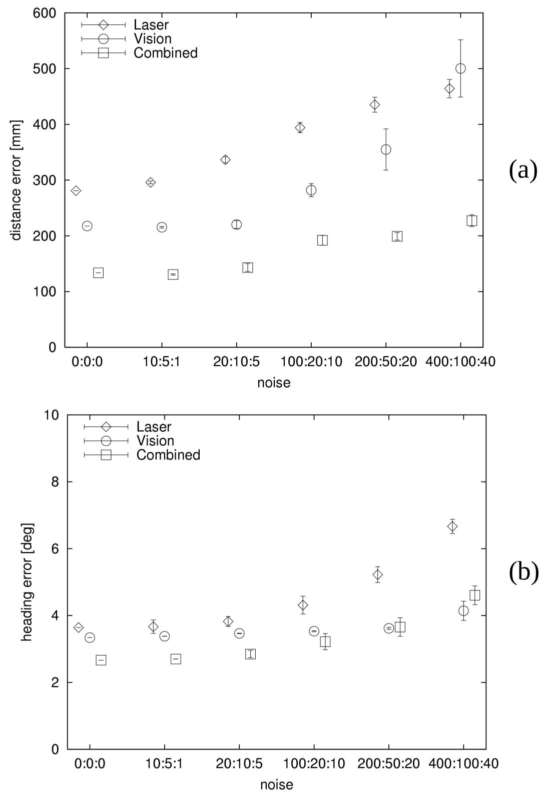


Figure 10: (a) Distance and (b) angle error to the reference position for different levels of Gaussian noise.

ber of times when the robot position was considered to be lost. Obviously, the robustness of the algorithm is clearly increased if the combined mode is used.

In a final test we evaluated our approach's ability to solve the *wake-up*-problem, i.e. to localize globally. We calibrated the vision system for detecting the field lines as good as possible and positioned the robot at 15 different locations on the empty field (see Figure 12). After startup we checked by visual inspection if the robot was localized correctly. The test was succesful for all positions and thus demonstrates the ability of the proposed algorithm to unambiguously localize the robot globally in the RoboCup environment.

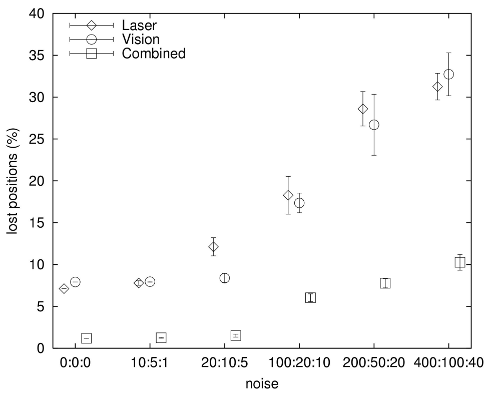


Figure 11: Number of times (in %) where the position error was above 0.5m or above 30°.

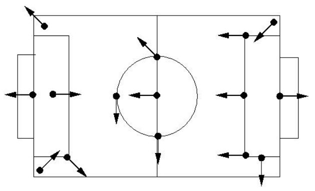


Figure 12: Robot positions for the *wake-up* test.

6 Conclusion and Outlook

We proposed a method for the combined correlation of line features detected by an omni-directional camera and a laser range finder with an a priori model of the environment. The resulting localization method has been evaluated in a typical RoboCup scenario which represents a highly dynamic and noisy real world environment. It has been shown that the algorithm is efficient, accurate and robust at the same time. In the RoboCup context global localization is possible instantly and without ambiguities.

In the RoboCup domain only few line features are detectable by a LRF. Under ideal lighting conditions enough field lines can usually be detected in the camera images. Under such conditions, combining the vision data with LRF data doesn't improve the localization performance considerably. However, if the vision process is disturbed and less field lines are visible, the combined algorithm performs clearly better than the purely vision-based or purely LRF-based mode. Obviously, the combined algorithm compensates for the lack of data if only the data of one sensor would be used.

The algorithm's combined version adds considerably to the generality of the approach. Since vision and laser lines can be utilized interchangeably it should be possible to easily apply the method in other environments such as office environments, ware houses or even outdoor scenarios.

Future work will include extensions and tests for these environments, as well as a further parameter refinement for achieving a higher accuracy.

Acknowledgment

The authors would like to thank Jens-Steffen Gutmann for his valuable comments for improving the presentation of this paper.

References

[1] K. Arras, N. Tomaris, B. Jensen, and R. Siegwart. Multisensor on-the-fly localization: Precision and reliability for applications. *Robotics and Autonomous Systems*, 34(2-3):131–143, 2001.

[2] J. Bruce, T. Balch, and M. Veloso. Fast and inexpensive color image segmentation for interactive robots. In *Proc. Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2061 – 2066, Takamatsu, Japan, 2000.

[3] J. A. Castellanos and J. D. Tardós. Laser-based segmentation and localization for a mobile robot. In *Second World Automation Congress (WAC'96)*, pages 101–108, Montpellier, France, May 1996.

[4] H.-M. Gross, A. Koenig, H.-J. Boehme, and C. Schroeter. Vision-based monte carlo self-localization for a mobile service robot acting as shopping assistant in a home store. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2002.

[5] J. Gutmann, T. Weigel, and B. Nebel. A fast, accurate, and robust method for self-localization in polygonal environments using laser-range-finders. *Advanced Robotics Journal*, 14(8):651–668, 2001.

[6] J.-S. Gutmann. *Robuste Navigation autonomer mobiler Systeme (in German)*. PhD thesis, Albert-Ludwigs-Universität Freiburg, Institut für Informatik, 2000.

[7] R. Hanek, T. Schmitt, M. Klupsch, and S. Buck. From multiple images to a consistent view. In Stone et al. [14].

[8] L. Iocchi and D. Nardi. Hough localization for mobile robots in polygonal environments. *Robotics and Autonomous Systems*, 40:43–58, 2002.

[9] P. Lima, A. Bonarini, C. Machado, F. Marchese, F. Ribeiro, and D. Sorrenti. Omni-directional catadioptric vision for soccer robots. *Robotics and Autonomous Systems*, 36(2–3):87–102, 2001.

[10] F. Marchese and D. Sorrenti. Omni-directional vision with a multi-part mirror. In Stone et al. [14], pages 179–188.

[11] L. Paletta, S. Frintrop, and J. Hertzberg. Robust localization using context in omnidirectional imaging. In *IEEE Int. Conf. on Robotics and Automation (ICRA 2001)*, pages 2072–2077, 2001.

[12] M. Restelli, D. G. Sorrenti, and F. M. Marchese. A robot localization method based on evidence accumulation and multi-resolution. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2002.

[13] E. Schulenburg. Selbstlokalisierung Im Roboter–Fußball unter Verwendung einer omnidirektionalen Kamera (in German). Diplomarbeit, Albert-Ludwigs-Universität Freiburg, Institut für Informatik, 2003.

[14] P. Stone, G. Kraetzschmar, T. Balch, and H. Kitano, editors. *RoboCup-2000: Robot Soccer World Cup IV*. Lecture Notes in Artificial Intelligence. Springer-Verlag, Berlin, Heidelberg, New York, 2001.

[15] T. Weigel, J.-S. Gutmann, A. Kleiner, M. Dietl, and B. Nebel. CS-Freiburg: Coordinating Robots for Successful Soccer Playing. *IEEE Transactions on Robotics and Automation*, 18(5):685–699, October 2002.

[16] N. Winters and J. Santos-Victor. Mobile robot navigation using omni-directional vision. In *Irish Machine Vision and Image Processing Conference (IMVIP'99)*, 1999.