Abstract. This paper describes the localization method used in the 4 Stooges robotic soccer team [2]. A soccer field is a challenging environment for localization, since it provides few unique landmarks.
The methodology described in the paper uses vision feedback from straight lines to determine the motion parameters. These parameters are used both to update the current location estimate as well as an actuator model of the robot.

Efficient Localization for Mobile Robots

Jacky Baltes
Centre for Image Technology and Robotics
University of Auckland,Auckland
New Zealand
j.baltes@auckland.ac.nz
http://www.citr.auckland.ac.nz/~jacky

No Institute Given

1 Introduction

This paper describes the localization method used in the 4 Stooges, a robotic soccer team from the University of Auckland. The hardware used by the 4 Stooges is based on simple toys and thus lacks the high accuracy DA converters, high accuracy AD converters, and shaft encoders found in other robotic platforms.

Figure 1 shows pictures of our mobile robots. The 4 Stooges use a differential drive mechanism. They are fully autonomous and include a CMOS camera sensor on the robot. Processing power is provided by an Eyebot controller ([3]), a small embedded controller using a Motorola MC68332 processor.

One of the most difficult problems for mobile robots is localization, that is to determine the position of the robot in the world. Clearly, to be useful a mobile robot must be at the right place at the right time. Therefore, recognizing the current location is a fundamental ability of a mobile robot.

Robotic soccer is a very difficult environment for localization since it does contain few unique landmarks. A unique landmark is one which allows a robot to uniquely determine its location. For example, the only landmarks visible from the images shown in Fig. 2 are the two colored goals. In most images, only a wall or corner can be identified, but this does not allow a unique determination of the location of the robot.

In the robotic soccer domain, the location of the robot is determined by its position on the playing field (i.e., its x,y coordinate) as well as its orientation.

The paper is organized as follows. Section 2 introduces two representative approaches to ego motion estimation as found in the literature. However, these approaches are based on the optical flow and are therefore computationally too expensive to be implemented in real-time on an embedded system.

Section 3 shows a block diagram of our architecture and shows the importance of ego motion estimation in our design.

Section 4 describes our approach to motion parameter estimation based on views of a line segment. The motion parameters are limited to only two parameters, the right and left wheel velocities for a differential drive robot.

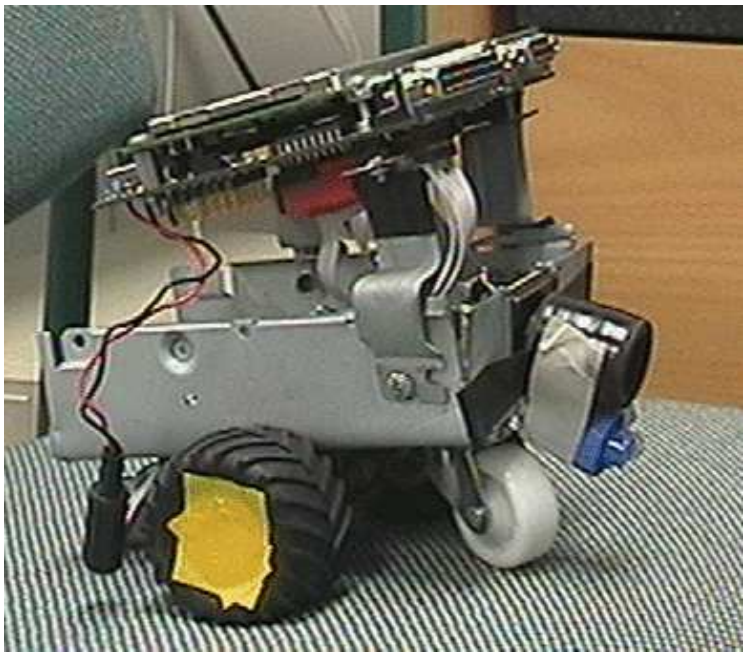

Fig. 1. The 4 Stooges Robotic Platform

Fig. 2. Some Sample Views from the Robot in the Soccer Domain

Section 5 describes some issues in the implementation of the method described in section 4. In particular it describes a fast method for camera calibration and detection of lines in the image.

The paper concludes by summarizing the results and providing directions for future research in section 6.

2 Related Work

Ego-motion estimation, i.e., the problem of determining the motion of a robot from a given a sequence of images taken during the motion, is a very popular research area in computer vision.

Since vision systems have become cheaper and smaller over time, they have been used in more applications. Computer vision also provides a very powerful general purpose sensor that can be used in a great variety of situations.

However, although it is easy for a human to detect the motion of a robot from a sequence of images, it is hard to do on a computer.

The ego-motion is formally defined as determining the translation and rotation parameter of a camera view given two different views of a scene. This estimate is usually based on unconstrained motion of a calibrated camera with respect to a plane. In this case, there are eight parameters to be determined.

Most approaches use the *rigid world* assumption where the scene is considered static and all movement in the image is due to the motion of the robot. In the robotic soccer domain, this assumption does not hold true very often. For a local vision robot, large movements in the image may be due to other robots moving in the image as well.

Stein et al. propose a method for robust ego-motion detection of a moving vehicle using a direct method. All pixels in the region provide support for a particular motion. A global probability distribution function is computed for the whole image. The method is robust in the presence of many outliers [6]. Stein reduces the number of parameters to three parameters (translation, pitch, and yaw).

Zhang describes another approach for estimating motion and structure from line segments [9]. This approach is based on the assumption that line segments overlap in 3D space, which allows the reduction in motion parameter.

Most of these approaches, however are computationally too expensive to be performed on an embedded controller in real-time. As described in section 4, we use a two parameter model appropriate for our mobile robots.

Also, given a fast camera calibration routine and a fast wall detection method, we are able to estimate the motion on the embedded system.

3 System Design

This section describes the model used in our work as shown in Fig. 3. The robot interacts with the environment using actuators and receives feedback from the environment via a set of sensors.

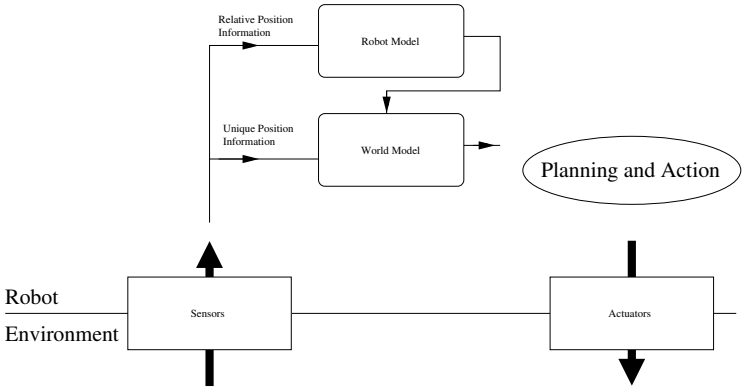

Fig. 3. System Design of our Localization Method

The sensor information can be classified into two types.

1. sensor information which allows the unique identification of the robot state is called *absolute position information*. Examples of absolute information are the robot two unique landmarks or one landmarks which are unique within the current uncertainty region.
2. sensor information which only provides feedback about the motion of the robot from the previous step to the current step is called *incremental position information*. Examples of relative position information are changes in the views of a wall.

As in most other related work, the sensors are used to update the world model of the robot, in our case the orientation and position of the robot on the playing field as well as the position of any other robots.

However, as described in the previous position, not all sensor feedback is useful to update the world model. Most other systems use dead reckoning to supplement the localization in this case.

As mentioned previously, our robots are not equipped with dead reckoning sensors (e.g., shaft encoders). Therefore, the system maintains an internal model of the robot for dead reckoning style navigation.

The dead reckoning navigation and internal robot model uses incremental sensor feedback.

The sensor feedback for incremental location information is also used to update a robot model, which models the behavior of the robot to the motor commands. This relationship changes drastically over time, since the batteries use their charge after only a few minutes.

Furthermore, the response to a motor command varies greatly depending on a number of factors, including battery charge, tire pressure, and so on. The robot model provides an actual model of the reaction of the robot to different motion

commands and is adapted when new incremental location information becomes available.

4 Incremental Location Information from Straight Lines

One of the most striking features in the pictures of the robotic soccer domain shown above are the walls. In most pictures, walls are a predominant and easily distinguishable feature.

Therefore, we decided to develop a method which uses walls in the image to provide at least incremental location information.

The following analysis depends on the assumption that the velocities of the right and the left wheel were constant during the time period for which the velocities are to be estimated. This is necessary since the proposed approach uses the differences between the start and end locations of the robot motions to derive values for the wheel velocities.

This is impossible if the wheel velocities are not constant during the time period in question. In this case, the robot can perform any one of an infinite number of possible movements and returned to the end location.

The kinematic model of a differential drive robot is shown in Fig. 4.

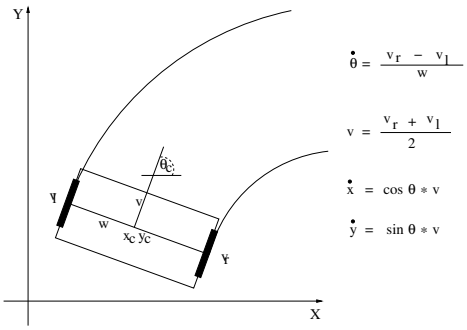

Fig. 4. Kinematic Model of a Differential Drive Robot

We use the example in Fig. 5 to motivate our analysis. At time $t0$, the robot is at position $P_{t0} = (x_{t0}, y_{t0}, \theta_{t0})$. A straight line (or wall) is in front of the robot. Let W_{ti} be the point on the wall in the center of the camera view of the robot at time ti and let d_{ti} be the distance between P_{ti} and W_{ti}. Let θ'_{ti} be the angle between the orientation of the robot θ_{ti} and the angle of the wall θ_w.

At time $t1$, the robot is at position $P_{t1} = (x_{t1}, y_{t1}, \theta_{t1})$. The distance d of the robot to the wall as well as the angle θ' between the robot and the wall will have changed.

The problem is to derive from this information the current wheel velocities v_r and v_l for the right and left wheel respectively.

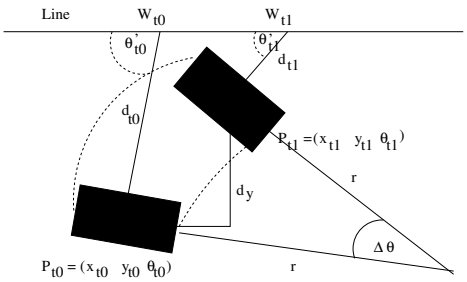

Fig. 5. Kinematic Model of a Differential Drive Robot

Given the kinematic model of the differential drive robot, it is easy to see that

$$\dot{\theta}' = \dot{\theta} = \frac{v_r - v_l}{w}$$

Since the width w of the robot is known, this allows us to compute the difference in velocities for the two wheels.

However, there is not sufficient information in the turn rate alone to determine the average velocity.

We assign a coordinate system X' with the origin at P and the x-axis parallel to the line or wall. In this coordinate system, the robot is at the origin and its orientation is θ'.

We can derive the distance from the robot to the closest point on the wall $d_W = \sin\theta' * d$. This allows us to compute

$$\Delta y = -(d_W(t1) - d_W(t0))$$

As can seen in Fig. 6, the linear velocity of the robot can be computed from its rate of turn and the offset in the direction of the robot $\Delta y'$.

From a simple geometric relationship, we can compute the radius of the circle r

$$r = \frac{\Delta y'}{\sin(\theta'_2 - \theta'_1}$$

The linear velocity is then equal to the length of the circle segment $(r * \Delta\theta)$ divided by the time to cover this distance.

$$v = \frac{r * (\theta'_2 - \theta'_1)}{\Delta t}$$

In case the robot is facing the wall directly (i.e., $\theta' = 90°$), the above equation yields the correct solution for the linear velocity of the robot.

As can be seen in Fig. 7, similar reasoning allows us to compute the circle distance $\Delta y'$ from a given wall distance.

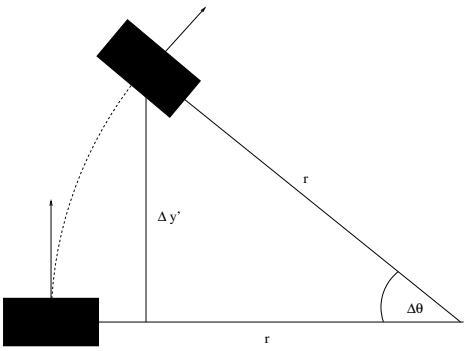

Fig. 6. Determination of the linear velocity v of a differential drive robot facing a wall

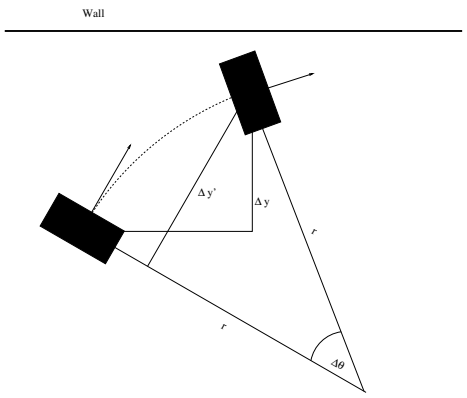

Fig. 7. Determination of the linear velocity v of a differential drive robot with an arbitrary angle to the wall

From the figure, one can derive that

$$\frac{c}{dy} = \sin\theta_1' - \frac{\theta_2' - \theta_1}{2}$$

In this case

$$\Delta y = \Delta y' * \frac{\sin\frac{\theta_2-\theta_1}{2}}{\cos\frac{\theta_2+\theta_1}{2}}$$

The right and left wheel velocities can then be calculated as

$$v_r = \frac{2v + \theta' * w}{2} v_l = \frac{2v - \theta' * w}{2}$$

Note that any straight line segment in the image can be used as a guideline. We find that many environments, including robotic soccer, provide many different line segments that can be used to compute the incremental location of the robot.

5 Practical Implementation

Even though the method described above is simple and straightforward, there are practical issues which makes it difficult to implement this idea on an embedded controller.

This section discusses some practical aspects of our implementation.

5.1 Feature Extraction

The first problem is that our method relies on the recognition of a straight line in the image.

There are numerous methods described in the literature for determine the orientation of a line. The most popular method is to perform edge detection followed by a Hough transform of the resulting image [5, 1]. The hough transform is a powerful method for detecting line segments in noisy images. However, it is computationally expensive.

In the images from the robotic soccer domain, the wall or playing field lines are the most striking features. Therefore, we implemented a very fast and cheap method for finding these lines.

The other advantage is that the lines on the playing field do not move. Although our method would work equally well with lines on other robots, these lines can introduce errors if the other robots are moving.

We use up to seven *drill points* across the image. We search seven columns in the image (20, 40, 60, 80,100, 120, 140) from the bottom to the top and look for the transition from the playing field to the wall.

The playing field is determined by a minimum number of playing field colored pixels (in this example, green pixels) in a certain region. Similarly, the wall is

defined as a minimum number of white pixels in a region. This makes the system more robust to noise.

The slope of the wall segment is then approximated by computing the difference in y coordinates for adjacent columns. If more than three columns match a specific line implementation then the algorithm returns that a wall was found and the angle between the center line of the robot and the wall.

Figure 8 shows an example of the wall following algorithm.

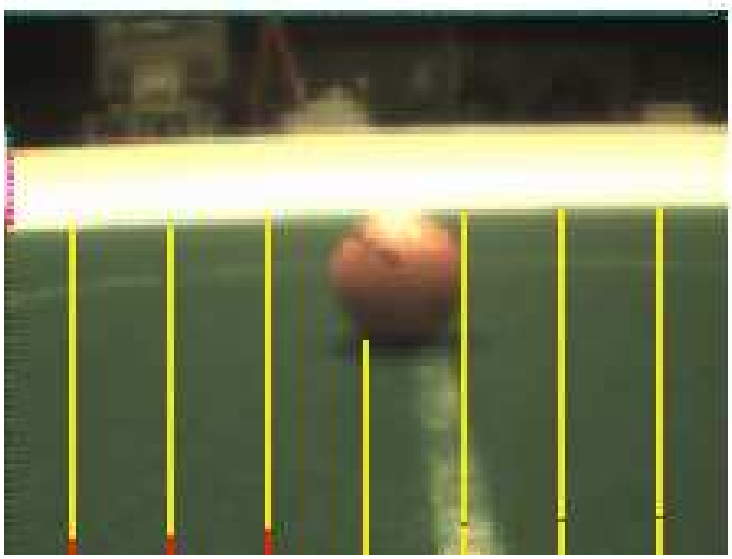

Fig. 8. Image of the performance of the wall detection algorithm

Note that the wall finding algorithm is conservative. It only returns the detection of a wall or line, if it there is significant evidence that the wall exists. In other words, the conditions for detecting a wall are *sufficient* in our algorithm.

The wall detection code does not provide *necessary* conditions for the existence of a wall. There are cases in which our algorithm fails to find a wall. In this case, the motor model will not be updated, but otherwise the performance of the robot is unaffected.

For example, in the two scenes shown in Fig. 9, the wall detection algorithm fails to find sufficient evidence for a wall.

5.2 Camera Calibration

Since there are only 120 rows and only seven columns of interest in our image, we use a simple lookup table to calibrate the distance from objects on the ground

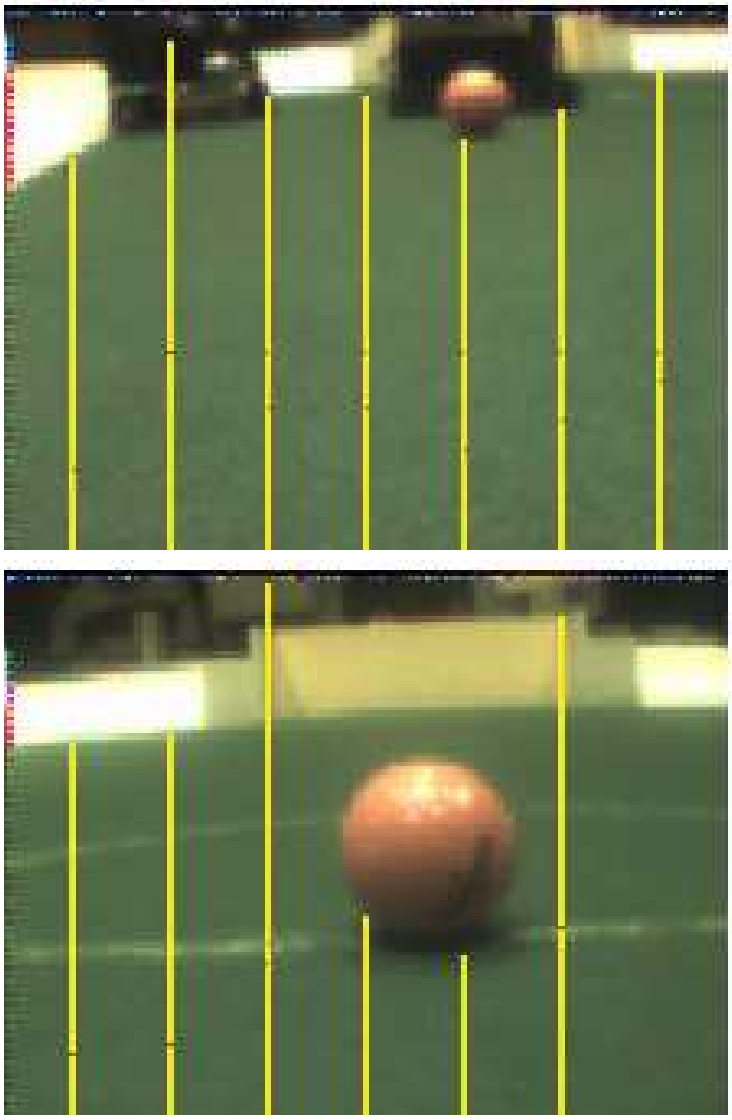

Fig. 9. In these images, the wall detection algorithm fails to detect a wall.

plane to the robot. More elaborate methods are possible [7, 8], but the accuracy achieved by this crude calibration is sufficient for our purposes.

5.3 Extensions to 3D lines

So far, we have assumed that the lines are in the ground plane. The algorithm can simply be extended by using the camera calibration method developed in subsection 5.2, assuming that the height of the feature points are constant.

Therefore, it is also possible to use this method to determine the ego motion of the robot with respect to a line painted at a constant height on the walls of the playing field.

5.4 Extension to Internal Parameters

As mention in section 4, the turn rate alone of a differential drive robot does not contain enough information to find the right and left wheel velocities.

In our current system, we use the motion of the robot towards the wall as another input, which then allows us to compute all parameters of the kinematic model.

However, it is interesting to look for other possibilities to augment the data about the turn rate of the robot. In particular, internal parameters of the robot would be desirable, since they are usually easier to measure.

One approach that we intend to implement in the future is to calibrate the torque on the motor based on the motor setting and the current charge status of the battery.

Given different motor settings and their resulting difference in velocities, we hope that it is possible to determine the wheel velocities without additional information from the outside world.

This would then allow us to extend our approach to single point correspondences.

6 Conclusion

This paper describes the methodology used in the 4 Stooges for localization. Walls or straight line segments are used to provide incremental motion information.

In contrast to other methods, the described method uses easily determinable features of prominent landmarks for its calibration.

At the same time, the incremental location information is used to update an internal robot model. This greatly improves the dead reckoning capabilities of the robot. Since the information is based on observed motion of the robot it is independent of errors in the odometry and work even on rough terrain, where traditional odometry fails due to excessive wheel slip.

Currently, the effectiveness of the system is evaluated using an empirical evaluation. Preliminary results are promising. The system will also be used in the 2002 FIRA competition ([4]).

We are also investigating methods of providing ego motion estimation based on simply the turn rate of the robot.

References

1. D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.
2. Jacky Baltes. The 4 stooges homepage. WWW, November 1999. http://www.citr.auckland.ac.nz/~jacky.
3. Thomas Bräunl. Thomas bräunl's homepage. WWW, November 2002. http://robotics.ele.uwa.edu.
4. Federation International de Robotic Soccer. Fira homepage. WWW http://www.fira.net, May 2002.
5. K. Y. Hough. Mehod and means for recognizing complex patterns. U.S. Patent 3069654, 1962.
6. Gideon P. Stein, Ofer Mano, and Amonon Shashua. A robust vehicle ego-motion. In *Proceedings of IV-2000*, 2000.
7. Roger Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, FL, 1986.
8. Z. Zhang, O. Faugeras, and R. Deriche. An effective technique for calibrating a binocular stereo through projective reconstruction using both a calibration object and the environment. *Videre: A Journal of Computer Vision Research*, 1(1):58–68, 1997.
9. Zhengyou Zhang. Estimating motion and structure from correspondences between line segments between two perspective images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1129–1139, 1995.