THE UNIVERSITY OF NEW SOUTH WALES
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

New Methods for Improving Perception in RoboCup SPL

Peter Anderson (3293464)

Submitted as a requirement for the degree
Bachelor of Computer Engineering

August 2012

Supervisor: Dr Bernhard Hengst

Abstract

This thesis introduces a number of new techniques motivated by the desire to improve robots' perception of their environment in RoboCup SPL. These methods include a unified field-feature inverse sensor model, a natural landmark localisation system, a visual odometry module, and a robot detection system that combines vision with sonar sensors. All of these techniques were demonstrated in the 2012 RoboCup competition.

Using a variation of ICP, the field-feature sensor model combines multiple simultaneous observations of aliased field-features into a single robot pose observation by: (1) matching observed features to the field map in a hierarchical fashion, and (2) minimising the squared positioning error simultaneously over all observed features. Results indicate that this sensor model is able to localise the robot from a single observation in 42% of field positions where multiple field-features are visible. However, it is not possible to distinguish between one end of the field and the other using field-features; this capability is provided by a natural landmark localisation system. Using a 'bag of words' image representation, the natural landmark localisation system stores up to 40 images of each goal area, and then matches camera frames to these stored images in real time on the Nao to resolve the field-end ambiguity. In a static environment this system is shown to perform flawlessly in repeated kidnap tests.

To further improve robot localisation, a fast and unique method for calculating visual heading odometry is also presented. Experiments indicate that this system can reduce the odometric uncertainty of an uncalibrated Nao robot by 73%. The visual odometry module is also able to detect collisions with unseen objects, while remaining robust to the presence of moving objects in the environment. Both the visual odometry module and the natural landmark localisation system are based on modified 1D SURF image features extracted from pixels on the robot's horizon. Consistent with the original SURF algorithm, the extracted features are robust to lighting changes, scale changes, and small changes in viewing angle or to the scene itself, while achieving a speed up of several orders of magnitude over SURF. This makes 1D SURF features suitable for visual navigation of resource constrained mobile robots.

Finally, a combined vision and sonar robot detection system is presented that uses a novel sonar hardware control scheme to introduce a third sonar detection sector in front of the robot. Evidence suggests that during a penalty shoot-out, this system enables the striker to localise a stationary goalie to within 28 mm to 60 mm before shooting. This capability also enabled rUNSWift to develop coordinated role-switching behaviours that remain operational during total wireless failures.

Acknowledgements

First and foremost, I would like to acknowledge the ongoing support and enthusiasm of my supervisor, Dr Bernhard Hengst. He helped me to understand the RoboCup competition environment, the existing rUNSWift system, and to focus on the right issues. Bernhard's experience was invaluable.

Secondly, I would like to thank the other members of the 2012 rUNSWift team: Sean, Youssef, Bel, Roger, Ritwik, Carl, Sam and Richard. It was a privilege to work with such a great team. In particular, I would like to acknowledge Youssef Hunter who worked with me implementing the unified field-feature inverse sensor model, and deserves credit for that work. I would also like to thank my girlfriend, Christina, for listening patiently to many soliloquies about robots and for occasionally reminding me that I am not really a World Cup soccer player.

The rUNSWift code base has evolved over many years, with many people contributing to the development of the underlying infrastructure that we use. For this I would also like to acknowledge all past members of the rUNSWift team, and associated staff and students in the school's robotic laboratory. Finally, I am grateful for the generous financial support of the School of Computer Science and Engineering, which allowed us to compete in Mexico. It was a lot of fun.

Related Publications

Peter Anderson, Yongki Yusmanthia, Bernhard Hengst, and Arcot Sowmya. Robot Localisation Using Natural Landmarks. In *Proceedings of the RoboCup International Symposium 2012*, Mexico City, Mexico, 18-24 June, 2012. (Nominated for best paper).

Table of Contents

1 Introduction

The RoboCup Standard Platform League (SPL) is an international robot soccer competition, in which all teams compete with identical autonomous robots. The stated intention of the RoboCup initiative is to foster research into robots and artificial intelligence by organising competitions under challenging real world conditions. The UNSW RoboCup SPL team, rUNSWift, has been involved with the competition since 1999 and placed third in the 2012 RoboCup SPL competition in Mexico.

Since the introduction of the humanoid Aldebaran Nao robot as the league's standard platform in 2008, the quality of SPL game play has improved significantly each year. However, observations of matches indicate that despite the many improvements, perception of the field environment and other robots remained one of the biggest hurdles facing most teams in 2011, including rUNSWift.

Supporting this claim, in 2011 rUNSWift still used localisation behaviours. These are behaviours that are intended to help robots localise, but offer no strategic value, such as standing still and head-scanning. Unfortunately, reliance on these behaviours frequently interrupts game play at critical moments. Furthermore, in 2011 matches we can frequently observe behaviour indicative of a robot being 'lost' or unaware of the positioning of other robots.

This thesis was motivated by the desire to bridge the gap between the state of rUNSWift game play in 2011, and the ideal in which each robot is precisely localised and aware of other robots at all times, without interrupting play with localisation behaviours. After all, it is only after this objective is attained that higher level strategic behaviours, such as team formations, passing, and marking, will become most valuable.

In pursuit of this aim, this thesis introduces several new techniques that were developed in 2012 to improve the robots' perception of their environment. These enhancements include:

1. A unified field-feature inverse sensor model,
2. A natural landmark localisation system,
3. A visual odometry module, and
4. A robot detection system combining vision and sonar.

All of these techniques were demonstrated in the 2012 RoboCup competition. The remainder of this chapter is devoted to individually introducing and outlining the motivation for each of these initiatives. The common thread running through all of them is the desire to improve the quality of the raw information extracted from the robot's cameras and sonar sensors. The selection and development of the most appropriate localisation filter for the robot is left to others.

1.1 Unified Field-Feature Sensor Model

Incremental development of the rUNSWift localisation system up until 2012 resulted in a proliferation of sensor models for observed field features. Each of these sensor models used geometry and heuristics to generate a normal distribution representing a hypothesis for the robot's pose, given a field-feature observation and the robot's last filtered pose. Separate sensor models were used for observations of field edges, field-line corners, field-line T-junctions, a single goal post, two goal posts, a goal post and a T-junction, parallel field-lines, and the centre circle [12].

By 2011 some of the limitations of this approach were becoming apparent. Field-line information is aliased due to the existence of two axes of symmetry in the field markings. When the robot perceives a field-line corner, it may be one of eight on the field and this does not indicate unambiguously where the robot is. However, if a field-line corner and a goal post are both perceived in a single frame, then the list of possible robot locations can be reduced considerably. Unfortunately however, intelligently combining information from a large number of separate sensor models is a difficult task. There is no clear indication which feature update should be applied to the localisation filter first, nor is there any guarantee that the position hypotheses generated by the sensor models will be mutually consistent. The introduction of the improved Nao v4 in 2012 exacerbated this problem, as situations where multiple field-features were observed in one camera frame became more common. This occurred since the Nao v4 allowed simultaneous access to both cameras for the first time, and provided increased processing power that facilitated field-feature observations at greater range.

The unified field-feature inverse sensor model overcomes these difficulties, and includes observations of single field-lines (the most common field-feature) in a sensor model for the first time. The unified sensor model works by representing all field features as 2 dimensional points in the field plane, and uses an adaptation of the Iterative Closest Point (ICP) algorithm to find the robot pose observation that minimises the squared positioning error over all observed features simultaneously. Unlike the previous approach that performed independent feature updates, the unified approach uses all the observed features in the frame to generate a single hypothesis for the robot's pose. This ensures that the pose hypothesis is always logically consistent with the combination of features that can be perceived.

A further advantage of the ICP-based approach is that it allows field-features to be associated in a hierarchical fashion, such that only the most distinctive features (such as goal posts) are used in the first few iterations of the pose estimate, and less unique observations such as field-lines are not associated with field positions until subsequent iterations. This helps ensure that observed field-lines are associated correctly.

1.2 Natural Landmark Localisation System

The SPL field set-up has changed over the years to progressively remove navigation beacons and other colour coded visual cues. In keeping with this trend, in the 2012 SPL competition the goal-posts at either end of the field were made the same colour for the first time [25]. As a result, a robot forced to localise from an unknown starting position is not able to resolve one end of the field from the other. In RoboCup matches this requirement can arise after a complicated fall, for example when robots become entangled, slip, and are rotated unwittingly.

B-Human's 2011 Open Challenge demonstration addressed the field-end ambiguity challenge by using a team-wide ball model, enabling a kidnapped robot to recover by fusing their own ball observations with those of their team-mates [26]. The authors acknowledged, however, that this approach could fail in situations where a robot is alone, unaware that it has been kidnapped, or if the team-wide ball model is incorrect. An own goal is the potentially disastrous result of one of these localisation failures. To avoid these problems and to allow a single robot to localise (for example, in the event of wireless failure), a method for extracting unique natural landmarks from images of the around-field environment is required. In this context, a natural landmark is defined as a set of scale-invariant local features that can be used to find point correspondences, and ultimately a perspective transformation, between two images containing the same object.

SURF (Speeded Up Robust Features) [5], [4] and SIFT (Scale-Invariant Feature Transform) [22] are two existing methods for extracting invariant local features from images. However, these methods are relatively computationally expensive and difficult or impossible to implement in real time on a resource constrained robot. To overcome these resource limitations, an optimised feature detector consisting of a modified one dimensional variant of the SURF algorithm was developed (1D SURF). This method was then applied to a single row of grey-scale pixels captured at the robot's horizon. The horizon image was chosen for analysis because, for a robot moving on a planar surface, the identified features cannot rotate or move vertically, and must always remain in the same order. The use of a 1D horizon image and other optimisations dramatically reduces the computational expense of the algorithm, while exploiting the planar nature of the robot's movement and still providing acceptable repeatability of the features.

Consistent with the original SURF algorithm, 1D SURF features are relatively robust to lighting changes, scale changes, small scene changes and small changes in viewing angle. Using these features, a bag of words natural landmark localisation system was developed that enables each robot to memorise the landmarks behind each goal area while walking on to the field at the start of each half. These landmarks can then be recognised in real time during the match, to help ensure that robots do not confuse their opponents goal with their own.

1.3 Visual Odometry Module

Odometry refers to the ability of the robot to estimate its motion. It is used in the process update of the localisation filter to account for the robot's transition to a new position. In previous years rUNSWift odometry has been quite crude; effectively taking information from the walk-engine and assuming that the robot will move as directed. However, practical experience indicates that individual bipedal robots slip to varying degrees while walking, and often do not walk in straight lines. On the soccer field this is exacerbated as robots are often bumped by other robots and the goal posts, or impeded by the touching of arms or feet.

During these manoeuvres it is apparent that the greatest odometry inaccuracy is observed in the robot's heading, which can change very quickly, rather than in the forward or sideways component of the robot's movement. As the Nao is not fitted with a z-axis gyroscope, a visual method is required to correct inaccuracies in walk-engine heading odometry. With this motivation, a visual odometry module was developed using the same 1D SURF features already used by the natural landmark localisation system. By matching these features across subsequent frames, this module can accurately measure changes in the robot's heading over time. Using this method, the robot's walk-engine heading odometry can be corrected, resulting in a significant improvement in the accuracy of the odometry information provided to the localisation filter. Furthermore, by analysing heading discrepancies between walk-engine odometry and visual odometry, obstacle collisions on the left and right sides of the body can be immediately detected and the appropriate avoidance behaviour taken. For example, it was observed that allowing the robot's arms to become limp during a collision avoided a significant number of falls at competition, as the robot was more easily able to brush past other robots.

1.4 Robot Detection using Vision and Sonar

To engage in top-level strategic behaviours, SPL teams require a complete world model that includes the location of both opposition robots and team mates. Although previous years saw the early developments of rUNSWift robot detection [21], this work was not deemed to be competition-ready. In 2012, work on visual robot detection was completed, and integrated with a new sonar filter. The new sonar filter has the ability to process sonar returns at multiple ranges on one sensor, in comparison to the previous filtering scheme which discarded all observations beyond the first return.

As part of the sonar filter development, it was discovered that the resolution of sonar direction could be improved from two sectors to three, by selectively controlling the left and right hand sonar transmitters and receivers. For example, when transmitting selectively on the left hand side of the robot body and receiving on the right hand side, only objects that are

directly in front of the robot are detected. This observation lead to a hardware control scheme that could accurate resolve detected objects into a left hand, middle, or right hand sector relative to the robot. This improvement in sonar direction resolution was a key factor in the successful integration of sonar ranges with visual robot detection.

These developments enabled robots to be detected and tracked at ranges of over 2 m. As a result of this capability, rUNSWift was able to develop strategic team behaviours that remained operational during the total wireless failures that characterised RoboCup 2012. This was done by role-switching to supporter or defender behaviour when a robot observed one of it's team mates closer to the ball than itself.

The remainder of this thesis is organised as follows: Chapter 2 describes the unified field-feature inverse sensor model in detail, and presents results illustrating the on-field performance of the sensor model. Chapter 3 details the calculation and characteristics of 1D SURF features. Chapter 4 describes the natural landmark localisation system, and illustrates its performance using repeated kidnap tests. Chapter 5 outlines the use of 1D SURF features for visual odometry, and presents results from odometry error benchmarking. Chapter 6 describes the methodology used for the combined sonar and vision robot detection and tracking system, which is evaluated using a penalty shoot-out scenario.

2 Unified Field-Feature Sensor Model

This chapter describes the unified field-feature sensor model in detail, and presents results illustrating its on-field performance.

2.1 Background

Recursive localisation filters require well-specified sensor models to work effectively. A sensor model describes the process by which sensor measurements are generated by the physical world. Forward sensor models specify a probability distribution over possible measurements conditioned on the robot state and the map of the environment. Inverse sensor models specify a distribution over possible robot states conditioned on the measurement and the map. The rUNSWift field-feature sensor model is specified as an inverse model, which is a common design choice when the measurements are more complex than the robot state, for example when the measurement is a camera image (as it is in RoboCup SPL) [32].

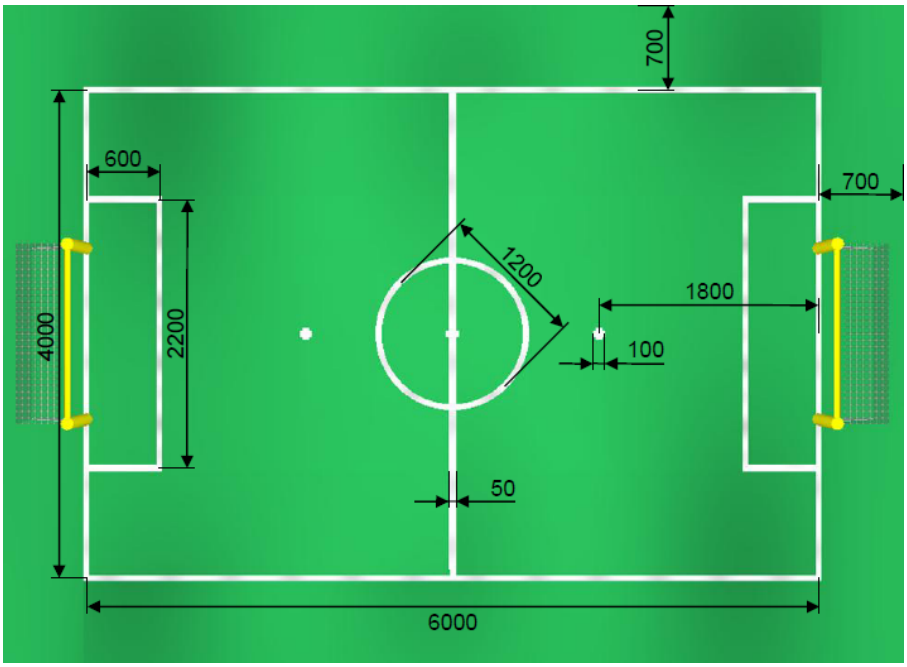

Fig. 1. Scale diagram of the RoboCup SPL field environment, illustrating axes of symmetry in the x and y dimensions. Measurements are in millimetres. Sourced from [25].

Consistent with the inverse approach, the sensor model used for localisation in RoboCup is required to generate a distribution over the robot's location, given field-feature observations and a map of the SPL field environment. Since the SPL field environment is heavily aliased, due to

the two axes of symmetry evident in Figure 1, the robot's latest filtered position is also used as an input to the sensor model. In the rUNSWift architecture, field-feature observations can be of various types, including observations of single field-lines, T-junctions, field-line corners, parallel field-lines, and the centre circle, as well as field-edge observations and goal post observations. These observations are generated from raw camera frames using image processing techniques documented in [24] and [18], and projected on to the ground plane in robot relative coordinates.

Although the specification of the SPL field environment is very particular, in many ways it is representative of a more general class of problems where a robot is required to localise in a 2D environment given observations of various types of features and a map. The application of recursive localisation filters to these problems is well documented [32]. However, in practise the general assumption of one sensor update per time step is often violated. In practical filter applications, there are often many time steps with no observations, and some time steps with multiple simultaneous observations. The appropriate course of action to take as a filter designer in these circumstances is one of the more neglected aspects of the literature.

One course of action in these circumstances is to perform several consecutive sensor updates. This was the approach used by rUNSWift in 2011, however it suffers from a number of disadvantages. There is typically no guidance as to which sensor update should be performed first. Furthermore, as shown by [31], if the errors in the observations are correlated (as would be expected if the observations are all extracted from the one camera frame), the use of consecutive updates results in a significant deterioration in localisation accuracy. Even worse than this, in applications such as RoboCup where observations are heavily aliased, there is no guarantee that the updates generated by independent sensor models will be mutually consistent.

The approach recommended by [31] in these circumstances is to perform a joint sensor update using all observations at once, ensuring that the observation covariance matrix accurately reflects the estimated level of error correlation between observations. This chapter outlines an alternative approach, in which an adaptation of the Iterative Closest Point (ICP) algorithm is used to estimate the robot pose that minimises the squared positioning error over all observed features simultaneously. This pose estimate can then be used as an observation to perform a single sensor update. The advantage of the ICP-based approach is that by associating detected features to the map in a hierarchical fashion starting with the most unique landmarks, we can help to ensure that the robot pose observation is consistent with all observed features. This is particularly important in applications where map features are heavily aliased, such as RoboCup.

By way of background, ICP is a widely used algorithm for geometric alignment of 2D or 3D point clouds when an initial estimate of the relative pose is known. Since the introduction of

ICP by [11], many variants of the algorithm have been proposed, but in general the basic steps of the algorithm as outlined by [27] include:

1. **Selection** of some set of points in one or both point clouds.
2. **Matching** these points to samples in the other point cloud, for example by using the nearest-neighbour match.
3. **Weighting** the corresponding pairs appropriately.
4. **Rejecting** certain pairs based on looking at each pair individually or considering the entire set of pairs (optional).
5. **Assigning** an error metric based on the point pairs, for example the sum of squared Euclidean distance between paired points.
6. **Minimising** the error metric by finding a transformation between the two point clouds.
7. **Iterating** the above steps to further improve the alignment.

By representing both observed field-features and the SPL field map as 2D points, the ICP algorithm lends itself to the construction of an inverse sensor model, as described below.

2.2 Method

Given a set of observed field-features in robot relative coordinates, a map of the SPL field, and a prior robot pose estimate, the unified field-feature sensor model provides a robot pose observation consistent with the observed field-features. In this application the robot pose is defined by its 2D planar coordinates x and y, along with its angular orientation θ. The first step in the calculation of the robot pose observation is to pre-process the field-features, with the aim of making each field-feature as distinctive as possible. This is done by increasing the number of field-feature categories through an examination of the spatial relationships between field-features observed in the same frame.

Prior to pre-processing, the rUNSWift vision system identifies field-lines (11), field-edges (4), field-line corners (8), field-line T junctions (6), goal posts (4), parallel lines (2) and the centre circle (1). The numbers in brackets indicate how many of each feature are found in the SPL field map. It is clear that before pre-processing, field-lines, field-line corners and field-line T junctions are highly non-unique. After pre-processing, three subcategories of field-line T junction are recognised, depending on the relative position of the observed goal post in the same frame (if any). Similarly, four sub-categories of field-line corner are recognised, depending on the position of observed field-edges relative to the corner. Field-lines are also restricted to a subset of all the possible field-lines on the SPL field map, again depending on the relative

position of observed field-edges. After pre-processing, the field-edge observations themselves are discarded, and the richer set of remaining field-features is retained for further processing.

After pre-processing, the key steps in the calculation of an updated pose observation include:

1. **Mapping** observed field-features from robot relative coordinates to field coordinates, using the prior robot pose estimate x_{k-1}, y_{k-1} and θ_{k-1}. In the first iteration this robot pose estimate is provided by the localisation filter, in subsequent iterations the computed pose is used.
2. **Hierarchically Matching** observed field-features with the nearest equivalent features on the SPL field map, starting with the most distinctive feature in early iterations, and gradually including less distinctive features in subsequent iterations.
3. **Representing** matched field-features using 2D point pairs.
4. **Weighting** the corresponding pairs appropriately, given the confidence level of the observation.
5. **Assigning** a squared Euclidean distance error metric based on the distance between point pairs.
6. **Minimising** the error metric by finding a new robot pose estimate x_k, y_k and θ_k.
7. **Iterating** the above steps to further improve the robot pose and to allow any incorrectly associated features to be matched correctly.

The similarities between the algorithm described above and the ICP algorithm are quite apparent. Each of these steps is described in further detail below.

During hierarchical matching (step 2), observed field-features in field coordinates are associated with the nearest map feature of the same type. For field-features with no orientation, such as single goal posts, the distance between features is calculated using Euclidean distance. For field-features that have orientation, such as a field-line T junctions, the distance also includes the orientation difference between features after the application of an arbitrary scaling factor to convert from radians to mm. In the case of field-lines, the distance between features is based on the Euclidean distance between the points at each end of the observed line, and the nearest point on the map line.

Matching is done hierarchically in step 2 to prevent observed field-features from being incorrectly associated with map features, in the event that the robot pose initialisation is poor. Although the iterative nature of ICP provides some robustness to the incorrect association of field-features (because features are re-associated at each iteration), the algorithm is still vulnerable to getting stuck in local minima. By associating only the most distinctive field-features in early iterations of the algorithm, we can ensure that the robot pose is approximately

correct before the most heavily aliased field-features (such as field-lines) are incorporated to further improve the robot pose estimate.

In the implementation of hierarchical matching for RoboCup SPL, the matching priority of features is defined (in order of distinctiveness) as: Two goal posts, parallel lines, field-line corners, field-line T junctions, a single goal post, a centre circle, and finally single field-lines. At each iteration of the algorithm a new field-feature is included in the match. In many cases, of course, no distinctive field-features are seen, and the sensor model will base its observation on a single field-line. However, the hierarchical approach ensures that when a distinctive field-feature eventually is seen, the sensor model is likely to output the correct robot pose, even if the robot pose initialisation is poor.

Once observed field-features have been matched to the field map, matched field-features must be represented as 2D point pairs for the ICP based approach to be used (step 3). These points exist in the global field coordinate frame. In each point pair, one point represents the observed position of some field-feature, transformed from robot relative coordinates to field coordinates using the current robot pose estimate. The second point represents the true position of that field-feature in the SPL field map. Features such as a goal post can be represented using a single point pair. Single field-lines are represented using two point pairs, with a point pair used to represent each end of the line segment. Finally, more complex field-features such as T junctions and corners are also represented with two point pairs, since this is sufficient to encode the location and orientation of the feature.

In step 4, the weighting on each point pair is scaled according to the inverse distance between the robot and that observed feature. The effect of this step is to put less emphasis on features that are observed at a greater distance, and are therefore less precisely located than closer features, when estimating the new robot pose. In steps 5 and 6, a new robot pose is estimated by rotating and translating the prior robot pose to approximately minimise the squared positioning error over all observed features simultaneously. Given a set of N source points $\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N\}$ and N target points $\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_N\}$ with weights $\{w_1, \ldots, w_N\}$, the mean squared positioning error e is given as a function of rotation matrix $R(\delta\theta)$ and translation vector $\boldsymbol{t}$ by:

$$e(R, \boldsymbol{t}) = \frac{1}{N} \sum_{i=1}^{N} w_i ||\boldsymbol{q}_i - R\boldsymbol{p}_i - \boldsymbol{t}||^2 \tag{1}$$

To find an approximate minimum for this error function, the centroids $\bar{\boldsymbol{p}}$ and $\bar{\boldsymbol{q}}$ are deducted from both source and target point clouds (in order to separate rotation from translation). The rotation matrix R is approximated using a first order Taylor series approximation such that

$sin(\delta\theta) \approx \delta\theta$ and $cos(\delta\theta) \approx 1$. Equating the first order partial derivatives of e with respect to t_x, t_y and $\delta\theta$ to zero gives the following matrix system for each point pair i:

$$\begin{pmatrix} 2w_i & 0 & -2w_i p_{i,y} \\ 0 & 2w_i & 2w_i p_{i,x} \\ -2w_i p_{i,y} & 2w_i p_{i,x} & 2w_i p_{i,y}^2 + 2w_i p_{i,x}^2 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \\ \delta\theta \end{pmatrix} = \begin{pmatrix} 2w_i q_{i,x} - 2w_i p_{i,x} \\ 2w_i q_{i,y} - 2w_i p_{i,y} \\ -2w_i p_{i,y}(q_{i,x} - p_{i,x}) + 2w_i p_{i,x}(q_{i,y} - p_{i,y}) \end{pmatrix} \tag{2}$$

Since each point pair i contributes three equations, if $N > 1$, this system is overdetermined, and the least squares solution can be found using a singular value decomposition. The new estimate for the robot pose observation can then by calculated from the previous pose estimate as follows:

$$\begin{pmatrix} x_k \\ y_k \end{pmatrix} = T \begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} = R(\delta\theta)[\begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} - \bar{\boldsymbol{p}}] + \begin{pmatrix} t_x \\ t_y \end{pmatrix} + \bar{\boldsymbol{q}} \tag{3}$$

$$\theta_k = \theta_{k-1} + atan2(y_k - \hat{p}_{i,y}, x_k - \hat{p}_{i,x}) - atan2(y_{k-1} - p_{i,y}, x_{k-1} - p_{i,x}), i \in \{1, \ldots, N\} \tag{4}$$

$$\hat{\boldsymbol{p}}_i = T(\boldsymbol{p}_i) \tag{5}$$

Step 7, iteration, is continued until all observed features have been included in the process, and the squared positioning error $e(R, \boldsymbol{t})$ falls below a threshold or does not reduce for two consecutive iterations, or if a maximum number of iterations is reached. At this point, if the mean squared error $e(R, \boldsymbol{t})$ is sufficiently small, the robot pose observation estimate x, y and θ can be provided to the localisation filter. If the position error is large, it may indicate that the observed combination of field-features is inconsistent, for example due to a false positive feature identification. It can also indicate that the algorithm became stuck in a local minima due to a poor initial pose, potentially indicating that the robot is lost.

The sensor model as described provides a method for integrating multiple observations of non-unique landmarks into a single observation. However, it does not deal well with observations of field-features that have length, such as field-lines, which typically leave one x or y coordinate of the robot's location unconstrained. Using the sensor model as described, suppose the robot observes both a field-line feature and a point feature such as a single goal post. During the iterative process, the use of a point representation for the field-line will artificially constrain the robot's pose from sliding along the field-line to improve alignment with the goal post. However,

since the field-line features in the SPL field are always aligned to the x or y axis of the field coordinate system, this problem can be improved by only minimising the error function $e(R, \boldsymbol{t})$ with respect to two out of three components of the change in the robot's pose; either θ and t_y (if the field-line parallels the y-axis), or θ and t_x (if the field-line parallels the x-axis). As such, the matrix system for a point pair i that belongs to a field-line parallel to the y-axis is:

$$\begin{pmatrix} 2w_i & 0 & -2w_i p_{i,y} \\ 0 & 0 & 0 \\ -2w_i p_{i,y} & 0 & 2w_i p_{i,y}^2 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \\ \delta\theta \end{pmatrix} = \begin{pmatrix} 2w_i q_{i,x} - 2w_i p_{i,x} \\ 0 \\ -2w_i p_{i,y}(q_{i,x} - p_{i,x}) \end{pmatrix} \tag{6}$$

The matrix system for a point pair i that belongs to a field-line parallel to the x-axis is:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 2w_i & 2w_i p_{i,x} \\ 0 & 2w_i p_{i,x} & 2w_i p_{i,x}^2 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \\ \delta\theta \end{pmatrix} = \begin{pmatrix} 0 \\ 2w_i q_{i,y} - 2w_i p_{i,y} \\ 2w_i p_{i,x}(q_{i,y} - p_{i,y}) \end{pmatrix} \tag{7}$$

2.3 Results

Usage Examples Before presenting more formal results, this section attempts to illustrate some characteristics of the sensor model by providing several examples. In the first example illustrated in Figures 2 and 3, multiple field-features are observed simultaneously by a robot standing near the centre of the field. The centre circle and a goal post are seen in the top camera frame, whilst in the bottom camera the centre circle and a field line can be observed. The unified sensor model was designed for exactly this sort of scenario. In this case, the observation of the centre circle and its intersecting field line helps to ensure that the single goal post is correctly matched to the left hand side of the goal.

Using this example, as illustrated in Figure 2, even if the heading of the initial robot pose is extremely poor (plus or minus 90 degrees from the true heading), the sensor model still converges on the correct robot pose from approximately 50% of all possible field locations. This illustrates the robustness of the sensor model to poor robot pose initialisation, provided several field-features are seen. However, it is not possible to distinguish one end of the field from the other using only field-features. As such, when the robot pose initialisation is closer to the aliased field position than the robot's true field position, it is no surprise that the sensor model converges to the aliased position. In practise, an area of convergence close to 50% with 90 degrees heading error therefore indicates that the sensor model can localise the robot from a single observation. Interestingly, however, if the heading error in the initial pose is reduced to 45 degrees, the area of convergence to the correct robot pose increases to cover the entire field, as illustrated in Figure 3.

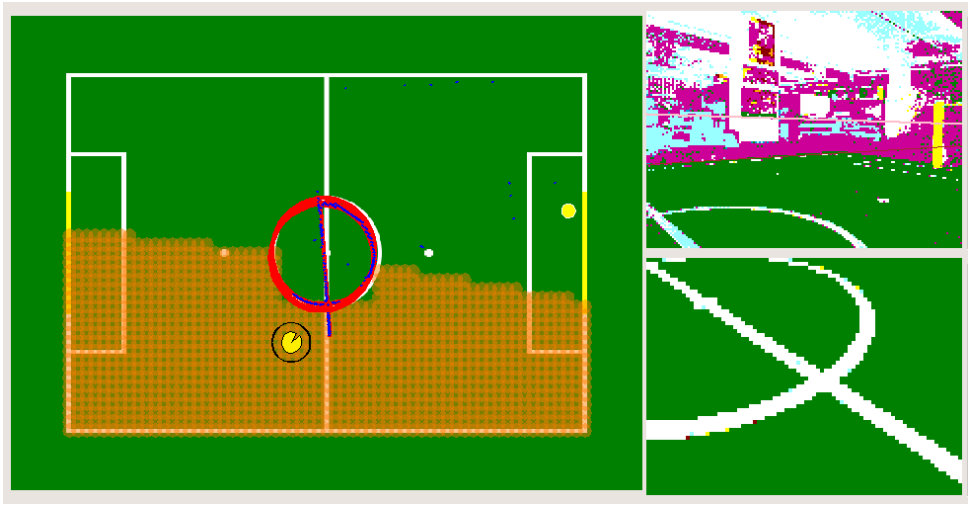

Fig. 2. Left: Sensor model area of convergence. Orange shaded areas indicate the set of initial robot poses that converge to the correct robot pose observation, shown in yellow, when initialised with a robot heading that is plus or minus 90 degrees from the true robot heading. Right: Colour classified source images from the robot's top and bottom cameras. Observations consist of a centre circle and a field line in the bottom camera, and a centre circle and goal post in the top camera.

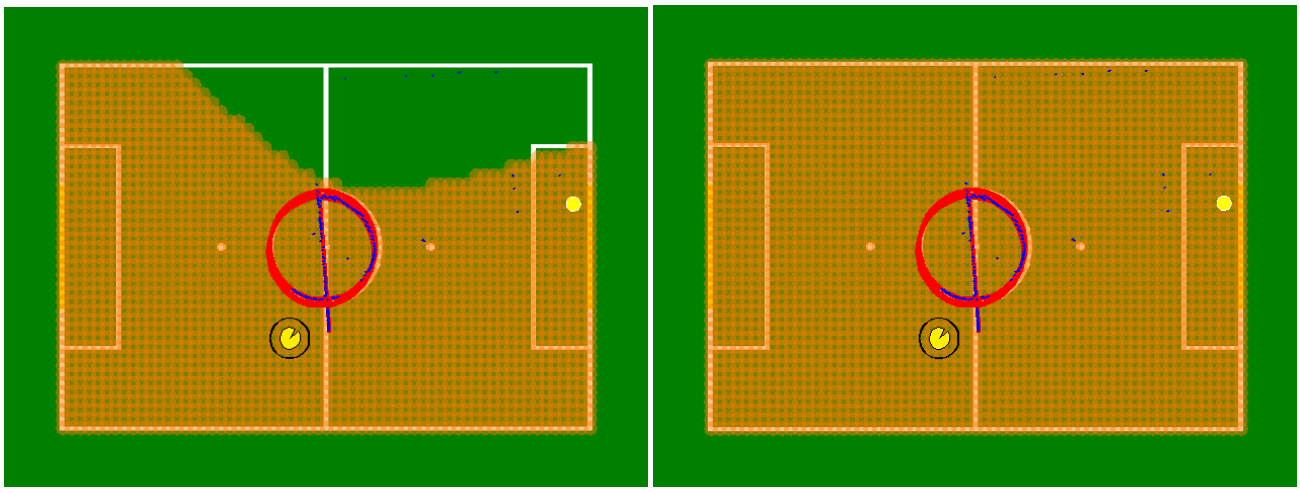

Fig. 3. Sensor model area of convergence after reducing the initial heading error to less than 60 degrees (left) and less than 45 degrees (right). These results indicate that when several field-features are observed, the sensor model is able to localise the robot from a single observation even if the robot pose initialisation is quite poor.

In the remainder of this section, four more examples are provided of field-feature observations with the corresponding area of convergence given 45 degrees initial heading error. In all of the results in this chapter, the sensor model observation is considered to be correct if it is within 10 cm and 10 degrees of the true robot pose.

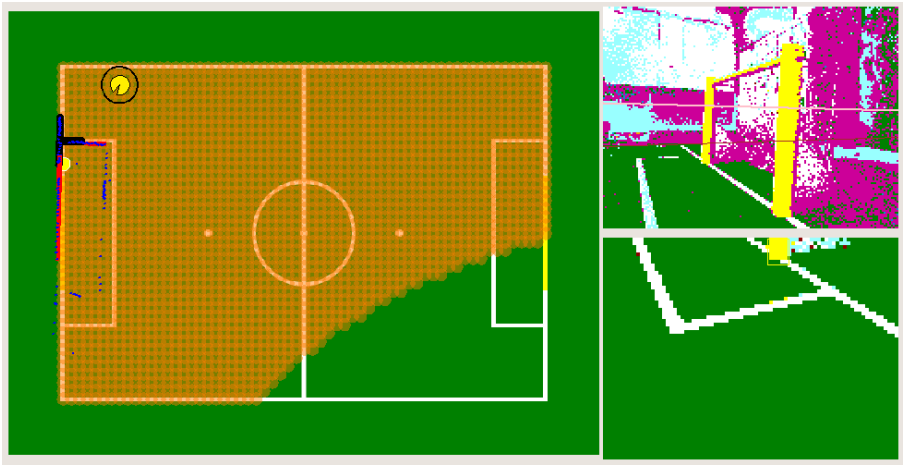

Fig. 4. Sensor model area of convergence given simultaneous observations of a field-line T junction and a goal post. The area of converge to the correct robot pose (shaded in orange) assumes the initial robot pose heading error is less than 45 degrees.

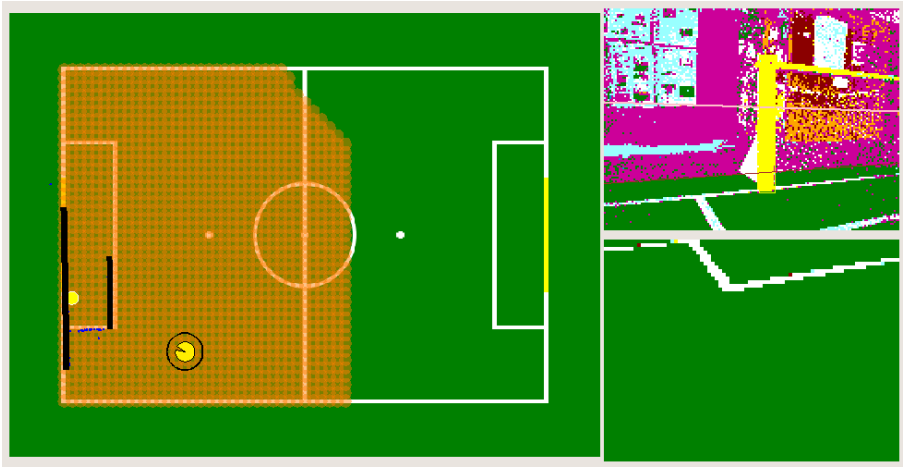

Fig. 5. Sensor model area of convergence given simultaneous observations of parallel field-lines and a goal post, assuming initial heading error of less than 45 degrees.

As to be expected, the area of convergence is largest when multiple field-features are observed or if the field-features seen are relatively unique. As shown in Figure 7, when only a single field-

line is observed, the sensor model can only converge to the correct robot pose if the component of robot pose in the direction of the line is already correct.

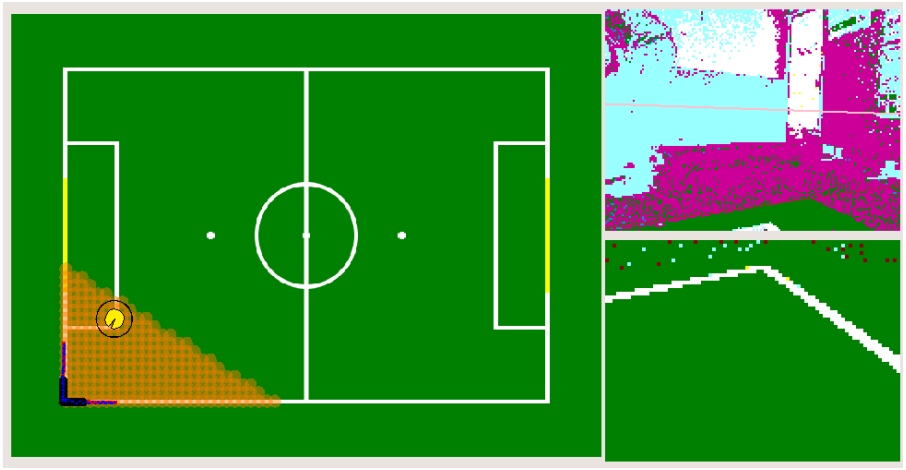

Fig. 6. Sensor model area of convergence given a single observation of a field-line corner, assuming initial heading error of less than 45 degrees.

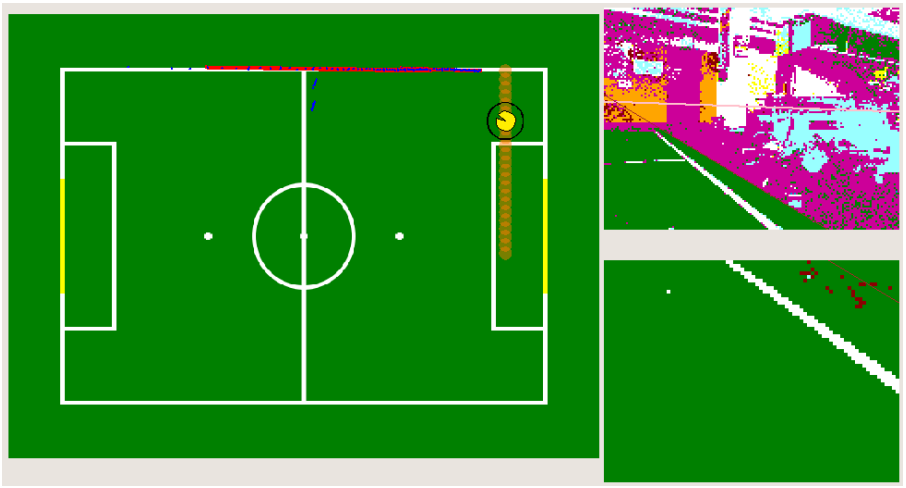

Fig. 7. Sensor model area of convergence given a single observation of a field-line, assuming initial heading error of less than 45 degrees. In this case the sensor model can only converge to the true robot pose if the component of the robot pose in the direction of the field-line is already correct.

Randomly Sampled Field Positions This section evaluates the effectiveness of the unified field-feature sensor model more formally, by quantifying the sensor model's area of convergence to the correct robot pose, and its positioning error, at a range of different field locations. To represent the diversity of robot poses achieved during an SPL match in this test, 20 locations on the SPL field were chosen using uniform random sampling from the robot's x, y, θ field configuration space, as shown in Figure 8.

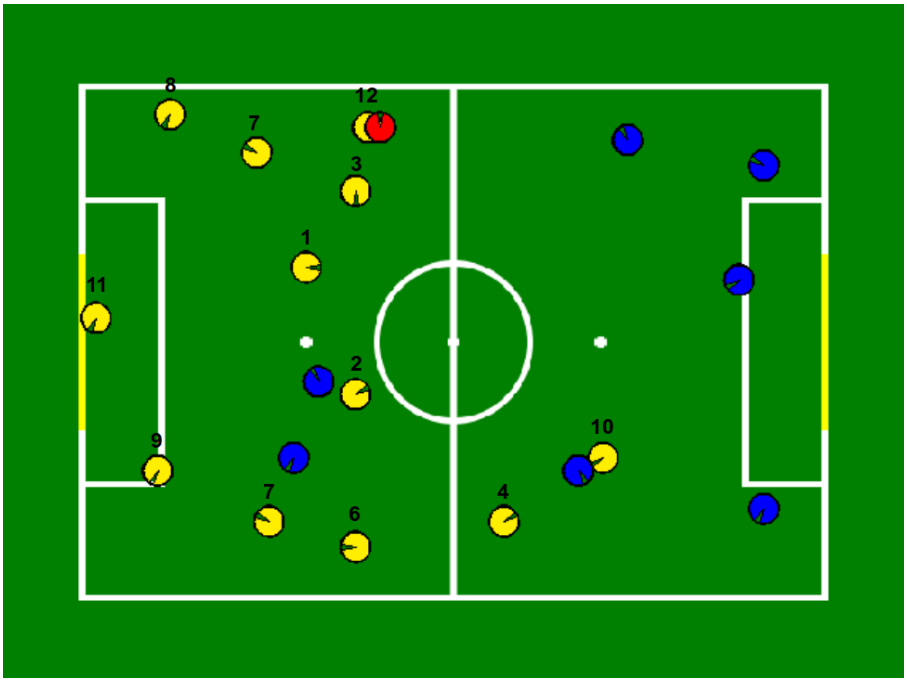

Fig. 8. Randomly determined robot poses used for sensor model evaluation. In positions marked with yellow, multiple field-features were visible. From blue positions, only a single field-line can be observed, and from the red position no field-features were visible. Position numbers correspond to Figure 9.

This data confirms how frequently the general assumption of one sensor update per time step is violated. In this data set of 20 randomly selected field positions, after excluding field edges, 5% of field positions yielded no useful observations (shown in red), 35% of positions yielded observations of a single field-line (shown in blue), and in 60% of positions multiple field-features could be observed (shown in yellow). The incidence of various types of field-feature observation in this dataset is presented in Table 1. Note that the frequency of each observation owes more to the capabilities of the rUNSWift vision system than any inherent feature of the SPL field. While the rUNSWift vision system performs reasonably well, goal posts, corners and T junctions are not always detected, and the range for detection of field-lines is limited.

Table 1. Incidence of various field-feature observations in 20 randomly selected field positions.

Field-feature	Incidence
Field edge	80%
Single goal post	15%
Two goal posts	10%
Centre circle	20%
Field-line corner	5%
Field-line T junction	5%
Parallel field-lines	10%
Field-lines	95%

After discarding positions where only one or zero field-features could be observed, the performance of the sensor model in the remaining 12 positions with multiple field-feature observations was analysed. The area of convergence to the correct robot pose, given initial pose heading errors of plus and minus 15 degrees, 30 degrees, 45 degrees and 90 degrees respectively was used to evaluate the sensor model. As previously discussed, since field-features can not resolve one end of the field from the other, an area of convergence close to 50% with 90 degrees heading error indicates that the sensor model can localise the robot from a single observation. As indicated in Figure 9 Left, this standard was achieved in 5 out of 12 (42%) of randomly determined field positions with multiple visible field-features, and 5 out of 20 (25%) of all field positions. The most informative field positions (positions 1 - 3) all contained observations of the centre circle, whilst the least informative field positions (10 - 12) consisted of observations of two field-lines not combined into a T junction or corner observation, or a single field-line and a goal post.

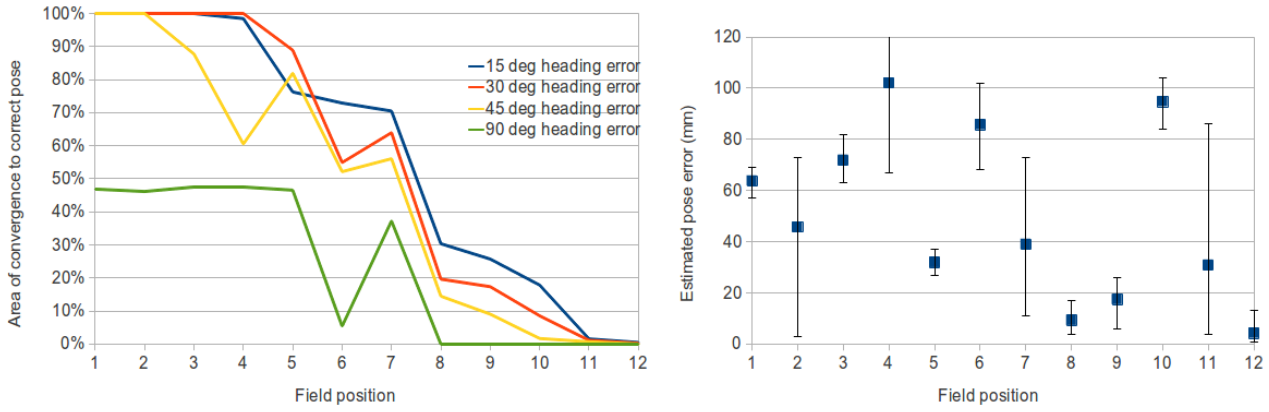

Fig. 9. Left: Area of convergence to the correct robot pose over 12 randomly determined field positions with multiple visible field-features. Right: Positioning error in mm of the final sensor model pose estimate at the same field positions.

Figure 9 Right illustrates the mean, minimum and maximum positioning error in mm achieved by the sensor model in repeated trials at the same field positions. In general, the positioning error of the sensor model is primarily determined by the accuracy of the field-feature observations, which is a function of the distance to the observation, the resolution of the camera image, and errors in the robot's kinematic chain. The mean positioning error of the sensor model across all trials in all 12 positions was 50 mm.

2.4 Evaluation

The unified field-feature sensor model presented in this chapter overcomes a number of the limitations of more conventional sensor models when observed features are heavily aliased, and multiple simultaneous observations are made. It does this by ensuring the robot pose estimate uses all the available information at once. This is achieved by matching field-features to the map in a hierarchical fashion, representing all field-features as point pairs, and drawing on the ICP algorithm to minimise the squared positioning error over all field-features simultaneously.

The results indicate that the sensor model is able to localise the robot from a single observation in 42% of randomly determined field positions with multiple field-features visible. However, it is not possible to distinguish one end of the SPL field from the other using field-features. As such, the sensor model is at risk of converging to an aliased field position if the robot pose initialisation is very poor. Chapter 4 introduces a natural landmark localisation system to resolve this ambiguity.

3 1D SURF Features

Both the natural landmark localisation system in Chapter 4 and the visual odometry module presented in Chapter 5 are based on 1D SURF local image features. This chapter describes how these features are detected and extracted from an image, and presents results addressing the repeatability of the features in the presence of lighting changes, scale changes, and small changes in viewing angle or to the scene itself.

3.1 Background

SIFT [22] and SURF [5], [4] are examples of existing feature representations that are well documented in the literature. Both feature representations are designed to be stable under scale and viewpoint changes. Each method identifies potential features by searching for extrema at all possible scales of a grey-scale image. In SIFT, this step is implemented efficiently by using the difference of Gaussians function applied in scale-space to a series of smoothed and re-sampled images. Once features have been identified, they are accurately localised in both scale and location by interpolating from a 3D quadratic function fitted to local sample points. Next, feature points that are poorly located along an edge are eliminated and an orientation is assigned to each feature, so all future operations can be performed in a rotation invariant manner. SIFT calculates a 128-dimension descriptor vector for each identified feature based on the 8-bin histogram of the image gradient in 4x4 subregions around the feature point location. This, combined with the use of a Gaussian weighting function and normalisation of the descriptor vector, produces features that are invariant to scaling and rotation, as well as small viewpoint and illumination changes.

SURF is related to SIFT, but instead of using a Difference of Gaussian filter, SURF uses simple box filters which can be evaluated very efficiently using integral images. Box filters are used to approximate Gaussian second order partial derivatives and find the determinant of the Hessian matrix, which is referred to as the blob response at a particular location and scale. Features are yielded at local maxima of this response, found by thresholding the response and applying non-maximal suppression in a 3x3x3 neighbourhood over the image and in scales. Like SIFT, SURF also involves feature localisation by interpolating from a fitted 3D quadratic function, and orientation assignment. The SURF feature descriptor uses integral images in conjunction with Haar wavelets to calculate a 64-dimension descriptor vector. This is calculated by summing both the signed and absolute values of both the horizontal and vertical Haar wavelet response over 25 sample points to generate a 4-dimension vector in each of 4x4 subregions around the feature point location.

A comparison of SIFT and SURF using a standard testing procedure based on a range of real-world images found SURF to be faster and more accurate than SIFT [20]. For this reason, SURF has been chosen as the the basis for the 1D feature representation presented in this chapter. As previously noted, both SURF and SIFT are relatively computationally expensive and difficult or impossible to implement in real time on a resource constrained robot in their original formulation.

Several other papers have adapted feature representations to operate on 1D data, however in these cases SIFT was chosen as the candidate rather than SURF. In the closest work to ours [8], [9], [10], a 1D variation of SIFT is used to localise a mobile robot fitted with an omni-directional camera. This was achieved by identifying SIFT-like features in a 1D circular panoramic image, calculating feature descriptors based on colour and curvature information, and using a circular dynamic programming algorithm to match features between images. In comparison to this work, our target platform is the Nao humanoid robot, which has a horizontal camera viewing angle of only 47.8 degrees (v3.2) or 60.9 degrees (v4). This dramatically reduces the amount of information available in a 1D horizon image. Furthermore, for reasons of computational efficiency we do not use colour information, and use SURF rather than SIFT as the basis for our method as previously mentioned.

3.2 Method

In many respects the 1D SURF algorithm represents the equivalent of SURF, but using only one image dimension rather than two. SURF searches for blob response extrema in a 3D scale-space consisting of horizontal location, vertical location and scale. In 1D SURF, the search is conducted in a 2D scale-space consisting of horizontal location and scale only. However, there are also some other significant modifications and simplifications which were made to the original algorithm, as outlined below.

As indicated in Figure 10 Left, the input to the 1D SURF algorithm is a single row of grey-scale image pixels. The intensity values of these pixels are calculated by sub-sampling every 4 pixels along the robot's horizon, and taking the sum over a band of 30 vertical pixels at each sample point. The vertical sum minimises the sensitivity of extracted features to errors in the location of the horizon, and the sum is faster to compute than the mean. The resulting increase in pixel intensity values can be compensated in the response threshold. The position of the horizon in the image is determined by reading the robot's limb position sensors and calculating the forward kinematic chain from the foot to the camera, in accordance with the Denavit-Hartenberg parameters previously determined by the rUNSWift team [19].

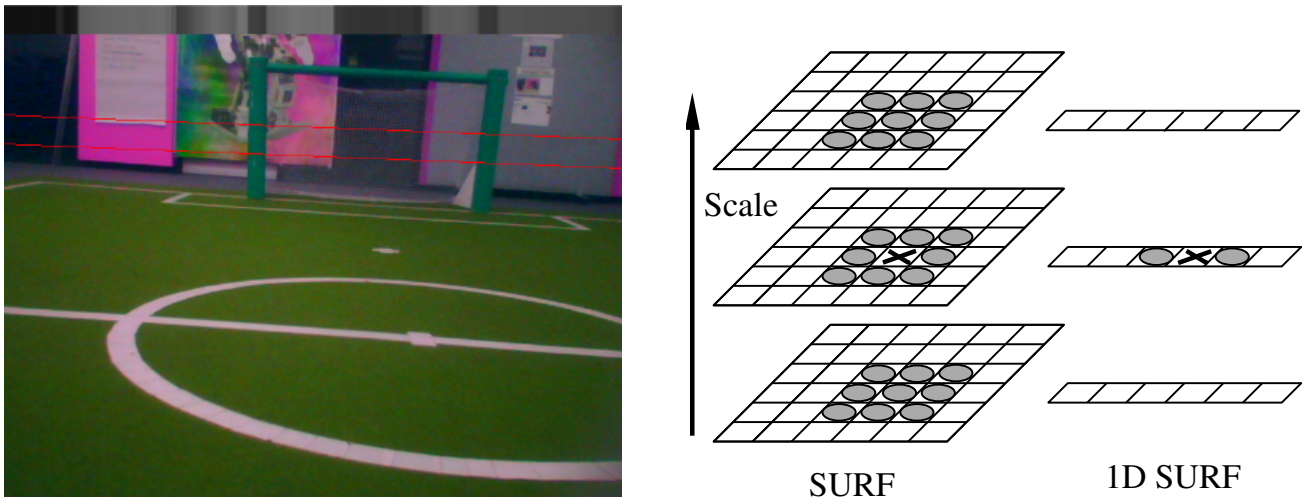

Fig. 10. Left: Image captured by the Nao robot showing superimposed 30 pixel horizon band in red, and the extracted grey-scale horizon pixels at the top of the image. Right: Identification of local maxima in scale-space. Pixel 'X' is selected as a maxima if it is greater than the marked pixels around it.

To identify local maxima of the blob response in scale-space, SURF thresholds the responses, then each pixel in 3D scale-space is compared to its 26 neighbours in a 3x3x3 neighbourhood to determine if it is a local maximum. In the case of 1D SURF, rather than searching for local maxima in a 3x3 scale-space neighbourhood, we apply a weaker test and only require that responses be extrema in the single space dimensional, as illustrated in Figure 10 Right. This relaxation ensures that sufficient feature points will be detected. It is an important aspect of the approach that a large number of relatively poor-quality features are generated, rather than relying on a small number of very distinctive features. A typical 1D horizon image containing 640 pixels might generate 50 - 70 features, depending on the parameter values chosen. In our case we use a scale-space consisting of 4 octaves of 3 intervals each.

Since in 1D SURF all features are defined with reference to the horizon, the SURF orientation assignment step is no longer necessary and can be disregarded. SURF interpolates the location of features in both space and scale to sub-pixel accuracy by fitting a 3D quadratic curve to the local image function. In our application, we found that the additional accuracy provided by this step was not worth the computational burden, and it was also discarded. Finally, although the 1D SURF feature descriptor is calculated analogously to the SURF feature descriptor, due to the reduction in sample space and by using 3 subregions instead of 4, we produce a 6-dimension feature descriptor rather than a 64-dimension feature descriptor, allowing for much faster matching of descriptors across images.

Application to Natural Landmark Recognition In order to illustrate the characteristics of 1D SURF features, simple naive nearest neighbour (NN) based matching techniques are presented here to store, and subsequently recognise, natural landmarks using 1D SURF features. Ultimately these naive techniques were not used for the natural landmark localisation system presented in Chapter 4, since they do not scale efficiently. However, these techniques do provide a good insight into the characteristics of the features. The NN method works as follows: Given a test image and a stored image, landmark recognition is performed by matching features in the test image to their nearest neighbours in the stored image, based on the Euclidean distance between feature descriptors. As before [22], feature matches are considered to be valid if the nearest-neighbour distance ratio is less than 0.7. Similarly [10], we then assign a recognition score to the test image calculated as the sum over all valid matched features of the inverse distance between feature descriptors. A high recognition score indicates that the test image contains the same natural landmark as the stored image with high likelihood.

The above method, which we will refer to as NN matching, does not preclude feature matches that are out of order, or otherwise inconsistent in terms of scale or horizontal displacement. Therefore a second matching method is presented, which first matches nearest neighbour features, and then uses RANSAC [17] to discard feature matches that do not agree on a consistent landmark pose, before recalculating the recognition score. A consistent pose is defined as a set of matched features that conform to a straight line matching function as follows, where $x_{test,i}$ and $x_{stored,i}$ represent the horizontal pixel location of the ith matched feature in the test and stored images respectively, and β_s and β_d are scaling and displacement parameters:

$$x_{test,i} = \beta_s x_{stored,i} + \beta_d \tag{8}$$

Given the robot's limited horizontal field of view, we find a straight line matching function is a reasonable approximation of the true feature matching function, which is curved in the presence of translation. Compared to NN matching, recognition scores calculated with this method will be lower, but have potentially greater discriminatory power. We will refer to this method as nearest neighbour matching with RANSAC (NN with RANSAC). The further advantage of this method is that it provides useful information about the robot's motion between the two images. The use of RANSAC to discard inconsistent matches generated by NN matching is shown in Figure 11.

3.3 Results

Two experiments were used to evaluate the performance of 1D SURF features for robot localisation. In each experiment, the images used were captured using the Aldebaran Nao RoboCup

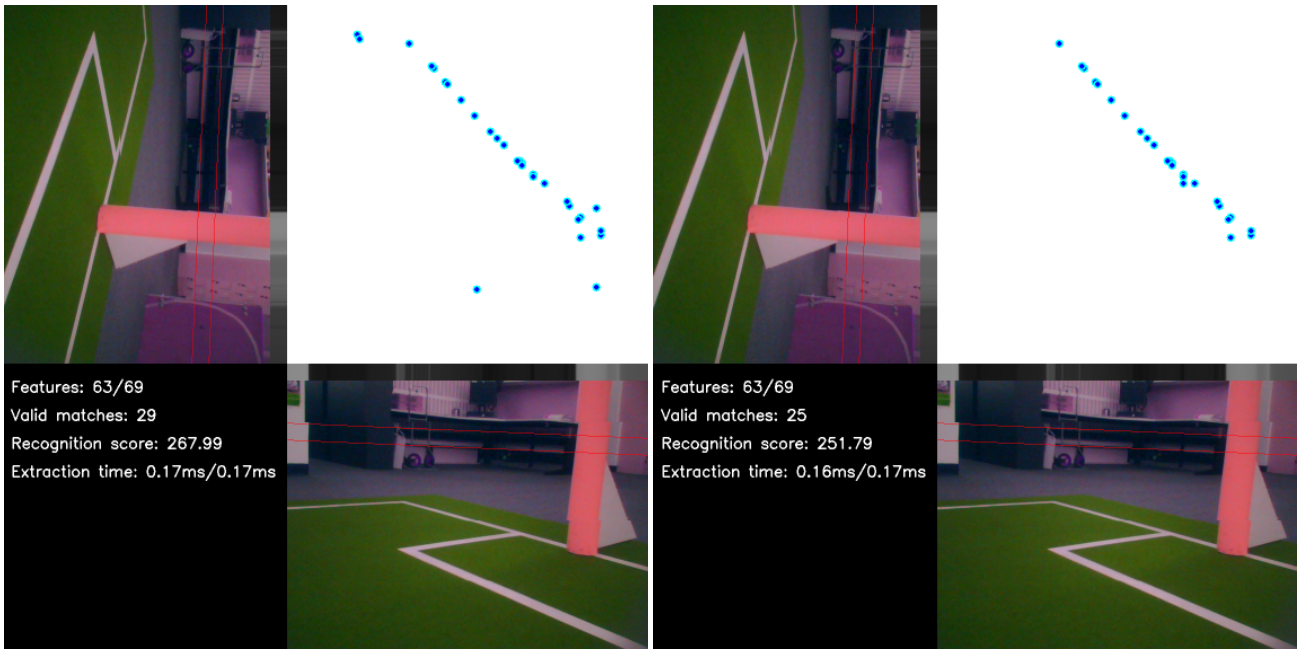

Fig. 11. Left: Matching features in two similar images based on nearest neighbour (NN) matching. Right: Matching features in the same two images after using RANSAC to discard matches that don't agree on a consistent pose (NN with RANSAC). As in Figure 10, each image displays the horizon band in red and the extracted grey-scale horizon pixels at the top of the image. Matching features are plotted in the top-right panel against their horizon location in each image. The text panel illustrates the number of features detected in each image, the number of matches, the recognition score and the time taken to extract the features on a 2.4GHz laptop.

edition v3.2, a humanoid robot equipped with a 500MHz AMD Geode LX800 processor. The Nao has two 640x480 pixel 30 fps digital cameras, each with a horizontal field of view of 47.8 degrees, which can be accessed one at a time.

Classification Experiment The first experiment was designed as a classification task, to assess whether the recognition score between two images could be used by the robot to determine whether both images contained the same landmark. Data for the experiment was captured by rotating the robot at a single location on the field, and capturing 88 images at approximately 4 degree increments. During this process the background around the field consisted of a typical office environment. From this image library we generated a test bank of 480 matched images and 2,065 unmatched images. Two images were considered to match if the angle between them was less than 20 degrees, implying at least 58% of each image horizon overlapped with the other image. Example images from the test bank and the resulting recognition scores are shown in Figure 12.

Fig. 12. Left: Two images that almost completely overlap. Although the features on the horizon are not very distinctive, a high recognition score is generated using 1D SURF and NN with RANSAC feature matching. Right: Two images with no overlap, resulting in a low recognition score using the same method. As before, matching features are plotted in the top-right panel against their horizon location in each image, and the text panel contains key statistics.

Although this experiment contains no changes in scale, illumination or viewing angle, it provides a useful baseline against which to tune parameters and assess the likely rate of false positive landmark recognitions. Feature extraction and matching was performed off-board the robot using a 2.4GHz Core 2 Duo Processor laptop. This enabled the classification accuracy and speed of 1D SURF to be easily compared against SURF, for which we used the OpenSURF[1] library implementation.

The sensitivity and specificity of SURF (using NN matching) and 1D SURF (using NN matching, and NN with RANSAC matching) with variation in the recognition score discrimination threshold is shown in Figure 13. 1D SURF (using the horizon pixels only) is clearly less robust than SURF (processing the entire image). However, as illustrated in Table 2, 1D SURF uses only a fraction of the features, and runs more than one thousand times faster than SURF in this experiment. With the mean extraction time below 0.2ms, real-time feature extraction on the Nao during RoboCup soccer matches is a clear prospect. Also, using RANSAC to enforce a consist landmark pose results in a small improvement in classification accuracy.

[1] http://www.chrisevansdev.com/computer-vision-opensurf.html

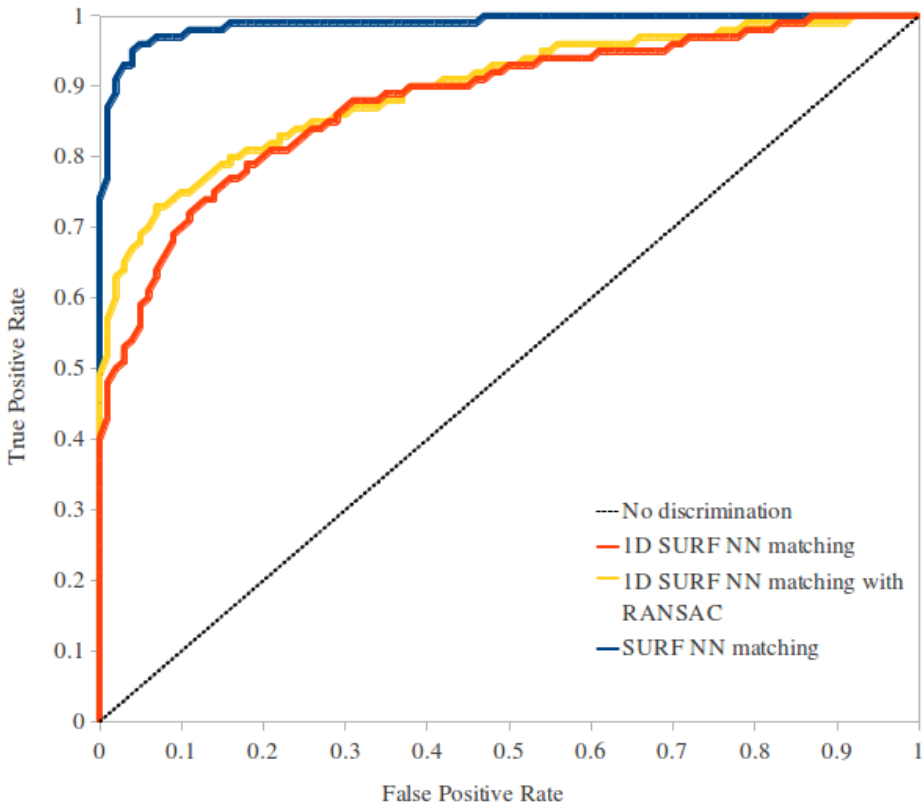

Fig. 13. ROC curve for classifying test images as matched or unmatched using the recognition score. Using NN with RANSAC matching on this data set, a threshold recognition score of 100 captured 70% of true positives with a 5% false positive rate.

Field Experiment Having validated the performance of 1D SURF on highly similar images, the second experiment was designed to assess the performance of 1D SURF under changes in scale, viewing angle, illumination and with small scene changes. It was performed on-board the Nao robot to provide a clearer assessment of the processing speed of the method with constrained hardware. In this experiment, we used NN with RANSAC matching and evaluated 1D SURF the way it might be used in a SPL match; to distinguish one end of the field from the other. To do this, we positioned the Nao in the centre of the field, captured one image of each goal area, and stored the extracted feature vectors. Next, we moved the Nao through a 1 m grid of positions covering a 4 m x 4 m area of the field (25 positions in total), and recorded the recognition scores at each point when manually positioned to face approximately towards each goal. By moving the Nao around the field, large changes in scale and viewing angle were generated. At each point we hoped to observe a large recognition score for the stored goal the robot was actually facing, and a low recognition score for the other goal, indicating that this technique could be used to reliably distinguish field ends during a match. During this entire

Table 2. Running time of feature extraction and matching algorithms evaluated on a 2.4GHz Core 2 Duo laptop.

Feature extraction technique	Feature matching technique	Mean no. features	Mean extraction time (ms)	Mean matching time (ms)	Area under ROC curve
SURF	Nearest neighbour (NN)	429	222.3	19.1	98.8%
1D SURF	Nearest neighbour (NN)	59.2	0.158	0.069	88.0%
1D SURF	NN with RANSAC	59.2	0.158	0.076	89.6%

experiment both goals were coloured yellow, and background objects were approximately 2 m behind the goals themselves.

The recognition scores recorded during this exercise are overlaid on a field map in Figure 14. Using a recognition threshold of 100 (as determined during the classification experiment), each field end is correctly recognised from the single stored image in more than half of the 4 m x 4 m test area. A very strong recognition response (greater than 200) is observed in a radius of approximately 1 m around the location of the original stored image. Finally, there were zero false positives recorded when facing the opposite end of the field. The recognition response to the opposite end of the field is almost always less than 50. Overall, these results indicate that even with just two stored images, a kidnapped robot could resolve one end of the field from the other from most mid-field positions. To provide a clearer indication of the field environment used during the experiment, Figure 16 depicts the stored goal images and examples of views from different areas of the field.

In many robot navigation applications, including RoboCup SPL, robots are subject to varied lighting conditions and the natural landmarks in a given scene will change over time. To test the robustness of 1D SURF in the face of these challenges, we repeated the experiment with the overhead field lighting turned off and both goals removed (to simulate some measurable change to the original scene). The stored features extracted from the original goal images were not changed. As shown in Figure 15, the recognition response to the correct field end is less peaked than before, but the recognition area is still large and again there are no false positives. It is interesting to note that the recognition area for the left-hand goal actually increases once the goal itself is removed. The goal itself can actually be something of a nuisance in the recognition process, since with large perspective changes it occludes features in the background that might otherwise be identified. Using a representative sample of field locations, the mean execution time to extract 1D SURF features on the Nao v3.2 robot was 12 ms. Although this is considerably slower than the 0.158 ms extraction time achieved on the laptop, it is still fast enough to enable features to be extracted in real time at the full 30 fps frame rate of the Nao camera. In later

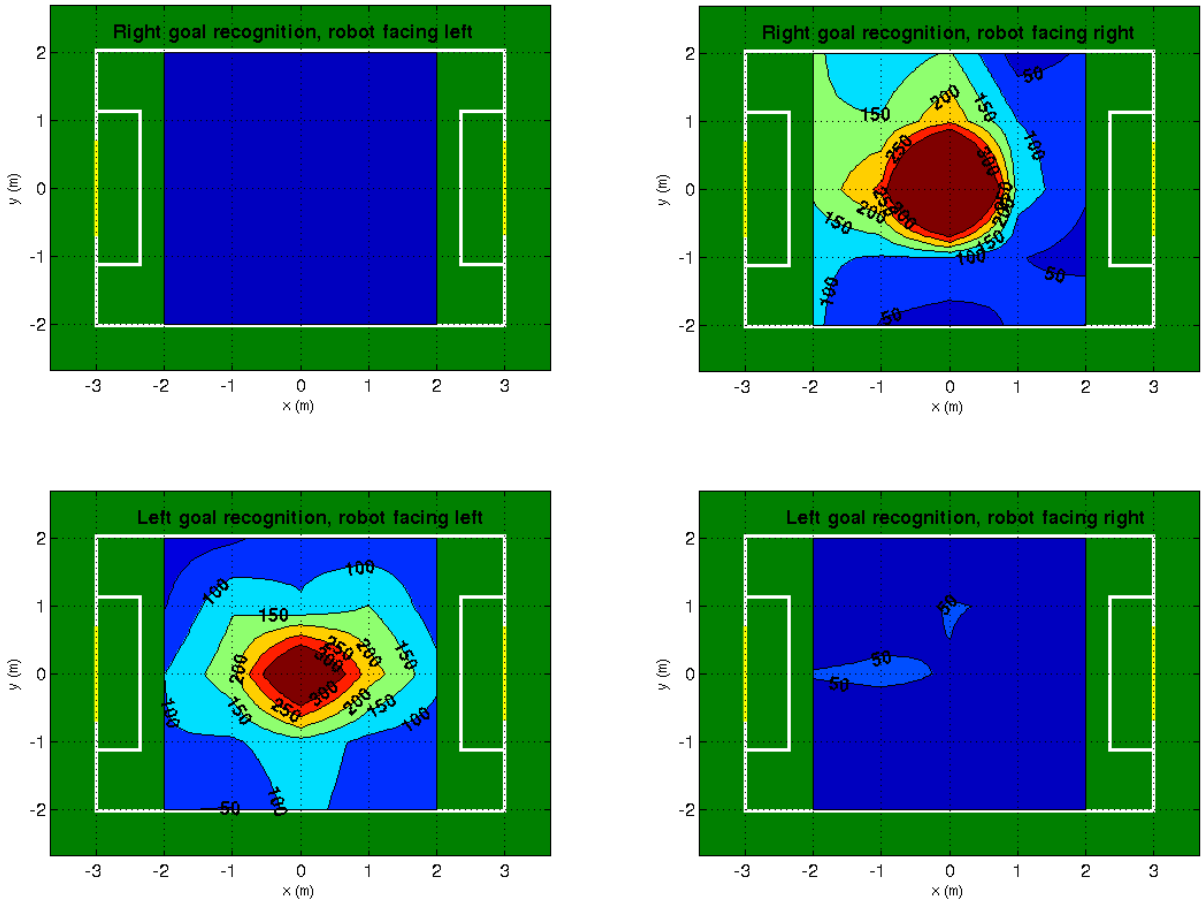

Fig. 14. Recognition scores of a single goal image from different areas of the field. Clockwise from top left: Recognition of right-hand goal when facing left, recognition of right-hand goal when facing right, recognition of left-hand goal when facing right, recognition of left-hand goal when facing left.

experiments with the Nao v4 robot, the mean feature extraction time dropped from 12 ms to 2 ms per image.

3.4 Evaluation

This chapter has presented an optimised method for extracting local features from 1D images of a mobile robot's horizon. The extracted 1D SURF features are robust to lighting changes, scale changes, and small changes in viewing angle or to the scene itself, making them suitable for robot navigation in indoor environments. Results indicate that by using 1D SURF features and a NN with RANSAC matching technique, it is possible to resolve the RoboCup SPL field

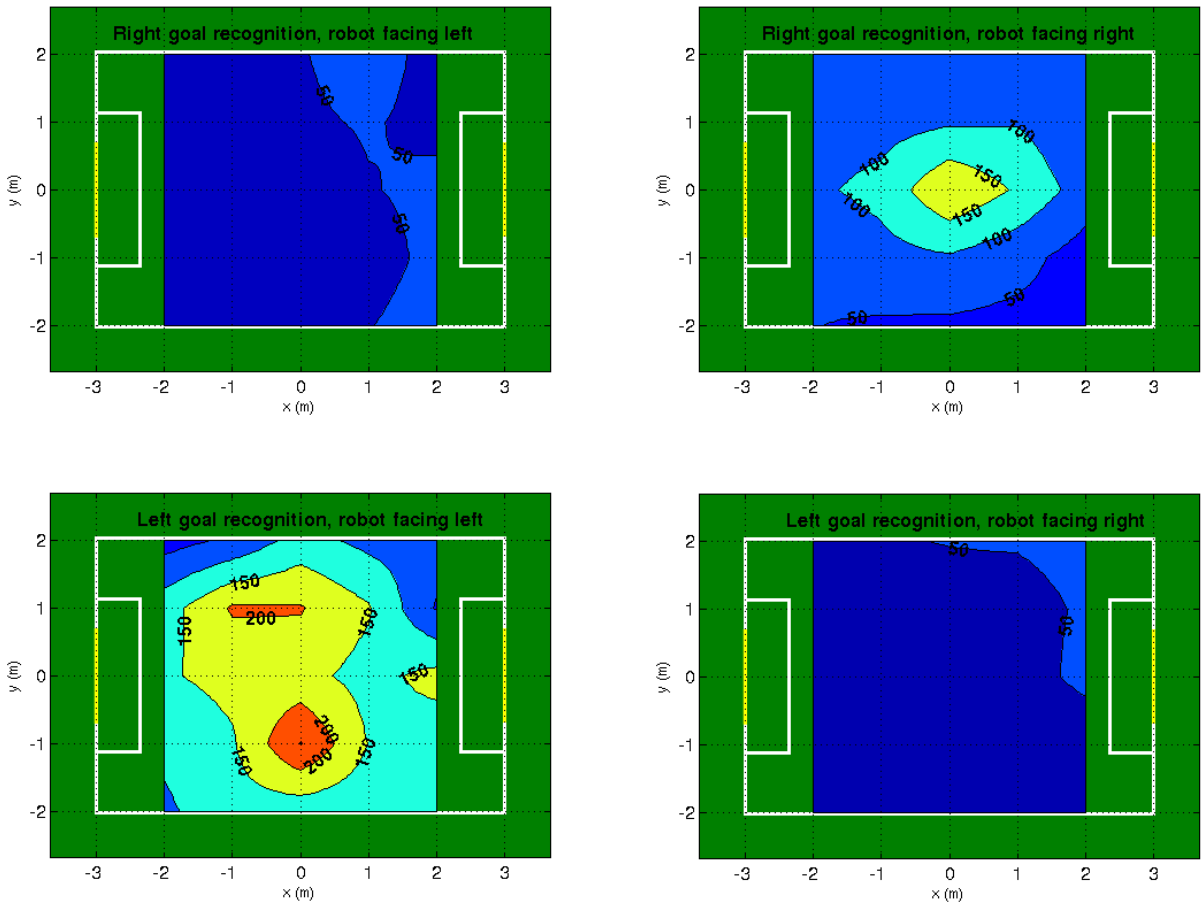

Fig. 15. Recognition scores of a single goal image from different areas of the field, with the overhead field lighting turned off, and the goals themselves removed. Clockwise from top left: Recognition of right-hand goal when facing left, recognition of right-hand goal when facing right, recognition of left-hand goal when facing right, recognition of left-hand goal when facing left.

end ambiguity in real time using just two stored images, at least in a relatively distinctive environment with few scene changes.

In actual RoboCup matches, the background environment is generally much more challenging than these laboratory tests due to the coming and going of spectators during the match. As such, in practise it becomes necessary to store more than two images, and more scalable matching methods are required. These techniques are presented in Chapter 4.

Fig. 16. Top row: Stored images of the left-hand and right-hand goal areas respectively. Row 2: Examples of correctly matched field views. Markers indicate the scale and position of the match. Row 3: Examples of correctly matched views with goals removed and overhead lights turned off. Bottom row: Some field views that could not be confidently matched to the stored images, possibly due to overexposure and occlusion of key features respectively.

4 Natural Landmark Localisation System

This chapter describes how the 1D SURF features presented in Chapter 3 are used as the basis of a natural landmark localisation system for the Nao. The object of the system is only to disambiguate one end of the SPL field from the other. Fine-grained observations are provided by the unified field-feature sensor model introduced in Chapter 2. In order to achieve this objective, the system records a database of up to 40 images of each goal area while walking on to the field at the start of each half. During the match, every camera frame is matched against all the images in this stored database in real time, to help ensure the Nao does not confuse the opponents goal with its own.

4.1 Background

As previously outlined in Chapter 3, the mean execution time required to extract 1D SURF features from a single camera frame on the Nao v4 is approximately 2 ms. The time taken to match these features to features extracted from another image, and calculate a matching score, is typically 3 - 4 ms using naive nearest neighbour (NN) feature matching techniques. For each additional database image, an additional 3 - 4 ms is required. This is too slow given the 30 fps frame rate of the Nao camera, and the myriad of other competing processing requirements for vision, localisation, walking and behaviour. A faster matching method is required.

There are a large number of exact and approximate alternative algorithms that aim to improve the performance of the naive NN algorithm. Exact NN methods typically make use of space-partitioning data structures, that require some preprocessing to form the structure but offer increased performance on repeated queries [34]. Approximate NN methods come in a large range of varieties, with [23] offering a systematic evaluation a range of methods. As documented by [23], these methods offer significant speed ups over the naive NN implementation on datasets containing hundreds of dimensions and hundreds of thousands of points. Unfortunately however, as outlined in Chapter 3, this application requires matches in only six dimensions with typically 50 - 70 points. As a result of this small data size, it was found that the additional overhead required to maintain these more advanced data structures overwhelmed the available efficiency improvement for small numbers of points. Instead, the NN feature matching approach was abandoned in favour of a method that doesn't require features to be matched at all.

An alternative approach to scene recognition in a database of images borrows heavily from methods used for text retrieval. Rather than using features in their raw form for direct image matching, each image is represented as a "bag of words". A bag of visual words is a sparse frequency vector. It counts the number of occurances in that image of each word from a vocab-

ulary of visual words pre-learned from typical features [29]. The location of features within the image is ignored in this representation.

Once both the query image and the database images are represented by frequency vectors, object or scene recognition can be performed by ranking database images according to the cosine of the angle (normalised scalar product) between the query frequency vector and database frequency vectors. However, since not all visual words are equally distinctive, as in [29] some weighting is usually applied to the components of the frequency vectors before retrieval. The standard weighting scheme borrowed from text retrieval is term frequency - inverse document frequency (tf-idf) [3]. Using tf-idf weighting, the ith component of the frequency vector for image d is given by:

$$t_i = \frac{n_{i,d}}{n_d} log \frac{N}{n_i} \tag{9}$$

where $n_{i,d}$ is the number of occurrences of word i in image d, n_d is the total number of words in the image d, n_i is the number of occurrences of word i in the whole database and N is the number of images in the whole database. The effect of this weighting scheme is to increase the weighting on visual words that appear frequently in the image, while decreasing the weighting on words that appear frequently throughout the whole database and are therefore not distinctive.

Having outlined bag of words based scene retrieval, it remains to consider the construction of the visual vocabulary which is used to vector quantise the original local image features (in our case, 1D SURF features). Typically, as in [29], features extracted from a training set of images are clustered using K-means or similar methods. Clusters can then be represented by their centroid, a visual word. When a new image is observed, extracted features are mapped to the nearest cluster. However, since K-means clustering is only locally optimal, the quality of the visual vocabulary produced is dependent on the initial cluster locations used to initialise the K-means algorithm. Evidence suggests that random cluster initialisation often leads to poor quality visual vocabularies. This is because the image feature space tends to have very large variations in density. With random initialisation, the final clustering tends to over-segment the dense region, and provide poor representation of less dense regions [13]. Furthest-first is an alternative initialisation technique that tends to spread the initial clusters more than random initialisation. Subset-furthest-first (SFF) initialisation [33] achieves the same effect but is less likely to pick outliers for the initial cluster centres.

4.2 Method

The pseudo code for the natural landmark localisation system to disambiguate one end of the field from the other is illustrated in Algorithm 1. The system uses the bag of words rep-

resentation for images. It works by keeping a database which contains the frequency vector representation of images of each goal area. At the start of each half, this database is cleared, and then as the robot walks on to the field for the first time the database is populated. It is very important that the database is only populated with correctly classified images, so images are only considered for inclusion if the localisation filter indicates that the robot is facing a goal area, and a goal post is actually visible. Images are populated to the database if they don't trigger a certain number of hits from the existing images in the database, and if the database size is below a maximum level (which was set to 80 total images for SPL games). The database uses tf-idf cosine scores for ranking image matches.

During normal game play, if the localisation system suggests the robot is facing a goal area, then each camera frame will be used to query the database. If the matching images clearly indicate which goal the robot is facing, through a voting system described in Algorithm 1 lines 11 - 21, then it will be considered to be an observation of that goal. To further improve the accuracy of the system, a sliding window of observations is then considered. The probability the robot is facing the away goal is calculated as the mean of a beta distribution, based on these observations, as indicated in lines 22-24 of Algorithm 1. If the away goal probability is above a required confidence threshold (or below one minus the threshold, indicating confidence in the home goal direction), then this is considered to be a certain observation by the rest of the localisation system. This will cause the localisation filter to be re-initialised facing that direction. Finally, note that detected 1D SURF features that are deemed to be part of another robot, according to the robot detection system outlined in Chapter 6, are discarded, in both the database building and retrieval stages.

Enhancements to the Bag of Words Retrieval System Although Algorithm 1 provides a high-level overview of the system, the system also includes a number of additional refinements to improve the quality of database query results. These enhancements include geometric validation, query expansion, subset-furthest-first (SFF) vocabulary initialisation, and use of a stop list, as described further below.

Arguably the greatest weakness of the bag of words approach to image representation is that it disregards the geometric relationships between features. To mitigate this problem, the database implementation actually preserves the original pixel locations of the detected 1D SURF features before they are mapped to frequency vectors. By preserving this information, a geometric verification can be performed to help improve database query results. To this end, all 1D SURF features that map to the same visual word are considered to be possible feature matches, and a RANSAC line is fitted through these possible feature matches as it was in

Algorithm 1: Overview of the natural landmark localisation algorithm.

Input: f_t: Set of 1D SURF features extracted from frame at time t, D: Database of labelled goal area images, V: Visual vocabulary

Output: $P(Away)_t$: Probability robot is facing the Away goal at time t

if *Start of new half* **then**
  clear D;
else if *Entering field for the first time this half, and localisation filter indicates robot facing a goal area, and a goal post is visible* **then**
  tf_t = mapToFrequencyVector(f_t, V);
  $hits$ = queryDatabase(D, (tf_t+tf_{t-1}));
  if *count(hits) < minRESPONSE and size(D) < maxSIZE* **then**
    addToDatabase(D, (tf_t+tf_{t-1}));
else if *Playing, and localisation filter indicates robot facing a goal area* **then**
  tf_t = mapToFrequencyVector(f_t, V);
  $hits$ = queryDatabase(D, (tf_t+tf_{t-1}));
  $awayGoalVotes = 0$;
  foreach *hit i* **do**
    if *i was recorded while facing the Away goal* **then**
      $awayGoalVotes++$;
    else
      $awayGoalVotes--$;
  $awayGoalObs_t = homeGoalObs_t = 0$;
  if $awayGoalVotes \geq minCONSENSUS$ **then**
    $awayGoalObs_t = 1$;
  else if $awayGoalVotes \leq$ *-minCONSENSUS* **then**
    $homeGoalObs_t = 1$;
  $\alpha = 1 + \sum_{i=t-N}^{t} awayGoalObs_i$;
  $\beta = 1 + \sum_{i=t-N}^{t} homeGoalObs_i$;
  $P(Away)_t = \alpha/(\alpha + \beta)$

Section 3.2. Using this system, database images are only considered to be a match to the query image if the cosine score is above a required threshold, *and* there is a sufficient number of inliers in the RANSAC line fitted through possible matching features.

Query expansion is a process by which geometrically verified images from the initial set of database hits are used to build a richer model for the query. The database is then re-queried using this richer model. Evidence suggests that this technique significantly improves the performance of bag of words scene retrieval, at little cost [2]. In this application, query expansion is implemented quite simply by adding the frequency vector from the prior camera frame to the current frame frequency vector, before querying the database, as indicated in Algorithm 1 lines 5, 7 and 10 and Figure 17. The advantage of this approach is that only one database query is made, rather than two queries in the case of methods such as Average Query Expansion [2]. Furthermore, the use of two query images also reduces the sensitivity of the system to errors in the location of the image horizon line (refer Section 3.2).

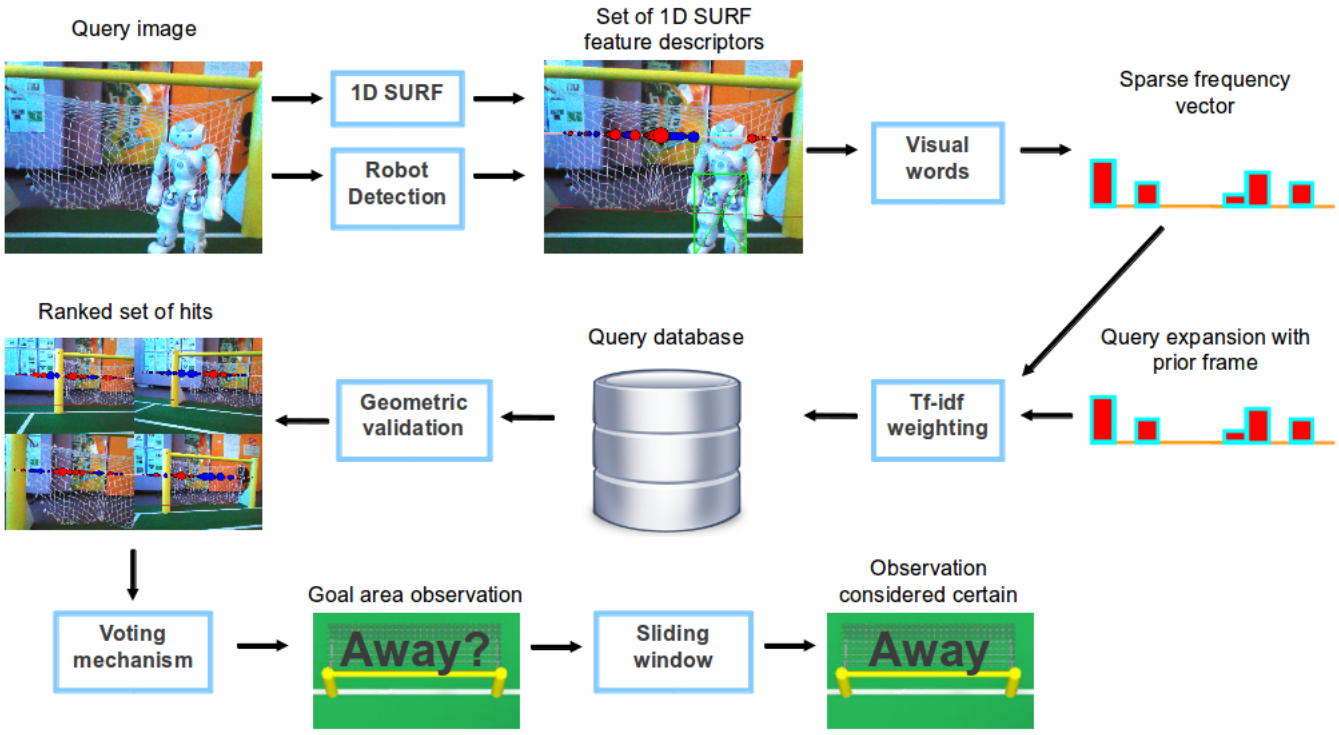

Fig. 17. Diagrammatic overview of the natural landmark localisation algorithm in retrieval mode. This occurs during normal game play whenever the localisation filter indicates the robot is facing a goal area. It is not necessary for a goal post to be visible.

The two remaining enhancements to the bag of words retrieval system relate to the construction of the visual vocabulary, which is very important for system performance. Firstly, consistent with [29], a stop list is used. Stop lists are used to repress visual words that are too common, leading to mismatches, or so rare that they are insignificant and don't contribute to retrieval performance. This is analogous to the removal of common words such as "the", and"a" from text searches. The size of the stop list is determined empirically, based on the frequency of visual words in the training image set. It appears that the optimal size of the stop list is related to the cluster initialisation technique used for visual vocabulary learning. In this application, subset-furthest-first (SFF) initialisation was used and the bottom 5% if visual words by frequency were stopped, relative to [29] who used random initialisation and stopped the top 5% and the bottom 10%. The size of the visual dictionary used for the natural landmark localisation system was 200 words, again determined empirically.

4.3 Results

Classification Experiment In order tune the parameters of the bag of words retrieval system, the classification exercise introduced in Section 3.3 was repeated. Rather than using the NN matching score as the classifier, however, the tf-idf cosine between two images was used. Since the tf-idf cosine calculation depends on the frequencies of visual words across the entire database, as previously described, it was necessary to load all 88 images into the database before any classification could take place. In effect, the bag of words retrieval system has the benefit of information about the entire database when performing scene recognition. The NN direct image matching approach does not have this global knowledge. This allows the bag of words method to outperform the NN matching approach as indicated in the results below, notwithstanding a significant loss of information (and corresponding efficiency improvement) from the vector quantisation of the 1D SURF features.

As illustrated in Table 3, using SFF initialisation with the bottom 5% of visual words by frequency stopped, the tf-idf cosine method outperformed NN with RANSAC as a classifier with an area under the ROC curve of 91.3% versus 89.6%, even without the query expansion and geometric validation steps that were subsequently implemented in the natural landmark localisation system. It can also be seen that SFF initialisation of the visual vocabulary offers a robust performance boost compared to random initialisation. Using SFF initialisation, the use of stop words provides only a small additional improvement in accuracy. Execution times are not listed since the tf-idf cosine method is not directly comparable to the NN matching techniques on this basis, as it is designed to search an entire database of images rather than perform matches between single pairs of images.

Table 3. Classification performance of tf-idf cosine scores vs. NN direct image matching approaches.

Feature extraction technique	Feature matching technique	Vocabulary initialisation	Area under ROC curve
SURF	Nearest neighbour (NN)	-	98.8%
1D SURF	Nearest neighbour (NN)	-	88.0%
1D SURF	NN with RANSAC	-	89.6%
1D SURF	Tf-idf cosine	Random	88.4%
1D SURF	Tf-idf cosine	Subset-furthest-first (SFF)	91.1%
1D SURF	Tf-idf cosine	SFF with bottom 5% stopped	91.3%

The ROC curve illustrating the sensitivity and specificity trade-off for each classifier is shown in Figure 18. Note that since the performance of the visual vocabulary is sensitive to the initialisation, which contains randomness even in the case of SFF, these results reflect the best tf-idf matching performance from 10 random initialisations. Figure 19 Right illustrates the distribution of performance of tf-idf classification with SFF and bottom 5% of words stopped, indicating on one out of 10 occasions it performed worse than NN with RANSAC matching. Figure 19 Left illustrates the differences in visual word distribution under random and SFF initialisation, indicating that SFF initialisation results in a more balance distribution of visual words in the training feature set.

Open Challenge Demonstration In order to evaluate the effectiveness of the complete natural landmark localisation system, a demonstration was devised which was also used as the rUNSWift SPL Open Challenge entry, which was awarded second place. During this demonstration, the robot is allowed 45 seconds to walk on to the field to the kick-off position, mapping landmarks as it walks. SPL matches also begin in this fashion. After the 45 seconds expires, the robot can be kidnapped to any field location and it is expected to return to the kick-off position on the correct side of the field. The robot is deemed to have arrived when it stops walking and the time taken to localise and return to the kick-off position is recorded. Although this experiment tests the complete localisation system, it also provides a good indication of the capabilities of the natural landmark localisation system to resolve one end of the field from the other, since after kidnapping this is not known by the robot except through natural landmarks.

As an experiment, this demonstration was undertaken in the rUNSWift lab with 10 kidnap positions as illustrated in Figure 20. Position 1 is both the kick-off position and also the first kidnap position (the robot was just picked up and replaced in the same location). Table 4 records the robot's final positioning error, the time taken, and whether the robot was positioned on

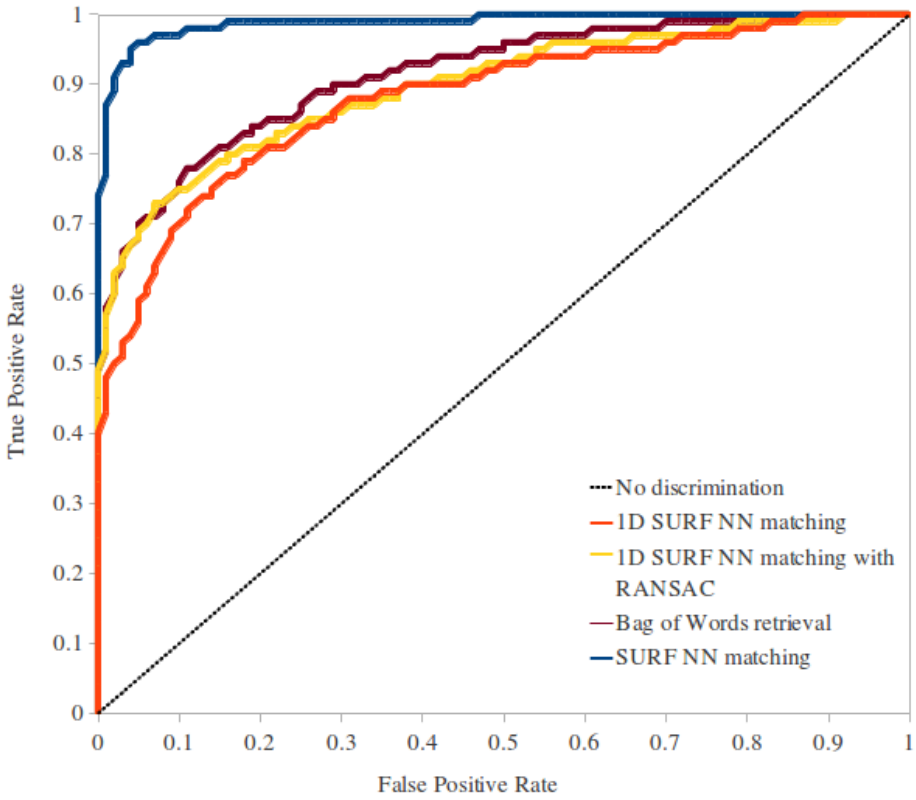

Fig. 18. ROC curve for classifying test images as matched or unmatched using the bag of words retrieval system with Subset Furthest First (SFF) initialisation and bottom 5% of words stopped. The bag of words method outperforms direct image matching techniques at this task.

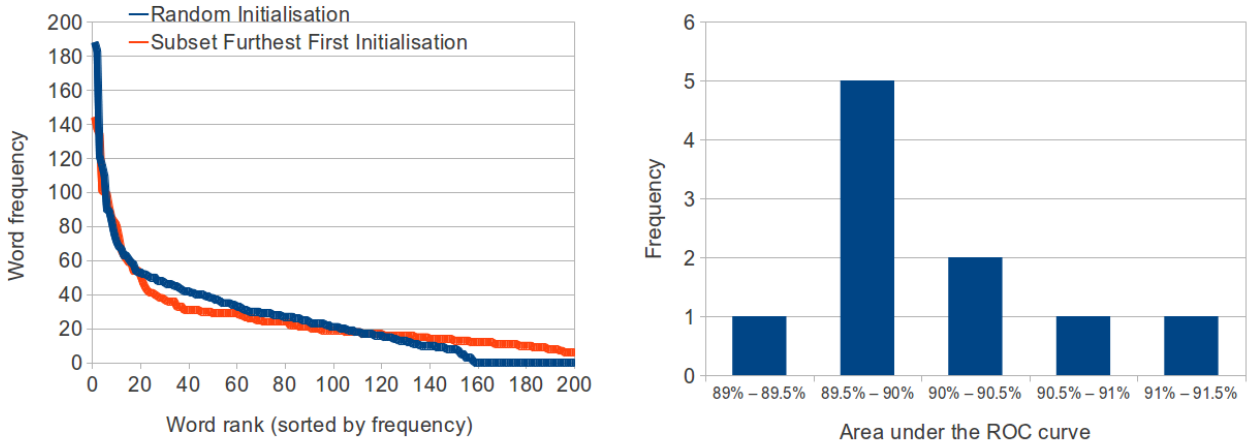

Fig. 19. Left: Resulting visual word distributions under random K-means initialisation and Subset Furthest First (SFF) initialisation. SFF results in a more balanced distribution of visual words than random initialisation. Right: Histogram of tf-idf classification performance based on SFF with bottom 5% of words stopped over 10 random initialisations. On one occasion, performance was worse than the NN with RANSAC matching approach.

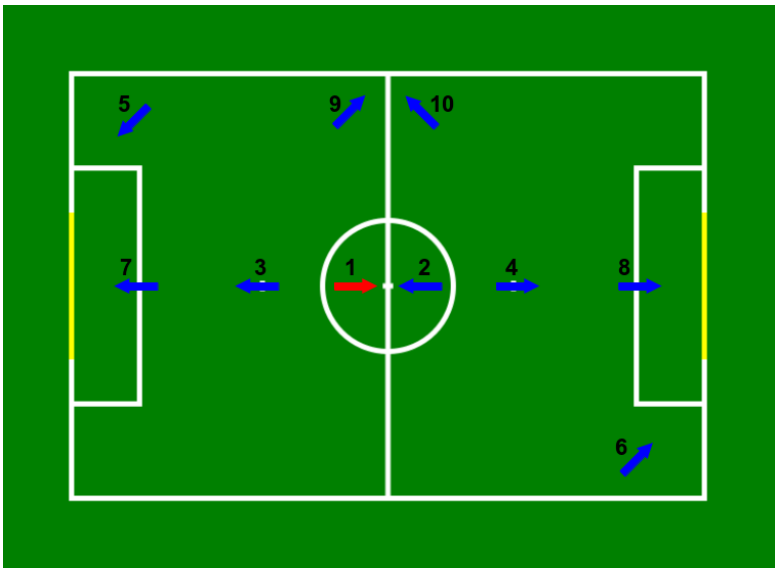

Fig. 20. Illustration of the positions used in the kidnap experiment. In each case the robot was required to return to the kick-off position labelled as position 1.

the correct end of the field in each of the 10 trials. A short low resolution video illustrating the kidnap and recovery process is also available[2]. As indicated by Table 4, in every trial the robot was able to correctly resolve field-ends and return to the correct side of the field. In trial 5, the robot initially walked to the wrong side of the field but then corrected itself (before it stopped walking), inflating the time taken on this trial. In trial 1, the robot was just picked up and replaced, and did not move, so the time taken is 0 seconds. The mean final positioning error over the the 10 trials was 44 mm. During this experiment, the mean execution time required to process calls to the natural landmark localisation system on a Nao v4 was 10.2 ms. This cost is only incurred when the localisation filter indicates the robot is facing a goal area. In this case the cost was based on searching over a database of 30 images of the home goal, and 31 images of the away goal.

Although this system exhibits excellent performance in the lab, it is worth noting that during the Open Challenge demonstration in Mexico, only two out of four attempted kidnap recoveries were successful. On the other two occasions, the robot positioned itself in the aliased position on the opposite side of the field. There are three likely reasons for this:

1. The dictionary of visual words used by the robots was learned in the rUNSWift lab, and was not updated for the competition environment. The need to mitigate the poor lighting conditions and frequent wireless failures in Mexico imposed significant time constraints on

[2] https://dl.dropbox.com/u/36660950/OpenChallengeDemo.mp4

Table 4. Kidnap performance of the natural landmark localisation system.

Trial	Time taken (s)	Correct field end (Y/N)	Positioning error (mm)
1	0	Y	15
2	12.1	Y	65
3	12.9	Y	53
4	17.1	Y	25
5	32.1	Y	22
6	21.7	Y	95
7	16.9	Y	7
8	25.4	Y	28
9	16.6	Y	88
10	19.7	Y	42

the team. In future it is recommended that a new dictionary of visual words be learned for the competition environment.

2. There was a large number of people clustered around the field during the Open Challenge demonstration. The constant movement of the audience provides a much more challenging environment than the static environment of the lab.
3. The confidence level requirement within the natural landmark localisation system is set very high, and was not tuned separately for either this experiment or the demonstration in Mexico. Whilst in matches it is very important to avoid false positive re-initialisation of the localisation filter, even at the expense of true positives, perhaps a lower confidence level would be more appropriate for demonstration in a situation involving a moving audience.

4.4 Evaluation

This chapter has presented a scalable method for matching camera frames to a database of stored images. Using this technique, a natural landmark localisation system was developed capable of storing up to 80 images of the field goal areas as the Nao robot walks on to the field. During the subsequent game, camera frames are matched to these stored images to resolve the field-end ambiguity in real time. Results indicate that this system is very effective in a static environment, but less effective when a large audience is clustered around the field. These results are consistent with the observed performance of the system during competition matches, where occasionally a robot would still occupy an aliased position on the incorrect side of the field. To further improve the system, it seems necessary to update the stored images of each goal area as the natural landmarks around the field change during the match. This requires a method to simultaneously localise and map (SLAM), rather than relying on a fixed set of stored images.

5 Visual Odometry Module

This chapter describes how the 1D SURF features presented in Chapter 3 are used as the basis of a fast visual odometry module for the Nao. The purpose of the module is to accurately measure changes in the robot's heading by matching 1D SURF features across subsequent camera frames. Using this method, the robot's walk engine odometry can be corrected, resulting in a significant improvement in the accuracy of odometry information provided to the localisation filter. A further benefit of the module is the ability to detect collisions and initiate appropriate avoidance behaviour.

5.1 Background

Visual odometry (VO) is the process of estimating the motion of an agent using only the output of one or more cameras attached to it. Typically, the most important step in a VO algorithm is estimating the agent's relative motion between two subsequent camera images. In the general 3D case, the agent's motion has 6 degrees of freedom (DoF). For agents fitted with multiple cameras with overlapping fields of view, stereo VO algorithms can be used to measure the 3D position of detected features by triangulation at each step, and use this to derive the relative 6 DoF motion of the agent [28].

In contrast, agents fitted with a single camera must use monocular VO methods, of which [28] outlines three categories: feature-based methods, appearance-based methods, and hybrid methods. Feature-based methods rely on the detection of repeatable local image features that can be tracked over multiple frames. Appearance-based methods use the intensity of all pixels in the image or a sub-region of the image to estimate optic flow. Hybrid methods are a combination of the two. Although the Aldebaran Nao is fitted with two cameras, the cameras have minimal overlap and so monocular VO methods must be used.

The general 6 DoF relative motion estimation problem cannot be solved using monocular VO methods in the absence of further information or motion constraints. Generally, only heading information can be accurately obtained, while the absolute scale of motion is ambiguous. However, in indoor mobile robotics, constraints on the motion of the agent, such as the planar motion assumption, have been used to simplify the VO problem [15] [16]. These approaches are typically based epipolar geometry; the assumption of planar motion on a flat surface makes relative pose estimation simpler and more efficient. Using this assumption, methods to determine the scale of motion have also been developed [30]. These methods use the surface context approach developed by [14] to split the single image into three geometric regions: the ground, sky, and other vertical regions. Using this technique, scale can be determined based on the motion of features on the ground plane.

Although impressive results have been reported using visual odometry in many application areas, the use of visual odometry in dynamic environments containing many independently moving objects remains challenging. Typical approaches to feature-based relative motion estimation are sensitive to wrong feature matches or feature matches on moving objects, even with the use of RANSAC based outlier rejection schemes [6]. To overcome these issues in dynamic environments, authors such as [6] have used image patch classification to improve rejection of independently moving objects such as other cars.

5.2 Method

The rUNSWift VO module is of the monocular, feature-based, heading-only variety. Since 1D SURF features are already extracted from camera frames for natural landmark localisation (refer Chapter 4), it is efficient to use these same features for VO. Although it is possible to estimate the scale of camera motion using a monocular VO system as in [30], the detected 1D SURF features are found only on the robot's horizon, not on the ground plane, and therefore do not lend themselves to these techniques. This is of little consequence, however, since as previously outlined, visual heading odometry is the primary interest for the RoboCup application domain. The error in the heading component of the robot's walk-engine odometry has a much greater effect on localisation accuracy than the errors in the forward and sideways components.

The RoboCup field is a dynamic environment, complete with multiple independently moving objects in the form of other robots, referees, and spectators in the background. As with the natural landmark localisation system, any 1D SURF features that are located on detected robots are discarded, to help prevent the movement of other robots on the field from influencing the visual odometry results. This approach is similar in spirit to the feature classification approach used by [6]. The system is also robust to the movement of undetected robots and referees in front of the camera, as described further below.

Having already outlined the 1D SURF feature extraction and matching techniques in Chapter 4, the additional infrastructure required to implement the VO module is relatively straightforward. The system uses the NN matching technique to find matching horizon features between two camera frames. The horizontal movement in pixels of each feature between frames is then calculated. Using this data, only a single parameter needs to be estimated: the robot heading change between the two frames. This is estimated using the mode of all feature movements. Knowing the resolution and horizontal field of view of the camera, it is trivial to convert the robot's heading change from pixels to radians, and to adjust for the movement of the robot's neck joint.

The mode is chosen as the measure of central tendency in this application because, provided the stationary background is always the largest contributor of features in the image, the mode will remain unaffected by the introduction of independently moving objects. Furthermore, if there are many or large moving objects in the frame, and the identification of the static background is uncertain, the distribution of feature movements will be strongly multi-modal. This enables this scenario to be easily detected, in which case the localisation filter reverts to using heading odometry from the walk-engine. In contrast, when the VO module is operating reliably, the distribution of feature movements will be approximately uni-modal. In this case the localisation filter uses visual heading odometry. Using this approach makes the system relatively robust to the movement of undetected robots and referees, as explained in Figure 21. At all times the forward and left components of odometry used for localisation are generated by the walk-engine.

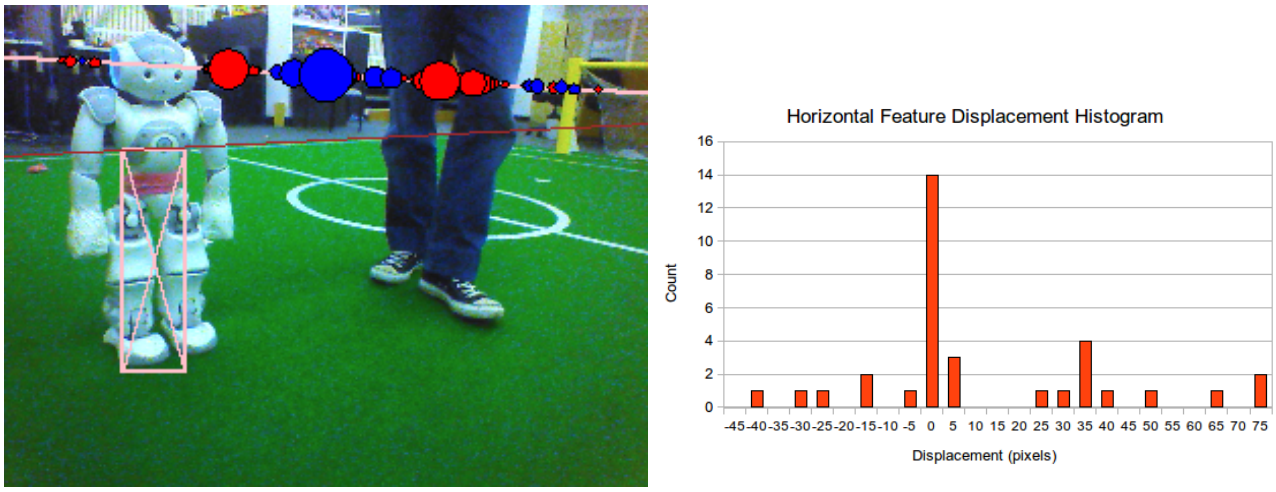

Fig. 21. Left: A typical camera frame captured by the Nao during a match could include both other robots and the referee. In this case the robot is detected and it's horizon features (indicated as blue and red blobs) are discarded. The referee cannot be detected, and constitutes an independent moving object. Right: The distribution of feature movement over subsequent camera frames indicates illustrates two modes: one representing the viewing robot's heading change, measured against static background features, and the other representing the independent motion of the referee. In this case, the larger mode can be easily identified and visual heading odometry determined using this mode. If the two modes were more similar in size the algorithm is able to fail gracefully.

So far, a method has been presented to find the change in heading between two camera frames, assuming that the images contain a reasonable overlap of horizon features. In order to track the robot's heading odometry over longer periods of time, the obvious approach is to integrate heading changes over all adjacent frames of a video sequence. However, to minimise drift from the accumulation of frame to frame errors, and to improve the robustness of the

system, it may not be optimal to include every adjacent frame in the odometry sequence. For example, a single image may be corrupted by horizon error, blur, or feature occlusion. In this case, if the robot's heading is changing slowly, the odometry will be more accurate if this frame is discarded from the odometry sequence.

In order to make the VO module robust to these single frame errors, at each time step the heading change between the current frame and each of the three previous frames is calculated. Current heading odometry is then calculated relative to the 'best' prior frame. The notion of the 'best' prior frame takes into account both the reliability of the prior frame's own odometry estimate, and the confidence level of the heading change calculation between the two images. More formally, at time t the robot's heading odometry is calculated relative to prior frame K_t given by:

$$K_t = \underset{k \in \{t-1, t-2, t-3\}}{\arg\max} \{\min\{reliability_k, confidence_{k,t}\}\} \tag{10}$$

where the $reliability$ of the odometry at frame t is determined recursively by the $reliability$ of the prior frame odometry and the $confidence$ of the heading change estimate between K_t and t:

$$reliability_t = \min\{reliability_{K_t}, confidence_{K_t,t}\} \tag{11}$$

$$reliability_0 = \infty \tag{12}$$

The measure of the confidence of the heading change estimate between two frames could be calculated in several different ways. In this case, it is taken to be the difference in the count of the first and second modes of the feature movement histogram. As such, the confidence measure is higher when the distribution of feature movements is more uni-modal, and lower when the distribution is more multi-modal (indicating difficulty in resolving independently moving objects from the stationary background). Overall, the choice to consider three previous frames in the odometry calculation is designed to make the system robust to single frame errors at reasonable computational cost.

The final aspect of the VO module to consider is collision detection. On the soccer field, robots are frequently bumped or impeded by other robots and obstacles that are not visible (for example, when the robot catches its arm on a goal post or robot to its side). Using visual heading odometry, these scenarios no longer lead to mislocalisation, but if not detected they will often lead to a fall. To detect collisions, visual heading odometry is differenced with walk-

engine heading odometry. When an exponential moving average of this quantity breaches certain positive and negative bounds, it indicates that the robot is slipping with a rotation to the left or right respectively. At this point it has been found that reducing the stiffness of the robots arms can avoid a significant number of falls. Figure 22 provides a comparison of the heading odometry from vision and the walk-engine during a left hand turn, illustrating the divergence of the data during a collision.

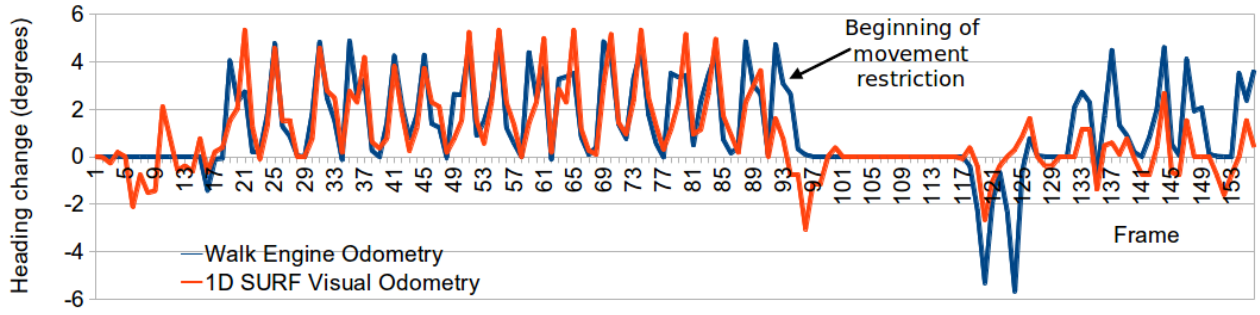

Fig. 22. Illustration of heading odometry from both the walk-engine and the visual odometry (VO) module during a 70 degrees per second left-hand turn on the Nao robot. Each spike in the graph represents a step. From frame 90 onwards, the movement of the robot was intentionally restricted. The walk-engine initially paused, before resuming an attempted left-hand turn with the feet slipping. The visual odometry indicates that these steps result in no net heading change, although the walk-engine odometry is indicative of a turn.

5.3 Results

In order to evaluate the performance and robustness of the VO module in comparison to naive walk-engine odometry, a quantitative benchmark test is required. In this paper the University of Michigan Benchmark test (UMBmark) is used [7]. In addition, tests include obstacle collisions that disrupt the natural motion of the Nao, and repeated observations of other moving objects at close range.

By way of background, the UMBmark is a procedure for quantifying the odometric accuracy of a mobile robot. In the test, the robot is pre-programmed to move in a bi-directional square path, in both the clockwise and anti-clockwise directions, and the accuracy of the return position is assessed. Although the test was designed for assessing wheel odometry error in differential-drive robots, several results of the paper are also relevant for bipedal robots. In particular, the paper illustrates that a uni-directional square path test is unsuitable for evaluating robot odometric accuracy due to the possibility of compensating turn and forward motion errors. Using the bi-directional square path test, however, these errors are revealed when the robot is run in the opposite direction.

To assess the performance of the VO module on the Nao, the UMBmark square path procedure was repeated five times in the clockwise direction, and five times in the counter-clockwise direction. In recognition of the greater inaccuracy of bipedal robots compared to wheeled robots, the side lengths of the square path were reduced from 4 m to 2 m. Prior to this test, the walk-engine odometry was calibrated to provide reasonable performance using the same settings across five robots, but it was not calibrated to suit this particular robot. The test was conducted on the SPL field, and the path of the robot was controlled autonomously using the full rUNSWift localisation system and a 'patrol' behaviour. In each test the robot did actually walk a near-square path at full speed, turn on the spot at each corner, and return to the starting position. The position of the robot calculated using only walk-engine odometry was then compared with the position of the robot using walk-engine odometry augmented with visual odometry (the VO module).

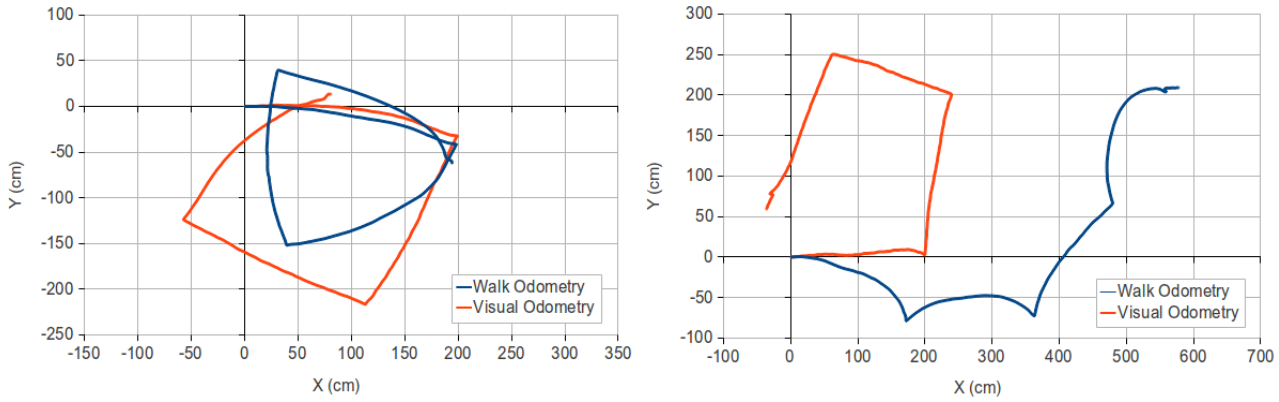

Fig. 23. Odometry tracks for one trial of an uncalibrated Nao robot walking in a 2 m x 2 m square path in both clockwise (left) and counter-clockwise (right) directions.

The odometry tracks for the first trial in each direction are shown in Figure 23, illustrating a substantial deviation from the true square path in the case of the walk-engine generated odometry, and a much smaller deviation for the visual odometry. It is clear from these diagrams that the robot used for the test has a systematic left turn bias. In order to walk around the square field, continuous right turn corrections was required, which can be observed in the paths generated by odometry dead reckoning. The VO module has compensated for a significant proportion, but not all, of this systematic bias.

Figure 24 illustrates the final positioning error using both odometry methods at the end of five trials in each direction. Using the centre of gravity approach outlined in [7], the walk-engine

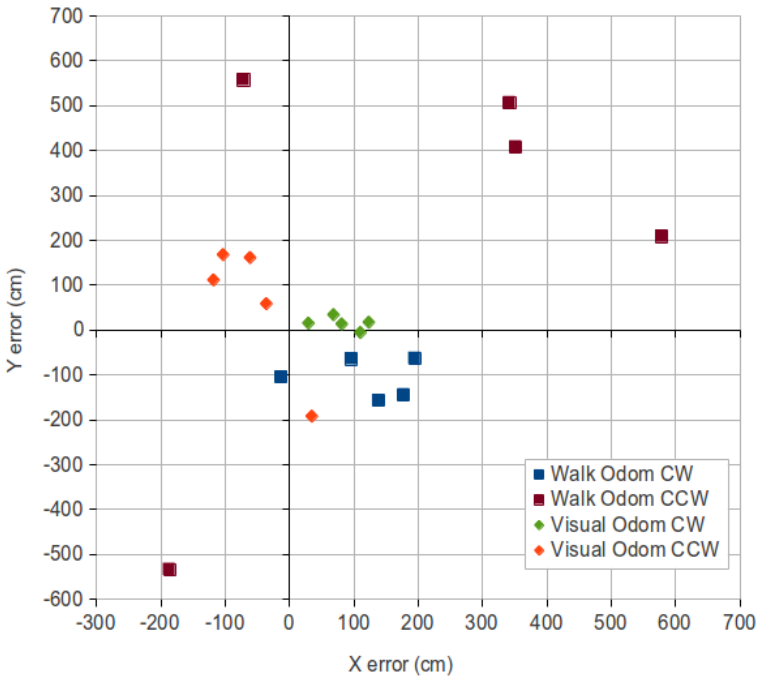

Fig. 24. UMBmark results for an uncalibrated Nao robot walking in a 2 m x 2 m square path in both clockwise (CW) and counter-clockwise (CCW) directions. Results indicate a significant increase in accuracy using visual odometry relative to naive walk-engine generated odometry.

odometry has a measure of odometric accuracy for systematic errors of 306 cm, versus the visual odometry approach with an accuracy of 83.8 cm, a reduction of 73%. These results suggest that the VO module can compensate for a significant proportion of the systematic odometry error of an uncalibrated robot. Furthermore, the standard deviation for the walk-engine return positions is 550 cm, relative to 160 cm for the visual odometry return positions. This indicates that the VO module also compensates for non-systematic odometry errors.

In the next test, the robot performed five trials of a simple out-and-back manoeuvre along a 2 m long straight line. However, on both legs of the path, a block of wood was placed in front of one shoulder of the robot to cause a collision, as illustrated in Figure 26. Figure 25 Left illustrates an odometry track for a trial during which the robot experienced a gentle collisions on the way out, and a more significant collision on the way back to the starting position that resulted in an uncommanded turn. Over five trials, the walk-engine odometry had a measure of odometric accuracy of 140 cm with standard deviation of 102 cm, versus the visual odometry approach with an accuracy of 43 cm and standard deviation of 53 cm.

The final test of the VO module was designed to illustrate the performance of the system when multiple moving objects are visible. For this test, the Nao was programmed to maintain position in the centre of the SPL field while walking on the spot, while a number of people

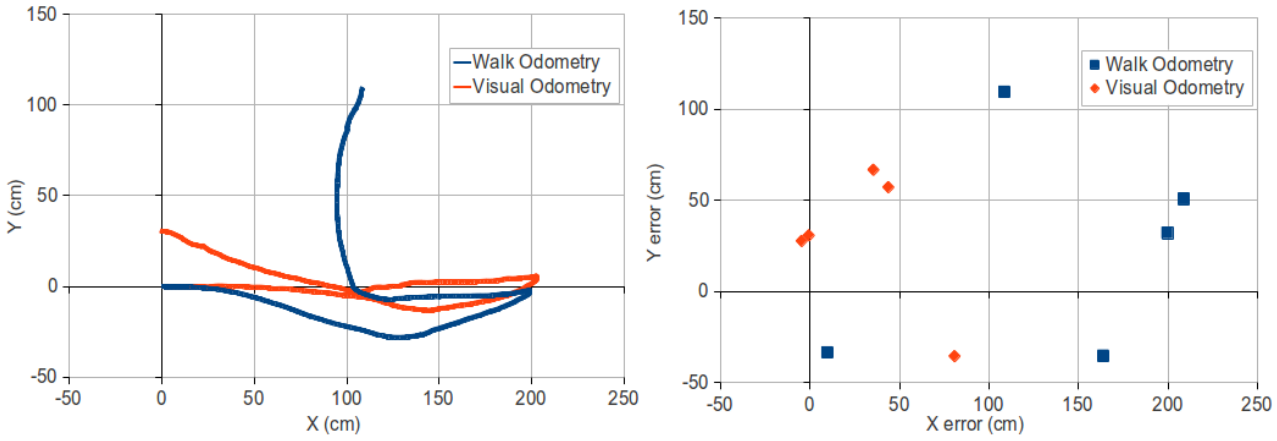

Fig. 25. UMBmark results for an calibrated Nao robot walking out and back with collisions on a 2 m path. Left: Odometry track from a single trial. Right: Final positioning error over five trials. Results again indicate a significant increase in accuracy using visual odometry relative to naive walk-engine generated odometry.

wearing blue jeans (simulating RoboCup SPL referees) were instructed to walk in front of the Nao at a range of around 50 cm. The first five people walked in the same direction, from right to left. Figure 26 illustrates that at this range the referees jeans will typically fill approximately half of the robot's horizontal field of view. The robot's visual odometry was monitored to see if the movement of the objects in front of the camera could trigger a false positive heading change. As illustrated in Figure 27, during this process there is no apparent drift in the robot's visual heading odometry in over one thousand frames (approximately 33 seconds). The oscillating pattern of the heading is attributed to the robot's constant adjustments to maintain position in the centre of the field. During these experiments, the mean execution time of the VO module on the Nao v4 (excluding the execution time required to extract 1D SURF features from the camera image) was 4.5 ms.

5.4 Evaluation

This chapter has presented a fast and unique method for calculating visual heading odometry using the same 1D SURF features used by the natural landmark localisation system. Using this technique, the VO module is able to reduce the odometric uncertainty of an uncalibrated bipedal robot by approximately 73%. The method is also able to detect collisions with unseen objects, and is robust to the presence of moving objects in the environment. As such, the VO module proved to be a significant aid to localisation during games, particularly in areas of the field where few features can be seen. The system was even demonstrated to improve localisation while the robot was being carried upright by the referee.

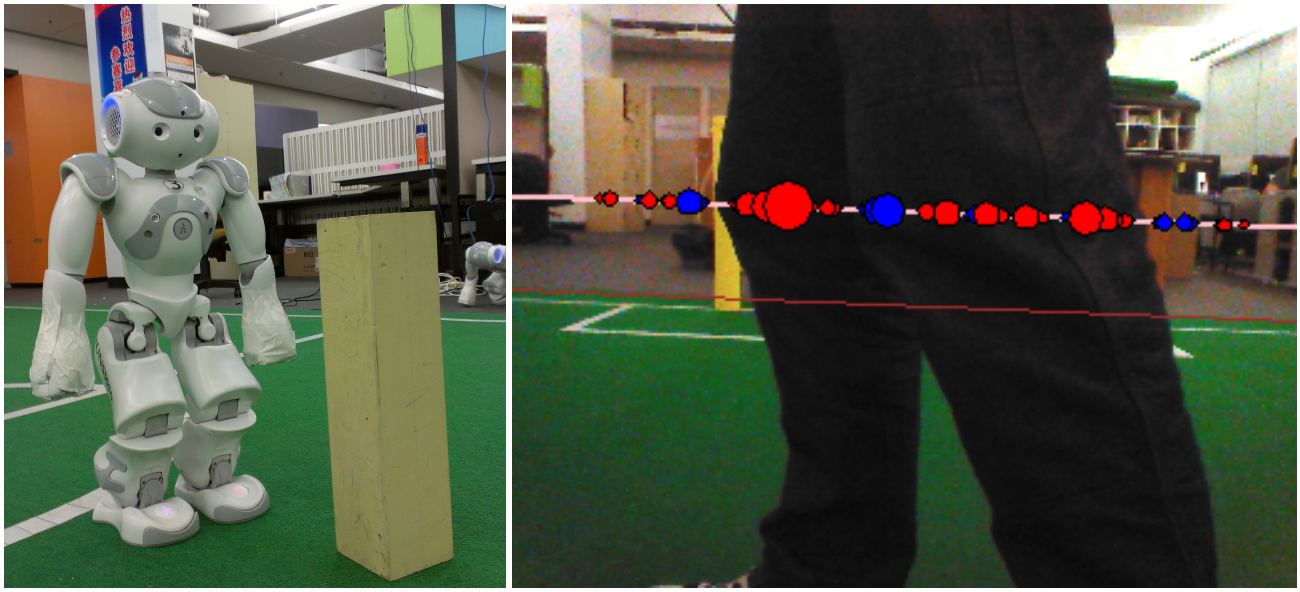

Fig. 26. Left: Block of wood used to cause collisions during testing. Right: Moving object tests as they appear to the robot.

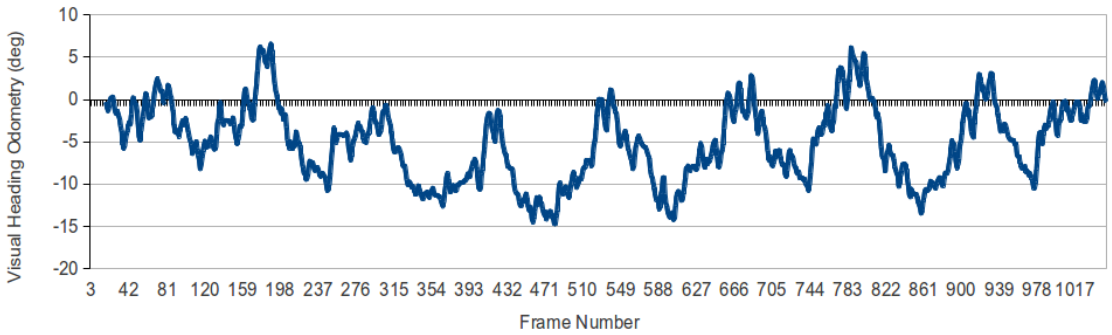

Fig. 27. Visual heading odometry with the Nao walking on the spot during the moving object test. There is no evidence of false positive heading changes or apparent heading drift during this 33 second period.

6 Robot Detection using Vision and Sonar

This chapter describes how a robot detection and tracking system for the Nao was implemented using both vision and sonar. The purpose of the system is to accurately track the positions of multiple other robots, while also determining their team from the colour of the waistband worn. This information is essential for behaviours such as obstacle avoidance, kicking around opposition players, passing, and utilising team formations in the absence of wireless communication.

6.1 Background

The RoboCup robot detection problem is an instance of a larger class of problems dealing with object detection in video. Unlike these general problems, however, many aspects of the RoboCup SPL environment are specified in advance. For example, it is known that robots of interest will always be found on a green field. These unique aspects of the RoboCup SPL environment, combined with the computational limitations of the robots, have tended to lead prior researchers towards quite specific solutions that rely heavily on this domain knowledge.

Fig. 28. Illustration of the concept of robot detection using field edge indents. No indent is shown around the goal post. Sourced from [21].

The basic premise of previous work in this area by the rUNSWift team [21] was that robots can be detected in a camera frame by examining indents in the field edge, as shown in Figure 28. The process for generating these field edge indents involves using colour calibration to detect the edges of the field, and region-building from non-green pixels detected below the field edge. Once field indents have been generated, [21] collected as much information as possible about

the region, including both camera frame and real-world dimensions, and colour information. These inputs were then fed into a C4.5 decision tree to attempt to classify the region as robot or non-robot. Although this approach showed some promising preliminary results, it was not considered to be ready for use in competition. As a result, in the 2011 RoboCup competition rUNSWift used a naive approach to robot avoidance based on reacting to the nearest sonar range return.

Another region-building approach to robot detection in RoboCup was outlined by [1]. In this approach, more importance was attached to detecting the presence of the robot's coloured waistband first, then scanning outwards from this point to verify the existence of the robot. The authors also outlined an enhanced Kalman filter based algorithm for tracking multiple robots. The algorithm increases the process noise for tracked robots that should be visible in the current frame but aren't, to ensure that false positives are quickly removed from the filter.

6.2 Method

The design of 2012 robot detection system utilises both the field indent detection technique (but not the decision tree) previously developed by rUNSWift [21], and the Kalman filter tracking algorithm outlined by [1]. The main contribution of this paper to robot detection is in the development of a more advanced sonar filter with higher directional resolution, and the incorporation of both vision and sonar information into the robot detection system.

The motivation for this development comes from the complementary nature of visual and sonar data. Experience suggests that vision based methods struggle to accurately determine the range of a robot observation. This is in part due to the difficultly in precisely determining the position of the detected robot's feet in the frame, for example, the feet could be cropped from the image, or the detected robot could be standing on a white line which is difficult to distinguish from a white foot. It is also due to the inaccuracy in transforming an image pixel location to real world coordinates, given noise in the robot's kinematic chain and other sources of error. In contrast, sonar observations provide very accurate range data, but cannot distinguish robots from other obstacles or determine an accurate direction to the robot. The best approach should merge both sets of observations, while extracting as much sonar directional resolution as possible.

To make sonar observations truly useful, it is necessary to filter the entire set of sonar range returns. The previous rUNSWift sonar filter only tracked the range to the first detected object [21], and so was not useful for tracking a field containing multiple robots and non-robot obstacles such as goal posts and referees. To overcome this limitation, a new sonar filter was devised that records a rolling window of sonar range returns in each available direction. It

then proceeds to cluster observations in a bottom up fashion using hierarchical agglomerative clustering, continuing to merge clusters of observations provided the difference in range is within a pre-determined cut-off of 5 cm. At the end of this process, any clusters above a certain cut-off size are considered to be reliable observations, with the range of the observation given by the mean of observations in the cluster. The advantage of this approach is that unlike histogram based approaches using bins, it doesn't reduce the accuracy of observations, yet it is still able to remove noise.

In order to merge sonar observations with vision, it is also important to extract as much directional resolution from the sonar sensors as possible. The Nao robot comes equipped with two sonar transmitter and two sonar receivers, with each sensor pair mounted on the left and right of the chest, as illustrated in Figure 29. As such, the sonar system is notionally capable of detecting obstacles on either the left or the right of the body, with each receiver reacting to a 60 degree cone at ranges of 25 cm to 255 cm[3].

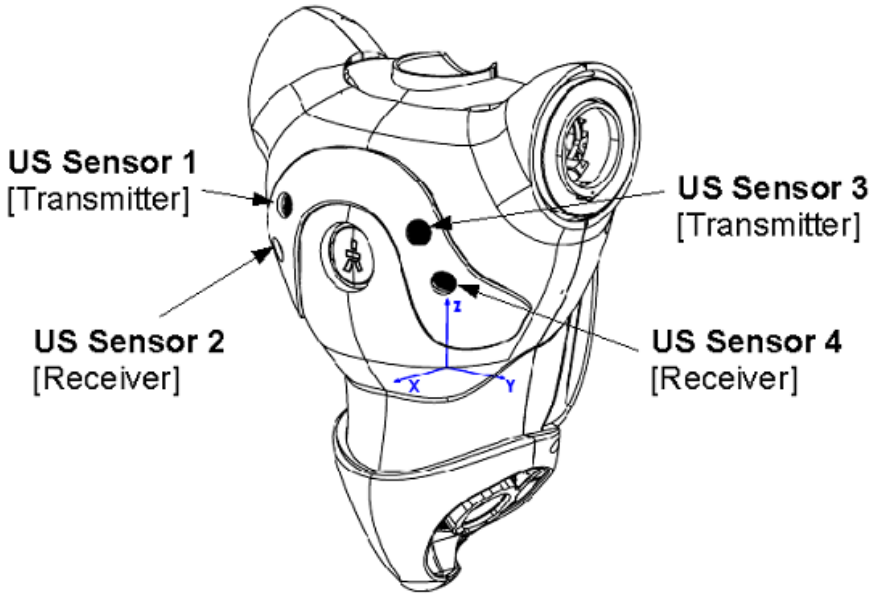

Fig. 29. Illustration of the location of sonar transmitters and receivers on the Aldebaran Nao robot. Source: http://www.aldebaran-robotics.com/documentation/nao/hardware/sonar.html.

To improve the resolution of the sonar sensors, a novel hardware control algorithm was developed to provide an additional central sonar detection cone. This allows the sonar sensors to reliably distinguish individual obstacles on the left, middle, and right of the robot body. The method works as follows: To detect obstacles on the left or right hand side of the body, only

[3] http://www.aldebaran-robotics.com/en/Discover-NAO/nao-datasheet-h25.html

the transmitter and receiver on that same side are switched on, while the opposite transmitter and receiver remain switched off. To detect obstacles directly in front of the robot, an alternate transmitter and receiver pair is used, for example the left transmitter and right receiver, or vice versa, and the remaining transmitter and receiver are switched off. The effect of this mode is that only obstacles within an approximately 40 degree cone in the centre of the robot will be detected, due to the geometry of the sonar sensor locations in the robot chest, as illustrated in Figure 30. In effect, using this hardware control algorithm is similar to having an additional pair of narrow cone sonar sensors mounted directly in the middle of the robot. It is important to realise that just checking in software for sonar returns on both sides of the robot is not equivalent to this algorithm. This is because sonar returns at similar ranges on both the left and right sensors could be generated by two distinct obstacles, or by a single central obstacle. The hardware control approach doesn't suffer from this ambiguity.

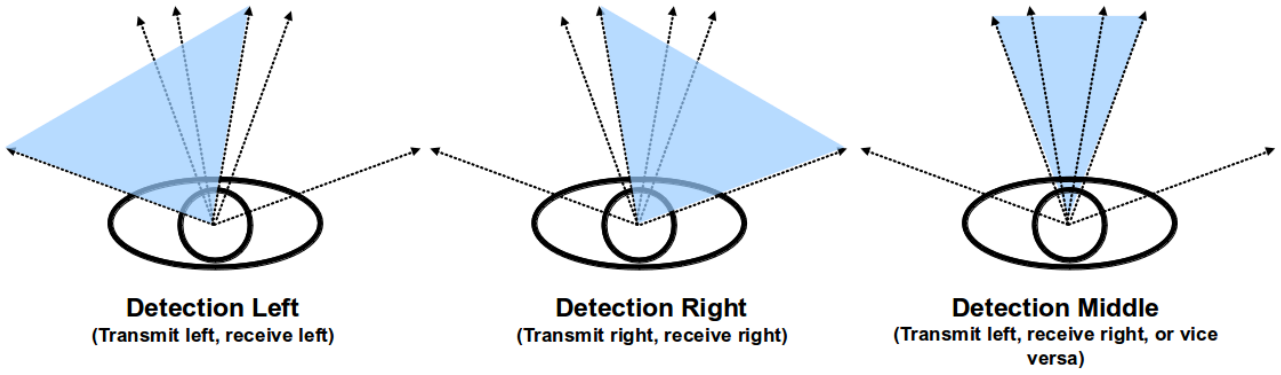

Fig. 30. Approximate location of the three sonar detection cones using the described hardware control algorithm.

The overall robot detection algorithm is outlined in Algorithm 2. In effect, it applies a series of sanity checks to field indentations, in order to determine if they represent a robot or not. Then, the base of the detected robot is used to project the robot's position into real world coordinates. However, if the base of the detected robot is at or near the bottom of the image, suggesting part of the robot might be cropped, then the waistband is used to project the robot's position into real world coordinates (assuming the robot is standing). At this point, filtered sonar returns are used to try to improve the robot range measurement estimated from the image. If an observation is made of a robot where the waistband cannot be detected, and the range doesn't match a sonar return, then the observation is discarded as it is very likely to be a false positive. Observations are then passed to the filter as outlined by [1].

Algorithm 2: Robot detection algorithm.

Input: I: Set of field edge indents, each represented by a colour classified bounding box, S: Set of filtered sonar range observations
Output: R: Set of robot observations

```
clear R;
foreach i in I do
    Shrink the bottom and sides of i if they are mostly green;
    if i is too wide, too narrow, or not tall enough then
        continue;
    if i contains a cluster of blue or pink pixels then
        SetWaistBand(i);
    if base of i is at or near base of camera frame and HasWaistBand(i) then
        i_world = WaistBand(i) projected into robot relative world coordinates, assuming
        robot is standing;
    else
        i_world = base of i projected into robot relative world coordinates;
    if i_world is too narrow to be a robot then
        continue;
    if Range(i_world) is close to a sonar range s in S then
        Range(i_world) = s;
    if TeamColour(i) is not set, and Range(i_world) is not sonar verified then
        continue;
    R.add(i);
```

6.3 Results

Penalty Shoot-Out Test In order to robustly evaluate a robot detection and tracking system, there are a huge number of scenarios that need to be considered, involving different configurations of opposition and friendly robots, and various amounts of observer and scene motion. However, given the broad scope of this thesis, this section does not intend to be an exhaustive evaluation, but merely to provide some indication of the typical performance of the system.

With this aim in mind, a test was conducted based on a penalty shoot-out scenario. The penalty shoot-out is a contest between one attacking robot (the striker) and one goalie. The goalie begins from inside the goal box, and the striker starts from the half way line, with the ball placed on the penalty spot (120 cm in front of the striker) [25]. This is an important game scenario for the robot detection system, for if the striker can accurately detect the position of the goalie, the kick can be aimed towards an open area of the goal. Similar scenarios arise in normal game play when an attacking robot shoots at the goal.

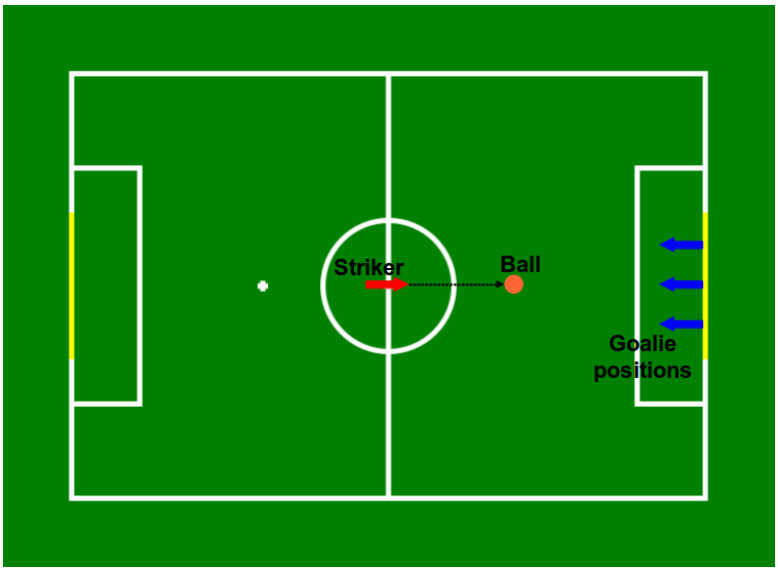

Fig. 31. Illustration of robot and ball positions at the beginning of the penalty shoot-out test. By placing the stationary goalie in different positions, the robot detection capability of the attacking robot could be evaluated.

To evaluate the robot detection system, three penalty shoot outs were undertaken. In each case a stationary, standing goalie was placed inside the goal area, either in the middle of the goal, 32 cm towards the left, or 32 cm towards the right of the goal, as indicated in Figure 31. The goalie was not wearing any coloured waistband during this test. The robot detection system was evaluated based on the range and accuracy with which the striker was able to detect the position of the goalie, while walking forward 120 cm to make contact with the ball. It is important to note that during this test, the accuracy of the goalie location estimate is dependent on the ability of the striker to detect the goalie, but also on the strikers ability to localise itself. This is because all observations of other robots are made in robot relative coordinates before being mapped to a global field location. As a result, this test is not a pure evaluation of the robot detection system. However, the movement of the striker improves the relevance of the evaluation to game situations.

Table 5. Performance of goalie detection during a penalty shoot-out test. Results indicate the range and error of the first observation of the goalie, and the range and error of the final goalie position estimate at the time of kicking the ball.

Goalie Position	First range (cm)	First error (cm)	Final range (cm)	Final error (cm)	Number of frames
Goal Middle	234	42	178	2.8	25
Goal Left	197	47	182	3.6	26
Goal Right	221	58	181	6.0	46

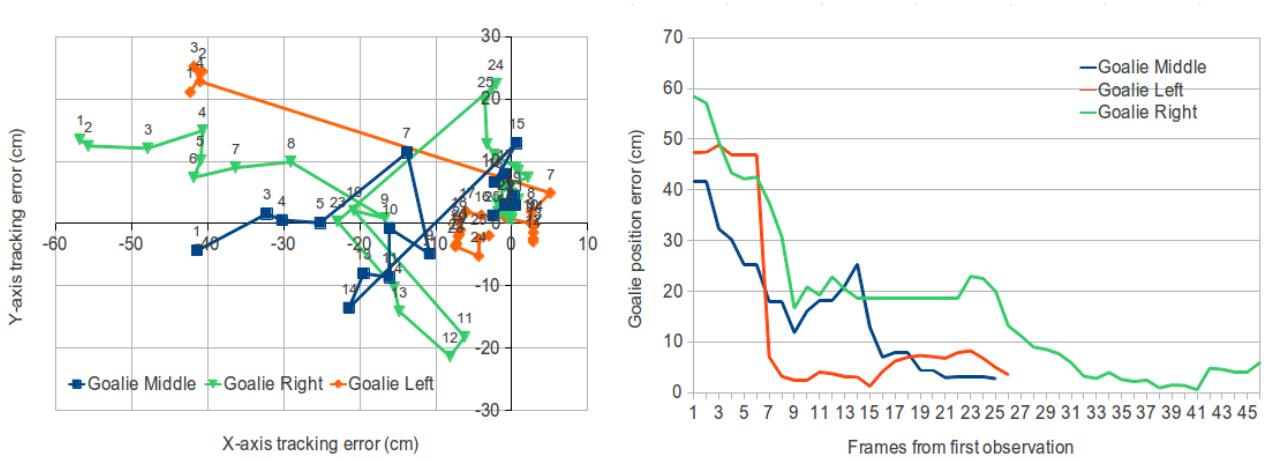

Fig. 32. Left: X and Y components of the goalie position error from first observation until the striker stops to kick the ball. Right: Reduction in goalie positioning error over the same time period.

As indicated in Table 5, the range at which the first observation of the goalie was made varied between 197 cm and 234 cm, with an initial error in the goalie position estimate of between 41 cm and 58 cm. While the initial goalie position error is quite large, as indicated in Figure 32 Left, this is mostly a result of underestimating the range to the goalie (negative X-axis error in the field coordinate system used here). Furthermore, as shown in Figure 32 Right, this error reduces quickly as more observations (including sonar range observations) are made. By the time the striker has reached the ball and stopped (a period of 25 to 46 camera frames), the goalie position error has reduced to between 28 mm and 60 mm. As previously noted, this error includes error in the strikers own localisation. Once the striker has reached this point, the range to the goalie is still approximately 180 cm. These results indicate that during a penalty shoot-out scenario the striker can accurately locate the goalie before kicking.

Match Performance The pool rounds of the 2012 RoboCup SPL competition were characterised by periods of near total wireless failure, during which robots were unable to communicate with each other or the game controller. Many teams, including rUNSWift, rely on wireless communication for assigning dynamic robot roles such as striker, defender, and supporter, based on location and proximity to the ball. However, in the absence of wireless connectivity, robots were unable to communicate their locations, and these role-switching behaviours collapsed. In the early matches this caused many instances of robots tackling their own teammates with the ball. As well as causing obvious negative interference with one's teammate, this behaviour frequently results in a penalty to the attacking robot under the player pushing rules [25].

To mitigate these problems, between the last pool match and the quarter final against the UPennalizers, rUNSWift implemented a new role switching behaviour. During wireless failures, the new behaviour used the robot detection system rather than relying on teammates to report their own positions using wireless. A robot that observed a team mate closer to the ball then itself would then role-switch to defender or supporter behaviour, rather than continuing to attack the ball.

The implementation of this behaviour in between matches provides a good opportunity to benchmark the performance of the robot detection system. For comparison, the pool match against B-Human and the quarter final against UPennalizers are chosen. In both matches, wireless availability was very poor, and rUNSWift played with a full team. rUNSWift's other second round pool fixtures are not suitable for use in the comparison, because much of these games were played with only one field player and a goalie to mitigate the teammate tackling problem.

Fig. 33. rUNSWift vs. B-Human (3:4), RoboCup SPL 2012 Pool I. This photo illustrates the high incidence of robot's crowding the ball and tackling their own teammates during periods of wireless failure (rUNSWift in pink waist bands). In later games rUNSWift was able to mitigate this problem using the robot detection system.

During the game against B-Human (result 3:4 to B-Human), there were 10 instances of a rUNSWift robot tackling its teammate with the ball. Of these instances, 6 resulted in a player pushing penalty, and by the end of the match virtually the entire rUNSWift team had been permanently removed for repeated penalties. The winning B-Human goal was scored with no rUNSWift robots on the field to oppose. Figure 33 is indicative of the ball-crowding behaviour that occurred frequently in this match. After implementing role-switching using the robot de-

tection system for wireless failures, the instances of teammate tackling in the UPennalizers match (result 8:0 to rUNSWift) were reduced to 7. Moreover, because rUNSWift wasn't constantly crowding the ball, only 2 of these incidents resulted in a player pushing penalty. This indicates that the robot detection system was effective in detecting both the location and the team colour of other robots during competition matches, and that this information could be used by the robots to make important strategic decisions.

6.4 Evaluation

This chapter has presented a robot detection system that builds on previous work by introducing a new sonar filter, and a novel hardware control scheme for improving the directional resolution of sonar measurements. Using these techniques, a combined robot detection and tracking system using both vision and sonar information was developed. Evidence suggests that in a penalty-shoot out, this system allows the striker to localise a stationary goalie to within 28 mm to 60 mm, and therefore kick towards the open area of the goal. Similarly, the system proved its worth in competition by allowing rUNSWift to develop coordinated team behaviours that remained operational during total wireless failures.

Notwithstanding this success, developing a robot detection and tracking system that works under all match scenarios is extremely difficult. Anecdotal evidence from competition matches indicates that robots were not always aware of other robots near them. At times this occurs because the robot is not looking in the right direction, indicating that it may be necessary for rUNSWift to move towards an active vision system and away from naive head-scanning techniques.

7 Conclusion and Directions for Future Development

This paper has presented a number of new methods for improving perception in RoboCup SPL, including a unified sensor model inspired by ICP, an optimised 1D SURF feature, a bag of words based natural landmark localisation system, a visual odometry system, and a combined sonar and vision robot detection system. Results have been presented in each chapter to illustrate the performance of each of these techniques, some of which are likely to have potential applications outside of RoboCup.

The overarching motivation for this thesis was to help bridge the gap towards a system where each robot is precisely located and aware of other robots at all times during SPL games, without the use of non-strategic localisation behaviours. With the work outlined in this thesis, along with the development of a new localisation filter, rUNSWift is close to realising this aim. While there were still occasions in games where it was evident that a robot was mis-localised, anecdotal evidence from watching many games suggests that for the vast majority of game time, each robot was well-localised and aware of other robots. Furthermore, this was achieved without any of the localisation behaviour 'crutches' that were previously used, such as localisation scans before kicking the ball. In general, rUNSWift was programmed to watch the ball whenever it was visible. From a behavioural perspective, localisation took place in the background, making our robots faster and more reactive then before. Aided by this and the many other innovations developed by the team, rUNSWift scored more goals than any other SPL team in RoboCup 2012.

There are a number of potential avenues for further development of the natural landmark localisation system, in order to make it more robust in the face of changing landmarks due to crowd movement. One obvious approach is to investigate simultaneous localisation and mapping (SLAM) techniques, rather than simply mapping the field environment at the start of the half and expecting it to remain fixed. Another avenue that is yet to be exploited is cooperative localisation, in the sense of robots sharing a single natural landmark map rather than using individual maps.

With regard to the localisation system more generally, this year we found it beneficial to simplify the localisation filter, and focus greater development effort on improving the sensor model and extracting more information from raw images. It seems likely that the easiest way to improve the localisation system further is by continuing in this direction, and focusing on image processing, rather than trying to compensate for poor information with complex filters. Many of the rUNSWift image processing techniques are still in their infancy. For example, the goal post detection module is incapable of detecting the goal cross-bar, and so cannot tell left-side posts from right-side posts unless both are visible. As a result, left and right goal posts are

aliased when they need not be. Furthermore, field-lines can only be detected at close range since rUNSWift has yet to use the full resolution of the robot's cameras. This is partly because some components of the code base, such as the field line detection module, are wasteful of CPU resources.

With regard to robot detection, anecdotal evidence suggests some collisions and player pushing penalties occur because the robot is not looking in the right direction. An active vision system could offer a material improvement over the naive head-scanning techniques being employed at the moment. Furthermore, the existing system does not share robot observations between team members. Implementing a team filter for opposition robots could provide a substantial improvement. With these opportunities and many others unrelated to perception, rUNSWift continues to offer an exciting development opportunity for future teams.

References

1. T. Röfer A. Fabisch, T. Laue. Robot recognition and modeling in the robocup standard platform league. *Proceedings of the Fourth Workshop on Humanoid Soccer Robots in conjunction with the 2010 IEEE-RAS International Conference on Humanoid Robots*, 2010.
2. R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
3. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, 1999.
4. H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.
5. H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer Vision–ECCV 2006*, pages 404–417, 2006.
6. Frank Moosmann Bernd Kitt and Christoph Stiller. Moving on to dynamic environments: Visual odometry using feature classification. *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2010.
7. Johann Borenstein and Liqiang Feng. Umbmark: A benchmark test for measuring odometry errors in mobile robots. *SPIE Conference on Mobile Robots*, 1995.
8. A. Briggs, C. Detweiler, P. Mullen, and D. Scharstein. Scale-space features in 1d omnidirectional images. In *Omnivis 2004, the Fifth Workshop on Omnidirectional Vision, Prague, Czech Republic*, pages 115–126, 2004.
9. A. Briggs, Y. Li, D. Scharstein, and M. Wilder. Robot navigation using 1d panoramic images. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 2679–2685. IEEE, 2006.
10. A.J. Briggs, C. Detweiler, Y. Li, P.C. Mullen, and D. Scharstein. Matching scale-space features in 1d panoramas. *Computer vision and image understanding*, 103(3):184–195, 2006.
11. Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Proceedings of the IEEE International Conference on Robotics and Automation*, 3:2724–2729, 1991.
12. David Claridge. Multi-hypothesis localisation for the nao humanoid robot in robocup spl. Technical report, School of Computer Science and Engineering, University of New South Wales, 2011.
13. Mark Cummins and Paul Newman. Appearance-only slam at large scale with fab-map 2.0. *Int. J. Rob. Res.*, 30(9):1100–1123, August 2011.
14. A. A. Efros D. Hoiem and M. Hebert. Recovering surface layout from an image. *International Journal of Computer Vision*, 75, 2007.
15. O. Naroditsky D. Nister and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23, 2006.
16. F. Fraundorfer D. Scaramuzza and R. Siegwart. Real-time monocular visual odometry for on-road vehicles with 1-point ransac. *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2009.
17. M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.
18. Sean Harris. Efficient feature detection using ransac. Technical report, School of Computer Science and Engineering, University of New South Wales, 2011.
19. R. S. Hartenberg and J. Denavit. A kinematic notation for lower pair mechanisms based on matrices. *Journal of Applied Mechanics*, 77:215–221, 1955.
20. L. Juan and O. Gwun. A comparison of sift, pca-sift and surf. *International Journal of Image Processing (IJIP)*, 3(4):143–152, 2009.
21. Jimmy Kurniawan. Multi-modal machine-learned robot detection for robocup spl. Technical report, School of Computer Science and Engineering, University of New South Wales, 2011.
22. D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

23. Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.
24. Adrian Ratter, Bernhard Hengst, Brad Hall, Brock White, Benjamin Vance, David Claridge, Hung Nguyen, Jayen Ashar, Stuart Robinson, and Yanjin Zhu. rUNSWift team report 2010 robocup standard platform league. Technical report, School of Computer Science and Engineering, University of New South Wales, 2010.
25. RoboCup SPL Technical Committee. Robocup standard platform league (nao) rule book. Website: http://www.tzi.de/spl/pub/Website/Downloads/Rules2012.pdf, 2012.
26. Thomas Röfer, Tim Laue, Judith Müller, Alexander Fabisch, Fynn Feldpausch, Katharina Gillmann, Colin Graf, Thijs Jeffry de Haas, Alexander Härtl, Arne Humann, Daniel Honsel, Philipp Kastner, Tobias Kastner, Carsten Könemann, Benjamin Markowsky, Ole Jan Lars Riemann, and Felix Wenk. B-human team report and code release 2011. http://www.b-human.de/publications/, 2011.
27. Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *3rd International Conference on 3D Digital Imaging and Modeling (3DIM 2001), 28 May - 1 June 2001, Quebec City, Canada*, pages 145–152. IEEE Computer Society, 2001.
28. D. Scaramuzza and F. Fraundorfer. Visual odometry: Part i - the first 30 years and fundamentals. *IEEE Robotics and Automation Magazine*, 18, 2011.
29. Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. pages 1470–1477, Nice, France, 2003.
30. Wonpil Yu Sunglok Choi, Ji Hoon Joung and Jae-Il Cho. What does ground tell us? monocular visual odometry under planar motion constraint. *11th International Conference on Control, Automation and Systems (ICCAS)*, pages 1480–1485, 2011.
31. Stefan Tasse, Matthias Hofmann, and Oliver Urbann. On Sensor Model Design Choices for Humanoid Robot Localization. In *RoboCup International Symposium 2012*, Mexico, June 2012.
32. S Thrun, W Burgard, and D Fox. *Probabilistic Robotics*. MIT Press, 2005.
33. Douglas Turnbull and Charles Elkan. Fast recognition of musical genres using rbf networks. *IEEE Transactions on Knowledge and Data Engineering*, 17:580–584, 2005.
34. Peter N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1993.