The Thesis Committee for Yuchen He
Certifies that this is the approved version of the following thesis:

LOCALIZATION USING NATURAL LANDMARKS OFF-FIELD FOR ROBOT SOCCER

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Professor Peter Stone (Chair)

Professor Dana H. Ballard

LOCALIZATION USING NATURAL LANDMARKS OFF-FIELD FOR ROBOT SOCCER

by

Yuchen He, B.E.

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Computer Science

The University of Texas at Austin

December 2013

Acknowledgements

I would like to give thanks to all members in the UT Austin Villa [1][2] team for their help and support. Great thanks to Samuel Barrett for his help in understanding and integrating in the team software infrastructures, to Todd Hester, Jacob Menashe and Katie Genter for developing localization and visual recognition models and providing great suggestions. Finally, I would like to express the deepest appreciation to my advisor Professor Peter Stone for his guidance, supports and encouragement on this research project. Without his supervision and constant help this thesis would not have been possible. And I would like to give great thanks to Professor Dana Ballard in regard to his helpful reviews and advice, and an excitement in regard to teaching and research.

Abstract

LOCALIZATION USING NATURAL LANDMARKS OFF-FIELD, FOR ROBOT SOCCER

Yuchen He, M.S. Comp.Sc.

The University of Texas at Austin, 2013

Supervisor: Professor Peter Stone

Localization is an important problem that must be resolved in order for a robot to make an estimation of its location based on observation and odometry updates. Relying on artificial landmarks such as the lines, circles, and goalposts in the robot soccer domain, current robot localization requires prior knowledge and suffers from uncertainty problems due to partial observation, and thus is less generalizable compared to human beings, who refer to their surroundings for complimentary information. To improve the certainty of the localization model, we propose a framework that recognizes orientation by actively using natural landmarks from the off-field surroundings, extracting these visual features from raw images. Our approach involves identifying visual features and natural landmarks, training with localization information to understand the surroundings, and prediction based on matching of features. This approach can increase the precision of robot orientation and improve localization accuracy by eliminating uncertain hypotheses, and in addition, it is also a general approach that can be extended and applied to other localization problems as well.

Keyword: *Localization; Robocup; Natural Landmarks; Feature Extraction, Matching; Humanoid Robot; SIFT, SURF, Bag Of Word*

Table of Contents

List of Tables

List of Figures

CHAPTER 1 INTRODUCTION

Robot localization is the process of making estimations of pose and orientation of the robot based on observations from surroundings and self-state modeling updates. In the robot soccer domain, one widely used observation is raw images from cameras in which manually defined artificial landmarks, such as lines, circles, crosses and goalposts with distinct colors, are detected by visual recognition. The inherent dependency on prior knowledge makes the localization less extendable to more general cases where no artificial landmarks have been previously defined. Using only limited on-field information, one observation could correspond to multiple different localization hypotheses, thereby creating many uncertainties in the estimation. The hypothesis we are going to verify is whether a robot could be trained to understand its surroundings with more perception from off-field, to then extract interesting features from each direction and, finally, to match the corresponding features in order to predict its orientation in real time, analogously to how human beings recognize direction. This process helps the robot improve localization accuracy when integrated with a probabilistic localization model. This is an experimental application of a more general visual recognition approach to robot perception and it could be generalized further to more general robot localization scenarios.

This thesis presents a framework for a humanoid robot to actively recognize orientation using natural landmarks off the field, so that the robot could better predict localization when localization using only the on-field landmarks yields only uncertain estimates. Humanoid robots are used since, compared to all other kinds of robots, disorientation and kidnapping are more likely to influence their hypotheses regarding

orientation and position because they are less stable, especially while and after walking, being pushed by obstacles, and falling over. After training that incorporated localization information to understand their surrounding, the robots could generate pose/orientation predictions by matching features extracted from raw camera off-field images instead of relying only on their prior knowledge of artificial on-field landmarks such as goalposts and lines. This orientation estimation based on natural landmarks from the robots' surroundings achieved a high level of accuracy. We set up experiments on the humanoid robot Nao used in the Robocup competition to test the performance of its orientation prediction on random walking and kidnapping scenarios in order to verify the results.

The thesis is organized in the follow way: in Chapter 2, we describe the background information, state the main problem we are solving as well as related research on localization using visual features; Chapter 3 gives descriptions of the proposed framework used to make orientation estimates based on features extracted from raw camera images such as natural landmarks off the field, as well as detailed descriptions of the training, testing and feature-matching algorithms; and Chapter 4 provides accounts of the experiments and assessments of our approach. Finally, we come to our conclusion and discuss future work in Chapter 5.

CHAPTER 2 BACKGROUND

Robots and artificial intelligence are popular research topics because they can provide insights into the way agents can gain knowledge by learning their from environments. Robocup is a competition examining and testing robot intelligence by making teams of robots play soccer against each other. This is a complicated task, requiring research and competence with techniques of motion control, vision, localization, estimation and more. In this thesis, we focus on the problem of localization, and our objective is to program the robot to localize itself based on off-field natural landmarks from its surroundings, like human beings do. In this section, we present some background information and described the robot system we examined along with the most important modules our approach relies upon. We analyze the actual problem and review the major constraints of current approaches. We propose a potentially more successful approach to solving the problem of localization using surrounding natural landmarks as visual information. Finally, we consider possible applications and improvements to this approach to achieving more accurate localization using surrounding natural landmarks.

2.1 NAO

Aldebaran Nao is a humanoid robot used in Robocup SPL (Standard Platform League), which is a competition designed for advancing research in robotics and artificial intelligence. The robot Nao is equipped with arms, legs with motors to control movement, and a head with processors, cameras and sonar for perception, as shown in Figure 1 [01] below.

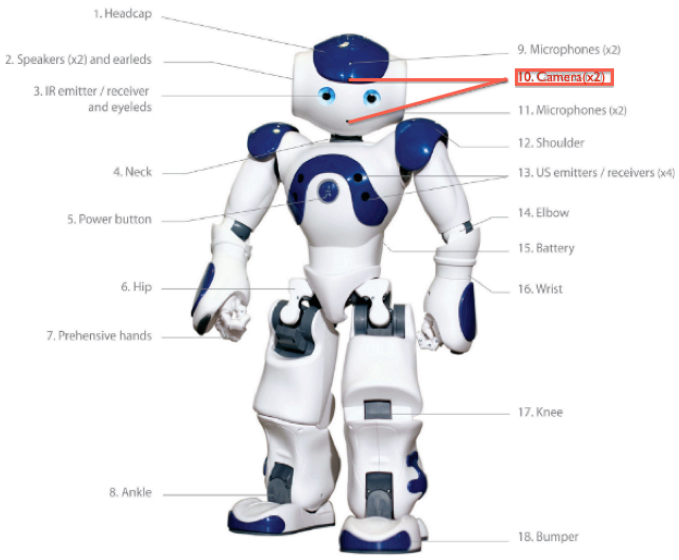


Figure 1: Aldebaran Nao Specification

The competition takes place on a green carpet field with white lines, center circles, penalty boxes and two goals whose posts are painted certain colors. Robots are programmed to recognize the artificial landmarks based on color segmentation and to autonomously play soccer as a team of five against other similarly autonomous teams of five robots. A few of the key capabilities required for robots to play soccer like human beings include motion planning, vision, localization, and strategies among others. A detailed description of how our team solved each of these problems can be found in our 2013 Team Report [2] and open source code [1]. Our team built a simulation tool to help analyze the real time performance of the approach. The simulator could replay and visualize the logs, recorded during practice and the actual games, of the robot's information, such as motion, vision, and localization. All the information can be processed using updated code compiled off-line, thus for the same set of logs we can vary

different algorithms and also debug. The scope of this thesis primarily involves improvements in vision and localization.

2.2 VISION

The vision system is composed of two 640x480 resolution 30 fps digital cameras, with 72.6°DFOV (60.9°HFOV, 47.6VFOV), as shown in Figures 2 below [20].

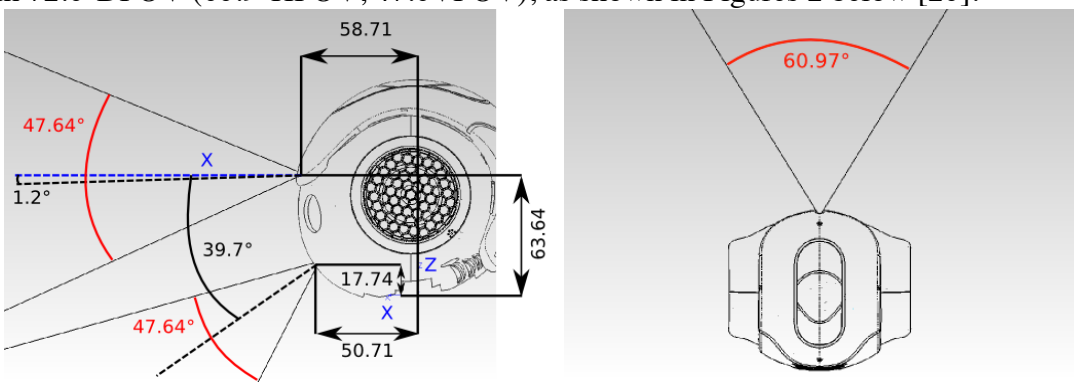


Figure 2: Nao Cameras Specification

As shown in Figure 2, the top camera is on the forehead and the bottom near the chain. In robot soccer, the top cameras are mainly used for perception of the in- and off-field images while the bottom cameras are for ball detection; together they provide streaming image frames as the main observation input. Visual processing for Robocup involves four stages: 1. Segmentation—the raw images are segmented using color tables; 2. Blob Formation—blobs are formed after scanning the segmented image horizontally and vertically; 3. Object detection—certain objects that have been predefined as lines, curves and goalposts are extracted by merging blobs; 4. Transformation—based on the pose of the robot, ground plane transformations are generated.

Since the 2012 Robocup, dual cameras are streamed simultaneously instead of alternately while still maintaining the hardware frame rate at around 30 Hz due to higher processing and bus speeds. The raw camera images are processed separately without

being merged together, using different color tables, with the bottom cameras focused more on ball detection. Though images from the top cameras are segmented, after the scanning, the off-field images were not further processed in our previous approach or in the current approaches of other teams. This is because the images are easily recognized using green and white color segmentation, and also because there are no significant artificial landmarks worth computing. This delivers high efficiency in image processing and useful observations of most of the artificial landmarks used for localization. However, the off-field images also contain useful information but take more time to process and analyze, as this involves general and complicated computer vision problems such as object recognition, feature extraction, and so on. In this experiment, we extracted "natural landmarks" under the assumption that most of the features of the surroundings remained stable during the short time period of the experiment. Then, the robot was able to match these visual features to their reoccurrence in order to help localization similarly to how artificial landmarks are used to help with localization.

2.3 LOCALIZATION

Localization is the process of estimating the pose and orientation of the robot based on observations from its surroundings and self-state modeling updates. Since sensors today provide only low feature specificity as observation input, the most effective localization algorithms currently are probabilistic modeling. Monte Carlo, such as Particle Filter, uses sampling-based density representation and can represent multi-modal probability distributions, but with low efficiency and more redundant information. As for the Kalman Filter, the state information is easier to share and maintain because of Gaussian representation. But it is less effective at handling complicated, dynamic, and ambiguous environments. EKF (Extended Kalman Filter) and UKF (Unscented Kalman

Filter) were developed to deal with non-linear models. Multi-model Kalman Filters are improvements of the Kalman Filter in regards to fast relocalization and representation of multi-modal belief distribution achieved by maintaining accurate and reasonable hypotheses using a Gaussian Mixture Model. Most of the localization approaches rely on artificial landmarks as observation input, but these do not function properly in beacon-free environments, thus could not be generalized to more complex and less specific environments.

The main goal of localization in Robocup is to detect the position of each robot itself, the ball, the goal for its team, and teammates and opponents as certainly and accurately as possible. Currently, most teams use the lines, circles, crosses, and goalposts as landmarks based on color, and anything beyond the field is ignored. The most commonly used modeling for localization is probabilistic; our team uses a multi-modal 7-state UKF [2][13], which is more computationally efficient and easier for team localization sharing and integration of ball information compared to other approaches, such as Monte Carlo. This approach can produce more reliable hypotheses of robot localization based on ambiguous landmarks and localization sharing. Information sharing between teammates will also help maintain a majority voted hypothesis among the entire team.

In 2012, in the standard platform league, the robot Nao V4 was upgraded to support streaming image frames from the top and bottom cameras simultaneously [2]; with this better processing capability, the robot receives more frequent and accurate updates through the combined raw camera input, allowing better observation of the surroundings. One of the main changes to the robot soccer field is that the goalposts on both sides are now the same shade of yellow, instead of the distinct colors yellow and blue. This has become a big challenge since robots need to differentiate between

symmetric locations on opposite ends of the field by relying on the tracking and updating of its states, especially orientation. Because the robot's odometry could be inaccurate and might not provide useful information for current localization, it is even harder if the robot gets kidnapped, bumped or falls over, which are quite common occurrences during games. And all these motion states (walking, stopped, kidnapped, bumped or falling over) are indicated in our team code base by analyzing the robot's coordinates along with its transform.

Our team [2] built a localization simulator tool [1] that provides visualization that uses simulation to analyze how one robot localizes itself based on its observation of the circle, ball, and intersections on the field using a multi-modal Kalman Filter. As we can see in Figure 3 below, the robot sees the circle, white line in the center of the field and the yellow goal post from its top camera (as indicated in Right Figure in the Vision Window), and it localize itself right in front of the circle facing the goal on the opposite direction (robot is represented in white near the middle circle, with black indicating the orientation, and the dashed white line as its visual cone) based on its observation of the circle and lines.

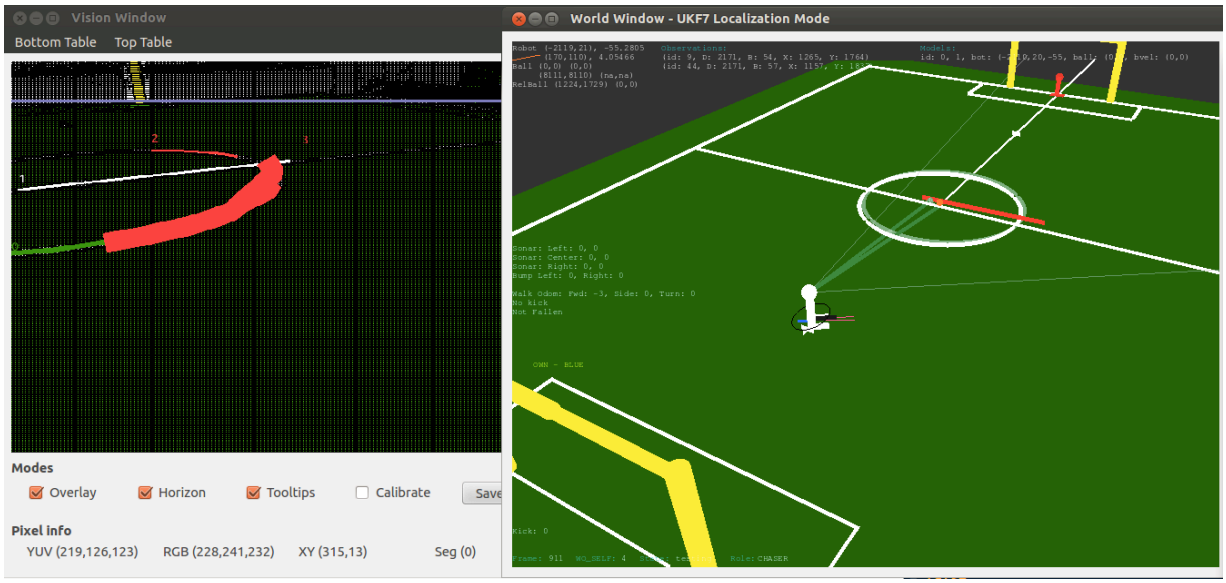


Figure 3: Localization Simulator with location estimation (Right Side, in white, single robot) and the visual image of Robot (Left Side, Sampled-pixel)

To solve the multi-hypothesis problem on symmetric sides of the field, more general visual information could be integrated rather than only recognizing artificial landmarks, which necessitate manual descriptions and prior knowledge of their relative position. The approaches using surrounding information, such as natural landmarks, are more like the approach of human beings, who would understand the surroundings based on the features learned from prior training processing and then predict their orientation according to features using recognition and feature matching. A necessary condition to ensure this approach works is that there must be significant differences in the off-field surroundings from different orientations on the field both horizontally and vertically such that distinct features can be generated as natural landmarks to represent orientation. The main tasks are feature extraction, training, and testing progress.

2.4 RELATED RESEARCH

In the area of visual recognition, there has been much recent work on invariant features in 2D images. One of the popular features is the Scale-Invariant Feature Transform (SIFT) [4] feature, which transforms images into a large collection of feature vectors and is based on the appearance of the object at particular interest points. It is invariant to image scale and rotation and is demonstrated to be robust after non-trivial changes in illumination and noise, and also after minor changes in viewpoint5. SIFT keys are selected at the maxima and minima of the results of a difference of Gaussians function applied in scale space to a series of smoothed and resampled images. It discards low contrast candidate points and edge response points along an edge. At each feature location, an orientation is selected according to the main peak of a histogram of local image gradient orientations. Each SIFT feature is associated with a subpixel image location, scale and orientation. Speeded Up Robust Features (SURF) [3] is another local feature detector related to SIFT, but is several times faster than SIFT and is more robust against different image transformations than SIFT. These features have been used for object recognition and image retrieval [17][18]. To further detect the key features, Bag of Words (BoW) approaches used in text retrieval could be adapted to vector quantize the descriptors into clusters which will be the visual 'words' for each labeled type for retrieval or matching [13][16][17][18]. The visual vocabulary is constructed from the images descriptor with same training labels. The vector quantization could be carried out by K-means clustering, K-medoids, histogram binning, mean shift, etc. and the search cost could be further reduced using a vocabulary tree or a randomized forest of k-d trees [19]. Also, the transformation between the planes, which is described using a homography matrix, could be calculated using RANdom SAmple Consensus (RANSAC)[23][24]. Robot localization generates probability distributions of pose

estimations over the entire possible mapping space using sensor observation and matching features. Thus, these features are also applicable for robot localization and navigation [14][15][16].

As for the application of visual recognition to robot localization, there are many research efforts, especially in Robocup application scenarios. In the 2012 Open Challenge, rUNSWift, the SPL team at the University of New South Wales, demonstrated a vision based localization approach running in real time using natural landmarks [8] to solve the problems resulting from both goalposts being the same color. They use modified 1D SURF extracted from the row pixels on the robot's horizon and the features are then matched using the nearest neighbor with order and scale constraints so that only features appearing in the same order are matched. This approach is demonstrated to be efficient enough for real time usage using SURF extracted from grey-scale pixels instead of SIFT since using SURF is faster [12]; in addition, it is robust in response to repeated landmarks, variations in illumination, changes and occlusions, as well as small changes in viewing angles and significant changes in scale. The rUNSWift team pointed out that the approach could be improved by using sequential frame matching to extract visual odometry information and to estimate robot rotation. Since the approach is not scalable to a large number of images, they suggested SLAM to combine features into one integrated map [11] or a BoW approach using discrete features clustered from training data [9][10]. Their approach are derived from Montemerlo *et al*. [11], which use a 1D variant of SIFT and a dynamic programming matching algorithm to localize a robot using an omnidirectional camera while in Robocup the cameras are restricted to viewing angles of only ~40 degrees. This approach relies on kinematics to obtain the 1D image ring used for feature matching, which is limited by the accuracy of the robot's kinematic calculations and pose estimation. It is also based on the assumptions that (1) the

horizontal features from each orientation remain almost the same, which might not be satisfied during the game, and (2) the processing is adequately quick due to the use of a 1D image, although this does not resolve the latency of processing issues. Another approach that is designed for more mobile vehicle robots and for use in large environments is FAB-MAP, which is an appearance-based place recognition and mapping model that uses a learned visual vocabulary. The approach is a probabilistic framework that is robust even in visually repetitive environments where many locations may appear essentially identical. It is a BoW based image retrieval system and has demonstrated that simple ranking algorithms operating according to image features are more effective and accurate than the tf-idf relevance measure borrowed from text retrieval. In the probabilistic framework, the joint distributions are approximated using a Chow Liu tree, and inference performance can be improved relative to a naive Bayes model. The approaches requires complicated computation and runs in real time for an autonomous vehicle robot that produces more stable image observations than humanoid robots.

The most similar research is found in [8][14][15][16] as we introduced before. Given the processing constraint and also the problem scope of localization in Robocup, the approaches adopted here are a sampled 2D SURF/SIFT feature exaction for training and testing images off-field, a BoW approach for refining the features, and then KNN/FLANN for matching with spatial constraints and RANSAC for transformation calculation. Next, the pose and orientation of the robot will be estimated, and an observation will be made of the natural landmarks in parallel with the artificial landmarks. More detailed approaches and problems will be described in Chapter 3.

CHAPTER 3 ORIENTATION ESTIMATION USING SURROUNDING NATURAL LANDMARK

To restate the central problem addressed in this thesis, we are striving to improve robot localization using off-field natural landmarks at times when on-field artificial landmarks can provide only insufficient localization information. In this chapter, we state our main methods to achieve our goal and present details concerning their implementation. We introduce a training and testing procedure and a framework for processing caputired images, extracting useful features, and matching these features to obtain an orientation hypothesis along with its localization certainty. We close with a demonstration of how we solve the parallelism and latency problem and how we fit this framework into our current localization framework.

3.1 TRAINING PROCESS

The training process consists of collecting the images and extracting features from raw camera input when the robot is moving and localizing itself using its known odometry. In our experiments, the process takes three to five minutes as the robot walks across the field near each long edge, orienting towards the center of the near and far long sides of the field. The ground truth directions are recorded for each run, images from only one direction will be recorded in the log, and since the walking is undisturbed from one side of the field to the other while facing the same direction, the localization is accurate enough to track the actual odometers and locations of each frame of images. These logs will be used as training data, with the raw images as input, the extracted feature progress as actual testing or gaming, and with the actual pose detected using localization as training labels.

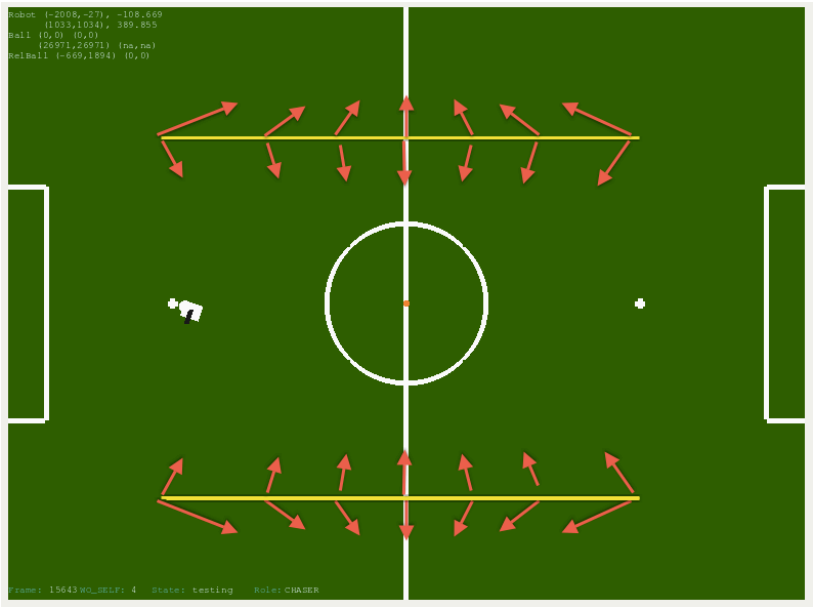


Figure 4(1): Training Process to collect natural landmarks

Figure 4(1) above is the visual description of the training procedure: the yellow lines are the trajectories the robot follows from one end to the other and the red arrows indicate the direction the robot is looking in. The robot looks either inside or outside during each run, and thus we need four runs in total to capture all of the required information. Since the robot is well localized in certain trajectories, the angles to face approximately at the center of the side of the target field are calculated in real time based on the odometer estimation; thus the robot can keep facing a certain area of the field. As indicated in figure 4(2) below, with the robot estimation of pose (x_t, y_t) and orientation θ_t obtained from normal localization at time t, we could calculate the angle at a certain time necessary to face exactly at the desired location (x_d, y_d) by using the formula $\Delta\theta = [\theta_t - < (x_t, y_t), (x_d, y_d) >_\theta]_{[0,180)}$. Our objective is to turn from current angle to the angle of lines between (x_t, y_t) and (x_d, y_d), then it is regulated to be in the range of [0, 180]. Here in our scenario, the desired facing point (x_d, y_d) are either of the two

center points on the field, as labeled in figure 4(1). For training it is facing one of the direction at one round of walking, while in the actual testing game, the turning angle are the smaller one in the left (y-positive) side and right (y-negative) side. We are picking these desired viewing points because more significant features are contained in the view.

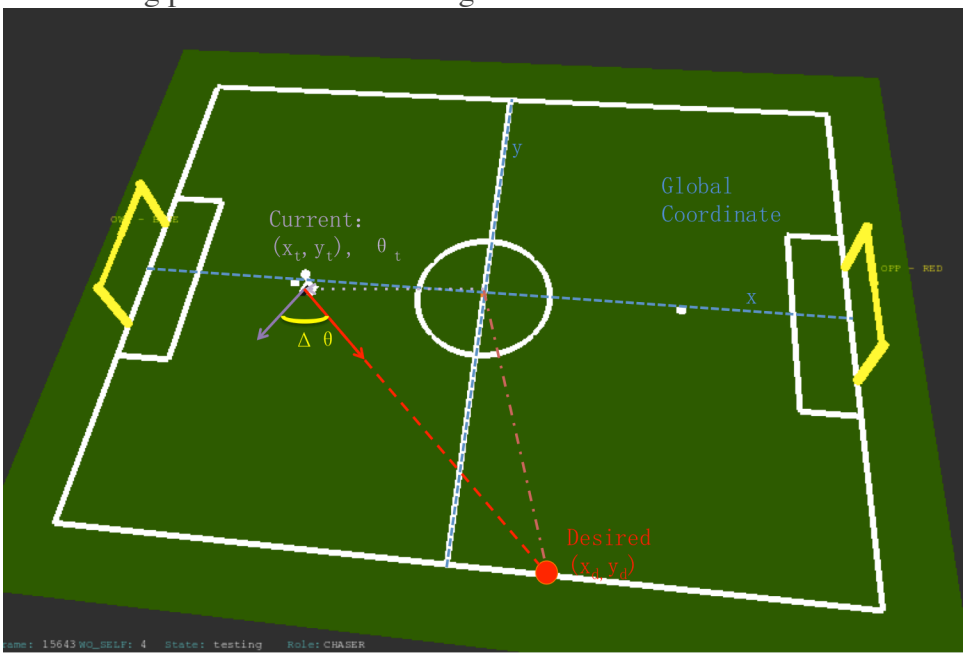


Figure 4(2): Training Process while facing certain desired points

3.2 IMAGE PREPROCESSING

We only use images from the top camera because we assume that only if robots are standing will all of the off-field images stay in the top camera's view and the orientation and pose predictions are stable and accurate enough to help the robot recover or to improve its localization hypothesis. Otherwise, when robots are in the middle of falling over, getting up, or being kidnapped, the robot's posture is not straight and thus the states are much more unstable, which results in uncertainty and mismatches in image

processing. We also choose not to process images to generate hypotheses when the image is blurred and no significant features can be extracted, or when the image is totally without significant green content, in which cases the horizon line detection process will result in imbalanced or no results.

The top camera images are preprocessed to get the off-field sections only. Instead of detecting the horizon based on kinematics and camera transforms, we are using the image itself, since the robots are not stable enough to help localize in most of the scenarios when we actually need the off-field landmarks. We need to keep only the off-field portion because the on-field features interfere with the process of matching the actually useful off-field features as the on-field features are much more common and also significant, meaning the white lines and yellow goalposts will be mistakenly matched to lines anywhere which will result in false positives. The detection of the boundary of the field is based on color similarity. To be more efficient and accurate, instead of calculating the transformation from feet to camera, we use a simple heuristic that generates the distribution of colors from sparsely sampled pixels to see if it exceeds the threshold percentage of green colors. To calculate the actual boundary, which we called the horizon line but which might be a gradient instead of a horizontal line, the images are separated into left and right sides from the middle line; we then calculate the boundary where green starts to dominate the pixels to be the key point on each half, and assume it is centered in the middle of each half. The two points on each side will determine the horizontal line. One of the heuristics for sampling is a 8-pixel step in the horizontal direction and 4-pixel steps in the vertical direction, and the iterated data is processed at least 32 times faster than the original raw images. Computational time is saved since we stop scanning early once the threshold is reached. The other options are similar to starting from an approximated one and then binary searching to keep lowering the computation to log

complexity. The performance and accuracy are enough to reach our expectation. The only issue is that, when there are other robots or large obstacles blocking the top camera's view, the calculation will provide more an unbalanced or a lower horizon line estimation. One fix is that if the gradient is too large to be normal, we will use a heuristic to trust the average of the left and right sides as the points and assume it is near horizontal. The visualization for the algorithm is as below.

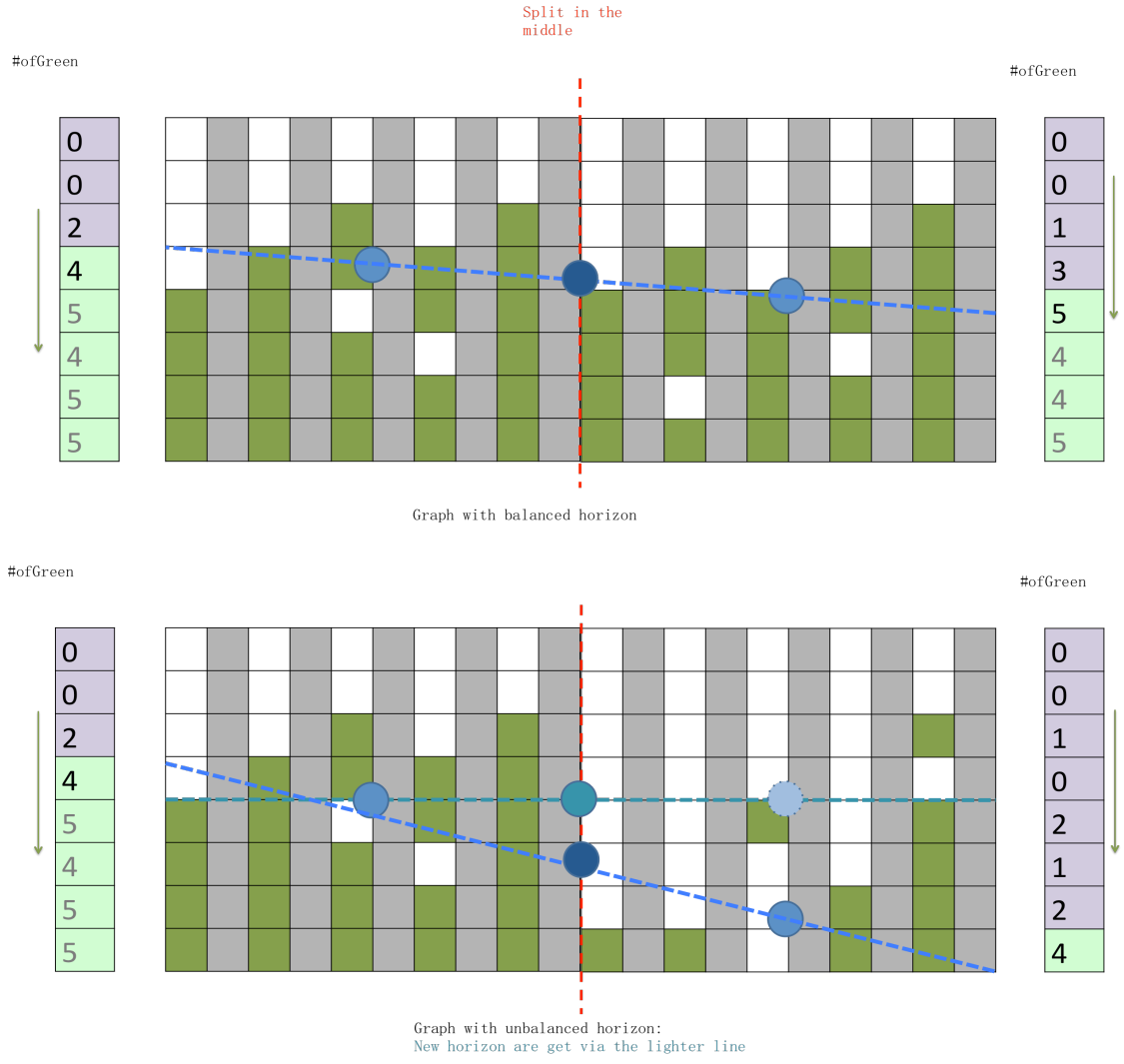


Figure 5: Horizon line estimation, for two situations (threshold=3)

In figure 5, the blocks in the middle are the images, and grey color blocks are the blocks we are not sampling; the green blocks are detected as green and the white as non-green. Here green is quickly determined in the color segmentation phase in YUV color. The graphs are partitioned in two, indicated by the red dashed line, and green color segment count increases from top to bottom as indicated in the blocks on the side. We stop once the threshold is met, which we assume is 3 here, and all the instances below are assumed to be green, as indicated again in the side blocks. Then center points on left and right halves are obtained (indicated with blue dots), so we can obtain the gradient and the point to fix our horizon line, as indicated by the blue dashed line. In the bottom images, since the left and right halves are too unbalanced, is indicated by the gradient, we get a different horizon line, as indicated by light blue dashed line, since we assume horizon line should within certain gradient range.

For example, the figures below are visualizations of the robot's top and bottom cameras from our tool [1]. The left large images are from the top camera. This visualization is after the color segmentation and only pixels are processed at a 2-pixel step interval in the interest of efficiency and thus it is sparse compared to other visualizations. The blue line is the horizon line， indicating the separation between on- and off-field. The four images in the right top corners are the top camera raw images, top camera segmented images, bottom camera raw images, and bottom camera segmented images. Here we verify the correctness of horizon line detection by comparing the blue line in the segmented image with the raw image, in which the horizon line is easily detected by human beings. As you can see in Figure 5 below, in general, the horizon line estimations are pretty accurate, and in Figure 6 below, labeled by the purple line, the horizon line is calculated correctly despite objects blocking the camera's view.

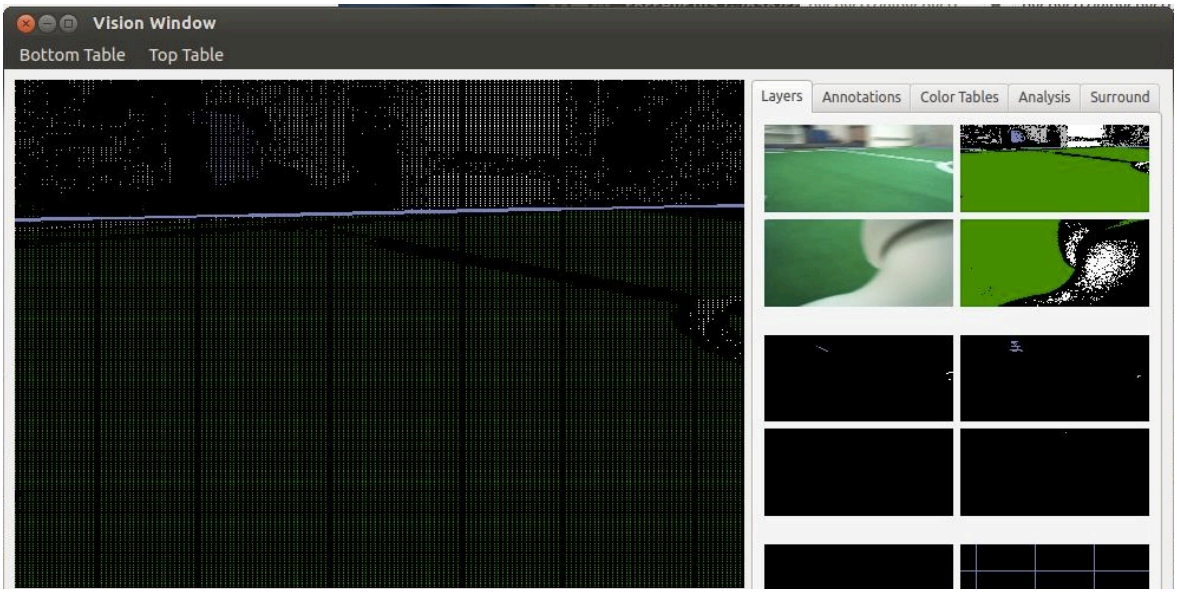


Figure 6(1): Horizon line estimation, non-blocking situations.

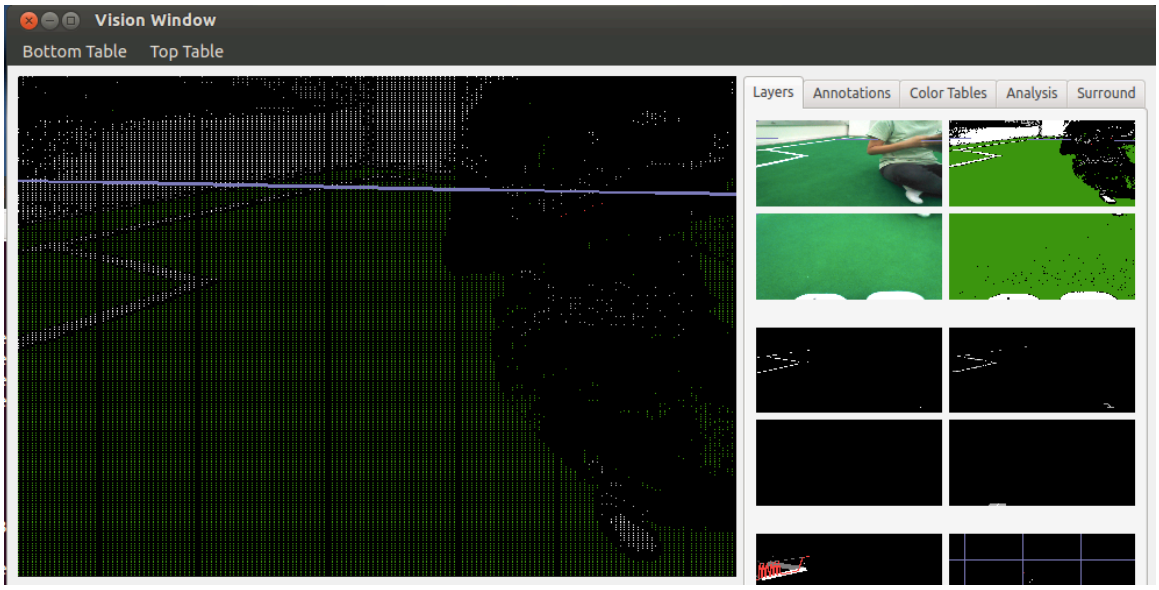


Figure 6(2): Horizon line estimation, blocking situation.

3.3 IMAGE FEATURE EXTRACTION

After processing and vision recognition, the direct features extracted from the off-field part are SIFT/SURF features. Here we use OpenCV [7] to extract features. The raw images from the robot we are using are in YUV space, thus we convert them to grey scale space after preprocessing to be used by our feature extraction. To show the final results, we also convert each image into RGB space but this is only done in the simulator instead of also onboard the robot, so that even if it takes an extra second or so in the simulator, the computational efficiency on the robot is still within an acceptable range. SIFT [4] is chosen as the basic feature extraction method since it is designed to be invariant to a shift of a few pixels in the region position compared to other descriptors, which is very often necessary given the movement of the robot. Combining the SIFT descriptors with covariant regions will give us the region description vectors that are invariant to transformations of images. The region descriptions are further used in matching to prioritize the matching that is spatial correlated and affine. The region descriptors are calculated using easy nearest-neighbor clustering in coordinate of the image, by which we can get a region described by the radius and center and, in the matched images. Our assumption is that features should remain in the same region in the testing and matching area. Each SIFT is 128-vector and each image contains 0-300 descriptors. To speed up the matching and improve the descriptive features for each image, after the descriptor generation, we perform a visual vocabulary analysis to generate vectors that quantize the descriptors into clusters, which are similar to "words" as they are widely used in information retrieval. Here the functions in OpenCV [7] are adopted, which applies K-means clusters to generate a randomized forest of k-d trees [18][21][22], and stop-list to filter out the most common features. We choose to generate 100 clusters to be "words" using the training images. Each feature descriptor then will be represented by the term

frequency inverse document frequency (tf-idf), which uses weighting instead of simple frequency in image processing. In general each valid images contains over 200 candidate feature descriptors, which results in less efficiency in representation and also matching; however, there are only a few distinct features that significantly determine the matching, which lowers the dimensions necessary for the next matching steps and is thus more efficient.

Though preprocessing, descriptor extraction processing, and then matching all take time, the most significant time consumption occurs in constructing the vocabulary. In our approach, the vocabulary dictionary is generated offline after the training procedure, and is saved to be directly used in matching and prediction, thus making our almost real-time matching both possible and easier.

3.4 MATCHING AND PREDICTION

While the robots are walking in real time, they will process raw images the same way as for the testing and training data set. When the robots lose their direction, in order to obtain an accurate estimation, based on the current transformation and estimation as described in Section 3.2, the robots are actively facing what we call either the lab side or the shelf side. For each off-field image corresponding to the queried frame, we obtain the similarity between itself and all of the training image feature vectors as described in Section 3.3. They then are ranked by the scalar product (cosine of the angle) using $sim(v_q.v_d) = \frac{v_q^T v_d}{||v_q||_2 ||v_d||_2}$, where $||v||_2$ is the L-2 norm. OpenCV[7] contains a collection of algorithms we found to work best for nearest neighbor search and a system for automatically choosing the best algorithm and optimum parameters depending on the dataset. Instead of selecting the best match (1-NN), in our scenario k-Nearest Neighbor

(k-NN) and its approximate variant Fast Approximate Nearest Neighbor (FLANN) [6] are considered since they perform better in high dimensional spaces such as image processing and they are optimal for multiple matching which introduces more uncertainty. To improve the multiple-to-multiple matching generated using k-NN/ FLANN [6] in OpenCV, we filter the matching to optimize and generate a one-to-one match with the highest matching score. Because the same features are not repeated in different places in feature matching scenarios, this type of bipartite matching produces the more accurate matching results than multiple matches.

Other than just the similarity between the feature descriptors, we also consider a special factor: the consistency with which a certain cluster of objects will remain in the same area. Each matching candidate is evaluated based on the best matching similarity distance and by how the points are consistently outlayed. The robot calculates spatial consistency using the distance between the current spatial region matching descriptor and the matching of all features in its spatial region description, since all features within the region should remain close to the matched images. In the figure below, we present an example of filtering the matching based on similarity and also spatial regional consistency. As indicated, the descriptors in the testing images (red dot) are matched to four descriptors (red, orange, purple, and pink dots) as candidates in the training images using K-NN in feature similarity. Then the region descriptors of the matched descriptors are matched back to the testing images by searching the multiple matching features. Region 1 is much more similar than regions 2 and 3 in spatial structure when matching back to testing image. Using both spatial consistency and similarity of features, the best matching is obtained and identified as Feature 1 (yellow highlighted circle).

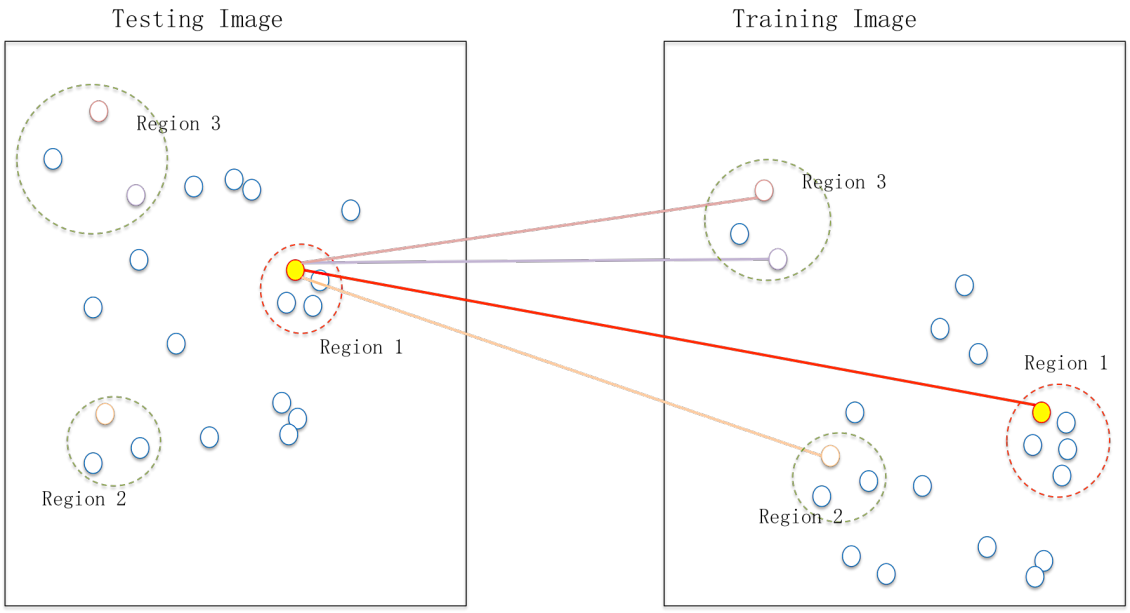


Figure7: Multiple Matching (4-nn in feature space) are filtered to one best matching based on spatial consistency of region description.

In our implementation we pick 4 candidates as defaults but it could be extended to all descriptors as candidates and, first, the majority of the clustered candidates are selected as the main hypothesis, and the score is the similarity (calculated from the percentage of matched features out of all detected features and a Cartesian distance in BoW features) from these good hypotheses. Next, the similarity of the outlier hypothesis will contribute negatively to the certainty. When there is no feature in test images (results in matches = 0), the certainty is 0 and thus we do not provide uncertain hypothesis for localization. If there is no feature extracted, there are similarities based on Cartesian distance, though they are quite small (< 0.5, good matches are > 0.97 usually). The robot said "None" only when certainty is less than 0.2, and the estimation based on surroundings will not be passed back to the localization system due to the uncertainty; otherwise it speaks the letter “S” for shelf (which is the positive y direction +90) or “L” for lab (which is the negative y direction -90) and saves the certainty and accuracy as

logs. Empty feature descriptor will return certainty 0 and it is caused because of movement speed and images are too blurring to obtain reliable estimation. When uncertainty results are generated, they will not be passed back to localization in main thread as observation, thus we only keep the useful hypothesis to help improve the total localization.

ALGORITHM: *Feature_Matching*(img1, img2, img1_descriptor, img2_descriptor)

Initialize empty best matching pair, BestMatchingSet = {}

multi-matching= K_nn(k=4, img1_descriptor, img2_descriptor);

foreach matching (d1, list[] d1_matching) in multi-matching:

sp1 = spatial_descriptor(d1)

foreach d2_i in d1_matching:

d2_i_match = multi-matching.getBestMatchFor(d2_i)

sp2_i = spatial_descriptor(d2_i_match);

sim2_i = similarity_feature(d1, d2_i);

score_i = η distance(sp2_i, sp1) + sim2_i

choose best candidate i with highest score, $best = \max_i\{score\_i\}$

add matching_pair, BestMatchingSet.add (d1→d2_best)

return BestMatchingSet

ALGORITHM: *Similarity_Feature*(v_d, v_t)

return $sim(v_q . v_d) = \frac{v_q^T v_d}{||v_q||_2 ||v_d||_2}$

Table 1: Pseudo Code for the matching algorithm.

The block above shows the pseudo code for the entire feature matching processing. The system is built on the UT Austin Villa 2012 SPL code base [1], which supports an interface for real robot connection and simulation simultaneously, as well as supplying a debugging tool that provides different functional testing by locally running all processing. Using this framework, we use the localization model to monitor the hypothesis for the robot's orientation and pose, and we also create a new visual model to understand how each frame could be matched to training sets. The screenshot of the new matching with the surrounding feature tool is in Figure 8 below. The left window is our SPL software used to select different testing modules. After taking logs, we could, for example, transfer the logging file to the debugging tool, load each frame, and run the processing the same as real time. Currently, the processing goes to frame 33 out of 65 frames. The right side is the Vision Window with only the surrounding match modules opened; the left side is the enlarged pictures of one of the candidates from the right side. Each matching contains left and right images, which are currently the vision frame and the candidates from training. Each matching is labeled using the colorful small circle dot and if matched, they will be connected using a colored line from the raw current frame images to the training images, which have best matching scores. The off-field part is indicated with a red square, and the corresponding RANSAC transform on the matched image. And the colored lines with circle on both end are the matching between the features, while the circles are the features in the original SIFT or SURF space. As we can see, there are still mismatches (false negatives), but in general, the matching is accurate, as we can see using the matching analysis tool in the vision system. Even though images are sometimes blurred, the bag of word approach could still provide some similarity information, but usually with less certainty.

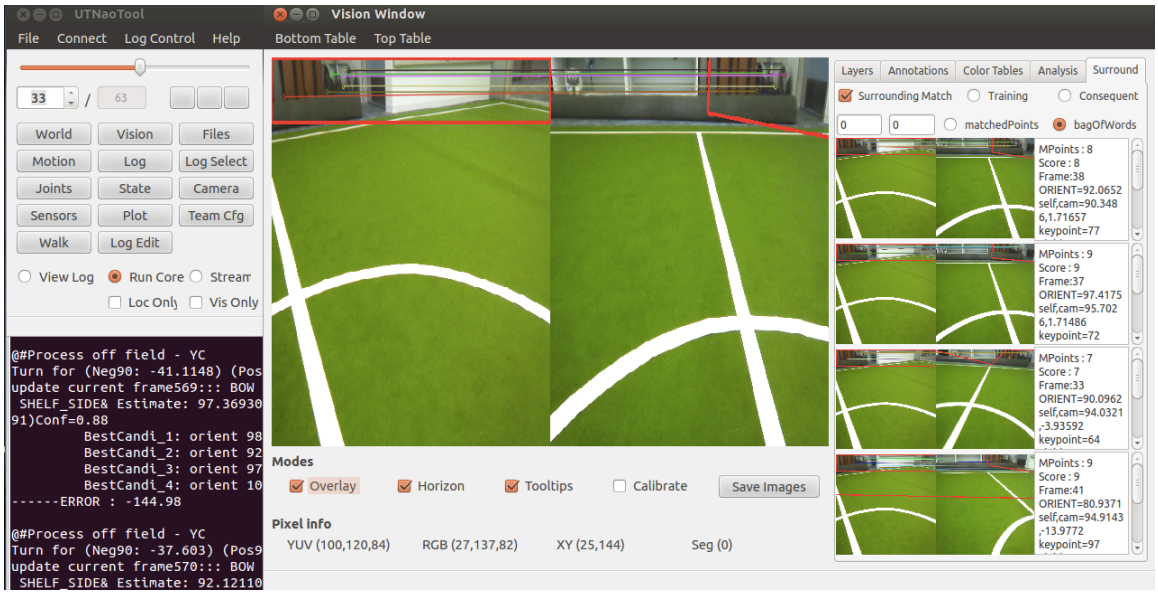


Figure 8(1): matching analysis tool in vision module, normal images

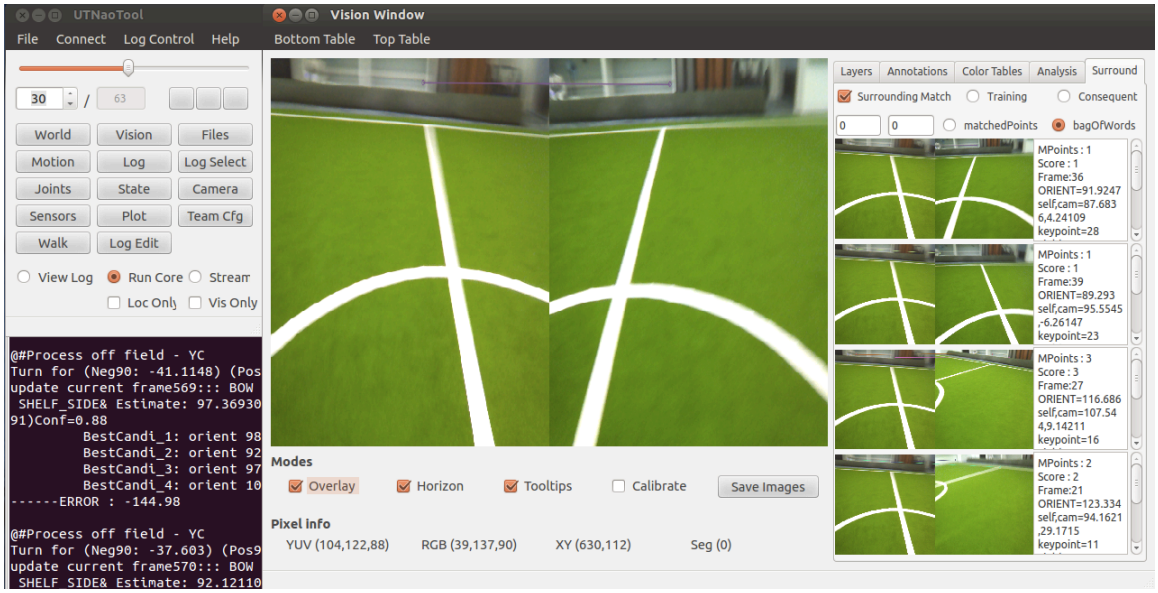


Figure 8(2): matching analysis tool in vision module, blurring image

3.5 PARALLELISMS AND LATENCY

On one hand, the localization using off-field information is a highly computational processing since much more off-field pixels of raw images need to be analyzed; then feature extracting, classifying and predicting orientation based on matching all also take memory space and computational time. On the other hand, the pose and orientation prediction based on off-field information, the natural landmarks, is not necessary while we are in stable and accurate odometry and localization updating cycles. It is only if the certainty of localization hypothesis drops severely that the off-field information would significantly help in resetting and recovering the correct localization hypothesis. Thus, our localization using natural off-field landmarks are run parallel on another background thread with less priority rather than in main thread. This allows more important and general processing, such as motion planning, vision recognition, localization to be run smoothly without disturbance from our background processing. Only the final results—the hypothesis of orientation prediction based on natural landmarks from off-field—are simultaneously sent to the main thread as external input for the localization modules as an extra fix for the pose-orientation. This prediction will only have significant influence on our localization hypothesis in the Kalman filter when our robot motion states are *falling or getting up*, *being bumped*, or *kidnapped*, in which cases the hypothesis and odometry updates based on actual on-field landmarks such as lines and circles are less certain. Since processing takes time, when returned, the estimations we obtained from analyzing off-field features and matching natural landmarks are latent compared to current the localization estimation. We also label the frame time t_{PREV} and localization hypothesis H_{PREV}, which is uncertain before processing, and the frame time t_{CURR} and H_{CURR} when sent back, and the odometry updates O_{CURR} since the images are cached for background processing; our processing will generate the

hypothesis based on natural landmarks NL_{PREV}, which will improve the H_{PREV} to a more accurate $H_{PREV}^{/}$, and accumulate the odometry updates to obtain a more accurate hypothesis of current states approximately, $H_{CURR}^{/} = H_{PREV}^{/} + O_{CURR}$, rather than $H_{CURR} = H_{PREV} + O_{CURR}$, with lower confidence. This update solves the problems of latency and asynchrony for the background processing as we are updating simultaneously to use the previous hypothesis to enhance our current hypothesis. Compared to running the entire feature extraction and matching process in the main thread, which drops our vision frame rate from ~29.7 fps to ~27.3 fps, the approach with parallel processing and simultaneous updating, maintains the main frame rate at ~29.6 fps while returning processed results after around 0.1 s processing time. The entire parallelism is built on a Posix Thread (pthread) with boost interprocess mutex to maintain simultaneous accessing of image memory as well as noticing the main localization module. The results are written into memory using memory access interface defined in the team code base [6].

CHAPTER 4 EXPERIMENT AND RESULTS

In this Chapter, we introduce the method we used to verify the effectiveness of our framework and algorithm. The experiments are set up using different feature extracting and matching algorithms and the results are provided. We also performed the experiment of localization during walking and other unstable states, e.g. kidnapping, to verify that the recognition of surrounding natural landmarks provides accurate estimation on orientation to help localization.

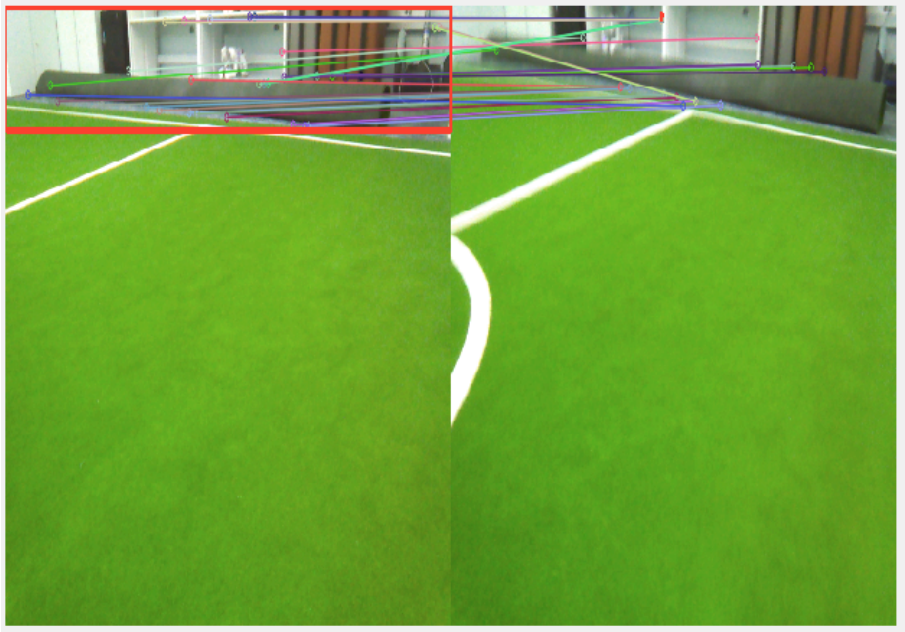
4.1 EXPERIMENT

Here we evaluate robot localization using off-field natural landmarks over a few testing scenarios. During the experiments, image and localization logs are recorded using the tool developed by the UT Austin Villa team [1][2] during the training and testing procedure, so that different algorithms can be examined and rerun on the same data sets off-line. Also for debugging and testing concerns, the robot is speaking out its orientation of "L" (lab side), "S" (shelf side), or "None" (low certainty) as described in section 3.4 to indicate its orientations prediction. We introduce the ground truth using odometry updates from the localization process and also manual placement of the robot at certain locations, to compare the localization performance and accuracy. The certainty levels are recorded and saved together with the ground truth, estimation orientation, and estimation errors in an extra log file for further analysis.

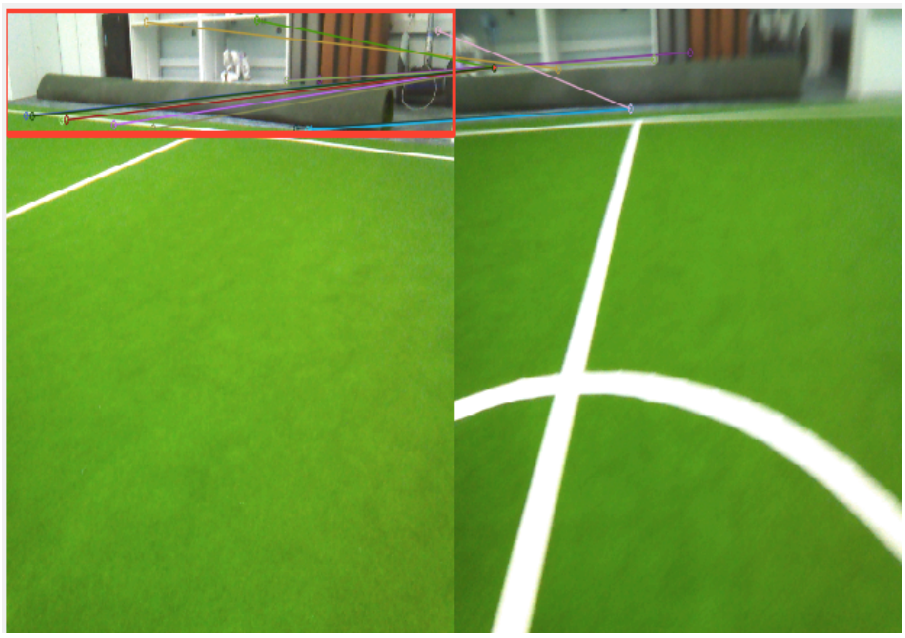
4.2 RESULTS: MATCHING

Figure 9 below illustrates a selection of typical output from our image processing and matching system, with different features and matching algorithms, and each of the

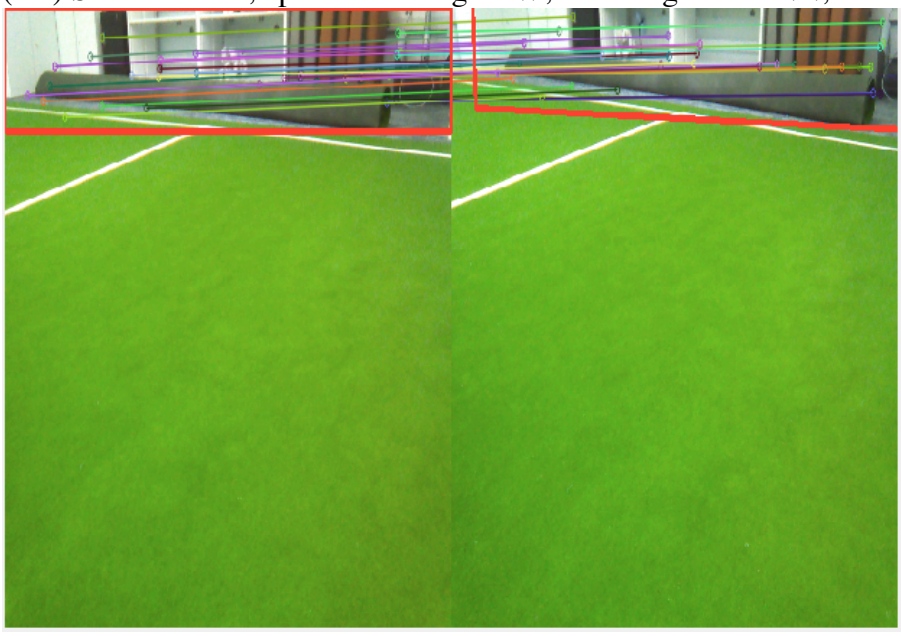
corresponding top four candidates (Figure 10). The left half in each image is the current queried image while the right half is the matched image from the training data. More colored lines between them indicates that more feature matches are found, which might not necessarily result in better matching; but we can get the ground truth using the orientation estimation data in Table 1 or just from manual observation. As we can see, all of the left-half images, representing the queried image, are the same out of many possible representative examples, and thus we can compare the performance of each algorithm..



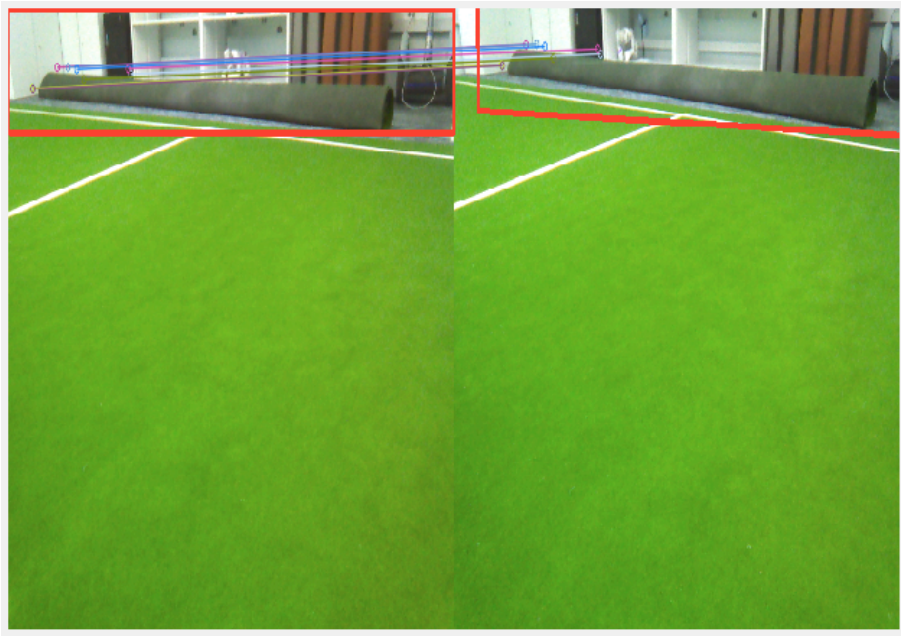
(1.a) SIFT features, optimized using BoW, matching with FLANN



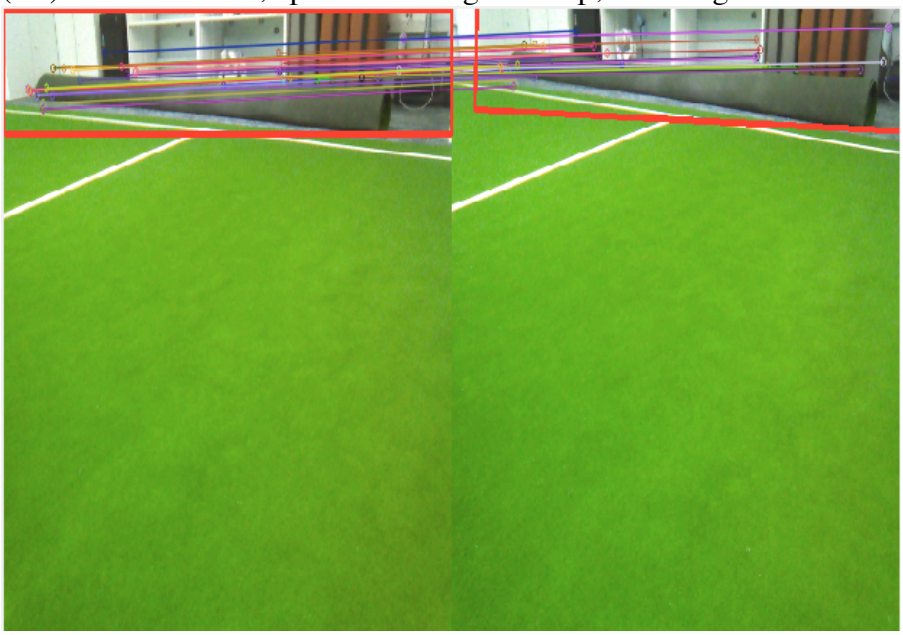
(1.b) SIFT features, optimized using BoW, matching with KNN, where n = 4



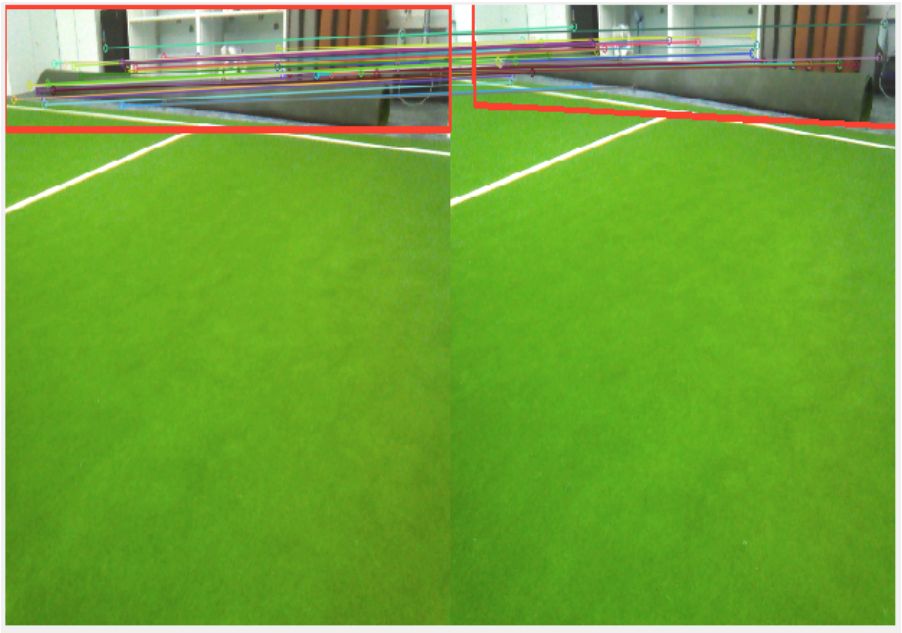
(1.c) SIFT features, optimized using BoW, matching with KNN, then space constraint to filter results as one-to-one matching.



(2.a) SURF features, optimized using FabMap, matching with FLANN



(2.b) SURF features, optimized using FabMap, matching with KNN, where k = 4



(2.c) SURF features, optimized using FabMap, matching with KNN, then space constraint to filter results as one-to-one matching.

Figure 9: Feature Extraction and Matching Comparison

Figure 10: Feature Extraction and Matching Comparison (top 4 candidates)

	FLANN		KNN		BoW+ KNN	
SIFT	23.25	10.9110118°	15.75	4.54011111°	19.5	**0.84820513°**
SURF	14.25	3.67049122°	19.75	3.48810126°	**25.25**	3.66770298°

Table 2: Average Matching Points (L) and average absolute errors (R, in degree) of features and matching algorithms

Table 2 depicts performance using the average number of matching points and the average absolute errors for different combinations of features along with different matching algorithms. The larger the average number of matching points is, the more certain the estimated results we obtained. The average absolute errors are the absolute differences between the ground truth orientation from accurate localization and the voted weighted average of top four candidates' estimation. We experimented with different combinations of feature extracting and matching algorithms, attempting to eliminate false positive and to improve accuracy. Features using the BoW optimization is invariant of the space transform, but it does not slow our processing since though it consume more time in training, then cost much less time in matching due to lower dimensions and more significant feature representations. The best performance is from the implementation using BoW (with SURF/SIFT) features and matching with k-NN, then filtering with space constraints to obtain a one-to-one match. We can see that BoW improves performance by generating more significant features for matching, and using the spatial constraint to filter the original k-NN matching makes the matching more specific and accurate for our scenario. The one-to-one filtered matching outperforms the k-NN matching in accuracy since it eliminates much more false-positive matching between features, which is consistent with our assumption that repeated patterns are not distributed across the field, and that there are significant features for each orientation in 2D, thus the features are unique. Even though images are sometimes blurred, BoW can still provide some similarity information. One other condition is that the feature extraction on the lab side does change (due to people moving, the robot moving, or light changing) during recording and testing, but this does not influence matching in terms of distinguishing between the two sides.

4.3 RESULTS: LOCALIZATION

To verify that surrounding natural landmark recognition helps improve the accuracy of localization we also video record our experiments and verify the results by keeping track of the correct orientation when the robot is in a stable state for localization, or by keeping track of ground truth manually when it is unstable. And the errors between the localization based on on-field artificial landmarks and the estimation based on our off-field natural landmarks recognition with respect to the ground truth are recorded. And the results are shown in the table below.

Statistic From Videos	Lab Side				Shelf Side				Accuracy
# of Testing and Estimation	Correct	Incorrect	Uncertain	Overall	Correct	Incorrect	Uncertain	Overall	
Kidnappmoving.mp4	42	9	4(3)	55	42	3	8(5)	53	79.6 %
Kidnap.mp4	37	3	1(0)	41	25	6	4(2)	35	
RandomWalkCenterlab.mp4	17	2	1(1)	20	10	1	0	11	87.3 %
RadomWalkCenterShelf.mp4	13	3	2(1)	18	32	2	1(1)	35	
RadomWalkCross.mp4	21	1	0	22	13	2	2(1)	17	

Table 3: The number of Localization Estimation and Errors

One experiment (randomwalking_*.mp4 [25]) represents a stable situation: the robot is starting at a known spot, randomly walking and turning, and then stopping to detect its orientation on the field. Since the robot remain stable in the random walking

experiments, the localization based on on-field artificial landmarks generate pretty accurate estimation of location pose and orientation using Kalman Filter, which assumes robot movement is almost continuous and smooth. We assume the localization provides ground truth and compared it with our approach. In our experiments, the orientation predictions are 87.3% accurate in reporting the correct orientation when it should be able to identify direction in the 3 two-minute long experiments, which include 184 orientation estimations. However, when it is in a corner or on the far side of the field from the off-field images it is facing toward, which result in fewer obtainable off-field images, detection was significantly less accurate, down to 22%, which is indicated by the uncertainty part, with the number in the bracket showing the uncertainty caused by facing the corner.

The other experiment (kidnap_*.mp4 [25]) represents an unstable state and tests the robot's ability to recognize its direction while being kidnapped. All image are taken by the top camera while the robot is being held. It achieves an accuracy of 79.6%, compared to 23.2% estimation accuracy from localization using artificial on-field landmarks, which is much less certain and accurate. More detailed accuracy information can be found in the videos, along with all the features and logs located at address [25]. As we can observe, the robot can better localize itself based on surrounding natural landmark recognition using image feature matching, especially if the state is unstable, as in the kidnapping or falling-down situations.

As we introduced before, the estimation of localization using natural landmarks will only be transferred back when the robot is under unstable situation, which is pretty common in games, but less useful information from on-field landmarks results in uncertain or wrong localization, our approach could improve the accuracy and certainty of localization by provide support for the good hypothesis.

CHAPTER 5 CONCLUSION AND DISCUSSION

A framework for robots to recognize orientation using natural landmarks from surroundings for robot soccer domain is proposed in this thesis. Relying only on prior knowledge of artificial landmarks such as lines, circles and goalposts as observation input is less generic and the resulting predictions of hypotheses are less certain in complicated scenarios, for example, when goalposts for both sides are the same color. So, in addition to on-field information, we integrate orientation estimations based on off-field natural landmark matching to help improve the accuracy and certainty of localization hypothesis prediction.

Though the current approach enables our experimental humanoid robot to estimate its orientation approximately by distinguishing which side of field it is facing; it cannot however generate accurate enough estimations when it is in a corner of the field, where fewer images could be obtained due to the degree of freedom of the robot's motion, or when images are blurred because the robot is moving or turning at high speed. Another limitation is that we still need accurate labels as prior knowledge of orientation during the training process, which makes the approach less generic, compared to simultaneous localization and mapping approaches. And our approach needs further improvement to be accurate enough in more complicated environments, where certain patterns are repeated within the environments. Currently, the approach relies on an off-line training process to obtain information in advance; for more general purposes, a vision-based SLAM algorithm could be developed based on our current feature extraction and matching framework by tracking SIFT landmarks and building a 3D map

simultaneously. In this way, a robot could learn and update features in the surrounding environment by maintaining an adaptive global map while localizing itself.

The approaches proposed in this thesis demonstrate that natural landmark cues, generated from raw images by identifying interesting features, could provide more certain hypotheses for localization. More research can be done to adapt this integration of on- and off-field landmarks for robot localization to more general scenarios and applications.

References

[1] UT Austin Villa Team Source Code: http://www.cs.utexas.edu/~AustinVilla/?p=downloads/source_code_and_binaries

[2] Samuel Barrett, Katie Genter, Yuchen He, Todd Hester, Piyush Khandelwal,,Jacob Menashe, and Peter Stone, "UT Austin Villa 2012: Standard Platform League World Champions", 2013

[3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008

[4] Lowe, D. G, "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[5] Mikolajczyk, K., and Schmid, C., "A performance evaluation of local descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence, 10, 27, pp 1615--1630, 2005.

[6] Marius Muja, David G. Lowe. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration", 2009

[7] OpenCV http://docs.opencv.org/, Library in use: feature2d, openfabmap. http://docs.opencv.org/modules/features2d/doc/object_categorization.html http://docs.opencv.org/modules/features2d/doc/features2d.html http://docs.opencv.org/trunk/modules/contrib/doc/openfabmap.html

[8] Peter Anderson, Yongki Yusmanthia, Dr. Bernhard Hengst, "Natural landmark localisation for RoboCup" ,unpublished, University of New South Wales, August 2011.

[9] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, Cedric Bray, "Visual Categorization with Bags of Keypoints", 2004

[10] A. J. Briggs, Y. Li, D. Scharstein, M. Wilder, "Robot Navigation using 1D Panoramic Images," Proceedings of ICRA 2006

[11] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem". AAAI, July 2002.

[12] L. Juan and O. Gwun, "A Comparison of SIFT, PCA-SIFT and SURF". International Journal of Image Processing (IJIP), Volume(3), Issue(4). 2009.

[13] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0", The International Journal of Robotics Research 30:100, 2011.

[14] S. Se, D. Lowe, and J. Little, "Global localization using distinctive visual features," in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems EPFL, Lausanne, Switzerland, Oct. 2002

[15] S. Se, D. Lowe, and J. Little "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," International Journal of Robotics Research, pp. 735–758, Aug. 2002.

[16] Mark Cummins, Paul Newman, "FAB-MAP: Appearance-Based Place Recognition and Mapping using a Learned Visual Vocabulary Model", Proceedings of the 27th International Conference on Machine Learning, 2010

[17] Josef Sivic, Andrew Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos", Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV), 2003

[18] Josef Sivic, Andrew Zisserman, "Efficient visual search of videos cast as text retrieval", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008

[19] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. "Object retrieval with large vocabularies and fast spatial matching", In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[20] Nao Documentation Online: https://community.aldebaran-robotics.com/doc/1-14/family/robots/video_robot.html

[21] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages II:2161–2168, 2006.

[22] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object re- trieval with large vocabularies and fast spatial matching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007.

[23] Martin A. Fischler and Robert C. Bolles (June 1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. of the ACM 24 (6): 381–395. doi:10.1145/358669.358692.

[24] David A. Forsyth and Jean Ponce (2003). Computer Vision, a modern approach. Prentice Hall. ISBN 0-13-085198-

[25] https://www.dropbox.com/sh/8nzdp862kzm8znr/MUJinXFPdT.