In *International Workshop on Humanoid Soccer Robots* (HSR-10),
Nashville, USA, December 7 2010.

Online Detection of Instability for Robust Teamwork in Humanoid Soccer Robots

Kshirabdhija Nadarajan [*1], Mohan Sridharan [*2]

[*1] *Department of Electrical and Computer Engineering*
Iowa State University
Ames, IA 50011, USA
kshira90@iastate.edu

[*2] *Department of Computer Science*
Texas Tech University
Lubbock, TX 79409, USA
mohan.sridharan@ttu.edu

***Abstract*— Humanoid robots are increasingly being used in several domains as a result of the development of sophisticated algorithms for challenges related to vision, locomotion and team coordination. RoboCup is one such research initiative with the stated goal of creating a team of humanoid soccer robots that can beat the human soccer champion team on an outdoor soccer field by the year 2050 [1]. In order to achieve this goal, it is essential to enable a team of humanoid robots to collaborate effectively. In the Standard Platform League (SPL) of RoboCup [2], the instabilities induced by collisions, slippages and falls pose a major challenge to effective collaboration and provide a significant advantage to the opposing team. This paper presents an approach that enables a humanoid robot to autonomously learn a model based on sensory inputs to detect instabilities. The detected instabilities are communicated to teammates and incorporated in the team coordination strategy, resulting in a quick re-organization of the roles of the robots and more effective team coordination. All algorithms are implemented and tested on the Aldebaran Nao robot platform [3] in the robot soccer domain.**

I. Introduction

Developments in sensor technology and sensory input processing algorithms have resulted in the deployment of mobile robots in applications such as navigation, disaster rescue and health care [4], [5], [6], [7]. In recent years, there has been considerable interest in the design and deployment of humanoid robots in human-robot interaction scenarios [5]. Researchers have developed sophisticated methods for challenges in vision, locomotion (e.g., motion control, dynamic balancing) and team coordination on humanoid robots [8], [9], [10], [11]. However, the ability to accurately sense the environment and collaborate with other robots still remains an open problem.

RoboCup is a popular research initiative where teams of robots play a competitive game of soccer in simulated and physical robot leagues. In the Standard Platform League (SPL) of RoboCup [2], teams of three humanoid robots play soccer on an indoor soccer field. Typically, the robots in a team dynamically determine their roles (e.g., attacker, defender, supporter) based on the state of the game. The rules enforce strict penalties for collisions with other robots. However, despite the development of sophisticated methods for dynamic balancing and for detecting opponents [10], [12], collisions and slippage cannot be eliminated. In addition, recovering from falls or related instabilities takes a considerable amount of time. Unless the other robots in the team quickly re-organize themselves to account for the unstable robot, the opposing team can exploit the situation to their advantage.

In this paper, the term "instability" includes collisions and slippages that cause the robot to wobble, and the serious situations such as falls, i.e., all cases that can damage the robot and provide advantages to the opposing team. In the humanoid robotics community, researchers typically use models learned offline based on sensory inputs (e.g., accelerometers, cameras) to determine when the robot has fallen or to detect obstacles [10], [11]. During online operation, the robot executes predefined motion patterns to recover from falls or to avoid obstacles. However, although the robot is able to detect falls and communicate this information to teammates, detecting other instabilities accurately is a challenge. The robot can perform dynamic balancing routines to offset the effects of instabilities only if the instabilities are detected in the early stages. In addition, different predictive models need to be trained for different walks performed by a robot on different surfaces, and generating manually labeled samples can be very time-consuming. Furthermore, the learned models have to be revised over time to account for any degradation in the data provided by the sensors, and the output of the predictive model has to be incorporated in the team coordination strategy. The proposed approach addresses the above-mentioned challenges through the following significant contributions:

- It enables a humanoid robot to autonomously generate labeled sensory input vectors, which are used to learn and revise a model that detects instabilities robustly.
- It enables the robot to use the communicated instances of instabilities (from its teammates) in the team coordination strategy, in order to achieve effective collaboration.

All algorithms are tested on a humanoid robot platform (Aldebaran Naos [3]) in the robot soccer domain.

The remainder of the paper is organized as follows. Section II describes the proposed approach and the test platform. Section III describes the experimental setup and results,

followed by a review of related work (Section IV) and the conclusions (Section V).

II. Test Platform and Proposed Approach

This section describes the robot test platform, followed by a description of the proposed approach that detects instabilities to achieve better team collaboration.

A. Test Platform

The Aldebaran Nao humanoid robot [3] is used as the test platform in this paper. The 58cm tall robot has 23 degrees of freedom; five in each arm and leg, two in the head, and one at the pelvis. The primary sensors are the monocular color cameras in the forehead and nose, each with a 58^o diagonal field of view and a maximum resolution of 640×480. There are two ultrasound sensors in the chest, one each on the left and the right with a 60^o field of view. In addition, the robot has accelerometers and gyroscopes in its body and bump sensors in its feet, which may be used for collision avoidance, Furthermore, the robot has microphones, loudspeakers and LEDs that are primarily used for debugging purposes, and Wi-Fi to communicate with other robots or an off-board PC.

One application domain for the Nao is RoboCup, an international research initiative with the stated goal of creating a team of humanoid robots that can beat the champion human soccer team on an outdoor soccer field, by the year 2050. The Standard Platform League [2] of RoboCup has teams of Naos playing soccer on an $\approx 6m \times 4m$ indoor soccer field. Figure 1 shows images of the robot and the soccer field.

Fig. 1: The Nao [3] robot and the robot soccer field.

The robot soccer domain presents many of the challenges that need to be addressed for deploying a humanoid robot in the real-world (e.g. vision, motion, localization and team coordination), while providing a moderate amount of structure that makes the domain tractable to solutions. However, all processing for vision, locomotion, localization and team coordination is to be performed in real-time (30Hz) on board the robot, using an x86 AMD GEODE 500MHz CPU that runs embedded Linux. One major challenge in the domain is the ability to adapt team coordination strategies to account for instabilities. In addition, the sensory inputs are not completely reliable. For instance, the ultrasound sensors frequently fail to report the presence of obstacles in the robot's path, making collisions all the more likely. The proposed approach that addresses these challenges is described below.

B. Proposed Approach

This section first describes the method that enables the humanoid robot to autonomously learn a classification model to detect occurrences of instabilities. It then summarizes how this information is used in the team coordination strategy.

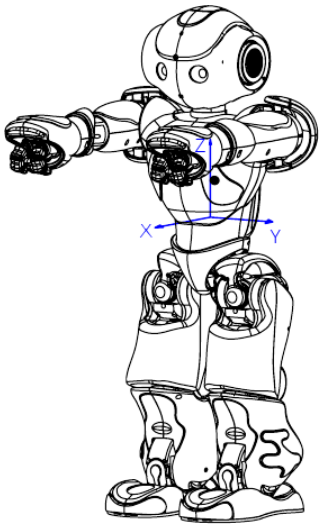


Fig. 2: The Nao robot's body axes [3].

The Nao is equipped with a three-axis accelerometer, which provides values in the range $\pm 2G$ along three axes, and a two-axis gyroscope—the body axes are shown in Figure 2. The proposed approach learns a predictive model using the accelerometer and gyroscope values. The input is hence a stream of five-dimensional vectors of the form:

$$\text{input stream} = \{\mathbf{s}_t\}_{t=1:T} \qquad (1)$$
$$\mathbf{s}_t = \langle a_{1,t}, a_{2,t}, a_{3,t}, g_{1,t}, g_{2,t} \rangle$$

where the sensory input vector at time t, i.e., $\mathbf{s}_t$ consists of the accelerometer values: $\{a_{1,t}, a_{2,t}, a_{3,t}\}$ and the gyroscope values: $\{g_{1,t}, g_{2,t}\}$. The data from these sensors represents the relative position and hence the stability of the robot's body. Our experiments indicate that using the proposed input vector provides much better performance than using just the accelerometers or the gyros. Note that this input vector does not include the force sensors or bump sensors that exist in the feet of the Nao robot. Though these sensors can be used to detect contact with the ground or other objects, they are typically too noisy to provide any useful information for the task of detecting instabilities robustly.

As with most sensory inputs on a mobile robot, the values obtained from the accelerometers and gyros are noisy. Prior research has shown that using a time series of data samples can help smooth out the noise. The input data stream is therefore

smoothed (i.e., averaged) along each dimension using an impulse response filter of the form:

$$v_{s,t} = \frac{k}{n} v_{s,t-1} + \frac{n-k}{n} v_{t,new} \quad (2)$$

where the smoothed value at time t, i.e., $v_{s,t}$ is determined as a function of the smoothed value in the previous time instant ($v_{s,t-1}$) and the new raw observation obtained at time t, i.e., $v_{t,new}$. The smoothed, i.e., the running average value is computed based on a window of n frames, and the parameter k can be modified to vary the effect that the raw observation $v_{t,new}$ has on the smoothed value.

Though the smoothing operation helps reduce the noise in the input data stream, the challenge with using these sensory values to obtain a predictive model is that it is time-consuming to get a sufficient number of labeled samples corresponding to the *stable* and *unstable* conditions. In addition, the sensory values can be different on each robot, and they can change over time even on a specific robot as a result of repeated use. There is hence a need to automate the collection of labeled data to learn a suitable predictive model. In the proposed approach, the data collection is automated by exploiting the fact that the accelerometer and gyro values corresponding to instabilities are significantly different from the values obtained when the robot is stationary or engaged in a stable motion, at least along one of the five dimensions of the input vector.

Consider Figure 3(a), which shows a the plot of the (smoothed) accelerometer values along the x-axis ("AccX") over a period of time. While the robot is walking normally, the $AccX$ values are close to the default value. The two large deviations in the observed values correspond to two instances where the robot displayed signs of instability. Figure 3(b) shows the corresponding accelerometer values along the y-axis. The $AccY$ plot shows deviations at similar points in the sequence but they are not as substantial as those along the x-axis. This specific data sample was generated when a humanoid robot slipped on the ground to cause a tilt (of its body) in the x-z plane. The key observation is that most instabilities result in measurably significant deviations in at least one of the observed accelerometer or gyro values, though the extent of deviation differs for the different instabilities. The robot can therefore autonomously generate labeled samples by looking for significant deviations along one or more dimensions of the 5D input vectors. When such a deviation is observed, the values within a fixed interval in *all five dimensions* are considered to represent *unstable* samples—the size of the interval is based on the size of the window used in Equation 2. An example of this automatic labeling is shown in Figure 3(c). Though the automatic labeling introduces some false positives and false negatives, this error is offset by the elimination of the effort involved in manually labeling the samples.

Once the robot has collected sufficient data samples corresponding to the two classes, the next step is to learn a predictive model that helps it classify subsequent samples as *stable* or *unstable*, thereby determining instances where the robot's motion is unstable. As mentioned above, the input vectors are points in 5D space, and the *unstable* class includes collisions, slippages and falls. As a result, the data collected by the robot is sampled from a complex distribution in 5D space. A robust classification scheme is therefore required to accurately partition this space. In this work, two different schemes are explored to learn the desired classifier: a decision-tree [13] and a Support-Vector Machine (SVM) [13], [14]. Experimental results show that the SVM-based model performs much better, and hence the corresponding theory is summarized below.

Given a set of training samples (i.e., instance-label pairs): $(\mathbf{x}_i, y_i), i \in [1, l]$, where $\mathbf{x}_i \in R^n$ and $\mathbf{y} \in \{0, 1\}^l$, SVMs compute the solution to an optimization problem, as defined by the following equation [15]:

$$min_{\mathbf{w},\xi} \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \quad (3)$$
$$\text{Subject to} \quad y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

where the function ϕ maps the input instances $\mathbf{x}_i$ to higher-dimensional space. In the current work, the sensory input vectors form the instances $\mathbf{x}_i$, while the labels $\mathbf{y}$ consist of *stable* and *unstable*, represented as '0' and '1' respectively. The SVM algorithm attempts to find linear hyperplanes that best separate the instances in this higher-dimensional space. The term $\mathbf{w}$ represents the normal to the hyperplane. The term C is the penalty for the error in classification, while ξ_i are the *slack variables* that represent the degree of misclassification for each individual instance. The input values are scaled in order to learn a more appropriate model—see [15] for details.

In SVM algorithms, the mapping of points to higher dimensions is typically performed using a *kernel function* that is defined by the following equation:

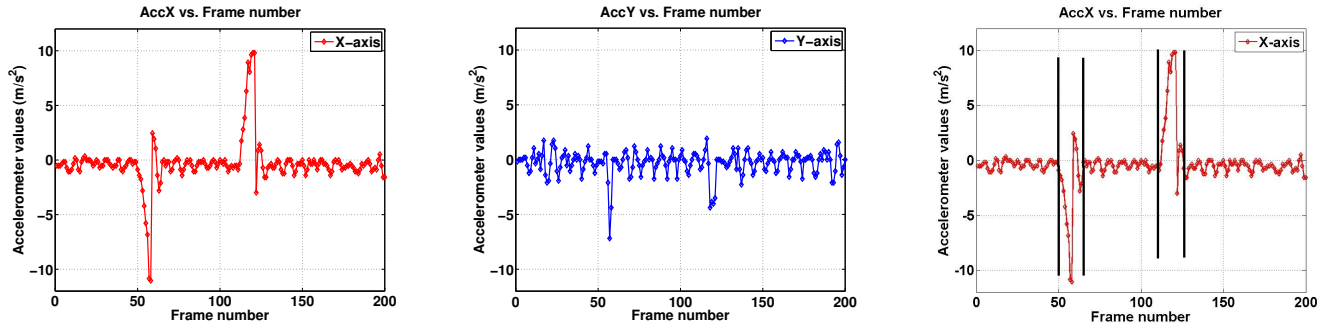
$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) \quad (4)$$

Some of the basic kernels used in SVM models are: linear, polynomial, radial basis functions (RBF) and sigmoid. The SVM model learned in this work is based on the RBF kernels:

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2), \ \gamma > 0 \quad (5)$$

where γ is a kernel parameter. RBF kernels are used because they are experimentally found to provide the best performance. The kernel projects the labeled instances to a higher dimension, and the hyperplane that best separates these instances is computed. New instances are then classified based on their location relative to this hyperplane.

Once an SVM model has been learned by each robot, it is used to predict occurrences of instabilities based on sensory inputs that have been smoothed as shown in Equation 2. When a robot detects that its motion is unstable, it: (a) tries to take corrective action; and (b) communicates the instability to its teammates. Currently, the robot chooses the corrective action from a set of predefined options, based on the task it is pursuing. For instance, it stops moving if it is not engaged in chasing the ball. More sophisticated dynamic balancing strategies [9] can be included here. In addition, the robot ensures the validity of the learned model by periodically



(a) AccX Plot. (b) AccY Plot. (c) Computed Bounds.

Fig. 3: (a) Sample plot of the accelerometer values along the x-axis; (b) Sample plot of the accelerometer values along the y-axis; (c) Computed intervals corresponding to the situation when the robot is unstable.

revising it with more current labeled samples. Furthermore, any undetected instability that leads to a fall (i.e., radically different sensor values) automatically triggers a getup routine and causes a new model to be learned.

In addition to the corrective action, each robot also uses the communicated instances of instability (from teammates) in its coordination strategy. The robots in our team perform different roles such as: *attacker, supporter, defender, keeper*, and combinations of these roles such as: *attacking supporter* and *defensive supporter*. Each robot (other than the goal keeper) dynamically determines its current role based on its estimate of the state of the game and the information provided by the teammates. The data communicated among teammates includes: (a) ball position; (b) robot pose (position and orientation); (c) opponent poses (if detected); and (d) current role and state. The robot's *state* includes information about the robot's actions (e.g., *kicking*). The role assignment has some simple constraints to ensure that the other team does not gain any advantage. For instance, there is always at least one robot chasing the ball. In addition, if a robot has not heard from its teammate for a long period of time or receives information that the teammate has had a fall, it assumes that the corresponding teammate does not exist and suitably revises its role choices. See [11] for details on the dynamic role assignment. However, other instabilities (e.g., wobble due to slippage) occur more frequently than falls, and the goal of this work is to robustly detect these instabilities and achieve effective collaboration.

In order to incorporate the stability information in the communicated data, the corresponding data structure is augmented with an entry that corresponds to the "physical state" of the robot. The physical state is currently represented using a single number that denotes the stability of the robot, i.e., $\{0, 1\}$ for $\{stable, unstable\}$. However, the framework can be readily extended to accommodate a probabilistic estimation of stability. As a result of this augmentation, each robot communicates detected instances of instabilities to its teammates. In order to give the teammate a chance to correct its instability (and to filter spurious detections of instability), a robot changes its coordination strategy only if a set of consecutive communicated vectors from a teammate indicate an instability. The robot responds to this situation by assuming the absence of the corresponding teammate. This modification to the existing coordination strategy results in robust operation in the presence of different types of instabilities.

III. Experimental Setup and Results

This section describes the experimental setup and the experimental results. The goal is to evaluate two hypotheses: (H_1) the learned predictive model accurately detects occurrences of instabilities in the robot's state; and (H_2) the communication of such instabilities between teammates and the incorporation of this information in the coordination strategy results in robust operation in the presence of instabilities.

In order to test H_1 (the first hypothesis), experiments were conducted in the natural game environment, with the humanoid robot playing soccer on the Standard Platform League soccer field. The robot therefore performs all the normal actions such as walking, kicking, falling down and recovering from a fall. While playing the game, the robot collects the corresponding accelerometer and gyroscope values in the form of 5D vectors—Equation 1. The collected values are filtered to smooth out the noise, as described in Equation 2, with $n = 20$. The robot analyzes the smoothed data stream for significant deviations along any of the five dimensions, and assigns appropriate labels to the input vectors: $\{0, 1\}$ for $\{stable, unstable\}$. Within a short period of time, the robot is able to collect many labeled samples. If the robot does not get sufficient *unstable* samples, it is possible to artificially place obstacles in the robot's path to induce the desired instabilities. However, no human intervention was required during the experiments to induce instabilities or modify the labels. In addition, bias is eliminated by selecting the same number of samples from each of the two classes to learn the predictive models. As mentioned in Section II-B, two different schemes were used: a decision-tree classifier and a SVM classifier. The J48 scheme [16] was used to learn the decision-tree classifier, while the libSVM code [15] was used to learn the SVM classifier after scaling the input values appropriately. The algorithms were revised to enable real-time execution on the humanoid robot. The learned predictive models were then

Approach	False positives (%)	False negatives (%)
J48 decision trees	5.7	8.1
RBF SVMs	0.5	0.7

TABLE I: False positive and false negatives using the two different classification schemes. SVM-based classifier performs much better.

used to perform online classification of the smoothed sensory input vectors on the robot.

In order to make the experiments more challenging, trials were conducted on three different Nao robots while performing different types of walks (i.e., different walk parameters) in different directions at different speeds. The trials also included different surfaces (robot soccer field, tiled floor and carpeted floor). The robot autonomously learned appropriate models for the different walks and surfaces, and these models were evaluated on instabilities induced by injecting obstacles in the robot's path. The results averaged over a set of ≈ 200 (total) trials are shown in Table I.

The results in Table I show the false-positive and false-negative rates of the classification schemes, i.e., smaller numbers in the table indicate a better classification scheme. In other words, the SVM-based scheme performs much better than the decision-tree classifier, with hardly any false positives or false negatives. In addition, even the observed errors did not occur for more than a frame at a time. The robot therefore does not take any unnecessary corrective action. The SVM technique provides good performance because it uses a RBF kernel to project data into a higher dimension where it is easier to separate the vector clusters corresponding to stable motion from those corresponding to the instabilities. Figures 4(a)–4(d) show snapshots taken during an experimental trial—the robot is pushed in different directions and the LEDs in the chest and eyes are turned on to indicate the instability. Currently the robot stops its motion or performs predefined motions to compensate for the instabilities, but more sophisticated balancing strategies can also be used.

Next, a multirobot coordination task was setup in order to test H_2, i.e., the hypothesis that using the communicated estimates of instabilities results in better coordination. First, a team of two robots was tasked with playing soccer with stationary opponents, i.e., two robots were placed at fixed points on the field. The performance of the team was evaluated by computing the number of goals scored over a period of five minutes. Once a goal was scored, the robots were informed and the ball was placed at the center of the field. The results over a set of 15 trials were averaged to obtain the numbers reported in Table II. The trials were first run without the proposed approach, i.e., the robots did not take corrective action for instabilities and did not use the communicated instabilities in the decision-making process. However, occurrences of falls are detected (the robot performs a getup routine), communicated and used in the decision-making. Then the trials were repeated with the robot using the proposed approach. As shown in the first row of Table II, detecting different types of instabilities and using the communicated instances of instabilities in the coordination strategy results in better performance—the team scores twice as many goals.

Situation	Goal difference	
	Default approach	New approach
Stationary opponents	3	6
Moving Opponents	1	4

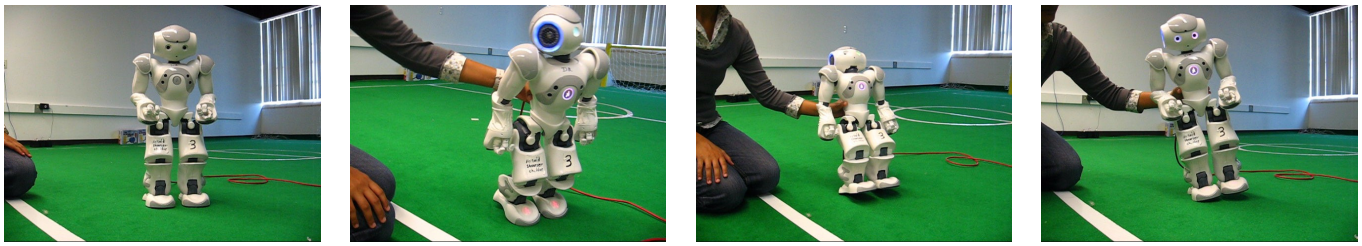
TABLE II: Results of using the proposed strategy in a multirobot scenario. The team scores more goals when the communicated estimates of instabilities are included in the coordination strategy.

Next, a similar set of trials were run with a team of two robots playing against moving opponents, i.e., another team of two robots. Each trial now lasted ten minutes and the difference in the goals scored by the two teams was used as a measure to evaluate the performance. Once again, the trials were first conducted without the proposed strategy, i.e., both teams used the same software. In this case, there was no significant difference in the performance of the two teams, as noted in Table II. Next, the robots in one team were allowed to use the proposed approach to detect instabilities and to include it in the team coordination strategy. This team is now able to coordinate better despite instabilities caused due to collisions or slippages. As shown in Table II, this team ends up scoring a significantly larger number of goals against the opponents who used the default coordination strategy. The results documented above show that the proposed strategy enables each robot to robustly detect (and account for) instabilities, and that it enables a team of robots to use the communicated estimates of instabilities to revise the coordination strategy.

IV. Related Work

There has been a significant amount of research on stable locomotion and team coordination for mobile robots. For instance, Fierro et al. [17] proposed an architecture using vision and other sensors to build a control system for robust multirobot coordination. The work described in this paper can be used in conjunction with such architectures to improve the robot's real-time performance while collaborating with other robots. In the robot soccer setting, most teams follow a collaborative strategy that involves assigning roles to robots dynamically based on the state of the game [10], [11].

On legged robots, stability continues to be a major requirement for robust coordination. Mericli et al. [18] proposed a method that used temporal accelerometer readings and interpreted them in the frequency domain to identify unstable states on quadruped robots (Aibos). On the other hand, Quinlan et al. [19] described an approach based on traction monitoring for collision detection on legged robots. Their method used the joint positions and calculated the deviations from the expected values to determine the occurrence of collision. However, joint values are typically noisy and not a good factor to differentiate between normal motion and small instabilities. There has also been considerable work on designing omni-directional walks that are more stable [10]. However, the occurrence of collision and slippage cannot be avoided. Instability is all the more important on biped platforms because recovery from falls can be a time-consuming process. More recently, Mericli and Veloso [20] proposed an approach based on real-time corrective feedback, which can be mapped to the changes



(a) Image 1 (stable). (b) Image 2 (unstable). (c) Image 3 (unstable). (d) Image 4 (unstable).

Fig. 4: (a) Snapshot of a robot during a stable walk; (b)-(d) Three snapshots of the detected instabilities. Here the robot is pushed in different directions as it is walking on the field. Detected instabilities are indicated by lighting up the LEDs in the eyes and chest.

in sensor values when the robot is unstable. In dynamic domains such as robot soccer, supervised learning is not the most appropriate approach because different models may be required for different robots and different surfaces. This paper therefore presents an approach that enables a humanoid robot to autonomously learn predictive models based on samples collected online, and to use these models in the team coordination strategy.

V. Conclusions

Stable motion is a key requirement for humanoid soccer robots. However, given the dynamic nature of the domain, collisions, slippages and other instabilities cannot always be avoided despite extensive research in stable bipedal locomotion and collision avoidance. In addition, the time spent in recovering from such instabilities can provide a significant advantage to the opponents, unless the other robots in the team re-organize themselves appropriately to perform the desired roles. This paper described a method that enables a humanoid robot to learn a predictive model in order to robustly detect all occurrences of instabilities. The robot automatically generates labeled samples to learn the parameters of this model. Furthermore, each robot uses the communicated estimates of such instabilities in the team coordination strategy.

The SVM models learned in this work use the data obtained from the accelerometers and gyroscopes. The input vector can be augmented to include other sensors such as the ultrasound sensors (or even vision). The robot will then build a model that detects and avoids situations that are likely to result in a collision or fall, *and* responds to the unavoidable instabilities caused by collisions and slippages. The goal would be prevent the falls that cause physical damage and require a time-consuming recovery process. A dynamic balancing strategy can also be incorporated to enable smoother recovery from such instabilities. Using such a rich representation of sensory inputs and a better recovery scheme will also contribute towards a more robust team coordination strategy. Ultimately, the goal is to enable a team of humanoid robots to collaborate effectively in the presence of a variety of instabilities and changes in the environment.

Acknowledgment

The authors would like to thank the members of the TT-UT Austin Villa team for their efforts in developing the soccer-playing software used to run the experiments reported in this paper. This research was supported in part by the ONR Science of Autonomy award N00014-09-1-0658.

References

[1] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "Robocup: The Robot World Cup Initiative," in *ICRA*, February 1997, pp. 340–347.
[2] SPL, "The robosoccer standard platform league," 2008, www.tzi.de/spl/.
[3] Nao, "The aldebaran nao robots," 2008, www.aldebaran-robotics.com/.
[4] J. Casper and R. R. Murphy, "Human-robot interactions during urban search and rescue at the wtc," in *Transactions on SMC*, 2003.
[5] M. A. Goodrich and A. C. Schultz, "Human-Robot Interaction: A Survey," *Foundations and Trends in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007.
[6] J. Hoey, P. Poupart, A. Bertoldi, T. Craig, C. Boutilier, and A. Mihailidis, "Automated Handwashing Assistance for Persons with Dementia using Video and a Partially Observable Markov Decision Process," *Computer Vision and Image Understanding*, vol. 114, no. 5, pp. 503–519, 2010.
[7] S. Thrun, "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.
[8] J. Pratt and B. Krupp, "Design of a Bipedal Walking Robot," in *Proceedings of the SPIE*, 2008.
[9] J. Rebula, F. Canas, J. Pratt, and A. Goswami, "Learning Capture Points for Humanoid Push Recovery," in *ICHR*, 2007.
[10] T. Röfer and Others, "B-human team report and code release," 2010, only available online: http://www.b-human.de/file_download/33/bhuman10_coderelease.pdf.
[11] T. Hester, M. Quinlan, and P. Stone, "UT Austin Villa 2008: Standing on Two Legs. Technical Report UT-AI-TR-08-8," The University of Texas at Austin, Tech. Rep., 2008.
[12] M. Sridharan and X. Li, "Autonomous Information Fusion for Robust Obstacle Localization on a Humanoid Robot," in *International Conference on Humanoid Robots*, 2009.
[13] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2008.
[14] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
[15] C.-C. Chang and C.-J. Lin, *LIBSVM: A Library for Support Vector Machines*, 2001, software available at: http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[16] R. J. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
[17] R. Fierro, A. Das, J. Spletzer, J. Esposito, V. Kumar, J. P. Ostrowski, G. Pappas, C. J. Taylor, Y. Hur, R. Alur, I. Lee, G. Grudic, and B. Southall, "A Framework and Architecture for Multi-Robot Coordination," *International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 977–995, October-November 2002.
[18] T. Mericli, C. Mericli, and H. L. Akin, "A Robust Statistical Collision Detection Framework for Quadruped Robots," in *RoboCup 2008: Robot Soccer World Cup XII*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 145–156.
[19] M. J. Quinlan, C. L. Murch, R. H. Middleton, and S. K. Chalup, "Traction Monitoring for Collision Detection with Legged Robots," in *RoboCup 2003 Symposium*. Springer, 2003, pp. 374–384.
[20] C. Mericli and M. Veloso, "Biped Walk Learning Through Playback and Corrective Demonstration," in *International Conference on Artificial Intelligence*, 2010.