2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Detecting Anomalies in Humanoid Joint Trajectories

Fernando Marcolino and Jiuguang Wang

***Abstract*— We present a semi-supervised anomaly detection system for humanoid robots that operates on trajectories with varying lengths, resolutions, and time shifts. The proposed approach utilities optimization to extract a model from joint trajectories under normal operation and seek to identify anomalous behaviors that deviates significantly from the known model. Compared to previously proposed approaches in humanoid anomaly detection that identified only high-level faults, our approach can detect subtle defects in the robot and at the same time, is capable of generalizing to higher-level behaviors. The system is demonstrated on a simulated model of the Atlas humanoid robot, with several experimental scenarios demonstrating detection of both joint-level anomalies and behaviors such as falling.**

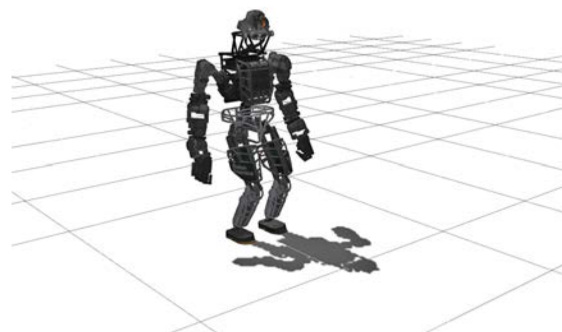
***Index Terms*— Humanoid robots, anomaly detection, optimization**

I. Introduction

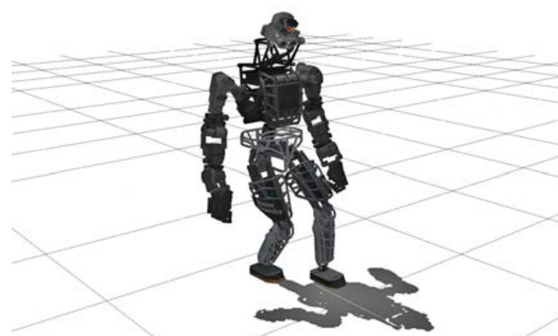
One significant barrier to the practical applications of humanoid robots in everyday environments is safety. Compared to statically stable robots, a dynamically balancing bipedal humanoid is an inherently unstable structure. Combined with heavy structural elements and high-powered actuators, any failure in a humanoid robot can lead to catastrophic physical damage to its surroundings. Therefore, for a humanoid robot to be practically useful, it is necessary to implement a system of error detection so that any failures in sensors or actuators can be detected.

Anomaly detection for humanoid robots can be simplified to the problem of detecting deviations in locomotive behaviors. By having a humanoid robot execute a known behavior (such as walking), an operator can compare the resulting trajectories against a collection of previously obtained data to discern any anomaly. This is similar to the Field Sobriety Tests utilized by law enforcement to identify drunk drivers through behaviors such as "walk-and-turn". For our task, an automatic anomaly detection system is highly desirable because manually identifying anomalous trajectories would be prohibitively expensive in a high degrees-of-freedom system such as a humanoid robot.

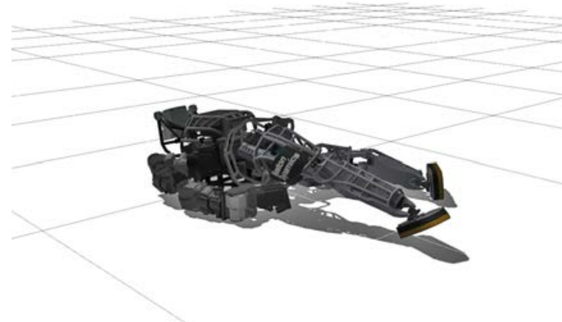
While there exist an extensive collection of anomaly detection methods in other fields of engineering [1], few can be directly applied to humanoid robots. Supervised methods require labeled datasets, which are difficult to obtain due to both the large degrees-of-freedom in a humanoid and the wide range of potential faulty behaviors. Most existing unsupervised and semi-supervised methods require training data to be well-organized, but in humanoid robots, multiple executions of the same experiment can have different start and end times, causing the trajectories to be time-shifted and have different lengths. Measurement outputs can also be irregular due to network delays and other factors, causing the resolution of the trajectories to be varied. These issues



(a) Balancing



(b) Walking



(c) Fallen

Fig. 1. A model of the Atlas humanoid robot in the Gazebo simulation environment. Our detector is designed to be a calibration tool where the robot executes a known behavior in order to identify anomalies in the system, both for low-level anomalies (degradation in the actuators) and high-level behaviors (falling).

Fernando Marcolino is with the Universidade do Estado da Bahia, Salvador, Brazil. Email: `fmarcolino2005@gmail.com`

Jiuguang Wang is with the Robotics Institute at Carnegie Mellon University, Pittsburgh, Pennsylvania, 15213, USA. Email: `robot@cmu.edu`

must be addressed in a successful anomaly detection system for humanoid robots.

The main contribution of this work is a semi-supervised anomaly detection system for humanoid robots that operates on trajectories in the joint space. It is designed to be a calibration tool where the robot executes a known behavior in order to identify anomalies in the system. The detector is semi-supervised in the sense that we train the model using good trajectories generated under normal operation of the robot and classify a new trajectory as anomalous if it deviates considerably from this model. While identification of faulty behaviors such as tripping, slipping, and falling has been addressed in previous work in this domain, we focus on detecting subtle defects in the robot due to degradation and miscalibration, which manifest themselves as small joint-level deviations. In addition, our detector is capable of identifying anomalies in trajectories that are incomplete, shifted, and varying in resolution, encountered frequently in real-world experimental data for humanoids. Finally, we demonstrate that the proposed detector can also be generalized to detecting high-level behaviors such as falling.

The remainder of this paper is organized as follows. Section II gives an overview of existing methods of anomaly detection. Section III describes the basic formulation of the proposed detector in two parts, model extraction and anomaly detection. Section IV gives the experimental results of the proposed approach as validated on a simulated Atlas humanoid robot. Section V contains concluding remarks and directions for future work.

II. Related Work

Several previous studies in humanoid robots have proposed approaches that detect faulty behaviors such as falling. Most notably, Renner [2], Karssen [3], and Kalyanakrishnan [4] have proposed strategies for instability and fall detection using probabilistic and machine learning-based methods. While these approaches are useful for specific behavior-based control design (for example, designing falling control in Wang [5]), they are limited to detecting a single faulty behavior and producing a binary decision. More recently, Lynch [6] have proposed a computational geometry based method for humanoid fault detection, but the approach was unable to address potential spatial and temporal shifts in the trajectory data, limiting its utility to extremely well-defined training data.

Previous approaches to the problem of fault detection in engineering can be broadly classified as signature-based or anomaly-based. In signature-based system such as Brodie [7], runtime data was actively checked against a library of known faulty states. While this approach is effective in applications where the range of possible faulty states is limited, it cannot be used with humanoid robots as the space of potentially faulty behaviors is very large. In anomaly-based systems, it is assumed that a collection of normal behaviors for a given system is available, and large deviations in the runtime data indicates a fault has occurred. Instead of simply matching the symptoms of faulty behaviors, anomaly detection requires building a model of correct behaviors and a way of classifying runtime behaviors against this model.

Many approaches have been proposed for anomaly detection using clustering-based techniques, for example, in Fu [8] and Hu [9] where trajectories were classified based on similarity metrics, possibly using kernel transformations [10]. While simple to implement, these approaches require well-organized trajectories of equal length and uniformly sampled in time. Goel [11] and Hu [12] have trained neural networks as a base model, but this and other black-box models such as Jakubek [13] suffer from a lack of transparency. Morris [14] and Bashir [15] used Hidden Markov Models which worked well for small trajectory segments, but the time complexity of the forward-backward algorithm for training the HMM is cubic in terms of the length of the trajectory, which made the approach difficult to apply for large trajectories. Similarly, Rosen [16] used an nonlinear optimization to obtain a parametric model, which relied on expensive numerical solutions. Other works such as Piciarelli [17] identified only complete trajectories and was unable to label specific samples as anomalous.

III. Approach

A. Model extraction

We model a humanoid robot as a planar rigid-body system in the sagittal plane whose equations of motion can be written as a series of first-order nonlinear equations

$$\dot{q} = f\left(q, u\right), \tag{1}$$

where $q \in \mathbb{R}^{n_q}$ defined in generalized coordinates is the state of the system comprised of joint angles and velocities for the robot; $u \in \mathbb{R}^{n_u}$ contains the actuator commands (torques) that are the controlled inputs to the system; together, q and u are referred to as the trajectory of the system.

We define the trajectory of the system to be a new vector

$$x = \left[\begin{array}{c} q \\ u \end{array}\right] \in \mathbb{R}^{n_q + n_u}, \tag{2}$$

where we use the subscript x_j to represent a row in this concatenated vector that is a component of the trajectory.

Each component of the trajectory x_j contains a series of measured data points

$$x_j = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_i\} \in \mathbb{R}^{n_{x_j}}, n_{x_j} \le T, \tag{3}$$

where the samples α_i are obtained for the duration of the trajectory T. Here, we do not assume x_j to be sampled uniformly and hence the lengths n_{x_j} are different for each x_j.

Let X denote a collection of k trajectories

$$X = \left\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(k)}\right\} \in \mathbb{R}^{n_X}, \tag{4}$$

where we use the superscript $x^{(k)}$ to represent a single trajectory in the collection. Combining notations, we use α_{ijk} to denote the i-th sample from the j-th component of the k-th trajectory.

Given X, we seek to determine a model $r(t) \in \mathbb{R}^{n_q+n_u}$ with components $r_j(t)$ that represents the k trajectories as a single parametric function where $0 \leq t \leq 1$. Let us assume that $r_j(t)$ takes the form

$$r_j(t) \approx \sum_{i=1}^{n_{x_j}} \psi_i \phi_i(t) = \Psi\Phi(t), \qquad (5)$$

where $\Phi(t)$ is a set of basis functions and Ψ is a set of corresponding coefficients. To determine Ψ, we formulate an optimization problem to minimize the sum of the squared distances between the points in the trajectories α_{ijk} and $r_j(t)$, in the form

$$\min_{\Psi} \quad \sum_{k=1}^{n_X} \sum_{i=1}^{n_{x_j}} \lambda_i \|r_j(t) - \alpha_{ijk}\|^2 + \sum_{m=1}^{p} \int_C \kappa_m \left\| \frac{d^m r(t)}{dt^m} \right\|^2 dt, \qquad (6)$$

with an additional integral that is a regularization term of the order p. Here, κ_m are nonnegative weights for the regularization terms and λ_i are positive weights. In this form, computing the coefficients Ψ is then a sequential quadratic programming (SQP) problem [18]. We take a similar approach but formulate the optimization problem differently to yield a more efficient solution.

For clarity, we simplify the notations below by dropping the subscripts to demonstrate how the parametric model can be obtained for a single trajectory component. Let α_i denote a sample obtained at time t_i and r^* represent the model constructed from a collection of N samples in time T.

First, we assume that the parametric representation of the model r^* to be in the form

$$\begin{aligned} \dot{x}^* &= Ax^* + Bu^* \\ r^* &= Cx^* \end{aligned} \qquad (7)$$

that is both controllable and observable [19]. Here, the matrices A, B, and C are non-unique and a canonical example is

$$A = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ a_1 & a_2 & a_3 & \dots & a_n \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}. \qquad (8)$$

This form of the system matrices along with the relative degree assumption

$$CB = CAB = CA^2B = CA^{n-2}B = 0 \qquad (9)$$

ensures maximal smoothness for the output. The control theoretic interpretation of (7) is to design a control law u^* that "drives" an output trajectory close to a sequence of data points α_i at fixed times. This is in fact a smoothing spline cast as an optimal control problem where we seek to minimize the distance between the trajectory and the data, which takes the form

$$\min_{u^*} \quad J(u^*) = \sum_{i=1}^{N} w_i (r^*(t_i) - \alpha_i)^2 + \rho \int_0^T {u^*}^2(t)\, dt, \qquad (10)$$

subject to an affine constraint

$$r^*(t) = Ce^{At}x_0^* + \int_0^t Ce^{A(t-s)}Bu^*(s)\, ds, \qquad (11)$$

where with $x_0^* = 0$, w_i and ρ are positive weights. The first term in (10) measures the distance between the solution $r^*(t)$ and the given data α_i. The integral term in (10) and the affine constraint in (11) ensures smoothness of the solution. This interpretation is very similar to (6) but yields a more efficient solution described below.

To simplify the subsequent notations, let

$$g_{t_i}(t) = \begin{cases} Ce^{A(t_i - t)}B & t \leq t_i \\ 0 & \text{otherwise} \end{cases} \qquad (12)$$

and $L_{t_i}(u^*)$ be a linear operator

$$L_{t_i}(u^*) = \int_0^T g_{t_i}(t) u^*(t)\, dt. \qquad (13)$$

Rewriting (11), we have

$$r^*(t_i) = \int_0^T g_{t_i}(t) u^*(t)\, dt = L_t(u^*). \qquad (14)$$

Similarly for the objective function, we have

$$J(u^*) = \sum_{i=1}^{N} w_i (L_{ti}(u^*) - \alpha_i)^2 + \rho \int_0^T {u^*}^2(t)\, dt. \qquad (15)$$

This optimization problem was shown to yield a unique, global minimum in [20], in the form of the necessary condition

$$\sum_{i=1}^{N} w_i g_{ti}(t)(L_{ti}(u^*) - \alpha_i) + \rho u^*(t) = 0. \qquad (16)$$

The solution of $u^*(t)$ takes of the form

$$u^*(t) = \sum_{i=1}^{N} \tau_i g_{ti}(t), \qquad (17)$$

where $\tau = (\tau_1, \tau_2, \dots, \tau_N)$ is a set of coefficients, which can be determined by solving a set of linear equations

$$(DG + \rho I)\tau = 0. \qquad (18)$$

Here, D is a diagonal matrix of weights and G is the Gramian

$$G = \begin{pmatrix} L_{t_1}(g_{t_1}) & \dots & L_{t_1}(g_{t_N}) \\ \vdots & & \vdots \\ L_{t_N}(g_{t_1}) & \dots & L_{t_N}(g_{t_N}) \end{pmatrix} \qquad (19)$$

Effectively, (17) and (18) gives the final $r^*(t)$, which can be modified to take into account multiple trajectories in the form of (5).

B. Anomaly detection

Given the parametric model $r(t)$, we can compute the tangent direction $w(t)$ by

$$w(t) = \frac{dr(t)}{dt} \Big/ \left\| \frac{dr(t)}{dt} \right\| . \tag{20}$$

Let $n(t)$ be the normal vector orthogonal to $w(t)$. A data point in the trajectory α_i is similar to $r(t_i)$ if

$$(r(t_i) - \alpha_i) \perp w(t_i) \tag{21}$$

and

$$\|r(t_i) - \alpha_i\| \leq \delta. \tag{22}$$

The first condition implies that the point lies on the normal vector and the second condition implies that the distance from the point to $r(t)$ is within a predetermined width δ.

Given the parametric model and a metric, there are many methods available to classify whether a new observation is an anomaly. We use the simple hypothesis testing framework described in [16]. Let ζ be a set of points that are in a similar position to $r(t)$. Let μ and σ be the mean and covariance of ζ. We label a point α_i as anomalous if

$$(\alpha_i - \mu)^T \sigma^{-1} (\alpha_i - \mu) > \chi^2_\beta, \tag{23}$$

where χ^2_β is the p-value at a significance level of β for a χ^2 distribution. We assume that the data follow a multivariate normal distribution.

IV. EXPERIMENTAL RESULTS

A. Setup

To validate the proposed anomaly detection system, we designed several experiments in the Gazebo simulation environment with a model of the Atlas humanoid robot (Fig. 1) used in the DARPA Robotics Challenge. The simulation environment and the associated models can be found in the *DRCsim* package, which is freely available on the web [21]. The simulation data and animations can be found at [22].

As a baseline, we designed a simple forward walking behavior and used 10 recorded joint trajectories to train the detector. In the results below, we illustrate detecting anomalies in inertial properties, actuator parameters, and high-level behaviors such as falling.

We used a simplified model of the robot that contained 32 degrees of freedom with a 96-dimensional state trajectory (torque inputs & measured joint positions and velocities), with other computed values such as the positions and velocities of the center-of-mass (CoM). Because the upper body of the robot was mostly unused during basic walking, we focused on detecting anomalies in the lower body. In the experiments below, we show a subset of interesting trajectories in the full state detector.

All experiments were conducted on a 64-bit system using a Core i7 2.8GHz machine with 16GB of RAM. While the model extraction step was meant to be executed offline, it can be done in real time for sufficiently small number of training trajectories (10 trajectories in the experiments below).

B. Detecting changes in inertial properties

We designed a simple periodic walking behavior where the robot began by standing at rest, took a prescribed number of steps forward, then came to a complete stop. Fig. 2 shows the trajectory of the link *l_lleg* in the left leg of the Atlas robot under normal walking and the torque output for the corresponding joint *l_leg_kny* under a 1 kHz control loop. The trajectories have different resolutions since there are missing measurements throughout the data. Because the start and end time of multiple trials are slightly different, the trajectories also become time-shifted and truncated differently.

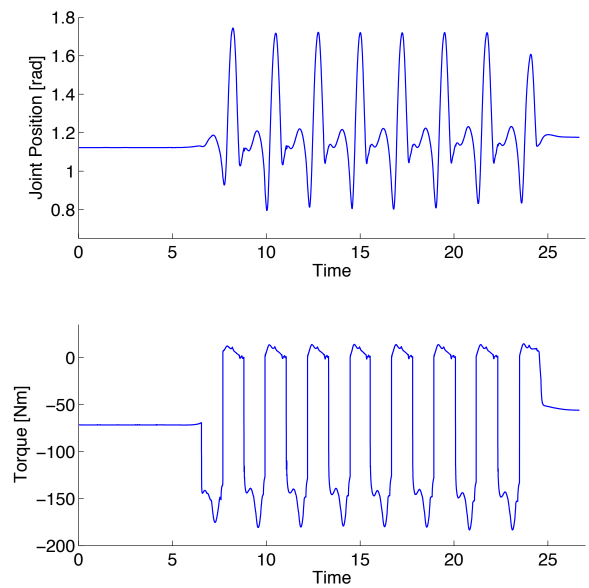


Fig. 2. Trajectory of the link *l_lleg* in the left leg of the Atlas robot under normal walking and the torque output for the corresponding joint *l_leg_kny*.

After obtaining a set of 10 normal walking trajectories, we introduced an anomaly by artificially tripling the mass of the *l_lleg* link from $4.367kg$ to $13.101kg$. This represented a physical change to the robot unknown to the controller that is not severe enough to to be easily detected visually by examining the walk. As shown in the outputs Fig. 3 and Fig. 4, the walking controller compensated by increasing the torque output on the joint *l_leg_kny*, which resulted in a slightly different walk. By looking at the trajectories at the joint level, the detected anomalous trajectory (red) is easily distinguished from the normal profiles (blue).

C. Detecting changes in actuator parameters

In this scenario, we altered the parameters for the actuator at the joint *l_leg_kny* by increasing the damping coefficient (from 1 to 20) and adding friction (from 0 to 20). This represented a typical degradation in the actuators and is also

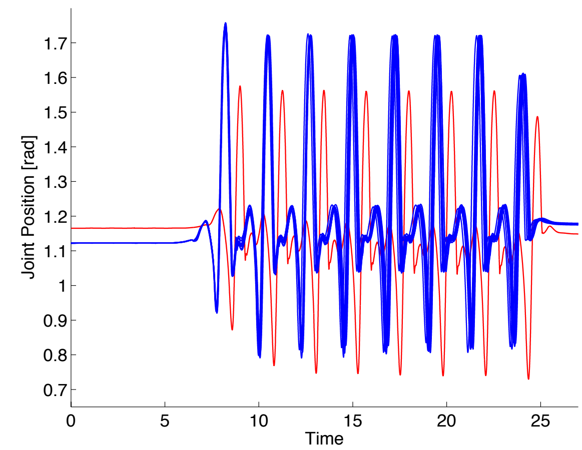


Fig. 3. Blue: 10 trajectories of the position (rad) of the joint *l_lleg*. Red: detected anomaly after tripling the mass of the *l_lleg* link from $4.367kg$ to $13.101kg$.

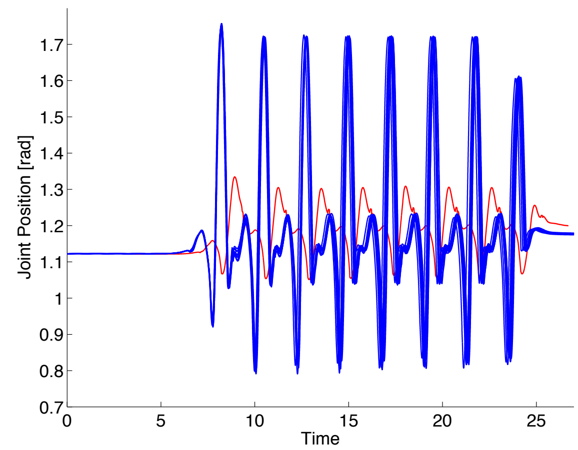


Fig. 5. Simulated anomaly in the joint *l_leg_kny* after increasing the damping coefficient (from 1 to 20) and adding friction (from 0 to 20). Detected anomaly (red) in the profile of the joint positions (rad) in the *l_lleg* link.

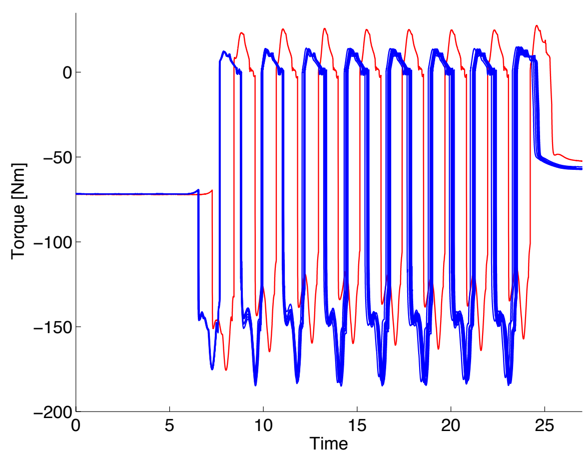


Fig. 4. Blue: 10 trajectories of the torque output of *l_leg_kny* (Nm). Red: detected anomaly after tripling the mass of the *l_lleg* link from $4.367kg$ to $13.101kg$.

a small variation that cannot simply detected through the high-level behaviors of the robot. Fig. 5 shows the result of the detection where in the joint positions of the *l_lleg* link, the anomaly (red) is easily distinguished from the normal profile (blue).

D. *High-level detection - falling due to tripping*

The previous example of anomaly detection in the joint space of the robot was geared primarily towards low-level detection of miscalibrated sensors and degraded actuators. The proposed approach is also capable of identifying high-level behaviors by examining the relevant parameters. Here, we give an example of identifying falling while walking by implementing the proposed detector on the CoM position, which is a trajectory calculated using the joint angles and the forward kinematics of the robot, assuming one foot on the ground.

Fig. 6 shows the vertical CoM (COM-y) position for the walking gait, which followed a sinusoidal pattern as the CoM sways. 10 trajectories were generated to train the detector (shown in blue), which was then used to identify a fall (red), where there is a significant downward movement in the CoM-y position until the robot was on the ground. Fig. 7 shows the corresponding plot for the CoM velocity in the vertical direction.

V. Conclusions

In this paper, we presented a method of anomaly detection for humanoid robots that operated on trajectories in the joint space. Unlike previously proposed systems in this domain that emphasized detection for high-level faults such as tripping, slipping, and falling, we focused on detecting subtle defects in the robot due to degradation and miscalibration, which appeared in the form of joint-level deviations. Using a semi-supervised system which extracted a model using optimization, our system was able to train on trajectories that were incomplete, shifted, and varying in resolution, encountered frequently in real-world experimental data for humanoids. Through the experimental results, we demonstrated detection of both joint-level anomalies and behaviors such as falling.

Our approach was semi-supervised in that we only trained the detector on good data, which required the operator to ensure that the robot was perfectly calibrated at the time of acquiring the training set. This is not always a realistic assumption and one future direction is to improve the robustness of the detector with respect to bad data in the training set. In addition, we have not addressed the use of unlabeled data in the model extraction, which could further be used to enhance the availability of data and make the training setup easier.

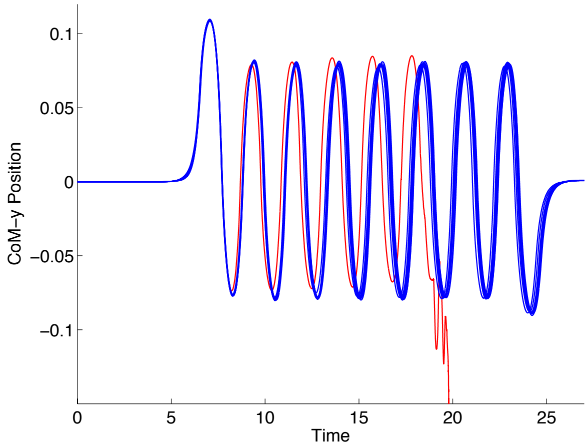


Fig. 6. The vertical Center-of-Mass (CoM-y) position during walking. Detected anomaly (red) as the robot falls to the ground, resulting in a large downward movement in the CoM-y position.

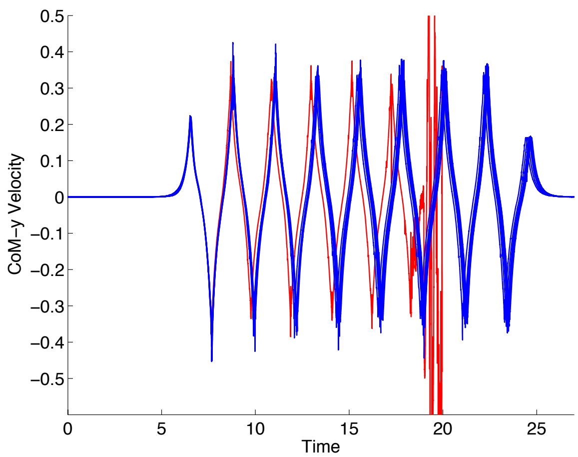


Fig. 7. Velocity of the vertical Center-of-Mass (CoM-y) during walking. Detected anomaly (red) as the robot falls to the ground, resulting in a large downward movement in the CoM-y velocity.

Our immediate future goal is to implement the proposed method on a real humanoid robot. While the simulation results we have presented here contained very realistic data, real-world experiments contain a wider range of noise, disturbances, discontinuities, etc. Further testing is needed to validate the performance of the proposed anomaly detector in the context of these external factors.

Acknowledgments

We thank Weiwei Huang and Siyuan Feng for their assistance in setting up the simulation of the Atlas humanoid and other colleagues at CMU for their valuable comments on an earlier version of the paper.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, 2009.
[2] R. Renner and S. Behnke, "Instability Detection and Fall Avoidance for a Humanoid Using Attitude Sensors and Reflexes," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 2967–2973.
[3] J. G. D. Karssen and M. Wisse, "Fall Detection in Walking Robots by Multi-Way Principal Component Analysis," *Robotica*, vol. 27, no. 02, pp. 249–257, 2009.
[4] S. Kalyanakrishnan and A. Goswami, "Learning to Predict Humanoid Fall," *International Journal of Humanoid Robotics*, vol. 8, no. 2, pp. 245–273, June 2011.
[5] J. Wang, E. C. Whitman, and M. Stilman, "Whole-Body Trajectory Optimization for Humanoid Falling," in *American Control Conference*, 2012, pp. 4837–4842.
[6] K. Lynch, D. M. Lofaro, and P. Oh, "A N-Dimensional Convex Hull Approach for Fault Detection and Mitigation for High Degree of Freedom Robots Humanoid Robots," in *International Conference on Control, Automation and Systems*, 2012, pp. 790–797.
[7] M. Brodie, S. Ma, G. Lohman, L. Mignet, M. Wilding, J. Champlin, and P. Sohn, "Quickly Finding Known Software Problems via Automated Symptom Matching," in *International Conference on Autonomic Computing*, 2005, pp. 101–110.
[8] Z. Fu, W. Hu, and T. Tan, "Similarity Based Vehicle Trajectory Clustering and Anomaly Detection," in *IEEE International Conference on Image Processing*, 2005, pp. II–602–5.
[9] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A System for Learning Statistical Motion Patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1450–1464, 2006.
[10] R. Fujimaki, T. Yairi, and K. Machida, "An Approach to Spacecraft Anomaly Detection Problem Using Kernel Feature Space," in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2005, pp. 401–410.
[11] P. Goel, G. Dedeoglu, S. I. Roumeliotis, and G. Sukhatme, "Fault Detection and Identification in a Mobile Robot Using Multiple Model Estimation and Neural Network," in *IEEE International Conference on Robotics and Automation*, 2000, pp. 2302–2309 vol.3.
[12] W. Hu, D. Xie, T. Tan, and S. Maybank, "Learning Activity Patterns Using Fuzzy Self-Organizing Neural Network," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 34, no. 3, pp. 1618–1626, 2004.
[13] S. Jakubek and T. Strasser, "Fault-Diagnosis Using Neural Networks with Ellipsoidal Basis Functions," in *American Control Conference*, 2002, pp. 3846–3851 vol.5.
[14] B. T. Morris and M. M. Trivedi, "Learning, Modeling, and Classification of Vehicle Track Patterns From Live Video," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 425–437, 2008.
[15] F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "Object Trajectory-Based Activity Classification and Recognition Using Hidden Markov Models," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1912–1919, 2007.
[16] O. Rosen and A. Medvedev, "An On-Line Algorithm for Anomaly Detection in Trajectory Data," in *American Control Conference*, 2012, pp. 1117–1122.
[17] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Anomalous Trajectory Patterns Detection," in *International Conference on Pattern Recognition*, 2008, pp. 1–4.
[18] L. Fang and D. C. Gossard, "Multidimensional Curve Fitting to Unorganized Data Points by Nonlinear Minimization," *Computer-Aided Design*, vol. 27, no. 1, pp. 48–58, 1995.
[19] W. L. Brogan, *Modern Control Theory*, 3rd ed. Prentice Hall, 1990.
[20] S. Sun, M. B. Egerstedt, and C. F. Martin, "Control Theoretic Smoothing Splines," *IEEE Transactions on Automatic Control*, vol. 45, no. 12, pp. 2271–2279, 2000.
[21] [Online]. Available: http://gazebosim.org/wiki/DRC/
[22] [Online]. Available: http://jw.nebulis.org