

Activity Monitoring and Prediction for Humans and NAO Humanoid Robots using Wearable Sensors

Saminda Abeyruwan and Faisal Sikder and Ubbo Visser and Dilip Sarkar

University of Miami

Department of Computer Science

1365 Memorial Drive, Coral Gables, FL, 33146, USA

{saminda|f.sikder|visser|sarkar}@cs.miami.edu

Abstract

While humans or biped humanoid robots perform activities such as jogging and running, an accident event such as a fall may occur. This might cause damage to the human body or to the structural components of the robot. For humans, immediate identification of a fall will allow fast responses, while for a robot, early prediction can be used to take corrective measures to prevent a fall. Modern wireless sensing devices can be attached to humans or robots to collect motion data. We propose: 1) methods to learn and predict different activities for humans and robots; and 2) software tools to realize these functions on embedded devices. Our contributions include: 1) detection of falls for both humans and robots within a unified framework; and 2) a novel software development environment for embedded systems. Our empirical evaluations demonstrate that with high accuracy different types of fall-events are predicted using the same learning algorithms for humans and biped humanoid robots.

Introduction

The humans as well as the biped humanoid robots complete apparently simple activities — such as, jogging, and running etc. — that requires complex computational tasks. While performing these routine activities, accidents such as falls may occur causing damage to the human body or to the structural components of the humanoid robot (Li et al. 2009). In the near future human-robot will cooperatively work together for solving problems that are difficult for both groups. For instance, there has been an increasing demand in domains such as rescue to use autonomous or teleoperated humanoid robots to complete high-risk tasks, otherwise would have been lethal to a human subject. Therefore, we envision environments where humans and humanoid robots collaboratively work to complete tasks.

Since both humans and biped humanoid robots have almost identical movements and are susceptible to similar accidents, we believe that the same set of learning algorithms are suitable for both groups. To validate our hypotheses, we develop a generalize approach for learning and predicting activities of both groups. We have attached wearable sensing

devices to collect motion data and have used software tools to interpret the sensor data to distinguish between regular activities and fall events. The sensing devices are assembled using (*i*) off-the-shelf hardware component boards and (*ii*) a set of generalized software tools that we have developed. The software tools collect data from the sensors, which we used for learning to distinguish between routine activities and fall events as well as predicting activities. The software tools provide a modular functionalities to use for collecting data from any sensing devices, and to learn and predict on/off-line. While attempts for identifying human motions have already been investigated, to the best of our knowledge, we are the first to investigate the prospect of using external embedded devices to identify activities on a NAO humanoid robot.

Related Work

The modern activity detection methods can be broadly categorized in to two groups based on: 1) inexpensive wearable embedded devices; and 2) smart-phones. Wearable embedded devices with add-on sensors provide options to develop effective activity recognition methods for humans and biped humanoid robots alike. The existing activity detection methods focus on special cases of fall detection in humans and humanoid robots. These methods were primarily used in isolation. (Ojetola, Gaura, and Brusey 2011) reported method using accelerometers and gyroscopes to detect fall incidence among the elderly people. The work reported detecting four falling events: forward, backward, right, and left. They used a decision tree to learn and classify falls and activities of daily living (ADL). The method identified fall events with precision of 81% and recall of 92%.

(Baek et al. 2013) proposed a fall detection system using necklace-shaped tri-axial accelerometer and gyroscope sensors to classify the behavior and posture of the detection subject. Their method distinguished between ADL and fall, with sensitivities greater than 80% and specificities of 100%. They experimented with ADLs such as: standing, sitting in the chair or floor, laying, walking, running, going up-stairs/downstairs, and bending, while, falling forward, backward, leftward, rightward, and fall on the stairs were treated as abnormal events.

(Leone, Rescio, and Siciliano 2013) prosed a system to detect event that cause trauma, and disabilities using a tri-

axial MEMS wearable wireless accelerometer. They used support vector machine (SVM) for robust classification of different events. (Bao and Intille 2004) reported of using five small biaxial wire-free accelerometers attached on the left bicep, right wrist, left quadriceps, right ankle, and right hip to recognize 20 activities starting from walking to riding elevator to strength training to bicycling. They reported using decision table, instance-based learning, decision tree, and naïve Bayes classifiers, where, decision tree showed the best performance with 84% accuracy. Similar efforts have been reported to detect human motions using motion tracking, e.g., (Dumitrache and Pasca 2013; Kumar and Pandey 2013; Krishnan and Cook 2014; Gao, Bourke, and Nelson 2014; Álvarez-García et al. 2015).

(Moya et al. 2015) proposed a fall detection, avoidance, and damage reduction mechanism for biped humanoid robots. They tried to simulate the real world environment where humanoid robots have to walk over irregular surface, running or playing sports, collusion with other robots. Their framework detected instability and did fall avoidance or at least low-damage falling mechanism was invoked. Therefore, embedded devices with add-on sensors provide a flexible platform to build may real world applications.

There is an increasing popularity in using smart-phones to detect activities in health care domain. Using only the accelerometer readings from a smart-phone, (Bai, Wu, and Yu 2013) analyzed five actions of human walking, running, standing up, sitting down, and jumping. They compared the acceleration characteristics of these actions with three different fall accelerations to infer the direction of the fall. Their method recognize the fall activity, only when a predefined set of conditions were met. But the method did not provide any prediction or indication value, that a fall may occur in future.

(Steidl, Schneider, and Hufnagl 2012) reported that the sensors of the smart-phones from different manufactures record values significantly incompatible ranges for identical tasks. Therefore, they trained a SVM classifier based on the features extracted from raw accelerometer readings and the directional changes of the constraining force exerted on an accelerometer to detect fall events. They compared these events to non-fall activities such as walking, running, jumping, and some actives which resembles falls such as sitting down on a chair. Their method detected fall events 84.8% average accuracy across different smart-phones.

(Dernbach et al. 2012) explored methods to detect simple and complex activities using inertial sensors (accelerometer and gyroscope) of an Android smart-phone. The simple activities included: biking, climbing stairs, driving, running, sitting, standing, walking, and state of the phone not on the person. The complex activities were: cleaning, cooking, medication, sweeping, washing hands, and watering plants. Six different classifiers, multi-layer perceptron (MLP), naïve Bayes, Bayesian network, decision table, best-first tree, and K-star, were trained on using the same feature extractor. For simple activities, 93% accuracy using MLP were reported, while, 50% success was achieved for complex activities.

(Shen, Lai, and Lai 2015) used a high-level fuzzy Petri net for the analysis and the development of identifying normal

human actions such as sitting-down, squatting, walking, running, and jumping and abnormal events such as falling forward, backward, sideways, and vertical. Their fall detection method reported 94% accuracy. One disadvantage of their process was that, some complex situations and movements cannot be detected accurately; e.g., falling down from stairs, multiple collisions, or temporal unbalance motions.

Similar to existing methods, we have defined different normal and abnormal activities for humans and NAO humanoid robots. In order to validate our hypotheses that the learning and predicting methods unifies across the two groups, firstly, we have restricted the activities that can be performed on a NAO robot. Therefore, in our experiments, the human performed activities that are similar to the robot, such as, walk or falling forward/backward so forth. Secondly, we have extended the human activities to more complex events. We have used Texas Instruments (TIs) microcontrollers and boosterpacks for our experiments, but, one can use microcontrollers and sensor boards from other sources.

Our Approach and Contributions

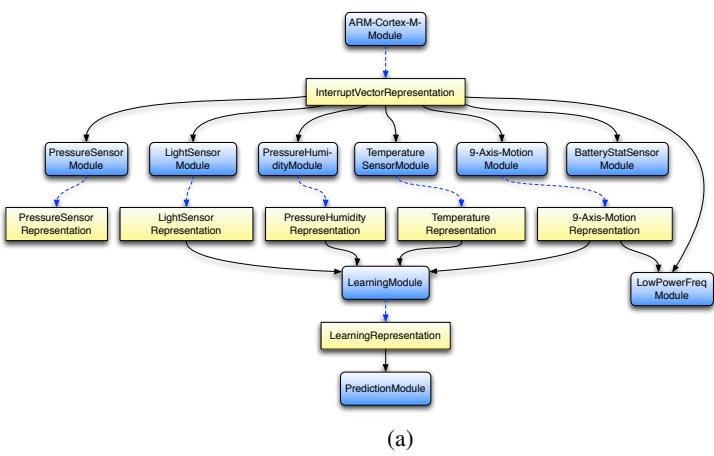
Many off-the-shelf hardware boards, such as, TI microcontroller boards (MSP430TMLaunchPad, TivaTM C Series TM4C123G LaunchPad, Tiva C Series TM4C129 Connected LaunchPad) and add-on BoosterPacks with sensors as well as wireless transmitter and receiver provide flexible platform for developing smart-devices for a wide rage of low power and portable applications. For our experiments, we have assembled a wireless sensing-device with a Tiva C Series TM4C123G microcontroller board, and three boosterpacks — a Sensor Hub BoosterPack for sensing 9-axis motion, a CC2533 BoosterPack for wireless networking, and a Fuel-Tank BoosterPack for power. We also assembled a wireless data collection device with a Tiva C Series TM4C123G microcontroller board, and a CC2533 Booster-Pack. These two devices were networked to create a wireless sensor network (WSN) for collecting motion data from humans and NAO humanoid robots.

We have developed a set of software tools to create a framework that allow us 1) to setup the WSN, 2) to collect data, 3) to learn from sample examples, and 4) to monitor and predict events. This framework is general enough for other practitioners to use the available functionalities for creating WSNs, collecting data, learning, and prediction.

The rest of the paper is organized as follows. First, we describe the software development framework that we have proposed and implemented. In the next section, we briefly describe the experimental setup. In the penultimate section, we present evaluation results for humans and NAO humanoid robots. We have utilized two machine learning algorithms and a thresholding based method to learn and identify normal and abnormal activities. Then we also have presented our observations and discussion. Finally, the paper is concluded with a summary and future work.

Framework

The framework provides generic functionalities to develop applications or rational agents on embedded devices that



(a)

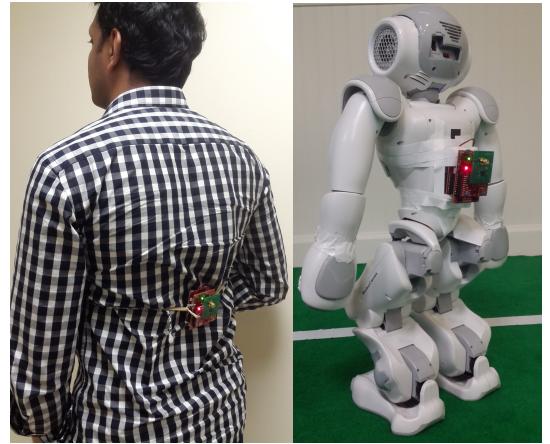


Figure 1: (a) Currently available software modules in our framework and a directed-graph representation of their functional relationship; (b) a wireless sensor device (assembled from a TI Tiva C Series TM4C123G LaunchPad, a Sensor Hub BoosterPack, a CC2533 BoosterPack, and a Fuel Tank BoosterPack) attached to the back of a human subject; the device is running our framework with only three motion-sensing modules; and (c) the same device configuration was used on the back of a NAO humanoid robot.

sense and actuate using add-on boards. The execution paths between sensors to actuators could contain complex behavior manipulations and modeling decisions that needs to be developed efficiently. Hence, the framework takes these considerations into account and provides a topologically sorted graph, based on the decision points provided by practitioners. The framework includes: 1) tools to develop modules and representations that execute on the microcontrollers or off-line, 2) the methods to access functionalities for physical robots, and 3) a real-time visualization system. Our framework is lightweight, flexible, and consumes minimum memory and computational resources. We have tested our framework on multiple microcontrollers and on boosterpacks as stated above. We have written and distributed software solutions to access devices on the boosterpacks such as: (1) InvenSense MPU-9150: 9-axis MEMS motion tracking (three-axis gyro, three-axis accelerometer, and three-axis magnetometer); (2) Bosch Sensortec BMP180 pressure sensor; (3) Sensirion SHT21 humidity and ambient temperature sensor; (4) Intersil ISL29023 ambient and infrared light sensor; and (5) TI's TMP006 non-contact infrared temperature sensor.

Our development framework, *μ Energia* (*pronounced as*: “micro-Energia” and *site*: <http://muenergia.saminda.org>), uses a notion of *modules* and *representations* to perform computations. The modules implement functions, while the representations exchange information from one module to another. They are connected using directed arcs to form a directed graph. In Figure 1a, the blue arrows show the provided representations, and the black arrows show the requested representations. A module can provide multiple representations. The framework computes the topologically sorted graph out of the nodes. This is computed once, and the nodes in the queue will be executed one after the other.

If there were to be cycles in the graph, the framework will detect them and indicate them to the users.

The Figure 1a shows the modules and representations related to our experiments, where the boxes represent the computational modules, while the rounded-boxes represent the input to a module or an output from a module. For brevity, in the rest of the paper, the *computational modules* are referred as *modules*. As an example, the module *9-Axis-Motion Module* contains logic to read from or write to MPU-9150 9-Axis (Gyro+Accelerometer+Compass) MEMS MotionTracking device on the sensor hub booster pack. The representation *9-Axis-Motion Representation* contains all the values that this module shares with other modules in the framework. In this graph, *Learning Module* requests values from *9-Axis-Motion Representation* to implement the learning function.

Experimental Setup

We have conducted our experiments on detecting motions (activities) on humans and humanoid robots (NAO robots). We have used a InvenSense MPU-9150 motion tracking sensor on the Sensor Hub BoosterPack. We have used 3-axis accelerometer readings, 3-axis gyroscope readings, 3-axis magnetometer readings, the quaternion rotation axis of the device, and Euler angles roll, pitch, and yaw. In order to calculate Euler angles, we have used the Direction Cosine Matrix (DCM) algorithm. We have configured one device as a remote sensing device, and the other device is configured to receive data from the sensing device. The rest of the section reports our data collection, data processing, learning, and prediction results.

Activity Annotation: The annotation for different human and robot motion activities are shown in Figure 2. Accelerometer and gyroscope’s *x*, *y* and *z* axis are shown for

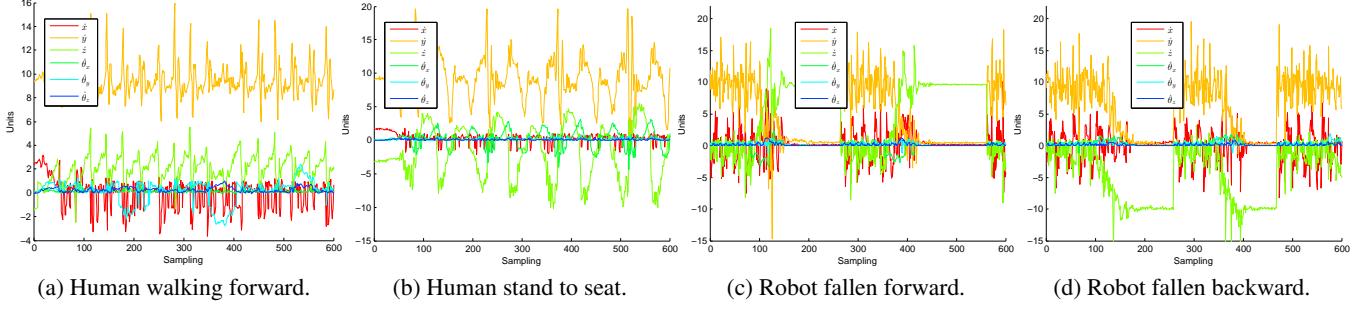


Figure 2: Figures (a–b) shows 3-axis accelerometer and 3-axis gyroscope graph for human motions walking forward and stand to seat. Figures (c–d) shows 3-axis accelerometer and 3-axis gyroscope graph for robot’s fallen forward and backward motions.

different motions. We can observe that an actual event takes between 180–250 ms. Figure 2 (a–b) shows the activity annotations for human motions (walking forward and sitting down). Figure 2 (c–d) shows activity annotations for the NAO robot. The following section provides the evaluation matrices of our experiments.

Evaluation Results

We have structured this section in a way that we discuss feature extraction, data processing, machine learning, and experimental results for both the human and robotic experiments. We have tested our hypotheses using *Regularized Logistic Regression* (LLR) and *Support Vector Machine* (SVM) algorithms. We have used a randomly generated 80%–20% partition for training and cross-validation on our learning dataset. We have used an independent test set to report our results. We have used standard parameter sweeping techniques to find the classifier parameters that provide the optimal trade-off between the bias and the variance, while precision, recall, and F₁-scores have been used to obtain the optimal value.

Feature Extraction

We have configured the motion sensor to sample at every 20ms, which is equivalent to 50Hz sampling rate. In order to identify activities: 1) we have used a window size of 400ms (20 samples); and 2) we have allowed 10 samples (200ms) to overlap between windows. The selection of the window size is based on the observation that a fall event takes between 200 – 300ms. Thus, a window size of 400ms will include both fall event and non-fall event data for classification. For each window, we have calculated the running average of all 9-axis values. This has produced nine values per window, which has been used as features. We also added a bias term to provide additional capacity for the learning algorithms. The accelerometer readings are in 3-axis in meter per square second (m/s^2), gyroscope readings are in 3-axis in radian per square second ($rads/s^2$), and the magnetometer readings are in 3-axis in Tesla (T).

As a preprocessing step, the features, except the bias, have been subjected to feature standardization. We have independently set each dimension of the sample to have zero-mean

and unit-variance. We achieved this by first computing the mean of each dimension across the dataset and subtracting this from each dimension. Then each dimension is divided by its standard deviation.

Experiments with a Human

We have attached the sensors on the mid-back of the human body as shown in Figure 1b for collecting our datasets. We have collected data for several different activities. For the studies reported here we have used the following datasets: (1) walking forward; (2) from standing to sitting down and back to standing; and (3) from standing to falling and back to standing. Example plots for 1) walking forward; and 2) from standing to sitting down are shown in Figures 2a and 2b respectively.

Table 1: Logistic regression and SVM classification for human activities.

Activity	Logistic regression				SVM classification			
	TP	TN	FP	FN	TP	TN	FP	FN
Walking forward	91%	90%	10%	9%	96%	93%	7%	4%
Walking backward	82%	86%	14%	18%	81%	84%	16%	19%
Walking left	86%	86%	14%	14%	89%	90%	10%	11%
Walking right	86%	86%	14%	14%	89%	90%	10%	11%
Falling forward	94%	93%	7%	6%	96%	93%	7%	4%
Falling Backward	84%	88%	12%	16%	84%	87%	13%	16%
Falling left	92%	91%	9%	8%	93%	93%	7%	7%
Falling Right	92%	91%	9%	8%	92%	93%	7%	8%
Marching	91%	90%	10%	9%	95%	93%	7%	5%
Rotate counter-clockwise	91%	89%	11%	9%	93%	91%	9%	7%
Rotate clockwise	92%	89%	11%	8%	94%	91%	9%	6%
Stand to seat	96%	92%	8%	4%	96%	92%	8%	4%

Results: Table 1 shows the final results for LLR classification, where, true positive, true negative, false positive, and false negative are abbreviated with TP, TN, FP, and FN respectively. For walking forward on an average, the accuracy is 91%, similarly, for falling forward and marching the accuracies are 94% and 91% on an average, respectively. We found comparatively low accuracy in walking backward and falling backward. Rotation has accuracy on an average more than 90%. We also observed that detection of the walking activity is less than other activities that we experimented with.

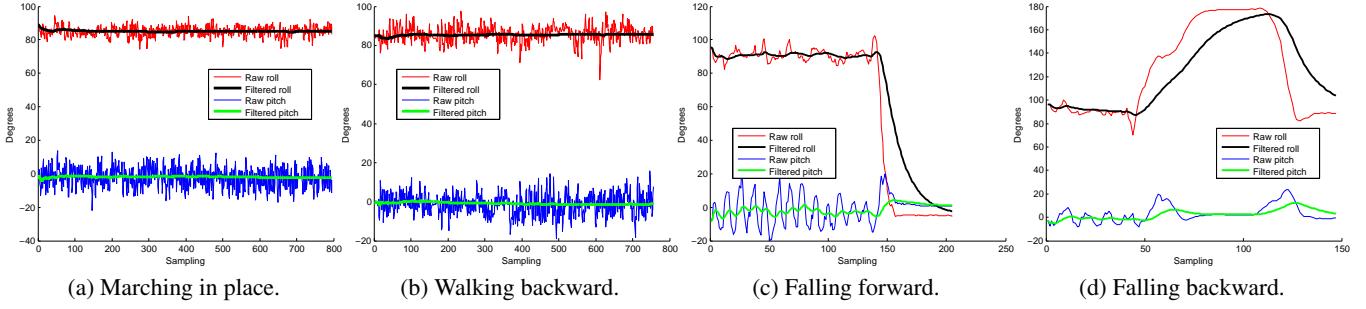


Figure 3: Figures (a–b) show the roll and pitch angles (raw and filtered) for normal behaviors marching in place and walking backward. Figures (c–d) show the raw and filtered roll and pitch angles for fallen forward and backwards states of NAO humanoid robot.

The results for SVM classifier is summarized in Table ???. On an average, accuracy for each type of activity is higher than LLR classifier with on an average 96% for walking, 95% for marching, and 96% for falling forward. Both type of rotations have little higher accuracy over LLR. As a consequence, SVM classifier has less false positives and false negative compared to LLR. Our experiments reveal that the SVM classifier performs better than the logistic regression classifier. However, due to memory and computational restrictions on the embedded devices, we have found that the logistic regression classifier is a better choice. Our feature vector consists of the mean normalized sensor reading and a bias term. We plan to combine and prune features to improve the classification accuracy for future work.

Experiments with a NAO Robot

Our NAO robots have an omni-directional walking engine. Walking motions are regulated by linear velocities \dot{x}_v and \dot{y}_v , and an angular velocity $\dot{\theta}_v$ as input parameters. Depending on these values we can let the robot walk (forward, backward, and side-ways) and rotate (clockwise and counter-clockwise) with different speeds. In this study, we have considered (1) marching ($\dot{x}_v = 0, \dot{y}_v = 0, \dot{\theta}_v = 0$); (2) walking forward/backward ($\dot{x}_v = \pm 200, \dot{y}_v = 0, \dot{\theta}_v = 0$); (3) walking side-ways ($\dot{x}_v = 0, \dot{y}_v = \pm 200, \dot{\theta}_v = 0$); and (4) rotating clockwise/counter-clockwise ($\dot{x}_v = 0, \dot{y}_v = 0, \dot{\theta}_v = \pm 0.5$) as *normal states*, while all the other activities are classified as *fallen states*.

We have attached the sensors on the back of the NAO humanoid robot as shown in Figure 1c and used the following protocol to collect the datasets. First, we set the velocity parameters and activate the walking engine of the robot. Then, we collect the motion data transmitted by the wireless device attached at the back of the robot. Finally, we marked the start point of the sample stream and collect data approximately for a minute and marked the end of the sample stream. This is an episode of a normal or a fallen state. We repeat the about described protocol for collecting multiple episodes to obtain our learning ensemble. To collect samples for fallen states, we have configured the walking engine to marching activity. Then, we safely pushed the robot to different direc-

Table 2: Logistic regression and SVM classification for robot activities.

Activity	Logistic regression				SVM classification			
	TP	TN	FP	FN	TP	TN	FP	FN
Walking forward	91%	90%	10%	9%	93%	91%	9%	7%
Walking backward	90%	90%	10%	10%	93%	91%	9%	7%
Walking left	92%	90%	10%	8%	94%	90%	10%	6%
Walking right	89%	90%	10%	11%	90%	91%	9%	10%
Falling forward	94%	93%	7%	6%	98%	93%	7%	2%
Falling Backward	94%	93%	7%	6%	98%	93%	7%	2%
Falling left	95%	93%	7%	5%	99%	94%	6%	1%
Falling Right	94%	93%	7%	6%	98%	93%	7%	2%
Marching	91%	89%	11%	9%	90%	91%	11%	10%
Rotate counter-clockwise	97%	92%	8%	3%	97%	93%	7%	3%
Rotate clockwise	98%	92%	8%	2%	96%	93%	7%	4%

tions to inducing the fallen state. Similarly, we marked the sample streams to generate several episodes for the learning ensemble.

Results: The results for LLR and SVM classifiers are summarized in Table 2. The results are similar to identification of human activities, for conserving space, we omit detail discussion of the results. It is worthwhile to note that the SVM identified events more accurately than LLR as observed for human activities.

Even though the above learning methods provide accurate results, there are microcontrollers with very limited computational and memory capacities. In such devices, it is better to use less memory expensive methods such as Kalman filtering, which is reported in the next section.

Kalman Filtering: We have designed a thresholding base activity predictor in these experiments. Using the the same datasets, we have calculated the roll, pitch, and yaw angles using the DCM algorithm. In order to achieve an effective threshold-based decision making, we have filtered the roll, pitch, and yaw values using a Kalman filter (Welch and Bishop 1995). Figures 3 (a–b) show the raw and the filtered values of the raw and the pitch values for marching and walk backward. The other activities show similar plots (we have refrained from plotting them due to space constraints).

The thresholding method suggests that, if the filtered roll values are within the range $[90 \pm 15]$ (degrees) and the fil-

tered pitch values are within the rage $[0 \pm 15]$ (degrees), then with 100% accuracy, the NAO robot will be in a normal state. Otherwise, we can safely assume that the robot is in a fallen state. Figures 3 (c–d) show the roll and pitch angles (raw and filtered) for a typical fallen robot. If the filtered roll value is less than 60 degrees, we can safely assume that the robot is falling forward. If the filtered roll values is more than 100° we can assume that the robot is falling backward. To detect the events falling to the left and right, we have used the filtered pitch values. If the filtered pitch value is less than -50° , we can assume that the robot is falling to the left side, while if the filtered pitch values is more than 50° , we can assume that the robot is falling to the right side. With these thresholds for a separate test cases, the thresholding method has detected fallen state with 100% accuracy.

Even though the thresholding method successfully distinguished between normal and fallen activities, it was unable to detect states within the normal activities, i.e., walking forward to walking backward so on. This one of the limitation of using thresholding methods, which has been overcome by the learning methods. We use Kalman filter in this experiments because of its efficacy and the choice of the filter is entirely arbitrary depending on the computational resources of the microcontroller.

Conclusions and Future Work

Falling events could cause damage to both humans and robots. Using prediction information, corrective measures can be engaged to avoid many fall events. Similarly, for humans, after a fall is identified, rescue services can be called upon. We have reported a wireless sensing device that have assembled using off-the-shelf hardware components. Using two devices, we have created a sensor network for collecting motion data from humans and biped humanoid robots. The sensing devices were connected to the back of the subjects and the data collecting device was connected to a laptop which archived the data. We have used two machine learning algorithms and thresholding methods to identify both normal and abnormal activities within 91%–100% accuracy. Comparing the encouraging experimental results we achieved, our future work will be to use multiple sensing devices to create a sensor network to detect complex activities with higher sampling rates.

References

- Álvarez-García, J. A.; Morillo, L. M. S.; de La Concepción, M. Á. Á.; Fernández-Montes, A.; and Ramírez, J. A. O. 2015. Evaluating wearable activity recognition and fall detection systems. In *6th European Conference of the International Federation for Medical and Biological Engineering*, 653–656. Springer.
- Baek, W.-S.; Kim, D.-M.; Bashir, F.; and Pyun, J.-Y. 2013. Real life applicable fall detection system based on wireless body area network. In *Consumer Communications and Networking Conference (CCNC), 2013 IEEE*, 62–67. IEEE.
- Bai, Y.-W.; Wu, S.-C.; and Yu, C. H. 2013. Recognition of direction of fall by smartphone. In *Electrical and Computer Engineering (CCECE), 2013 26th Annual IEEE Canadian Conference on*, 1–6. IEEE.
- Bao, L., and Intille, S. S. 2004. Activity recognition from user-annotated acceleration data. 1–17. Springer.
- Dernbach, S.; Das, B.; Krishnan, N. C.; Thomas, B. L.; and Cook, D. J. 2012. Simple and complex activity recognition through smart phones. In *Intelligent Environments*, 214–221. IEEE.
- Dumitache, M., and Pasca, S. 2013. Fall detection algorithm based on triaxial accelerometer data. In *E-Health and Bioengineering Conference (EHB), 2013*, 1–4. IEEE.
- Gao, L.; Bourke, A.; and Nelson, J. 2014. Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems. *Medical engineering & physics* 36(6):779–785.
- Krishnan, N. C., and Cook, D. J. 2014. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing* 10:138–154.
- Kumar, P., and Pandey, P. C. 2013. A Wearable Inertial Sensing Device for Fall Detection and Motion Tracking. In *Proc. Annu. IEEE Conf. INDICON, Mumbai, India*, 1–6.
- Leone, A.; Rescio, G.; and Siciliano, P. 2013. Supervised wearable wireless system for fall detection. In *Measurements and Networking Proceedings (M&N), 2013 IEEE International Workshop on*, 200–205. IEEE.
- Li, Q.; Stankovic, J. A.; Hanson, M. A.; Barth, A. T.; Lach, J.; and Zhou, G. 2009. Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. In *6th International Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, 2009, 138–143.
- Moya, J.; Ruiz-del Solar, J.; Orchard, M.; and Parra-Tsunekawa, I. 2015. Fall Detection and Damage Reduction in Biped Humanoid Robots. *International Journal of Humanoid Robotics*. DOI: 10.1142/S0219843615500012.
- Ojetola, O.; Gaura, E. I.; and Brusey, J. 2011. Fall detection with wearable sensors—safe (smart fall detection). In *Intelligent Environments (IE), 2011 7th International Conference on*, 318–321. IEEE.
- Shen, V. R.; Lai, H.-Y.; and Lai, A.-F. 2015. The implementation of a smartphone-based fall detection system using a high-level fuzzy petri net. *Applied Soft Computing* 26(0):390 – 400.
- Steidl, S.; Schneider, C.; and Hufnagl, M. 2012. Fall detection by recognizing patterns in direction changes of constraining forces. In *Proceedings of the eHealth 2012*, 27–32. OCG.
- Welch, G., and Bishop, G. 1995. An Introduction to the Kalman Filter. Technical report, Chapel Hill, NC, USA.