A simple method for reliable footstep detection on embedded sensor platforms

Ryan Libby

June 25, 2008

Abstract

Classification of human activity in real-time enables automatic health monitoring and other applications that require accurate and rapid feedback of activity. Classifiers vary in design and efficacy for particular tasks and data sets, but generally benefit from using features which correlate well with classes. For at least the classes walking, running, and bicycling, one set of such features is based on footsteps; when they occur, at what rate, etc. I present a method for reliable step detection suitable for use on embedded sensor platforms equipped with 3-dimensional accelerometers, such as are available now in specialty devices or will soon be incorporated into some mobile phones. Performance on a set of data collected by our group is discussed. Further, step detection enables the extraction of intra-step features by providing delimiting events.

1 Introduction

Step detection is the automatic determination of the moments in time at which footsteps occur, such as is performed by pedometers to generate step counts. Accelerometers are becoming increasingly ubiquitous in commercially sold devices, such as mobile phones like the Apple iPhone or the Nokia N95. While previous work is able to detect steps from foot-mounted accelerometers, this paper demonstrates step detection working reliably from hip-mounted accelerometers, a conceivable location for a device such as a mobile phone. In addition, the sensor platform used is no more powerful than many modern mobile phones. More than enabling pedometer functionality on mobile-phone-like devices, the step detection algorithm presented here could lead to more accurate automatic classification of bipedal human activities.

Inferring human activities in real-time is a major goal of health applications. One general approach is to train a classifier to recognize activities. The input may be generated by sensors on an embedded device. The raw data are featurized before being fed to the classifier.

When inferring common human activities such as walking or running, features related to bipedal motion are especially useful in distinguishing classes.

Some accelerometer-based features are already used to aid activity recognition, such as the spectral distribution of the signal over a window. Using accelerometers to detect steps opens up not only some of the more obvious step features, such as step count and stride length, but also features that use steps to delimit sections of the signal. Features delimited by detected steps can be called intra-step features. One such feature might be the quality of a match between a time-normalized step and a template signal.

The step detection algorithm is new, but simple. It uses an adaptive filtering technique to smooth the acceleration signal and locate steps. This technique is demonstrated to work while subjects walk or run at a range of paces from 1.9 to 2.8 Hz. Bicycling performance is not yet as reliable.

2 Related work

Accelerometer signals have previously been shown to be useful for step detection[1]. In 2007, Ying et al evaluated three different step detection algorithms for a 2-axis, fixed-orientation, foot-mounted acceleration sensor. One of them, the Pan-Tompkins method, was the starting point for the work presented in this paper. All three of the algorithms presented by Ying et al are online and potentially suitable for use on embedded sensing systems. However, a direct comparison of the algorithms, performance on their data set, and the actual embedded implementation were all left to future work.

Apart from step detection, accelerometer features have been used successfully for classification of human activities, both offline[2] and in real-time[3]. Using a variety of simple features from accelerometer, barometer, and microphone input, Lester et al were able to classify distinct human activities with high accuracy. Consolvo et al used real-time classification of human activities to provide rapid-feedback health monitoring for a fitness application. The eventual goal of the work presented here is to enable the generation of features that will allow a finer-grain classification of human activities.

Lester et al and Consolvo et al used the same sensor hardware as this step detection algorithm, although they sampled the accelerometer at different rates.

3 Sensors

Input was collected using the Mobile Sensing Platform (MSP) developed by the University of Washington and Intel Research Seattle, shown in Figure 1. The MSP is an embedded sensing platform consisting of the Multi-Modal Sensor Board 2 (MSB2), a small sensor package, stacked on the IMote2, a small mobile-phone-like computing device. The MSP and related software comprise a comprehensive system for sensing, analyzing, and classifying the physical environment. They are described in detail elsewhere[4].

The MSB2 features a sensor board with a 3-axis accelerometer, which was sampled at 512 Hz (the STMicro LIS3L02DS). Each sample gives a 3-dimensional acceleration vector. The sensor board was worn clipped to clothing at the hip. It is therefore in position to sense

(a) (b)

Figure 1: The MSP is shown without a protective case or battery in (a). The upper daughter board is the MSB2. Photo credit University of Washington, http://ubi.cs.washington.edu/wiki/index.php/Sensor_Boards. The MSP is shown in (b) (black box) worn clipped to clothing at the right hip (in this case, a belt).

hip acceleration during bipedal motion.

The juxtaposition of sensing platform (MSB2) and moderate computing power (IMote2) opens the possibility of processing and analyzing data sensed from the environment in real-time.

4 Characteristics of the step detection algorithm

An important goal was to have steps be detected online. Online step detection is most easily run on the device, near where the sensor data are collected. The eventual goal of using step detection-derived features to inform a classifier also relies on step detection being accomplished on the device, near where other features are computed.

Live detection on the device implies several constraints.

Available computational resources are limited. The ARM processor of the IMote2 has 32 MB of RAM available to it and runs at a 400 MHz clockrate. This put a practical limit on, for example, the amount of time over which the algorithm can "look ahead." (Look ahead is also limited by the desire to minimize detection latency.) It also lacks hardware for integer division and floating point operations. These facts influenced the choices for representations and transformations of the data.

Online step detection is subject to real-time considerations; if newly sensed data are not serviced in time, they are lost. (For reference, new data are introduced at an average of almost every 2 ms.)

The Library for the Inference of Real-time Activities (LIRA), developed by Intel Research Seattle, provides a data-processing framework for use with the IMote2. The key advantage it offers is the infrastructure for processing streaming data, such as continuous sensor readings. LIRA helps to address the real-time issues by managing sensor input and the dataflow

between logical steps of the algorithm.

The end result with respect to performance is that the entire system, including the step detection algorithm and basic feature generation, runs on the MSP at 25% CPU usage. The latency between physical impulse and output of a detected step is never more than ten seconds. There is enough excess processing power for more advanced feature extraction and running a classifier.

5 Design process

The algorithm was developed over a period of months during which several approaches were considered. The initial implementation was done in Matlab and closely followed Ying et al's description of the *Pan-Tompkins Method*. The other two algorithms described by Ying et al were not attempted. The *Template Matching Method* was avoided for its complexity. The *Peak-detection method based on combined dual-axial signals* relied on assumptions about the correlation between vertical and lateral accelerations of the foot which could not be demonstrated on the hip.

The Pan-Tompkins implementation did not work on our data set because the sensor setups were different. Ying et al collected data from foot-mounted accelerometers and predetermined the orientation of the accelerometers. In contrast, no assumptions were made about the orientation of our accelerometers. Moreover, the device in our case was clipped to clothing at the hip. (In practice, the clip did enforce an orientation of the device, but use of this fact was deliberately avoided.)

Several attempts were made to make simple modifications which would enable the use of the Pan-Tompkins method. These attempts were largely confounded by difficulties designing a low-pass filter. Some of the filters applied were a simple average over a sliding window with various choices of window size, and several Butterworth filters.

Their performance was not found to be satisfactory. They were generally unable to remove components of the signal which were at frequencies only moderately higher than the dominant ones. The result was that rather than there being one or two peaks per step in the Pan-Tompkins output, there would be several more.

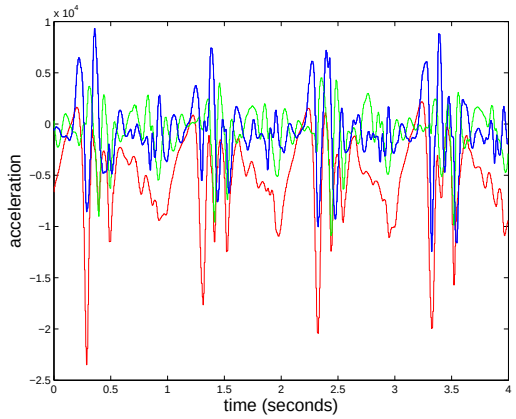
The excess peaks often had different characteristics, such as being of generally lower amplitude. This led to attempts to address the excess peaks directly, by, for example, only accepting peaks of a certain relative size within a window. This approach did not work well when peaks from one foot were much bigger than peaks from the other, a common occurrence.

The accumulation of small adaptations led to seeing the problem in a different light: as an attempt to develop a low-pass filter that would leave the rest of the algorithm trivial. The Pan-Tompkins method was abandoned for one that did most of its work at the low-pass filter.

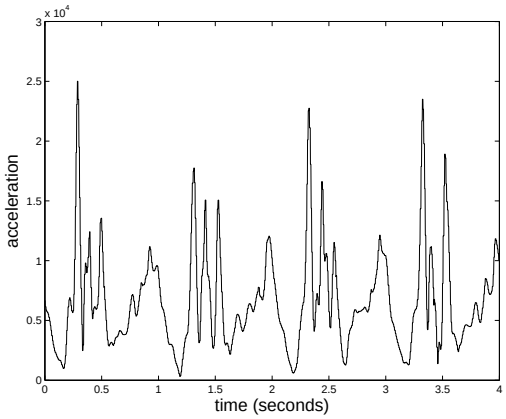
6 Algorithm

The algorithm to detect peaks is simple in concept. It uses an obvious technique to find local maxima in the magnitude of the sensed acceleration vector. The local maxima are assumed to correlate to footfalls.

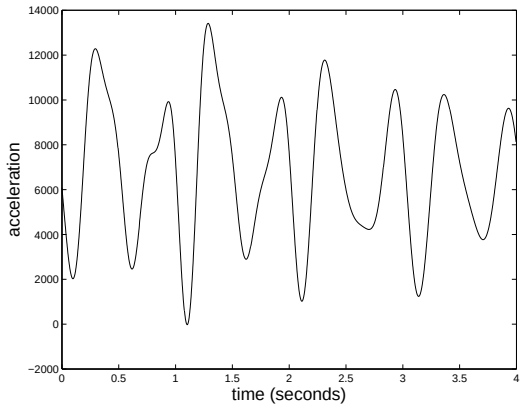
First a low-pass filter is applied to the acceleration magnitude signal, giving the smoothed acceleration signal (Figure 2b to Figure 2c). Then a function approximating a derivative is applied to the smoothed acceleration, giving the jerk (Figure 2c to Figure 2d). When the jerk crosses zero from positive to negative, the smoothed acceleration is at a local maximum.



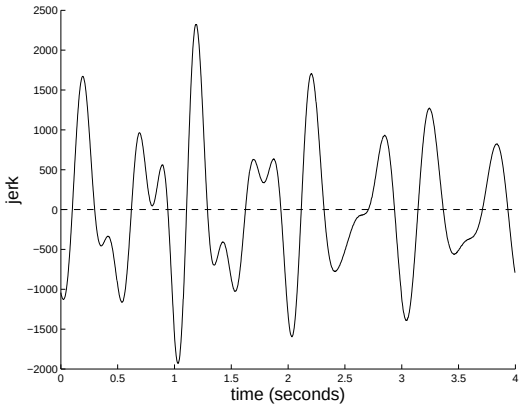
(a) Raw accelerometer readings.



(b) Magnitude of the 3D accelerometer vector.



(c) Output of the low-pass filter.



(d) Derivative of the low-pass filter.

Figure 2: A short section of a typical trace at the major stages of the step-detection algorithm.

6.1 Input

Data are sampled from the three orthogonal accelerometers at 512 Hz (see Figure 2a for an example). The three accelerometer readings compose a three-dimensional acceleration vector. The algorithm considers only the magnitude of that vector. It is therefore not sensitive to the orientation of the sensors, and so has no dependence on any particular orientation.

Determining the magnitude of the vector is simple and relatively cheap, but it has a disadvantage. It has the potential to introduce high-frequency artifacts when one of the components periodically crosses zero. (Consider $sin(x)$ vs. $|sin(x)|$.) High-frequency artifacts, noise, and acceleration not obviously correlated with bipedal motion give the motivation for the low-pass filter.

6.2 Low-pass filter

The ideal output of the low-pass filter is one where every local maximum corresponds to a footfall. Its input is far from that. The general shape is apparent, but there are many departures. They are of varying amplitudes, but they tend to be short in duration. Therefore a filter which discards high-frequency events while retaining the general shape of the signal was desired.

A simple solution would be a standard FIR filter to pass a low-frequency band. The biggest problem with such an approach is choosing the cutoff frequency. A typical walking pace may be under 2 Hz, while a running or bicycling one may be double that. A choice of cutoff frequency that accommodates faster paces may leave high-frequency components in a signal collected from a slow-paced activity. Conversely, a choice that removes high-frequency components from a signal from a slow-paced activity may not faithfully preserve important aspects of a signal collected from a faster-paced one.

Although in both cases a low-pass filter with a certain cutoff frequency is desired, the choice of that frequency may discard either too much or too little of the signal. One way to quantify the portion of a signal in a given frequency band is the portion of energy in that band.

The signal energy of a discrete-time signal is the sum of the squares of its time-domain values, i.e. $\sum x(t)^2$. Parseval's theorem says that this quantity is the same as the sum of the squares its frequency-domain values, i.e. $\sum X(f)^2$, where X is the discrete Fourier transform (DFT) of x.

A signal from a faster-paced activity should have relatively more energy in higher frequencies bands than a signal from a slower-paced activity.

The low-pass filter was therefore designed to preserve a certain portion of the energy of the input signal. It does this by using the lowest frequency cutoff that preserves some portion of the total energy of a window of the signal.

The low-pass filter utilizes the DFT of its input to implement a hard frequency cutoff. The underlying idea is to utilize the DFT as an expression of a time-domain input as a frequency-domain output. The frequency-domain representation is useful for direct manipulation of

the components to produce the final output of the filter.

Although transforming the acceleration magnitude signal from the time domain to the frequency domain is an expensive operation, it is useful for more than the low-pass filter. Classifiers using accelerometer-based features such as those used by Lester et al and Consolvo et al already compute the DFT of the magnitude of the acceleration. In such an environment, this operation alone incurs no additional computational penalty. Unfortunately, the need to operate on overlapping windows, explained below, means that the DFT is actually computed more often than it might be under the feature generation scheme alone.

First the real FFT of a window of the input is taken. Then, an "energy filter" is applied. Figure 3 visualizes an example input to and output of the energy filter.

The energy filter first considers the total energy inside the window. To find this quantity, the energy of each of the bins (except the DC) is summed. Again, by Parseval's theorem, this sum is equivalent to the total energy of the window less its mean.

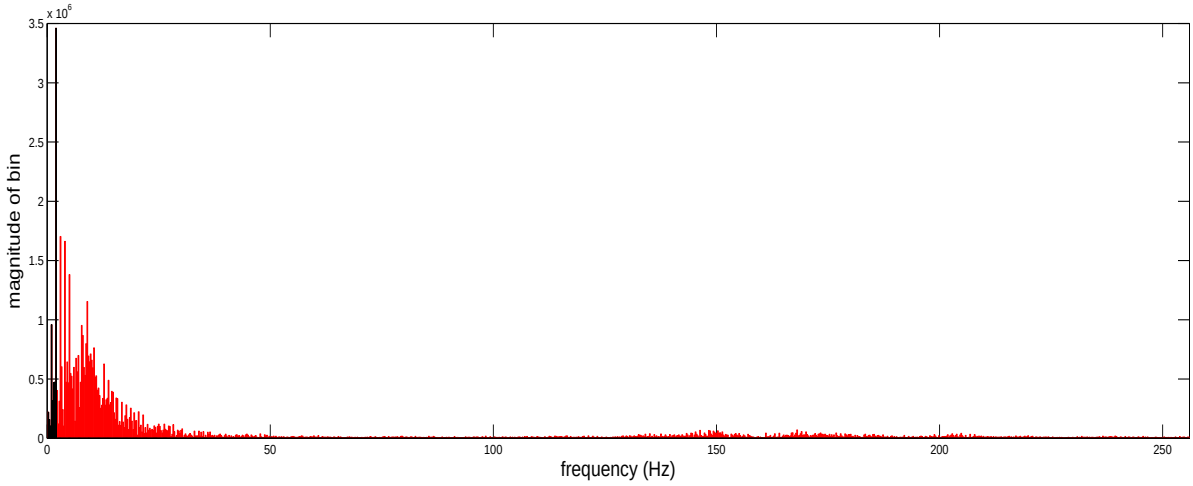
The total energy is multiplied by a parameter to the filter, the "relative energy threshold," to attain a threshold energy. Then, starting from the bin representing the lowest frequency, the value of the bin is either passed through or set to zero. It is set to zero if the sum of the energies of the lower frequency bins is at least the threshold energy. In effect, the filter chooses to pass through the fewest number of low-frequency bins that explains at least the portion of the total energy specified by the relative energy threshold.

It is important to note that the above method leads to a dynamic choice of the cutoff frequency; that is, a potentially different filter is applied to every window. For a given window, recovering a time-domain representation is as simple as taking the inverse FFT, but the output signal cannot be expected to be smooth at window boundaries. Further, the dynamic frequency cutoff precludes some common convolution techniques.

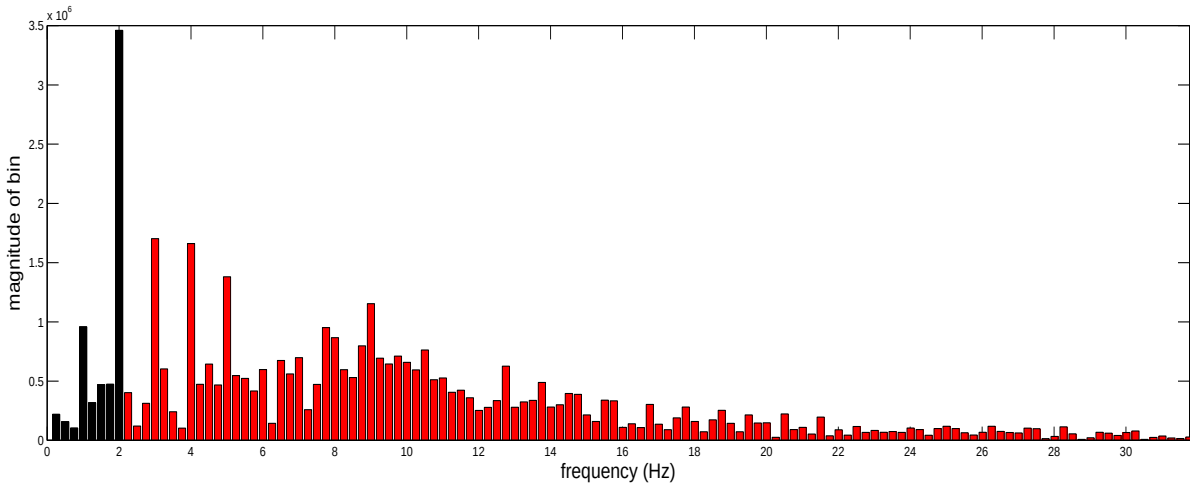
The solution used is to apply the energy filter to windows that overlap by half. The value of a given point is then determined by a weighted average of the two values covering it. The weight function increases linearly toward the center of a window (i.e. it is a triangle weight function). The function used is equivalent to

$$w_x = \begin{cases} 2x + 1 & x \leq n/2 \\ 2n - (2x + 1) & x > n/2 \end{cases}$$

where x is the zero-based index into a window, and n is the number of samples in a window. Triangular weights which overlap by half have the property that the sum of the weights is constant (see Figure 4 for a visualization). Particular choices of the window size (n) allow the overlapping weighted average to be computed cheaply with two multiplies, an add, and a shift for every point. For the above function, when n is a power of 2, the sum of weights is also a power of 2.



(a)



(b)

Figure 3: An example window of data samples passes through the energy filter. (a) shows the full spectrum while (b) focuses on the lower $\frac{1}{8}$ of frequencies. The energy filter either passes bins through unchanged (black) or removes them entirely (red) and sets them to zero. The adaptive nature of the energy filter causes the cutoff frequency to vary with the relative energy distribution.

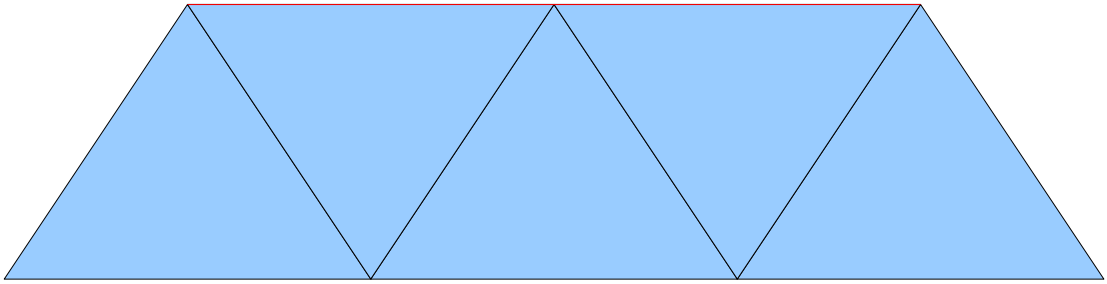


Figure 4: A visualization of the triangle weight function showing how the sum of the weights of overlapping windows is constant. Five windows (triangles) are shown. Each sample is covered by two windows. The vertical distance from the base to the border with the adjacent triangle is proportional to that window's contribution to the output value of the point at that time.

6.3 Derivative

The derivative operation applied is almost exactly that described by Ying et al in their eqn. (2).

$$y(n) = \frac{1}{8}[2x(n) + x(n-1) - x(n-3) - 2x(n-4)]$$

The one difference is that the division by 8 was omitted since no downstream operations depend on the scale of the derivative in this algorithm.

7 Methodology

7.1 Data collection

Three members of our group wore the MSP and collected raw sensor data from several activities. In addition to the accelerometers, several other sensors that were not used for step detection were sampled and logged. These data are the unmanipulated readings from the sensors, sampled at fixed intervals. From the logged sensor readings it is possible to reconstruct all of the physical environmental information availabe to software running live on the device. At all times during data collection, the MSP was worn clipped to clothing at the right hip, usually a belt or waistband (see Figure 1b).

Data were collected over short periods of time during which primarily only one activity was performed. The sensor data logs from such a period are referred to as a "trace." Over 8 hours of traces in total were collected by our group.

The activities included bicycling, sitting and standing, walking, and several variation on walking. Table 1 summarizes the activities and durations of collected data. Due to some

Table 1: A summary of collected data.

Activity	Minutes collected	Minutes usable
Sitting/Standing	86	47
Walking	199	108
Bicycling	82	65
Vehicle passenger	129	60

issues with software beyond the scope of the project, not all of the logs were usable. Corrupt or missing data from some sections of the traces lessened the amount available for analysis. Although most of the lost data would probably be recoverable with some effort, they were instead simply discounted.

In addition, a related project collected running data with a compatible sensor setup, some of which was use in the analysis.

7.2 Evaluating performance

Determining the accuracy of step detection is complicated by the question of "ground truth." The only data collected directly related to bipedal motion were from the accelerometers; no independent scheme is used to assess accuracy. As such, determining the accuracy of a detected step, in the sense of how near in time the detected moment is to the true one, is not feasible with our data.

The correctness of step detection can still be assessed on a large scale. A bipedal activity conducted at a steady pace will have a narrowly-focused step period distribution. (The length of time between two detected steps is here called a step period.) If a step fails to be detected, the result will be a step period which is roughly two or more times the expected period. If a step is detected when none occurred, the result will be one or more unusually short step periods.

This still leaves the problem of the expected step period. Much of our data were collected in natural settings where the expected period was not known, calculated, or measured. Still, if step detection is reasonably good, it can be expected that the expected period is near the center of the step period distribution.

The median is chosen to represent the center of the distribution. Two other simple choices for the center might be the mode or the mean. The problem with the mode is that many of the step period distributions turn out to be bimodal, and there is no obvious reason to choose one mode over the other. The median is chosen over the mean to avoid sensitivity to outliers.

Using these observations, it is possible to devise a quantitative scheme for characterizing false detections. Step periods less than .6 times the median period likely represent false positives. (.6 is used instead of half because one conceivable failure mode is a section of slightly slower steps being double-counted.) Steps periods greater than 1.5 times the median

period likely represent false negatives. (1.5 is used instead of 2 to reveal the case where slightly faster steps are half-counted.)

It is important to note that there is not a one-to-one correspondence between unusually short or long steps and false positives or negatives. For example, in a section of a trace where steps are double-counted, every two unusually short periods may actually only represent one false positive. Conversely a step period three times the expected value may represent two false negatives, rather that one.

8 Results

The major results of running the step detection algorithm on our data set are presented in Table 2, which summarizes the detected step period distributions from a number of sections of sensor traces.

Of the 18 walking and running segments examined, in only 4 do less than 90% of the step periods fall between .6 times the median and 1.5 times the median. The lowest percentage in the middle for walking and running is 75.39%.

Detected step periods from bicycling vary much more. Never more than 70% of the detected step periods fall in the middle. Significant portions of the step period distributions for bicycling segments fall both in the "unusually short" and "unusually long" categories.

Figure 5 gives a visualization of the data presented in Table 2 using segment 22 as an example. Figure 5 also demonstrates a typical characteristic of the step period distribution: bimodality.

Many but not all of the step period distributions are bimodal. The bimodality is caused by alternating short and long periods in the detected steps, as shown by Figure 6.

Step detection was also run on data collected when no bipedal motion was occurring, such as when the subject was sitting or was a passenger in a vehicle. No steps were detected in these cases.

9 Discussion

The step detection algorithm is quite consistent when operating on running or walking data. Step detection of bicycling data works, but suffers both from under- and over-counting steps. Under-counting happens especially when the energy filter is satisfied by the energy in the two-step period.

9.1 Bimodal step period distribution

The distribution of stride lengths is in many cases bimodal. There are two possible explanations: the subject may be exhibiting an uneven gait; or it may be an artifact of the step detection algorithm.

It would be possible to determine which effect was dominant with a simple experiment. Step detection would have to be run with the device attached to both hips. It could then

Table 2: A summary of the step period distribution of detected steps in segments of traces. "n steps" is the number of steps detected in the segment; "% fast" is the percentage of "unusually short" step periods (those less than or equal to .6 times the median); "% slow" is the percentage of "unusually long" step periods (those greater than or equal to 1.5 times the median); "% middle" is the percentage of remaining step periods.

segment	activity	length (s)	n steps	median (s)	% fast	% middle	% slow
1	bicycle	516	1029	.354	12.73	57.63	29.64
2	bicycle	516	769	.606	31.73	65.93	2.34
3	bicycle	256	430	.354	10.00	61.40	28.60
4	bicycle	516	687	.391	4.95	61.72	33.33
5	bicycle	516	1028	.381	4.18	63.04	32.78
6	run-treadmill	175	431	.408	1.62	98.14	0.24
7	run-treadmill	171	432	.398	2.31	99.77	0.00
8	run-treadmill	224	711	.359	24.61	75.39	0.00
9	run-treadmill	126	380	.357	13.16	86.84	0.00
10	run-treadmill	175	432	.416	3.70	96.30	0.00
11	walk	516	913	.529	1.20	96.17	2.63
12	walk	338	695	.509	5.32	94.68	0.00
13	walk	203	403	.506	0.50	99.50	0.00
14	walk	222	356	.520	1.69	92.70	5.62
15	walk	278	591	.494	6.43	93.40	1.69
16	walk	516	637	.506	8.95	82.26	8.79
17	walk	367	750	.486	0.53	99.33	0.13
18	walk	418	864	.488	0.58	99.31	0.12
19	walk	516	1015	.498	1.77	98.23	0.00
20	walk-treadmill	516	984	.498	0.91	96.75	2.34
21	walk-treadmill	256	517	.480	1.93	96.91	1.16
22	walk-treadmill	516	1025	.496	1.37	98.15	0.49
23	walk-treadmill	516	1070	.529	15.42	83.93	0.65

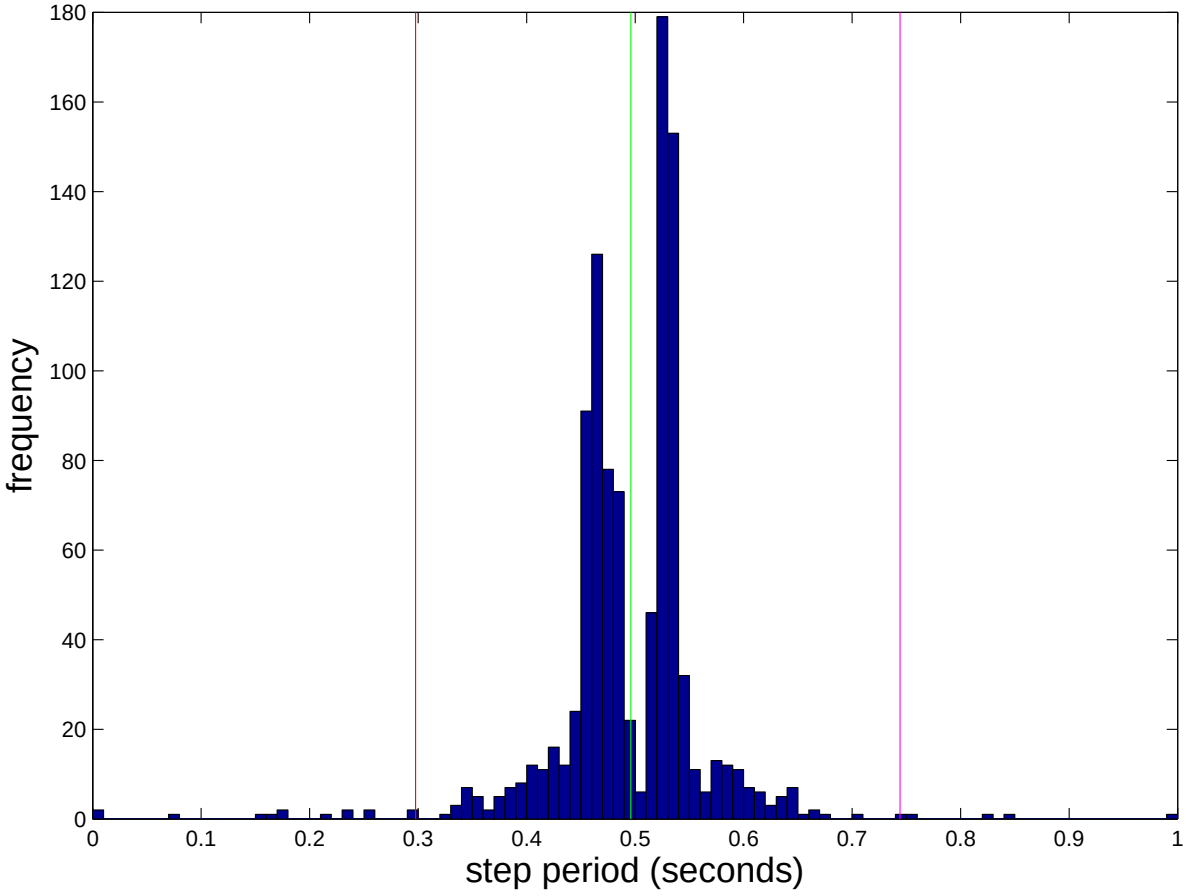
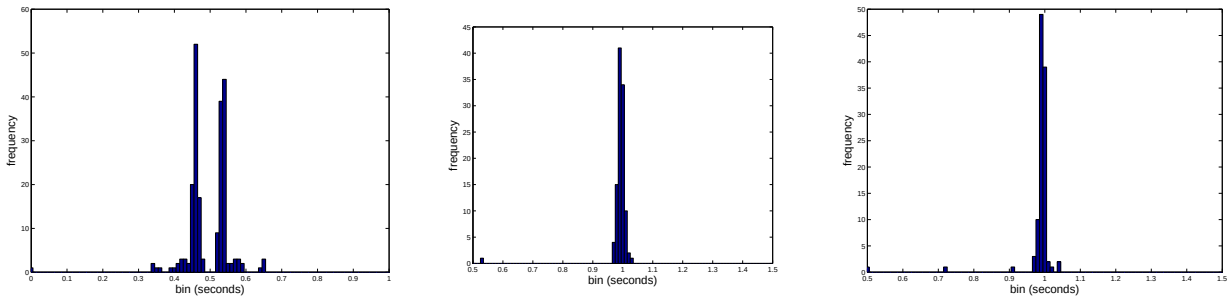


Figure 5: Histogram of the step period distribution of segment 22. A green vertical line is drawn at the median of the distribution. A red vertical line is drawn at .6 times the median; and all step periods shorter than that value are considered "unusually short" (probably false positives). A magenta line is drawn at 1.5 times the median; and all step periods longer than that value are considered "unusually long" (probably false negatives).



(a) (b) (c)

Figure 6: The bimodal characteristic of the step period distribution is due to alternating short and long periods. (a) is the step period distribution of a section of segment 22. (b) is the period distribution when every other detected step is skipped. (c) is the same as (b) but starting on the opposite step. (b) and (c) could be said to represent the two-step period distribution.

be determined whether the short (or long) period is associated with the left or right leg or with being near the same leg as the device.

Bimodality has implications for further feature extraction. If it cannot be avoided, then it may not be possible to treat all steps the same. Right and left steps could be examined separately, or the step could be examined over a two-step period.

9.2 Improvements to the evaluation

An independent method for verifying the results of the step detection algorithm is desirable. One possibility would be to use an off-the-shelf pedometer to maintain a step count which could be used as a sanity check. A similar concept could be used to count bicycle pedal revolutions. Neither would allow an examination of the accuracy of the timing of detected steps.

A major weakness of the analysis used is that the number of detected step periods outside the middle range does not actually give the number of mistakes. Many short periods may represent many mistakes over only a short stretch of the trace; and only a single long period may represent a failure to detect steps over a large stretch of the trace.

The bicycling data is particularly hard to evaluate. Most of the mistakes appear to be in failing to detect "steps" (pedaling). The problem is that it is unknown whether the absences of detection reflect real gaps (such as if the subject were coasting) or if they are in fact false negatives. More controlled bicycling data is needed in order to determine which is the case. Ways to collect bicycling data without coasting gaps might be using a stationary bicycle, riding in a velodrome, or riding on an otherwise unobstructed course.

On the other hand, the large portions of bicycling step periods which seemed too fast are likely real mistakes. Segment 2, the one with the highest percentage of "unusually short" periods is an interesting case which may not be evidence of the above point. The median of the distribution is .606 seconds, higher than any other segment examined, which means any

period of .364 seconds or less would contribute to its short-period category. .364 seconds is higher than the median of two of the other bicycling segments, leaving open the possibility that many of the "short" periods are actually not unusual. Segment 2 may then be more indicative of a weakness with the analysis than with the algorithm.

9.3 Improvements to the algorithm

A number of improvements to the algorithm are possible. As it stands, the algorithm is simple, fast, and works well for walking and running. A few small changes might avoid some of the failure modes.

The first step of the algorithm takes the magnitude of the acceleration vector. It might be better instead to select the component of the acceleration in the vertical direction. This would avoid high-frequency artifacts introduced by lateral accelerations. Orienting the device could still be avoided by keeping a running estimate of the direction of the acceleration of gravity. (Such a capability already exists in LIRA.) Removing high-frequency artifacts simplifies the job of the low-pass filter.

The low-pass filter could use refinement in general. The key property that made it perform well from walking at 1.9 Hz to running at 2.8 Hz was an adaptive cutoff frequency. Now that that concept and an application of it has been explored some, it would be worth trying it with a cutoff that isn't so dramatic.

A problem that contributes to step detection working at half the proper rate on some traces is asymmetric contributions to the energy in the signal from the left and right steps. If it is uneven enough, the energy filter passes through the signal at half the desired frequency. Refinements of the low-pass filter should consider how to address this.

10 Conclusion

The method for step detection here presented is simple, effective, and capable of running on embedded hardware. It opens up the possibility of extracting intra-step features, such as the shape of the accelerometer signal normalized over a step, which could in turn be used to characterize steps from different activities. In more specific terms, choosing a set of delimiters opens up a set of features which are not shift-invariant, unlike features such as band energy used heretofore.

While this algorithm is a reasonable start, more work must be done to bring it to full effectiveness on some activities, such as bicycling. Important prerequisites for those improvements are understanding under which conditions it fails, some of which are presented here and some of which need more carefully controlled input to uncover.

References

[1] H. Ying, C. Silex, A. Schnitzer, S. Leonhardt, M.Schiek, "Automatic Step Detection in the Accelerometer Signal," 4th International Workshop on Wearable and Implantable

Body Sensor Networks, Springer Berlin Heidelberg, 2007.

[2] Jonathan Lester, Tanzeem Choudhury, Gaetano Borriello, "A Practical Approach to Recognizing Physical Activities," In *Proceedings of Pervasive 2006*, pp. 1-16, May 7, 2006.

[3] Sunny Consolvo, David W. McDonald, Tammy Toscos, Mike Y. Chen, Jon Froehlich, Beverly Harrison, Predrag V. Klasnja, Anthony LaMarca, Louis LeGrand, Ryan Libby, Ian E. Smith, James A. Landay, "Activity Sensing in the Wild: A Field Trial of UbiFit Garden," CHI 2008: 1797-1806.

[4] Tanzeem Choudhury, Gaetano Borriello, Sunny Consolvo, Dirk Haehnel, Beverly Harrison, Bruce Hemingway, Jeffrey Hightower, Predrag "Pedja" Klasnja, Karl Koscher, Anthony LaMarca, James A. Landay, Louis LeGrand, Jonathan Lester, Ali Rahimi, Adam Rea, Danny Wyatt, "The Mobile Sensing Platform: An Embedded Activity Recognition System," IEEE Pervasive Computing, vol. 7, no. 2, pp. 32-41, Apr-Jun, 2008.