Gait Cycle Analysis and Inconsistency Detection using Single-Axis Accelerometer

Som Bose ¹, Siddharth Srivastava ¹*, Aditya Agarwal ¹ and Vidushi Goyal ¹

¹ Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology Kharagpur, Kharagpur – 721302, India

Abstract. Gait inconsistencies are a direct measure of a variety of diseases. Despite the fact that they should be one of the simplest parameters to measure, the laboratory-based analysis of gait cycle makes it inaccessible to a lot of people due to large area requirements and high cost. This paper proposes a novel method for gait cycle calculation and inconsistency detection. A simple setup consisting of a single-axis accelerometer and a microcontroller has been used to analyze the accelerometer data in a computer simulation; hence, the feasibility of making a low-cost embedded system for gait cycle calculation has been shown.

Keywords: gait cycle, batch-wise analysis, single-axis accelerometer, gait inconsistency

1. Introduction

Gait analysis has primarily been carried out inside a laboratory with the use of computer vision technology and image processing algorithms. Quality research work has been performed in recognizing gait and its related parameters, primarily by the use of video analytics [1] and silhouettes [2]. In the work of Boulgoris et al. [2], the peaks observed in the autocorrelation of the foreground sum signal were used to determine the walking period. This method requires maintenance of labs and hence precludes the analysis of walk cycles of a human being from being done in a natural environment and on different terrains. The method being expensive is also not accessible to people in general.

Gait analysis related work has also been performed using FSR (Force Sensitive Resistors) by measuring the pressure beneath the foot [3]-[5]. These methods use FSR signal and pattern recognition algorithm for gait cycle detection.

More recent works [6]-[8] have been focusing on the use of MEMS-based devices like accelerometers and gyroscopes. Xuana et al. [7] describe a method in which inertial signal data passes through a low pass filter, and then various data analysis techniques are used for gait analysis.

Gait irregularities including slow pace of walking or variations in stride may be an indicator of inhibited illnesses. Studies have suggested that walking irregularities may even be a sign of cognitive decline in addition to being an indicator of chronic walking disorders. Gait parameters such as gait velocity, cadence and stride length have shown to be associated with cognitive decline. Gait analysis can be used to assess symptoms of Parkinson's disease, which is characterized by abnormalities such as involuntary motor movement or dyskinesia. Dyskinesia can be precisely described in terms of rhythm, speed, duration and pattern of motor movements.

The structure of the paper is as follows: Section 2 describes the proposed solution and algorithm. Section 3 elaborates the experimental results. Section 4 concludes the paper by showing the future scope of this work.

*Corresponding author. Tel.: +9122-29662049.
E-mail address: sid.m.srivastava@gmail.com.
2. Proposed Solution

The paper suggests a novel method to extract data from a single-axis accelerometer, digitize it and then use an algorithm based on experimental observations so as to get the required cycle time. It is critical that the entire operation is done on a data acquisition system, which must be carefully placed.

The accelerometer should see a good amount of swing in the component of gravitational force it experiences while the person is walking. Hence, the entire system is placed on the person’s leg, as shown in Fig. 1.

The sensor acquired data needs to pass through an algorithm. An infinite number of values cannot be stored on an embedded processor, and hence old values must be deleted as new data is acquired.

Thus, an algorithm has been developed that makes the system batch-wise real-time, i.e., data for N samples is collected and processed to give an output, and then another set of N samples is taken.

If the digital output values of the accelerometer after attaching the device to the person’s leg are recorded, we observe a reasonably periodic curve with peaks and troughs. These peaks and troughs occur when the foot is at maximum deviation on either side. Hence, the aim of calculating the gait cycle time is narrowed down to capturing the peaks or the troughs in the accelerometer data. The difference in time between two consecutive peaks provides the cycle time, and the deviation from mean will give the maximum angular deviation that can occur, by

$$\theta = \cos^{-1}\left(\frac{g_{\text{obs}}}{g}\right)$$

where $g_{\text{obs}}$ is the observed value from accelerometer and $g$ is the value when the accelerometer axis coincides with the gravitational force direction.

2.1. Algorithm

The algorithm described in this section focuses on calculating cycle time and angular deviation from the mean while walking. The data received from the accelerometer is noisy. This would create a lot of false positives if we were to apply simple difference-based peak finding. Hence, we carry out a first order difference-based local minima/maxima finding and then apply the same to the set of extrema we get after the first iteration. However, before the main algorithm is implemented, a real-time moving average filtering algorithm [9] is used, which makes sure that the noisy high frequency components are removed from the data acquired. This filtering can be done on the fly while data is being acquired.

After the moving average filtering removes the noise, we can apply our peak finding algorithm. The pseudo code for the peak finding algorithm is given below.

*Input:* A set of acceleration values $P$ in number and array name `data`

*Output:* Cycle time and deviation from mean: ‘meantime’ and ‘dev’

```plaintext
i = 0, 1, 2, · · ·, P − 1
1: for $i \leftarrow 1$ to $P − 1$ do
2: \hspace{1em} `sum \leftarrow sum + data[i]``;
3: end for
4: `mean \leftarrow sum/P``;
5: //finding initial local minima
6: `flagmin \leftarrow 0``;
```

Fig. 1: Data Acquisition System and Placement on Leg
7: for j ← 1 to P do
8:   if data[j] < mean and flagmin = 0
9:      flagmin ← 1;
10:  else if data[j] > mean and flagmin = 1
11:     peakcnt ← peakcnt + 1;
12:     peaks1[peakcnt] ← maximum possible value;
13:  end if
14:  if flagmin = 1
15:    if data[j] < peaks1[peakcnt]
16:       peaks1[peakcnt] ← data[j];
17:       minidx[peakcnt] ← j;
18:  end if
19: end if
20: end for
21: for j ← 1 to peakcnt-1 do
22:   if peaks1[j] < peaks1[j-1] and peaks1[j] < peaks1[j+1]
23:      peaks2[peakcnt2] ← peaks1[j];
24:      minidx2[peakcnt2] ← minidx[j];
25:      peakcnt2 ← peakcnt2 + 1;
26: end if
27: end for
28: dev ← mean - peaks2[0];
29: for j ← 1 to peakcnt2-1 do
30:   sum1 ← sum1 + minidx2[j] - minidx2[j-1];
31:   dev ← dev + mean - peaks2[j];
32: end for
33: meantime ← sum1/(peakcnt2 - 1)*0.05;
34: dev ← dev/peakcnt2;

The complexity of the cycle time detection algorithm is O(n), as in the worst case we would have three loops with n elements, but none of them nested. This means that the computation power required for its implementation is very low, for a considerably large sample space. The number of samples, however, will depend on the memory available with the microcontroller or processor in the system.

3. Results

For the purposes of our analysis, first, a collection of varying data was to be compiled. Using an accelerometer with a range of ±3g and a supply of 3.3V (ADXL335 by Analog Devices) tied just above the ankle, as explained before, the change in voltage data across the Z-axis of the device was recorded. This data was sent to an Arduino serial monitor (on the processor ATMEGA328), using a Bluetooth Mate (RN-41).

The received data (12000-15000 samples per set, with a sampling time of 1 ms) was then passed through the algorithm described in the solution. Four different data sets were collected for different speeds of walking - fast, slow, irregular and transitional from fast to slow. The algorithm was applied to each of these and the average cycle time recorded.

3.1. Filtering Simulations

In Fig. 2, the results of applying a moving average filter with a window size of 20 and 50 samples to the collected data are shown. Since our algorithm proposes a batch-wise real-time execution, the moving average filter was used to process each batch of data in real-time; while the person was walking, the data was recorded in multiple batches and then processed accordingly.

As can be seen, most of the noise is filtered out from the raw data in the 50-sample window filtered output and the periodicity of the cycles can be clearly seen. This also allows for better peak detection by removing the anomalous spikes that do not contribute to the actual calculation.

3.2. Peak Detection and Cycle Time Calculation
Previous research on cycle time calculations places a regular walk cycle as less than 1.1 seconds [8]. This is the benchmark for the correctness of our output.

The peak detection algorithm was applied to the filtered datasets as shown in Fig. 3. The peaks are detected twice: first on the original filtered data, and then on the first set of peaks. The reason for this is clear from the given plots - the first iteration may include some local minima that are not actually global; the second iteration thus helps us filter out these minima.

In each figure, the upper plot shows the filtered data from the accelerometer plotted against time. The green and the red curves denote the first and second iteration of finding the local minima. The cycle times are clubbed in the histogram (lower plot). The number of cycle times between the mean and standard deviation (μ-σ to μ+σ) is the first bar of the plot; the second bar is the number of cycles whose period is outside this standard range.

From the results given in Table 1, it was observed that the cycle time for a regular walk falls between 0.8 s and 0.9 s. Also, the number of peaks outside the standard statistical range is negligible. In keeping with the consistency of our algorithm, for the non-uniform walks, the cycle time was still averaged out to around 0.7 s; however, the histogram data, having 7-8 peaks out of the total falling outside the standard range, clearly signified that this data was irregular and the cycle time could not be accurately judged, just averaged to a certain value.

Thus, from such a result set, we can conclude the algorithm is consistent for both regular and irregular walking cycles, and clearly distinguishes between the two in terms of cycle time as well as the peak distribution.

Table 1: Results of Different Walk Types

<table>
<thead>
<tr>
<th>Walk Type</th>
<th>Peaks within (μ-σ to μ+σ)</th>
<th>Peaks outside (μ-σ to μ+σ)</th>
<th>Average Cycle Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast</td>
<td>12</td>
<td>2</td>
<td>0.811</td>
</tr>
<tr>
<td>Irregular</td>
<td>10</td>
<td>7</td>
<td>0.681</td>
</tr>
<tr>
<td>Slow</td>
<td>8</td>
<td>2</td>
<td>0.956</td>
</tr>
<tr>
<td>Transitional</td>
<td>11</td>
<td>4</td>
<td>0.822</td>
</tr>
</tbody>
</table>

Fig. 2: Filtered Output of Accelerometer Data: (a) Fast Walk (b) Irregular Walk (c) Slow Walk (d) Transitional Walk. Each figure – top to bottom: Original date, 20-sample moving average filtered output, 50-sample moving average filtered output.
4. Conclusion

We have presented a novel approach to calculate gait cycle time and also detect inconsistencies, if any. Our software solution requires light-weight computational power (such as that of Atmega328) and can hence be implemented directly on an embedded system.

![Fig. 3: Algorithm Output on Filtered Data: (a) Fast Walk (b) Irregular Walk (c) Slow Walk (d) Transitional Walk. Each figure – top to bottom: Peaks marked with red, Histogram showing statistical distribution of peaks within standard error range.]

The proposed algorithm has O(n) complexity in both filtering as well as peak detection. Our result is consistent with Bamberg et al. [8], where the paper matches its result with Massachusetts General Hospital, Bio Motion Laboratory. Also, our results for slow and fast walk cycles are consistent with results in Jordana et al. [10], where they measure variation in gait cycle with walking speed. Hence, our approach provides a direction towards embedded system based gait cycle detection.

5. References


