Improved real-time gait phase detection using simple force sensors, sequencing conditions and smart fault management

Clemence Drouot, Nathanael Jarrasse

Improved real-time gait phase detection using simple FSR sensors, sequencing conditions and smart fault management

Clémence Drouot and Nathanaël Jarrassé

***Abstract*— Gait event analysis by wearable devices is still a current challenge. Ground foot contacts, lower body segment accelerations or rotations or joint kinematics are commonly used, associated with functional analysis or machine learning algorithms, and currently allow to detect up to 5 different phases during a gait cycle. However, these detection are not always reliable or usable in real time because of calculation duration or sensor faults and drifts. To get more precise and realistic sequencing of the gait cycle, we chose to only use the information of heel or toe contacts with Force Sensing Resistors (FSR) to decompose the cycle in 8 different phases that we determined previously by a bottom-up analysis of the possible contacts during walk. Based on this definition different detection algorithms are presented, which introduces sequencing conditions and smart fault management of sensors information to improve the performance and robustness of the detection. Proposed algorithms are assessed on the recorded walking data of 12 participants and in an online experiment, showing an average of 99% recognition rate of gait phases during the walk. These proposed approaches could offer new possibilities for using simple and low-cost instrumented soles to perform quantitative assessment of walking strategies.**

I. Introduction

Older adults falls are a common, serious, and growing public health problem [1]. Those falls are induced by postural instabilities, muscle weaknesses or buckling of lower-limb joints [2] and can lead to severe injuries, long term impairment up to death (second leading cause of unintentional injury deaths worldwide according to WHO). Numerous technical devices are currently being developed to address this issue, such as advanced orthotic devices such as the C-Brace developed by Ottobock [3], exoskeletons such as the Keeogo from B-temia [4], smart canes [5], wearable airbags [6], etc. One key challenge for those devices is to be able to efficiently detect the fall through sensors in a fast or anticipatory way. One common approach for those systems is to measure and analyze some characteristics of the gait cycle to detect abnormalities, through the decomposition of the cycle into specific phases and comparison with reference characteristics.

Wearable solutions that are analysing the gait cycle can use information from force, angular and accelerometric measurements [7]. Historically, simple foot switches were used to detect *heel strike* and *heel off* events. For example, Lyons and al. [8] used such sensors and information to apply Functional Electrical Stimulation (FES) during patient walk. Low processing of kinematic data also shown good results in *heel strike* and *toe off* detection. For example, O'Connor and al. [9] used motion capture system to get vertical velocity of the foot or Hanlon and al. [10] used accelerometers to measure the knee vertical and foot horizontal accelerations. However, the first solution currently uses motion capture which limits the use to clinical indoor trials and the second one tends to lose accuracy when used in real-time. To increase precision, Pappas and al.[11] used a sole fitted with 3 sensors (Force Resistive Sensors (FSR)) and a gyroscope measuring the rotational velocity of the foot. Thanks to this hardware, they were able to differentiate 4 different phases during one gait cycle (*Heel strike*, *Stance*, *Heel off* and *Swing*), cadenced by 7 possible triggering events, with a maximum delay of 90 ms. Skelly and al. [12] used a sole fitted with four individual FSR sensors — two at the heel, one under the head of the first metatarsal (medial sensor), and one under the head of the fifth metatarsal (lateral sensor). These sensors, processed by a fuzzy logic classifier, added to a knee angle sensor allowed them to distinguish 5 phases per gait cycle (*Heel strike*, *foot flat*, *Heel off*, *Toe off*, *Knee max*) after having added supervisory rules on the primary gait phase estimations. Williamson and Andrews [13] also differentiated 5 gait phases in real-time, using tree-directional accelerometer on the shank and machine learning.

Wearable solutions based on IMU sensors are however time and energy consuming and have a tendency to drift through time[7] requiring regular calibration. FSRs, while commonly used, are known for their limited reliability and fragility leading to inconsistencies in the measurement. Detection can also show incoherent states because of the difficulty of defining mappings between the ground contacts states and associated gait phases. Indeed, most of exiting studies do not take into account the cycling aspect of the gait phases and generally considers 4 or 5 possible states at most while biomechanic literature (such as Whittle[15]) defines up to 8 different phases. Besides, most of these gait detection algorithms are using machine learning or artificial intelligence [7]. But, as highlighted by Burkart and al. [14], there are some domains that require explainability, such as the medical area because of the complexity, unpredictability and variations of the functioning of the human bodies.

We believe that sequencing more precisely - and in an explainable way- the gait cycle using information on feet (heel and toes) contacts with the ground, would allow to recognize more efficiently abnormalities in the walking of human subjects. In this article, we thus introduce a new sequencing of the gait cycle, based on the possible ground contacts of the heel and toes of both feet. We also propose a dedicated

Clémence Drouot and Nathanaël Jarrassé are with the Institute of Intelligent Systems and Robotics, Sorbonne Université, with CNRS UMR7222, and with INSERM U1150, Paris, France. E-mail: surname.name@sorbonne-universite.fr.

gait phase detection algorithm which takes advantage of the cyclical aspect of walking and takes into account the possible defects of the sensors of the instrumented insoles.

In the following section, we first analyse the gait cycle with respect to the possible combinations of feet contact with the ground to determine the expected sequence. We then introduce three gait-phase detection algorithms, based on this definition. First a simple algorithm that links the phases directly to the detected ground contact. Then, cycling conditions are added to get more relevant on the phase detection based on the previously detected phase. And finally, a solution is proposed to take into account the possibility of a sensor fault while pursuing the phase detection. A dataset of signals collected on a group of asymptomatic participants walking with instrumented soles is then created (Section IV) and used to assess the performance of proposed algorithms in Section V. In this section, the results of an online detection experiment are also shown, and finally discussed in Section VI.

II. Definition of a gait cycle

A. Global definition

The walking can be described as a succession of repetitive events and movements of the lower body : the *gait cycle* is thus defined as the time interval between two successive occurrences of one of these events[15]. In the sagittal plane, three joints (the ankle, knee and hip) and four segments of body are involved in the lower limb movement, as it can be seen in the decomposition of the Figure 1.

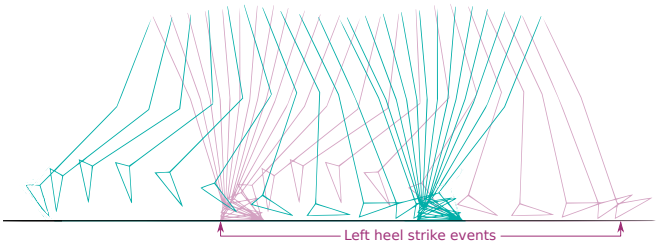

Fig. 1: Position of 3 DoF legs in the sagittal plane at 40 ms intervals during a single gait cycle. Adapted from [15] with the right leg in green and the left one in red

Numerous characteristics of one given leg thus exhibit a typical pattern during walking and can be used as a reference to describe the cycle, such as the joint angles or segments orientation. The other leg goes through the exact same series of events, but shifted in time by half a cycle duration. In this study, we define the beginning of the gait cycle by the left heel strike event and focus our analysis on the ground contact sequences to define and decompose the gait cycle.

B. Ground contacts

Gait phase detection literature generally only considers 2 to 5 different phases during a cycle. We here propose a more precise decomposition based on systematic ground contacts analysis. We here consider two ground contact areas for each foot, heel and toes. To identify all possible occurring contacts during the walking cycle, we use a bottom-up strategy, considering that at least one foot part remains in contact with the floor anytime and that only one foot part can land or leaves the ground at each instant.

Thus, starting by a ground contact combination unavoidable during the walk (such as the Left Heel/Left Toe combination when the right leg is swinging), the next possible events for the 4 possible foot parts are evaluated (considering a forward walk). In this configuration for example, either the Left Heel leaves the floor, or this is the Right Heel that lands on ground first.

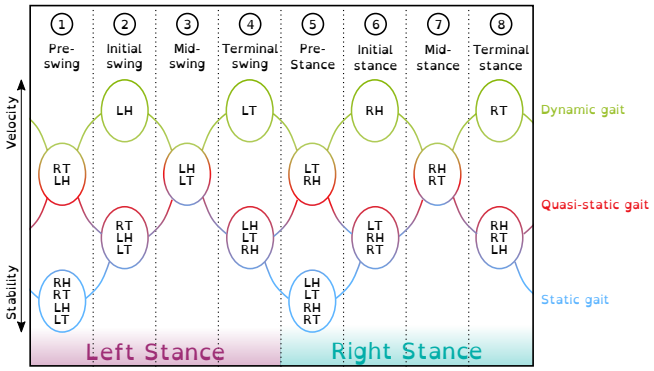

Fig. 2: State machine of the different possible ground contacts during a gait cycle. R refers to the Right foot. L the left one. H refers to the heel and T the toes.

This analysis leads to the graph of the Figure 2 which represents the possible ground contact sequences observable during walk. R and L letters corresponds to the Right of Left foot and H or T refers to the Heel or Toe part of the foot. In this figure, contact combinations have already been organized around similarities, highlighting the possibility to define **8** different phases sequencing the walking gait cycle. In what follows, those will be named and numbered as shown in this Figure. This organization also highlights the different ground contact sequences which minimizes the number of contact with the ground (upper part of the graph) and at the opposite maximizes them (lower part). This is why they have been named *dynamic*, *quasi-static* and *static gaits*. A visualisation of each of these gaits is shown in the Figure 3.

Based on the proposed gait cycle decomposition, we introduce in the next section a dedicated and improved walking phase detection algorithm.

III. Gait phase detection algorithms

A. Direct ground contact decoding

The standard way to detect the gait phase P is to associate each one to its possible ground contacts, as shown in Figure 2 leading to the following correspondence Table I. The binary state of right and left (R and L) heel and toes (H and T) sensors are combined in a quadruplet and converted into a unique decimal variable G_c, varying between between 0 and

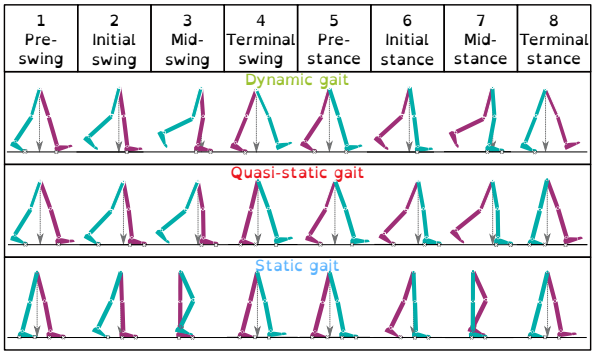

Fig. 3: Leg positions according to their contacts with the ground. The grey arrow represents the approximate projection of the center of mass on the ground.

15, representing the 16 possible states corresponding to all the combinations of the 4 sensors states, as summarized in Table I.

Thus, the algorithm for gait phase P detection is the following:

input : RT, RH, LT, LH
parameter: $G_c \leftarrow \text{bin2dec}('RT\ RH\ LT\ LH')$
prev. phase P_{prev}
output : phase P
switch G_c **do**
  case *0* **do**
    $P \leftarrow NW$;
  case *1* **do**
    $P \leftarrow ISw$;
  ⋮
  case *15* **do**
    switch P_{prev} **do**
      case *8,1* **do**
        $P \leftarrow PSw$;
      case *4,5* **do**
        $P \leftarrow PSt$;
    end
end

Algorithm 1: Identification of the gait phase depending on the detected ground contact

B. Considering the cycling aspect

The proposed algorithm do not check if the sequences of detected phases is coherent with the sequential graph of Figure 2. We here propose to derive it into a *weighted oriented graph* that will be used to confirm the veracity of the detected phase within in a larger sequence. This will allow us to determine the current gait phase P based on both ground contacts G_c and previously detected gait phase P_{prev}: when a gait phase P is determined from ground contacts, it has to be following some specific phases (i.e. to respect a given sequence) to be really detected by the algorithm, which otherwise will detect it as a Non Walking (NW) state.

To do so the previous walking sequential graph is turned into a graph with the values of ground contact G_c determine by the table I as seen in Figure 4. A graph is a finite set of *nodes*, linked together in pairs. The bind that links 2 nodes is

G_c	RT	RH	LT	LH	phase P acronym
0	0	0	0	0	**NW**
1	0	0	0	1	ISw
2	0	0	1	0	TSw
3	0	0	1	1	MSw
4	0	1	0	0	ISt
5	0	1	0	1	**NW**
6	0	1	1	0	PSt
7	0	1	1	1	TSw
8	1	0	0	0	TSt
9	1	0	0	1	PSw
10	1	0	1	0	**NW**
11	1	0	1	1	ISw
12	1	1	0	0	MSt
13	1	1	0	1	TSt
14	1	1	1	0	ISt
15	1	1	1	1	PSw or PSt

TABLE I: Coding table of the ground_contact variable and associated gait phase P (where NW means *Not Walking*)

defined as an *edge*. A specific kind of graph will be used here : a *weighted oriented graph*. This means that each edge has a associated *direction* and *weight* (or *cost*). Then, a Dijkstra function [16] can be used to determine the shortest path in the defined graph and evaluate its associated cost.

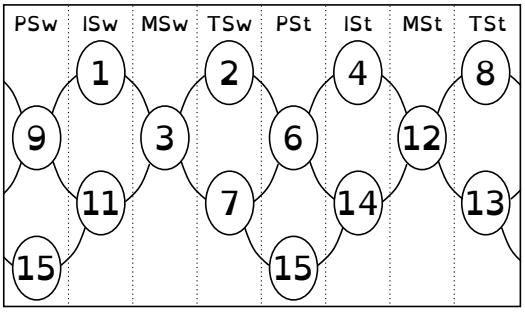

Fig. 4: Numbered translation of the state machine of the Figure 2

To numerically encode the graph of Figure 4, a 16×16 matrix D is used. Each row represents the previous estimated ground contact number (previous *node*) and each column the current one (current *node*). Thus, an edge between two nodes is representing by a numbered *cost* $d_{i,j}$ stored in the appropriate element of the matrix. In this approach, all edges are considered as bidirectional and with the same cost $usual_cost = 1$. Thus, the edge between ground contacts 9 and 1 (i.e. $RT\ LH$ and LH) for example will be mathematically defined as $d_{1,9} = d_{9,1} = 1$. All non-existent edges are set to an high cost (100) and each node is linked to itself by an edge of 0 cost.

The gait phase detection algorithm is here using the Dijkstra method to determine, in this graph D, the shortest possible path between two successive ground contacts and the correspondent cost d_tot. It has been observed on preliminary data analysis that it is common that 2 sensors can change of state simultaneously. To take this possibility into account, the algorithm considers that the path belongs

to walk if the total cost d_{tot} is lower or equal to 2. If so, the current gait phase will be set to the phase corresponding to the current ground contact thanks to the *get_phase* function which links both. Otherwise, the wearer is considered as not walking and the detected phase is set to 0 (NW) even if the ground contact is one belonging to the walking graph. In the specific case where the total cost d_{tot} is equal to 0 (i.e. 2 successive times with the same combination), the algorithm maintains the previous detected phase. All this corresponds to the algorithm 2.

input : RT, RH, LT, LH
parameter: $G_c \leftarrow \text{bin2dec}('RT\ RH\ LT\ LH')$
$G_{c_{prev}}$ (previous ground contact)
P_{prev} (previous phase)
D
output : P

$[d_{tot}, path] = dijkstra(D, G_{c_{prev}}, G_c)$;
if $d_{tot} == 0$ **then**
  $P \leftarrow P_{prev}$;
else if $d_{tot} <= 2$ **then**
  $P \leftarrow get\_phase(path(end))$;
else
  $P \leftarrow NW$;
end
$G_{c_{prev}} \leftarrow G_c$
$P_{prev} \leftarrow P$

Algorithm 2: Identification of the gait phase depending on the cycling sequence with one phase skip allowed

C. Managing the possibility of sensor fault

To consider the possibility of a sensor fault, new nodes and edges are added to the previous graph. The translations of this graph without the glitched sensor involved have been written, considering that only one sensor is glitching at a time (assuming that a glitch remains an uncommon event). These graphs are the colored ones shown in Figure 5. To allow some adjustments of the variables, the edges of these graphs have their own costs. In what follows, they will be named g_cost and be set to a higher value than the standard cost of 1 to give priority to usual behaviors.

There are now several nodes that have the same G_c value but which refers to different phases. For example, $G_c = 2$ can refer to the MSw phase when the LH sensor is glitching or to the TSw in the usual case or to the PSt phase when RH sensor is glitching. Thus, to differentiate each node despite the fact that they are sharing the same G_c value, a new node numbering is implemented as follows :

$$\text{node\_num} = P + 9 * G_c \quad (1)$$

Thus, nodes have now values between 0 and 143 so the matrix (D_g) defining the new graph will now be (143×143). The equation 1 is used to find the line and column number corresponding to the previous and current node of the graph. So the edge costs of the 5 small graphs are implemented in D_g and to gather them all, edges are added between each "usual nodes" and its glitched counterpart. An example for the node corresponding to $G_c = 15$ and $P = 5$ is shown in the figure 5. A glitch_appearance_cost and glitch_disappearance_cost are assigned to the latter, with a value of between the usual_cost and the g_cost to make sure that a (return to) usual situation will be preferred to a glitched one. Here, they are set to 1, 1.1 and 1.2.

Eventually, now that the definitions of this graph's nodes depend on both the ground contact and the phase, new nodes corresponding to the ground contact detected during a NW phase have to be added. Thus, the ground contact sequences of the usual graph are also added for $P = NW$.

The next phase being unknown at each time step, the algorithm will evaluate the total cost of each path that possibly link the current node and the ones corresponding to the detected ground contact and all the possible next phases. Thus, a temporary cost variable C_{temp} is used to store the smallest path cost found while evaluating all the possibility for the next phase. The new algorithm is thus as follows, in which *find_node* is the implementation of the equation 1 and *find_phase_number* its corollary:

input : RT, RH, LT, LH
parameter: $G_c \leftarrow \text{bin2dec}('RT\ RH\ LT\ LH')$
$G_{c_{prev}}$
P_{prev}
D_g
C_{temp}
$cost_{temp}$
output : P

$C_{temp} = 100$;
for $p = 1$ **to** 8 **do**
  $[cost_{temp}, path] =$
  $dijkstra(D_g, find\_node(G_{c_{prev}}, P_{prev}, find\_node(G_c, P))$;
  if $cost_{temp} == 0$ **then**
    $P \leftarrow P_{prev}$;
  else if *($cost_{temp}$ <= 2) & ($cost_{temp} < C_{temp}$)* **then**
    $C_{temp} = cost_{temp}$;
    $P \leftarrow find\_phase\_number(path(end))$;
  end
end
if C_{temp}==*Inf* **then**
  $P \leftarrow NW$;
end
$G_{c_{prev}} \leftarrow G_c$
$P_{prev} \leftarrow P$

Algorithm 3: Identification of the gait phase depending on the cycling sequence with one phase jump allowed and sensor glitches management

IV. Materials and methods

We conducted an experimental campaign to collect ground contact data during the walk of different individuals to assess the performance of the proposed detection algorithms.

A. Instrumented soles

The figure 6 shows a picture of a participant wearing the recording device, with a zoom on one of the sole disposed on one shoe. Each sole is supporting 2 FSRs (pointed by the white arrows in the figure), one at the front (detecting toe part contact with the ground) and one at the back (for

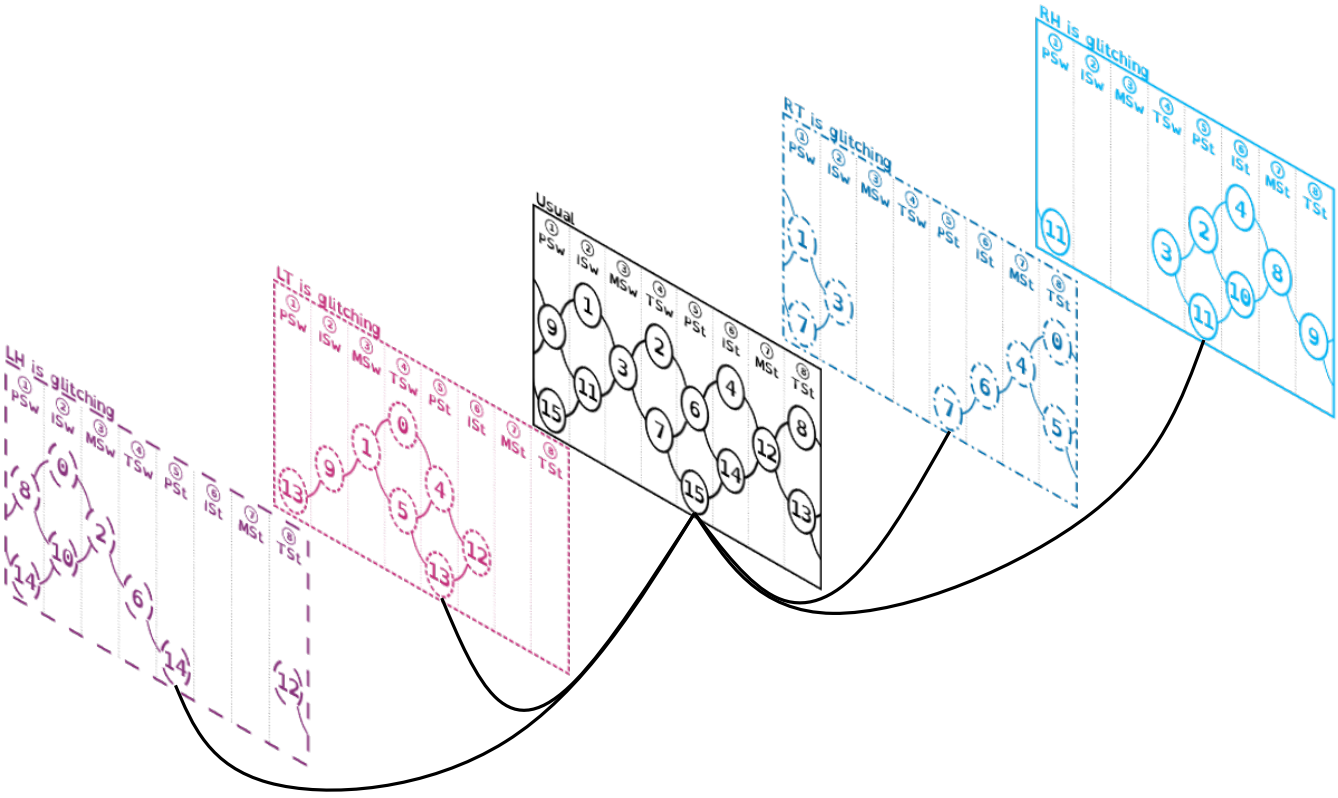

Fig. 5: Graphical showing of the graph defining possible ground contact sequences during walk, including contact glitching. Links between the 5 graphs are not represented except for the node corresponding to a ground contact equal to 15 in phase 5 in the usual situation

the heel part). A smooth sole (at the bottom of the figure) is added over the one that bears the FSRs to improve the user feel and the pressure distribution on the sensors. The placement of each FSR is adapted to the wearer foot before the recording.

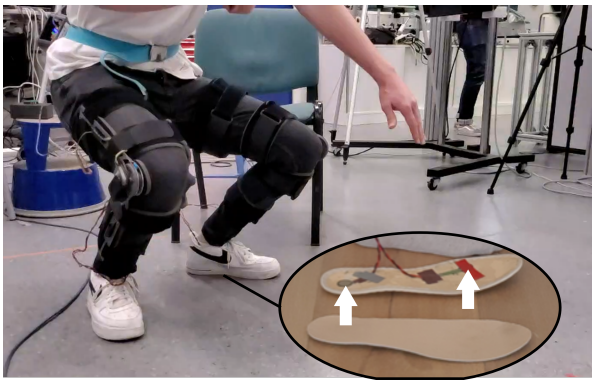

Fig. 6: Picture of 2 FSRs set on the right sole.

The 4 FSR raw signals are collected by an Arduino MKR1000 which is able to either send the recorded raw data to a computer via Wi-Fi for offline analysis or applied itself the detection algorithm and send visible or audible feedback. Accordingly, colored LEDs and a buzzer are controlled by the Arduino to provide possible feedback to the experimenter. Data are collected and send via Wi-Fi at a rate of 50Hz.

B. Data treatment

FSR information are collected by the Arduino (through a 12 bit resolution ADC) and sent to the computer. Examples of received data during a gait cycle are shown in the figure 7 A. and A' for each sole. These sensors are here used as simple ground contact detectors : their values oscillate between a minimum and a maximum depending if the foot part is in the contact with the ground or not. They do not provide precise measurement of the amount of force applied.

FSR signals can be influenced by several phenomenon. For example, sensors can detect the pressure exerted by the tightening of the shoe, this is especially true during swinging phase when the toes extend and the shoe bends a little. It explains why the raw FSR signal of the right toe shown in figure 7.A'. doesn't get down to 0 during the flight phase of the right foot. Besides, walkers can be sometimes unbalanced, especially when they are asked to exhibit a particular gait and have to focus on their leg movements. So, as the toe FSRs do not cover the whole width of the soles, their high level value can also vary during standing, as shown in figure 7.A' : after having risen above the threshold, it can be seen that the signal of the right toe FSR oscillates a little rather than just going up. For all these reasons, the binarization is done with a personalized threshold chosen for each walker and each FSR. This threshold is manually defined as an intermediate value between the maximum

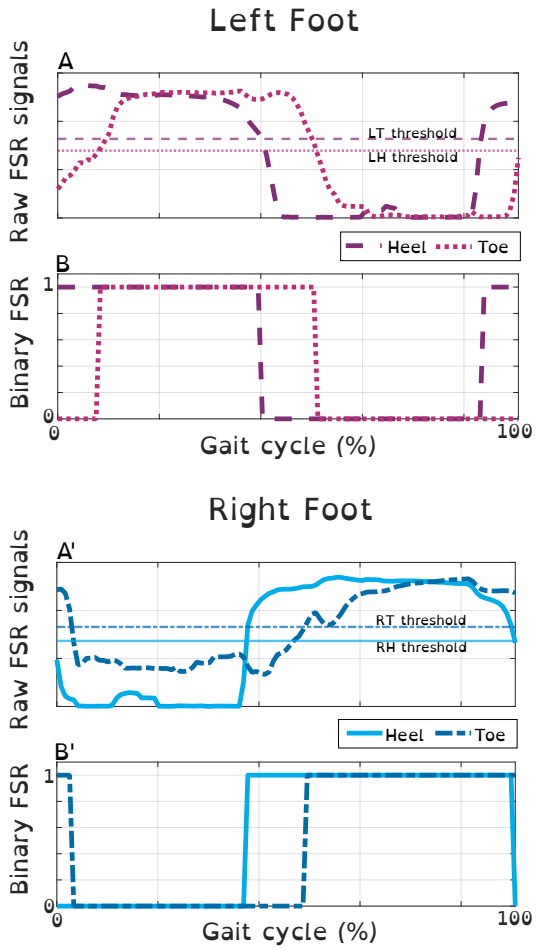

Fig. 7: FSR raw and binarized signals for the left and right foot during a walking gait cycle

and minimum FSR values reached by the user during a preliminary walking experiment. The chosen thresholds for the participant of which data are shown on the figure 7 are visible on graphs A. and A'. and the result of the binarization on the graphs B. and B'.

C. Participants and protocols

1) Participants: All participants were adults, aged between 20 and 30 years old, with no evidence of gait abnormality. They were asked to wear the 2 soles in their own shoes and the collecting data card were worn on a belt. The study was carried out in accordance with the recommendations of the Sorbonne Université ethic committee CER-SU, which approved the protocol. All participants provided informed consent to participate in the study. The protocol was performed in accordance with the Declaration of Helsinki.

2) Protocol used to collect preliminary data: The first experiment, which goals was to collect walking data, were conducted on 12 asymptomatic participants. They were asked to walk on flat floor. The recording area were composed by a straight line and a slight turn (i.e. without having to rotate on themselves), corresponding to a 30 meter walk. Each participant were going through this track 6 times, with a break every 2 repetitions.

3) Protocol used to assess real time detection: The validation experiment consisted on the recording of different kinds of walk (according to its speed or stride length) exhibited by 3 participants. On a straight flat track of 10 meter long, they were first asked to go through 2 times, walking as they are used to. Then they were asked to go back and forth with a high speed and wide strides and to gradually lower both until they reached a slow walk with tight steps. During this experiment, the wearer legs were filmed and the output of a Graphical User Interface (showing through which ground contacts the participant goes) has been recorded.

V. Results

A. Qualitative performance offline

The figure 8 shows the output of the algorithms during 3 successive gait cycles of one participant as a representative detection example. Phases are detected in the correct order, from PSw to PSt, for the third algorithm ("Fault robust") while the others show inaccuracies during the second cycle.

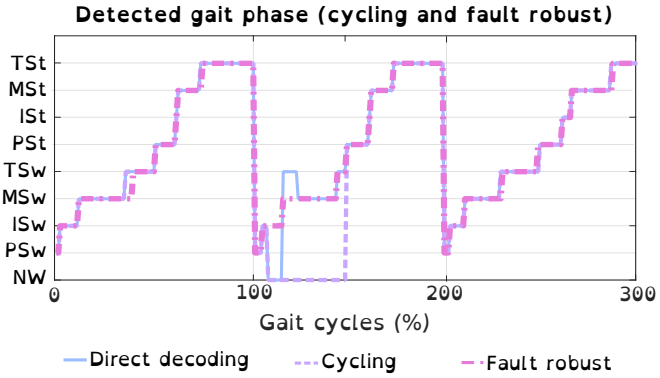

Fig. 8: Phases return by the algorithms for one participant during 4 gait cycle

A sequence of the detected phases during a damaged gait cycle is shown on the figure 9 for all the presented algorithms. It can be seen that the first approach, that directly links phases to sensed ground contact, do not manage to identify the phases that succeeded to the second detected phase and then detects a sequence that is not progressive despite the fact that the wearer was still walking straight. The cycling approach do not detect the gait phases at the beginning of the cycle but return a "Not Walking" (NW) of phase instead. The most robust algorithm detects phases going incrementally from PSw to PSt which was what the participant were exhibiting.

B. Quantitative performance offline

Figure 10 shows the percentage of time instants that were detected as belonging to the walk cycle (i.e. all phases different from NW. It was computed on all the recordings of all participants (8 min of recording for each, i.e. 8 475 data point with a 50 Hz frequency).

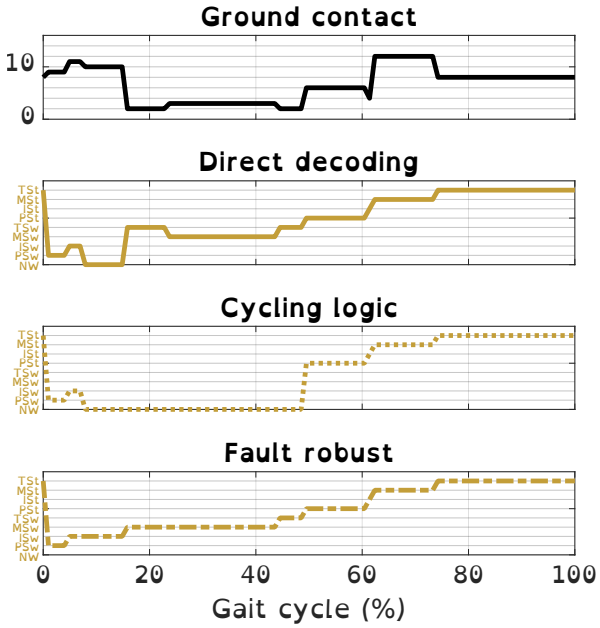

Fig. 9: Portion of detected phases for one participant during a gait cycle including a sensor fault

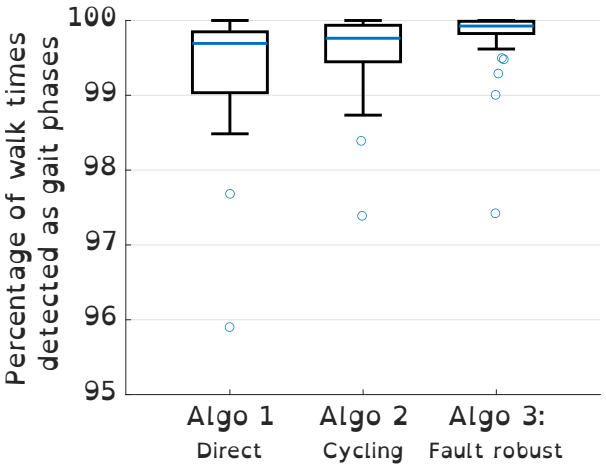

Fig. 10: Rate of gait phases detected during walk of all participant for the 3 proposed algorithms.

The blue line on each boxplot represents the median of the data. Thus, the 3 algorithms detected walk phases during more than 99% of the walk. Besides, the rate of detected phases increases with the evolution of the algorithm (for all the participants) and is equal to at least 96% approximately.

C. Online detection of gait phase with GUI

During the online experiment, participants were asked to exhibit different kind of walk (i.e. with different speeds) while the last algorithm where evaluated the walking phase at any time and sent it to the computer. Figure 11 shows freeze frames of a gait cycle with wide strides (*dynamic* gait of Figure 2). Each frame is associated to the matching picture that appeared on a dedicated Matlab GUI fed by the Arduino. This GUI reproduces the graph from Figure 2, with the phase names at the top and the possible ground contacts under. It displays in dark blue the current sensed ground contact and its relative detected phase. Besides, the information of the previous sensed ground contact remains on light blue to keep a residual track of the sequence the wearer goes through. An example of the sequences displayed on the GUI during others kinds of gait are illustrated on Figure 12, with the quasi-static gait to the left and the static gait one to the right (as defined in Figure 2).

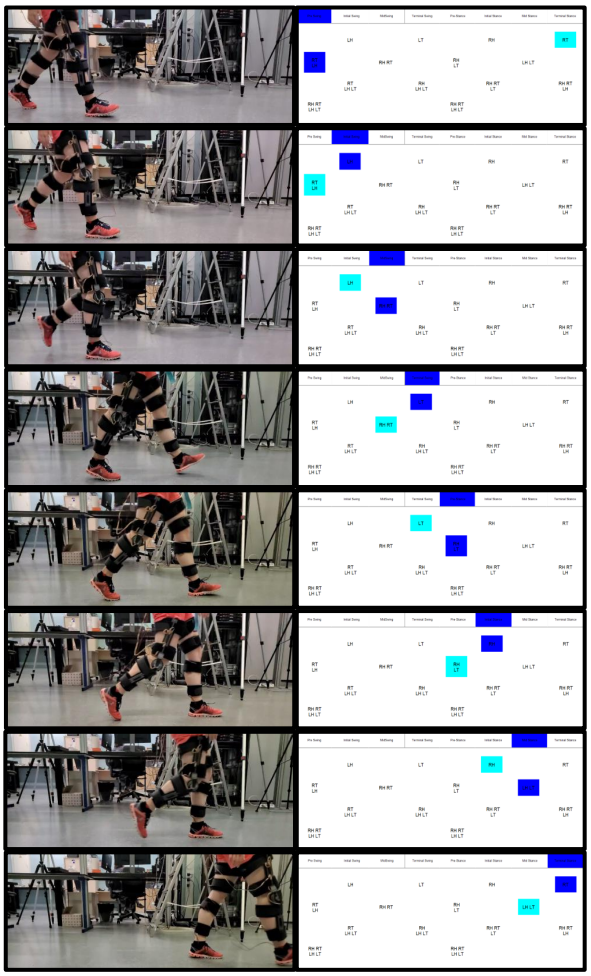

Fig. 11: Freeze frames of a *dynamic* gait with each associated state of the GUI

VI. Discussion & Conclusion

As it can be seen on Figure 8, the detection algorithm correctly detects the 8 walking phases, from PSw to PSt, while a participant is walking normally. The example of Figure 9 shows that the third algorithm allows to retrieve walking phases that were missed or incorrect because of FSR sensor faults or misreads. The third algorithm using sequence conditions and fault management features thus allows to get more reliable information about the gait evolution with a precise decomposition of the walking gait in 8 different phases, offering improved possibilities to characterize and

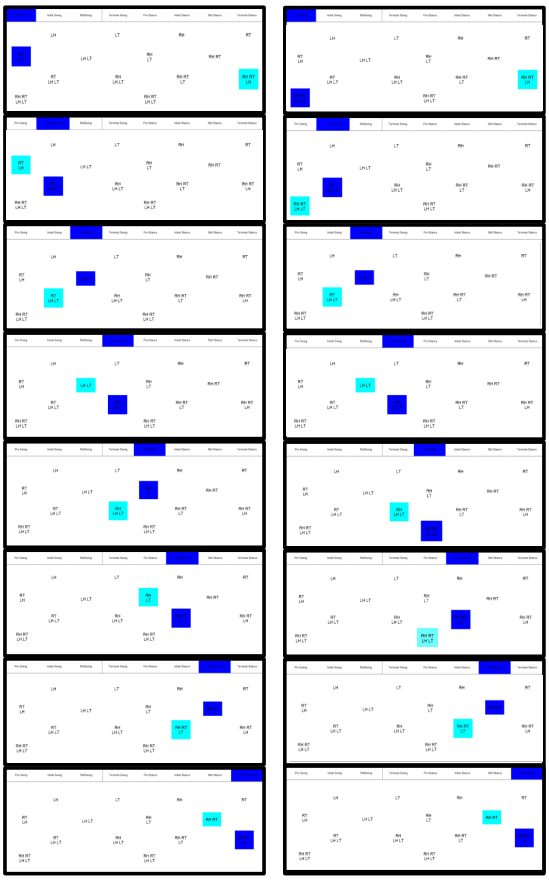

Fig. 12: Freeze frames of the GUI for a *quasi-static* gait to the left and a *static* gait to the right

analyze the walking strategies. This algorithm to detect gait phases in more than 99% of the walking time for 12 participants. However, a wider quantitative performance evaluation will have to be done to assess more in depth the relevance of the detection, with external control data (with a motion capture system for example or a force instrumented walking platform). Robustness towards variations of multiple parameters (walking rhythms, ground surface and even physiological condition of participants with more unusual walking strategies) should also be studied in future experimental campaigns.

Additionally, proposed algorithm currently only works during walking. If the system is used during other activities, numerous foot contact states will be shown in sequences unknown to the algorithm and will thus lead to numerous incorrect gait phase detection or constant NW phase detection. A perspective of this work will therefore be to implement recognition of the specific sequences of other activities to be able to adapt the sequencing conditions accordingly and be able to characterize gait phase in more realistic everyday life scenarios.

The real time experiment shows that our device (instrumented soles and embedded electronic) and proposed algorithm can be used in an online manner. During the walk of participant, the GUI correctly displays in real-time the sequences of gait phases and ground contacts and allows to observe subtle variations in walking patterns. This could be useful for clinicians to get more precise information on a patient gait such as walking asymmetries, nature of ground contacts patient is exhibiting (assessment of foot rolling impairment), the duration of each phase or even information of any other sensor (such as kinematics measurements) that could be categorised depending on the analysed gait phase(s). A specific sound or visual feedback could also easily be added for the clinician or the user to have real-time information and take them into account, for rehabilitation purposes for example. We believe that proposed gait phase detection approaches could pave the road toward generalization of the use of simple and low-cost instrumented soles to perform clinical quantitative assessment of walking or to offer smarter rehabilitation care.

REFERENCES

[1] Florence CS, Bergen G, Atherly A, Burns E, Stevens J, Drake C. Medical Costs of Fatal and Nonfatal Falls in Older Adults. J Am Geriatr Soc. 2018 Apr;66(4):693-698.

[2] Salzman, B. (2010). Gait and balance disorders in older adults. American family physician, 82(1), 61-68.

[3] Auberger, R., Pobatschnig, B., Russold, M., Riener, R., & Dietl, H. (2020). Activities with a Microprocessor-Controlled Leg Brace for Patients with Lower Limb Paralysis: A Series of Case Studies. IEEE Transactions on Medical Robotics and Bionics, 1–1.

[4] Mcleod, J. C., Ward, S. J. M., & Hicks, A. L. (2019). Evaluation of the KeeogoTM Dermoskeleton. Disability and Rehabilitation: Assistive Technology, 14(5), 503–512.

[5] Trujillo-León, A., Ady, R., Reversat, D., & Bachta, W. (2020). Robotic cane controlled to adapt automatically to its user gait characteristics. Frontiers in Robotics and AI, 105.

[6] T. Tamura, T. Yoshimura, M. Sekine, M. Uchida and O. Tanaka, "A Wearable Airbag to Prevent Fall Injuries," in IEEE Transactions on Information Technology in Biomedicine, vol. 13, no. 6, pp. 910-914, Nov. 2009

[7] Rueterbories, J., Spaich, E. G., Larsen, B., & Andersen, O. K. (2010). Methods for gait event detection and analysis in ambulatory systems. Medical Engineering and Physics, 32(6), 545–552.

[8] Lyons, G.M.; Sinkjaer, T.; Burridge, J.H.; Wilcox, D.J. (2002). A review of portable FES-based neural orthoses for the correction of drop foot. , 10(4), 260–279.

[9] O'Connor, C. M., Thorpe, S. K., O'Malley, M. J., & Vaughan, C. L. (2007). Automatic detection of gait events using kinematic data. Gait and Posture, 25(3), 469–474.

[10] Hanlon, M., & Anderson, R. (2006). Real-time gait event detection using wearable sensors. Gait & Posture, 24, S127–S128.

[11] Pappas, I. P. I., Popovic, M. R., Keller, T., Dietz, V., & Morari, M. (2001). A reliable gait phase detection system. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 9(2), 113–125.

[12] M. M. Skelly and H. J. Chizeck, "Real-time gait event detection for paraplegic FES walking," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 9, no. 1, pp. 59-68, March 2001

[13] Williamson, R., & Andrews, B. J. (2000). Gait Event Detection for FES Using Accelerometers and Supervised Machine Learning. IEEE TRANSACTIONS ON REHABILITATION ENGINEERING, 8(3), 312–319.

[14] Burkart, N., & Huber, M. F. (2021). A survey on the explainability of supervised machine learning. Journal of Artificial Intelligence Research, 70, 245–317.

[15] Whittle M. "Gait Analysis, an introduction" (2007)

[16] Dijkstra, Edsger W., et al. A note on two problems in connexion with graphs. Numerische mathematik, 1959, vol. 1, no 1, p. 269-271.