Computer-based Application to Assess Gross Motor Skills Using Microsoft Kinect Sensor

by

Arina Bragança Moraes

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Applied Science (MASc)
in Natural Resources Engineering

The Faculty of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Faculty of Graduate Studies/Faculté des études supérieures

Title of Thesis Titre de la thèse	Computer-based Application to Assess Gross Motor Skills Using Microsoft Kinect Sensor
Name of Candidate Nom du candidat	Bragança Moraes, Arina
Degree Diplôme	Master of Science
Department/Program Département/Programme	Engineering
Date of Defence Date de la soutenance	March 07, 2018

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Brahim Chebbi
(Co-Supervisor/Directeur de thèse)

Dr. Line Tremblay
(Co-Supervisor/Co-directrice de thèse)

Dr. Brent Lievers
(Committee member/Membre du comité)

Dr. Tilak Dutta
(External Examiner/Examinateur externe)

Approved for the Faculty of Graduate Studies
Approuvé pour la Faculté des études supérieures
Dr. David Lesbarrères
Monsieur David Lesbarrères
Dean, Faculty of Graduate Studies
Doyen, Faculté des études supérieures

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

Abstract

The performance of a fundamental motor skill in early childhood can be investigated using diverse assessment tools. Although the classical tests are reliable and useful for motor skills assessment, they have some limitations. The launch of the Microsoft Kinect marked a revolutionary advancement for developers thanks to the depth camera and its affordable price. This work aims to develop a reliable computer-based application using the Microsoft Kinect sensor to implement the third version of the Test of Gross Motor Development (TGMD-3). The assessment consists of customized algorithms that verify if the 3D position of the most relevant joints for each subtest varies along time according to the respective performance criteria. The proposed system returns an immediate feedback to the participant, indicating if s/he passes or fails the selected subtest. The results revealed the computer-based application for assessing gross motor skills is accurate, although it is limited by the space requirements.

Keywords

Kinect Sensor, Gross Motor Skills, Virtual Reality, Assessment, Computerized Test.

Acknowledgments

I sincerely thank my supervisor Dr. Brahim Chebbi for his patience, guidance and support during the development of this thesis. He trusted me with this project when I asked him about a master's opportunity supervised by him. This experience has brought many positive impacts to my academic, professional and personal life, and I will be always grateful for that. He is a great person and an example of a true professional who loves to teach.

I also thank Dr. Line Tremblay for being always positive and supportive regarding to the project. She is a great person to work with — a happy, motivated and ready to help person who is passionate for what she does.

I must thank my mother for believing in my potential to grow and encouraging me to leave my comfort zone. Without her I might not have found the courage to begin this journey in another country. Mom, I love you!

Finally, I acknowledge all other great people that have helped me through this accomplishment. This includes Larinni, Natalie, Maureen, Daniel, John and my uncle Rodrigo. Thank you all for the different point of views, for caring about me and making this passage smoother.

Table of Contents

List of Tables

List of Figures

1 Introduction

Usually the term *motor skill* is used to describe a type of skill that emphasizes the body movement and how it is performed (Newell, 1991). According to Eliason & Jenkins (1981), motor skills can be broken down into two major categories known as *fine motor skills* and *gross motor skills*, where both of them are considered to be fundamental motor skills. Although for the present study we consider that distinction, it is important to mention that there is conflicting information in the literature regarding the classification of motor skills.

Burton & Miller (1998) consider *fundamental motor skills (FMS)* as a base to more advanced or specific movement skills. FMS assist children to keep dominion of their bodies, manage their daily activities, and develop more complex skills and other movements necessary to enroll in sports or other recreational activities (Goodway, 2003). The improvement of fundamental motor skills, primarily during the childhood, strongly contributes to the development of interest and success in sports and physical activities (Booth, et al., 1999).

Fine motor skills relate to small muscles recruitment and include skills that evolve, for example, finger dexterity, wrist flexibility, arm and hand steadiness, and finger speed. On the other hand, *gross motor skills* relate to large muscles recruitment and include skills that evolve, for example, static and dynamic balance, strength and agility, and general body coordination (Wang, 2004).

Since the 1920s, studies involving human motor behavior have shown considerable progress. However, the mechanism to acquire and improve a new skill is still not very well understood. It is known that within a few months after a child is born, his/her motor development starts and continues radically changing with the time, being able to be enhanced or modified through

education and training, especially during the early years. Although there is no commonly accepted theory to explain the exact course of development of motor skills, many models have been proposed to provide an explanation of motor development (Haubenstricker & Seefeldt, 1986). The simplest model was proposed by Seefeldt (1979). In his model, four levels of skills are organized in a pyramid style, and each level, from the base to the top, represents a progression of motor skills acquirement at a specific stage of life. A barrier is placed between the second and the third levels to indicate the importance of developing proficiency in the first two level skills, which are reflexes-reactions and fundamental motor skills, before developing more complex skills, which are represented by the third and fourth level skills (Haubenstricker & Seefeldt, 1986).

1.1 Motivations

Although fine and gross motor skills are learned during growth, these basic skills must also be taught and practiced over and over again to refine them (Haywood, 1993). Children who show a delay or lack of motor skills development tend to experience frustration and difficulty in acquiring more complex skills, reducing their enthusiasm for an active lifestyle (Booth, et al., 1999). The lack of participation in physical activities directly affects the improvement of childhood health status and quality of life, the advancement of adult health status and the probability of maintaining adequate activity into adulthood (Boreham & Riddoch, 2001).

Goodway et al. (2003) hypothesized that children who show developmental delay of fundamental motor skills can improve their motor proficiency if they follow adequate motor skills instructional programs. They found that children who received motor skills instructions benefited greatly showing improved abilities to demonstrate skills with proficiency and efficiency. Their hypothesis

was supported by other studies (Kelly, et al., 1989) (Zittel & McCUBBIN, 1996) that proved the positive effects of early motor skills programs on motor skills development for young children.

It was also suggested that different types of instructions yield positive impact on the development of fundamental motor skills. For example, Connor-Kuntz & Dummer (1996) studied the effects of incorporating language instruction to physical education lessons and found that "physical activity appears to be an effective environment in which to enhance the cognitive development of preschool children of all abilities". Moreover, Hamilton, Goodway, & Haubenstricker (1999) investigated the "effectiveness of parental involvement on the acquisition of object-control skills" and found that the experimental group achieved significant improvement, compared to the control group that did not get any instruction.

Considering the studies found in the literature suggest that motor skills instruction can positively affect the fundamental motor skills, while taking into account the model proposed by Seefeldt (1979) suggesting that the basics motor skills are determinant to the performance of more complex motor skills, it is clear that early diagnostic of poor motor skills proficiency in a child is very important. The earlier this deficiency is detected, the more likely it can be reversed, or at least improved upon using instructions and training programs performed by a qualified professional.

Virtual reality applications have been introduced into the fields of human kinetics and rehabilitation due to their capability of simulating real environments in an interactive, intuitive and safe manner. For instance, Wang, Mihailidis, Dutta, & Fernie (2011) have developed a feedback interface integrated to a simulated collision-avoidance power wheelchair for users with cognitive and complex physical impairments. The system helped the users to avoid hitting obstacles by

giving them auditory, visual and haptic feedback, which successfully contributed to the improvement of their navigability.

The literature illustrates diverse types of motor skills tests used to evaluate, identify developmental delays in motor skills acquisition and aid the process of improving the maturity of these skills. However, the literature review section details the shortcomings of administrating these classical tests, such as the dependence on human observation and high costs to administrate, among others. In addition, to address the need to improve the classical tests administration and reduce their limitations, there is also a need to develop a tool to be used to assess, develop and train gross motor skills, and to possibly diagnose gross motor skills deficit in children and adults, which would require less time to administrate, offer repeatability and as well, a better benefit-cost ratio to train professionals.

The first prototype of a computerized tool to implement the TGMD-2 was developed by Tremblay et al. (2013). Despite the fact that the pilot version implemented only four skills of the previous version of the TGMD-3, it was shown to be a promising tool "to provide researchers, educators and other experts a more effective and accurate mode to test gross motor skills in both children and adults" (Tremblay, et al., 2013).

1.2 Objectives

This work aims to develop a computerized application to implement the third version of the Test of Gross Motor Development (TGMD-3), developed by Ulrich (2013). The Microsoft Kinect will be used for this application due to its versatility and affordability. Despite the amount of research

conducted in human motion tracking area, until now there has been no published work using Microsoft Kinect sensor to implement the TGMD-3 test.

The Kinect-based application is intended to be interactive and provide assessment of all the thirteen skills assessed by the TGMD-3, described in Table 1. In addition, it has to collect the participants' information and provide them with instantaneous feedback after performing each skill.

The specific objectives of this work are to:

- Develop an integrated system composed of the Kinect sensor and a customized computational program to acquire, treat and analyze the collected data
- Develop a contact sensor to assist the data acquisition for the criteria that the Kinect sensor by itself cannot track
- Develop an algorithm for ball detection to track it during the ball-skills subtests
- Investigate the performance of the Kinect-based application for gross motor skills analysis

The following chapter reviews some key concepts from the literature regarding motor development, the types of classical tests used most often to assess gross motor skills, the limitations of the classical tests, and the advantages of using virtual reality and human-computer interaction applications for human motion tracking.

2 Literature Review

2.1 Understanding Motor Development

In the field of human kinetics, most people use the top-class athletic performance as a criterion for skilled behavior, or just see a skill in terms of its results, not paying attention to what it intrinsically involves. A more accurate understanding is provided a quote from Kelso (1982): “skill implies spatial precision: the appropriate sets of muscles for a given activity must be ordered correctly. Also, skill has a temporal or timing component; not only do muscles have to function in the proper sequence, but they have to operate at the right time”.

The terminology *motor,* when used by itself, indicates the mechanical and biological elements that effect movement. Usually, this terminology is used with another word, e.g. motor skill, motor development, motor behavior, motor learning, etc. (Thakur, et al., 2004).

According to (Rintala & Loovis, 2013) *motor development* is defined as “the sequential, continuous age-related process whereby movement behavior changes”. The changes in the movement behavior are usually classified as gross motor development and fine motor development. Humphrey (2003, 2004) defines motor development as “progressive change in motor performance”, considering that “development is sequential and cumulative” (Clark & Metcalfe, 2002).

The *term motor behavior* relates to the whole-body movements, including the eyes movements and the capability to control the head (Adolph, 1997). Also, Haywood (1993) argues that the motor behavior refers to the combination of motor development and motor learning; in other words, it is a result of the biological and environmental influence.

Motor learning refers to the acquisition of new skilled movements (Pavildes, et al., 1993). Moreover, "motor learning is an internal process or state that reflects a person's current capability for producing a particular movement" (Schmidt & Wrisberg, 2008). It can be measured observing the changes in the capability of producing a movement when submitted to practice or experiences, since the best way to measure motor learning is by observing a person's movement performance and pointing out the changes that might happen when additional practice is included (Schmidt & Wrisberg, 2008).

Finally, *a motor skill* can be described as "a skill for which the primary determinant of success is the quality of the movement that the performer produces" (Schmidt & Wrisberg, 2008). Piaget's sensorimotor period refers to the phase from birth to 2 years of age in a child's life that may be one of the most crucial phase of his/her brain development. From birth to 1-month-old, the child shows a set of reflexive and automatic actions, such as closing the hands, moving the eyes and sucking objects (Lefmann & Combs-Orme, 2013). Later on, the child progressively starts taking control of some muscles, performing specific movement patterns instead of random ones. Those movements are divided into two different groups: gross response and fine response. The fine response corresponds to movements performed with the fingers, hands or forearm, or those involved in hand co-ordination depending on small muscles. Conversely, gross responses refer to those movements of the trunk, the legs and the shoulders that depend on large muscular activity (Thakur, et al., 2004).

Fine response and gross response bring the concept of *gross motor skills* and *fine motor skills*. According to (Wang, 2004) "fine motor skills refer to such skills as finger dexterity, wrist flexibility, arm and hand steadiness, and finger speed. Gross motor skills refer to skills such as static and dynamic balance, strength and agility, and general body coordination". Harrow (Harrow,

1972) associates gross motor skills to locomotor movements, or locomotor skills, that change the child from one location to another, e.g. crawling, creeping, walking, running, leaping, jumping, hopping, skipping, galloping, rolling, and climbing. On the other hand, fine motor skills are associated to the non-locomotor movements, such as writing, drawing, object manipulation, etc. Fine motor coordination can be described as "skills of the hand that are needed to attain and manipulate objects" (Vries, et al., 2015).

According to Gallahue & Ozmun (1998), the growth in the ability to differentiate and integrate motor and sensory mechanisms evidences the neuromotor maturation. Since birth, a differentiation process is observed in a child's motor development, evidenced by a gradual progression from overall movement patterns to more controlled functional movements. The child progressively starts to distinguish different muscle groups and starts to establish more refined movements. Then, an integration process begins when the child starts to bring different muscle and sensory systems to interact in a coordinated manner with one another. The differentiation and integration processes are crucial to normal development.

2.2 Motor Skills Assessment Tools

The performance of a fundamental motor skill in early childhood can be investigated using a diverse range of assessment tools. Which assessment tool is used depends on the type of test, the context and the availability. Also, most of the tests are aimed at a specific group, so it is useful to focus on the skills that are most relevant to each group (Cools, et al., 2009). There are two major classifications for the testing instruments: *norm-referenced* and *criterion-referenced*. The norm-referenced scale is used to compare a standard norm, that was established previously, to a group or person. On the other hand, the criterion-referenced scale points out if a person matches the skill

level known to be acquired in sequence. The individual is compared to his/her own previous performance instead of to a group norm. The majority of the tests used to access a child's motor development are norm-referenced (Haywood, 1993).

Another approach classifies motor skill measurement tools as *process-oriented* or *product-oriented* assessment. Product-oriented assessments "are based on the time, distance, or number of successful attempts resulting from the performance of a skill". Basically, this approach is focused only on the final outcome of the performance. On the other hand, process-oriented assessments "are concerned with how the skill was performed, or the process responsible for the performance output" (Burton & Miller, 1998).

The most common assessment tools found in the literature are: Motoriktest für Vier- bis Sechjärige Kinder (MOT 4-6), Movement Assessment Battery for Children (Movement-ABC), Peabody Developmental Scales (PDMS), Test of Gross Motor Development (TGMD), the Maastrichtse Motoriek Test (MMT) and the Bruininks-Oseretsky test of Motor Proficiency (BOTMP) (Cools, et al., 2009).

2.2.1 Motoriktest für Vier- bis Sechjärige Kinder (MOT 4-6)

The norm-referenced test MOT 4-6 is a tool specifically developed for pre-school children that aims to identify motor disorders and to assess motor development. Due to its very specific age range, the MOT 4-6 test is recommended for research purposes. The test was created in Germany and there is no translation of its manual in English, so it is not widely used and is more common in German speaking counties (Kambas, et al., 2012).

The MOT 4-6 consists of performing, in approximately 20 minutes, eighteen items: an introductive forward jump in a hoop, walking forward, making dots on a sheet, grasping a tissue with toes, jumping sideways, catching a dropped stick, carrying balls from box to box, walking in backward direction, throwing a ball to a target, collecting matches, passing through a hoop, jumping in a hoop on one foot standing on one leg, catching a ring, jumping jacks, jumping over a cord, rolling around the length axe of the body, standing up holding a ball on the head, jumping and turning in a hoop (Zimmer & Volkamer, 1987).

According to Kambas et al. (2012) there are diverse ways to score the performance in each task of the MOT 4-6 test. Variables that involve for example, the number of jumps, total time to perform a task, etc., are called "raw scores" and are noted and posteriorly converted into a ranking scale of three-levels. Therefore, the possible scores for each task range from '0', which means that the skill is not mastered, to '2', which means the skill is mastered. Then, the final score for the test consists of the sum of all seventeen tasks scores, ranging from '0' to '34'.

The MOT 4–6's psychometric characteristics were investigated by Kambas et al. (2012). The test showed high values for test–retest reliability, which means the test results can be considered stable and consistent.

2.2.2 Movement Assessment Battery for Children (Movement-ABC)

The Movement Assessment Battery for Children (Henderson & Barnett, 1992) is a product-oriented test that is one of the most often employed methods to assess child's movement and has evolved for identifying children with slight motor impairment (Croce, et al., 2001).

The aim of the test is to "identify and describe children with movement difficulties", since the test is focused on detection of delay or deficiency (Henderson & Barnett, 1992) (Cools, et al., 2009). The test was created for child between 4 and 12 years old and comprises 32 items, divided into four age bands. Each of them contains of eight items that assess the movements towards three categories: manual dexterity skills, ball skills and balance skills. Also, each item is scored from ‘0’ to ‘5’ points, being the rate ‘0’ the best performance. The whole test can be done in approximately 30 minutes and the final score is the sum of each individual score in each item (Cools, et al., 2009).

Croce et al. (2001) studied the psychometric characteristics of the Movement Assessment Battery for Children test and concluded that the test-retest reliability of the Movement-ABC, estimated using intraclass correlation coefficients, was high across all age groups. The results indicate the usability of the test to assess motor skills of children from 5 to 12 years old, which means the test results can be considered stable and consistent.

2.2.3 Peabody Developmental Scales – Second Edition (PDMS-2)

The Peabody Developmental Motor Scales-2 (Folio & Fewell, 1983) is a norm-referenced test that assesses both gross and fine motor skills. The test contains two different scales, one for each type of fundamental motor skill, i.e. gross or fine. Both scales aim to assess if a 6-year-old child or younger has some delay in his/her motor development (Van Waelvelde, et al., 2007).

The PDMS-2 classifies the motor development and FMS of a child from when he/she is born until the age of six. The tasks are divided into six skill categories; four of them are related to gross motor skills (reflexes, static balance, locomotion and ball skills) and two of them are related to fine motor

skills (grasping and visual-motor integration). The whole test takes around one hour to be completed since each category has a number of items to assess. The reflexes category is evaluated only for children under one year of age; the static balance category contains 30 items, the locomotion 89 items, the ball skills 24 items, the grasping 26 items and the visual-integration 72 items. There is a specified performance criterion for each item and the child's performance is scored from '0' to '2' (Van Waelvelde, et al., 2007).

The total motor skill is given by the sum of the six categories, where '0' means a non-acquired skill, '1' means a developing skill and '2' means an attained skill. This way of assessment permits the professional to follow the child's progress (Cools, et al., 2009).

The PDMS-2 psychometric characteristics were investigated and the test demonstrates good "internal consistency, test-retest reliability, and inter-rater coefficients" (Tremblay, et al., 2013).

2.2.4 Maastrichtse Motoriek Test (MMT)

The MMT (Vles, et al., 2004) test assesses, both quantitatively and qualitatively, the motor skills of 5 to 6-year-old children. It measures the FMS taking into account four categories: static balance (14 items), dynamic balance (20 items), ball skills (8 items) and diadochokinesis and manual dexterity (28 items). Diadochokinesis refers to the ability to perform antagonistic movements in quick succession. From a total of 70 items, 34 consist of quantitative aspects and 36 of qualitative aspects (Wassenberg, et al., 2005). The total amount of time to finish the test is approximately 25 minutes and "the scores of the four areas are combined in a total score, a quality score and a quantity score" (Wassenberg, et al., 2005). The psychometric characteristics for this test were investigated and the test was shown to be stable and consistent.

2.2.5 Bruininks-Oseretsky test of Motor Proficiency (BOTMP/BOT-2)

BOTMP is an abbreviation for Bruininks-Oseretsky Test of Motor Proficiency and BOT-2 refers to its revised second edition (South & Palilla, 2013). The test assesses gross and fine motor proficiency for people from the ages of 4 to 21 years old. The complete version of the test comprises 53 items divided into 8 subtests: the fine motor precision subtest consists of 7 items; the fine motor integration subtest consists of 8 items; the manual dexterity subtest consists of 5 items; the bilateral coordination subtest consists of 7 items; the balance subtest consists of 9 items; the running speed and agility subtest consists of 5 items; the upper limb coordination subtest consists of 7 items and the strength subtest consists of 5 items. The difficulty of each item increases progressively in every subtest (Cools, et al., 2009). Later on, those 8 subtests are categorized into: strength and agility, manual coordination, body coordination, and fine manual control. The whole test takes around one hour to complete (South & Palilla, 2013).

Moreover, the score of the test changes depending on the individual item. The minimum score is 2-point scale and the maximum score is 13-point scale, where the raw scores can be transformed into a standard numerical score. The total motor composite consists of the sum of the scores of each test. The outcome of the scores can be broken down into a fine manual control composite, a manual coordination composite, a body coordination composite and a strength and agility composite (Cools, et al., 2009).

The test psychometric characteristics were investigated and it “proved test validity for BOT-2 for individuals with developmental coordination disorder (DCD), mild to moderate mental retardation (MR), and high-functioning autism /Asperger’s Disorder” (Cools, et al., 2009).

2.2.6 Test of Gross Motor Development-3 (TGMD-3)

The purpose of the TGMD (Ulrich, 2000) is to measure the gross motor skills of children from 3 to 10 years of age, based on qualitative aspects of their performance. According to Ulrich D. A. (2000) and Burton & Miller (1998) the test can be described as process- and product-oriented, as well as a norm- and criterion-referenced assessment tool. It is used to identify whether a child's motor skill development is significantly behind his/her peer's development. In addition, the test is useful to plan an instructional program to assist in motor skill development, to assess the child's progress, to evaluate the success of the motor skill program, and to serve as an instrument of measurement for research involving gross motor development.

The third edition of the TGMD was released in 2015. In the present study we implement this version. The original version, the TGMD, was released in 1985 (Cools, et al., 2009). Based on feedback received on the TGMD-2, released in 2000, two skills were eliminated and three others were added to the new TGMD-3, which are the one hand side arm strike, underhand throw and skip (Ulrich, 2013).

The test is divided into the locomotor subtest and ball skills subtest. The locomotor subtest includes run, gallop, hop, skip, horizontal jump and slide. The ball skills subtest includes two-hand strike of a stationary ball, one-hand forehand strike of self-bounced ball, one-hand stationary dribble, two-hand catch, kick a stationary ball, overhand throw and underhand throw (Ulrich, 2013).The 13 items take around 20 minutes to assess, and each of them has a number of performance criteria to be demonstrated and used to assess if the skill performance is considered adequate or poor (Johnson, et al., 2016). The child has to perform each item twice and every attempt receives a '0' score if performed incorrectly and '1' score if performed correctly. A raw score for each item is

obtained by the sum of the two trials. The raw scores are then translated to standard scores considering the age of the child (Cools, et al., 2009).

The examiner follows the instructions provided on the TDMG-3 performance criteria sheet, shown in Table 1, which presents the performance criteria for each skill, used to obtain an evaluation of the quality and maturity of the child's movement (Ulrich, 2013).

Table 1. TGMD-3 Form for Collecting Norms. Adapted from Ulrich D. (2013).

Skill	Performance Criteria
	Locomotor Subtest
1. Run	1. Arms move in opposition to legs with elbows bent. 2. Brief period where both feet are off the surface. 3. Narrow foot placement landing on heel or toes (not flat-footed). 4. Non-support leg bent about 90 degrees so foot is close to their buttocks.
2. Gallop	1. Arms flex and swing forward to produce force. 2. A step forward with lead foot followed with the trailing foot landing beside or a little behind the lead foot (not in front of the lead foot). 3. Brief period where both feet come off the surface. 4. Maintains a rhythmic pattern for 4 consecutive gallops.
3. Hop	1. Non-hopping leg swings forward in pendular fashion to produce force. 2. Foot of non-hopping leg remains behind hopping leg (does not cross in front of). 3. Arms flex and swing forward to produce force. 4. Hops 4 consecutive times on the preferred foot before stopping.
4. Skip	1. A step forward followed by a hop on the same foot. 2. Arms are flexed and move in opposition to legs to produce force. 3. Completes 4 continuous rhythmical alternating skips.
5. Horizontal Jump	1. Prior to take off both knees are flexed and arms are extended behind the back. 2. Arms extend forcefully forward and upward reaching above the head. 3. Both feet come off the floor together and land together. 4. Both arms are forced downward during landing.
6. Slide	1. Body is turned sideways so shoulders remain aligned with the line on the floor. 2. A step sideways with the lead foot followed by a slide with the trailing foot where both feet come off the surface briefly. 3. 4 continuous slides to the preferred side. 4. 4 continuous slides to the non-preferred side.
	Ball Skills Subtest
Skill	Performance Criteria
7. Two-hand strike of a stationary ball	1. Child's preferred hand grips bat above non-preferred hand. 2. Child's non-preferred hip/shoulder faces straight ahead. 3. Hip and shoulder rotate and derotate during swing. 4. Steps toward ball with non-preferred foot. 5. Hits ball sending it straight ahead
8. One- hand forehand strike of self- bounced ball	1. Child takes a backswing with the paddle when the ball is bounced. 2. Steps toward the ball with non-preferred foot. 3. Strikes the ball toward the wall. 4. Paddle follows through toward non- preferred shoulder.
9. One-hand stationary dribble	1. Contacts ball with one hand at about waist level. 2. Pushes the ball with fingertips (not slapping at ball). 3. Maintains control of the ball for 4 bounces without moving their feet to retrieve the ball.
10. Two-hand catch	1. Child's hands are positioned in front of the body with the elbows flexed. 2. Arms extend reaching for the ball as it arrives. 3. Ball is caught by hands only.
11. Kick a stationary ball	1. Rapid, continuous approach to the ball. 2. Child takes an elongated stride or leap just prior to ball contact. 3. Non-kicking foot placed close to the ball. 4. Kicks ball with instep or inside of preferred foot (not the toes).
12. Overhand throw	1. Windup is initiated with a downward movement of hand and arm. 2. Rotates hip and shoulder to a point where the non-throwing side faces the wall. 3. Steps with the foot opposite the throwing hand toward the wall. 4. Throwing hand follows through after the ball. Release across the body toward the hip of the non-throwing side.
13. Underhand throw	1. Preferred hand swings down and back reaching behind the trunk. 2. Steps forward with the foot opposite the throwing hand. 3. Ball is tossed forward hitting the wall without a bounce. 4. Hand follows through after ball release to chest level.

Webster & Ulrich (2017) recently published a study investigating the psychometric properties of the TGMD-3, which found that the test "exhibit(ed) high levels of validity and reliability, providing confidence for the usage and collection of new norms". Moderate to large correlations with age were found during the reliability test; the ball skills subtest showed a higher correlation and the locomotor skills subtest showed the lower correlation. The investigation of the test's psychometric properties showed that the test results can be considered stable and consistent.

2.3 Classical tests limitations

Regardless of the type of tool used to assess the motor skills, all of them involve measurement, i.e. data collection, and posterior evaluation of the information collected. Although classical tests are reliable and useful for motor skills assessment, they have some limitations. For example, depending on the test, the examiner has to meet some degree of qualification to be able to conduct it. There are three levels of qualification: level 'A' does not require any special qualification; level 'B' requires some expertise about the test; and level 'C' requires a high level of expertise. Depending on the test and its qualification requirements, the cost to administer it can be very high, especially considering that most of these tests require more than one professional and training to conduct them is necessary (Tremblay, et al., 2013).

Another limitation of the classical tests is related to human observation. The results are obtained by professionals who have to observe the performance of the movement to score it. Sometimes the judgment has to be done quickly and some movements are performed very fast, which can lead to misconceptions and mistakes. Also, each examiner potentially has a subjective judgment for each task, what can result in a conflict when the final scores are combined. To overcome this problem, some researchers choose to videotape the child's test (Bonifacci, 2004) to double check the results,

which can be very impractical and add a substantial amount of time to perform the test (Tremblay, et al., 2013).

Moreover, according to Mousavi Hondori & Khademi (2014), the majority of the assessments used in physical therapy and rehabilitation rely on human judgment and observation, which directly depend on the visual feedback from the therapist that watches the patient performing the movement. Mousavi Hondori & Khademi (2014) also suggest the use of sensors and computing techniques for motion capture, considering that technology has advanced quickly and has become more cost-effective.

2.4 Alternatives for the classical tests

The use of Virtual Reality (VR) and motion-based games for rehabilitation is gaining ground (Chang, et al., 2011). The user of a VR tool is able to interact and explore a computer-made environment in real-time (Sveistrup, 2004). The main characteristic of a VR application is the user's interaction with the environment (Sveistrup, 2004). The advantages of using VR applications for rehabilitation and human movement assessment are: repetition, feedback and motivation. Repetition is fundamental for motor learning. By itself, repetition does not lead to motor learning, and instead must be linked to the feedback regarding the performance in order to reach a goal for a particular task. Also, the user must be kept motivated, since practicing a movement over and over again can get boring (Holden, 2005).

According to Burdea (2003), an application based on VR has numerous advantages such as: the immediate feedback and accurate measurement of movements; the instantaneous data storage of the movement into a computer to be subsequently processed and analyzed; the possibility of

assessment of more complex and more assorted types of motor abilities in a very practical way and the possibility of creating a database to evaluate fundamental motor skills. Virtual Reality also permits the creation of a virtual teacher to perform a task repeatedly to provide an enhancement of "learning by imitation". Moreover, according to Holden (2005), virtual reality makes it possible to offer instantaneous feedback to the user in a friendly interface which is easy to interpret. In this way, the participant is able to see their own performance from the same point of view as that of an external observer.

2.5 Human motion tracking

Human motion tracking systems are essential to human kinetics assessment and have to dynamically extract the main characteristics of a performed movement while storing the real-time data for further analysis. The recorded data is expected to faithfully reproduce the whole-body movement variation with time (Zhou & Hu, 2008).

Motion capture consists of recording the movement of a target in real-time as a set of Cartesian coordinates in 3D space. Usually, this motion capture is performed by optical systems known as Optical Motion Capture (OMC). These systems use cameras to reconstruct body movements performed by a subject with a very high degree of accuracy. OMCs can be markerless or marker-based systems. The most common marker-based OMCs used to track human movement are the Vicon system and the Optotrak system. Both of them consist of employing “a set of multiple synchronized cameras to capture markers placed in strategic locations on the body” (Guerra-Filho, 2005). The marker-based systems are usually very accurate, but they can be very expensive and require a relatively high level of expertise to operate. On the other hand, markerless systems do not require the placement of markers on the body, and thus are a cheap and convenient alternative.

Their principle of operation is based on extracting silhouettes or other features, e.g. edges from the subject's body (Guerra-Filho, 2005). The Microsoft Kinect sensor is one of the most famous markerless system used in human motion tracking.

2.6 Using Microsoft Kinect as a tool to capture movements

The launch of Microsoft Kinect on November 4th of 2010 marked a revolutionary advancement to developers especially thanks to its depth camera and its affordable price (Seguin, et al., 2012). The proof of its success is found in the enormous amount of papers citing its use that have been published each year since the Kinect was released. A simple search on Google scholar shows that from 2010 to the present time 70,800 papers have mentioned the word 'Kinect'. Using the Kinect for rehabilitation has also been intensely studied and a search for the term 'Kinect + Rehabilitation' during the same time period reveals 8,460 results. However, the literature does not show many sources for the use of the Kinect to assess motor skills. A search for this term showed only 133 investigations, many of which are irrelevant to this topic.

The authors of (Galna, et al., 2014) studied the accuracy of the Microsoft Kinect sensor for measuring movements in people with Parkinson's disease. They chose nine people diagnosed with the disease and ten people for the control group to perform a series of movements. They measured the movements with a Vicon 3-D motion analysis system, considered a gold-standard reference, and the Kinect. The movements performed included gross motor skills, such as standing, reaching and walking; and fine motor skills, such as hand clasping, finger tapping and hand pronation. The measurements included the time length to perform the task and its range of motion. They concluded that the Kinect is very accurate in measuring the timing of movements, is excellent for measuring

gross movements and is very poor for measuring fine motor movements when compared to the Vicon system.

Another study was performed by Dutta (2012), who investigated the reliability of the Kinect to be used as a portable 3-D motion capture system in terms of range, field of view and accuracy. Compared to a Vicon system, the results of the study showed that the Kinect has a root mean-squared errors of 0.0065 m, 0.0109 m, 0.0057 m in the x, y and z directions, respectively. The author concluded that it is possible to assess kinematic measurement in the workplace using the Kinect.

Clark, et al. (2012) compared the validity of the Kinect anatomical landmarks to those of the Vicon system during three commonly performed postural control tests: single leg standing balance, forward reach and lateral reach. The authors concluded that the Kinect can be successfully used in providing anatomical landmark displacement and trunk angle data. For them the major benefits are the cost, the portability and the availability, if compared to a 3D motion analysis system. In addition, another important advantage of the Kinect is that it is markerless (Clark, et al., 2012).

González-Ortega, et al. (2014) used the Kinect as a 3D computer vision system in order to perform a rehabilitation and cognitive assessment of body scheme dysfunctions and left-right confusion. They presented the Kinect as a markerless, portable and inexpensive system. They concluded that the Kinect device is an affordable and very useful sensor to develop applications in many fields such as rehabilitation. The study assessed fifteen users and achieved a successful monitoring percentage of 96.28%.

3 System Description

3.1 Proposed System

The Kinect-based application consists of software that interacts with the user to assess his/her gross motor skills based on the performance criteria defined on the TGMD-3 performance criteria sheet (Table 1). The first thing to do when starting a new test with a new user is to create a registration for the user. The registration form to be filled out contains the participant's information, such as name, address, age, gender, socioeconomic status, race, weight status, date of the test, country, preferred-hand, among other information.

It is important to keep the participant's registration record for future research, or in case the participant wants to keep track of his/her improvements in a specific skill.

After filling out all the information required and pressing the save button, the participant is ready to start a new test. After the s/he selects the type of test for the assessment (Figure 1), the respective test page pops up (Figure 2).

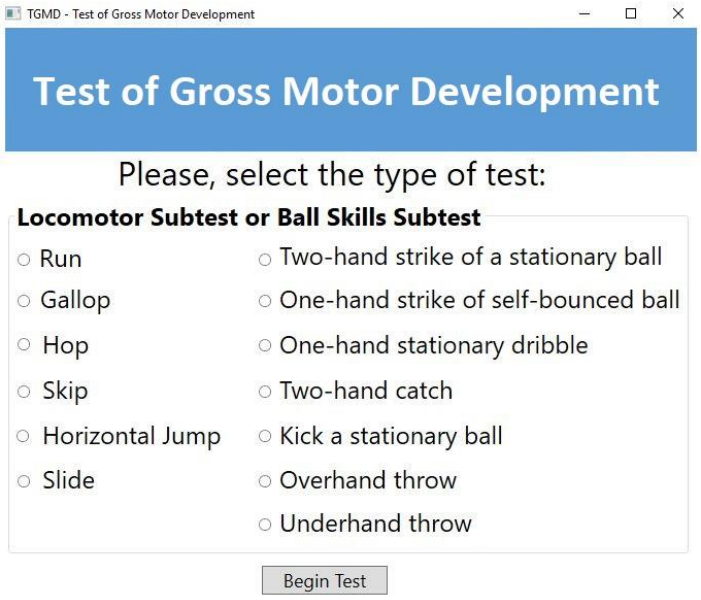

Figure 1. The test selection screen contains all the thirteen movements present on the TGMD-3 test. The user of the computer-based application is expected to select the test that s/he wants to perform.

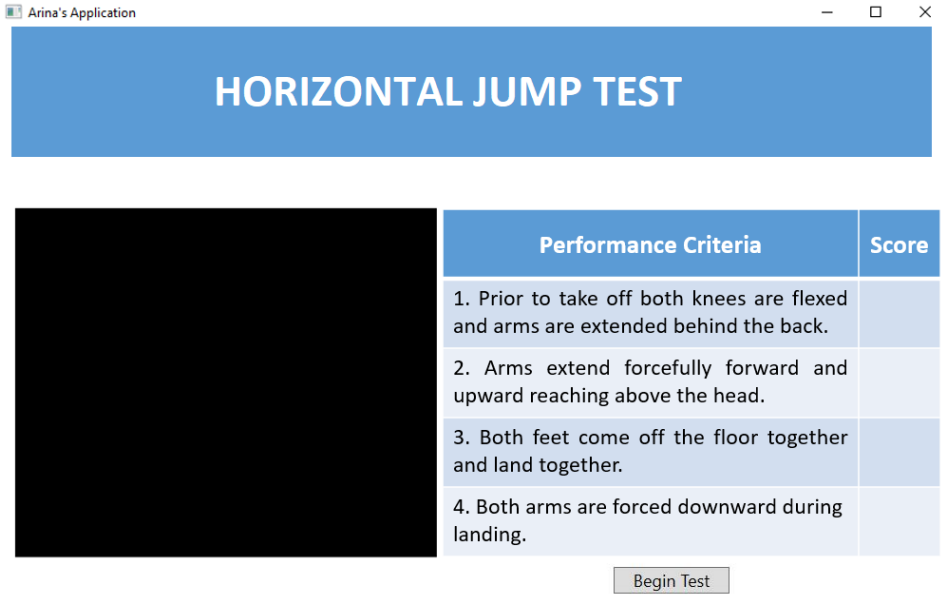

Figure 2. Example of a test page. After the participant selects the test that s/he wants to perform, the respective test page pop-up. It contains the expected performance criteria expected for the chosen test and a column that will show the score for each criterion. The black canvas on the left represents where the participant's skeleton will be drawn while the test is performed.

The testing page contains a start button, a table describing the performance criteria for that test and a column used to display the achieved score for each criterion. When the 'Begin Test' button is clicked, if a person is detected to be standing in front of the Kinect sensor, the test begins and their skeleton is drawn on the screen during the whole movement. A quick interaction is given to inform the user to 'prepare', to 'go' and 'end of recording'. After instructing the user to begin the movement, the software starts recording the position of twenty-five joints, storing all the coordinates during the whole movement. When the movement is finished, the algorithms execute the data analysis and then display the feedback for each performance criteria in the table. The computerized test marks '1' if the user passes that criterion and '0' if he/she fails (Figure 3).

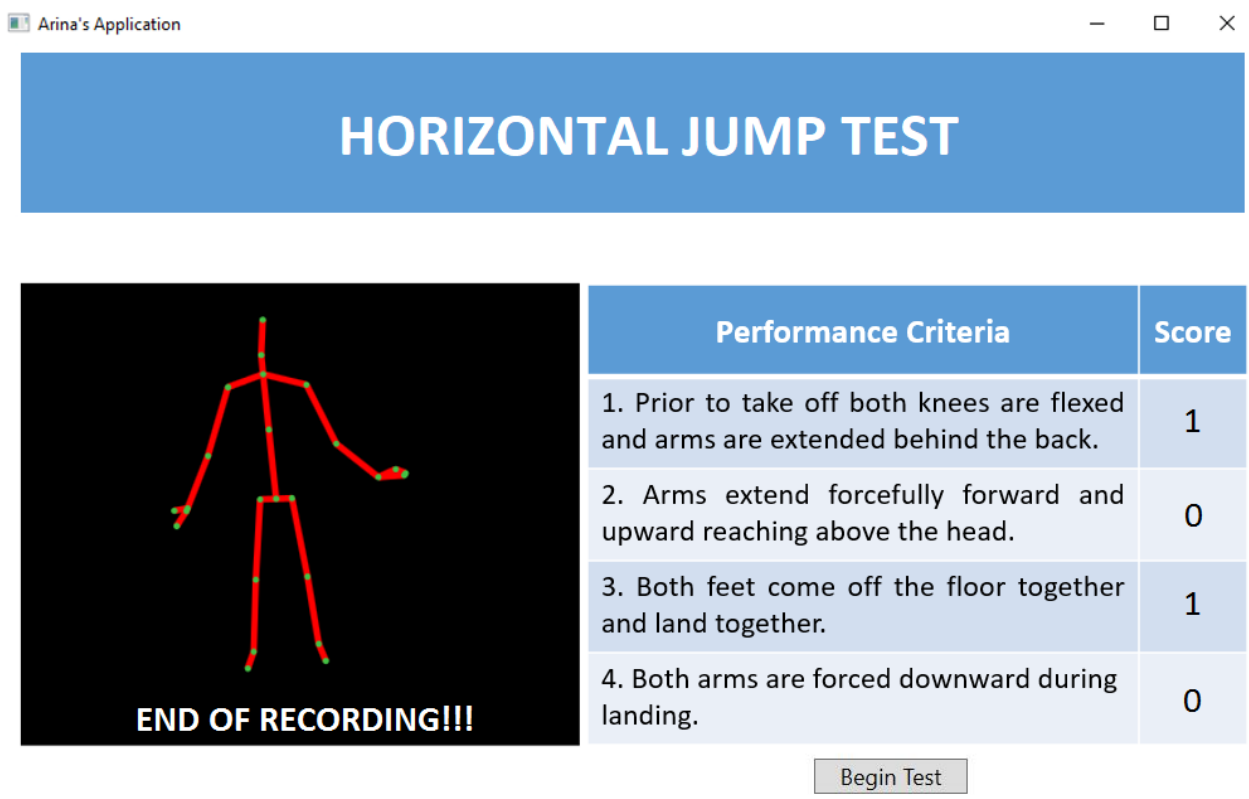

Figure 3. Example of feedback. After the participant performs the test, a score for each performance criterion will be display in the column 'score'. If the participant performs correctly the criterion, s/he receives a score '1', otherwise '0'. When the test is completed, it displays 'End of Recording' on the canvas where the skeleton is drawn.

3.2 Microsoft Kinect Sensor

Kinect's first version was known by the code name ‘Project Natal’ and it was created for motion capturing for the Xbox 360 gaming console. What stands out most about the Kinect is the absence of a hand-controlled device; instead it detects a person’s body position, motion and voice (Jana, 2012). Kinect literally uses people as the replacement for the controller as it provides a Natural User Interface (NUI). "A natural user interface is a system for human-computer interaction that the user operates through intuitive actions related to natural, everyday human behavior. Microsoft’s Kinect, for example, is a motion sensor for the Xbox 360 gaming console that allows users to interact through body motions, gestures and spoken commands. Kinect recognizes individual players’ bodies and voices" (Rouse, 2011).

Kinect sensor has been widely used in many applications since its first release. The current version, Kinect v2 launched in 2015, has an improved hardware and a better performance compared to the previous versions.

The next paragraphs are dedicated to describing the device and its components. As shown in Figure 4, Kinect sensor includes some key components: a color camera, a depth sensor (infrared camera + infrared emitters), a microphone array and a power light.

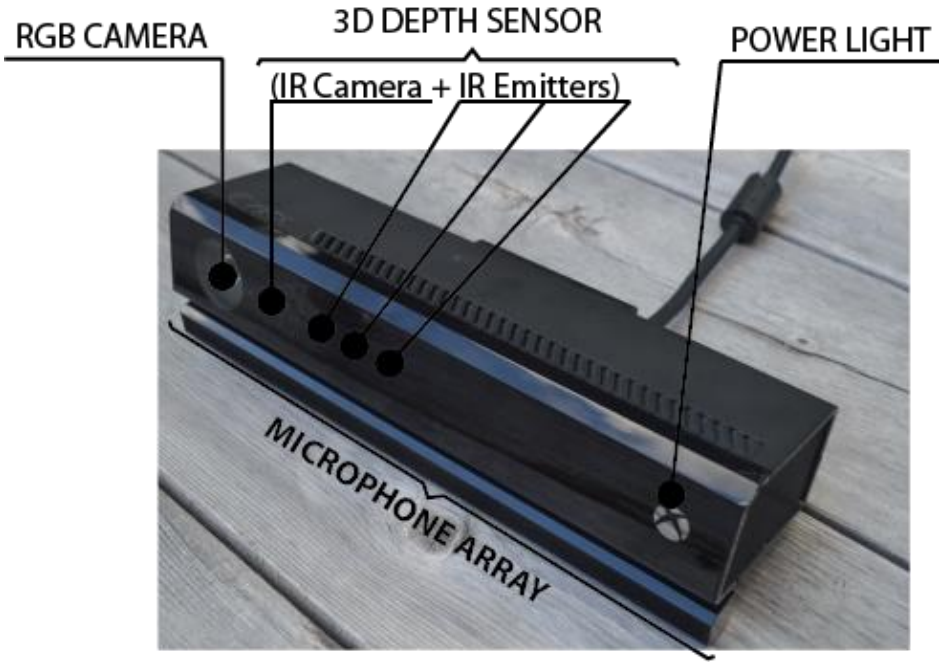

Figure 4. Microsoft Kinect sensor and its main components: RGB camera, IR camera, IR emitters, power light and microphone array.

3.2.1 Color Camera

The color camera is responsible for capturing and streaming the color information. It is also called a RGB camera since it detects the red, green and blue colors from the objects in front of it. The camera has resolution of 1920 x 1080 pixels at a speed of 30 frames per second (FPS). Its range of view is 70 degrees horizontal by 60 degrees vertical (Microsoft, 2015).

3.2.2 Depth Sensor

The depth camera collects data between the camera and the object, which is the principle of 3D measurement. Raw 3D information is very useful for model construction, object tracking, movement detection and other applications (Yang, et al., 2015). The depth data stream supports a resolution of 512 x 424 pixels at a speed of 30 FPS and the sensor's field of view is the same as the color camera, i.e. 70º x 60º. The range of the depth sensor lies between 0.5 meters and 4.5

meters (Microsoft, 2015). Basically, the depth sensor tells how far away things are from it. Figure 5 (a) shows an example of a stream captured by a depth camera, where brighter colors are closer and darker colors are further away from the camera. Each pixel in Figure 5 (a) is represented by a shade of grey that varies from 0 to 255. The closest the pixel value is to 0, the darker and consequently the further from the sensor. Figure 5 (b) shows the 3D plot of that depth image. Note that as the person holds the ball in front of his/her body, the ball is closer to the sensor than the body, so in Figure 5 (a) the ball appears brighter and in Figure 5 (b) the ball appears 'coming out' of the paper plane. This means the ball's pixels in Figure 5 (a) are closer to 255, which is the maximum value of the depth scale.

(a) (b)

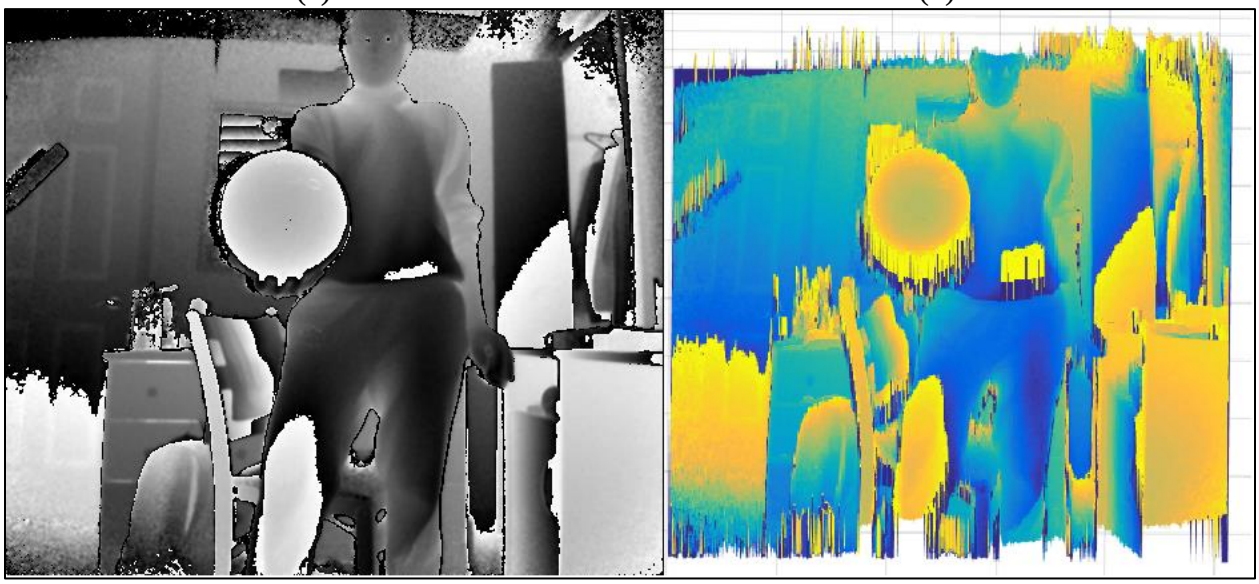

Figure 5. (a) Depth image captured by Kinect's depth camera. (b) 3D plot of the depth image captured by Kinect's depth camera.

The principle of Kinect's depth camera is based on the Time-of-Flight method, which uses active sensors to calculate the distance from a target by measuring the round-trip elapsed time of a light ray (Yang, et al., 2015). Time-of-flight (TOF) systems are optical systems able to rebuild a 3D scenery by calculating the period of time that a light ray takes to travel from its emission source to

a reflector target and then back to the collector. “TOF cameras are particular TOF systems able to acquire a map of distances through the diffusion of the light on the whole scene, collimating the returning light on a camera matrix sensor and measuring for each single pixel the phase shift between sending and returning light” (Corti, et al., 2016).

The main components of a TOF camera are an infrared (IR) emitter, a matrix of IR sensors and an electronic circuit that receives the reflected signal and then performs the calculations to estimate the round-trip distance (Corti, et al., 2016).

3.2.3 Microphone Array

The Kinect’s microphone array consists of 4 channels, placed in a linear order, of 16-bit audio at a sampling rate of 16 kHz (Mousavi Hondori & Khademi, 2014). The order that the channels are placed in is ideal for the detection of the direction of the audio wave. Also, the array arrangement contributes to a more effective voice recognition, better noise suppression and echo cancelation (Jana, 2012).

3.2.4 Developing Applications Using the Kinect

In order for a developer to interact with the Kinect sensor and its components it is necessary to use a Software Development Kit, or simply SDK. On June 16th of 2011, Microsoft released the first version of the Kinect for Windows SDK to allow developers to write Kinect app in C++, C# or Visual Basic. The current version, the Kinect for Windows SDK 2.0, includes the drivers for using Kinect sensor v2 on a computer; application programing interfaces (APIs) and device interfaces; and some code samples (Microsoft, 2014).

Connecting the Kinect to a PC requires an adapter to supply additional power, since the USB port of the PC does not deliver enough power for the Kinect to function. Also, the adapter contains a USB outlet to be connected to the computer's USB port.

Besides the Kinect for Windows SDK, two pieces of software are also necessary to be able to create an application using the Kinect sensor: the Microsoft Visual Studio 2010 Express or higher editions and a Microsoft .NET Framework 4.0 or higher. It is important to mention that the Kinect for Windows SDK runs only on the Windows operation system (Windows 7, 8 or 10) and it also requires a minimum hardware configuration (32- (x86) or 64-bit (x64) processor; a dual core 2.66 GHz or faster processor; a dedicated USB 3.0 bus; and at least 2 GB RAM) (Jana, 2012).

Once the installation of those software is complete, the user is ready to plug the Kinect to the computer. The first time it is plugged in, the drivers will be installed automatically. After that, the user is able to start developing applications using Kinect's sensor features.

The Kinect SDK provides managed (C# or VB.NET) and unmanaged (C++) libraries which interact with the cameras (RGB and IR) and the microphone array. In this way, the Kinect SDK offers several possibilities for developers to create applications to be used in diverse fields such as healthcare, robotics, education, security, VR, coaching, military, etc. The following list gives an example of operations performed using Kinect for Windows SDK (Jana, 2012):

- Capturing and processing the color image data stream
- Processing the depth image data stream
- Capturing the infrared stream
- Tracking human skeleton and joint movements
- Human gesture recognition

- Capturing the audio stream
- Enabling speech recognition
- Getting data from the accelerometer
- Controlling the infrared emitter

3.2.5 Skeletal Joints

Kinect for Windows SDK offers the possibility of tracking body information about the users standing in front of it. It can track up to 25 joints each of up to 6 players at the same time. Figure 6 represents those 25 joints (Microsoft, 2016).

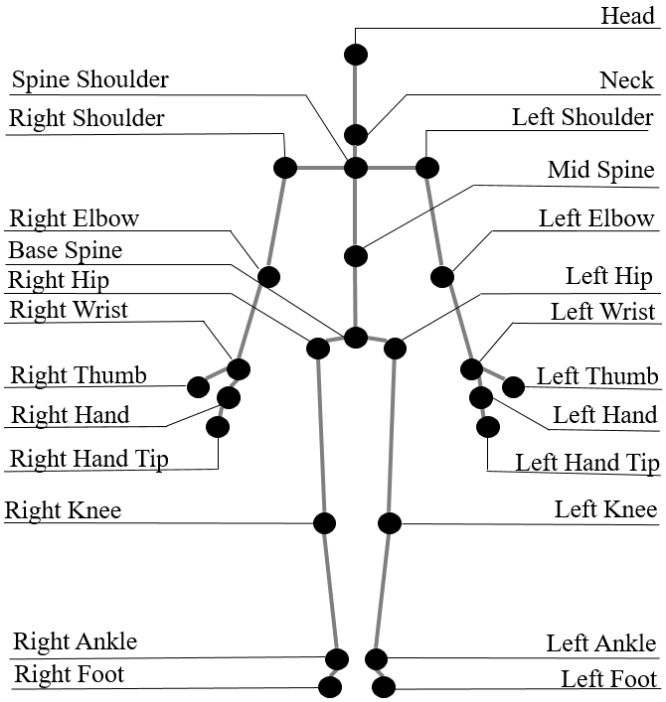

Figure 6. Scheme that represents the twenty-five joints tracked by the 'BodyFrame' class. The class represents in a frame all the computed real-time tracking information about the person standing in front of the sensor.

Thanks to the depth camera, the Kinect sensor can return the distance between the sensor and each joint in a 3D measurement. In other words, it is possible to track the X, Y and Z coordinate of each joint in real time and draw the whole movement as it varies along time. Moreover, Kinect operates on a frequency of 30 Hz, which means that the sensor captures 30 frames per second.

3.3 Contact Sensor

Some of the TGMD-3 criteria consist of analyzing how the child hits the ball (e.g. slaps at the ball or pushes it using fingertips) or how the child lands on the surface (e.g. landing on heels or toes, not flat-footed). It is clear that by only using the body-tracking feature, it is not possible to evaluate these criteria, since they need a better accuracy than the feature can provide. Therefore, a contact sensor was developed to be used in association with the Kinect to be able to better assess all the performance criteria.

The contact circuit was built using an Arduino Uno, one pushbutton and one Bluetooth module HC-05 to transmit the data wirelessly (Figure 7). A code snippet was created on the Arduino environment to program the Bluetooth to send logic level ‘0’ when the pushbutton is not pressed and ‘1’ when it is pressed. The device was then attached to a glove or a shoe. Padding was provided to avoid breaking of contact sensor depending on the skill to be tested. The transmission via Bluetooth informs the main software when contact occurs.

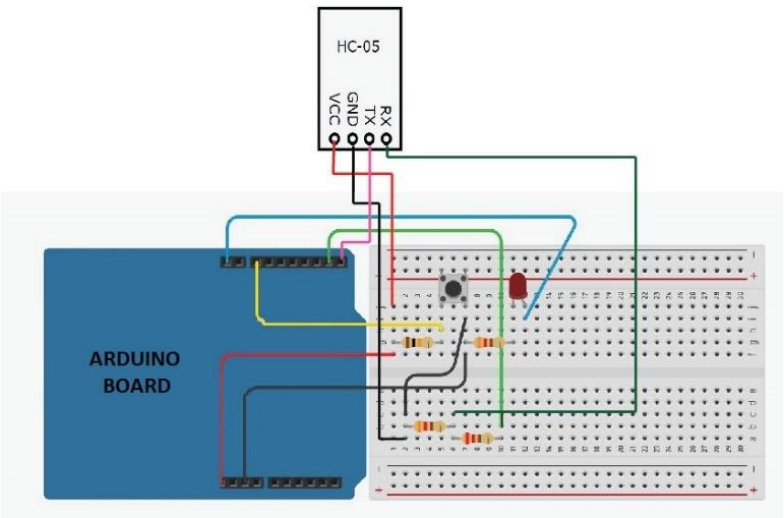

Figure 7. Electronic circuit for contact sensor. It consists of an Arduino chip, resistors, a LED, a push-button and a Bluetooth module.

3.4 Ball Detection

In order to assess the ball skill subtests of the TGMD-3, it was necessary to develop a ball detection algorithm based on its color information. The algorithm detects the ball by its color using the Kinect's RGB camera and EmguCV, a cross platform .Net wrapper to the OpenCV image processing libraries (EmguCV, 2008). OpenCV brings a set of useful optimized algorithms that can be used for computer vision techniques, such as object recognition (OpenCV, 2012). The library has more than 2500 optimized algorithms. These algorithms are used in diverse applications such as face recognition, object identification, human motion tracking, camera tracking, moving object tracking, 3D shape extraction from objects, image matching, pattern recognition, reality enhancement, etc. (OpenCV, 2012).

Each RGB image frame captured by Kinect's color camera was converted to HSV space and then segmented into its three channels: hue, saturation and value. Working on the HSV space provided better results, since it offers more robustness to lighting changes (Chipantasi, et al., 2017).

With the HSV values separated into three single channels, thresholds were then applied to filter the image according to the color of the ball, resulting in a processed black and white image that contains only the ball (Figure 8). After detecting the ball on the frame, the color frame of the original picture was then mapped to the depth frame to also acquire the coordinates of the ball. Having the depth information and the position of each pixel that comprises the ball, it was possible to track the 3D position of ball in real time.

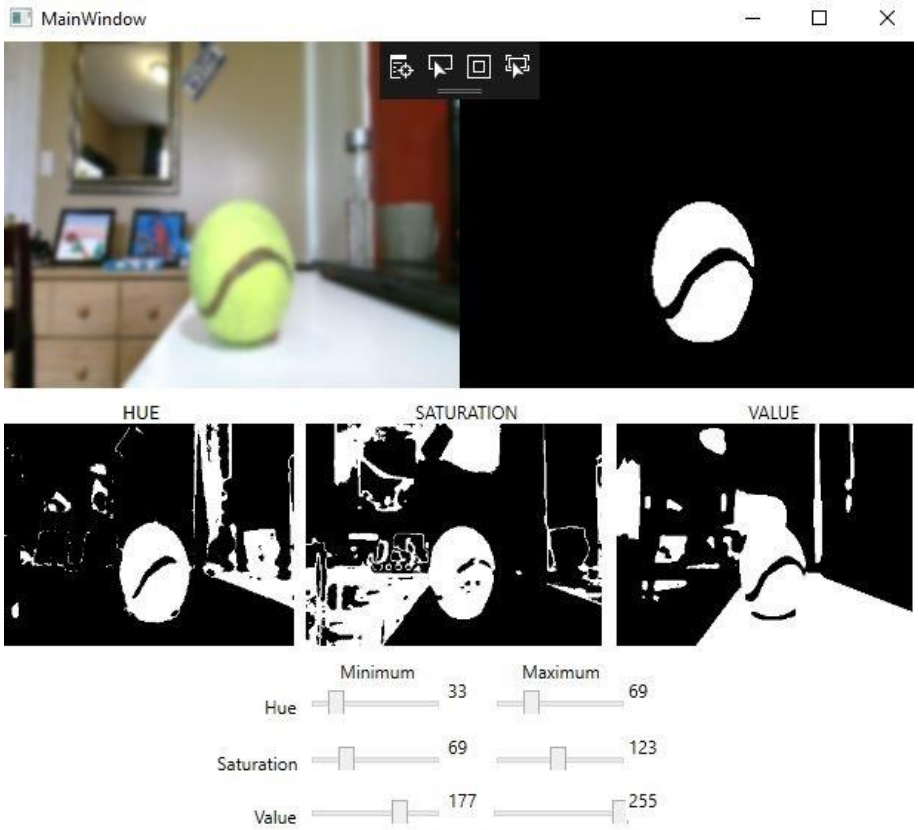

Figure 8. Page where the user of the computerized application selects the parameters for the HSV filter. The user needs to manually adjust hue, saturation and values until the object that s/he wants to track is completely isolated in the top-right canvas.

4 Implementation

The proposed system architecture is shown in Figure 9. The application was developed to specifically address the research requirements, and it was developed using Visual Studio 2013, as the integrated development environment, in connection with the Microsoft Kinect SDK, as the development kit that accesses the Kinect's raw data. All the code was written in C#. The software MATLAB was also used to plot the Kinect's data in order to provide a better comprehension of the performed movement. The rest of this chapter is divided into sections to better organize the steps performed along the software development.

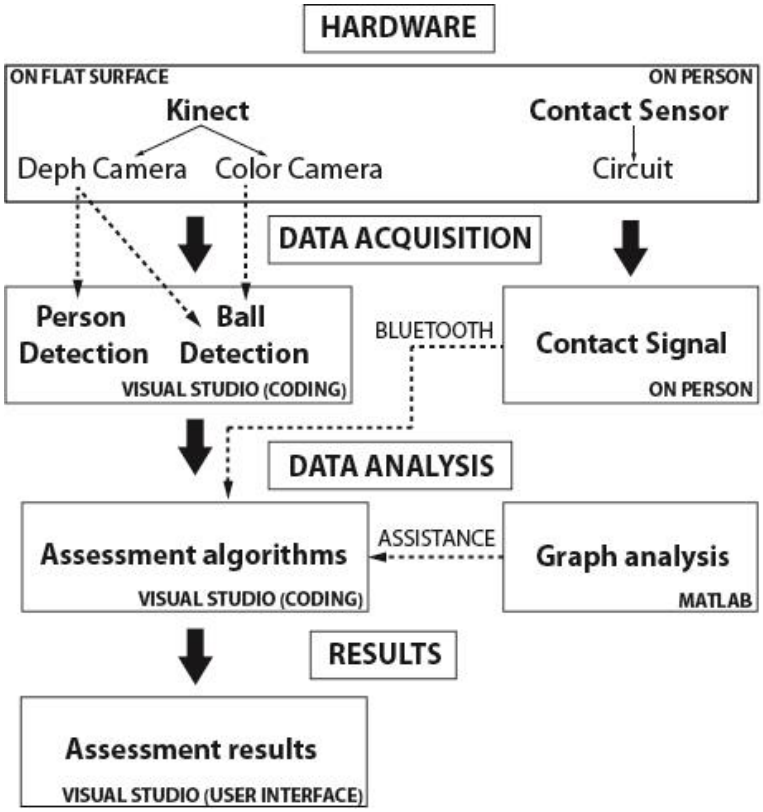

Figure 9: Overall diagram of the proposed system showing its main components and the steps performed until the feedback is returned to the user.

4.1 Hardware

The hardware is composed of the Kinect sensor and the contact sensor. During the administration of the test, the Kinect has to be placed on a flat surface facing the user and the contact sensor must be attached to the body part that might come in contact with the ball or the surface.

4.2 Data Acquisition

The body tracking capabilities of the Kinect for Windows SDK is used to perform the 3D skeleton detection of the user by requesting data from the depth camera stream. The color camera and depth camera streams work together with OpenCV libraries to detect the ball coordinates by its color. The code for the contact sensor was written in Arduino IDE and loaded onto the chip. Then, the pushbutton signal state (pressed or not pressed) was sent to Visual Studio via Bluetooth connection to integrate with the main code. A set of algorithms was written to determine whether the user passed or failed the test.

4.2.1 Kinect Sensor Placement

According to Microsoft (2010), "the sensor works best when it is positioned somewhere between 2 feet (0.6 m) and 6 feet (1.8 m) off the floor". Also, the best accuracy of the depth data happens when the user keeps a distance from 0.5 meters to 4.5 meters away from the sensor (Microsoft, 2015). Therefore, the Kinect was placed at the laboratory following the Microsoft recommendations, facing the participant.

4.2.2 Body Tracking

A sample from Microsoft written in C# called “Body Index Basics – WPF” was adapted to obtain and visualize the body index frame from the sensor. Additional algorithms were added into the sample in order to store the 3D position of the 25 body joints along a pre-determined time. The Body Index Basics sample provided a great baseline for getting started with human motion tracking in the project.

4.2.3 Kinect Sensor Calibration

Each body joint is measured in a 3D space and its values are given in X, Y and Z coordinates. The origin (X = 0, Y = 0 and Z = 0) is located at the center of the IR camera. The X axis extends to the sensor’s left, the Y axis extends up and the Z axis extends out in the direction the sensor is facing (Microsoft, 2014).

The classical tests are applied by a professional who must watch the child's performance to assess the quality of the movement. The observer analyzes the whole movement based on real world coordinate system, i.e. he/she considers the floor as the reference basis for the body joints instead of the Kinect’s IR camera basis. Figure 10 below represents the 3D axis with respect to Kinect's coordinate system and to observer’s coordinate system. A transformation from Kinect’s Coordinates to Real World Coordinates was done in order to use the floor as the reference for body joints displacements.

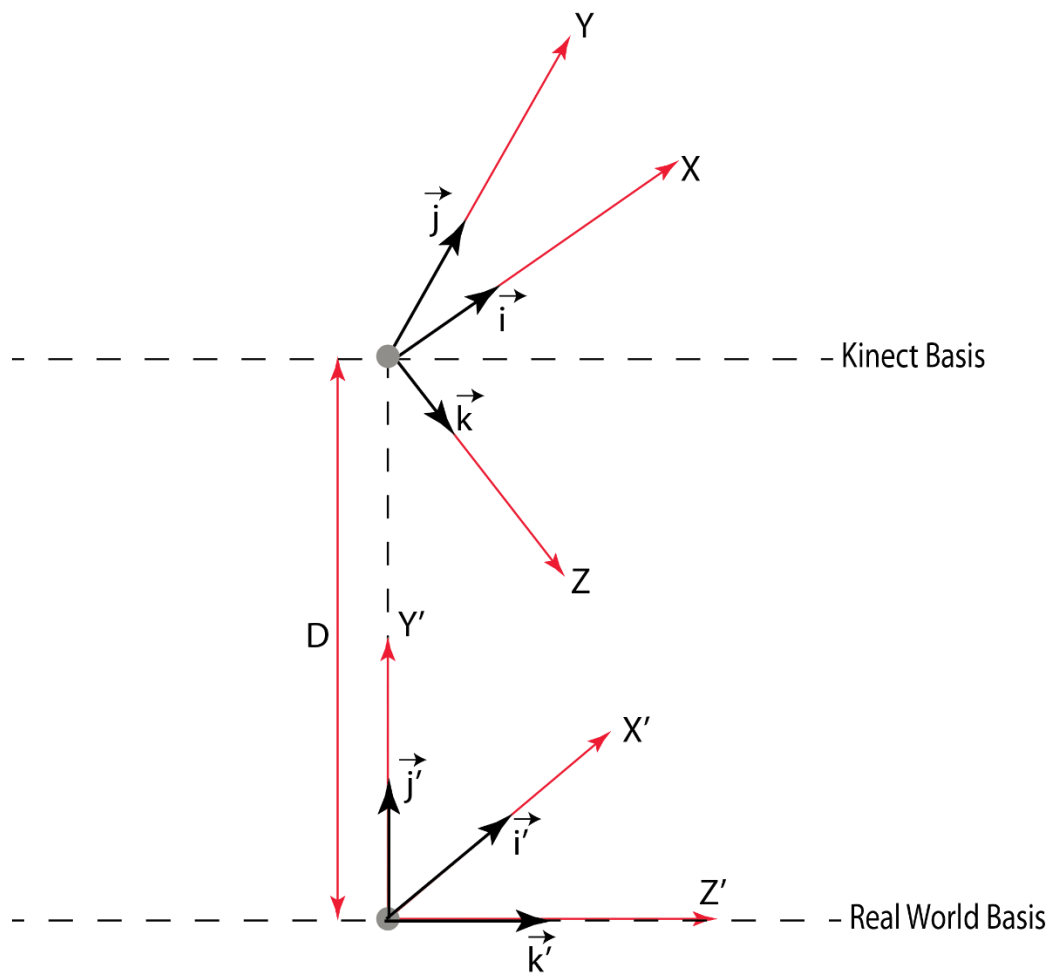

Figure 10: Representation of Kinect's coordinate system vs. Real World coordinate system. The Kinect's coordinate system is located in the center of its IR camera while the real-world coordinate system consists of a rotation around X axis followed by a translation along the new Y axis.

Kinect SDK provides a property called "BodyFrame.FloorClipPlane" that describes the orientation of the floor plane relatively to the camera. Using that property makes it possible to find the coordinate of the real-world coordinate system from the Kinect's coordinate system coordinates. The SDK provides the coordinates of the joints with reference to (X, Y, Z). To transform from (X, Y, Z) to a new system of coordinates (X', Y', Z'), the equation of the floor plane in (X, Y, Z) is:

$$Ax + By + Cz + D = 0 \tag{1}$$

where,

A = bodyFrame.FloorClipPlane.X

B = bodyFrame.FloorClipPlane.Y

C = bodyFrame.FloorClipPlane.Z

D = bodyFrame.FloorClipPlane.W

A, B, C, D are determined by the SDK from measurements by the sensor.

Note from Figure 10 that the real-world coordinate system (X', Y', Z') consists of a rotation around the X axis followed by a translation along the Y' axis. The length of the translation is equal to the distance D, which represents the distance from the sensor to the floor.

Since the X' axis is parallel to the X axis, then:

$$\vec{\imath}' = \vec{\imath} \tag{2}$$

The vector $\vec{\jmath}'$ is perpendicular to the floor plane, which is described in (1), so:

$$\vec{\jmath}' = A\vec{\imath} + B\vec{\jmath} + C\vec{k} \tag{3}$$

The vector $\overrightarrow{k'}$ is perpendicular to the vectors $\vec{\imath}'$ and $\vec{\jmath}'$, then:

$$\overrightarrow{k'} = \vec{\imath}' \times \vec{\jmath}' \tag{4}$$

$$\overrightarrow{k'} = \begin{bmatrix} \hat{\imath} & \hat{\jmath} & \hat{k} \\ 1 & 0 & 0 \\ A & B & C \end{bmatrix} \tag{5}$$

Finally, the vector $\overrightarrow{\imath'}$ can be calculated by the cross product between the vectors $\vec{\jmath}'$ and $\vec{k}$:

$$\vec{\imath}' = \vec{\jmath}' \times k' \tag{6}$$

Therefore, the transformation from Kinect's coordinate system to Real World coordinate system can be described by the homogeneous transformation matrix:

$$\begin{bmatrix} \vec{i}'_x & \vec{i}'_y & \vec{i}'_z & 0 \\ \vec{j}_x & \vec{j}'_y & \vec{j}'_z & D \\ \vec{k}_x & \vec{k}_y & \vec{k}_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Jx \\ Jy \\ Jz \\ 1 \end{bmatrix} = \begin{bmatrix} Jx' \\ Jy' \\ Jz' \\ 1 \end{bmatrix} \tag{7}$$

where,

$\begin{bmatrix} \vec{i}'_x & \vec{i}'_y & \vec{i}'_z \\ \vec{j}_x & \vec{j}'_y & \vec{j}'_z \\ \vec{k}_x & \vec{k}_y & \vec{k}_z \end{bmatrix}$ represents a rotation

$\begin{bmatrix} 0 \\ D \\ 0 \end{bmatrix}$ represents a translation

$\begin{bmatrix} Jx \\ Jy \\ Jz \end{bmatrix}$ are the body joints with respect to Kinect's coordinate system

$\begin{bmatrix} Jx' \\ Jy' \\ Jz' \end{bmatrix}$ are the body joints with respect to real-world coordinate system

After all body joints have been stored into arrays, an algorithm was developed to apply the transformation described by (7) in order to have all coordinates related to the floor.

4.3 Data Analysis

After transforming the collected data to real world coordinate system, the data was filtered, since the raw data contains a substantial amount of noise. An algorithm that implements a low-pass Butterworth filter was developed. The parameters of the filter shown to be the most suitable for the application used a cut-off frequency of 5 Hz and an order equal to four. The same parameters

were used by (Bailey & Bodenheimer, 2012). The filter was developed based on Equation (8), where "G" represents the Butterworth filter gain, "n" represents the filter order, "f" represents the frequency and "f_c" represents the cut-off frequency.

$$G^2(f) = \frac{1}{\left(1 + \left(\frac{f}{f_c}\right)^{2n}\right)} \tag{8}$$

The body tracking data captured and store in arrays is a 3D type of data that represents distances in meters of a body joint with the Kinect's IR camera as reference.

The TGMD-3 Scoring Sheet was used as reference to develop the algorithm to automatically assess each test. The data analysis was based on the proper performed movement, since the TGMD-3 is a criterion-referenced test which considers qualitative forms to correctly perform some specific movements and then checks if the performance of the user matches that pre-determined form. The movement performed by the user is then compared to this standard form to determine the results. In this way, samples of the body joints of the standardized performance were collected. In order to do this, a person who is very familiar with the TGMD-3 test executed the movement in front of the Kinect while all the body joints were being saved to an Excel file. Then, the software MATLAB was used as an auxiliary tool to plot the variation along time of the joints to provide a better visualization of the movement. The graphs were used to comprehend the behavior of each relevant joint at a specific frame. The steps for testing can be summarized as follows:

1. Identify movement phases and/or its critical points
2. Identify the joints or members important to evaluate the movement based on performance criteria of TGMD and movement execution

3. Establish geometrical criteria for angles and XYZ positions of joints, members and used objects (ball, bat and paddle)
4. Execute movement properly and plot variables of joints and members (angles and XYZ positions)

In order to keep the text organized, each of the following subsections contains a figure of the standardized performance of the respective skill and an individual section for each criterion describes the logic behind the algorithms used to assess it.

4.3.1 Run

Kinect's range of operation lies between 0.4m to 4.5m, and the running test requires at least 18.3 meters of clear space, according to the TDMG-3 scoring sheet. Because of that, the participant had to perform the running movement on a treadmill and the sensor was positioned facing the participant's back, as shown in Figure 11. It would have been better to place the Kinect facing the participant and remove the posts, but this was not allowed in the gymnasium where the tests were performed due to safety reasons.

Figure 11: Set up for the running test. The participant runs on a treadmill and the Kinect is placed on a flat surface facing the participant's back.

An example of a running movement is shown in Figure 12 below, where the Kinect is positioned facing the participant's back.

Figure 12. Sequence of images to represent a standardized running movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

Note from the picture that running is a movement characterized by a synchronized stride and an aerial phase whereby both feet come off the surface for a brief period of time. Also note that, as the movement progresses, the head goes up and down for each stride, which causes the Y coordinate of the head to draw peaks along time, just like illustrated in Figure 13. In this way, the movement can be divided into cycles, which can be determined by each head peak. Note in Figure 12 that for each cycle, i.e. for each head peak, the leading leg alternates and the movement follows this synchronized pattern until the participant decides to stop running.

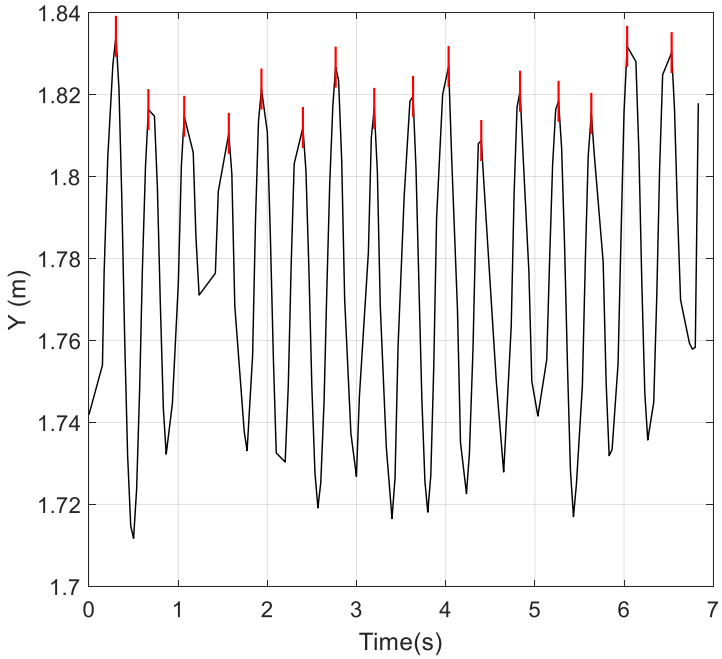

Figure 13. Y coordinates of head. The solid black line represents the variation along time of the Y coordinates of the head and the red traces represent each head peak.

In order to determine the moment when each stride occurs, an algorithm that identifies peaks is used to save the index of all frames when the participant performs a stride. Before starting the performance of the skill, the application runs a body calibration in order to store the participant's joint information when s/he is standing on the floor. The algorithm that identifies peaks uses the Y coordinate of the head obtained during the calibration to determine a threshold for the peak identification. For example, when the participant stands on the floor, the Y coordinate of the head is approximately the participant's height. When s/he leaves the floor, the Y coordinate of the head reaches a value higher than the one calibrated. In this way, for every moment the participant leaves the surface, the algorithm stores the values above that threshold and then takes the maximum one; consequently, by the time the movement is done, the algorithm knows at which frame all the head peaks occur.

The following flowchart illustrates how to determine the head peaks:

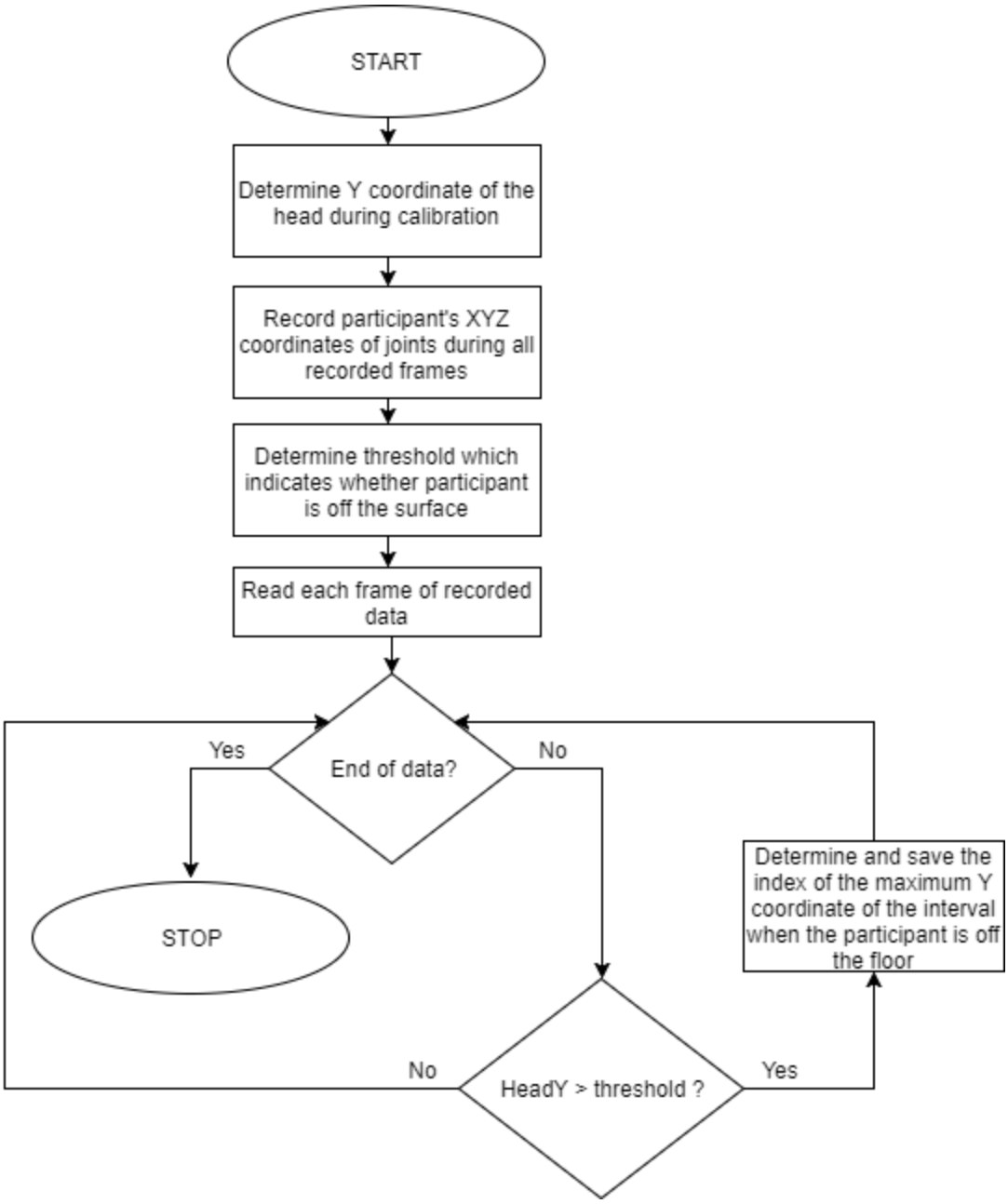

Figure 14. Algorithm for head peaks identification.

Criterion 1: Arms move in opposition to legs with elbows bent.

The first criterion consists of analyzing whether arms move in opposition to legs with elbows bent. In order to do that, the algorithm checks if for each stride the participant has his/her arms in opposition to legs. As mentioned before, the moment when the head reaches a peak can be considered one stride of the run and so on. Figure 15 shows the behavior of the Z coordinate of knees and elbows at each stride. Taking the graph of the knees as an example, note that for each stride one knee is closer to the sensor than the other, and they exchange positions rhythmically during the whole movement. The same behavior repeats for the elbows. Observe that for the same stride, the leading knee, the one closer to the sensor, is in opposition to the leading elbow and same pattern repeats for all strides. This means that while the right knee is leading, the left elbow is also leading and vice-versa, which indicates that the participant is running while moving the arms in opposition to legs. Therefore, by knowing the frames when each stride occurs, the algorithm checks if in between two strides the leading knee and the leading elbow are in opposite side of body.

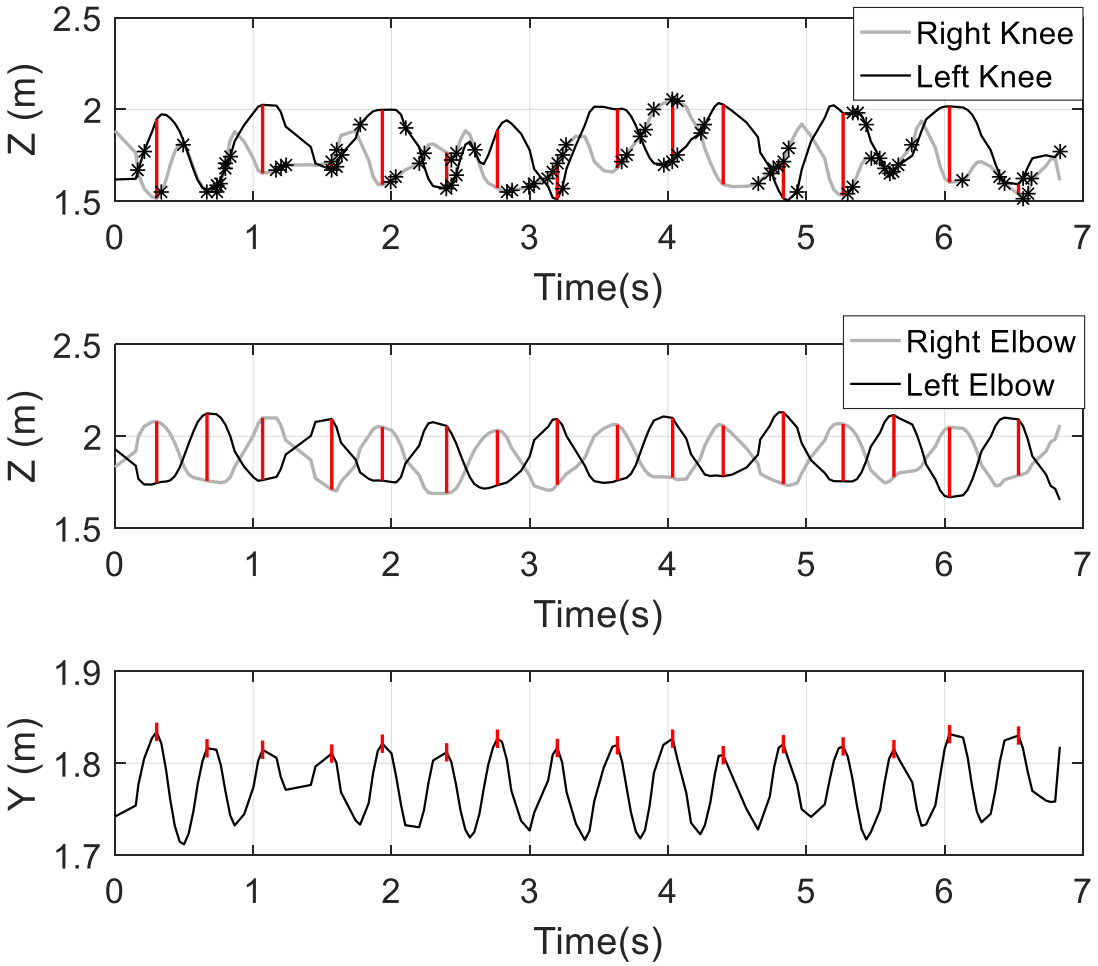

Figure 15. (a) Variation along time of Z coordinates of knees. (b) Variation along time of Z coordinates of elbows. (c) Variation along time of y coordinates of head. The traces in read represents the moment when each head peak occurs. The (*) in the graphs represents the moment when the 3D position of the joint was inferred.

It is important to mention that the body tracking of the Kinect is optimized for when the participant performs the movement facing the Kinect. According to (Microsoft, 2012), "Skeletal Tracking is optimized to recognize users standing or sitting and facing the Kinect". Moreover, as the running movement is performed quickly, it could lead to confusion for the body tracking algorithm. In situations like that, when some joints cannot be fully tracked by the sensor, their position is "calculated from surrounding joint data rather than captured by the camera. Since the data is

calculated, confidence in the data is very low" (Microsoft, 2017). An example of a confusion in depth sensor is illustrated in Figure 16; note that the left leg is drawn with thinner lines than the other joints, which is an indication of inferred joints. For all graphs that illustrates the tracked joints behavior along time it is used a star ("*") marker to represent if the joint was inferred or not tracked at that moment. Therefore, if a star is present on the graph, it means that at that specific moment, the joint position is not completely accurate.

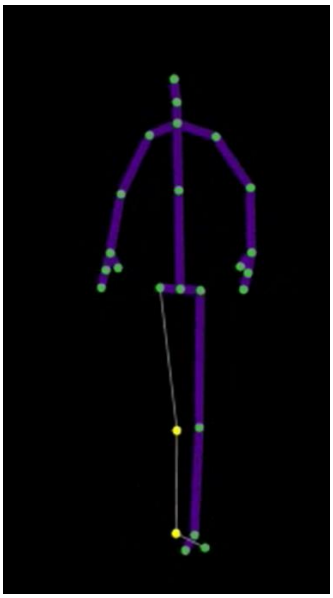

Figure 16. Example of a moment when the joints of the knee and the ankle were inferred. Note that the member that contains those joints were drawn with thinner lines.

Considering possible confusion in the depth sensor, despite the fact that the algorithm checks for all strides if the leading knee is in opposition to the leading elbow, it gives a tolerance of 20% for the condition not meeting the requirement. Hence, the algorithm determines if for at least 80% of the total number of strides the leading knee is in opposition to the leading elbow.

The following flowchart illustrates how to determine if arms move in opposition to legs:

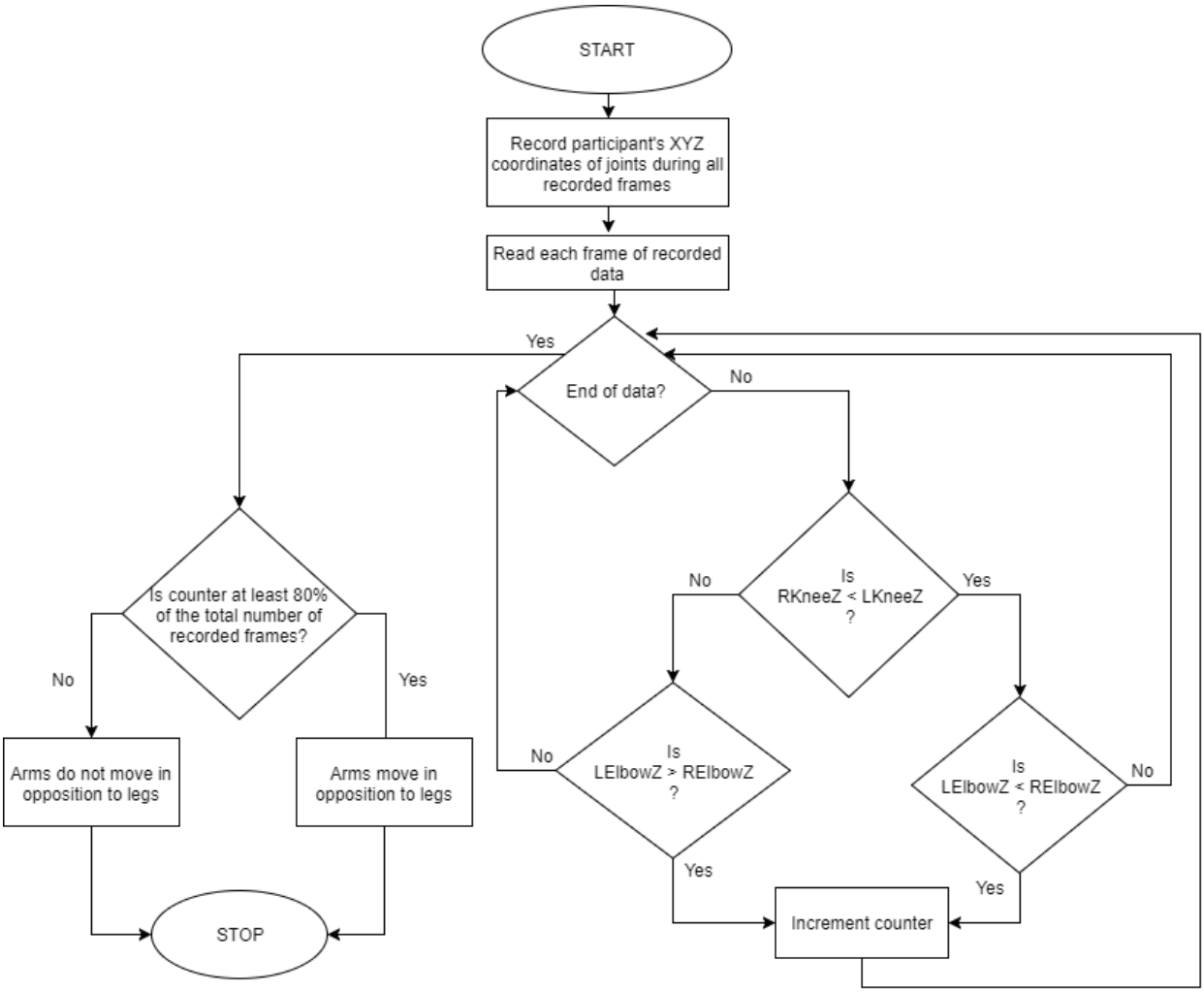

Figure 17. Algorithm to assess whether arms move in opposition to legs.

The algorithm also needs to check if the elbows are bent to complete the analysis of the first criterion. Figure 18 illustrates a situation when the arm is extended and when the arm is flexed at about 120°. Note from the picture that as the distances between shoulder and elbow and between elbow and hand are constant, if the participant flexes his/her arm, the Y coordinate of the hand will be close to that of the shoulder. Therefore, it is possible to calculate a ratio that relates the position of the hand to the position of the shoulder to estimate the value of the elbow's angle.

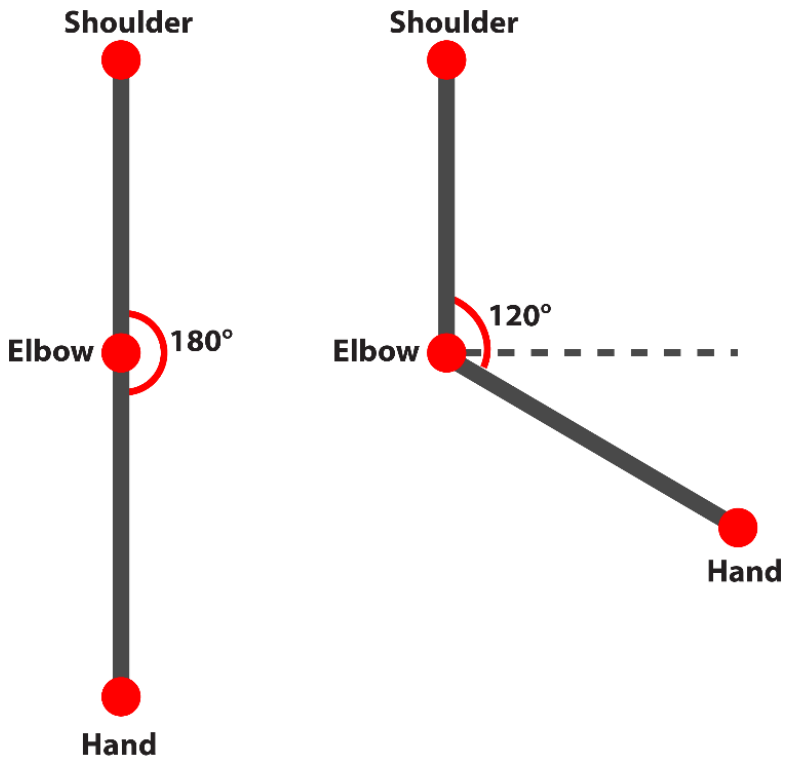

Figure 18. Representation of arms extended (left) and arms flexed (right).

During the calibration the participant is asked to stand in front of the sensor with arms extended in order to save the participant's measurements. That way, the ratio is given by the formula:

$$ratio = \frac{ShoulderY - HandY}{ShoulderY_{calibration} - HandY_{calibration}} \tag{9}$$

Note that when the arm is extended, the value of $ShoulderY$ is equal to $ShoulderY_{calibration}$ and $HandY$ is also equal to $HandY_{calibration}$. So, when the arm is fully extended, the ratio is equal to or very close to "1". However, the more the participant starts flexing the arm, the more the ratio decreases in the direction of "0". Moreover, a threshold equal to 120° is adopted to consider if the participant has his/her arms flexed or not. Any value of angle below that threshold is considered to constitute flexed arms.

Considering that the distance between the shoulder and the elbow and the distance between the

hand and the elbow are approximately the same, then the Y coordinate of the shoulder is approximately equal to the double of the Y coordinate of the hand. Therefore, an approximation for the ratio when the arm is flexed at 120° is equal to:

$$ratio = \frac{ShoulderY - HandY}{ShoulderY_{calibration} - HandY_{calibration}}$$

$$ratio = \frac{2 \times ShoulderY - \sin(30°) \times ShoulderY}{2 \times ShoulderY}$$

$$ratio = 0.75$$

Any value equal or below 0.75 means that the participant has the arms flexed, otherwise not flexed.

Figure 19 illustrates the behavior of the ratios of the arms. Note that for the majority of the head peaks, the ratio is below 0.75. Due to the position of Kinect sensor be facing the participant's back, assuming that the participant maintains his/her elbows flexed, the joints of the hands will be hidden most of the time. As mentioned before, Kinect deals very well with cases like this, when the joints are hidden. In these situations, the sensor infers the position of the hidden joints and despite that it does this faithfully, sometimes the inferred position can be miscalculated, which explains some cases when the ratio is above 0.75 in Figure 19.

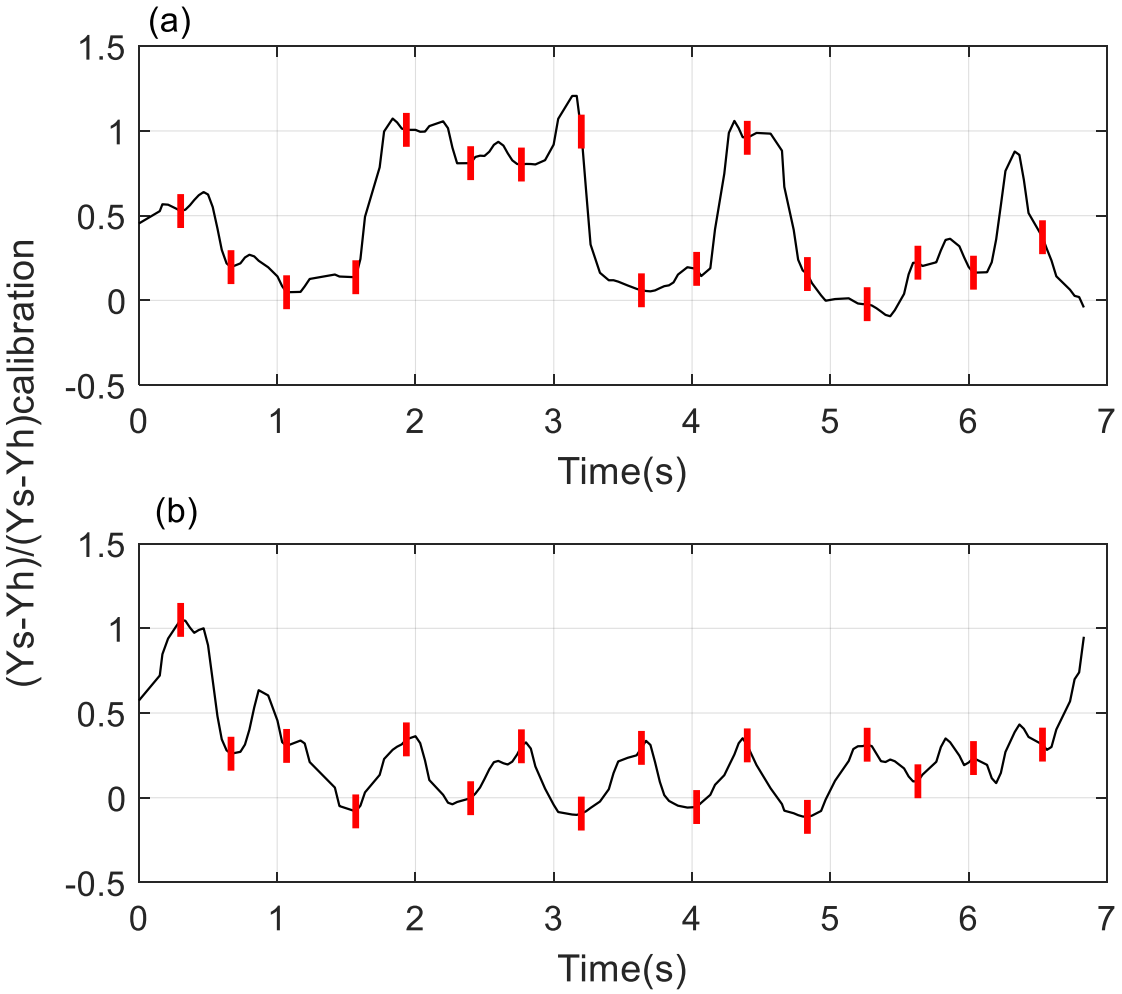

Figure 19. (a) Ratios of right arm at each critical point. (b) Ratios of left arm at each critical point. The traces in red represent the head peaks.

Therefore, to assess whether the participant runs with elbows flexed, the algorithm checks if at each stride, the ratios of both elbows are equal or lower than 0.75. In order to prevent miscalculations due to hidden joints, the algorithm gives a tolerance of 20% of occurrences when the ratio is above 0.75 during the performance of the running test.

The following flowchart illustrates how to determine if the participant runs with elbows flexed:

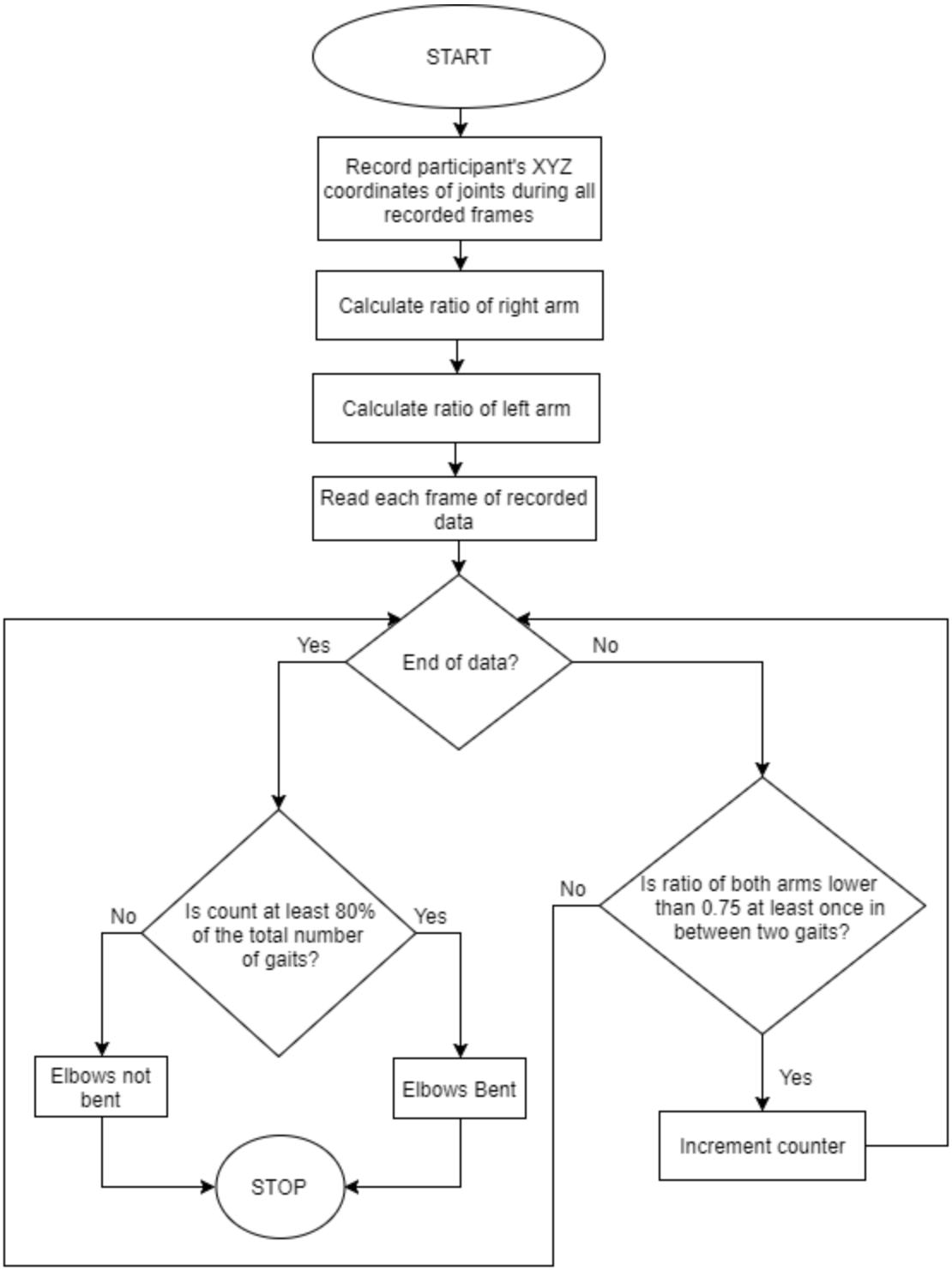

Figure 20. Algorithm to assess whether elbows are bent.

The following flowchart illustrates the feedback for the first criterion:

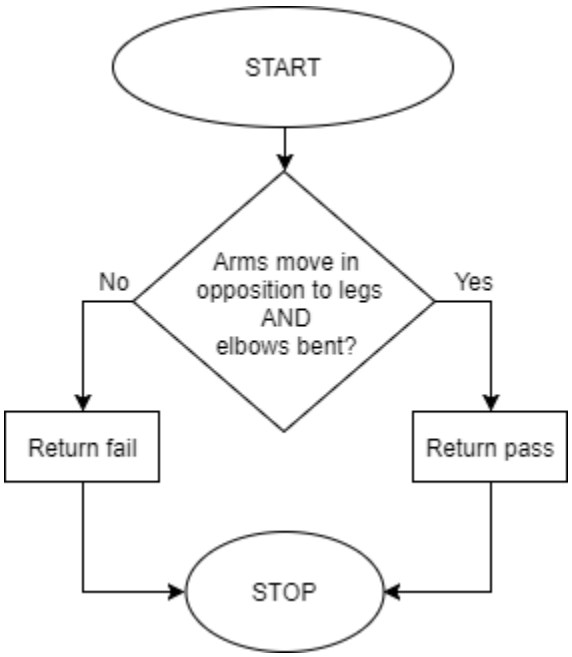

Figure 21. Algorithm to return feedback for the first criterium.

<u>Criterion 2: Brief period where both feet are off the surface.</u>

Auvinet et. al (2015) suggests that feet joints are often incorrectly recognized by the sensor since "depth images could lead to confusion between the ground and the feet around the contact phase". Therefore, it is used the joints of the knees instead of the feet to analyze when the participant is off the surface, since it can be more accurate.

The second criterion of the running test consists of analyzing whether both feet come off the surface for a brief period of time. In order to assess this, the algorithm checks if both Y coordinates of the knees are off the floor at each stride. The algorithm that detects the head peaks stores in an array all the frames that a stride happens. So, it compares if at each stride the Y coordinates of both knees are at least 10% above its respective value acquired during the calibration. The value of the

calibration is based on the distance from the surface that each knee is at when the participant is standing on the floor level. Figure 22 shows the Y coordinate of both knees, as well as the standing level for the knees and the threshold. Note that with the exception of one stride, for all other stride cycles both knees are above the threshold, i.e. both knees are off the surface.

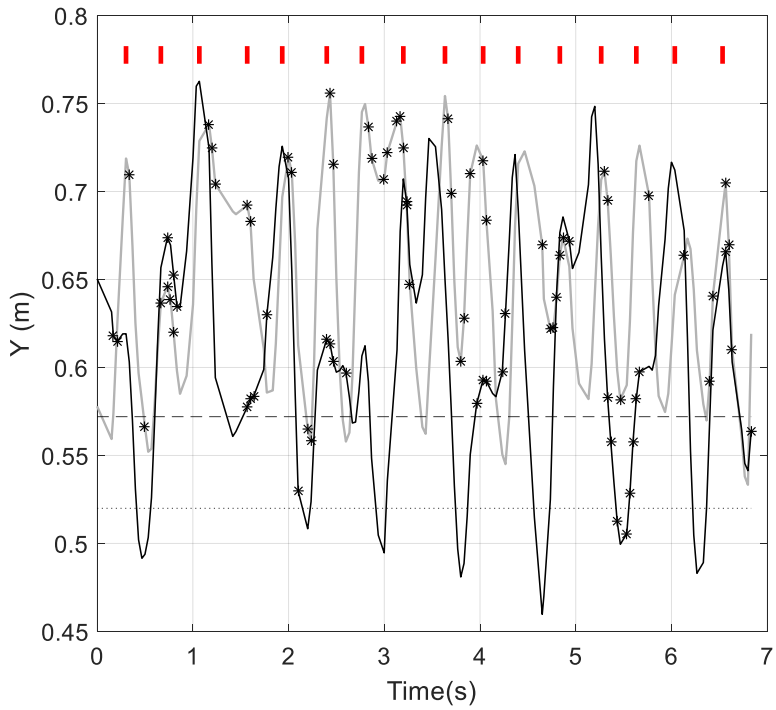

Figure 22. Variation along time of Y coordinates of knees. The grey solid line represents the right knee, the black solid line represents the left knee, the dotted line represents the standing level, the dashed line represents the threshold, the (*) represents the moment when the 3D position of the joint was inferred, the traces in red represent the head peaks.

The algorithm gives a tolerance of 20%, i.e. both of knees have to be above the threshold for at least 80% of the total number of strides. This is done to prevent false results due to possible depth image confusion. If the participant meets the condition for at least 80% of the occurrences, the algorithm returns “pass”.

The following flowchart illustrates how the algorithm assess the second criterion:

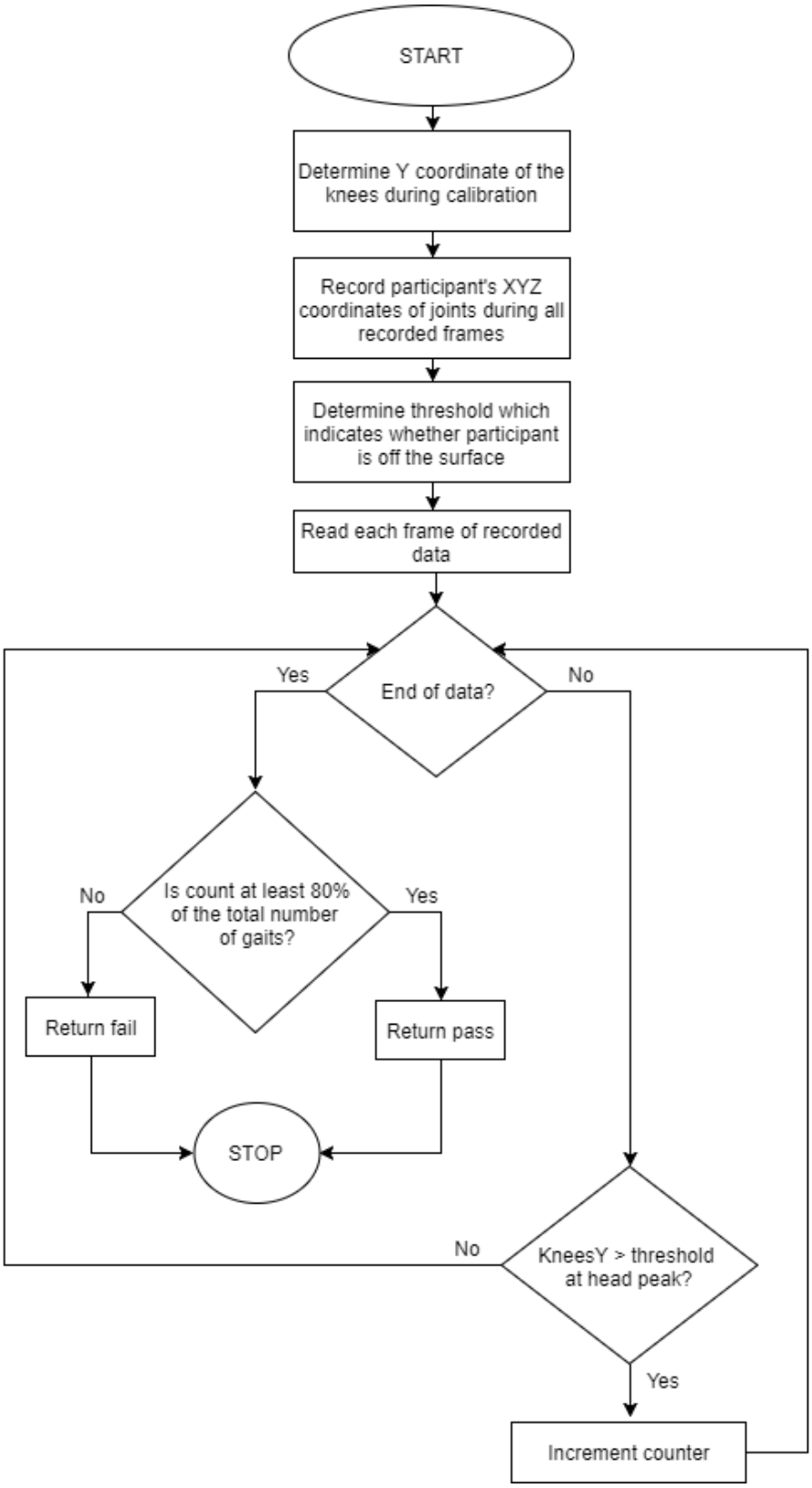

Figure 23. Algorithm to assess and return feedback for the second criterion.

Criterion 3: Narrow foot placement landing on heel or toes (not flat-footed).

The third criterion checks for narrow foot placement landing on heel or toes (not flat-footed). In order to analyze if the participant runs with narrow foot placement, the algorithm compares the distance between the X coordinate of knees acquired during the calibration to the distance between them during the performance of the movement. During the calibration, the distance in X between knees is considered to be the normal distance when the participant is standing. Therefore, if the participant runs with wide foot placement, the distance between the knees will be larger than the distance from the calibration, otherwise the distances should be around the same. Following this idea, the code compares the average distance between the X coordinate of the knees during the whole movement to the value acquired during the calibration. If the mean is lower that the distance acquired during the calibration, the foot placement is considered narrow. Figure 24 shows an example of narrow foot placement. Note that the mean is lower than the calibration value.

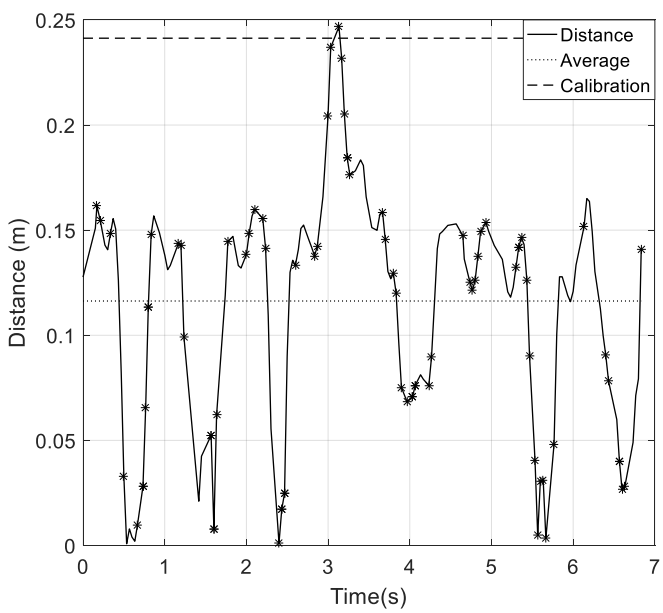

Figure 24. Variation along time of the distance between knees, its average and calibration values. The (*) represents the moment when the 3D position of the joint was inferred.

The following flowchart shows how the algorithm checks for narrow foot placement:

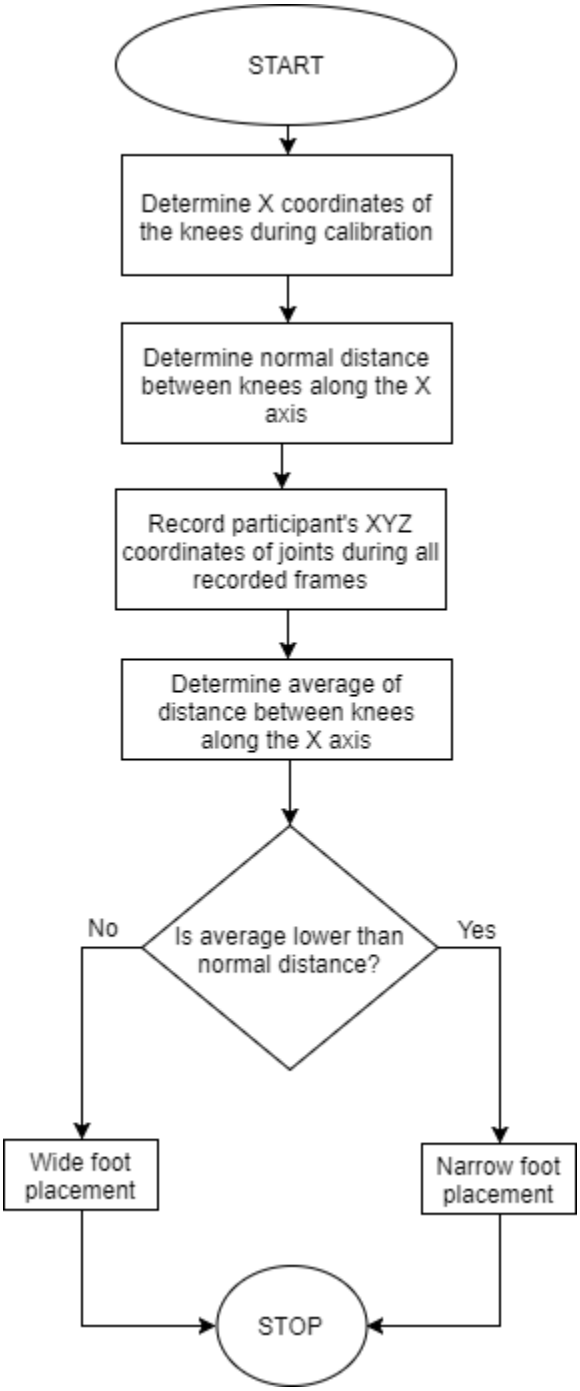

Figure 25. Algorithm to assess whether participant runs with narrow foot placement.

The third criterion also checks if the feet land on heels or toes, not flat-footed. In order to assess that the participant is asked to run with socks that contains contact sensors positioned under each heel and under each ball of the foot. Depending on how the participant lands on the floor, the sensors will trigger in some specific order. If flat-footed, the sensors of heels will trigger almost at the same time of the ones located under the ball of the feet. Otherwise, they will trigger a fraction of a second apart. Figure 26 shows an example of when the participant lands on the surface on flat-footed. Note that the time lapse between the trigger of the two sensors is equal to 0.097 seconds. Figure 27 illustrates when the sensor located on the ball of the foot fires before the sensor located on the heels. Note that the time lapse is equal to 0.499 seconds. The last scenario is when the sensor located on the heels fires before the sensor located on the ball of the foot. Note in Figure 28 that the time lapse is equal to 0.701. Based on those examples it is clear that the time difference between flat-footed and not flat-footed is significantly large. The algorithm then tracks the output of the Bluetooth during the whole movement. If at any moment it detects that both sensors located on the same foot is triggered within a threshold of 0.15 seconds apart, it considers that the participant has landed flat-footed. If flat-footed happens more than 10% of the total number of strides, the algorithm returns “fail” to the criterion.

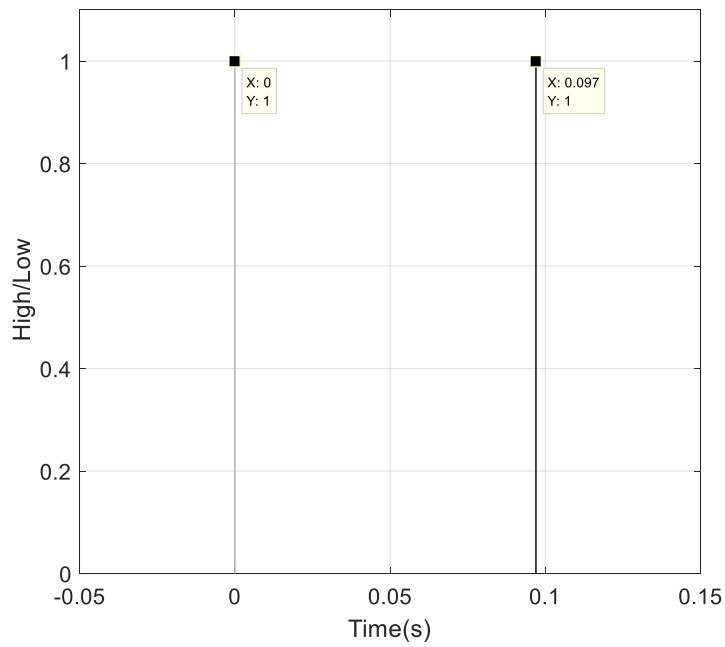

Figure 26. Example of situation when the sensor located on the heel, represented by the black curve, and the sensor located on the ball of foot, represented by the grey curve, trigger at the same time. Note that the time lapse between them is very short (97 ms).

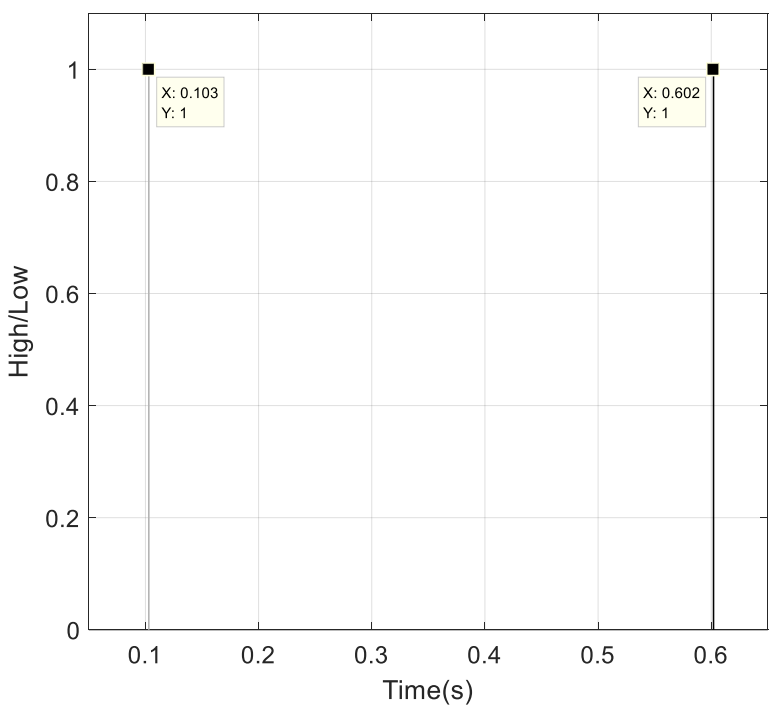

Figure 27. Example of situation when the sensor located on the ball of foot, represented by the grey curve, fires first than the sensor located on the heel of foot, represented by the black curve. Note that the time lapse between them is larger (499 ms) than when they trigger together (97 ms).

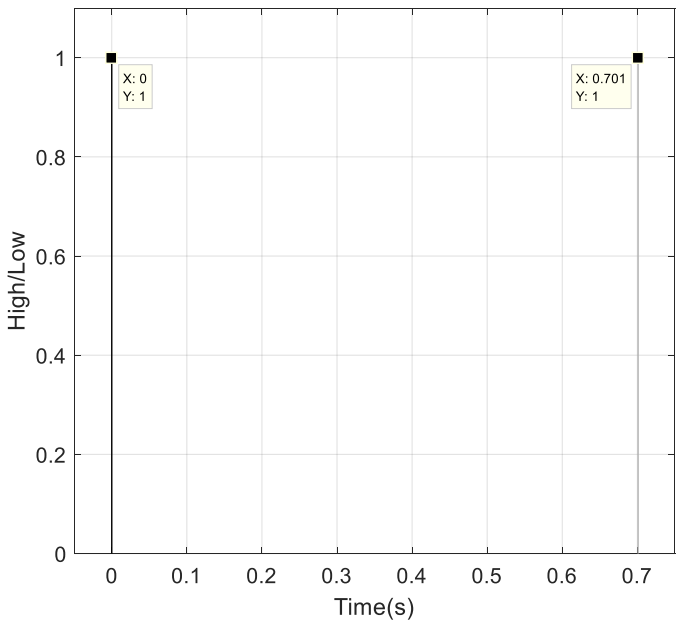

Figure 28. Example of situation when the sensor located on the heel of foot, represented by the black curve, fires first than the sensor located on the ball of foot, represented by the grey curve. Note that the time lapse between them is larger (701 ms) than when they trigger together (97 ms).

The following flowcharts illustrates how to determine if participant lands flat-footed and how the algorithm gives the feedback for the third criterion:

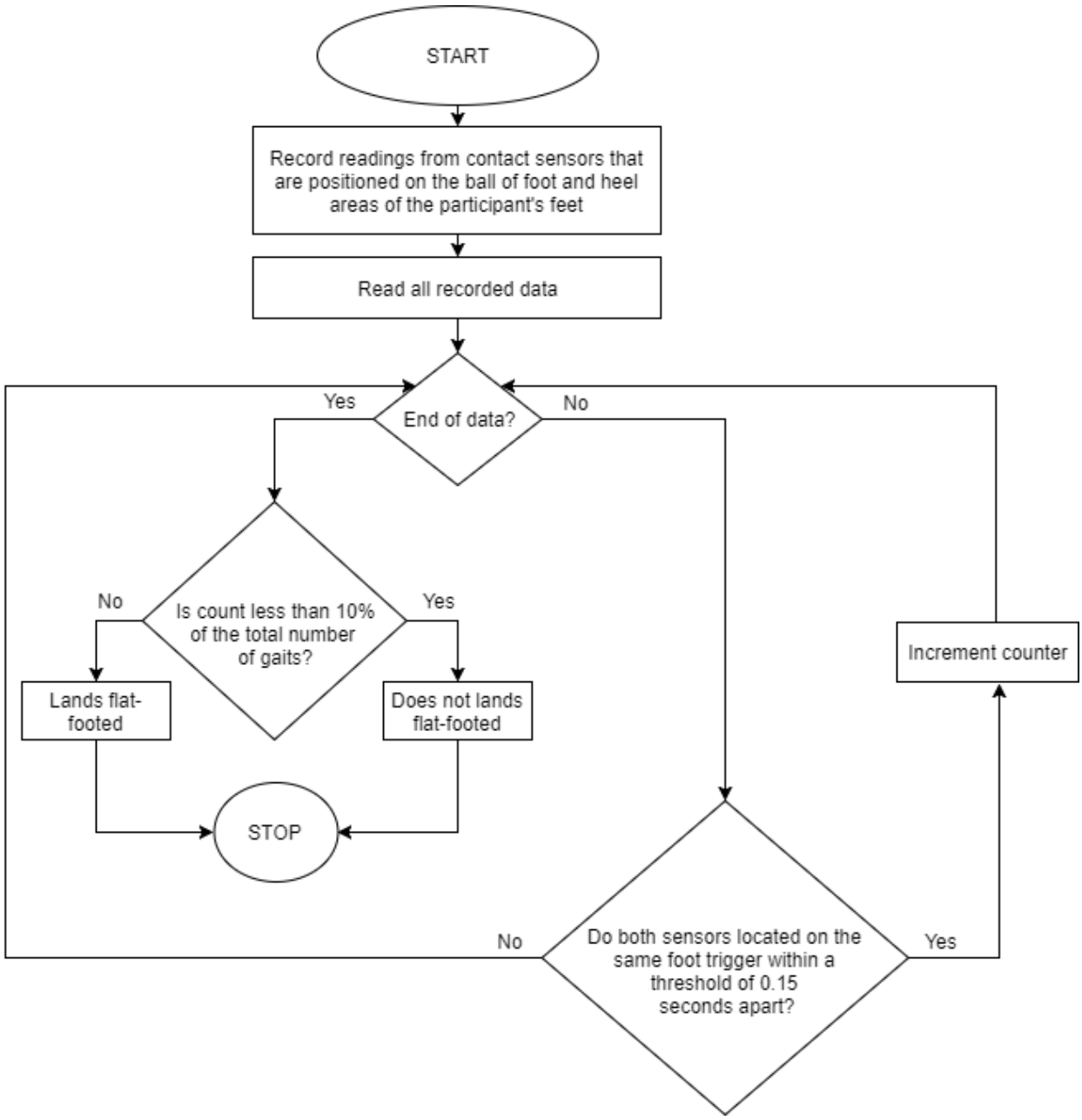

Figure 29. Algorithm to assess whether participant lands flat-footed.

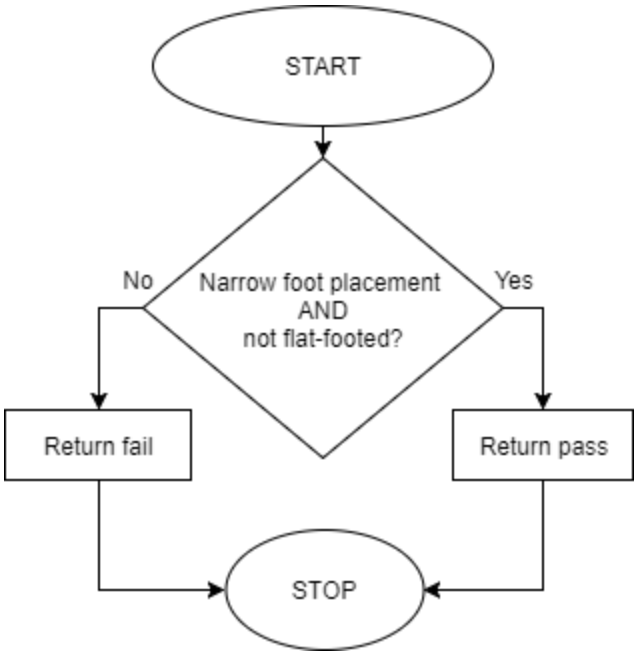

Figure 30. Algorithm to return feedback for third criterion.

<u>Criterion 4: Non-support leg bent about 90 degrees so foot is close to their buttocks.</u>

Note in Figure 12 that the moment when the non-support leg bends about 90° happens in between two consecutive stride cycles. Hence, when the running test is performed as the standardized movement, it is expected to have one of the legs bent at roughly 90° in between two head peaks.

As previously mentioned, the Kinect sensor is optimized to track human body when the participant performs the movement facing the sensor. As in this situation the sensor is facing the participant's back and the running test is performed faster than the other skills, the results for this criterion are shown to be poor. Note that in Figure 12, in between two consecutive strides, one of the legs is bent about 90°, however the graph of the angles of the knees along time shows a different behavior than expected. Note in Figure 31 that the angles of the knees crossed the threshold of 100° just a few times, despite the sequence of frames for the skill in Figure 12 shows clearly that the knees reached angles below 100° in between two head peaks.

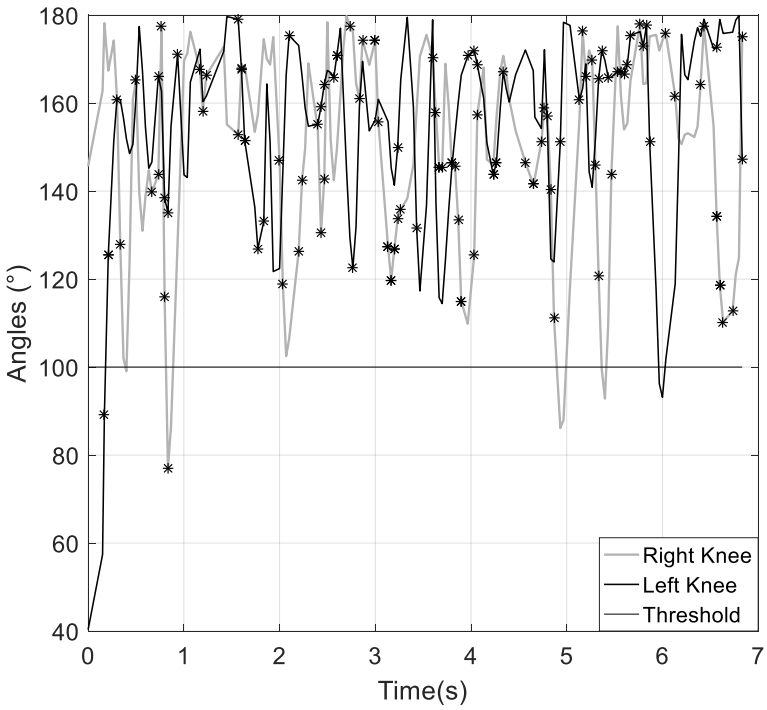

Figure 31. Variation along time of angle of knees. The (*) represents the moment when the 3D position of the joint was inferred.

During the performance of the running test shown in Figure 12, the skeleton tracking drawn by Kinect sensor was also recorded in order to compare to the results. Figure 32 shows some frames of the movement according to the sensor's point of view; note that there are many occurrences where the lower body joints are inferred. This illustrates situations when the depth sensor gets confused about the position of the joint, which leads it to infer the position. Because of that, some of the joint values do not match the real position, which explains the inaccurate graph shown in Figure 31.

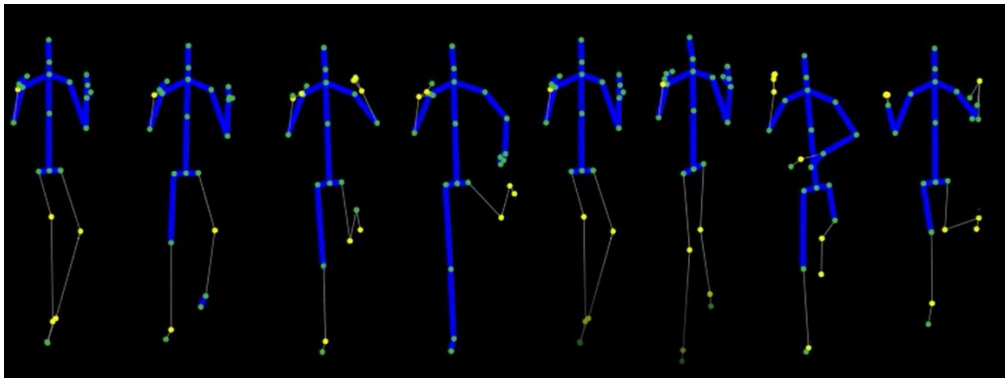

Figure 32. Some examples of skeleton tracked during running test. The left side of body is shown as right and the right side of body is shown as left, since the sensor is positioned facing the participant's back.

Therefore, to assess the criterion the algorithm takes the stride frames and analyzes whether for at least 80% of the total number of stride frames the angle of the non-supporting leg is below 100°. If the condition is satisfied, the algorithm returns "pass" for the criterion. However, if the condition is not satisfied, it returns "fail". Despite of the limitations to assess this criterion due to the confusion in depth sensor, it is important to mention that for most of the trials the algorithm will return "fail" even if the participant has performed the skill correctly.

The following flowchart illustrates how the algorithm assess the last criterion:

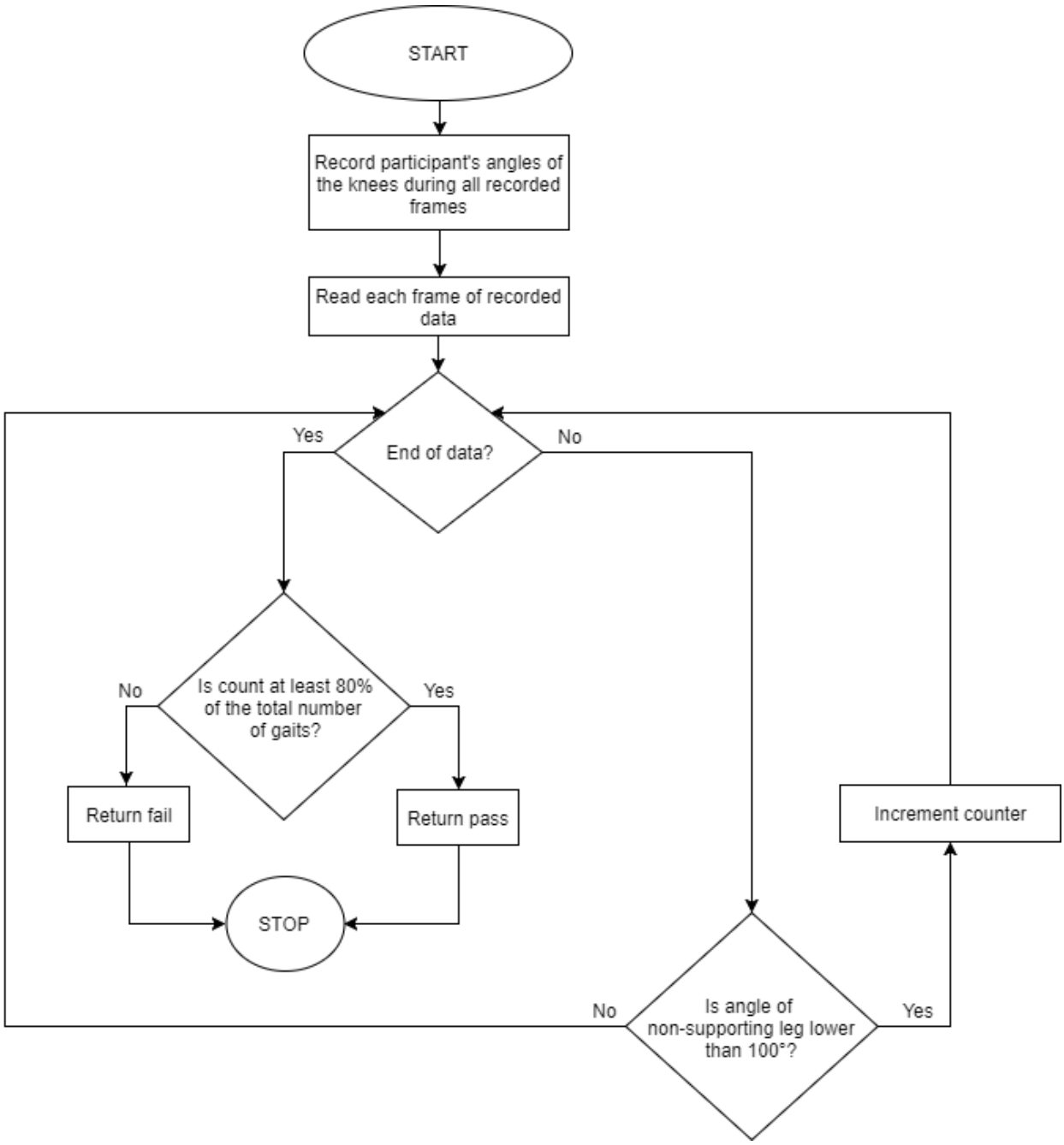

Figure 33. Algorithm to return feedback for fourth criterion.

4.3.2 Gallop

According to the TGMD-3 scoring sheet, the participant is asked to perform four gallops conforming to the specified performance criteria. An example of a gallop movement is shown in Figure 34 below, where the Kinect is positioned facing the participant. It is important to mention that before starting the performance of the skill, the application runs a body calibration in order to store the participant's joint information when s/he is standing on the floor. Those values will be used as a reference for some criteria analysis.

Figure 34. Sequence of images to represent a standardized gallop movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

Note that in Figure 34, the Y coordinate of the head, i.e. the distance from the floor to the head, draws peaks over time. Therefore, each gallop phase is characterized by the period between two head peaks, where each peak detected is a critical point for the analysis. Figure 35 shows the behavior of Y coordinate of the head along time when the participant performs four consecutive gallops as well as the critical points. Note that the head Y coordinate draws peaks over time.

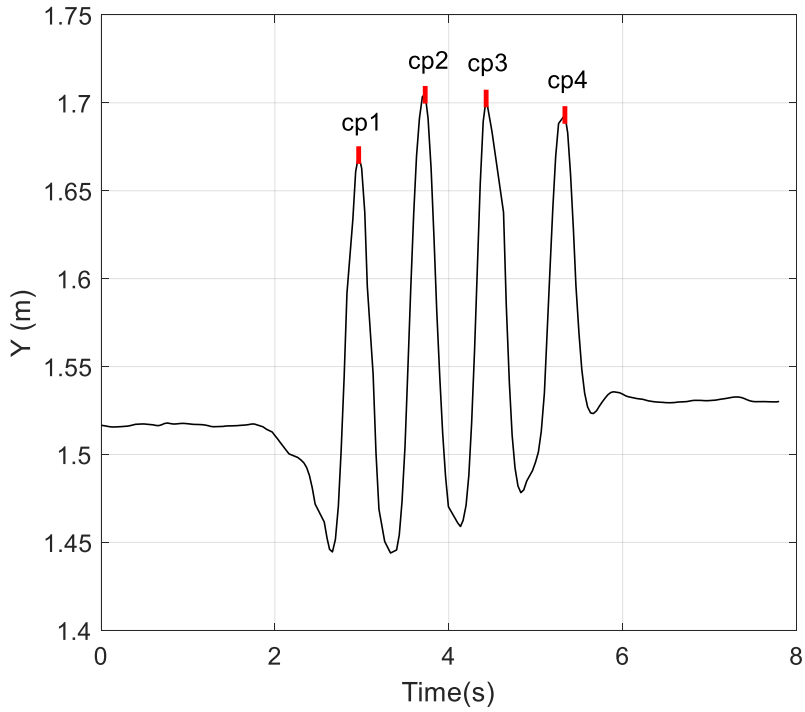

Figure 35. Variation along time of Y coordinates of head. The traces in red represent the head peaks.

<u>Criterion 1: Arms flex and swing forward to produce force</u>

As was seen with the running test, and referring to equation (9), the arm is flexed when the ratio, $\frac{ShoulderY - HandY}{ShoulderY_{calibration} - HandY_{calibration}}$, is less or equal than 0.75. Figure 36 illustrates the ratio of the arms during the performance of the gallop test. Note that in between one critical point to the next one, both arms reach values of ratios below 0.75, which means that the participant had his/her arms flexed less than 120° for each gallop.

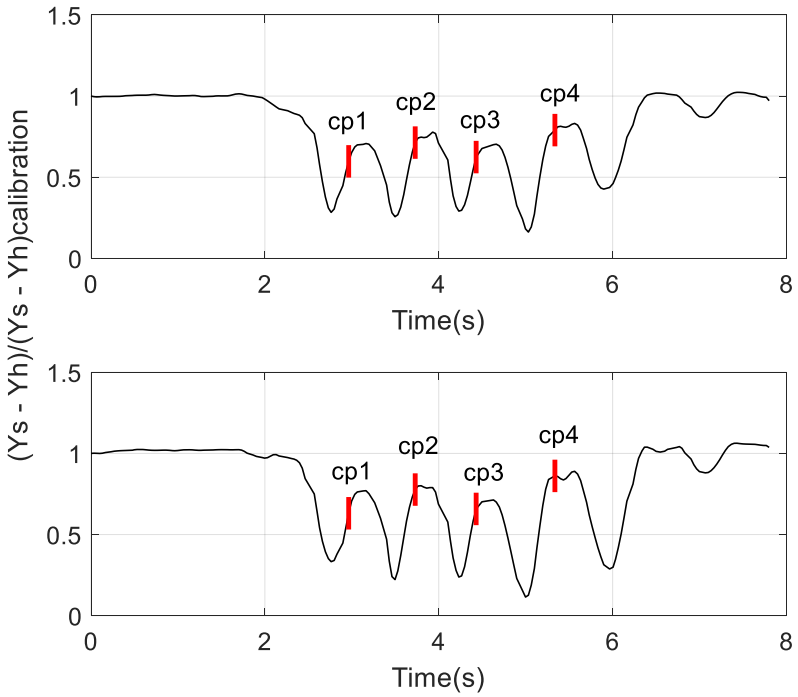

Figure 36. (a) Ratios of right arm at each critical point. (b) Ratios of left arm at each critical point. The traces in red represent the head peaks.

The flowchart that illustrates how to checks if arms are flexed is:

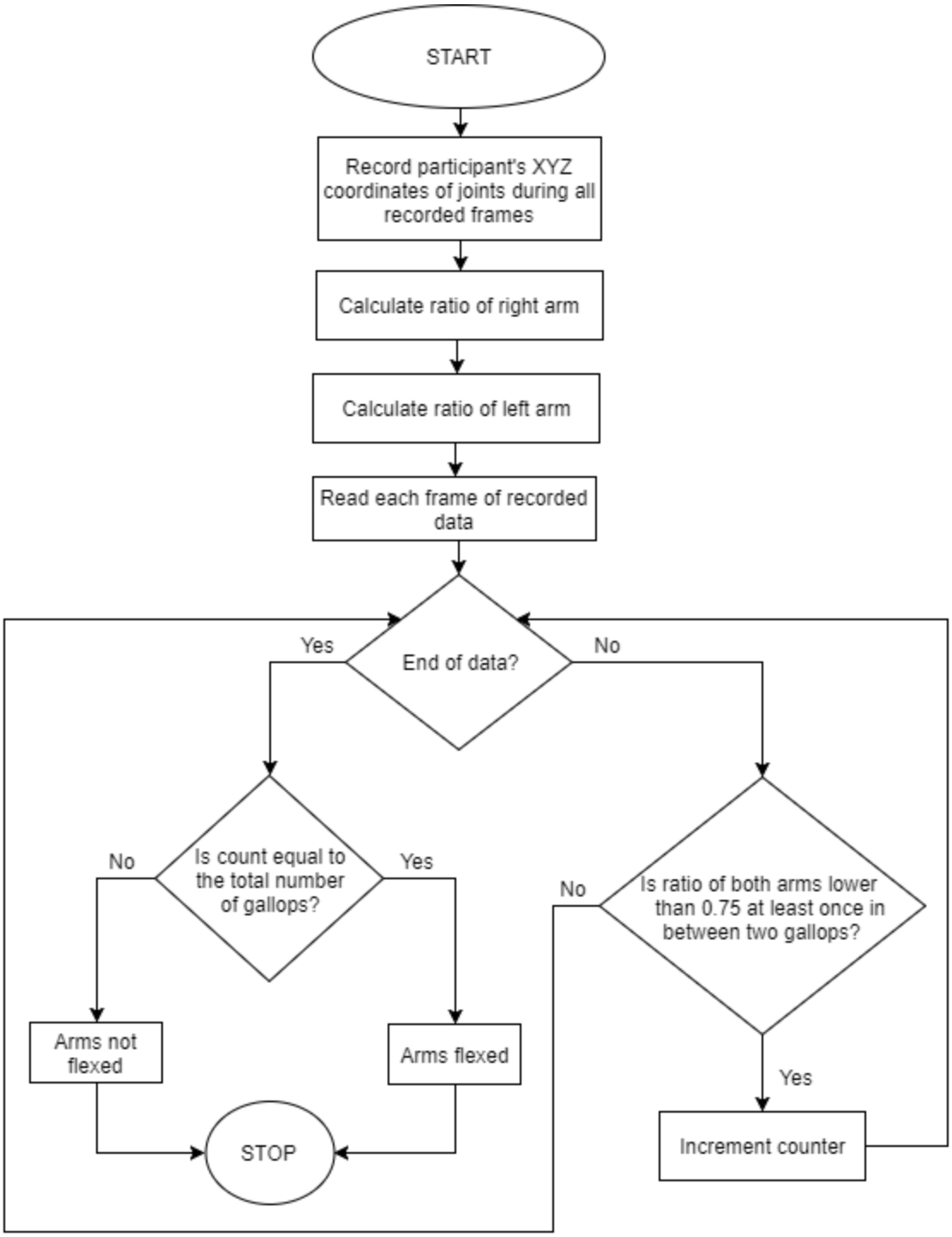

Figure 37. Algorithm to assess if participant flexes arms during the movement.

In addition, the algorithm also has to check if the arms swing forward to produce force. Figure 38 shows an example of a swing forward and a swing backward using the arms. Note in Figure 34 that

as the movement progresses, the participant has to swing the arms as illustrated in Figure 38 to produce force.

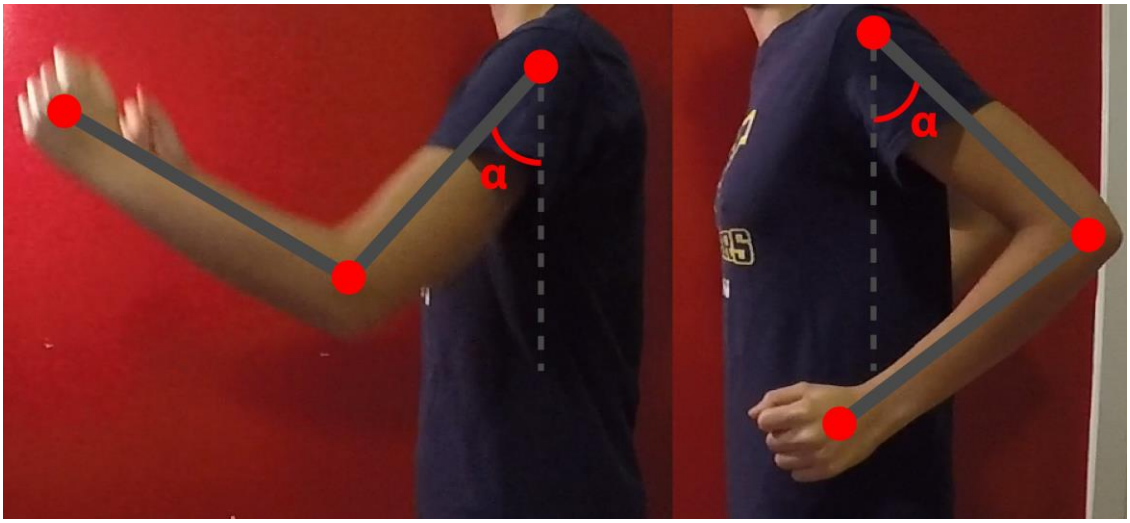

Figure 38. Illustration of how the angle alpha varies during a swing using arms.

A swing can be identified by an angle α formed between the upper arm and the line that represents the trunk. The angle α can be calculated as follows:

$$\alpha = \tan^{-1}\left(\frac{Elbow\ Z - ShoulderZ}{ShoulderY - ElbowY}\right) \tag{10}$$

Moreover, the angle α is positive or negative depending on whether the elbow is in front of or behind the trunk. Figure 39 illustrates the variation of the angle α during the performance of the movement.

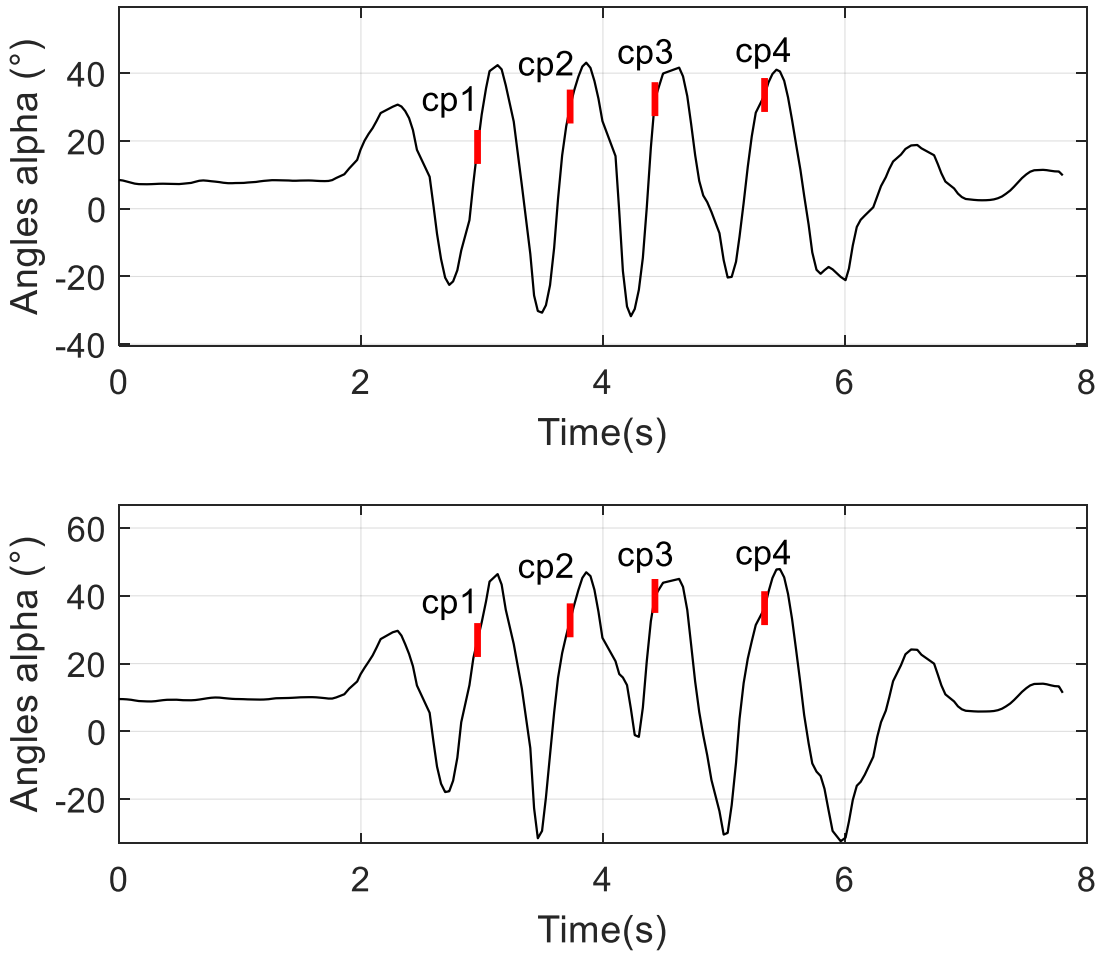

Figure 39. (a) Angles alpha of right arm. (b) Angles alpha of left arm. The traces in red represent the head peaks.

Note that in between two gallops there is at least one value of α above 0° and at least one value of α below 0° for both arms, which means that the participant swings his/her arms to produce force. Therefore, the flowchart that illustrates how to check if the arms swing is:

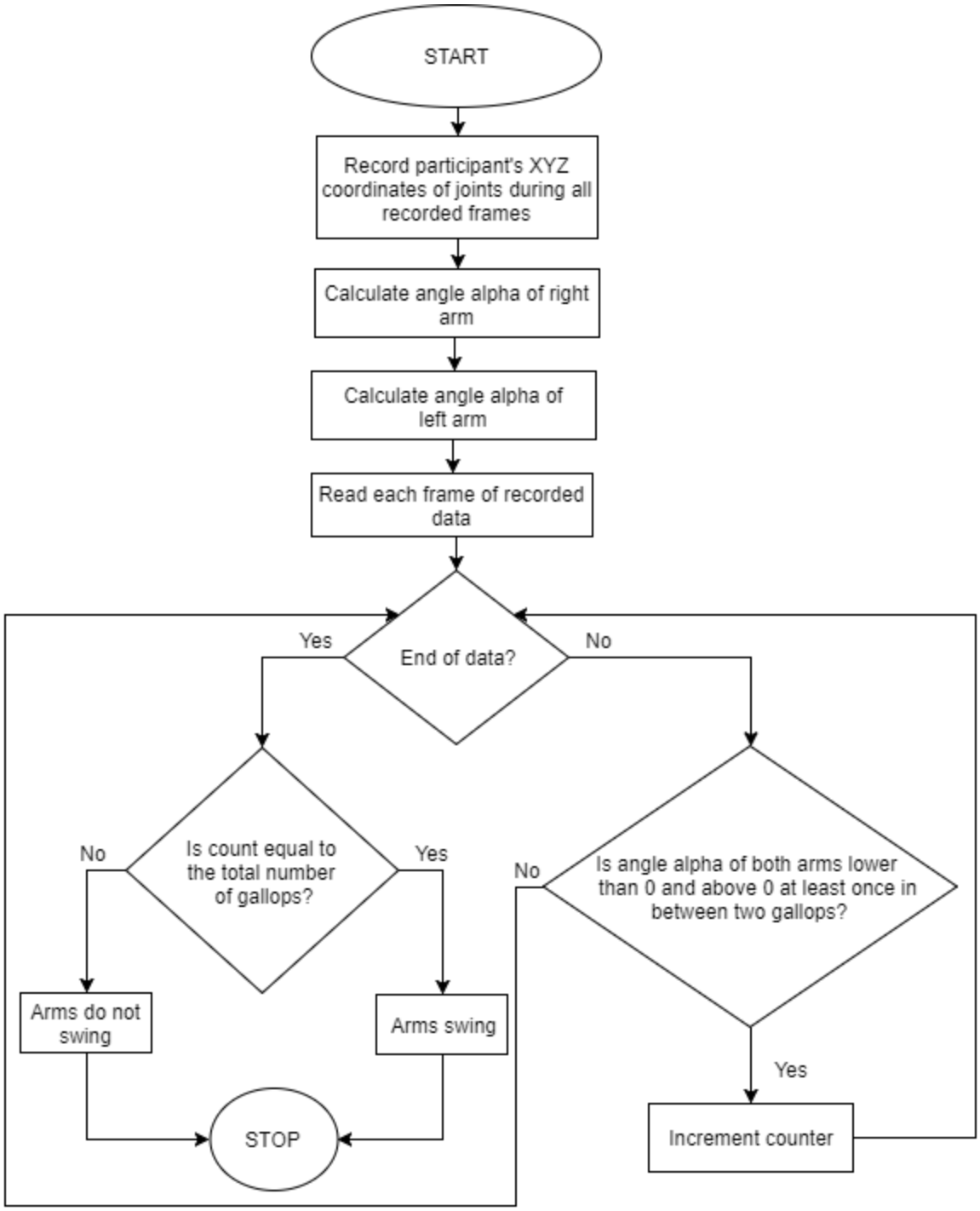

Figure 40. Algorithm to assess if participant swings arms forward during the movement.

The following flowchart illustrates the feedback for the first criterion:

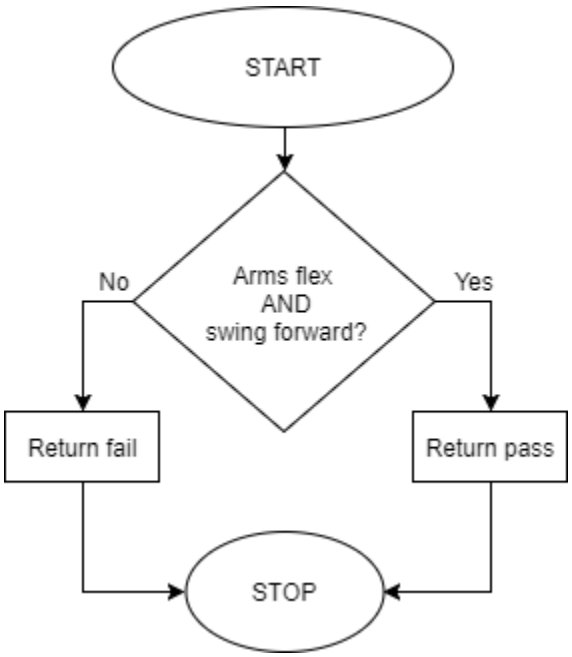

Figure 41. Algorithm to return feedback for first criterion.

<u>Criterion 2: A step forward with lead foot followed with the trailing foot landing beside or a little behind the lead foot (not in front of the lead foot).</u>

The first thing to do to analyze this criterion is to determine which foot the participant starts the movement with. In order to do so, the algorithm checks the difference in Z axis between the feet, considering that initially both feet will be at approximately the same distance from the sensor. Figure 42 shows the difference in the Z axis between the feet along time.

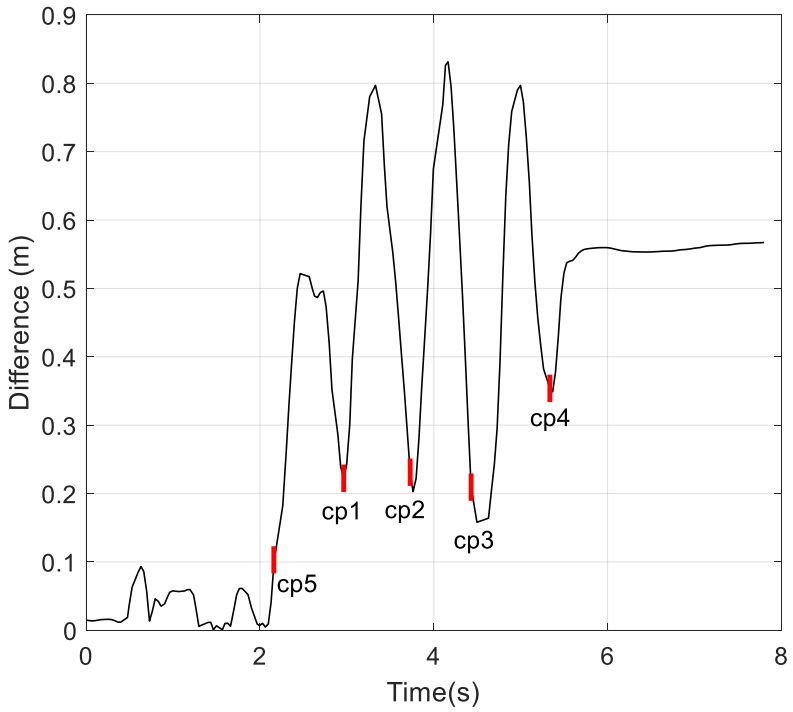

Figure 42. Difference between feet along Z axis. The red traces represent the critical points for the movement.

Note from the picture that initially both feet are the same distance from the Kinect, and as the participant starts the movement, the distance between the feet starts increasing. The algorithm considers the moment when they reach a distance superior or equal to 10 cm from each other to be the moment when the step begins (“cp5”).

The flowchart that illustrates how to check when movement starts is:

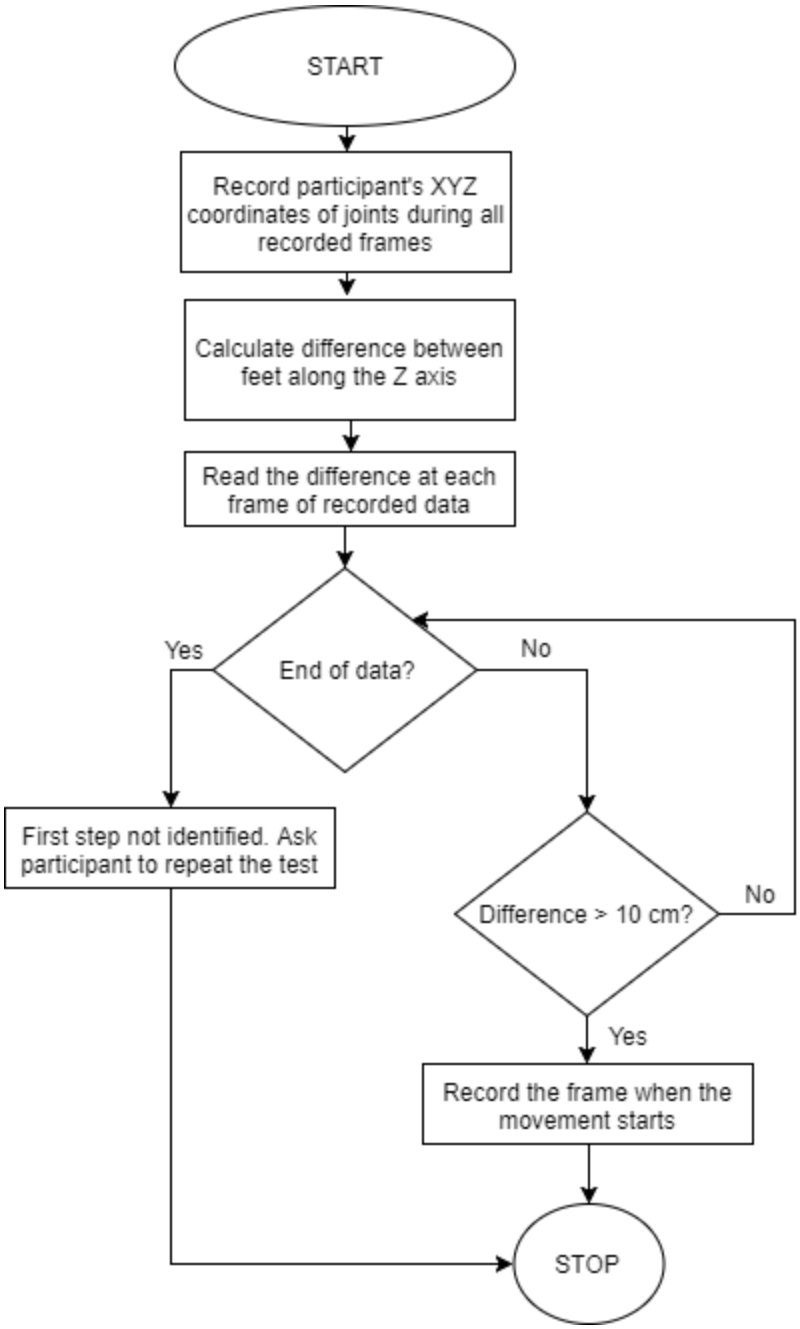

Figure 43. Algorithm to determine when movement starts.

After determining when the first step occurs, the algorithm still has to determine which foot performed the first step in order to determine the preferred leg. Figure 44 shows an example of the moment when the participant steps with the right foot.

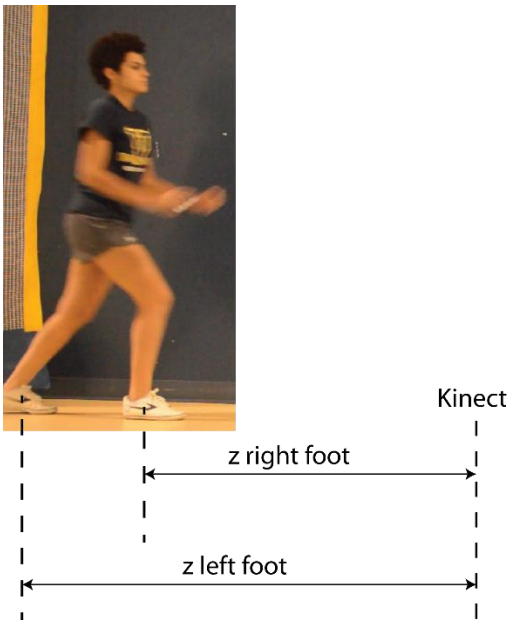

Figure 44. Example of a situation when the participant starts the movement with right foot.

Initially, the participant is standing and the distances along Z axis from each participant's foot to the Kinect sensor are approximately the same. As the participant steps towards the sensor, the Z coordinate of the stepping foot decreases in comparison to the non-stepping foot. Following this idea, the algorithm checks which foot is closer to the Kinect at "cp5", i.e., the moment when the distance between them is greater than 10 cm. Figure 45 illustrates the behavior of feet in Z axis along time.

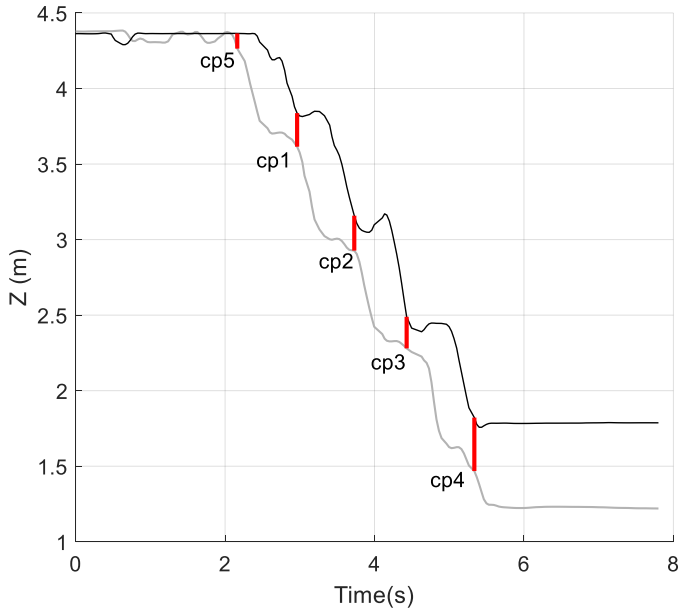

Figure 45. Variation along time of Z coordinates of feet. The right foot is represented by the grey curve and the left foot is represented by the black curve. The red traces represent the critical points for the movement.

A closer look at “cp5” is shown in Figure 46.

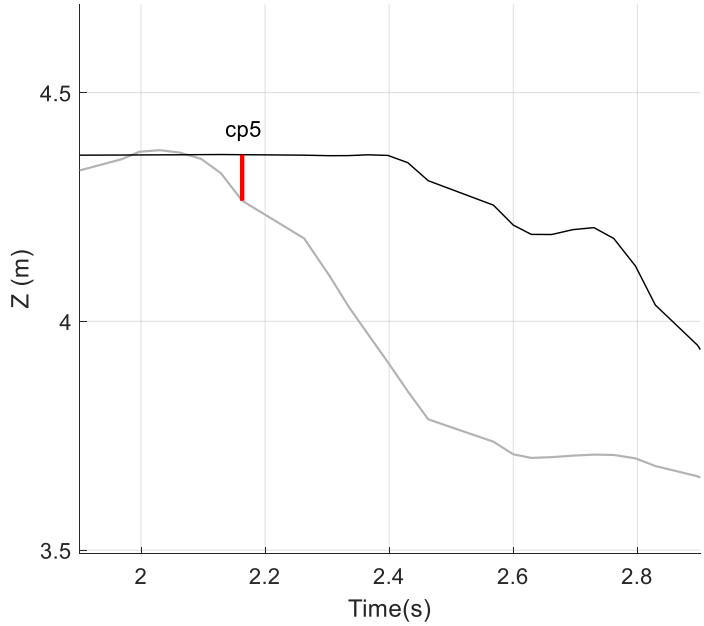

Figure 46. A closer look at "cp5", the moment when the right foot starts stepping forward. The right foot is represented by the grey curve and the left foot is represented by the black curve.

Note that at “cp5” the right foot is closer to the sensor than the left foot, which indicates that the participant’s lead foot is the right one. The flowchart that illustrates how to determine which foot starts the movement is:

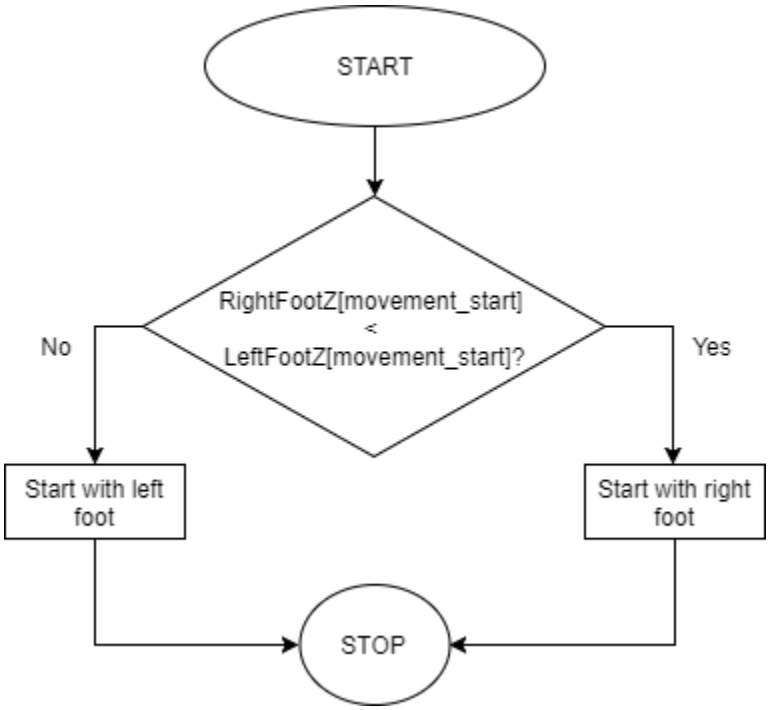

Figure 47. Algorithm to determine which foot starts the movement.

After knowing which foot is the lead foot, the algorithm has to check if a step forward happens for each gallop cycle. As shown in Figure 34, the participant starts the movement with the feet close to each other, then s/he performs the first step, and a pattern follows for the rest of the movement: At each head peak the feet are close together and as soon as the participant lands on the floor, a new step with the lead feet initiates; during the aerial phase the feet get closer to each other again. When a step occurs, the distance between the feet increases, and at the same time the foot opposite to the leading foot stays placed on the floor, without moving, for a brief period of time. Figure 48 illustrates this situation using as an example the first gallop.

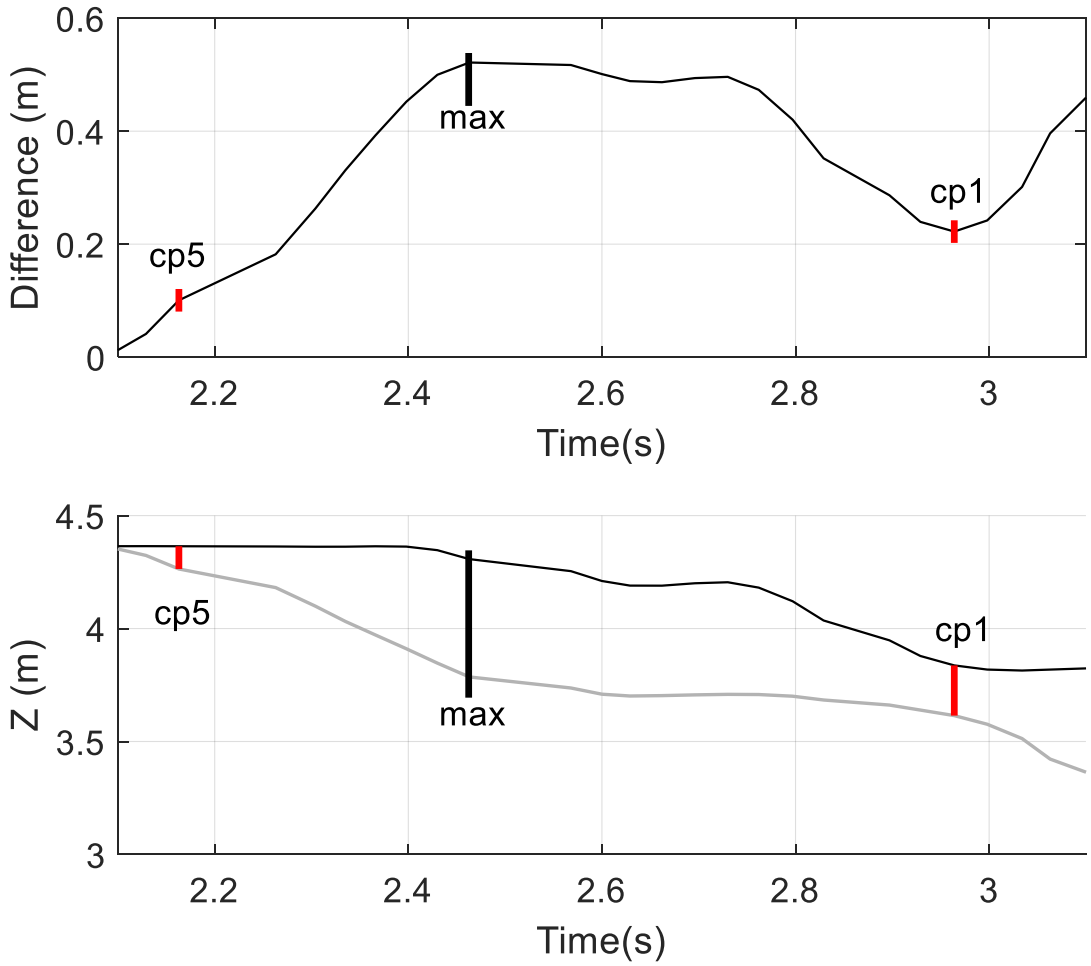

Figure 48. A closer look at the first step that happens between “cp5” and “cp1”. (a) Variation along time of distance between feet. (b) Variation along time of Z coordinates of feet; the right foot is represented by the grey curve and the left foot is represented by the black curve.

Note from the picture that a step is initiated at “cp5” by the right foot, and as the right foot moves away from the left foot until it reaches the maximum distance, the left foot stays in place on the floor without moving. When the distances reach the maximum value, it means that one step is completed and then the participant performs the impulse from the floor to leave the ground, reaching the first head peak at “cp1”. The same pattern repeats for the other gallops. Figure 49 shows that in between two gallops, the distance between the feet reaches a maximum value within that period, which indicates when each step is completed.

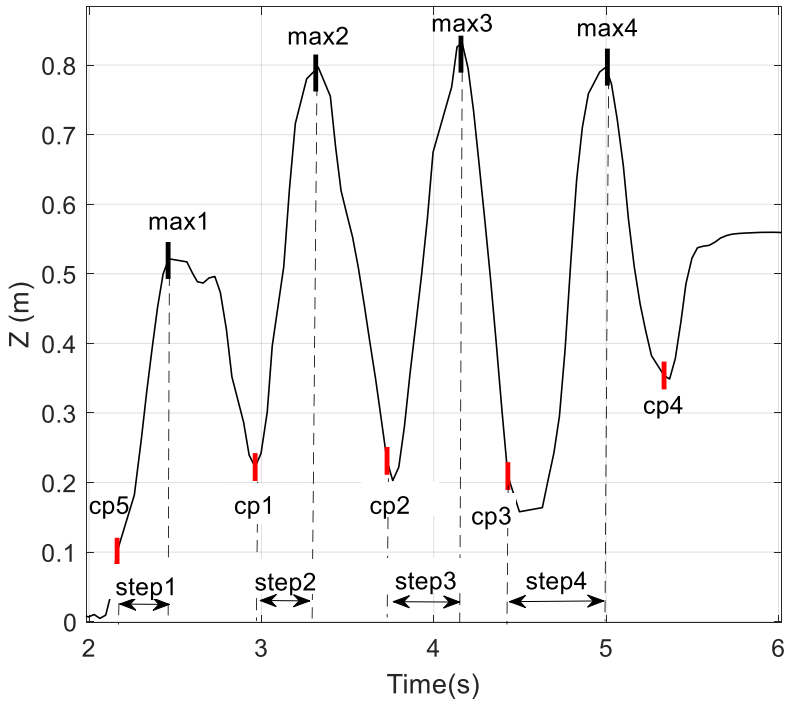

Figure 49. Difference between feet along Z axis and moment when each step occurs based on the maximum difference in between two consecutive critical points. Each step happens in between a critical point and a local maximum.

Basically, each step begins at a critical point and finishes when the distance between feet reaches a maximum value in between two critical points, as shown in Figure 50.

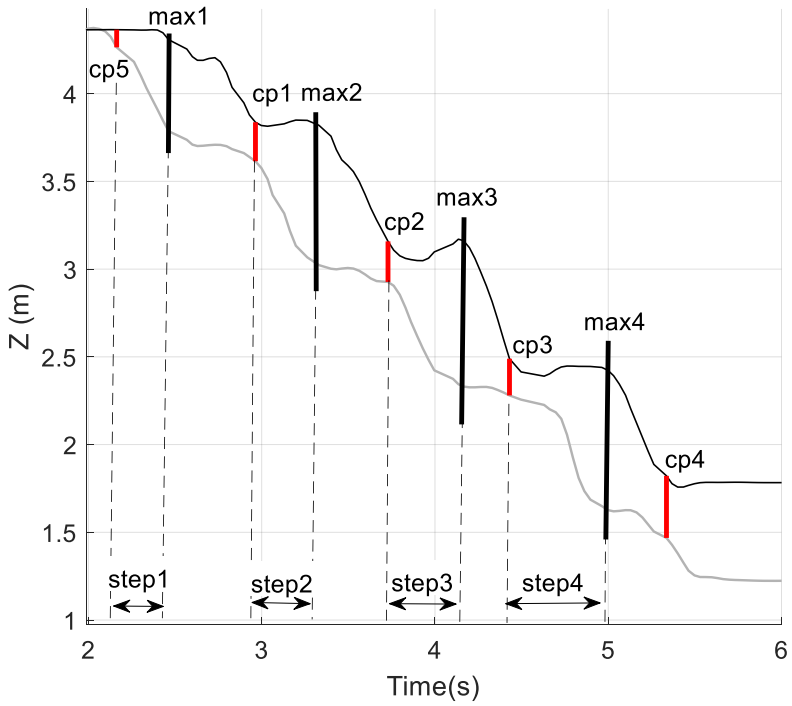

Figure 50. Variation along time of Z coordinates of feet and moment when each step occurs based on the maximum difference in between two consecutive critical points. Each step happens in between a critical point and a local maximum. The right foot is represented by the grey curve and the left foot is represented by the black curve.

Note in Figure 50 that the foot that is performing the step — in this case the right foot — decreases its coordinate substantially along Z as it moves towards the Kinect, while the non-stepping foot remains approximately at the same value of Z, which indicates that the non-stepping foot stays placed on the floor and does not move during the performance of the step. Therefore, in order to check if the participant performs a step forward with the leading foot, the algorithm checks the Z coordinate of the non-stepping foot and the stepping foot first at the beginning and then at the end of the step. If the difference between initial and final values of the non-stepping foot is lower than the difference between initial and final values of the stepping foot for all steps, the algorithm considers that the participant performed a step forward with the leading foot. In essence, it considers the slope of the curve. The final value also needs to be lower than the initial value, since

the movement is performed towards the Kinect.

In addition, the algorithm also needs to check if the trailing foot lands beside or a little behind the leader. Note in Figure 45 that the grey curve, represented by the leading foot, remains closer to the Kinect since the beginning of the movement ("cp5") until the end of it ("cp4"). Also, the black curve, represented by the trailing foot, never crosses the grey curve, which indicates that the trailing foot never crosses in front of the leading foot. Also, by observing Figure 34, it is clear that during a standardized performance of the gallop test, the trailing foot never crosses the leading foot. Following this idea, the algorithm checks each frame to see if the Z coordinate of the leading foot is lower than the Z coordinate of the trailing foot; if that condition is true for all frames, it means that during the whole movement the trailing foot remained behind the leading foot and the participant passes the criterion.

The flowchart that illustrates how to check the maximum value of the difference between feet in between two consecutive critical points is:

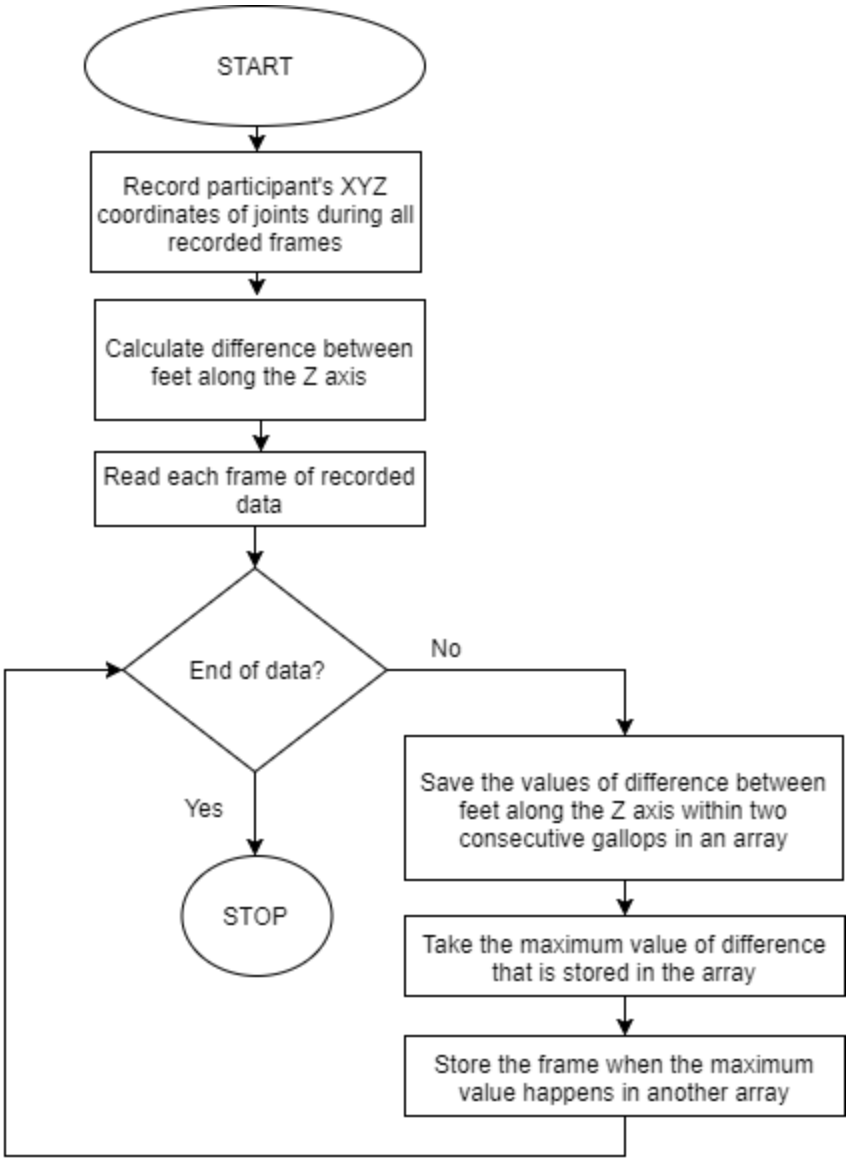

Figure 51. Algorithm to determine the maximum distance between feet within two consecutive gallops.

The flowchart that illustrates how to check for step forward in between each critical point is:

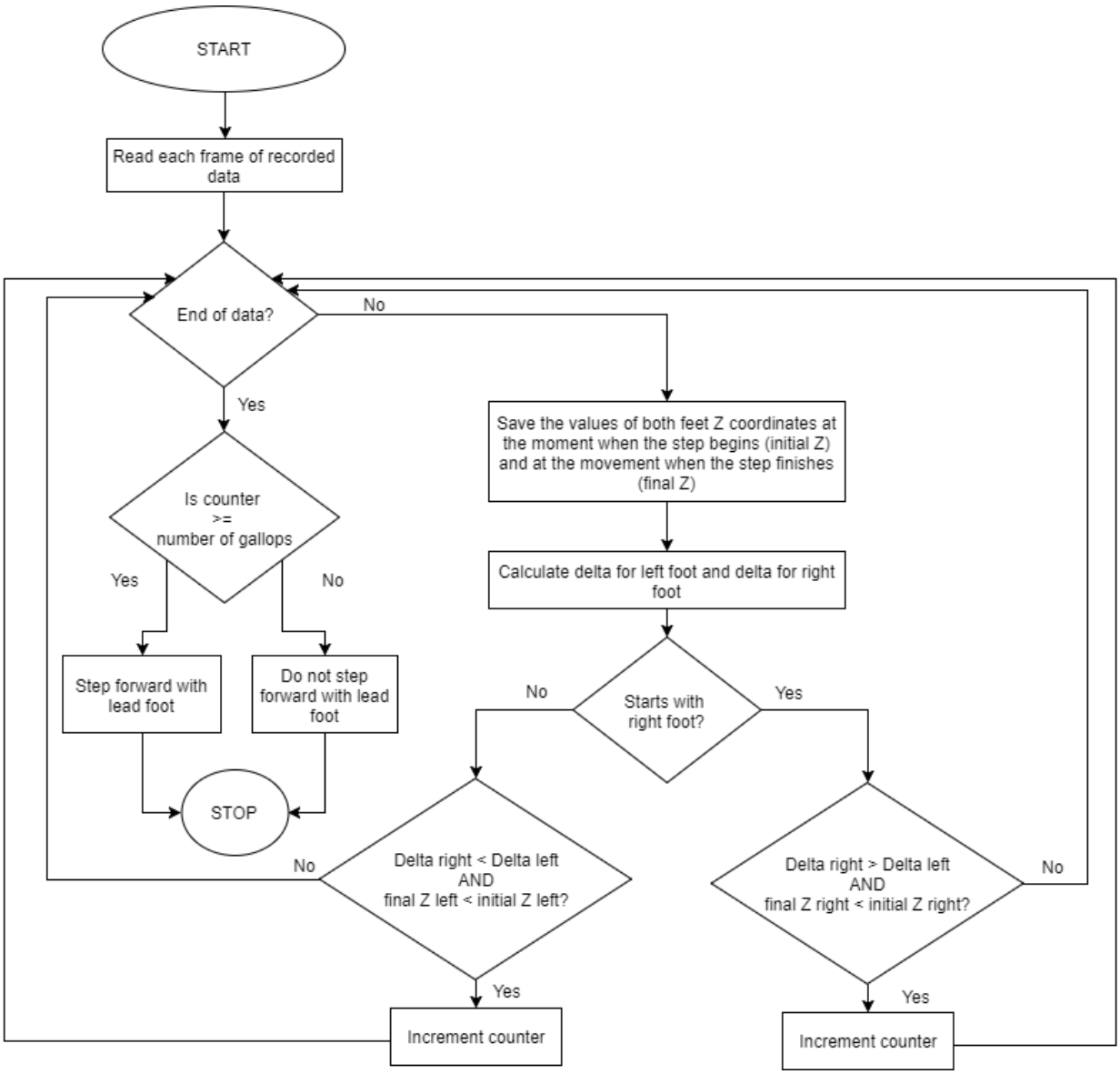

Figure 52. Algorithm to determine if participant steps forward with lead foot.

The flowchart that illustrates how to check if non-stepping foot crosses step foot is:

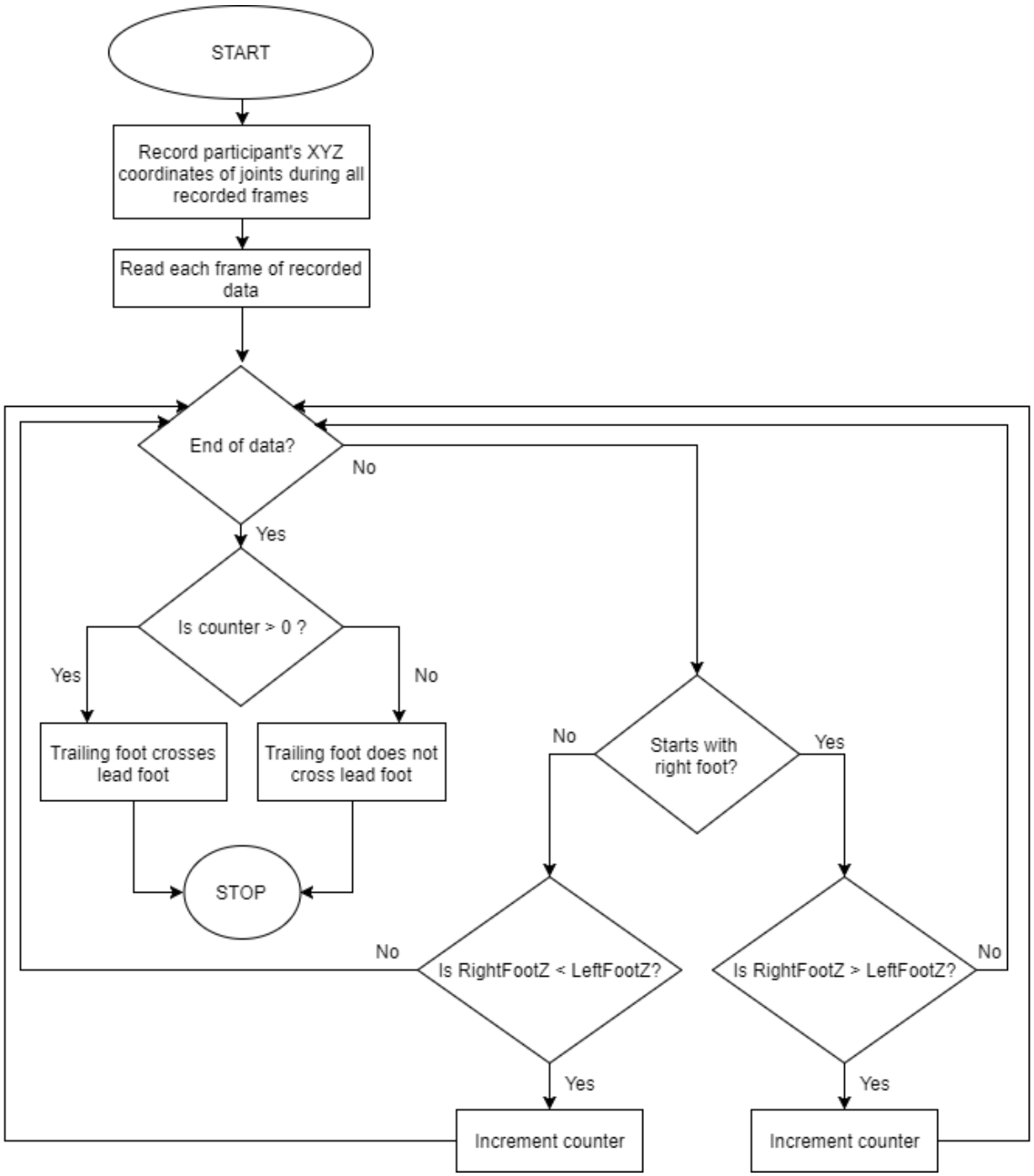

Figure 53. Algorithm to determine if participant's trailing foot crosses lead foot.

The following flowchart illustrates the feedback for the second criterion:

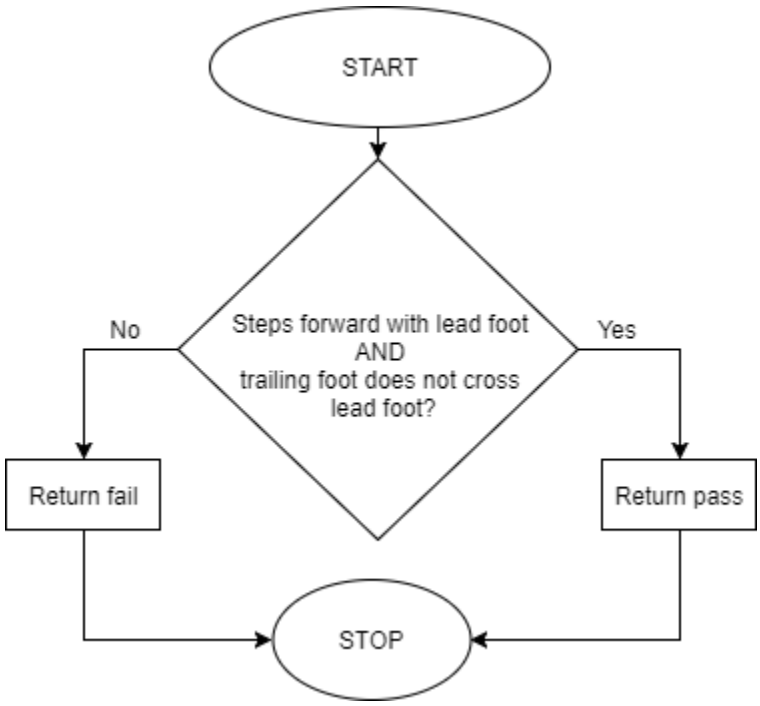

Figure 54. Algorithm to return feedback for second criterion.

<u>Criterion 3: Brief period where both feet come off the surface.</u>

As mentioned before, the gallop movement consists of four phases very similar to each other, since the user is asked to maintain a rhythmic pattern for four consecutive gallops. In order to distinguish one phase from another, the algorithm uses the Y coordinate of the head because it varies rhythmically along time, as shown in Figure 35. Just like it was done for the running test, the algorithm that identifies the peaks is used to save the index of all frames where the critical points occur. At those frames, the participant is off the surface, seeing that the Y coordinate of the head at a peak is higher than the participant's height, which was determined during the calibration phase. Then, for each of those frames, the algorithm needs to check if both Y coordinates of the knees are above standing level. Again, the joints of the knees are used instead of the feet to analyze the criterion, since this is more accurate.

In order to check if the knees are above standing level, the algorithm first needs to know the distance the knees are from the floor when the participant is standing on the floor level. In order to do so, the algorithm uses the values of Y coordinate of the right knee and the left knee obtained during the calibration. These values represent the knee's distance from the floor when the participant is standing. Therefore, the participant's standing level is obtained by taking the average between right knee's distance from the floor and left knee's distance from the floor obtained from the calibration.

Once the standing level is obtained, the algorithm checks if at each critical point the Y coordinate of each knee is at least 10% greater than the standing level. This 10% above the standing level is the threshold used to consider whether the knees are off the standing level. Figure 55 shows the Y coordinate of both knees and head as well as the standing level and the threshold. It is clear in the figure that both knees are off surface when the peaks occur.

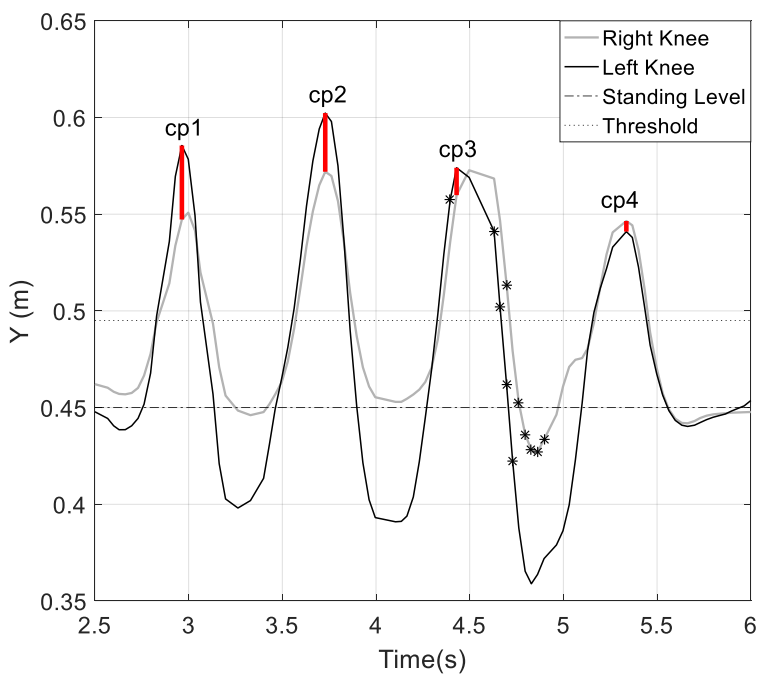

Figure 55: Variation along time of Y coordinates of knees. The (*) represents the moment when the 3D position of the joint was inferred.

The flowchart that illustrates how to check if both knees are off surface at each critical point and returns the feedback for the criterion is:

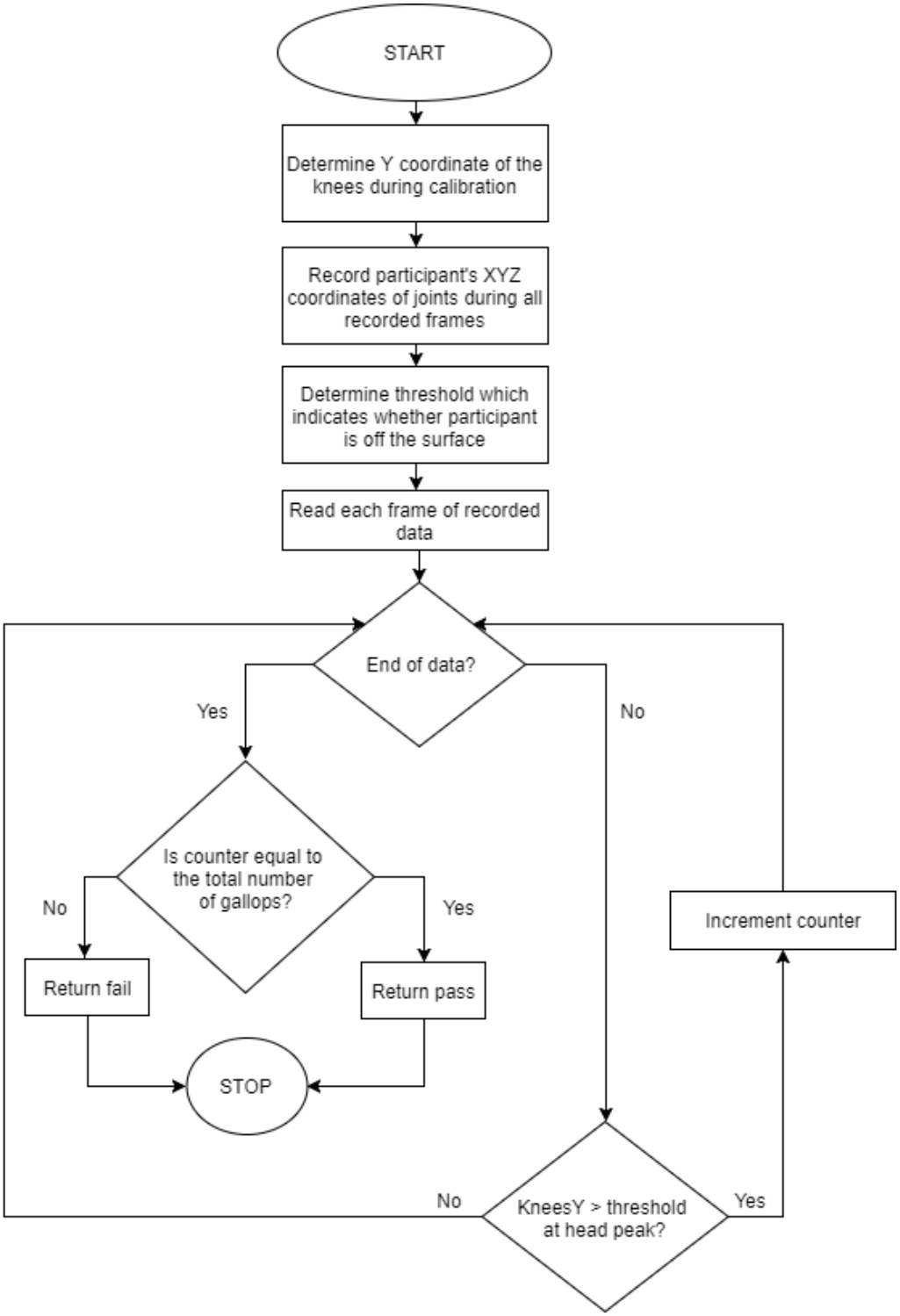

Figure 56. Algorithm to return feedback for third criterion.

Criterion 4: Maintains a rhythmic pattern for 4 consecutive gallops.

The last criterion consists of analyzing if the participant maintains a rhythmic pattern for 4 consecutive gallops. In order to check that, the algorithm uses the number of peaks present on the Y coordinate of the head to count how many gallops are performed. If the number of peaks is equal to four, and if the interval of time between them is approximately the same, it means the participant performed four consecutive gallops while maintaining a rhythmic pattern.

Note in Figure 57 that the participant performs 4 gallops and the time lapse between each gallop remains approximately the same. In this example, the maximum difference between them occurs from “cp3” to “cp4” and does not exceed 23%. The algorithm gives a tolerance of 30% that allows one peak to be apart from the next peak to still consider the movement rhythmical. It is important to mention that the time period can vary and a 30% difference was found to be reasonable.

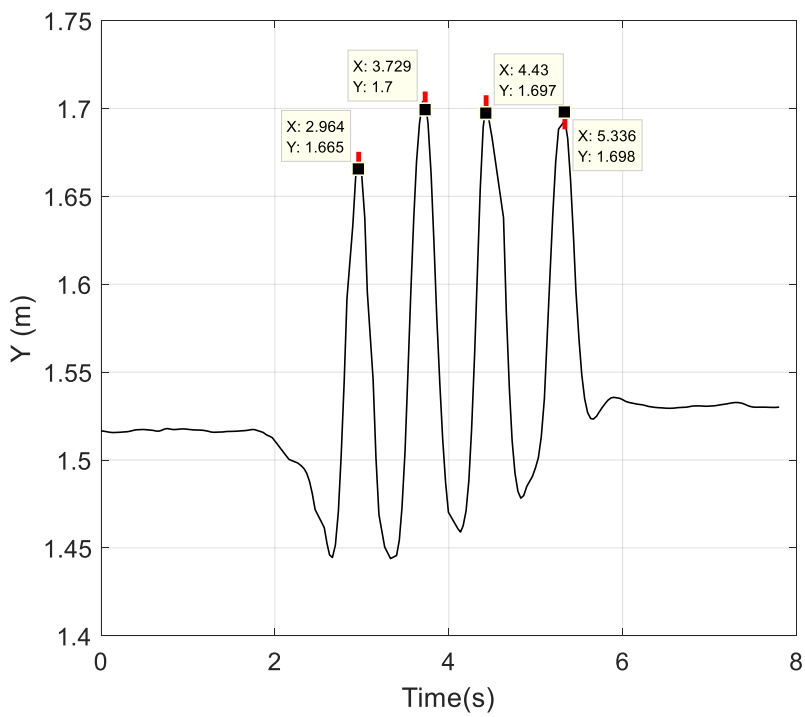

Figure 57: Moment when each head peak, i.e. each critical point, occurs.

The flowchart that illustrates how to analyze the last criterion is:

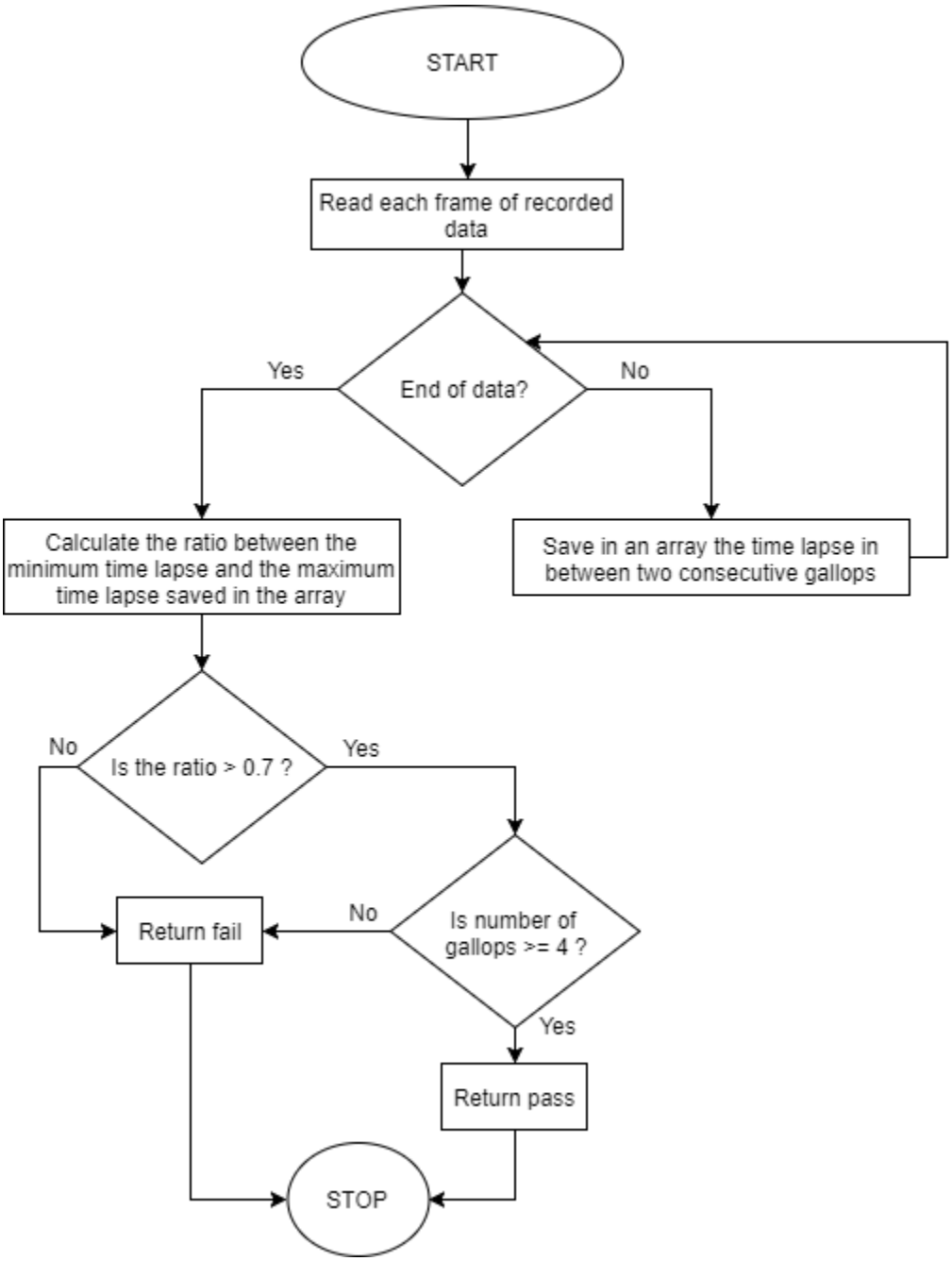

Figure 58. Algorithm to return feedback for fourth criterion.

4.3.3 Hop

According to the TGMD-3 scoring sheet, the participant is asked to perform four hops conforming to the specified performance criteria. An example of a hop movement is shown in Figure 59 below, where the Kinect is positioned facing the participant.

Figure 59. Sequence of images to represent a standardized hop movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

Note in Figure 59 that the Y coordinate of the non-hopping knee, i.e. the distance from the floor to it, draws a peak along time. Therefore, each hop phase is characterized by the period between two non-hopping knee peaks. In general, each hop phase can be determined by the frames in between two non-hopping knee peaks, so each peak detected is a critical point for the analysis. Figure 60 shows the behavior of Y coordinate of the knees along time when the participant performs four consecutive hops. Different from the gallop and running tests, where the Y coordinates of the head are used to identify the peaks and critical points, the hop test uses the Y coordinates of the non-hopping knee instead, since this joint exhibits a greater variation along compared to that of the head.

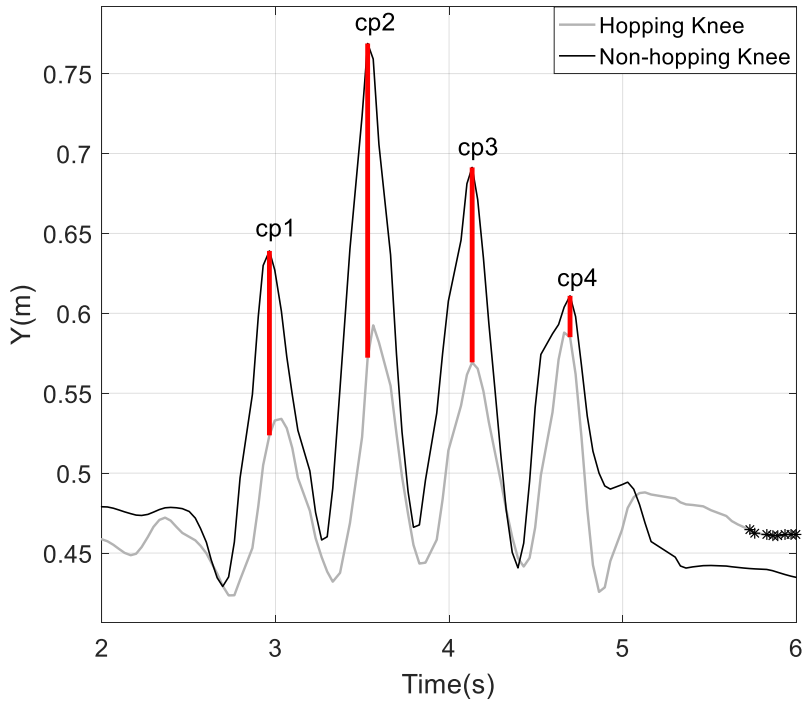

Figure 60. Variation along time of Y coordinates of knees and the critical points.

<u>Criterion 1: Non-hopping leg swings forward in pendular fashion to produce force.</u>

The analysis of the first criterion of the hop tests is similar to the first criterion of the gallop test. First, the algorithm needs to identify which leg is the hopping one. Figure 60 shows an example of a situation when the participant hops with the right leg. Note that the knee of the non-hoping leg, in this case the left one, reaches higher values along the Y axis, since the non-hoping leg needs to swing to produce force. Figure 61 also illustrates this.

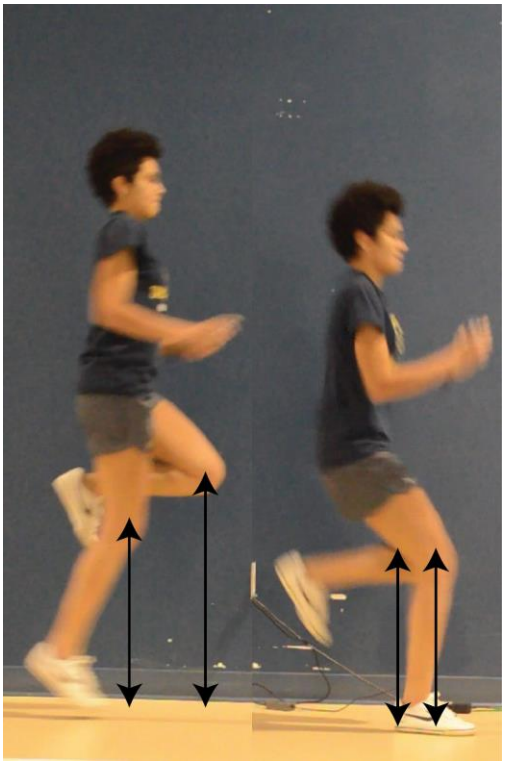

Figure 61. Example of knees displacement along Y axis when the participant hops with right leg.

Note that the non-hopping knee draws a greater displacement along Y axis than the hopping knee. Therefore, based on the maximum values reached by the knees in the Y axis, the algorithm knows which leg is the hopping one.

The following flowchart illustrates how to identify the hoping leg:

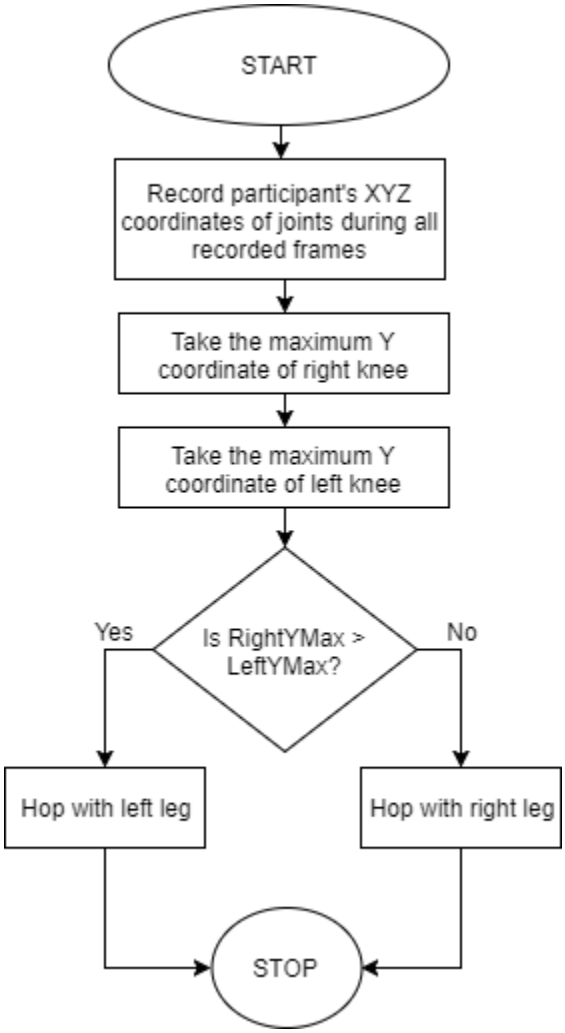

Figure 62. Algorithm to determine the hopping leg.

Note in Figure 59 that as the movement progresses, the participant has to swing the non-hoping leg to produce force. Figure 63 shows an example of a swing forward and a swing backward using the leg.

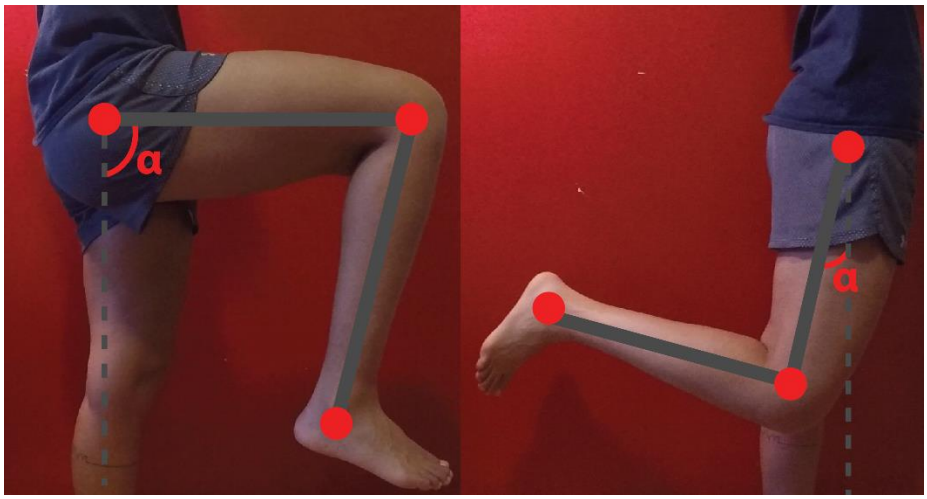

Figure 63. Illustration of how the angle alpha varies during a swing using legs.

A swing can be identified by an angle α formed between the hip and the line that represents the trunk. The angle α can be calculated as follows:

$$\alpha = \tan^{-1}\left(\frac{Hip\ Z - KneeZ}{HipY - KneeY}\right) \tag{11}$$

Figure 64 illustrates the variation of the angle α during the performance of the movement.

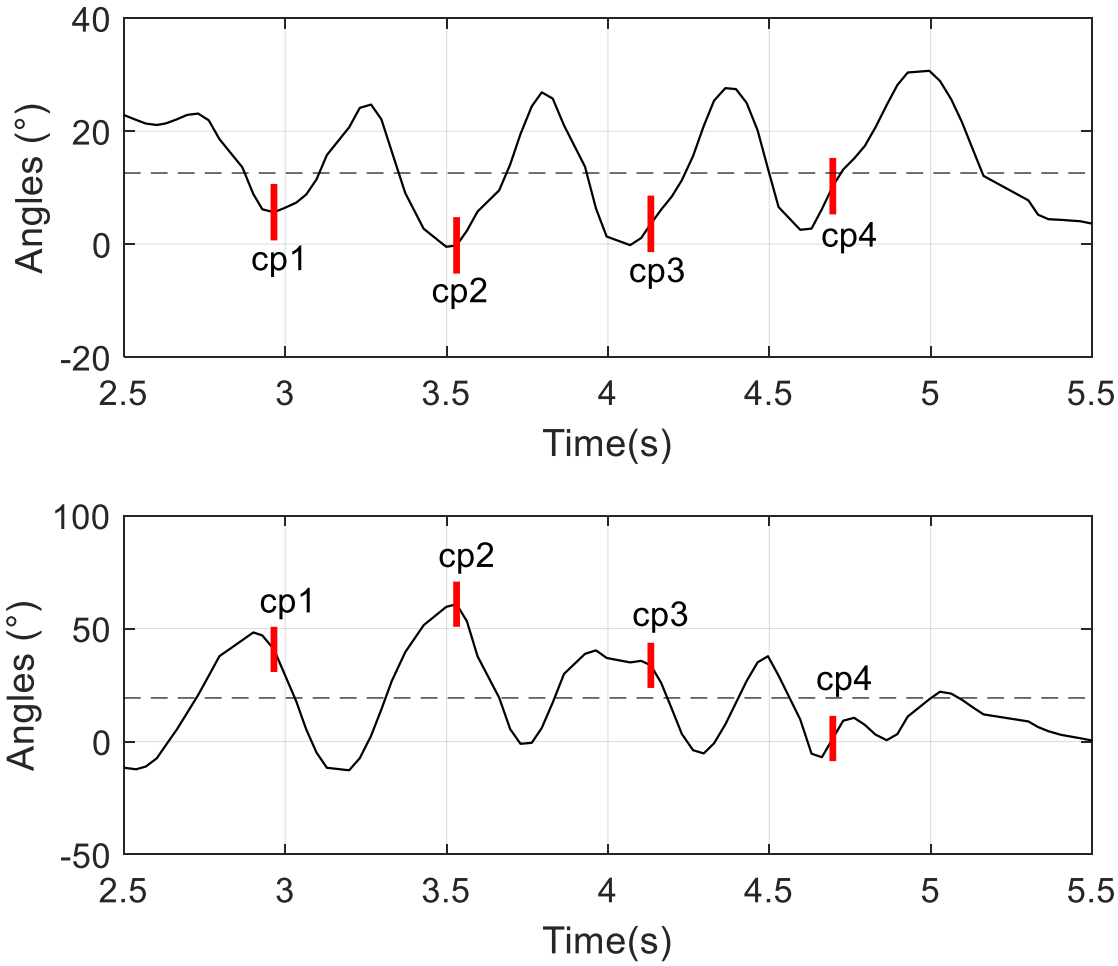

Figure 64. (a) The solid line represents the angles alpha of right leg and dashed line represents the average of the angles. (b) The solid line represents the angles alpha of left leg and dashed line represents the average of the angles.

Note in the picture that depending on whether the swing is forward or backward, the angle α varies between a greater or a lower value. Therefore, in order to check if the participant swings the non-hoping leg forward, the algorithm checks to see if there is at least one value of α above the average of the values of α and at least one value below the average in between two consecutive hop cycles for the non-hoping leg. In this way, the algorithm knows that the participant has performed a swing for each hop.

The flowchart that illustrates how to check if non-hoping leg swings is:

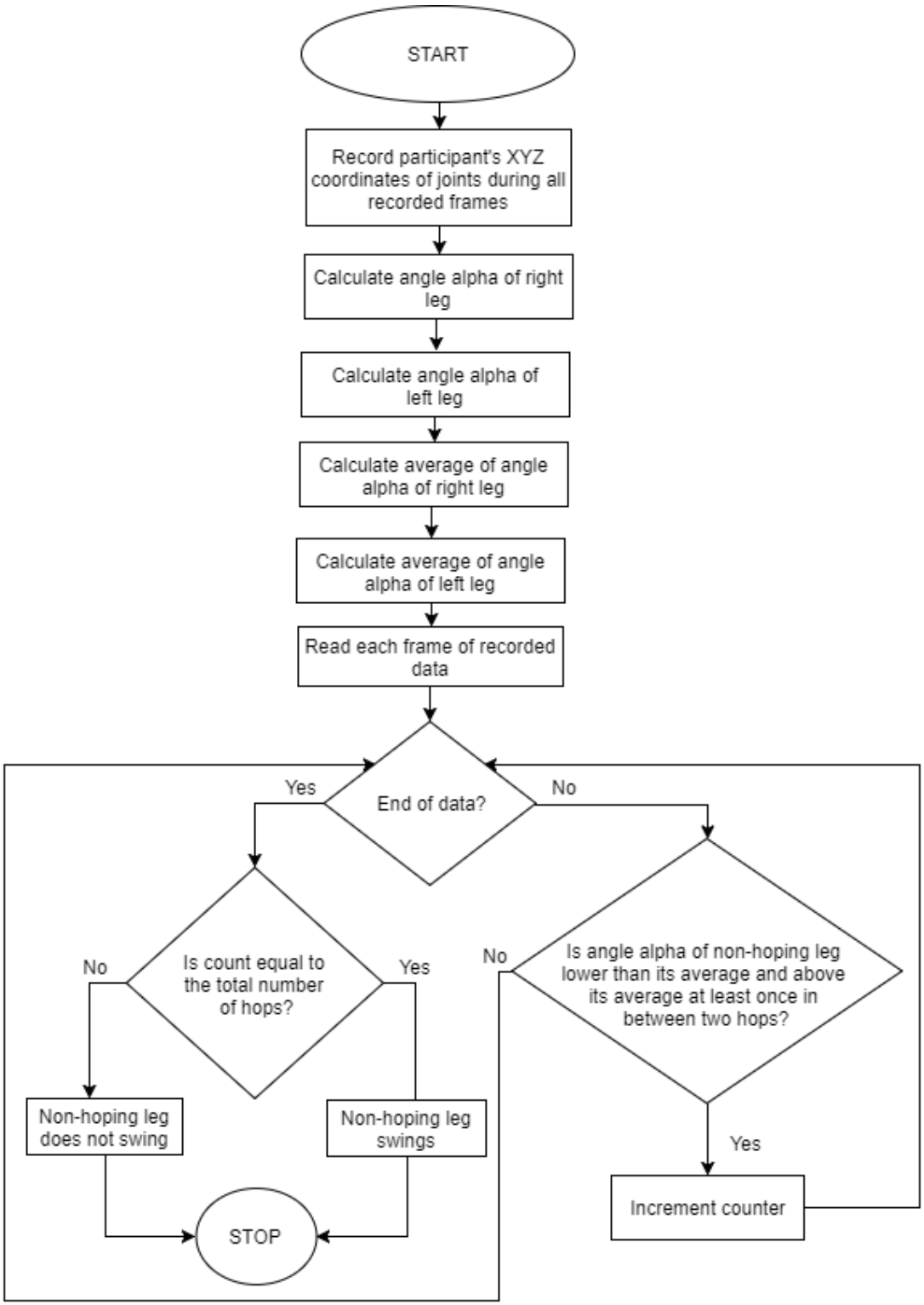

Figure 65. Algorithm to determine whether legs swing to produce force.

In addition, to check if the movement of the non-hoping leg is pendular, the algorithm compares the Y coordinate of the non-hoping foot to the Y coordinate of the hoping foot. If the non-hoping leg swings and its foot never touches the floor during the hops, the movement is considered to be pendular. Therefore, for all frames recorded during the hops, the Y coordinate of the non-hoping foot is expected to be greater than the Y coordinate of hoping foot.

Note in Figure 66 that the left foot, the non-hoping one in this example, does not touch the floor during the hops, being above the hoping foot within that period.

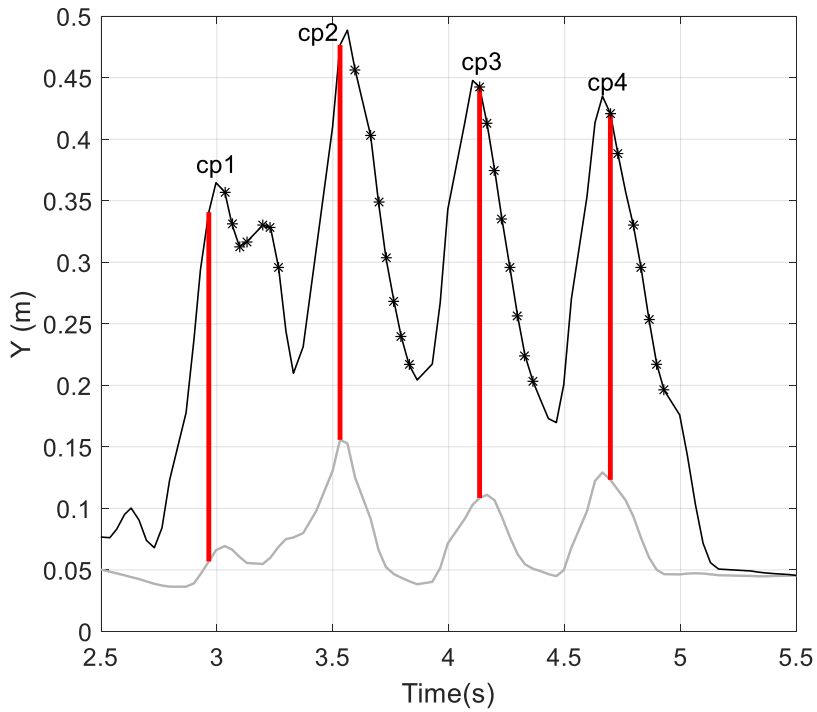

Figure 66. Variation along time of Y coordinates of feet. The black curve represents the left foot (non-hoping foot), and the grey curve represents the right foot (hoping foot). The (*) represents the moment when the 3D position of the joint was inferred.

The flowchart that illustrates how to check for pendular fashion is:

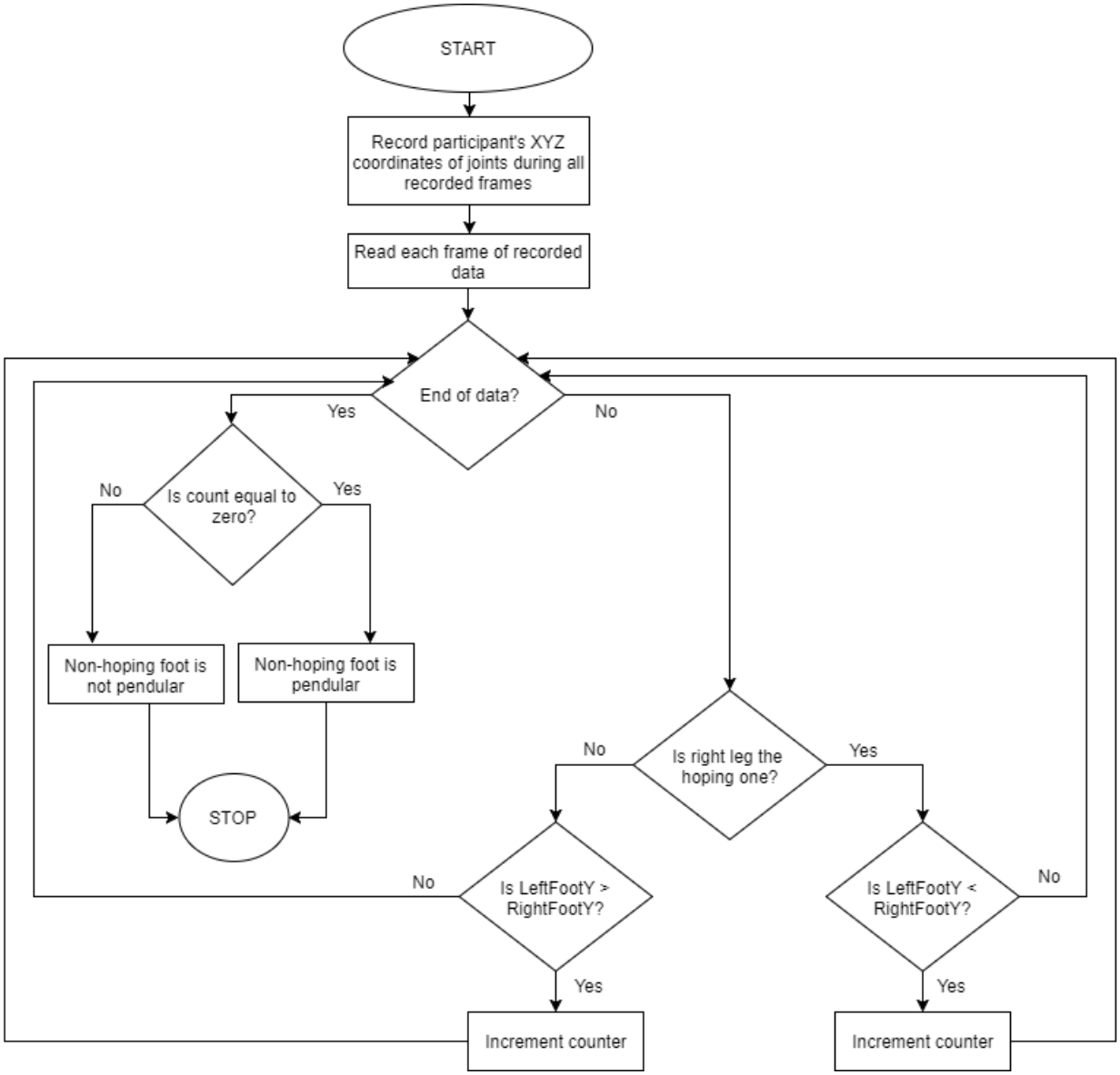

Figure 67. Algorithm to determine whether non-hopping leg swings in pendular fashion.

The following flowchart illustrates the feedback for the first criterion:

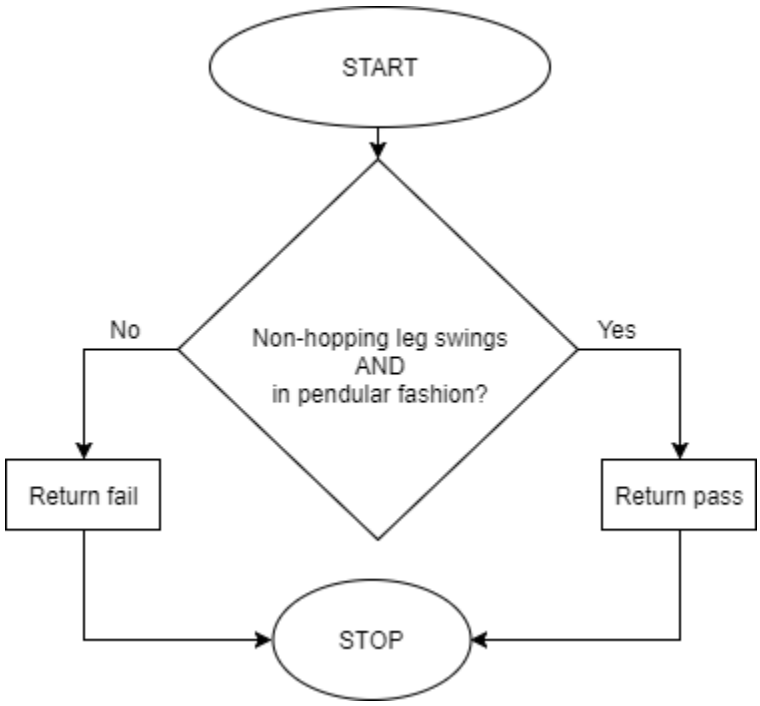

Figure 68. Algorithm to return feedback for the first criterion.

<u>Criterion 2: Foot of non-hopping leg remains behind hopping leg (does not cross in front of).</u>

The second criterion checks if the foot of non-hopping leg remains behind the hopping leg, i.e. does not cross in front it. For this assessment, a comparison between the Z coordinate of the non-hopping foot and the Z coordinate of the hopping knee was used to determine if the non-hopping foot crosses the hopping leg or not. As the movement is performed facing the Kinect, if the non-hopping foot crosses the hopping leg, the Z coordinate of the non-hopping foot will be lower than the Z coordinate of the hopping knee at that moment, since it will be closer to the sensor.

Figure 69 shows the behavior of the non-hoping foot during the standardized movement.

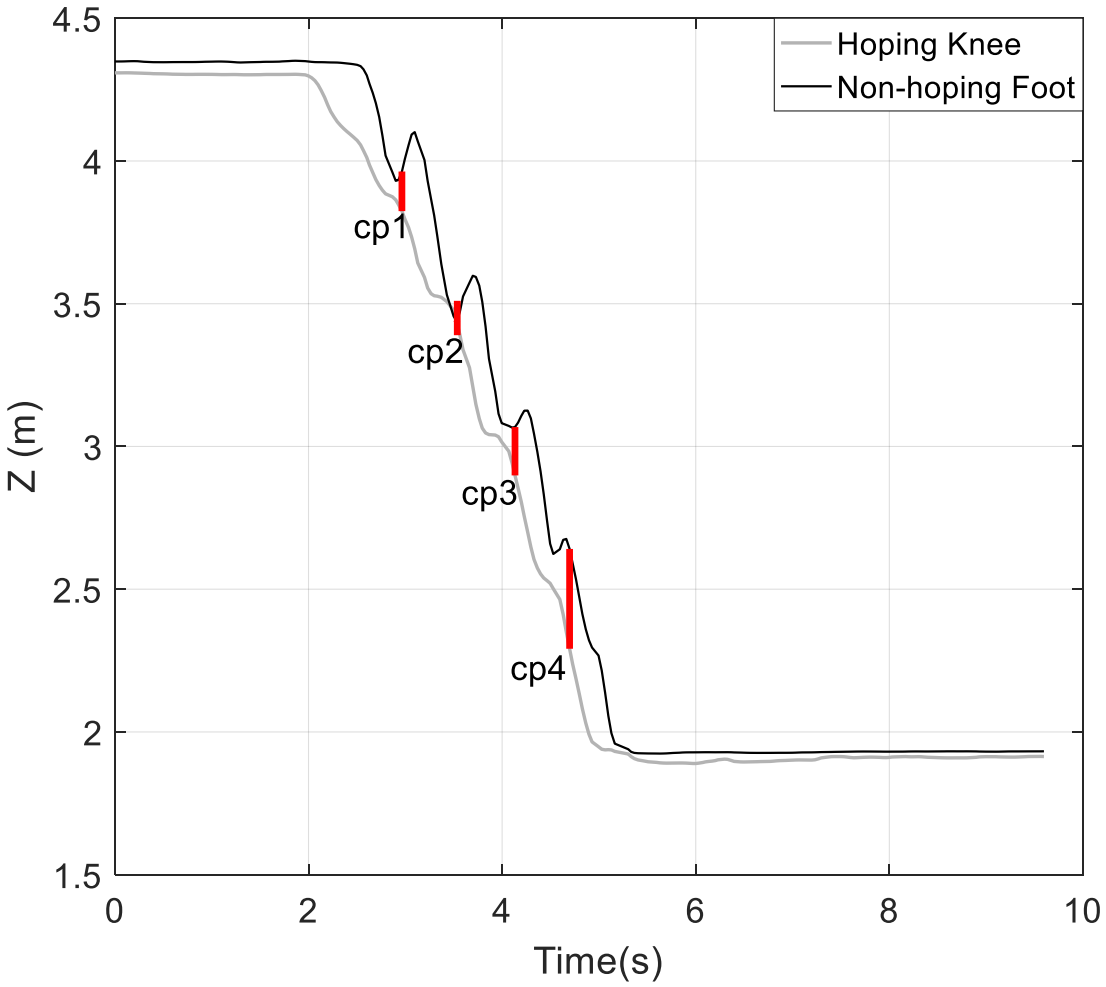

Figure 69. Variation along time of the Z coordinate of hoping knee and non-hoping foot.

The standardized movement is performed by meeting all the criteria for the skill, i.e. the participant does not cross the non-hopping foot in front of hopping leg. Note in the figure that as the non-hoping foot does not cross the hoping leg, the Z coordinate of the non-hoping foot remains greater, i.e. further away from the Kinect, than the Z coordinate of the hoping knee during the whole movement.

Based on that comparison between the Z coordinates of the non-hopping foot and the Z coordinate of the hopping knee, the software checks for all frames if a crossing occurs. If it does not cross more than 10% of all frames, the participant passes the criterion.

The flowchart that illustrates how to determine if a crossing occurs is:

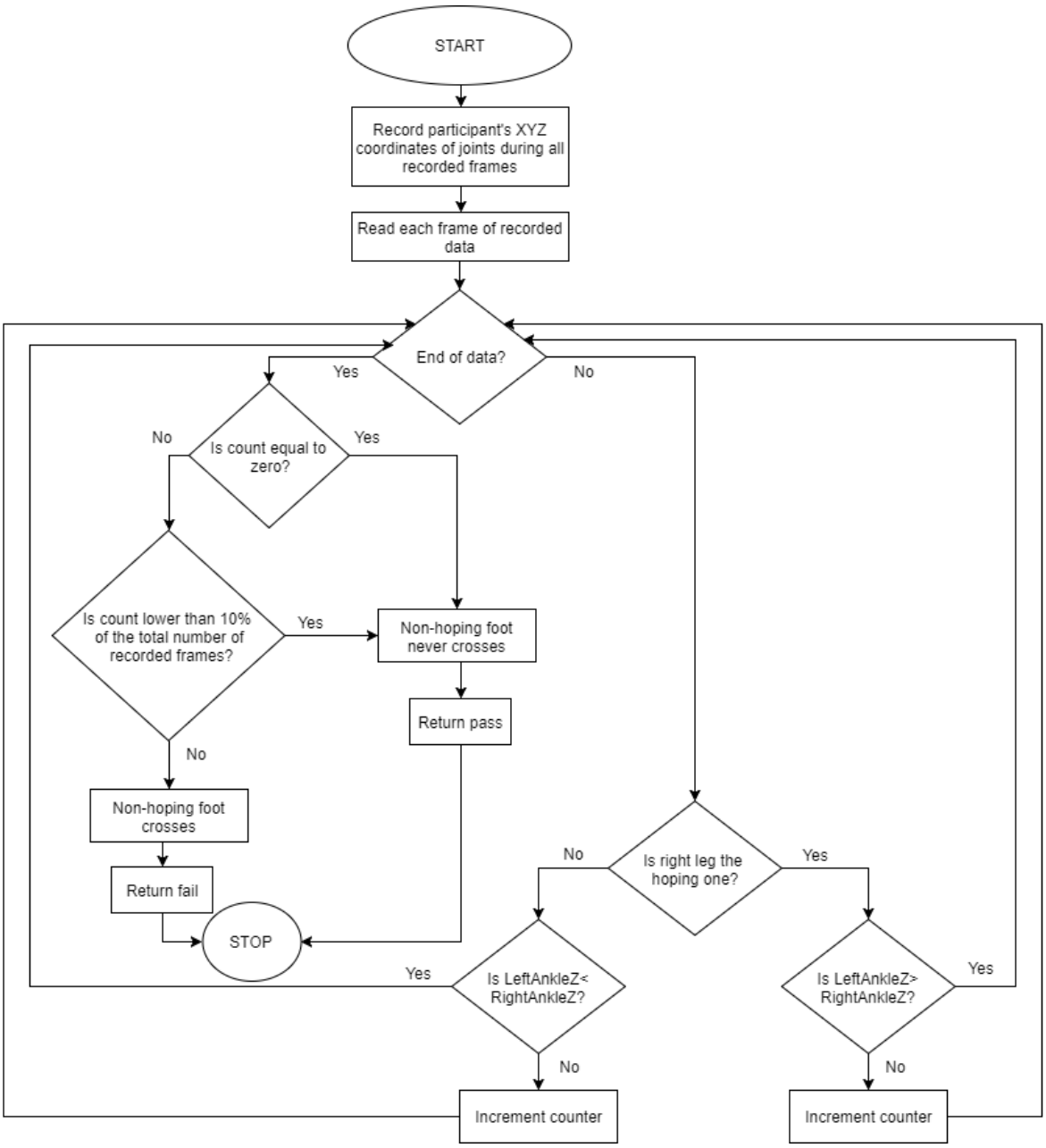

Figure 70. Algorithm to determine whether non-hopping foot crosses hopping leg.

Criterion 3: Arms flex and swing forward to produce force.

The third criterion checks if the arms flex and swing forward to produce force. This criterion is identical to the first criterion of the gallop test; therefore, they were assessed exactly the same way. Using the ratio equation to determine the ratio for when the arms flex equal or less than 120°, any value below 0.75 means that the participant has his/her arms flexed; otherwise they are not flexed. Figure 71 illustrates the ratio of the arms during the performance of the gallop test. Note that from one critical point to the next one, both arms reach values of ratios below 0.75, which means that the participant had his/her arms flexed for each gallop.

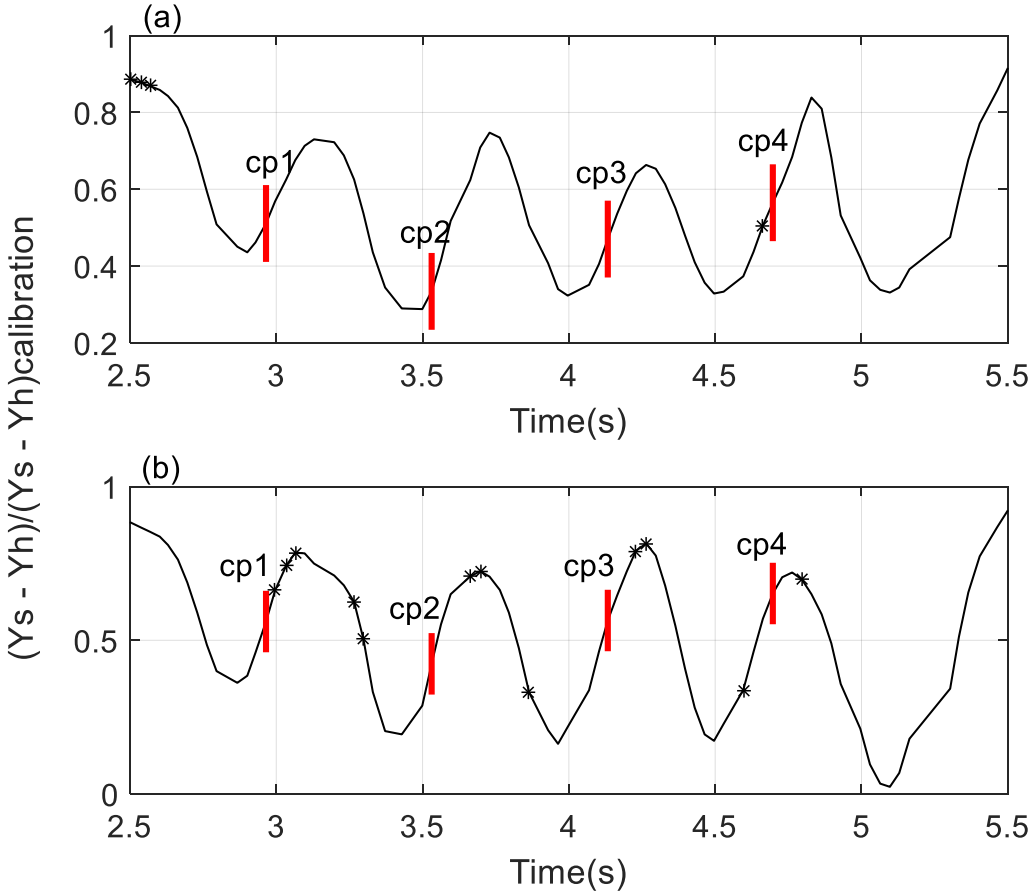

Figure 71. (a) Ratios of right arm. (b) Ratios of left arm. The (*) represents the moment when the 3D position of the joint was inferred.

The flowchart that illustrates how to check if the arms flex is:

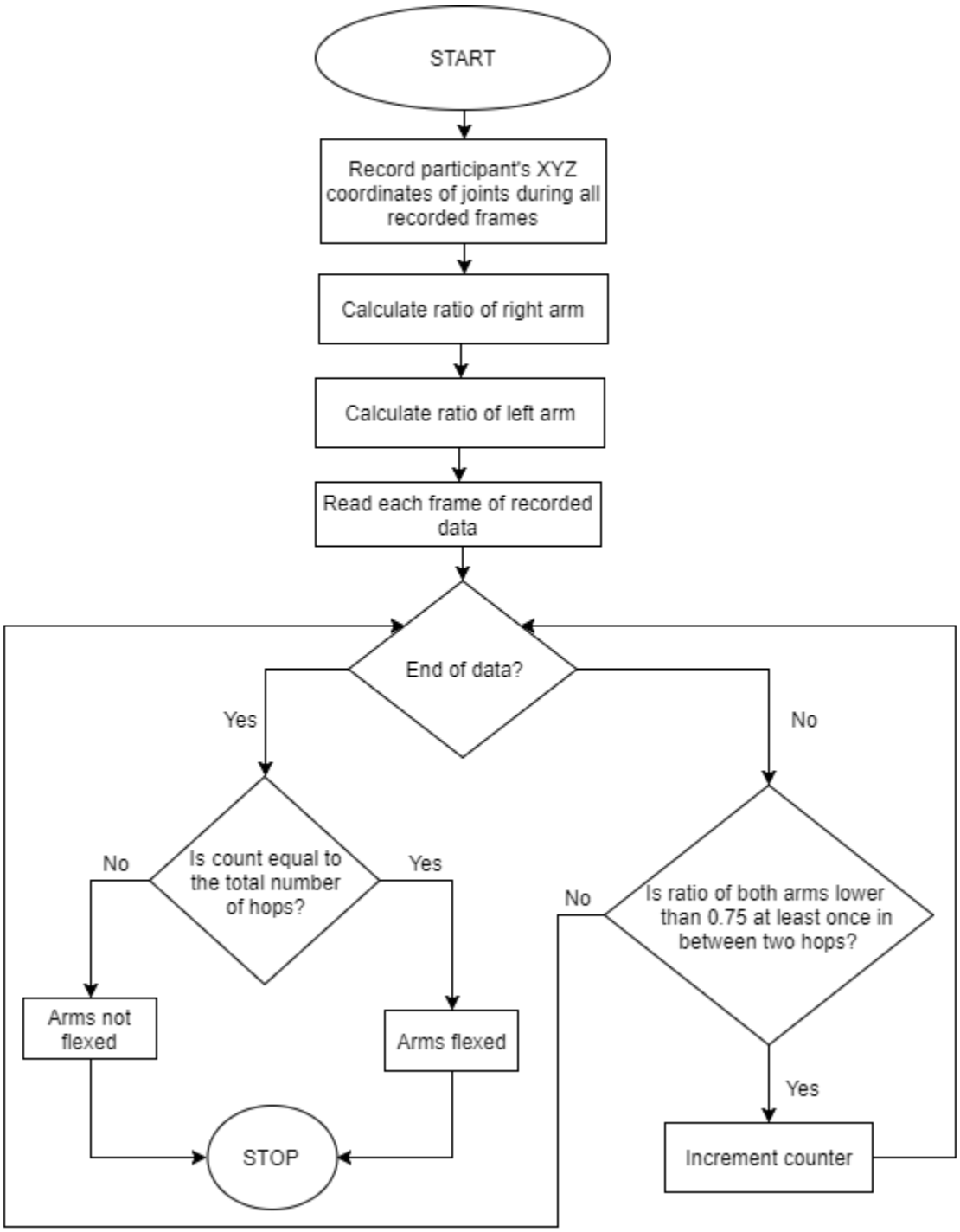

Figure 72. Algorithm to determine whether participant flexes arms during the movement.

In addition, the algorithm also has to check if the arms swing forward to produce force. Figure 38

shows an example of a swing forward and a swing backward using the arms.

As previously mentioned, a swing can be identified by an angle α formed between the upper arm and the line that represents the trunk. The angle α can be calculated as follows:

$$\alpha = \tan^{-1}\left(\frac{Elbow\ Z - ShoulderZ}{ShoulderY - ElbowY}\right) \tag{12}$$

Moreover, the angle α is positive or negative depending on whether the elbow is in front of or behind the trunk. Figure 73 illustrates the variation of the angle α during the performance of the movement.

Note that in between the two hops there is at least one value of α above 0°, and at least one value of α below 0° for both arms, which means that the participant swings the arms to produce force.

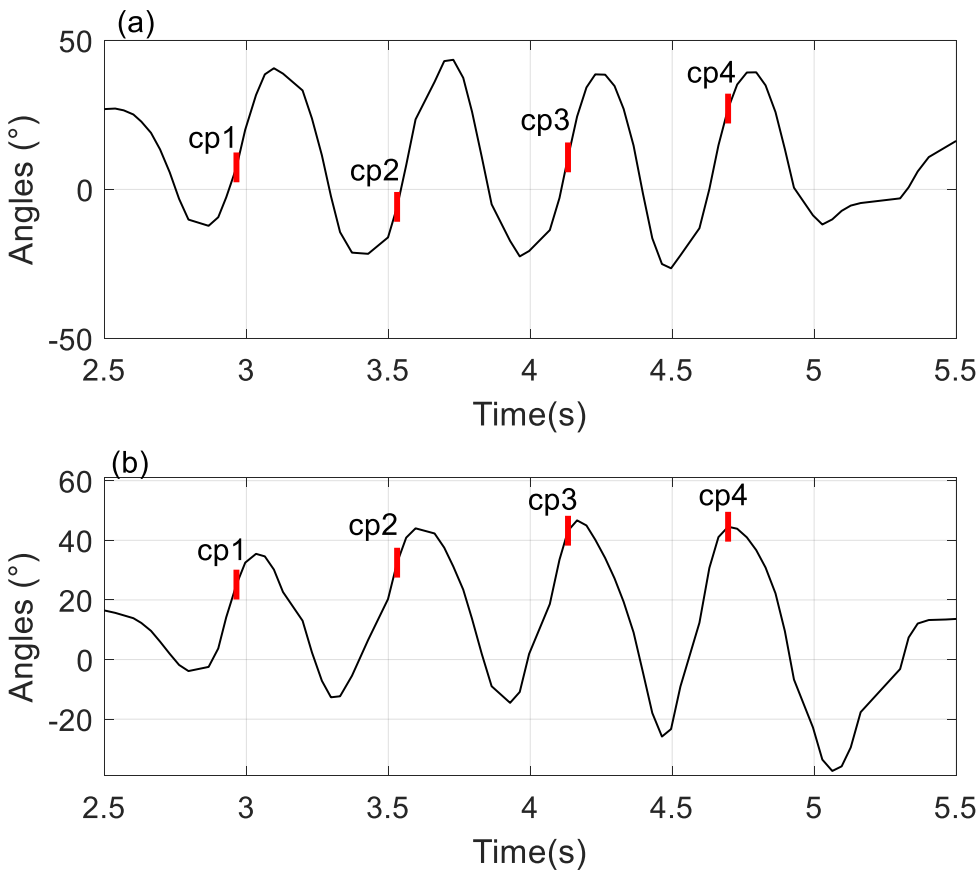

Figure 73. (a) Angles alpha of right shoulder. (b) Angles alpha of left shoulder.

The flowchart that illustrates how to check if the arms swing is:

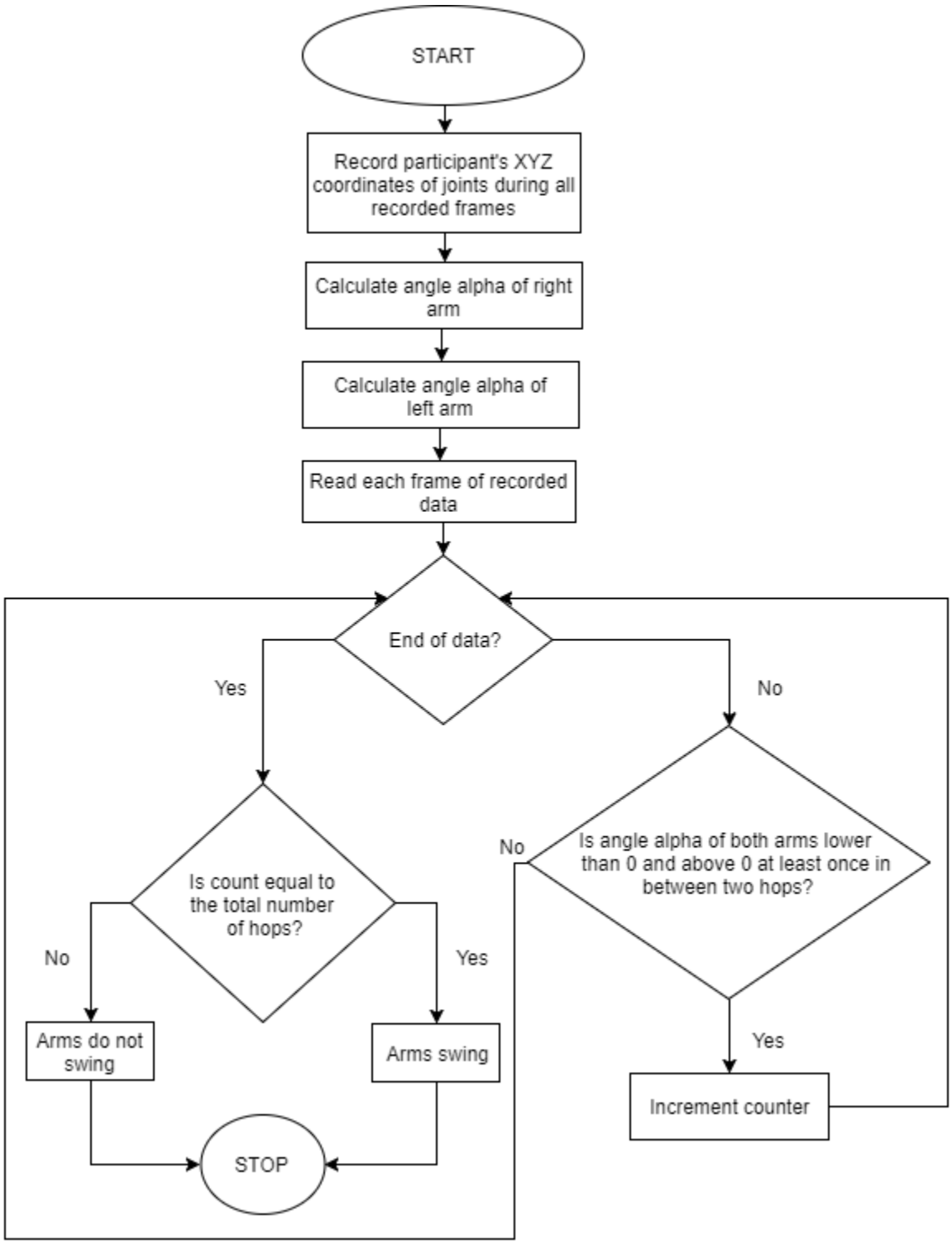

Figure 74. Algorithm to determine whether arms swing.

The following flowchart illustrates the feedback for the third criterion:

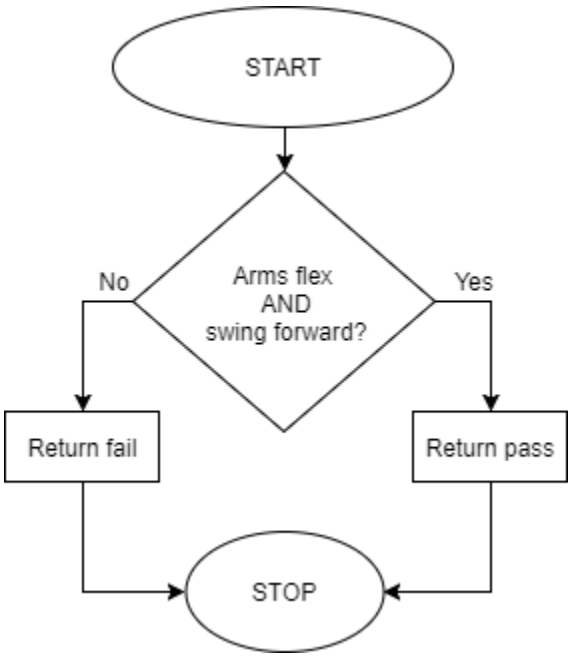

Figure 75. Algorithm to return feedback for third criterion.

<u>Criterion 4: Hops 4 consecutive times on the preferred foot before stopping.</u>

The last criterion consists of analyzing if the participant performs 4 hops before stopping. In order to check this, the algorithm uses the number of peaks present on the Y coordinate of the hopping knee to count how many hops are performed. If the number of peaks is equal to four, it means that the participant has performed four consecutive hops. In addition, in order to ensure that all the hops were performed on the preferred leg, the algorithm checks that the non-hoping foot never touches the floor during the hops. Therefore, for all frames recorded during the hops, the Y coordinate of the non-hoping foot is expected to be greater than the Y coordinate of hoping foot.

The flowchart that illustrates how to check if participant hops 4 consecutive times on the preferred foot is:

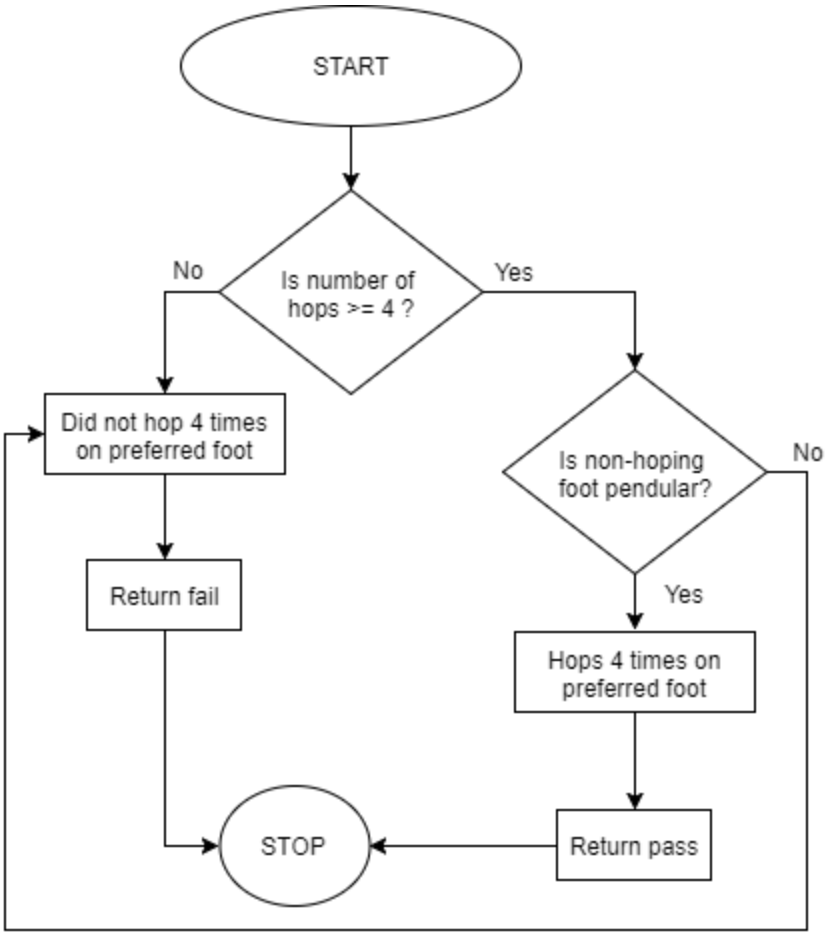

Figure 76. Algorithm to determine whether participant passes or fails the fourth criterion.

4.3.4 Skip

According to the TGMD-3 scoring sheet, the participant is asked to perform four skips conforming to the specified performance criteria. An example of a skip movement is shown in Figure 77 below, where the Kinect is positioned facing the participant.

Figure 77. The sequence of images used to represent a standardized skip movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

As was done for the running and gallop tests, the peaks drawn by the Y coordinates of the head along time were used to determine the critical points, and distinguish one skip phase from another, as shown in Figure 78.

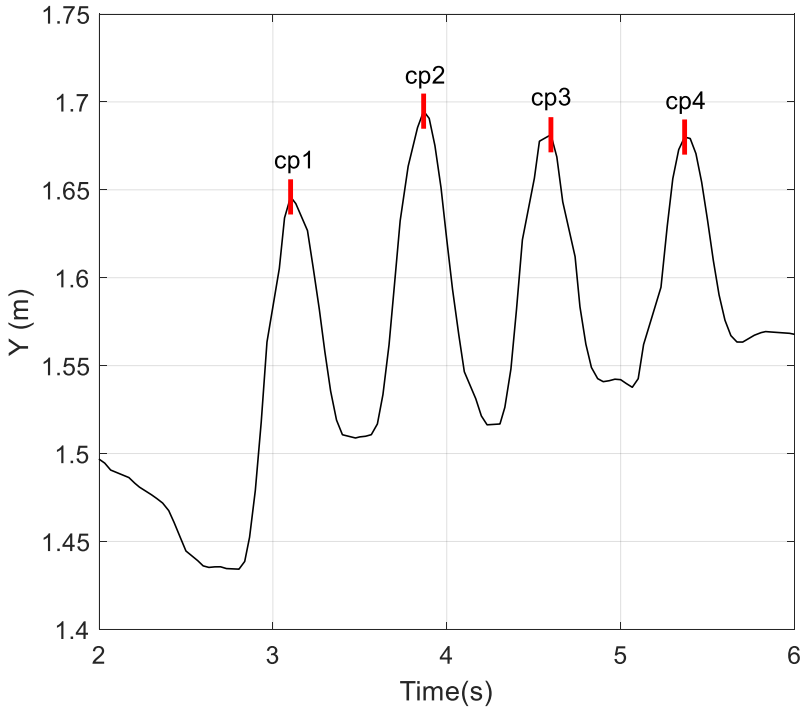

Figure 78. Variation along time of Y coordinates of head and critical points.

Note from Figure 77 that for the skip test, when the participant hops on a leg, s/he is supposed to raise and flex the non-hoping leg. Moreover, at each critical point s/he has the non-hopping leg flexed and the hopping leg performing the hop.

Criterion 1: A step forward followed by a hop on the same foot.

The first criterion for the skip test consists of analyzing if the participant performs a step forward, followed by a hop on the same foot. The algorithm first needs to detect which foot the participant starts the movement with. By analyzing the behavior of the distance between the feet along time, the algorithm considers the first step starts when the difference between the feet is equal or superior to 10 cm. Note from Figure 79 that initially, both feet are roughly the same distance from the Kinect, and as the participant starts the movement, the distance between the feet begins to increase.

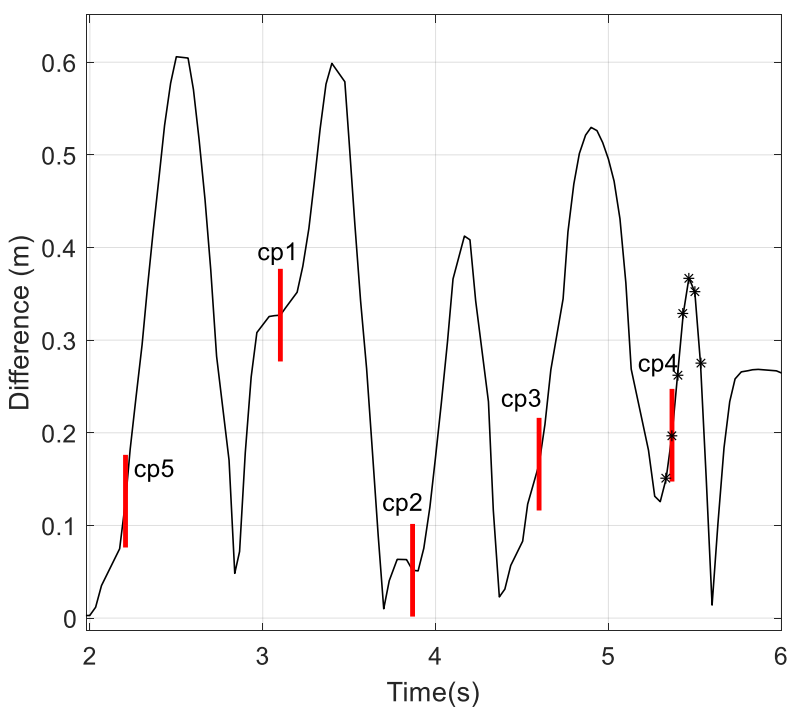

Figure 79. The difference in the Z axis between feet along time and critical points. The (*) represents the moment when the 3D position of the joint was inferred.

The point at which they reach a distance superior or equal to 10 cm from each other (“cp5”) is when the step begins. Then, to check which foot starts the movement with, the algorithm checks which one is closer to the sensor at “cp5” — the same idea used for the gallop test.

Figure 80 shows a closer look at the Z coordinates of both feet at “cp5”. Note that the left foot is closer to the Kinect than the right foot for that moment, which means the left foot starts the step.

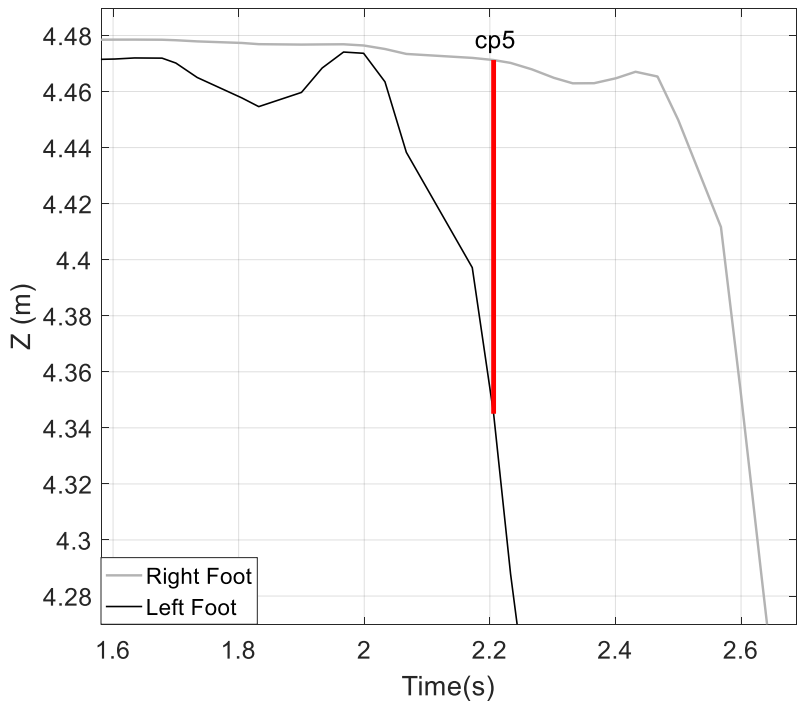

Figure 80. A closer look at the Z coordinates of feet at "cp5".

Similar to the analysis used for the gallop test, note in Figure 79 that in between two skips, the distance between the feet increases to a maximum value within that period, which indicates that a step is being performed. Figure 81 shows the behavior of the Z coordinate of the feet during the performance of the steps.

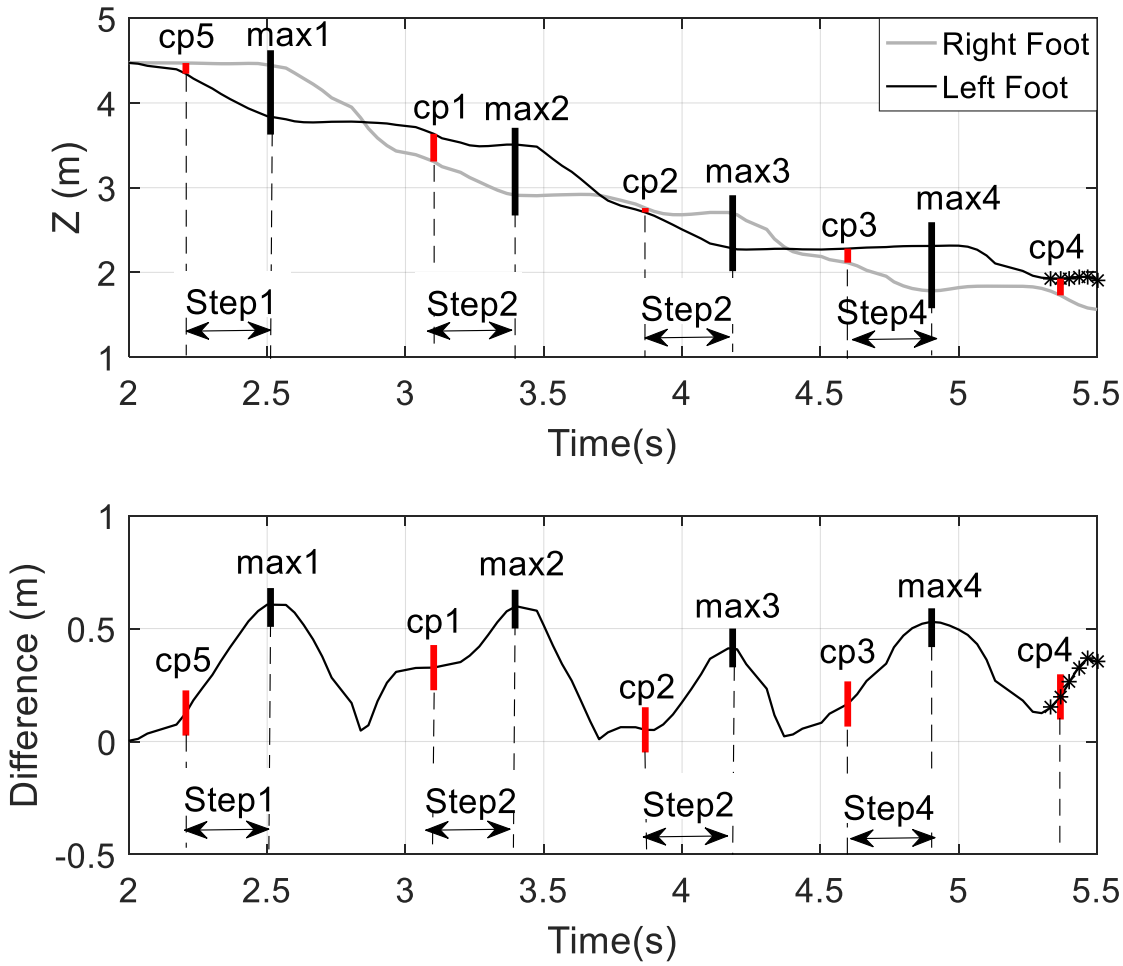

Figure 81. (a) Variation along time of the Z coordinates of feet and moment when each step occurs. (b) The difference between feet along Z axis and moment when each step occurs. The (*) represents the moment when the 3D position of the joint was inferred.

Note that for each skip, i.e. in between two critical points, one step is performed. The step is completed when the distance between the feet reaches a maximum value within a skip cycle. Note that during the step, the foot performing the step substantially decreases its coordinate along Z, as it moves towards the Kinect, while the non-stepping foot remains at approximately the same value of Z, which indicates that the non-stepping foot stays placed on the floor and does not move during the performance of the step. Also, note that the foot performing the step alternates for each skip.

Therefore, in order to check if the participant performs a step forward, the algorithm checks the Z coordinate of the non-stepping foot and the stepping foot at the beginning and at the end of the step. If the difference between the initial and final values of the non-stepping foot is lower than the difference between initial and final values of the stepping foot, for all steps, the algorithm considers that the participant has performed a step forward with the leading foot, i.e., it basically considers the slope of the curve. The final value also needs to be lower than the initial value since the movement is performed towards the Kinect.

The following flowcharts illustrate how to check if a step forward is performed alternating with the stepping foot:

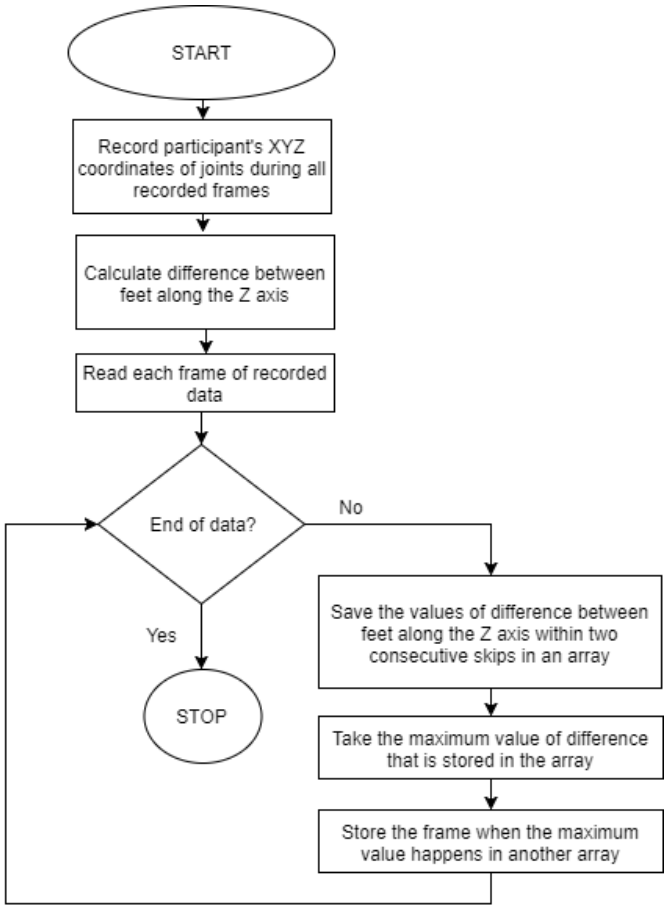

Figure 82. Algorithm to determine the maximum distance between feet within two consecutive skips.

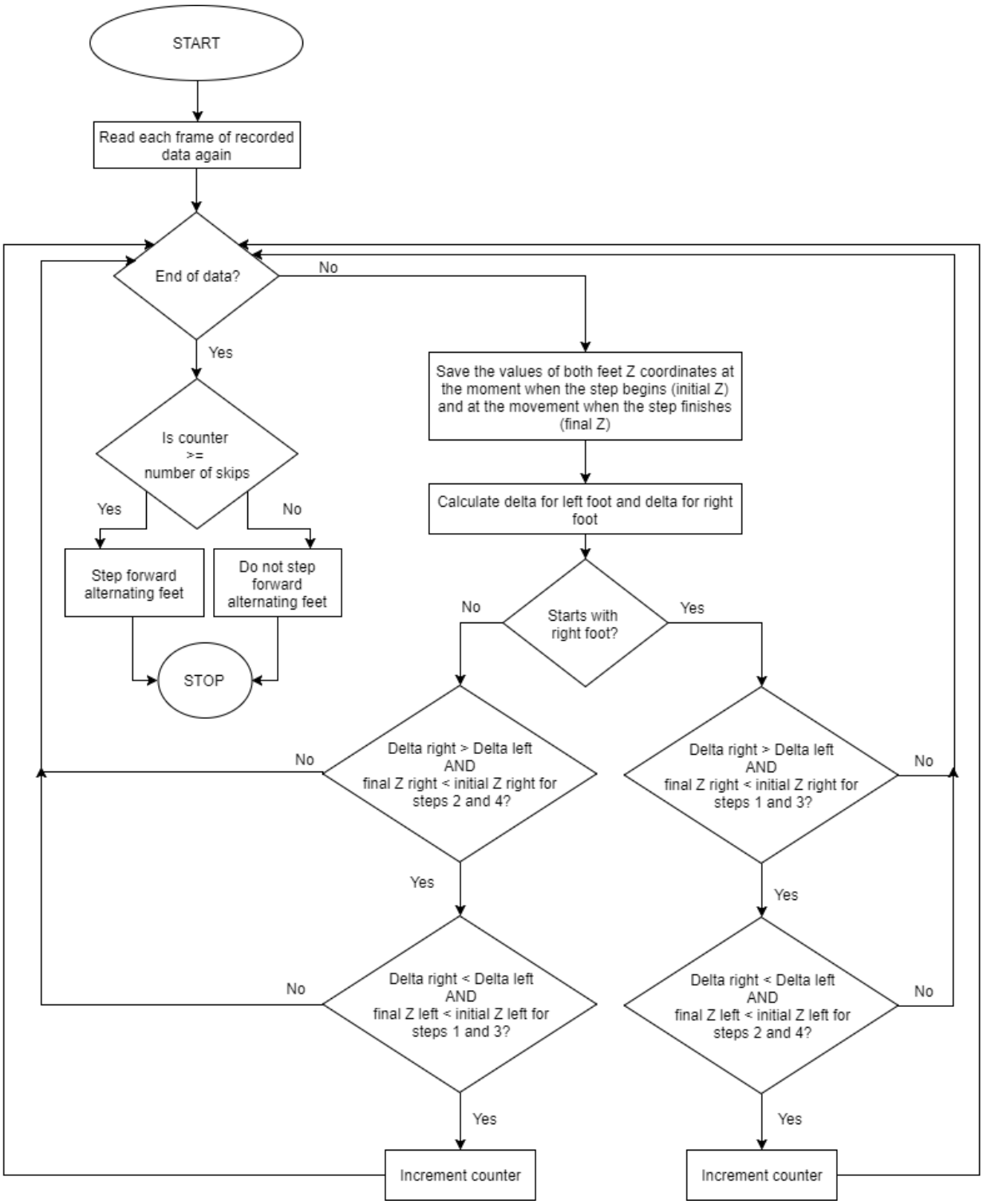

Figure 83. Algorithm to determine whether participant steps forward alternating feet.

In addition, the algorithm also needs to check whether after the step, the participant performs a

hop on the same foot. Note that the hops happen at “cp1”, “cp2”, “cp3” and “cp4”, moments when the participant is off surface. Note in Figure 84 that the foot that performed the step is the same as the foot performing the hop.

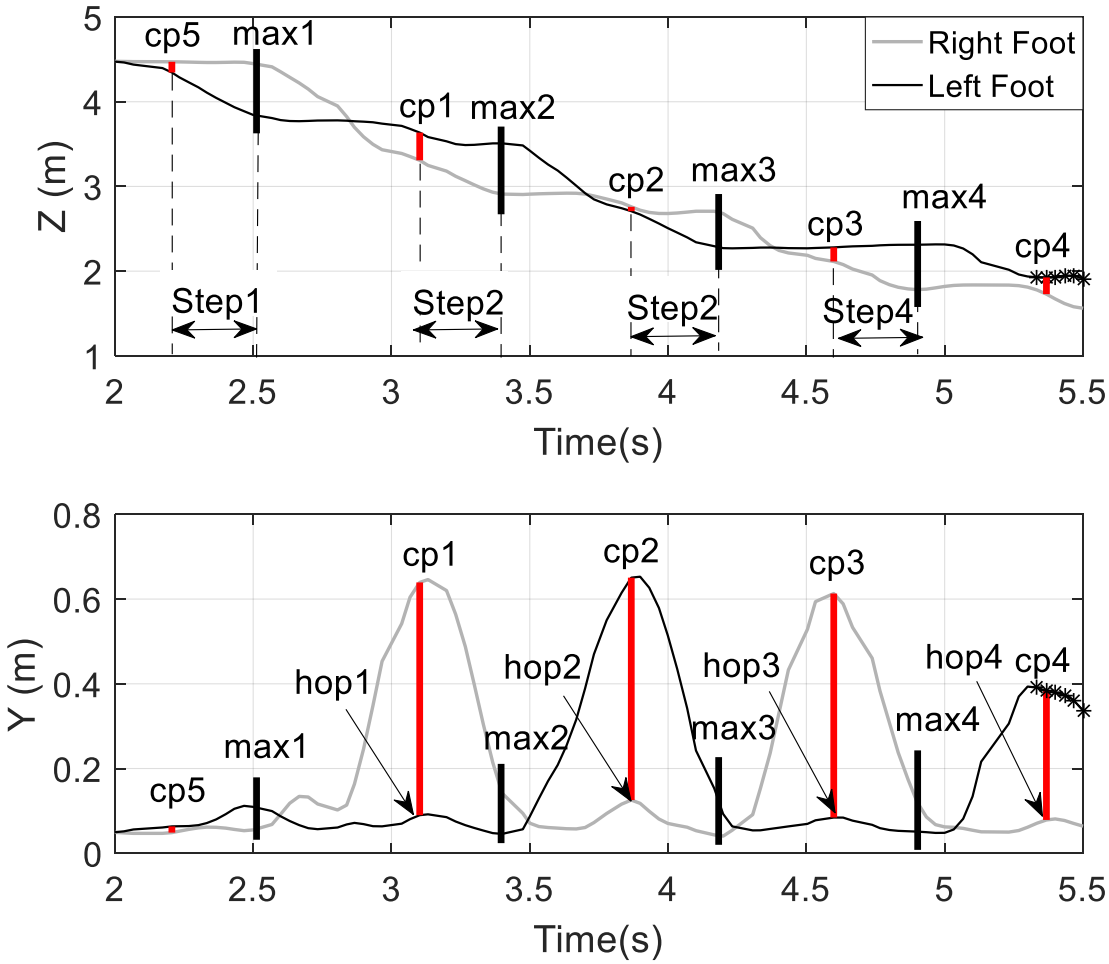

Figure 84. (a) Variation along time of Z coordinates of feet and moment when each step occurs. Variation along time of Y coordinates of feet and moment when each hop occurs. The (*) represents the moment when the 3D position of the joint was inferred.

It is important to mention that by observing Figure 77, it is noticeable that at each critical point, the non-hopping foot draws a high peak as the participant raises his/her leg, while the hopping foot draws a slight peak. This behavior means that while the non-hopping leg is raised and flexed, the hopping leg performs the hop. Therefore, the hops are represented by the small peaks, i.e. the

moments when the Y coordinates of the hoping foot are at least 10% above the floor level acquired during the calibration.

The following flowchart illustrates how to check if stepping leg performs a hop on the same foot:

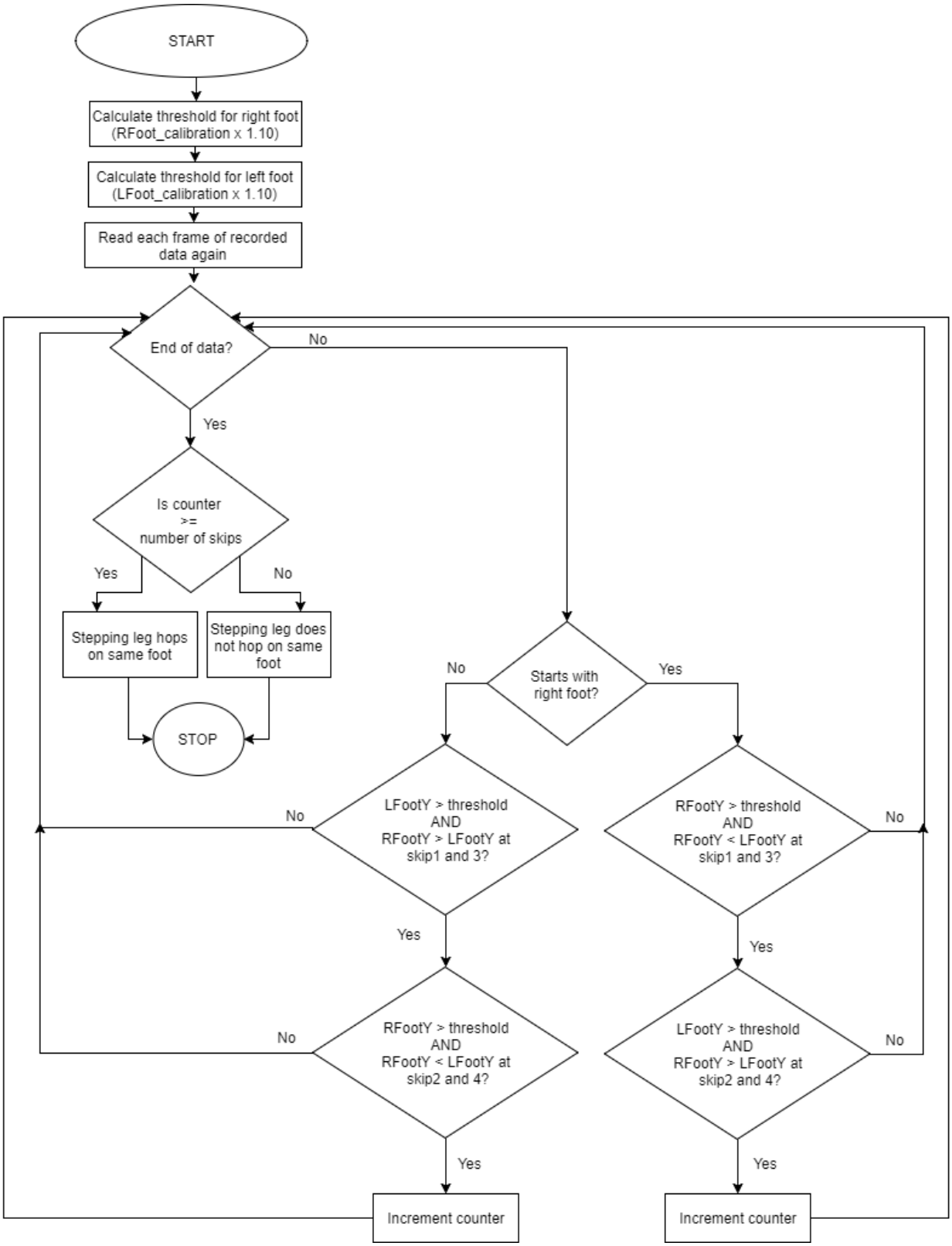

Figure 85. Algorithm to determine whether participant hops on the same foot.

The following flowchart illustrates the feedback for the first criterion:

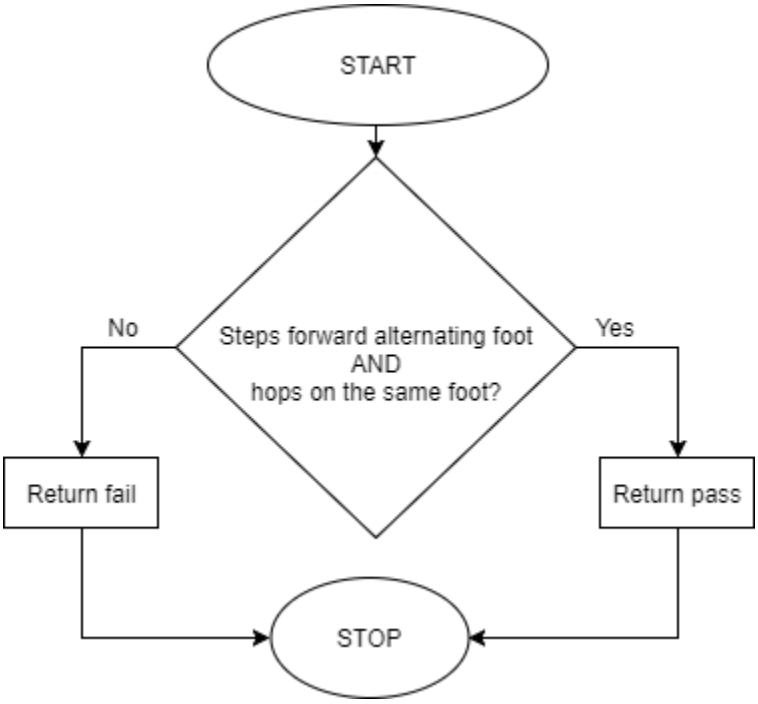

Figure 86. Algorithm to return feedback for the criterion.

<u>Criterion 2: Arms are flexed and move in opposition to legs to produce force.</u>

The second criterion is similar to the first criterion of the running test. The participant is asked to flex his/her arms and move them in opposition to legs to produce force. Figure 77, Figure 87, and Figure 88 show the pattern for the standard movement.

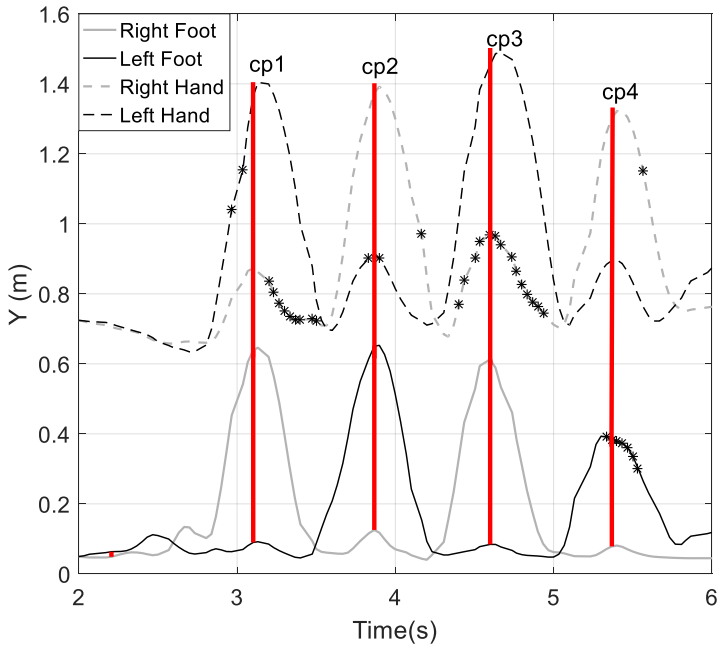

Figure 87. Variation along time of Y coordinates of hands, feet and critical points. The (*) represents the moment when the 3D position of the joint was inferred.

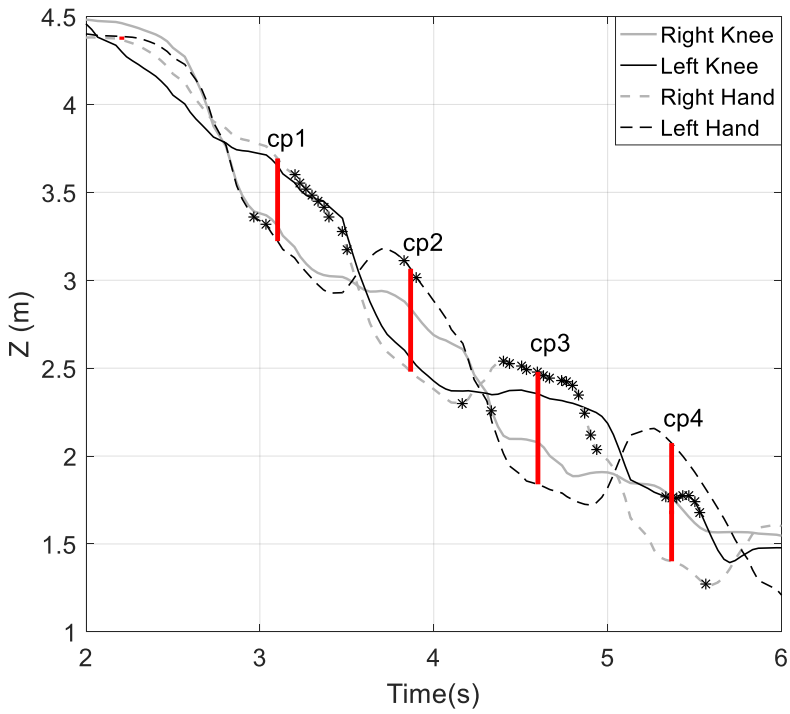

Figure 88. Variation along time of the Z coordinates of hands, feet and critical points. The (*) represents the moment when the 3D position of the joint was inferred.

Note that for each critical point, the hands are in opposition to the legs. Also note that the hand of the same side of the hopping leg is closer and higher to the Kinect than the other hand at each critical point, and the knee of the hopping leg is lower and further away from the Kinect than the other knee.

Therefore, to assess this criterion, the algorithm checks whether or not at each critical point the hand of the same side of the hopping leg has a higher Y coordinate and a lower Z coordinate than the other hand. It also checks if the knee of the same side of the hopping leg has a higher Z coordinate and a lower Y coordinate than the other knee. In this way, the algorithm knows if the hands are moving in opposition to legs during the whole movement.

The flowchart that illustrates how to check if arms are opposite to the legs is:

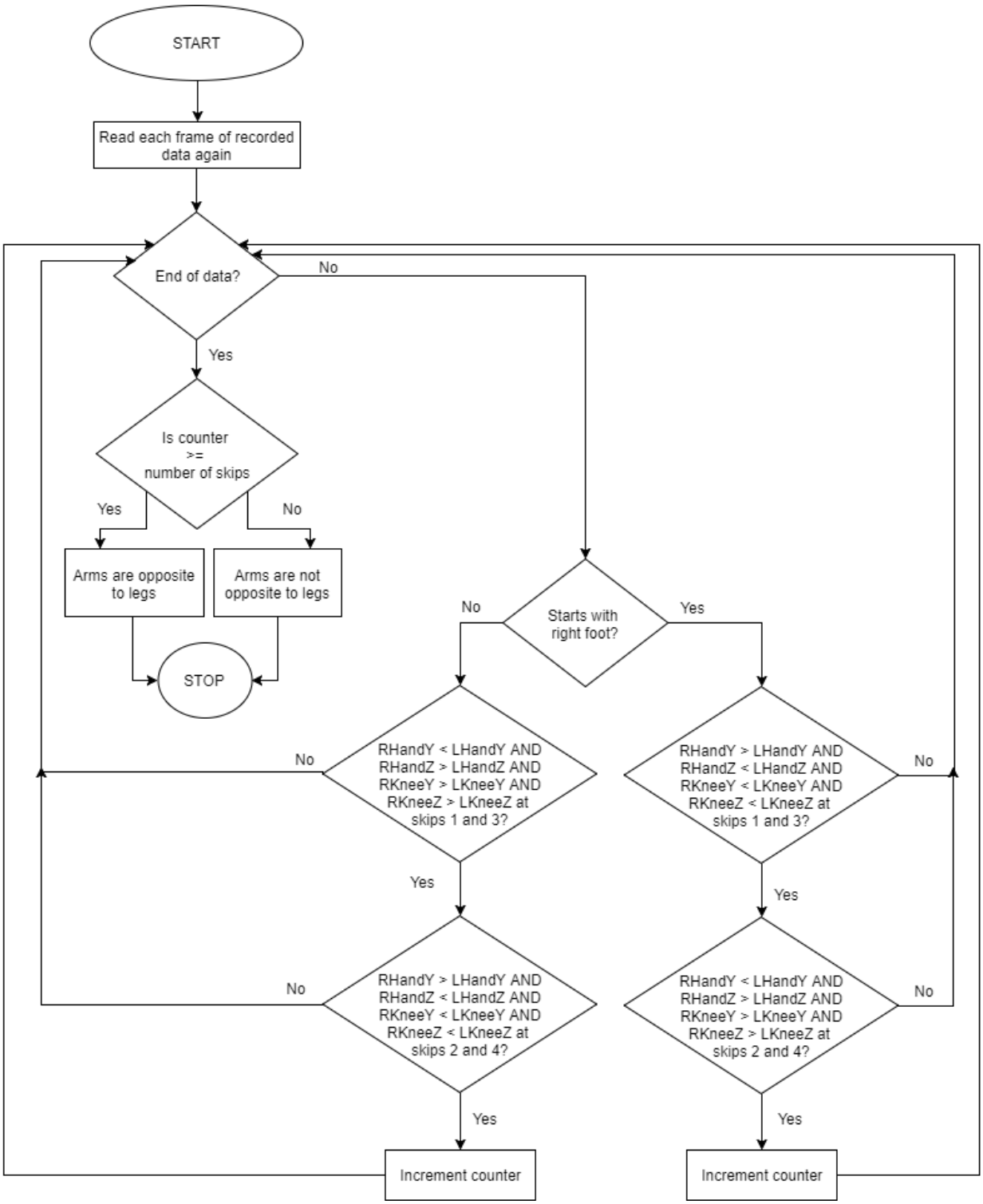

Figure 89. Algorithm to determine whether arms are opposite to legs.

The algorithm also checks if the arms flex using the ratio equation. As mentioned for previous

tests, the ratios for when the arms flex equal or less than 120° corresponds to any value below 0.75. Figure 90 illustrates the ratios of the arms during the performance of the skip test.

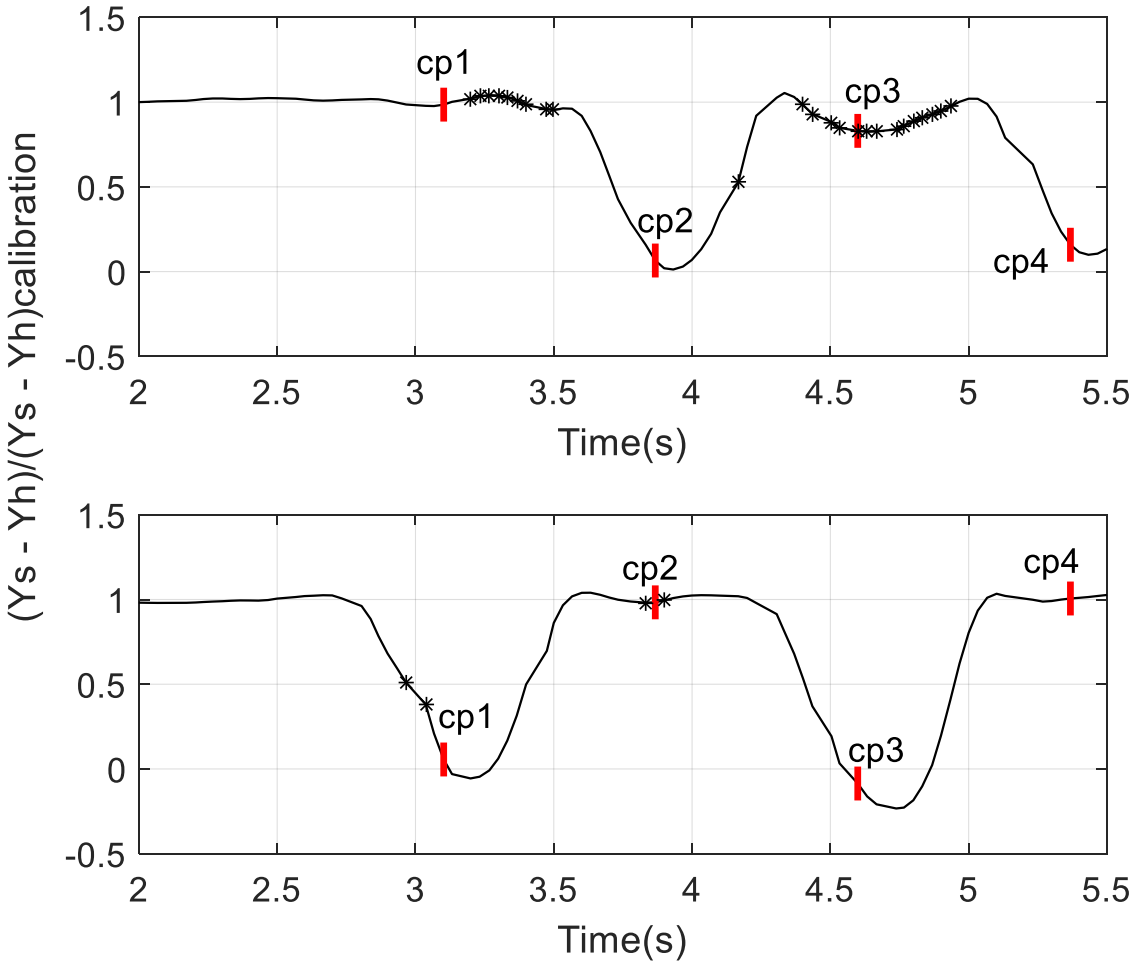

Figure 90. (a) Ratios of right arm and critical points. (b) Ratios of left arm and critical points. The (*) represents the moment when the 3D position of the joint was inferred.

Note that from one critical point to the next one, both arms reach values of ratios below 0.75, which means that the participant had his/her arms flexed for each skip.

The flowchart that illustrates how to check if the arms are flexed is:

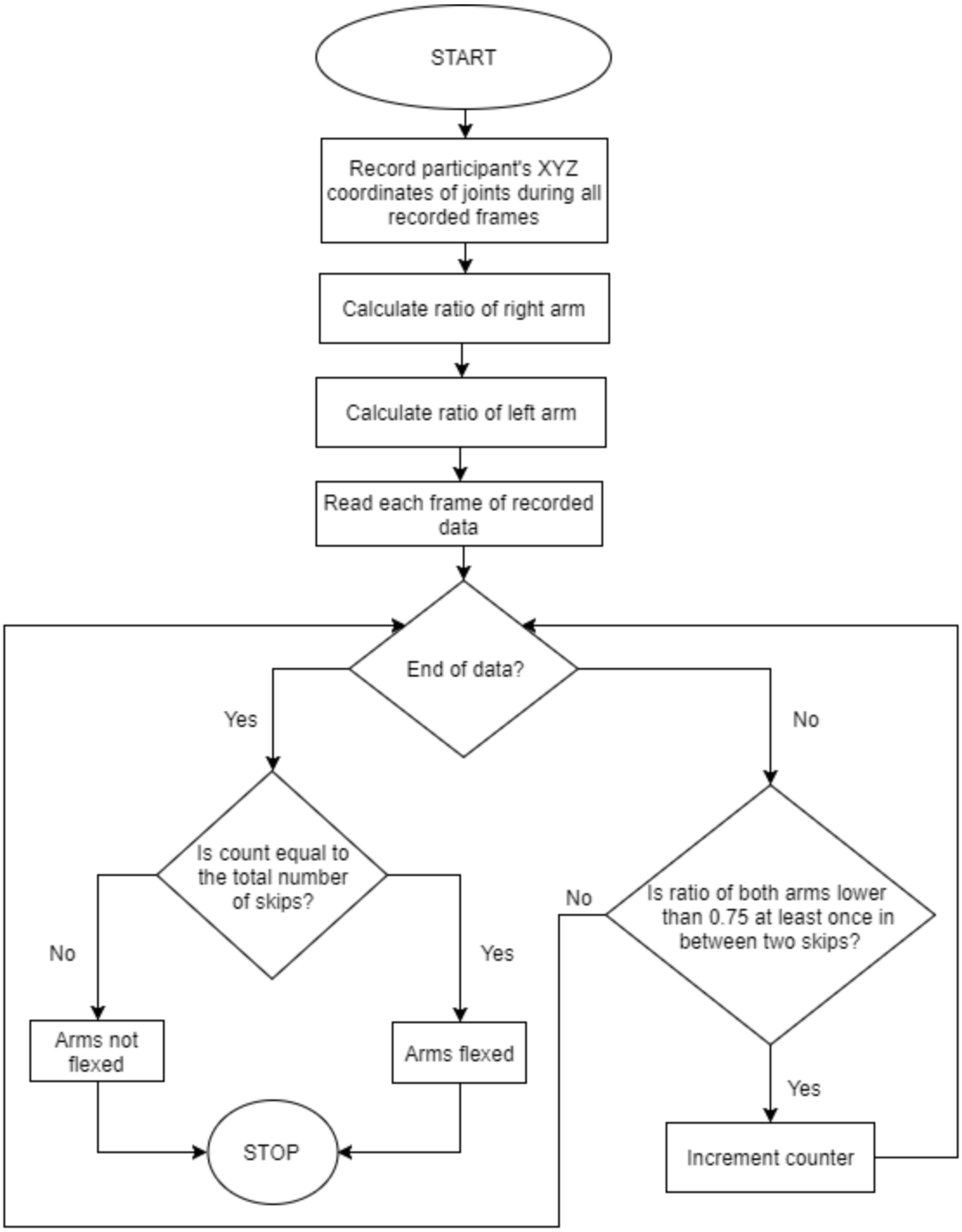

Figure 91. Algorithm to determine whether participant flexes the arms.

The following flowchart illustrates the feedback for the second criterion:

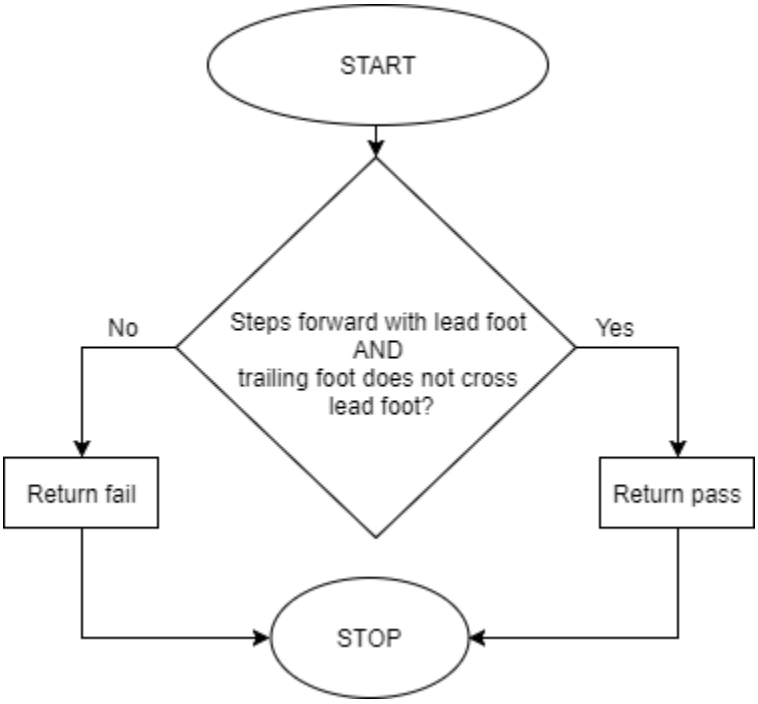

Figure 92. Algorithm to determine whether participant passes or fails the second criterion.

<u>Criterion 3: Completes 4 continuous rhythmical alternating skips.</u>

The last criterion checks if the participant completes 4 continuous rhythmical alternating skips. By knowing which foot starts the movement, the algorithm knows which one is supposed to be performing the hop at each critical point and then it checks if the participant alternates the hopping leg at each skip cycle. It also counts the number of skips by counting how many peaks are drawn by the Y coordinate of the head, i.e. the number of critical points. Moreover, to determine if the movement is performed rhythmically, the algorithm measures the time lapse between critical points, the same as was done for the fourth criterion of the gallop test.

The flowchart that illustrates how to assess the last criterion is:

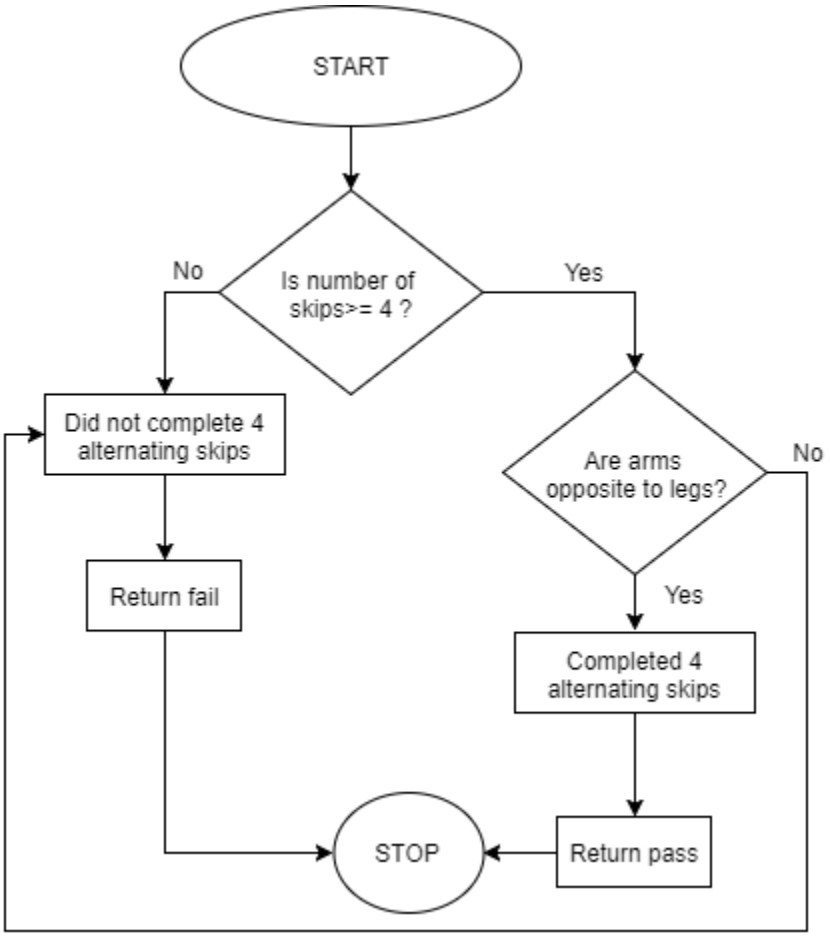

Figure 93. Algorithm to determine whether participant passes or fails the third criterion.

4.3.5 Horizontal Jump

An example of a horizontal jump movement is shown in Figure 94 below, where the Kinect is positioned facing the participant.

Figure 94: The sequence of images to represent a standardized horizontal jump movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

The horizontal jump test is divided into three distinct phases: the preparation phase (1), the aerial phase (2) and the landing phase (3). To assess the criteria, it is necessary to identify when the participant starts the movement and when each phase occurs. On the phase 1, the preparation phase, the user is standing facing the Kinect and as soon as s/he receives the command “prepare”, s/he bends the knees to be ready to initiate the jump. The critical point “cp1” for this phase is when the head Y coordinate drops to its lowest position, which indicates to the software that the user is about to begin the jump. Figure 95 illustrates this.

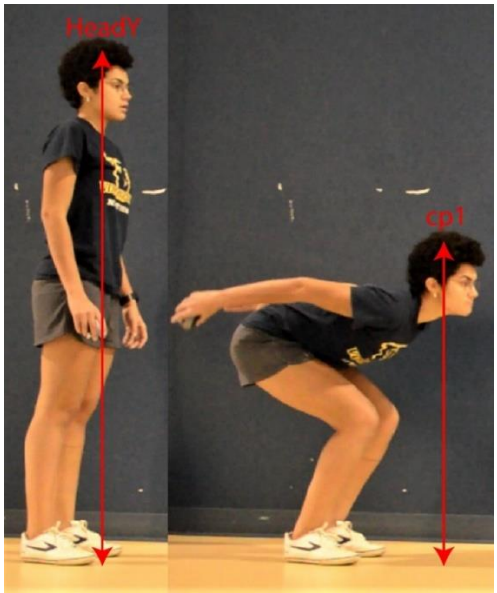

Figure 95. Example of "cp1". Note that the Y coordinate of the head reaches the minimum value before the jump.

During phase 2, the participant performs the jump, coming off the floor. During this phase, the Y coordinate of the head reaches a maximum value and then drops again until the participant lands back on floor. The critical point “cp2” for this phase is when the Y coordinate of the head reaches the highest value of the whole movement. Figure 96 illustrates this.

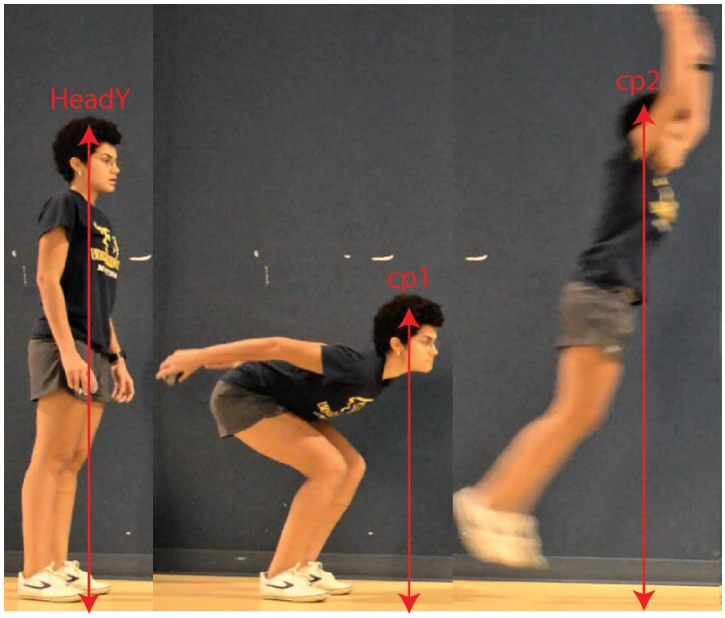

Figure 96. Example of "cp2". Note that the Y coordinate of the head reaches the maximum value during the movement.

Finally, in the landing phase the participant lands on the floor. Its critical point, “cp3”, is when the Y coordinate of the head reaches the lowest value after the maximum value, as shown in Figure 97.

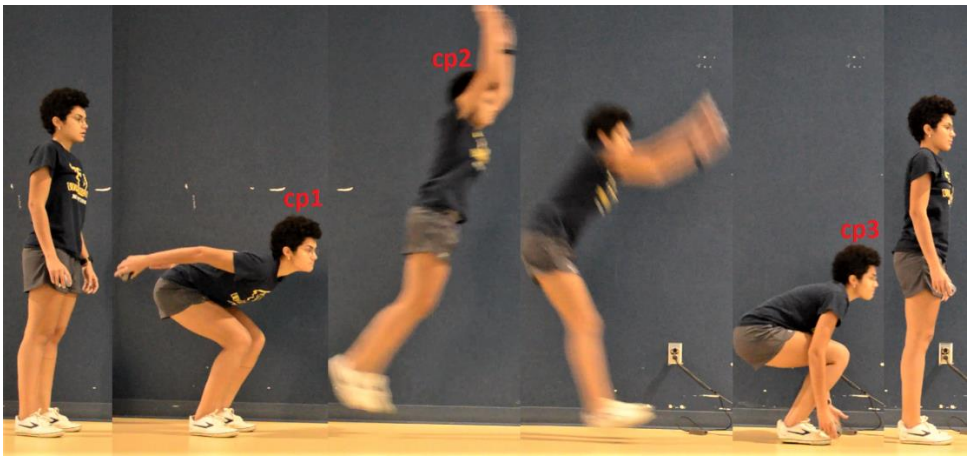

Figure 97. Example of "cp3". Note that the Y coordinate of the head reaches the minimum value after the jump.

By analyzing the variation of the Y coordinate of the head along time using the standardized coordinates, it is possible to determine its behavior. Figure 98 shows the behavior of the Y coordinate of the head, as well as the critical points.

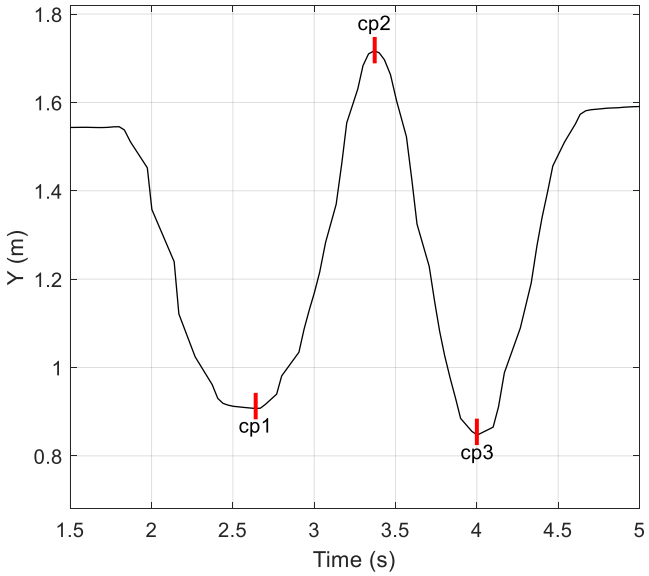

Figure 98. Variation along time of Y coordinate of head and its critical points.

Note that before beginning the test, the participant is standing, and the head Y coordinate stays constant until the participant starts the movement. When the user finishes the phase 1 at “cp1”, the Y head reaches its smallest value before it reaches its greatest value at “cp2” during phase 2. Again, in phase 3, the Y head reaches the smallest value at “cp3”. This way, it is possible to determine in which frame each critical point happens.

The following flowchart illustrates how to calculate the critical points for the movement:

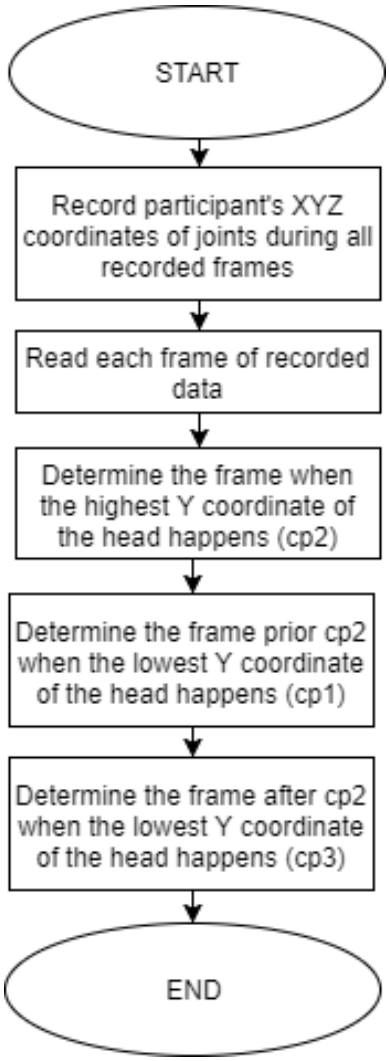

Figure 99. Algorithm to determine the critical points for the movement.

Criterion 1: Prior to take off both knees are flexed and arms are extended behind the back.

The first criterion analyzes if the participant flex both knees and if their arms are extended behind the back at the "cp1". By knowing the frame where the "cp1" occurs, the algorithm analyzes if these conditions apply. The algorithm then uses the ratio of the legs to determine if both ratios are below 0.75 at "cp1", where the ratio can be calculated as follows:

$$ratio = \frac{HipY - FootY}{HipY_{calibration} - FootY_{calibration}} \tag{13}$$

Note in Figure 100 that the ratios of both legs are below 0.75 at "cp1", which indicates that the participant has his/her knees flexed prior the jump.

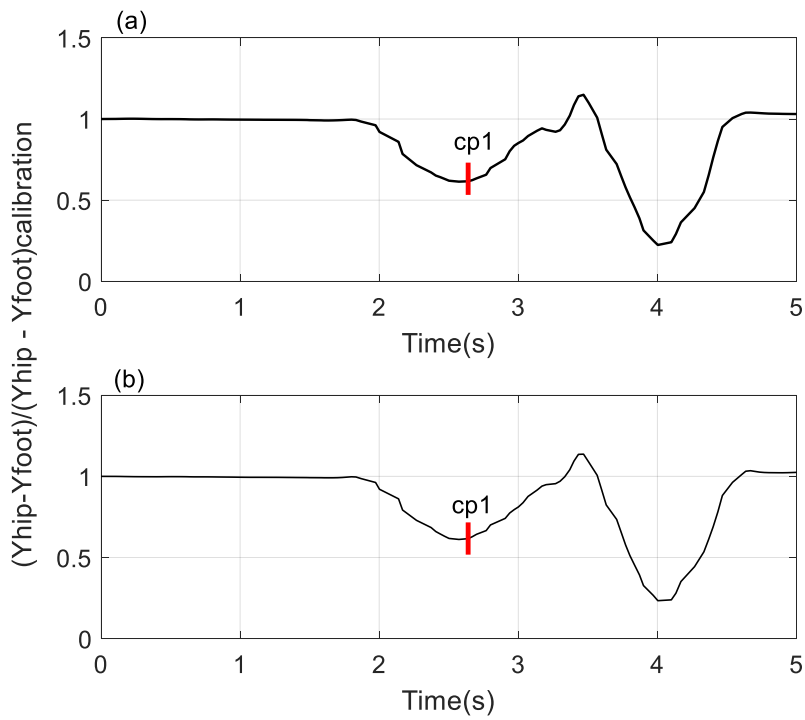

Figure 100. (a) Ratio of right knee at "cp1". (b) Ratio of left knee at "cp1".

The following flowchart shows how the algorithm checks if knees are flexed at “cp1”:

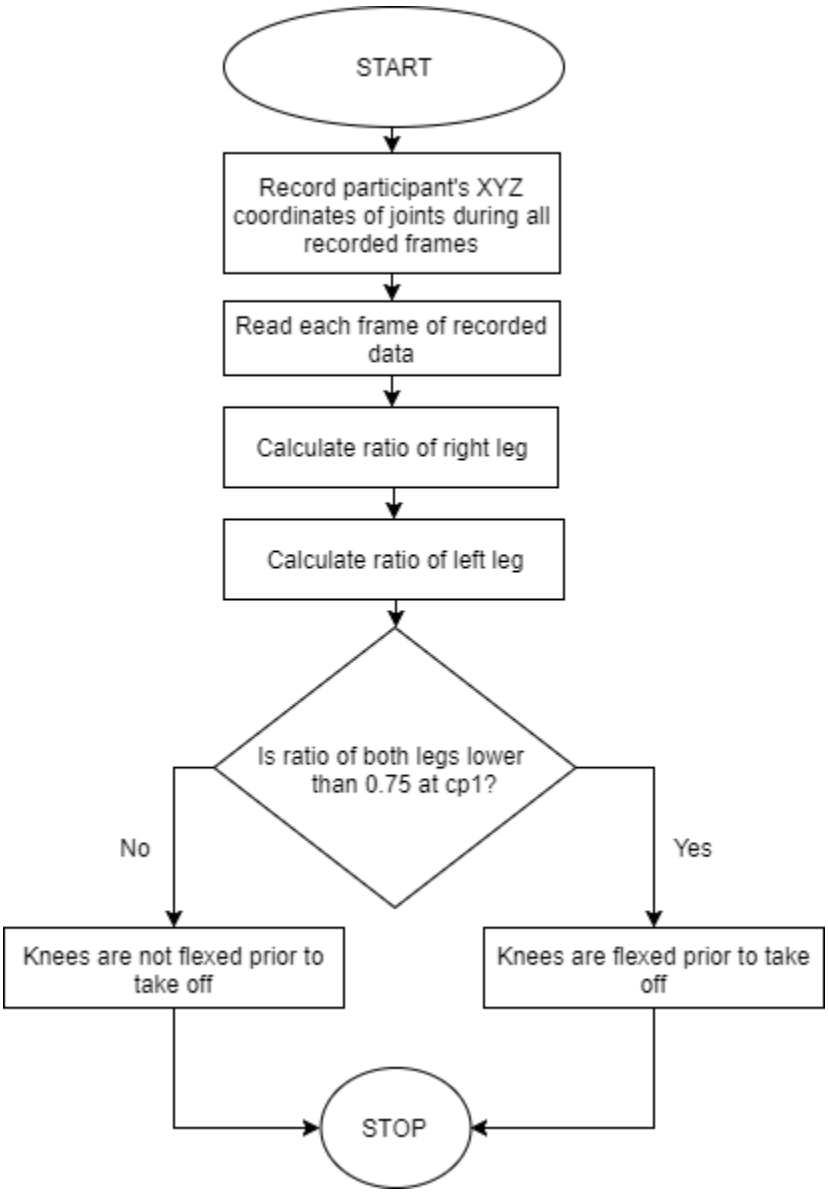

Figure 101. Algorithm to determine whether participant flexes the knees prior to take off.

In order to determine if the arms are extended behind the back, the algorithm checks if the Z coordinates of both hands are further from the sensor than the Z coordinate of the head at “cp1”. As the Z axis increases out in the direction the sensor is facing, and as the user performs the entire movement facing the Kinect, if both hands have a bigger Z coordinate than the head, this means

that the hands are behind the head. Note in Figure 102 that the hands are behind the head, and consequently, behind the back, at “cp1”.

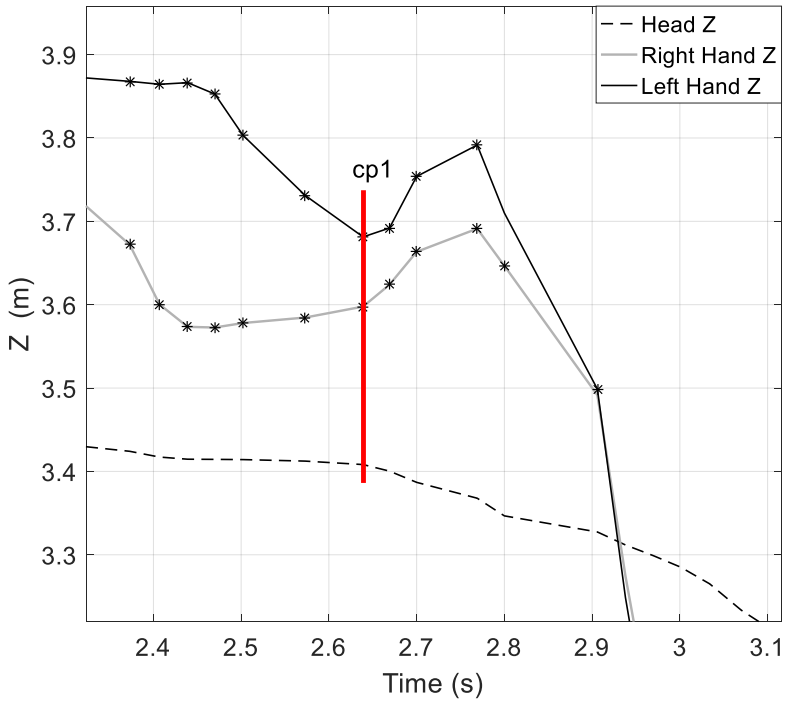

Figure 102. Z coordinates of head and hands at “cp1”. The (*) represents the moment when the 3D position of the joint was inferred.

The following flowchart illustrates how to check if arms are extended behind the back at “cp1”:

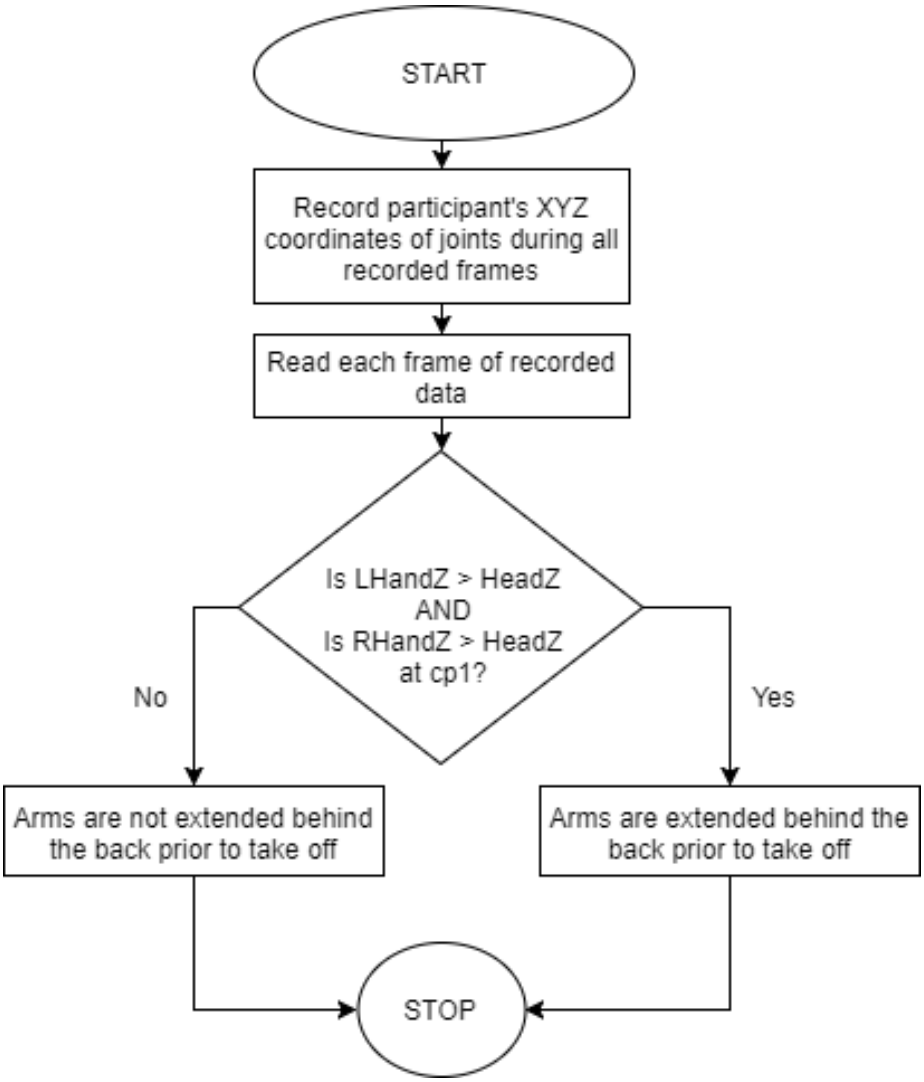

Figure 103. Algorithm to determine whether participant extends arms behind the back prior to take off.

The following flowchart illustrates the feedback for the first criterion:

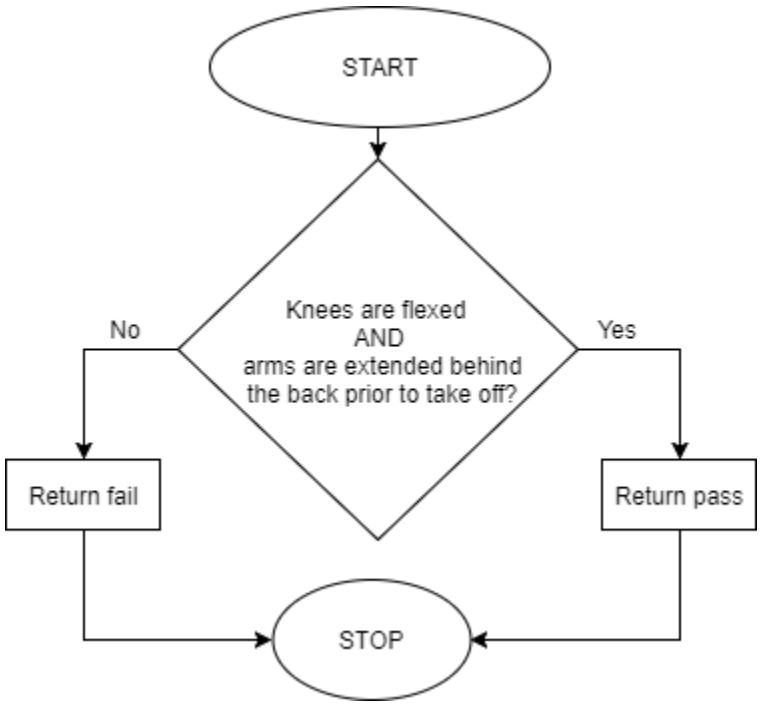

Figure 104. Algorithm to return feedback for the first criterion.

<u>Criterion 2: Arms extend forcefully forward and upward reaching above the head.</u>

The second criterion analyzes if the arms extend forcefully forward and upward, reaching above the head at “cp2”. By checking if the Y coordinates of both hands are greater than the Y coordinate of the head at the frame in which “cp2” occurs, the algorithm can assess if the participant has performed the criterion correctly. Note in Figure 105 that both hands reach above the head at the “cp2”.

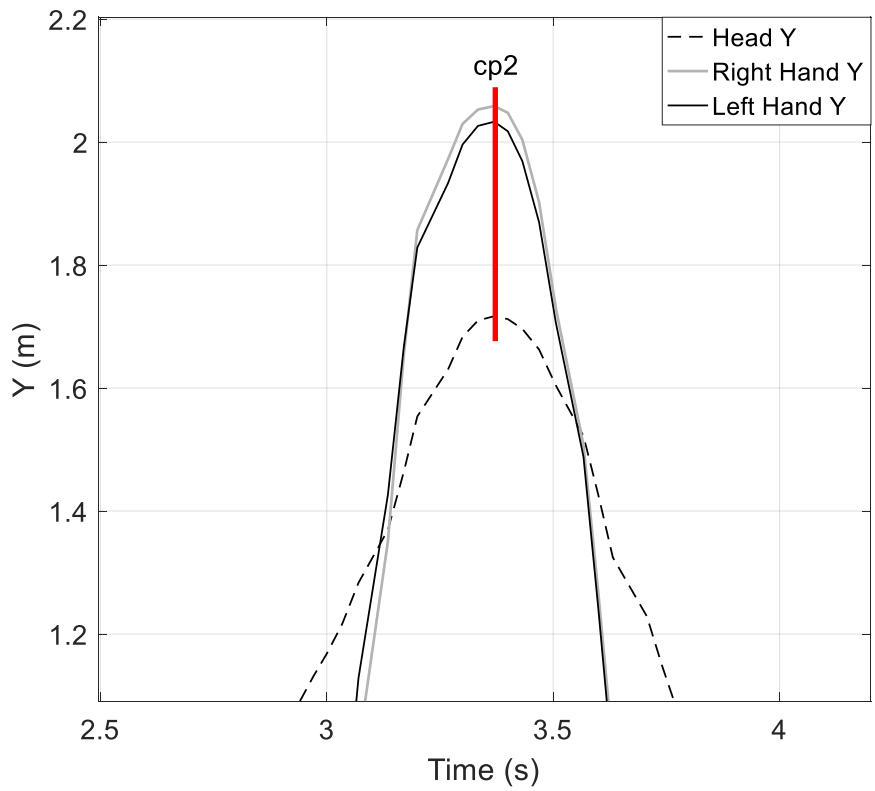

Figure 105. The Y coordinates of head and hands at “cp2”.

The following flowchart shows how to check the second criterion:

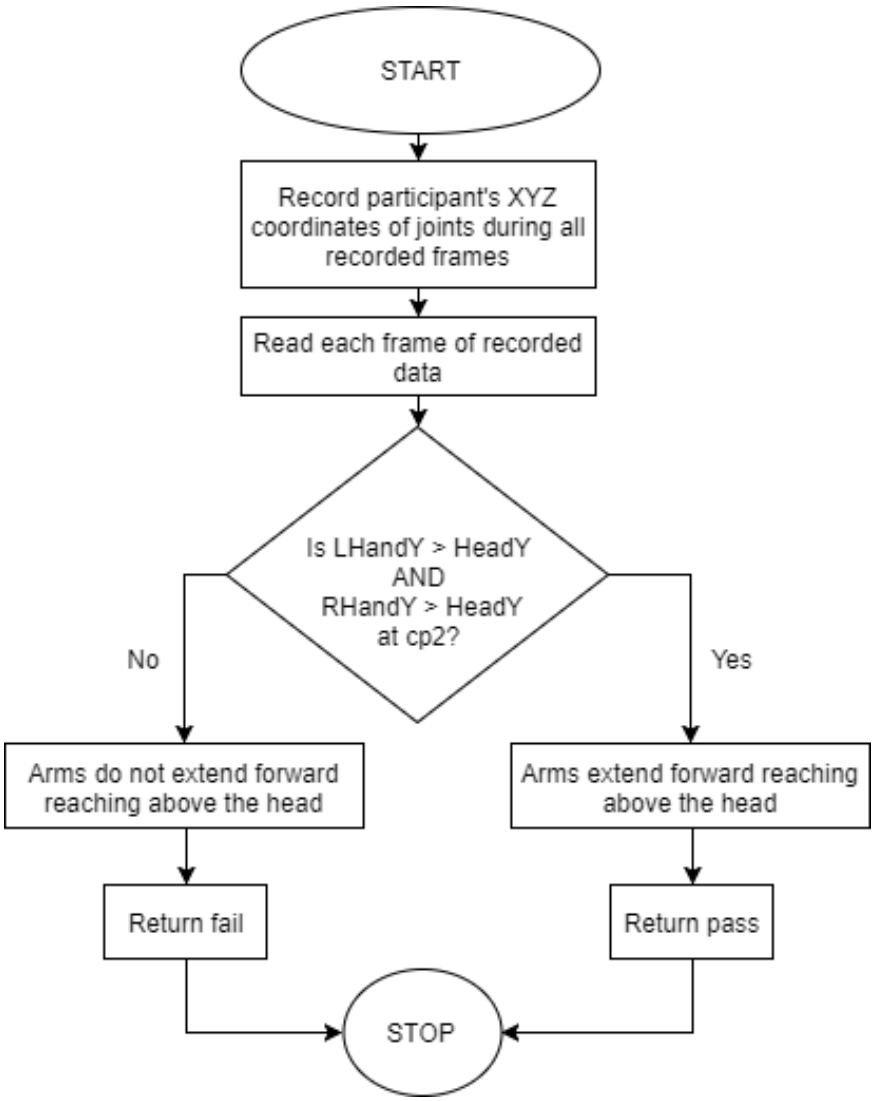

Figure 106. Algorithm to determine whether participant passes or fails the second criterion.

<u>Criterion 3: Both feet come off the floor together and land together.</u>

The third criterion analyses whether both feet come off the floor together and land together. To determine if they leave the floor together, the algorithm checks the moment when both feet are 20% above floor level, which is determined from the calibration, and then checks the time lapse between the moment when the right foot is above the threshold and the moment when the left foot is above the threshold. To determine if they land together, the algorithm checks the first occurrence

when both feet reach 20% below floor level, after they have left the floor, and then checks the time lapse between the moment when the both the right and left feet are below the threshold.

In addition, the algorithm allows a tolerance of three frames (0.1 seconds) for them to be apart, since it is very hard for this action to occur at the exact same fraction of second, due to the high sample frequency. For an observer, they would still be considered to be happening at the same time. Note in Figure 107 that the first occurrence of feet above the threshold happens at the same time equal to t = 3.17s and the first occurrence of feet below the threshold happens at the same time equal to t = 3.70s.

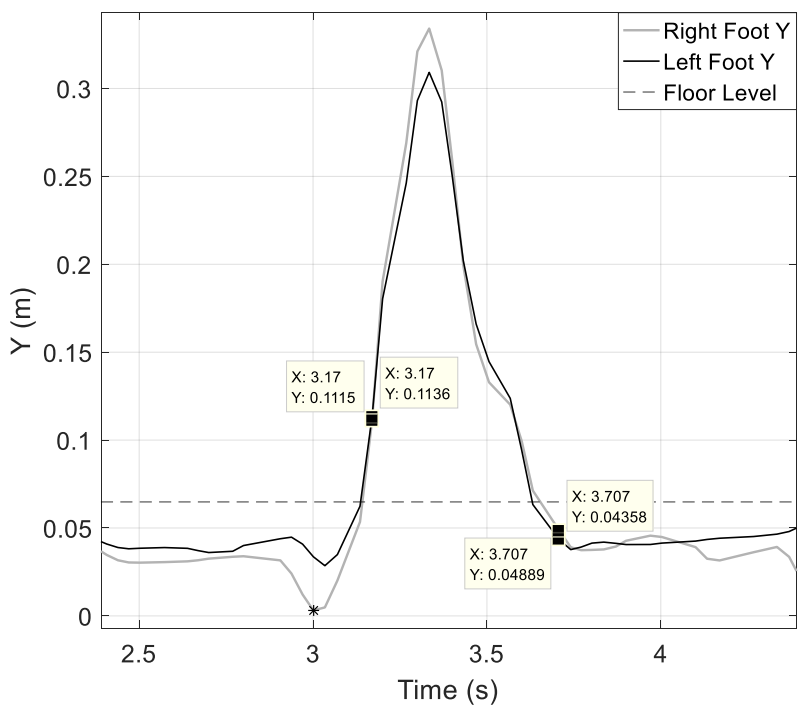

Figure 107. Variation along time of Y coordinate of feet.

The following flowchart illustrates how to determine if feet leave together and land together:

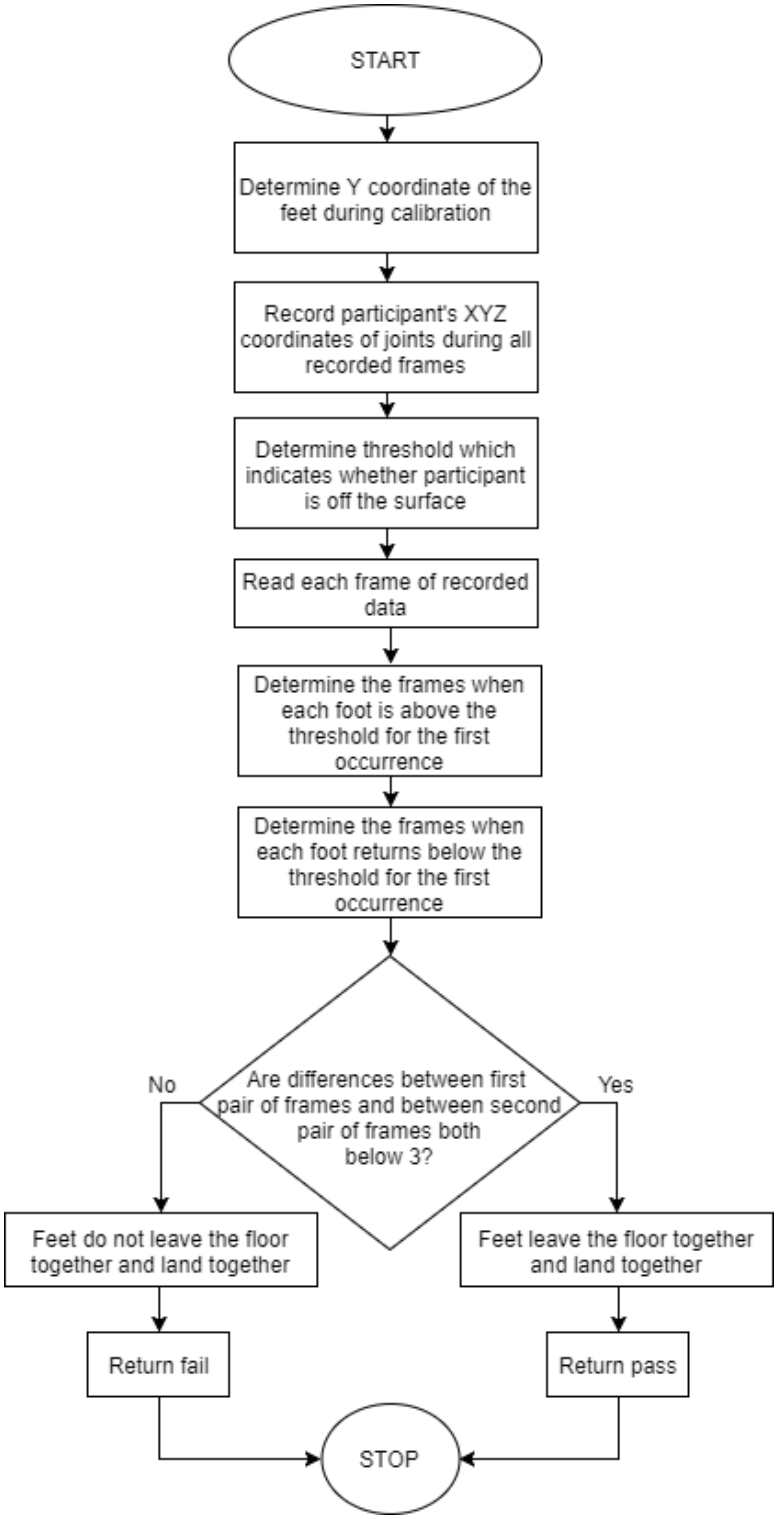

Figure 108. Algorithm to determine whether participant passes or fails the third criterion.

Criterion 4: Both arms are forced downward during landing.

The last criterion analyzes if both arms are forced downward during landing. To determine this, the algorithm checks if both the Y coordinates of the hands are lower than the Y coordinate of the knees at “cp3”.

Note in Figure 109 that both hands are below the knees at “cp3”. This means that the arms were forced downward during the transition from phase 2 to phase 3.

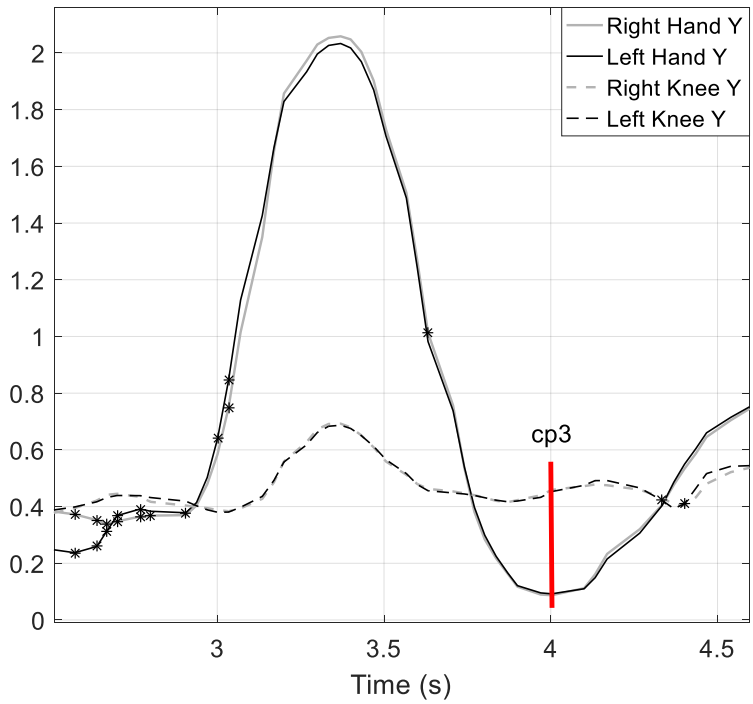

Figure 109. The Y coordinates of hands and knees at “cp3”. The (*) represents the moment when the 3D position of the joint was inferred.

The following flowchart illustrates how to check this:

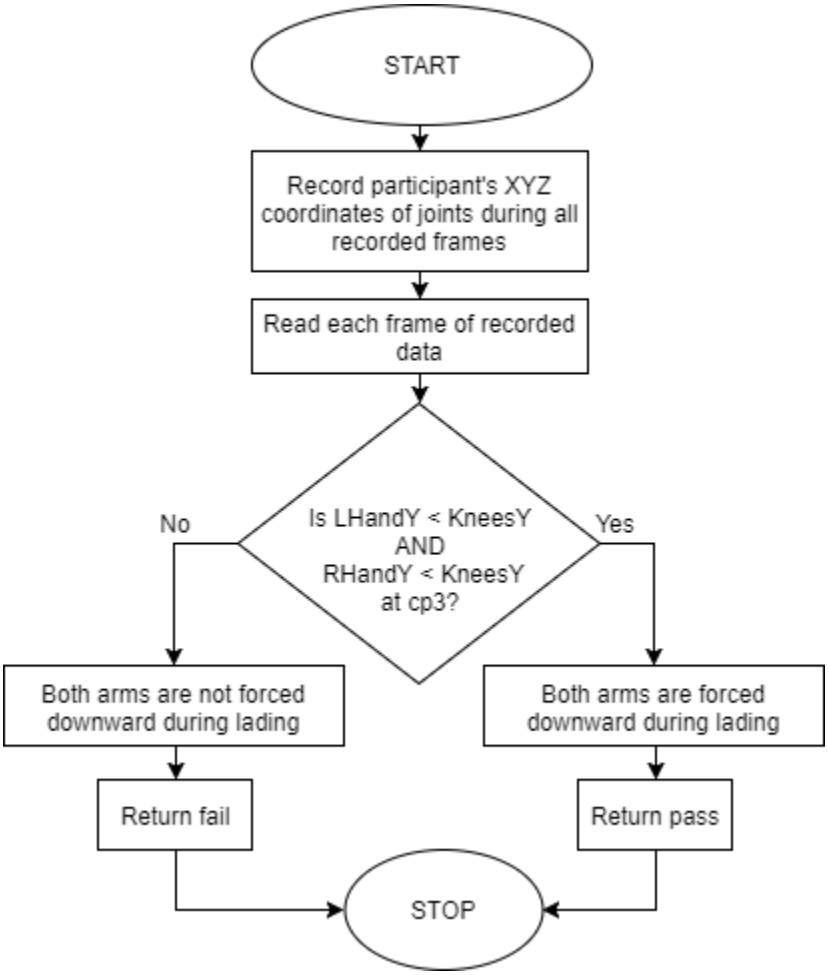

Figure 110. Algorithm to determine whether participant passes or fails the fourth criterion.

4.3.6 Slide

An example of a slide movement is shown in Figure 111 below, where the Kinect is positioned facing the participant. A line is drawn on the floor, parallel to the Kinect's X axis, and the participant is asked to perform the slide along the line. For this test, as the participant faces the Kinect during the whole movement, the movement displacement occurs along the X axis.

Figure 111. The sequence of images to represent a standardized slide movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

Note in Figure 111 that the Y coordinate of the head, i.e. the distance from the floor to the head, draws a peak along time. Therefore, each slide phase is characterized by the period between two head peaks. After completing four slides from the sensor's left to the right, the participant returns to the origin of the movement, sliding four times from the right to the left of the sensor.

<u>Criterion 1: Body is turned sideways so shoulders remain aligned with the line on the floor.</u>

As mentioned before, the movement is performed along the X coordinate of the Kinect sensor, and in this way the joints displacement is done along X axis. Figure 112 shows a diagram illustrating the top-view position of the joints and the Kinect sensor for the slide test. In the figure, J1 represents the right shoulder joint, J2 the head joint and J3 the left shoulder joint.

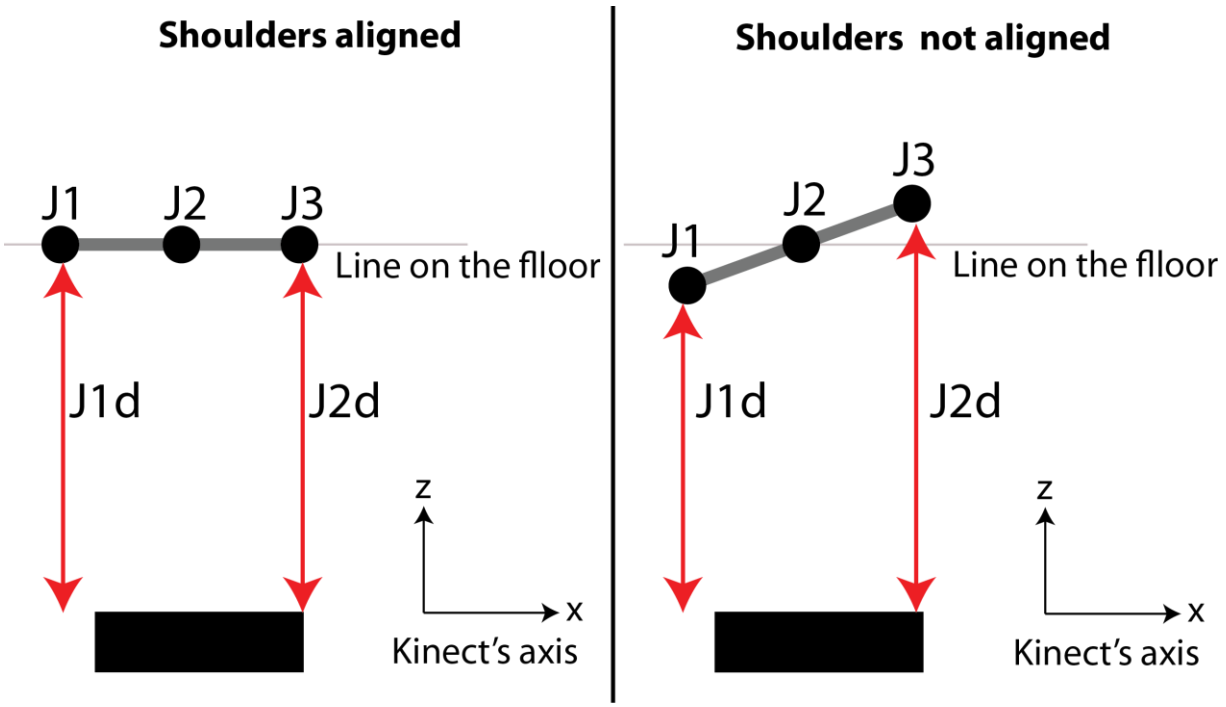

Figure 112. Diagram of the shoulders aligned and not aligned with line on the floor.

Note in the figure that when the participant keeps the shoulders aligned with the line on the floor, the distances on the Z axis between each shoulder and the sensor are approximately the same, since the line is parallel to the Kinect's X axis. On the other hand, if the participant does not keep his/her shoulders aligned to the line, one of the shoulders will have a greater distance from the sensor than from the other one. Considering this, to analyze the first criterion, the algorithm checks the difference between the Z coordinate of the shoulders from the first head peak to the last one; if the shoulders remain aligned, the difference between them should be very close to zero. The algorithm then checks to see if the difference between them during the whole movement is less than 10 cm. It also allows for a tolerance of 20% of occurrences to be outside of this range. Note in Figure 113 that during the whole movement, the difference between them did not exceed 10 cm.

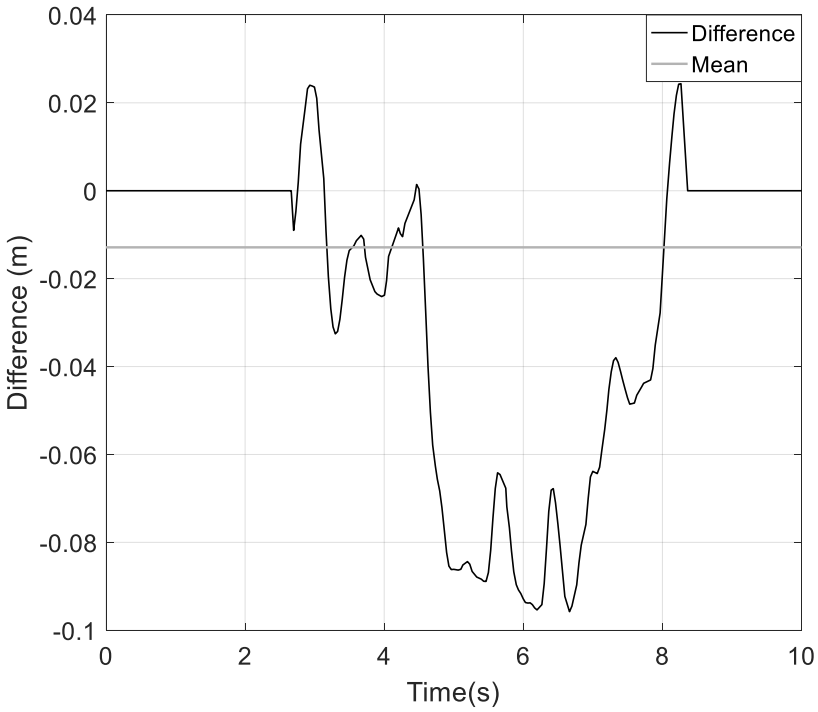

Figure 113. Variation during time of difference between shoulders along Z coordinate and its average.

The flowchart that illustrates how to check if the shoulders are aligned is:

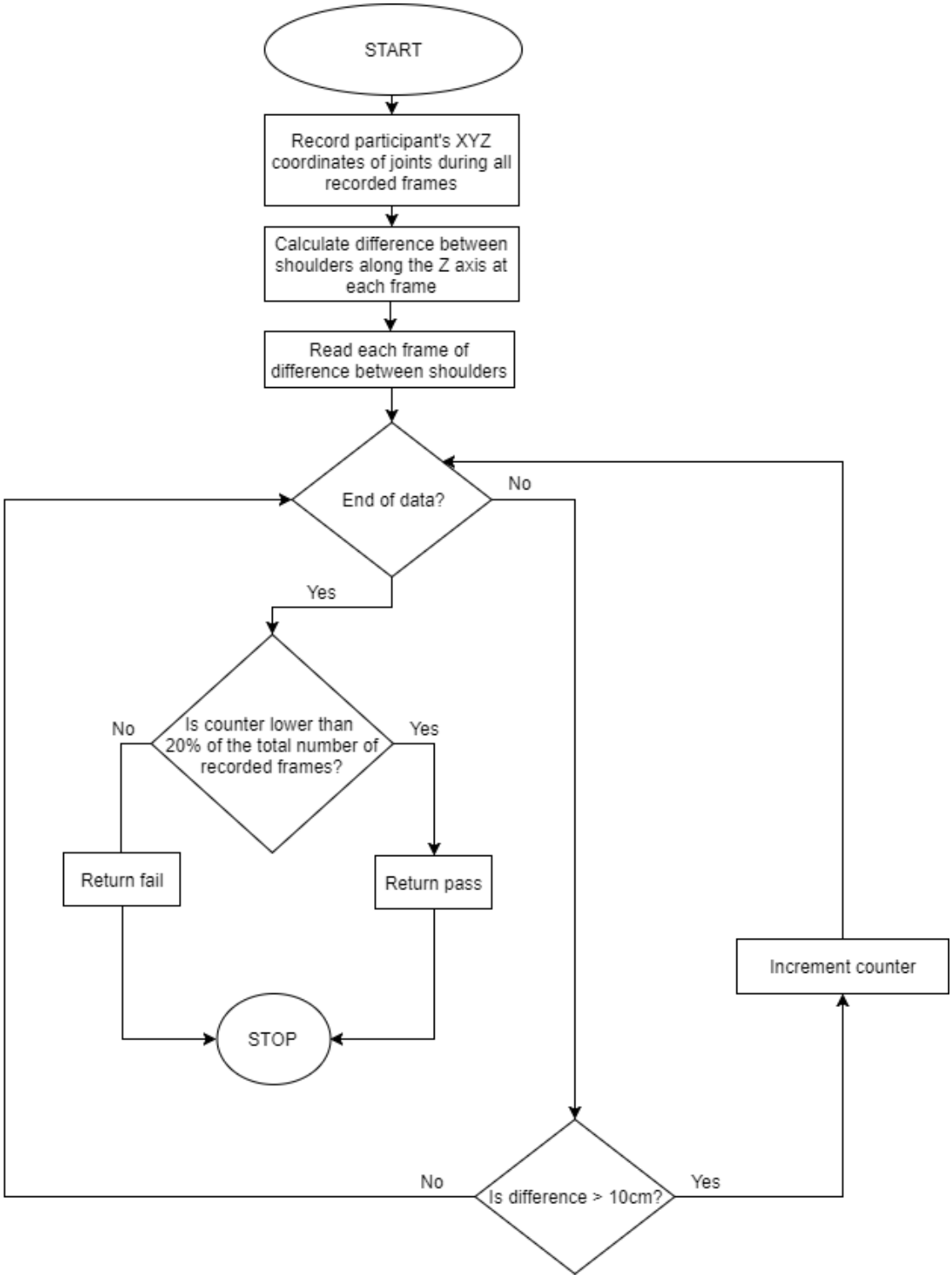

Figure 114. Algorithm to determine whether participant passes or fails the first criterion.

Criterion 2: A step sideways with the lead foot followed by a slide with the trailing foot where both feet come off the surface briefly.

In order to analyze the second criterion, it is necessary to detect on which side the user starts sliding. According to the Kinect's coordinate map, if the participant is standing facing the Kinect, movements to the sensor's right side return positive values for X and movements to the sensor's left side return negative values. Figure 115 shows the behavior of the X coordinate of the head when the participant starts sliding to the right side.

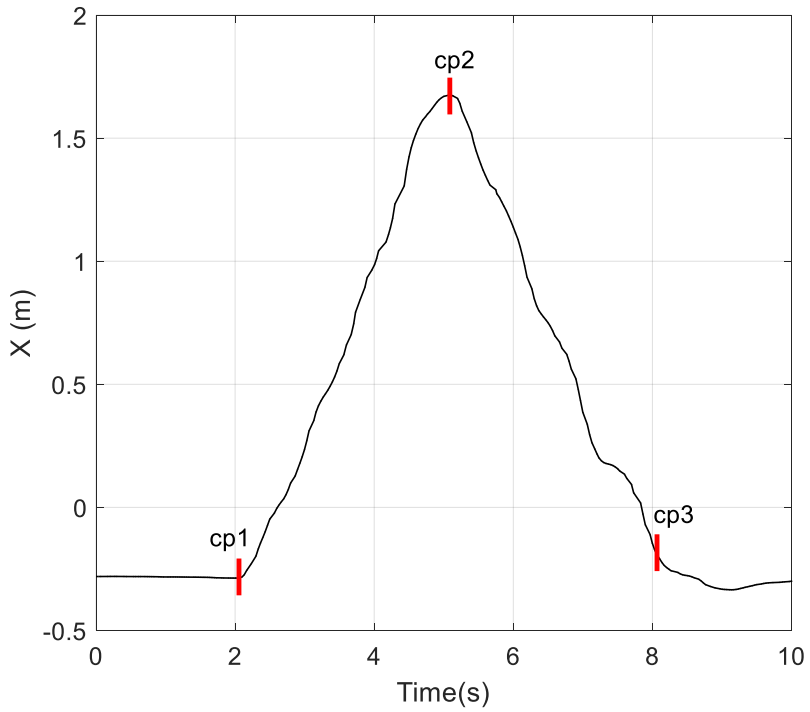

Figure 115. Variation along time of X coordinates of head and critical points.

The movement consists of four slides to the preferred side and then four slides to the non-preferred side, returning to the initial position. Note that the participant starts moving to the right at critical point 1 ("cp1"). In this way, the x coordinate increases until s/he completes four slides to the preferred side at critical point 2 ("cp2"). Then, s/he starts sliding to the non-preferred side, i.e. to

the left side, until reaching the initial position at critical point 3 (“cp3”). Note that the behavior of the head during the moment draws a peak on the X coordinate. Therefore, by checking if the peak of the head on the graph is a maximum or a minimum point, the algorithm can determine the participant’s preferred side. If the participant starts sliding to the right, the peak will be a maximum point, otherwise it will be a minimum point.

After determining the lead side, the algorithm has to check for a step sideways with the lead foot followed by a slide with the trailing foot. By analyzing the behavior of the distance between the feet along time, note that as the slide is performed along the X axis, initially the feet will be apart from each other in a constant value as the participant is standing waiting to begin the test. As s/he starts sliding, the participant’s preferred foot will start the step and consequently the distance from the non-stepping foot increases until reaching a maximum distance, moment when the first step is complete. Figure 116 illustrates the difference between the feet along the X axis, as well as the moment when the difference between the feet is equal or greater than 20% of the initial value, i.e. the moment when the algorithm considers that the first step begins (“cp1”).

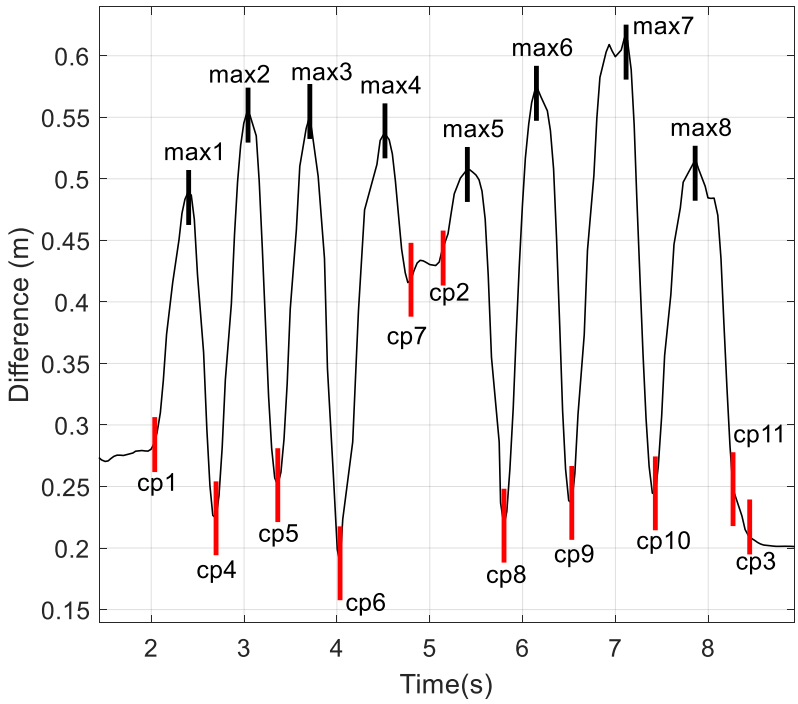

Figure 116. The distance between feet along X axis and critical points.

Each slide phase, determined by the head peaks, is represented by “cp4” to “cp11”, where “cp2” represents the moment when the participant starts sliding to the non-preferred side and “cp3” represents the moment when the movement is completed.

The flowcharts that show how to check when the movement starts and when it changes sides are:

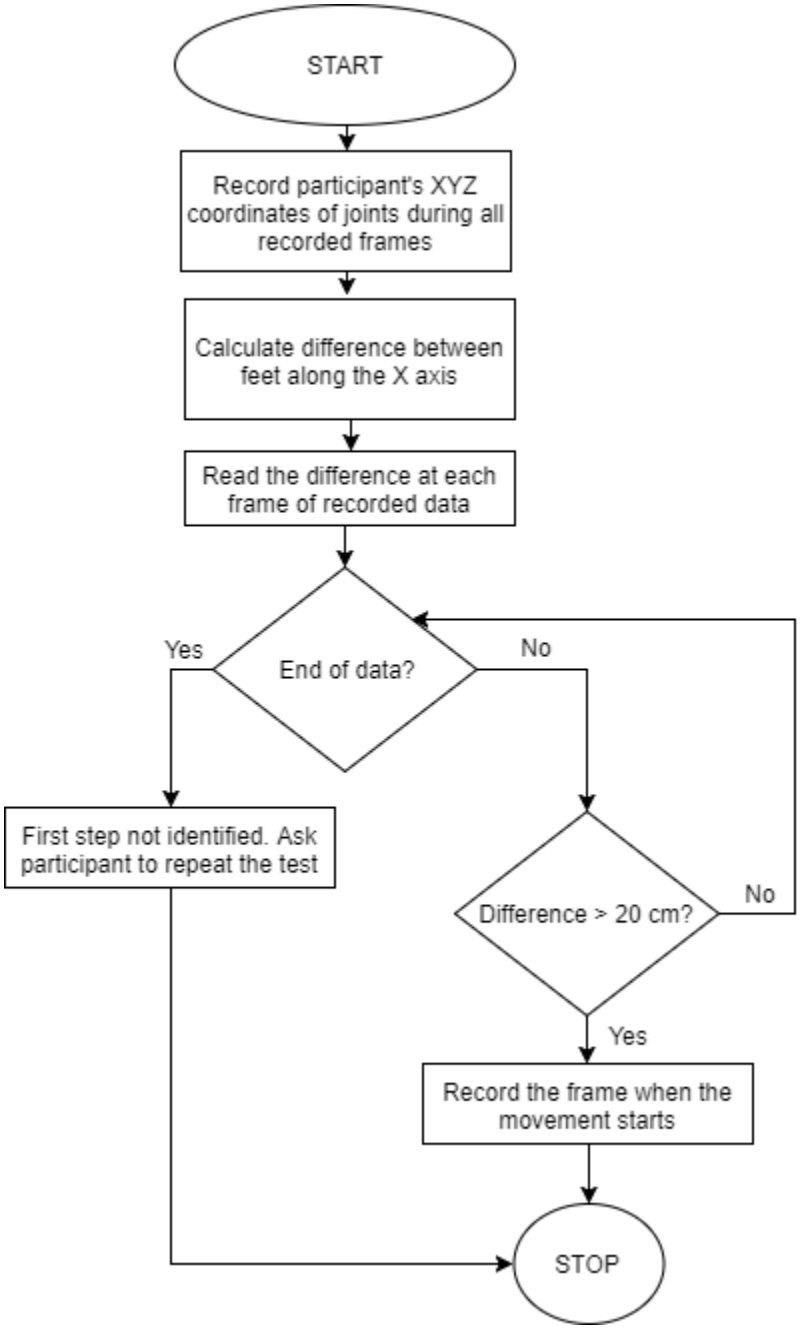

Figure 117. Algorithm to determine when movement starts.

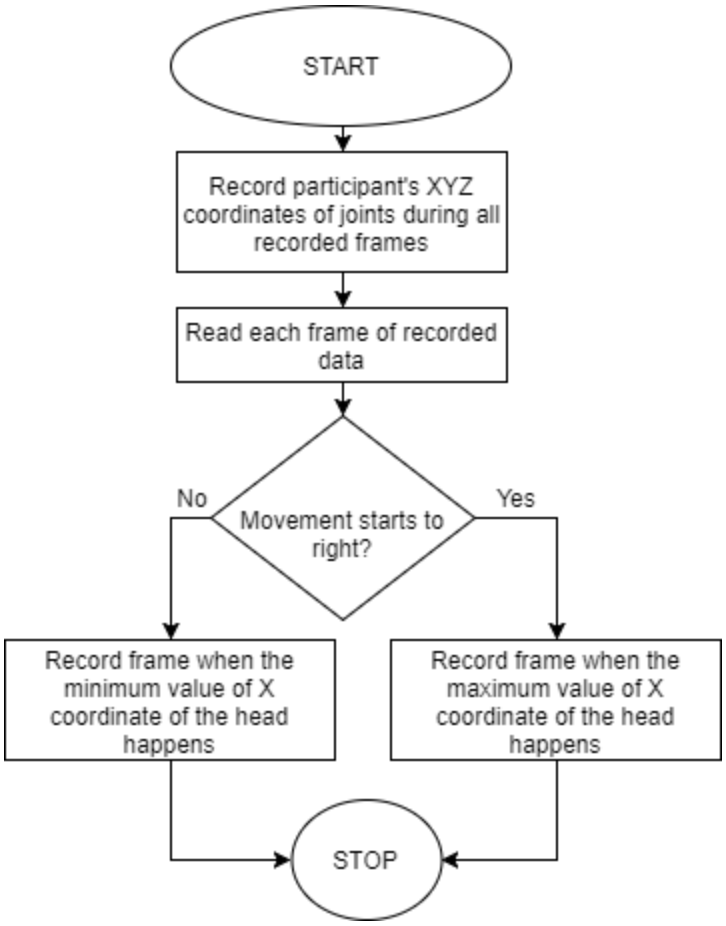

Figure 118. Algorithm to determine when movement changes sides.

Note that as a step has to happen for each slide, in between two consecutive slides, the distance between the feet reaches a maximum value within that period. As was done for the previous tests, the algorithm has to check to see if while the stepping foot is performing the step, the non-stepping foot is placed on the floor without moving. Figure 119 illustrates this situation, where “s1” to “s8” stands for “step 1” to “step 8”.

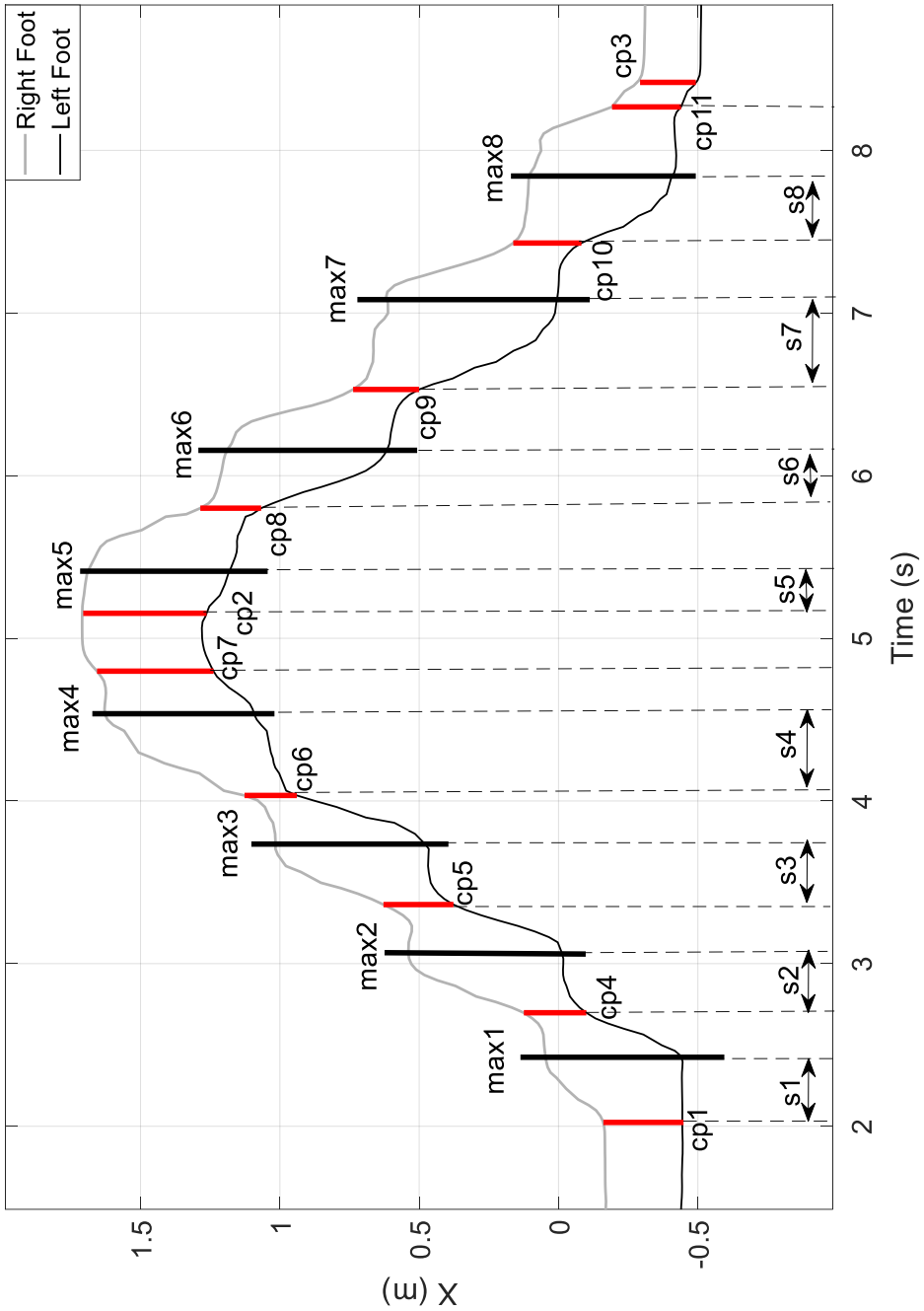

Figure 119. Variation along time of X coordinates of feet and the moment when each step occurs. The red traces represent the critical points and the black traces represent the moment when the distance between the feet reaches the maximum value in between two consecutive critical points.

In order to check if the participant performs a step sideways, the algorithm checks the X coordinate of the non-stepping foot and the stepping foot at the beginning and at the end of the step. If the difference between initial and final values of the non-stepping foot is lower than the difference between the initial and final values of the stepping foot, for all steps, the algorithm considers that the participant has performed a step with the leading foot, i.e. it basically considers the slope of the curve. Note from the picture that the foot performing the step changes at “cp2” as the participant starts sliding to the other side after that critical point.

In addition, in order to check if a slide with the trailing foot occurs, the algorithm checks to see if after the step the trailing foot is the one moving, while the foot that performed the step barely moves. The same idea that was used to check the step is applied to the slide with the trailing foot, however in a different interval, as illustrated in Figure 120. In the figure, “S1” to “S8” starts for “slide1” to “slide8”.

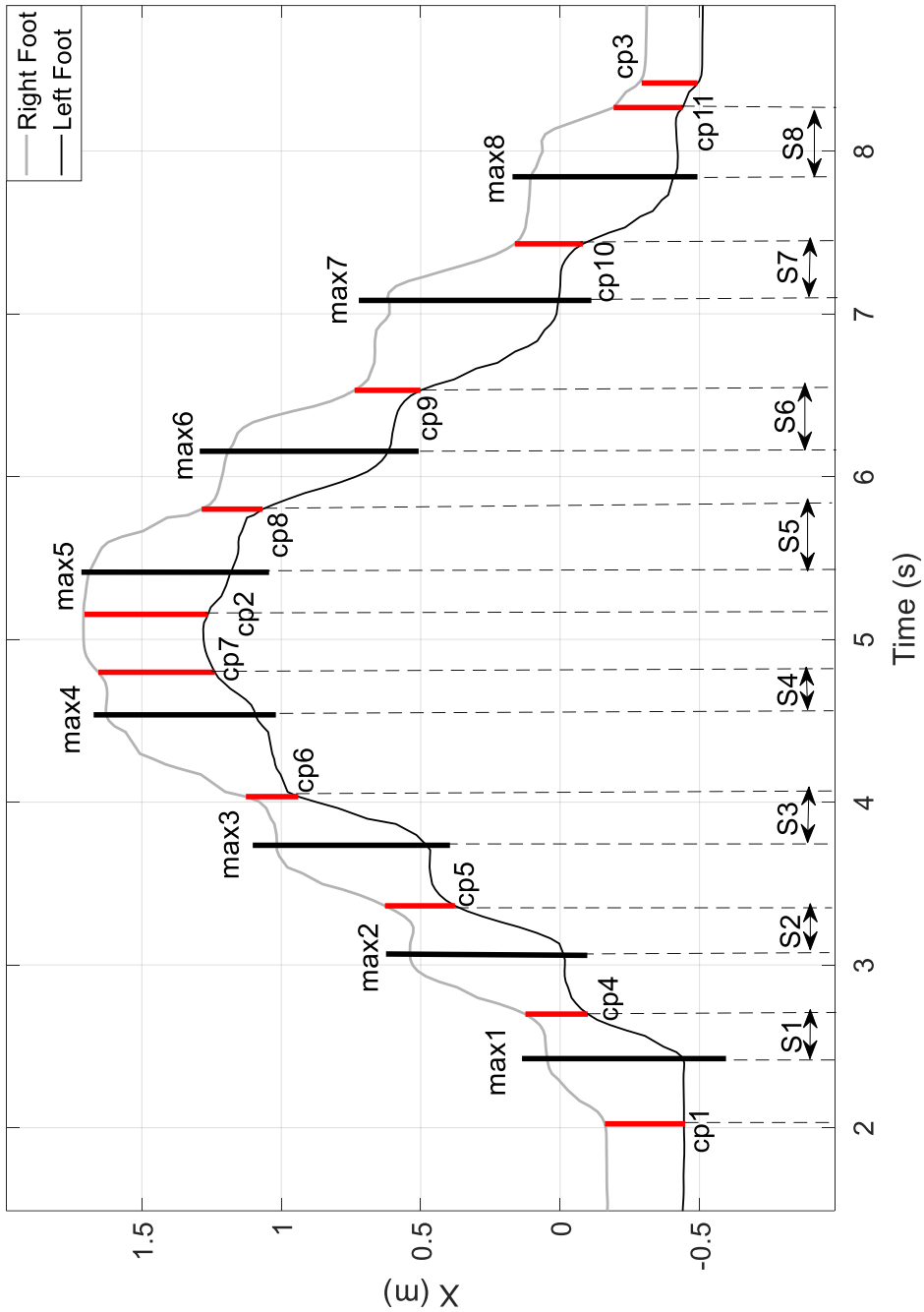

Figure 120. Variation along time of the X coordinates of feet and the moment when each slide occurs. The red traces represent the critical points and the black traces represent the moment when

the distance between the feet reaches the maximum value in between two consecutive critical points.

The following flowcharts illustrates how to determine if the participant steps sideways with leading foot followed by slide with trailing foot:

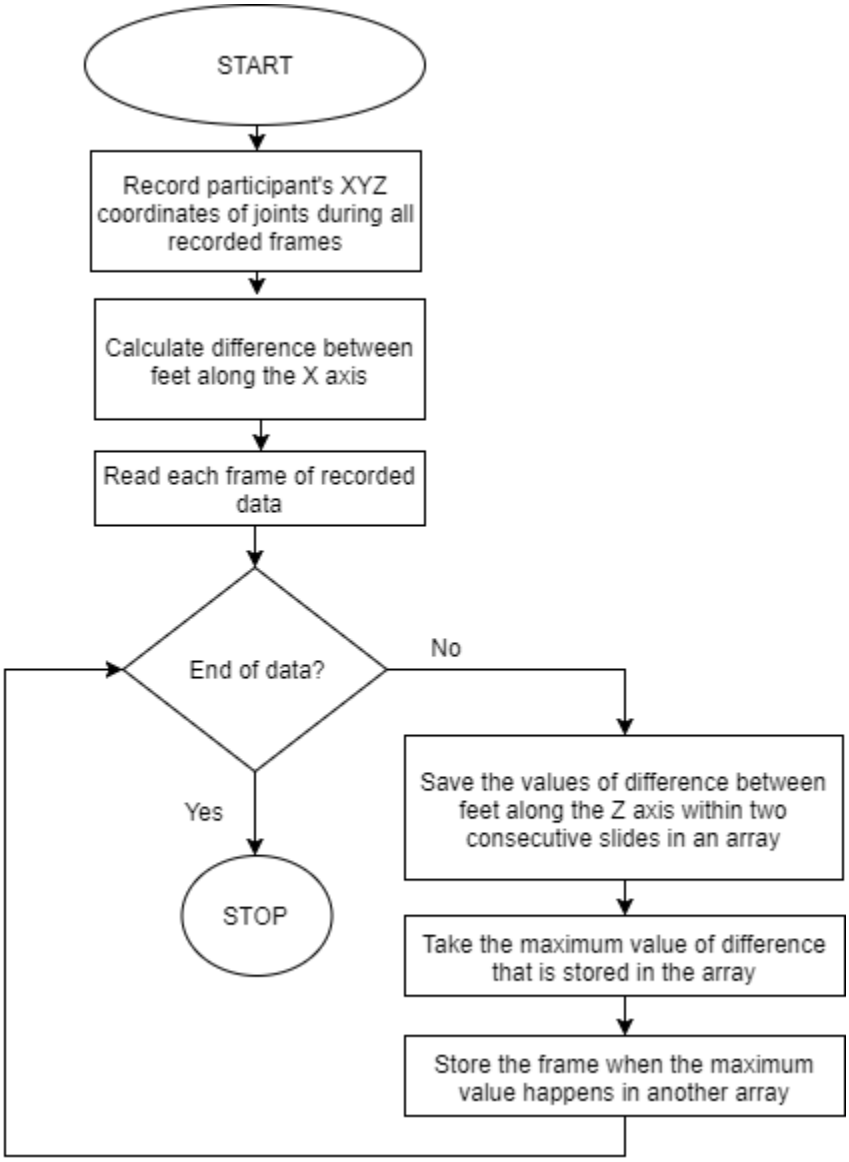

Figure 121. Algorithm to determine the maximum distance between feet within two consecutive slides.

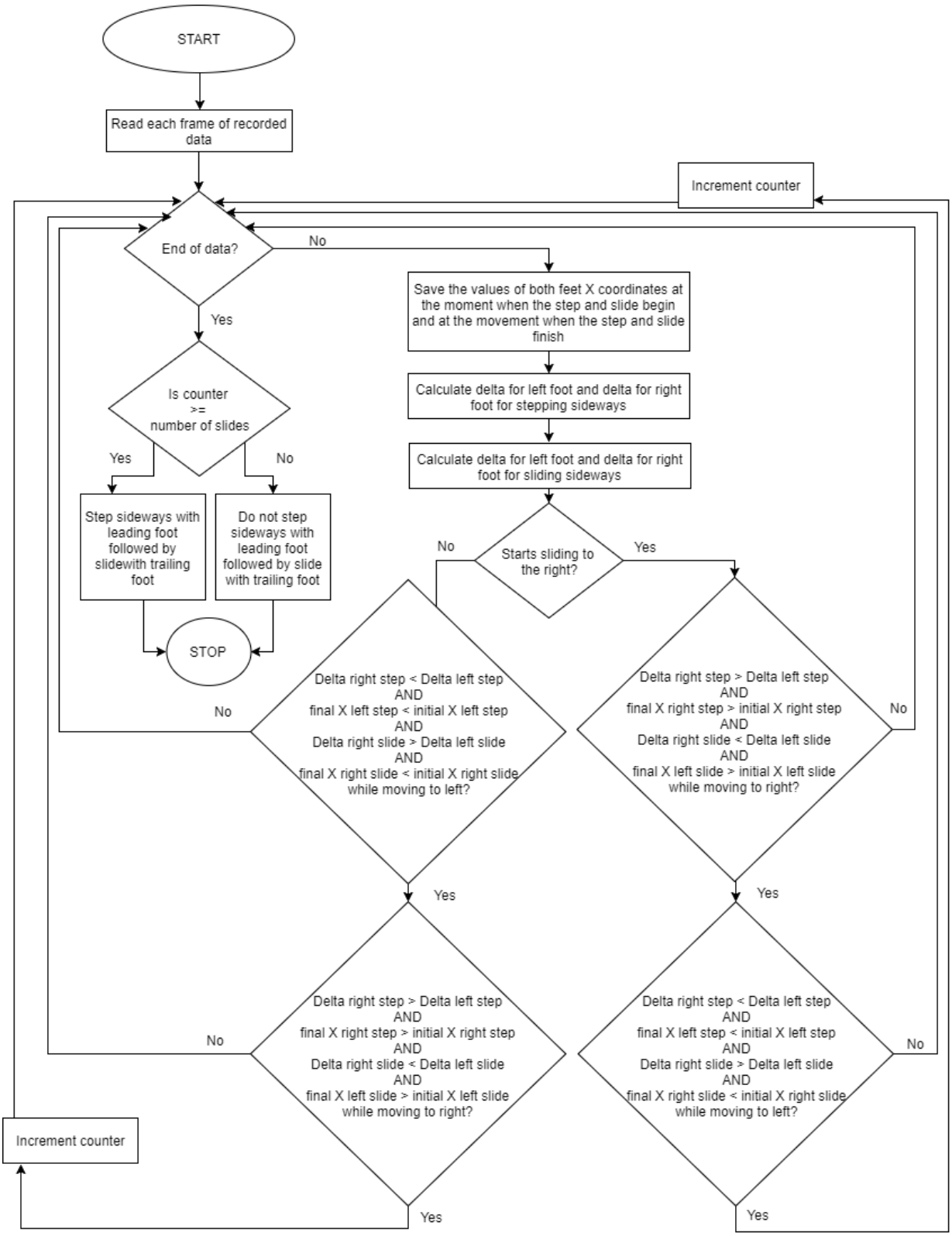

Figure 122. Algorithm to determine if participant steps sideways with lead foot followed by slide with trailing foot.

Finally, to check if both feet come off the surface briefly, as was done for some previous tests, the algorithm that identifies peaks is used to save the index of all frames when the participant is off the surface. Then, for each of these frames, the algorithm checks if both Y coordinates of the knees are at least 10% greater than the Y coordinate acquired during the calibration. This value from the calibration represents the distance between knees and floor when the participant is standing on the floor level, and 10% above the standing level is the threshold considered to be off the surface. Figure 123 shows the Y coordinate of both knees and head, as well as the standing level and the threshold. It is clear from Figure 123 that both knees are off surface when the peaks occur; the same logic used for the third criterion of the gallop test was applied to assess this criterion.

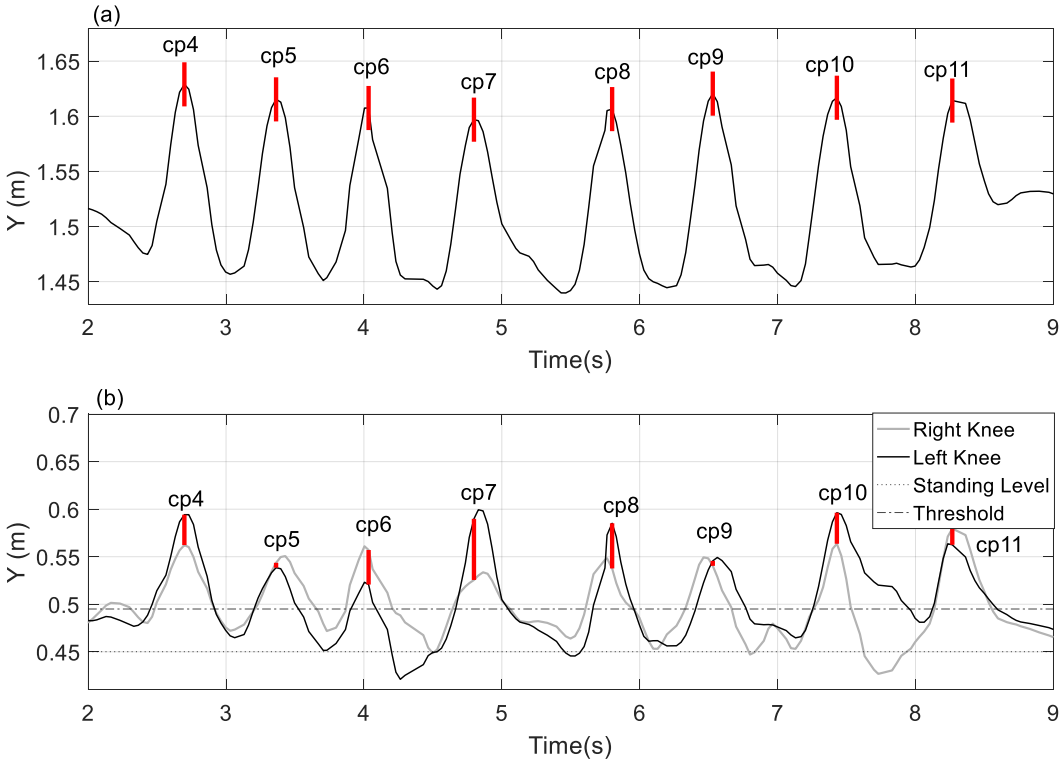

Figure 123. (a) Variation along time of the Y coordinates of head and head peaks. (b) Variation along time of the Y coordinates of knees and moment when each head peak occurs.

The flowchart that illustrates how to check if both feet leave the floor is:

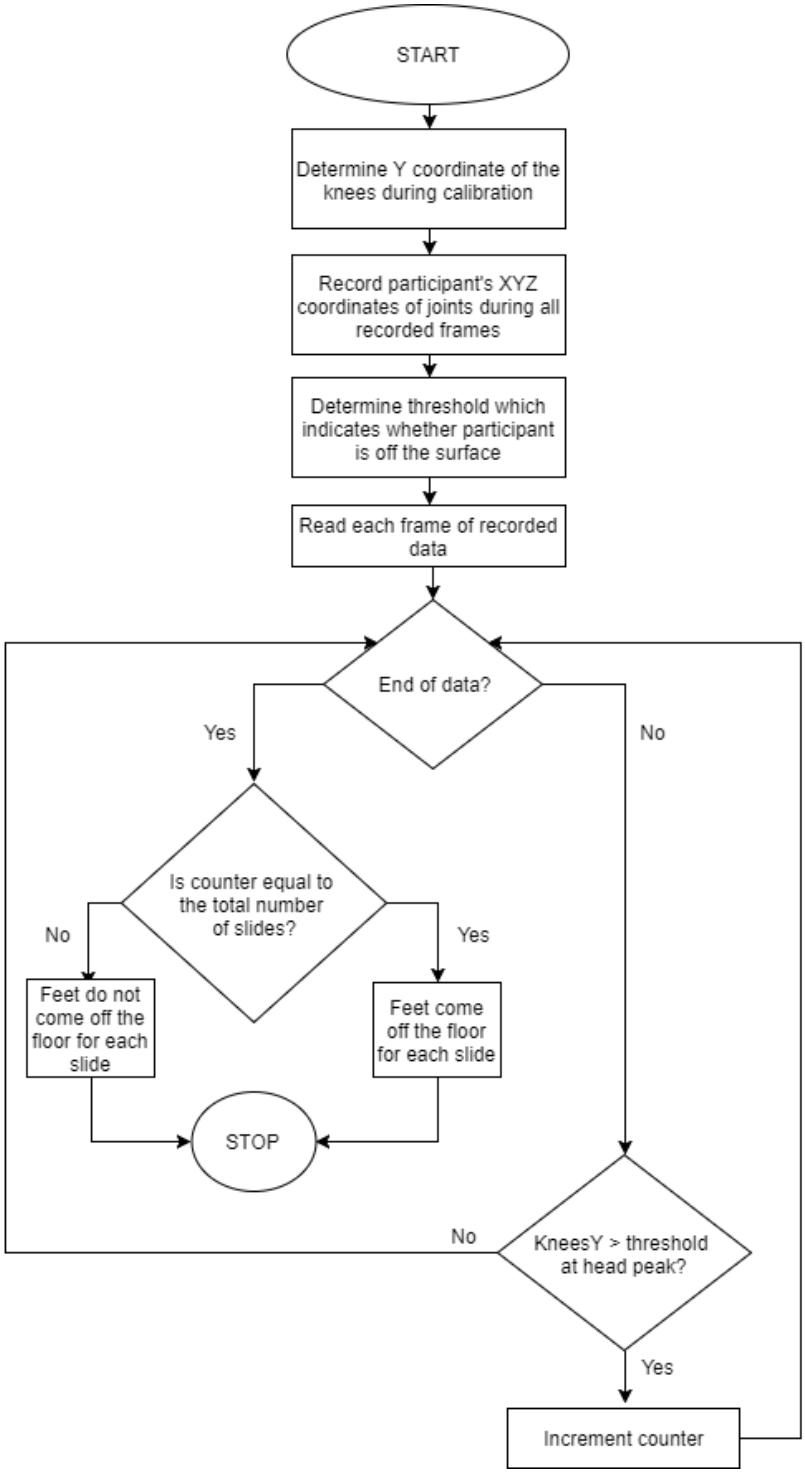

Figure 124. Algorithm to determine whether both feet come off the surface.

The following flowchart illustrates the feedback for the second criterion:

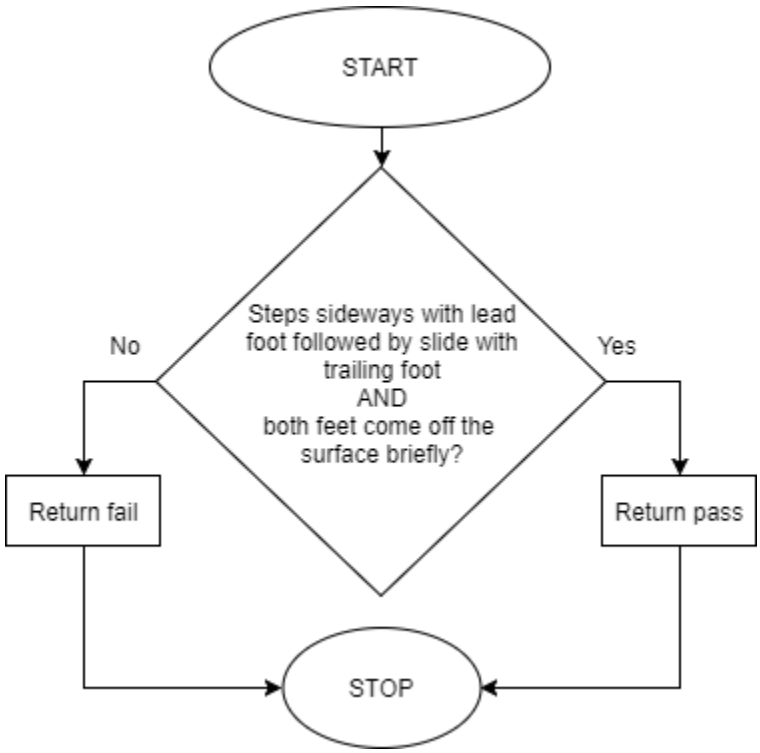

Figure 125. Algorithm to return feedback for second criterion.

<u>Criterion 3 and 4: 4 continuous slides to the preferred and 4 continuous slides to the non-preferred side.</u>

The third and fourth criteria check if the participant slides four times to the preferred side and four to the non-preferred side. As shown in Figure 126, the participant reaches the most extreme distance to the preferred side and then returns to the initial position sliding to the non-preferred side. By checking the frame when the peak occurs for the X coordinate of the head, i.e. the frame when the participant changes sides, the algorithm divides the whole movement into two parts: movement to the left and movement to the right. Then, using the algorithm to detect the head peaks in the Y coordinate, the logic counts how many peaks happened during the movement to the right

and how many peaks happened during the movement to the left. If four of them occur during the right phase and four of them occur during the left phase, the algorithm returns a pass or a fail accordingly.

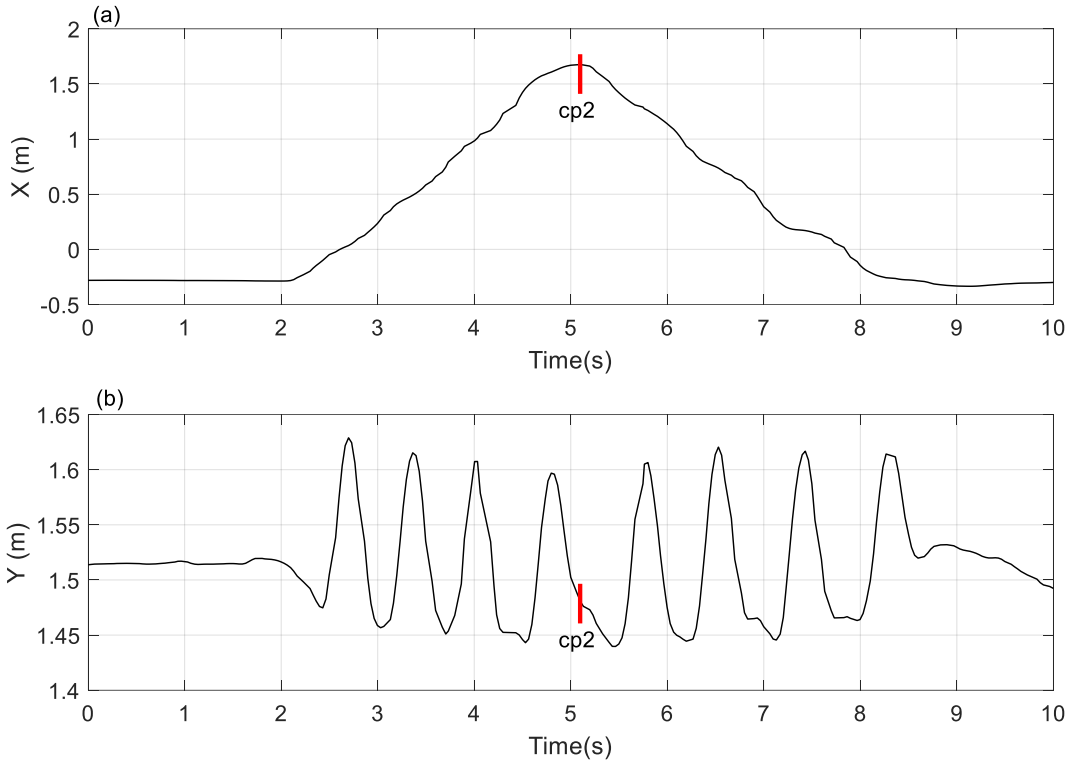

Figure 126. (a) Variation along time of the X coordinates of head and “cp2”. Variation along time of the Y coordinates of head and “cp2”.

The flowchart that illustrates how to check this is:

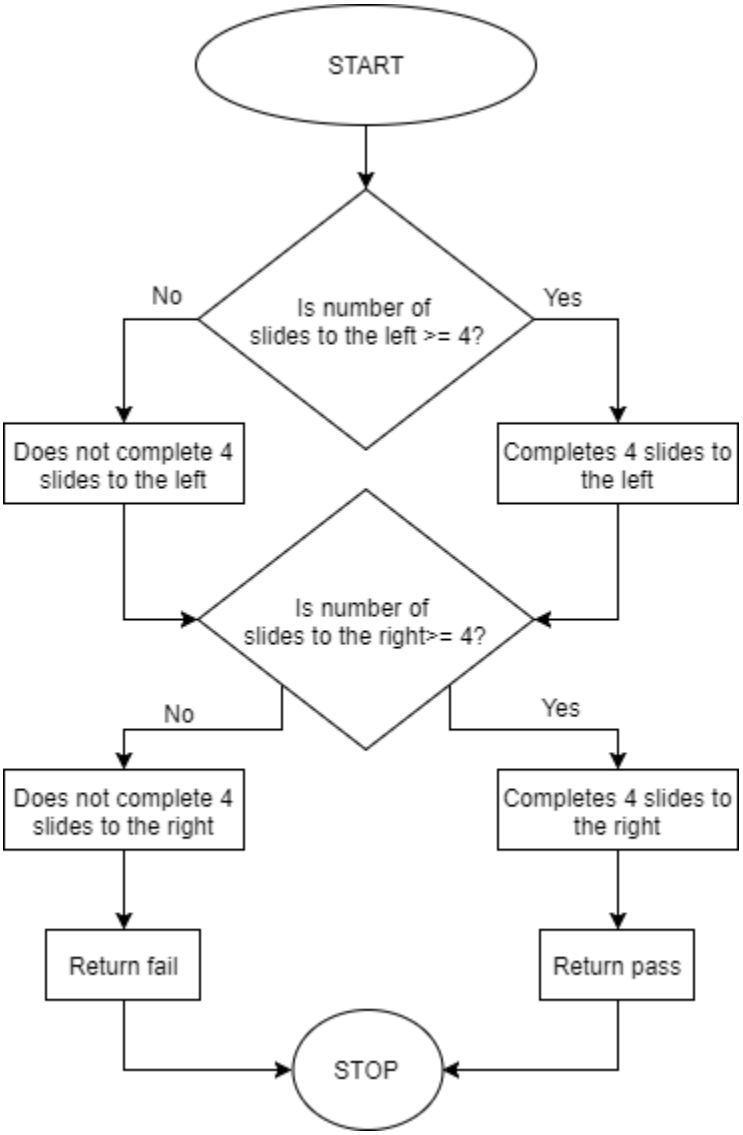

Figure 127. Algorithm to evaluate whether participant passes third and fourth criteria.

4.3.7 Two-hand strike of a stationary ball

An example of a two-hand strike movement is shown in Figure 128 below, where the Kinect is positioned facing the participant.

Figure 128. The sequence of images to represent a standardized two-hand strike of a stationary ball movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

Note that initially the participant will be holding the bat with his/her preferred hand above their non-preferred hand. As s/he starts the movement, prior to the strike, her/his trunk turns to side, away from the ball. For the algorithm's perspective, the movement actually starts when the participant straightens his/her elbows behind the trunk holding the bat. This moment is characterized by the stage when both hands reach the furthest distance from the Kinect, which is represented by the first critical point, "cp1". Prior to the analysis of the movement, the participant is asked to state his/her preferred hand and the answer is given as an input to the algorithm. Figure 129 illustrates an example of the behavior of the Z coordinates of the hands during the entire movement when the participant strikes the ball, where the right hand is the preferred one.

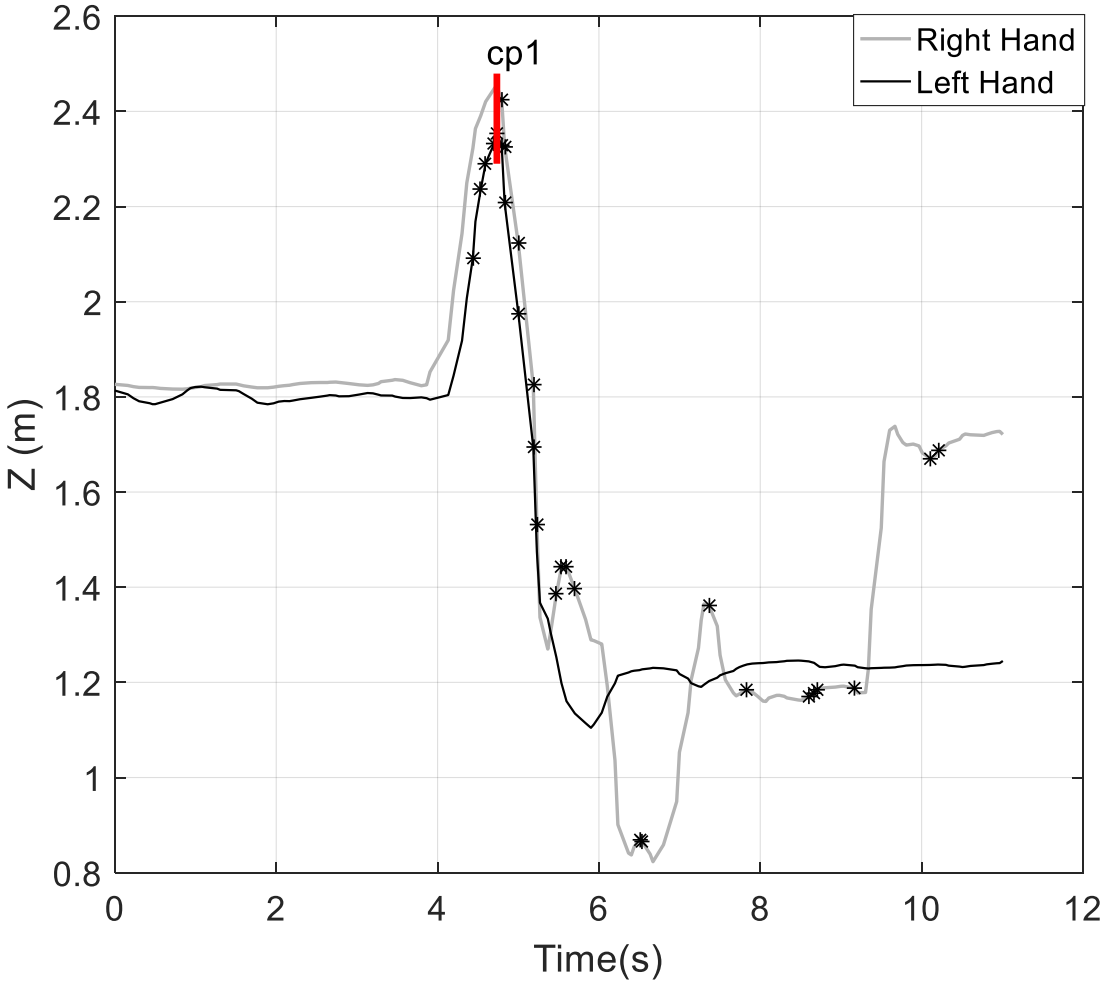

Figure 129. Variation along time of the Z coordinates of hands and "cp1". The (*) represents the moment when the 3D position of the joint was inferred.

The algorithm knows which hand is the preferred according to the participant's answer, so it checks the maximum value of the Z coordinate of that hand to determine "cp1", as shown in the figure.

The following flowchart checks when “cp1” happens:

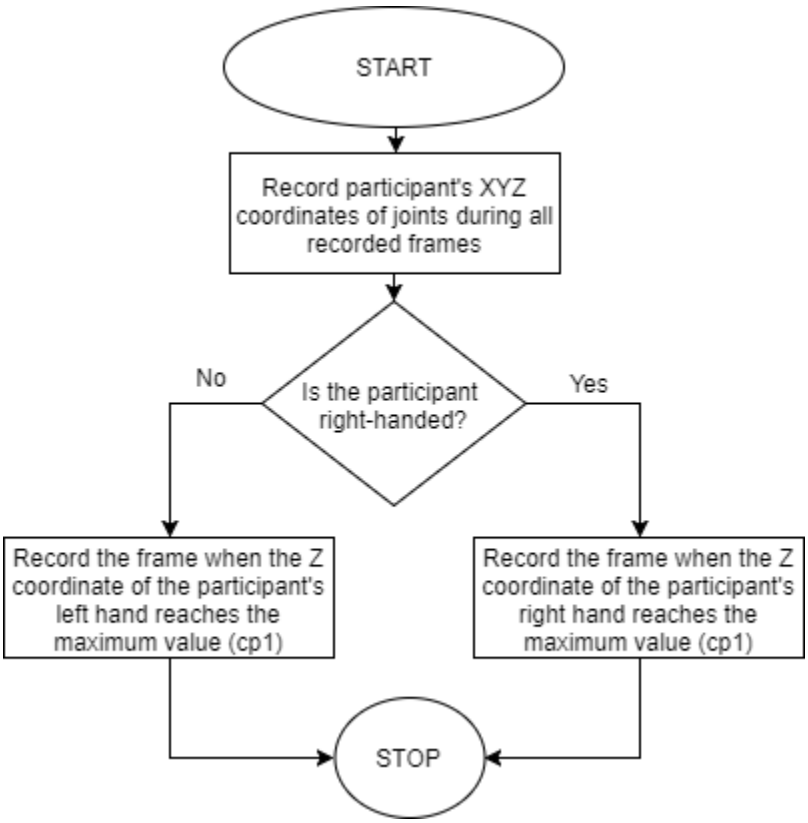

Figure 130. Algorithm to determine cp1.

<u>Criterion 1: Child’s preferred hand grips bat above non-preferred hand.</u>

The first criterion consists of checking if child’s preferred hand grips the bat above his/her non-preferred hand. Figure 128 shows an example of a two-hand strike, where the participant uses the right hand as the preferred one. Note that during the movement, the participant has to transfer his/her weight back prior to strike the ball. During this process, both hands reach behind the trunk and the preferred hand is the one that reaches the furthest distance from the sensor, since it grips the bat above non-preferred hand. Additionally, note that at frame 8, the preferred hand, which in this example is the right one, is further away from the sensor than the left hand.

Consequently, in order to evaluate whether the preferred hand grips bat above non-preferred hand, the algorithm checks if at “cp1” the preferred hand is further away from the Kinect than the non-preferred one, as illustrated in the following flowchart:

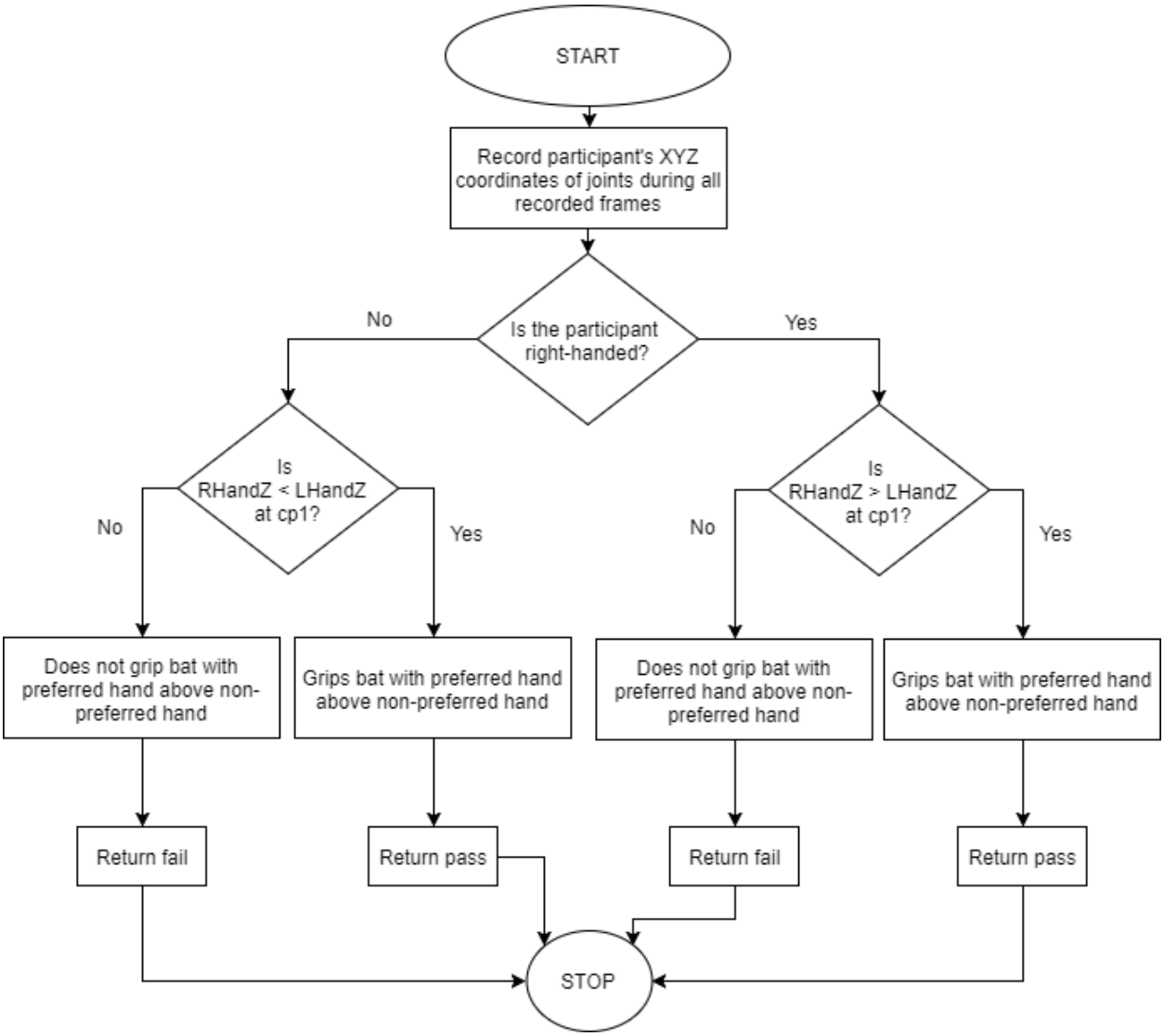

Figure 131. Algorithm to determine whether participant passes or fails the first criterion.

Note in Figure 129 that the preferred hand (the right one) is further away from the Kinect than the non-preferred one, which indicates that the participant is gripping the bat correctly.

Criterion 2: Child's non-preferred hip/shoulder faces straight ahead.

The second criterion checks if child's non-preferred hip/shoulder faces straight ahead, i.e. faces the Kinect sensor. By knowing the child's preferred side, the algorithm checks if at "cp1" the Z coordinates of non-preferred shoulder and non-preferred hip are lower than the respective preferred side; it must be lower than at least half of the distance between them acquired during the calibration. Note in Figure 128 that at "cp1" (e.g. at frame 8), the non-preferred hip and shoulder are closer to the Kinect, i.e. they face straight ahead — the same idea illustrated in Figure 112.

Figure 132 shows the Z coordinate of shoulders and hips at "cp1" when the participant performs the moment with his/her preferred right side.

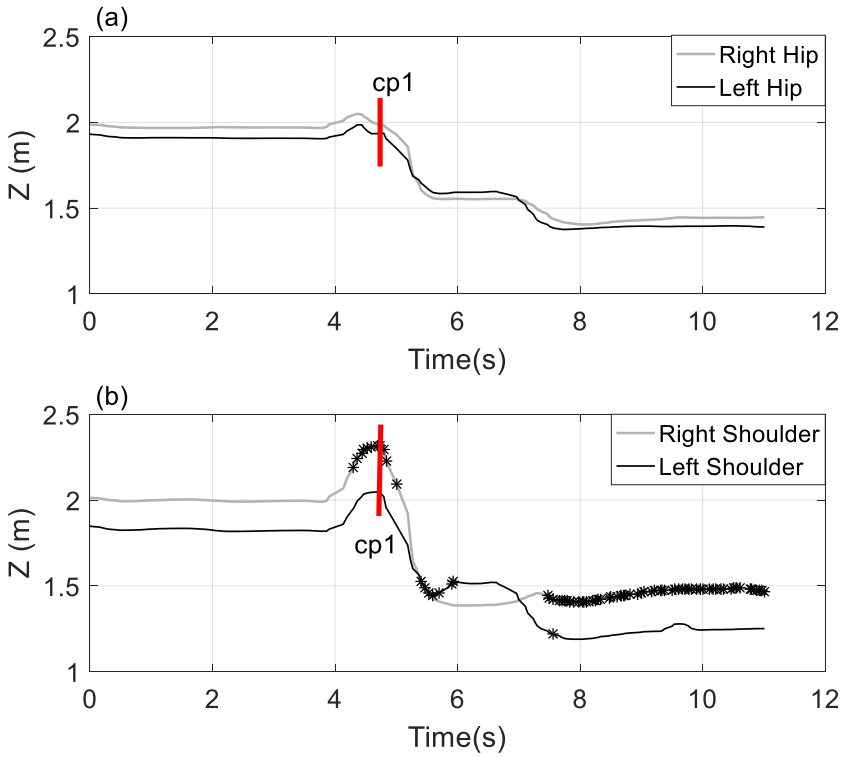

Figure 132. (a) Variation along time of the Z coordinates of hips. (b) Variation along time of the Z coordinates of shoulders. The (*) represents the moment when the 3D position of the joint was inferred.

Note that the left hip has a lower Z coordinate than the right hip at “cp1”, and the left shoulder has a lower Z coordinate than the right shoulder at “cp1”. This means that at “cp1” the non-preferred hip/shoulder faces the sensor.

Therefore, to assess the second criterion, the algorithm checks whether at “cp1” the non-preferred hip and shoulder are closer to the sensor than the respective preferred ones. If this condition is true, the algorithm returns “pass”.

The following flowchart illustrates this:

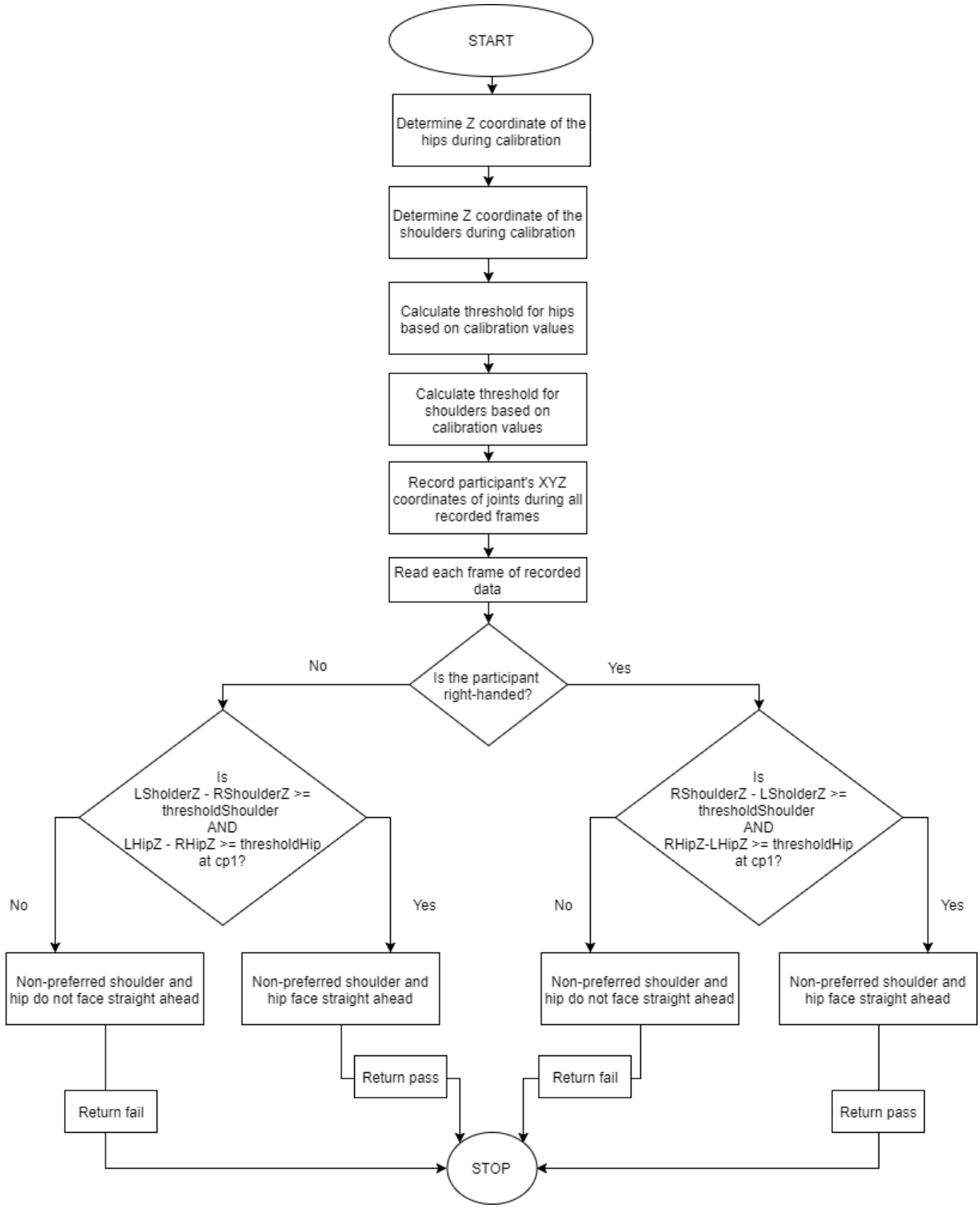

Figure 133. Algorithm to determine whether participant passes or fails the second criterion.

Criterion 3: Hip and shoulder rotate and de-rotate during swing.

Figure 134 represents an example of an ideal rotation. In the figure, J2 represents the non-preferred joint (hip or shoulder) and J1 represents the preferred joint.

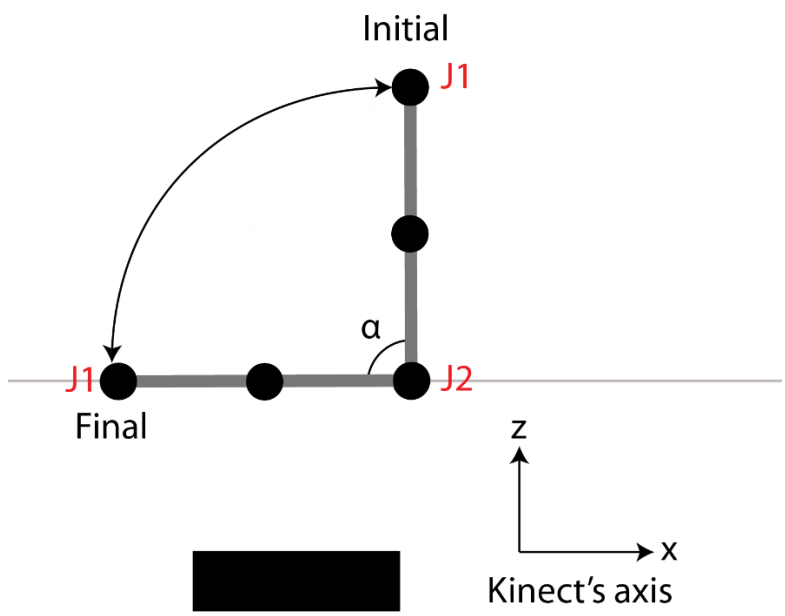

Figure 134. Example of rotation.

Note that initially the angle α in the XZ plane is equal to zero, since the preferred shoulder/hip is aligned with the non-preferred shoulder/hip along Z axis, where the non-preferred one is facing the Kinect. As the participant starts rotating the shoulder/hip, angle α increases towards 90° until the preferred shoulder/hip is aligned with the non-preferred shoulder/hip along X axis. When the participant starts de-rotating the shoulder/hip, angle α decreases towards 0° until the preferred shoulder/hip is again aligned with the non-preferred shoulder/hip along Z axis.

In an ideal situation, angle α changes from 0° and 90°; however, as the participant may not perform a perfect rotation, to identify if a rotation happens, it is enough to check whether the angle α increases and then decreases within a period when a rotation is expected to happen.

Therefore, in order to check if a rotation and a de-rotation happens, the algorithm checks if between "cp1" and the end of the movement, angle α increases at least 45° and then decreases at least 45°. Angle α can be calculated as follows:

$$\alpha_{shoulders} = \tan^{-1}\left(\frac{Right\ Shoulder\ X - Left\ Shoulder\ X}{Right\ Shoulder\ Z - Left\ Shoulder\ Z}\right) \tag{14}$$

$$\alpha_{hips} = \tan^{-1}\left(\frac{Right\ Hip\ X - Left\ Hip\ X}{Right\ Hip\ Z - Left\ Hip\ Z}\right) \tag{15}$$

Figure 135 shows angle alpha for the shoulders and hips from "cp1" until the end of the movement.

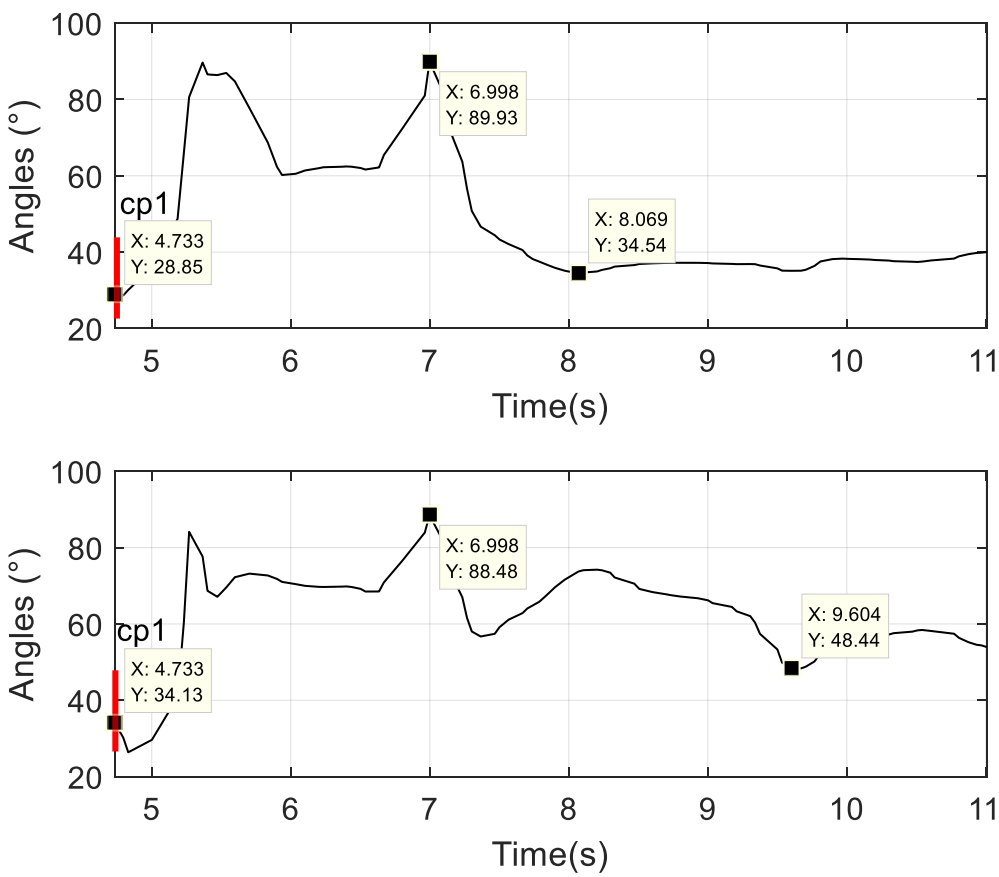

Figure 135. (a) Variation along time of angle alpha of the shoulders and "cp1". (b) Variation along time of the alpha angles of hips and "cp1".

Note that within this interval, the alpha angles increase by at least 40° and then decrease again by at least 40°. This behavior means that the participant rotated and then de-rotated his/her shoulders and hips.

The flowchart that illustrates how to check if the participant rotates and then de-rotates his/her hips and shoulders is:

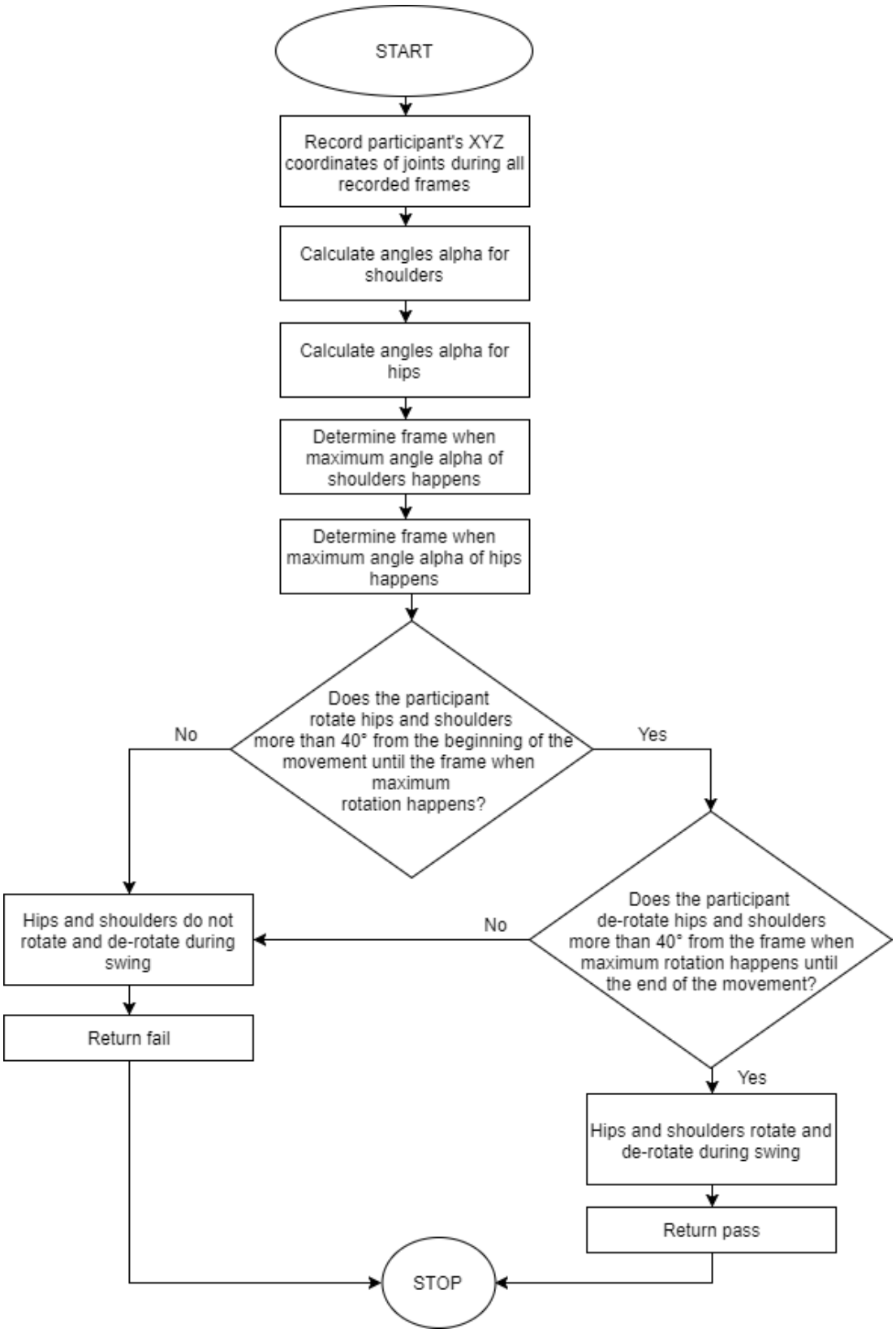

Figure 136. Algorithm to determine whether participant passes or fails the third criterion.

Criterion 4: Steps toward ball with non-preferred foot.

The fourth criterion checks if the participant steps toward the ball with his/her non-preferred foot. In order to check this, it is necessary to determine when the ball is struck. This moment can be estimated using the instant when the ball starts moving, which is the second critical point "cp2", as shown in Figure 137.

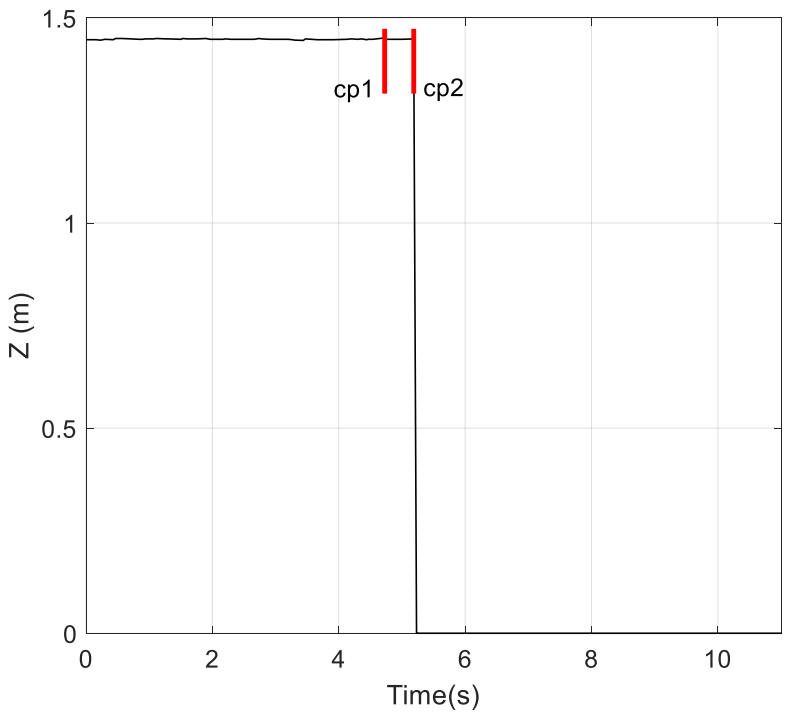

Figure 137. Variation along time of the Z coordinates of ball.

Figure 138 shows the behavior of feet within the period that goes from "cp1" to "cp2".

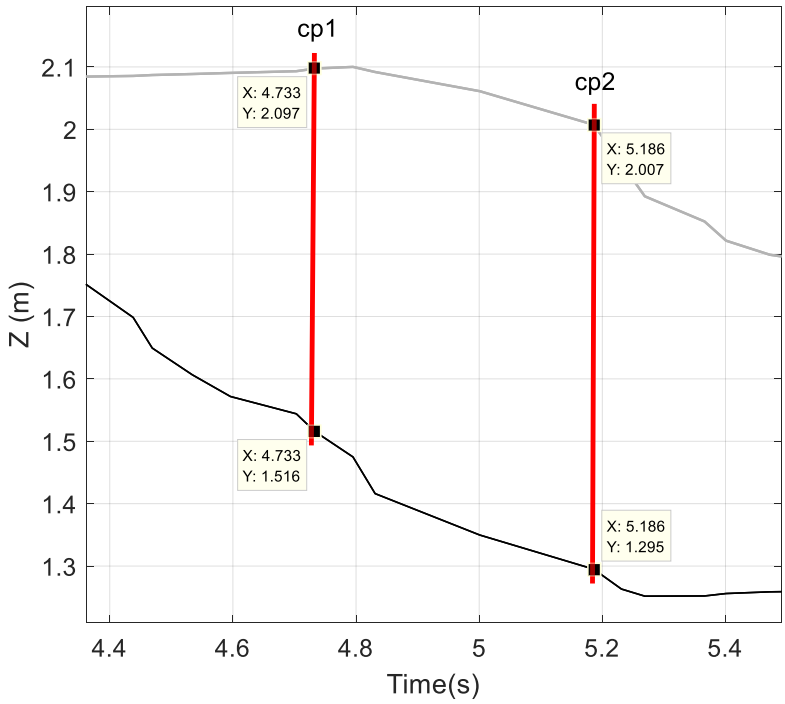

Figure 138. Variation along time between “cp1” and “cp2” of Z coordinates of feet. The black curve represents the left foot and the grey curve represents the right foot. The (*) represents the moment when the 3D position of the joint was inferred.

Note that the step occurs in this interval, since the participant must step towards the ball prior to the strike. Note that the slop of the curve of the non-preferred foot — in this example, the left one, is sharper than the slop of the preferred foot within this interval. This means that while the non-preferred foot performs the step, the preferred one stays placed on the floor. Also note that the position of the non-preferred foot at “cp2” is lower than the position at “cp1”, which means that the step was performed towards the ball, which is placed on the tee in front of the participant’s body.

Therefore, to assess this criterion, the algorithm checks the slope of the curve of both feet within “cp1” and “cp2”, and then compares which one is sharper. It also checks to see if the step was

performed towards the ball by analyzing the Z coordinate of the stepping foot at the beginning and at the end of the step. The flowchart that illustrates this idea is:

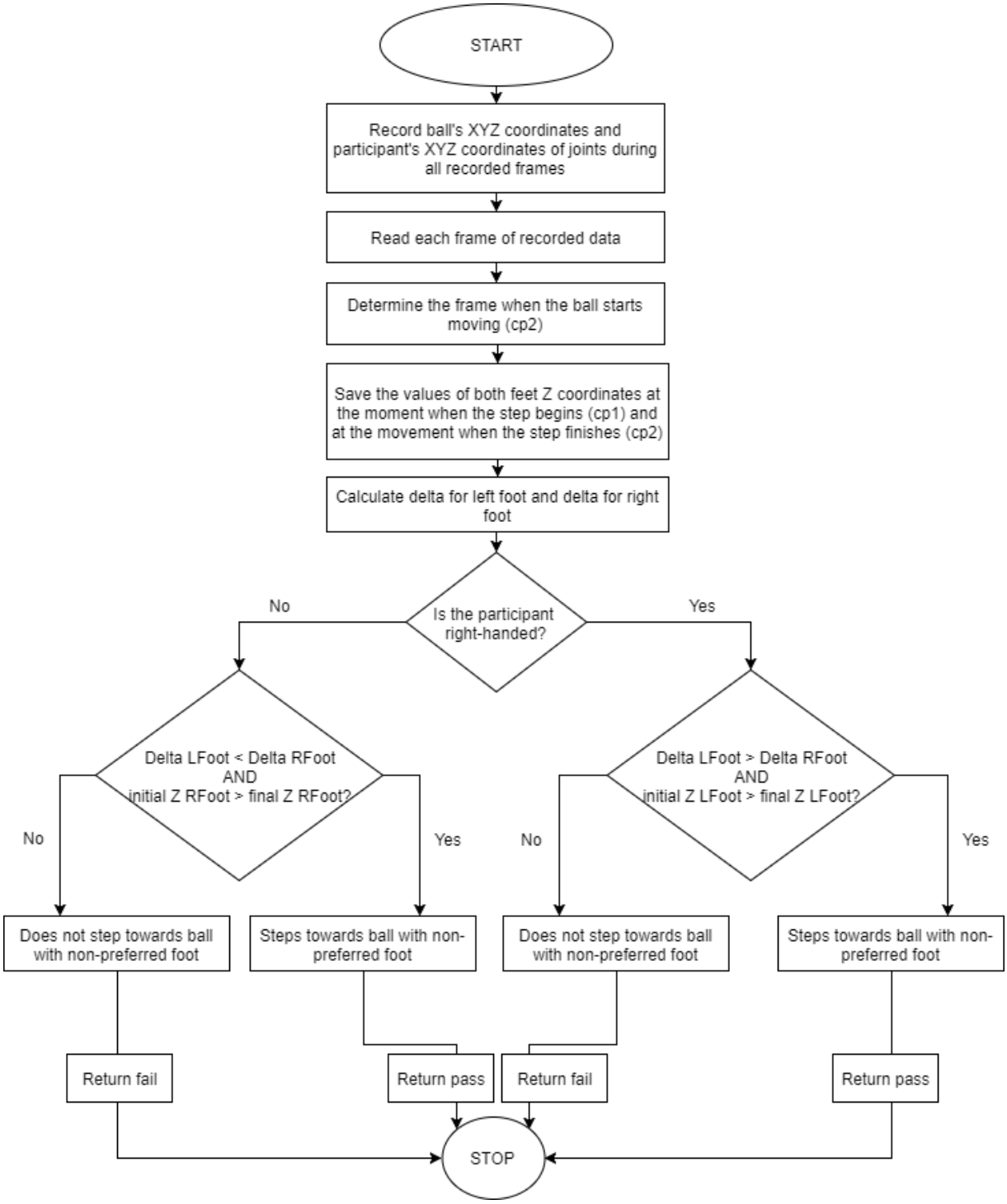

Figure 139. Algorithm to determine whether participant passes or fails the fourth criterion.

Criterion 5: Hits ball sending it straight ahead.

Note in Figure 137 that from when the ball is struck at "cp2" until it comes out of the range of the sensor, i.e. the moment when its coordinates are equal to zero, its Z coordinate decreases, which means the ball was struck towards the Kinect. For this scenario, the wall would be anywhere behind the Kinect.

Therefore, to check if ball is sent straight ahead, the algorithm checks if from "cp2" until it gets out of the range of the Kinect, its Z coordinates decrease. The flowchart that illustrates how to check this is:

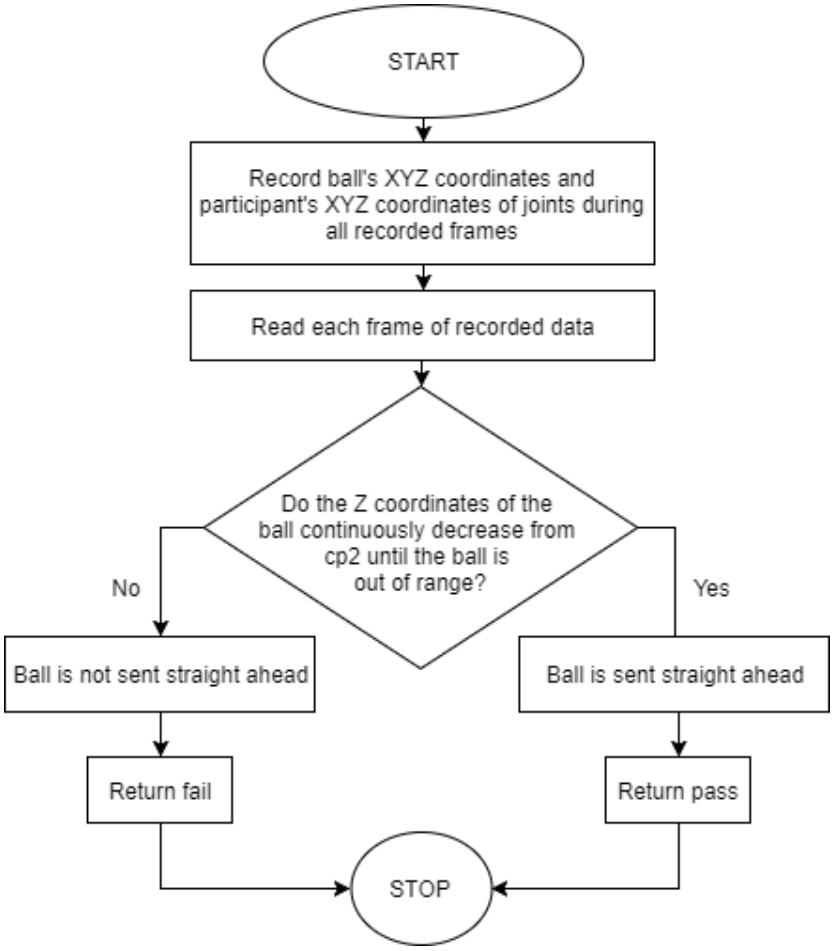

Figure 140. Algorithm to determine whether participant passes or fails the fifth criterion.

4.3.8 One-hand forehand strike of self- bounced ball

An example of a one-hand forehand strike movement is shown in Figure 141 below, where the Kinect is positioned facing the participant.

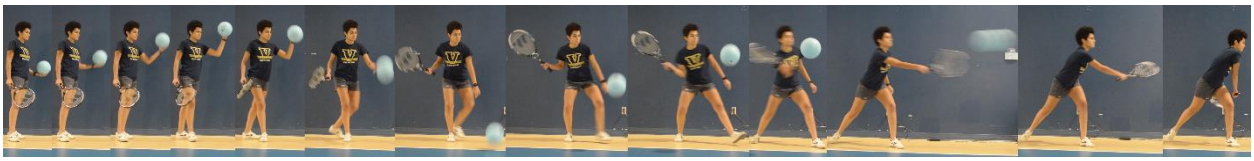

Figure 141. The sequence of images to represent a standardized one-hand forehand strike of self-bounced ball movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

The first thing the algorithm does to assess this skill is to identify the critical points, according to the position of the ball. As the movement is performed facing the Kinect, the participant will be holding the ball (frame 1) and when s/he drops it (frame 6), this marks the beginning of the fall at critical point 1 ("cp1"). Then, the algorithm starts saving the Y coordinate of the ball in an array when it drops to lower than 50cm from the floor, which means the ball is actually falling. During the fall, the Y coordinate of the next frame is always lower than the current frame, since the ball is falling towards the floor. The first occurrence of a Y coordinate of the next frame greater than the current frame marks the moment when the ball bounces, since after the bounce, the ball starts going up again. The moment when the ball bounces represents the critical point 2 ("cp2").

Note that the algorithm detects the center of the box that contains the ball, so the ball will never reach the level of the floor due to its radius, so we have to work with approximate values. Figure 142 illustrates this.

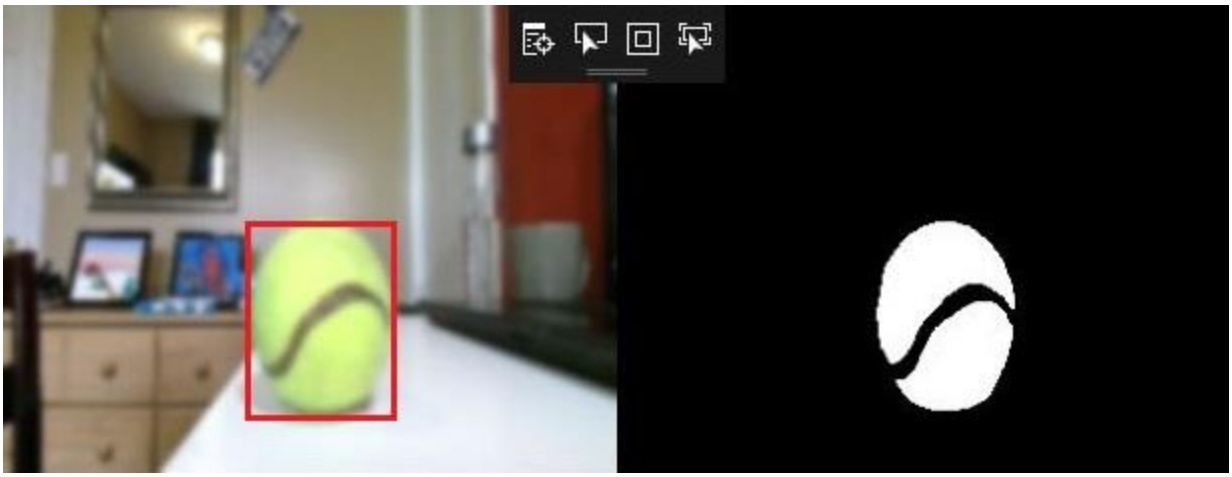

Figure 142. An example of ball detection. Note that due to its radius, when the ball is at surface level, its Y coordinate is at the center of the square.

After bouncing, the ball rises again and is struck by the participant at critical point 3 ("cp3"). At critical point 4 ("cp4"), the ball is out of range of the Kinect sensor and all the following values are set to zero. Figure 143 illustrates this.

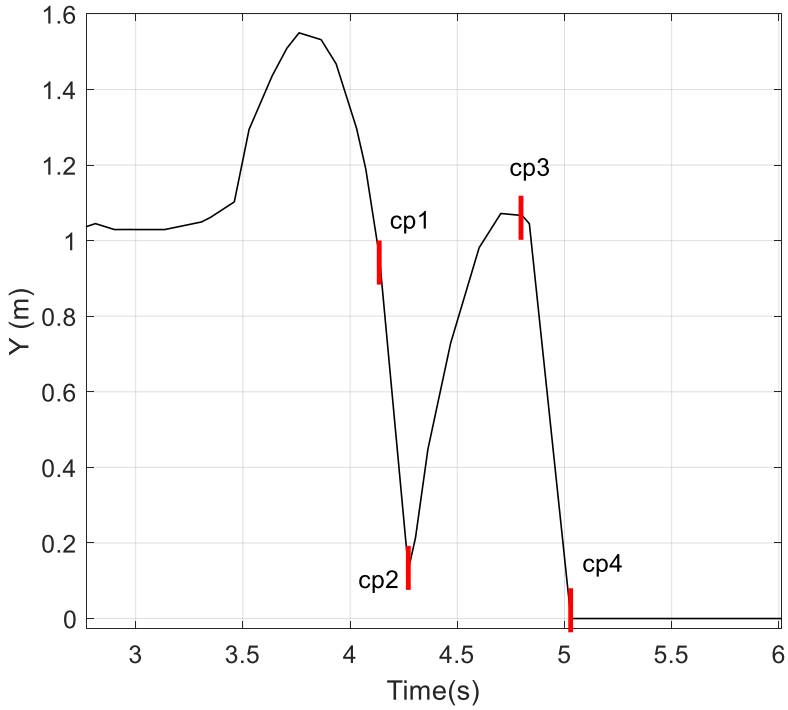

Figure 143. Variation along time of the Y coordinates of ball and critical points.

The flowchart that illustrates how to determine the critical points is:

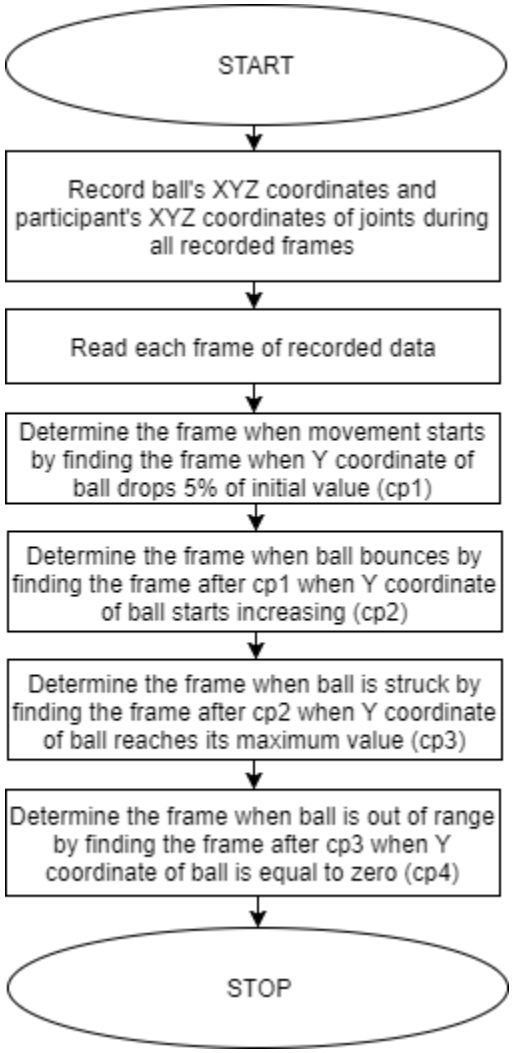

Figure 144. Algorithm to determine critical points.

<u>Criterion 1: Child takes a backswing with the paddle when the ball is bounced.</u>

The first criterion consists of analyzing if the child takes a backswing with the paddle when the ball is bounced. By knowing the critical points, the algorithm can check if the child takes a backswing with the paddle at the moment when the ball is bounced. Prior to the analysis of the movement, the participant is asked his/her preferred hand and the answer is given as an input to

the algorithm. Therefore, the code analyzes whether the preferred hand reaches behind the head in a range between 5 frames, before and after “cp2”. It also checks if the Y coordinate of the preferred hand at five frames before “cp2” is lower than at five frames after “cp2”. If the Y and Z coordinates of preferred hand increases during the interval, this means that the child has elevated the paddle behind the trunk to strike the ball, which describes the back swing. The reason the algorithm checks the whole interval of 5 frames before and 5 frames after the ball is bounced is because the backswing might not happen at the exact frame when the ball bounces, due to the high sampling rate of Kinect.

Figure 145 illustrates the Z coordinate of both hands and the head during the interval of five frames before and five frames after “cp2”.

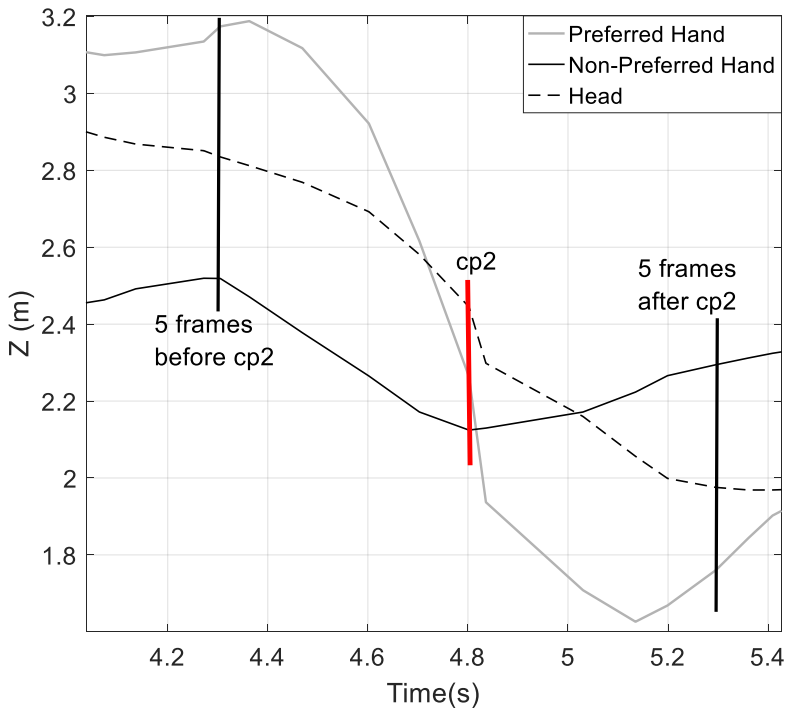

Figure 145. Z coordinates of hands and head at five frames before “cp2” and at five frames after “cp2”. The (*) represents the moment when the 3D position of the joint was inferred.

Note that in this example, the hand represented in grey is the preferred one, and it reaches behind the head within the range when the ball bounces at “cp2”. Figure 146 shows that the Y coordinate of the preferred hand increases within this range when ball bounces.

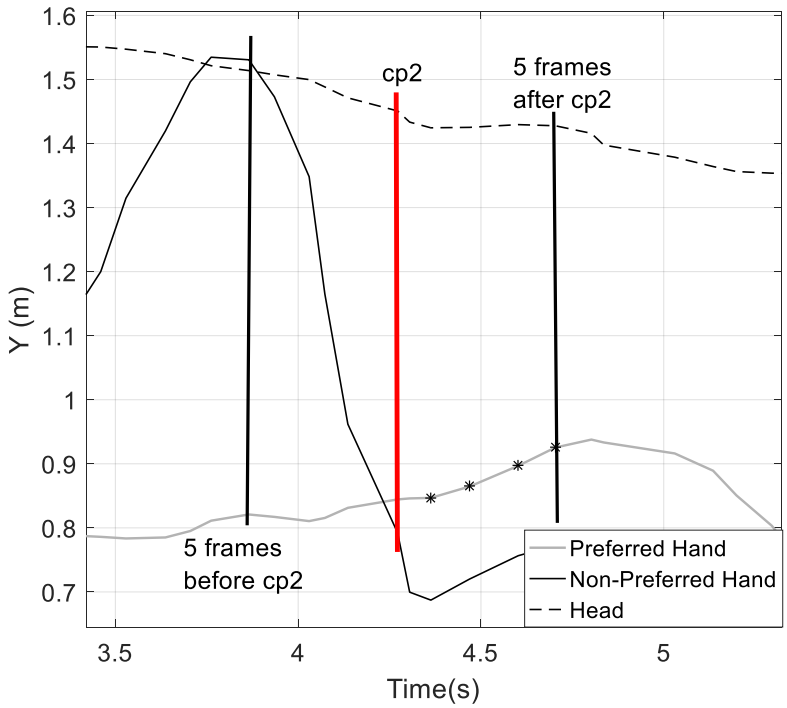

Figure 146. The Y coordinate of hands at five frames before “cp2” and at five frames after “cp2”. The (*) represents the moment when the 3D position of the joints was inferred.

So, as the preferred hand reaches behind the trunk while its Y coordinate increases, this means the participant has performed a backswing with paddle.

The flowchart that evaluates the first criterion is:

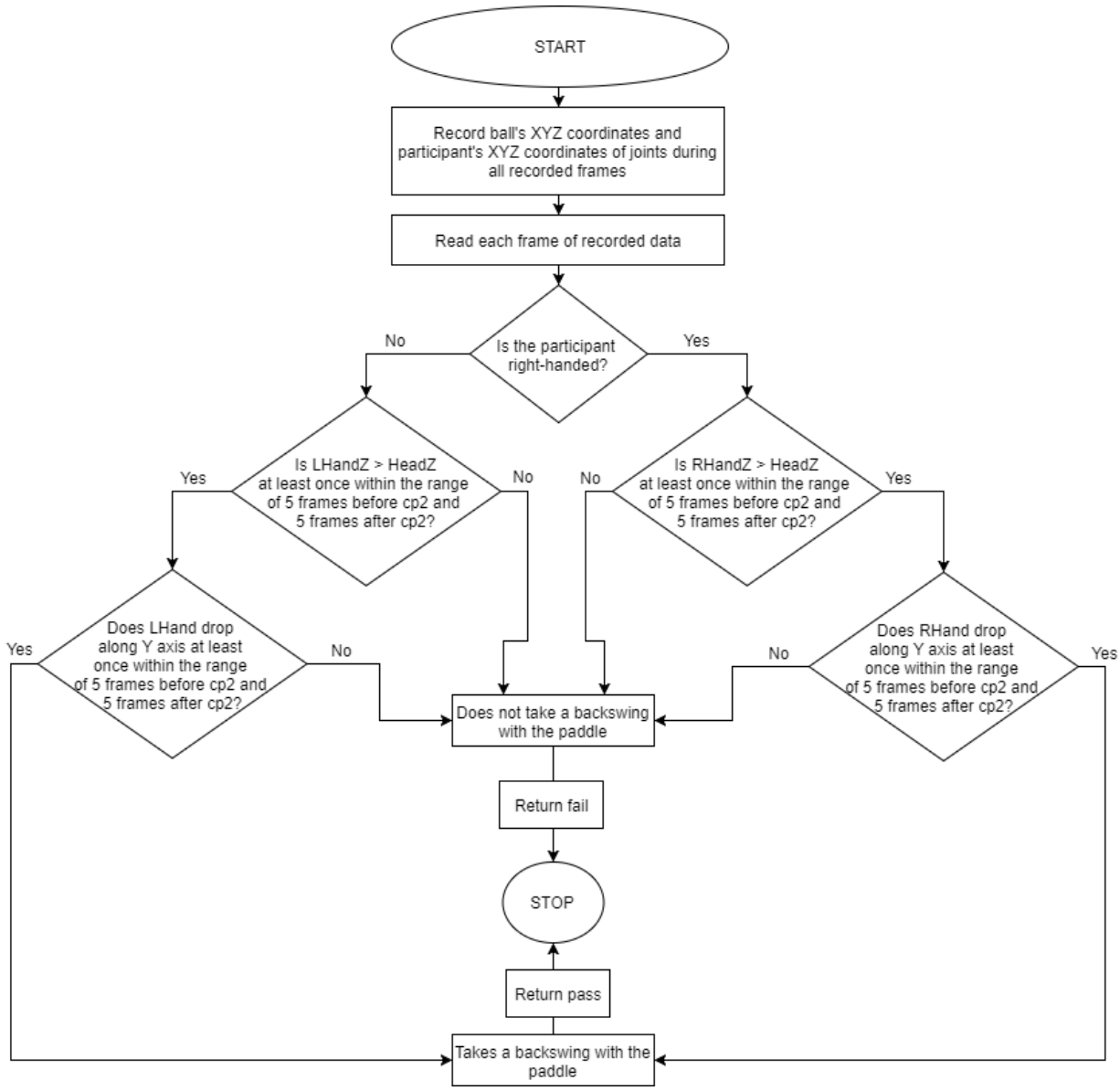

Figure 147. Algorithm to determine whether participant passes or fails the first criterion.

<u>Criterion 2: Steps toward the ball with non-preferred foot.</u>

The second criterion checks if the participant steps toward the ball with their non-preferred foot. Note that the step happens between the moment when the ball bounces and the moment when it is

struck, i.e. between “cp2” and “cp3”. As mentioned previously, the slope of the curve of the stepping foot is sharper than the slope of the curve of the non-stepping foot within the moment when the step happens. Therefore, in order to assess this criterion, the algorithm compares the variation of the Z coordinate of the preferred feet to the non-preferred feet at “cp2” and “cp3” to determine if a step has happened. It also checks if the Z coordinate of the stepping foot at “cp3” is lower than at “cp2”, in order to make sure the step was performed towards the ball and the Kinect.

Note in Figure 148 that between “cp2” and “cp3” the z coordinates of the non-stepping foot, which in this case is the right one, remains approximately the same value, while the Z coordinates of the stepping foot vary substantially. Also note that the Z coordinate of stepping foot decreases, which means it moves towards the Kinect and the ball.

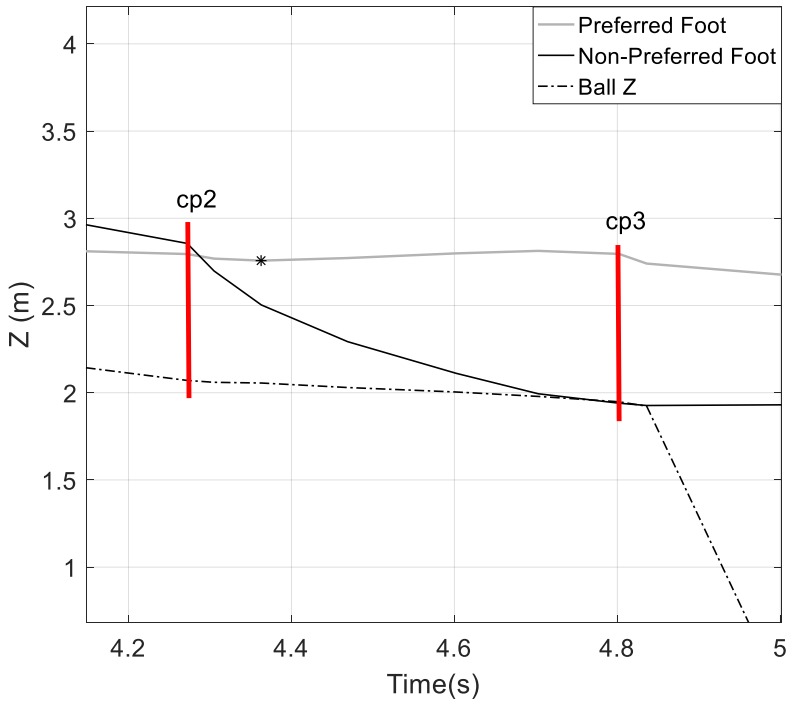

Figure 148. The Z coordinates of feet and ball between “cp2” and “cp3”. The (*) represents the moment when the 3D position of the joint was inferred.

The flowchart that illustrates how to evaluate the second criterion is:

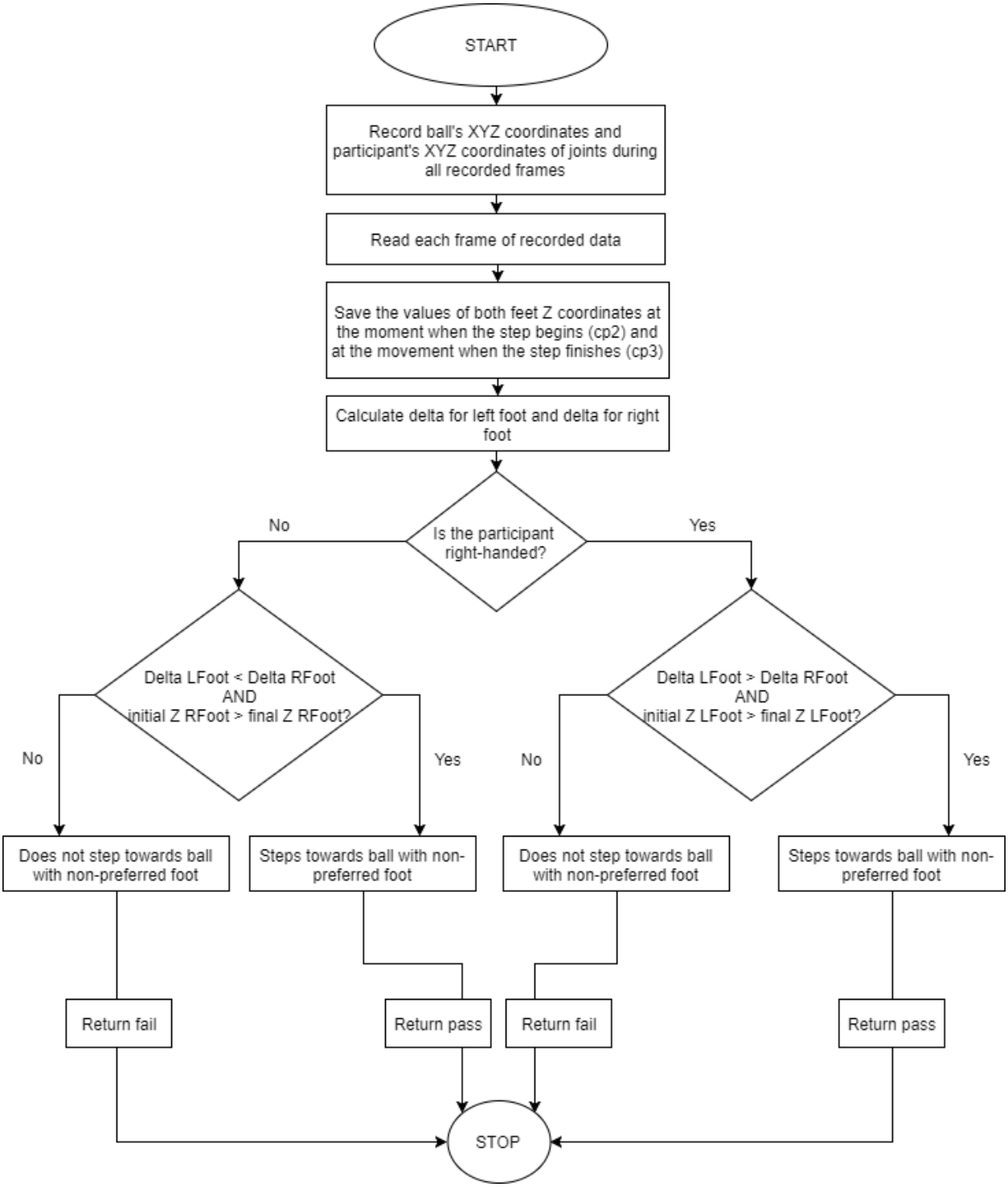

Figure 149. Algorithm to determine whether participant passes or fails the second criterion.

Criterion 3: Strikes the ball toward the wall.

The third criterion checks if the participant strikes the ball toward the wall. Note in Figure 148 that after the ball is struck at "cp3" until it comes out of the range of the sensor at "cp4", its Z coordinate decreases, which means the ball was struck towards the Kinect. For this scenario, the wall would be anywhere behind the Kinect.

The flowchart that evaluates the third criterion is:

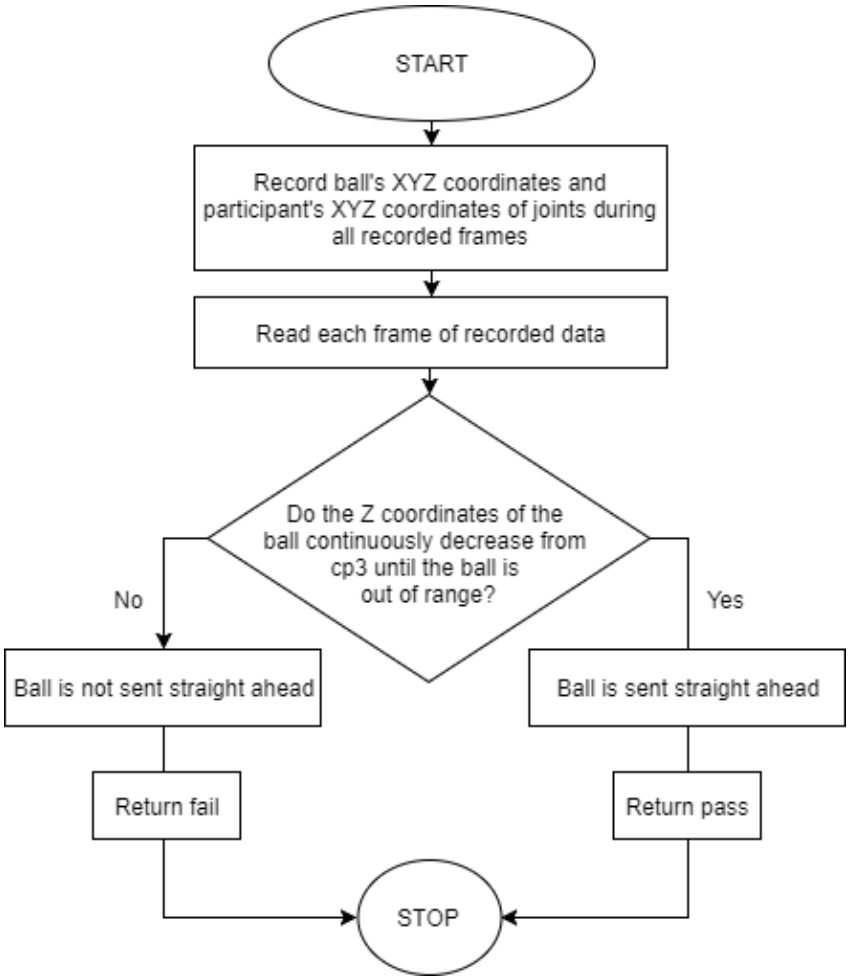

Figure 150. Algorithm to determine whether participant passes or fails the third criterion.

Criterion 4: Paddle follows through toward non - preferred shoulder.

The last criterion checks if paddle follows through towards the non-preferred shoulder (e.g. frame 13). As the X axis divides the right and left sides of the sensor into positive and negative respectively, the algorithm needs to check the lowest value reached by the preferred hand along the X axis and then compare it to the lowest value reached by the non-throwing side shoulder along the X axis. In this example, as the participant strikes ball with right hand, the algorithm checks if after "cp3", the throwing hand reaches lower values than the left shoulder. Figure 151 illustrates this situation.

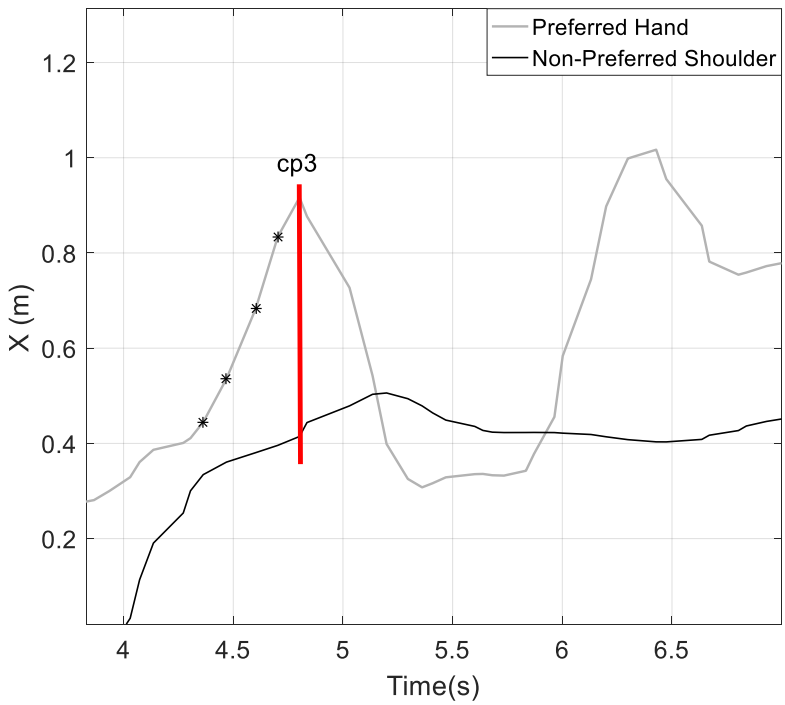

Figure 151. The X coordinates of preferred hand and non-preferred shoulder from "cp3" to the end of the movement. The (*) represents the moment when the 3D position of the joint was inferred.

Note that after "cp3" the preferred-hand follows through towards the non- preferred shoulder.

The flowchart that evaluates the last criterion is:

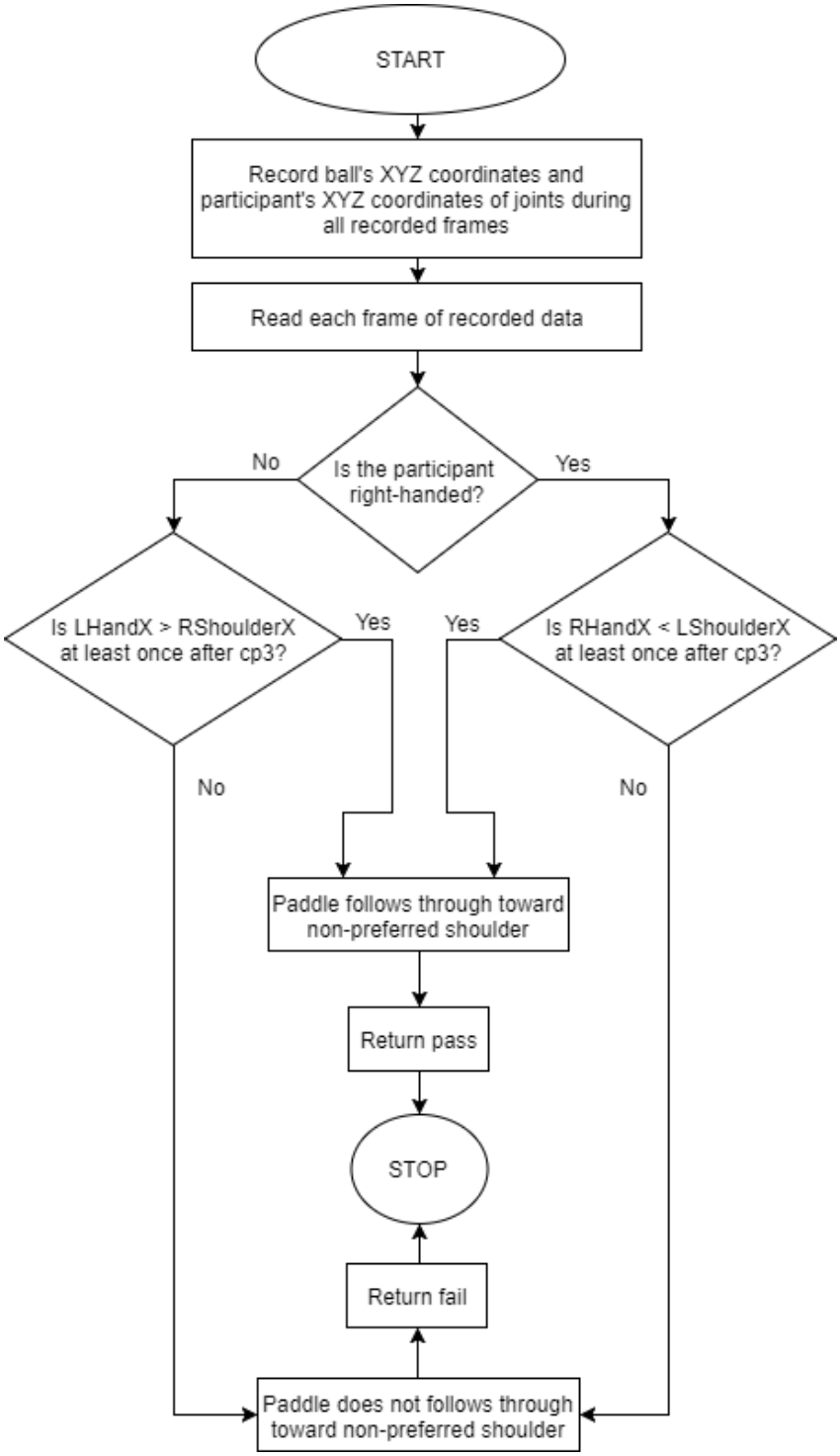

Figure 152. Algorithm to determine whether participant passes or fails the fourth criterion.

4.3.9 One-hand stationary dribble

An example of a one-hand forehand strike movement is shown in Figure 153 below, where the Kinect is positioned facing the participant.

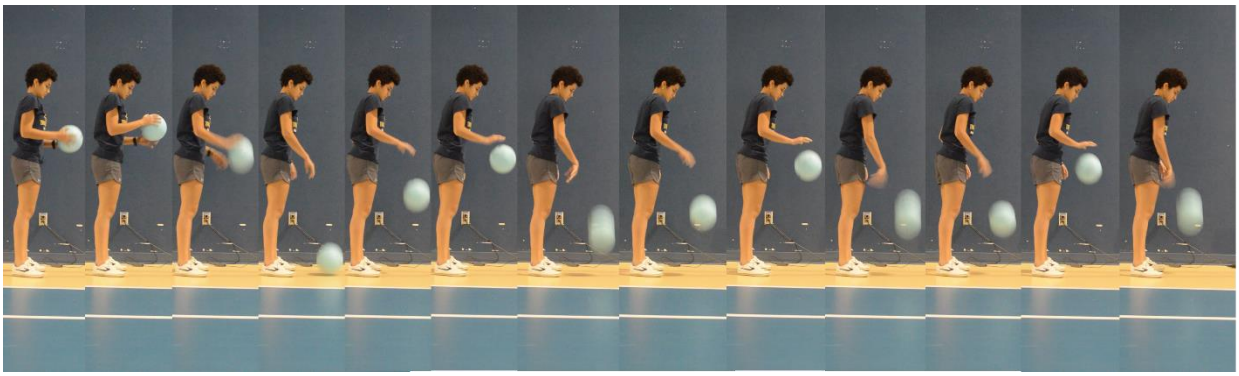

Figure 153. The sequence of images to represent a standardized one-hand stationary dribble movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

The participant is asked to start the movement holding the ball at about the waist level, so the moment when the dribble begins ("cp1") can be determined by the first occurrence when the Y coordinate of the ball reaches below the base spine level. The moment when the movement ends can be determined by the last occurrence, when the Y coordinate of the ball reaches below the base spine level ("cp2").

Note in Figure 153 that as the ball bounces along time, it draws peaks on the Y axis and each peak can be used to determine the critical points for the movement. In this way, from one critical point to another, it marks that one dribble was performed. The algorithm then finds the critical points for the movement based on the Y coordinate of the ball, and based on when the movement starts and finishes, as illustrated in Figure 154.

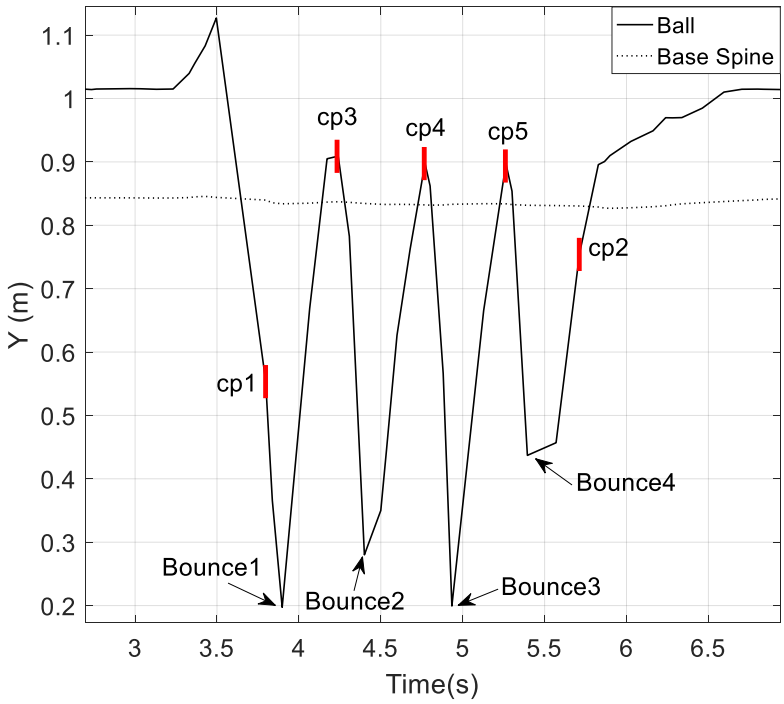

Figure 154. The Y coordinates of ball, base spine, critical points and moments when the ball bounces.

In this figure, “cp1” represents the moment when the movement begins, “cp2” represents the moment when the movement finishes, and “cp3” to “cp5” represents the ball peaks.

The following flowchart shows how the critical points were determined:

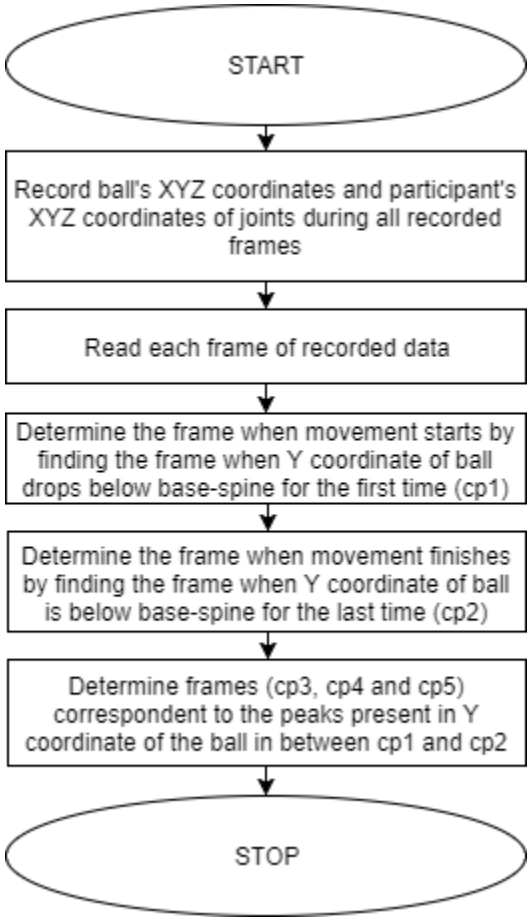

Figure 155. Algorithm to determine critical points.

<u>Criterion 1: Contacts ball with one hand at about waist level.</u>

The first criterion of the one-hand stationary dribble consists of analyzing if the participant makes contacts with the ball with one hand at about waist level. The participant's waist level, i.e. the distance from the participant's waist to the floor, can be estimated as anywhere between the Y coordinate of the participant's base-spine and mid-spine. The algorithm then checks if the Y coordinate of the ball, at each critical point that represents the ball peaks, is above the Y coordinate

of the hips and below the Y coordinate of the mid-spine. If the Y coordinate of the ball is in that range for all ball peaks, this means that the participant connects with the ball at about waist level.

Figure 156 illustrates this situation.

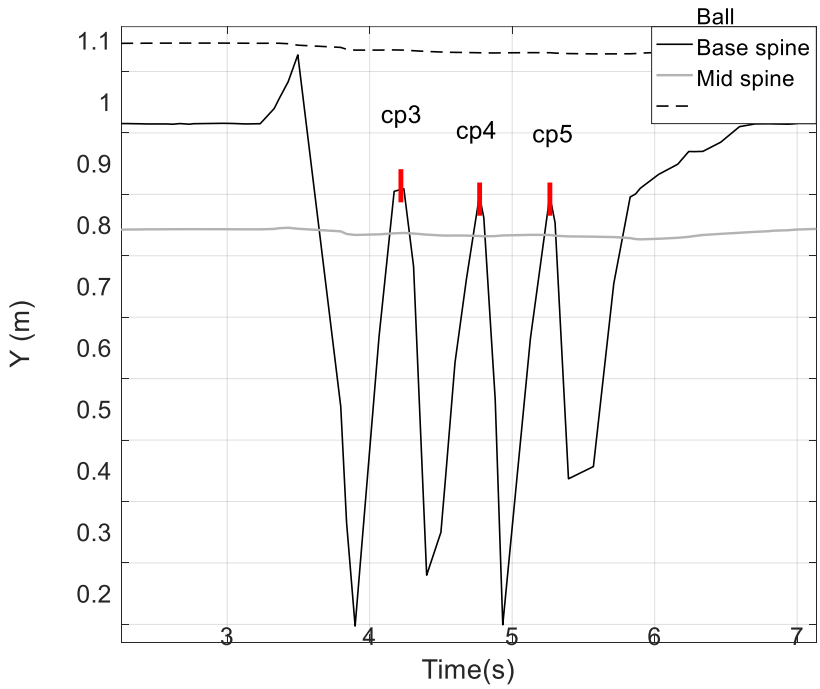

Figure 156. The Y coordinates of ball, base spine, mid spine and critical points.

Note that at each ball peak, i.e. at “cp3”, “cp4” and “cp5”, the Y coordinate of the ball is above the Y coordinate of base-spine and below Y coordinate of mid-spine.

The flowchart that analyses if the participant contacts ball at about waist level is:

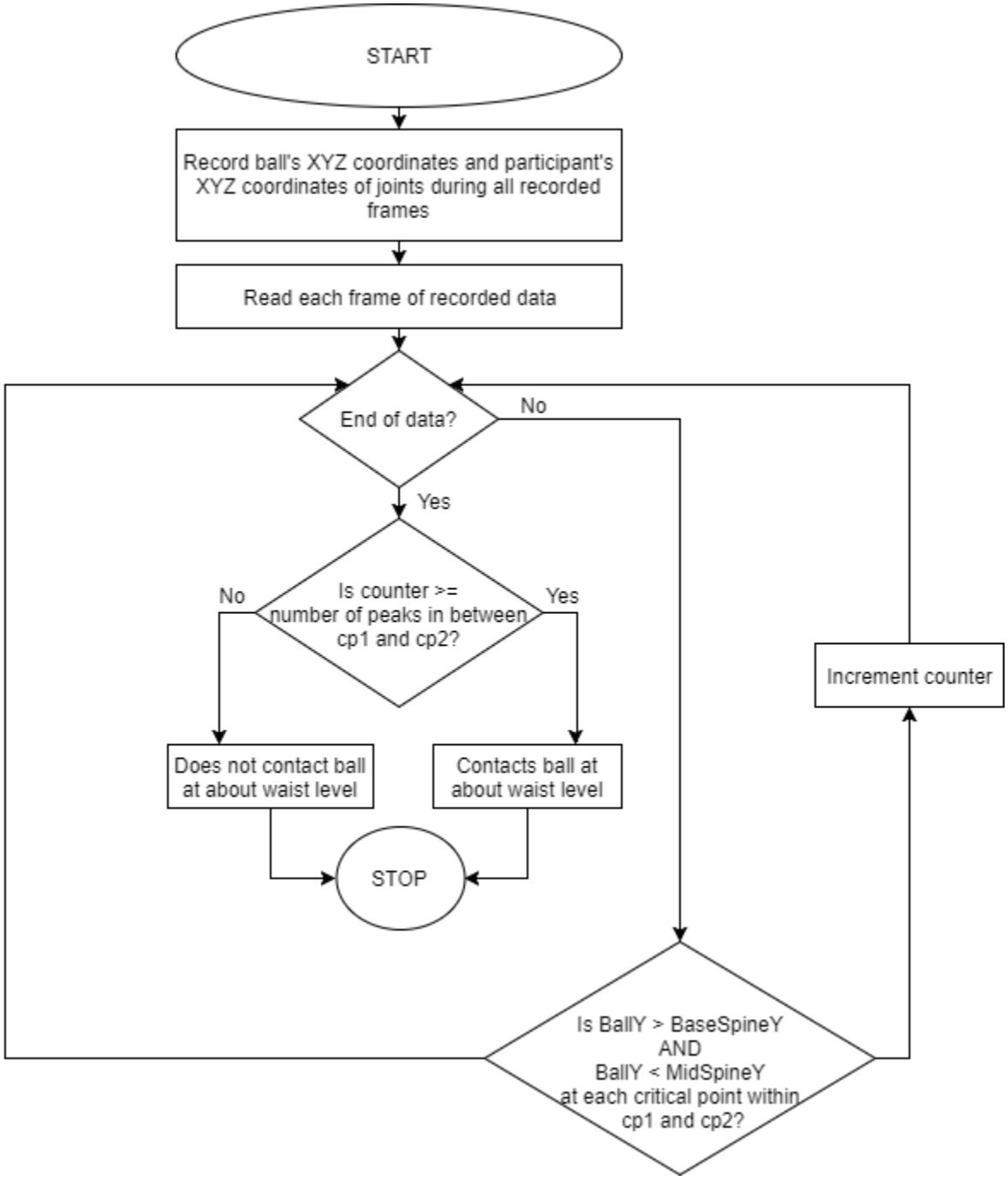

Figure 157. Algorithm to determine whether participant contacts ball at about waist level.

In addition, the algorithm also needs to determine if the participant pushes the ball with one hand. Note from Figure 153 that as the participant performs the dribble with his/her preferred hand, the non-preferred arm stays extended, which causes the non-preferred hand to stay at an approximately constant level from the floor. As the ball bounces, the preferred hand accompanies the ball movement up and down. Therefore, for each ball peak, the Y coordinate of the preferred hand is above the Y coordinate of non-preferred hand, as illustrated in Figure 158.

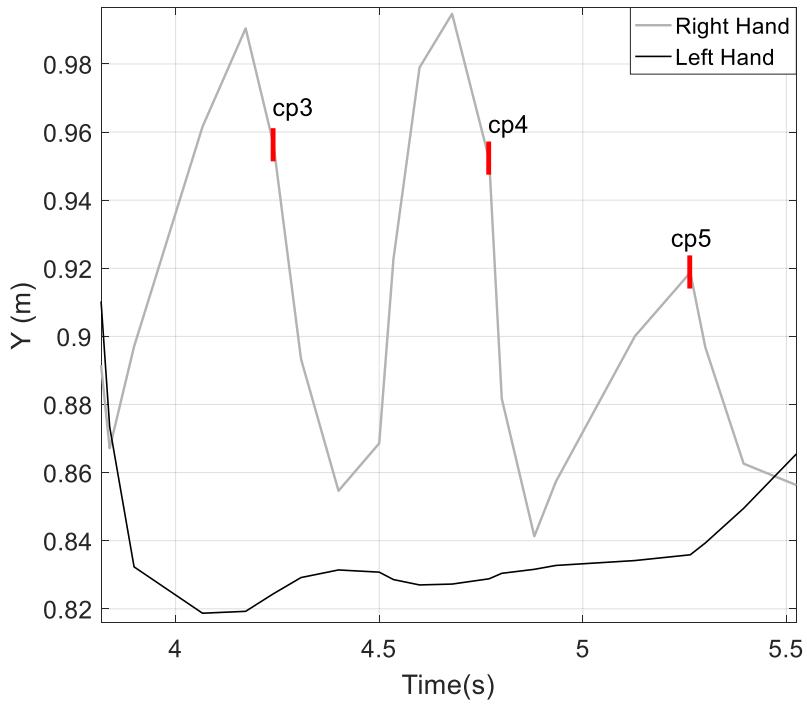

Figure 158. The Y coordinates of hands at “cp3”, “cp4” and “cp5”.

Considering this, the algorithm checks if the same hand that performs the first bounce is above the other hand for all ball peaks, and then returns the feedback for the criterium, as shown in the code flowchart:

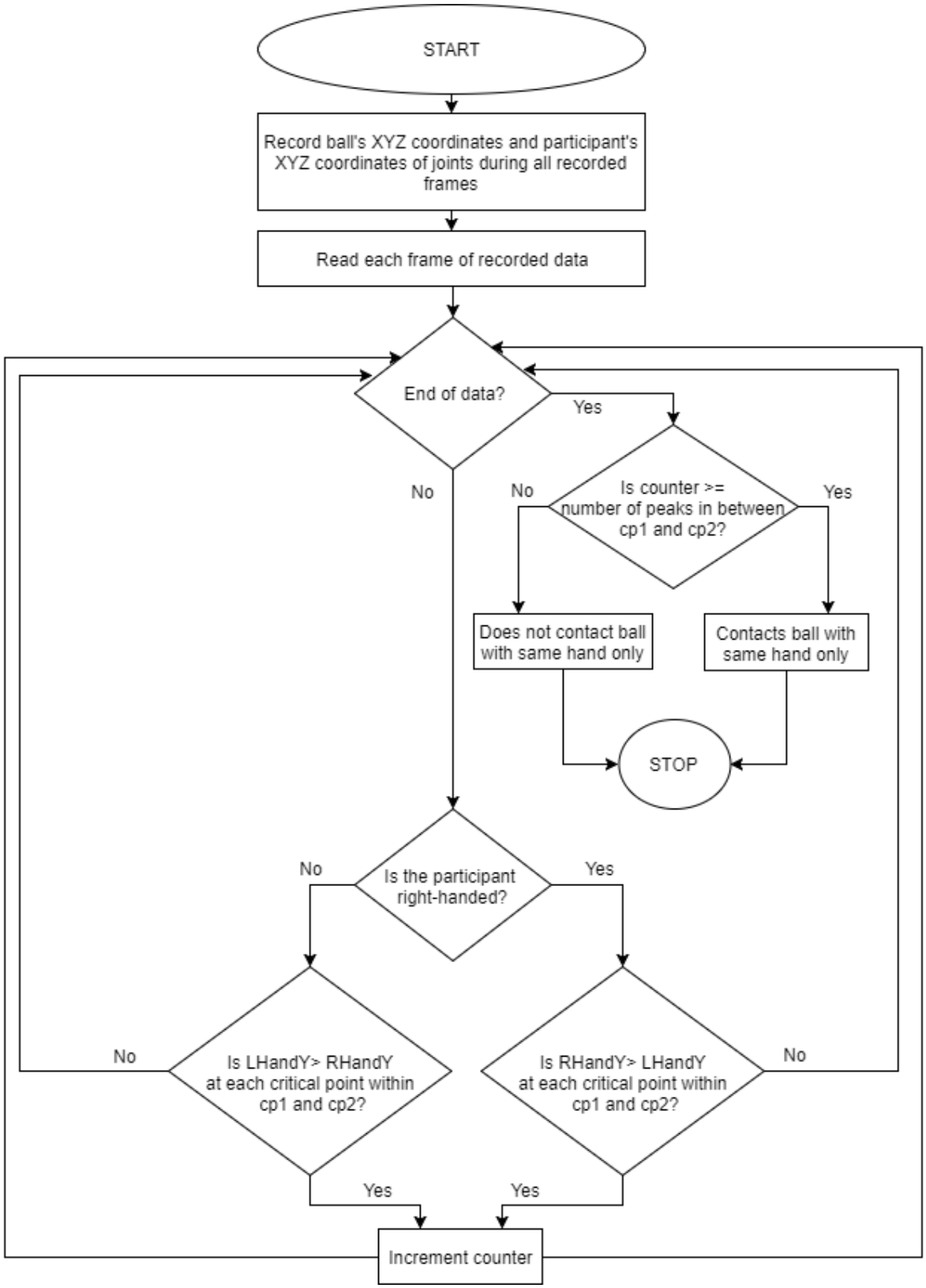

Figure 159. Algorithm to determine whether participant contacts ball with same hand.

The following flowchart illustrates the feedback for the first criterion:

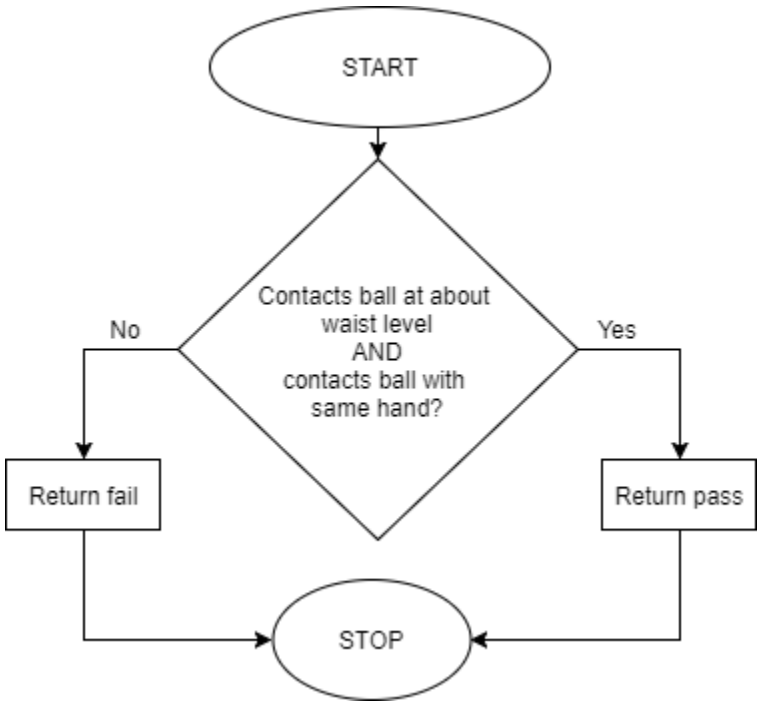

Figure 160. Algorithm to return feedback for first criterion.

<u>Criterion 2: Pushes the ball with fingertips (not slapping at ball).</u>

The second criterion checks if the participant pushes the ball with his/her fingertips, without slapping at ball. For this assessment, the algorithm uses the information from the contact sensor to determine if a slap at the ball occurs. The contact sensor is attached to a glove and positioned in the palm of the preferred hand. If a slap occurs, it means that the participant touched the ball with the palm of the hand. When a contact occurs, the contact sensor sends logic level “1”, otherwise it sends logic level “0” via Bluetooth to the code. During the whole performance of the movement, the algorithm is storing the message transmitted from the Bluetooth to an array. When the movement is completed, the algorithm checks the sum of the array. If the sum is equal to zero, it means that no slap at the ball has happened during the movement; otherwise, if the sum is different

than zero, it means that at some moment the participant slapped at ball. According to the sum of the Bluetooth output the algorithm identifies if s/he passes or fails the criterion, as shown in the following code snippet:

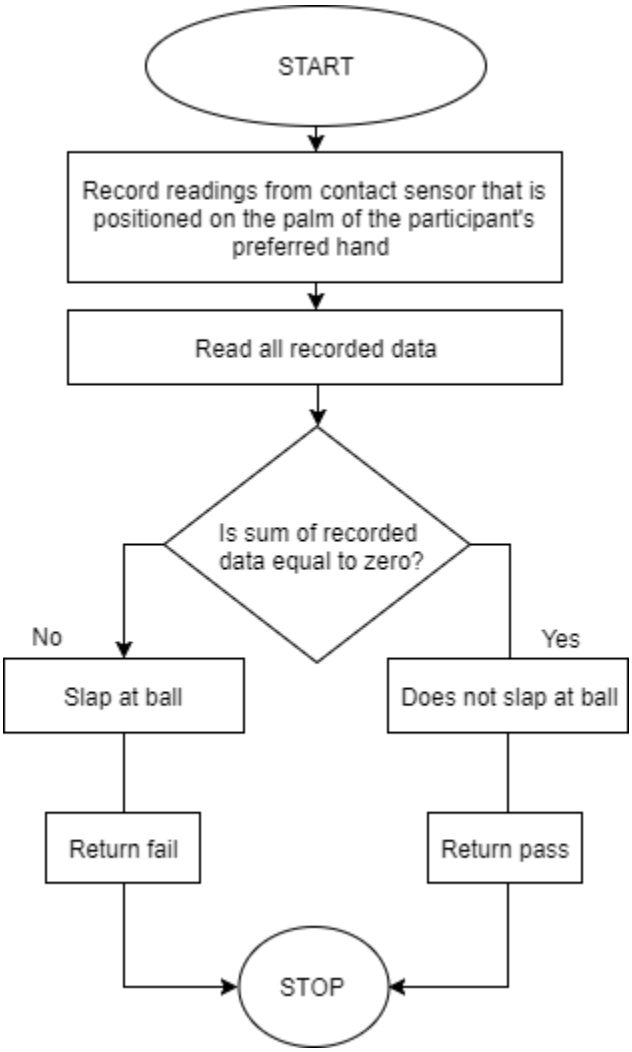

Figure 161. Algorithm to return feedback for first criterion.

Criterion 3: Maintains control of the ball for 4 bounces without moving their feet to retrieve the ball.

During the standardized one-hand dribble skill, the participant is not supposed to move his/her feet. Therefore, to assess this criterion, the algorithm checks from the moment when the ball starts moving until the end of the movement to see if both feet stay placed on the floor without moving. To determine if the feet move or not, the algorithm uses the minimum and maximum values of X, Y and Z positions of both feet reached during the movement to analyze the displacement along each coordinate. For example, if the difference between maximum and minimum values of the Y coordinate of a foot does not pass a threshold of 10 cm, it means that during the performance of the skill, the foot did not move on that axis. The code does this analyzes for both feet for the X, Y and Z coordinates. If the differences between maximum and minimum do not pass 10 cm, the participant passes the criterion. Figure 162 shows an example of a situation when the feet did not move. Note in this example that the feet do not move more than 10 cm along any axis during the entire performance of the skill. The threshold is used to overcome possible noise, just like the one in the second and third graphs in Figure 162.

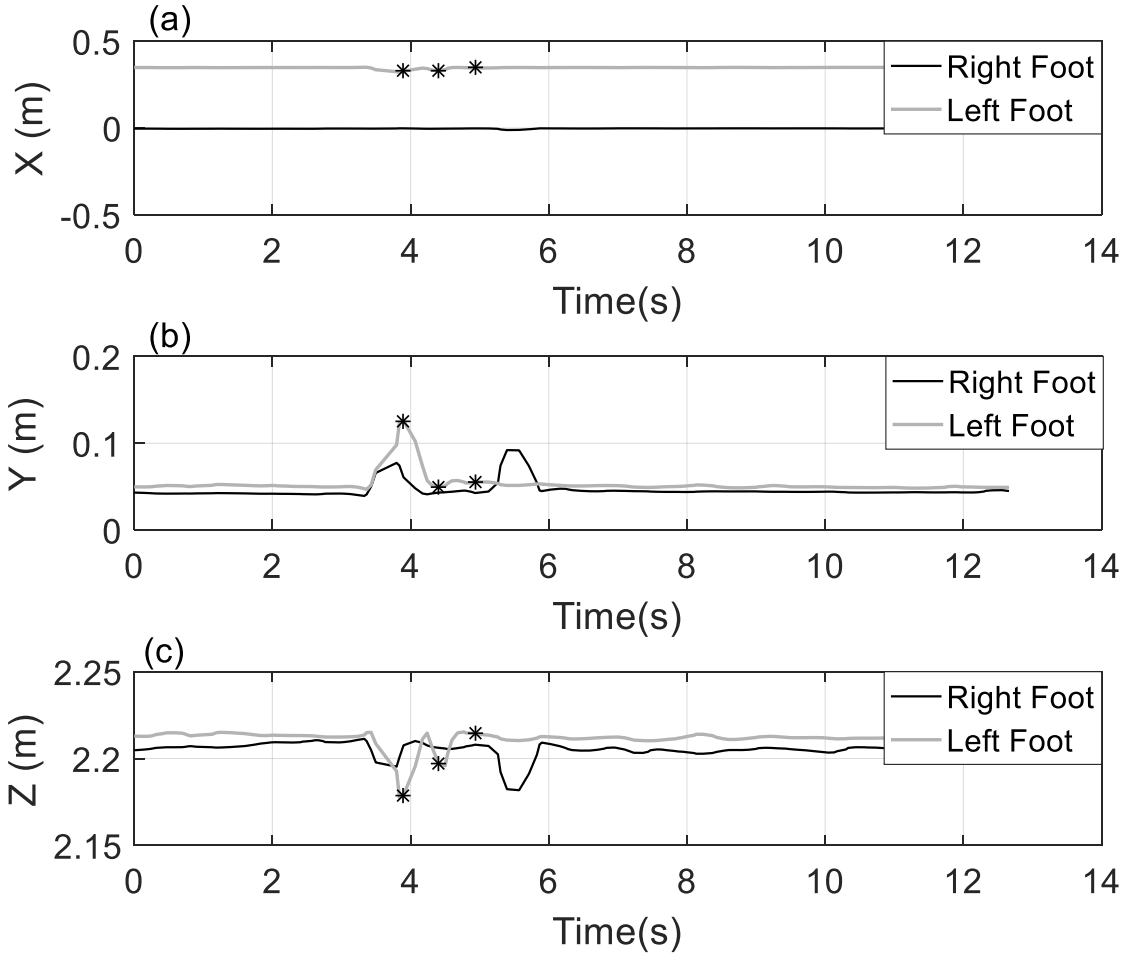

Figure 162. (a) Variation along time of X coordinates of feet. (b) Variation along time of Y coordinates of feet. (c) Variation along time of Z coordinates of feet. The (*) represents the moment when the 3D position of the joint was inferred.

In order to assess if the participant maintains control of the ball for four bounces, the algorithm uses the number of critical points, i.e. the number of bounces, to count how many times the ball bounces. It also checks the time lapse between each bounce to determine if the movement is performed in a controlled manner — the same idea used in the fourth criterion of the gallop test.

The following flowchart illustrates how to check that:

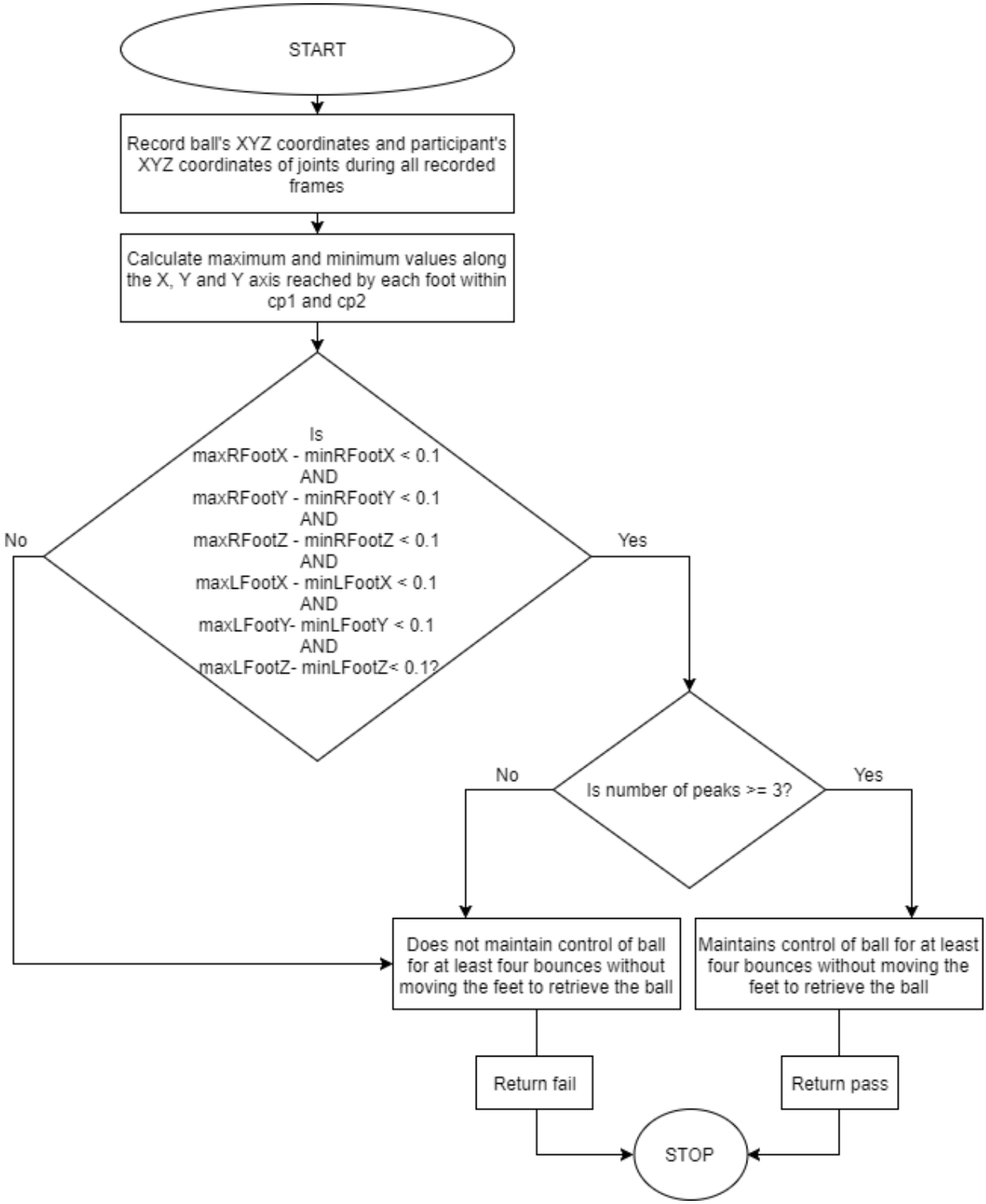

Figure 163. Algorithm to return feedback for third criterion.

4.3.10 Two-hand catch

An example of a two-hand catch movement is shown in Figure 153 below, where the Kinect is positioned facing the participant.

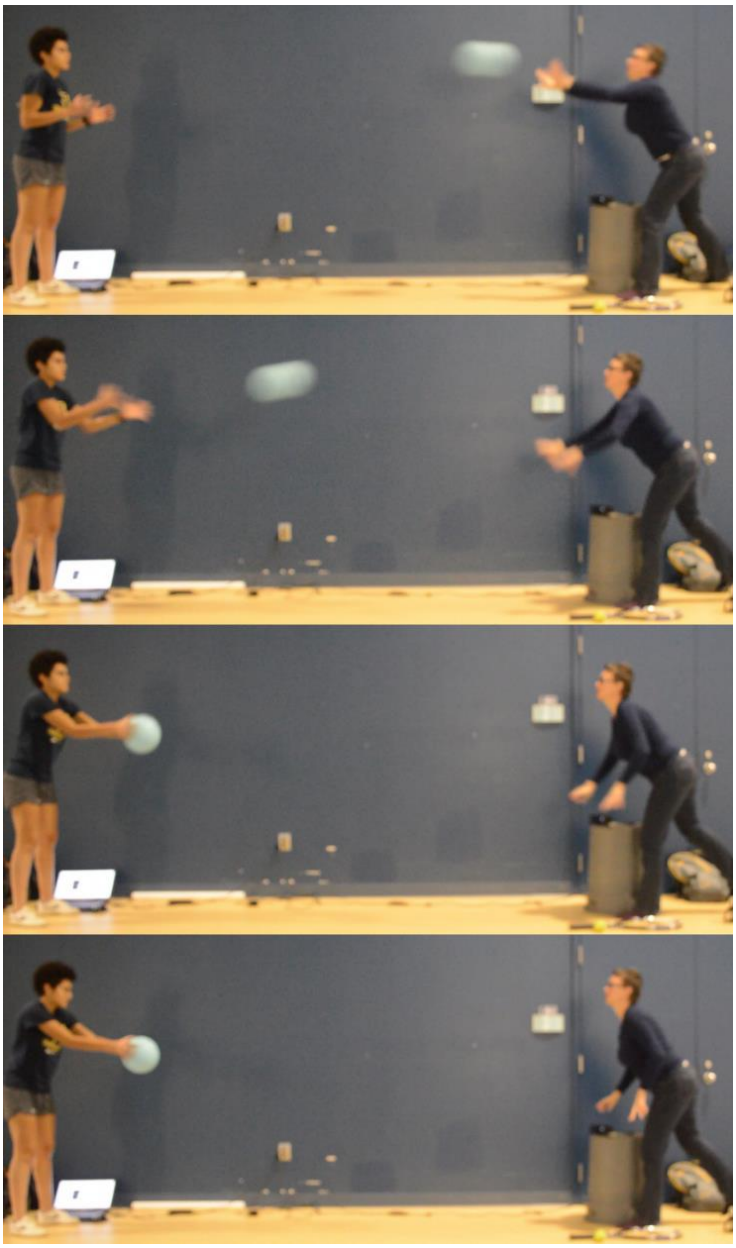

Figure 164. The sequence of images to represent a standardized two-catch movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

The algorithm uses the moment when the ball appears in the frame to determine the first critical point “cp1”, since the child could be with his/her arms extended before the ball is launched towards him/her. Therefore, the algorithm checks the frame at which the ball appears by taking the first frame at which the Z coordinate of the ball is different than zero — since when the ball is out of the Kinect’s field of view, its XYZ coordinates are set to zero.

The flowchart that determined “cp1” is:

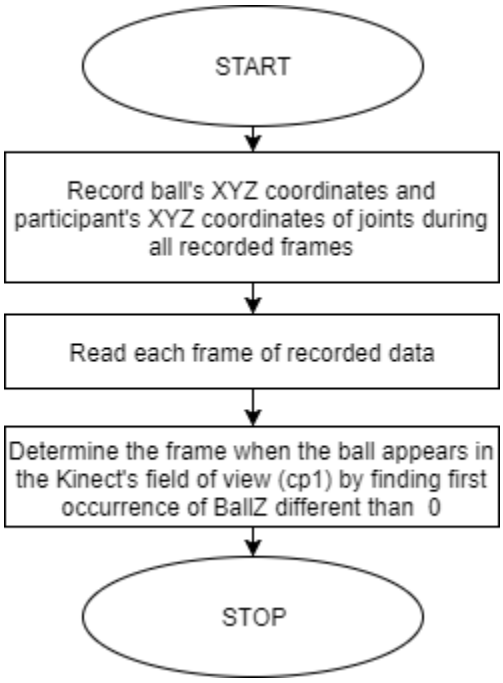

Figure 165. Algorithm to determine cp1.

The second critical point of the moment is when the ball is caught. Note that the participant will be standing within the range of the depth sensor, which is around 4.5 m from the Kinect, waiting for the ball to be thrown. As the ball is thrown from behind the sensor, its Z coordinates increase as it moves towards the participant, since it gets farther away from the sensor. Therefore, in order to determine when the ball is caught (“cp2”), the algorithm checks when the Z coordinate of the

ball stops increasing, i.e. the moment when the ball stops increasing in distance from the sensor. The following flowchart shows how to identify “cp2”:

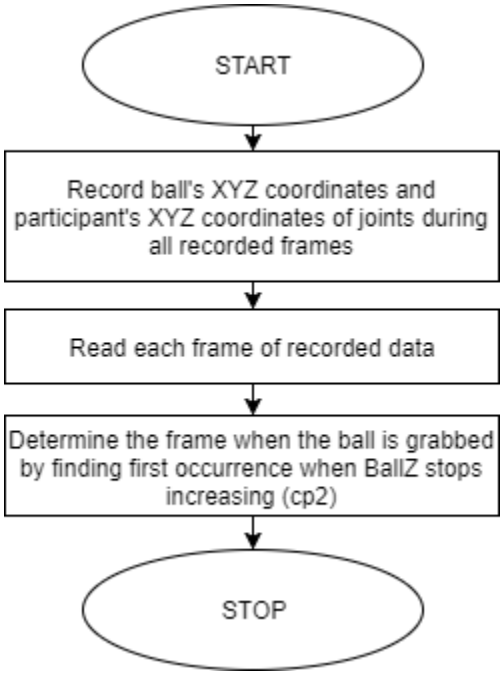

Figure 166. Algorithm to determine cp2.

<u>Criterion 1: Child’s hands are positioned in front of the body with the elbows flexed.</u>

The first criteria of the two-hand catch test consists of determining if the child’s hands are positioned in front of the body with his/her elbows are flexed. Figure 167 (a) illustrates the moment when the ball appears to the frame and Figure 167 (b) illustrates the behavior of the hands along Z.

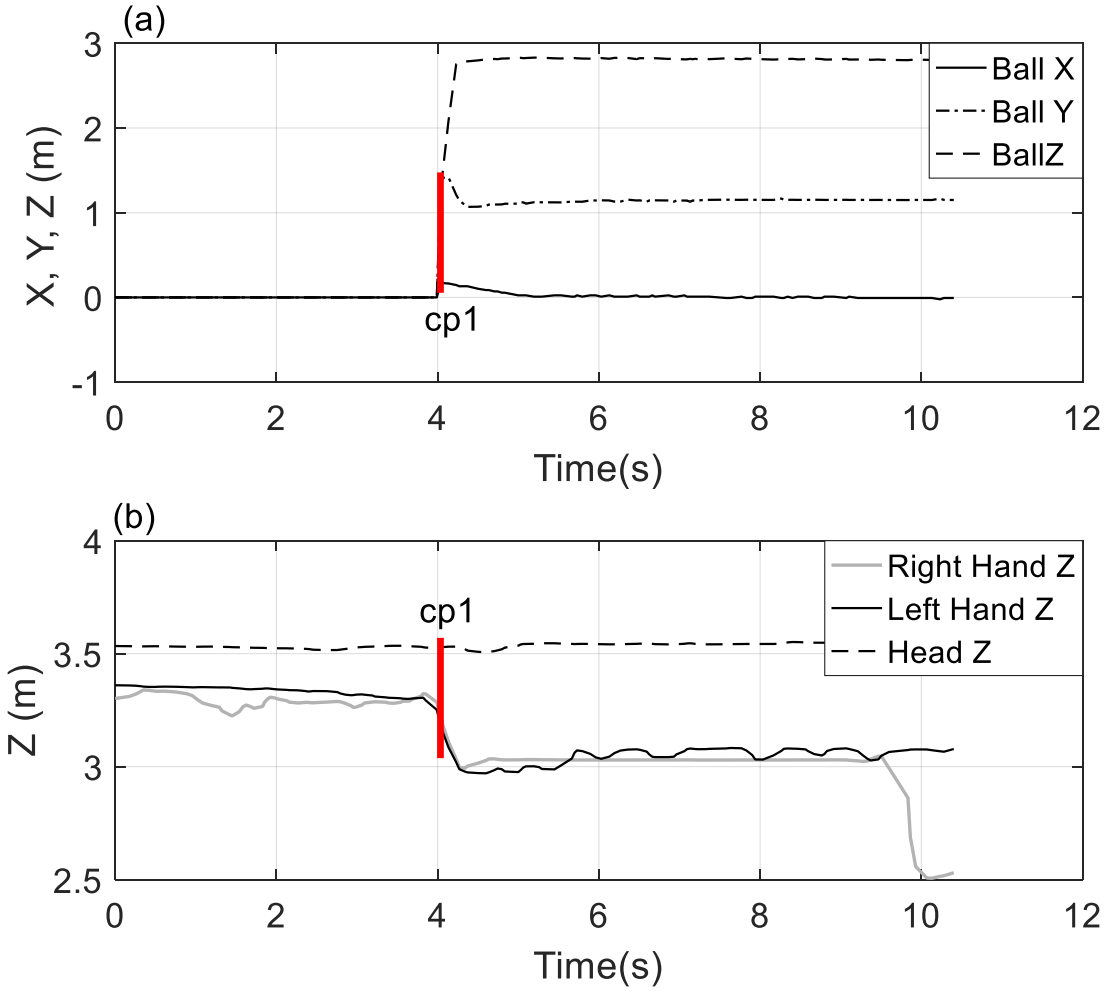

Figure 167. (a) 3D coordinates of ball. (b) Variation along time of Z coordinates of hands and head.

Note that at "cp1", both hands are closer to the sensor than the head, which means both hands are positioned in front of the body.

Figure 168 illustrates the ratios of the arms during the performance of the two-handed catch test.

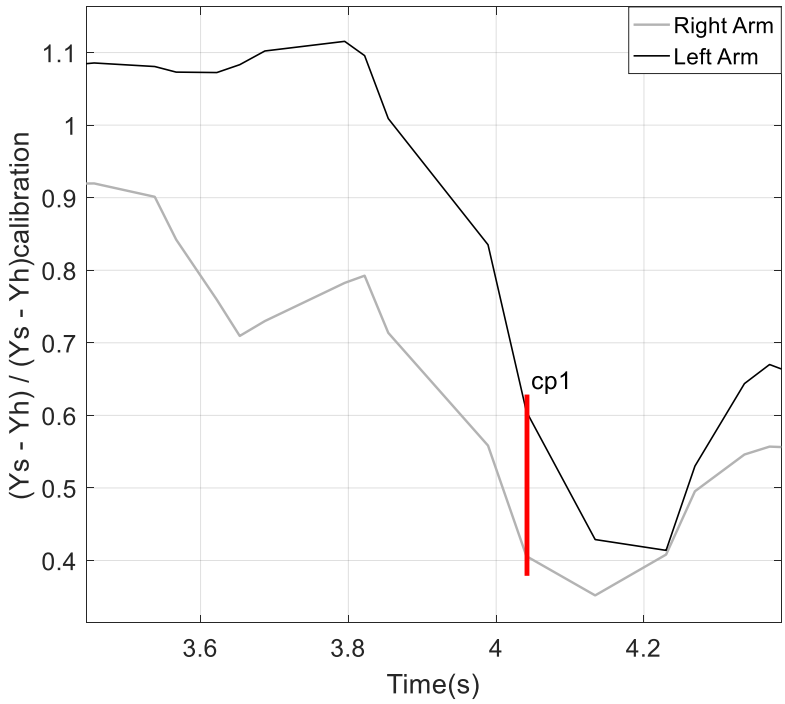

Figure 168. Ratios of right and left arms at “cp1”.

As mentioned previously, any value below 0.75 means that the participant has his/her arms flexed less than 120°. Note that at “cp1”, both arms reach values of ratios below 0.75.

Thus, by knowing the frame when the ball starts approaching the participant, the algorithm checks if at that frame the Z coordinates of the hands are closer to the sensor than the head, and it also checks if the ratios of both arms are lower than 0.75. Note that the Z coordinate of the head is used to estimate the trunk position; so, when the hands have lower Z coordinates than the head, it assumes that the hands are positioned in front of the trunk.

The flowchart that illustrates how to check the first criterion is:

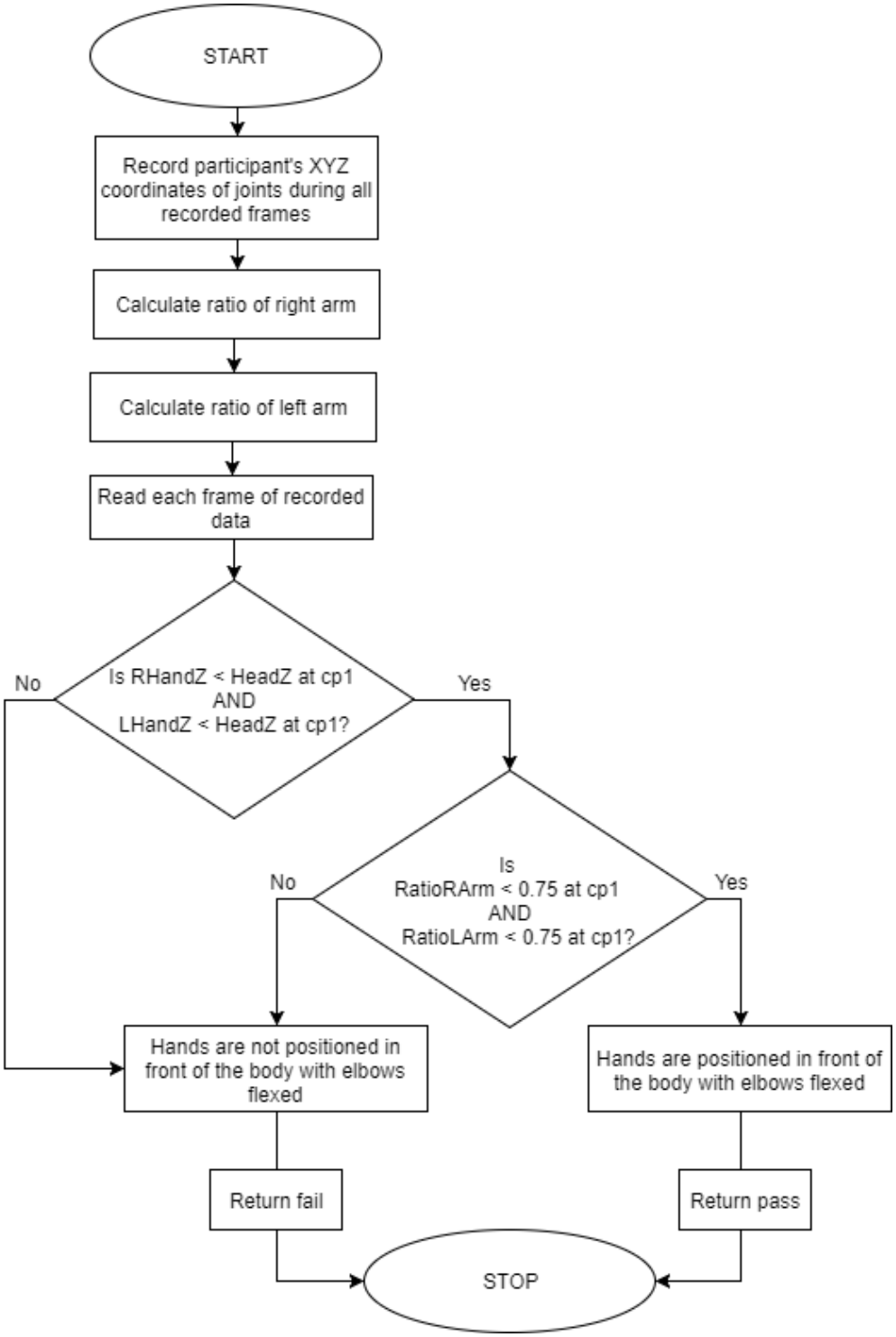

Figure 169. Algorithm to return feedback for the first criterion.

Criterion 2: Arms extend reaching for the ball as it arrives.

When the participant sees that the ball has been thrown from behind the sensor, s/he starts extending his/her arms towards the ball. Therefore, the algorithm checks if, from the moment when the ball appears at the frame until the moment it is caught, the Z coordinates of the hands continuously move forward towards the ball. Figure 170 illustrates this; note that the ball appears at "cp1" and it is caught at "cp2"; within this period, the Z coordinates of the hands decrease, until they grab the ball at "cp2".

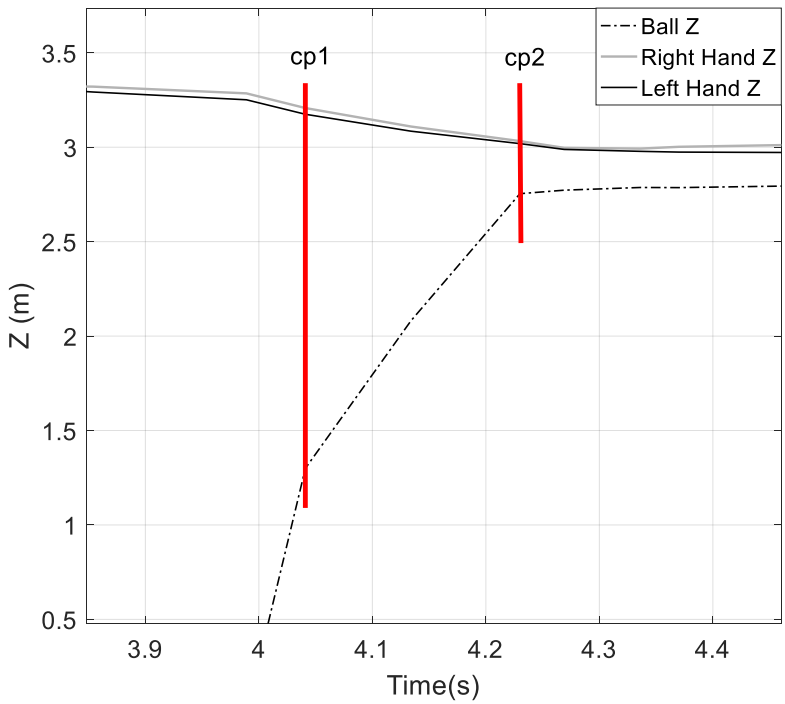

Figure 170. The Z coordinates of hands and ball between "cp1" and "cp2".

The flowchart that assess criterion 2 is:

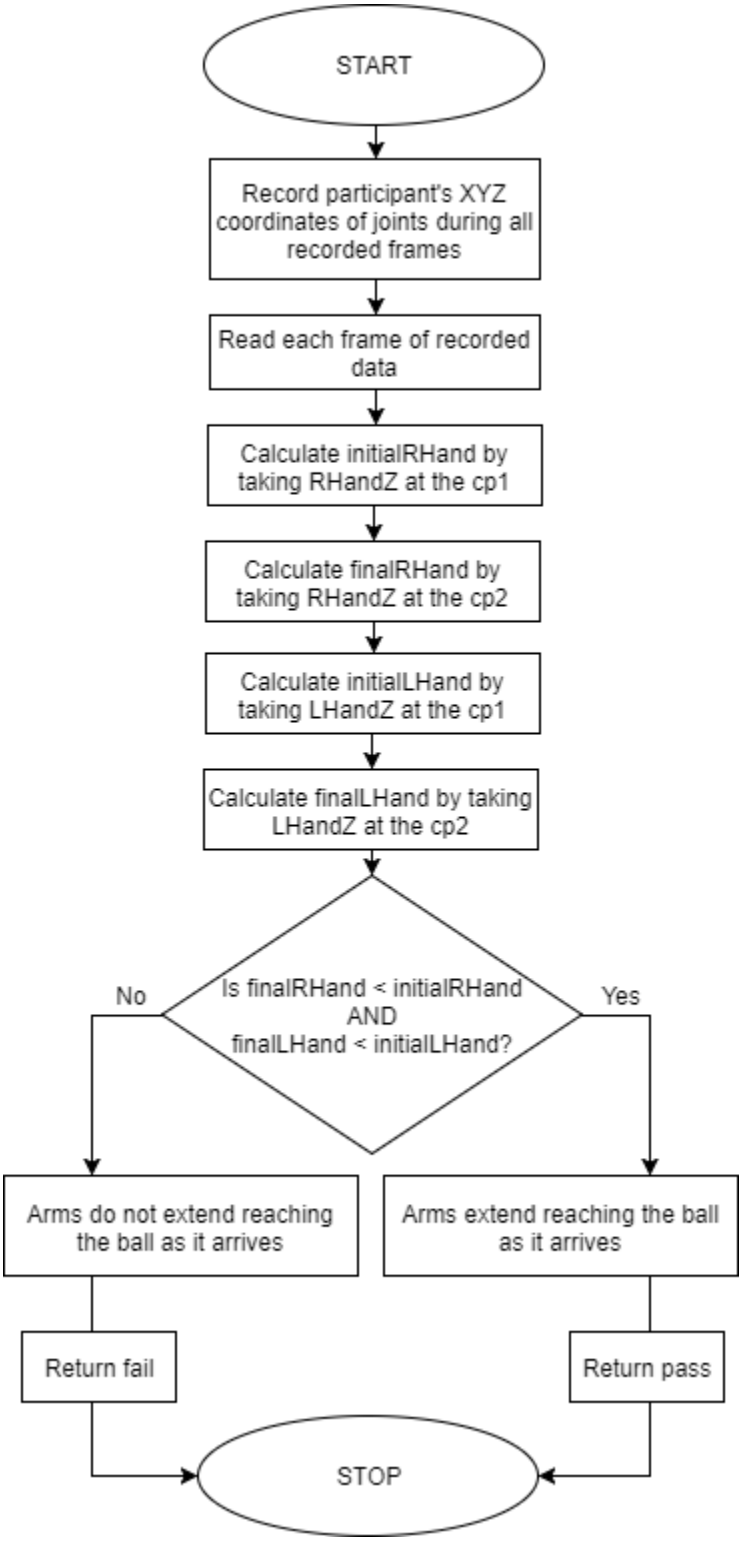

Figure 171. Algorithm to return feedback for the second criterion.

Criterion 3: Ball is caught by hands only.

In order to check the last criterion, "ball is caught by hands only", the algorithm checks if at "cp2", the moment when the ball is caught, the Z coordinate of the ball is in front of the trunk. That analysis takes into account that when the ball is caught by hands only, its Z coordinate keeps a certain distance from the joints that belong to the trunk.

The algorithm uses mid-spine, knees and head as a reference for the trunk; therefore, following the aforementioned idea, at the moment when the ball is caught ("cp2") its Z coordinate should be lower, i.e. in front of the Z coordinate of the mid-spine, knees and head. This comparison assumes that if the ball is caught by hands only, it will be caught at a certain distance from the body. Figure 172 illustrates this.

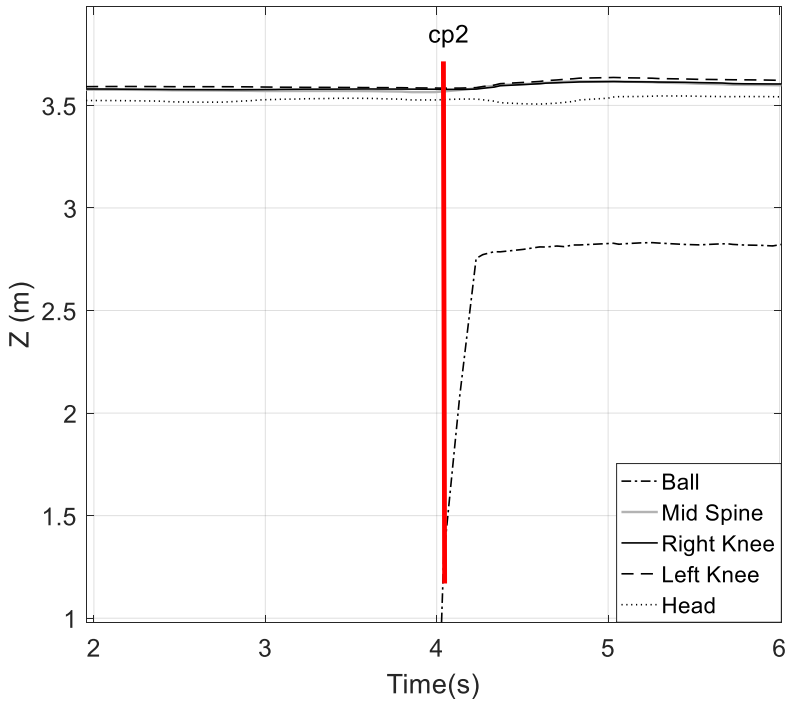

Figure 172. The Z coordinates of ball and trunk at "cp2". Trunk is represented by the mid spine, knees and head.

Note that at “cp2”, the ball is caught by the hands and is far from the trunk. Hence, the algorithm checks if at “cp2” the Z coordinate of the ball is at least 30 cm lower than the Z coordinate of the knees and mid-spine. If the condition is true, the participant passes the criterion.

The flowchart that assess criterion 3 is:

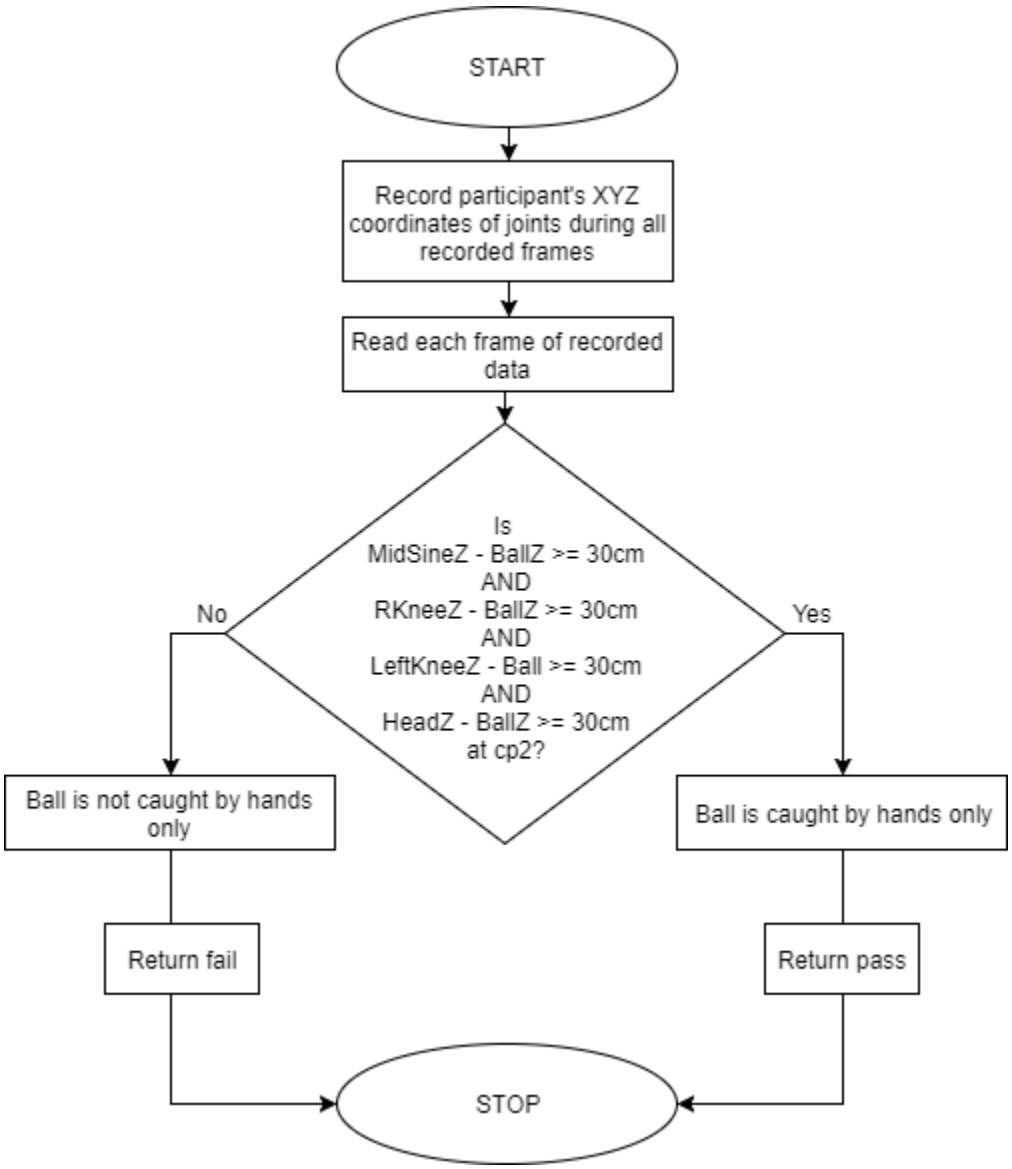

Figure 173. Algorithm to return feedback for the third criterion.

4.3.11 Kick a stationary ball

An example of a kick a stationary ball movement is shown below, where the Kinect is positioned facing the participant.

Figure 174. The sequence of images to represent a standardized kick a stationary ball movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

The critical point for this movement is the moment when the kick happens. To determine the frame when the kick happens, the algorithm checks when the Z coordinate of the ball starts dropping. As the ball is static at the beginning of the movement, after it is kicked, it moves toward the Kinect, and the Z coordinate decreases. Figure 175 illustrates that situation.

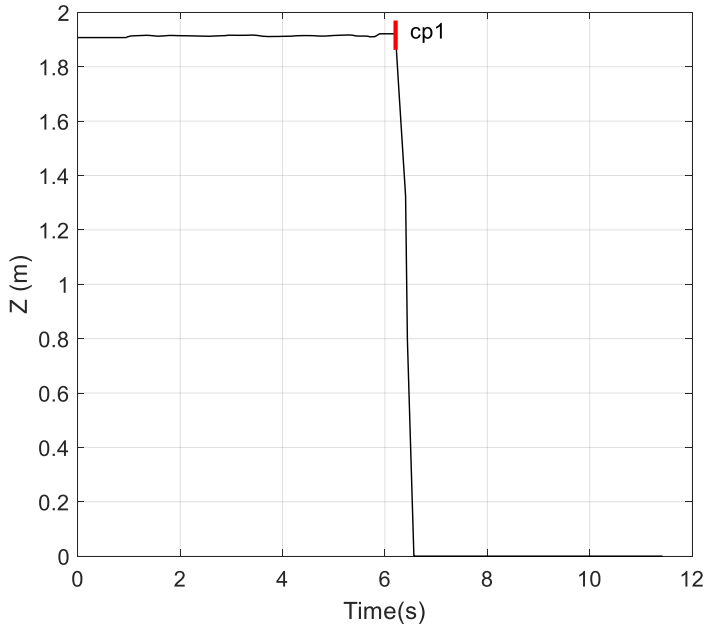

Figure 175. Variation along time of the Z coordinate of ball and moment when “cp1” occurs.

Note that the kick happens at “cp1”, and then the ball moves towards the Kinect. After it goes out of Kinect’s field of view range, its Z coordinate remains constant at zero.

The flowchart that checks “cp1” is:

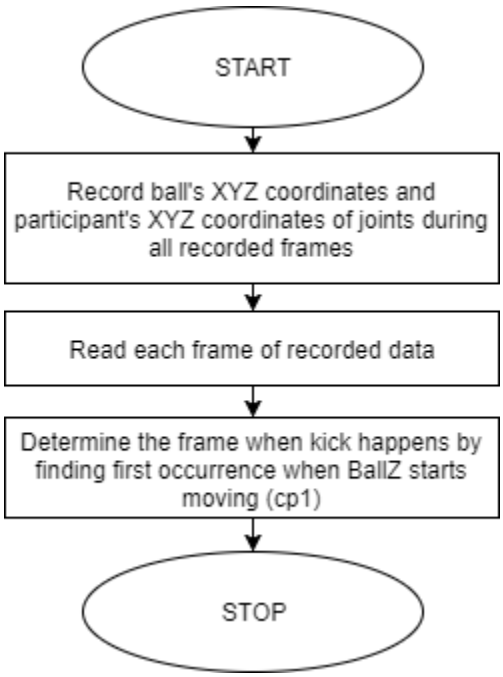

Figure 176. Algorithm to determine cp1.

<u>Criterion 1: Rapid, continuous approach to the ball.</u>

The first criterion for this skill consists of analyzing if the participant approaches the ball continuously and rapidly. After determining the frame when the kick happens, the algorithm checks the behavior of the feet from the beginning of the movement until “cp1”. Figure 177 illustrates the Z coordinate of the feet when the participant kicks the ball with the right foot.

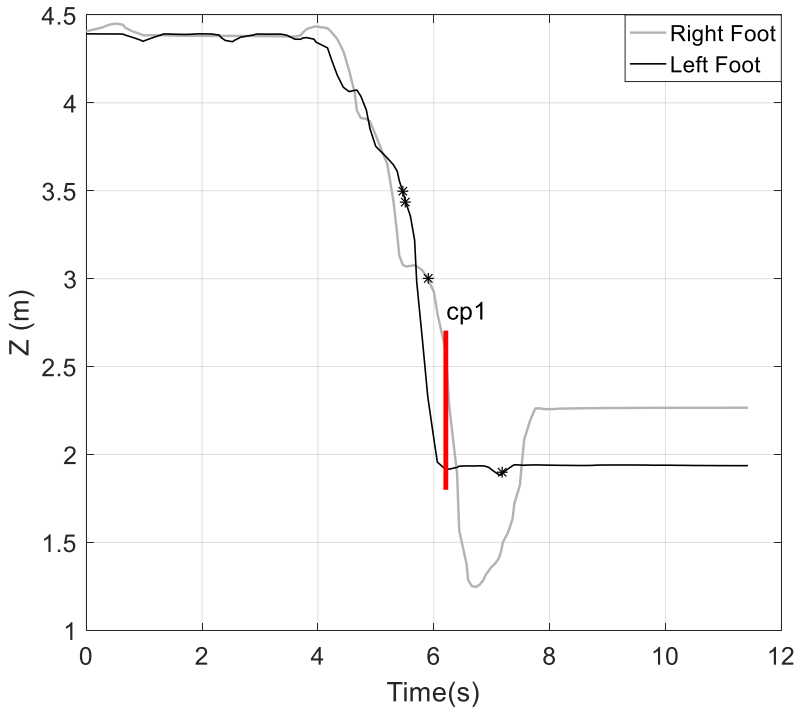

Figure 177. Variation along time of Z coordinates of feet and "cp1". The (*) represents the moment when the 3D position of the joint was inferred.

Note that before the kick, the leading foot alternates between right and left, which represents the participant's pace towards the ball. The algorithm checks if the approach to the ball is continuous by analyzing the time interval between the paces; if they are approximately the same, it means the movement is continuous — the same logic that was used for the fourth criterion of the gallop test. Within two paces, the distance between the feet reaches a maximum value; so, the algorithm checks for the time interval between two consecutive peaks in the graph shown in Figure 178. It also gives a tolerance of 30% that allows one peak to be apart from the next peak to still consider the movement rhythmical. Note in Figure 178 that the movement is considered continuous, since the time lapse within two consecutive maximum points does not exceed 30%.

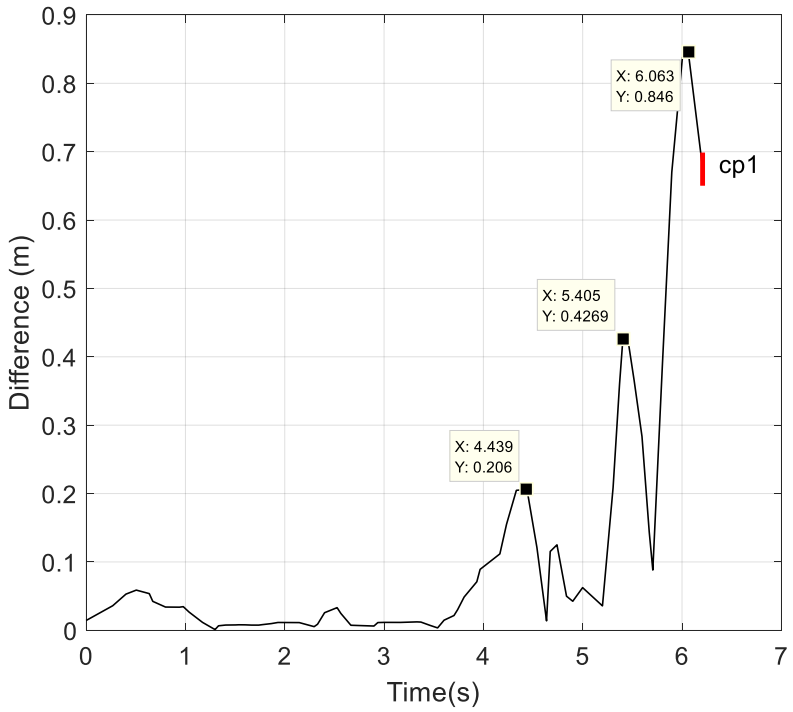

Figure 178. Variation along time of distance between feet along Z axis, its maximums and “cp1”.

In order to check if the approach to the ball is rapid enough, the algorithm checks the time lapse between the beginning of the movement and the moment when the kick happens; if kick happens within the first half of the movement’s total number of recording frames, the performance is considered rapid. Using that logic, it is possible to determine if the participant fails or passes the first criterion.

The following code snippet illustrates that:

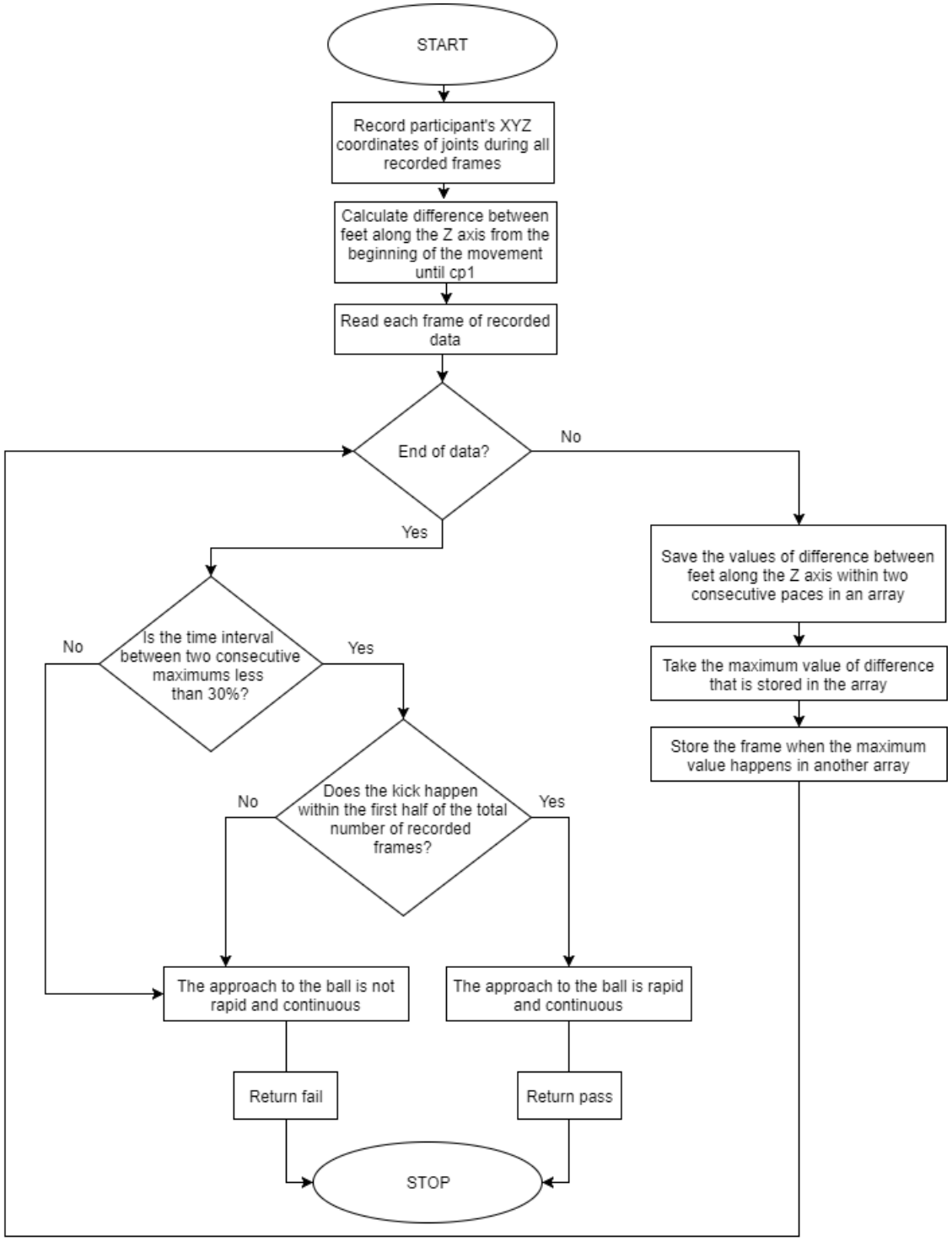

Figure 179. Algorithm to return feedback for the first criterion.

Criterion 2: Child takes an elongated stride or leap just prior to ball contact.

The second criterion checks if the participant takes an elongated stride or leap just prior to ball contact. Figure 180 shows the distance between the feet along the Z axis.

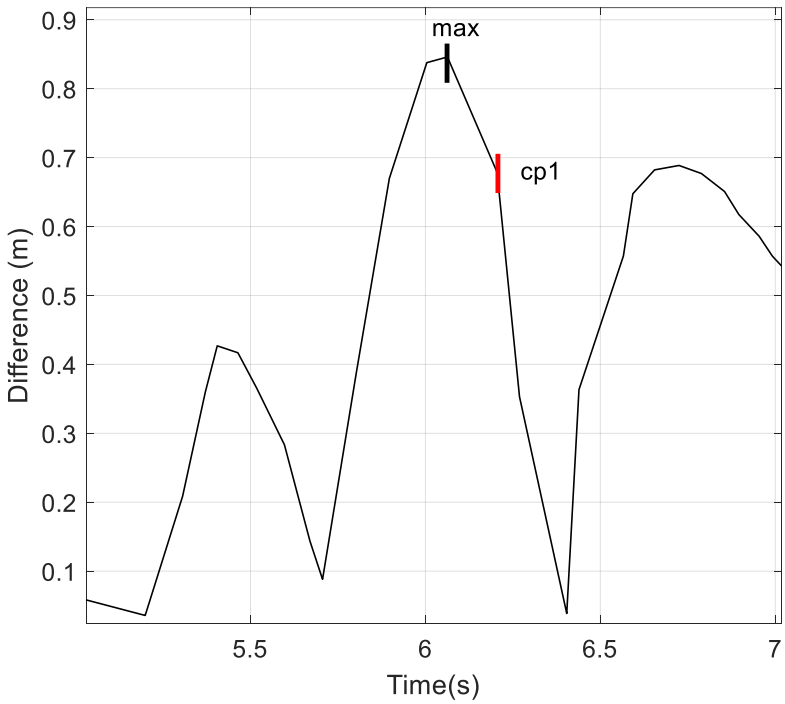

Figure 180. Distance between feet along the Z axis at "cp1" and its maximum point. The (*) represents the moment when the 3D position of the joint was inferred.

Note that the largest difference between the Z coordinate of the feet happens a few frames before the ball is kicked at "cp1". In this example, it happens one frame before the kick. This means that right before the kick happens, the participant takes an elongate stride/leap. Therefore, the algorithm checks if the maximum distance between the Z coordinate of the feet happens within a threshold of 5 frames prior to the kick and then returns pass or fail if the participant meets that requirement.

The flowchart that evaluates the criterion 2 is:

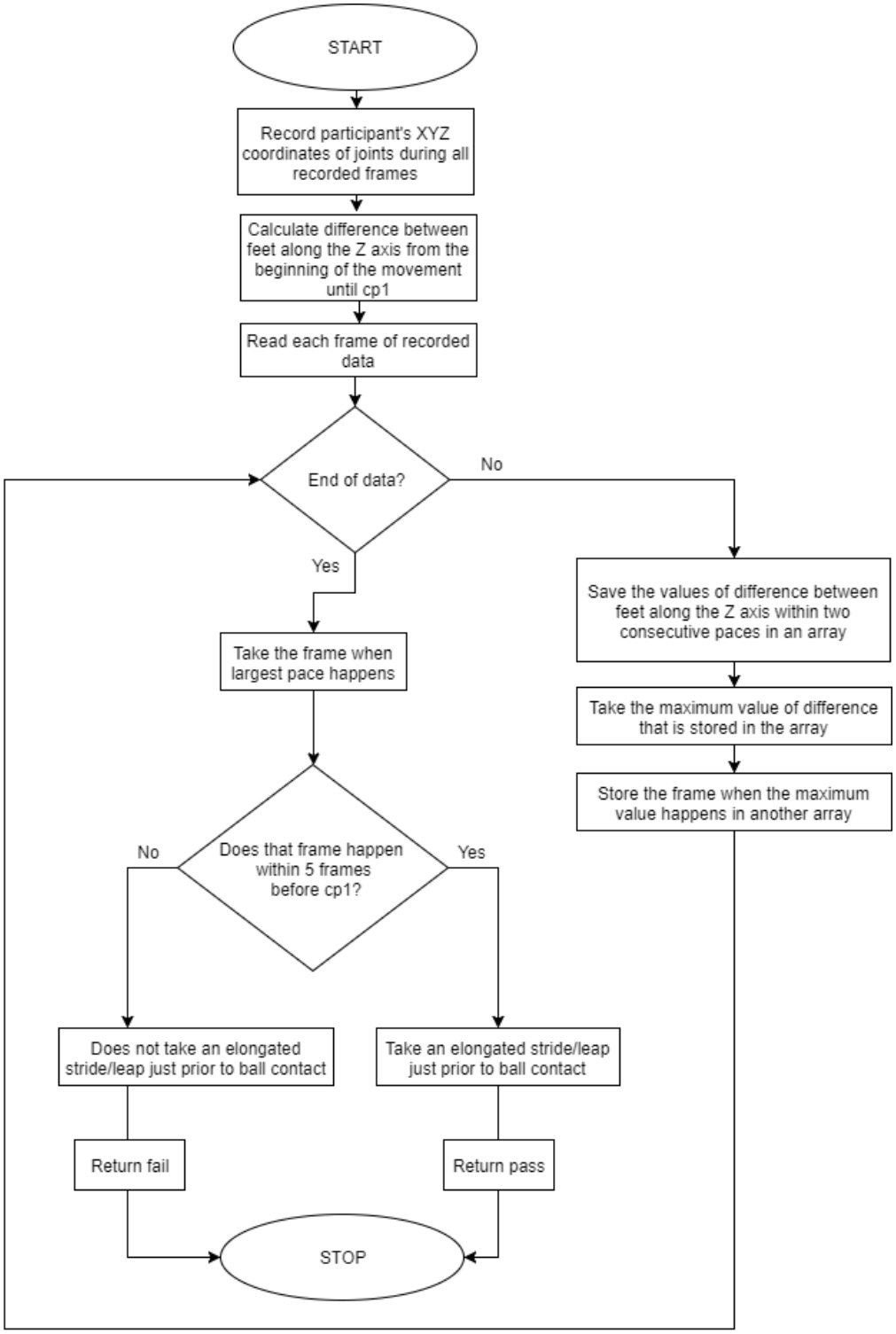

Figure 181. Algorithm to return feedback for the second criterion.

Criterion 3: Non-kicking foot placed close to the ball.

The third criterion checks if the non-kicking foot is placed close to the ball. In order to assess this criterion, the algorithm needs to first determine which foot performs the kick. Note in Figure 174 and Figure 177 that right after the kick, the kicking foot is the joint that reaches the closest distance from the Kinect sensor, i.e. it is the joint that reaches the minimum value for the Z coordinate within a few frames after the kick. Following this idea, the algorithm takes the minimum value for the Z coordinate reached by each foot within the period of 3 frames after the kick and compare; the foot that has the minimum value for the Z coordinate is the foot that kicks the ball. The following flowchart shows how to determine the kicking foot:

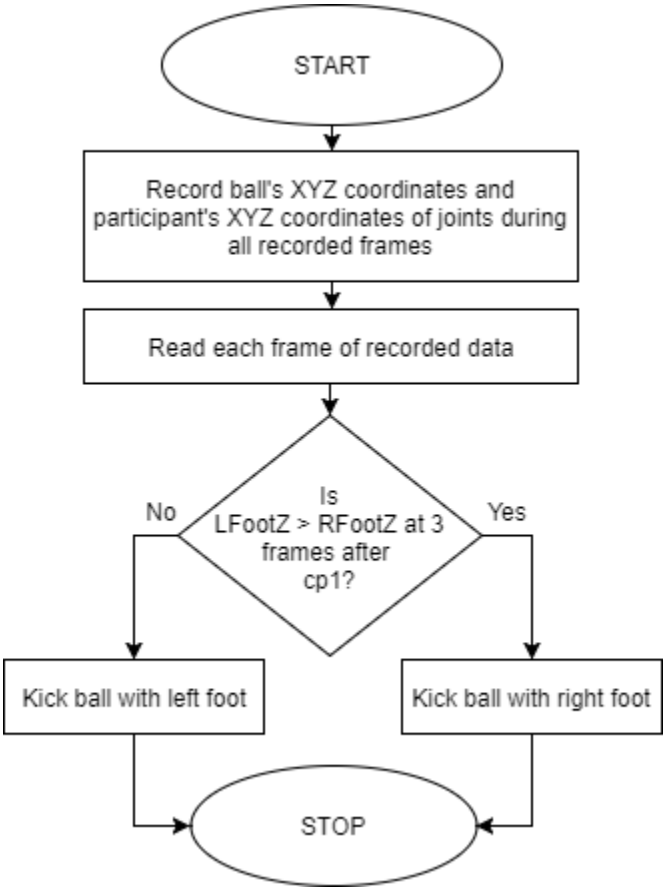

Figure 182. Algorithm to determine which foot kicks the ball.

After determining the kicking foot, the algorithm checks if the non-kicking foot is placed close to the ball prior to the kick, as is illustrated in frame 15 of Figure 174. Figure 183 illustrates the Z and X coordinates of the feet and ball when the participant kicks the ball with the right foot.

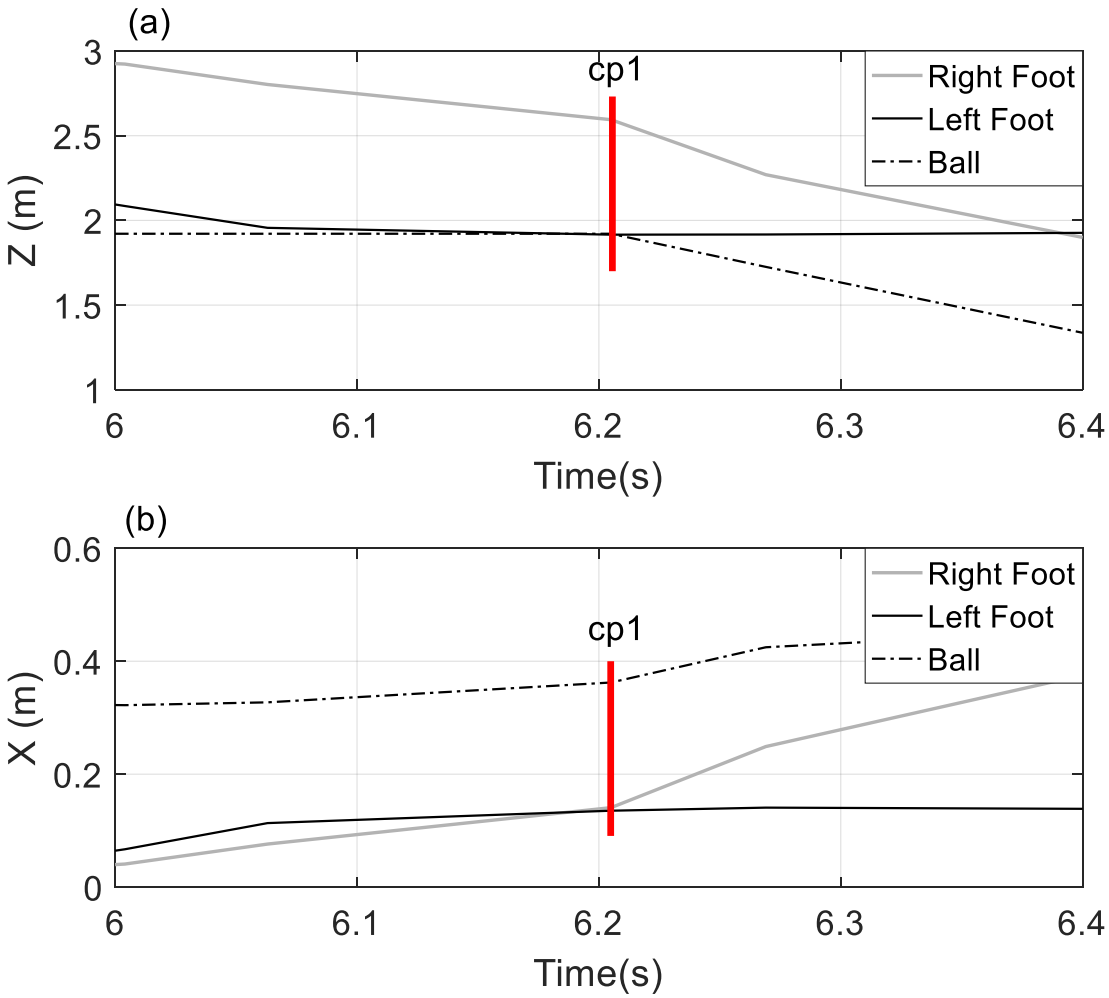

Figure 183. (a) The Z coordinates of ball and feet at “cp1”. (b) The X coordinates of ball and feet at “cp1”. The (*) represents the moment when the 3D position of the joint was inferred.

Note that the left foot is already placed closed to the ball, while the right foot performing the kick that happens at “cp1”. The algorithm then checks if at “cp1” the non-kicking foot is within a range of 15 cm apart from the ball along the Z axis and X axis. If the non-kicking foot meet this condition, it returns pass, otherwise fail.

The flowchart that shows how to evaluate the third criterion is:

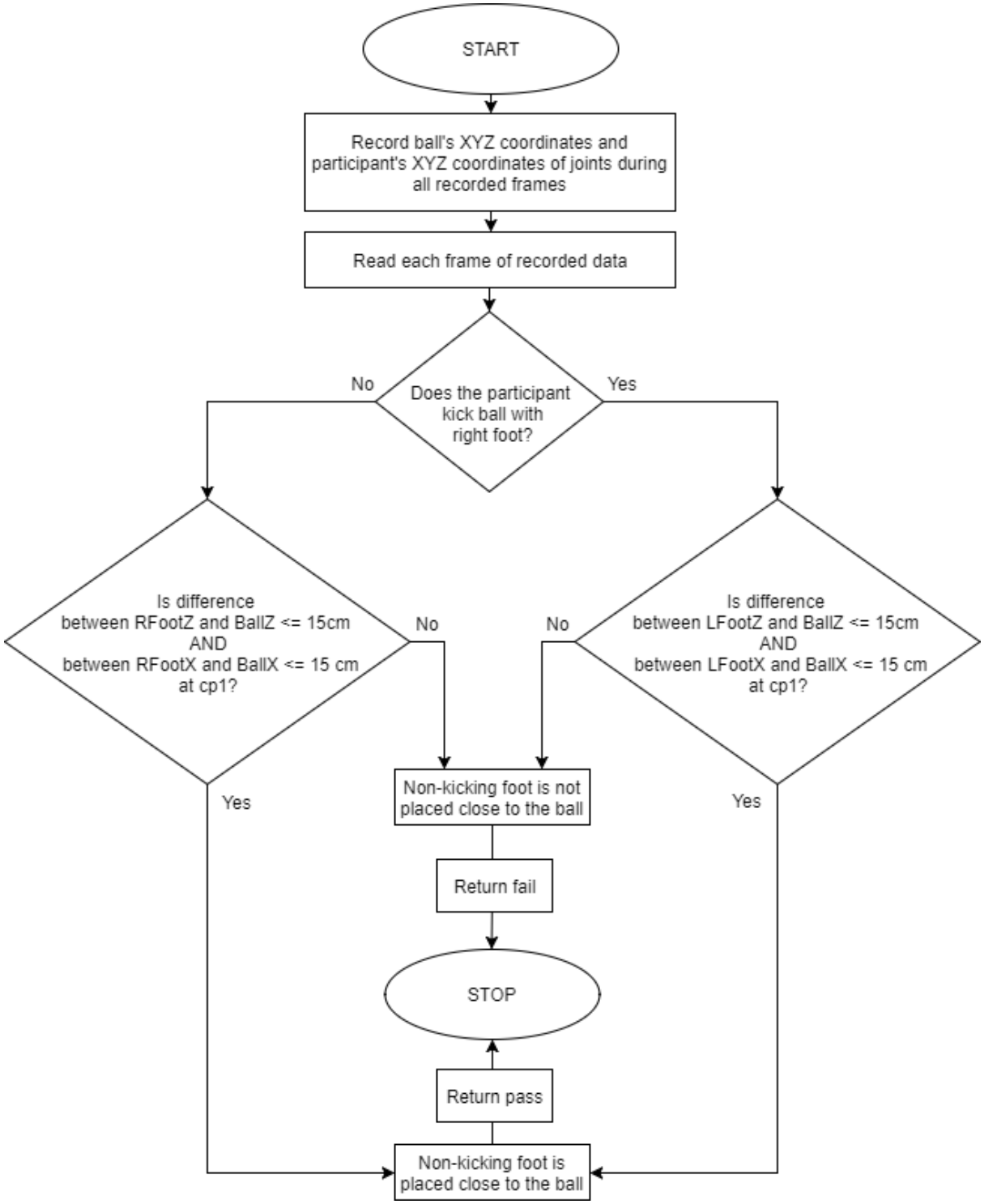

Figure 184. Algorithm to return feedback for the third criterion.

Criterion 4: Kicks ball with instep or inside of preferred foot (not the toes).

To assess this criterion the algorithm uses the information from the contact sensor to determine if the participant kicks the ball with his/her instep or the inside of their preferred foot. In order to check, the participant is asked to perform the movement with shoes that contain contact sensors covering the whole toes area, as illustrated in Figure 185.

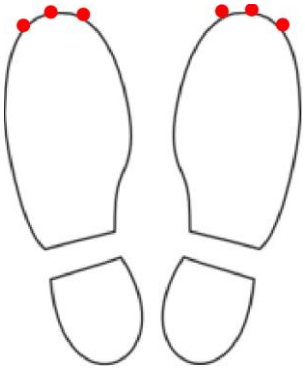

Figure 185. Representation of how the contact sensors are positioned on the shoes.

If the participant kicks the ball with his/her toes, the contact sensor sends logic level “1”, or logic level “0” otherwise. Therefore, the algorithm tracks if any logic level “1” is sent from any one of the contact sensors, and then returns pass or fail to the criterion according to it — similar logic was used for the second criterion of the one-hand dribble test. If any logic level “1” is sent, it means the participant kicks the ball with toes, and for this reason s/he fails the criterion.

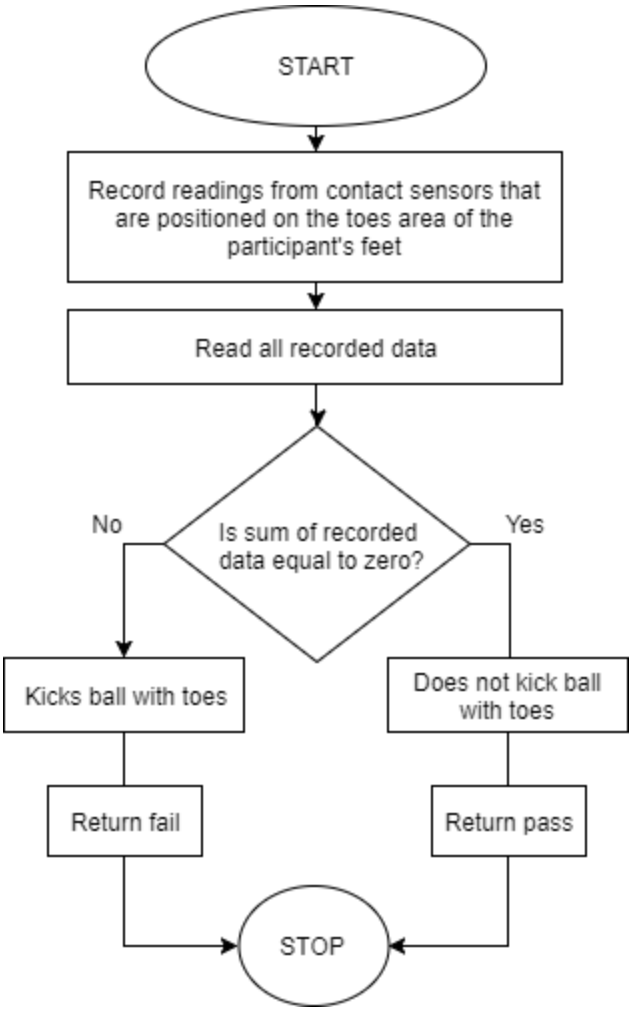

Figure 186. Algorithm to return feedback for the fourth criterion.

4.3.12 Overhand throw

An example of an overhand throw movement is shown in Figure 187 below, where the Kinect is positioned facing the participant.

Figure 187. The sequence of images to represent a standardized overhand throw movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

According to (Seroyer, et al., 2010) “during windup and stride, the pitcher keeps his center of gravity back (over stance leg) for as long as possible to allow maximum generation and transfer of momentum and force to the upper extremity and ball.” Note in Figure 187 that during the windup phase, the participant’s Z coordinate of the preferred hand reaches the furthest distance from the Kinect sensor (frame 8), in order to dislocate the center of gravity back. That moment, i.e. when the Z coordinate of the participant’s preferred hand reaches the furthest distance from the Kinect, marks the first critical point “cp1”, which indicates the beginning of the movement. Prior to the analysis of the movement, the participant is asked his/her preferred hand and the answer is given as an input to the algorithm. Note in Figure 188 that the Z coordinate of the preferred hand (the right hand) reaches the maximum value at “cp1”.

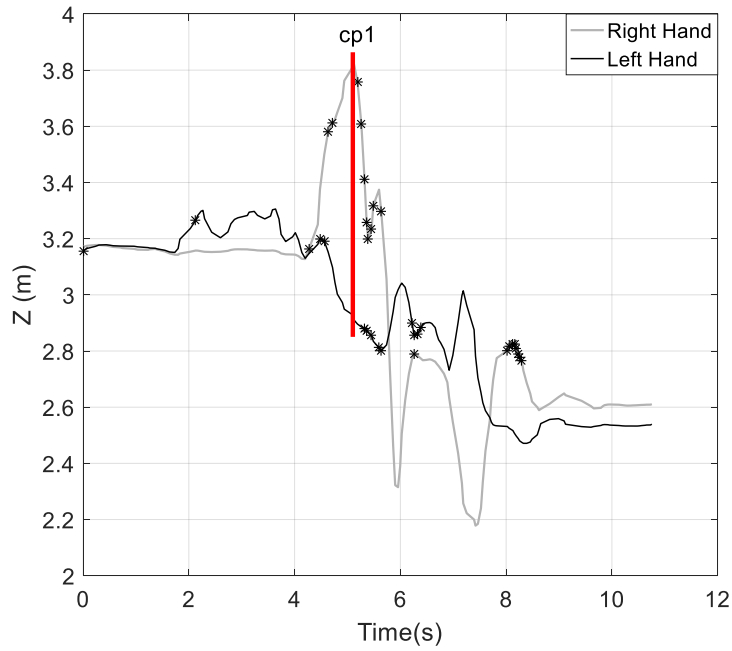

Figure 188. Variation along time of Z coordinates of hands and "cp1". The (*) represents the moment when the 3D position of the joint was inferred.

The flowchart that illustrates how to check for preferred hand and "cp1" is:

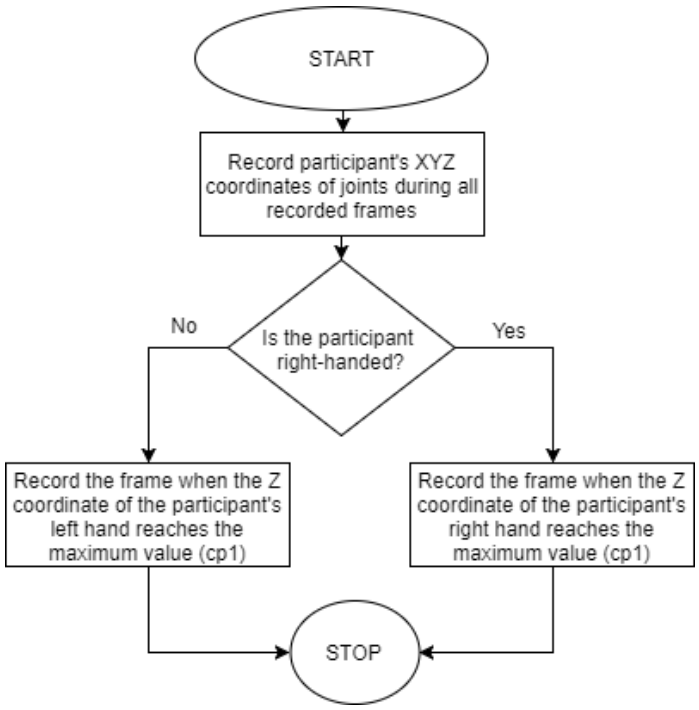

Figure 189. Algorithm to determine cp1.

Criterion 1: Windup is initiated with a downward movement of hand and arm.

The first criterion for the overhand throw consists of analyzing if the windup is initiated with a downward movement of hand and arm of the preferred hand. Figure 190 shows the behavior of the hands along the Y coordinate when the participant performs the test with his/her right hand.

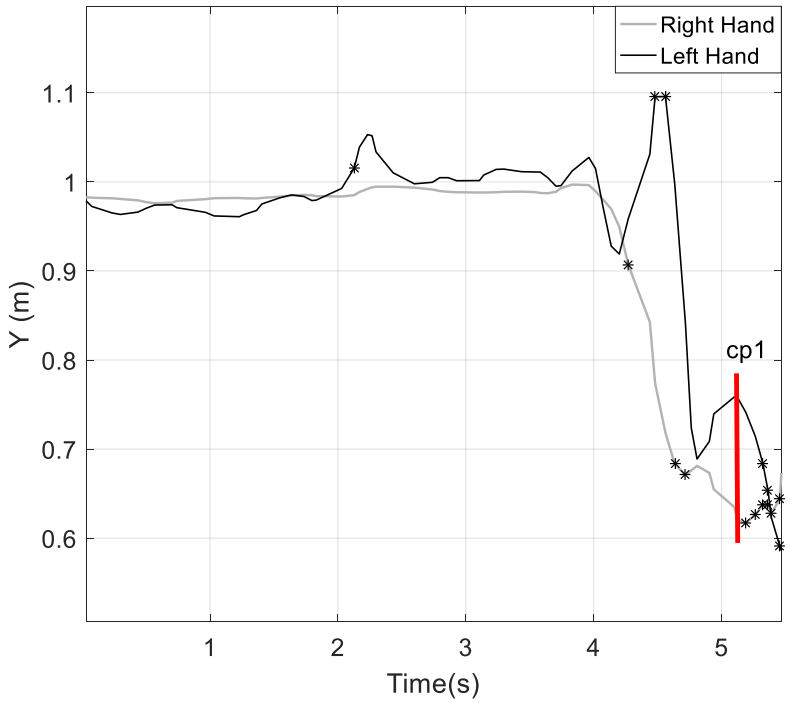

Figure 190. The Y coordinates of hands at "cp1". The (*) represents the moment when the 3D position of the joint was inferred.

Note that the Y coordinate of his/her preferred hand drops from the beginning of the movement until "cp1". Therefore, the algorithm checks if at "cp1" the Y coordinate of preferred hand is below the initial value given in the first frame. If the condition is satisfied, the participant passes the criterion.

The flowchart for the first criterion is:

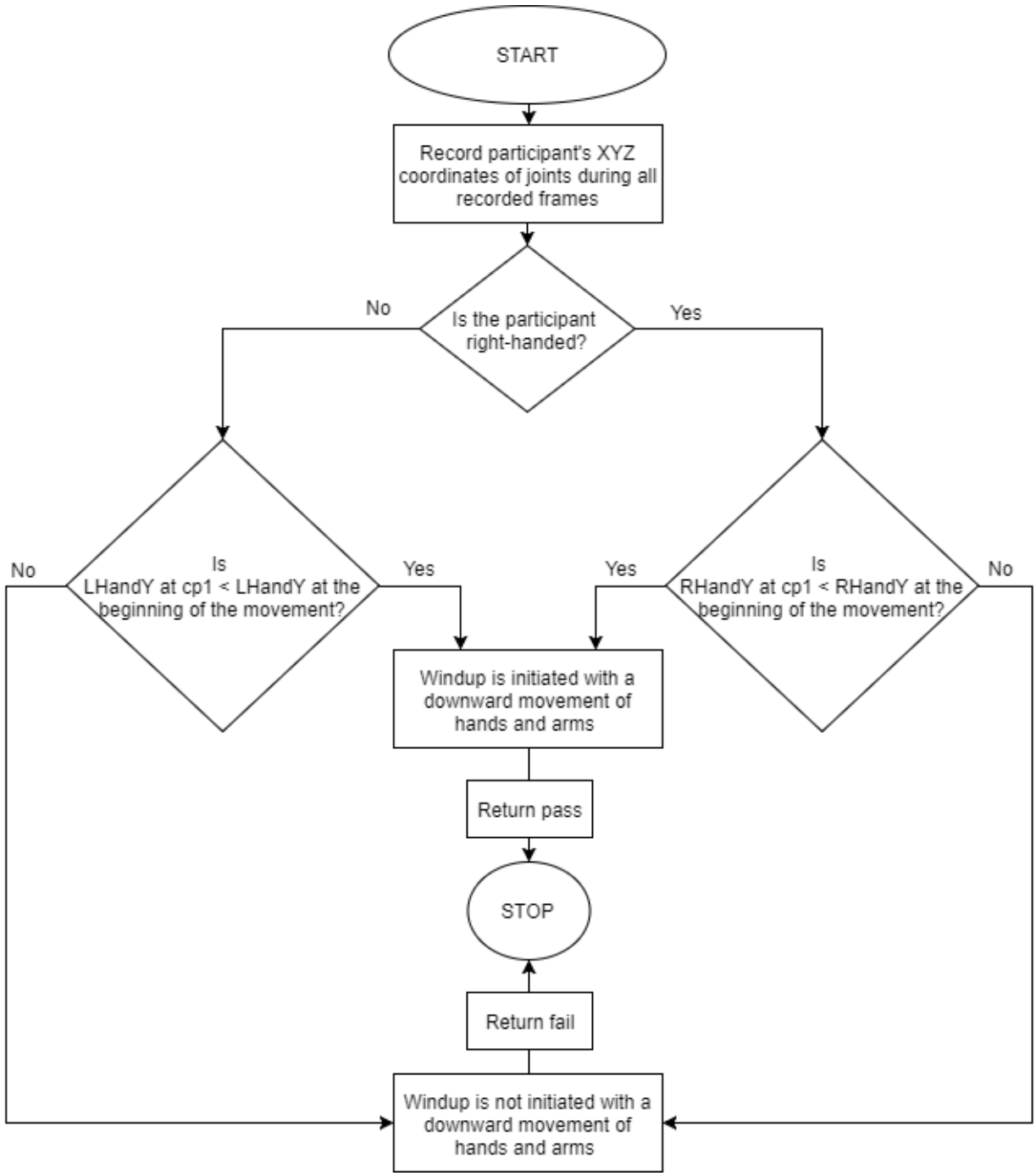

Figure 191. Algorithm to return feedback for the first criterion.

Criterion 2: Rotates hip and shoulder to a point where the non-throwing side faces the wall.

The second criterion analyses if the participant rotates his/her hip and shoulder to a point where the non-throwing side faces the wall. As the movement is performed facing the Kinect, at "cp1" both Z coordinates of the hip and shoulder of the throwing side must be greater than the Z coordinates of the non-throwing side, which means that non-throwing side faces the Kinect. Figure 192 shows that situation.

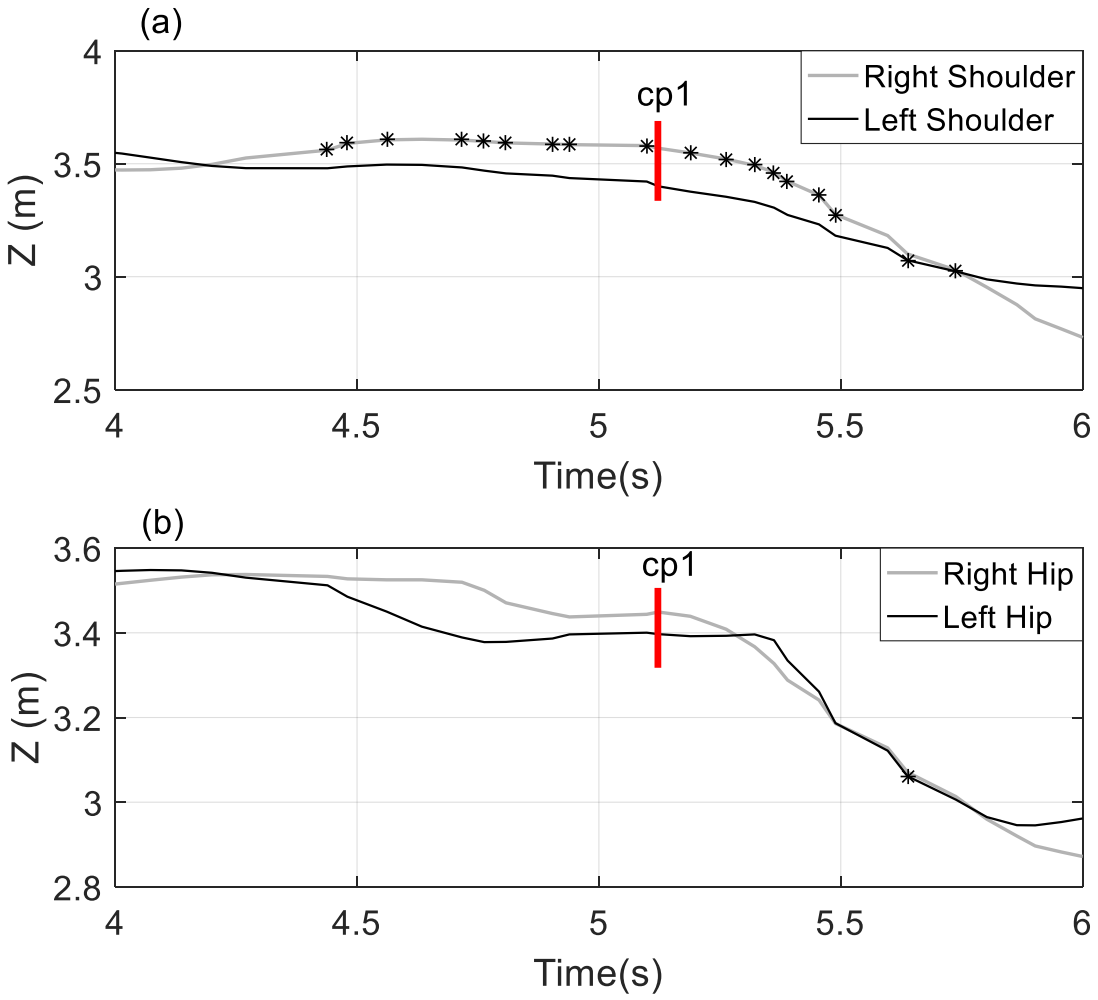

Figure 192. (a) Variation along time of Z coordinates of shoulders and "cp1". (b) Variation along time of the Z coordinates of hips and "cp1". The (*) represents the moment when the 3D position of the joint was inferred.

Note that initially both Z coordinates are approximately the same, since the participant is facing the Kinect without any rotation of hips and shoulders. When the participant rotates the body so that the non-throwing side faces the Kinect, the Z coordinates of non-throwing side diminish, since they move closer to Kinect, and the Z coordinates of throwing side increase, since they move further from Kinect. Therefore, the algorithm checks if at “cp1” the Z coordinates of the hip and shoulder of the non-preferred side are lower than the respective coordinates of the preferred side. By doing this analysis at “cp1”, the algorithm checks if the participant passes the criterion or not.

The flowchart for the second criterion is:

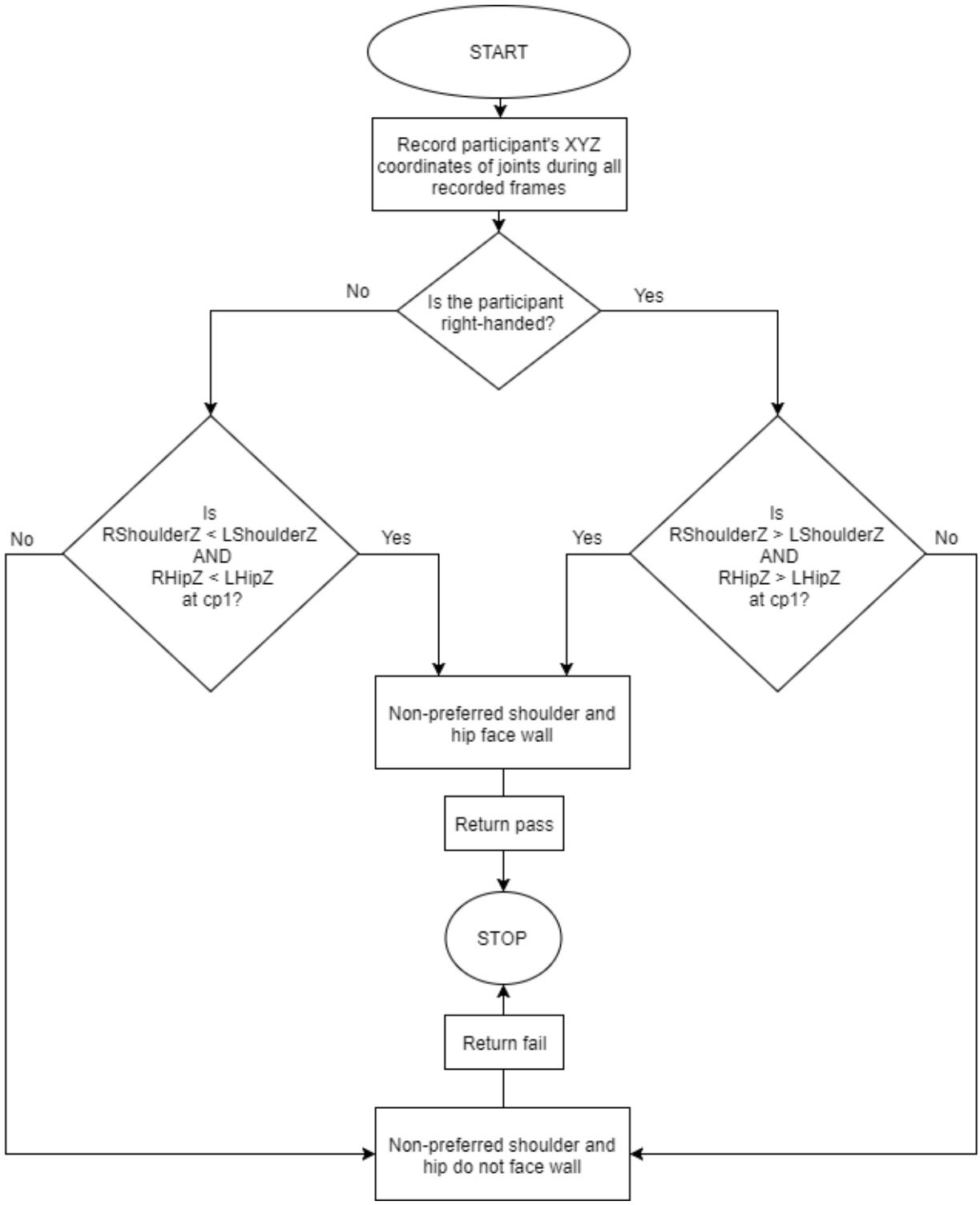

Figure 193. Algorithm to return feedback for the second criterion.

Criterion3: Steps with the foot opposite the throwing hand toward the wall.

The third criterion checks if the participant steps with the foot opposite the throwing hand toward the wall. Note that at "cp1", the participant is ready to throw the ball, so the step will happen at any moment after "cp1". Also note that when the participant throws the ball, it will travel towards the Kinect until it gets out of the sensor's field of view, which is moment when the ball coordinates are equal to zero, as illustrated by the second critical point "cp2" in Figure 194.

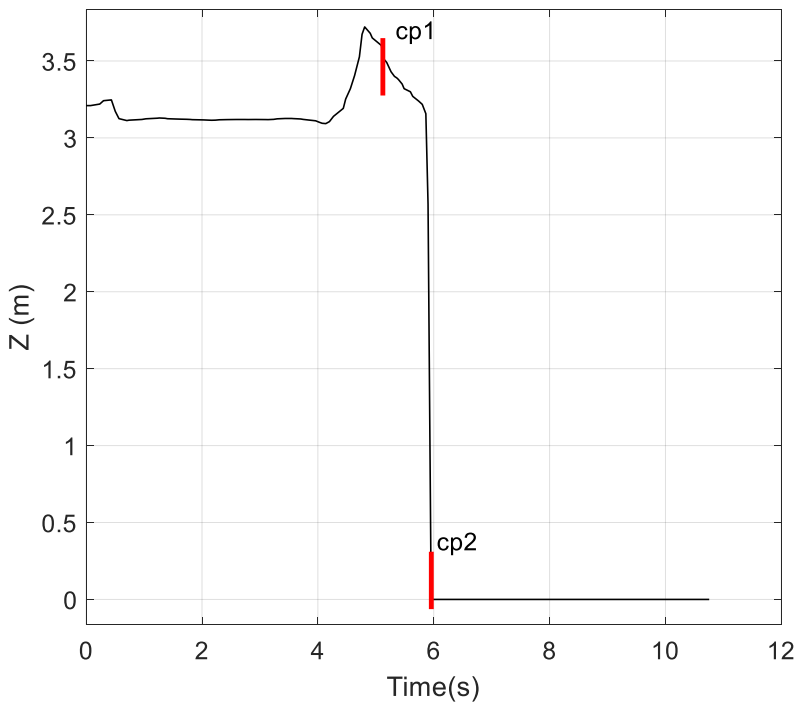

Figure 194. Variation along time of the Z coordinates of ball and critical points "cp1" and "cp2".

Therefore, a step towards the wall will happen within the interval between "cp1" and "cp2". It is important to mention that the wall will be anywhere behind the Kinect sensor.

Figure 195 shows the behavior of the Z coordinate of feet in between "cp1" and "cp2".

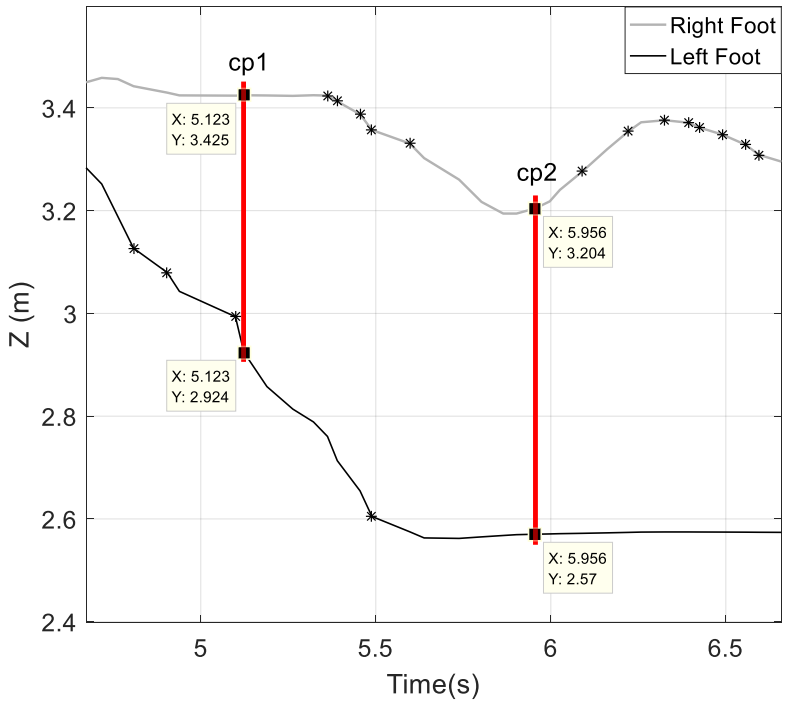

Figure 195. Variation along time of Z coordinates of feet and critical points “cp1” and “cp2”. The (*) represents the moment when the 3D position of the joint was inferred.

Note that a step is performed within that interval, and it is determined by the moment when one of the feet stays placed on the floor while the other one moves, performing the step. Therefore, by analyzing the slope of the curve of both feet and by knowing the preferred side for the movement, the algorithm can determine if a step was performed within “cp1” and “cp2”. In addition, it also checks if at “cp2”, the Z coordinate of the stepping foot is closer to the sensor than it was at “cp1”, that way the algorithm knows that the step was performed towards the wall.

The flowchart for the third criterion is:

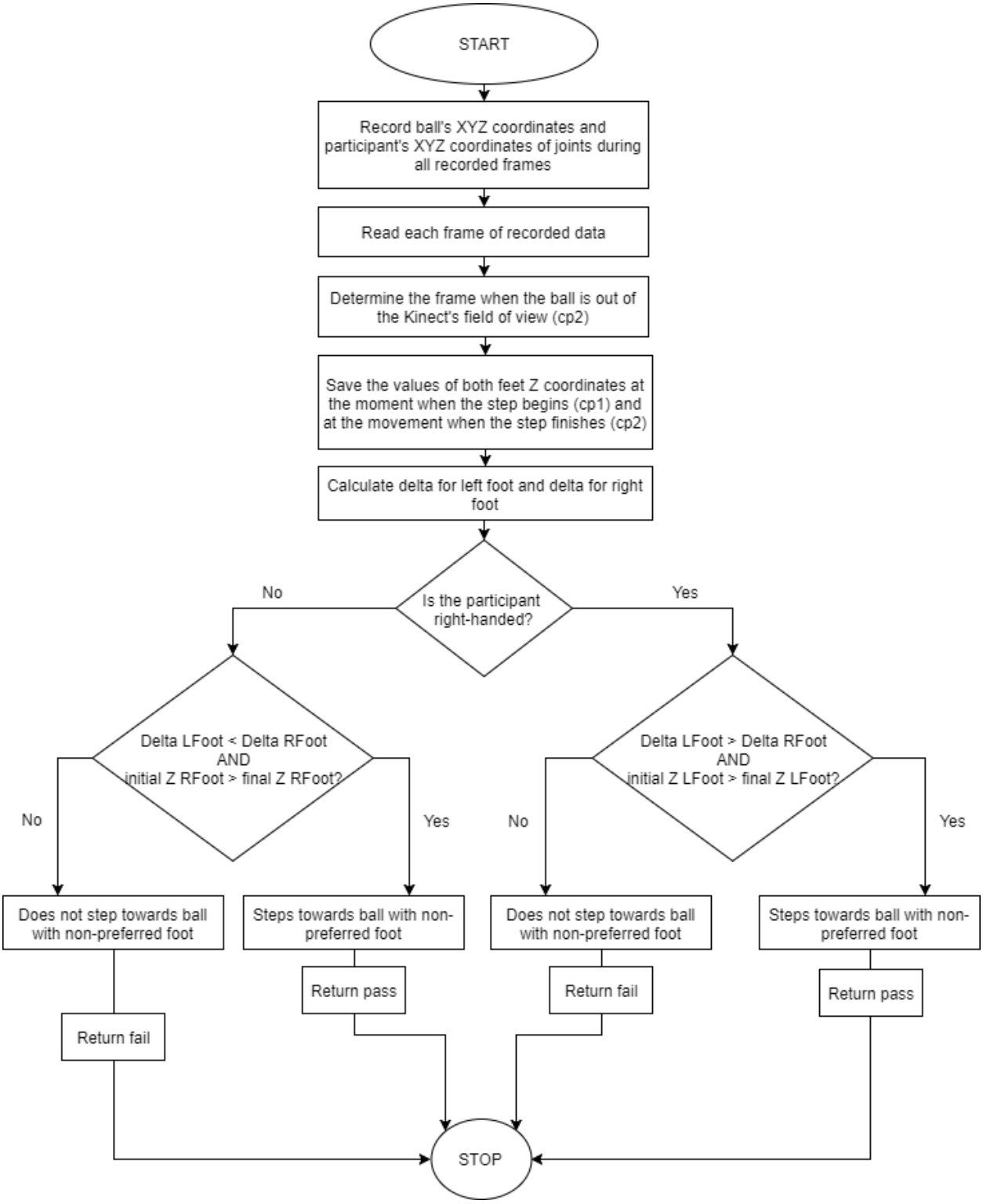

Figure 196. Algorithm to return feedback for the third criterion.

Criterion 4: Throwing hand follows through after the ball. Release across the body toward the hip of the non-throwing side.

The last criterion checks if the throwing hand follows through, after the ball is released, across the body toward the hip of the non-throwing side. The algorithm considers that the ball is released at "cp2", the moment when it goes out of the range of the sensor.

As the X axis divides the right side of the sensor as positive and left side as negative, the algorithm checks if after "cp2", the throwing hand reaches lower values than the hip of non-preferred side, in case the participant is right-handed; or if the throwing hand reaches greater values than the hip of non-preferred side, in case the participant is left-handed. Figure 197 illustrates this situation.

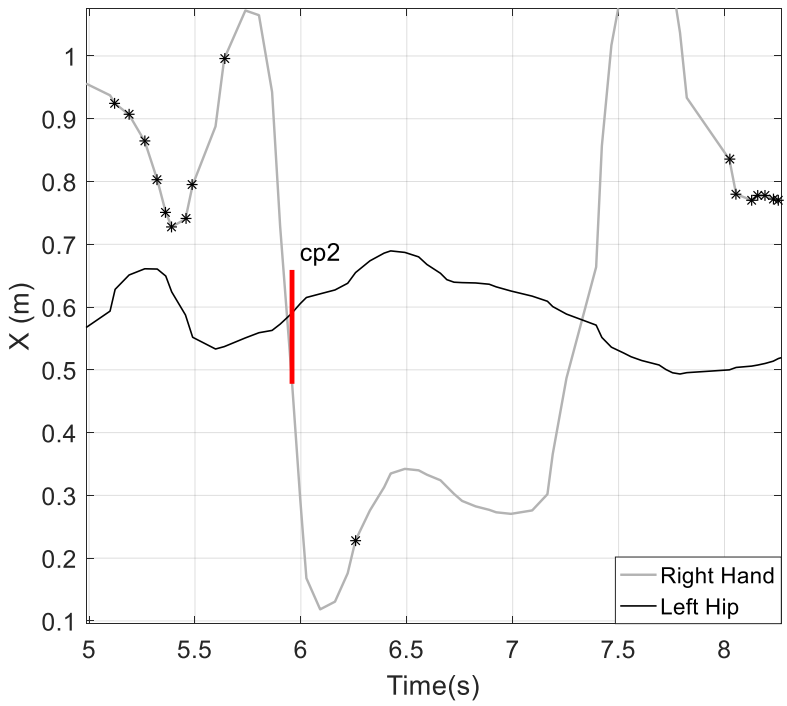

Figure 197. X coordinates of right hand and left hip and "cp2". The (*) represents the moment when the 3D position of the joint was inferred.

Note that since the beginning of the movement, the right hand shows greater values for X coordinates if compared to the left hip. Since the right hand is to the right side of the left hip, its coordinates are more positive than the left hip coordinates. The moment when the right hand follows through, crossing the body to the non-throwing side after the ball is released, for some period of time, its X coordinates are lower than the X coordinates of left hip, which means for that period of time the right hand is to the left side of the left hip. Therefore, to access this last criterion, the algorithm checks if after “cp2” the x coordinate of the throwing hand crosses the X coordinate of the non-throwing hip at least once.

The flowchart for the last criterion is:

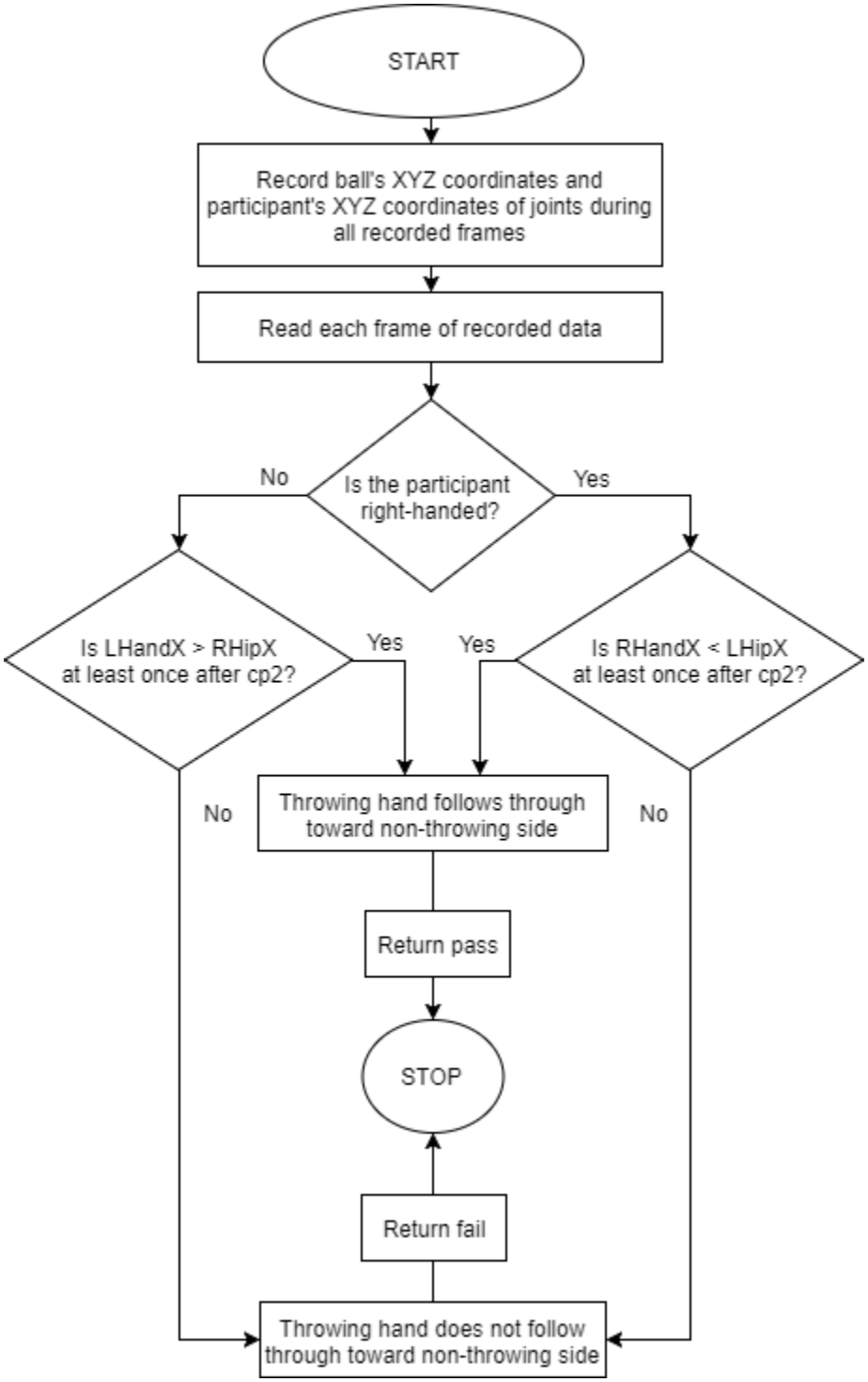

Figure 198. Algorithm to return feedback for the fourth criterion.

4.3.13 Underhand throw

An example of an underhand throw movement is shown in Figure 199 below, where the Kinect is positioned facing the participant.

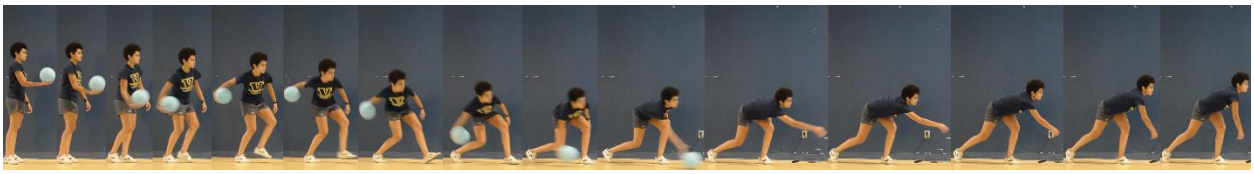

Figure 199. The sequence of images to represent a standardized underhand throw movement. The participant performed the movement according to the performance criteria described in the TGMD-3 scoring sheet.

Prior to the analysis of the movement, the participant is asked his/her preferred hand and the answer is given as an input to the algorithm. Note in Figure 199 that during a standardized performance of an underhand throw, the preferred hand reaches its furthest distance from the sensor at the moment when the participant is ready to initiate the throw; this moment is the first critical point "cp1" for the movement.

Criterion 1: Preferred hand swings down and back reaching behind the trunk.

The first criterion checks if the participant's preferred hand swings down and back reaching behind the trunk. Figure 200 illustrates the behavior of both hands and mid spine along the Z axis when the participant performs the movement with the right hand.

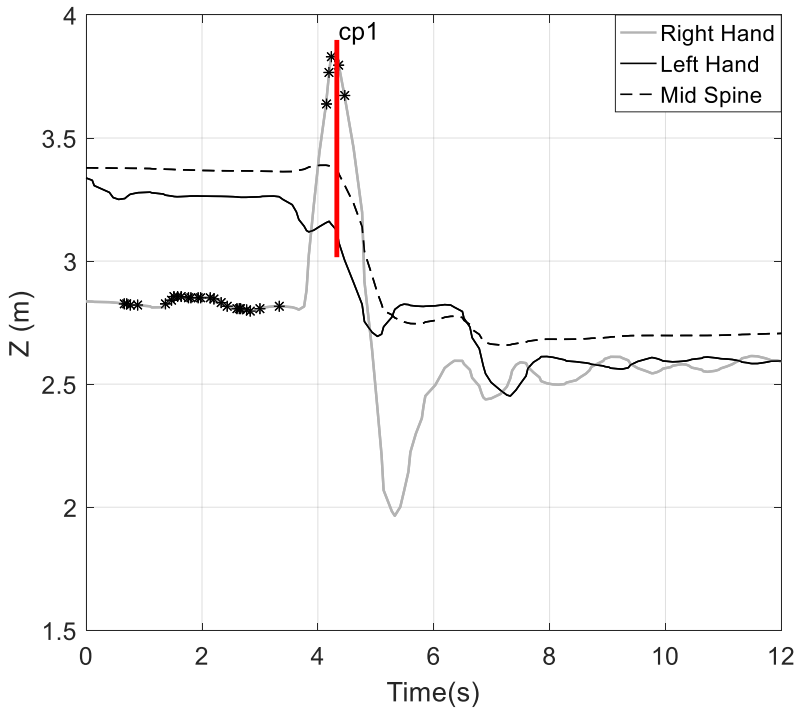

Figure 200. Variation along time of Z coordinates of hands and mid spine and "cp1". The (*) represents the moment when the 3D position of the joint was inferred.

Note that the right hand, i.e. the preferred one, is behind the mid-spine at "cp1", i.e. the right hand is further away from the Kinect than the mid-spine at "cp1", which is used as a reference for the trunk.

Therefore, the algorithm takes the moment when the maximum distance for the Z coordinate is reached by the preferred hand to identify "cp1". Then, it uses the mid-spine position as a reference for the trunk and checks if at "cp1" the preferred hand reaches at least 20 cm behind the mid spine. If that condition if true, it means that the preferred hand reaches behind the trunk.

In order to check if the preferred hand swings down, the algorithm analyses its variation along the Y axis. Note in Figure 201 that the preferred hand performs a decline along the Y axis before "cp1" and at "cp1", its Y coordinate is lower than the initial value at the beginning of the movement.

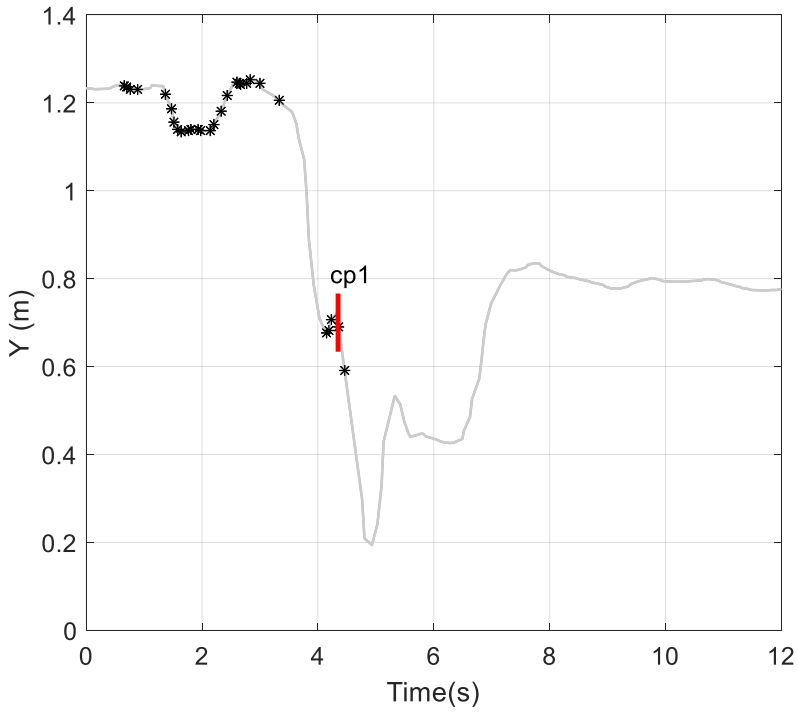

Figure 201. Variation along time of the Y coordinate of preferred hand and “cp1”. The (*) represents the moment when the 3D position of the joint was inferred.

In this way, the algorithm compares the Y coordinate of the preferred hand at “cp1” to the Y coordinate of the hand during the initial frame. If the Y coordinate at “cp1” is lower, it means that the participant’s hand performed a downswing.

By analyzing those variations along Y and Z, the algorithm can determine if the participant passes or fails the first criterion, as shown in the following flowchart:

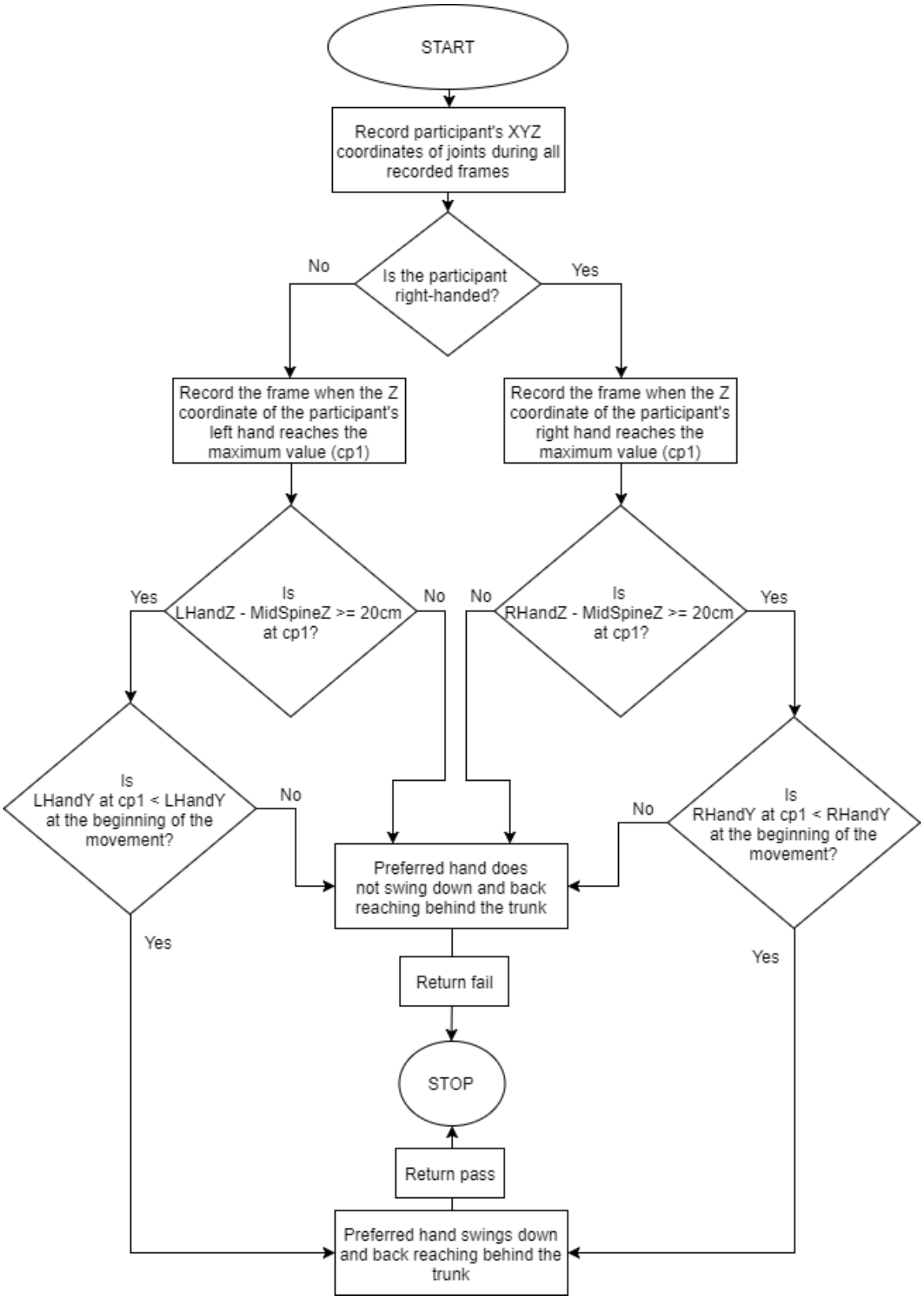

Figure 202. Algorithm to return feedback for the first criterion.

Criterion 2: Steps forward with the foot opposite the throwing hand.

The second criterion analyses if the participant steps forward with the foot opposite the throwing hand. Note in Figure 200 that at “cp1”, the preferred hand reaches the furthest distance from the sensor, and that moment characterizes the beginning of the throw. Then, the preferred hand reaches the chest level at critical point 2 (“cp2”). Considering the mid-spine as a reference for chest level, at “cp2” the Z coordinate of the preferred hand is at the same distance as the Z coordinate of the mid spine; therefore “cp2” can be estimated by the moment when the distance between preferred hand and mid-spine, along the Z axis, reaches the lowest value after “cp1”. The following flowchart shows how to determine “cp2”:

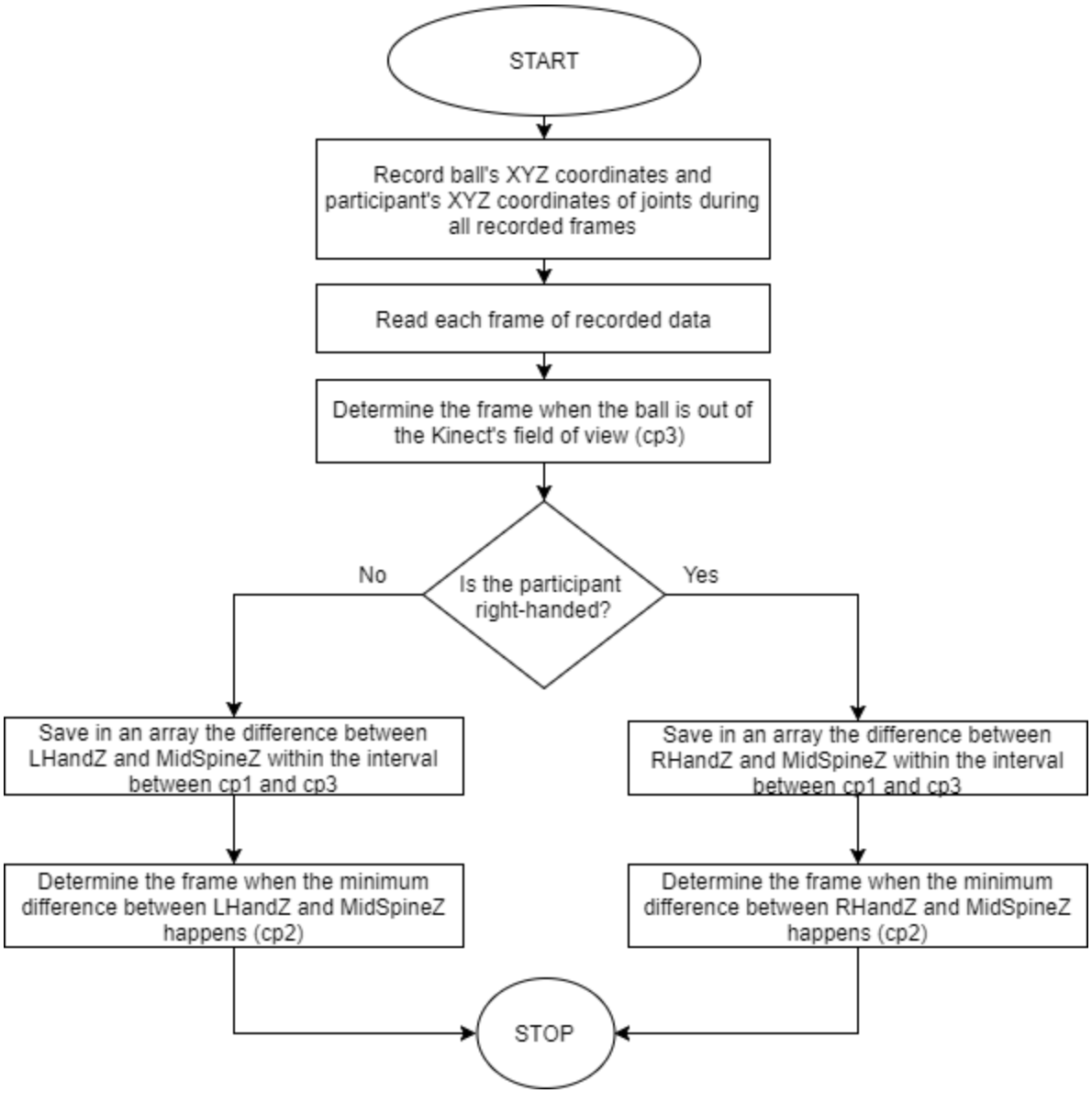

Figure 203. Algorithm to determine cp2.

Moreover, as shown in Figure 199, the step is performed in between “cp1”, the moment when the participant is ready to initiate the throw, and “cp2”, moment when the participant releases the ball crossing the chest level. Therefore, in order to check if the participant steps forward with a foot opposite to the throwing hand, the algorithm compares the slope of the curve of both feet and the

final Z coordinate of the stepping one, as was done previously for other tests. Figure 204 illustrates the slope of the curve for each foot.

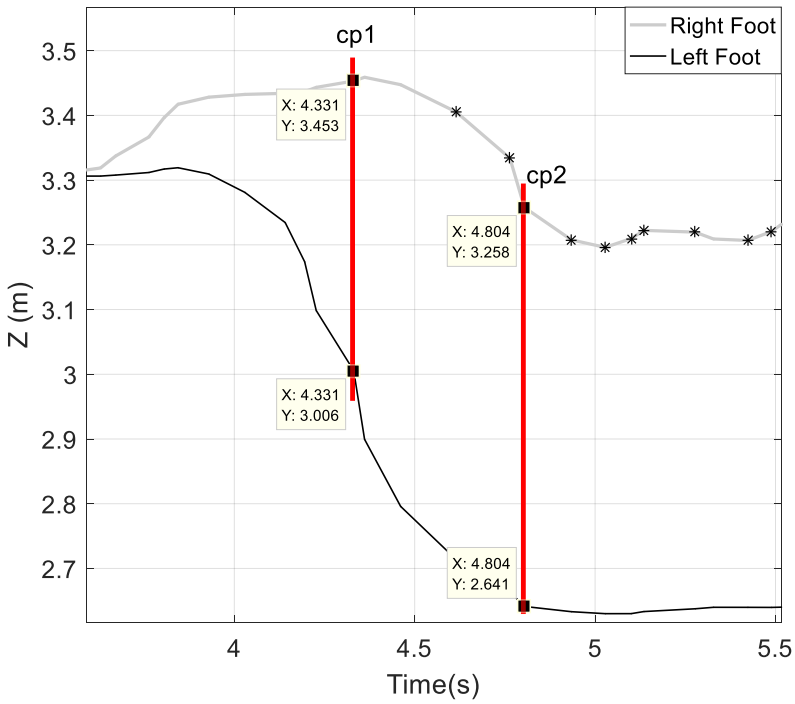

Figure 204. The Z coordinates of feet between “cp1” and “cp2”. The (*) represents the moment when the 3D position of the joint was inferred.

Note that the stepping foot, in this example the left one, performs the step and its slope is sharper than the slope of the non-stepping foot.

The following flowchart illustrates the analysis of the second criterion:

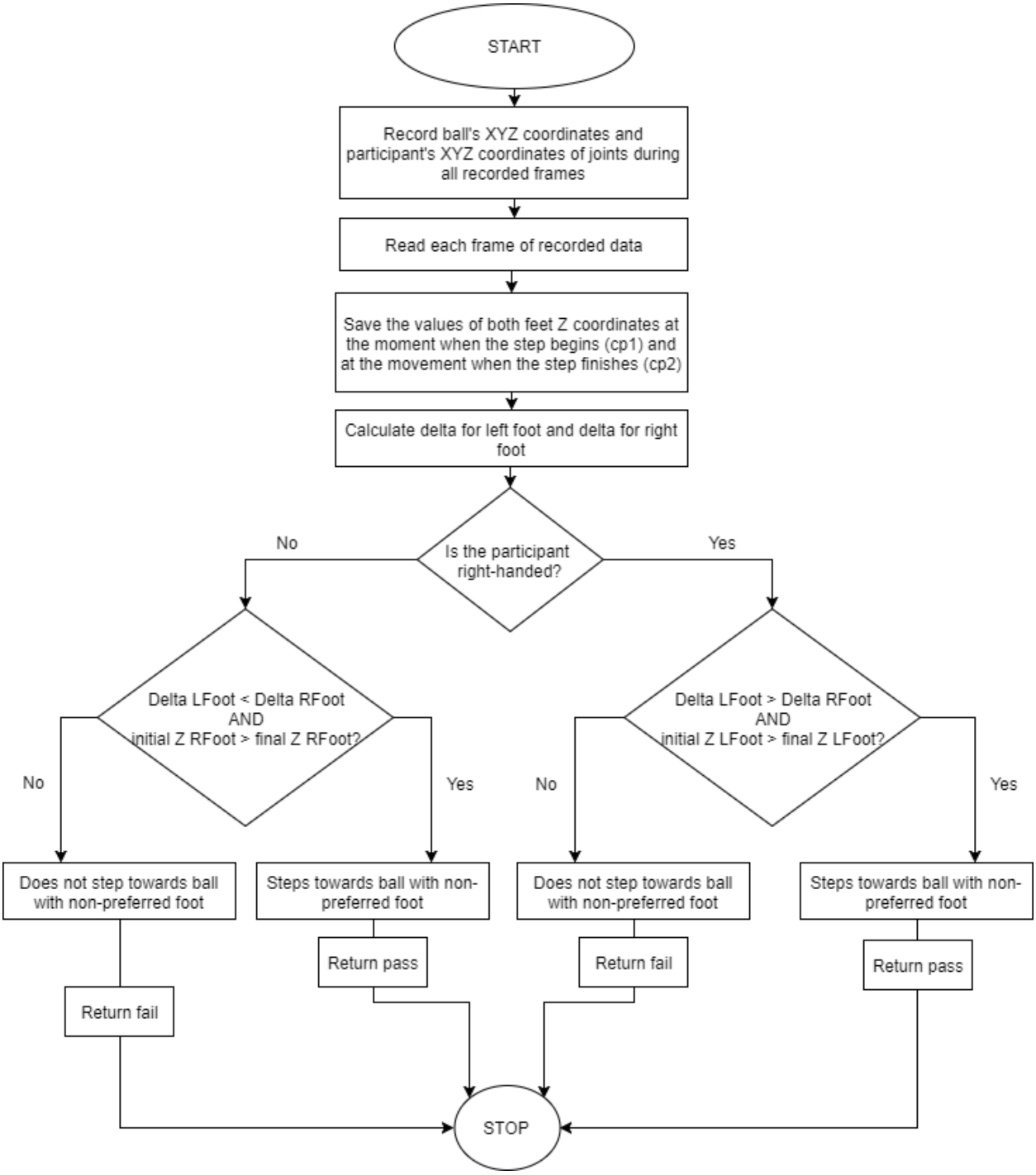

Figure 205. Algorithm to return feedback for the second criterion.

Criterion 3: Ball is tossed forward hitting the wall without a bounce.

The third criterion consists of checking if ball is tossed forward hitting the wall without a bounce. Figure 206 shows the Y coordinate of the ball from “cp1” until it goes out of the range of the sensor at “cp3”.

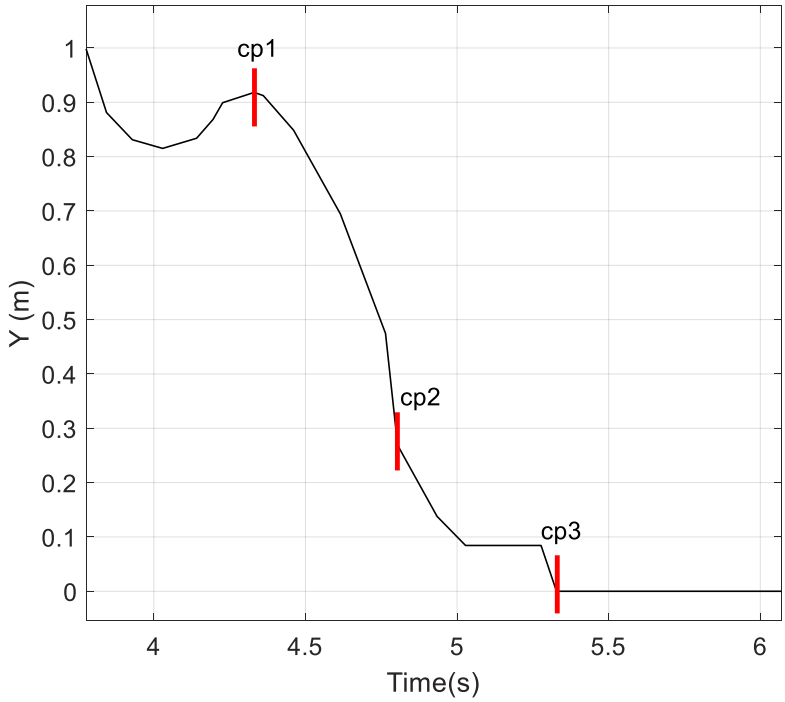

Figure 206. The Y coordinates of ball at “cp1”, “cp2” and “cp3”.

According to the characteristic of the underhand throw movement, the ball is released from the hand contact at any moment after the throwing hand reaches the chest level. The algorithm then checks the behavior of the Y coordinate of the ball from “cp2” until the ball goes out of the range of the Kinect at “cp3”. Note that from “cp2” to “cp3”, the Y coordinates of the ball do not draw any peaks, which means the ball does not bounce.

The following flowchart illustrates the analysis of the third criterion:

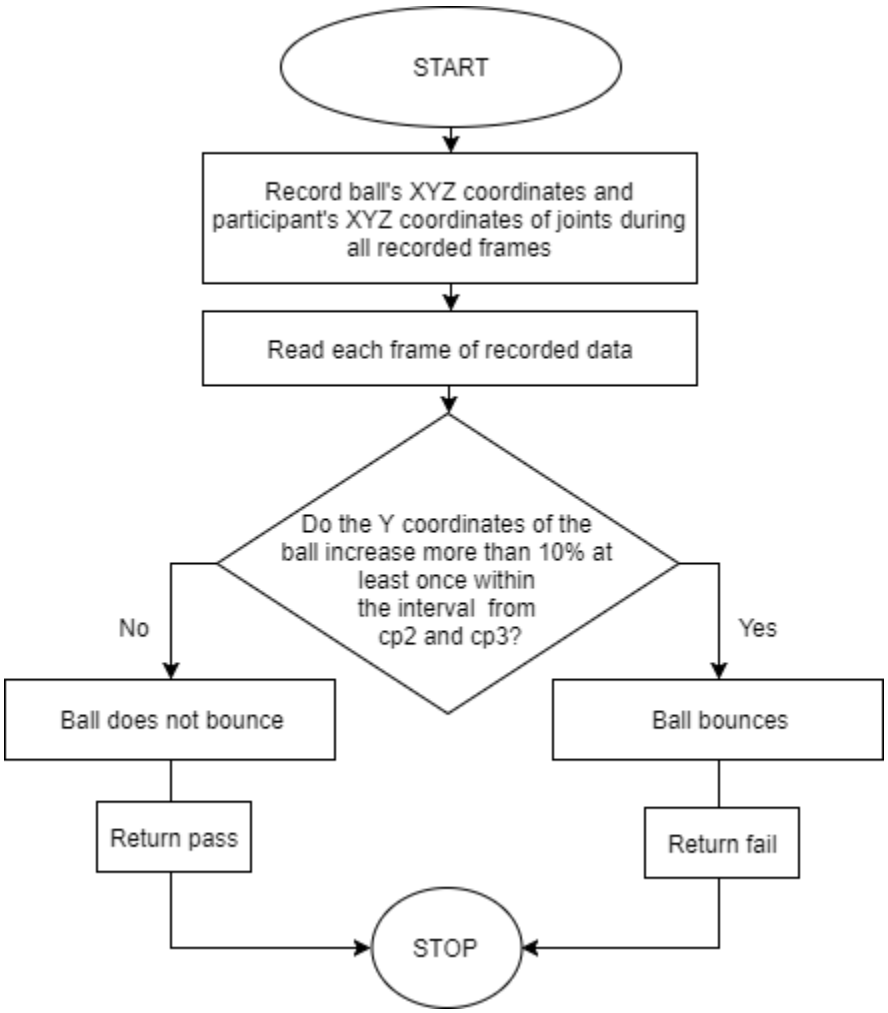

Figure 207. Algorithm to return feedback for the third criterion.

<u>Criterion 4: Hand follows through after ball release to chest level.</u>

The last criterion checks if throwing hand follows through after the ball release to chest level. Figure 208 shows that after the throwing hand, in this example the right one, reaches the chest level at “cp2”, its Z coordinate continues decreasing, getting closer to the Kinect sensor.

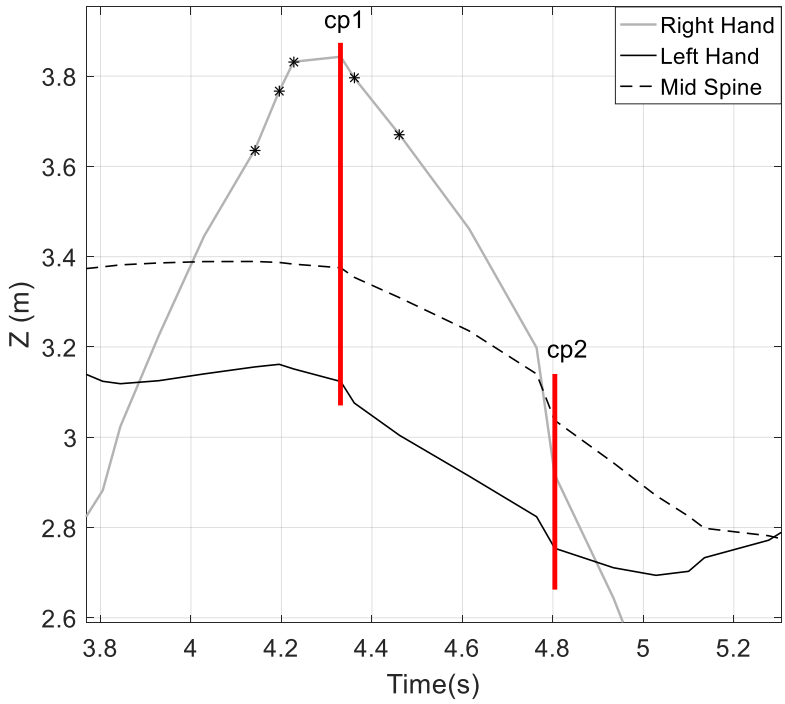

Figure 208. The Z coordinates of hands and mid-spine between "cp1" and "cp2". The (*) represents the moment when the 3D position of the joint was inferred.

This behavior indicates that the throwing hand follows through after ball is released. Therefore, the algorithm checks if the Z coordinate of the preferred hand continues decreasing a few frames after "cp2". If that condition is met, the software returns pass, otherwise it returns fail.

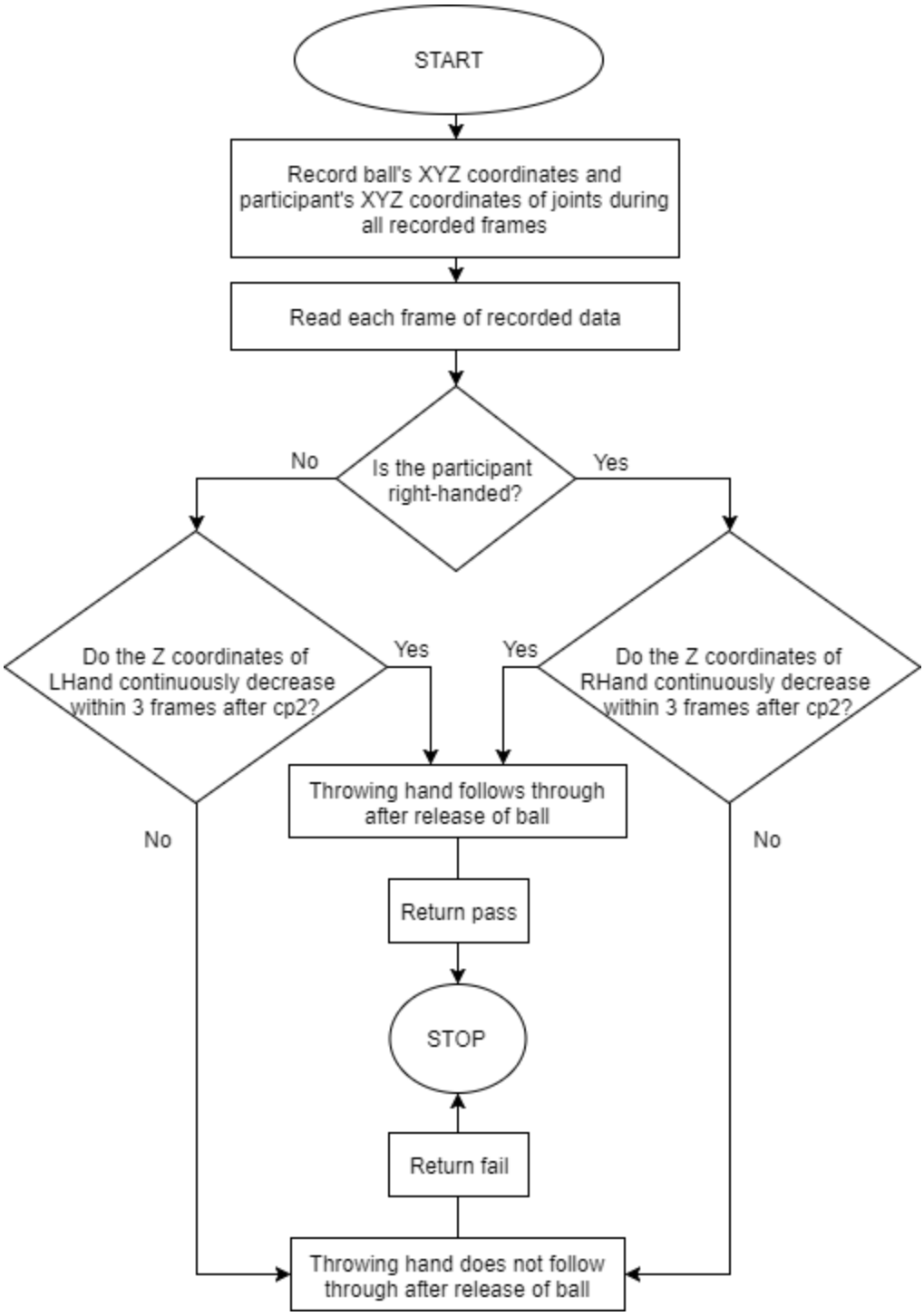

Figure 209. Algorithm to return feedback for the fourth criterion.

5 Methods and Results

The experimental phase consisted of using the developed system to test the skills of the TGMD-3. The assessment is based on customized algorithms for each subtest that verify if the 3D position of the most relevant joints for each subtest varies according to each performance criteria along time. The proposed system then returns if the participant passes or fails the respective subtest.

Experiments were conducted at Laurentian University Ben Avery Gym, in an indoor environment with a clear space. The Kinect sensor was positioned on a desk facing the participant for all tests except for the running test. The range of operation of the Kinect sensor lies between 0.4m to 4.5m, and the running test requires at least 18 m of clear space, according to the TDMG-3 scoring sheet. Because of that, the participant had to perform the running movement on a treadmill and the sensor was positioned facing the participant's back.

In order to evaluate the sensitivity and specificity of the algorithms to detect passes and fails, the author of this study performed all the test skills in two stages. First, each of the thirteen skills were performed ten times following the standardized movement to check if the proposed system would return only "pass" for all criteria, as expected. Then, the author performed each criterion incorrectly five times, for all the thirteen tests, to check if the software would detect each fail as expected. It is important to mention that these experiments were not intended to validate or test the reliability of the software; instead, the purpose was to investigate the feedback from the algorithms.

Table 2 shows the results given by the software when the author executed all the movements according to the standardized performance (please, refer to Table 1 for numbering of skills and

performance criteria). The results showed a very good response of the algorithms to assess gross motor skills. They also indicated that the algorithms and the strategy of dividing the movement into phases and critical points were very suitable for this application.

Table 3 shows the results given by the software when the author executed some criterion incorrectly on purpose, keeping the other ones executed as the standard, in order to check if the software would return a “fail” to the criterion executed incorrectly. The other criteria that were executed as the standard were omitted from the table in order to provide a better visualization of the results. For this second experiment, the software returned excellent results as it identified all the criteria executed incorrectly.

The reason why some movements did not acquire 100% of detection can be explained by the situations when the body joints were inferred or because of space limitation. The main example is the running test. Since the participant is running giving his/her back to the Kinect, the hand joints used to calculate the angle of the elbows are hidden for the majority of the movement, as the participant is supposed to have the arms flexed, what may cause incorrect calculations for the angles. The same situation applies to those tests that used the ball; sometimes during the test, the joints of the feet can get hidden by the ball, which may jeopardize the results.

Another important observation is the space limitation to perform the tests. Due to the Kinect’s field of view, the optimum range to perform the movement lies between 0 to 4.5 meters from the sensor; that way, movements that involve four cycles, e.g. four gallops, four skips, etc., forces the participant to perform the whole movement within that range. This situation makes the participant shorten the movement to fit the range, which may jeopardize the performance and, consequently, the feedback from the computerized application.

Table 2. Experiment 1 - Participant performed all the skills according to the standardized performance criteria.

Skill	Performance Criteria	Trial 1	2	3	4	5	6	7	8	9	10	% Results
1	1	1	1	1	0	1	1	0	1	1	1	75
	2	1	1	1	1	1	0	1	1	1	1	
	3	1	1	1	1	1	1	0	1	1	1	
	4	0	0	1	0	1	1	0	0	1	0	
2	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
3	1	1	1	1	1	1	1	1	1	1	1	97.5
	2	1	1	1	1	1	1	1	0	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
4	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
5	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
6	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
7	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
	5	1	1	1	1	1	1	1	1	1	1	
8	1	1	1	1	1	1	1	1	1	1	1	95
	2	0	1	1	1	1	1	0	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
9	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
11	1	1	1	1	1	1	1	1	1	1	1	95
	2	1	1	0	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	0	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
12	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	
13	1	1	1	1	1	1	1	1	1	1	1	100
	2	1	1	1	1	1	1	1	1	1	1	
	3	1	1	1	1	1	1	1	1	1	1	
	4	1	1	1	1	1	1	1	1	1	1	

Table 3. Experiment 2 - Participant performed each criterion of each skill incorrectly.

Skill	Performance Criterion Executed Incorrectly	Trials 1	2	3	4	5	% Results
1	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
2	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
3	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
4	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
5	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
6	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
7	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
	5	0	0	0	0	0	
8	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
9	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
10	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
11	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
12	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	
13	1	0	0	0	0	0	100
	2	0	0	0	0	0	
	3	0	0	0	0	0	
	4	0	0	0	0	0	

One of the purposes of the TGMD-3 is to identify if a child's gross motor skills development is significantly behind his/her normative group, according to the child's age. For this reason, it is important to also investigate the computer-based application sensitivity and specificity. The sensitivity, also know as "true positive rate", is the proportion of positive values that are actually identified as positives and the specificity, also known as "true negative rate", is the proportion of negative values that are actually identified as negative (Boyce, 2017). The investigation of the proposed system sensitivity and specificity is very important because the software should not misidentify a child that actually presents poor gross motor skills, and also should not identify a child whose gross motor skills development is considered normal for his/her age as a child who needs instruction and practice to improve his/her gross motor skills.

The sensitivity and specificity can be calculated using the following equations (Boyce, 2017):

$$Sensitivity = \frac{number\ of\ true\ positives}{number\ of\ true\ positives + number\ of\ false\ negatives} \tag{16}$$

$$Specificity = \frac{number\ of\ true\ negatives}{number\ of\ true\ negatives + number\ of\ false\ positives} \tag{17}$$

In the first experiment, the author performed the complete TGMD-3 ten times according to the standardized performance criteria. In this way, the performance of 500 criteria were assessed by the software. All 500 criteria were executed correctly, however instead of having 500 true positives values, the computer-based application returned 486 true positive and 14 false negative values. Therefore, the sensitivity of the system in this experiment is equal to 0.972.

In the second experiment, the author performed each criterion of the complete TGMD-3 incorrectly five times. In this way, the performance of 250 criteria were assessed by the software. All 250 criteria were executed incorrectly and the computer-based application returned 250 true negative values. Therefore, the specificity of the system in this experiment is equal to 1.

It is important to mention that these first two experiments were performed by the author with the main objective to investigate if the algorithms were detecting the passes and fails. Because the author is the developer of the computerized application, a bias is inevitably introduced in the analysis and there is subjectivity in the determination of the specificity and sensibility of the computer-based application. As mentioned above, these tests were conducted mainly to check the algorithms and a better analysis of sensitivity and specificity should be performed in the future with more independent volunteers.

To this end, another experiment was performed in order to verify how the software scoring performance is comparable to the classical test. For this experiment, a volunteer performed all the skills, with exception of the running test, while being assessed by the computerized application and by a graduate student conducting her studies in human kinetics and very familiar with the TGMD. More similar tests with more volunteers should be conducted in the future.

The results obtained by the software judgment and by the graduate student judgment are presented in Tables 4 to 15.

Table 4. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Horizontal Jump.

Horizontal Jump												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	1	1	1	0	1	1	1	0	1	1	1
Trial 2	1	1	1	1	1	1	1	1	1	1	1	1
Trial 3	1	1	1	1	1	1	1	1	1	1	1	1
Trial 4	1	1	1	1	1	1	1	1	1	1	1	1
Trial 5	1	1	1	1	1	1	1	1	1	1	1	1
Trial 6	1	1	1	1	1	1	1	1	1	1	1	1
Trial 7	1	1	1	1	1	1	1	1	1	1	1	1
Trial 8	1	1	1	1	1	1	1	1	1	1	1	1
Trial 9	1	1	1	1	1	1	1	1	1	1	1	1
Trial 10	1	1	1	1	0	1	1	1	0	1	1	1
	Percentage of success								Percentage of agreement			
	100 %				95 %				95 %			

Table 5. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Gallop.

Gallop												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	1	1	1	1	1	0	1	1	1	0	1
Trial 2	1	1	1	1	1	1	1	1	1	1	1	1
Trial 3	0	1	1	1	1	1	0	0	0	1	0	0
Trial 4	1	1	0	1	1	0	0	0	1	0	1	0
Trial 5	1	1	1	1	1	1	1	0	1	1	1	0
Trial 6	1	1	0	1	1	0	0	1	1	0	1	1
Trial 7	1	1	0	1	1	1	0	1	1	1	1	1
Trial 8	1	1	1	1	1	1	1	1	1	1	1	1
Trial 9	1	1	1	1	1	1	1	0	1	1	1	0
Trial 10	1	1	0	1	1	1	1	1	1	1	0	1
	Percentage of success								Percentage of agreement			
	87.5 %				72.5 %				75 %			

Table 6. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Hop.

Hop												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	0	1	1	0	0	1	1	0	1	1	1
Trial 2	0	0	1	1	0	0	1	1	1	1	1	1
Trial 3	0	0	1	1	0	0	1	1	1	1	1	1
Trial 4	0	0	1	1	0	0	1	1	1	1	1	1
Trial 5	1	0	1	1	0	0	1	1	0	1	1	1
Trial 6	1	0	1	1	1	0	1	1	1	1	1	1
Trial 7	1	0	1	0	0	0	1	1	0	1	1	0
Trial 8	0	0	1	1	0	0	1	1	1	1	1	1
Trial 9	1	0	1	0	0	0	1	1	0	1	1	0
Trial 10	1	0	1	1	0	0	1	1	0	1	1	1
	Percentage of success								Percentage of agreement			
	60 %				52.5 %				82.5 %			

Table 7. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Skip.

Skip									
	Graduate Student			Software			Agreement		
Criterion	1	2	3	1	2	3	1	2	3
Trial 1	1	1	1	1	1	1	1	1	1
Trial 2	1	1	1	1	1	1	1	1	1
Trial 3	1	1	1	1	1	1	1	1	1
Trial 4	1	1	1	0	0	0	0	0	0
Trial 5	1	1	1	1	1	1	1	1	1
Trial 6	1	1	0	1	1	1	1	1	0
Trial 7	1	1	1	0	0	0	0	0	0
Trial 8	1	1	1	1	1	1	1	1	1
Trial 9	1	1	1	0	0	0	0	0	0
Trial 10	1	1	1	1	1	1	1	1	1
	Percentage of success						Percentage of agreement		
	96.67 %			70 %			66.67 %		

Table 8. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Slide.

Slide												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	1	1	1	1	1	0	1	1	1	0	1
Trial 2	1	1	1	1	1	1	1	1	1	1	1	1
Trial 3	1	1	1	0	1	1	1	1	1	1	1	0
Trial 4	1	1	1	1	1	1	0	1	1	1	0	1
Trial 5	1	1	1	1	1	1	0	0	1	1	0	0
Trial 6	1	1	1	1	1	1	0	1	1	1	0	1
Trial 7	1	1	1	1	1	0	0	1	1	0	0	1
Trial 8	1	1	1	1	1	1	1	1	1	1	1	1
Trial 9	1	1	1	1	1	0	0	0	1	0	0	0
Trial 10	1	1	1	1	1	1	1	1	1	1	1	1
	Percentage of success								Percentage of agreement			
	97.5 %				75 %				72.5 %			

Table 9. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Two-hand Strike.

Two-hand strike															
	Graduate Student					Software					Agreement				
Criterion	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
Trial 1	1	1	1	1	1	1	1	1	0	1	1	1	1	0	1
Trial 2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Trial 3	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1
Trial 4	1	0	1	1	1	1	0	1	0	0	1	1	1	0	0
Trial 5	1	0	1	0	1	1	1	1	0	0	1	0	1	1	0
Trial 6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Trial 7	1	0	1	1	1	1	1	1	1	1	1	0	1	1	1
Trial 8	1	0	1	1	1	1	0	1	0	1	1	1	1	0	1
Trial 9	1	0	1	1	1	1	1	1	0	0	1	0	1	0	0
Trial 10	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1
	Percentage of success										Percentage of agreement				
	80 %					76 %					76 %				

Table 10. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for One-hand Strike.

One-hand Strike												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	0	1	1	1	0	0	1	1	1	0	1
Trial 2	1	0	1	0	1	1	1	1	1	0	1	0
Trial 3	1	0	1	0	1	0	1	1	1	1	1	0
Trial 4	1	0	1	1	1	0	1	1	1	1	1	1
Trial 5	1	0	1	0	1	0	1	1	1	1	1	0
Trial 6	1	0	1	0	1	1	1	0	1	0	1	1
Trial 7	1	0	1	1	1	0	1	1	1	1	1	1
Trial 8	1	0	0	0	1	0	1	0	1	1	0	1
Trial 9	1	0	1	0	1	1	1	0	1	0	1	1
Trial 10	1	0	1	0	1	0	1	1	1	1	1	0
	Percentage of success								Percentage of agreement			
	55 %				72.5 %				77.5 %			

Table 11. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Two-hand Catch.

Two-hand Catch									
	Graduate Student			Software			Agreement		
Criterion	1	2	3	1	2	3	1	2	3
Trial 1	1	1	1	1	1	1	1	1	1
Trial 2	1	1	1	1	1	1	1	1	1
Trial 3	1	1	1	1	1	1	1	1	1
Trial 4	1	1	0	1	1	1	1	1	0
Trial 5	1	1	1	1	1	1	1	1	1
Trial 6	1	1	1	1	1	1	1	1	1
Trial 7	1	1	1	1	1	1	1	1	1
Trial 8	1	1	1	1	1	1	1	1	1
Trial 9	1	1	1	1	1	1	1	1	1
Trial 10	1	1	1	1	1	1	1	1	1
	Percentage of success						Percentage of agreement		
	96.67 %			100 %			96.67 %		

Table 12. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for One-hand Dribble.

One-hand Dribble									
	Graduate Student			Software			Agreement		
Criterion	1	2	3	1	2	3	1	2	3
Trial 1	0	1	1	1	1	0	0	1	0
Trial 2	0	1	1	0	1	0	1	1	0
Trial 3	0	1	1	1	1	1	0	1	1
Trial 4	0	1	1	1	1	1	0	1	1
Trial 5	0	1	1	1	1	1	0	1	1
Trial 6	0	1	1	0	1	1	1	1	1
Trial 7	1	1	1	1	1	1	1	1	1
Trial 8	0	1	1	0	1	1	1	1	1
Trial 9	0	1	1	1	1	1	0	1	1
Trial 10	0	1	1	1	1	1	0	1	1
	Percentage of success						Percentage of agreement		
	70 %			83.33 %			73.33 %		

Table 13. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Kick Ball.

Kick Ball												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	1	1	1	1	1	1	1	1	1	1	1
Trial 2	1	1	1	1	1	1	1	1	1	1	1	1
Trial 3	1	1	1	1	1	1	1	1	1	1	1	1
Trial 4	1	1	1	1	1	1	1	1	1	1	1	1
Trial 5	1	1	1	1	1	1	1	1	1	1	1	1
Trial 6	1	1	1	1	1	1	1	1	1	1	1	1
Trial 7	1	1	1	1	1	1	1	1	1	1	1	1
Trial 8	1	0	1	1	1	1	1	1	1	0	1	1
Trial 9	1	1	1	1	1	1	1	1	1	1	1	1
Trial 10	1	1	1	1	1	1	1	1	1	1	1	1
	Percentage of success								Percentage of agreement			
	97.5 %				100 %				97.5 %			

Table 14. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Overhand Throw.

Overhand Throw												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	1	1	0	1	1	1	1	1	1	1	0
Trial 2	1	1	1	0	1	1	1	1	1	1	1	0
Trial 3	1	1	1	0	1	1	1	1	1	1	1	0
Trial 4	1	1	1	0	1	1	1	1	1	1	1	0
Trial 5	1	1	1	0	1	1	1	1	1	1	1	0
Trial 6	1	1	1	0	1	1	1	1	1	1	1	0
Trial 7	1	1	1	0	1	1	1	1	1	1	1	0
Trial 8	1	1	1	0	1	1	1	1	1	1	1	0
Trial 9	1	1	1	0	1	1	1	1	1	1	1	0
Trial 10	1	1	1	0	1	1	1	1	1	1	1	0
	Percentage of success								Percentage of agreement			
	75 %				100 %				75 %			

Table 15. Comparison between graduate student assessment, software assessment and agreement between these two methods of evaluation for Underhand Throw.

Underhand Throw												
	Graduate Student				Software				Agreement			
Criterion	1	2	3	4	1	2	3	4	1	2	3	4
Trial 1	1	1	1	1	1	1	1	1	1	1	1	1
Trial 2	0	1	1	1	1	1	1	1	0	1	1	1
Trial 3	1	1	1	1	1	1	1	1	1	1	1	1
Trial 4	1	1	1	1	1	1	1	1	1	1	1	1
Trial 5	1	1	1	1	1	1	1	1	1	1	1	1
Trial 6	1	1	1	1	1	1	1	1	1	1	1	1
Trial 7	1	1	1	1	1	1	1	1	1	1	1	1
Trial 8	1	1	1	1	1	1	1	1	1	1	1	1
Trial 9	1	1	1	1	1	1	1	1	1	1	1	1
Trial 10	1	1	1	1	1	1	1	1	1	1	1	1
	Percentage of success								Percentage of agreement			
	97.5 %				100 %				97.5 %			

The results from the comparison between the software assessment and the graduate student assessment showed levels of agreement that vary from 66.7% to 97.5%. The results indicate that skills that have their movement phases very distinct from each other, e.g. the horizontal jump test, showed higher levels of agreement since their phases can be easily assessed by the software or by human observation, with less risk for double interpretations or misjudgments. On the other hand, skills that do not have a clear distinction between their phases showed a lower level of agreement, which is the case of the skip test, for example.

Another factor that led to disagreements between the two types of assessment can be explained by the level of difficulty in performing the skills. The software uses thresholds and tolerances in order to return a feedback to a given skill and because of that, if those thresholds and tolerances are not well adjusted, it can cause the software judgment to be too generous or too biased. The same applies for the human assessment; sometimes the participant can perform one of the phases of the movement incorrectly but all the other phases correctly and while the software would return a "fail" because of that single mistake, the human judgment could return a "pass" considering all the others correctly performed phases. The judgement from the software uses a "true" or "false" idea, i.e. a Boolean logic, while the human judgment is a form of many-valued logic in which the truth values may be anything between 0 and 1, i.e. a fuzzy logic.

In addition, the software has some weakness with regards to the field of view. While the human observer is able to see the entire frame and distinguish precisely the participant's joints, the software sometimes misinterprets the scene due to the occurrence of inferred joints, which can cause the software to distort the actual performance of the participant, leading to a "fail" when it should be a "pass".

Human subjectivity is another reason that can cause disagreement between the human and computer assessments.

The analysis of sensitivity and specificity for this last experiment can be roughly calculated by taking the judgment of the graduate student as a reference for true positives and true negatives values. The total number of feedbacks returned by the software was equal to 460, where 343 were true positives, 35 were true negatives, 34 were false positives and 48 were false negative. Therefore, the sensitivity and specificity analysis for this experiment is 0.88 and 0.50, respectively.

6 Discussions and Future Work

This study describes the second version of a computerized tool to measure all the thirteen movements of the TGMD-3. This version amended some limitations and improved upon the pilot version. The pilot version implemented only four skills of the TGMD-2 and used the first version of the Kinect sensor, while this present version implemented all the thirteen skills of the TGMD-3, used the second version of the Kinect sensor, implemented image processing to detect objects based on their colors and integrated a contact sensor to the application.

The experiments used to test the second version of the computer-based application do not have the same empirical value as a control study. The first two experiments had the author of this study performing the skills correctly and incorrectly to measure the responsivity of the algorithms. This method of experiment may include a bias in the results, since the author is very familiarized with the TGMD-3 skills and with the algorithms. In addition, having only one external volunteer testing the software in the third experiment while being assessed by a graduate student that is also an expert in TGMD-3 may still include a bias in the results.

All the algorithms were developed based on the author performance of the skills, and because of that all the thresholds and tolerances used were determined according to the author's best judgment and body type features such as height and weight. It is still necessary to perform more tests using different volunteers in order to collect enough data to adjust the thresholds and tolerances to optimized values, which should be determined by a professional in the Human Kinetics field. The future versions of the test should be able to use the participant's information such as age, height and weight as an input for the test so that the algorithms evaluate the performance according to the participant and the test purpose. In addition, the thresholds and tolerances could also be used to

determine the level of difficulty of the test. Depending on the purpose of using the software, e.g. if used to evaluate athletes' performance, the tolerances and thresholds could be adjusted to more rigid values in order to increase the rigor of the assessment.

The limitations with regards to the Kinect's field of view revealed an important impact on the software assessment. Different from the classical tests, where the participant can move freely while performing the movement, when the computer-based application is used for the assessment, the participant must perform the movement within the field of view of the device that is capturing the body motion. Since Microsoft revealed in October of 2017 that the Kinect sensor was going to be discontinued, the assessment algorithms should be adapted to be used with another device that captures 3D body motion. It is important to select a device that provides a larger field of view and a better resolution than the current Kinect's cameras, so that the participants would have more freedom when performing the movements, without interfering with their actual performance. In case the new selected device still presents the same field of view of the Kinect sensor, the attachment of colored markers and/or IMUs (Inertial Measurement Units) to the most critical body joints should be considered, in order to overcome this limitation.

Once the new device to replace the Kinect sensor is identified and the study to optimize the threshold and tolerances used in the algorithms according to the user profile and purpose of testing is complete, it is crucial to develop a study to investigate the psychometric properties of the computer-based application to implement the TGMD-3. According to Webster & Ulrich (2017) "validity, or the degree to which a measurement adequately assesses constructs of interest, is a critical component of test criterion and revision" and "reliability, or the consistency of performance across multiple occasions, is imperative to ensure that an assessment is free from random error measurements". In other words, it is important to investigate the computer-based

application validity in order to determine how the score measured by this proposed software actually represents the best estimate of the real motor skill, and the reliability investigation will determine the consistency of the score obtained with this software over time.

The computerized system has some drawbacks compared to classical human testing in terms of versatility, such as the need for setting up the computer and the different sensors and making the proper adjustments for the field of view; and also, in terms of usability, such as the need to wear special sensors or markers. Even with these shortcomings, we believe its advantages in terms of repeatability, consistency and the possibility of recording the tests and registration of the data and results outweigh these shortcomings.

As previously mentioned in the Methods and Results chapter, it is also very important to evaluate the sensibility and specificity of the computer-based application. The calculation of these parameters in the last experiment, the one that compares the classical test results to the software, was merely to give a sense of the agreement that exists between the two methods. In the future, a true test administered according to standard protocols, with two observers should be performed in parallel to the computer-based application, in order to truly evaluate the software's ability to match the performance of the human administered test. More data collection from a variety of volunteers of different ages and physical structure should be used.

Moreover, a market impact survey should also be investigated in order to evaluate if the replacement of the classical test by the proposed system is actually worth it in terms of usability. One of the goals of this study is to create a computerized tool to implement the TGMD-3 that is practical to use and inexpensive. Depending on the device that is going to substitute the Kinect and if there is a need to increment the system with IMUs and markers to improve the accuracy and

usability of the system, the final product should still be simple to setup, offer the user comfort and not interfere with the performance of the skills.

It is important to mention that the software was developed based on adults' dimensions and because of that the tolerances and thresholds used may not be appropriate for children. Therefore, adaptations in the algorithms must be done in order to scale the assessment to children as well. This can be achieved by calculating the thresholds and tolerances based on the calibration values, so that the thresholds and tolerances values will remain proportional to the participant's physical dimensions.

Studies performed by Webster & Ulrich (2017) showed that the higher the score on the TGMD-3 test, the better the performance on the assessment, and as well, the raw scores on the test improved as the age of the participants increased. This means that a young child is not expected to master the skills of the TGMD-3; instead, his/her performance will be compared to the normative score for his/her correspondent age, as is done in the classical tests.

7 Conclusion

The present study had as an objective the development of a computerized assessment tool to implement the third version of the Test of Gross Motor Development (TGMD-3) using Microsoft Kinect sensor as the main component of the proposed system. The Kinect-based application was intended to be interactive and provide an accurate assessment of all the thirteen skills assessed by the TGMD-3, as described in Table 1. In addition, it had to collect the participants' information and provide them with instantaneous feedback after performing each skill.

A motivation for this study was in response to the limitations of the classical gross motor skills test, in particular with regards to the cost of having at least two trained coaches to apply the test, and also to the inconsistencies made as a result of human observation and human bias. Another motivation for this project was the advancement in human motion tracking combined with high-tech devices at an affordable price. In order to address the aforementioned motivations, a pilot study was initiated using the first version of the Kinect sensor and implemented only four skills of the TGMD-2. The present version of the computerized tool aimed to continue and improve the limitations of the pilot version.

This study was divided into four phases. The first phase consisted of developing an integrated system composed by the Kinect sensor and a customized computational program to acquire, treat and analyze the collected data. The second phase was intended to develop a contact sensor to assist the data acquisition for the criteria that the Kinect sensor couldn't track by itself. The third phase consisted of developing an algorithm for ball detection to track it during the ball-skills subtests. Finally, the last phase was dedicated to investigate the performance of the Kinect-based application for gross motor skills analysis, and it was divided into three experimental approaches. The first

one consisted of performing all the movements correctly in order to check if the proposed system would return only “passes” for each criterion. The second approach consisted of performing each criterion for each skill incorrectly in order to check if the proposed system would return only “fails” for those criteria. The last approach consisted of comparing the software scoring performance to the classical test scoring performance. Therefore, a volunteer performed all the skills, with exception of the running test, while being assessed by a graduate student and by the software.

The results revealed the computer-based application is excellent for detecting performance skills executed incorrectly, and it is also very good at detecting performance skills executed correctly. However, some shortcomings must be addressed. The running test results were shown to be poor when compared to the other tests. The reason for that is mainly the space limitations, which obligated the participant to perform the test on a treadmill with the sensor positioned facing the participant’s back. That procedure was shown to be less satisfactory for the test, since it intensified the occurrence of inferred joints, which made the results less accurate. The attempt of attaching circular markers on the shoes sole in order to try to eliminate the inferred joints occurrence on the feet was unsuccessful, considering that the size of the ratio of the markers would have to be bigger than the shoe area in order to be tracked by the Kinect’s camera and the algorithms for ball detection. Therefore, not much could be done in this study to improve the running test assessment.

Another drawback that must be addressed is the fact that it was not possible to attach markers on the bat in order to improve the assessment of the first criterion of the two-hand strike of a stationary ball test. The idea was attaching a marker on the bat in the space between the two hands in order to detect if the child grips the bat with the preferred hand above the non-preferred hand. In this case also the marker had to be larger than the bat size to be captured by the camera. Figure 210 illustrates an example of a marker attached to the bat. Note that a contour around the marker is

determined by the algorithm that identifies the objects based on their color. The 3D coordinates of the detected object refer to the point in the center of the white rectangle drawn in the black/white canvas on the right side of the picture. As the marker gets further away from the Kinect, the contour diminishes until it is barely visible. To overcome this problem the participant would have to perform the test very close to the Kinect or the size of the marker would have to be bigger than the one in the figure, and both options would not be possible for this test.

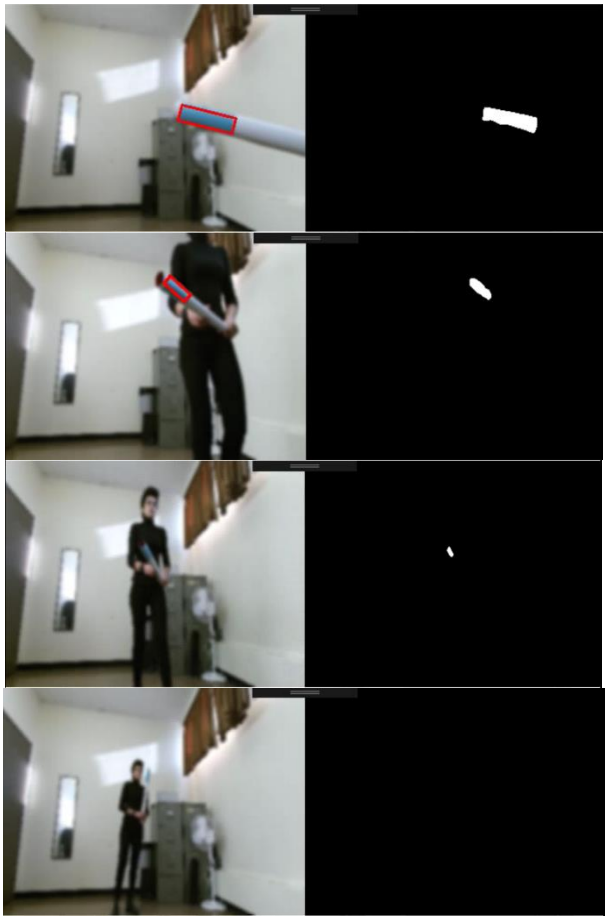

Figure 210. Example of marker attached to the bat.

Due to the same reason mentioned above, the participant used a volley ball for as two-hand strike and one-hand strike tests, even if the recommendation was to use a tennis ball to perform the tests.

A tennis ball was too small to be detected when the participant was more than 2 meters far from the Kinect.

Despite the above shortcomings, which can be easily overcome by selecting an infrared sensor and an RGB camera with a higher resolution and longer depth of field, the computer-based application was shown to be a promising tool as a substitute for the classical test, since it can be faster, more accurate and interactive. In this work, the Kinect sensor v2 was used to continue the development of a powerful application to implement the third version of the Test of Gross Motor Development (TGMD-3). Since Microsoft announced the discontinuation of the Kinect device, future studies should be performed to adapt the developed algorithms to be used with another body tracking device. Despite the fact that the algorithms and its thresholds and tolerances are still in a developmental phase, the results have already revealed that the use of the application for assessing gross motor skills is promising, although it is limited by the space requirements and by situations that lead to confusion in the depth sensor when determining the 3D position of the joints.

References

Adolph, K. E., 1997. Learning in the Development of Infant Locomotion. *Monographs of the Society for Research in Child Development,* 62(3).

Auvinet, E. et al., 2015. Detection of gait cycles in treadmill walking using a Kinect.. *Gait & posture,* 41(2), pp. 722-725.

Bailey, S. W. & Bodenheimer, B., 2012. A comparison of motion capture data recorded from a Vicon system and a Microsoft Kinect sensor.. *Proceedings of the ACM Symposium on Applied Perception,* pp. 121-121.

Beth, T. et al., 2003. Characteristics in human motion–from acquisition to analysis. *IEEE International Conference on Humanoid Robots,* pp. 56-75.

Bonifacci, P., 2004. Children with low motor ability have lower visual-motor integration ability but unaffected perceptual skills. *Human Movement Science,* 23(2), pp. 157-168.

Booth, M. L. et al., 1999. Mastery of fundamental motor skills among New South Wales school students: prevalence and sociodemographic distribution. *Journal of Science and Medicine in Sport,* 2(2), pp. 93-105.

Boreham, C. & Riddoch, C., 2001. The physical activity, fitness and health of children. *Journal of sports sciences,* 19(12), pp. 915-929.

Boyce, D., 2017. Evaluation of Medical Laboratory Tests. In: *Orthopaedic Physical Therapy Secrets.* s.l.:Elsevier, p. 125–134.

Burdea, G. C., 2003. Virtual rehabilitation-benefits and challenges. *Methods of Information in Medicine-Methodik der Information in der Medizin,* 42(5), pp. 519-523.

Burton, A. W. & Miller, D. E., 1998. *Movement skill assessment.* s.l.:Human Kinetics.

Chang, Y. J., Chen, S. F. & Huang, J. D., 2011. A Kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities. *Research in developmental disabilities,* 32(6), pp. 2566-2570.

Chipantasi, D. J. M., Velasco, N., Rivas, D. & Andaluz, V. H., 2017. Robust Color Tracking to Ambient Light Changes. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC),* pp. 83-87.

Clark, J. E. & Metcalfe, J. S., 2002. The mountain of motor development: A metaphor. *Motor development: Research and reviews,* pp. 163-190.

Clark, R. A. et al., 2012. Validity of the Microsoft Kinect for assessment of postural control.. *Gait & posture,* 36(3), pp. 372-377.

Connor-Kuntz, F. J. & Dummer, G. M., 1996. Teaching across the curriculum: Language-enriched physical education for preschool children. *Adapted Physical Activity Quarterly,* pp. 302-315.

Cools, W., De Martelaer, K., Samaey, C. & Andries, C., 2009. Movement skill assessment of typically developing preschool children: A review of seven movement skill assessment tools. *Journal of sports science and medicine,* 8(2), pp. 154-168.

Corti, A., Giancola, S., Mainetti, G. & Sala, R., 2016. A metrological characterization of the Kinect V2 time-of-flight camera. *Robotics and Autonomous Systems,* pp. 584-594.

Croce, R. V., Horvat, M. & McCarthy, E., 2001. Reliability and concurrent validity of the movement assessment battery for children. *Perceptual and motor skills,* 93(1), pp. 275-280.

Dutta, T., 2012. Evaluation of the Kinect™ sensor for 3-D kinematic measurement in the workplace.. *Applied ergonomics,* 43(4), pp. 645-649.

Eliason, C. F. & Jenkins, L. T., 1981. A practical guide to early childhood curriculum. *CV Mosby.*

EmguCV, 2008. *Main Page.* [Online] Available at: http://www.emgu.com/wiki/index.php/Main_Page [Accessed 05 January 2017].

Folio, M. R. & Fewell, R. R., 1983. Peabody developmental motor scales and activity cards.

Gallahue, D. & Ozmun, J., 1998. *Understanding motor development: Infants, children, adolescents, adults.*. 4th ed. Boston: McGraw-Hill Humanities, Social Sciences & World Languages.

Galna, B. et al., 2014. Accuracy of the Microsoft Kinect sensor for measuring movement in people with Parkinson's disease.. *Gait & posture,* 39(4), pp. 1062-1068.

González-Ortega, D., Díaz-Pernas, F. J., Martínez-Zarzuela, M. & Antón-Rodríguez, M., 2014. A Kinect-based system for cognitive rehabilitation exercises monitoring.. *Computer methods and programs in biomedicine.,* 113(2), pp. 620-631.

Goodway, J. D., 2003. Effects of motor skills instruction on Fundamental Motor Skill Development. *Adapted Physical Activity Quarterly,* Volume 20, pp. 298-314.

Guerra-Filho, G., 2005. Optical Motion Capture: Theory and Implementation. *RITA,* pp. 61-90.

Hamilton, M., Goodway, J. & Haubenstricker, J., 1999. Parent-assisted instruction in a motor skill program for at-risk preschool children. *Adapted Physical Activity Quarterly,* pp. 415-426.

Harrow, A. J., 1972. *A taxonomy of the psychomotor domain: A guide for developing behavioral objectives*.. s.l.:Addison-Wesley Longman Ltd.

Haubenstricker, J. & Seefeldt, V., 1986. Acquisition of motor skills during childhood. *Physical activity and well-being,* pp. 41-102.

Haywood, K., 1993. *Life Span Motor Development.* Champaign: Human Kinetics.

Henderson, S. E. & Barnett, A. L., 1992. *Movement assessment battery for children.* London: Psychological Corporation.

Holden, M. K., 2005. Virtual environments for motor rehabilitation: review. *Cyberpsychology & behavior,* 8(3), pp. 187-211.

Humphrey, J. H., 2003. *Child development through sports.* New York: Howorth Press.

Humphrey, J. H., 2004. *Childhood stress in contemporary society.* New York: Haworth Press.

Jana, A., 2012. *Kinect for windows SDK programming guide*.. Birmingham: Packt Publishing Ltd.

Johnson, T. M. et al., 2016. Does playing a sports active video game improve young children's ball skill competence?. *ournal of Science and Medicine in Sport,* 19(5), pp. 432-436.

Kambas, A. et al., 2012. The Motor-Proficiency-Test for children between 4 and 6 years of age (MOT 4–6): An investigation of its suitability in Greece. *Research in developmental disabilities,* 33(5), pp. 1626-1632.

Kelly, L. E., Dagger, J. & Walkley, J., 1989. The effects of an assessment-based physical education program on motor skill development in preschool children. *Education and Treatment of Children*, pp. 152-164.

Kelso, J. S., 1982. *Human motor behavior: An introduction.* Hillsdale: Lawrence Erlbaum Associates.

Lefmann, T. & Combs-Orme, T., 2013. Early brain development for social work practice: Integrating neuroscience with Piaget's Theory of Cognitive Development.. *Journal of Human Behavior in the Social Environment*, 23(5), pp. 640-647.

Microsoft, 2010. *More about Kinect sensor placement.* [Online] Available at: https://support.xbox.com/en-CA/xbox-360/accessories/sensor-placement [Accessed 23 November 2016].

Microsoft, 2012. *Skeletal Tracking.* [Online] Available at: https://msdn.microsoft.com/en-us/library/hh973074.aspx

Microsoft, 2014. *Coordinate mapping.* [Online] Available at: https://msdn.microsoft.com/en-us/library/dn785530.aspx [Accessed 23 November 2016].

Microsoft, 2014. *Kinect for Windows SDK 2.0.* [Online] Available at: https://www.microsoft.com/en-ca/download/details.aspx?id=44561 [Accessed 21 November 2016].

Microsoft, 2015. *Kinect hardware.* [Online] Available at: https://developer.microsoft.com/en-us/windows/kinect/hardware [Accessed 18 November 2016].

Microsoft, 2015. *Kinect hardware.* [Online] Available at: https://developer.microsoft.com/en-us/windows/kinect/hardware [Accessed 18 November 2016].

Microsoft, 2016. *Developing with Kinect for Windows.* [Online] Available at: https://developer.microsoft.com/en-us/windows/kinect/develop [Accessed 23 November 2016].

Microsoft, 2017. *JointTrackingState Enumeration.* [Online] Available at: https://msdn.microsoft.com/en-us/library/microsoft.kinect.jointtrackingstate.aspx?f=255&MSPPError=-2147217396

Mousavi Hondori, H. & Khademi, M., 2014. A review on technical and clinical impact of microsoft kinect on physical therapy and rehabilitation. *Journal of medical engineering.*

Mousavi Hondori, H. & Khademi, M., 2014. A review on technical and clinical impact of microsoft kinect on physical therapy and rehabilitation.. *Journal of medical engineering*, Volume 2014, p. 16.

Newell, K. M., 1991. Motor skill acquisition. *Annual review of psychology*, 42(1), pp. 213-237.

OpenCV, 2012. *About.* [Online] Available at: http://opencv.org/about.html [Accessed 06 January 2017].

Pavildes, C., Miyashita, E. & Asanuma, H., 1993. Projection From the Sensory to the Motor Cortex is Important in Learning Motor Skills in the Monkey. *Jornal of Neurophysiology,* 70(2), pp. 733-741.

Rintala, P. & Loovis, E. M., 2013. Measuring motor skills in Finnish children with intellectual disabilities. *Perceptual and motor skills,* 116(1), pp. 294-303.

Rouse, M., 2011. *What is natural user interface (NUI)?.* [Online] Available at: http://whatis.techtarget.com/definition/natural-user-interface-NUI [Accessed 18 November 2016].

Sarbolandi, H., Lefloch, D. & Kolb, A., 2015. Kinect range sensing: Structured-light versus time-of-flight kinect. *Computer Vision and Image Understanding,* pp. 1-20.

Schmidt, R. A. & Wrisberg, C. A., 2008. *Motor learning and performance: A situation-based learning approach.* s.l.:Human Kinetics.

Seefeldt, V., 1979. Developmental motor patterns: Implications for elementary school physical education. *Psychology of motor behavior and sport,* pp. 314-323.

Seguin, C., Geary, B., Livingstone, D. & Hainey, T., 2012. Can stereographic 3D displays improve interaction and feedback full body motion controlled application?.

Seroyer, S. T. et al., 2010. The kinetic chain in overhand pitching: its potential role for performance enhancement and injury prevention. *Sports Health,* 2(2), pp. 135-146.

South, M. & Palilla, J., 2013. Bruininks-Oseretsky Test of Motor Proficiency. *Encyclopedia of Autism Spectrum Disorders,* pp. 480-482.

Sveistrup, H., 2004. Motor rehabilitation using virtual reality. *Journal of neuroengineering and rehabilitation,* 1(1), p. 1.

Thakur, A., Sharma, S. & Rani, R., 2004. Assessment of Mental and Motor Development of Infants in Hamirpur District of Himachal Pradesh. *Anthropologist,* 6(2), p. 4.

Tremblay, L. et al., 2013. *Description and Features of a New Computerized Gross Motor Skills Measurement Test.* Madrid, Universidad de Alcalà.

Tremblay, L., Thorpe, M., Chebbi, B. & Theriault, V., 2016. Description and Features of a New Computerized Gross Motor Skills Measurement Tool. *Journal of Technologies and Human Usability,* 12(3-4), pp. 31-49.

Ulrich, D., 2013. The Test of Gross Motor Development - 3 (TGMD-3): Administration, scoring, and international norms. *Spor Bilimleri Dergisi,* pp. 27-33.

Ulrich, D. A., 2000. *Test of gross motor development-2.* Austin(Texas): s.n.

Van Waelvelde, H., Peersman, W., Lenoir, M. & Engelsman, S., 2007. Convergent validity between two motor tests: movement-ABC and PDMS-2. *Adapted Physical Activity Quarterly,* 24(1), p. 59.

Vles, J. S. H., Kroes, M. & Feron, F. J. M., 2004. MMT: Maastrichtse Motoriek Test.

Vries, L. et al., 2015. Evaluating fine motor coordination in children who are not ready for handwriting: which test should we take?. *Occupational therapy international,* pp. 61-70.

Wang, J. H. T., 2004. A study on gross motor skills of preschool children. *Journal of research in childhood education,* 19(1), pp. 32-43.

Wang, R. H., Mihailidis, A., Dutta, T. & Fernie, G. R., 2011. Usability testing of multimodal feedback interface and simulated collision-avoidance power wheelchair for long-term-care home residents with cognitive impairments. *Journal of Rehabilitation Research and Development,* 48(7), p. 801.

Wassenberg, R. et al., 2005. Relation between cognitive and motor performance in 5-to 6-year-old children: Results from a large-scale cross-sectional study. *Child development,* 76(5), pp. 1092-1103.

Webster, E. & Ulrich, D., 2017. Evaluation of the psychometric properties of the Test of Gross Motor Development—Third edition. *Journal of Motor Learning and Development,* 5(1), pp. 45-58.

Yang, L. et al., 2015. Evaluating and improving the depth accuracy of Kinect for Windows v2. *IEEE Sensors Journal,* 15(8), pp. 4275-4285.

Zhou, H. & Hu, H., 2008. Human motion tracking for rehabilitation—A survey. *Biomedical Signal Processing and Control,* 3(1), pp. 1-18.

Zimmer, R. & Volkamer, M., 1987. MOT 4-6: Motoriktest für vier-bis sechsjährige Kinder: Manual. *Beltz.*

Zittel, L. L. & McCUBBIN, J. A., 1996. Effect of an integrated physical education setting on motor performance of preschool children with developmental delays. *Adapted Physical Activity Quarterly*, 13(3), pp. 316-333.