Modeling and Development of Human Interface for Pedestrian Simulator

Romesh Khaddar
University of Nevada, Las Vegas, romesh.khaddar@gmail.com

Follow this and additional works at: https://digitalscholarship.unlv.edu/thesesdissertations

Part of the Computer Engineering Commons, Computer Sciences Commons, Electrical and Computer Engineering Commons, and the Kinesiology Commons

Repository Citation
https://digitalscholarship.unlv.edu/thesesdissertations/2187

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Theses, Dissertations, Professional Papers, and Capstones by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
MODELING AND DEVELOPMENT OF HUMAN INTERFACE FOR PEDESTRIAN SIMULATOR

by

Romesh Khaddar

Bachelor of Technology
Electronics and Communication Engineering
Indian Institute of Technology, Guwahati, India
2008

A thesis submitted in partial fulfillment of the requirements for the

Master of Science in Electrical Engineering – Electrical Engineering

Department of Electrical and Computer Engineering
Howard R. Hughes College of Engineering
The Graduate College

University of Nevada, Las Vegas
December 2013
We recommend the thesis prepared under our supervision by

Romesh Khaddar

entitled

Modeling and Development of Human Interface for Pedestrian Simulator

is approved in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering
Department of Electrical and Computer Engineering

Pushkin Kachroo, Ph.D., Committee Chair
Ke-Xun Sun, Ph.D., Committee Member
Yingtao Jiang, Ph.D., Committee Member
Zhonghai Ding, Ph.D., Graduate College Representative
Kathryn Hausbeck Korgan, Ph.D., Interim Dean of the Graduate College

December 2013
ABSTRACT

Modeling and Development of Human Interface for Pedestrian Simulator

by

Romesh Khaddar

(Dr. Pushkin Kachroo), Examination Committee Chair
Lincy Professor of Electrical Engineering
University of Nevada, Las Vegas

According to Traveler opinion and perception survey of 2005, 107.4 million Americans use walking as regular mode of travel, which amounts to 51% of American population. In 2009, 4092 pedestrian fatalities have been reported nationwide with a fatality rate of 1.33 which totals 59,000 crashes. Also, pedestrians are over represented in crash data by accounting more than 12% of fatalities but on 10.9% of trips. This makes a perfect case for understanding the causes behind such statistics, calling for a continuous research on pedestrians walking behavior and their interactions with surroundings.

Current research in pedestrian simulation focuses on surveys and mathematical simulation models such as macroscopic and microscopic dynamic models involves autonomous entities. The surveys represent the perception of individual while mathematical simulation severely limits the capacity to capture effect of human factors in the understanding of pedestrian interactions. Complicated psychological models
are used to a certain extent for understanding of such problems but are incapable to estimate the diversity of human behavior. To capture tendencies of people, they need to be a part of research, under a safe and controlled environment.

In this thesis, an attempt has been made to develop a module which can be used to track human walk gesture and map it to actual human walk. Then, this module could be implemented in a system aimed to understand pedestrian behavior. Following are the accomplishments of this thesis.

- Built an API to use with software interface to capture human motion
  - Explored arduino based wearable interface to capture human motion.
  - Explored Kinect based video interface to capture human motion.
  - Defined gestures and identified configurations for least difficult setup and calibration process.
  - Wrote the software interface for a Kinect based system (video interface).

- Built a mathematical framework for abstracted dynamical system, for the purpose of pedestrian interface in simulation engine.
  - Obtained mathematical model for human walk.
  - Obtained conversion to non-holonomical system for human walk.
  - Programmed the mathematical model into the API.
Eventually this is expected to contribute towards state-of-the-art researches which aim at understanding pedestrian dynamics in transportation safety and planning. The module described is expected to work real-time as a separate entity.
ACKNOWLEDGEMENTS

Every project depends on the right guidance and enough motivation to ensure quality and completion. I would like to express my sincere thanks and gratitude towards my advisor Prof. Pushkin Kachroo for inspiring, motivating, guiding and helping me understand the math behind the universe, throughout my masters degree and the thesis. I cannot say enough words to appreciate the mentorship provided by him during the course of my master’s degree.

I am also grateful to my thesis committee Prof. Hongtao Jiang, Prof. Ke-Xun Sun and Prof. Zhonghai Ding for their constructive comments and warm encouragement.

I would like to take this opportunity to show my appreciation towards the love of my parents and younger brother who supported my endeavour in sun and rain.

I am also grateful to my friends Anuj, Atul, Himanshu, Naveen, Pankaj, Pratik, Shaurya and Sourabh for all their support and help during the period of my degree.
# TABLE OF CONTENTS

ABSTRACT iii

ACKNOWLEDGEMENTS vi

LIST OF TABLES x

LIST OF FIGURES xi

1 INTRODUCTION & LITERATURE SURVEY 1

2 ABSTRACTION 6

2.1 Abstraction of Systems . . . . . . . . . . . . . . . . . . . . . . . . . . 6
2.2 Mathematical Preliminaries . . . . . . . . . . . . . . . . . . . . . . . 8
2.3 Abstracting Maps . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
2.4 Abstraction of Dynamical Systems . . . . . . . . . . . . . . . . . . . 11
2.5 Abstractions of Control Systems . . . . . . . . . . . . . . . . . . . . . 12

3 THE HUMAN WALK 16

3.1 Bio-Mechanics of Human Movement . . . . . . . . . . . . . . . . . . 16
3.1.1 Movement Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . 17
3.2 Gait . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
3.3 Kinematics Analysis of Gait . . . . . . . . . . . . . . . . . . . . . . . 21
3.4 Simulating an Actual Human Walk . . . . . . . . . . . . . . . . . . . 22
3.5 Abstraction of Human Walk . . . . . . . . . . . . . . . . . . . . . . . 23
3.5.1 Definition of on the spot walk . . . . . . . . . . . . . . . . . . . 24
3.5.2 Analysis of on the spot walk . . . . . . . . . . . . . . . . . . . . 26
3.6 Model of Virtual Entity in 3D World . . . . . . . . . . . . . . . . . . 26
3.7 Abstracting Map . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
3.8 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

4 CAPTURING HUMAN WALK 31

4.1 Simulated Human Walk . . . . . . . . . . . . . . . . . . . . . . . . . . 31
4.2 Mapping actual to simulated walk . . . . . . . . . . . . . . . . . . . . 32
4.3 Extracted Parameters . . . . . . . . . . . . . . . . . . . . . . . . . . . 32
4.4 Construction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
4.4.1 Architecture . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
4.4.2 Hardware Construction . . . . . . . . . . . . . . . . . . . . . . . 37
LIST OF TABLES

4.1 Gesture Parameter mapping ........................................ 32
5.1 Comparision Table of Available Products ........................ 46
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Relationship between pedestrian and traffic environment</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Relation between Systems and Abstractions</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>Mapping Between Spaces</td>
<td>11</td>
</tr>
<tr>
<td>3.1</td>
<td>A Single Step shown for the forward progression of body</td>
<td>17</td>
</tr>
<tr>
<td>3.2</td>
<td>Free Body diagram of a foot</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>Link Segment Model</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>On The Spot Walk</td>
<td>25</td>
</tr>
<tr>
<td>3.5</td>
<td>Mapping of walking phases</td>
<td>27</td>
</tr>
<tr>
<td>3.6</td>
<td>The rolling disc</td>
<td>29</td>
</tr>
<tr>
<td>4.1</td>
<td>Flowchart of complete system</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>Information Flow</td>
<td>35</td>
</tr>
<tr>
<td>4.3</td>
<td>Flowchart of hardware system</td>
<td>36</td>
</tr>
<tr>
<td>4.4</td>
<td>Flowchart of software system</td>
<td>42</td>
</tr>
<tr>
<td>4.5</td>
<td>Placement of sensors</td>
<td>43</td>
</tr>
<tr>
<td>4.6</td>
<td>Calibration Position</td>
<td>43</td>
</tr>
<tr>
<td>5.1</td>
<td>Out of the box</td>
<td>47</td>
</tr>
<tr>
<td>5.2</td>
<td>Power Cord</td>
<td>48</td>
</tr>
<tr>
<td>5.3</td>
<td>Mounting the Kinect Sensor</td>
<td>48</td>
</tr>
<tr>
<td>5.4</td>
<td>Kinect Interaction Space [14]</td>
<td>49</td>
</tr>
<tr>
<td>5.5</td>
<td>Kinect Architecture [15]</td>
<td>50</td>
</tr>
<tr>
<td>5.6</td>
<td>Kinect SDK Components Architecture [15]</td>
<td>51</td>
</tr>
<tr>
<td>5.7</td>
<td>Pedestrian Data Extraction Architecture</td>
<td>54</td>
</tr>
<tr>
<td>6.1</td>
<td>Perpendicular kinect sensors scenario 1</td>
<td>60</td>
</tr>
<tr>
<td>6.2</td>
<td>Perpendicular kinect sensors scenario 2</td>
<td>61</td>
</tr>
<tr>
<td>6.3</td>
<td>Geometry of relating body constants</td>
<td>62</td>
</tr>
<tr>
<td>6.4</td>
<td>Scenario for hand gesture</td>
<td>64</td>
</tr>
<tr>
<td>7.1</td>
<td>Complete Architecture of Pedestrian Simulator API</td>
<td>66</td>
</tr>
<tr>
<td>7.2</td>
<td>User in front of Kinect Sensor</td>
<td>69</td>
</tr>
<tr>
<td>7.3</td>
<td>User in changing heading</td>
<td>69</td>
</tr>
<tr>
<td>7.4</td>
<td>User stepping and increasing distance travelled</td>
<td>70</td>
</tr>
</tbody>
</table>
Pedestrian safety is a primary concern in traffic situations since they are in the most vulnerable position. According to a report by NHTSA, 4092 pedestrians were killed and an estimated 59000 were injured in traffic crashes in United States in 2009. The numbers are very high as they account for 12% of fatalities in crash data [18]. Each pedestrian injury or fatality has many facets to it, in terms of cost to those affected directly and indirectly.

Traffic congestion is a very important aspect of travel planning inside city networks. The costs attributed to any congestion are at multiple levels and can be broadly classified in direct and indirect costs. Direct costs include delays and fuel consumption, while the indirect costs include inability to calculate precise travel time and pollution. But the buck doesn’t stop here and creates tertiary effects like road rage and slow emergency vehicle response [17].

Traditionally, larger part of travel planning is done with aim to minimize travel time with preference to vehicle traffic over pedestrian traffic [10]. Some studies have attempted to plan for reducing traffic congestion by optimizing travel time costs to both pedestrians and vehicles. These studies are more relevant to traffic in heavily congested areas.
travelled areas, but point towards a more subtle area of pedestrian safety. After the completion of planning, study on pedestrian safety is required to understand the effect of the new improvements. Such studies are incomplete in a broader sense of safety unless actual human subjects experience such systems. Some methods used for study of pedestrian safety are surveys and observations on an actual implemented system, over the course of time.

Pedestrian safety is also attributed to drivers of vehicles travelling through the traffic system. In this way pedestrian safety is a bi-party relationship where it is a responsibility of both driver and pedestrian. If any of the two does not understand it or fail to respect others right, eventually affect both. The study of such issues is under the broad topic of Human Factors Research.

Studies show that the demographics of an area affect the transportation behavior there. Transportation behavior here covers the interaction between transportation system and people ranging from mode choice to trip frequency and distance as well as the ways citizens affect the transportation policy. Demographics is a broad term and has many variables. With respect to the transportation behavior, certain variables have been found in high correlation such as age distribution, race and ethnicity, education level etc [8]. Statistically, a set of individuals can be identified as the representative of demographics for a region to conduct human factors research. Such research is an important topic in the field of transportation since it assesses effects on transportation systems subject to variations in user behavior due to their personal traits.
Though, individual behavior cannot be truly studied or analyzed for the entire population, due to the complexity involved between transportation behavior and demographics. Statistical methods provide the capability to represent demographical information with a smaller set of individuals, thereby creating a significant representation of the population inhabiting in that region. Such methods are based on surveys in a safe and controlled environment of a lab and have widely been accepted for study of human factors research.

As mentioned above, till now pedestrian safety has been mostly studied on established transportation systems by surveys and observations at the locations/site. Such methods though address in understanding many real life problems, have certain implied assumptions and therefore lack on a few grounds. For example, since such surveys and observations are taken on an existing system and the results are used for suggesting modifications to it, implies the system might be running in a potentially unsafe condition.

The standard in conducting pedestrian Level of Service (LOS) analysis is laid out by Highway Capacity Manual (HCM) in USA. Although, a standardized set of practices is defined for data collection and quantifying congestion in pedestrian facilities, many studies identify amendments and new methods for HCM to analyze LOS, respectively [4].
According to HCM, LOS for pedestrian part of a transportation system can be improved upon three primary areas viz. pedestrian characteristics, sidewalk environment and flow characteristics; relationship between these categories have emerged in the literature for pedestrian studies and can be illustrated as in figure 1.1 [4].

Figure 1.1: Relationship between pedestrian and traffic environment

Pedestrian characteristics identified can be broadly classified as personal characteristics, trip purposes & expectations and behavior. Personal characteristics relate variables like pedestrian speed and sidewalk widths with age, gender, group size and other demographic factors [5], [12], [27]. Trip purpose and expectations with pedestrian perceptions like safety, comfort and convenience have also been found to affect their behavior, though have not been addressed by HCM. Researchers have confirmed that pedestrians perception of environment affect their behavior significantly
In general, have a tendency to put a cost to each sidewalk facility for a destination on their personal expectations [9]. Similarly, individual behaviors like use of music players and mobile handsets during walk, has been criticized by various writers [2] but researchers merely have anecdotal evidences for the same and wish to understand it more.

In this thesis, we propose a methodology alongwith hardware and software development to quantify and study the effect of individual differences on pedestrian transportation system. A parallel to this methodology has existed in principle as human centered simulation studies for driving and has been proven helpful. A result expected from this work is to develop a module using which a platform can be constructed for conducting studies on pedestrian related transportation systems.

The thesis is structured by starting with introduction to theory of abstraction which forms an important pillar of the complete work, this is followed by explanation of how the human walk is studied and what are the important parameters involved. This is followed by details on hardware implementation of a system for capturing human walk and associated parameters as a proof of concept. Later a Kinect based solution is studied with software and hardware capabilities and limitations for prototype implementation. In the end we discuss about the problems, limitations and future applications.
CHAPTER 2

ABSTRACTION

Capturing massive data and computing it to obtain values for necessary model which can simulate a complex system is tedious, resource intensive, and complicated. For reducing the complexity of analysis for such systems, simplified models which can capture the behaviour of interest in the original system can be obtained. Such models, called abstractions, are easier to analyze as compared to the complex model. Therefore, a mathematical framework is required to filter unnecessary data and utilize the necessary information as per requirement. This chapter studies an important mathematical ideology and framework which would form the basis behind the choice of modeling in subsequent chapters.

2.1 Abstraction of Systems

The webster’s dictionary define abstraction as "the act or process of separating the inherent qualities or properties of something from the actual physical object or concept to which they belong". In system theory, the objects are usually dynamical or control systems, the properties are usually the behaviors of certain variables of interest and the act of separation is essentially the act of capturing all interesting behaviors. Thus, the Webster’s definition can be modified to define the abstraction
of a system as another system which captures all system behavior of interest [20].
This set of behaviors are captured by an abstracting map $\alpha$ and are dependent on what information is of interest (figure 2.1).

Therefore, the classic model reduction techniques can be explained as approximate abstractions under this framework. It is not necessary that the abstracted and the original systems are similar from a modeling perspective. An example, can be that the original model be an ordinary differential equation but the abstracted system may be a discretized system. Therefore, under this definition, the problem can be rephrased as the following: Given an original system and an abstracting map, find an abstracted system which generates the abstracted behaviors either exactly or approximately.

Figure 2.1: Relation between Systems and Abstractions
2.2 Mathematical Preliminaries

Differential geometry takes an important part in understanding the following. Multiple texts can be used as reference for better understanding [23], [1].

**Tangent Space:** Let there be a differentiable manifold $M$. The set of all tangent vectors at $p \in M$ is called the tangent space of $M$ at $p$ and is denoted by $T_p M$.

**Tangent Bundle:** The collection of all tangent spaces of the manifold $M$ is called a tangent bundle. Mathematically it can be represented as

$$TM = \bigcup_{p \in M} T_p M$$  \hspace{1cm} (2.1)

**Projection Map:** The projection map is defined from tangent bundle to manifold as $\pi : TM \to M$ taking a tangent vector $X_p \in T_p M \subset TM$ to the point $p \in M$. The tangent space $T_p M$ can then be thought of as $\pi^{-1}(p)$. The tangent space can be considered as special case of an object called fibre bundle.

**Fiber Bundles**[19]: A fiber bundle is a five-tuple $(B, M, \pi, U, \{O_i\}_{i \in I})$ where $B, M, U$ are smooth manifolds called total space, the base space and the standard fiber respectively. The map $\pi : B \to M$ is a surjective submersion and $\{O_i\}_{i \in I}$ is an open cover of $M$ such that for every $i \in I$ there exists a diffeomorphism $\phi_i : \pi^{-1}(O_i) \to O_i \times U$ satisfying

$$\pi_o \circ \phi = \pi$$  \hspace{1cm} (2.2)

where $\pi_o$ is the projection from $O_i \times U$ to $O_i$. The submanifold $\pi^{-1}(p)$ is called the fiber at $p \in M$. If all the fibers are vector spaces of constant dimension, then the fiber...
bundle is called a vector bundle.

Let $M$ and $N$ be smooth manifolds and $f : M \to N$ be a smooth map. Let $p \in M$ and let $q = f(p) \in N$. We push forward tangent vectors from $T_pM$ to $T_qN$ using the induced push forward map $f_* : T_pM \to T_qN$. If $f : M \to N$ and $g : N \to M$ then

$$(g \circ f)_* = g_* \circ f_*$$ (2.3)

A vector field or dynamical system on a manifold $M$ is a continuous map $F$ which places at each point $p$ of $M$ a tangent vector from $T_pM$. Let $I \subseteq \mathbb{R}$ be an open interval containing the origin. An integral curve of a vector field is a curve $c : I \to M$ whose tangent at each point is identically equal to the vector field at that point. Therefore an integral curve satisfies for all $t \in I$,

$$c' = c_*(I) = X(c)$$ (2.4)

$f$-related Vector Fields: Let $X$ and $Y$ be vector fields on manifolds $M$ and $N$ respectively and $f : M \to N$ be a smooth map. Then $X$ and $Y$ are $f$-related iff $f_* \circ X = Y \circ f$.

2.3 Abstracting Maps

For system analysis, reduction in complexity is associated with a avoidance of unnecessary information, thereby working with a simplified model with reduced com-
plexity. So, if $M$ is the state space of a system, the state $p \in M$ is thus mapped to an abstracted state $q \in N$. It can be safely said that the complexity reduction requires that the dimension of $N$ should be lower than the dimension of $M$.

The relevant information for mapping $M$ is dependent on the required properties from the system under consideration ($M$). The desired specification can be quite different even in the same system as functionality may be different in various modes of its operation. Hence we cannot obtain a system abstraction without prior knowledge of the system functionality. For example, a person can move forward on two limbs or all four limbs or even a single limb. Therefore, depending on mode of operation functionality changes and hence the system specification will change.

This functionality of the system helps in identifying the states of interest for information extraction. Once the functionality of the system is identified, a set of equivalent states can be defined in the form of an equivalence relation on the state space. Thus, the quotient space $M/\sim$, determined by the chosen equivalence relation, is the state space of the abstracted system.

For this quotient space to have a manifold structure, the equivalence relation must be regular [1]. The surjective map $\alpha : M \rightarrow M$ which sends each state $p \in M/\sim$ to its equivalence class $[p] \in M/\sim$ is called the quotient map and is the mapping from each state to its abstracted state. Therefore it can be defined as following [20]

**Abstracting Maps:** Let $M$ and $N$ be given manifolds with $\text{dim}(N) \leq \text{dim}(M)$. A surjective map $\alpha : M \rightarrow N$ from the state space $M$ to the abstracted space $N$ is called an abstracting map.
2.4 Abstraction of Dynamical Systems

The interesting portion after determining the abstracting map \( \alpha \) is obtaining the evolution of dynamics obtained from the state evolution on \( M \) governed by a vector field \( X \) on \( M \). This is characterized by integral curves and is defined as following.

**Definition:** Let \( X \) and \( Y \) be vector fields on \( M \) and \( N \) respectively and let \( \alpha : M \to N \) be a smooth abstracting map. Then vector field \( Y \) is an abstraction of vector field \( X \) with respect to \( \alpha \) if and only if for every integral curve \( c \) of \( X \), \( \alpha \circ c \) is an integral curve of \( Y \) (figure 2.4).

\[ c' = c_* (I) = X(c) \]

implies

\[ c' = c_* (I) = X(c) \]
Moreover, it also implies from the definition that two different abstracting maps \( \alpha_1 \) and \( \alpha_2 \) over some vector field \( X \) do not create same abstracted vector field \( Y \).

When modeling large scale systems, a hierarchical approach may be chosen, thereby modeling at many levels of abstraction. Therefore, the following proposition:

**Transitivity of Abstractions**: Let \( X_1, X_2, X_3 \) be vector fields on manifolds \( M_1, M_2 \) and \( M_3 \) respectively. If \( X_2 \) is an abstraction of \( X_1 \) with respect to the abstracting map \( \alpha_1 : M_1 \to M_2 \) and \( X_3 \) is an abstraction of \( X_2 \) with respect to abstracting map \( \alpha_2 : M_2 \to M_3 \) then \( X_3 \) is an abstraction of \( X_1 \) with respect to abstracting map \( \alpha_2 \circ \alpha_1 \).

**Theorem.** Vector field \( Y \) on \( N \) is an abstraction of vector field \( X \) on \( M \) with respect to the map \( \alpha \) if and only if \( X \) and \( Y \) are \( \alpha \)-related.

The above theorem is equivalent to the definition of abstraction of dynamical systems. It is important because rather than explicit computation of integral curves, it allows to check condition on vector fields. Also, the \( \alpha \)-relatedness of two vector fields is a very restrictive condition as it limits cases where one dynamical system is an exact abstraction of another.

### 2.5 Abstractions of Control Systems

In this section, the theory of abstraction for dynamical systems is extended to control systems.

**Control System** [6]: A control system \( S = (B, F) \) consists of a fiber bundle \( \pi : 
B → M called the control bundle and a smooth map F : B → TM which is fiber preserving and hence satisfies

$$\pi' \circ F = \pi$$

where $\pi' : TM \to M$ is the tangent bundle projection.

Essentially, the base manifold $M$ of the control bundle is the state space and the fibers $\pi^{-1}(p)$ are the state dependent control spaces. In a local coordinate chart $(V, x)$, the map $F$ can be expressed as $\dot{x} = F(x, u)$ with $ueU(x) = \pi^{-1}(x)$.

**Integral Curves of Control Systems [20]:** A curve $c : I \to M$ is called an integral curve of the control system $S = (B, F)$ if there exists a curve $c^B : I \to B$ satisfying

$$\pi \circ c^B = c$$

$$c' = c_\ast(I) = F(c^B)$$

**Abstractions of Control Systems [20]:** Let $S_M = (B_M, F_M)$ with $\pi_M : B_M \to M$ and $S_N = (B_N, F_N)$ with $\pi_N : B_N \to N$ be two control systems. Let $\alpha : M \to N$ be an abstracting map. Then control system $S_N$ is an abstraction of $S_M$ with respect to abstracting map $\alpha$ iff for every integral curve $c_M$ of $S_M$, $\alpha \circ c_M$ is an integral curve of $S_N$. 
In the above definition of abstractions of control systems, it is clear that two abstracting maps for same abstraction may or may not be different. However, it is difficult to decide whether one control system is the abstraction map of other by directly using the definition since it would require integration of the system. Therefore, a set of conditions for such identification were derived as following theorem. It is analogous to the theorem about abstraction of the dynamical system detailed above.

Conditions for Control System Abstractions [20]: Let $S_N = (B_N, F_N)$ and $S_M = (B_M, F_M)$ be two control systems and $\alpha : M \to N$ be an abstracting map. Then $S_N$ is an abstraction of $S_M$ with respect to abstracting map $\alpha$ iff:

$$\alpha \circ F_M \circ \pi_M^{-1}(p) \subseteq F_N \circ \pi_N^{-1} \circ \alpha(p)$$

at every $p \in M$.

But this theorem does not require the commutativity. This allows the theorem to be applied in every control and dynamical system and concludes that they can be abstracted by another control system. This can be seen in following corollaries:

- **Abstractable Control Systems**: Every control system $S_M = (B_M, F_M)$ is abstractable by a control system $S_N$ with respect to any abstracting map $\alpha : M \to N$.

- **Abstractable Dynamical System**: Every dynamical system on $M$ is abstractable by a control system with respect to any abstracting map $\alpha : M \to N$. 14
Once a system abstraction has been obtained, it is useful to propagate properties of interest from the original system to the abstracted system. For control systems, one of those properties is controllability.

**Controllability**: Let $S = (B, F)$ be a control system. Then $S$ is called controllable iff given any two points $P_1, P_2 \in M$, there exists an integral curve $c$ such that for some $t_1, t_2 \in I$ we have $c(t_1) = p_1$ and $c(t_2) = p_2$.

**Controllable Abstractions**: Let control system $S_N = (B_N, F_v)$ be an abstraction of $S_M = (B_M, F_M)$ with respect to some abstracting map $\alpha$. If $S_M$ is controllable then $S_N$ is controllable.

Other properties, such as local accessibility, also propagate. Stability, however, does not propagate since the abstracted system allows redundant evolutions which could be unstable.
CHAPTER 3

THE HUMAN WALK

In this chapter we analyze the primary objective of the module under consideration for development, i.e., The Human Walk. This chapter explains an actual human walk and associated kinematics, discusses the challenges of simulating a human walk, then defines a simulated human walk, analyses it and eventually provides a method of mapping to actual human walk from a simulated motion.

3.1 Bio-Mechanics of Human Movement

In this section we assume that the reader is familiar with basic concepts of force, energy, momentum and laws of motion and methods for their analysis. We start here from analyzing the pattern of human motion in terms of angular rotation of joints and translation.

In general, a typical walk involves rotation of limbs to achieve a motion through space. This motion is usually curvilinear in nature due to fixed length of rotating arm at every joint. This rotation reduces effective height of that particular joint from the ground. The following figure[3.1] shows the same for a human walk [25]. It shows a single step in a sequence of ankle(b), hip (a’) and ankle (b”) rotation providing for the forward progression of the body.
3.1.1 Movement Analysis

Majority of human movements are quite complex due to movement of body parts relative to each other and environment. A quantitative analysis of movement can clarify the muscles required to be active during that period in context of forces acting on the body. For this purpose, Inverse Dynamics is an often used technique, and two types of models are used for studies, namely Free Body Diagram and Link-segment...
Model. They view the entire system for locomotion as a whole and in its parts, respectively. Eventually this calculation translates into joint forces and torques.

![Free Body diagram of a foot](image)

**Figure 3.2: Free Body diagram of a foot**

Such analysis requires three types of information:

- Anthropometric information on the segments (mass, length etc).

- Kinematic information like linear and angular moment information for each segment.

- External forces acting on body.

With the following assumptions for link segment models:
Figure 3.3: Link Segment Model

- Each segment has a fixed mass located at its center of mass.
- The joints are considered as hinge joints.
- The moment of inertia is fixed during movement.
- The length of each segment is constant.

3.2 Gait

Gait is defined as pattern of movement of limbs for locomotion for animals including humans. Every mammal follows a set of different gaits corresponding to different type of motion (walking, jogging, running etc.). However in this text we are discussing
the human walk and henceforth, gait will be used in reference to human gait only.

Characteristics of a gait are defined by differences in point of contact with surface, potential and kinetic energy cycles, overall velocity and forces experienced. Of these, the most popular and referenced is the identification of various parts of a gait by point of contact of foot with the surface.

Generally, a gait is classified as normal or pathological. The pathological gait is different from normal gait because of certain factors about the concerned subject. The deviation from normal gait can be permanent or transient and is a field of research for bio-mechanics aimed towards individuals requiring help with their walking ability. Many factors affecting a gait are identified in the literature and hence can be classified as follows [3]:

1. Extrinsic
2. Intrinsic
3. Physical
4. Psychological
5. Physiological
6. Pathological

On basis of point of contact of foot, a complete gait consists of total analysis of all the body movements, the associated mechanics and related muscle activity, for
overall pattern of limbs between re-contact of initial reference point of the foot, with the surface. To study a gait, following parameters are taken into account [24]:

1. Step length
2. Stride length
3. Cadence
4. Speed
5. Dynamic base
6. Progression line
7. Foot angle

Essentially a gait is a repetitive motion of the limbs, each sequence of this cyclic movement of limbs is called a gait cycle and is an essential derived parameter of any gait. It is an important parameter because various phases of a typical gait are defined in terms of percentage of gait cycle completed since it provides a common reference method. This will be discussed in detail during kinematic analysis of the gait.

3.3 Kinematics Analysis of Gait

Kinematics is the branch of classical mechanics describing motion of a point or a collection of points or a rigid body or group of rigid bodies and does not consider force. It is the method for quantifying and measuring the kinematic quantities for
analysis of gait. Kinematic quantities for any point or rigid body are described as position, velocity and acceleration of concerned points on it. Thus, it describes the motion of human body by studying the trajectories of various important points and lines identified by careful observation.

The requirement from the kinematic analysis of gait is defined by the objective of the problem. In the considered problem, we need the information about movement of a subject in space and time. Determination of clinical aspects of gait are not in scope of the problem. Therefore, the requirement from kinematic analysis is in terms of following parameters:

- Speed of subject
- Instantaneous acceleration of subject
- Distance travelled by the subject

3.4 Simulating an Actual Human Walk

Walking is a method of transportation wherein one moves from point A to point B by following a set of movements repeatedly. This repeated pattern called the gait is an important action to be captured as it contains requisite information for the analysis of the walk. In past, solution to this problem has been attempted by many methods. Video recording of small walks, Marker based data collection via video camera and infrared sensors, treadmills etc. have been used for data collection to analyze gait.
All these methods attempt to capture a complete gait and then extract the relevant information from it for further analysis.

Though these methods for capture of complete gait are quite robust, there are certain limitations to them. Most of these methods face the problem of limited space of operation and hence are unable to work if gait capture is required for longer period of time. Some methods can capture long walks, but are unable to capture motion in two dimensions. Those systems which can capture have very complex and large setups, which limit their capability for easy and movable installation. Also such systems are quite expensive to procure.

Thus the challenges in simulating a human walk from the actions performed by a human can be listed as follows:

1. Limited space against longer time duration walks
2. Amount of setup required
3. Cost of setup required
4. Trade off between straight line walks and walk with turns

3.5 Abstraction of Human Walk

As discussed above, a human walk is composed of multiple components. Moreover a human subject has multiple degrees of freedom which create a capability for highly dextrous movements which are complex in nature. Each movement is associated with
multiple muscles and joints coordinating to provide a reliable motion.

Capturing such massive data and computing it to obtain necessary model which can simulate a human gait is tedious, resource intensive, and complicated. For reducing the complexity of analysis for such a system, simplified models which can capture the behaviour of interest in the original system can be obtained. Such models, called abstractions, are easier to analyze as compared to the complex model. Therefore, a mathematical framework is required to filter unnecessary data and utilize the necessary information as per requirement. Such a framework for abstraction of a dynamical system is discussed in previous chapter.

The proposed pedestrian simulator will capture the walk of subject and then identify the gait information from it for recreating the movements of subject in the simulated reality. This can be visualized as an abstraction of the subject’s walk in the real world to simulated world. Looking at the limitations in previous section, a much simpler, cost effective approach was required so that it can be setup and reused easily. To address this problem we first defined the gesture that can be used for long walks and hence forth identify the solution for its implementation and analyze it.

3.5.1 Definition of on the spot walk

The solution visualized was defining an on the spot walk pattern, using which certain parameters can be defined and extracted which were then transformed into gait parameters to simulate the walking. This motion is hereby defined as:
the alternating motion of taking one foot up in air, while keeping the other foot on the ground, and returning it back to ground at the same point where it was before lifting off. The knee and hip joints will rotate appropriately to allow the thigh to come as closely parallel to the surface as anatomically feasible, thereby raising the knee joint close to waistline.

Hence an on the spot walk can be visualized as a continuous sequence of above defined gesture in figure 3.4.

Figure 3.4: On The Spot Walk
3.5.2 Analysis of on the spot walk

The on the spot walk or rather the simulated walk is divided into two phases the Lift when one foot is rising, while the other phase is called Fall which occurs when the risen foot is coming back on the surface. This nomenclature implies that there are two lift phases and two fall phases in each simulated gait for any person. Various parameters are extracted from this simulated walk, which enable it to be mapped to an actual gait. The relationship between the phases of an actual gait and simulated walk are shown in figure 3.5.

3.6 Model of Virtual Entity in 3D World

Since the final objective of the pedestrian simulator is to interface between the physical subject and their representation, it is an important aspect to define the entity in the 3-Dimensional virtual world.

The pedestrian in the virtual world is assumed to operate under following boundaries:

- A pedestrian always moves in a straight line.
- A pedestrian can move forward, backward and rotate by any angle.
- A pedestrian is able to vary speed and stride length.
- A pedestrian never takes side steps.
Figure 3.5: Mapping of walking phases
The above assumptions define that the pedestrian is nonholonomic in nature with linear and angular motion only. To model the assumptions mathematically, a pedestrian can be assumed as a rolling disc. Since the disc cannot side step itself, i.e., the disc cannot move in the direction of its area vector, hence, it becomes a nonholonomically constrained system. Moreover, now the disc can have linear velocity, and angular velocity along the direction of vector in the plane containing the disc. Therefore the disc can reach any point on the 2D plane.

The rolling disc model can be described as follows: Consider a disk of radius $\rho$, that rolls without slipping on a plane, as shown in figure 3.6, while keeping its midplane vertical. Its configuration is completely described by four variables: the position coordinates $(x, y)$ of the point of contact with the ground in a fixed frame, the angle $\theta$ characterizing the disk orientation with respect to the $x$ axis, and the angle $\phi$ between a chosen radial axis on the disk and the vertical axis.\[7\]

Therefore, the disc must satisfy the kinematic constraints

$$\dot{x} - \rho \cos \theta \dot{\phi} = 0 \quad (3.1)$$

$$\dot{y} - \rho \sin \theta \dot{\phi} = 0 \quad (3.2)$$

These dynamic constraints are not integrable, therefore disk can reach any point in its state space from any point in it, by any path. It is due to this reason this can
be assumed as the appropriate model for a pedestrian as each point of the virtual world is reachable.

3.7 Abstracting Map

Hence our state space $M$ can be considered as state space that belongs to the motion of a skeleton in physical world. This state space has multiple dimensions and encompasses many complex interactions. Moreover, this motion is required to be translated into the motion of a virtual entity in the 3D virtual world as the rolling disc model discussed above. Hence we can say that a particular higher dimensional, dynamical control system is being abstracted into a lower dimension dynamical control system.

For the reasons discussed before, this abstraction is the key to simplification of analysis and control for navigation by human subjects for the entities in the virtual world.
3.8 Conclusion

Though above explained approach is an inexpensive way and addresses all the limitations listed in previously tested approaches, it has a major drawback of comparing on the spot walk gait to an actual gait/walk. It is a limitation because it takes a user away from how they actually walk to a different approach which is less natural and more monotonous. This takes a user from their usual behavior and demands efforts in a continuously different situation thereby creating a greater psychological divide from reality. Therefore, it still cannot be taken without proof that eventual experience will not be engaging.
CHAPTER 4

CAPTURING HUMAN WALK

Before moving on to a project, we need to do a feasibility study and identify the possible challenges in the project. This chapter details the requirements for the prototype implementation of capturing the walking gesture and its analysis.

4.1 Simulated Human Walk

In the last chapter a gait was defined and analysis methods were identified and thus defined a simulated walk for a person to be performed inside a lab environment and associated mathematics was discussed. In short, the simulated walk can be described as on the spot walking. The motion is defined as lifting the knee up to above a certain position to be classified as a step. Thus each virtual step is established by a spot gesture and relevant definitions for various parameters of a gait are defined by it and calculated accordingly.

The definition of such gesture is an important objective to prove the simulation of a Human walk and is required to be validated. For validation of this definition and the concept, a prototype was made which has been discussed in this chapter. This prototype was built using Arduino and IR sensors.
4.2 Mapping actual to simulated walk

As observed in previous chapters, an actual gait is composed of many parameters, variables, motions and patterns. Hence, it is a complicated series of events. Therefore there is a need to filter the available information for efficient computation and implementation of pedestrian simulator. For this reason, we decided to map certain parameters of an actual walk to be implemented by the above defined on the spot walking gesture (Definition 3.5.1). This mapping has been given as below.

<table>
<thead>
<tr>
<th>On The Spot Walk Gesture</th>
<th>Simulated Gait</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Knee Height</td>
<td>Step Length</td>
</tr>
<tr>
<td>Knee Lift Frequency</td>
<td>Speed</td>
</tr>
<tr>
<td>Knee Position</td>
<td>Gait Position Cycle</td>
</tr>
</tbody>
</table>

Table 4.1: Gesture Parameter mapping

4.3 Extracted Parameters

For development of proof of concept, the requirement was to ensure a feature extraction for the defined action of walking. The primary problem in such situation is to sense physical movements, process them, and extract features and gestures and map their values into parameters in real time. As per the mapping defined in table 4.2, information about speed of walking, stride length and position with reference to gait cycle has to be extracted. This created a problem of identifying the right sensors,
and appropriate locations for tracking simultaneously minimizing the computations for fast and efficient response from the system.

The solution to this problem was implemented with the use of Infrared Sensors, placed on the feet of the subject, and two panels both in front and back of subject till knee height. The sensors on the feet were placed pointing in three directions bottom, front and back. The sensor facing towards ground was required to measure the rise of the foot and those towards front and back were to measure distance from their respective panels. The placement of sensors was aimed at recording the height of foot and the distance of foot from the front and rear panel. These features helped in computing all the required information for simulated gait as follows:

\[
\text{Distance between front and rear panel} = L \\
\text{Height of foot} = h \\
\text{Position from front panel} = x \\
\text{Step Length (s)} = k \times \frac{x}{L} \\
\text{frequency of steps (f)} = \text{No. of times foot raised above ground/second} \\
\text{Speed} = f \times s
\]

4.4 Construction

This section explains the construction for proof of concept for pedestrian simulator. The architecture is explained followed by the hardware construction and software design.
4.4.1 Architecture

The overall architecture has been divided into two parts viz hardware and software architecture. The architecture has also been detailed with respect to semantic breakdown of information flow and trigger actions. The complete flow of information can be visualized as follows:

Complete: The complete system can be visualized (figure 4.1) as a data acquisition and processing system where the sensor feedback is classified and assigned with relevant gestures to compute appropriate information. Actions and information: The

![Flowchart of complete system](image)

Figure 4.1: Flowchart of complete system

information flows from user’s actions into the system. Here a single action of 'on the spot walking' is further broken into constituent actions as follows:

- Walking
• Stride Length

• Direction of Movement

The information flows from the actions to sensors. The data of these sensors is acquired by a microcontroller and then processed to classify between the above three actions as well as compute related information. This information flow can be visualized as in figure 4.2.

![Information Flow Diagram](image)

Figure 4.2: Information Flow

**Hardware**

The hardware architecture is explained as follows: As shown in figure 4.3. There are three infrared sensors for each foot amounting to six sensors in total. These sensors are pasted along the feet in various orientations. Two sensors are facing front and back respectively and one sensor is facing downwards. Each of the sensor is connected to an analog input channel in the micro-controller from where the information goes to
Software

The software architecture is explained as follows: The software reads the data available at the analog input channel of the microcontroller and computes the formulae. Since the microcontroller has a single core, sequential instruction execution, we first read all inputs, compute all formulae and then store and display data and identify status of the user as displayed in the figure 4.4.

Figure 4.3: Flowchart of hardware system
4.4.2 Hardware Construction

The hardware has been constructed using readily available infrared sensors, Arduino Uno prototyping board, wireframe to hang the sensors on a foot and some cardboard boxes for localized location reference.

Installation Positions

Positioning sensors is always crucial in a setup as they minutely affect the calibration for the system software. For this setup also sensors required special installation positions. Sensors were placed at following positions (figure 4.5):

- In the gap between feet and ankle in the subjects shoe.
- At the toe of the shoe, facing outward, aligned perpendicular to the foot
- At the ankle of the shoe, facing outward, aligned perpendicular to the foot and parallel to toe sensor.

Circuit diagram and Components

The following components were used for the construction of the system. These components constitute the data acquisition and processing modules from the flowchart (figure 4.1).

1. IR Sensors, part number
2. Arduino UNO board
3. 1 usb A to B cable for Arduino

4. A wireframe to hold sensors.

5. Front and rear reference panels

6. Scotch tape and connecting panels

The infrared sensors give an analog signal as output, which need an ADC to convert to digital value which can be used for execution of decisions in the microprocessor. This ADC is included in the Arduino prototyping board. Thus it completes the necessary requirement. Similarly, the board is required to be connected to a PC for display and storage of values. This has been accomplished via usb connection to the computer system which displayed the data in a text window.

**Positioning of sensors**

Setup of a sensor system is an important step as it affects the calibration values which in turn will affect the calculations via code and hence affects the recorded values for gait conversion. For this reason a wireframe was designed to be hooked into any shoe of any size. The sensors were able to slide and the ideal condition is depicted in figure 4.5.

The sensors at toe and heel are required to be parallel and facing outwards, while the sensor at the side of the foot is facing down and may or maynot be perpendicular to the plane of sensors at toe or heel.
4.4.3 Software Design

There are two parts to the software design viz data acquisition and processing and frontend display of information. The software was written for Arduino and the output was observed on the serial console of a computer for the same. This output to serial port was then used in another software for display of calculated heading and speed of the subject based on theory discussed in previous chapters.

Platforms used

The platform used for programming the micro controller is called Arduino and the display platform used is called Processing. Both the programming platform are developed in java but the code developed in them follows C programming style and structure. The installation of these platforms is very simple and detailed instructions can be found at [ref] and [ref].

Files

The code for the software developed is attached in the appendix. It is divided into three primary parts as calibration code for microcontroller, running code for the microcontroller and frontend visualization.
Calibration

The hardware sensors do not behave same during their lifetime. Moreover two units do not always give the same output. The calibration algorithm is written inside the microcontroller along with the computational code. For process of calibration, a switch determines the state of execution for the code. This is implemented on microcontroller by setting PIN# as “HIGH” for calibration mode, and low implied the regular data acquisition and processing mode.

The calibration mode is to determine the initial point of reference as well as the range for the data acquisition by defining central position of the subject and stopped feet condition when both are resting on ground. Once the subject is in position as shown in figure 4.6 while the calibration pin (PIN#) is switched to LOW, it calibrates the sensor values to the existing situation. It sets the minimum value for the bottom foot sensor and sets the center between the front and rear panel.

4.5 Limitations

After completion of this process, three main limitations were observed which can create problems in the pedestrian system.

1. The downside of this procedure is to calibrate the setup for every subject manually.
2. The detection of turn is not natural and requires push button input.

3. The system is bulky and clunky, and requires carefulness even during operation.

4.6 Testing and Results

This system was attached with shoes of a subject and then the results were obtained on the screen. Since this was a prototype, not much emphasis was given on visualization, except raw data on console.

Eventually the result was that we were able to successfully capture the walking behavior of person performing a walking gesture using a microcontroller and sensor based system. These parameters were enough to model a human gait.

There were two main conclusions drawn out of it:

1. We can easily extract the information and employ them to generate gait parameters; therefore, the features have been correctly identified.

2. An attached sensor based system is difficult to calibrate and monitor. Also such system will take time to setup for every subject. Moreover the system becomes clunky due to involvement of multiple wires. Therefore such systems are impractical in nature.
Figure 4.4: Flowchart of software system

1. Read Sensor Data
2. Check Calibration Pin: If No, go to Send to PC via Serial Port; if Yes, go to Store Calibration Constants.
3. Check Any Foot in Air: If No, go to Grounded Foot Close to Wall?; if Yes, go to Compute % Gait Completed.
4. Grounded Foot Close to Wall?: If No, go to Both Feet on Ground?; if Yes, go to Change Speed per Distance.
5. Both Feet on Ground?: If Yes, go to Compute Heading Deviation; if No, go to Send to PC via Serial Port.
6. Send to PC via Serial Port.
Figure 4.5: Placement of sensors

Figure 4.6: Calibration Position
CHAPTER 5

IMPLEMENTATION USING NATURAL INTERACTION

This chapter studies about natural interaction, its benefits and the implementation details using ms kinect for the purpose of implementation of pedestrian simulator.

5.1 **What is natural interaction**

In general, our method of communication is through gestures, expression and movements and which may also be aided with audio or speech. In this process they may interact with each other or with the environment. There is no need to wear any devices or learn some new instruction, it is completely intuitive and is considered natural mode of interaction with environment. Thus Natural interaction is defined in term of experience as

People naturally communicate through gestures, expressions, movements, and discover the world by looking around and manipulating physical stuff; the key assumption here is that they should be allowed to interact with technology as they are used to interact with the real world in everyday life, as evolution and education taught them to do [26].
5.2 Why is natural interaction required

Natural interaction capable devices have a number of advantages to traditional button and sensor based interfaces as follows:

1. They eventually remove complicated user interfaces.
2. They are easier to maintain.
3. They are considerably lower in cost
4. They, above all are intuitive, hence have a really fast learning curve

Natural Interaction is a design methodology for the next generation devices and physical interaction spaces, and provides capability to general users for using what is intuitive and leave the bulky and complicated interfaces. Mostly this in itself is a research topic which has been researched heavily for many different usage scenarios under the subject of user interaction design. We have identified and covered those during earlier chapters.

5.3 Available options

Currently market has two major options available capable of defining natural interactions viz, Microsoft Kinect and Nintendo Wii. One third option is development using a video camera and OpenCV based solution. Effectively this brings us to a situation of selecting the most cost effective and durable device among the available options.
5.3.1 Why MS Kinect

are available in market currently To identify the device fitting our requirements,

Table 5.3.1 shows a comparison on list of features among the three alternatives.

<table>
<thead>
<tr>
<th>Features</th>
<th>MS Kinect</th>
<th>Nintendo Wii</th>
<th>OpenCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor type</td>
<td>Multiple Video Camera</td>
<td>Accelerometer</td>
<td>Video Camera</td>
</tr>
<tr>
<td>Controller</td>
<td>Not required</td>
<td>Atleast one controller</td>
<td>Configurable</td>
</tr>
<tr>
<td>Setup</td>
<td>Plug and play</td>
<td>Plug and play</td>
<td>Various configurations</td>
</tr>
<tr>
<td>Depth Perception</td>
<td>Available</td>
<td>Not available</td>
<td>Algorithm &amp; hardware</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>dependent</td>
</tr>
<tr>
<td>Development Support</td>
<td>Excellent</td>
<td>Moderate</td>
<td>Open source community</td>
</tr>
<tr>
<td>Special</td>
<td>Skeleton &amp; face detection</td>
<td>fast movements</td>
<td>none</td>
</tr>
<tr>
<td>Precision</td>
<td>High</td>
<td>High</td>
<td>medium</td>
</tr>
<tr>
<td>Speed</td>
<td>High</td>
<td>High</td>
<td>Slow</td>
</tr>
</tbody>
</table>

Table 5.1: Comparision Table of Available Products

Based on the above table, we selected MS Kinect because of the various features available as well as simplicity in setting up. Moreover the Kinect is backed up by software giant Microsoft, hence a good support both online and community is at disposal as well as a trouble free sdk is available for use. Moreover, Kinect is a commercial grade product aimed to provide many aspects of game development for the developers. This opens a wide window of opportunities for a better product in every aspect.
5.4 Setting up kinect

This section gives a technical walkthrough of how to install a kinect on a PC and then how to install the kinect for pedestrian simulator.

- Setting up Kinect Sensor: Open the kinect from the box.
  
  Attach the power cord to sensor and main power supply.

![Image of kinect](image)

**Figure 5.1: Out of the box**

Mount the sensor on a stable surface above or below the display.

- Install Kinect SDK


  Go through the guided installation steps. Please note that it works only with
Visual Studio 2011 and upwards. connect the usb to PC and wait for all the sensors to get installed

Done, The Microsoft Kinect is set up and ready to use.

5.5 Kinect Interaction Space

The interaction space is defined as the area in front of the kinect sensor where the infrared and color sensors have unblocked view of everything in front of the sensor.
The Assumption is that the lighting is not too bright and not too dim, and that
the objects being tracked are not too reflective. While a Kinect is often placed in
front of and often at the level of a users’s head, the sensor can be placed in a wide
variety of positions [14]. Eventually this is defined by the field of view of the kinect
cameras. Moreover this is also supported by tilt angle which greatly increases the
interaction space. This tilt angle is controlled by a motor inside the sensor. This can
be visualized in figure 5.5.

Figure 5.4: Kinect Interaction Space [14]
5.6 Kinect for Windows Architecture

The hardware software interaction using a kinect sensor can be visualized as shown in figure 5.5. Kinect is a complex device and provides video image, depth image (using IR camera) and audio signal. These are interfaced with Natural User Interaction library provided by Kinect SDK, which is eventually used by the application to perform programmed actions. The Kinect for windows SDK has three broad types of components which can be visualized in its architecture and is shown in figure 5.6.

These components include the following:

1. Kinect hardware - The hardware components, including the Kinect and the USB hub through which the sensor is connected to the computer.

2. Kinect drivers - The Windows drivers for the Kinect, which are installed as part of the SDK setup process as described in this document. The Kinect drivers support:

   - The Kinect microphone array as a kernel-mode audio device that you can access through the standard audio APIs in Windows.
Figure 5.6: Kinect SDK Components Architecture [15]

- Audio and video streaming controls for streaming audio and video (color, depth and skeleton).
- Device enumeration functions that enable an application to use more than one Kinect.

3. Audio and Video Components Kinect Natural User Interface for Skeleton Tracking, Audio, Color and Depth Imaging

4. DirectX Media Object (DMO) for microphone array beam-forming and audio source localization.

5. Windows 7 standard APIs - The audio, speech, and media APIs in Windows 7, as described in the Windows 7 SDK and the Microsoft Speech SDK.
5.7 Programming Models

[the programming architectures followed by SDK] Kinect operates using two cameras viz RGB and Infrared. Each camera is composed of a Charge Coupled Device (CCD) which acts as a screen to capture an image and transfer the relevant bits and bytes of information for processing. Each such snapshot of a subject in front of kinect sensor is called a frame.

Application using Kinect sensor retrieves image frames from it. The process of retrieval involves requesting a frame from the sensor where each frame is passed to a buffer whenever requested. A frame is passed only when it is completely captured inside the CCD of sensor. The sensor data stream never provides the same frame to the application more than once. To get the frame there are two programming models followed viz Polling Model and Event Model [13].

5.7.1 Polling Model

It is the simplest model for reading data frames. The application opens the image stream and then requests the frame after every predefined time interval. The request method returns a new image frame when it is ready or at the expiry of wait time, whichever comes first.

On successful return the new frame obtained is ready for processing. If the time-out value is set to zero, the application code can poll for completion of a new frame while it performs other work on the same thread [13].
5.7.2 Event Model

In this approach, the code passes an event handle (a pointer to a function) to image retrieval method from SDK. Its processing can be visualized as when the image is ready to be read from the buffer of the sensor, it is termed as an event and the associated function in the code is called for processing the data. During this data processing, the event is reset by the NUI Image Camera API.

The event model in Kinect SDK supports the ability to integrate retrieval of a skeleton frame into an application engine with more flexibility and accuracy [13].

5.8 Software architecture

Objective of pedestrian simulator is to track a walking human gesture and then process data to draw inferences for creating simulation of a walking human. This code structure has the flowchart as shown in figure 5.7 below. The architecture primarily consists of three layers Data Extraction, Data Processing and Event Generation where each of these semantic layers specify a classification of objectives addressed in the code structure:

5.8.1 Skeleton identification

Natural User Interface (NUI) library provides a built-in class to extract skeleton related information as the embedded algorithms inside the Kinect sensor can detect and track two human skeletons at the same time.

In our scenario, we always assume that we are tracking only a single skeleton at
Figure 5.7: Pedestrian Data Extraction Architecture
a given time and there are no more skeletons present in the picture. The Kinect sensor sends a signal and the skeleton data every time it confirms one and finishes the computation for it. On this signal skeleton data is read and then transferred to the data reading function where gesture identification is done and appropriate information is extracted.

5.8.2 Gesture definition

The skeleton data is actually tracking a set of points mapped in two dimensions and providing real time location of 21 joints. Each of these joints is accessible by using a macro defined in the SDK and hence are easily tracked. As discussed in chapter 2 and 3, we defined the gesture to be an on the spot walk to map the parameters for a human walk. It is this gesture that we have defined earlier, is what we want to extract.

We also define an intent to walk occurrence as a small lifting of the foot as to the situation where the subject might be thinking of taking the next step but then stops right in the middle of it. If he brings it back before a certain position or height, we can call it intent to walk and ignore to avoid the subject taking an unintentional step.

5.8.3 Feature extraction

This on the spot walk gesture is defined by the knee movements up and down where their height determines the step length and the frequency of up down determines the speed of the pedestrian. These features are extracted from the gesture by continuous
monitoring of the knee joints from the available skeleton data. These values are then de-noised to remove false positives. Also the intent to walk occurrences are also detected and ignored by not triggering and changing the values for gait parameter conversion.

5.8.4 Gait parameters conversion

The feature extraction from the skeleton data is used to convert to gait parameters. Recall that a gait is defined as the pattern of movement of limbs. For this pattern to continue, we identified the parameters in chapter 2. These parameters of an actual gait are then related to the on the spot walk gesture and the relations were introduced in chapter 2.

These generated parameters are continuously updated in the walking pedestrian model (rolling disc model) to generate current state of the pedestrian which will be required for a detailed graphical modelling.

5.8.5 Step event generation

When a subject takes a step, it is characterized by one full lift of the foot and then back to the resting position. Therefore we call that a subject has taken a step only when the foot goes up and down, and this generates a step event. A step event marks the completion of feature extraction and then triggers the computation of the subject’s global parameters like current speed, location etc.
5.8.6 Global parameters conversion

Global parameters are those which define the state of individual pedestrian in the complete visualization world. These parameters are speed, direction of heading and current location. They effectively point a state of pedestrian but are updated by the computed gait parameters over time.

5.9 Conclusions and limitations

[for this approach] This approach makes the development of this simulator modular and thereby allowing it to be integrated in any software and application framework. However this architecture has limitations as follows:

- Cannot observe direction of motion using Natural Interaction.
- Requires calibration for each new user.

In further chapters we attempt to address these limitations.
CHAPTER 6

NAVIGATION ON 2D PLANE

Kinect is a versatile sensor which can be used in multiple scenarios. In this thesis, it was used to extract information about the gestures performed by a subject to help them navigate through a virtual network. The algorithm discussed in previous chapter allows to extract information related to human gait. But navigation in 2D plane is an important aspect and such a framework should be able to address it. In this chapter methodologies are discussed to be able to perform the same.

6.1 Current Limitations

Movement in single dimension or a curve implies motion in forward or backward direction. But on a two dimensional plane it also requires taking perpendicular turns or turning on a curve. With these things in perspective, the current framework is able to capture the walking gesture, but is incapable of the following

- Inability to distinguish between walking backward and forward.

- Taking left and right turns during walking gesture.

- Taking a graduated turn along some curvilinear path.
It is imperative to have a solution to solve these problems in the framework, which otherwise will be fall short of providing access to multiple possible paths and motions. Therefore following three methods have been identified:

- Method 1: Using multiple (two) kinect sensors to track
- Method 2: Tracking decrement in body constants (hip width, etc.)

6.2 Method 1

Kinect for Windows support multiple kinect sensors to be connected to a same system, being governed by the the same application. This allows for many wonderful and new interaction spaces that can be thought of, hence, leading to new application architectures that can be applied to many problems.

In case of recreating two dimensional motion, inside a virtual network with the inputs by the subject, it is important to detect the position and degree of rotation (turn) taken by the subject. Since motion is two dimensional, it is intuitive to keep the two kinects perpendicular to each other as shown in figure 6.1.

This method will help in generating more accurate detection of how much the subject has turned, as evident from the figure 6.1. This is accomplished with skeleton detection and face detection from the video stream from the kinect sensor. This will help to identify that towards which kinect sensor is the subject facing and sub-
sequently feature extraction will be from that sensor. Although this approach will also suffer from the problem of differentiating between forward and backward motion.

Another arrangement for perpendicular placement of two kinect sensors can be seen in figure 6.2. This arrangement will again help in generating a more accurate detection of the turn taken by the subject. As seen in figure 6.2, in this approach one kinect sensor is mounted over the head facing the floor. This kinect sensor will be used to identify the direction in which the subject is facing by using a directional marker on their head. This directional marker can be color coded, or shape coded. Also it is not necessary to use a kinect sensor over the head, and another video camera can be placed for the same. But since kinect sdk is being used already, it would greatly simplify the programming and hence using a kinect sensor is more sensible approach.

The limitation of this method is more practical in nature. Due to involvement of multiple kinect sensors, the setup of the interaction space will be difficult. The
sensors will have to be placed accurately and calibration will require repositioning them. Hence the setup of this will require a lot of time and will be susceptible to any accidental movement of the sensors.

6.3 Method 2

For every subject, while performing on the spot walk gesture in front of kinect sensor, the plane containing the shoulder and the hip region (plane of subject) is parallel to that of the sensor (plane of sensor). This information can be used to define whether the plane of subject is parallel to the plane of sensor. Therefore, the hip and shoulder width observed is a result of taking a projection of the plane of subject on plane of sensor (refer figure 6.3). This will accurately define the tilt of the
subject with respect to the sensor and hence the direction of movement.

To check whether the subject is moving towards the sensor or away from the sensor, face detection can be used. If the face is found in the video stream, it will imply that user is moving in a specified direction in the virtual world, let it be North. In case if is not found and the projection value of shoulder width is nearly same as that when the planes are parallel, the subject is assumed to be moving South.

This points to the limitation of this approach as it becomes difficult to point whether the subject is moving east or west while turning. In other words left and right turns cannot be distinguished. An alternative solution to this limitation can be wearing

Figure 6.3: Geometry of relating body constants
colored markers perpendicular to the plane of subject, placed on each shoulder.

6.4 Method 3

With the advent of gesture based control, each interaction is nowadays thought of in terms of gestures. Similarly we have defined and used an on the spot walk gesture for obtaining information for the gait parameters for any person. Hence, it is intuitive to define another gesture for turning while walking.

In this method, a new set of gestures is defined as follows (and seen in figure 6.4):

To turn left, raise the left hand above torso. The turn command will be relayed till the hand remains above torso, in constant increments. This will be independent of whether the person is performing on the spot walking gesture or not. Similarly, the gesture for right turn is defined.

Therefore, this method will allow to turn with ease with a limitation on the speed of turn. A way to dynamically alter the speed of turn as per the requirement of subject is desired. Also, such a method constrains a subject to behave in a non-intuitive manner as it is more intuitive to turn left yourself, rather than raise hand to turn.

6.5 Selected Final Method

After the above analysis, a number of issues for selection of final method surfaced. The primary being ease of deployment and complexity of setup, followed by ease of development and number of kinect sensors. Also this linked to an indirect part of
problem, the calibration. Calibration is most important procedure for any system as it makes the system adaptable to varying conditions.

Based on these issues, method 3 was selected as it was easiest and most robust implementation strategy. Although this method is less intuitive in nature, but given the increase in cost and complexity of the system in other two methods, it seems to be a worthy trade-off.
CHAPTER 7

FINAL IMPLEMENTATION

This chapter details the development and execution of the final software framework, based on the research detailed in previous chapters. At the end of chapter suture work and limitations are also discussed.

7.1 System Architecture

This project is aimed at development of an application programming interface (API) for user interface engine of a pedestrian simulator, to allow a subject navigate through a virtual network. The output is aimed to be generic, returning various gait parameters in realtime, the number of steps taken and turning values. As mentioned, the api has to be realtime, maintaining total and instantaneous parameters. For this the system architecture is as shown in figure 7.1. This architecture has been explained in following subsections.

7.1.1 Initialization

This part of the API architecture is called to initialize various functions necessary for tracking skeleton and hand genstures. In this function a thread is initialized which
Figure 7.1: Complete Architecture of Pedestrian Simulator API
communicates to NUI SDK for Kinect sensor to update skeleton data. This function also initializes another thread which updates current values for a pedestrian object from the skeleton data.

This initialization is a part of the constructor for the object. A constructor is that part of the program which is invoked on creation of an object. This part ensures that default values are loaded into the variables inside the pedestrian object and avoid the problem of loading garbage value in them. Hence it ensures smooth functioning of the program.

7.1.2 Calibration

This part of the API architecture is called immediately after initialization or at any point during processing to define certain setup dependent variables for computation of various gait parameters. These variables are shoulder width, hip width, maximum and minimum knee height and wrist height from ground.

This method ensures that subject of all body types can participate and navigate. This ensures the independence from height, width and similar body features. Hence it allows for all kinds of subjects in every age group. It also allows to switch subjects during a single run if needed.

The calibration process is guided in nature, and hence the API will only support calibration on a handshake type protocol. The method to communicate with user has to be built around it.
7.1.3 Get Current Gait Values

A gait has various parameters which can be called as instantaneous parameters. These define the current phase of the gait for a subject. The mapping can be seen in figure 3.5. Using the value returned from this module, the graphics engine can represent the status of the subject in one of these phases.

This module returns the current phase of walking i.e. stance or swing phase and the percentage completion of that phase. Therefore a suitable graphics engine can render appropriate views in the 3D environment for the subject.

7.1.4 Get Pedestrian Data Values

A subject while walking has many important secondary data points. These include, but are not limited to, instantaneous speed, a complete step taken, deviation from geographic North of virtual environment etc. This data is required to be fetched at every point of time while the application is running. Therefore, it is important to keep this method in a thread associated with graphics.

7.2 Test Case Implementation

To employ the above mentioned API, a text based user interface was implemented which printed various parameters on the screen. The following screenshots show the implemented solution using the pedestrian API built during this project.
Figure 7.2: User in front of Kinect Sensor

Figure 7.3: User in changing heading
Figure 7.4: User stepping and increasing distance travelled
CHAPTER 8

CONCLUSION & FUTURE WORK

8.1 Conclusion

This thesis is a summary of implementation about a system, for the development of human based pedestrian simulator, which can be used in conjunction with other simulator systems [21] or in a standalone setup. As a result we have been able to iteratively define and design a robust system and architecture for data acquisition and conditioning for use with an envisioned pedestrian simulator.

Another problem with system development for use with human subjects is idea of one size fits all, for which we have been able to correctly define and implement a calibration procedure to ensure uniformity in the experience and results.

8.2 Future Work

The developed system, as mentioned above, is aimed at providing the necessary data acquisition framework for tracking a human movement and converting it for simulator. However such kind of simulator still needs to be developed and remains as a goal for future work. Moreover as per the co-authored publication [21], A more integrated simulator system is envisioned, where this pedestrian simulator will be a...
part of the bigger system.

The code of this thesis is planned to be made available as open source under GPL license, thus fostering research and development with the help of a wider community.
CHAPTER 9

BIBLIOGRAPHY


[2] KEN BELSON. Sidewalk smackdown; no, you can’t walk and talk at the same time. NY Times, 8 2004.


[17] Lee Morgan. The effects of traffic congestion. *USA Today*.


VITA

Graduate College
University of Nevada, Las Vegas
Romesh Khaddar

Degrees:
Bachelor of Technology, Electronics and Communication Engineering, 2008
Indian Institute of Technology (IIT), Guwahati, India

Special Honors and Awards:
Graduate and Professional Student Association Travel Award (Summer 2012)
Graduate and Professional Student Association Travel Award (Spring 2012)
Third Place Recipient, The Howard R. Hughes College Of Engineering 2nd Graduate Celebration (2012)

Publications:


Thesis Title: Modeling and Development of Human Interface for Pedestrian Simulator

Thesis Examination Committee:
   Chairperson, Dr. Pushkin Kachroo, Ph.D.
   Committee Member, Dr. Ke-Xun (Kevin) Sun, Ph.D.
   Committee Member, Dr. Yingtao Jiang, Ph.D.
   Graduate Faculty Representative, Dr. Zhonghai Ding, Ph.D.