UNIVERSITY OF NEWCASTLE UPON TYNE

SCHOOL OF MECHANICAL & SYSTEMS ENGINEERING

Title:

Human Recognition, Identification and Tracking using Microsoft Kinect Interfaced with DaNI Robot

Author: Diyar Khalis Bilal

Course Title: MSc Mechatronics

Supervisor: Dr. Michele Pozzi

Academic Year: 2012 – 2013

This dissertation is submitted in partial fulfillment for the Degree of MSc in Mechatronics

SCHOOL OF MECHANICAL & SYSTEMS ENGINEERING

MSc IN MECHATRONICS

PROJECT DECLARATION FORM

Student Name:	Diyar Khalis Bilal
Student Number:	120015661
Project Title:	Human Recognition, Identification and Tracking using Microsoft Kinect Interfaced with DaNI Robot
Admission Date:	September 2013

DECLARATION:

I hereby declare that the contents and results of this MSc dissertation are the source of my own research, analyses and construction and this dissertation is intended only for the completion and submission of my MSc programme at the School of Mechanical and Systems Engineering.

I also certify that no part of this dissertation has knowingly been copied, directly or even indirectly, from someone else's work, without providing due credit of their contribution.

Student Name	Student Signature	Date
Diyar Khalis Bilal		

Supervisor Name	Student Signature	Date
Dr. Michele Pozzi		

Supervisor Comments:

Acknowledgments

I would like to express my gratitude to Dr **Michele Pozzi** for his valuable and constructive suggestions during the development of this work, and for his constant support and guidance throughout the year.

I would also like to offer my special thanks to my wife, whom without her limitless support I could not have finished this work. Moreover, my infinite thanks to my parents for their endless love and support.

Abstract

In this study, the functionality of an autonomous mobile robot is improved by interfacing it with Microsoft Kinect for finding and tracking a previously identified human target while avoiding obstacles both while searching and tracking. A suitable platform for fixing Microsoft Kinect and a laptop on a mobile robot is designed considering simplicity and practicality. The human identification and tracking methodology and algorithm is analysed for real life scenarios. The developed algorithm is applied to the mobile robot using robotics software and evaluated through real life scenarios achieving pleasing results. The mobile robot is able to identify humans, search for them and track them while avoiding static obstacles both while searching and tracking at indoor environments where no direct sunlight is present.

List of Tables and Figures

Table of Contents

1 Introduction

1.1 Autonomous Mobile Robots

In recent years, there has been an increasing interest in autonomous mobile robots due to decreased cost of manufacture of robots with powerful processors on board, reduced overall size of robots, easy to program and control because of availability of powerful and easy to use software at affordable prices. One of the leading companies in this field is National Instruments, which provides an industrial grade robotics platform designed for prototyping a robotics system called NI LabVIEW Robotics Starter Kit[1] known as DaNI. The kit includes an ultrasonic sensor, encoders and an NI Single Board RIO device for embedded control, which can be programmed using the LabVIEW graphical development software developed by National Instruments.

1.2 Microsoft Kinect

One of the most significant inventions of 2010 is perhaps the Microsoft Kinect[2] because of the high-resolution depth and visual (RGB) features that it provides for a relatively much lower cost when compared to other 3D cameras such as stereo cameras[3] and Time-Of-Flight cameras[4]. In November 2010, Microsoft launched Kinect, a motion sensing input device for the Xbox 360 gaming console, which makes control and interaction with Xbox 360 possible without using a game controller but by using hand gestures and spoken commands. Later on June 2011, the Kinect for Windows Software Development Kit (SDK)[5] was released to allow developers to write Kinecting applications in C++, C# or Visual Basic. After that, a Windows version of Kinect was released in February 2012. In 2011-2012, a group of mechanical engineering students from the University of Leeds developed a toolkit for LabVIEW named Kinesthesia[6] to allow access to RGB video, depth camera and Skeletal tracking functions of Microsoft Kinect, which was initially developed for medical rehabilitation and surgical tools.

1.3 The Problem

So far, however, there has been little work done on interfacing Microsoft Kinect with an autonomous mobile robot for human recognition and tracking. In addition, no work has been found that makes use of the skeletal tracking capability of Kinect while interfaced with a mobile robot. Recently a similar work was performed by Susperregi (2013)[7] who successfully interfaced Microsoft Kinect with a mobile robot for human recognition and tracking. However, the skeletal data obtainable from Kinect was not used for human identification, instead a vision based colour technique was used to identify certain emergency staff.

1.4 Aims and Objectives

The main aim of this project is to improve the functionality of a mobile robot by finding, recognizing and tracking a previously identified human target using Microsoft Kinect while avoiding static obstacles and continuously feeding back the location of the target to the user. In order to satisfy the above aim, the following objectives have been defined:

- Random exploration of an indoor environment to find the pre-identified human target.
- Using skeletal figures to recognise humans from environment.
- Identifying a certain person and tracking only the identified person using skeletal data.
- Human target tracking while keeping a minimum distance to the target.
- Avoiding static obstacles both while exploring and tracking.
- Localization to feedback the position of Robot and Target to the User.

1.5 Outline of Project

The overall structure of the study takes the form of six chapters, including this introductory chapter. Chapter 2 discusses the current knowledge in the field of human recognition and tracking using mobile robots.

Chapter 3 begins by laying out the specifications of the robotics platform and sensors used as well as the software involved in programming them are described in detail. In addition the design of the developed platform and the hardware connection is shown at the end of this chapter. Chapter 4 is concerned with the methodology used for human identification and tracking with detailed analysis of the proposed methodology.

In Chapter 5 the calibration and evaluation of Microsoft Kinect, as well as the practical results obtained are discussed and presented in detail. Finally, the conclusion and future works proposed are conferred in chapter 6.

2 Literature Review

In this chapter the current knowledge in field of human identification and tracking as well as the sensors used for human detection are discussed.

2.1 Microsoft Kinect

In recent years, there has been an increasing amount of literature on Microsoft Kinect due to its depth sensing quality, which it offers at a much affordable price when compared to similar ranging sensors available now on the market such as SwissRanger[8] and PMD[9]. Microsoft Kinect's operation principle is based on triangulation[10] process as described by inventors, which is fundamentally different from the time of flight measurement process used in traditional ranging sensors. The advantage of depth cameras over conventional intensity sensor cameras is that the depth camera can work in low light conditions, are unchanging by change of colour and texture and are capable of resolving contour uncertainties in pose[11]. The geometric quality of depth data acquired by the Kinect sensor was theoretically and experimentally analysed by Khoshelham (2012)[12] and concluded that at a maximum range of 5 meters the random error of depth measurement was only 4 cm and the error increased quadratically with increased distance from the sensor. Khoshelham (2012) recommended data to be acquired within 1-3 m distance to the sensor for general mapping applications due to the degraded quality of data at larger distances because of noise and low resolution of the depth measurements.

Another innovative feature of Microsoft Kinect, which this project is concerned about, is its advanced skeletal tracking capabilities that offers new possibilities in the area of human recognition, accurate measurement of skeletal joints, natural user interface (NUI) and human activity analysis. Shotton (2011)[11] represents one of the core components of the Kinect's skeletal tracking algorithm. Shotton (2011) proposes a new method to predict 3D positions of body joints from a single depth image quickly and accurately without using temporal information, but by taking an object recognition approach that maps the difficult pose estimation problem into a simpler per-pixel classification problem by designing intermediate body parts representations. The joint prediction accuracy achieved by Shotton (2011) was a pleasing average precision of 0.731 mAP (mean average precision) on synthetic test set, while on the real test set an outstanding mAP of 0.914 was achieved on head, shoulders, elbows and hand joints.

2.2 Human Recognition, Identification and Tracking using Mobile Robots

A considerable amount of literature has been published on human recognition, identification and tracking. These studies mainly focus on recognising humans from environment based on vision techniques or laser sensors to detect legs' position[13], or a combination of both. In most of the studied cases, the camera or the sensor is stationary as in security surveillance systems. So far, however, there has been little work done on using cameras or sensors to recognise and track humans interfaced with mobile robots. This is may be due to the unavailability of a reliable and affordable device for human recognition until recent years.

A recent study by Susperregi (2013)[7] combines colour-depth, laser and thermal sensors to follow humans in a mobile robot. Microsoft Kinect was used as a RGB-Depth camera due to the rich data set it offers at a relatively low cost when compared with other range sensors. According to Susperregi (2013) because of Kinect's limitation in people detection on a mobile platforms due to the fact that its algorithm relies on images captured by a static camera, an additional Hokuyo laser and a thermopile array sensor (HTPA) developed by Heimann were used to cope with this and some other practical limitations of Microsoft Kinect. The information obtained from these three sensors were combined in a real-time particle filter to calculate the position of the target based on image features, probabilistic leg and thermal patterns and optical flow to this end. The three sensors are mounted on a RMP Segway mobile platform as shown in the figure 1 below:

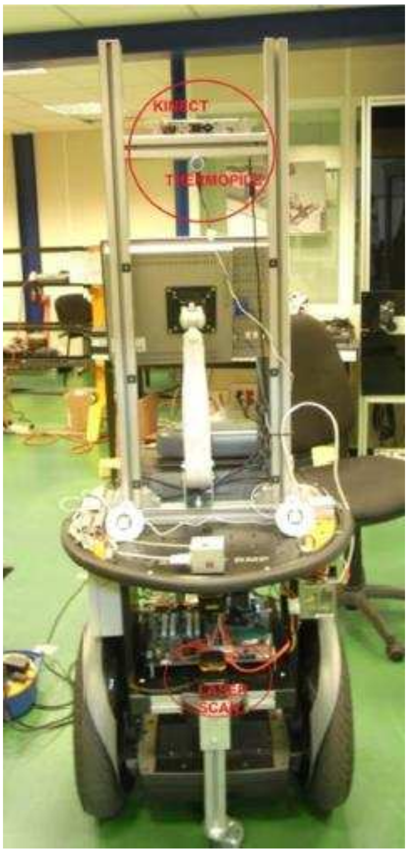


Figure 1 Segway RMP 200 Robotics platform, interfaced with Microsoft Kinect, a Hokuyo laser and HTPA thermal laser[7]

Susperregi (2013) proposes a parallel processing and filtering of multiple sensors and the implementation of the proposed algorithm is based on ROS system as shown in figure 2 below:

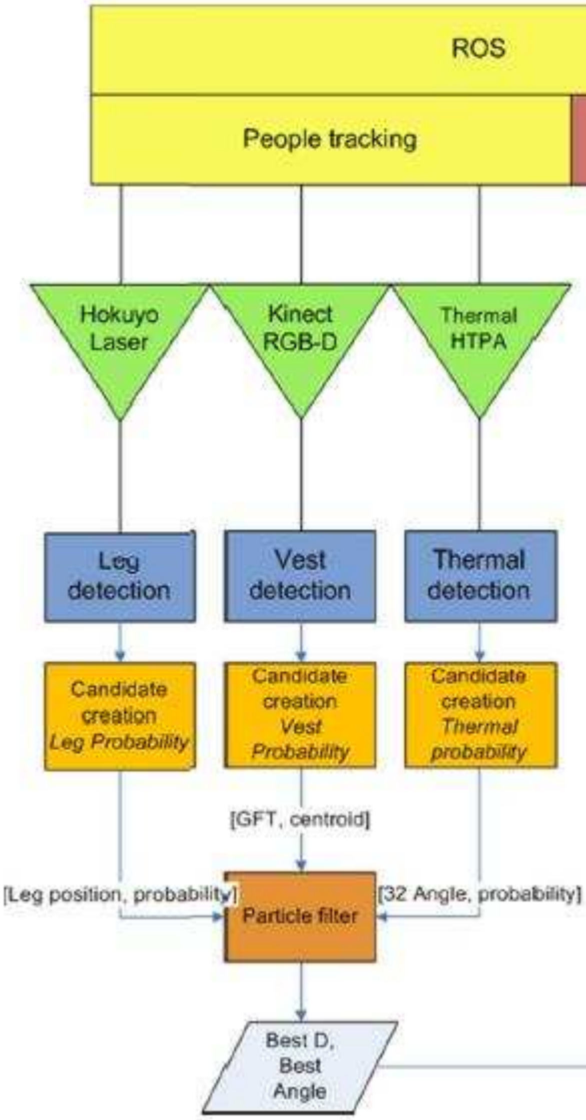


Figure 2 Approach combining three input clues from RGB-D sensor, laser and thermal sensor using a particle filter approach[7]

Each sensor is responsible for a particular task:

- The HTPA thermal sensor provides temperature distribution of the environment that can be used for several applications such as fire, hotspot, infrared radiation or person detection. The information obtained is converted to into an image with each pixel corresponding to a temperature value. In this project it was used for human detection since the temperature of a person is usually around 37 Celsius, which is correspondent to the pixel values, hence the location of a human can be easily estimated if no other significant heat source is available to interfere.
- Microsoft Kinect provides both the Colour (RGB) and Depth images with a resolution of 640x480 at 30 frames per second (fps). The colour image is used for human identification based on the colour of the vest worn by a person. While the depth map is used to estimate the position of the human target.
- The Hokuyo UTM-30LX scanning laser is used for detection of leg and obstacles. With a range of 0.1 to 30 m and an angular resolution of 0.25 degrees (1080 readings per scan) a measuring area of 270 angular degrees is provided. For leg detection, the probabilistic leg pattern method presented by Martinez-Otzeta[13] was used.

The output from each sensor was fed to a Particle filter, which chose the best (most reliable) input to set the position of target human to be followed. The experimental results obtained by each method individually and combined are as shown in tables 1 and 2 below:

Sensor	Mean	Standard deviation
Leg Detection (L)	66.17	53.37
Vest Detection (V)	30.68	28.26
Thermal Detection (T)	69.35	32.11
0.5V 0.5T	69.43	31.65
0.5L 0.5T	56.26	49.18
0.5L 0.5V	28.42	25.58
0.3L 0.3V 0.3T	33.29	32.75
0.15L 0.7V 0.15T	17.44	22.54

Table 1 Results in terms of error in angle (degrees). The numbers before the letters refers to the weight given[7]

Sensor	Mean	Standard deviation
Leg Detection (L)	0.51	0.45
Vest Detection (V)	0.27	0.33
0.5V 0.5T	0.62	0.25
0.5L 0.5T	0.50	0.45
0.5L 0.5V	1.87	0.28
0.3L 0.3V 0.3T	0.25	0.29
0.15L 0.7V 0.15T	0.10	0.31

Table 2 Results in terms of error in depth (m). The numbers before the letters refers to the weight given[7]

The experimental results obtained by this approach were rather in favour of Microsoft Kinect, which the vest detection was based on. Since the best estimate for both angle (between the centre of robot and the target human) and depth (distance to target) individually and in combination with other methods was achieved by vest detection, hence Microsoft Kinect. However, it is clearly seen that when all three methods are fused together the estimation of the angle and distance is better when compared with any other methods individually.

From these studies it is made clear that the most reliable and cost effective sensor for human recognition and tracking for indoor environments is Microsoft Kinect, even if it is used on a mobile robot.

3 Design and Specifications

In this chapter, the specifications of the robotics platform and any additional sensors used are described in detail along with any modification done to the platform.

The robotics platform and the sensor used are as follows:

1. Robotics Platform: NI LabVIEW Robotics Starter Kit 1.0 (DaNI) for Education
2. Additional Sensor: Microsoft Kinect for Windows

3.1 NI LabVIEW Robotics Starter Kit 1.0 (DaNI)[15]

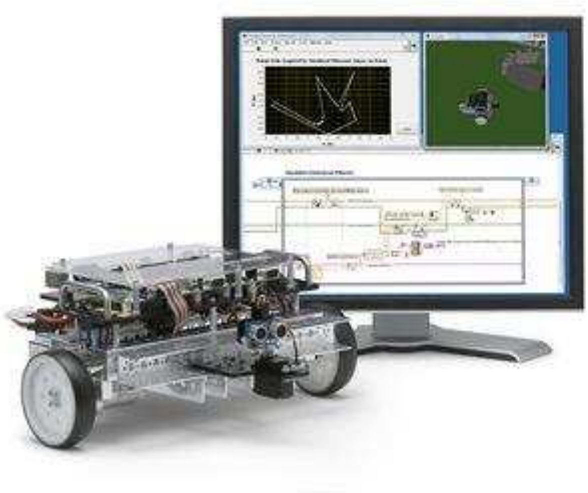


Figure 3 NI LabVIEW Robotics Starter Kit 1.0 (DaNI) for Education

The Starter Kit 1.0 is a 4-wheeled autonomous mobile robot, which is designed, manufactured and sold by National Instruments. The robot comes with multiple sensors, motors and NI Single-Board RIO-9631 for embedded control. The robot can be programmed and controlled by using the LabVIEW Robotics Module, which is included in the Kit. The Module comes with a ready vector field histogram (VFH) obstacle avoidance program which makes the robot roam while avoiding obstacles based on the readings from the ultrasonic sensor included in the Kit.

The Starter Kit includes the following sensors and actuators[14]:

Name	Specifications
Pitsco Education DC Motors	Supply voltage 12V, Speed 152 rpm, Torque 2.12 N.m.
Optical Encoders	Supply voltage5V, Cycles per revolution 100 CPR, Pulses per revolution 400 PPR
PING))) Ultrasonic sensor	Supply voltage 5V, Range 2cm to 3m, Burst frequency 40 kHz for 200 µs
PING))) Mounting bracket for Ultrasonic sensor	Sweeping angle 180 degrees
Pitsco Education TETRIX wheels	Four 101.6 mm wheels

Table 3 Sensors and Actuators Included in NI LabVIEW Starter Kit 1.0

It should be noted that the four-wheel design of the robot was changed to three wheels (2 TETRIX wheels and one omnidirectional wheel) to increase the manoeuvrability of the robot.

3.1.1 NI sbRIO-9631[15]

NI Single-Board RIO 9631 features a real-time processor, a user-reconfigurable field-programmable gate array (FPGA), and I/O all incorporated on one printed circuit board (PCB). The sbRIO-9631 is designed to be easily embedded in high-volume applications that require flexibility, reliability, and high performance.

The NI sbRIO-9631 includes the followings:

Parts Name	Specifications
Controllers	Integrated real-time controller, reconfigurable FPGA, and I/O on a single board
FPGA	1M gate Xilinx Spartan FPGA
Processor	266 MHz
Memory	64 MB of DRAM, 128 MB non-volatile memory
Serial Ports	3 RS232 serial ports
Digital I/O Lines	110 3.3 V (5 V tolerant/TTL compatible)
Input Channels	32 single-ended/16 differential 16-bit analog channels at 250 kS/s
Output Channels	Four 16-bit analog channels at 100 kS/s
Network	10/100 Mbits/s Ethernet port
Power Supply Input Range	19 to 30 VDC

Table 4 NI sbRIO-9631 Components

3.1.2 LabVIEW Overview

The embedded sbRIO-9631 can be programmed by the LabVIEW graphical development environment[15]. The LabVIEW Real-Time Module is run on the Wind River VxWorks real-time operating system (RTOS) by the real time processor. Both graphical and textual syntax can be combined by deploying .m files to NI real-time hardware through the LabVIEW MathScript RT Module.

High-speed control and inline signal processing can be achieved by programming the reconfigurable FPGA through the LabVIEW FPGA Module. The data transfer between the FPGA and real time processor is controlled by the drivers and APIs, which are included in LabVIEW. A comprehensive robotics library is offered by the LabVIEW Robotics Module, which requires the LabVIEW development environment to run. The library includes:

- Connectivity to robotic sensors
- Foundational algorithms for intelligent operation and robust perception
- Built-in physics-based environment simulator
- Motion functions for making the robot or vehicle move
- Real-world application examples
- Forward and inverse kinematics
- Libraries for protocols including I^2C, SPI, PWM, and JAUS

A variety of robots, from educational robots to sophisticated autonomous systems can be developed by the tools available from the LabVIEW Robotics Module. A robot executing multiple complex subsystems in parallel can be easily programmed by the templates or the software architectures offered by this software.

3.1.3 Platform Specifications

Size	405 mm x 368 mm x 150 mm
Mass	3.6 kg
Battery charge time	1.7 hours
Battery charge time (with the motors turned on)*	1 hour
Battery charge time (with the motors turned off)	4 hours

Table 5 NI LabVIEW Platform Specifications

* Assuming that the robot is running the start-up application (obstacle avoidance) on a fully charged battery.

3.1.4 Ultrasonic Sensor Specifications

Short ultrasonic bursts are emitted and then listened for the echo by the Parallax PING))) ultrasonic sensor to detect objects. A short 40 KHz (ultrasonic) burst is emitted under the control of a host microcontroller. The emitted burst travels at 344.4 meters per second and when it hits an object, it bounces back to the sensor. When the echo is detected, the PING))) sensor sends an output pulse to the host to be terminated; hence, the distance to the target corresponds to the width of the pulse[15].

3.1.5 Block Diagram

The figures 4 & 5 below show the block diagram of the LabVIEW Starter Kit and a schematic of the sbRIO-9631 board, respectively:

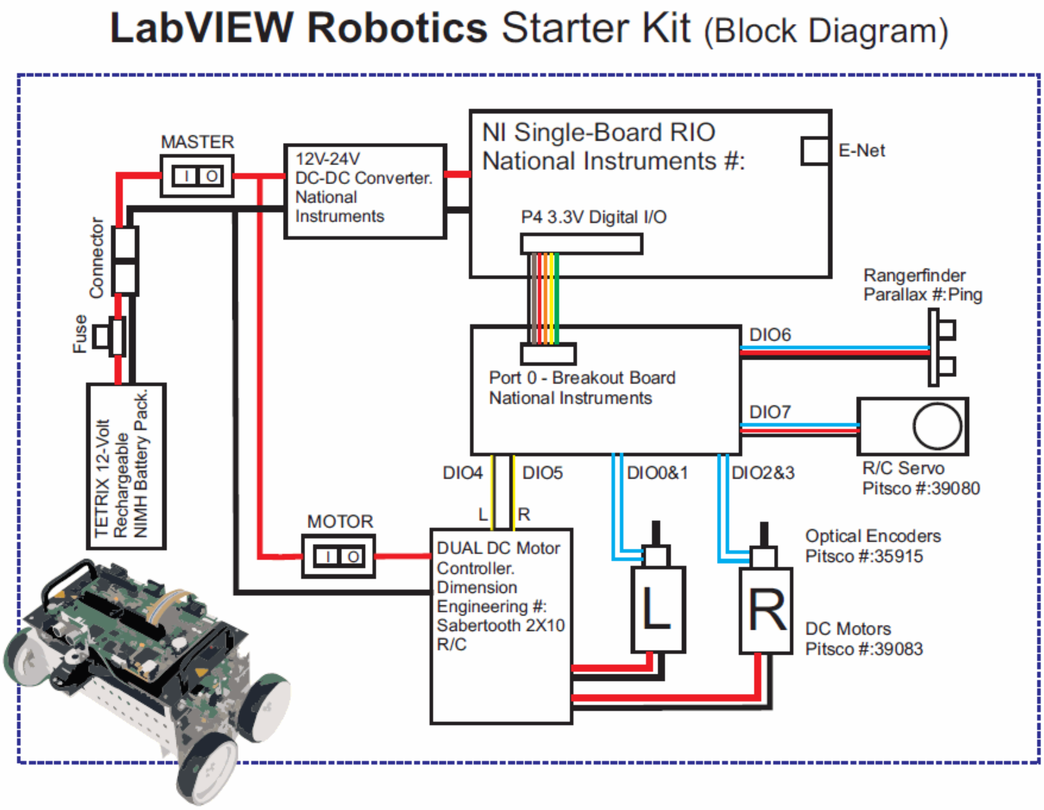


Figure 4 NI LabVIEW Robotics Starter Kit 1.0[15]

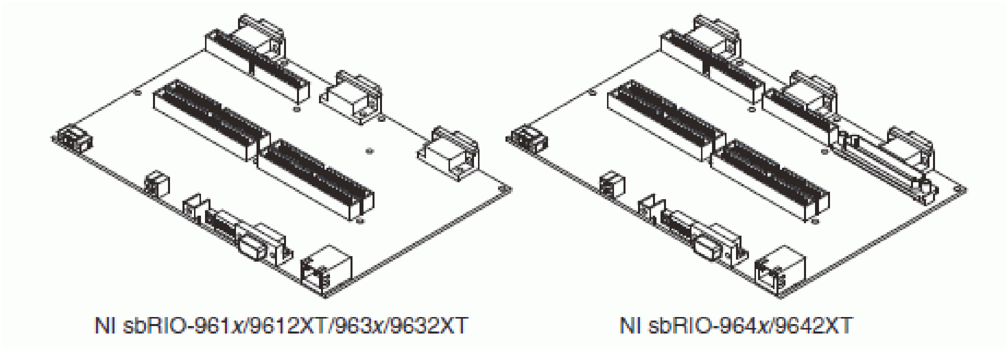


Figure 5 A schematic of NI sbRIO-961x/9612XT/963x/9632XT and NI sbRIO-964x/9642XT [15]

3.2 Microsoft Kinect

In November 2010, Microsoft launched Kinect, a motion sensing input device for the Xbox 360 gaming console, which makes control and interaction with Xbox 360 possible without using a game controller but by using hand gestures and spoken commands. Later on June 2011, the Kinect for Windows Software Development Kit (SDK) was released to allow developers to write Kinecting applications in C++, C# or Visual Basic. After that, a Windows version of Kinect was released in February 2012.

The Kinect sensor is composed of the followings[16]:

- An RGB camera with a 640 x 480-pixel resolution which captures colour images at 30 FPS (frames per second)
- An infrared (IR) emitter and an IR depth sensor to measure the distance between an object and the sensor by converting the reflected beams into depth information. The device can capture depth images with a 640 x 480-pixel resolution at 30 FPS.
- An array of four microphones that make it possible to record audio, determine the location of the sound source and the audio wave's direction.
- A 3-axis accelerometer to determine the orientation of the Kinect

The figure 6 below shows various components of Microsoft Kinect:

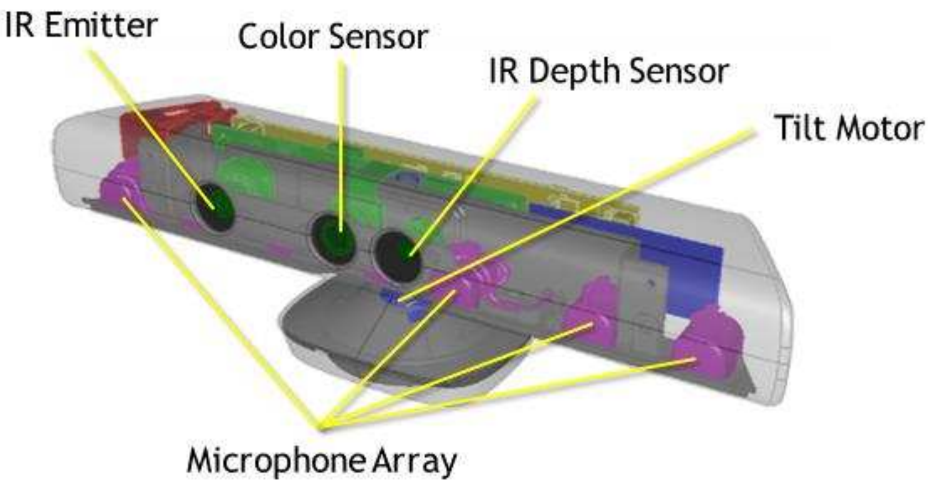


Figure 6 Microsoft Kinect and its Peripherals[16]

3.2.1 Specifications of Microsoft Kinect

Below is table 6 showing the specifications of Microsoft Kinect as stated by Microsoft[16]:

Characteristics	**Specifications**
Viewing angle	43° vertical by 57° horizontal field of view
Vertical tilt range	±27°
Frame rate (depth and color stream)	30 frames per second (FPS)
Audio format	16-kHz, 24-bit mono pulse code modulation (PCM)
Audio input characteristics	A four-microphone array with 24-bit analog-to-digital converter (ADC) and Kinect-resident signal processing including acoustic echo cancellation and noise suppression
Accelerometer characteristics	A 2G/4G/8G accelerometer configured for the 2G range, with a 1° accuracy upper limit.

Table 6 Specifications of Microsoft Kinect

3.2.2 Hardware Requirements of Microsoft Kinect

The minimum and recommended hardware requirements to develop applications using the Kinect for Windows SDK as stated by Microsoft is as shown table 7 below:

	Minimum[17]	Recommended[18]
Processor Type	Dual-core	Core i5
Clock Speed	2.66 GHz	3.0 GHz
Memory	2 GB	4 GB DDR3
Operating System	32-bit or 64-bit Windows 7	64-bit Windows 7 or Windows 8

Table 7 Minimum and Recommended Hardware requirements for using Microsoft Kinect

The recommended hardware configuration is for applications designed to make intensive use of Kinect Skeletal Tracking, which should have been used for this project. However, due to unavailability of such laptop, a laptop with much less processing power was used as shown in Table 3. The laptop's performance was low as to be expected but proved to be sufficient for a slow moving indoor autonomous robot.

Since DaNI robot has a processing power of only 266 MHz, much less than the minimum clock speed of 2.66 GHz, this makes it almost impossible to run Kinect on DaNI robot.

A solution to this problem was to run and process data from Kinect on a laptop, then to transfer the acquired data to DaNI robot using Ethernet cable.

The hardware specifications of the laptop used in this project are as shown in the table 8 below:

	Laptop Used in this Project
Processor Type	Core2Duo
Clock Speed	2.1 GHz
Memory	4 GB DDR3
Operating System	32-bit Windows 7

Table 8 Hardware Specifications of the Laptop used in this Project

3.2.3 Microsoft Kinect SDK and OpenNI

After the hardware was chosen, work was done to see how Microsoft Kinect can be used in LabVIEW. Since the robot is be programmed in LabVIEW and it must be interfaced with Microsoft Kinect, it was necessary to run and process the data from Kinect in LabVIEW in order to make transmission of data between Kinect and DaNI as smooth as possible.

After doing some research it was found that before Microsoft released a driver of Kinect for Windows, OpenNI[19] released an unofficial driver for Windows platforms, which Microsoft called it hacking but then it was stated by Microsoft that they are consent with this, and later on Microsoft itself released a driver along a software development kit (SDK) [20]. With these two tools available, one had to be chosen by looking at their capabilities. Below is a comparison table between these tools in terms of their algorithmic capabilities[21]:

	OpenNI	Microsoft SDK
Operating Platform	Multi-platform	Windows 7 onwards
Initialization Requires a Certain Pose	Yes	No
No. of Joints Tracking Capability	15	20
Can Track while Seated	No	Yes
Body Gesture recognition	Yes	Yes
Hand gesture Analysis	Yes	Yes
3D Depth Practical Ranging Limit	0.8m – 3.5m	0.8m – 3.5m

Table 9 Comparison between OpenNI and Microsoft SDK

After the comparison, it was clear that Microsoft SDK was superior to OpenNI, especially in terms of Initialization since Microsoft SDK does not require a certain pose in order to recognize humans, which in this work was practically proved to be correct, making it very practical in real time operation.

3.3 Microsoft Kinect SDK

A comprehensive software library and tools are offered in the newest version (1.7) of Microsoft Kinect SDK to provide the developers access to all of the features of Kinect to its fullest extent. The interaction between Kinect along with the software library and a user application is shown in figure 7 below:

.

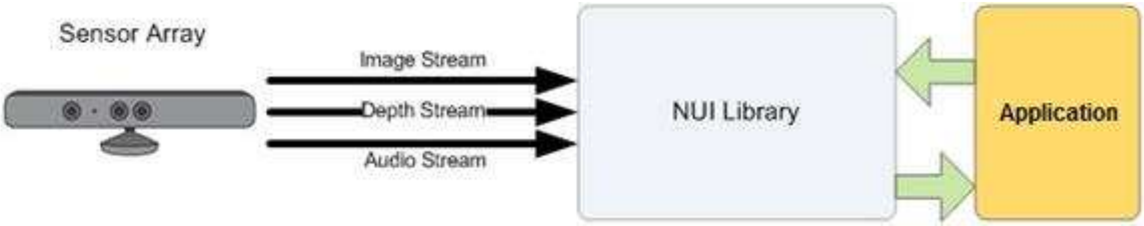


Figure 7 Hardware and Software Interaction of Microsoft Kinect with an Application

The Microsoft Kinect SDK architecture and its components are shown in figure 8 below:

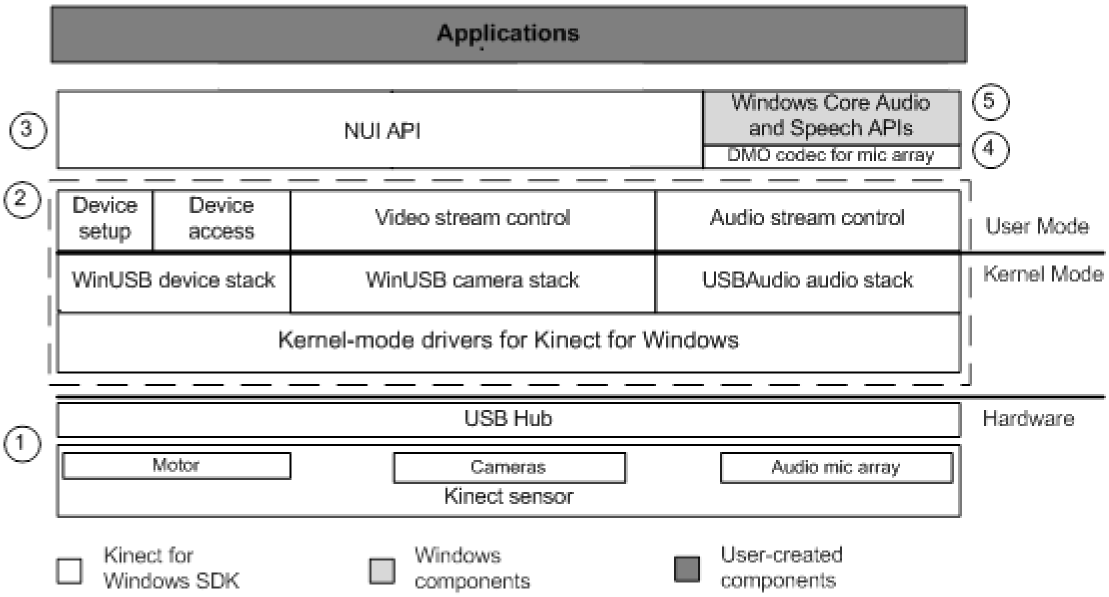


Figure 8 Microsoft Kinect SDK Architecture

The system is composed of the following:

1. Microsoft Kinect Hardware.
2. Windows drivers for Microsoft Kinect, which are a part of the SDK software.
3. Video and Audio components: Kinect natural user interface for skeleton tracking, audio, and colour and depth imaging.
4. Microphone array beamforming and audio source localization by using DirectX Media Object (DMO).
5. The audio, speech, and media APIs of Windows 7

3.3.1 Setting up a Kinect Sensor[22]

The setup of Microsoft Kinect and its operation conditions as stated by Microsoft are as follows:

1. The sensor must be mounted on a stable surface and not placed in front of or on a speaker or a vibrating surface as the vibration disrupts the data received by the Kinect sensor.
2. The Kinect must not be placed in direct sunlight.
3. The operating temperature range is from 5 to 35 degrees Celsius.
4. Microsoft Kinect must not be manually tilted on its base as it is controlled by software otherwise the Kinect will be damaged.

The following software must be installed on a computer before connecting the Kinect as the drivers are not available readily in Microsoft Windows:

1. Latest Kinect for Windows SDK.
2. Latest Kinect for Windows Developer Toolkit

After the software installation is finished the Kinect can be connected to the computer as follows:

1. One end (adapter) of the power supply cord of Kinect must be connected to an external power source.
2. The other (USB) end of the supply cord must be connected to the USB port on the computer.

Microsoft provides the following instructions for the operating conditions and the environment in which the Kinect should be used:

1. Both ends of the power cord must be connected for the Kinect to function otherwise only the LED would be on if only one end of the power cord is connected and the Kinect will not function.
2. No calibration is required for both video and audio.
3. The only device plugged in to a USB hub on a computer should be Kinect. If more than one Kinect is intended to be connected to a computer, they must be connected to different USB controllers.
4. If the temperature reaches 90 degrees Celsius, the sensor's firmware will turn off the camera. The sensor is cooled by a fan.
5. The lightning conditions should be neither very dark nor very bright for the RGB camera to capture images properly. The device can capture colour images without any considerable disruptions under Incandescent, Fluorescent and natural lightning conditions. However, the RGB sensor can be blinded if a constant or intense source of light is directed to the camera.
6. In typical or reduced lightning conditions the depth sensor functions adequately. However, noise increases in near darkness conditions.
7. The depth sensor might not identify highly reflective (shiny metal) or highly absorptive (dark materials) objects correctly.

3.3.2 Coordinate Spaces

The coordinate spaces for each of colour, depth and skeleton data types that are streamed out by Kinect one frame at a time are described in this section[23].

3.3.2.1 Colour Space

The colour sensor at each frame captures a colour image of everything visible in the field of view of the sensor. The number of pixels a frame is composed of depends on the frame size, which can be specified by software. The red, green and blue value of a particular pixel at a specific (x, y) coordinate in the colour image is contained in each pixel.

3.3.2.2 Depth Space

The depth sensor at each frame captures a grayscale image of everything visible in the field of view of the sensor. The number of pixels a frame is composed of depends on the frame size, which can be specified by software. The Cartesian distance (in millimetres), from the camera plane to the nearest object at that specific (x, y) coordinate as shown in figure 9, is contained in each pixel. The location of a pixel in the depth frame is represented by the (x, y) coordinates of a depth frame. They do not represent the physical units in the room.

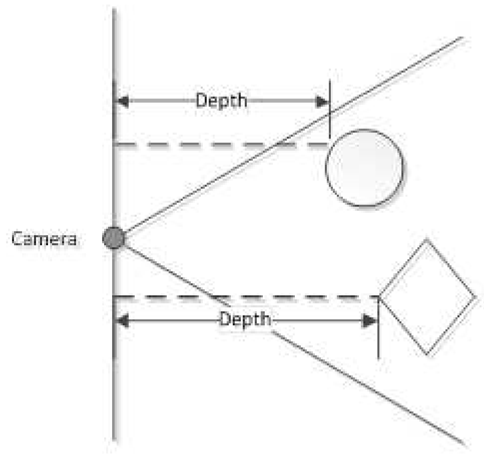


Figure 9 Cartesian distance between an object and depth camera in Depth Space[23]

3.3.2.3 Depth Space Range

There are two depth ranges available from the depth sensor: the default range and the near range. The Kinect for Windows sensor has both default and near ranges while the Kinect for Xbox 360 has only the default range. The sensor depth ranges in meters are illustrated in the figure 10 below:

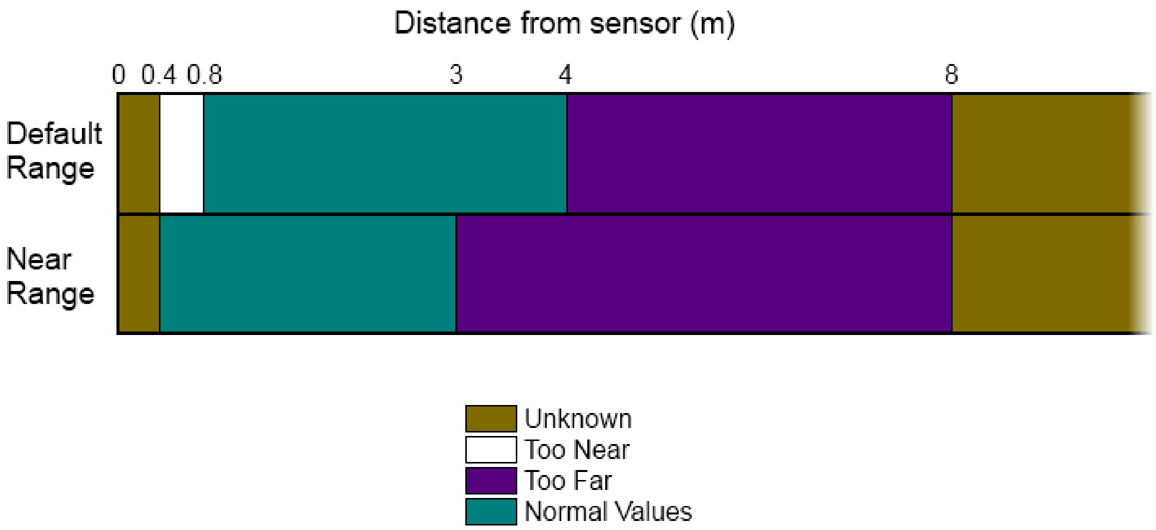


Figure 10 The Operational Range of Depth Sensor[23]

- The “Unknown” represents no object is detected.
- The “Too Near” represents an object was detected, but no reliable distance measurement can be provided as it is too near to the sensor.
- The “Too Far” represents an object was detected, but no reliable measurement can be provided as it is too far from the sensor.
- The “Normal Values” represent an object was detected and reliable distance measurement can be provided.

3.3.2.4 Skeleton Space

The Kinect runtime processes the capture depth image into skeleton data with each frame. The 3D position data for human skeletons are contained in the skeleton data for up to 2 persons visible in the field of view of the depth sensor. The (x, y, z) coordinates of skeleton space which are expressed in meters represent the position of a skeleton and each of the skeleton joints.

The figure 11 below shows the x, y and z-axes of the body in the depth sensor:

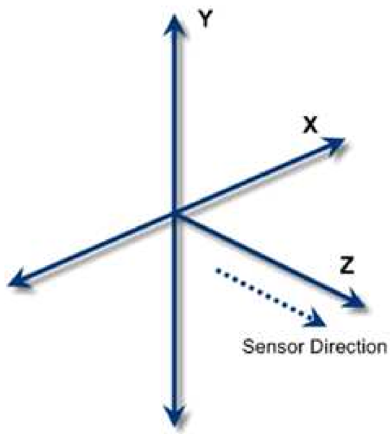


Figure 11 X-Y-Z Coordinate-axis of a Skeleton in Depth sensor[23]

The Kinect is placed at the origin of the coordinate system, with the direction Kinect facing being the positive z-axis. The positive x-axis extends to the left and the positive y-axis extends upward. When the sensor is placed on a surface that is not level or is tilted to optimize the field of view of the sensor, the generated skeletons can appear to lean instead of standing upright.

3.3.3 Skeletal Tracking

Skeletal tracking[24] is one of the most important features provided by Microsoft Kinect. By using, the infrared (IR) camera up to 6 persons in the field of view of the camera can be detected. Furthermore, up to 2 from the 6 detected persons can be tracked in detail. The joints of the tracked users can be located in space and their movements be tracked by an application. This can be illustrated in the figure 12 below:

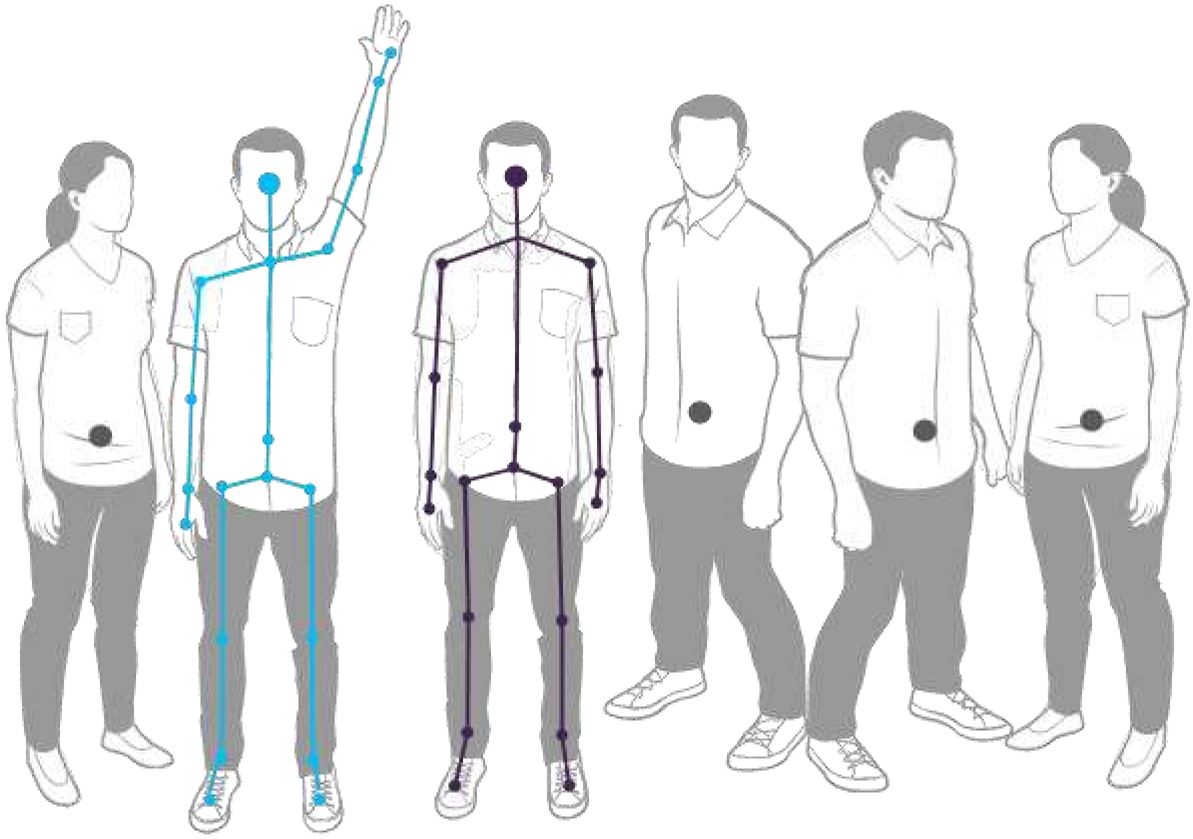


Figure 12 Two persons out of six recognized persons can be tracked at the same time[24]

Users can be detected either while standing or sitting and facing the Kinect. However sideways poses could prove to be difficult to be detected because of the parts of the user that are not visible to the camera. No calibration action or specific pose is required for the user to be recognized and tracked. The user only needs to be in front of the sensor and make sure their head and upper body can be seen by the sensor to be recognized. This is illustrated in the figure 13 below:

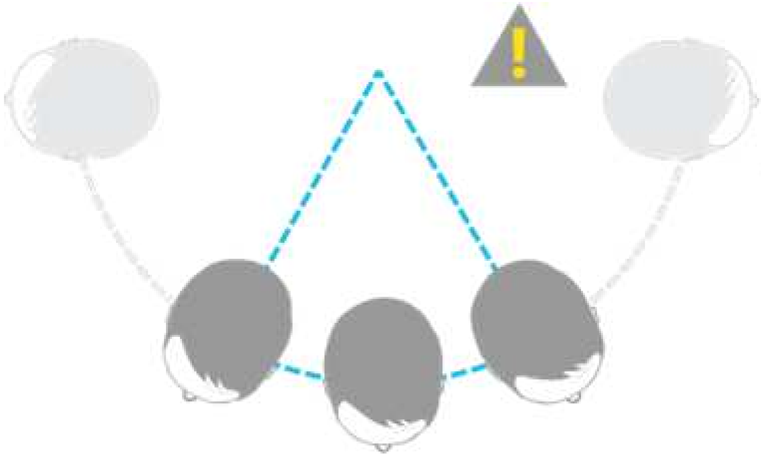


Figure 13 Skeleton tracking is designed to recognize users facing the sensor[24]

3.3.4 Field of View

The settings of the infrared (IR) camera determines the Kinect's field of view of the users. Kinect can see people from 0.8 meters to 4.0 meters away from the sensor while standing in default mode, however users will not be able to use their hands at these distances, therefore a practical range of 1.2 to 3.5 meters is suggested. This is illustrated in the figures 14 & 15 below:

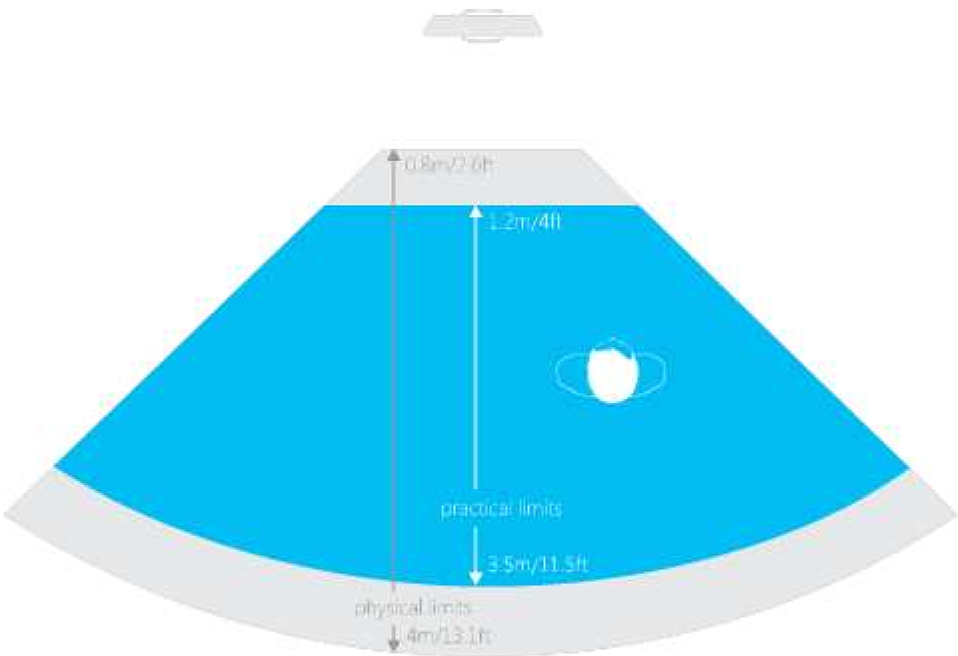


Figure 14 Horizontal Field of View of Microsoft Kinect in default range[24]

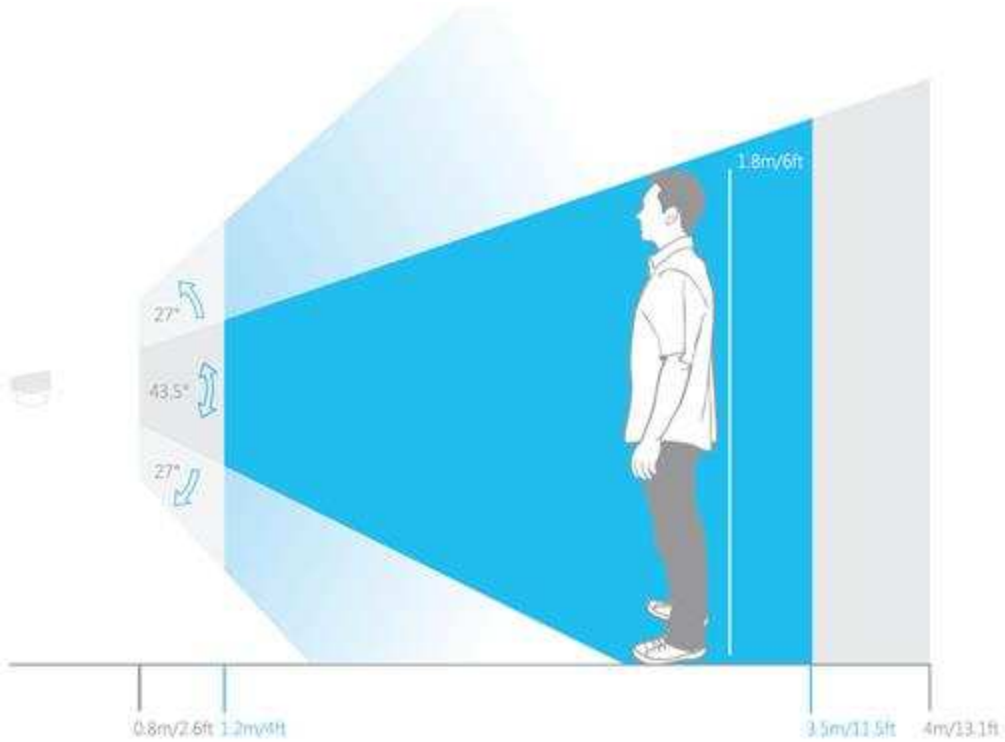


Figure 15 Vertical Field of View of Microsoft Kinect in default range[24]

While Kinect can see people from 0.4 meters to 3.0 meters while standing in near range mode. However again for the users to be able to use their hands, a practical range of 0.8 to 2.5 meters is suggested. This is illustrated in the figure 16 below:

.

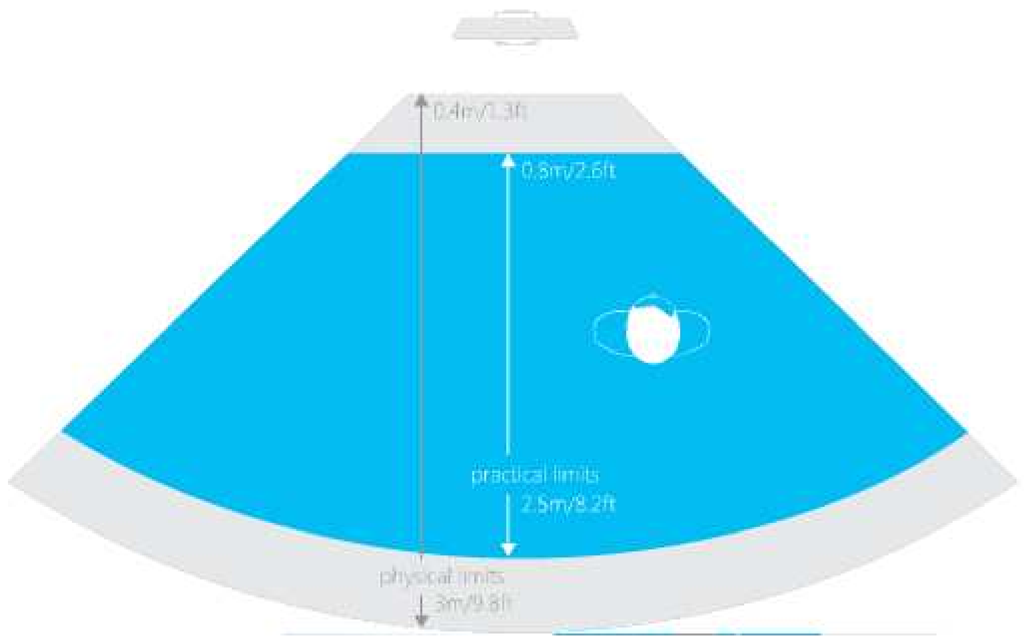


Figure 16 Horizontal Field of View of Microsoft Kinect in near range[24]

3.3.5 Skeletal Tracking Precision and Multiple Kinect Sensors

For different people and different body parts to be recognized in the field of view, the depth of the people is calculated by using a pattern of infrared light, which is projected by Kinect's infrared emitter. The precision and accuracy of skeletal tracking may reduce when more than one Kinect is used to illuminate the target area due to interference of the infrared light sources. Hence it is recommended that no more than one infrared light source (or Kinect sensor) should be directed to a field of view where skeletal tracking is done.

3.3.6 Skeleton Position and Tracking State

There are 2 tracking states available for a skeleton in a frame, one state being "tracked" which offers detailed information about the position of 20 joints of a user's body in the field of view of the camera as shown in figure 17. The other state being "position only" which provides information about the position of the user only without any information about the position of the joints[25].

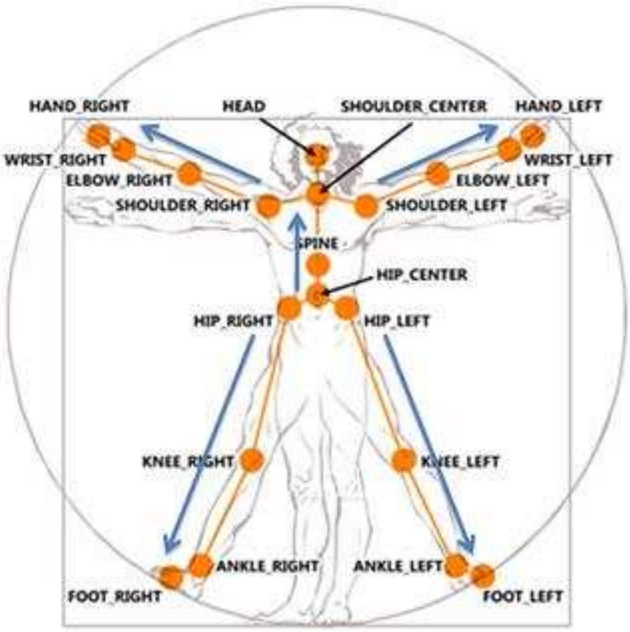


Figure 17 Joints tracked by Microsoft Kinect[25]

3.3.7 Active User Tracking

Skeleton to be tracked from the 6 recognized users can be chosen by an application. In the default settings, the application is set to track the first 2 recognized users, but this can be overridden and a custom logic be implemented to choose users to track. For instance an application can choose to track the user who is raising his/her hands or the user who is farthest. An application can achieve this by cycling through all the recognized skeletons in the field of view and then choose the one that matches the desired criteria. The matched skeleton's tracking ID is then sent to the skeletal tracking API for full tracking. [26]

The skeletal tracking system is set not to take control back once the users to track has been taken control over by an application. Therefore, the application must choose which new user to track, if the tracked user is goes out of the field of view of the camera. It should be noted that if a user goes out of the field of view of the camera and then comes back, a new random tracking ID will be given to the user, which is not related to the previous ID the user had. An application can choose to track 1 or 2 users by sending the corresponding tracking ID's to the skeletal tracking APIs or it can choose to not track any user by sending a tracking ID of 0.

3.3.8 Player ID in depth map

The location of a user in the field of view of Kinect is specified in the depth map when the skeletal tracking identifies the user. The players' index information are contained in the three lowest order bits of every pixel in the depth map. A pixel of "1" to "6" designates the existence of a user at that particular pixel, while an index of "0" depicts no user is present. By subtracting "1" from the player index, and entering the resulting value into the array of skeletons, it is possible to look at the skeleton of a user at a specific depth pixel.[26]

3.4 Kinesthesia

In 2011-2012, a group of mechanical engineering students from the University of Leeds developed a toolkit for LabVIEW named Kinesthesia to allow access to RGB video, depth camera and Skeletal tracking functions of Microsoft Kinect, which was initially developed for medical rehabilitation and surgical tools.

Kinesthesia toolkit is freely available to download through the JKI's VI Packet Manager to be used in LabVIEW. The toolkit comes with an example to show how it works in LabVIEW as shown in the figure 18 below:

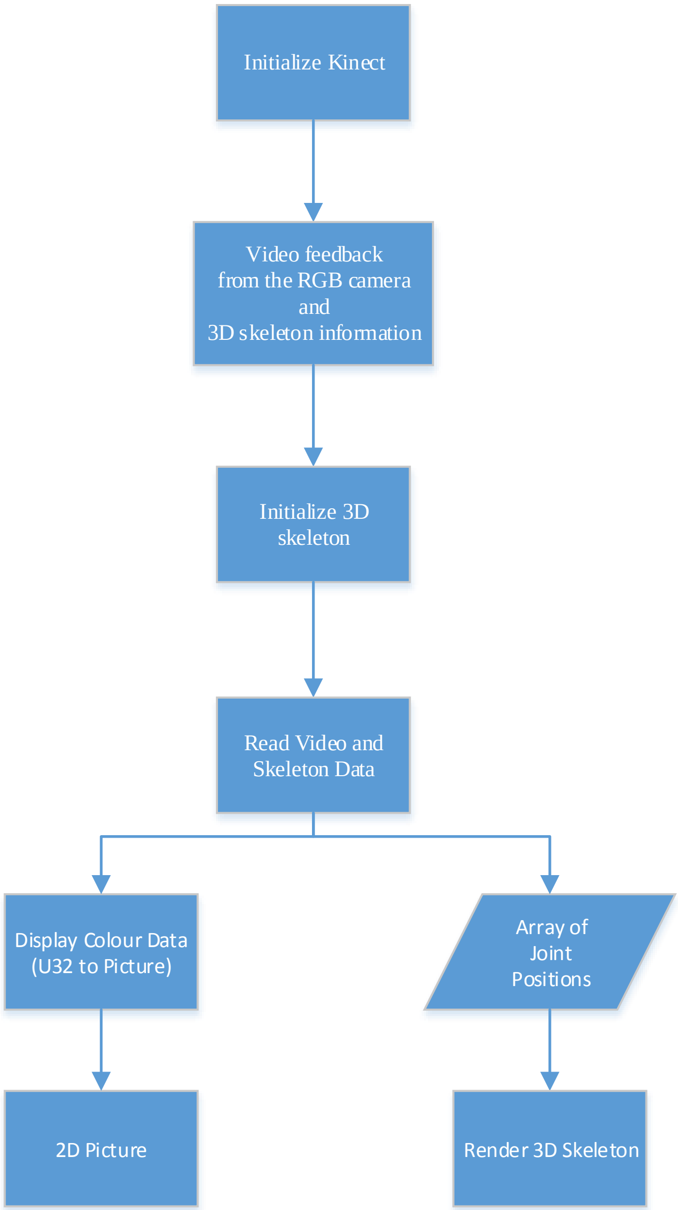


Figure 18 Kinesthesia Example

The Kinesthesia toolkit is designed for LabVIEW and requires the following to function:

- Microsoft .NET 4.0 Framework
- Microsoft Kinect SDK 1.0 or newer Versions
- NET 4.0 Assemblies LabVIEW Hotfix

The main polymorphic VIs, which provide access to the colour and depth videos as well as the skeletal data, are as follows:

- Initialise VI: An instance of Kinect is created in the computer's memory to Initialize it.
- Configure VI: Provides the option for a user to choose which data (video, depth or skeleton) to be streamed from Kinect. The user can choose to stream one, two or all three of them together.
- Read VI: The data streams from Kinect are processed in this VI to be displayed or ready it for further processing.
- Close VI: All the references which were created during the operation of Kinect are closed by this VI.

The main VIs in LabVIEW are as shown in figure 19 below:

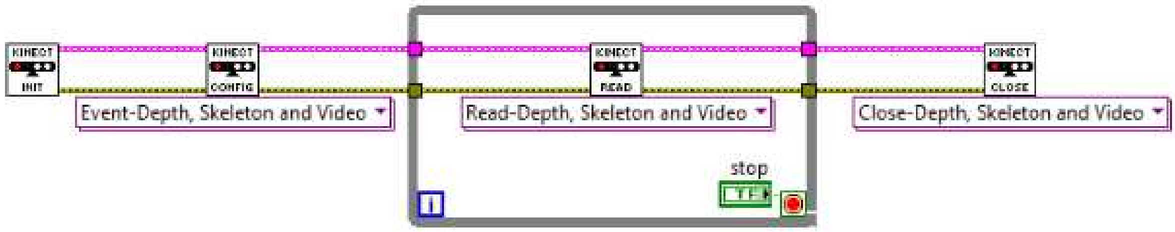


Figure 19 Main Polymorphic VIs of Kinesthesia[27]

In addition to the main VIs a number of sub-VIs are also provided in this toolkit, which are as follows:

- Display Colour Data (U32 to picture) VI: The raw U32 array is converted to a 2D picture format to be displayed in LabVIEW.
- Display Depth Data (U16 to picture) VI: The raw U16 array is converted to a 2D picture format to be displayed in LabVIEW.
- Initialise 3D Skeleton: The scene for skeleton plotting is made available by this VI for 3D skeleton rendering.
- Render 3D Skeleton: The output of "Initialise 3D Skeleton" and the "Array of Joints positions" from Kinect is fed to this VI to produce visual feedback for the joint data.
- Joint Coordinates VI: Returns The x, y and z coordinates of a selected joint
- Displacement and Distance between Joints: Returns the distance and absolute displacement between two selected joint or between a joint and a custom coordinate system.
- Angle between Joints: The angle between three connected joints or between a custom vector and a vector created by two joints is calculated in this VI.

While originally Kinect can recognise 6 people and track 2 at the same time as described before, Kinesthesia toolkit provides the tracking of only 1 person and that is the first person it recognizes in its field of view. Once it recognizes and tracks a person, it will not recognize anybody else entering the field of view of camera. This limits the capability of Kinect greatly especially for this project, but no other toolkit is available to use Microsoft Kinect in LabVIEW.

3.5 The Platform

A platform was designed to be fixed on DaNI, which enables the positioning and fixing of both the laptop and the Kinect to DaNI robot. The design of the platform was made as simple as possible for ease of mounting and dismounting of laptop and Kinect. The platform is made of Aluminium to be as light as possible for decreasing the load on the robot motors' and yet strong enough to carry both the laptop and Kinect.

The platform is made of two Aluminium sheets, the connection both between the sheets and between the lower sheet and DaNI robot is done by using hexagonal columns to be as firm as possible and not to vibrate during operation.

The 2D CAD designs of the platform is provided in Appendix A.

The Kinect is fixed to the upper sheet by two brackets, and are fixed to the upper sheet by screws as shown in the figures 20 & 21 below:

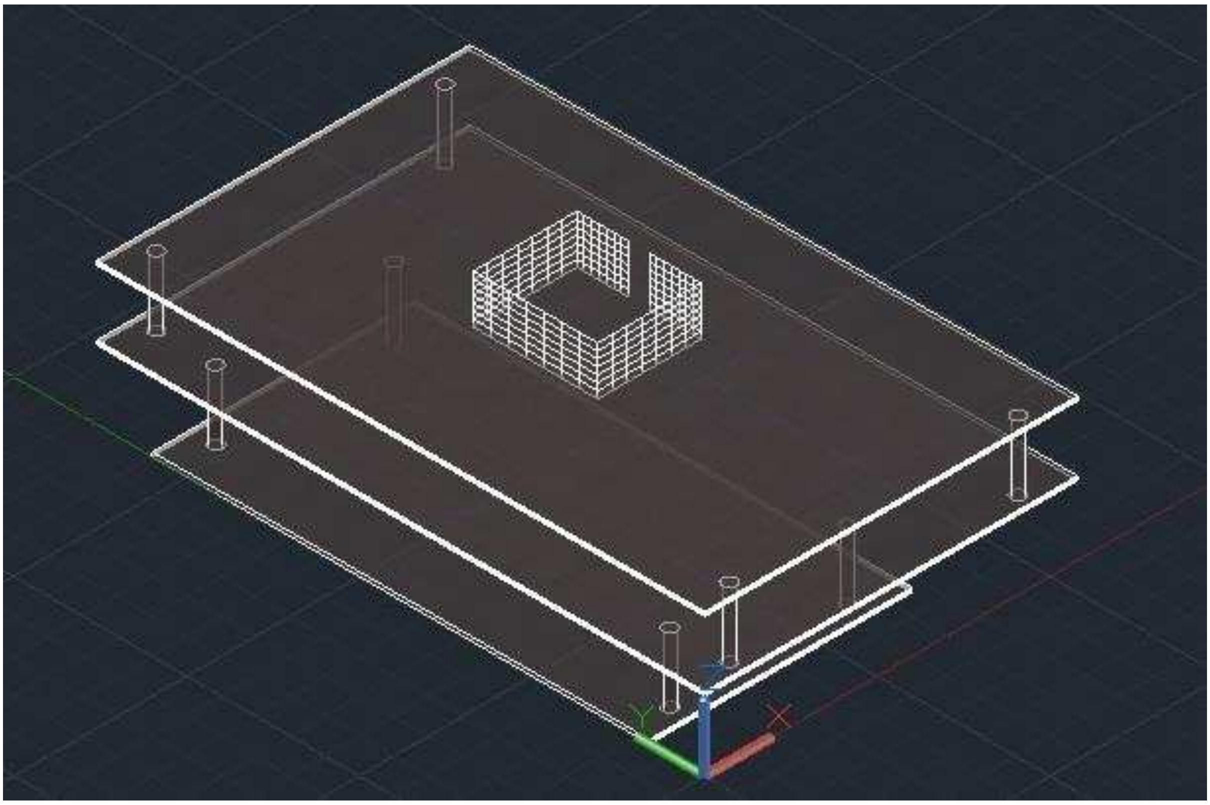


Figure 20 3D Design of the Platform to be Fixed on DaNI Robot

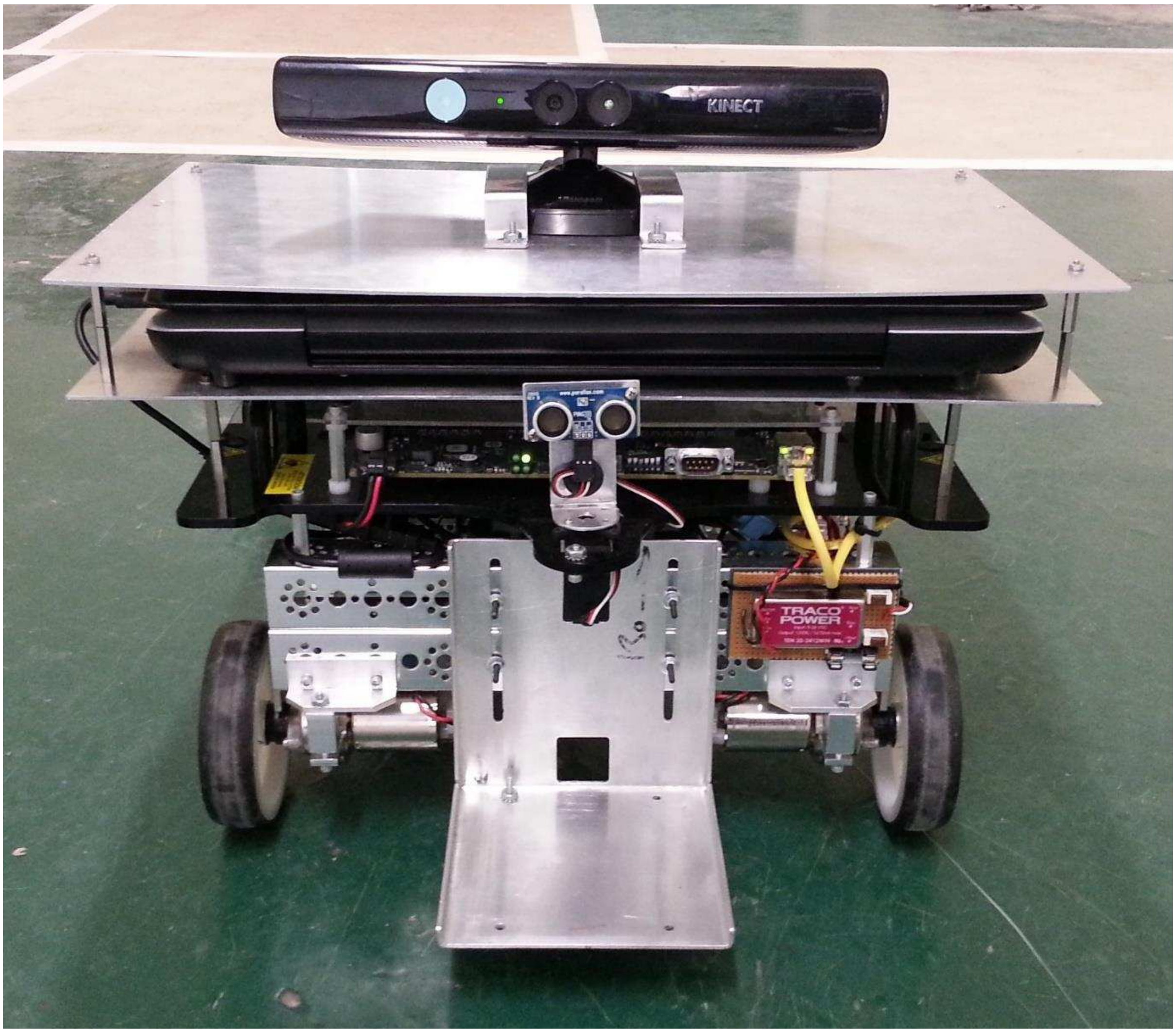


Figure 21 Final Assembly of NI LabVIEW Starter Kit 1.0 with Microsoft Kinect

3.6 Hardware Connection

Microsoft Kinect is powered by two sources; one from mains (12 volts DC) and one from a USB (5 volts DC) port. Since the Kinect will be fixed on the robot, the Kinect was powered from the robot's battery, which supplies the whole robot, and the USB port was connected to the laptop fixed on the robot. The robot was connected and controlled by the laptop via an Ethernet cable. The figure 22 below illustrates the hardware connection:

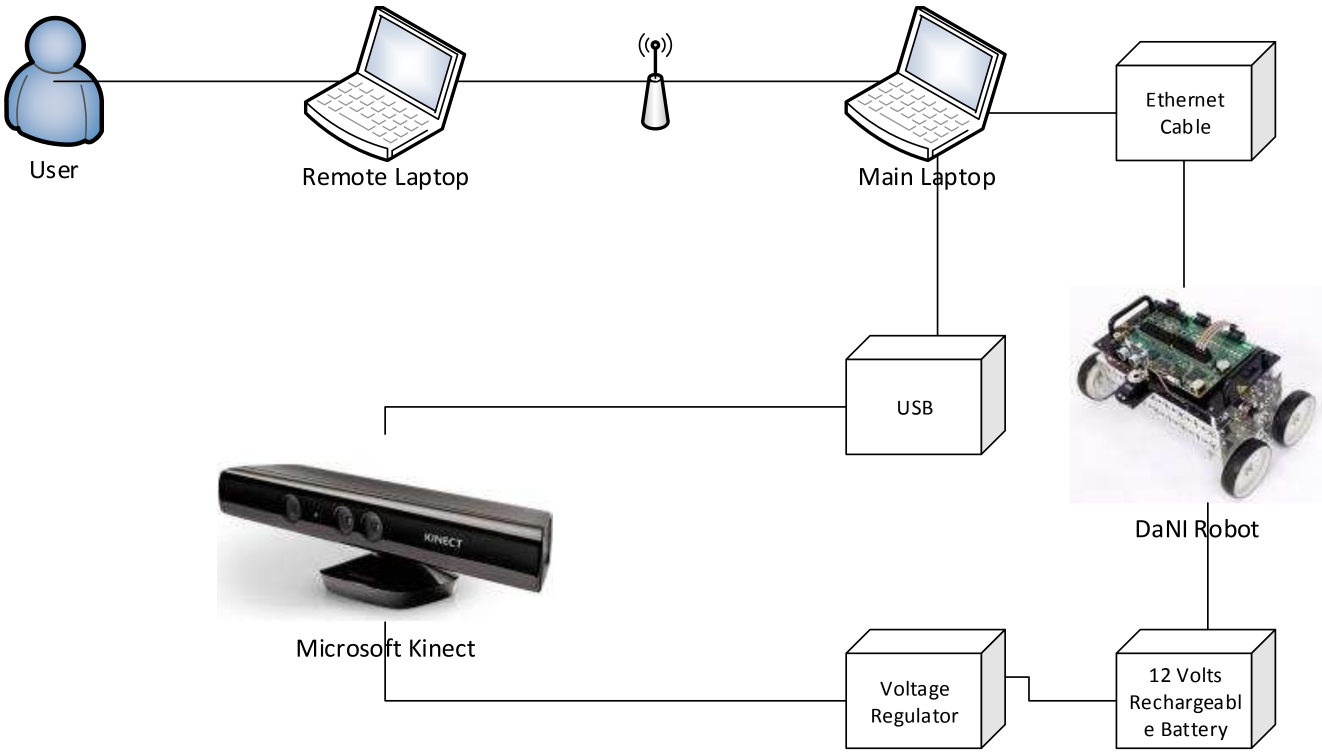


Figure 22 Hardware Connection of the Overall Project

4 Methodology and Analysis

This chapter describes the human recognition, Identification and tracking methodology developed for this project, which is then followed by a practical analysis to determine the limitations of the proposed method.

4.1 Overall Algorithm (Methodology)

Initially the person to be tracked must move to in front of the robot, facing the robot at a distance of 2 meters, then the Kinect will start recognizing the human and will start to take data to Identify the human, the overall methodology can be divided into following sequences:

1. Robot starts to recognize and save data of the first person coming into field of view of Kinect.
2. The person to be identified must move forwards and backwards from 2 m to 3.5 m at least twice for the skeletal data to be saved and be used to identify the target when robot starts roaming.
3. Then the person must move at least 3.6 m away from Kinect for the robot to start roaming.
4. The robot starts roaming randomly and search for the target using Microsoft Kinect while avoiding any obstacles found on the way.
5. When any person comes into the field of view of Kinect, the target's skeletal data is compared with the first identified person's skeletal data and then the algorithm decides whether the person in the field of view is the target or not.
6. If the person is identified as the target then the robot starts tracking him/her while avoiding any obstacles and non-target humans and will keep a minimum distance of 2.2 m to the target.
7. If the person is not identified as the target then the robot will identify it as an obstacle and continue its search while avoiding any obstacles and humans, which are not the target until the target human is found.
8. Both while roaming and tracking, the robot's location and the target human's location will be displayed on an XY-graph in LabView's front panel.

All the VI's developed and used for achieving the objectives of this project are provided in Appendix B. In addition, Each VI is illustrated in detail using flow charts in the following sections in this chapter.

4.2 Human Identification Methodology

Identification of a human as a target to be found and tracked is the most important aspect of this project and thus the VI's involved in this process. The Identification algorithm proposed relies on the skeletal tracking ability of Microsoft Kinect, which over recent years proved to be reasonably accurate for a low cost range camera. One of the most important VI's Kinesthesia toolkit provides is the "Distance and Displacement between Joints" VI which forms the backbone of the Identification algorithm proposed. By using this VI it is possible to obtain the absolute distance between 2 adjacent joints, hence absolute distance between all 20 joints can be acquired by using 19 "Distance and Displacement between Joints" Vis starting from head until feet.

The operation principle of "Distance and Displacement between Joints" VI and the steps involved in the Identification process are described in the following sections.

4.2.1 Operation Principle of Distance and Displacement between Joints VI

This VI is provided in the Kinesthesia toolkit and was used without any modifications. Initially 2 adjacent Joints are selected via "Index Array" function to obtain the X, Y and Z coordinates of each Joint from the "Array of Joint Positions VI". Then by using the "Array Subset" function only X and Y coordinates are extracted. The extracted sub arrays of Joints 1 and Joint 2 are subtracted from each other, squared, summed and finally the square root is taken to get the absolute distance between the 2 joints. The overall operation of this VI can be seen in the figure 23 below:

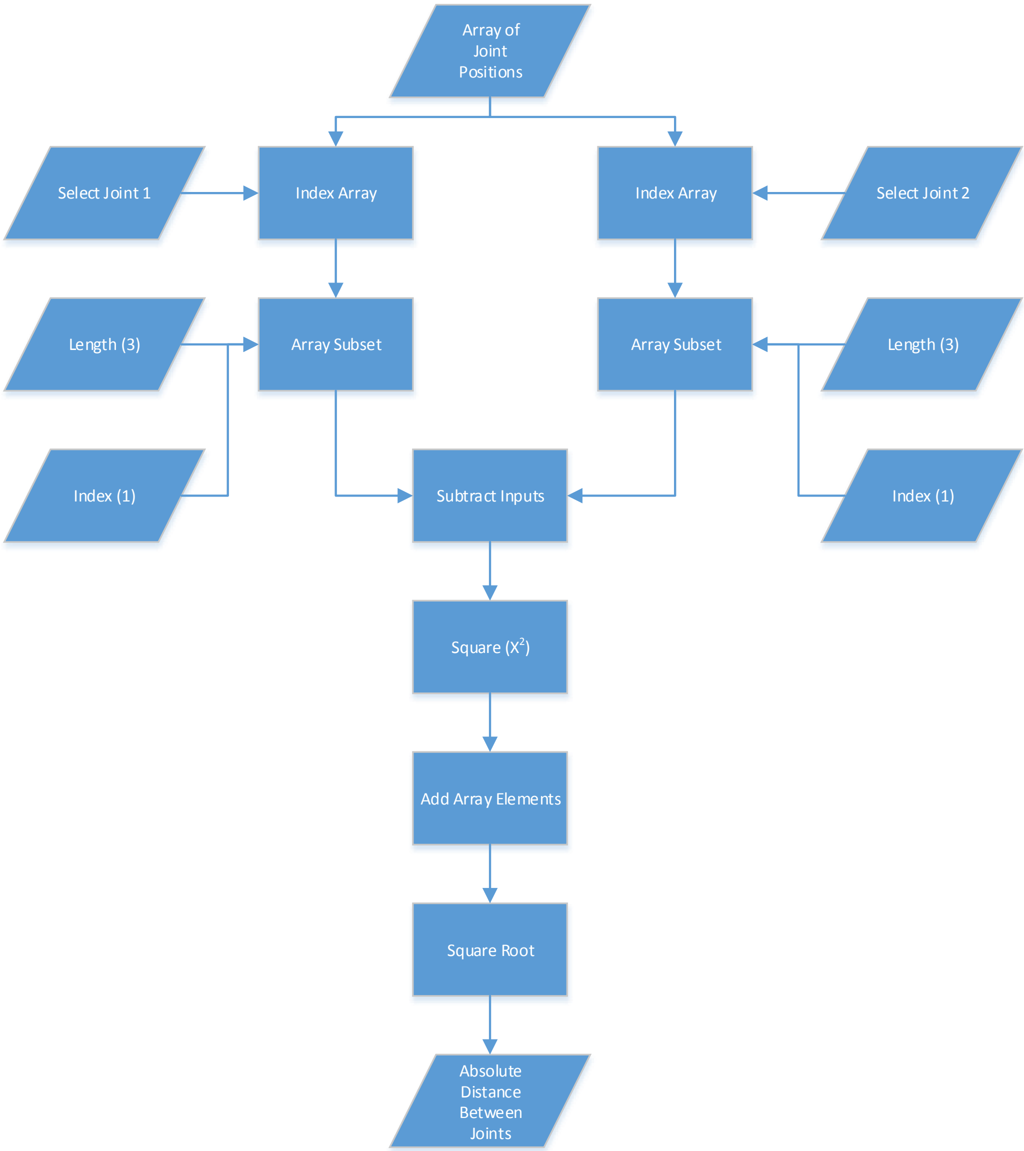


Figure 23 Block Diagram of Absolute Distance between Joints

4.2.2 Obtaining Absolute Distance between 20 Joints

By applying the "Displacement and Distance between Joints VI" between all of the 20 adjacent joints it was possible to obtain 19 absolute distances between the joints. The figure 24 below illustrates the 20 joints of a human:

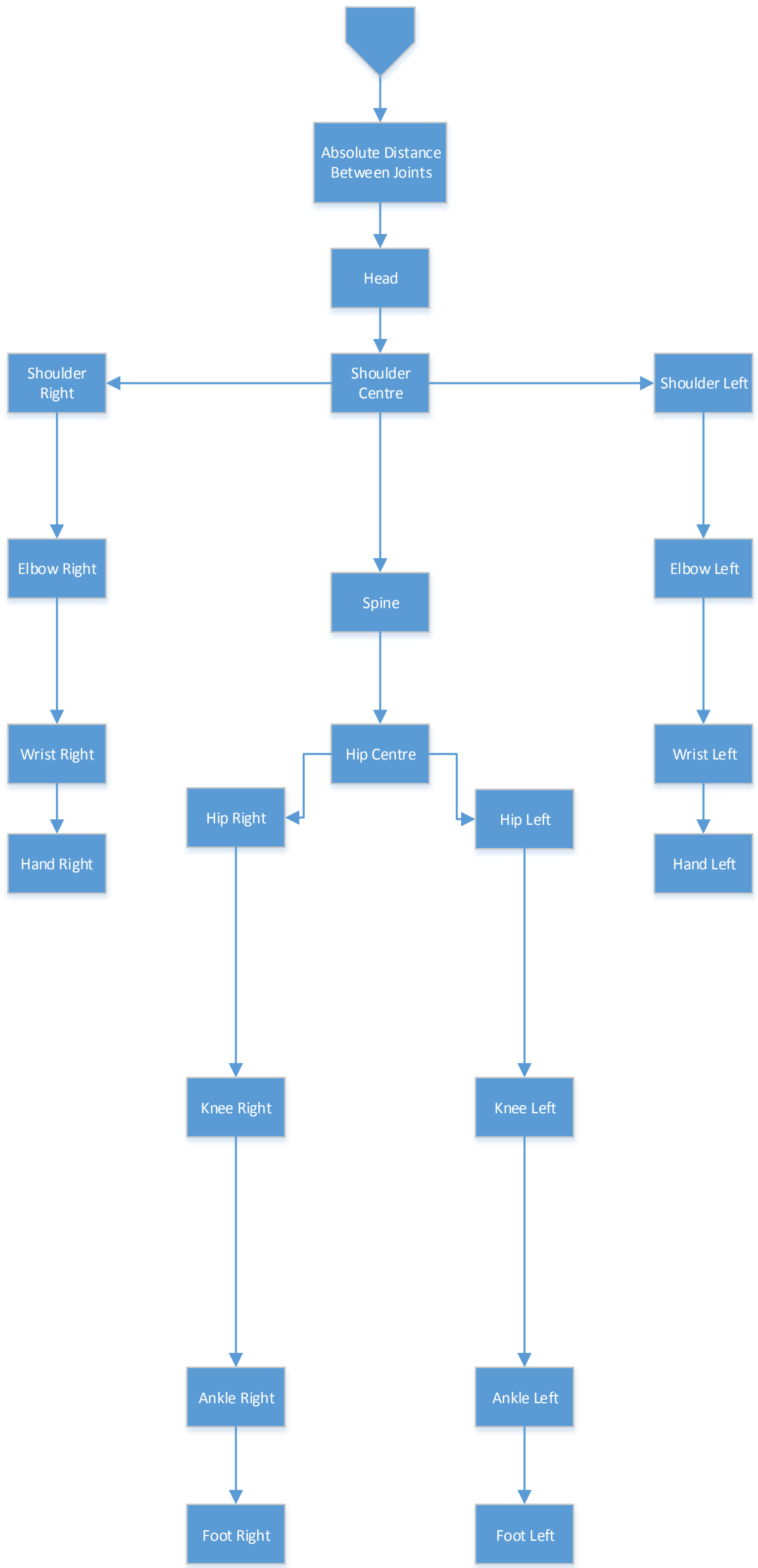


Figure 24 Block Diagram of 20 Body Joints

4.2.3 Saving and Reading Data obtained from Microsoft Kinect

When the robot starts operating, the person to be tracked must move into the field of view of Microsoft Kinect and stand at a minimum distance of 2 meters from the Kinect. Then the person should move backwards while holding his/her arms next to hips, because as the arms rise the shoulder joints and the spine extend in Microsoft Kinect's perspective thus giving inaccurate data. The person must continue moving forwards and backwards from 2 meters to 3.4 meters at least twice while facing the robot all the time. After moving twice, the person must move to at least 3.5 meters away from the robot for it to start roaming.

During this period (from 2 meters to 3.4 meters), the 19 absolute distances between all 20 adjacent joints are being written to 19 different spreadsheet files continuously. When the distance from Kinect reaches 3.5 meters, the writing stops and reading from each 19 spreadsheet starts. The figure 25 below illustrates the flow of the programs described:

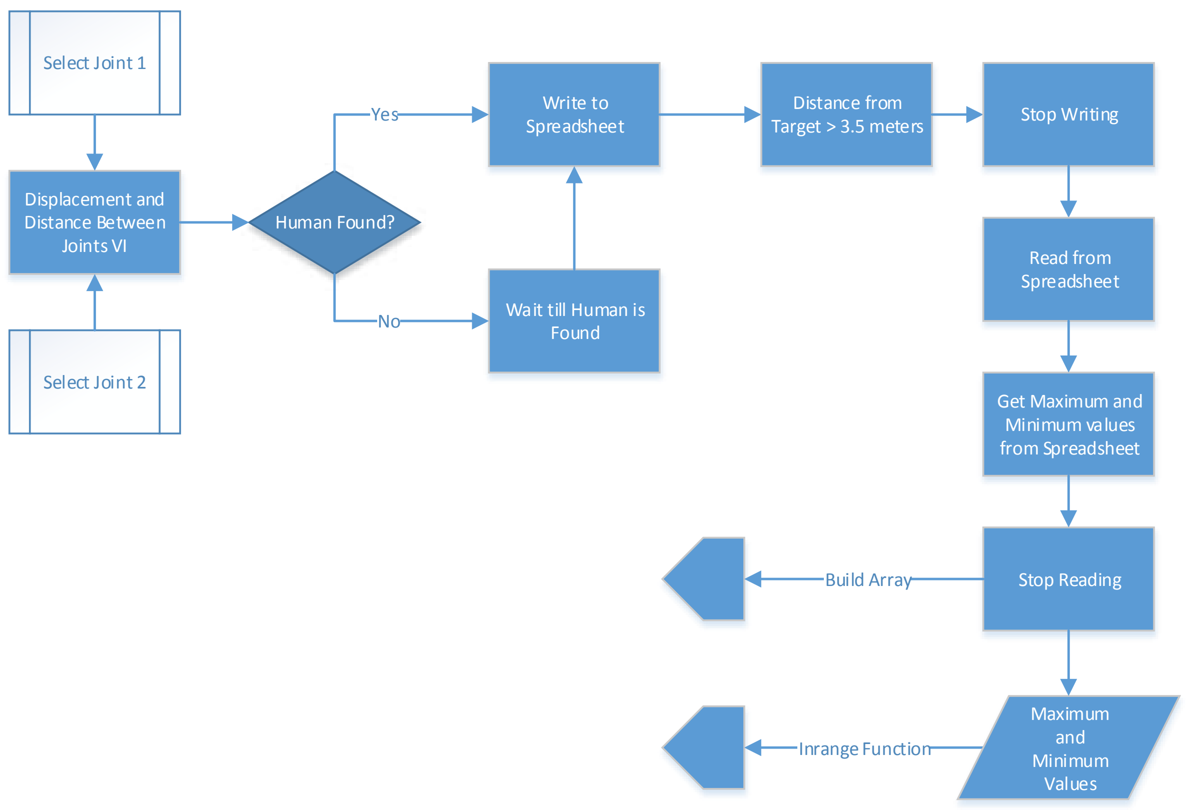


Figure 25 Block Diagram of Data Acquisition from Microsoft Kinect

4.2.4 Reading Data obtained from Microsoft Kinect and Inrange Function

The below VI,applies to all of the 19 absolute distances obtained. After each spreadsheet is read, the “Array Max and Min” function is applied to each read spreadsheet and the max and min values are extracted. The extracted maximum and minimum values are sent to an “Inrange and Coerce” function, which decides if the new recognized human’s absolute distance between joints is between the max and min values of the saved absolute distance between joints, if it is within the limits then the output is logic 1 if not then the output is logic 0. This operation is illustrated in the figure 26 below:

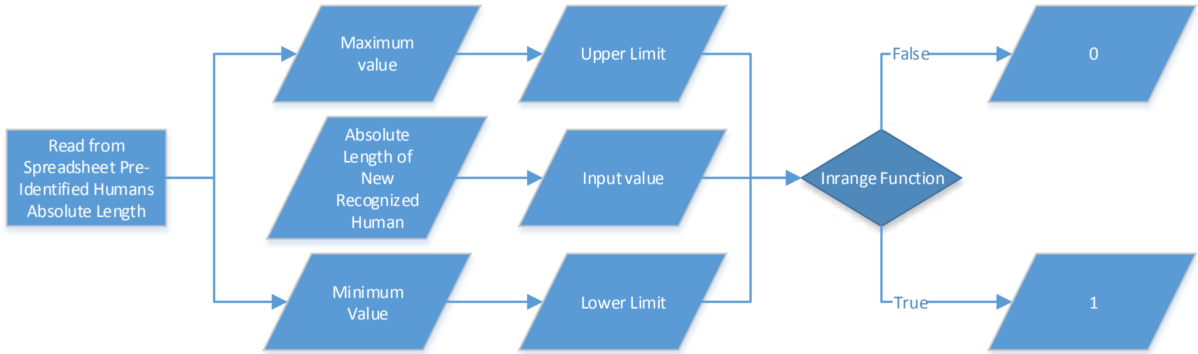


Figure 26 Block Diagram of Inrange function

4.2.5 Sum of Booleans

After the “Inrange and Coerce” function is applied to each 19 absolute distances of the next recognized human, the 19 outputs are either logic 1 or logic 0. Then all of these are summed using “Compound Arithmetic” function and choosing the “Add” option. The figure 27 illustrates the described process for only 2 absolute distances, but in practice this is applied to all 19 absolute distances obtained:

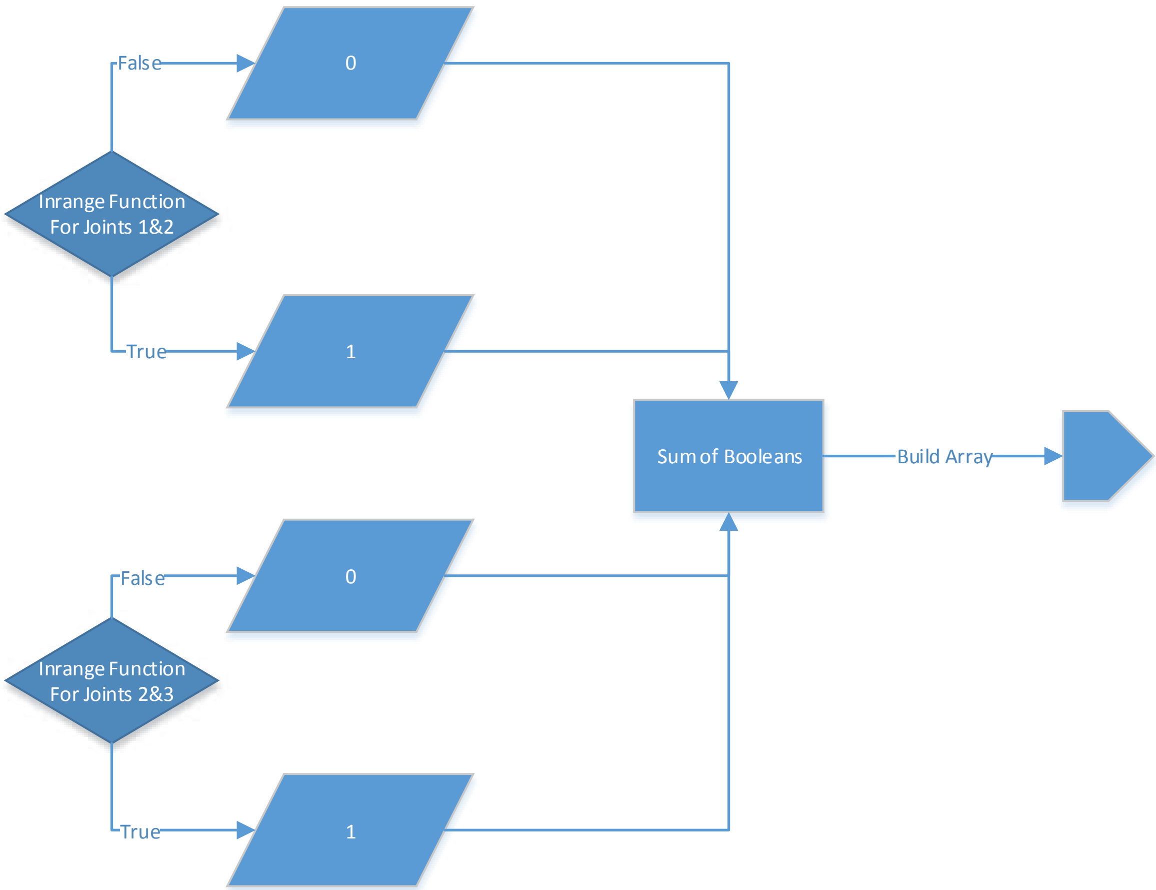


Figure 27 Block Diagram of Sum of Booleans

4.3 Initializing Microsoft Kinect and Skeletal Recognition

Kinesthesia toolkit was used to initialize and configure Microsoft Kinect to start skeletal recognition and 3D skeleton rendering in LabVIEW. From the skeletal data, X and Z coordinates of a selected joint could be extracted and in this case, Spine was chosen as the desired joint since it is located approximately at the centre of a human and the data obtained from it was most stable at different ranges. From the skeletal data, the obtained array of joint positions was fed to a sub VI called “Displacement and Distance between Joints” and from it, 19 absolute distances between 20 joints were extracted. Then the extracted X and Z coordinates as well as the 19 absolute distances were written to network stream to be read from the robot. The overall flow of the program can be seen in the figure 28 below:

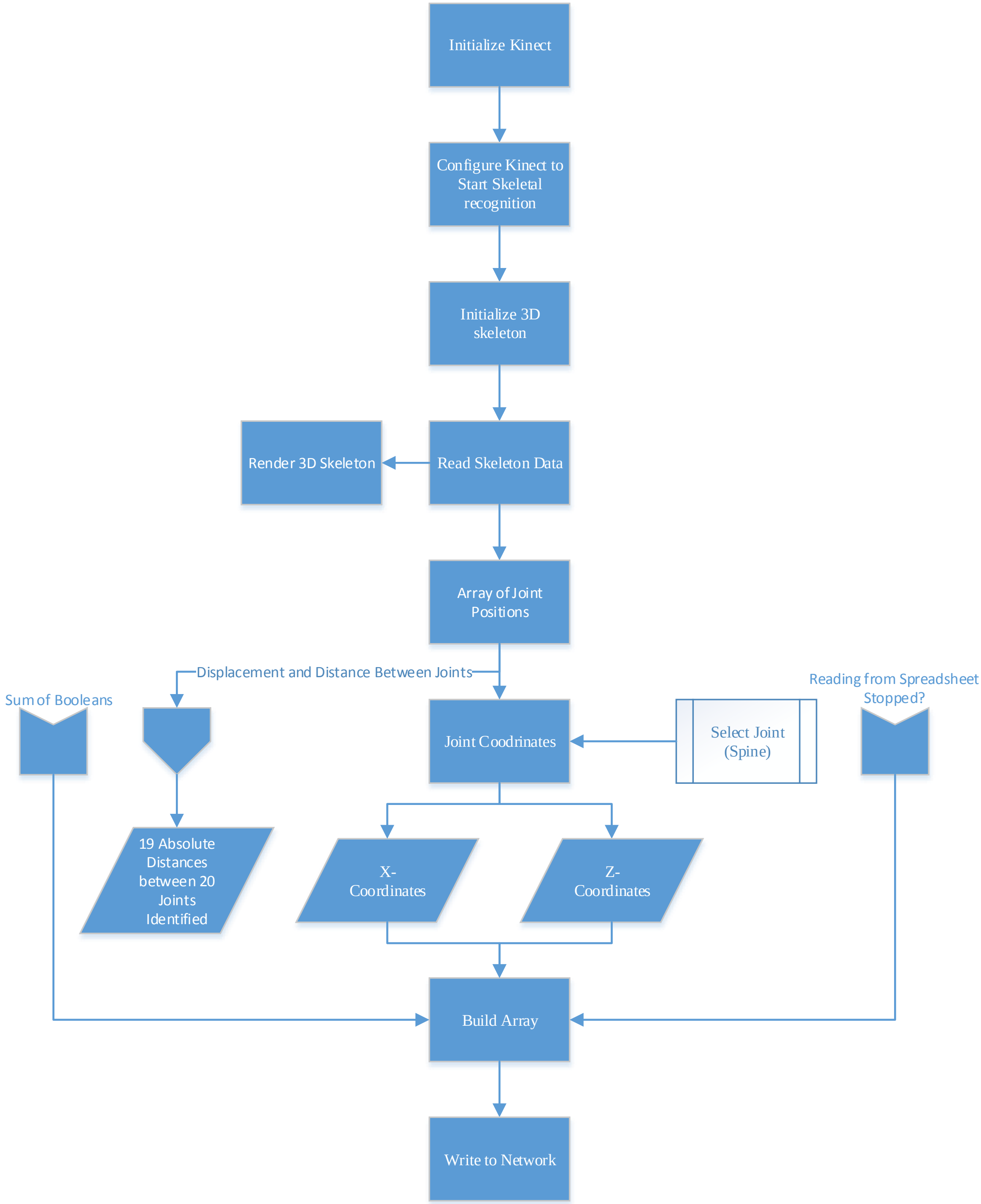


Figure 28 Block diagram of Initialization of Microsoft Kinect and Skeletal Tracking

The extraction of X and Z coordinates is described in the subsequent sections.

4.3.1 Writing to and Reading from Network

Since the laptop processes data from Microsoft Kinect, a network stream was necessary to transfer data acquired from laptop to DaNI robot via wireless connection.

The figure 29 illustrates how writing to network functions in LabVIEW.

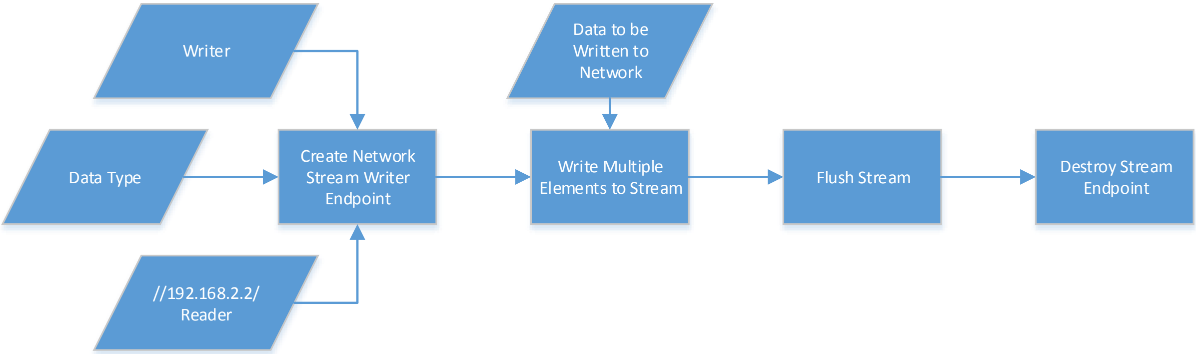


Figure 29 Block Diagram of Writing to Network

The figure 30 illustrates how reading from network functions in LabVIEW.

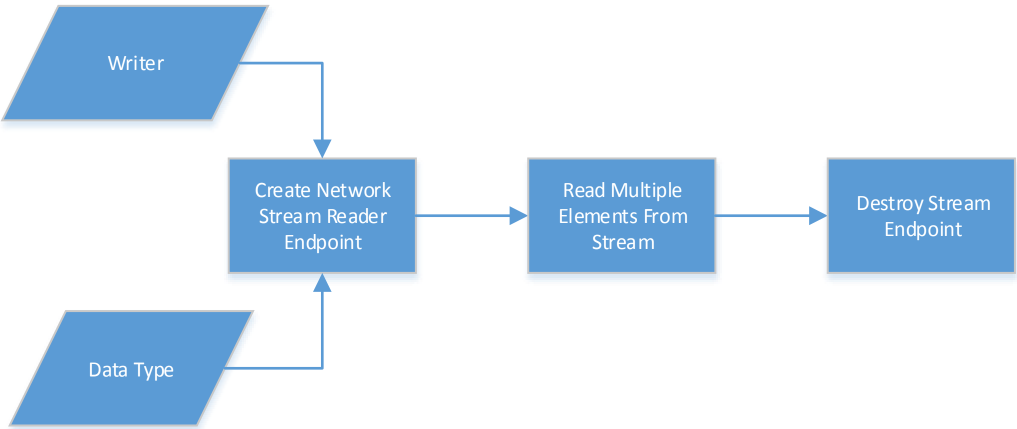


Figure 30 Block Diagram of Reading from Network

4.3.2 Extracting Joint Coordinates

The extraction of Joint coordinates X and Z was done by modifying the “Joint Coordinates VI” already available from Kinesthesia toolkit. The VI extracted all of the X,Y and Z coordinates of a joint together in a matrix form which was not useful for this project. That is why this was modified by using “Decimate 1D Array” function in LabVIEW to get each coordinates separately since each of X and Z coordinates were required separately. The figure 31 illustrates the modified “Joint Coordinates VI” which shows how each coordinate was extracted from the whole array of joint positions.

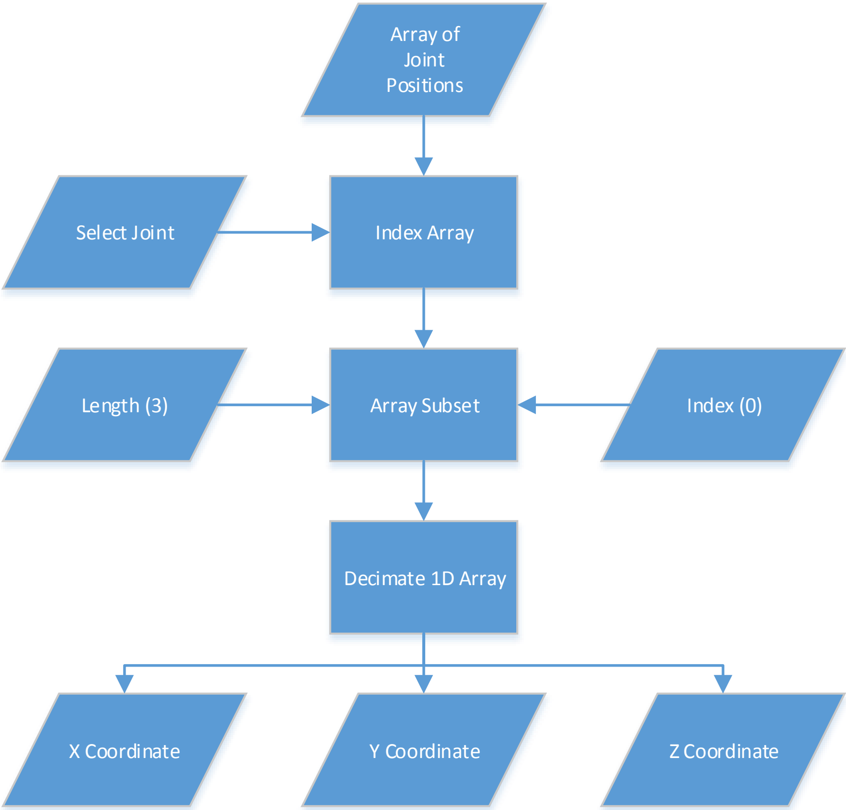


Figure 31 Block Diagram of Extraction of Joint Coordinates

4.4 Initializing DaNI

After all the required data were extracted from Microsoft Kinect and processed on laptop, they were sent to the DaNI robot via wireless network. Until now, the robot was functioning but was waiting for the processed data to be received via network streams. The robot is set to start roaming and tracking only after all the identification processes are done. The signal for the robot to know if the processing is done is when the laptop stops reading from spreadsheet after obtaining the maximum and minimum values. If the reading is stopped it means the identification process is over and the robot will start roaming, search for the target identified and track the target when found. If not the robot will wait for the signal. During both the searching and tracking, the robot will avoid any obstacles on the way by using the readings from the Ultrasonic sensor available on DaNI. All the data acquired from Kinect was fed to a modified “Calculate Driving Direction VI” which controls the entire autonomous locomotion of the DaNI robot. The figure 32 illustrates the overall processes executing in DaNI robot:

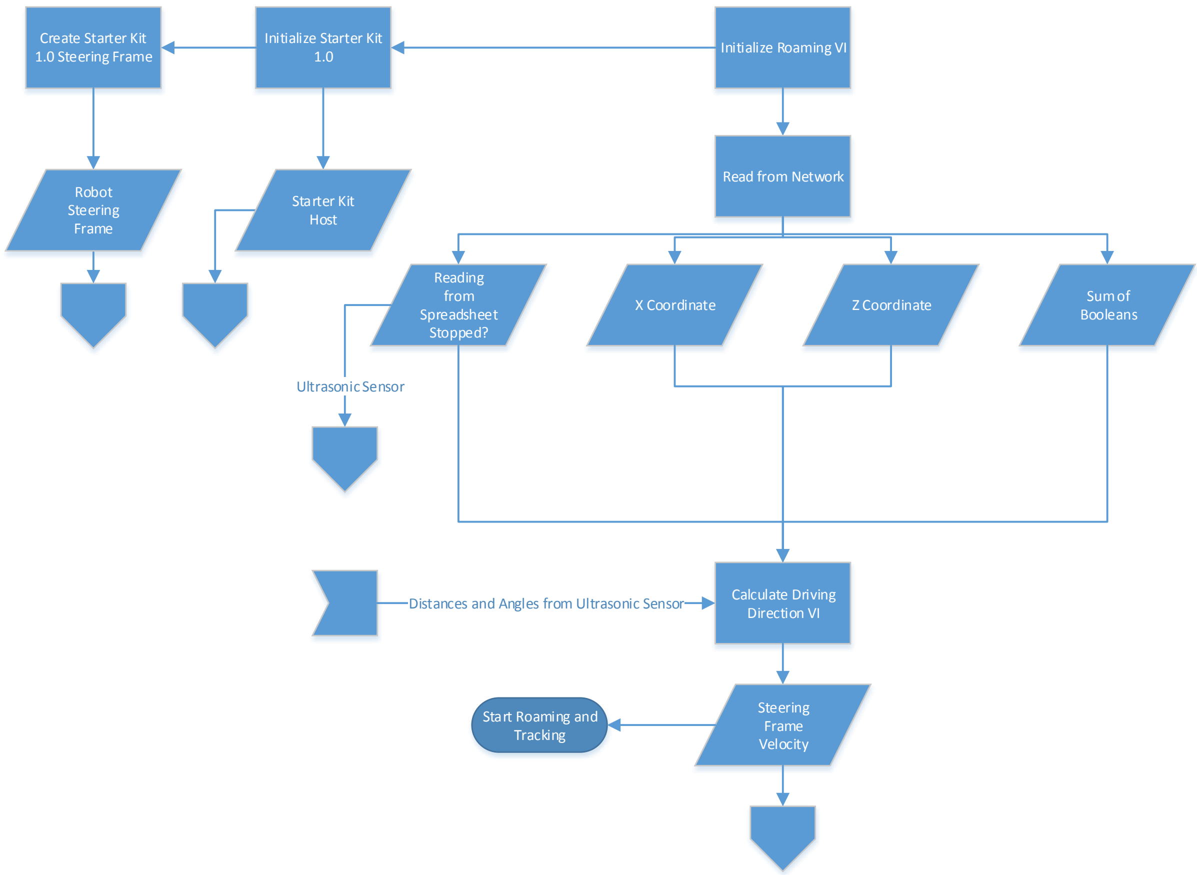


Figure 32 Block diagram of Initialization of DaNI

The extraction of data from Ultrasonic sensor and the sub VI's involved in achieving the roaming, tracking and obstacle avoidance, are described in the subsequent sections.

4.4.1 Initializing Ultrasonic Sensor

When the reading from spreadsheet stops, the ultrasonic sensor starts to operate immediately. The ultrasonic sensor is fixed on a servo motor to provide a sweeping motion to the sensor. The total sweeping angle was set to 130 degrees, 65 degrees clockwise and 65 degrees anti-clockwise with a scan angle of 4 degrees. The ultrasonic sensor made it possible to identify any obstacles on the way of DaNI robot by sending ultrasonic waves and receiving them. If there were any obstacles on the way the wave would be reflected back to the sensor at a definite time thus enabling to know how far is the obstacle from the robot. From the ultrasonic sensor the "Distance from obstacle" and the "angle" at which the obstacle is located is extracted and fed to the "Calculate Driving Direction VI" to control the motion of the robot based on the readings from ultrasonic sensor as well. The below figure 33 shows how the ultrasonic sensor is initialized and data is obtained from it in LabVIEW:

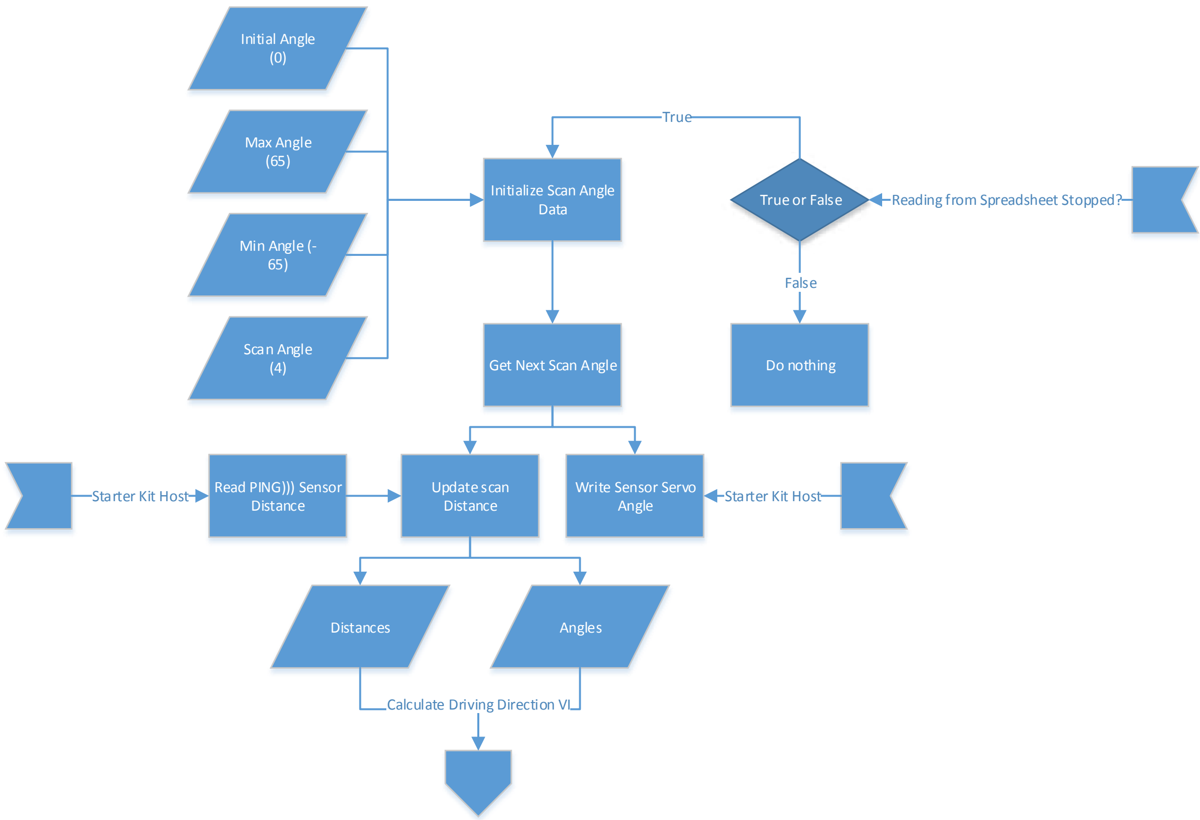


Figure 33 Block Diagram of Initialization of Ultrasonic sensor

4.4.2 Modified Calculating Driving Direction VI

One of the most important VIs of this project is the "Calculating Driving Direction" VI because this VI controls each of the locomotion, tracking and obstacle avoidance capabilities of the robot. All of the obtained data from both Microsoft Kinect and ultrasonic sensor are fed to this VI, which enables the implementation of a sophisticated and reliable algorithm for roaming, tracking and obstacle avoidance processes to run in parallel and not disrupt one another during operation. This VI is set to execute only after the identification process is complete so that the robot does not move before that. Stopping reading from spreadsheet, as described before, signals the end of identification and it signals the start of this VI. This VI is a totally modified "Calculate Driving Direction" VI available in LabVIEW Robotics. The one in LabVIEW only uses the data from ultrasonic sensor to avoid obstacles while roaming while the modified VI developed besides using the ultrasonic sensor it also uses the data from Microsoft Kinect which enables the tracking of an Identified person possible. From the Z coordinate obtained from Microsoft Kinect the distance of a human from the robot can be known and by using the X coordinate then getting arctan of (X-Coordinate/Z-Coordinate) the angle at which the human is facing the robot can be calculated. Here the arctan(X/Z) is multiplied by (-) since left and right in Kinect's perspective is in positive and negative respectively, while in robot's perspective it is the exact opposite, right being positive and left being negative. The output angle is in radians and since the whole VI is in a continuous loop the angle becomes rad/sec and is fed directly to the motors when tracking. However, the turning speed was found to be very slow so the angle is multiplied by 2 to increase the turning speed. Most important data here is the “Sum of Booleans” because this decides whether a recognized human is the identified target or not.

As described before, the identification process is based on comparing the “Absolute Distance between Joints” of the first person seen by Microsoft Kinect and any other human seen after that. From Kinect 19 “Absolute Distance between Joints” were extracted and sent to this VI. The identification criteria is based on choosing a threshold as how many Joints need to match to successfully identify the recognized person as the correct target. After doing many practical tests, the threshold level was chosen to be 16 joints out 19 joints must match for the person to be correctly identified as the target. This could have been 19 out of 19 which would make the algorithm much more reliable but this drawback was because the data obtained at different distances and circumstances were not precise enough. It was found that even after data were obtained at different distances to increase the precision it was still not enough to obtain 100% precise “Absolute Distance between Joints” at every conditions. This is due to the fact that whenever the robot starts moving inevitably the Kinect vibrates so disrupting the data obtained from Kinect. This is because the “Absolute Distance between Joints” is based on the Y coordinate, which changes whenever either the position of the target is changed or the elevation of Microsoft Kinect is changed. This threshold level proved to be accurate enough to distinguish most of the persons provided that no hands are raised above the hips, which causes inaccurate readings from Kinect as described before. Because there is a loss of 4 joints to match, 2 persons with nearly same body built could be mistakenly identified as the same person.

After receiving all these data, now it is up to this VI to control the behaviour of the robot. Initially the robot will start normal roaming and avoid any obstacles on the way until a human is found in its field of view. The person is immediately compared with the previously identified person to see if the recognized human is the target or not, if at least 16 out of 19 joints match then the target is recognized as the target and the robot will start tracking him/her using the X and Z coordinates as described in Identification methodology. Whenever any obstacle comes before the robot while tracking, the robot will stop tracking and will avoid the obstacle and then again search for the target. If the target is not recognized then the robot will ignore and treat him/her as an obstacle and will continue the search for the target. The final output of this VI is the “Steering Frame Velocity” which is fed to “Apply Steering Frame Velocity to Motors” VI for the autonomous locomotion to be accomplished.

The figure 34 below illustrates in detail the flow of this program in LabVIEW:

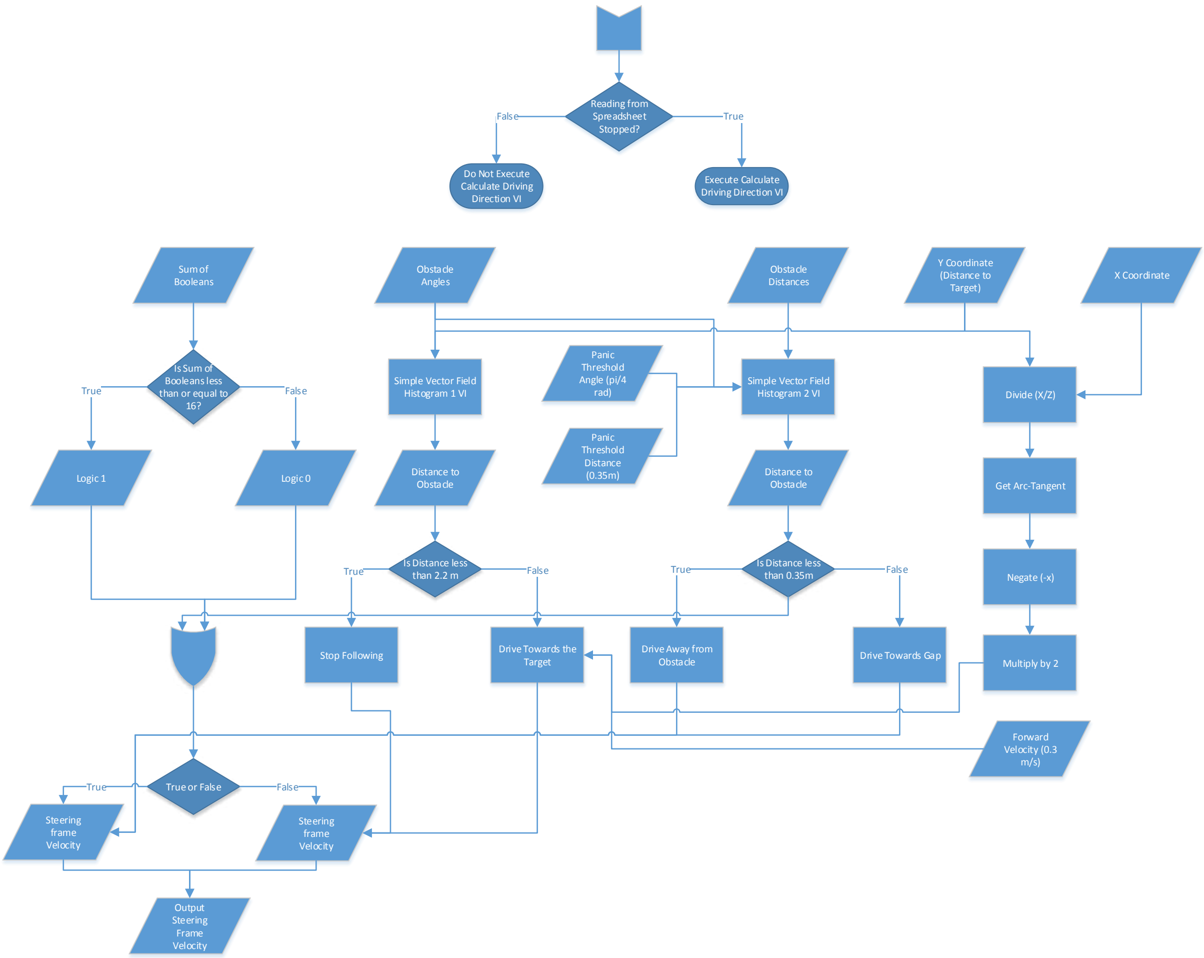


Figure 34 Block Diagram of Modified Calculating Driving Direction VI

4.4.3 Locomotion and Dead Reckoning

After every calculation is done the “Steering Frame Velocity” is sent to the “Apply Steering Frame Velocity to Motors” VI which controls the motors controlling the 2 wheels on the robot and thus the autonomous locomotion is accomplished based on the cases described in “Calculate Driving Direction” VI.

A simple localization method called Dead Reckoning was implemented to show the location of the robot on a XY graph in LabVIEW based on the readings from the encoders mounted on the motors. The estimation of position of robot is described in detail in the following section.

The figure 35 illustrates the steps involved in applying the velocities to the DC motors and the Dead Reckoning method to show the location of the robot.

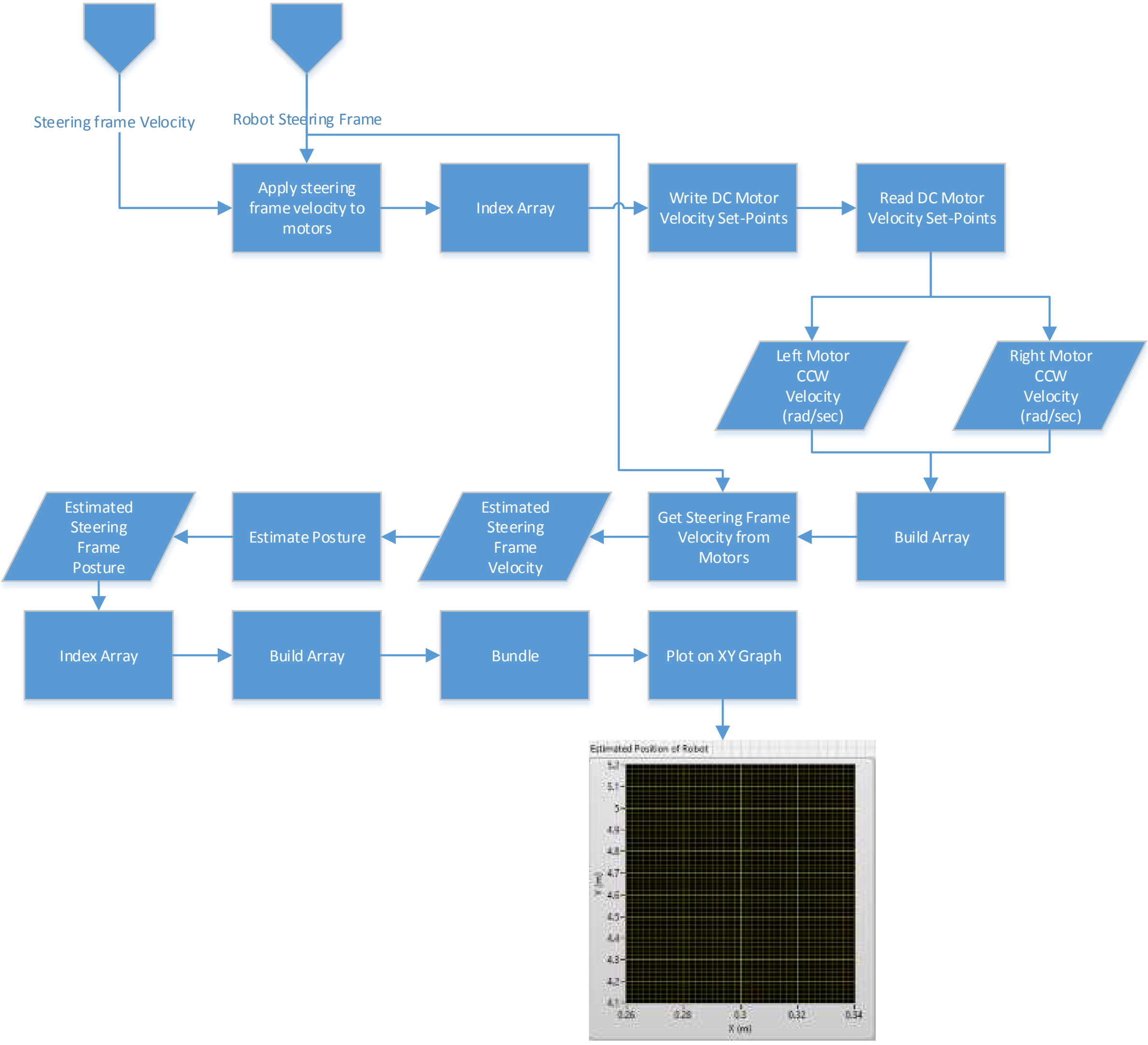


Figure 35 Block Diagram of Locomotion and Dead Reckoning

4.4.4 Estimating Posture from Velocity

The estimation of posture is based on the estimated steering frame velocity and the initial posture of the robot. From the both the “Estimated steering frame velocity” and “Initial Posture” each of the “Lateral velocity, Forward Velocity and Angular velocity” are obtained and are used to estimate the position of robot on a XY graph. The calculation can be divided into the following steps:

1. Integrate the angular velocity component to estimate the angular position.

2. Rotate the X and Y velocity components to be aligned with the global frame of reference.

3. Integrate the rotated X and Y velocity components to estimate the X and Y position.

Below is a detailed figure 36 showing how the position is estimated in LabVIEW as described above:

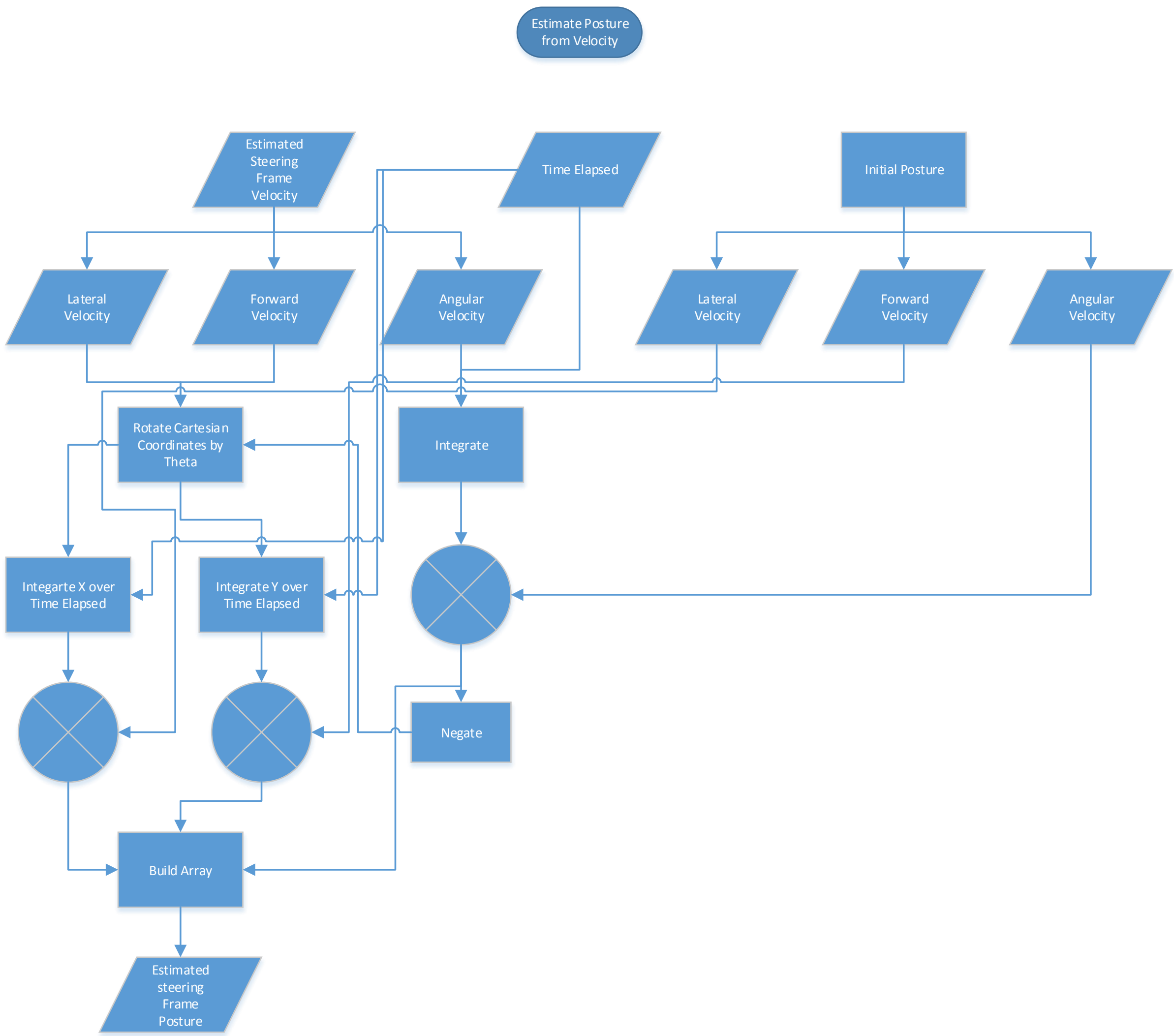


Figure 36 Block Diagram of Estimating Posture from Velocity VI

5 Results and Discussions

In this chapter, the calibration of Kinect's tilt angle, evaluation of Microsoft Kinect's depth sensor and the practical results achieved after implementing the proposed methodology are demonstrated and discussed in details.

5.1 Kinect Sensor Tilt Angle Calibration

Since the Kinect has a 43° vertical by 57° horizontal field of view it was necessary to position and tilt the Kinect at an angle which will see the whole body of a target without the view being obstructed by the platform. After practically observing the field of view by using the RGB camera the tilt angle was chosen to be 17 degree.

5.2 Optimum Operational Range of Microsoft Kinect for Skeletal Tracking

After doing several practical tests it was found that the Kinect can not detect humans at a distance less than 2 m away from Kinect, which in contrast as told by Microsoft it should detect humans at a range of 1.2 m. This is because the Kinect's height is only 30 cm above the ground when it is placed on the robot, and since it has a 43° vertical field of view the whole body of a human does not fit totally into the field of view of Kinect at a distance less than 2 meters away from Kinect. But it was noted that when the Kinect starts recognizing a human at a distance of 2 meters and then the human starts moving forward, the Kinect still was able to recognize humans until 1.2 meters away from Kinect as claimed by Microsoft. However, since the whole body is not in the field of view of camera the accuracy of most of the joints of upper and lower body parts fell drastically.

For precision purposes, the 19 "Absolute Distance between Joint" obtained from Kinect was observed at different distances away from Kinect, starting from 1.43 to 4 meters.

The data taken at different distances away from Kinect are as shown in the figures from 37 to 46 below:

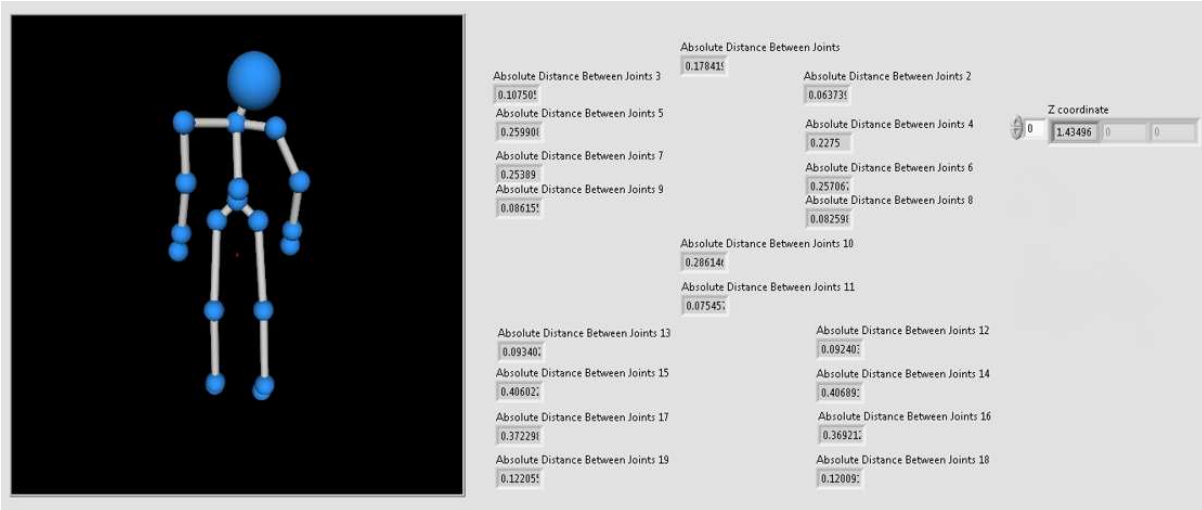


Figure 37 At 1.43 meters away from Kinect

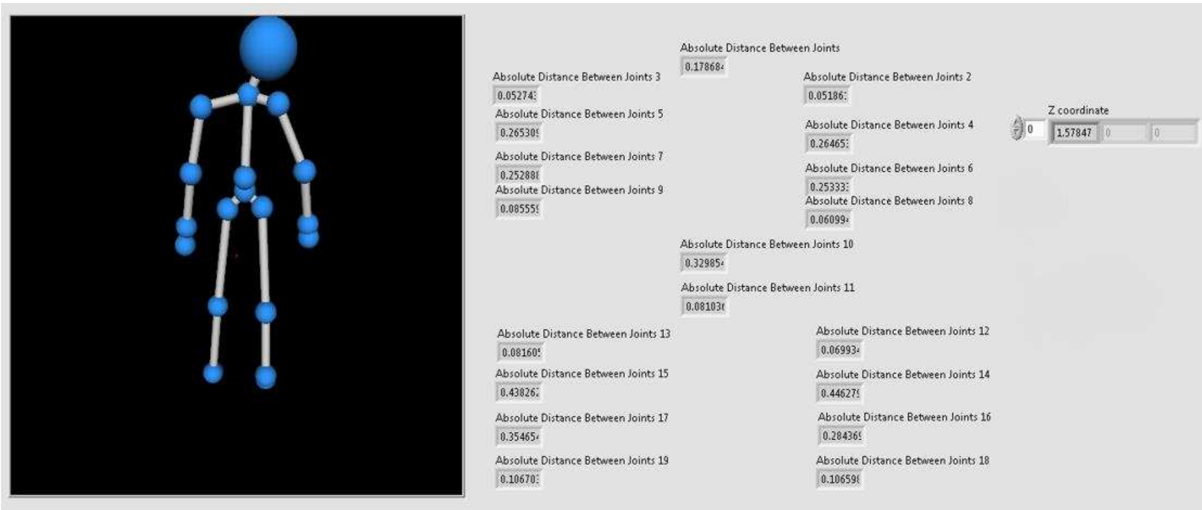


Figure 38 At 1.57 meters away from Kinect

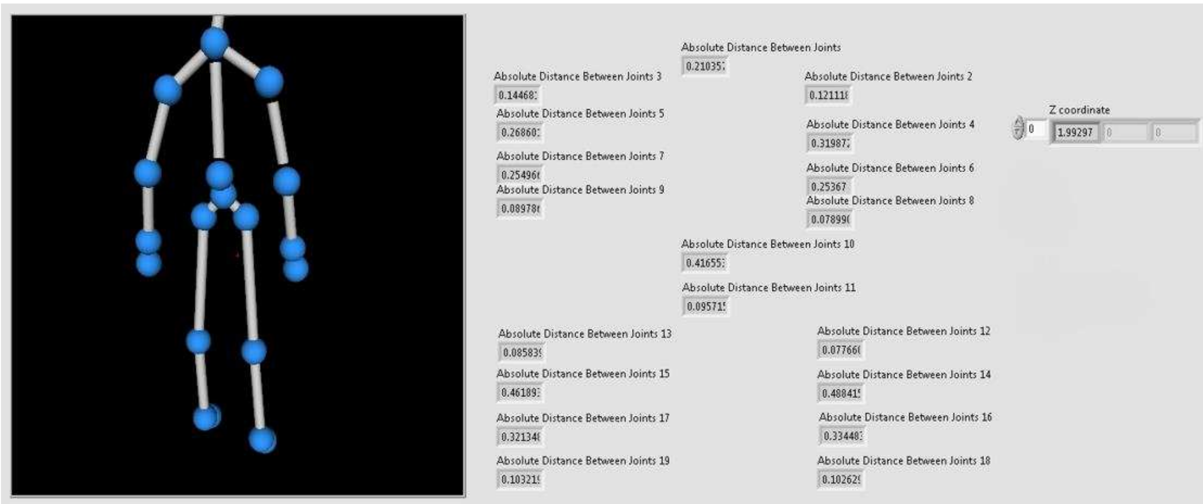


Figure 39 At 1.99 meters away from Kinect

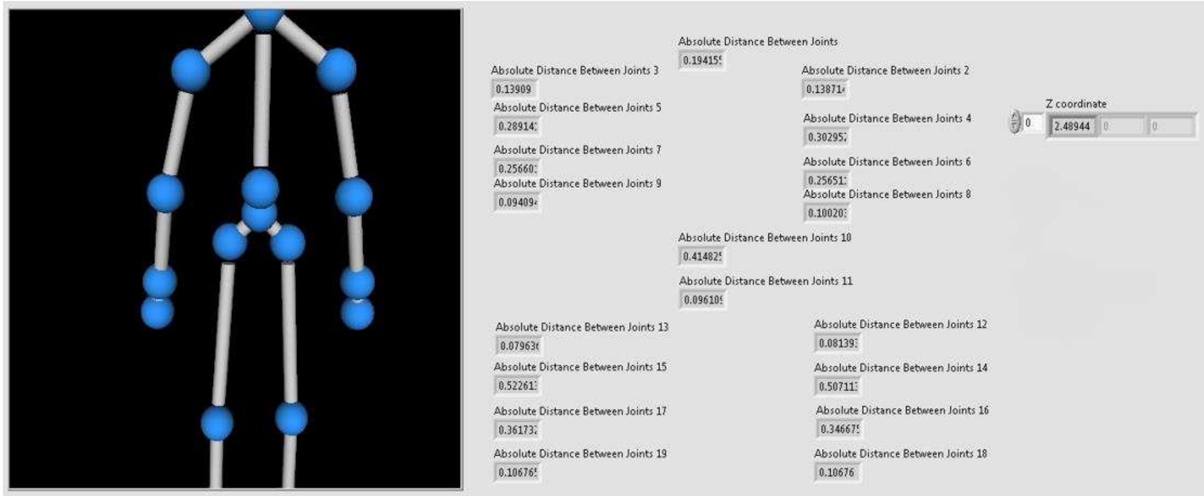


Figure 40 At 2.48 meters away from Kinect

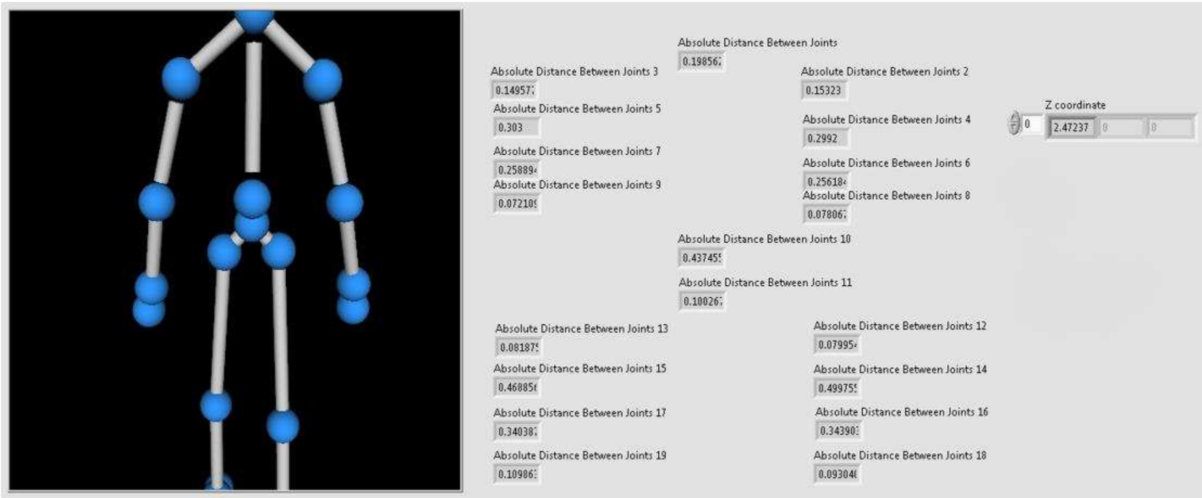


Figure 41 At 2.47 meters away from Kinect

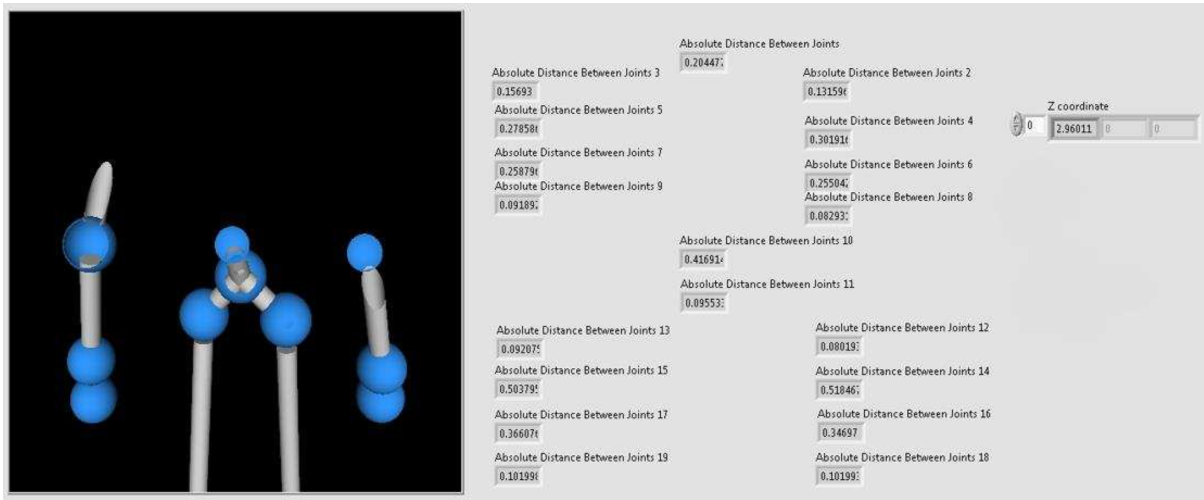


Figure 42 At 2.96 meters away from Kinect

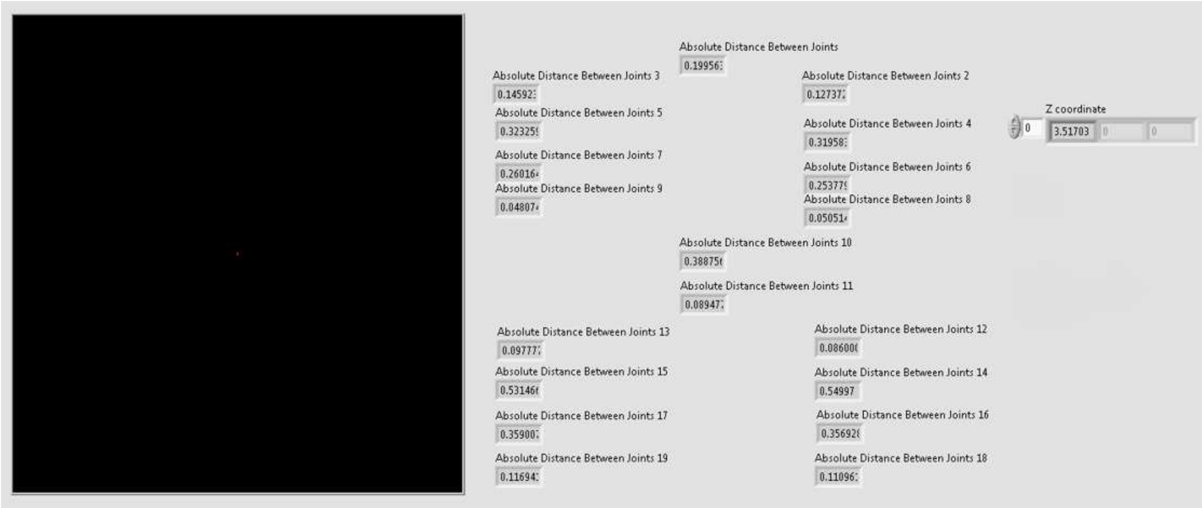


Figure 43 At 3.51 meters away from Kinect

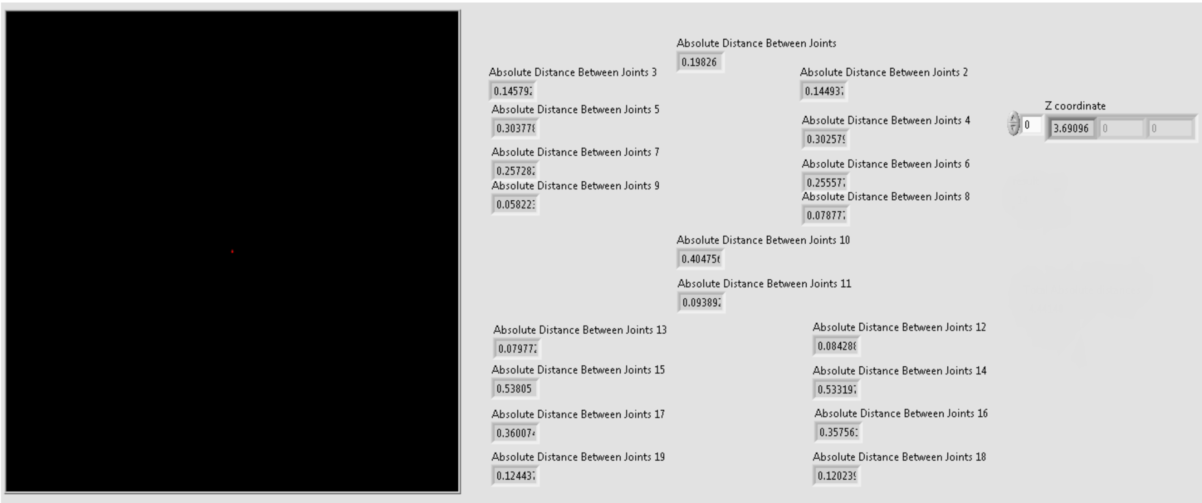


Figure 44 At 3.69 meters away from Kinect

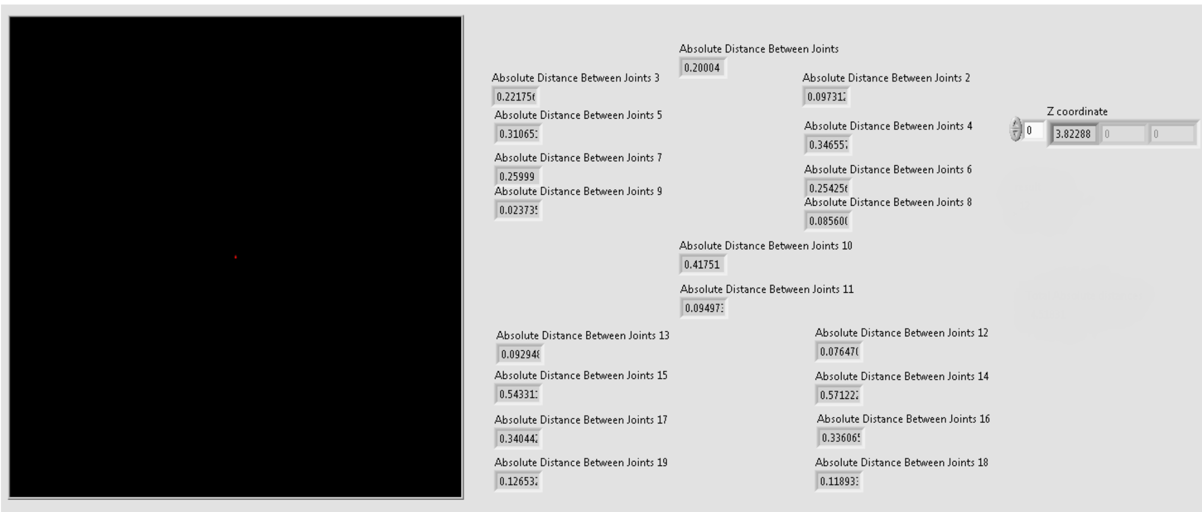


Figure 45 At 3.82 meters away from Kinect

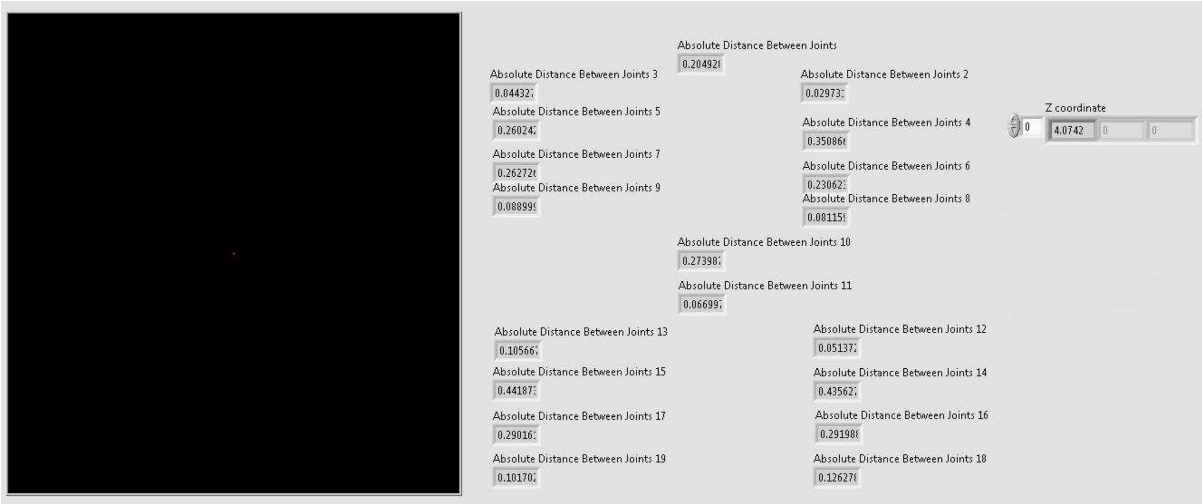


Figure 46 At 4.07 meters away from Kinect

It should be noted that a mirroring technique is implemented into the toolkit making the avatars more visible as moved nearer to the Kinect; hence, the avatars become less visible as the target moves further away from Kinect.

By observing the 19 “Absolute Distance between Joints” from figure 37 to figure 46 it can be easily noticed that the data is precise only from a distance of 1.99 m (Figure 39) to a distance of 3.69 m (Figure 44) away from Kinect. However, at distances nearer than 1.99 m and further than 3.69 m the data are inconsistent and are unreliable.

It should be noted that even between the ranges of 1.99 m to 3.69 m sometimes the data can be inconsistent as well as seen in figure 40 and 41. Figure 40 is data taken at 2.48 m and is consistent when compared with the data taken from 1.99 m to 3.69 m, however, in figure 41 the data is taken at the exact location (at 2.47 m) at the same time (there is a gap of 1 second between the two data taken) but the data is inconsistent and there is a considerable loss in precision of the data taken. After taking so many data and analysing it, no particular reason was found to be the cause of this imprecision. The data were taken indoors, under no sunlight conditions and the Kinect was stable as the robot was stationary during the whole data acquisition process, hence all the guidelines as stated by Microsoft were followed but still there were imprecisions detected even at optimal range of operation.

From the data taken, and taking into account the fact that the robot will be mobile during roaming and tracking (hence the precision will decrease further since the Kinect will be subject to vibrations due to either the surface not being smooth and/or the slight/sudden change in speed of robot when avoiding obstacles). It was decided that the operation range will be between 2 meters and 3.5 meters hence, the identification will be done within this range. As for tracking, the robot will stop tracking if the distance to the target drops below 2.1 meters.

5.3 Human Pose Limitations

Several poses were analysed to see their effect on the precision of data taken by Microsoft Kinect and their impact on the Identification algorithm developed. The figures 47 to 49 show another person from the one present from figures 37 to 46. In these figures 47 to 49 the person is standing at the same position while raising her hands at different angles for the changes to be observed during these poses, as shown in figures below:

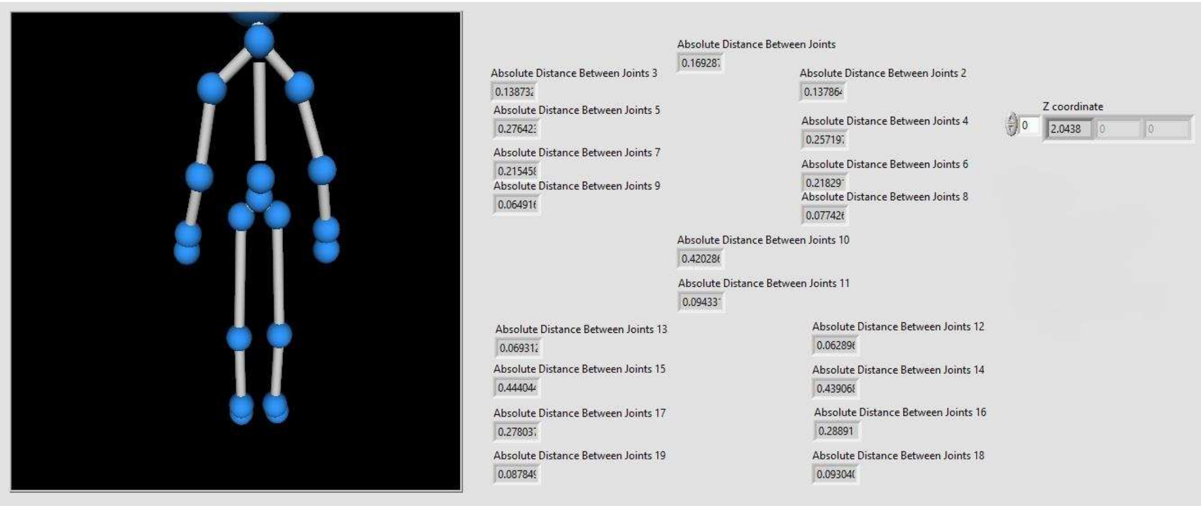


Figure 47 Standing at 2.04 meters away from Kinect while arms at natural lowered position

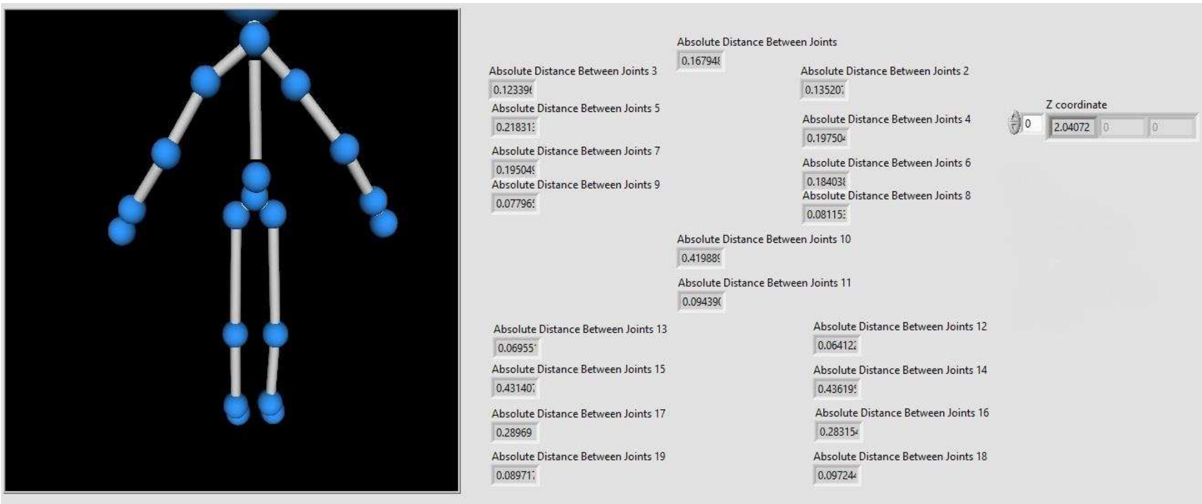


Figure 48 Standing at 2.04 meters away from Kinect while arms raised making an approximate angle of 40 degrees with the spine

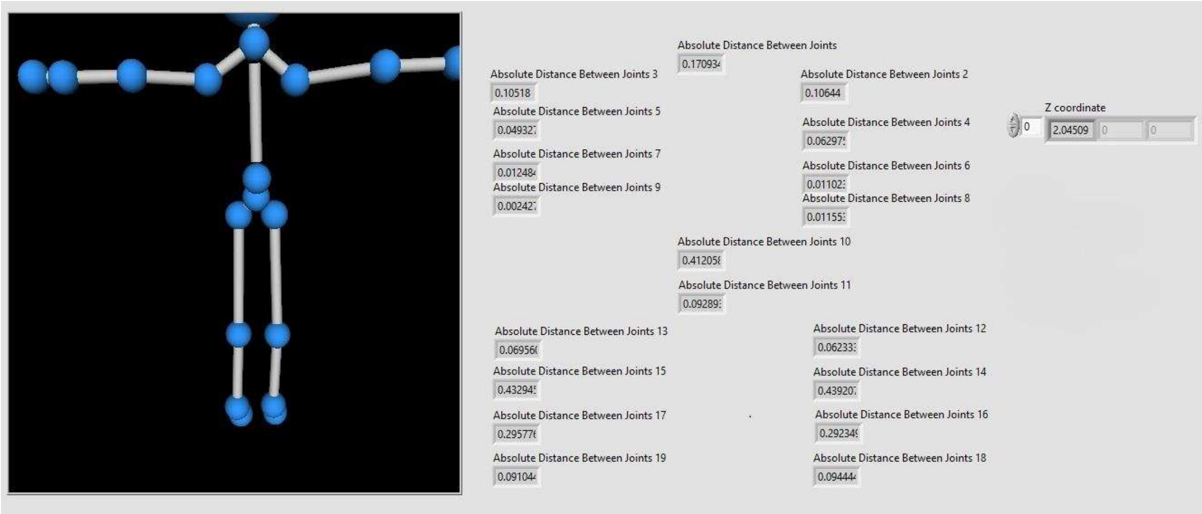


Figure 49 Standing at 2.04 meters away from Kinect while arms raised making an approximate angle of 90 degrees with the spine

From these figures 47 to 49 it can be clearly seen that as the hands are raised the data of the upper body joints change significantly especially after the point where the arm makes an angle of 40 degrees with the spine. Since the identification algorithm is based on the maximum and minimum values of absolute distance between joints, as the arms are raised the maximum and minimum range will significantly increase thus decreasing the precision of the algorithm greatly. Therefore, the person to be identified must never raise his/her hands above 30 degrees. A lower angle was chosen to increase the efficiency of the algorithm since even at 40 degrees as shown in figure 12 there is a considerable change in values of the upper body joints.

5.3 Practical Results

In this section, practical results achieved for the objectives of this project is discussed in detail.

5.3.1 Human Identification, Tracking and Obstacle Avoidance

The algorithm involved in each of human identification, tracking and obstacle avoidance were described in detail in chapter 4. The figures below show the robot during its operation in sequence.

5.3.1.1 Tracking the Identified Human

After the human to be tracked was identified (the one wearing green and white shirt), the robot started roaming and searching for the target. After the target human was found, the robot started tracking him/her by using the X and Z coordinates of the target obtained from Kinect as described in chapter 4. This can be seen in the figure 50 below:

Figure 50 Human Target Tracking

5.3.1.2 Obstacle Avoidance while Tracking

While tracking and roaming if any obstacle comes in front of the robot, the robot will quit tracking/roaming and will avoid the obstacle and then start roaming and its search for the human target. This can be seen in figures 51 and 52 below:

Figure 51 Obstacle Avoidance while Tracking

Figure 52 Continuation of Target Tracking after Avoiding Obstacle

5.3.1.3 Tracking Only the Identified Human Target

The robot is developed to track only the previously identified human and ignore any other humans recognized in the field of view of Kinect as described in detail in chapter 4. From figure 53 it can be seen as the robot is tracking the target if anybody else comes into the field of view of Kinect, the Kinect will not recognize the new human and will track the target without interruption. This is because Kinesthesia toolkit allows the recognition of only the first person in Kinect's field of view.

Figure 53 Tracking Only the Identified Human Target

In figure 54, it can be seen that although there are 2 persons in Kinect's field of view, the robot does not track them and continues roaming since they are not the identified target.

Figure 54 Just Roaming since the Previously Identified Human Target is not among the Recognized Humans in the field of view of Kinect

It should be noted that the robot was able to distinguish between the identified human target and anybody else in its field of view because no one had the same body built as the identified target. The algorithm developed is based on scanning the body of a human target then tracking the human based on his/her physical body characteristics. This is this the limitation of this approach but proved to be sufficient in most of the cases if the human to be tracked has not the same body built as the humans to be scanned as shown in the figures above. The whole algorithm is dependent on the precision of data taken from Kinect and can prove to be very practical in real life scenarios if the precision of Kinect is improved.

5.3.2 Dead Reckoning method for Localization

The Dead Reckoning was implemented successfully to show the position of the robot on an XY graph. However, after a few minutes of operation due to the accumulation of errors resulting from the slipping of wheels, the estimated position started to be inaccurate thus becoming unreliable in real life scenarios. As a result, different localization methods were proposed but not implemented since it was beyond the scope of this project.

6 Conclusion and Future Work

In this chapter, the conclusions and related recommendations for future work are discussed.

6.1 Conclusions

The aim of the current study was to improve the functionality of a mobile robot by finding, recognizing and tracking a previously identified human target using Microsoft Kinect. This dissertation has investigated and evaluated in detail the practicality of using Microsoft Kinect in real life scenarios for human recognition, Identification and tracking at indoor environments where no light sources that interfere with the infrared sensor of Kinect such as direct sunlight is present. Human identification algorithm based on the absolute distance between joints was developed, analysed and implemented successfully on a mobile robot followed by a detailed evaluation for real life scenarios.

The present study confirms previous findings and contributes additional evidence that suggests Microsoft Kinect is the most reliable sensor for human recognition and Identification usable on a mobile robot for indoor environments at significantly lower cost when compared with other range sensing devices.

In conclusion, after the identification and tracking algorithms were developed and then successfully implemented on a mobile robot, which showed quite pleasing results, it is safe to assume that the main aim and primary objectives of this study has been successfully achieved. However, a number of important limitations need to be considered. First, the person to be identified and tracked must never raise his/her arm further than 30 degrees (the angle between the arm and the spine at the shoulder joints). Secondly, since the identification is based on the absolute distance between joints, thus body-build, the algorithm can misidentify two persons having approximately the same body-build. Finally, the Microsoft Kinect uses infrared for range sensing thus making it dependable only indoors where there is no direct sunlight.

6.2 Future Work

1. Kinect originally can detect 6 people and track 2 at the same time, but with Kinesthesia toolkit in LabVIEW only one person is recognized and tracked which reduces the practicality of the algorithm developed very much. Therefore, the Kinesthesia toolkit can be improved to recognise all the six detectable people and track two, which will increase the practicality of the algorithm developed greatly.
2. An additional supporting identification algorithm such face recognition can be implemented to distinguish two persons have approximately the same body-build.

References

[1] http://sine.ni.com/nips/cds/view/p/lang/en/nid/208010

[2] http://www.microsoft.com/en-us/kinectforwindows/

[3] Stereo camera, http://en.wikipedia.org/wiki/Stereo camera

[4] S. Gokturk, H. Yalcin, and C. Bamji, A Time-Of-Flight Depth Sensor System Description, Issues and Solutions, Proc. IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 35-45, 2004

[5] http://www.microsoft.com/en-us/kinectforwindows/develop/new.aspx

[6] http://sine.ni.com/nips/cds/view/p/lang/en/nid/210938

[7] Loreto Susperregi, Jose Maria Martínez-Otzeta, Ander Ansuategui, Aitor Ibarguren and Basilio Sierra, RGB-D, Laser and Thermal Sensor Fusion for People Following in a Mobile Robot, International Journal of Advanced Robotics Systems, 2013.

[8] MESA Imaging. Available online: http://www.mesa-imaging.ch/ (accessed on 14 December 2011).

[9] PMD [vision] CamCube 3.0. Available online: http://www.pmdtec.com/products-services/ pmdvisionr-cameras/pmdvisionr-camcube-30/ (accessed on 14 December 2011)

[10] Freedman, B.; Shpunt, A.; Machline, M.; Arieli, Y. Depth Mapping Using Projected Patterns. U.S. Patent 2010/0118123, 13 May 2010.

[11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, Real-Time Human Pose Recognition in Parts from a Single Depth Image, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1297-1304, 2011.

[12] K. Khoshelham, and S. Elberink, Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications, Sensors, vol. 12, pp. 1437-1454, 2012

[13] J. M. Martinez-Otzeta, A. Ibarguren, A. Ansuategui and L. Susperregi. Laser based people following behaviour in an emergency environment. Proceedings of the 2nd International Conference on Intelligent Robotics and Applications, ICIRA '09, pages 3342, Berlin, Heidelberg, 2009. Springer- Verlag. ISBN 978-3-642-10816-7, 2009

[14] http://sine.ni.com/ds/app/doc/p/id/ds-217/lang/en

[15] http://sine.ni.com/nips/cds/view/p/lang/en/nid/205894

[16] http://msdn.microsoft.com/en-us/library/jj131033.aspx

[17] http://msdn.microsoft.com/en-us/library/hh855359.aspx

[18] http://msdn.microsoft.com/en-us/library/jj131032.aspx#ID4EVC

[19] http://www.openni.org/

[20] http://msdn.microsoft.com/en-us/library/jj131023.aspx

[21] Jungong Han, Ling Shao, Dong Xu and Jamie Shotton, Enhanced Computer Vision with Microsoft Kinect Sensor: A Review, www.ntu.edu.sg/home/dongxu/KinectReview_TC2013.pdf

[22] http://msdn.microsoft.com/en-us/library/hh855356.aspx

[23] http://msdn.microsoft.com/en-us/library/hh973078

[24] http://msdn.microsoft.com/en-us/library/hh973074.aspx

[25] http://msdn.microsoft.com/en-us/library/jj131025.aspx

[26] http://msdn.microsoft.com/en-us/library/jj131025.aspx#Active_User_Tracking

[27] https://decibel.ni.com/content/docs/DOC-20973

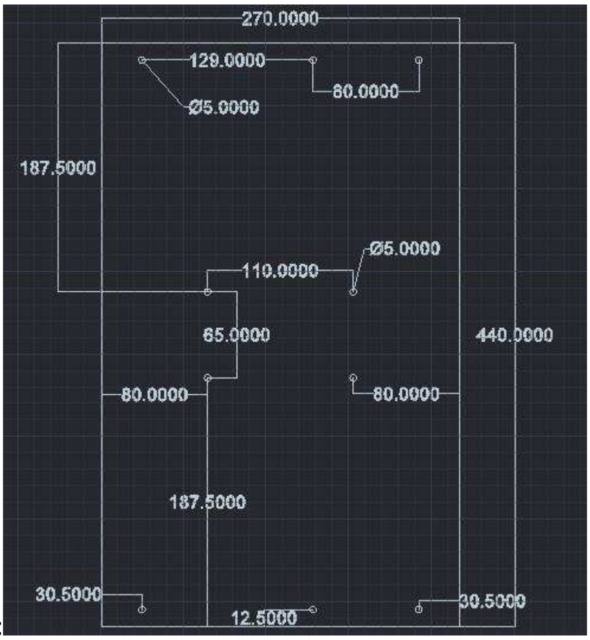
Appendix

Both appendixes A and B are provided in this section.

Appendix A

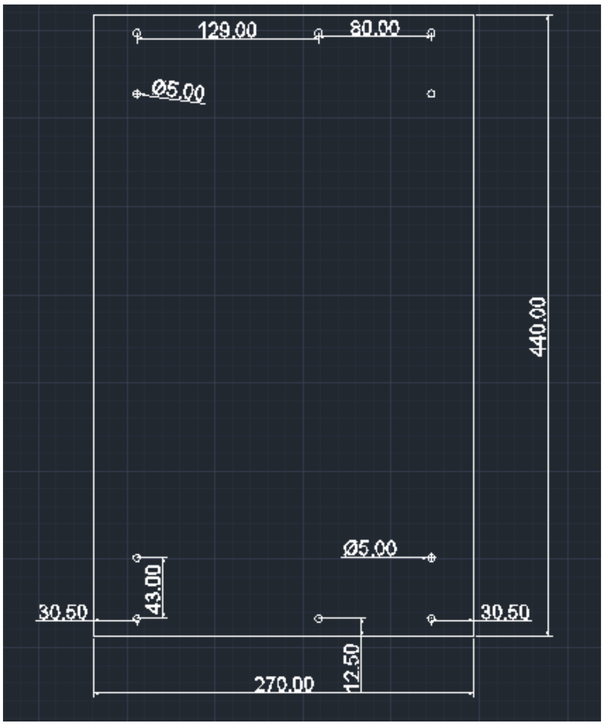
Below are the 2D drawings of the two sheets manufactured for the platform:

Upper Aluminium Sheet:



:

Lower Aluminium Sheet:

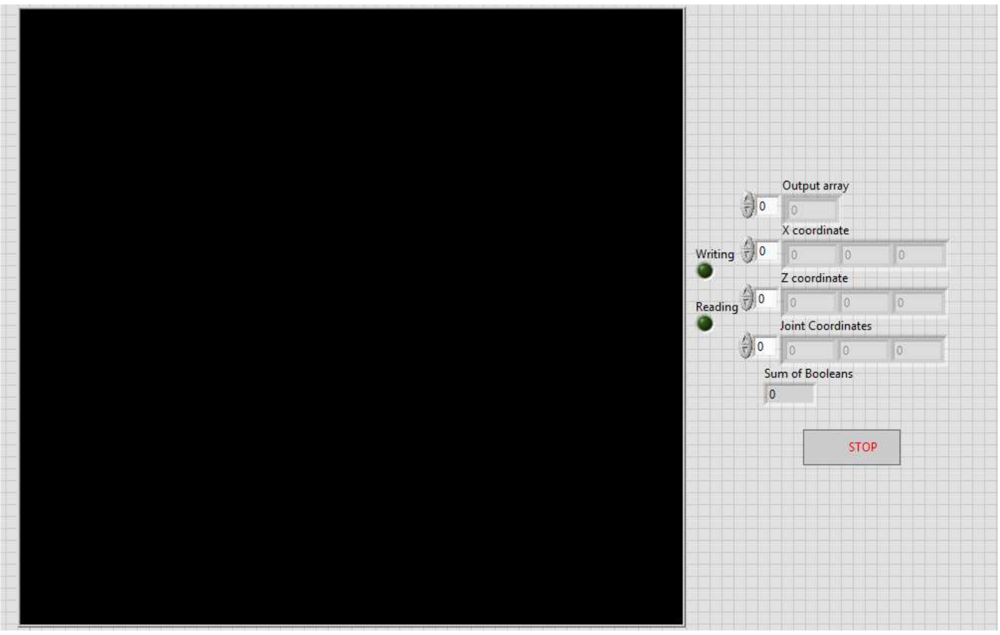


Appendix B

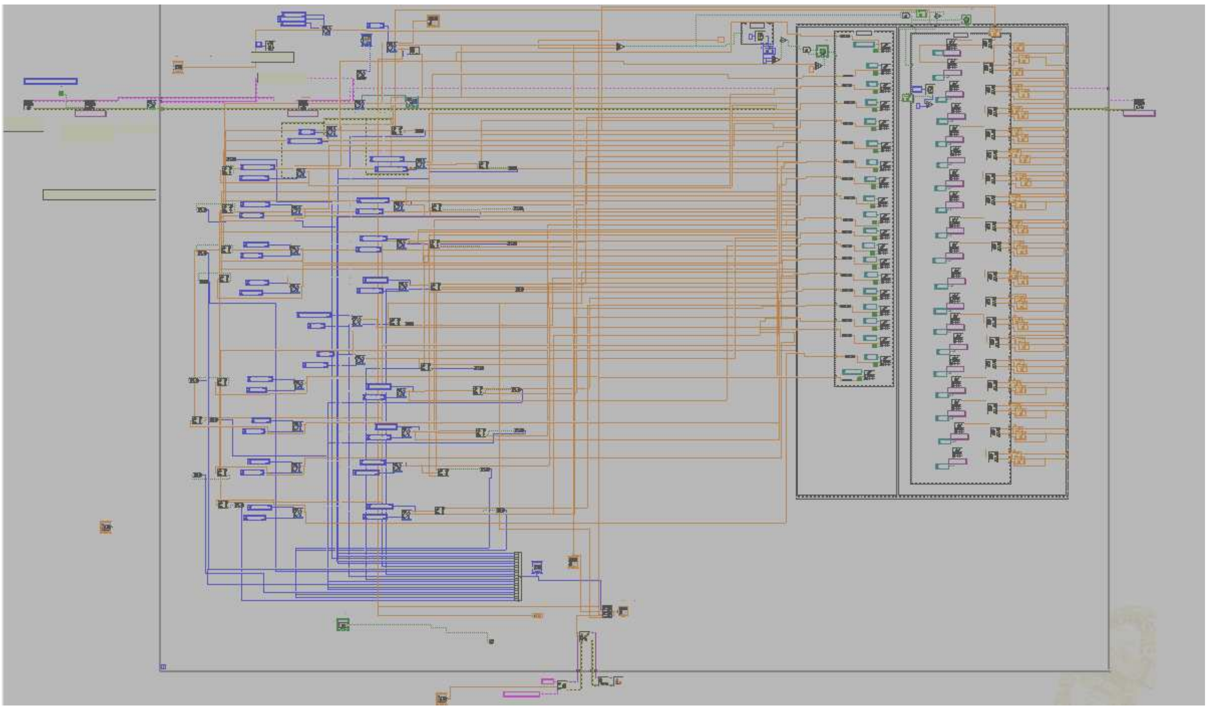
The screenshots of all of the VIs used in this project are provided in this section. Only some parts of certain VIs are shown because of their enormous sizes.

All of the processes involved in using Microsoft Kinect are processed in a single VI as shown below:

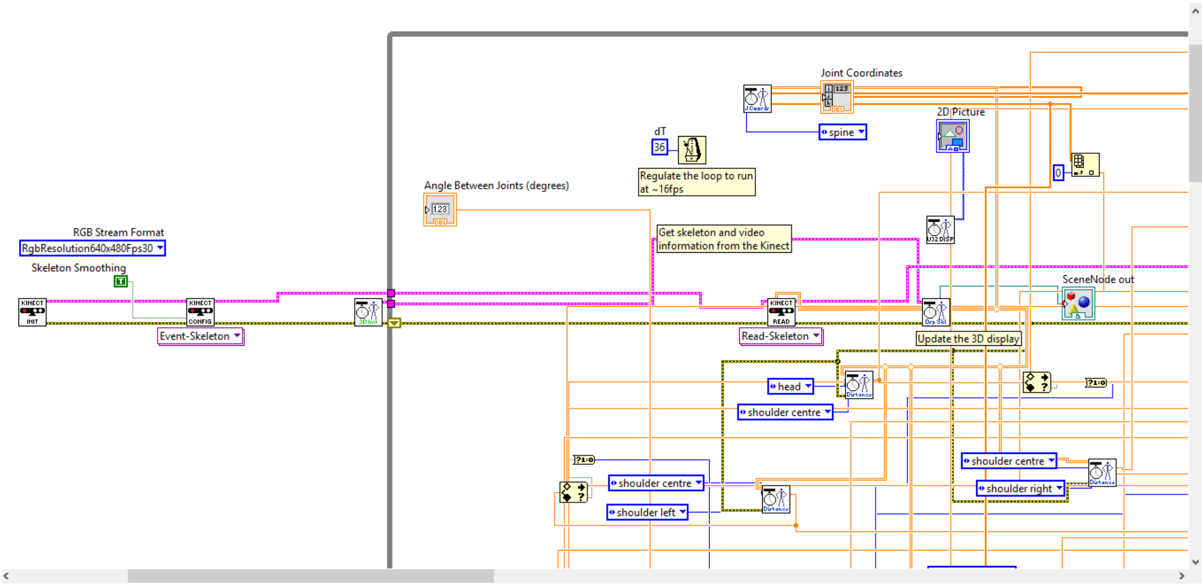
The Front Panel:



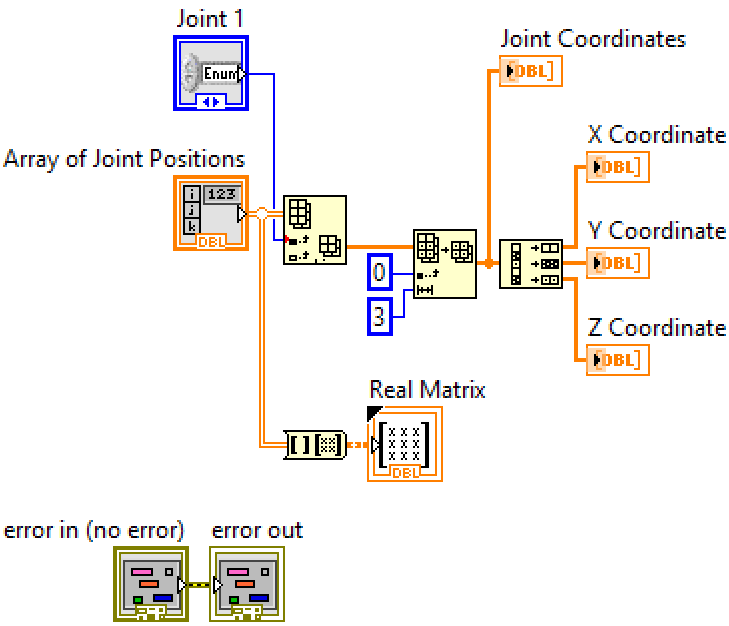
The Block Diagram of the whole VI:



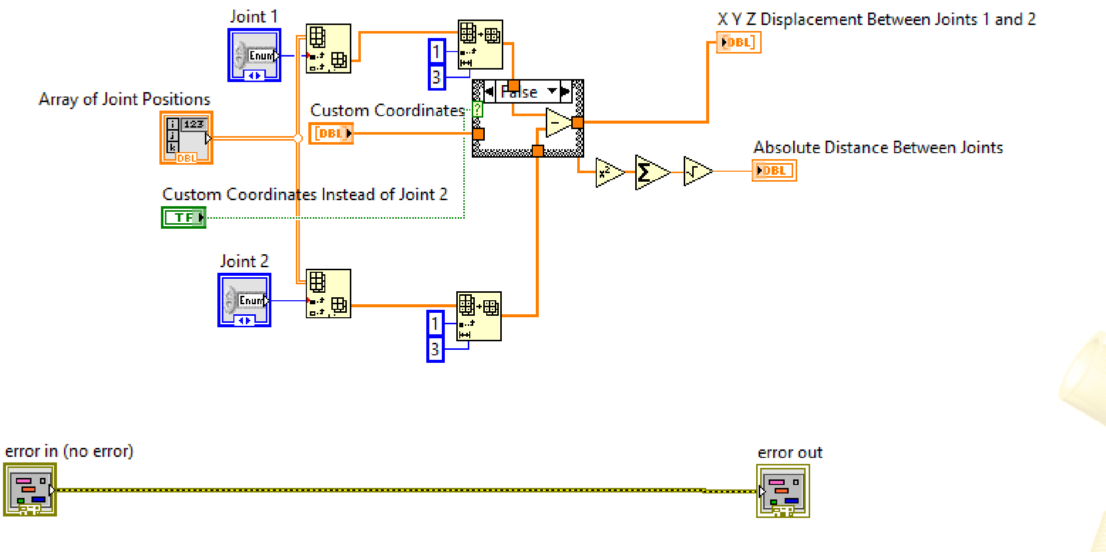
Initialization of Kinect and Extraction of Joint Coordinates



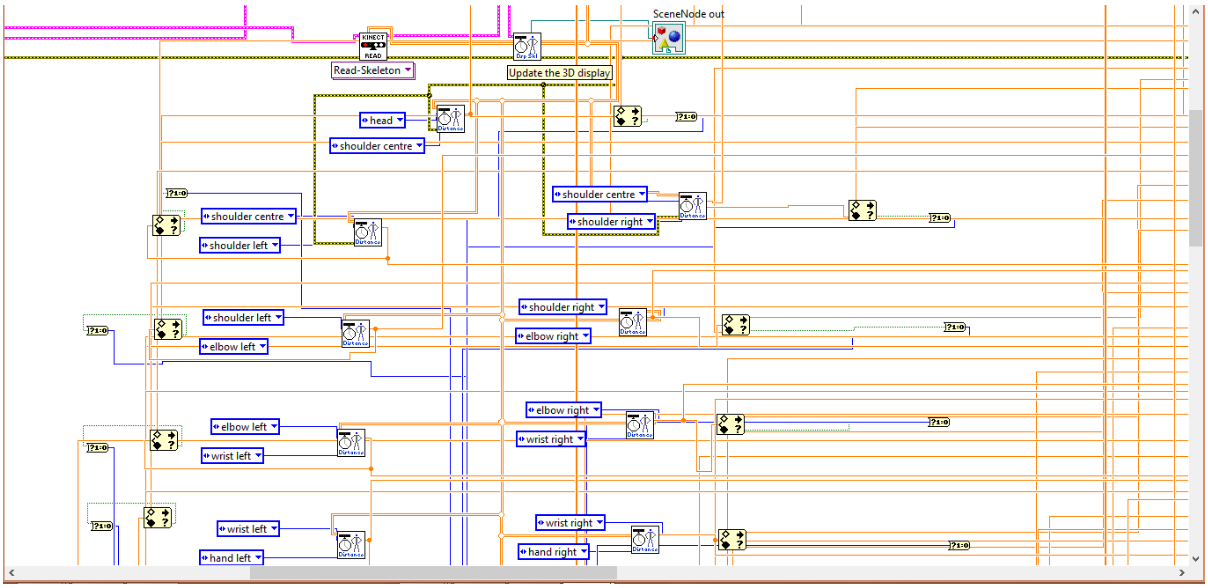
Modified Joint Coordinates VI



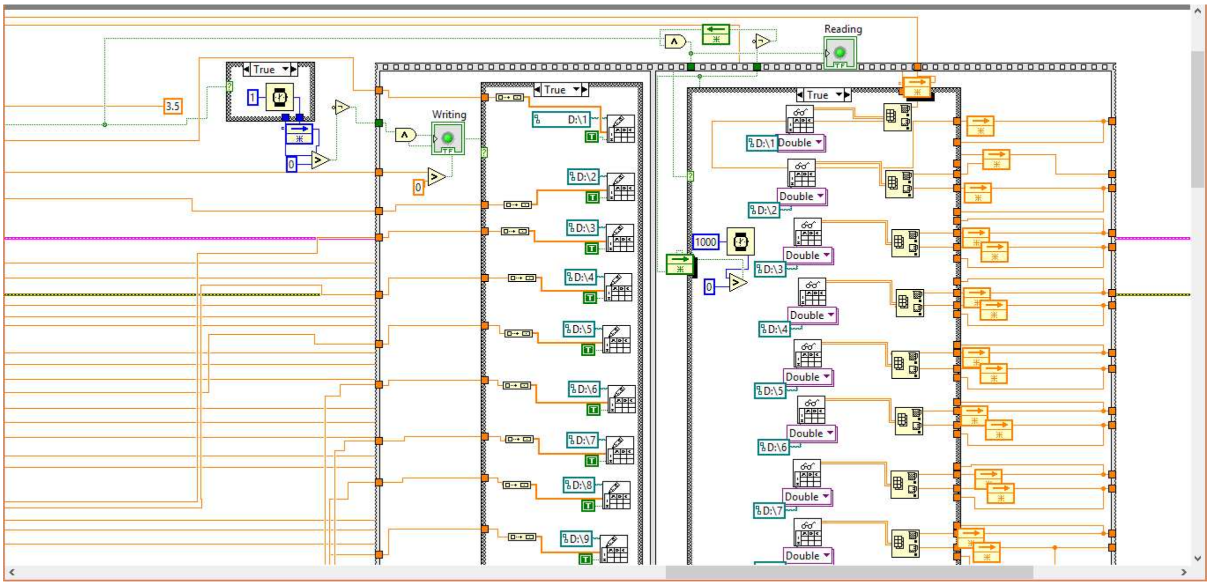
Original Absolute distance Between Joints VI



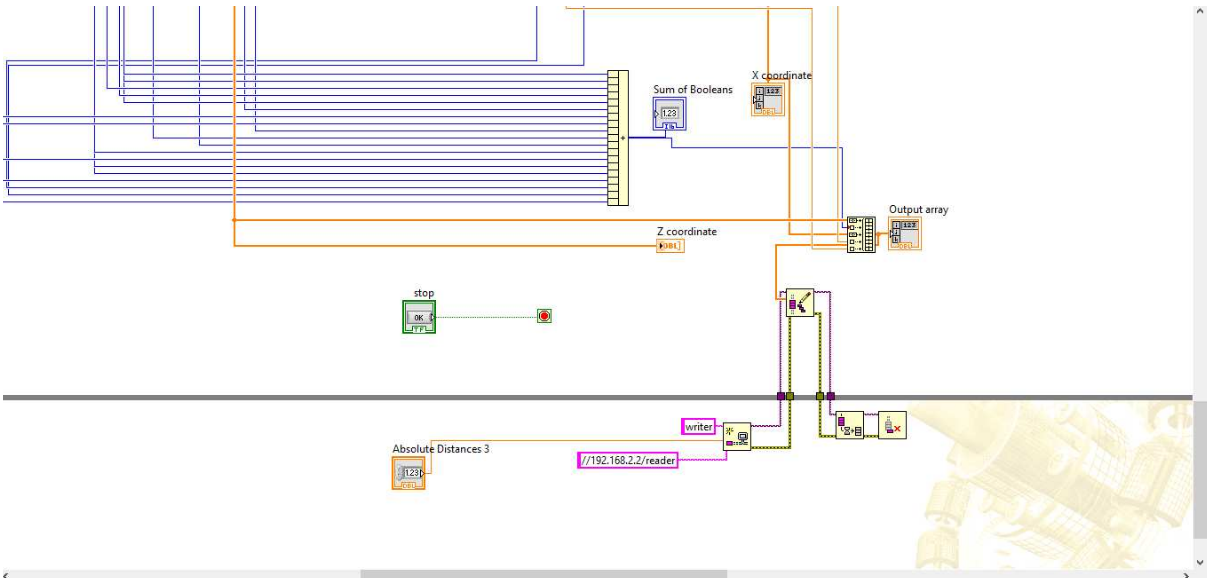
A part of Obtaining Absolute Distance Between Joints followed by Inrange Function



A Part of Data Acquisition from Kinect

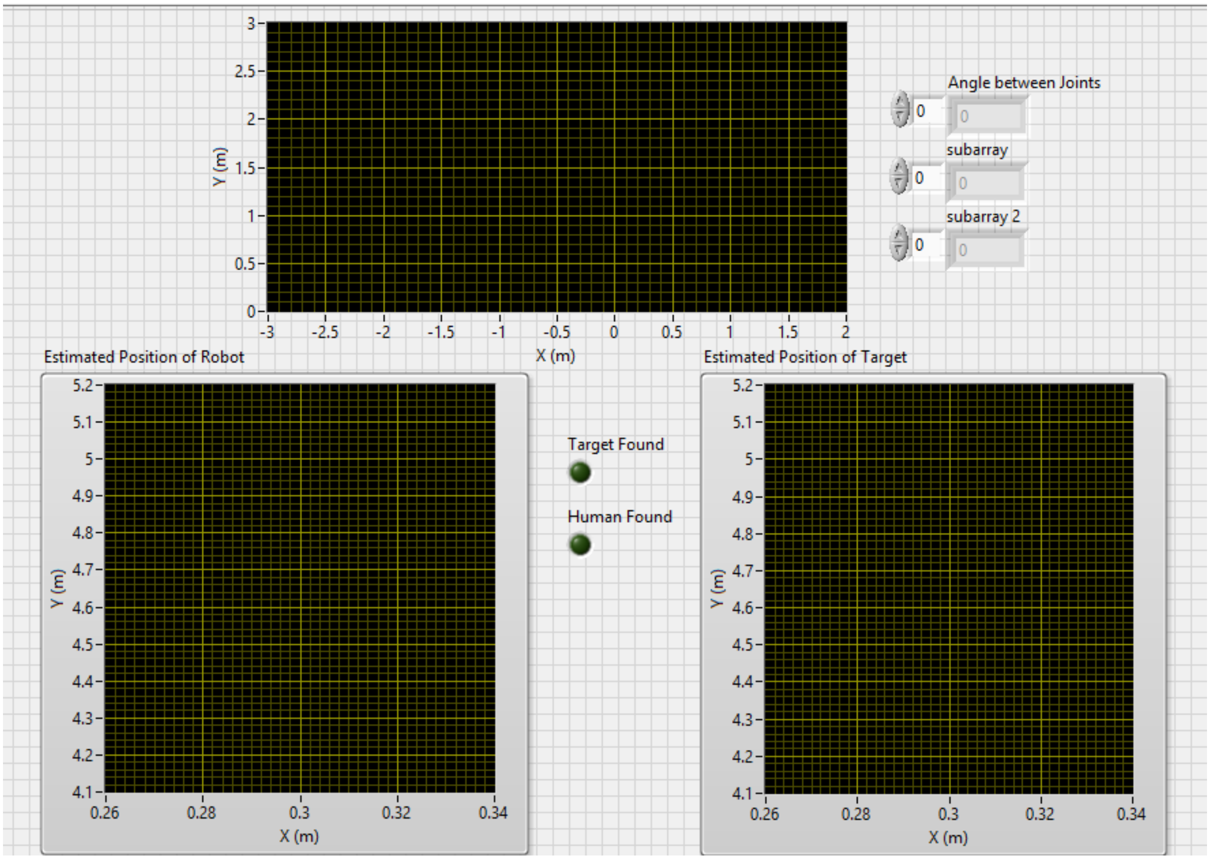


Obtaining Sum of Booleans and Writing to Network part of VI

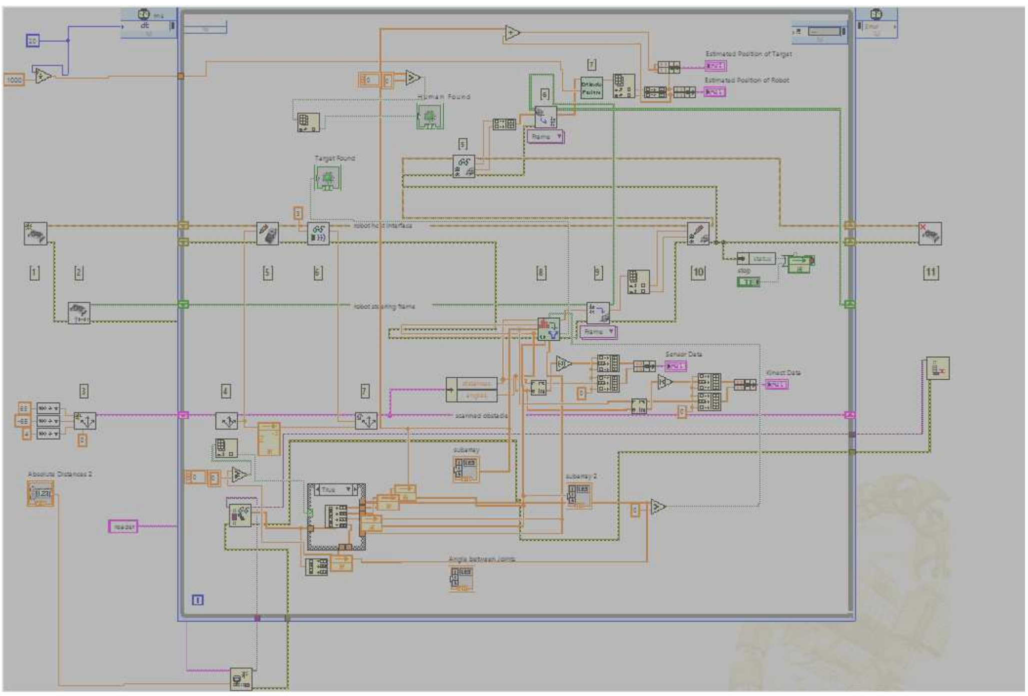


All of the processes involved in navigation of robot, including tracking and obstacle avoidance are as run in a single VI as shown below:

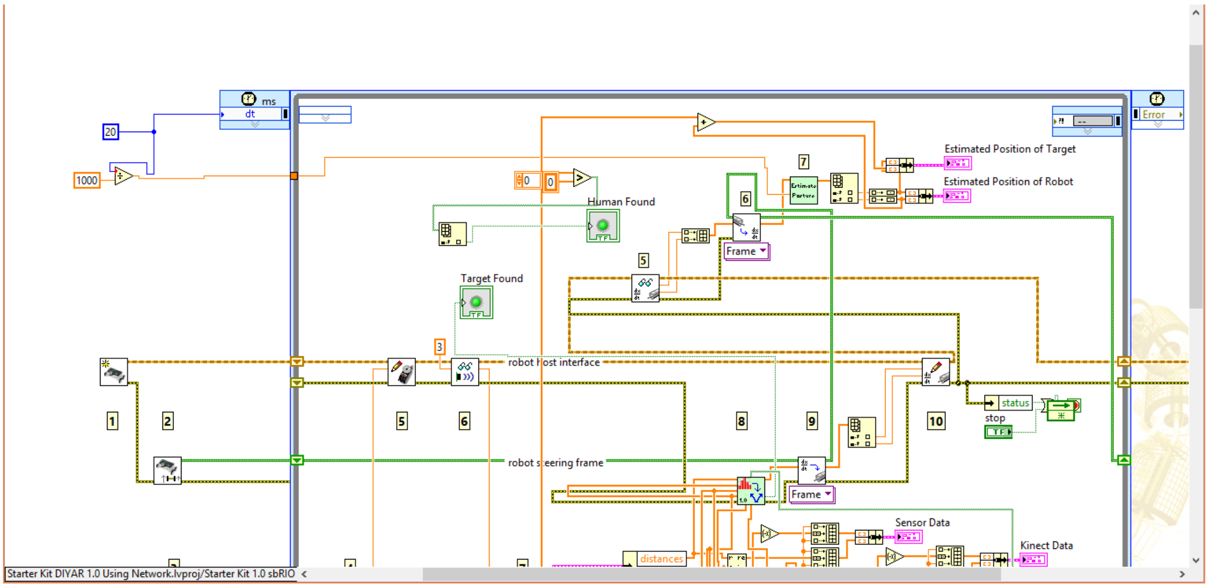
Front panel



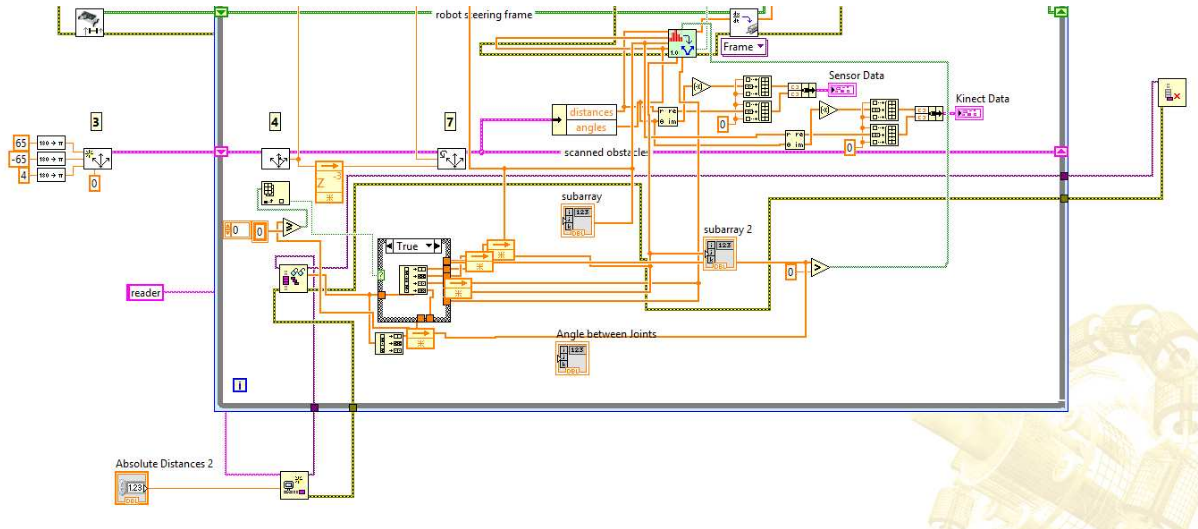
The Block Diagram of the whole VI:



A part of the VI showing the Initialization of DaNI robot as well as the Dead Reckoning Localization Method



A part of the Block diagram showing initialization of ultrasonic sensor and Reading from Network for the data to be fed to Calculate Driving Direction VI



Modified Calculate Driving Direction VI