Control of an Omnidirectional Walking Simulator

Manish Chauhan, Rajeevlochana C.G,

Subir Kumar Saha, S.P Singh

Department of Mechanical Engineering, Indian Institute of Technology Delhi, Hauz Khas, New Delhi – 110016, India.

[1]manish_me@student.iitd.ac.in

[2]rajeevlochana-me@student.iitd.ac.in

[3]saha@mech.iitd.ac.in

[4]singhsp@mech.iitd.ac.in

Abstract - Simulators are a unique way of replicating virtually, any real world scenario. It gives one the opportunity to be in a place virtually without being present there physically. The motivation behind making this simulator was to replicate real world terrains and make a human walk in that environment. The basic application which acted as motivation was training people like soldiers, sportspersons and others on various terrains. Control of a prototype is reported in this paper. The prototype is an Omnidirectional walking simulator that allows one to walk on it to get the feeling of walking on a plane in any direction.

Keywords - **Omnidirectional, walking simulator, inertial measurement unit, cable potentiometer, DAQ card, LabVIEW.**

1. INTRODUCTION

In the real world scenario, soldiers and players have to travel some distance before reaching the place of training. There are various places particularly, for soldiers, which are not easily accessible due to their geographical locations. This adds to unwanted physical strain on travelling, increase in cost and other unforeseen challenges. Though the real world situation is tough to replicate but our attempt was to replicate a terrain which is flat in nature without the temperature, pressure and other environmental conditions. In the present development, motion realization was first focussed. Later, a visual display was added to provide an immersed environment to the users. Such a simulator will help in reducing the cost of training the soldiers and players.

1.1 RELATED WORK

Walking simulators have been evolving over the years and they can be broadly classified [1] as 1) Pedaling devices, 2) Walk-in place systems, 3) Programmable foot platforms, 4) Sliding surface systems, 5) Planar treadmills etc. Further in planar treadmills, work has been carried out by Darkenet al.[2], Noma et al.[3], Iwata[4][5], Sughiara et al.[6], Hollerbach et al. [7]-[10], Huang et al.[11], Fernandes et al.[12], Luca et al.[13], Nagamori et al.[14], Templeman[15] et al., Zitzewitz et al.[16]

The proposed simulator prototype concept is based on the ATLAS (Noma et al. [3]) as shown in Fig. 1. The ATLAS (ATR Locomotion Interface for Active Self Motion) system comprises of an active treadmill which is installed on a motion platform with 3-axis rotation. The human walking velocity is tracked using infra-red (IR) sensors and accordingly the speed of treadmill is varied. Similarly rotation of the human is tracked and the yaw motion is given to the simulator.

Due to its simplicity, the concept is adopted for the development of our Omnidirectional walking simulator, as shown in Fig. 2.

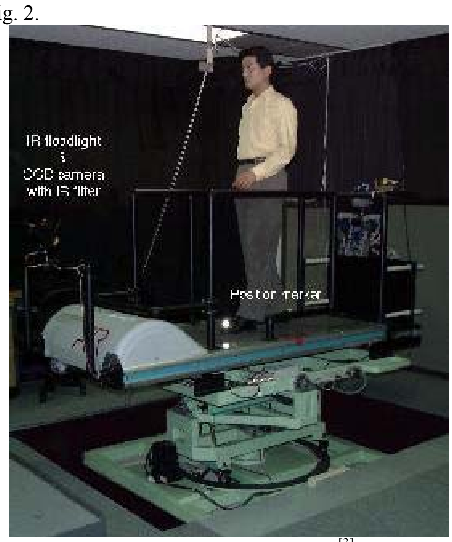
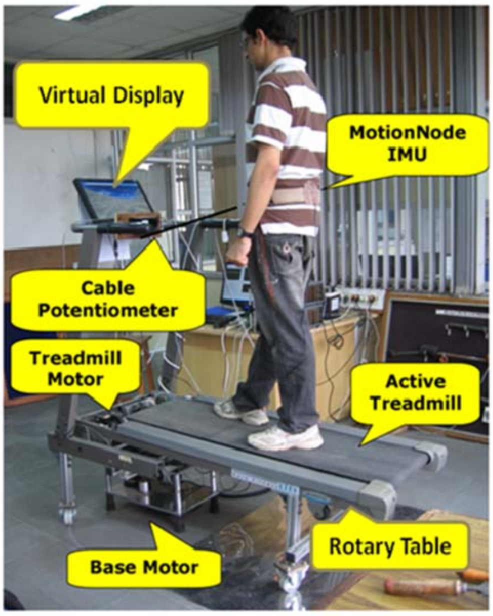


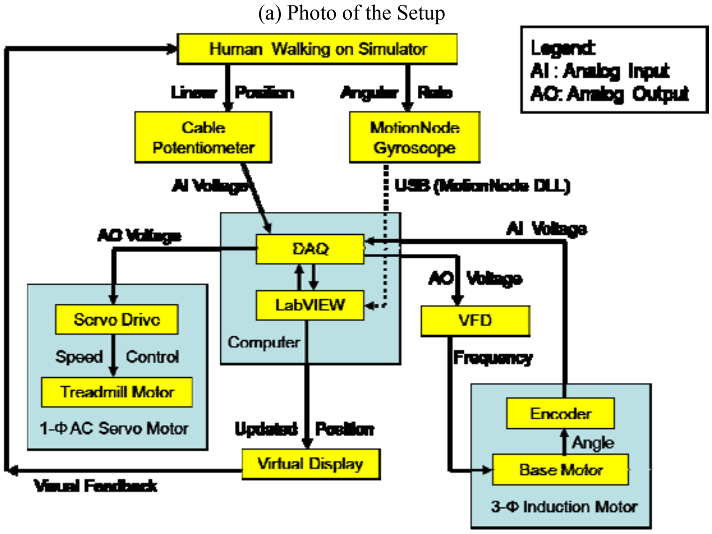
Fig. 1: ATLAS (Noma et al.[3])

1.2 PROPOSED SIMULATOR

The proposed Omnidirectional walking simulator as shown in Fig. 2(a) consists of an "*Active Treadmill*" mounted on a "*Rotary Table*." The motors driving both the treadmill and rotary table (base motor) are integrated with a host computer so that they can be independently controlled. A "*Visual Display*" (TFT screen) is mounted on the front side of the treadmill so that the walking human can look at it all the time. The display is connected to the host computer to provide appropriate scenes based on the movement of the human. A pre-defined terrain is displayed to the human and he/she takes a decision accordingly at the start of walking on the treadmill. The setup layout is shown in Fig. 2(b).



(a) Photo of the Setup



(b) Setup Layout

Fig. 2: Proposed Omnidirectional Walking Simulator

To track the motion of the human, two devices are used viz,

1). A *cable potentiometer* which is mounted on the front side of the treadmill and its cable end is attached to the human's waist. It acts as a voltage divider circuit depending on the length of extension of the cable, which is directly proportional to the relative position of the walker on the treadmill. The voltage signal acquired by the computer can be used to take control decision of increasing, decreasing or no change of the speed of the treadmill motor.

2) *MotionNode* device that has a *gyroscope* and it is put on the waist of the human. When the human takes a turn, gyroscope gives the angular rate about its vertical axis and the signal is used to rotate the base motor so that the human continues to walk on the treadmill. The updated position and the angle are sent to the virtual environment and it updates itself, thus completing the loop.

2. CONTROL OF THE SIMULATOR

As Fig. 2 briefly shows the basic components of the walking simulator setup, the following items discuss the LabVIEW programs to control the system:

1) *Control of treadmill and base motors*

The formulas for various trajectories are given in Table 1. The corresponding callout 1 is shown in Fig. 3. The sampling frequency, amplitude of the signal can be controlled from callout 2 of the block diagram. The output signal from callout 2 is sent to the treadmill motor via DAQ (*Data Acquisition*) card (*PCI 6221*) as shown in callout 3 of Fig. 3. Figure 4 shows the screenshot of the LabVIEW window when the trajectory control program was run

Table . Trajectories used in LabVIEW

TYPE	EQUATION
Sinusoidal wave	A*sin(ωt)
Noise wave	A*sin(ωt)*(3-sin(10πt)
Noise wave	A*sin(ωt)+2(rand(-1)-0.5)
Steadily increasing wave	5t/fs
Cosine wave	A*cos(ωt)

A:Amplitude; ω: Angular velocity; t: Time period; fs: Samples/second.

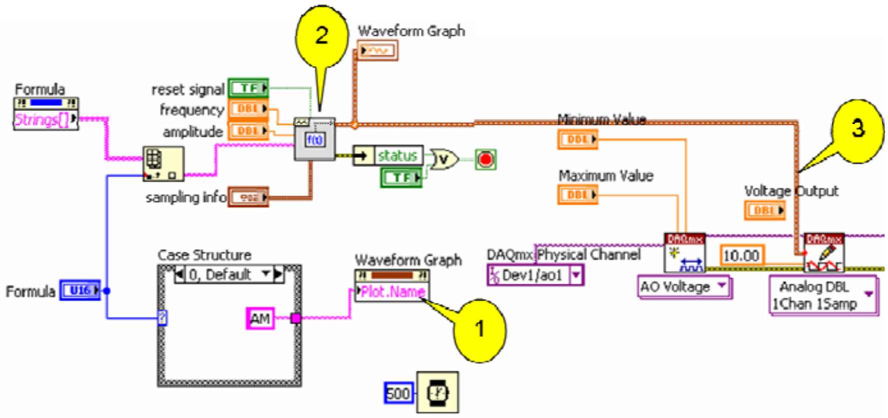


Fig. 3: LabVIEW program for trajectory control

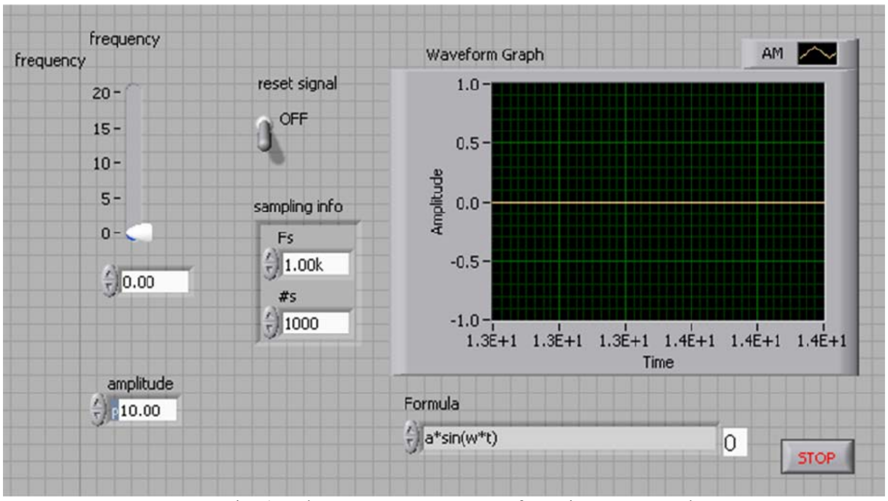


Fig. 4: LabVIEW program output for trajectory control

2) *Calibration of treadmill belt speed*

In order to obtain suitable control of command function for the treadmill belt motor which is a *single phase ac servo motor*, it was necessary to calibrate it. To perform the calibration, voltage was increased from 0 to 10V and corresponding value of treadmill angular speed was recorded using a tachometer. Similar procedure was followed by reducing the voltage values from 10V to 0V and again finding the treadmill angular speed.

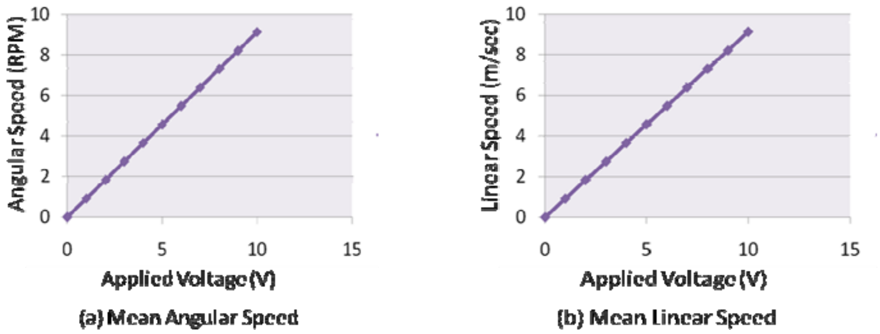


Fig. 5: Calibrated graph for Treadmill belt

Mean angular speed of the treadmill motor vs. voltage, which was calculated from the speeds due to increasing and decreasing voltages are plotted in Fig. 5(a). Corresponding mean linear speeds of the treadmill belt vs. Voltage are shown in Fig. 5(b). Both the plots clearly show a linear behaviour. Hence, the speeds of belt are directly proportional to voltage.

3) *Calibration of base motor*

Similar to the calibration steps of the belt motor, the treadmill base motor, which is a 3-phase induction motor controlled through a *variable frequency drive* (VFD), was made to turn in both clockwise and anti-clockwise direction without a person standing over it. For this voltage was varied within the range of 0 to 5V. Three trials of the experiment were performed for clockwise and anticlockwise direction rotation of 90 degrees and corresponding graphs were plotted to know the behaviours. They are shown in Figs 6 and 7. From Fig. 6 we can notice that for varying voltages on horizontal axis in the range of -5 to +5V, there is almost linear change in rotational speed of the base motor. From this plot, we were able to calculate the slope of the curve as 5.35degree/second/voltage i.e., for every voltage change supplied to the base motor, we find a change in angular speed as 5.35 degrees/second.

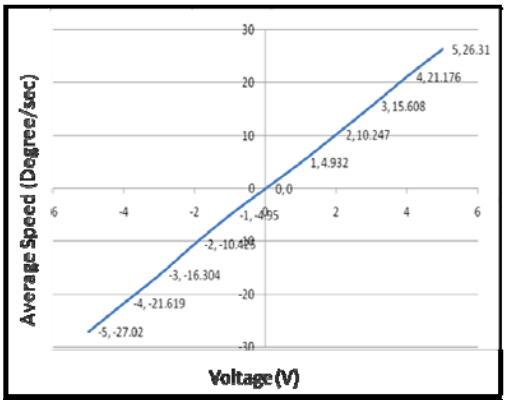


Fig. 6: Calibration of base motor without a person

The treadmill was then made to turn in both clockwise and anti-clockwise direction with a person standing over it whose average weight was 75kg. For this, voltage was also varied within the range of 0 to 5V. Three trials of the experiment were repeated as before.

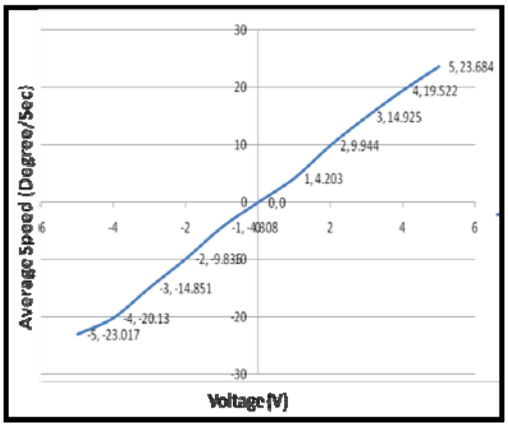


Fig. 7: Calibration of base motor with a person

Fig. 7 shows the variation of angular speed vs. Voltage. It is also almost a linear behaviour, for which the slope is 4.58degree/second/voltage, i.e., for every voltage change supplied to the base motor, we find an angular speed variation of 4.58 degrees/second.

4) *Feedback data from MotionNode sensor*

In order to get the rotation signals from *MotionNode sensor* which the user was wearing on his/her torso, a *digital link library* (.dll) corresponding to the sensor mode of the MotionNode device was called and the signal obtained from the gyroscope is as shown in callout 1 of Fig. 8. As the input signal had a lot of noise signal, it was passed through a Bessel’s filter and the filtered value is as shown in callout 2 which was then sent to a formula node. A scaling factor which relates MotionNode signal to the voltage variations of the base motor was multiplied and sent to base motor as shown in callout 3 of Fig. 8.

As it is already discussed, there is a linear relationship between voltage and angular speed of the base motor. If an approximation of slope is done, it leads to a value of 5 degree/second/voltage. Hence the corresponding voltage signal coming out of MotionNode is fed to the base motor.

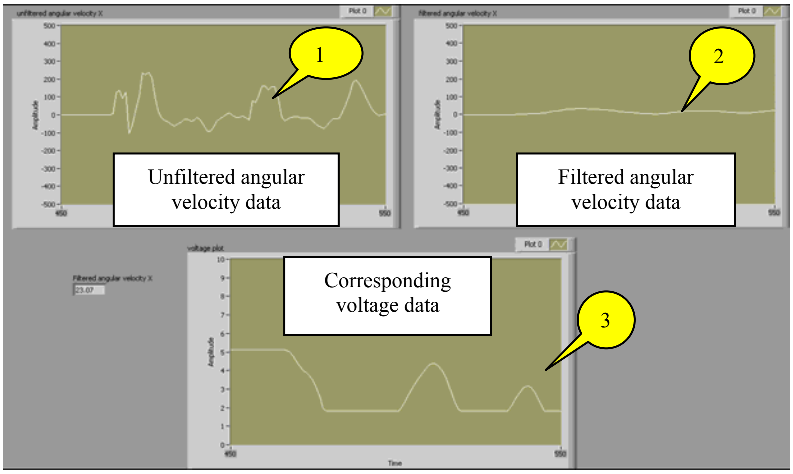


Fig. 8: Integration of MotionNode with base motor

Next, to get the *encoder* feedback a DAQ assistant is setup as shown in Fig. 9 (callout 1). Encoder feedback is acquired from Analog input channels on DAQ card which are shown in callout 2. These analog input channels correspond to channel's A and B of the encoder feedback which are fed into the Formula Node of the Block Diagram as shown in callout 3. In Formula Node, the program is written to count the feedback signals of A and B channels in both clockwise and anti-clockwise directions of rotation for the base motor. After finding the input channels A and B, the degree of rotation is calculated. Angular velocity is determined in callout 4.

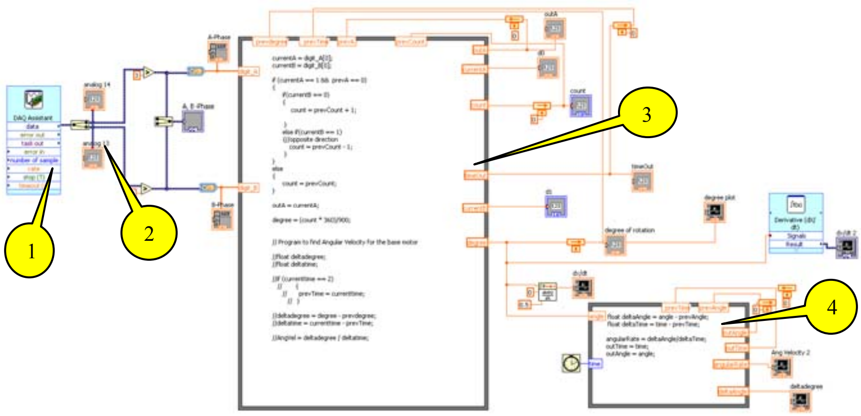


Fig. 9: Block diagram for base motor encoder feedback

5) *Control of belt speed using cable potentiometer signal.*

After successful control of the base motor, the idea was to control the speed of the treadmill according to the variation in speed of the user. If the user accelerates or decelerates on the treadmill, the treadmill should respond accordingly and re-centre the user in the centre of the platform to avoid him/her or her to fall down. In order to achieve the above mentioned task, a cable potentiometer was fitted on the treadmill.

Note, in Fig. 10, a DAQ assistant is setup to acquire cable potentiometer signal via *DAQ Card* as shown in callout 1. The acquired signal is fed into formula node at callout 2. Here a corresponding voltage scale is setup for cable potentiometer signal which is fed to the treadmill motor. The code tracks the acceleration and deceleration of user on the treadmill. If the user accelerates, then he/she approaches the front of the treadmill and correspondingly voltage reduces. This change in voltage is tracked and treadmill belt speed is increased. Similarly when deceleration occurs, then the voltage increases, and the belt speed reduces. There is range of voltage calibrated such that if the user wants to walk at constant speed, then he/she is kept on the middle of the treadmill. Figure 11 shows the output of the program.

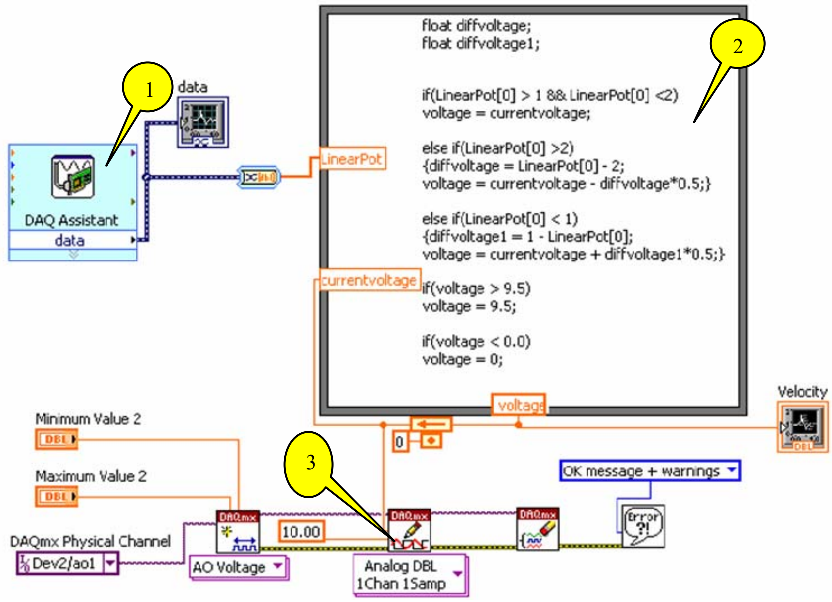


Fig. 10: Block Diagram for treadmill belt control using Cable Potentiometer

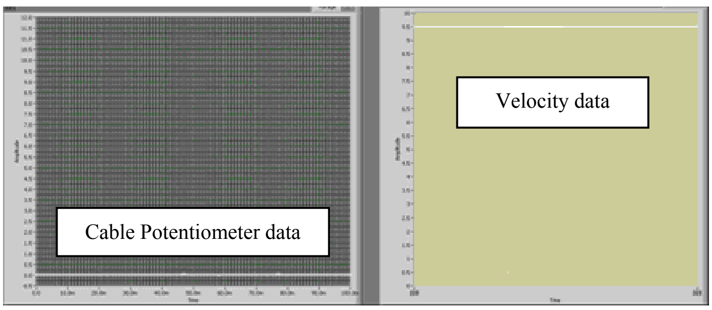


Fig. 11: Output of the treadmill belt speed using Cable Potentiometer

6) *Visual Feedback using TrueVision3D*

TrueVision3D [17] is a *3D Software Development Kit* (SDK), using which a 3D virtual environment can be developed. A *VisualC#* application was developed on *VisualStudio* and *TrueVision3D* which displayed a 3D virtual environment as shown in Fig. 12. It loads up a terrain (in this case an 8 shaped route) and the position of the camera/person and the look-at angle of the camera have to be updated to move around in the virtual environment. The application has two properties which can be set to current position (position X and position Z) of the camera. Position can also be considered to include the altitude of the terrain as well. To move around in the environment, the current position needs to be set and the direction of viewing (look-at angle) can be determined from the vector from previous position to current position. The application has a 25 millisecond timer which reads the current position properties and updates the virtual environment. The program which was built using Visual Studio produces a *Dynamic Linked Library* (.dll) file which needs to be called from a LabView program.

A LabView program was developed as shown in Fig. 13. The working is as follows:

- Create an instance of LabViewTrueVisionLink program (from the above .dll) using .Net Constructor Node as shown in callout 1. A reference to the above C# application is established and passed on inside the while loop. A windows form is displayed as shown in Fig. 12.

- The while loop runs at a time interval of 25 milliseconds and the input data is fed to the formula node (callout 2).

- Formula node determines the position X and position Z programmatically and sets the current position properties in the C# program, which keeps updating the view as per the input sent from LabView. The X and Z values in the formula node get the points on two circles which form an 8 numbered shape.

- Once the LabView program is ended, free the C# program (callout 3).

Fig. 12: 3D Virtual Environment

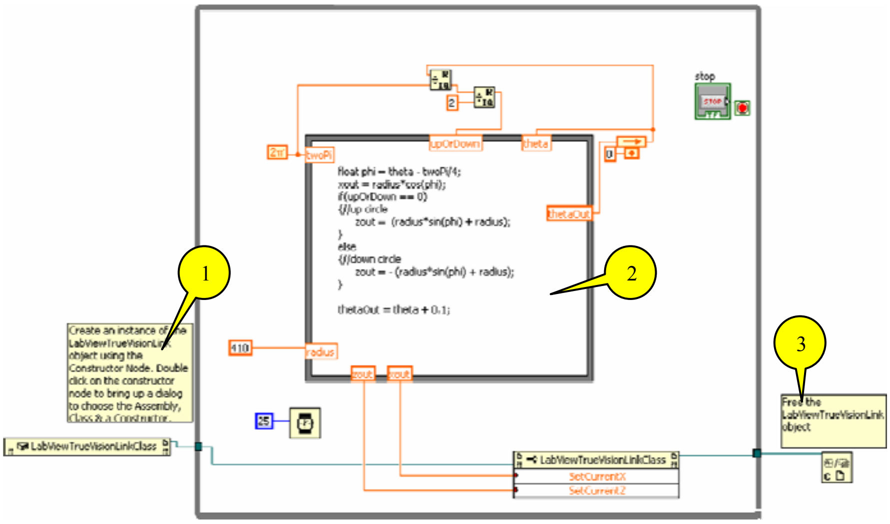


Fig. 13: Block Diagram to integrate 3D virtual Environment

3. CONTROL INTEGRATION

The control algorithm of the whole simulator was built in stages of modules and it was the final task to run all the programs simultaneously and integrate all its functions. Figure 14 shows the integration of various LabVIEW programs which are discussed individually in the previous section. Callout 1 shows the *DAQ assistant* which acquires signal for base motor encoder feedback. Cable potentiometer signal is obtained from DAQ assistant as shown by callout 2. The callout 3 shows the formula node for the base motor encoder feedback. Callout 4 shows the block diagram for the visual feedback screen fixed on the treadmill.

In order to test the functions of the simulator, the following experiments were performed:

1. Acquiring signal from MotionNode sensor in a LabVIEW program.
2. Controlling the base motor or yaw motor with the help of MotionNode sensor.
3. Acquiring and analyzing signals from cable potentiometer sensor
4. Controlling the belt speed according to acceleration and deceleration of the user on the treadmill.
5. Creating visual scenario for the user walking on the treadmill.
6. Interacting with the visual screen and making the simulator respond accordingly.
7. Experiments were conducted to control the belt speed with the help of MotionNode sensor.

The above experiments showed that the Omnidirectional walking simulator functioned properly, i.e., providing necessary abilities of walking in any direction without physically moving away from the centre of the treadmill.

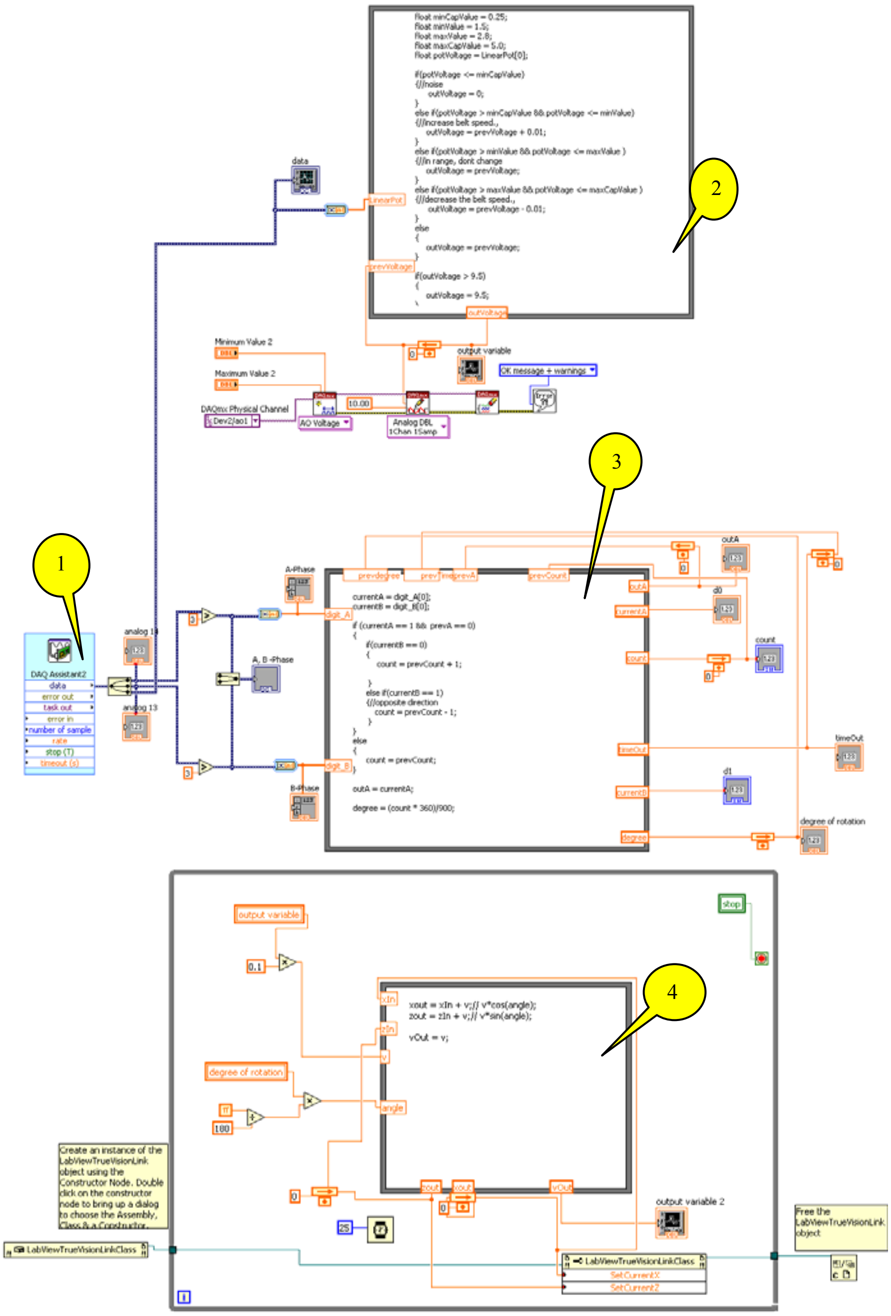


Fig. 14: Block diagram for System Integration

4 CONCLUSIONS

Simple but useful LabVIEW based functionalities were incorporated in a hardware set-up of an Omnidirectional walking simulator conceived and developed at Mechatronics Lab, IIT Delhi. It is a first of its kind in India and we hope that the device will boost the simulator R&D effort and the relevant industrial activities.

5 ACKNOWLEDGEMENT

The authors sincerely thank Mr. Vinay Khemka, an ex-M.Tech student, Mr. Hemant Kumar, Mr. Nitesh Garg, two ex-B.Tech students, for their efforts in developing the hardware set-up for the Omnidirectional walking simulator under the project entitled, “Omni-directional Advanced Simulator for Speed Walking”.

6 REFERENCES

[1] Khemka, V.K., 2009, “Conceptual Design of Omnidirectional Walking Simulator”, M.Tech Thesis, Mechanical Engineering Department, IIT Delhi.

[2] Darken, R.P., Cockayne, W.R., Carmein, D. (1997), “The Omnidirectional Treadmill: A Locomotion Device for Virtual Worlds”, Proc. Of UIST’97, pp. 213-221

[3] Noma,H. and Miyasato,T., (1998): "Design for Locomotion Interface in a Large Scale Virtual Environment", *Proc. of VRSJ’98*, pp. 185 – 188

[4] Iwata, H. (1999), “The Torus Treadmill: Realizing Locomotion in VEs,” IEEE Computer Graphics and Applications, Vol.9, No.6, 30-35.

[5] Iwata, H., Yano, H., and Nakaizumi, F. (2001),“Gait Master: A Versatile Locomotion Interface for Uneven Virtual Terrain,” Proceedings of IEEE Virtual Reality 2001 Conference, 131-137

[6] Noma, H., Sugihara, T., Miyasato, T. (2000), “Development of ground surface simulator for tel-E-merge system,” Proc. IEEE Virtual Reality 2000, New Brunswick, NJ, pp. 217-224.

[7] Hollerbach, J.M., Xu, Y., Christensen, R., and Jacobsen, S.C. (2000), `` Design specifications for the second generation Sarcos Treadport locomotion interface," Haptics Symposium, Proc. ASME Dynamic Systems and Control Division, DSC-Vol. 69-2, Orlando, pp. 1293-1298.

[8] Hollerbach, J. M. (2002), Locomotion Interfaces. In K. Stanney (ed.), *Handbook of Virtual Environments: Design, Implementation and Applications* (pp.239-254). Mahwah, NJ: Lawrence Erlbaum.

[9] Hollerbach, J.M. "Locomotion interfaces", http://www.cs.utah.edu/~jmh/Locomotion.

[10] Hollerbach, J.M., Grow, D., Parker, C. (2005),” Developments in Locomotion interface, “ IEEE 9th International Conference on Rehabilitation Robotics, Chicago, IL,USA

[11] Huang, J. Y., (2003),” An omnidirectional stroll-based virtual reality interface and its application on overhead crane training,” *IEEE Transaction on Multimedia, 5*(1), 39-51.

[12] Fernandes, K.J., Raja, V., and Eyre, J. (2003), “Cybersphere: the fully immersive spherical projection system,” Commun. ACM,46(9):141-146

[13] Luca, A. De, Mattone, R., and Giordano, P. R. (2006), “The motion control problem for the CyberCarpet,” in Proc. 2006 IEEE Int. Conf. on Robotic and Automation,(Orlando,FL)

[14] Nagamori, A.K., Wakabayashi, K., and Ito. M. (2005), “The ball array treadmill: A locomotion inter-face for virtual worlds,” in Workshop on New Directions in 3D User Interfaces (at VR 2005). Bonn, Germany

[15] Templeman, J. N., Denbrook, P. S., and Sibert L. E. (1999), "Virtual locomotion: Walking in place through virtual environments," *Presence-Teleoperators and Virtual Environments,* 8(6), 598-617.

[16] Zitzewitz, J.V., Bernhardt, M., and Riener, R. (2007), "A Novel Method for Automatic Treadmill Speed Adaptation," IEEE Transactions on neutral systems and rehabilitations engineering, Vol.15, NO. 3, 2007

[17] TrueVision3D SDK http://www.truevision3d.com