Tech. Rep. IRIS-02-413
Institute for Robotics and Intelligent Systems, USC, 2002

Speed Control of a Pneumatic Monopod using a Neural Network

Kale Harbick and Gaurav S. Sukhatme

kale|gaurav@robotics.usc.edu

Robotic Embedded Systems Laboratory
Robotics Research Laboratories
Department of Computer Science
University of Southern California
Los Angeles, CA 90089-0781

Abstract

We discuss a speed controller for a hopping robot with a pneumatically powered leg. The controller uses a neural network to model the neutral point as a function of running speed and hopping height. The network is trained off-line using training data taken from a simulated hopper that is manually controlled by a human. Simulation experiments of hopping in the sagittal plane show improved performance over a Raibert PD controller, which uses a linear approximation for the neutral point.

I. Introduction

The benefits of legged systems over other forms of terrestrial locomotion are obvious; they can navigate obstacles which wheeled and treaded vehicles cannot. Dynamically-stable legged systems have advantages over statically-stable legged systems. Running machines can travel faster and can navigate terrain which has points of support that are spaced too far apart for walking machines to reach. Even for the simplest running machine, a one-legged hopper, research problems have not been completely explored.

Most running machines constructed thus far have served as little more than existence proofs of different forms of locomotion. We seek to contruct a running machine that builds on basic locomotion with additional behaviors that would be useful in a more functional robotic system. To this end we have studied ways to improve the traditional Raibert three-part control system [1].

The hopping machine considered in this paper is like a pogo-stick, except it uses a pneumatic rather than mechanical spring. The plant's motion is restricted to the sagittal plane. A diagram of the hopper model is shown in Figure 1. Table I defines the state variables and Table II defines the physical parameters of the hopper. The leg stroke, hip offset, and mass parameters are similar to those used by Raibert for his planar hopper [1].

Our controller is based on Raibert's three-part control system [1]. This method decomposes the control into three separate control loops: forward velocity control, body attitude control, and height control. The experiments reported in this paper use a model-based height controller [2], while the body attitude controller is unchanged. The speed controllers are described in Section III.

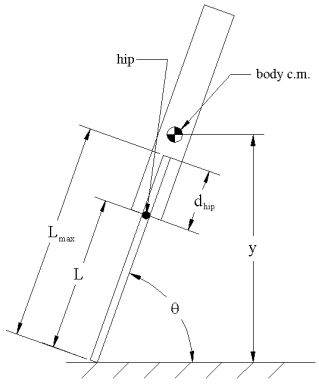


Fig. 1. 2D hopping machine in stance. The body is not shown.

We report results of speed regulation experiments, in which the desired forward speed was changed in a step-wise fashion while maintaining a fixed apex height. Simulations show that a neural network-based speed controller performs well over a wider range of desired speeds than the Raibert speed controller.

The paper is organized as follows. Related work is discussed in Section II. Section III describes the two speed controllers. Experimental results are presented in Section IV. Conclusions and future work are discussed in Section V.

II. Related Work

A three-part control system has been used to control monopod hopping robots [1]. This system used two proportional controllers to control forward speed. This controller is described in further detail in Section III.

A similar approach modifies the neutral point approximation with an additional factor of the ratio of leg length to rest

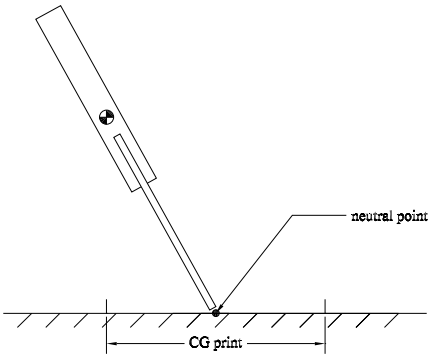


Fig. 2. The neutral point is approximated using an estimation of the CG print.

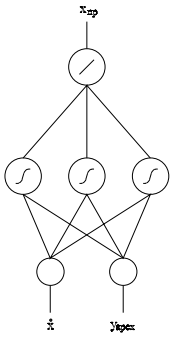


Fig. 3. Neural network.

length [3].

A discrete closed form trajectory model was used to control a bow leg hopper [4]. Model parameters were experimentally determined with a least squares fit to a set of recorded trajectories.

A simulation of a vertical hopping machine was controlled using a near-inverse discrete-time model [5]. Time-varying or unknown parameters were estimated with a recursive least-squares parameter estimator.

Other statistical learning methods such as RMRC have been used to learn inverse kinematic mappings in real time [6].

III. Speed Controllers

A. Raibert Controller

Raibert describes the neutral point of a hop as the point where the foot should be placed which will result in zero net acceleration [1]. This point is a function of hopping height and forward speed. Raibert used an approximation of the CG print, i.e. the locus of points on the ground over which the body center of mass travels during stance, to estimate the neutral point. This is represented in Figure 2.

He approximated the length of the CG print as the stance duration times the forward running speed, as in Equation 1.

$$x_f = \frac{\dot{x} T_s}{2} + k_{\dot{x}} \left(\dot{x} - \dot{x}_d \right). \tag{1}$$

This foot position is converted into a desired leg angle w.r.t. the body

$$\gamma_d = \phi - \arcsin \frac{x_f}{L}.$$

Finally a second PD loop is used to calculate the hip torque

$$\tau = -k_p(\gamma - \gamma_d) - k_d(\dot{\gamma}).$$

We used the following values for gains: $k_{\dot{x}} = 0.03$, $k_p = 47.0$, $k_d = 1.26$.

B. Neural Network-Based Controller

One problem with the Raibert speed controller, as he points out, is that the neutral point is nearly linear with forward speed only up to a certain velocity. This point varies with the physical parameters of the system, but for the plant described in this paper, it is approximately 0.7 m/s. To overcome this difficulty, we used a neural network to more closely represent the neutral point function.

The original controller is modified to replace the first term in Equation 1 with a neural network output. The second term is left unchanged. The network is shown in Figure 3. The hidden layer uses sigmoidal tangent activation functions and the output layer is purely linear.

Training data was generated by a human operator controlling the foot position at varying speeds and apex heights. The simulation recorded neutral point values for three different apex heights (0.4 m, 0.5 m, and 0.6 m) and seven different speeds (0.4 m/s to 1.6 m/s at 0.2 m/s intervals), for a total of 21 training points. The neural network was trained off-line using Levenberg-Marquardt optimization [7] [8].

Figure 4 shows a surface plot of the output of the trained network given apex values from 0.4 m to 0.65 m and speed values from -2.0 m/s to 2.0 m/s.

IV. Experiments

The simulations were started with the hopper at zero forward velocity. Two experiments were performed with each speed controller: one with an apex height of 0.45 m and another with an apex height of 0.5 m. In each experiment, the desired speed was increased in a step-wise fashion at 0.1 m/s intervals, up to approximately 1.8 m/s. Figure 5 shows the results of the experiment using the first height, and Figure 6 shows the results of the experiment using the second height.

It is apparent that there is a large error present at higher speeds with the Raibert controller. The neural network-based controller tracks the higher desired speeds with less error. Plots of error as a function of desired forward speed are shown in Figure 7 for the first experiment and Figure 8 for

L	leg length (m)
$\dot{x}$	body horizontal velocity (m/s)
x_f	distance from body c.m. to foot along horizontal dimension (m)
T_s	stance duration (s)
ϕ	body angle w.r.t. horizontal (rad)
γ	angle of leg w.r.t. body (rad)

TABLE I

VARIABLE DEFINITIONS.

g	gravitational acceleration	$-9.81\, m/s^2$
μ_k	viscous leg friction	$5.0\ N/s$
d_{hip}	distance from hip to body c.m.	$0.05\ m$
L_{max}	max leg length	$0.285\ m$
m_s	sprung mass	$8.375\ kg$
m_u	unsprung mass	$0.225\ kg$
A_p	area of piston	$4.91 \times 10^{-4}\ m^2$
P_0	nominal pressure of upper leg chamber	$4.0 \times 10^2\ kPa$

TABLE II

PHYSICAL PARAMETERS OF THE HOPPER.

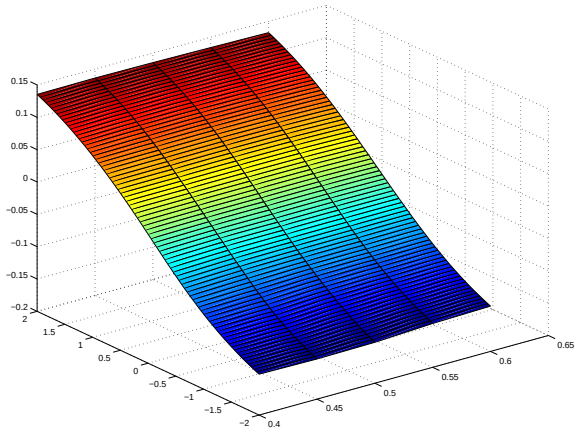


Fig. 4. Surface plot of neural network output.

the second experiment. Table III shows the average absolute and relative errors in speed.

V. CONCLUSIONS AND FUTURE WORK

We have shown that using a neural network to represent the functional mapping between forward speed and neutral point is a viable solution, and simulation results show that it tracks higher desired speeds with less error than the Raibert speed controller. The neural net controller adds little computational load to the system and is thus feasible to use on a real robot.

Future work in simulation includes 3D simulations and the introduction of sensor and actuator noise. Once construction of the real robot is completed, we will also test the two speed controllers on it.

We plan to implement additional behaviors on top of the basic locomotion. These include sitting and standing, leaning against a wall or corner, and hopping up and down inclines.

We have done some preliminary work in simulation with the incline problem, and have found that if the incline is sufficiently small, no changes are needed to the controller. We are exploring different strategies for negotiating steeper inclines.

Acknowledgments

The authors would like to thank Michael Poole for his work on designing and constructing the physical robot frame, presently under development, and Chad Jenkins for his expertise in training methods. The Vortex physics simulation toolkit developed by CM Labs was used for the experiments reported in this paper.

REFERENCES

[1] Marc Raibert, *Legged Robots that Balance*, MIT Press, Cambridge, MA, 1986.

[2] Kale Harbick and Gaurav S. Sukhatme, "Controlling hopping height of a pneumatic monopod," in *submitted to IEEE Intl. Conf. on Robotics and Automation*, 2002.

[3] M. Ahmadi and M. Buehler, "Stable control of a simulated one-legged running robot with hip and leg compliance," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 1, pp. 96–104, February 1997.

[4] G.Z. Zeglin and H.B. Brown, "Control of a bow leg hopping robot," in *IEEE Intl. Conf. on Robotics and Automation*, 1998.

[5] Joseph Prosser and Moshe Kam, "Control of hopping height for a one-

	Apex Ht. (m)	**Avg. Abs. Error** (m/s)	**Avg. % Error**
Raibert Controller	0.4	0.22	36.9
	0.5	0.19	23.6
N. Net-based Controller	0.4	0.03	8.3
	0.5	0.04	12.3

TABLE III

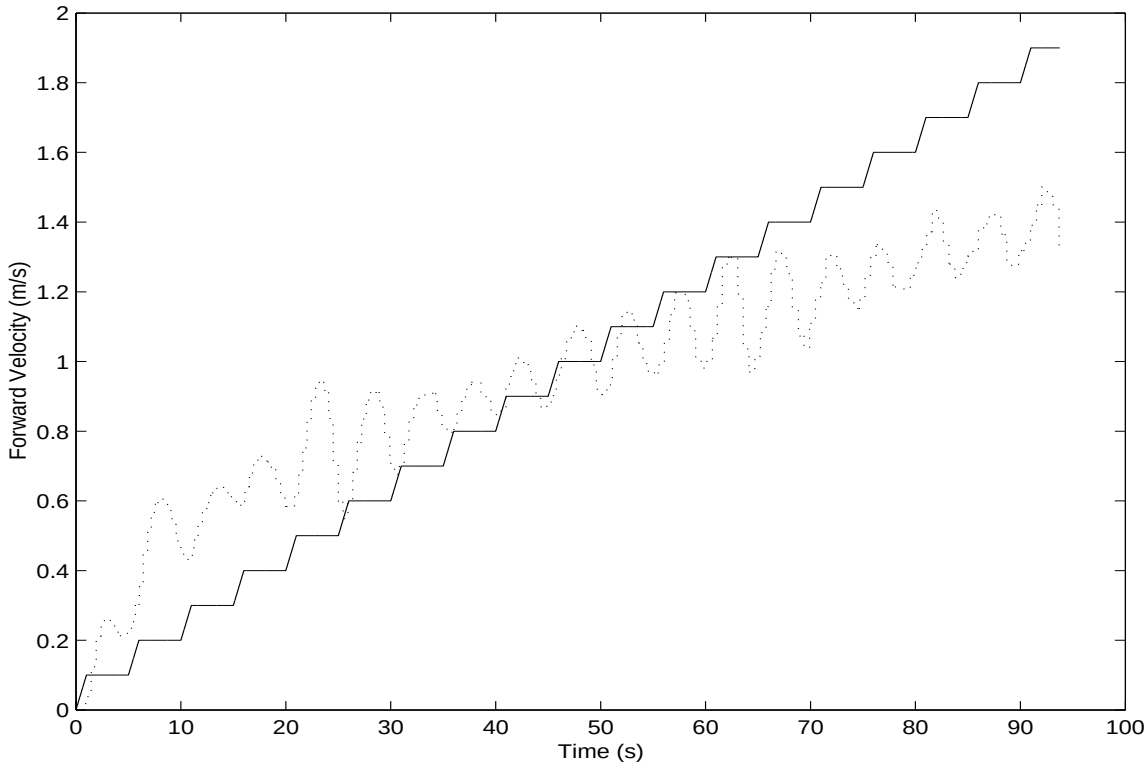
AVERAGE ABSOLUTE AND PERCENT ERRORS IN SPEED.

legged hopping machine," *Mobile Robots VII*, vol. 1831, pp. 604–612, November 1992.

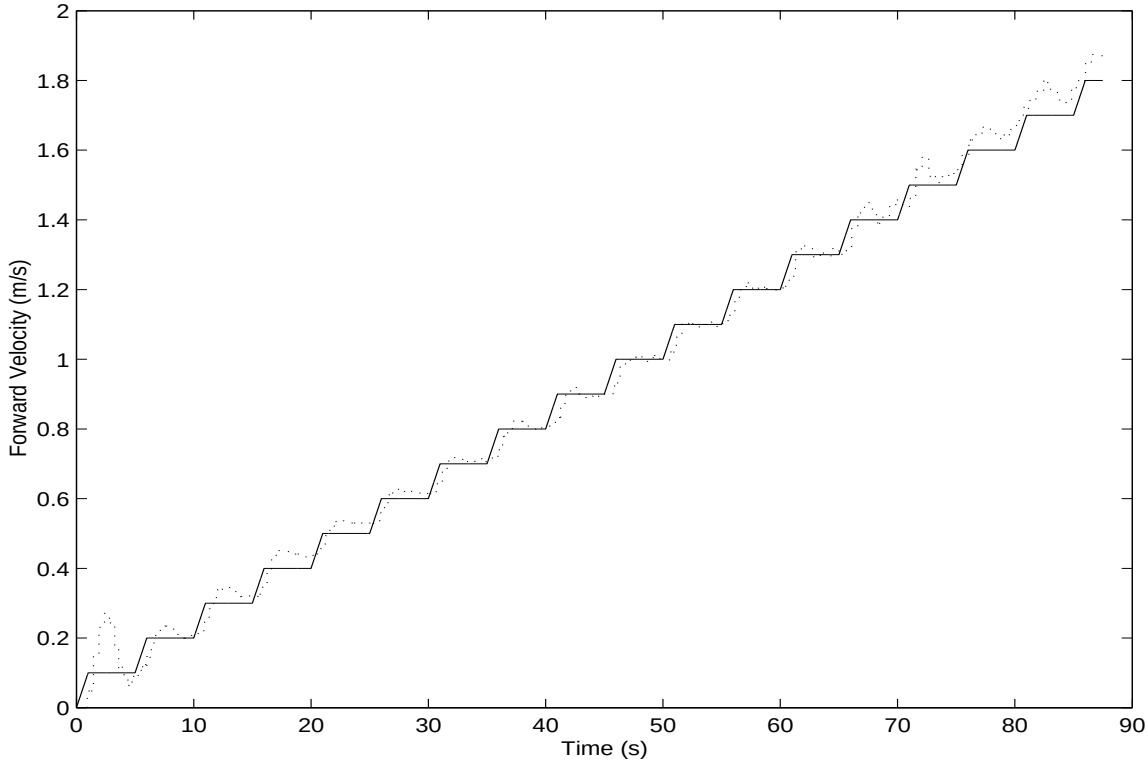
[6] A. D'Souza, S. Vijayakumar, and S. Schaal, "Learning inverse kinematics," in *IEEE Intl. Conf. on Intelligence in Robotics and Autonomous Systems*, 2001.

[7] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly Journal of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[8] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the Society of Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

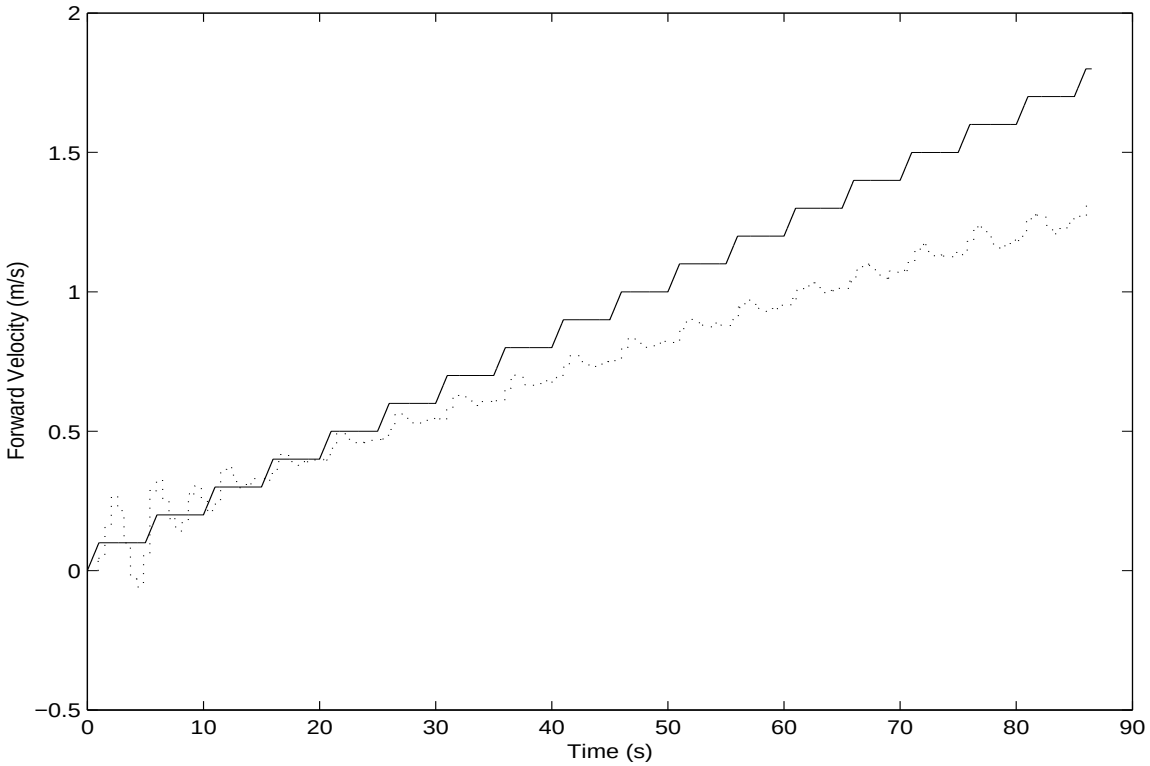


(a) PD controller.

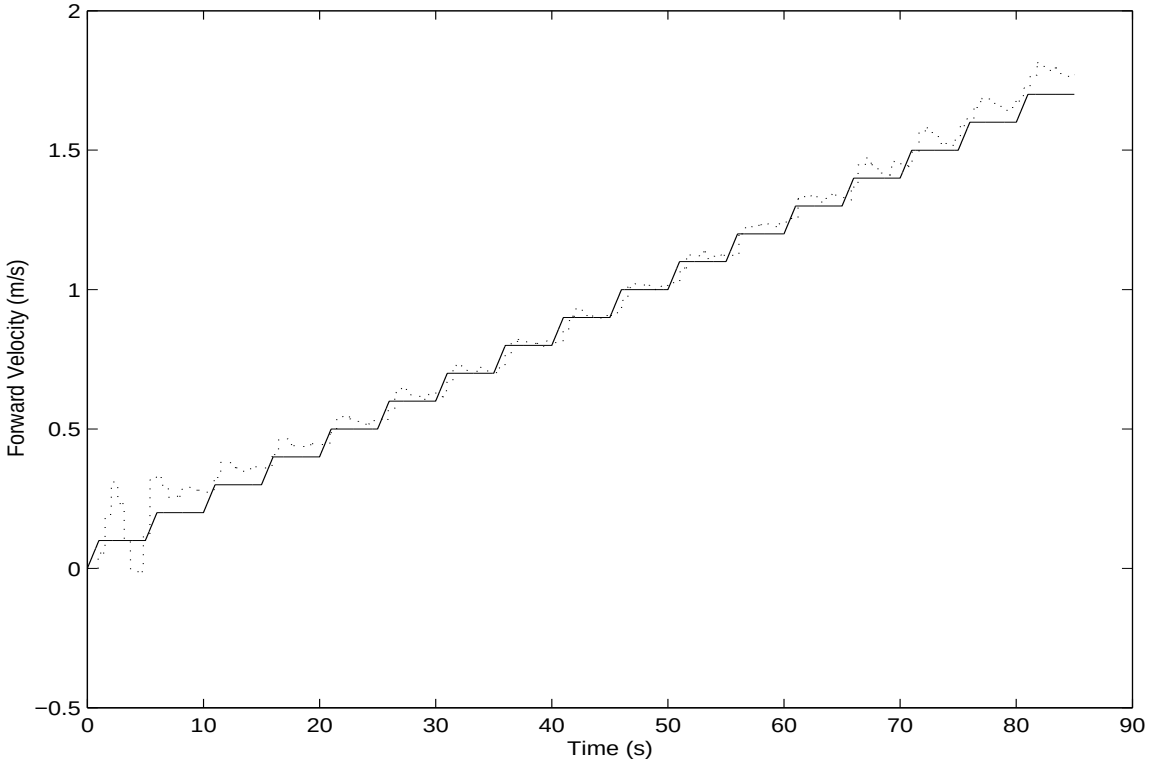


(b) Neural network-based controller.

Fig. 5. Speed performance, apex height: 0.4 m.

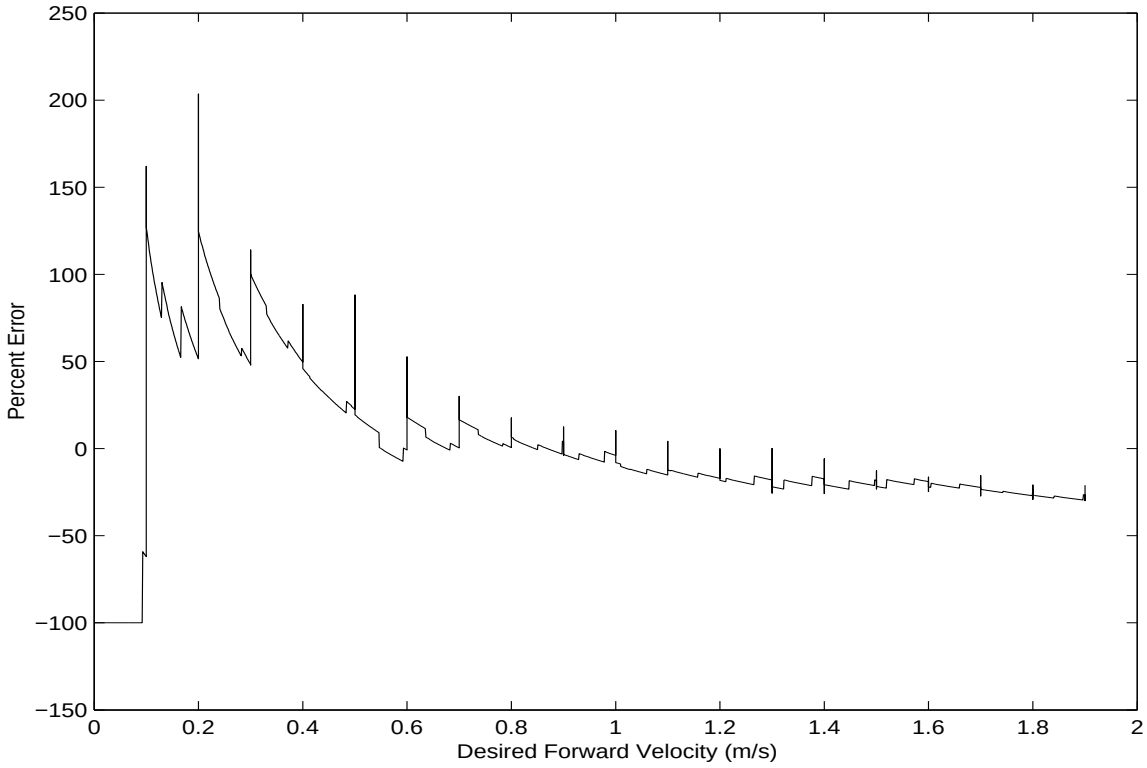


(a) PD controller.

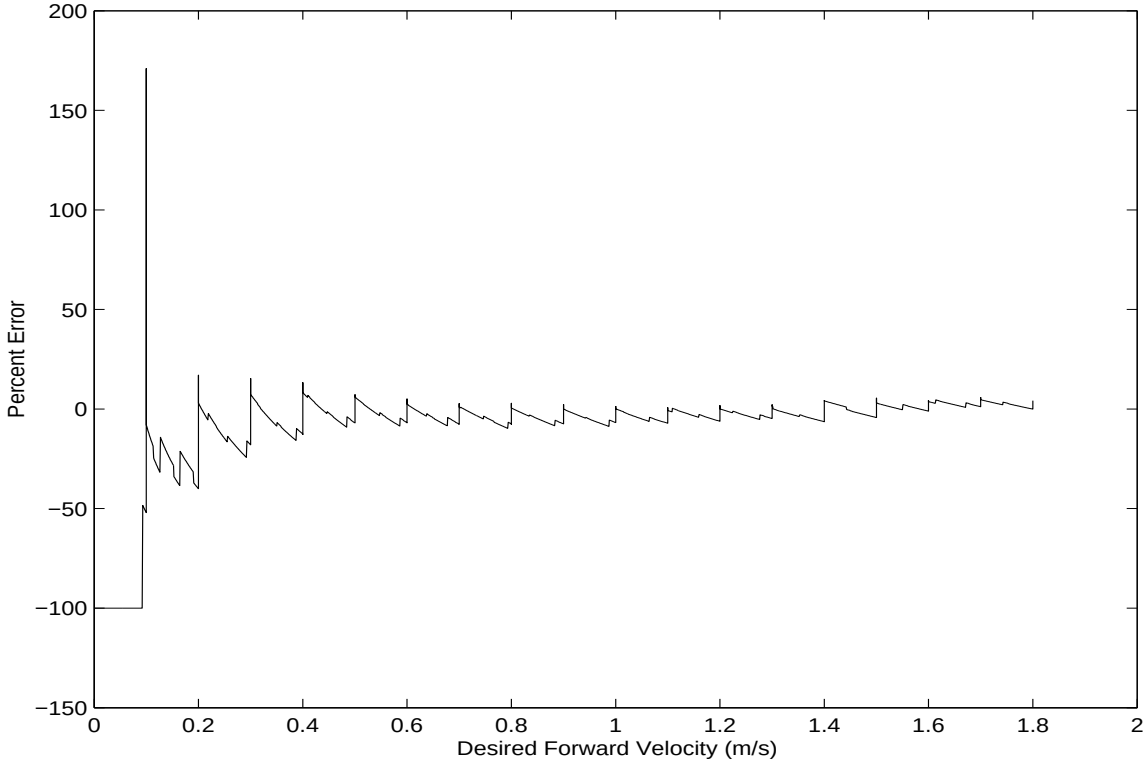


(b) Neural network-based controller.

Fig. 6. Speed performance, apex height: 0.5 m.

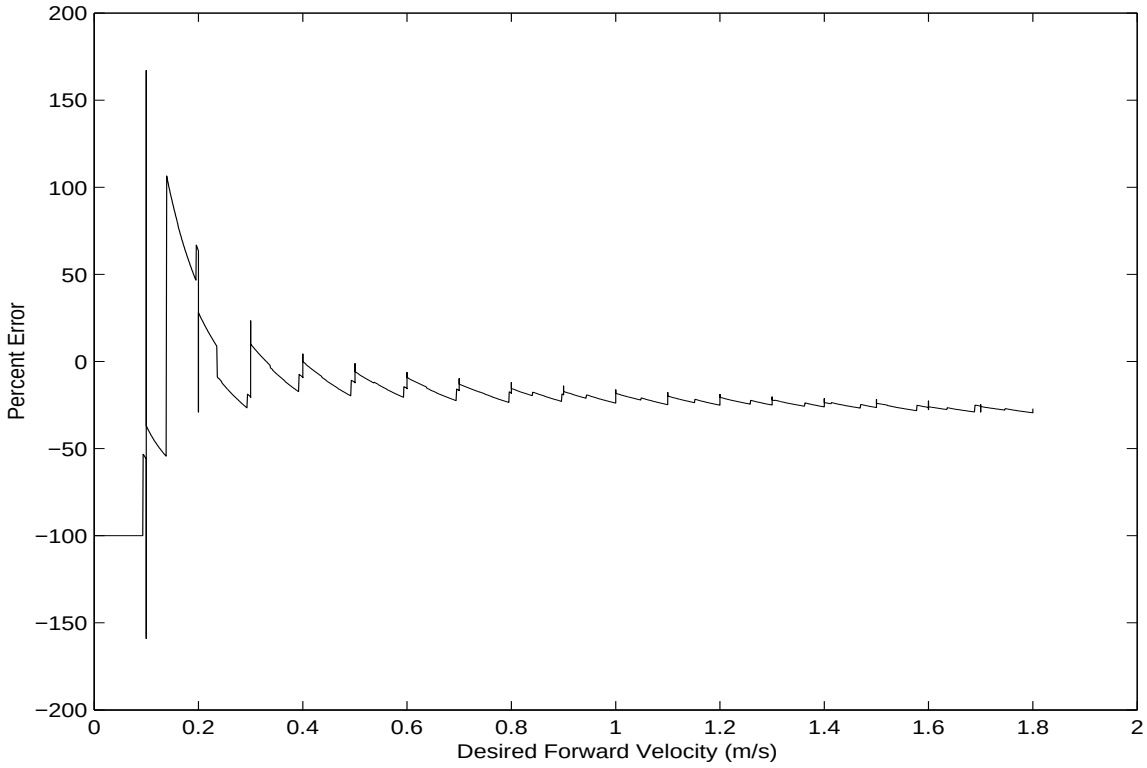


(a) PD controller.

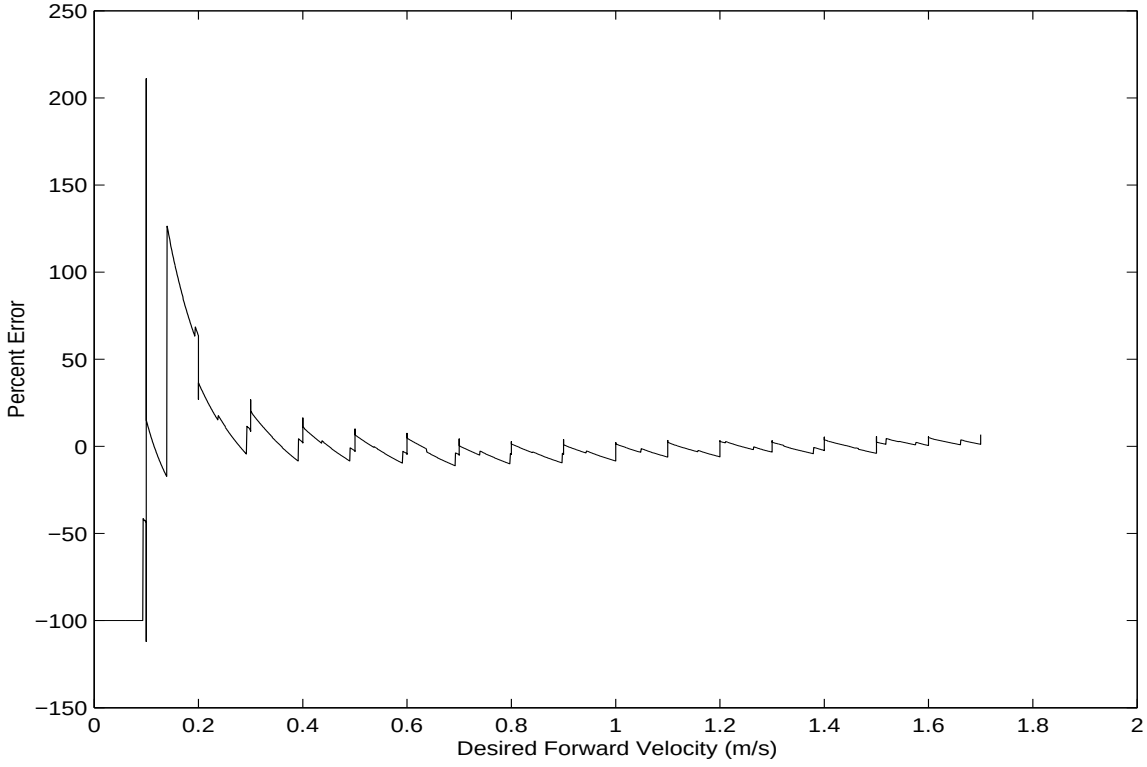


(b) Neural network-based controller.

Fig. 7. Percent error, apex height: 0.4 m.



(a) PD controller.



(b) Neural network-based controller.

Fig. 8. Percent error, apex height: 0.5 m.