2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

Control of Hopping Speed and Height Over Unknown Rough Terrain Using a Single Actuator

Nicholas Cherouvim and Evangelos Papadopoulos

***Abstract*— We present a method for controlling the forward speed and the apex height of a one-legged hopping robot over rough terrain, using a single actuator located at the robot hip. The control algorithm is comprised of two elements, the forward speed control and the height control. The only input to the system is the torque applied by the hip actuator. The control is demonstrated to perform tracking of desired forward speed trajectories and desired apex height trajectories. Simulation and experimental results on the SAHR (Single Actuator Hopping Robot) experimental setup are presented and compared. It is shown that the robot follows both trajectories closely in simulation as well as in experiment. Also the robot is tested successfully on a rough terrain course, which includes inclined ground and an abrupt drop in height of over 25% the length of the robot leg. The robot has no knowledge of its environment. Further, the robot is made to run over the course a number of times, to demonstrate the control robustness.**

I. INTRODUCTION

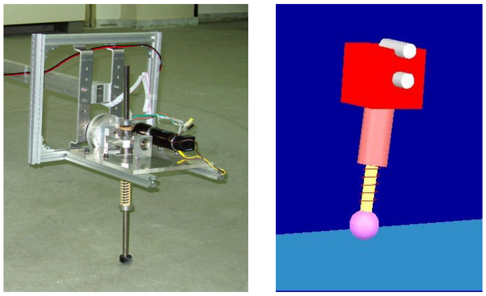
One of the primary goals in the field of legged robotics is the development of machines that will be able to transverse rough terrain, on this planet and others. Although legged machines have the potential to outperform wheeled vehicles on rough terrain, they are subject to more complex control challenges and must also consider the problem of balance in motion. Simply controlling the forward speed becomes a much more involving issue than in vehicles with wheels.

Considerable progress has been made towards legged robots that transverse rough terrain. Most earlier approaches to rough terrain locomotion required a known terrain profile. Generally, the robot would walk over the terrain with a gait that was statically stable, in contrast to running, hence simplifying the stability problem [1]. Some of the first works on a running robot negotiating rough terrain was done by Hodgins, who controlled a running biped over stair-like terrain although the terrain profile was known in advance [2].

More recently, the Tekken, RHex and BigDog robots have demonstrated successful motion over outdoor terrain [3], [4], [5]. Of these, the Tekken and BigDog robots both use three actuators on each robot leg, while the RHex robot uses only one actuator per leg, and also uses a different method of motion where the legs revolve continually rather than oscillate back and forth. The RHex robot runs mainly with open-loop control and forward speed is not directly controlled [8]. Finally, the RHex's unique design and use of tripod gaits make it difficult to generalize its method of locomotion to other robot designs. An important characteristic of the robots capable of locomotion on uneven terrain is the number of actuators used per leg, as this affects not only the robot weight, but also the complexity of the design, the robot cost and the power autonomy of the system. It is therefore desirable to be able to develop a legged system that can transverse rough terrain and control its forward speed, using as few actuators as possible.

For the purpose of potentially applying our results to a wider range of running robots, we study the case of a one-legged system. Robots with one leg are often used to explore the basic principles of legged locomotion [9], [10]. Perhaps the most outstanding example of this is the SLIP (Spring Loaded Inverted Pendulum) model, which has been widely adopted in much work, and has even been used to study the dynamics of the RHex hexapod [11]. It is reasonable, therefore, to seek to develop a control method with as few actuators as possible for robust locomotion over rough terrain, for the case of a one-legged robot.

In this work we present a controller for robust locomotion of a one-legged robot over unknown rough terrain, with the ability to control both forward speed and apex height. To date, this has not been accomplished in the literature. The robot has only the minimum actuation for locomotion, a single actuator, located at the robot hip. The control is tested both in simulation and on the SAHR (Single Actuator Hopping Robot) hardware, shown in Fig. 1a. The robot is successfully made to follow desired trajectories with regard to forward speed and height. Further, the robot is tested on rough terrain and shows consistent robust behavior.



(a) The SAHR setup. (b) ADAMS model.

Fig. 1: a) The hopping robot experimental setup, with a single actuator. b) The robot modelled in ADAMS software.

This work is co-funded by public funds (European Social Fund 80% and General Secretariat for Research and Technology 20%) and private funds (Zenon S.A), within measure 8.3 of Op.Pr.Comp., 3rd CSP - PENED 2003.

N. Cherouvim is with the Department of Mechanical Engineering, National Technical University of Athens, 15780, Athens, Greece ndcher@mail.ntua.gr

E. Papadopoulos is with the Department of Mechanical Engineering, National Technical University of Athens, 15780, Athens, Greece egpapado@central.ntua.gr

978-1-4244-2789-5/09/$25.00 ©2009 IEEE

II. HOPPING ROBOT DYNAMICS

In this paper we study a hopping robot with a single leg. The experimental setup is shown schematically in Fig. 2a. An arm ensures that the robot performs a circular motion around the central pivot and constrains the robot body pitching motion. In the literature, similar setups are used to study robots in a plane of forward motion [7]. The constraint of the pitching motion does not inhibit the robot from capturing dynamic characteristics present in many legged machines. The robot leg has two degrees of freedom (DoF), one which allows revolution about an axis parallel to the support arm (the hip axis), and a prismatic DoF which allows compression and extension. The revolute DoF is actuated, while the prismatic DoF is passive and has a linear passive spring.

As the setup is used to study the planar motion of the robot, the dynamics are represented by a 2D robot model, shown in Fig. 2b. The model has a body, with a mass m, corresponding to an effective mass which is due to the robot mass and partly due to the mass of the support arm. The length of the leg when not compressed is L, the stiffness of the linear spring is k, and the torque applied by the hip actuator is denoted by τ, also see Fig. 2b. The length of the leg at any time is r, while the angle of the leg with respect to the vertical is ϕ. The robot leg is modelled with a mass m_l, and energy losses in the leg prismatic DoF are modelled using viscous friction with a coefficient of b. Effects due to the impacts with the ground are included in the lumped parameters k, b.

During the stance phase, when the robot leg is on the ground, the dynamics is found using a Lagrangian approach using the Cartesian coordinates x, y of the robot body, see Fig. 2b. During the stance phase the effect of leg mass is omitted as it is negligible compared to the body inertial forces and spring forces. The right hand side of the dynamics is quite complex when written using only x, y. However it can be seen that these expressions may be replaced with expressions which are functions of r, ϕ. Then the dynamics for the stance phase assume a compact form:

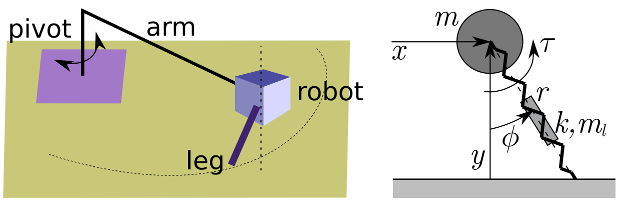
$$m\ddot{x} + k\,(L-r)\sin\phi - b\,\dot{r}\sin\phi = -\tau\cos\phi/r \tag{1}$$

$$m\;\ddot{y} - k\,(L-r)\cos\phi + b\;\dot{r}\;\;\cos\phi = -\tau\;\;\sin\phi/r \tag{2}$$

During flight, the robot center of mass (CoM) performs a ballistic trajectory. Omitting the effect of the leg mass on the robot motion, the flight dynamics is:

$$\ddot{x} = 0 \tag{3}$$

$$\ddot{y} = -g \tag{4}$$



(a) SAHR setup overview. (b) Dynamic model.

Fig. 2: a) The SAHR experimental setup. b) The dynamic model used for the control derivation.

III. ROBOT CONTROL

The control of the robot is split into two parts. The first has to do with the control of the robot forward speed, while the second part has to do with the control of the height attained by the robot at the apex of each flight phase. The only explicit input to the system is the torque from the single actuator. From a physical viewpoint, the actuator torque during stance acts mainly on the forward speed dynamics. This is verified by observing (1), (2), in which the torque τ is stronger in the forward dynamics, see (1), while in the vertical dynamics, see (2), it may disappear completely due to the sine of the leg angle. Additionally, it is clear that the angle with which the leg strikes the ground is an important parameter of the motion. To visualise this consider that, by choosing the leg angle ϕ with which the leg strikes the ground, more or less of the robot's energy from the flight phase can be translated into spring compression during stance. Further, spring compression is directly related to the vertical dynamics. For the above reasons, the hip actuator torque during stance, τ_{st}, and the leg with which the angle strikes the ground at touchdown, ϕ_{td}, are chosen as the quantities used as control inputs. During flight, the hip actuator will be used to position the leg to the desired touchdown angle.

To implement the speed and apex height control, proper feedback is required. The robot has two sensors, which provide feedback of the leg angle position with respect to the robot body, ϕ, and the length of the robot leg, r. More detail on the hardware will be given in Section V. From the sensory feedback, we are also able to compute the rates of change $\dot{\phi}$ and $\dot{r}$. Using this and simple geometry, see Fig. 2b, it is possible to know the robot states during the stance phase:

$$x - x_{fp} \;=\; -r\sin\phi \tag{5}$$

$$y \;=\; r\cos\phi \tag{6}$$

where x_{fp} is the x coordinate of the foot position on the ground during stance. By differentiating (5), (6), the forward and vertical velocities x, y of the body are also known:

$$\dot{x} \;=\; -\dot{r}\sin\phi - r\;\dot{\phi}\cos\phi \tag{7}$$

$$\dot{y} \;=\; \dot{r}\cos\phi - r\;\dot{\phi}\sin\phi \tag{8}$$

A. Forward speed control

The control of the forward speed is accomplished using the hip actuator during stance τ_{st}. We have found a proportional derivative (PD) control to provide a robust solution. The PD control is implemented using the actuator torque during stance, and hence controls the forward velocity during stance. In actual fact, there are two possible approaches to controlling the forward velocity using a PD control and they differ in the actual quantity that is controlled. The first directly controls the actual forward velocity $\dot{x}$ of the robot:

$$\tau_{st} = k_p\,(\dot{x} - \dot{x}_{des}) \tag{9}$$

Note that according to the formulation used in the dynamics, negative actuator torque propels the robot forward during stance, and positive leg angles are in front of the body. The proportional control in (9) may be intuitive, but it suffers from the fact that the forward velocity is not a directly measured quantity, but rather is given by (7). This means that the forward velocity will involve a greater number of measured quantities and will have increased noise and error.

A second approach to controlling the forward speed relies on the approximate relationship between forward speed and leg angle, set out in [7], whereby:

$$\phi = \phi_{td} - \dot{x}\ t/L \tag{10}$$

where ϕ_{td} is the leg touchdown angle, and time t starts at leg touchdown. Differentiating, we have:

$$\dot{\phi} = \dot{x}/L \tag{11}$$

and therefore controlling the leg angle velocity $\dot{\phi}$ will approximately control the forward speed. The desired rate of change for the leg angle is set as $\dot{\phi}_{des} = \dot{x}_{des}/L$. The control of the forward speed now has the benefit that only one measured quantity is involved, and it is implemented as:

$$\tau_{st} = -k_p \left(\dot{\phi} - \dot{\phi}_{des}\right) \tag{12}$$

B. Control of the apex height

When running over uneven terrain, proper control of the robot's vertical motion is particularly important to overall stability. To allow the robot to accommodate for changes in the ground incline, as well as sudden changes in the height of the terrain, the robot should be able to maintain a constant distance from the ground surface, despite the unknown variations in that surface. This is the purpose of the method we propose here for controlling the height of the robot body at the apex point of each flight phase.

As explained at the beginning of this section, the leg touchdown angle ϕ_{td} has a strong effect on the vertical dynamics, and we use this as a control input to achieve the desired apex height from the ground at each cycle. For the determination of the proper leg touchdown angle, ϕ_{td}, we start from the dynamics in (2), which is related to the vertical motion of the robot. Substituting the length of the leg r and the angle of the leg ϕ as expressions of the coordinates x, y of the robot body, and using trigonometric small angle approximations, the vertical dynamics becomes:

$$m\ \ddot{y} + b\ \dot{y} + k\ y = k\ L\ \cos\phi \tag{13}$$

The dynamics in (13) now has the form of an oscillator, with regard to the robot body height y. The oscillator is driven by a term dependent on the leg angle ϕ. Using (10), the leg touchdown angle ϕ_{td} appears in the driving term:

$$m\ \ddot{y} + b\ \dot{y} + k\ y = k\ L\ \cos\left(\phi_{td} - \dot{x}\ t/L\right) \tag{14}$$

where time t starts at the beginning of the stance phase. Integrating (14) twice yields the evolution of the body height y during stance. Then, the necessary leg touchdown angle can be computed so that a desired vertical liftoff velocity $\dot{y}_{lo}$ is achieved. As the robot flight dynamics are simple, this computation is equivalent to computing ϕ_{td} such that a desired apex height h_{des} is achieved:

$$\phi_{td} = f\left(\text{robot state at touchdown}, \dot{x}_{des}, h_{des}\right) \tag{15}$$

A limit of 15 deg is imposed on the calculated leg touchdown angle, as a preventative measure against foot slipping due to the steeper attack angle. The leg is brought to its desired touchdown position using the hip actuator during the flight phase. A proportional derivative position controller servos the leg, and the actuator command is:

$$\tau = k_{p,fl}(\phi_{td} - \phi) - k_{d,fl}\dot{\phi} \tag{16}$$

C. Control application

To give a clear view of the control algorithm, one complete cycle of the robot motion is described. Consider the robot to be just before the point of liftoff from the ground, at the end of the stance phase. At this point, the robot body velocities, $\dot{x}$, $\dot{y}$ and the body height y are computed from (7), (8), (6).

The robot now enters the flight phase and performs two calculations. First, the velocities $\dot{x}_{td}$, $\dot{y}_{td}$ are estimated for the impending touchdown after the flight phase, using the simple flight dynamics. Secondly, the necessary touchdown angle for the leg ϕ_{td} is calculated according to (15). These calculations take less than 2ms on the SAHR's computer.

In the remainder of the flight phase the leg is servoed to the desired touchdown angle, by applying the PD position control in (16). Once the robot hits the ground and enters the stance phase, the hip actuator is commanded to apply the torque in (12). From there on, the cycle repeats.

IV. SIMULATION RESULTS

The SAHR setup is simulated using a model in ADAMS simulation software shown in Fig. 1b. The ADAMS model is restricted to 2D motion, and this matches the SAHR setup as explained in Section II. The simulated model is much closer to the real setup than the simplified model, in Fig. 2b, used for the control derivation. The ADAMS model also includes inertia of the robot leg, and viscous friction losses at the hip as well as in the leg prismatic DoF. A detailed actuator model is used, which includes the inertia of the motor rotor and the motor torque-speed characteristic, as provided by [12], also see Table I. A Coulomb friction model between the foot and the ground is also included. The impacts of the foot with the ground are modelled as plastic collisions. In conjunction with the ADAMS software, MATLAB Simulink is used for running the control algorithm, receiving the feedback of the leg angle and leg length from ADAMS and passing back the actuator torque command. The control is applied to the model as described in subsection C of Section III.

The control is first applied on smooth ground, initially following a desired speed trajectory and then following a desired apex height trajectory. At the beginning of each simulated run, the robot is left to fall from some initial height. Results from following a desired speed trajectory are shown in Fig. 3. The robot must follow the desired

TABLE I: Main parameters of the SAHR experimental setup.

Quantity	Symbol	Value	Unit
Robot body mass	m	4	kg
Leg spring stiffness	k	4800	N/m
Leg rest length	L	0.275	m
Leg mass	m_l	0.5	kg
Leg viscous friction	b	6	N s/m
Leg inertia	-	0.009	kg m^2
Hip viscous friction	-	0.23	N m s/rad
Motor rotor inertia	-	7.87e^{-6}	kg m^2
Max. motor torque (continuous)	-	0.093	N m

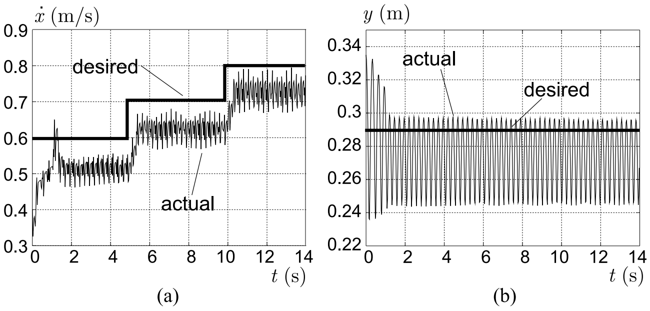
speed trajectory depicted by a thick line, while the attained apex height should remain constant, at a desired value of 0.29m. In Fig. 3 the robot speed is controlled well, and small deviations from the desired trajectory are due to the added complexity of the ADAMS model compared to the model used for the control derivation. In Fig. 3b, the body apex height corresponds to the highest point of each cycle of the body height curve. It is kept constant over the entire speed range, although it is slightly higher than the desired value.

Following a desired apex height trajectory is shown in Fig. 4. The robot must attain the desired apex heights, shown in Fig. 4b with a thick line, while maintaining the forward speed constant at 0.7m/s. The height trajectory is followed properly, and the forward speed shows only small changes.

Finally, the robot is simulated running on rough terrain, with uphill and downhill inclines of 10deg, and sudden drops of up to 7cm, see Fig. 5c. The robot is commanded to maintain a constant forward speed of 0.6m/s and an apex height of 0.29m. Snapshots from the simulation are shown in Fig. 5c and the forward speed and body height are shown in Fig. 5a, 5b. Despite the powerful environmental disturbances, the controller can be seen to continuously restore the forward speed to the desired value. The body height shows quite large deviations from 0.29m, but this is because it was measured with respect to the level ground.

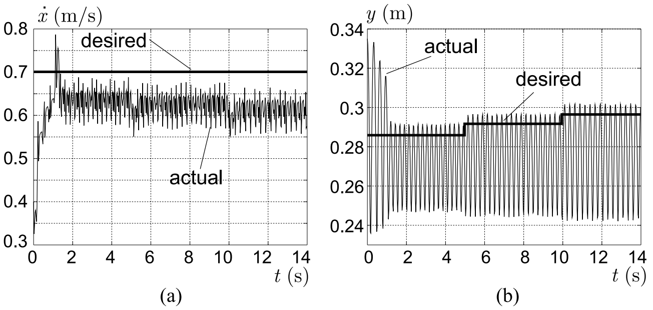
V. EXPERIMENTAL RESULTS

A. Experimental setup



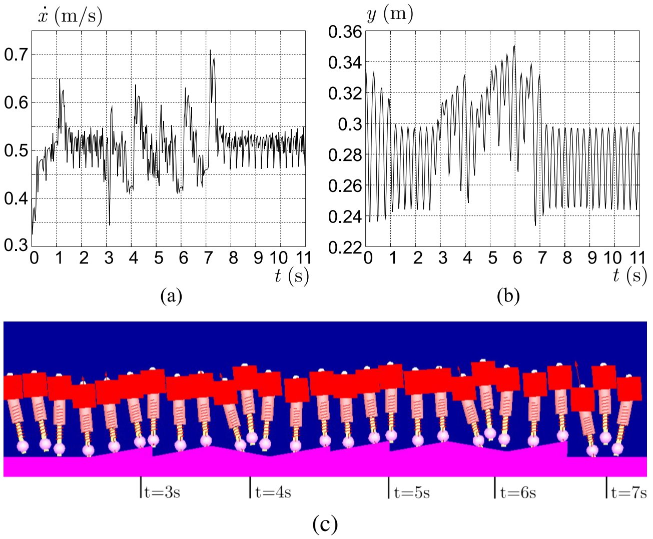
(a) (b)

Fig. 3: Robot following a desired forward speed trajectory in ADAMS. a) Forward speed. b) Body height from the ground.



(a) (b)

Fig. 4: Robot following a desired apex height trajectory in ADAMS. a) Forward speed. b) Body height from the ground.



(a) (b)

(c)

Fig. 5: Robot running over uneven terrain in ADAMS. a) Forward speed. b) Body height from the *level* ground. c) Simulation snapshots, with points in time marked.

In addition to simulating the controller behavior, we have developed the SAHR (Single Actuator Hopping Robot) setup for testing in practice. The setup has been described in brief in Section II, and the robot body and leg are shown in Fig. 1a. A simplified view of the complete setup is provided in Fig. 2a, which shows how the robot body is constrained by an arm to perform a cyclical motion around the main pivot. Power is taken from a mains outlet, and is passed down to the setup from the ceiling. The main parameters of the SAHR setup are shown in Table I.

The only actuator is a DC electric motor, which provides the torque input τ referred to in Section II. The motor is from Maxon, model RE35, coupled to a planetary gearhead with a reduction ratio of 26:1, [12]. A timing belt transmits the torque to the leg and provides an additional reduction of 2:1, making the reduction from motor shaft to robot leg equal to 52:1. As a result the system produces up to about 5.5 Nm of torque on the leg.

For the control algorithm implementation, a PC104 stack running Linux is used. Additionally a custom board of microcontrollers implement the low-level interface with the sensors and the motor drive unit which drives the actuator, see Fig 6a. On the custom microcontroller board we use the I2C protocol for communication. An Atmel AVR microcontroller acts as the I2C master, while Microchip PIC microcontrollers serve as I2C slaves. The I2C master interfaces with the PC104 using a custom protocol based on GPIO (General Purpose Input Output) pins, which are part of the PC104 stack.

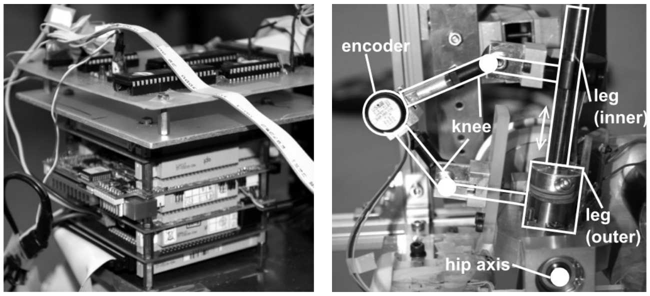
The two feedback quantities are the length of the robot leg and the leg angle. The first quantity is calculated indirectly by measuring leg compression with the use of a pseudo-knee and optical encoder placed at the knee joint, see Fig. 6b. The leg angle is known from the optical encoder mounted on the actuator. A software initialization routine ensures knowledge of the absolute leg angle. The feedback described can easily be used, as is, in the case of a fully autonomous robot.

B. *Smooth terrain results*

The robot is made to follow desired speed and height trajectories on the smooth terrain of the lab floor. It should be noted, however, that the lab floor is slightly inclined by construction. This acts as a constant disturbance to the robot motion, as it sometimes appears as uphill and other times as downhill, due to the robot's cyclical motion. Upon applying the controller, the only tuning required was the determination of proper gains for the PD control for positioning the leg in flight, see (16), and the gain for the speed control in (12). Once found, these did not have to be adjusted again. Finally, it was important to avoid unnatural combinations of forward speed and apex height, such as very low speeds with large apex heights, or high-speed running with small apex heights.

The first experiment on smooth terrain is following a desired speed trajectory, which is depicted in Fig. 7a as the thick line. The apex height of the robot is desired to remain constant at 0.29m. In Fig. 7a, 7b the forward speed at liftoff and the body height at apex are shown respectively. The forward speed can be seen to follow the desired trajectory faithfully, and the body apex height can be seen to vary slightly as the forward speed changes.

The next smooth terrain experiment is following a desired apex height trajectory, which is shown in Fig. 7d with a thick line. The forward speed is desired to be kept at 0.6m/s. In Fig. 7c, 7d the forward speed at liftoff and the apex height are shown. The robot successfully attains all the desired apex heights. The lower speeds at the beginning of the run are due to the transient of the robot starting from stationary.



(a) PC104 stack and custom electronics board.

(b) Pseudo-knee.

Fig. 6: a) The PC104 stack and the custom microcontroller board. b) The pseudo-knee for measuring leg compression.

C. *Rough terrain results*

The robot is also made to transverse an obstacle course, consisting of a 5deg uphill incline, and an abrupt drop of 7cm at the end of the incline, which is more than 25% of the robot leg length at rest. The robot moves in a cyclical path, and so passes over the course repeatedly. In the results shown here, the robot crossed over the course three times. The controller consistently kept the robot stable uphill and performed proper recovery of the robot balance after the sudden drop. The run can also be viewed in the video attached to this paper. Snapshots from the robot recovering from the 7cm drop are shown in Fig. 8. The robot was still running properly when the experiment was terminated. For this run the desired forward speed is set to 0.8m/s, while the desired apex height is set to 0.29m. For this run, the forward speed at each liftoff and the body height from the ground at each apex are shown in Fig. 9a, 9b respectively. Some decrease of the forward speed can be seen when running uphill, which is marked by the dark-band regions in Fig. 9a. The light-band regions in Fig. 9a, 9b show the recovery of the robot after the abrupt 7cm drop in the ground level.

Magnified data from a time-window around the first abrupt 7cm drop is shown in Fig. 9c, 9d. The shaded area corresponds to the snapshots of Fig. 8. Each leg touchdown angle shown in Fig. 9c is the angle used before the corresponding apex point shown in Fig. 9d. It can be seen that the abrupt drop causes a large robot rebound, which the controller counters by decreasing the leg touchdown angle. In about three cycles, the apex height has again been stabilised.

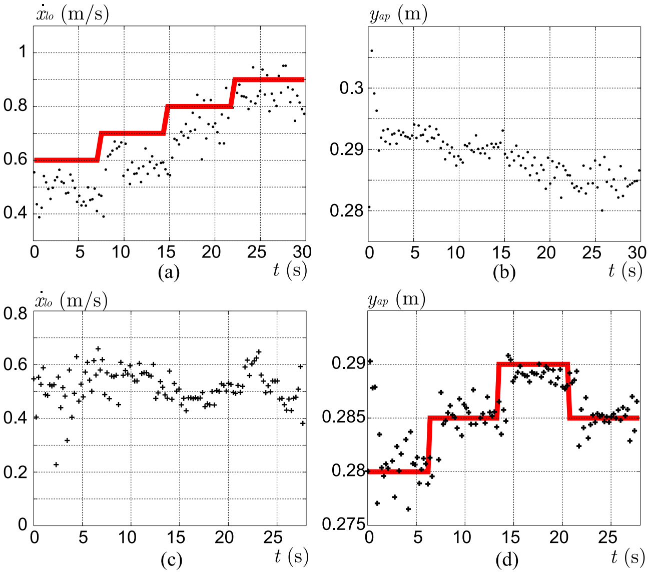


Fig. 7: Robot running on smooth terrain. Forward speed at liftoff, $\dot{x}_{lo}$, and body height at flight apex, y_{ap}, are shown. Following a desired speed trajectory in (a), (b). Following a desired apex height trajectory in (c), (d).

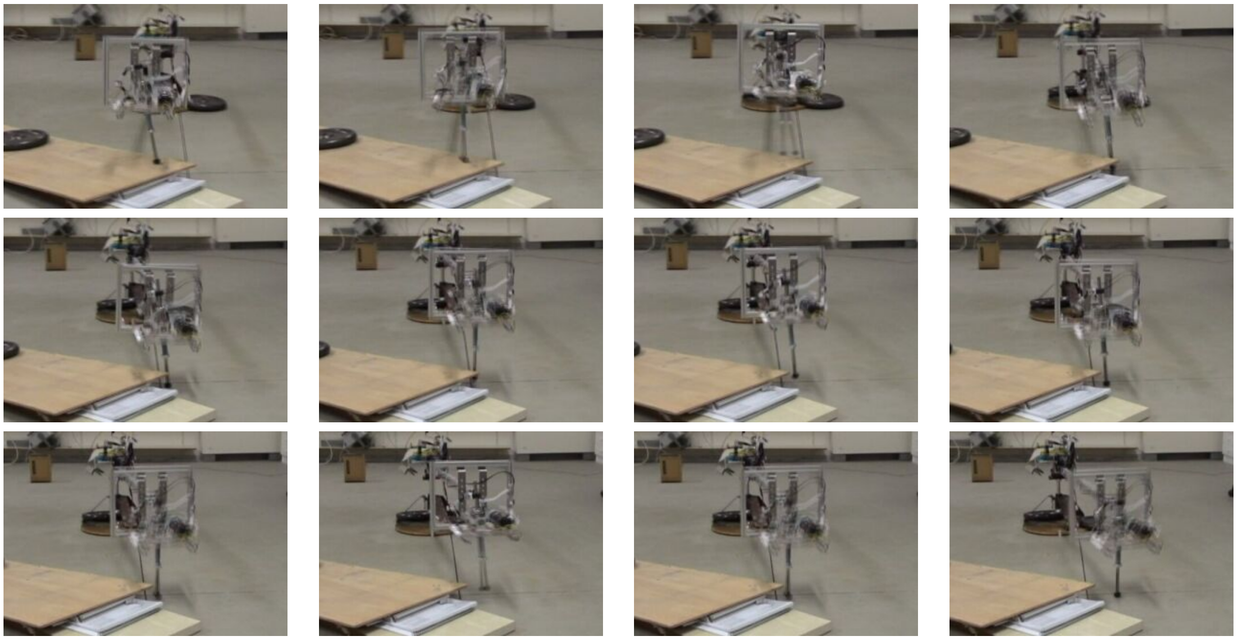


Fig. 8: Snapshots of SAHR recovering after a 70mm drop in the terrain. Snapshots are taken every 80ms.

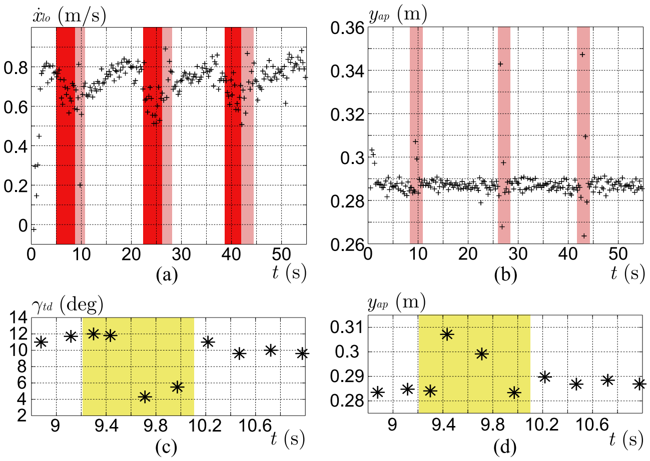


Fig. 9: Robot on unknown rough terrain. In (a), (b), dark-band regions show uphill motion, light-band regions show recovery after the abrupt drop. Banded regions in (c), (d) show the time window of Fig. 8. a) Forward speed at liftoff. b) Body apex height. c) Leg touchdown angle around abrupt drop. d) Body apex height around abrupt drop.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we presented a controller capable of controlling the forward speed and the apex height of a running robot with one leg, over unknown rough terrain. The robot possesses only a single actuator with which it drives the leg angle. The two parts of the control algorithm were shown, which correspond to the control of the robot's forward speed and the control of the robot's apex height. The only input to the system is the torque applied by the hip actuator. The control was shown to perform well both in simulation and when applied to the SAHR experimental setup. Following of desired apex height and forward speed trajectories was demonstrated. Also, the controller showed consistent robust behavior on uneven terrain, which included sudden drops in terrain height. Future work includes the extension of the control approach to the case of multi-legged systems.

REFERENCES

[1] R.B. McGhee and G.I. Iswandhi, "Adaptive Locomotion of a Multilegged Robot over Rough Terrain", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, 1979, pp. 176-182.

[2] J. Hodgins, "Legged Robots on Rough Terrain: Experiments in Adjusting Step Length", *in Proceedings of the 1988 IEEE Int. Conf. on Robotics and Automation*, Philadelphia, PA, 1988, pp. 824-826.

[3] Y. Fukuoka, H. Kimura and A. H. Cohen, "Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts", *The Int. Journal of Robotics Research*, vol. 22, 2003, pp.187-202.

[4] U. Saranli, M. Buehler and D.E. Koditschek, "RHex: A Simple and Highly Mobile Hexapod Robot", *The Int. Journal of Robotics Research*, vol. 20, 2001, pp. 616-631.

[5] M. Buehler, R. Playter, and M. Raibert, "Robots Step Outside", *in Int. Symp. Adaptive Motion of Animals and Machines*, Ilmenau, 2005.

[6] L.R. Palmer and D.E. Orin, "Quadrupedal running at high speed over uneven terrain", *in IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2007, pp. 303-308.

[7] M.H. Raibert, *Legged robots that balance*, MIT Press, Cambridge, MA, 1986.

[8] M. Buehler, U. Saranli, D. Papadopoulos and D. Koditschek, "Dynamic locomotion with four and six-legged robots", *in Int. Symp. Adaptive Motion of Animals and Machines*, Montreal, 2000.

[9] S. Hyon and T. Emura, "Energy-preserving control of a passive one-legged running robot", *Advanced Robotics*, vol. 18, 2004, pp. 357-381.

[10] N. Cherouvim and E. Papadopoulos, "Energy saving passive-dynamic gait for a one-legged hopping robot", *Robotica*, vol. 24, 2006, pp. 491-498.

[11] R. Altendorfer, D. E. Koditschek and P. Holmes, "Stability Analysis of a Clock-Driven Rigid-Body SLIP Model for RHex", *The Int. Journal of Robotics Research*, vol. 23, 2004, pp. 1001-1012.

[12] Maxon Motor AG, www.maxonmotor.com.