Atlas Ankle Balancing

Wyatt Newman
May 31, 2014

Introduction: These notes describe progress towards feedback control for Atlas balancing on one foot. The general case of Atlas dynamic balancing includes rapid accelerations of multiple body joints, and subsequent reconvergence to a stable, balanced pose. It is envisioned that a robust balance recovery strategy would invoke multiple phases of control, including a possible high-acceleration, emergency response to near-fall conditions, graceful repositioning to a nominal pose, and return to a nominal balance strategy requiring only ankle control. These notes describe only the ankle-control phase.

Robot model: For simple analysis of single-ankle control of balance, it is assumed that all body joints of Atlas are frozen at some pose, and only one joint (an ankle roll) is responsible for maintaining balance. For example, joint 15 of Atlas corresponds to rotation of the right ankle along the x-axis of the right foot frame. This is the most challenging axis, since Atlas's foot is less than 12cm wide (vs 26 cm long). Therefore, a shift of the robot's center of mass of merely +/- 6cm in the lateral direction results in inability to recover balance using only the ankle. Results of this demanding case can be utilized for the orthogonal ankle joint as well (e.g., joint 14) to maintain balance in the sagittal plane.

Balance control in the present approach assumes use of position and velocity commands, not torque commands. This assumption makes implementation on Atlas simpler, as it can use the existing control interface. Further, the kinematics and controls of the ankle joints are complex, since these degrees of freedom involve dual actuators in a closed-chain mechanism. Decoupling this system into equivalent orthogonal rotations (both sensing and control) is currently performed within the Boston Dynamics eBox. Attempting direct actuator effort control of the ankle joints would involve considerable additional complexity, and prospective benefits of such extra effort are uncertain.

For the simplified case presented here, the robot is assumed to be an inverted pendulum controllable via position and velocity commands to an ankle joint. The ankle joint can exert restoring torques on the robot indirectly, as a result of the foot-pad interacting with the ground. The achievable ankle torque reaches saturation corresponding to the robot's entire weight (150kg) directed over one edge of the foot. Since the foot is only 12cm wide, maximum ankle torque corresponds to the robot's center of mass shifted 6cm laterally, corresponding to a torque of 88N-m.

Atlas gravity modeling: Figure 1 shows results of gravity-model characterization tests on Atlas. In this test, Atlas had both feet on the ground, and he was controlled to lean forward and backward about his ankles (joints 14 and 8). Torques about his ankles (as measured by his foot-pad force-torque sensors) and tilt relative to gravity (as measured by his IMU) were analyzed to deduce the whole-body center of mass in this nominal pose, with knees slightly bent.

The blue trace shows that the torque sensor exhibits artifacts, but that it approximately indicates a (small-angle) linear model (the green trace) corresponding to a slope of approximately 1650 N-m/rad. Note that the nominal pose deliberately placed the center of mass slightly forward of the ankles (closer to the centroid of the foot pad), so the model fit for IMU tilt vs ankle torque does not pass through zero. Nonetheless, the slope of the model fit reveals the desired mass properties. The weight of the robot, as reported by the foot force/torque sensors, was 1500N, and thus the identified gravity model corresponds to a mass of 153 Kg located 1.1 meters above the ankles (in the nominal test pose).

For balancing using only an ankle joint, shifting the COM by 6cm laterally would result in falling. Equivalently, with the COM at 1.1m above the ankle, this corresponds to leaning a mere 0.054 rad of tilt angle.

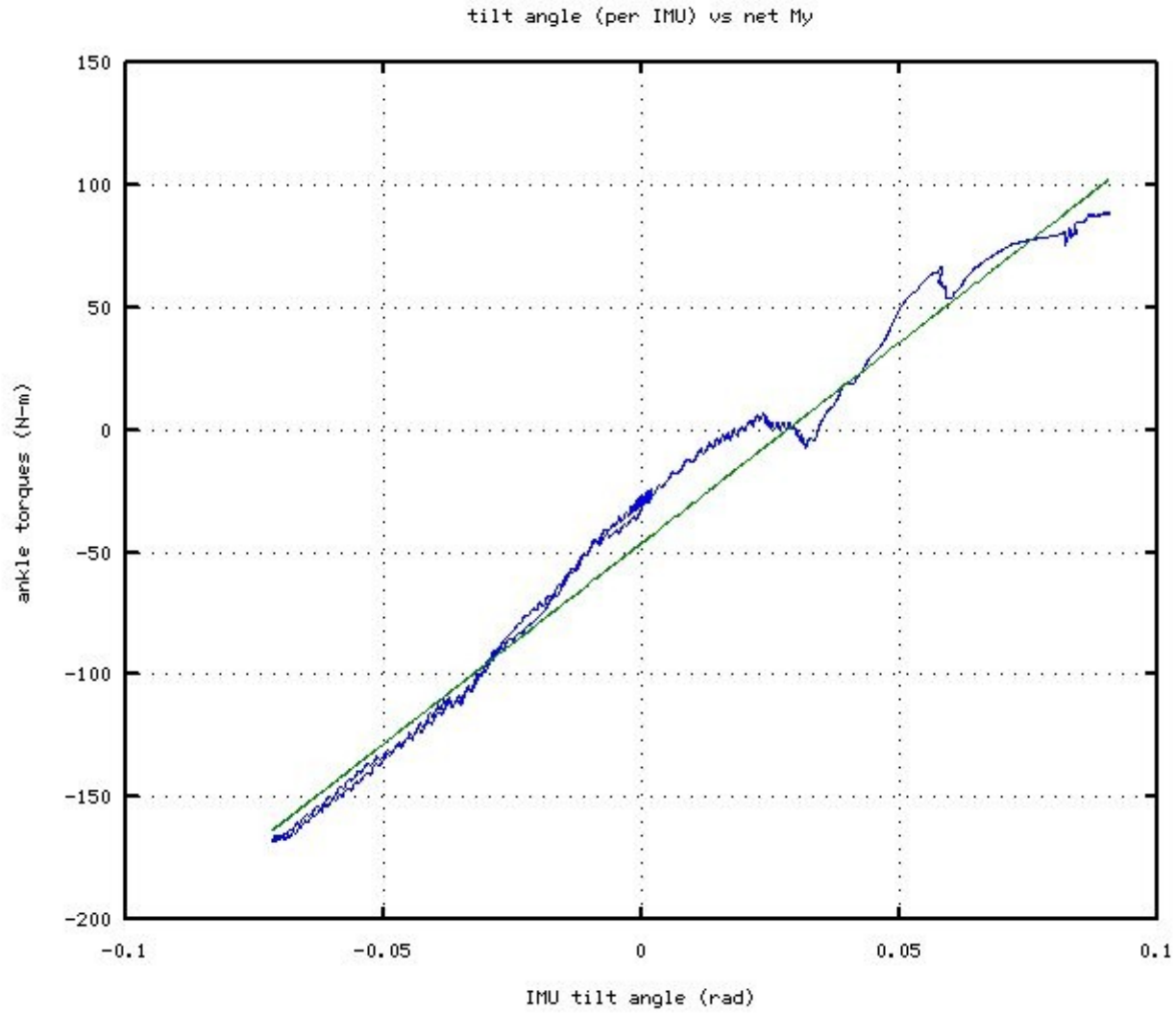

Illustration 1: Model of Atlas gravity torque function from tilting experiment

For subsequent control analysis, the two important quantities from this analysis are: torque due to gravity is approximately: $\tau_g = K_g \theta_g$, where τ_g is torque about the support ankle due to gravity, θ_g is the body tilt about the ankle, relative to gravity, and $K_g = m g L$ = 1,650 N-m/rad. The small-angle approximation is valid, since tilt of greater than 0.06 rad will result in falling. Additionally, with width of the foot limits ankle torque values to $\tau_{ankle,max}$ = 88N-m.

The same experiment and analysis were performed with simulated Atlas via drcsim. The gravity model compares favorably, at 1,580 N-m/rad.

Atlas Inertia Modeling: A second important parameter to identify is the inertia of the entire body rotating about the support ankle. In general, this is a function of the whole-body pose. For the present analysis, a specific pose was chosen, with Atlas balanced on his right foot, arms outstretched to the sides (the zero-ankle "home" pose of the arms), right knee slightly bent, left leg slightly raised, and torso leaning to the right, such that the center of mass was approximately in the center of his support (right) foot. (See Fig 2).

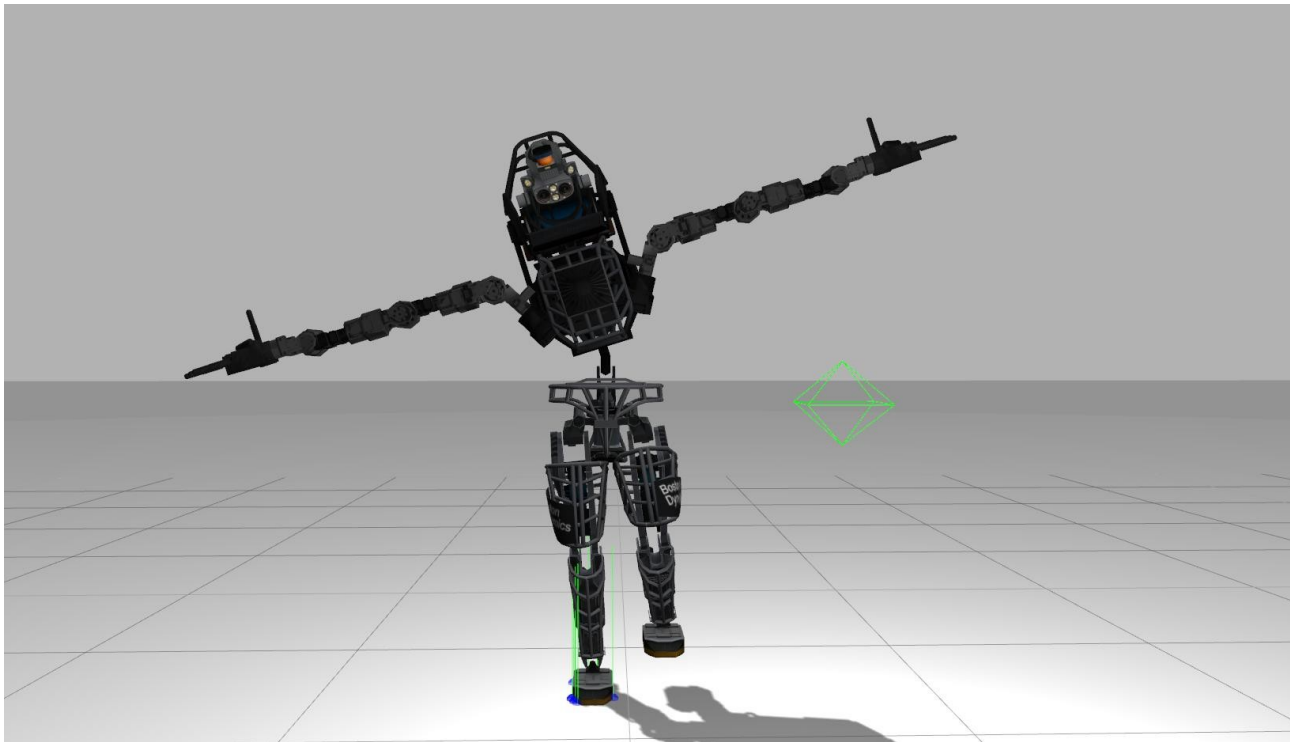

Illustration 2: Nominal balancing pose on right foot

The chosen pose was obtained experimentally, starting from a nominal pose for which Atlas can balance on one foot in drcsim. However, the actual robot was not able to balance in this pose, and thus waist tilt and ankle roll adjustments were made manually until example postures were found for which Atlas could balance without feedback. That is, with fixed angle setpoints, Atlas could barely maintain balance on one foot. It was observed that these poses were not entirely reproducible. Presumably Coulomb friction in the joints and perhaps hysteresis in the rubber foot pads resulted in these experimentally-determined poses being barely stable. With some trial and error, though, Atlas could be made to balance in such poses without a balance controller. At the same time, these experiences made it clear that active feedback for balance is essential.

After getting Atlas to tenuously balance in the nominal pose, a controller deliberately commanded the right (support) ankle to rotate slowly, causing Atlas to fall sidewise. IMU and foot-pad force-torque data were logged during falling, and this data was post-processed to identify inertia.

The process was first attempting insimulation, using drcsim. Figure 3 shows the data acquired during simulated falling, from the initial pose of balancing on the right foot. Figure 3 shows the measured ankle torque, the computed gravity torque about the support ankle and the net torque about the support ankle. The red trace (ankle torque) starts positive, as the ankle induces the fall, then saturates at approximately -83 N-m, as anticipated. The torque about the support ankle due to gravity was computed based on the identified gravity function and using the tilt angle reported by the (simulated) IMU. The sum of these effects—the blue trace—is the net torque acting on the body.

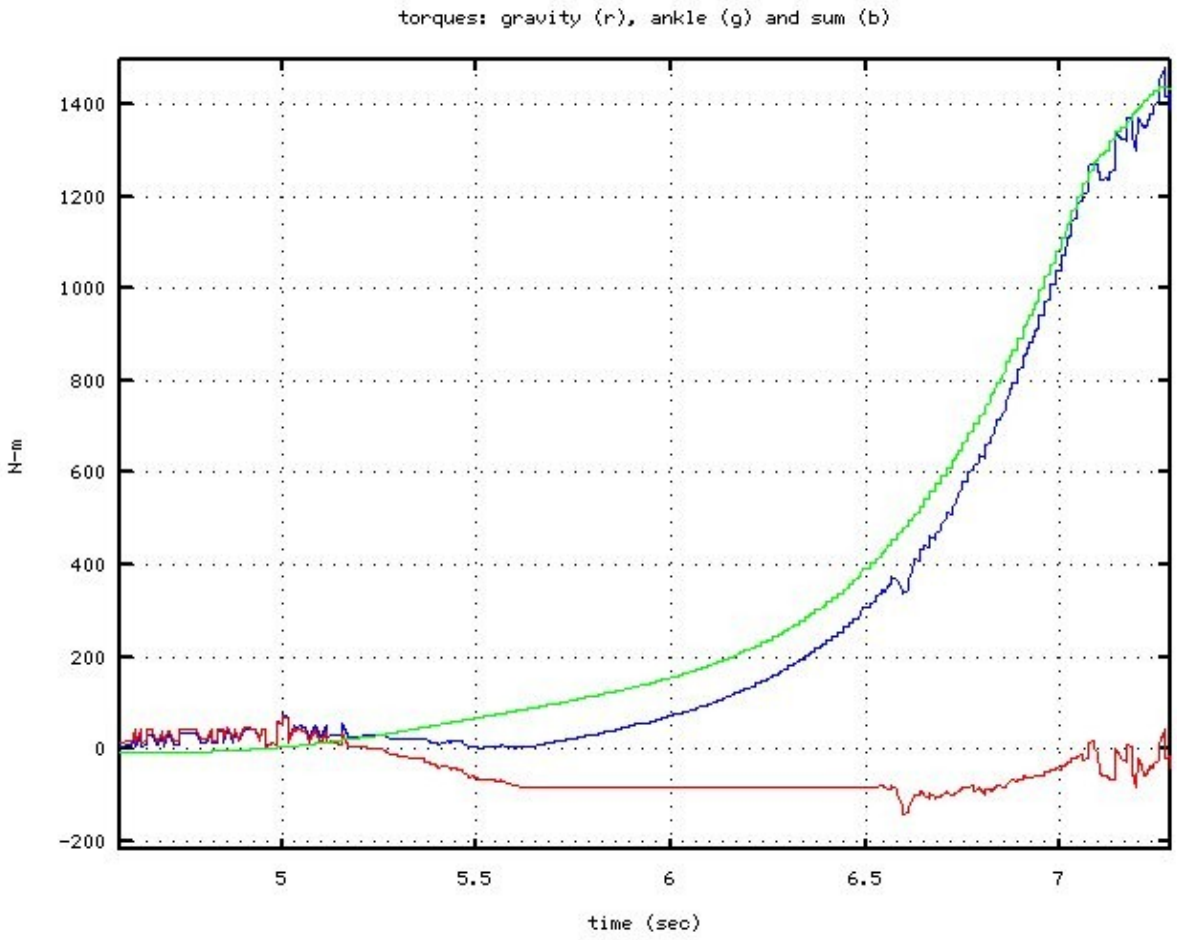

Illustration 3: drcsim gravity, ankle and net torques during falling experiment

To identify inertia, a model fit was performed comparing torque impulse (integral of net torque with respect to time) and angular momentum (angular velocity times rotational inertia about the support ankle). The result is shown in Fig 4. The blue trace is the time integral of net torque, and the red trace is the angular momentum (from the IMU) times a constant, H11. The constant H11 was adjusted manually, attempting to match the estimated angular momentum to the computed torque impulse. The result shown was obtained using a value of H11=240 Kg-m^2.

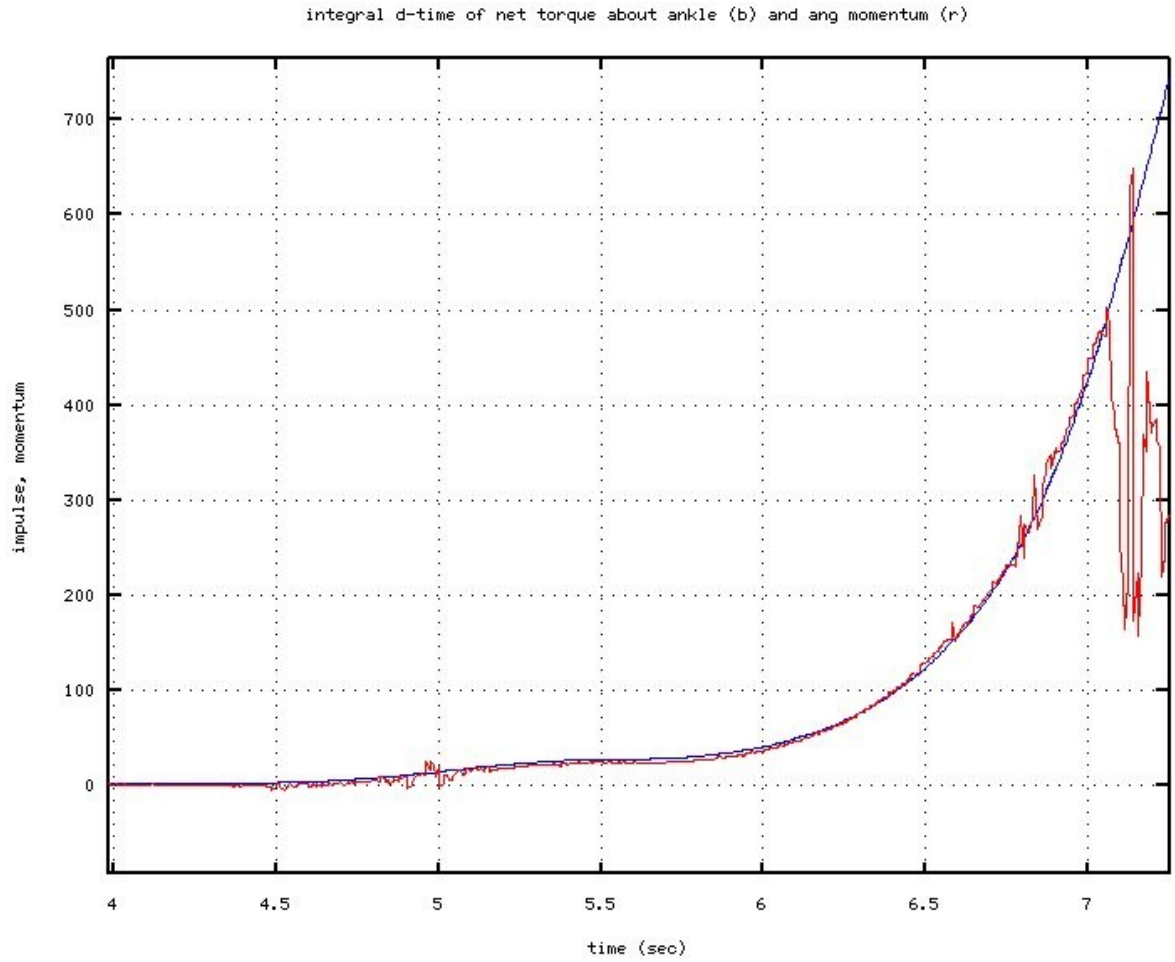

Illustration 4: drcsim inertia modeling from falling data: impulse and angular momentum

The same experiment was performed on Atlas. The resulting ankle torque, computed gravity torque (based on the prior gravity modeling) and net torque during the fall are shown in Fig 5.

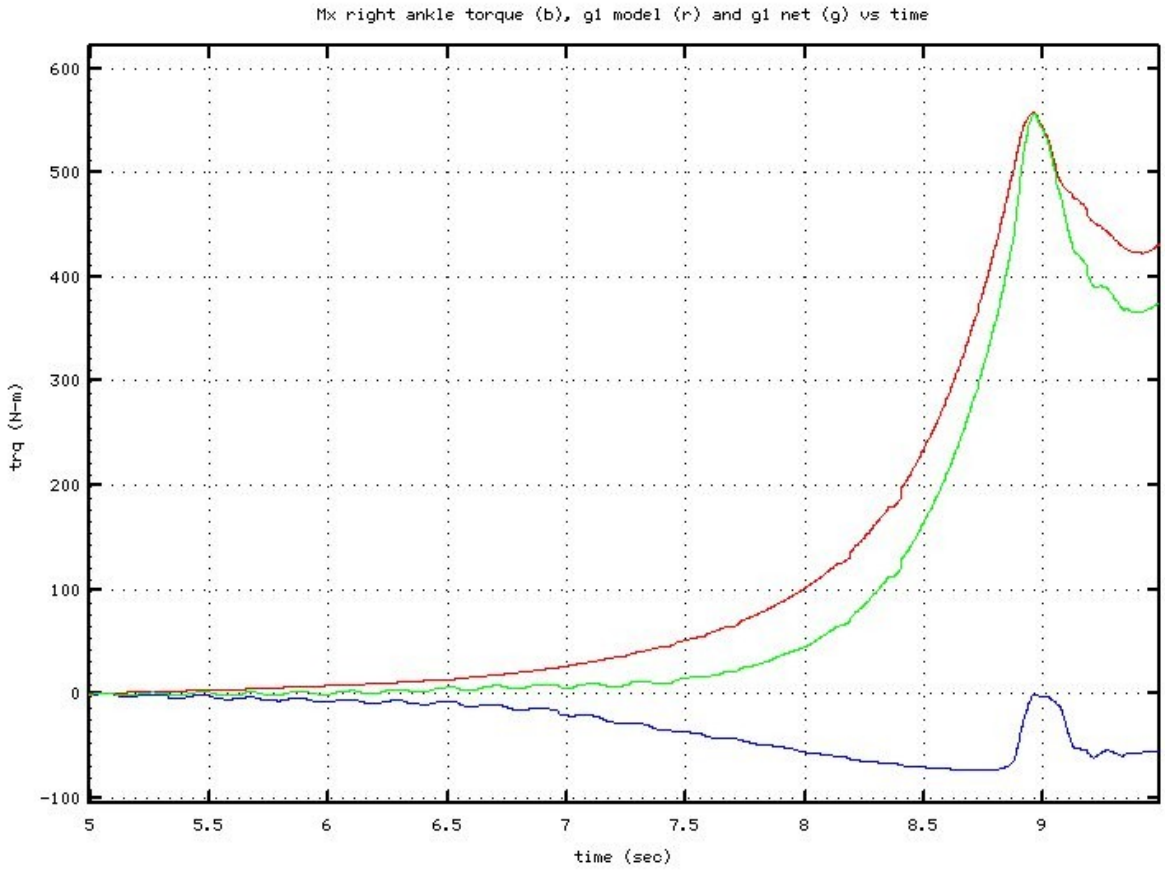

Illustration 5: Atlas experiment: Ankle, gravity and net torques during falling

Figure 6 shows the corresponding attempt to match impulse of torque to angular momentum. The experiment with Atlas had a shorter window of valid data (barely 2 seconds), as Atlas was rescued from falling earlier than the range of valid data in simulation. Nonetheless, the integral of the net torque compares well with a fit of angular momentum (based on IMU yaw rate data) using an inertia value of 260 Kg-m^2. This value is based on manual inspection, though the quality of fit visually implies that the inertia values of 250 and 270 are both worse than the identified value of 260, placing a rough bound on the precision. This value also compares well with drcsim at merely 8% higher.

From the record of Fig 5, the peak observed ankle torque was -73 N-m. This is lower than computed and lower than drcsim. The reason is uncertain. It may be that compression of the foot-pad, with weight concentrated on the edge, results in a lower net moment. It may also be that the foot-pad force-torque sensors yield incorrect values as the weight is concentrated on one edge. Nonetheless, it would be prudent to assume that a balance controller is limited to torques no larger than this value.

The conclusion of mass characterization is that the inertia of Atlas (in the chosen nominal pose) about one ankle is approximately: H11 = 260 Kg-m^2.

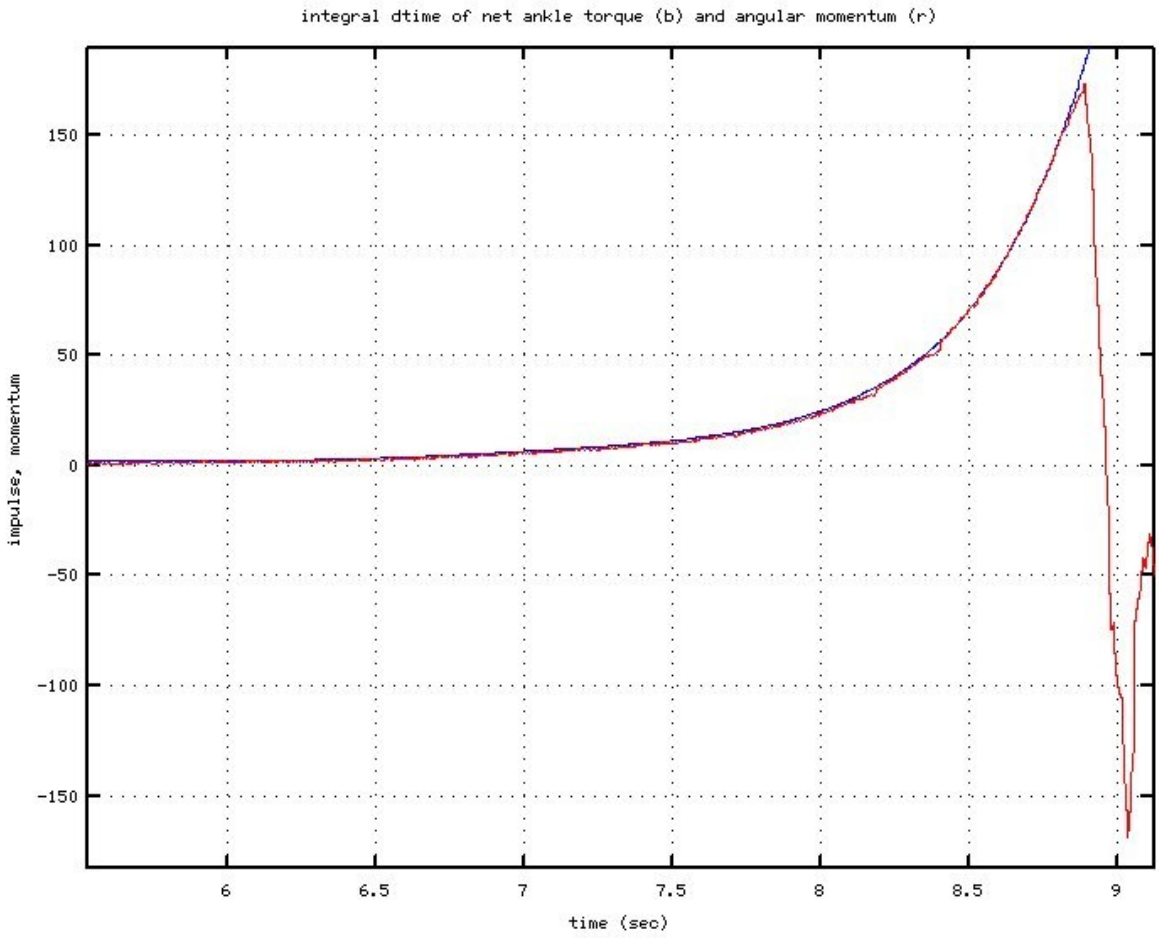

Illustration 6: Atlas falling analysis: fit of angular momentum at H=260

Atlas control interface: Maintaining Atlas' balance through use of ankle torques alone is conceptually a simple control problem corresponding to a single inverted pendulum. Typically, a linear controller would apply control torques proportional to tilt and to angular velocity (PD control) to maintain stability. However, direct access to control torques is a challenge with Atlas. There is no direct torque actuation available. Rather, ground reaction torques occur indirectly via the angle of contact between the footpad and the ground.

The Atlas interface does report equivalent ankle torques (deduced from fluid pressure differentials and closed-chain kinematics with the two ankle actuators). Further, the control interface does allow for performing feedback with respect to actuator effort. However, such an approach presents a variety of complications. One issue is that it is most often desirable to control the angle of the foot (position control), and this conflicts with attempting to control the torque on the ankle. For example, if the foot is lifted off the ground, only zero torques are possible. Any attempt to control torque in this pose would result in the ankle being driven to a hard-stop angle limit. Additionally, the torque reported by the actuator efforts is significantly affected by Coulomb friction. Figures 7 and 8 illustrate this point.

Figures 7 and 8 show data from Atlas standing on one foot while the ankle is commanded to oscillate. The commanded ankle roll velocity is shown in blue. (The simultaneous roll angle command was mathematically consistent as the integral of this velocity command). The corresponding response is shown in red. During this oscillation, the reported actuator efforts and the foot-pad ankle torque values were recorded. These are plotted against each other in Fig 8.

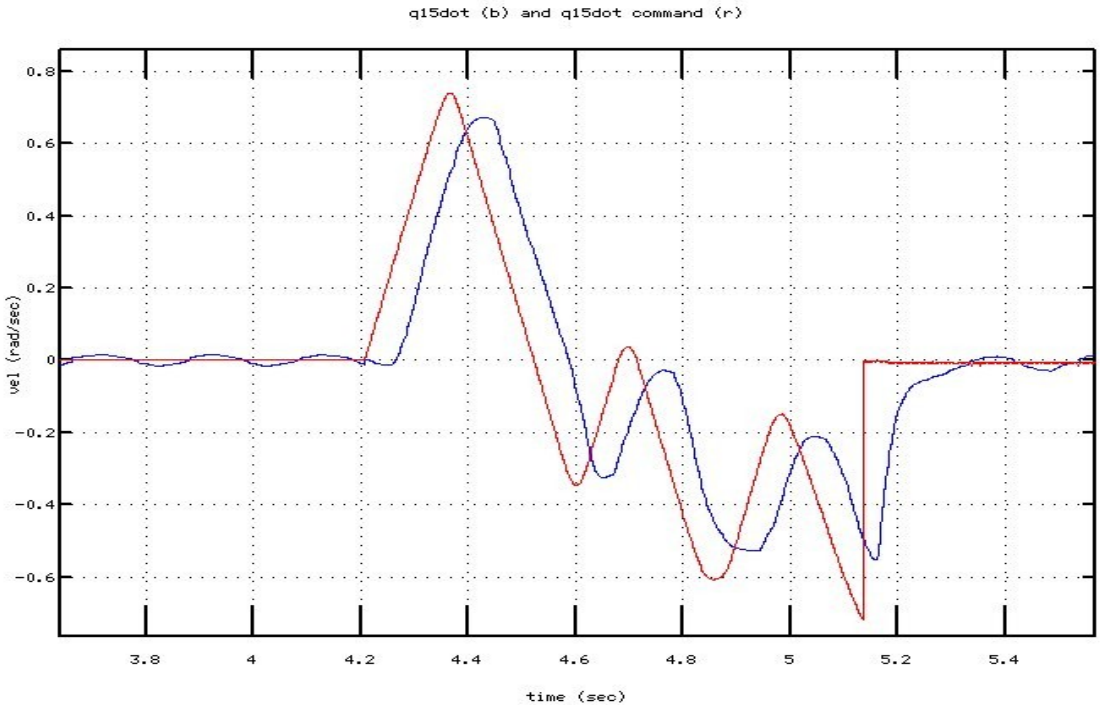

Illustration 7: Ankle oscillation experiment with Atlas standing on one foot

Figure 8 shows that there is substantial hysteresis between torques reported by differential fluid pressure within the actuator vs ankle torque reported by the foot-pad force/torque sensor. The hysteresis has been observed in other Atlas joints, and it is attributable to Coulomb friction within the actuators. As can be seen from Fig 8, commanding an actuator effort of 0 N-m can result in an actual ankle torque of anywhere between -5 and +10N-m. This uncertainty in control effort can make stability challenging.

In this presentation, it will be assumed that the balance control efforts must be exerted through position and velocity commands, although this also presents challenges. As can be seen from Fig 7, the ankle controller adequately followed velocity commands slewing at approximately 6 rad/sec^2. However, the angular velocity response was delayed by approximately 60msec. A controller that uses this velocity-command interface will have to tolerate this latency.

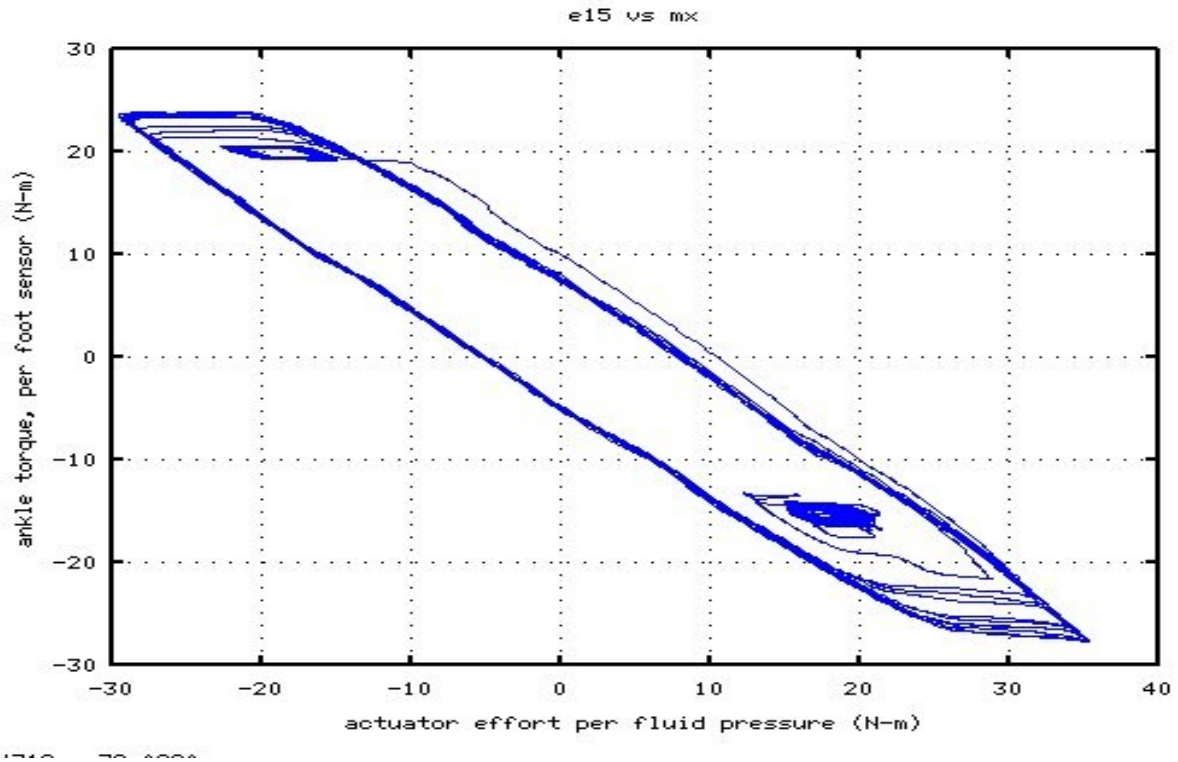

Illustration 8: Atlas actuator effort vs foot-pad torque during ankle oscillation

Another challenge to a position/velocity input control interface is that it is unclear what commands will lead to a required ground reaction torque. Tilt of the foot with respect to the ground results in reaction torques, which can be modeled as an equivalent torsional spring. For the excitation of Fig 7, the foot-to-ground angle was estimated, taking into account both whole-body tilt (per the IMU) and ankle angle (per the reported ankle sensor values). Figure 9 shows a plot of the ankle torques reported by the foot-pad sensor vs the estimated tilt of the ankle relative to ground. The blue trace is the sensed torque vs inferred angle. The red line is a model approximation treating the footpad as a linear torsional spring. The green trace shows part of nonlinear torsional stiffness model using piecewise-linear segments.

The linear model of footpad torsional stiffness is approximately 1,500 N-m/rad (although this is valid only over a small range of tilt angles). It is noteworthy that this torsional stiffness is *less* than the destabilizing equivalent gravity stiffness of $K_g = 1650$ N-m/rad. Further, once Atlas starts to tip, the footpad stiffness decreases, but the gravity influence does not. As a result, it is understandable why it is very difficult to balance Atlas at fixed joint angles, even if the the center of mass is initially well positioned over the ankle.

Although the foot-pad torsional stiffness is only comparable to (slightly lower than) the destabilizing gravity influence, the footpad torsional stiffness is nonetheless high enough to present control challenges, since quite small errors in estimated foot-to-floor angle result in large torque estimate errors. Further, the bias angle (angle at which the foot torque is zero) is uncertain and, in fact, variable. The blue trace of Fig 9 shows that foot torque vs foot angle is not reproducible. The compliant footpad have be introducing hysteresis. Further, debris picked up by the footpad can result in large changes to the torque vs angle curve. Additionally, the actual surface normal of the ground under the foot is not known a priori. In general, attempting to estimate an ankle angle that will produce a desired ground-reaction torque is impractical.

The approach taken here is to assume only a differential relationship. That is, with the foot in contact with the ground, a positive ankle angular velocity will lead to an ankle reaction torque that

is decreasing in time. The rate of decrease depends on the equivalent footpad torsional stiffness. For small ankle torques, this slope is approximately 1,500 N-m/rad. For larger angle torques, the torsional spring softens to much lower stiffness. The green trace of Fig 9 shows part of a nonlinear model derived from a piecewise-linear model of torsional stiffness as a function of ankle torque.

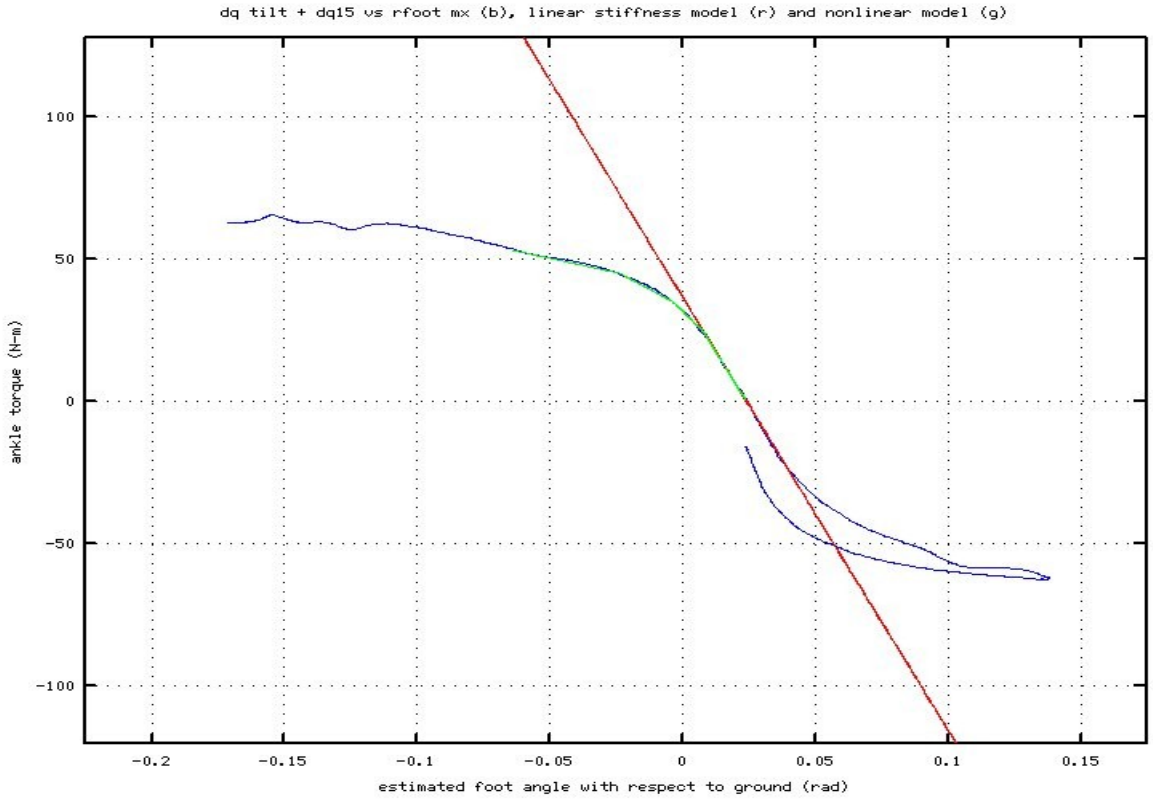

Illustration 9: footpad torsional stiffness modelling

From this analysis, the conclusion is that imposing a desired command torque is not directly achievable. Use of direct ankle actuator torque commands is problematic, and use of an interface accepting position and velocity commands is limited by maximum velocity, maximum acceleration and suffers from a time-delayed response. Further, one cannot expect to command an ankle angle that will achieve a desired ground-reaction torque. The mapping is too sensitive and irreproducible.

The approach taken in the following is assume that ankle velocity commands will be used to invoke time rate of change of ground reaction torques. Although we cannot expect to achieve a target torque at a knowable angle, we can expect that the rate of change of torque is related to the angular velocity of the foot with respect to ground (including both full-body tilting and ankle roll rate).

State-Space Model: The control approach taken here is to assume that angle angular velocity commands are followed well by the eBox controller. Per Fig 7, it is clear that this assumption is limited. Ankle commands are constrained by maximum angle limits, maximum angular velocity limits and maximum angular acceleration limits. Further, command following showed an approximately 60ms latency. A controller based on ankle angular velocity commands will have to tolerate these limitations. Proceeding under this assumption, the system is modeled as follows.

$$\tau_{net} = K_g \theta_{tilt} + m_{ankle}$$

In the above, the net torque about the ankle is the summation of the torque due to gravity and torque due to ground reactions with the foot. The torque due to gravity is estimated using the identified center-of-mass properties, linearized for small angles via the constant K_g. The torque from ground reaction with the foot is measured by the foot-pad force/torque sensor. The displacement of the center of mass relative to the ankle is inferred from the whole-body tilt angle via the IMU. Note, though, that there is an unknown bias angle that must be discovered—the IMU tilt ankle corresponding to zero gravity torque. The net torque computation is performed relative to this reference tilt angle.

Given the net torque on the body about the support ankle, and assuming all joints except the support ankle roll angle are held fixed, the dynamics of Atlas in the lateral plane is described by:

$$H_{11} \ddot{\theta}_{tilt} = \tau_{net}$$

where H_{11} is the inertia of the body (in the chosen pose) about the support ankle. The angular acceleration of the body in the lateral plane is not measured directly. However, the angular velocity is measured very well by the IMU (yaw rate about the pelvis x axis). Defining the "displacement", D, as:

$$D = H_{11} \theta_{tilt}$$

and the angular momentum, h, about that ankle as:

$$h = H_{11} \dot{\theta}_{tilt}$$

It follows that:

$$dD/dt = h$$

and

$$dh/dt = \tau_{net}$$

As discussed above, we cannot assume direct control of net torque. However, one can approximately control the rate of change of net torque, according to:

$$\dot{\tau}_{net} = (K_g / H_{11}) h + d\, m_{ankle}/dt = K_g \dot{\theta}_{tilt} + d\, m_x / dq (m_x)(\dot{\theta}_{tilt} + \dot{q}_{ankle})$$

Using a (nonlinear) model of the foot-pad torsional stiffness, one can obtain an estimate of $d\, m_x / dq$ (the incremental stiffness of the footpad) as a function of the current ankle torque, as measured by the foot-pad sensor. To achieve a desired value of $\dot{\tau}_{net}$ using the input command of ankle angular velocity, $\dot{q}_{ankle,cmd}$, one can compute the ankle velocity that should achieve the desired effect as:

$$\dot{q}_{ankle,cmd} = [\dot{\tau}_{net,desired} - (K_g + m')\dot{\theta}_{tilt}]/m'$$

From the above, one can attempt to achieve a desirable time-rate of change of net torque via ankle velocity command, subject to a variety of requirements and restrictions. The value of $\dot{\theta}_{tilt}$ can be obtained from Atlas's high-quality IMU. The appropriate value to use for $m' = d\, m_x / dq$, the incremental torsional stiffness of the foot pad, can be estimated from the current ankle torque (per the foot-pad force/torque sensor) and from a model, e.g. that of Fig 9.

Restrictions include maximum achievable ankle velocity, maximum ankle angle, and maximum ankle acceleration (lumped in with following latency). Additionally, the net torque cannot be increased if the foot has already reached saturation torque (approximately +/-70N-m), corresponding to Atlas standing on one edge of his foot. The incremental footpad torsional

stiffness, m', appears in the denominator of the derived ankle velocity command formula. Note that as the foot torque approaches its limit, this slope approaches zero, and the feedback linearization computation can suffer from a divide by zero (in the limit), or may yield impractically large ankle velocity commands. Such limitations must recognized in terms of saturation of commands in the control algorithm and experimental tuning of control implementations.

Subject to these restrictions, we can proceed with a control-system design for the above 3rd-order system. Assuming use of the above feedback linearization mapping ($\dot{q}_{ankle,cmd}$ that produces a desired $\dot{\tau}_{net}$), our system model is linear in D, h and τ_{net}, as described by:

$$\dot{\mathbf{x}} = \frac{d}{dt}\begin{bmatrix} D \\ h \\ \tau_{net} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} D \\ h \\ \tau_{net} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \mathrm{u}$$

where u is the control input, $\dot{\tau}_{net}$, which is to be implemented through the corresponding ankle velocity command, $\dot{q}_{ankle,cmd}$.

A linear controller could be designed for this simple plant. However, the actuator effort is limited by saturation, which is a nonlinear function of state. Designing a controller that always avoids actuator saturation is a possible approach, though assuring avoidance of actuator saturation can result in a highly conservative (i.e. unacceptably slow) controller. An alternative is to explicitly consider actuator saturation limits and implement a sliding-mode controller.

Under sliding-mode control, one chooses a desirable sliding surface, means to approach this surface, and means to stay on the surface. A candidate sliding surface is the following state-space constraint equation:

$$0 = \lambda^2 D + 2\lambda h + \tau_{net}$$

If the system can be coerced to approach and maintain satisfaction of this constraint equation (sliding surface), then the ensuing dynamics will correspond to convergence to the goal state, **x** = **0**, with time constants $1/\lambda$.

The constraint equation may be expressed as a scalar function of state, s(**x**), which is merely a linearly weighted sum of state variables. It is desired to achieve and enforce that s(**x**)=0. This is accomplished first in an "approach" phase to converge on s=0. To do so, a control effort is exerted to guarantee that the Lyapunov function $V = 1/2\, s^2$ is decreasing, i.e. $dV/dt = s\dot{s} < 0$. To guarantee that $s\dot{s} < 0$, the controller should enforce the sign of $\dot{s}$ such that $sign(\dot{s}) = -sign(s)$.

To find the appropriate control input, examine the terms of $\dot{s}$:

$$\dot{s} = \lambda^2 h + 2\lambda \tau_{net} + \dot{\tau}_{net}$$

To satisfy $sign(\dot{s}) = -sign(s)$, exert the control effort $\dot{\tau}_{net}$ according to:

$$\dot{\tau}_{net} = -\lambda^2 h - 2\lambda \tau_{net} - sign(s)\hat{\dot{\tau}}_{net}$$

$$= -\dot{s}_{nom} - sign(s)\hat{u}$$

With the above formula, the term $\dot{s}_{nom}$ approximately cancels the system's tendency to drift away from the sliding surface, and the term $sign(s)\hat{u}$ exerts an additional effort of magnitude $\hat{u}$ to drive the system towards the surface s(**x**)=0. Typically, however, exerting a switched control (using the sign of s) results in chatter near the sliding surface. An approximate fix to this problem is to replace the "sign" function with the "sat" (saturation) function, resulting in:

$$\dot{\tau}_{net} = -\lambda^2 h - 2\lambda\tau_{net} - \hat{\dot{\tau}}_{net} sat(s/K_s)$$

The saturation function is identical to the sign function for values of $|s| > K_s$.

This sliding-mode controller has 3 tunable parameters: λ , $\hat{u}$ and K_s . The value of λ determines how fast the system is intended to converge on goal state along the sliding surface (with larger values of λ implying faster convergence). This value should be chosen to be fast enough (if the system is too slow, it will fall past the ankle torque saturation limit before correction can be achieved), yet not too fast. (Demanding excessive speed of convergence will prescribe an unachievable convergence rate, with unpredictable results). The value of $\hat{u}$ must be chosen to be large enough to dominate modeling uncertainty, as well as to enforce fast enough convergence to s=0 (before the robot falls). However, it is also attractive to choose this value to be as small as practical, so as to minimize excitation of higher-order dynamics. The term K_s sets the "width" of the "boundary layer" about the sliding surface. Use of the "sat" function in place of the "sign" function is only an approximate fix to the chattering problem. Convergence of the system inside the boundary layer is not guaranteed. However, if the boundary layer is narrow enough (i.e., small enough K_s) then uncertainty of the plant state dynamics within this boundary layer is not of practical importance. Minimizing the boundary layer is desirable in terms of bounding the plant dynamics relative to the sliding surface. However, as this value is decreased to zero, the "sat" function approaches the "sign" function, and the problem of chatter is re-introduced. Finding an acceptable value for K_s is a matter of empirical tuning.

Experimental results: The program "right_ankle_balancer_v2.cpp" within the package "balance_behavior" implements the above 3rd-order sliding-mode balance controller. The following parameter values were tested (though not optimized): $\lambda = 2$ (rad/sec), $K_s = 50$ (N-m) and $\hat{u} = 200$ (N-m/sec). Atlas was balanced on one foot manually (with joints frozen, no feedback), after which the balancer program was run for approximately 24 seconds, which included an initial 4-second "warm-up" period before control was active.

While balancing, a sequence of 5 force disturbances were introduced by pushing on the right hand. These disturbances, shown in Fig 10, were logged via the wrist force/torque sensor ranging from 2.5N to 14N. The force disturbances integrated in time corresponded to impulses ranging from 2.5 to 14 N-seconds (and these values are roughly consistent with the observed changes in angular momentum).

Figures 11 and 12 show the robot's response to these disturbances. The ankle torques are shown in Fig 11. The largest response (rougly 60N-m) is close to the maximum achievable ankle torque (approx. 70 N-m). Figure 12 shows the response in terms of body tilt angle. The time integral of IMU-reported angular velocity is superimposed on the tilt angle, with remarkably good agreement. This comparison shows that the IMU yaw-rate signal is reliable and consistent with the tilt angle reported per Atlas's sensor outputs. Response to disturbances converges within 3 seconds with relatively low overshoot and good damping. The reaction of the controlled ankle torque vs body tilt angle achieves a stable net stiffness of approximately 1,000 N/rad (as compared to 0 or negative stiffness without feedback). Further, the damping exhibits an attractive response.

Figure 13 shows the ankle angle commands and angle response that achieved the balancing. These commands were derived from integration of the ankle velocity commands, which were computed from feedback linearization and sliding-mode control, subject to velocity saturation limits.

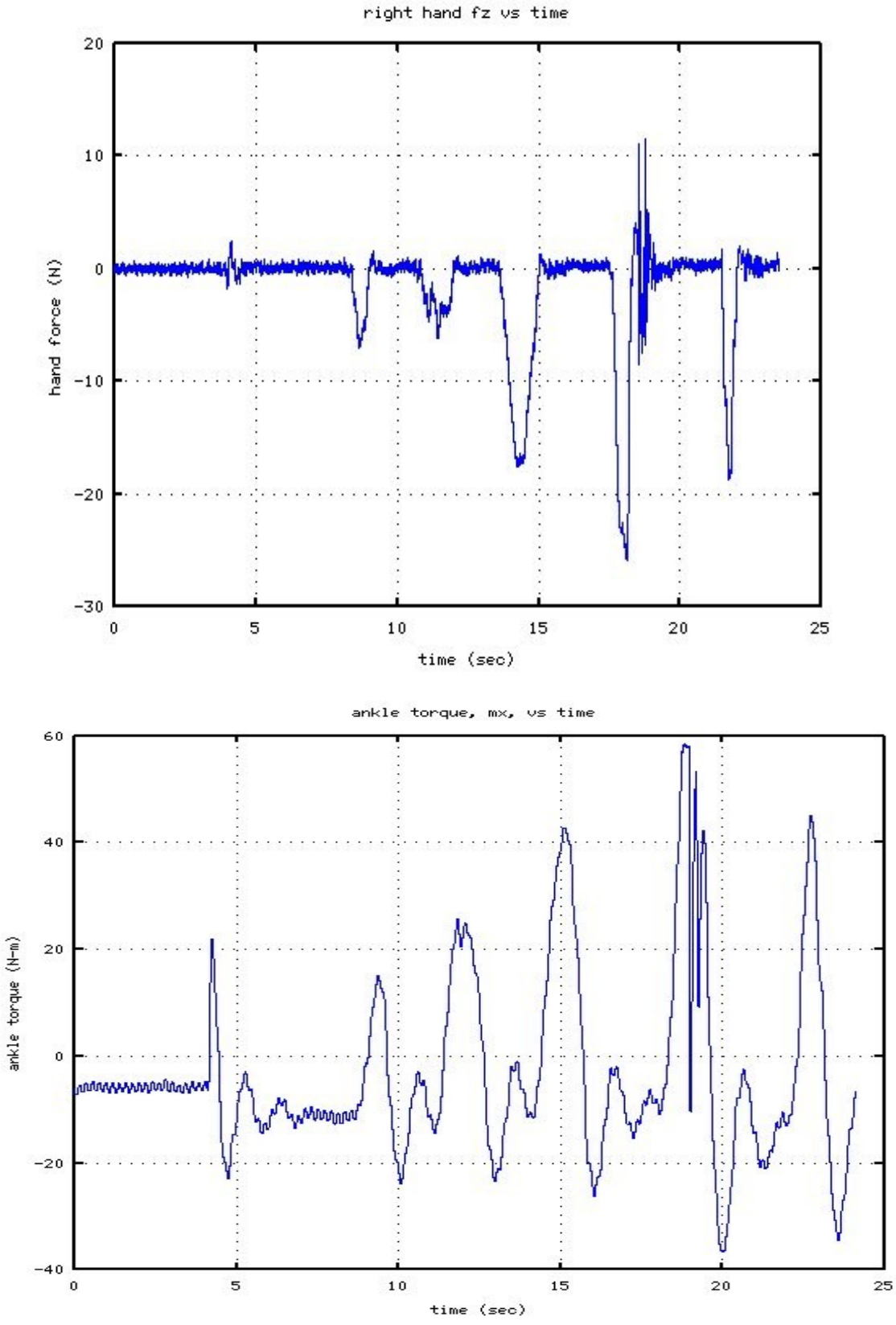

Illustration 11: Ankle torques in response to disturbance forces

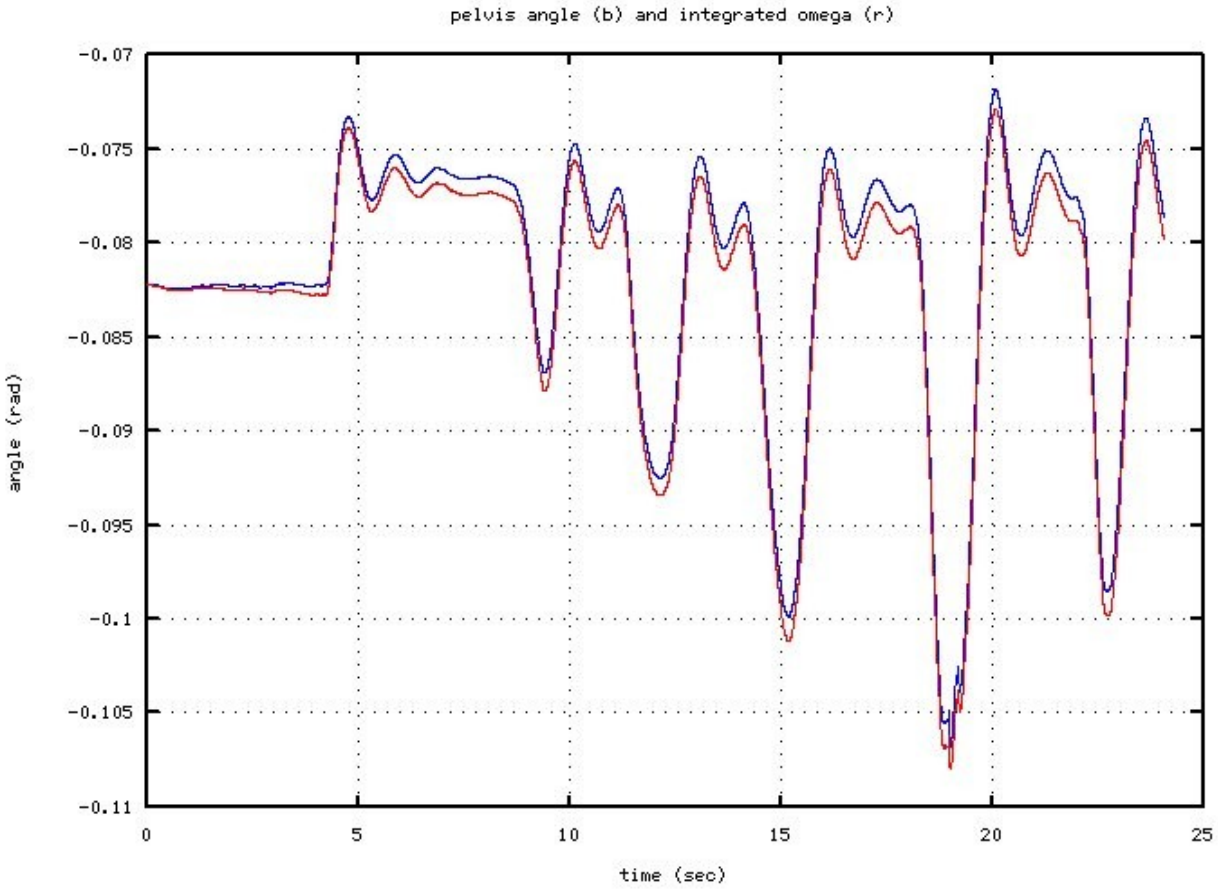

Illustration 12: pelvis angle and integrated yaw rate in response to disturbances

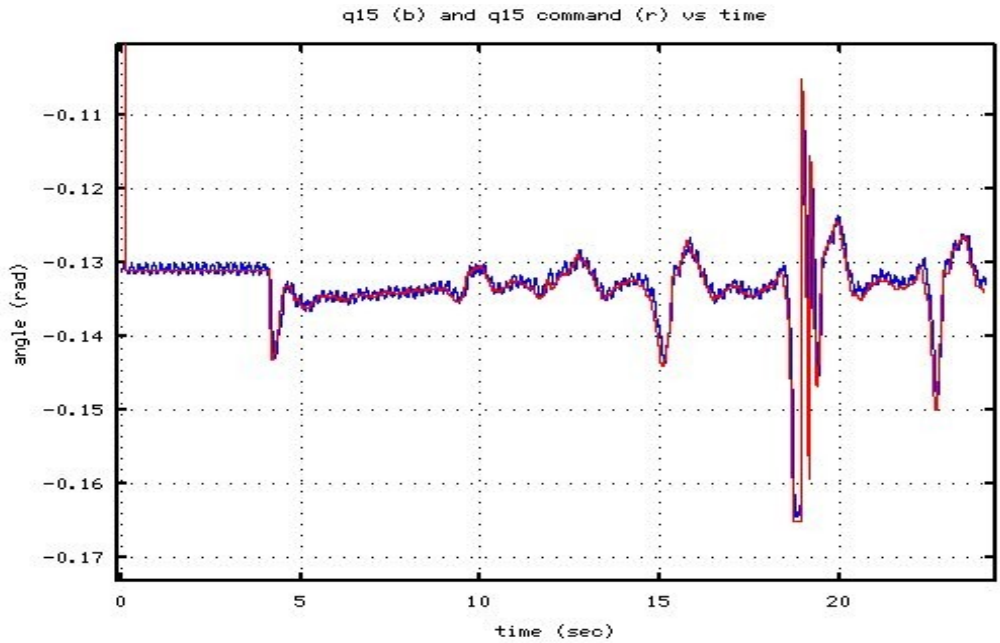

Illustration 13: Angle angle commands issued during balancing: red= command; blue=response

Figure 13 shows the dynamics of the sliding-mode controller. The sliding surface, s(t) is comprised of weighted components of body tilt, angular velocity, and net torque about the ankle. With consistent units of the weights, each of these components has units of N-m. Ideally, s(t) would

remain close to zero. The objective of the sliding-mode controller is to drive s(t) to zero then keep it at zero. The individual components of s may be large compared to s during transients, but as long as their weighted sum is close to zero, the plant will emulate a linear, 2nd-order system with time constants proportional to $1/\lambda$.

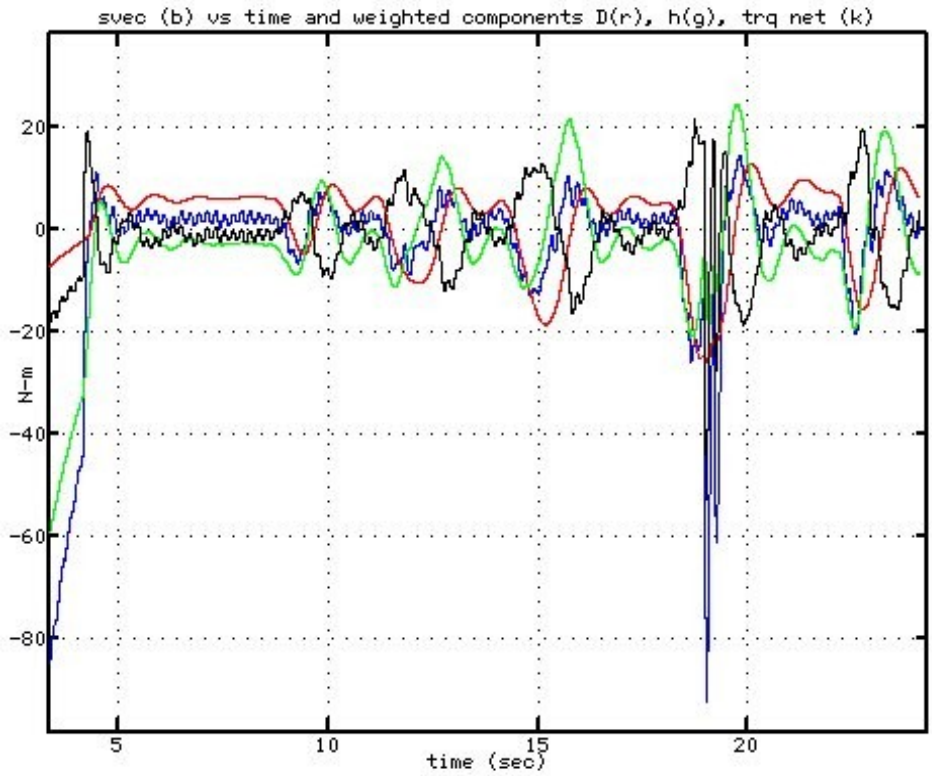

Illustration 14: s(t) and the constituent components during response to disturances

Additional Control Program Features: asdf
pelvis tilt wrt gravity: define frames;

trqnet_observer → tilt_x_obs and omega_x_obs2; tries to infer gravity from integration of net trq
observer ang vel is side effect (removes vibration noise)

third_order_smc(): returns ankle velocity cmd, saturated at max qdot; uhat set at 200
callback integrates qdot-> qcmd; but saturates at chosen joint limits, and limits more motion if at/near 50N-m of ankle torque

4-second warm-up for observers: with omega set at only 1.0; very slow to converge! (4sec)

xxxxxxxxxxxxxxx

drcsim Differences: asdf

Future Work: testing, tuning, integration w/ motion, transitions in stepping;
faster stepping (w/ inertial considerations);

integrate as 1 phase w/in switched controller for balance recovery;

establish IC's such that this controller remains valid (before transition to emergency response)