Identification and control of a pneumatic robot

Emanuel Todorov[1], Chunyan Hu[1], Alex Simpkins[1] and Javier Movellan[2]
[1]Applied Mathematics and Computer Science and Engineering, University of Washington
[2] Institute for Neural Computation, University of California San Diego

***Abstract*— Pneumatic actuators have a number of advantages over electric motors, including strength-to-weight ratio, tunable compliance at the mechanism level, robustness, as well as price. Their properties are in many ways similar to muscle properties, which further makes them a good choice for bio-inspired robotic designs. However they are considered harder to control, and often avoided. Here we report results on modeling and control of a 2-dof robot, as well as preliminary results on a state-of-the-art 38-dof humanoid. Contrary to popular belief, we found it surprisingly easy to work with these pneumatically-actuated robots and obtained high tracking performance. We were also able to achieve end-effector control of the highly redundant arm of the humanoid. This was done by building parametric models, fitting them to experimental data, designing model-based feedback controllers, and optimizing their performance.**

I. Introduction and prior work

Pneumatic systems are becoming increasingly popular [1] due to their high speed and force capability, as well as relatively low price and overall robustness. From the perspective of bio-robotics, pneumatic actuators are further desirable because they have many of the essential properties of biological muscle at the mechanism level – which we believe is quite different from trying to re-create those properties using feedback control. In particular: (i) as the joint moves in the direction of actuated force, the chamber volume increases and thus the pressure/force drops, resulting in stiffness; (ii) this stiffness can be tuned by activating opposing cylinders, much like a biological limb becomes stiffer when antagonist muscles co-contract; (iii) the actuator has an internal activation state (air mass in the case of pneumatics, calcium concentration in the case of muscles) whose dynamics make the entire system 3rd-order; (iv) the latter dynamics effectively introduce a low-pass filter between command signals and forces, with similar time-constants for muscles and pneumatic cylinders; (v) since the actuators are often linear they can be mounted in a way reminiscent of muscle attachment to the skeleton, resulting in moment arms which vary with joint angle; (vi) the high force output makes gears and other amplification mechanisms unnecessary, which in turn results in uniquely compliant systems capable of dynamic interactions with the environment. One could argue that some of these properties are unnecessary complications. However, if we are serious about understanding the principles of biological control and replicating those principles in synthetic systems, it would be a mistake to ignore the fact that biological control has evolved in the context of the musculo-skeletal plant and is profoundly shaped by the unusual properties of this plant.

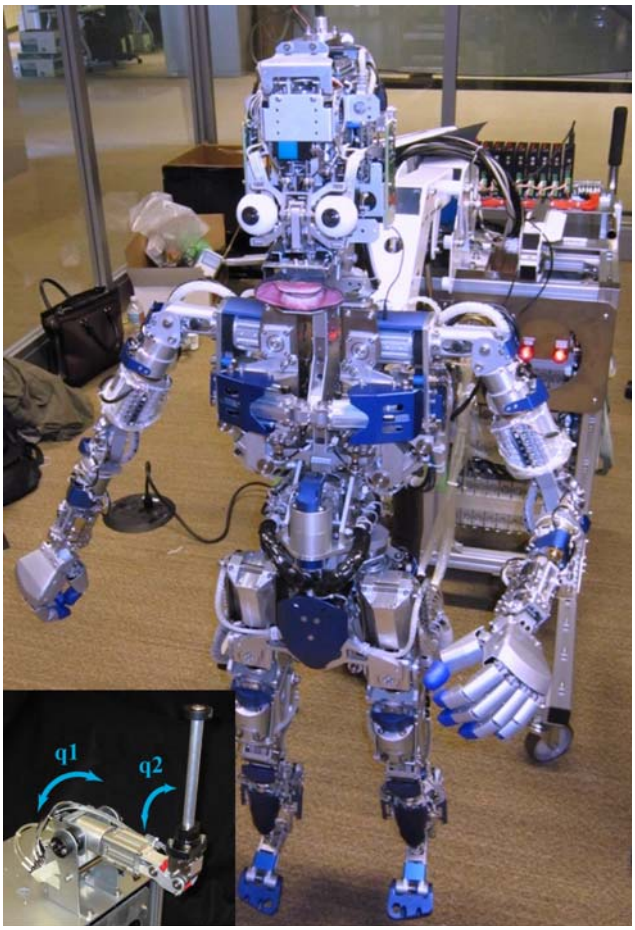

Fig. 1. Our 44-dof humanoid (DiegoSan) made by Kokoro Japan. The inset shows a simpler 2-dof robot made with the same sensors and actuators. All joints are actuated by pneumatic cylinders, with separate pressure sensors and valves for each chamber.

Due to the highly nonlinear properties of pneumatic modeling and uncertainties of various parameters, the control problems become challenging for systems with high precision requirement on force and position [2]. Various approaches have been proposed to cope with these problems. Early applications used a linearized state space model to develop an optimal regulator for a fixed operating point [1], [3]. Later, adaptive control was used for actuating an air power robot [4]. Other works on adaptive control include [5], [6], [7]. A literature also exist on applying PID control to pneumatic actuators [8], [9]. However it has been claimed that conventional PID feedback

controllers do not yield sufficiently effective performance for control purposes because pneumatic systems have quite complex pressure dynamics. Ref. [10] developed a robust controller using sliding mode techniques to drive the output tracking error to zero in finite time. Ref. [11] proposed proxy-based sliding mode control of a 2-dof planar manipulator actuated by pleated pneumatic artificial muscles, and obtained better tracking results compared with PID control. Additional sliding-mode controllers and other non-linear approaches can be found in [12], [13]. Thus far the literature on pneumatic control has focused on low-dimensional systems (1 or 2 dofs), and scaling to more complex robots remains an open question.

II. Robotic platforms

We recently acquired the humanoid shown in Figure 1, made by Kokoro in Japan. It has 44 pneumatically-actuated dofs; 6 of them are in the hands which were removed for the purposes of this paper, so here we are controlling "only" 38 dofs. Our long-term goal is to be able to make this humanoid perform various life-like movements. Before attempting such a challenging task, however, it makes sense to study a simpler system and find out what methodologies might work. The simpler system is the 2-dof robot arm shown in the inset in Figure 1. It is made by the same manufacturer using the same components. We now describe these robotic platforms in more detail.

Each joint is driven by either a linear or a rotary pneumatic cylinder. The drive is direct, without any gears, belts or cables (except for a couple of joints in the humanoid). This makes the system both more compliant and more robust – indeed the robot has been hitting its joint limits at high speeds during the system identification tests, without any damage. Each cylinder has two chambers fitted with solid-state pressure sensors. Each chamber is connected to a proportional valve, which can be open towards the compressor (supplying 7 bar pressure) or towards the room (where the atmospheric pressure is 1 bar) or can be closed. The joint angles are measured by potentiometers. Thus the 2-dof robot has 6 sensors and 4 controls. The humanoid has 114 sensors and 76 controls.

We are using National Instruments boards for both robots. The valves have 100Hz bandwidth, thus the control loop runs at 100Hz. The pressure sensors and potentiometers are analog devices that can be sampled at arbitrary rates. In the 2-dof robot we are sampling all sensors at 50KHz and averaging every 500 samples in order to obtain a 100Hz sensory-motor loop. In the humanoid we had to reduce this to 20KHz and 200 samples respectively, still yielding a 100Hz sensorimotor loop. Such averaging is beneficial because the sensor noise is essentially white. The software system consists of the NIDAQmx drivers, a mex function which reads the driver buffers and returns averaged sensor data, and Matlab scripts for everything else. To our pleasant surprise, we found that Matlab running under Windows (without any real-time extensions) results in a perfectly reliable real-time control loop – which we verified against the internal clock of the DAQ card.

III. Complexities of pneumatic actuation

Here we illustrate some of the unusual properties of pneumatic actuators using the 2-dof system. Unlike motor-actuated robots where the torque is proportional to the control signal, here the torque is proportional to the difference in pressure between the two chambers of the cylinder. What the control signal does is affect the rate of change of pressure. The way this happens will be discussed in detail later, but for now it is important to note that the pressure dynamics are inherently unstable: the pressure changes towards the maximum or the minimum value depending on which way the valve is open. Figure 2A shows the asymptotic pressures reached for different control signals (the valve controls are between 0V and 10V, with 5V being the closed position). If the valves were to operate as specified, the blue curve in the figure would be a step function. It turns out however that the valves operate slightly differently: for control signals near 5V, they are not completely closed but instead are somewhat open both towards the compressor and towards the room. As a result we observe asymptotic pressures at intermediate values. These asymptotes take much longer to reach (red curve), but can nevertheless be very useful because they introduce an element of proportionality in an otherwise all-or-nothing system. When designing linear feedback controllers in particular, we may want to operate in this regime.

Another effect of (almost) closing the valves is that the system becomes stiff at the mechanism level. It is easy to see why. Perturbing the arm changes the volume of the chambers. If the valves are closed, the perturbation causes a change in pressure which in turn generates a force opposing the perturbation. If the valves are open, the air can move in and out (usually faster than the mechanical perturbation) thus we have little stiffness. So the stiffness of the system can be controlled; high stiffness is obtained in the intermediate regime discussed above. This stiffness is rather complex, and involves what looks like an elasto-plastic phenomenon shown in Figure 2B. In this experiment all valves are closed (control = 5V). If the experimenter hits the robot arm (narrow pulses) it returns to its previous position in a slightly under-damped fashion. If however the experimenter moves the robot arm to a new position and holds it there for a couple of seconds, the equilibrium shifts to the new position. This happens because the valves are not completely closed.

Finally we illustrate the speed capabilities of the robot. Figure 2C shows the results of two experiments, in which the robot is oriented so that the second link moves in the horizontal plane. At time 0 the agonist (i.e. pushing in the movement direction) control changes from 0V to 10V. If the antagonist control is 0V, we observe a very rapid movement reaching peak velocity of 1450 deg/sec.

Note that towards the end of the movement the pressure in the antagonist chamber rises substantially. This is because the chamber volume is decreasing faster than the air can get out of it (even though the valve is fully open towards the room). The result is a non-linear viscosity effect present only at high

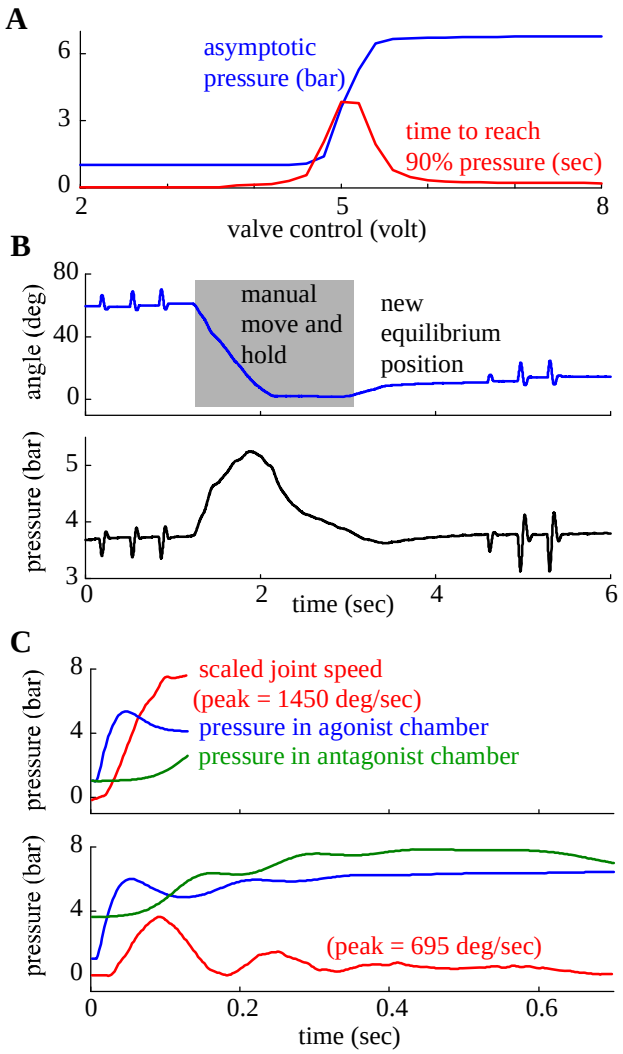

Fig. 2. **A**– Asymptotic pressure, recorded 5 seconds after setting the valve control to the value shown on the horizontal axis. The curve is the same regardless of the initial pressure. The time it takes to reach 90% of the asymptotic pressure is also shown (in red). **B**– Illustration of plasto-elasticity. The pulses correspond to the experimenter hitting the robot – which quickly returns to its previous equilibrium position. The equilibrium however can be changed by holding the robot for a few seconds in a different position. **C**– Response to a step control input to one of the valves. When the opposing valve is open we see a rapid movement. When the opposing valve is closed, we see a complex response ref1ecting the air dynamics.

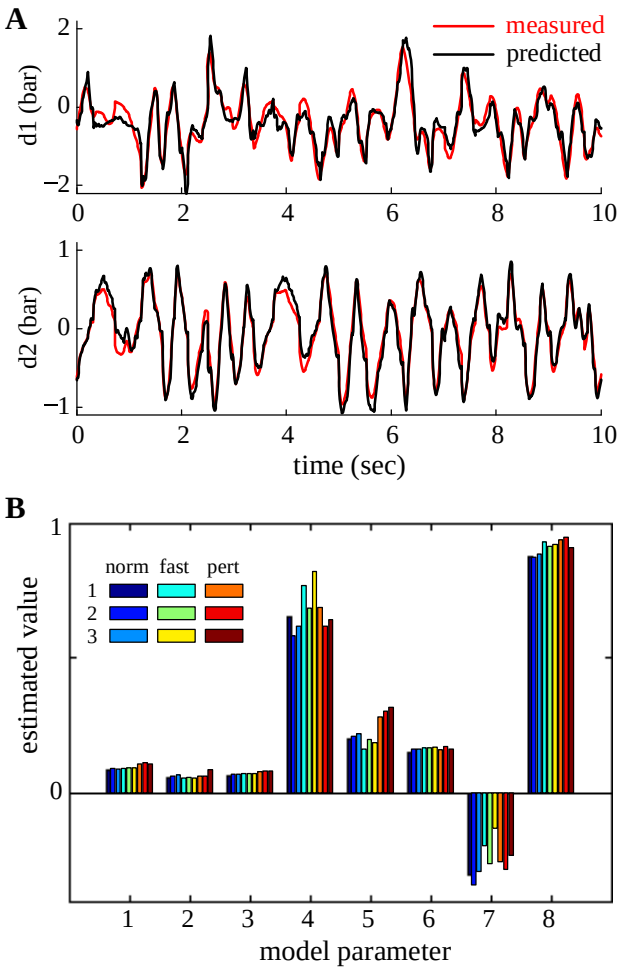

Fig. 3. **A**– typical example of measured and predicted pressure differences over a 10 sec time window taken from a 60 sec trajectory (normal speed, no self perturbtations). Overall the R^2 for the model fit is 0.87. **B**– The values of the 8 model parameters estimated from 9 different datasets. The first three parameters (encoding the inertia matrix) are scaled by 10 to be more visible.

velocities. If we now repeat the experiment (bottom row) with the antagonist control set to 5V, the result is very different: the arm begins to move rapidly, then comes to almost a complete stop, and then continues at a slow and somewhat oscillating velocity. This behavior is the result of constant control signals; all the complexity comes from the air dynamics.

IV. Modeling and System Identification

In this section we construct a physical model of the 2-dof robot and fit its parameters in a system identification phase. The model is later used for control purposes. The model has two parts. One is the dynamics of the 2-link mechanism, and relates joints positions, velocities and torques. The other is the dynamics of the pressures inside the chambers. We will use the following notation:

$i \in \{1,2\}$	joint number: 1-shoulder, 2-elbow
$q_i, \dot{q}_i, \ddot{q}_i$	joint angle, velocity, acceleration
p_i^+, p_i^-	pressures in the two chambers of cylinder i
τ_i	actuator torque acting on joint i
$d_i = p_i^+ - p_i^-$	pressure difference (proportional to τ_i)
u_i^+, u_i^-	control signals to the valves of cylinder i

Thus the state vector is $\mathbf{x} = \left[q_1, q_2, \dot{q}_1, \dot{q}_2, p_1^+, p_1^-, p_2^+, p_2^-\right]^\mathsf{T}$ and the control vector is $\mathbf{u} = \left[u_1^+, u_1^-, u_2^+, u_2^-\right]^\mathsf{T}$. The quantities d_i, τ_i are derived from the state variables.

A. Two-link arm dynamics

The equations of motion of a two-link arm are fairly standard, as derived below. The unusual aspect of our model is that we seek a minimal parameterization so as to make the

subsequent system identification more reliable. As a result, multiple physical quantities are lumped together into model parameters (w) which do not have intuitive meaning.

The configuration-dependent inertia matrix M of this system is in the form

$$M(\mathbf{q}) = \begin{bmatrix} w_1 + w_2 \cos(2q_2) & 0 \\ 0 & w_3 \end{bmatrix} \quad (1)$$

The off-diagonal terms are zero because the joint axes are orthogonal. The Christoffel symbols of M yield the Coriolis and centripetal torques $\mathbf{n}$:

$$\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = w_2 \begin{bmatrix} -2\sin(2q_2)\,\dot{q}_1\dot{q}_2 \\ \sin(2q_2)\,\dot{q}_1\dot{q}_1 \end{bmatrix} \quad (2)$$

The gravitational torques $\mathbf{g}$ can be computed using the fact that the elevation of the center of mass of link 2 is proportional to $\cos(q_1)\cos(q_2)$. Note that $q_1 = q_2 = 0$ is defined as the vertical configuration shown in Figure 1. Using the gradient of the elevation, we have

$$\mathbf{g}(\mathbf{q}) = w_4 \begin{bmatrix} \sin(q_1)\cos(q_2) \\ \cos(q_1)\sin(q_2) \end{bmatrix} \quad (3)$$

We also include viscous forces $\mathbf{b}$ of the form

$$\mathbf{b}(\dot{\mathbf{q}}) = \begin{bmatrix} w_5\dot{q}_1 \\ w_6\dot{q}_2 \end{bmatrix} \quad (4)$$

Finally, we allow for an offset between the pressure differences d and torques τ, which can account for sensor calibration issues. A scaling factor is not needed because the above model is already scalable. Thus the overall model is

$$\begin{aligned} \widehat{d}_1 &= (w_1 + w_2\cos(2q_2))\,\ddot{q}_1 - 2w_2\sin(2q_2)\,\dot{q}_1\dot{q}_2 \\ &\quad + w_4\sin(q_1)\cos(q_2) + w_5\dot{q}_1 + w_7 \\ \widehat{d}_2 &= w_3\ddot{q}_2 + w_2\sin(2q_2)\,\dot{q}_1\dot{q}_2 \\ &\quad + w_4\cos(q_1)\sin(q_2) + w_6\dot{q}_2 + w_8 \end{aligned} \quad (5)$$

This is an inverse dynamics model, predicting the pressure differences given the joint positions, velocities and accelerations. It has eight free parameters.

All parameters affect the model predictions linearly, which is a major advantage because we can find the globally-optimal parameter estimates using linear regression. As a sanity check, we constructed a model of our robot in the Matlab robotics toolbox (setting $d = \tau$ and excluding the pneumatics), generated data for randomly chosen positions, velocities and accelerations, and then fit our model to it. Since we are using lumped parameters it is not easy to see what the correct estimates should be, however the model fit produced zero error – which validates the above equations. We then proceeded to fit the model to actual data obtained from the robot. The data were collected while the robot was driven by a PID controller tracking a reference trajectory (see below). We collected nine dataset: three reference trajectories (each 60 seconds long), executed at normal speed, 2 times faster speed, and normal speed with self-generated control perturbations. The measured and predicted pressure differences are compared in Figure 3A over a 10 sec time window. Note the close correspondence. Overall the predicted values of d explained 87% of the variance of the measured values (i.e. $R^2 = 0.87$). We also tested if the estimated model parameters are similar over the nine datasets. This was indeed the case as shown in Figure 3B. Using the dataset-specific parameters instead of the average parameters improved the model fit only by 1% variance explained ($R^2 = 0.88$ averaged over the datasets).

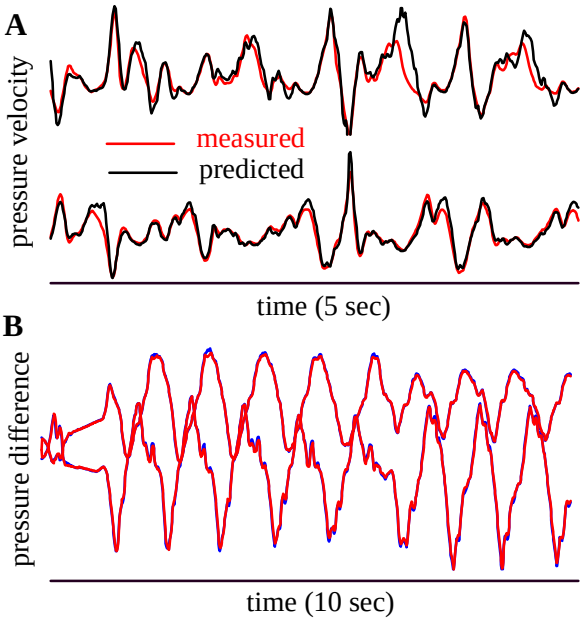

Fig. 4. **A.** Example of measured and predicted pressure velocities $\dot{p}$, for the two chambers of cylinder 1. **B.** Example of reference (blue) and model-predicted (red) pressure differences d for both actuators.

B. Air dynamics

A pneumatic cylinder is a device with two chambers separated by a piston. In a double acting cylinder, each chamber has an orifice controlled by a valve, which connects the chamber to the compressor or the room. When the orifice is open to the compressor, compressed air enters into the chamber. Similarly, when the orifice is open to the room, chamber air exhausts to the atmosphere. Increasing or decreasing pressure can force the piston to move in the desired direction.

1) Mass Flow Model: The key part of the cylinder model is to control air f1ow that causes chamber pressure difference. According to isentropic f1ow assumptions, the mass f1ow rate going into a chamber of cylinder i is defined as

$$\dot{m}_i = a_{i,c}(u_i)\,f(P_c, p_i) - a_{i,r}(u_i)\,f(p_i, P_r) \quad (6)$$

where $a_{i,c}, a_{i,r}$ are the orifice areas connecting the chamber to the compressor and room respectively, and P_c, P_r are the (constant) pressures in the compressor and room.

The nonlinear function $f(P_c, p_i)$ is given by

$$f(P_c, p_i) = \alpha P_c \sqrt{\left(\frac{p_i}{P_c}\right)^{\frac{2}{k}} - \left(\frac{p_i}{P_c}\right)^{\frac{k+1}{k}}} \quad (7)$$

for $P_c/p_i \leq \theta$, and otherwise

$$f(P_c, p_i) = \beta P_c \tag{8}$$

2) Chamber Pressure Dynamics: Air charging and discharging processes into the cylinder's chambers cause the pressure change. Using the law of conservation of energy, and assuming a constant temperature T and no heat exchange, the velocity of the chamber pressure can be derived as follows

$$\dot{p}_i = -k\frac{\dot{V}_i}{V_i}p_i + k\frac{RT}{V_i}\dot{m}_i \tag{9}$$

where V_i is volume of the chamber and $\dot{V}_i$ is the volume velocity. These equations depend on a number of constants which are inferred from the data.

3) Calibration of Orifice Open Area: In our two dimensional pneumatic system, there are two cylinders and four chambers. Each chamber has an orifice connecting to compressor or room, which is controlled by a valve. To calibrate the orifice area for each valve control signal, we keep the chamber volumes static, set initial chamber pressure to atmospheric pressure or compressor pressure, provide constant valve signal for 5 seconds, and then record the pressure. Since the chamber volume is static, the volume velocity is 0 and we have

$$\dot{p}_i = k\frac{RT}{V_i}\dot{m}_i = k\frac{RT}{V_i}\left(a_{i,c}\left(u_i\right) f(P_c, p_i) - a_{i,r}\left(u_i\right) f(p_i, P_r)\right) \tag{10}$$

As k, R, T are known constants, $\dot{p}(t)$, $f(P_c, p(t))$ and $f(p(t), P_r)$ can be calculated at each time step t, we are able to estimate the unknown constants a_c/V and a_r/V by linear regression for each voltage v, with notations $a_c(v)/V$ and $a_r(v)/V$. As $a_c(10)$ and $a_r(0)$ are known, we can calibrate other $a_c(v)$ and $a_r(v)$ by using

$$a_{i,c}(u_i) = a_{i,c}(10)\frac{a_{i,c}(u_i)/V}{a_{i,c}(10)/V} \tag{11}$$

$$a_{i,r}(u_i) = a_{i,r}(0)\frac{a_{i,r}(u_i)/V}{a_{i,r}(0)/V} \tag{12}$$

The resulting model was able to fit the experimental data quite well – see Figure 4A.

4) Model-predictive control: Given a reference pressure-difference trajectory $d\left(t\right), d\left(t+\Delta\right), \cdots d\left(t+n\Delta\right)$ over a short time horizon n with time step Δ, we can now use our model of air dynamics to compute valve commands which will instantiate the reference pressure differences. This approach is superior to using instantaneous pressure feedback (see below) because the valve commands have their largest effect a couple of time steps into the future. Note that there is a redundancy issue here: the reference trajectory only specifies the pressure differences/forces (because it is computed by some higher-level controller which ignores the air dynamics) and not the actual pressures. We resolve this redundancy by introducing a control cost on the valve commands – namely a quadratic centered at 5V. We then formulate a non-linear optimal control problem subject to control constraints, and solve it using our iterative LQG method [15]. This is done in real-time at every step of the control loop. Figure 4B shows the reference trajectory (blue) plotted at $d\left(t+\Delta\right)$, and the trajectory (red) predicted by our air dynamics model given the controls computed by iLQG. The two are indistinguishable, meaning that at least according to the model we can achieve the desired pressure differences perfectly, by taking into account a few time steps into the future (currently $n = 2$). We are currently experimenting with applying this control scheme to the physical robot.

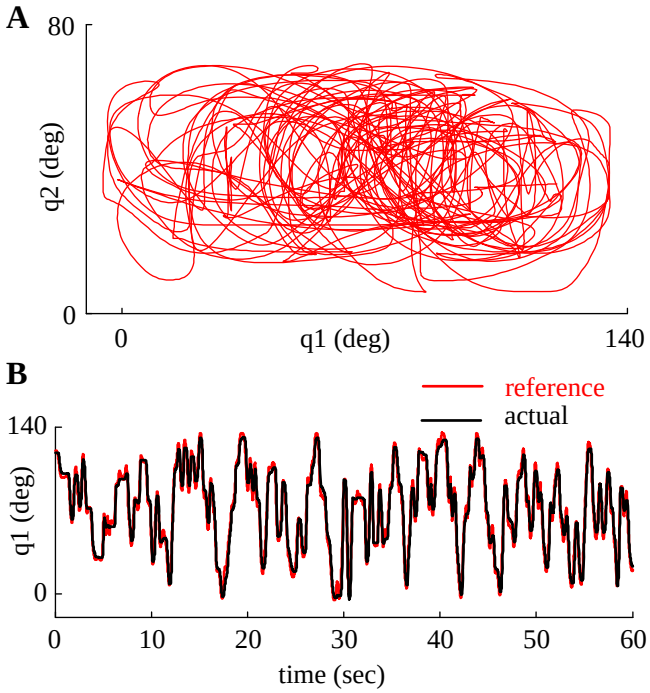

Fig. 5. **A**– One of the three reference trajectories recorded by the experimenter. It contains reaching movements that stop abruptly, circles, figure eights, and a variety of other movements. **B**– Tracking performance of the best controller we have designed so far for this robot (PIDF+model).

V. 2-DOF TRAJECTORY CONTROL

Here we focus on the task of tracking a reference trajectory in joint space. While many interesting control problems are not in this form, trajectory tracking is a basic control objective that one should be able to accomplish before moving on to more challenging problems. Three long reference trajectories (1 minute each) were recorded by the experimenter, who held the endpoint of the robot and moved it around while the computer recorded the joint angles. One of these trajectories is shown in Figure 5A. Each trajectory was used at the recorded speed (which was already quite fast), as well as a two times faster speed (referred to as "fast trajectories"). The experimenter attempted to generate as diverse a set of movements as possible. The data were scaled to fit the joint range minus a small safety margin. Cubic spline smoothing was used to clean up the position data (which was not really necessary) but also to compute reference velocities and accelerations.

Multiple controllers were designed as described next. The tracking performance of the best controller is illustrated in Figure 5B.

As we describe the different controllers below, we will repeatedly refer to Figure 6 which compares their performance. Figure 6A illustrates the typical tracking performance of the different controllers on a portion of one of the fast reference trajectories (performance at normal speed is much better which makes it hard to distinguish between the different controllers). Figure 6B shows a systematic performance analysis. For each speed (normal and fast) and each controller, we compared the reference and actual trajectories at different latencies, and computed the mean absolute error. The reason for analyzing different latencies is because we noticed that our controllers often produce trajectories very similar to the reference, but with a delay. Fig 6C illustrates an interesting phenomenon. Here we repeated a (shorter) reference trajectory three times. The first repetition was discarded and the next two repetitions were plotted in the figure. On the last repetition the experimenter delivered some perturbations to the robot (marked with arrows). The trajectories are virtually identical before the perturbation, and quickly converge after the perturbation. This illustrates the stability of the controller, as well as the fact that the system is rather deterministic.

A. PID control

The simplest possible feedback control methodology is proportional-integral-derivative (PID) control. For each joint, we compute an error signal of the form

$$\begin{aligned} e_i(t) &= k_P\left(q_i^*(t) - q_i(t)\right) + k_D\left(\dot{q}_i^*(t) - \dot{q}_i(t)\right) \\ &\quad + k_I \int_0^t \left(q_i^*(s) - q_i(s)\right) ds \end{aligned} \tag{13}$$

where '*' denotes the reference trajectory. In the case of electric motors $e_i(t)$ can simply be interpreted as a control signal. Here the situation is more complicated because the torques/pressure differences do not correspond directly to the control signals. Nevertheless we can pretend that they do, and define the controls as

$$\begin{aligned} u_i^+(t) &= 5 + e_i(t) \\ u_i^-(t) &= 5 - e_i(t) \end{aligned} \tag{14}$$

This will generally have the effect of increasing the pressure difference when $e_i > 0$, and decreasing it otherwise.

The resulting control scheme had plausible performance at normal speeds, however it was far from the reference trajectories at high speeds. See Figure 6A,B.

B. PID control with force feedback

Here we consider a simple extension to the above PID control scheme. Instead of treating the PID error signal e_i as a control, we treat it as what it really is, namely a desired pressure difference. Since we are measuring the actual pressure difference d_i in real time, we can create a low-level feedback loop aiming to match the desired and actual pressures. The resulting control law is

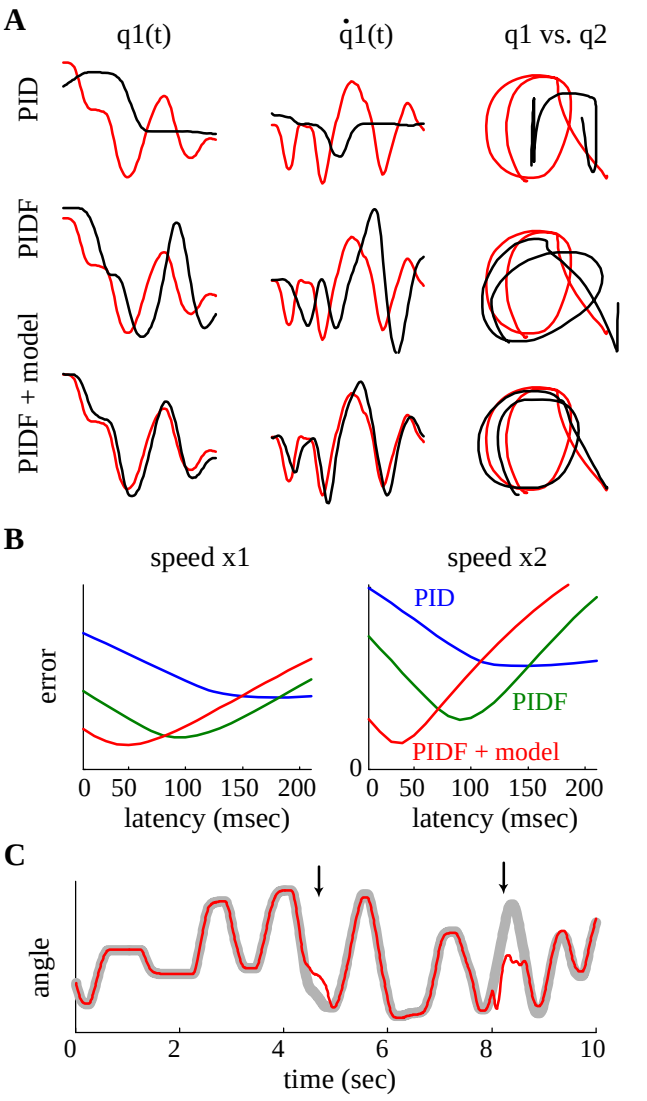

Fig. 6. **A**– Typical behavior of the three controllers on fast trajectories. In the illustrated time segment, the excursion of the reference velocity is around 1000 deg/sec. **B**– Mean absolute tracking error (averaged over the two joints) for each controller, trajectory speed, and time lag. The results are averaged over the three reference trajectories. **C**– Illustration of the repeatability of the PIDF controller. The two traces are two executions of the same reference trajectory. The arrows denote times when the experimenter perturbed the robot.

$$\begin{aligned} u_i^+(t) &= 5 + e_i(t) - d_i(t) \\ u_i^-(t) &= 5 - e_i(t) + d_i(t) \end{aligned} \tag{15}$$

We will call this PIDF control, where the "F" stands for "force". The reason for using the term "force" rather than "pressure" is because we believe this scheme is more general, and can be used for other third-order systems even if the actuation mechanism is not pneumatic. One particularly interesting case is biological motor control, where Golgi tendon organs provide fast force feedback to the spinal cord, and muscles have activation states similar to pneumatic actuators.

While PID and PIDF control are conceptually very similar, the latter turns out to be a much better control scheme. The quantitative comparisons can be found in Figure 6A,B. Another difference is that PIDF control can use larger gains – which is to be expected if it achieves smaller errors, although the argument is somewhat circular. We established this empirically by optimizing the gains of both controllers. The optimization was done automatically as follows. We implemented a Matlab function whose arguments specify the feedback gains. This function applies the corresponding control law to the physical robot (on a 10 second trajectory), measures the tracking error, and returns it to the calling function. We then used a numerical optimization procedure (the nonlinear simplex method) to minimize the resulting error. The reason for using the simplex method rather than a finite-difference-based gradient descent method is because the former evaluates the function at points that are spaced further apart. Given that the function evaluation is somewhat noisy, finite differences will produce unreliable results. Overall we found that the automatic gain tuning always finds somewhat better gains than our manual tuning, although there seemed to be plenty of local minima. In order to make this automated procedure safe, the evaluation function returned a large error whenever the robot hit any of the joint limits at high speed (which it did when the minimizer attempted to use large gains).

C. Model-based PIDF control

We designed a controller taking advantage of our model of the robot. The current version uses only the 2-link dynamics model and not the air dynamics model (the latter was used above in the model-predictive control setting but only in simulation for now). As in PIDF control, we compute a desired pressure difference at each point in time. However we now take the model predictions into account:

$$\overline{d}_i(t) = e_i(t) + \widehat{d}_i(t) \tag{16}$$

The feedback gains are the same as in the PIDF controller. Note that if tracking is perfect we will have $e_i = 0$, in which case the PID and PIDF controllers will generate zero output and thereby cause an error. The model-based controller can at least in principle track perfectly, modulo model errors of course. The control signals now have the form

$$\begin{aligned} u_i^+(t) &= 5 + \overline{d}_i(t) - d_i(t) \\ u_i^-(t) &= 5 - \overline{d}_i(t) + d_i(t) \end{aligned} \tag{17}$$

This model-based controller had markedly better performance, especially on fast trajectories. It was actually better in two (possibly related) ways. First, the tracking error at zero latency was smaller. Second, the latency at which it achieved the best performance was smaller – around 40 msec. See Figure 6A,B.

VI. Controlling a 38-dof humanoid

Here we describe results from controlling the humanoid robot. Since we are still in the process of constructing a dynamics model and fitting it to data (using a new physics engine we are developing), the present results are based on the PIDF control scheme described above. Before attempting to control the humanoid, we collected calibration data for every valve and cylinder, and designed (automatically) a non-linear transformation which compensates for the unique characteristics of every valve, resulting in a canonical sigmoid relationship between command signals and (inferred) orifice areas. These non-linear transformation were incorporated into our device driver and were transparent to the rest of the control system.

A. Trajectory tracking

We obtained a 1-minute reference trajectory by holding the arms, legs and head of the robot and moving them around while recording potentiometer data (it took 4 people to do this). We then adjusted the control gains to achieve the best tracking possible. This was done in a hierarchical manner. First, we sent a sinusoidal signal to each joint (one at a time), measured the resulting range of motion, and defined a joint-specific gain scaling factor equal to the inverse of this range. The rationale is that some joints have higher "impedance" that others (lumping multiple effects into a single number) and thus need larger control gains. We then tuned the master gains by hand.

Tracking was surprisingly good – see Figure 7. The median absolute error over all 38 dofs was 3.4% of the corresponding joint ranges. Most of this error was due to a timelag – which was aroound 100 msec. Note that the actuators of the humanoid have longer time constants compared to the 2-dof system, because of the longer air tubes connecting the (off-board) valves to the cylinders.

Similar to the 2-dof case, we found that the trajectories resulting from the PIDF controller were very repeatable as well as stable in the presence of perturbations. Comparing two repetitions of the experiment, we found that the difference is 0.03% of the joint range, or 100 times smaller than the differences shown in Figure 7. This is encouraging because it implies that a better model-based controller could in principle achieve almost perfect performance.

B. End-effector control

Finally, we developed a hierarchical control scheme capable of moving the robot's hand to a spatial target held by the experimenter. Let $x = f(q)$ denote the mapping from arm joint coordinates $q \in \mathbb{R}^7$ to hand Cartesian coordinates $x \in \mathbb{R}^3$, $J(q)$ the Jacobian of this mapping, and $x^* \in \mathbb{R}^3$ the current target position. Here x and x^* are obtained from optical motion capture, while q is obtained from the potentiometers. We compute the instantaneous spatial error $x^*(t) - x(t)$, interpret it as a desired force acting on the hand, and map it to desired joint torques as $\tau(t) = J(q(t))^{\mathsf{T}}(x^*(t) - x(t))$. Then $\tau(t)$ is treated in the same way as the quantity $e(t)$ generated by the PIDF controller, and converted into valve command signals. This control scheme alone is problematic because the arm can gradually reach some of the joint limits. To avoid this problem, we further define a default posture

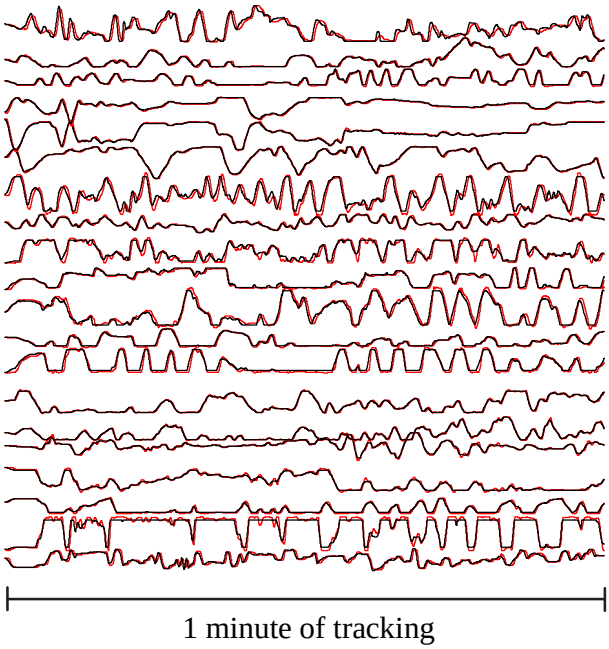

Fig. 7. Reference (black) and measured (red) trajectories for 20 out of the 38 dofs of the humanoid.

$\overline{q}$ in the middle of the joint range, and generate additional torques $\mathrm{Null}\left(J\left(q\left(t\right)\right)\right)\left(\overline{q}\left(t\right)-q\left(t\right)\right)$ in the null space of the Jacobian, pushing the arm towards the default posture in a way which does not interfere with target tracking. This resulted in a very responsive controller which tracked the target well, and resolved the redundancy of the arm automatically. We used a somewhat inaccurate kinematic model (since we do not yet have the CAD files for the robot), and furthermore two of the joints involved had more friction than what a linear feedback controller could handle well. So there is room for improvement here, but the initial results are encouraging. Movies of both robots in action can be found at `www.cs.washington.edu/homes/todorov`, under `Papers`.

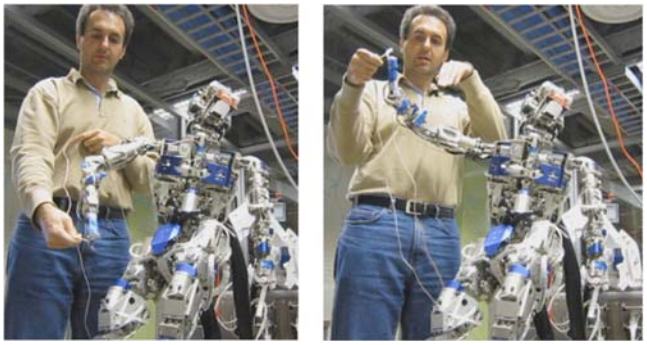

Fig. 8. End-effector control to spatial targets specified in real-time by the experimenter. The robot "sees" its hand and the target using off-board infrared motion capture (PhaseSpace system).

VII. Conclusion

We presented results on modeling, identification and feedback control of a 2-dof pneumatic robot and a 38-dof humanoid. We were able to obtain good tracking performance and end-effector control using relatively simple techniques. Our model of air dynamics has not yet been used in the control loop, but we plan to do that soon. We are also hopeful that the model-predictive approach based on iLQG can be extended to handle the entire control problem and not just reference pressure tracking. Modeling and system identification of the humanoid is another interesting problem which remains to be addressed. This will be done in future work.

Acknowledgment

This work was supported by NSF grant IIS0808767 and NIH-NINDS grant R01NS058633. Thanks to Joanne Jao, Tomoyuki Noda and Tingfan Wu for their technical help with the humanoid robot.

References

[1] S. Liu and J. E. Bobrow, "An analysis of a pneumatic servo system and its application to a computer-controlled robot," *Journal of Dynamic Systems, Measurement, and Control*, vol. 110, no. 3, pp. 228–235, 1988.

[2] K. Khayati, P. Bigras, and L.-A. Dessaint, "Lugre model-based friction compensation and positioning control for a pneumatic actuator using multi-objective output-feedback control via lmi optimization," *Mechatronics*, vol. 19, no. 4, pp. 535 – 547, 2009,

[3] C. M. J.Y. Lai and R. Singh, "Accurate position control of a pneumatic actuator," *Journal of Dynamic Systems, Measurement, and Control*, vol. 112, pp. 734–739, 1990.

[4] B. McDonell and J. Bobrow, "Adaptive tracking control of an air powered robot actuator," *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, p. 427, 1993.

[5] D. Caldwell, G. Medrano-Cerda, and M. Goodwin, "Control of pneumatic muscle actuators," *Control Systems Magazine, IEEE*, vol. 15, no. 1, pp. 40–48, Feb 1995.

[6] G. A. Medrano-Cerda and C. J. Bowler, "Adaptive position control of antagonistic pneumatic muscle actuators," in *IROS '95* 1995, p. 378.

[7] G. Tonietti and A. Bicchi, "Adaptive simultaneous position and stiffness control for a soft robot arm," in *Intelligent Robots and Systems*, 2002

[8] D. Caldwell, G. Medrano-Cerda, and M. Goodwin, "Braided pneumatic actuator control of a multi-jointed manipulator," in *Systems, Man and Cybernetics*, pp. 423–428 vol.1, 1993

[9] J. Schröder, K. Kawamura, T. Gockel1, and R. Dillmann, "Improved control of a humanoid arm driven by pneumatic actuators," in *Proceedings of Humanoids 2003*, 2003.

[10] T. Acarman, C. Hatipoglu, and U. Ozguner, "A robust nonlinear controller design for a pneumatic actuator," in *American Control Conference*, vol. 6, pp. 4490–4495, 2001

[11] M. Van Damme, B. Vanderborght, B. Verrelst, R. Van Ham, F. Daerden, and D. Lefeber, "Proxy-based sliding mode control of a planar pneumatic manipulator," *Int. J. Rob. Res.*, vol. 28, no. 2, pp. 266–284, 2009.

[12] P. Carbonell, Z. Jiang, and D. Repperger, "Nonlinear control of a pneumatic muscle actuator: backstepping vs. sliding-mode," in *Control Applications*, pp. 167–172, 2001

[13] Y.-C. Tsai and A.-C. Huang, "Multiple-surface sliding controller design for pneumatic servo systems," *Mechatronics*, vol. 18, no. 9, pp. 506 – 512, 2008.

[14] J. Movellan, "A Pneumatic Cylinder Model," Machine Perception Laboratory Tutorials, http://mplab.ucsd.edu, 2009.

[15] E. Todorov and W. Li. "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems", American Control Conference, pp 300–306, 2005.