Unedited posts from archives of CSG-L (see INTROCSG.NET):

Date: Wed Jan 20, 1993 9:50 am PST
Subject: Degrees of Freedom

[From Chris Malcolm]

Rick Marken writes: >Chris Malcolm -->Powers said:

>> Have you actually seen Little Man Version 2? It controls an arm with 3 degrees of freedom without computing inverse anythings.

> You reply:

>> 3 degrees is trivially easy to do almost any way you like. It's the last two degrees of freedom (5 & 6) that create the computational difficulties.

> So can we expect the PCT revolution to begin when the Little Man also controls the roll (4) pitch (5) and yaw (6) of a baton held in his hand? I would guess that after all the work to program these changes (the main work being simulation of environment; the torques of the baton and whatnot) the AI, robotics, and motor control community would look at it and say "yeah, but your doing it with a little man; most baton twirlers are girls". This critique is as cogent (and misses the point as grossly) as does the observation that only 3 df are controlled in the current demo.

I agree. But the kind of engineering types who build industrial robot controllers are not going to even bother examining the detail of any demonstration of a supposedly better way of controlling robots which only handles 3 DoF, since they know that 3 DoF can be done very computationally cheaply by a great variety of methods which fail horribly with 6 DoF, and students keep coming up to them with very wonderful methods which work a treat in 3 DoF and bog down horribly in 6. "Very interesting, come back and show me it handling 6 DoF, and then I'll be interested enough to bother trying to understand it" is the standard response.

Chris Malcolm

Date: Wed Jan 20, 1993 11:49 am PST
Subject: Robotics and degrees of freedom

[From Bill Powers (930120.1130)] Chris Malcolm (930120) --

Chris, could you explain just what the problem with 6 (or 7) degrees of freedom is as seen in conventional robotics? Without actually extending the Little Man to that degree of complexity I can't come up with a demonstration just now, but in my naivete it doesn't strike me as particularly difficult. I'm sure there are some problems, but without actually trying it I can't guess what they are.

Suppose we were trying just to position a fingertip on a hand which swivels in two dimensions at the wrist, the wrist mounted on a rotatable forearm, the forearm bending at the elbow, and the upper arm swinging in x and y and also rotating. That seems to come out to 7 degrees of freedom.

Positioning the fingertip in space (as visually perceived) uses up only three degrees of freedom, so there are four degrees left unspecified. The (visual) feedback signals representing x,y, and z error would be compared with reference signals, and the error signals would be distributed among the reference signals operating the 7 degrees of freedom of the arm. The requirement for the distribution is that a small perturbation in x should contribute (through each possible path) to an opposing net effect on the finger in the x direction, and the same for y and z. The actual effects don't have to be exactly opposed to the perturbation, as long as they have a component that is opposed to it. In this way the three errors would be corrected. The remaining degrees of freedom being unspecified, the arm would end up in some arbitrary configuration, but the fingertip would be in the specified position.

If the orientation of the hand were also to be controlled, there would be three more perceptual variables representing the hand in terms of rotation about the x, y, and z axes in space. Again, we could derive three error signals, which contribute to all seven degrees of freedom in the way required for net negative feedback about the three axes of rotation. This will use up three more degrees of freedom, leaving one degree unspecified.

Mechanical limits on position and rotation would introduce interactions so that when one control system reached a limit, the reference signals for others would automatically adjust through normal feedback action.

I realize that this picture involves a great deal of abracadabra, and I know that my intuition is far from infallible. I'm just trying to sketch in how I'd approach the problem in PCT terms without anticipating any problems.

The other approach, I can see, would be very complex. If you picked a final configuration of the arm, to know how to get the arm to that configuration from some other configuration by varying joint angles you'd have to solve for the inverse of the kinematic equations, and then devise a signal-generator that would produce just the signal waveforms that would accomplish the move. You'd have to avoid signal combinations that would exceed limits or pass through forbidden regions, and so forth.

With the PCT approach you don't have to compute any inverse relationships. Only forward computations are involved. From the current joint angles, you compute the current states of the variables to be controlled. You compare those states with the corresponding desired states, and convert the error into changes in joint angles that will make the error smaller. In simpler applications, what happens is that the systems acting together simply find a path from any initial configuration to the desired final configuration. Actually I've done this with as many as 50 controlled variables, admittedly without any complex environmental constraints to make the job harder.

If this straightforward approach fails, there is still a way out. By constructing intermediate control systems handling only a subset of the degrees of freedom, control can be established in an intermediate space where the original degrees of freedom are transformed into a more tractable set. In a minor way this is what I did with the Arm demo, by combining vertical control of the two arm segments into a single radial direction of control and a controlled vertical rotation in phi. Having done that, relating the visual error signals to changes in reference signals for the kinesthetic systems became relatively simple, with no ambiguities.

The impression of current robotics approaches that I have gathered is that the problem is thought of as producing a particular outcome by finding a set of inputs that will produce it. This necessarily involves working backward from the desired outcome to the inputs through the inverse of the relationships that turn inputs into outputs.

My approach, the PCT-style approach, is much more permissive. There isn't any attempt to specify the path by which the system gets from one state to another. Instead, we begin with the current state of the variables and their differences from the desired states. The error is then transformed into, or mapped into, a vector that, when added to the current state vector, makes the error vector smaller during one time-increment. If the error always gets smaller, the inevitable result is that the actual state vector will become equal to the desired one.

The path by which this result is accomplished may not be an ideal one, but in all cases I have looked at (simpler cases) such a result does come about, and not by any outrageous or prima facie undesirable paths. This makes sense if you assume that the human systems work the same way. We judge the reasonableness of the paths, after all, by comparing them with the way the real system works. If the real system doesn't precompute paths, but takes whatever results from the simultaneous actions of the control systems, then an artificial system that works the same way will move in ways that look perfectly "natural."

Best, Bill P.

Date: Wed Jan 20, 1993 12:17 pm PST
Subject: Re: Robotics and degrees of freedom

[From John Gardner (930120.1400) to: B. Powers and Chris Malcom

Perhaps I can help clear up the mis-understanding. I consider myself to be one of those "engineering types who make robot controllers" and I think I see the misunderstanding here. I agree with Chris Malcom-what works in 3 DOF may not help much at all in 6 dof.

I think the root of the problem lies in an unstated assumption in your PCT approach. It seems to me (though I'm not altogether sure) that in your 3 dof 'little man' that you somehow know which joint to move to affect which 'desired state' (by which I assume you mean x, y and z). IN your sub-space approach, I think you partition the problem in such a way so as to make the interaction between the joint-space and the cartesian space easier. With 3 dof, this kind of intuitive approach is possible. In general, and for a 6 DOF industrial robot, EVERY joint has some effect on EVERY coordinate of the end effector. Likewise, EVERY coordinate of the endeffector is affected by EVERY joint in some way or another. Note that this is true, in general. Clearly there are special geometries and special configurations for which the problem de-couples.

In robotics, we deal with this as a coordinate transformation relating small movements of the joints with corresponding small movements in the end-effector space. This relationship is represented in a matrix called the Jacobian matrix. This also represents the relationship between velocities in the two coordinate frames (divide both small motions by a small delta- time). In general, this relationship is defined by the geometry and allows us to compute a set of x-y-z (and rotational) speeds which result from a given set of joint speeds. To move in the other direction, we need to take the inverse of this matrix (which is 6x6 and very 'goopy').

In terms of control: Imagine I'm at point 1 in x-y-z space and I want to get to point 2. I can draw a 'desired velocity vector' which will tell me the way to get to point 2 from where I am. BUT the only way to ensure that I will move the robot in such a way that I get there is to transform that 'desired velocity vector' which is, by its very nature, defined in the x-y-z frame, into the coordinate frame of the joints. This is done by assembling the 6x6 Jacobian matrix for that 'pose' of the robot, inverting it and multiplying it by the 'desired velocity vector'. The result is a set of joint rates which will move the robot along the desired path. This process has to be update quite often (10-20 ms intervals) and I think now the problem of higher degrees of freedom becomes clearer.

One other point I'd like to address. You made mention of 'unspecified degrees of freedom'. This area is well known to the robotics field as redundant degrees of freedom, most often considered in the control of 7 DOF robots. The problem is that, in control, you can never say that the 'arm will assume some arbitrary configuration' When you control a piece of hardware, you must control it completely. The problem is, which of the infinite number of arbitrary configurations is best, according to some arbitrary criteria (min torque, obstacle avoidance, etc.)

Well, I've rambled on enough for now. What do you think? Does this seem to address the confusion?

-John Gardner
Mechanical Engineering Penn Sate Univ University Park, PA 16802

Date: Wed Jan 20, 1993 12:55 pm PST
Subject: Re: Robotics and degrees of freedom

[From Rick Marken (930120.1200)] John Gardner (930120.1400) --

One of your reasons for believing that the addition of df will be a problem is that they will have unpredictable (or, at least complex) effects on the other df. But this is precisely why the perceptual control model is the right organization for control of multiple df. The effects of the controlled df (or

the outputs that effect them) on other controlled df are treated as disturbances and compensated for automatically by the input control systems. That's why you don't need all those inverse kinematics and dynamics computations; all the system cares about is the controlled degrees of freedom (as perceived) -- the system must also be set up so that it can have (relatively) independent influences on these df; but after that (and some proper stabilization -- EASY) you're ready to boogy (er...robot).

Best Rick

Date: Wed Jan 20, 1993 3:57 pm PST
Subject: Re: robotics and degrees of freedom

[From Bill Powers (930120.1530)] John Gardner (930120.1400) --

It's wonderful to have you real control engineers speaking up. I've glad you broke your silence. I think that eventually you guys are going to be able to tell me why the PCT model works, even if you start out by telling me why it can't work.

There's one thought to hold firmly in mind: it is highly unlikely that the lower reaches of the brain and the cerebellum are computing the quantitative inverse of a 6x6 (or when you come down to it, a 26x26) matrix in order to move the limbs, walk, balance, and so on, all at the same time and in real time. SO THERE HAS TO BE A MUCH SIMPLER WAY.

> I think the root of the problem lies in an unstated assumption in your PCT approach. It seems to me (though I'm not altogether sure) that in your 3 dof 'little man' that you somehow know which joint to move to affect which 'desired state' (by which I assume you mean x, y and z).

Yes, that's true. To move the fingertip upward in visual space, either the vertical shoulder angle or the elbow angle should be changed, or both. Put that way, the problem sounds ambiguous, because whether crooking the elbow would raise or lower the visual y position of the fingertip depends on how the arm is configured relative to the line of sight from the eye.

Furthermore, to change the z coordinate (distance from the eye), the elbow angle must be altered -- but that will change the vertical coordinate, either raising or lowering the fingertip depending, roughly, on whether the interior elbow angle is less than or greater than 90 degrees.

I think this way of looking at the problem is at the root of the conventional approach. There seems to be no solution but to solve the inverse kinematic equations to find the angles that will fit the requirement of both a certain y position and a certain z position. This is only a two-dimensional case, so there's no big problem in doing this. Nevertheless, there's an easier way using control systems and without explicitly solving the simultaneous equations.

The solution is to build one control system that controls the vertical shoulder angle and a second one that controls the EXTERIOR elbow angle. The y dimension of the fingertip position, as seen, is altered by varying the shoulder reference signal. The z dimension is controlled by sending a reference signal r to the elbow angle control system, and a reference signal -r/2 to the shoulder vertical angle control system (adding to the other reference signal).

When this compound reference signal is made larger, the exterior angle at the elbow increases proportionally, and the shoulder angle is _depressed_ by half that amount. The result is that the fingertip moves nearly along a line between the fingertip and the shoulder, radially. This has a slight disturbing effect on the visual y coordinate, but it is no longer ambiguous and the magnitude of the interaction is small enough that the visual control systems can easily compensate for it. So we have converted from the ambiguous problem to an unambiguous one, and we have two axes of control that are sufficiently independent for normal control action to soak up any remaining interactions. The visual systems, of course, control the visual y and z coordinates of the fingertip by altering the two reference signals we have created.

This is only a two-dimensional illustration. When we bring in the shoulder x control system we have again an interaction among the control system actions, but in visual space the disturbances caused by one axis of motion on the other two axes are modest and the visual control systems (or a still higher level of kinesthetic control systems) easily compensate. This, by the way, is also how the dynamical interactions are prevented from any important effects.

I see no reason in principle why this approach can't be extended to four or more degrees of freedom. If we can agree that the brain is NOT computing quantitative inverses of 6x6 matrices to better than one percent accuracy, this seems to be the only remaining approach. Each degree of freedom is put under direct feedback control. Then the control systems are put under the control of a higher level of control that senses combinations of the variables, removing ambiguities not by literally solving simultaneous equations, but in the analogue mode, by assuming a solution and using it to feed back to alter the variables in the right direction to bring about that solution. This process bears some resemblance to solving large sets of equations by successive substitution, but it does so without the use of any inverse calculations. It's more like a method of steep descent.

> In general, and for a 6 DOF industrial robot, EVERY joint has some effect on EVERY coordinate of the end effector. Likewise, EVERY coordinate of the end effector is affected by EVERY joint in some way or another.

Yes. You should take a look at Rick Marken's spreadsheet model of hierarchical control systems. This model has three levels with six systems at each level; each system perceives a variable made up of ALL SIX of the lower-level perceptual variables (or external controlled variables), and acts by contributing to ALL SIX reference signals at the lower level (or all six external physical variables). Each system controls its own composite perceptual signal quite independently of what the other systems are doing (unless, by bad luck, the six perceptual functions compute linearly dependent functions of the external variables).

Marken's demo works in a simple linear space without any complex external constraints, so it's really only a start toward solving the arm problem. But it shows that there is no problem caused merely by having six control systems all of which sense all of the variables, in different ways, and act by affecting the states of all the variables. As I said before, I have done this in 50 dimensions, with no great problem, using randomly-chosen weights for each of the 50 inputs to each of the 50 perceptual functions in the 50 control systems.

Nonlinearities and limits are going to complicate this problem, to be sure, but I think there's a basic principle here that isn't being used in conventional robotics. It's important because it doesn't require any complex calculations, and the system can work with normal analogue tolerances.

> In terms of control: Imagine I'm at point 1 in x-y-z space and I want to get to point 2. I can draw a 'desired velocity vector' which will tell me the way to get to point 2 from where I am. BUT the only way to ensure that I will move the robot in such a way that I get there is to transform that 'desired velocity vector' which is, by its very nature, defined in the x-y-z frame, into the coordinate frame of the joints. This is done by assembling the 6x6 Jacobian matrix for that 'pose' of the robot, inverting it and multiplying it by the 'desired velocity vector'.

This is essentially how I thought it was done. Tell me something: in order to get, say, one percent accuracy in the final position in x,y, and z, what kind of computational accuracy must be maintained during the matrix calculations, and how accurately do parameters like muscle response to signals and arm mass distribution and geometry have to be known? If gravity were being switched randomly on and off during this movement at unpredictable times, what would happen to the outcome? Do you see any way at all that the human nervous system, working with frequency-coded analogue signals having a dynamic range of something like 50:1, could do the necessary matrix inversions with enough accuracy to explain the behavior we see? I think it's very important to question whether the nervous system has the properties necessary to solve this problem as you describe. It's important for the reason I stated at the

beginning: if the nervous system lacks the capacities needed to carry out this approach, yet produces the kind of exquisite control that we observe, it must be operating on a completely different, and much simpler, principle. I think that the PCT approach is at least a hint as to what that principle might be.

Best, Bill P.