Constructing fast hydraulic robot models for optimal motion planning

Murali Krishna, John Bares

The Robotics Institute

Carnegie Mellon University

Pittsburgh, PA - 15213 (mkrishna, bares@ri.cmu.edu)

Abstract

Computing optimal motions for any robot requires a good model, and a method to compute the optimal motions using that model. As research is conducted into automating operations in construction, excavation etc. there arises a need to compute optimal motions for the hydraulic machines used in these areas. Hydraulic machines disallow a simple extension of work done previously on optimal motion planning for electric drive robots.

We have constructed a fast model for a hydraulic excavator(HEX) that can capture the non-linear actuator interactions. This model can simulate 75 secs of machine motion in 1 sec. of real time on a Sun Sparc 20. We use a set of neural networks to approximate the actuator response functions. We use the HEX model with a simulated annealing optimization method to compute time-optimal motions for the HEX, for defined start and end states. We demonstrate the efficacy of the constructed model and show results from using it in optimal motion computation. Real testbed results are shown in both cases. This is the first time that such a result has been reported in the literature for hydraulic machines.

I. Introduction/Background

Researchers continue to study the problem of optimal motion planning for robots. Most work to date has focused on planning optimal motions for electric drive robots, which are the staple in the manufacturing industry. Focus has lately been directed towards building robotic versions of hydraulic machines used in the fields of construction, forestry, and mining. Hydraulic machines are commonly used in these applications due to their high force-to-weight ratios. Automation of such machines is practical only if the robotic machine offers increased productivity or cost benefit. This can be achieved by performing the tasks optimally to minimize a combination of performance objectives such as time per bucket load and fuel consumption.

Optimal motion computation in turn requires a robot model which defines the constraint surface for the path optimization problem. A complete robot model consists of an actuator model and a linkage dynamics model. While the linkage dynamics can usually be modeled using the well-known Newton-Euler equations, the actuator model is rather complex and non-linear. Besides the non-linearities due to the physical process of fluid flow (as opposed to current flow) there are also non-linearities due to dynamic energy re-distribution between the different joints when a multi-joint hydraulic machine reaches a power or fluid flow limit.

A typical hydraulic machine used in construction and excavation is shown in Fig 1. This machine has four joints and two independently controlled tracks for movement of the base. The HEX is powered by an onboard diesel engine. During normal operation it is quite common for the HEX to be power limited. In such a condition the available power is dynamically redistributed between the joints. The heavily loaded joints get the least amount of power while the lightly loaded joints are the less affected. These non-linearities must be modeled since they significantly affect machine operation.

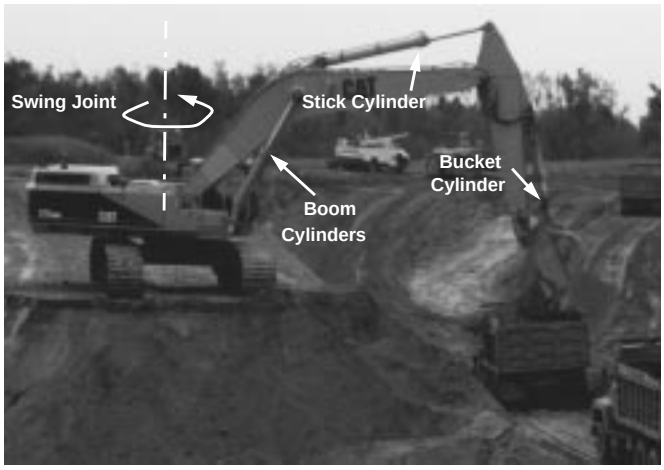

Fig. 1. A Hydraulic Excavator (HEX)

Therefore the two main challenges in optimal motion planning for hydraulic robots are -

- Constructing fast (computationally inexpensive) models of the robots that capture the important hydraulic system characteristics, and
- Using the models to compute optimal motions.

In this paper we address the first of the two problems, and present some preliminary results from an approach for the second. We use a 25-ton hydraulic excavator, similar to the HEX in Fig 1, as the testbed for our work.

We have previously [Krishna 98] described an approach to constructing computationally inexpensive hydraulic robot models using locally weighted linear regression. We improve upon that approach by using neural networks instead of the regression. Comparison of simulation results with results from the excavator testbed demonstrate the efficacy of our approach in modeling the important hydraulic system characteristics. The excavator model is able to accurately simulate a few seconds of excavator motion at a run-time ratio of 75:1, i.e. simulating 75 secs of excavator motion in 1 sec. on a Sun Sparc 20 workstation. This model is 3 times faster than the previously reported locally linear regression based model, and 30 times faster than an analytical model of comparable accuracy that we have constructed. However the new model is not much more accurate than the regression-based model. For a description of

related previous work refer [Krishna 98].

We also present some preliminary optimal motion planning results for the excavator for a given start and goal configuration. The optimal motions were computed by using the above described excavator model in a simulated annealing search approach. We also show results from executing the computed optimal motions on the testbed. The details of the motion planning will be the subject of a future paper.

II. Problem description

The testbed used for the work described in this paper is a Caterpillar 325 HEX, similar to the one in Fig 1. This machine has two independently actuated tracks which give it the ability to turn-in-place. The excavation activities are performed using four joints driven by hydraulic actuators. These are the swing (Sw), boom (Bm), stick (St), and bucket (Bk) joints.

The four joints only have one degree-of-freedom each. The Bm, St, and Bk joints move in the plane of the excavator's arm and are actuated by linear hydraulic cylinders labelled in Fig 1. The Sw and the tracks are actuated by separate hydraulic motors (rotary actuators) not visible in the figure. The tracks are usually not actuated when excavating; they are periodically used to reposition the excavator.

Fig 2 shows a simple schematic[1] of the hydraulic system of a CAT 325 HEX. The hydraulic system is driven by two hydraulic pumps which take low-pressure hydraulic oil from a tank (at atmospheric pressure) and output high-pressure oil. The power required to achieve the pressure rise is obtained from a single engine (not shown) which drives the pumps.

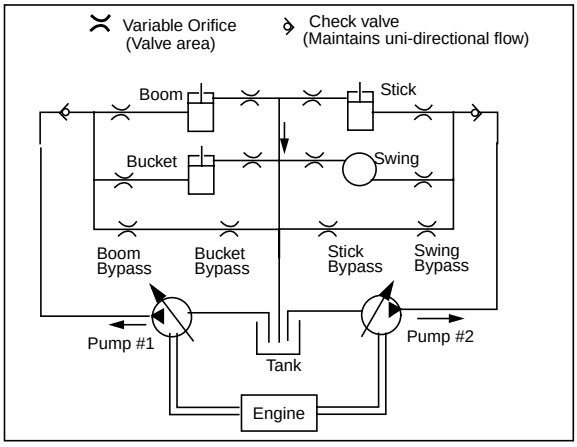

Fig. 2. Schematic of hydraulic system of a typical hydraulic excavator

The high-pressure oil flows to the hydraulic cylinders, which in turn actuate the different joints. Each of the two pumps supplies two implement circuits, i.e. one pump supplies the boom and bucket cylinders while the other supplies the stick cylinder and swing motor. (For the rest of the discussion the tracks will not be mentioned since they are not used during excavation. When they are used, one pump is dedicated to each track motor).

The flow from the pumps to the cylinders is controlled through variable orifices shown in Fig 2. If an orifice is completely closed no flow is supplied to that cylinder and no motion results. The hydraulic system shown above is an open-center system. In an open-center system the pumps do not reduce their output to zero. When no actuator flow is being demanded, the pumps still output a non-zero flow - between 10 and 20% of maximum flow for the testbed in Fig 1. This "idle" flow goes to the tank through the bypass (or "center") passages shown in the figure. When the actuators are being commanded to move, the bypass passages slowly close and are fully closed when maximum velocity is being demanded. The HEX hydraulic system also has non-linear components such as the check-valves (shown in Fig 2) which prevent oil flow from the cylinder to the pump.

The complete solution of the response of even the simplified hydraulic system in Fig 2 involves the simultaneous solution of multiple orifice flow equations, multiple compressibility equations for all the oil volumes, and force balance equations for each cylinder. A steady state solution would not include fluid compressibility and other dynamic effects. For a closer look at the equations involved refer [Krishna 98] and [Medanic 97].

A detailed model of the complete excavator hydraulic system has been constructed using a proprietary numerical solver, and its performance verified using results obtained from the CAT 325 testbed. This detailed dynamic model takes approx. 100 secs to simulate 1 sec. of a typical excavation cycle when running on a SUN Sparc20 workstation. A steady state approximation of the same model takes 1 sec. to simulate 2 secs of excavator motion.

III. Model construction

We are interested in developing a model for use in optimal motion planning which will capture the broad trends in robot behavior rather than the subtle effects. It is impossible to accurately characterize each HEX machine for which the optimal motion planning will be done. Hence, when implementing the motions computed using the model, it is necessary to use an approach, such as that suggested by Rowe [Rowe 97], which will perform a local optimization to adjust the motions for each machine.

The HEX model can be viewed as a combination of the linkage dynamic model and the actuator model. The linkage model refers to the force and torque balance relationships for all the moving masses of the excavator. The actuator model refers to a model of the entire apparatus (hydraulic pumps, fluid, valves, hoses etc.) involved in causing motion/force at the hydraulic cylinder pistons.

Linkage Model: The HEX is an open chain manipulator and its linkage dynamics can be written using the formulation common in the robotics literature ([Craig 89]):

$$M(\Theta)\ddot{\Theta} + V(\Theta, \dot{\Theta}) + G(\Theta) = \Upsilon \qquad (1)$$

[1] A different set of valves come into play when the cylinders move in the opposite direction. Also, the schematic shown is for the simplest operating mode - the flow routing is more complex for other operating modes.

where $M(\Theta)$ is the inertia matrix, $V(\Theta, \dot{\Theta})$ is the matrix of coriolis and centripetal terms, and $G(\Theta)$ is the gravity vector. Each joint torque τ_i in the torque vector Υ is related to the corresponding cylinder force via the transform:

$$\tau_i = f_{cyl_i} \cdot \frac{\partial}{\partial \theta_i} C_i(\theta_i) + \tau_{diggingForce} \quad (2)$$

where C is a non-linear function that maps the joint angles to cylinder extension, i.e. $x_{cyl} = C(\theta)$

The Υ vector should include the friction at the joints besides the cylinder force. However, since we do not wish to accurately capture the transient response and subtle effects in robot behavior, we neglect joint friction.

Actuator Model: The different orifice areas (Fig 2) are controlled by the position of a control spool. For example, the position of the boom control spool determines the boom pump-to-cylinder, cylinder-to-tank, and bypass areas. Thus a single spool position can represent all the orifice area variables for a joint.

From Fig 2 it can be seen that the steady state flow available to any cylinder/motor is a function of the orifice area of the relevant valves, and the ratio of force loads between the cylinders/motors competing for flow. For instance, if the boom inlet orifice area were zero, no flow could be supplied to it. To supply flow to the boom cylinder, the boom spool is shifted and the area opened. This allows high-pressure oil to flow from the pump to the cylinder. If the bucket spool were to be shifted now, the bucket cylinder will "steal" part of the boom's flow since the bucket cylinder is lightly loaded as compared to the boom cylinder. This is true if the total pump flow is inadequate to supply both cylinders at the same time - if the bucket or boom orifice areas are only partially open, the distribution of flows between the cylinders will be different.

It is important to capture these actuator interactions since their effect can be significant. Our approach to modeling these interactions is to use a non-linear function approximator to approximate this functional mapping from the space of inputs - the spool positions (which determine the different orifice areas) and cylinder forces, to the outputs - the flow to each cylinder/motor, which in turn determines its steady-state velocity. We settle for a steady-state actuator model since the transient response is not significant for cylinder motions much longer than the cylinder's time constant, and we need the model for just such a purpose, i.e. simulating a few seconds of machine motion.

A total of four neural networks are used - one for each joint of the HEX. The outputs of the boom, stick and bucket networks are steady-state cylinder velocity values, while the swing output is swing acceleration. The swing is different from the other joints since it is a rotational joint with a large rotational inertia. It therefore has long acceleration and deceleration phases, unlike the three joints driven by linear cylinders. The network inputs were chosen after studying the flow routing schematics for the most complex HEX operating mode, where the valve arrangement is quite different from that in Fig 2.

In Table 1, SP refers to spool position, F refers to force, I refers to rotational inertia, Vel refers to velocity and Acc refers to acceleration.

	Bm	**St**	**Bk**	**Sw**
Input #1	Bm SP	Bm SP	Bm SP	Bm SP
Input #2	St SP	St SP	St SP	St SP
Input #3	Bk SP	Bk SP	Bk SP	Bk SP
Input #4	Sw SP	Sw SP	Sw SP	Sw SP
Input #5	Bm F	St F	Bk F	Sw I
Input #6	Bk F	Sw I	Bm F	St F
Input #7	-	-	-	Sw Vel
Output	Bm Vel	St Vel	Bk Vel	Sw Acc

TABLE 1: Inputs/Output for HEX actuator model networks

Data Collection: Training data for the actuator model neural networks was collected using a slow but complete analytical machine model[2]. The slow model was driven through a number of motion sequences to adequately cover the operating space of each network. While the spool positions are directly controllable the cylinder forces are not. The motions were therefore repeated for a fully loaded bucket, half-empty bucket, and completely empty bucket to cover the cylinder load dimensions. All motions were performed slowly to minimize transient effects.

The spool positions have a range of ± 11 mm. Data was sampled at a resolution of 1mm along the spool position axes. The cylinder forces were determined by the excavator's configuration.

Implementation Detail: While training the networks it was found that splitting up the input space was desirable to allow the networks to better approximate the functions. Hence, each joint's input space was split into 8 non-overlapping parts resulting in a total of 4*8=32 networks. The 8 represents the number of permutations possible with regard to the direction of motion of the Bm, St, and Bk. Each cylinder can move in or out, resulting in 2^3 permutations, and hence 8 sub-spaces - the Sw is symmetric and hence clockwise and ccw rotation are identical. The network for each sub-space was trained using data that belonged to it.

Complete Model Construction: The complete excavator model is constructed by partitioning the actuator dynamics and linkage dynamics into two separate problems. Instead of solving them simultaneously they are solved in a serial fashion. First, the linkage dynamic model (Eqn 1) is used to compute the forces for a given excavator state. This force is assumed to remain constant over the time period that the actuator response is simulated using the learned actuator model. The results of

[2]The testbed was not used to collect training data due to our inability to measure spool positions directly on it. Also, the initial attempt at training used a large amount of training data which was much easier to collect from a simulation model (which had been verified to match the testbed).

the actuator simulation are used to compute a new state, which is used in the linkage dynamic model for force computation as the cycle continues. The steps are spelled out explicitly below.

- Step #1: The dynamic model of the excavator is used to compute the force loads on the different hydraulic cylinders.
- Step #2: The force loads computed in Step #1 are used with the input spool commands to compute the resulting cylinder velocities using the corresponding neural networks. The swing table uses the current swing velocity to compute the swing acceleration.
- Step #3: The computed velocities (or accelerations) are integrated to obtain an updated excavator state. (Repeat steps 1 through 3)

IV. Results (Modeling)

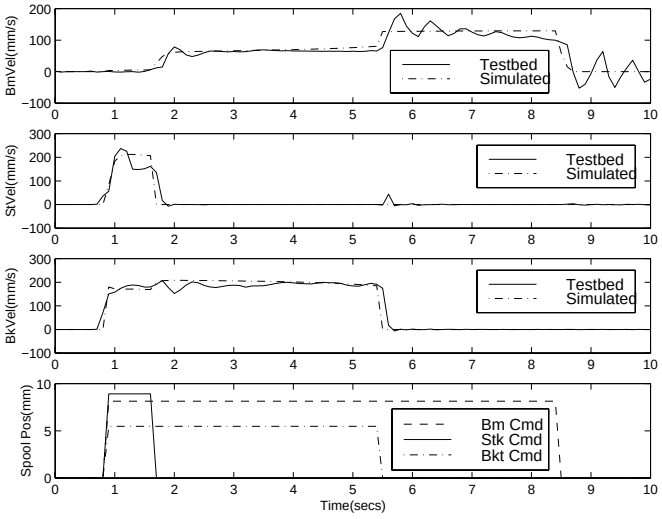

Fig. 3. Bm, St, Bk cylinder velocity plots (Test #1)

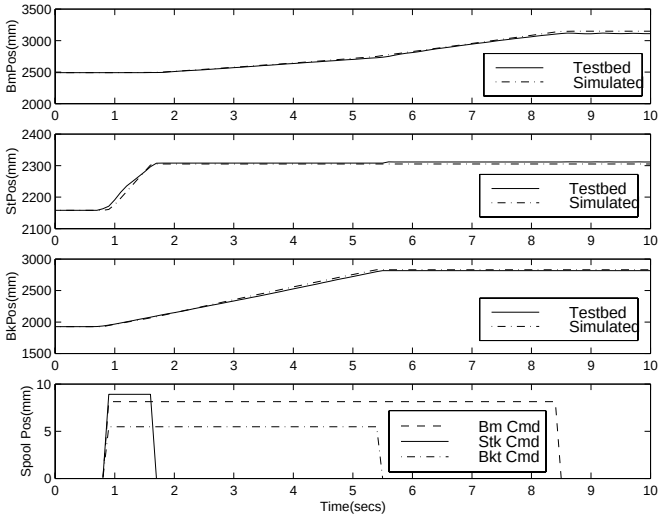

Fig. 4. Bm, St, Bk cylinder position plots (Test #1)

The above approach was used to construct a complete model of a CAT 325 HEX. The performance of the model was evaluated by comparing it to the performance of the excavator testbed. During the first test (Fig 3, Fig 4), the boom, stick and bucket cylinders were actuated simultaneously to demonstrate the interaction between them, while the second test (Fig 5, Fig 6) demonstrated interaction between the swing and stick joints. No digging was involved during any of the tests and therefore no digging forces at the bucket tip were applied. This was done since the excavator testbed is not setup to measure digging forces during interaction with the soil. (The structure of the model however does allow the model to simulate machine motions in response to external tip forces if the forces are known - see Eqn 2).

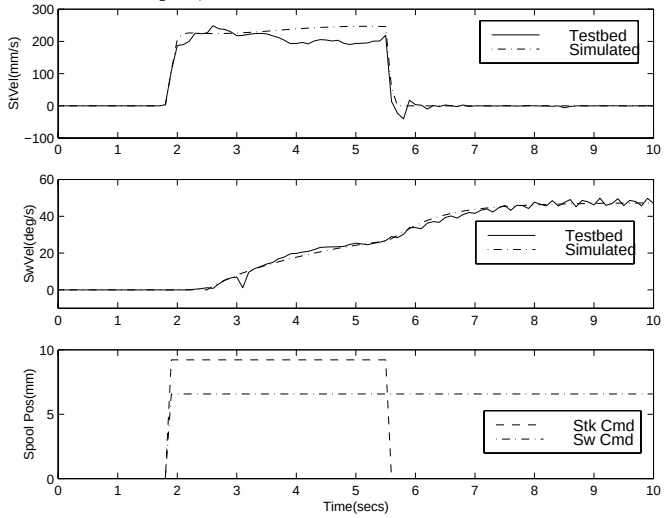

Fig. 5. St cylinder, Sw joint velocity plots (Test #2)

The current implementation of the HEX model uses 32 networks to cover the entire space for four joints. The model runs at a run-time:real-time ratio of 75:1, i.e. simulating 75 seconds of motion requires 1 sec. of computation time on a SUN Sparc 20 workstation.

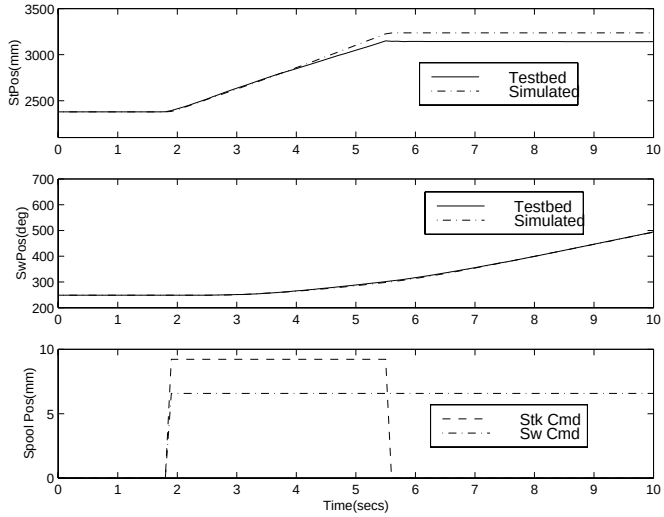

Fig. 6. St cylinder, Sw joint position plots (Test #2)

V. Optimal Motion Planning

One application for the machine model developed here is in

planning the optimal free-space motion between specified start and end states. It is not possible to use previously developed approaches like [Shiller 91] since they were developed for electric drive robots. They assume the ability to control actuator torque, and the knowledge of actuator torque limit curves. This concept cannot be transplanted easily to hydraulic machines. Our optimal motion computation approach involves discretizing the *input* space and then performing a search in that space to compute the most optimal command sequence. An advantage of using low-level joint controllers is that the closed-loop HEX (Fig 7) is a lower order system, thus allowing a coarser discretization of the command *input* space. The *inputs* for the closed-loop HEX are the joint position commands (u_{joint}; ref. Fig 7). The controllers also help insulate the robot from disturbances.

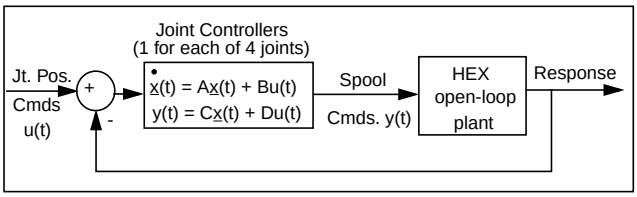

Fig. 7. HEX with joint position controllers

The HEX testbed has independent PD (proportional-derivative) joint position controllers for each of its four joints. The joint controllers are modeled in state-space form as:

$$\dot{\underline{x}}_{joint} = A_{joint}\underline{x}_{joint} + B_{joint}u_{joint} \quad (3)$$

$$y_{joint} = C_{joint}\underline{x}_{joint} + D_{joint}u_{joint} \quad (4)$$

where, u_{joint} is the position command to the joint, $\underline{x}_{joint}$ is a 5 element vector of state variables, y_{joint} is the spool position command, and $A_{joint}, B_{joint}, C_{joint}, D_{joint}$ are the state-space matrices for the controllers. These controllers cannot be described here in greater detail for proprietary reasons.

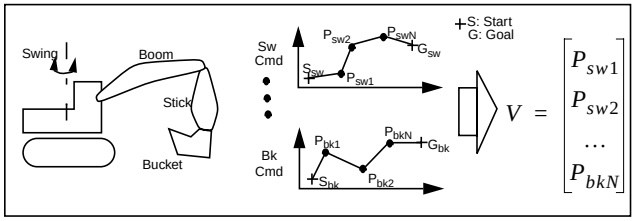

Fig. 8. A typical search vector (V = search vector)

Since we wish to primarily demonstrate the use of this modeling approach in optimal motion computation, we will treat the optimizer as a black box whose details will be described in a future paper. The optimizer searches the command space of the closed-loop HEX (joint position commands) to compute the set of joint command sequences (ref. Fig 8) that minimize the objective function. In the examples shown here we only use time in the objective function. We also have the ability to incorporate energy in the objective function.

The optimizer takes four inputs -

- Task specification - an initial and final position
- Robot model - describes a constraint surface for the optimization since it embodies the limitations of the robot.
- Task constraints - can include obstacles in the environment that the robot should avoid, in addition to task specific constraints such as - "The robot should not open the bucket until it reaches the truck".
- Initial guess - this provides the seed to start the search for the optimum. We use the Simulated Annealing algorithm to compute the optimum and this algorithm is quite insensitive to the choice of the initial guess.

VI. Results (Motion Planning)

Fig 9(a) and (b) show the HEX located on a digging bench with a truck parked behind it in a typical loading configuration. The terrain shown in the figures is not synthetic. It was created using a range scanner located on our HEX testbed. We show the results from two tests -

- Test #3: Starting from position #1 with a loaded bucket (35000N load) and unloading it in the truck (position #2),
- Test #4: Starting from position #2 with an empty bucket and returning to the dig face (position #1 - with the bucket open).

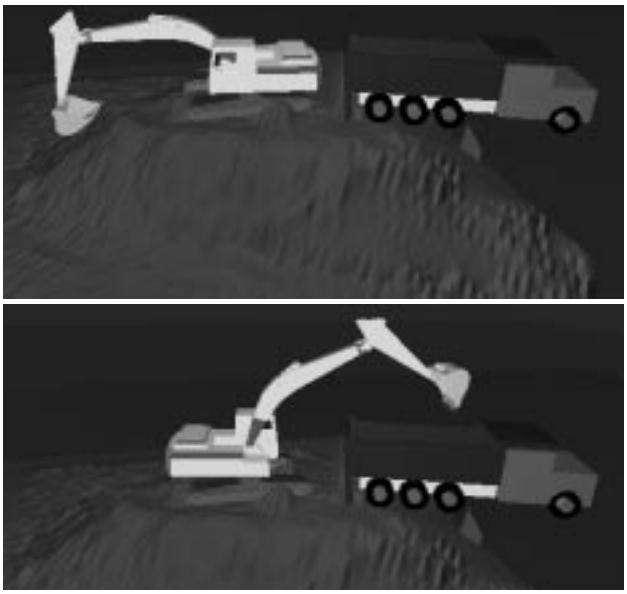

Fig. 9. (a) Position #1
(b) Position #2

In test #3 we enforce a spillage constraint that penalizes opening of the bucket prior to reaching the truck. This constraint is not enforced for test #4. Both tests have a collision constraint that severely penalizes collision with the environment. To demonstrate the insensitivity of the method to the choice of the *initial guess* we have chosen a motion (not shown) that collides with the side of the truck.

Fig 10 and Fig 11 show joint motions for tests #3, 4. Each figure has three traces - the thick line is the motion as demonstrated by a human expert. Since we do not have a benchmark of how close we are to the globally optimal way of performing the tasks, we chose to compare our motions to that of a human expert operator. The dash-dot traces are the optimized motions

computed by the optimization system using the model. The optimization system searches in the command space (position commands to the position controller) and its results are a series of position commands. These are not shown in order to reduce clutter, and also because the relevant result is the HEX motion, which is used by the optimization to compute the objective function. The dashed traces are the HEX testbed motions when the computed optimum is implemented on the testbed. We use the Denavit-Hartenberg convention ([Craig 89]) for the kinematics of the excavator.

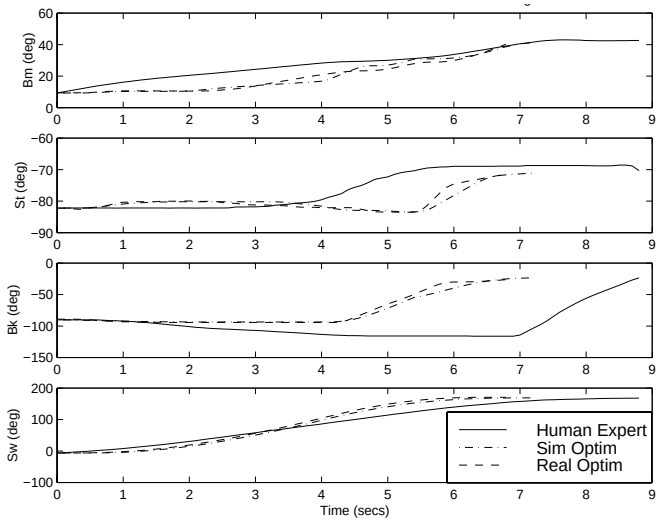

Fig. 10. Results from Test #3 (Position #1 -> Position #2; Loaded bucket)

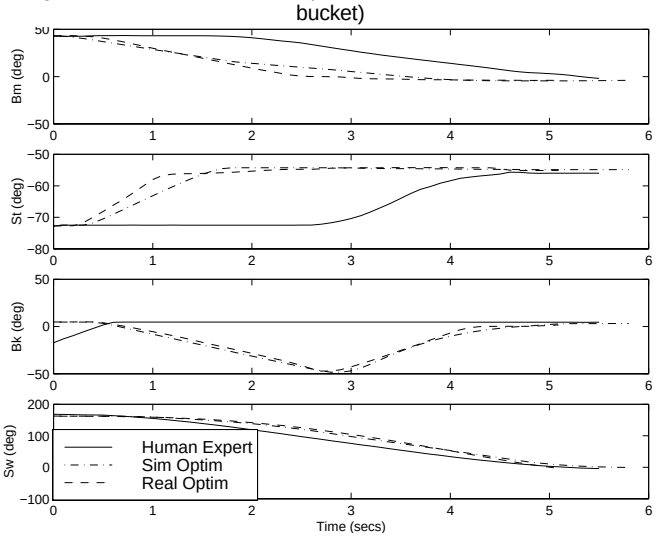

Fig. 11. Results from Test #4 (Position #2->Position #1; Empty bucket)

The results for Test #3 are faster than those of the expert human operator by 1.5 secs, while they match the human in Test #4. It should be noted that the human expert used for comparison is one of the very best, and the results shown are for the third loading pass in a 6-pass loading operation, where one pass is defined as transferring a load of soil in the truck and returning for the next dig. In both cases the simulated results agree very well with the testbed results.

The optimization takes 2 hours when running on four Sun Sparc20 workstations in parallel. We are not concerned by the run-time of this initial implementation of the optimization since we believe that it can be significantly improved by refining the way the optimization is performed. The important result we have demonstrated is the feasibility of computing optimal motions using this search approach for hydraulic machines. This is the first time that optimal motions have been computed and demonstrated for a hydraulic machine.

VII. Conclusions

It is possible to approximate the actuator interactions using sets of linear surfaces instead of a non-linear function approximator like a neural network. However breaking up the actuator response surfaces into linear sections will be non-trivial.

The down-side of using a function approximator is the need to supply it enough data to ensure that the approximated function is close to the desired function. Neural networks offer a significant(5X) speed advantage in our case over non-linear regression-based function approximation techniques. However, they offer very little control over the function approximation created, thus requiring more extensive coverage of the data space.

The simulated annealing algorithm used for the optimization is very robust against local minima. From closer analysis of our time-optimization cost surfaces we have ascertained that the cost surface is pocked with local minima. It is possible that a different form of the cost function may offer fewer minima, but we have no way of finding that form. We have tried our method on 8 different HEX tasks, using time as the objective functions. The results match or exceed the motions of the human expert. We have also used energy as the objective function for the same 8 tasks and the results offer interesting insights into the operations. We believe that this method can act as an important tool to aid designers in optimizing machine designs.

VIII. References

[Craig 89] Craig, John J., "*Introduction to Robotics: Mechanics and Control*", 2nd Ed., Addison-Wesley Publishing Company, 1989.

[Krishna 98] Krishna, M., Bares, J., "*Hydraulic System Modeling through Memory-based Learning*", IEEE Conf. on Intelligent Robots and Systems, (IROS), Victoria, B.C., Canada, Oct '98, pp. 1733-8.

[Medanic 97] Medanic, J., et. al., "*Robust Multivariable Non-linear Control of a Two Link Excavator: Part I*", IEEE Conf. on Decision and Control Systems, pp. 4231-6, Dec 1997.

[Narendra 90] Narendra, K.S., et. al., "*Identification and control of dynamical systems using neural networks*", IEEE Tran. on Neural Networks, 1990, Vol. 1, pp. 4-27

[Rowe 97] Rowe, P., et. al., "*Parameterized Scripts for Motion Planning*", IROS, Vol. 2, pp. 1119-24, Sep. '97.

[Shiller 91] Shiller, Z., et. al., "*On Computing the Global Time-Optimal Motions of Robotic Manipulators in the Presence of Obstacles*", IEEE Tran. on Robotics and Automation, Vol. 7, No. 6, Dec. '91.