An Optimal Sliding Mode Controller Applied to Human Motion Synthesis with Robotic Implementation

Adam Spiers, Guido Herrmann and Chris Melhuish

Abstract—The operational space formulation is applied to a practical robot system in order to generate realistic human reaching motion based on the minimisation of ‘effort’, a function of gravity and weighting gains. We present a novel optimal sliding mode controller that uses techniques of steepest descent to achieve this minimisation without affecting the task controller. The sliding mode optimal controller is verified both theoretically and by practical evaluation on simulated and physical two degree of freedom (dof) robotic arms. These arms produce redundant reaching motion that is similar to that observed from human subjects.

We also present our modifications to the effort function, for implementation of smooth joint limits. A separate sliding mode controller for task control is also presented. Both sliding mode controllers guarantee robustness to model uncertainty and actuator disturbances e.g. friction.

I. INTRODUCTION

Recent advances in humanoid robot technology have sparked growing interest in alternative control schemes that would aid the placement of robots into the human environment. At present many humanoid robots continue to use joint level, position (or it’s derivatives) based control schemes borrowed from classical industrial robot control and grounded in inverse kinematics. Such schemes have been designed for the high speed and accuracy requirements of industrial robots that work in highly structured environments, usually surrounded by a safety barrier that prevents humans from entering the robot’s workspace. For humanoid robots, where the environment is dynamic with potentially close proximity to humans, such control schemes seem highly inappropriate.

By implementing a dynamics rather than kinematics based control scheme one achieves robot motion without strict definition of joint level trajectories. This permits the robot to make full use of it’s redundancy to achieve task space goals (i.e. positioning of the end effector) in a number of different configurations, or ‘postures’.

During human robot interaction, it is important that the human maintains a high degree of confidence in the robot’s movements and that these movements are familiar and predictable. One way of achieving confidence by these means is to make the robot move in a familiar and human-like way, choosing it’s posture in a way that resembles the posture of a human. Biomechanic analysis of human motion has suggested that this can be achieved by making the motion

optimal with regard to some quantity (as will be discussed in section I-A).

In this paper we modify the operational space formulation developed by Khatib et al. [3] to achieve dynamic robot control. The control method implements two separate controllers which separately and hierarchically drive the end effector of the robot (the task controller) and control the redundant degrees of freedom (the posture controller).

In order to improve the applicability of the operational space controller to practical robot systems we provide an alternative, novel and robust posture controller that is based on sliding mode control techniques and may be verified by Lyapunov methods. Experimental results on a simulated and a practical robot system show that this controller is indeed more optimal than the original posture controller provided in [10]. The new controller is robust against model uncertainty and stability can be proven via Lyapunov stability. These are both important considerations in potentially dangerous practical systems, whose parameters change over time [7].

A. Human Motion though Optimisation

Human motion tends to follow certain paths, patterns and constraints to produce fairly consistent motion patterns [4]. Motion that does not fit into these definitions may be instantly recognised as being unusual and unnatural. It is our goal to create a controller that generates human-like motion in order to better facilitate human/robot interaction without compromising movement accuracy and goal achievement.

A great deal of evidence supports the claim that some form of optimisation is taking place during the planning and/or execution of motion. In [2], the authors propose a model that reproduces features observed in multi-joint planar reaching motion. This model was based on dynamic minimisation of ‘jerk’, the 3rd derivative of position. The resulting motion was smooth with a bell shaped velocity profile for point to point motions of the hand. However, this model was based solely on the kinematic hand trajectory. In [11], Soechting et al. demonstrated that posture of the arm during reaching motions cannot be defined kinematically and hypothesised that posture is formulated from minimisation of the work necessary to transport the hand to it’s end goal. This dynamic form of minimisation is similar to the approach of Uno et al. [13] who proposed a minimum torque model for reconstructing planar arm motion. Such dynamic optimisation approaches have been adopted for the work presented in this paper.

As stated in the introduction, the operational space control method provides a method of decoupling control of different
aspects of the robot via separate controllers. Additionally, the contribution of each controller may be placed in a ‘hierarchy’, where task level control gains precedence over the posture level control. This is directly analogous to the human motion observations made in [6] where subjects overcame exoskeleton introduced disturbances at the elbow to consistently repeat point to point motion with their finger. This analogy was extended in [10], where De Sapia et al. formulated a posture controller based on the minimisation of gross muscular effort, obtained via a complex simulation of the human musculoskeletal system. This produced convincing human like motion over the whole body.

Sliding mode control has been used previously in synthesising human motion. Rengifo et al. [9] have investigated the problem of optimal neuromuscular control based on joint position tracking errors and torque squared. Lister et al. [5] used a sliding mode controller to achieve gait trajectory tracking for a 10 dof human locomotor system. In both cases, sliding mode controllers have been used to track predefined trajectories in order to determine biological actuation schemes.

In our application it is necessary for the robotic system to dynamically generate human-like motion at run time without any predefined trajectory data. In order to test candidate control schemes, we have focused on vertical, planar reaching motions. This is a common task with interesting dynamic characteristics that may be repeated on a simple 2 dof robot with shoulder and elbow flexion.

Observations of this motion from ten volunteers confirmed in all cases that the motion of the arm follows a general pattern whereby the elbow is folded and unfolded as the arm is lifted at the shoulder (Figure 1). This is in line with the observations in [10] where it is suggested that human motion emerges from a minimisation of muscular effort. Indeed, folding of the elbow would reduce the moment of inertia about the shoulder and require less energy when reaching to an overhead object.

In order to extend the possible variation of the synthesised reaching trajectory, an element of redundancy was added to the reaching task. Rather than reaching to a target with a fully defined Cartesian location, the target is now defined only in terms of height. This permits horizontal redundancy in the 2 dof arm. Note that we do not define target orientation in our reaching task.

II. OPERATIONAL SPACE FRAMEWORK

The Operational Space Framework was introduced by Khatib in 1987 [3] as a method of robot control that hinges on the projection of system dynamics into the same coordinate system that is used to command the end effector of the robot. This method also serves to decouple the control scheme into task and posture controllers.

A. Task Controller Design

The operational space framework approach of [10] uses a feedback linearisation method in its primary task controller to estimate and compensate for the dynamics of the robot system under control. The joint space dynamics of a robot system are defined by \( \Gamma \), a vector of system torques, \( q \), the generalised (joint space) co-ordinate vector, \( \Lambda \), the mass/inertia matrix, \( b \), centrifugal/coriolis terms and \( g \), gravity terms.

\[
A(q)\dot{q} + b(q, \dot{q}) + g(q) = \Gamma,
\]

while joint co-ordinates are represented by \( q \), Cartesian, operational space co-ordinates are represented by \( X \). In the case of our redundant reaching task we are interested only in the height of the end effector \( X_y \). In order to project the (joint space) dynamics of the robot into task space, the dynamically consistent inverse of the Jacobian, \( J \), is employed (as the non-square Jacobian cannot be inverted by standard methods):

\[
J = \frac{\partial X_y}{\partial q}, \quad \bar{J} = A^{-1}J^T(JA^{-1}J^T)^{-1}
\]

The following task space dynamics are a projection of the robot dynamics into the Cartesian space of the end effector. Rather than joint torques, the task space dynamics equate to a vector of forces, \( f \), acting on the end effector.

\[
\Lambda(q)\dot{X_y} + \mu(q, \dot{q}) + \rho(q) = f,
\]

\[
\Lambda = (JA^{-1}J^T)^{-1}, \quad \mu = J^Tb - \Lambda q, \quad \rho = J^Tg
\]

Due to feedback linearisation \( f \) may be driven by a linear PD controller. Conversion back to joint space torques \( \Gamma \) via the Jacobian allows the control forces, \( f \), to be applied to the robot’s actuators:

\[
f^* = -K_s(X_y - X_{y0}) - K_v\dot{X}_y,\n\]

\[
f = \dot{\bar{J}}f^* + \ddot{\mu} + \dddot{\rho},\n\]

\[
\Gamma_T = J^Tf
\]

B. Sliding Mode Task Controller Modification

In [7], the authors express concern over the poor performance of force based operational space methods (the type described in [3], [10]) due to inevitable model mismatch with physical systems. In our case, initial hardware implementation of the operational space controller led to poor performance from the physical robot system for this reason. In particular, small changes in demand led to large steady state errors, where the low peak velocity & acceleration were insufficient to overcome un-modelled actuator stiction and friction. In order to overcome this error, a sliding mode element was introduced to the task controller scheme [12]. The role of the sliding surface is to force the position error of the end effector to zero. The sliding mode controller is based on the sliding surface \( s_t \), where \( X_{y0} \) is the desired position:

\[
s_t = (X_{y0} - X_y) + K_v(X_{y0} - X_y),
\]

\[
K_s = -(K_s - K_v)K_s, \quad K_v = (K_s + K_v^2)/K_s,
\]
and is integrated into the operational space control force \( f \) (4) as follows:

\[
usl = -Kssl s_t
\]

\[\]

\[
f = \hat{A}f^* + \hat{A} + \hat{A}X_0, \]

\[\]

\[
f^* = -K_s(X_y - X_{y0}) - K_v(X_y - X_{y0}), \] (6)

Note that the control terms have also been modified to create a tracking of the control signal where \( \hat{A}X_0 \) complements the tracking control law. When combined with a filtered step demand \( X_{yd} \) (from (7)) it is possible to overcome the jerkiness that may be associated with a task level sliding mode control scheme:

\[
X_{y0}(s) = \frac{\omega_n^2 s^2 + 2\zeta \omega_n s + \omega_n^2}{s^2 + 2\zeta \omega_n s + \omega_n^2}X_{yd}(s), \] (7)

Note that for simplicity, we use the same notation for the time domain and the Laplace domain variable \( X_{y0}(t) \) and \( X_{y0}(s) \).

By forcing \( s_t \) to zero, the controller also forces the position and velocity errors \( (e = X_y - X_{y0} \text{ and } \dot{e} = \dot{X}_y - X_{y0}) \) to zero. This modification significantly improves the performance of the physical robotic system, as has been reflected in practical and simulated experiments with deliberate model mismatch [12].

C. Original Posture Controller

In [10], the posture controller is a function of gravity terms (from 1) and an in-depth model of the human musculoskeletal system based on antagonistic muscle pairs with realistic dynamics. In previous work [12] we have shown that human-like motion can be achieved by substitution of this complex model with a simple activation matrix \( K_a \). Modification of the gains of this matrix alter the landscape of the effort cost function \( U(q) \) to produces different trajectories. The gains may also be regarded as preferential joint weightings:

\[
U(q) = s^T K_a^{-1} g, \quad K_a = \begin{bmatrix} K_{a1} & 0 \\ 0 & K_{a2} \end{bmatrix} \] (8)

We have further modified the cost function to include joint limits \( q_{L_n} \) as a smooth function of in the joint variables.

\[
U = s^T K_a^{-1} g + \frac{1}{|q_1 - q_{L_1}|^{K_{L_1}} + \frac{1}{|q_2 - q_{L_2}|^{K_{L_2}}}} + \cdots + \frac{1}{|q_n - q_{L_n}|^{K_{L_n}} + \frac{1}{|q_{L_n}|^{K_{L_n}}}} \] (9)

where the exponents \( K_{L_n} \) are even numbers whose magnitude corresponds to the magnitude and profile of the individual joint limit’s contribution to \( U_p \) (practically illustrated in Section IV). The cost function is minimised in [10] via the following controller.

\[
\Gamma_p = -\kappa_p (\nabla U)^T - K_a \dot{q}, \quad \nabla U = \left( \frac{\partial U}{\partial q} \right) \] (10)

This is integrated into the operational space formulation via the term \( N^T \), which decouples the posture control action from that of the task controller. This decoupling is at the heart of the operational space control scheme and means that the task control trajectory will remain static regardless of the action of the posture controller.

\[
\Gamma = \hat{f}^{T} f + N^{T} \Gamma_p, \quad N^T = (I - \hat{f}^{T} \hat{f}) \] (11)

III. OPTIMAL SLIDING MODE CONTROLLER DESIGN

A. Steepest descent sliding surface

In standard Lyapunov theory the stability of nonlinear systems can be verified by analysis of a positive definite function \( V \). This is historically regarded as the energy function of the system. If this function decreases then the system is expending energy and will eventually come to a rest, it is therefore stable. This can be verified by checking if the derivative \( \dot{V} \) is a negative definite function. By designing a controller that is able to make the energy function decrease, one is able to control the behavior of a system that could otherwise be unstable.

The optimal sliding mode control method presented here parallels basic Lyapunov theory but adapts the minimisation of the energy function for the minimisation of a cost function. The Lyapunov technique may be applied to the effort function \( U(q) \), the time derivative of effort is therefore:

\[
\dot{U}(q) = \frac{\partial U}{\partial q} \dot{q} \] (12)

The following choice achieves steepest descent:

\[
\dot{q} = -k \left( \frac{\partial U}{\partial q} \right)^T \] (13)

Substituting (13) into (12) gives:

\[
\dot{U}(q) = -k \frac{\partial U}{\partial q} \left( \frac{\partial U}{\partial q} \right)^T = -k \left| \frac{\partial U}{\partial q} \right|^2 \] (14)

\( \dot{U}(q) \) is negative definite, providing that \( U \) is strictly convex in \( q \). The choice of (13) may be adopted to create the sliding surface \( \tilde{s} \):

\[
\tilde{s} = \dot{q} + k \left( \frac{\partial U}{\partial q} \right)^T \] (15)

Thus, providing the posture controller achieves \( \tilde{s} = 0 \) to guarantee that (13) holds, then \( \dot{U}(q) \) is decreasing. This can be achieved via sliding mode approaches.

B. Lyapunov Function determination

A Lyapunov candidate for the sliding function \( \tilde{s} \) is:

\[
V_{\tilde{s}} = \frac{1}{2} \tilde{s}^T \tilde{s}, \quad V_{\dot{\tilde{s}}} = \tilde{s}^T \dot{\tilde{s}} \] (16)

\[
V_{\dot{s}} = \tilde{s}^T \left( \dot{q} + k \left( \frac{\partial U}{\partial q} \right)^T \right) \dot{q} \] (17)

by substituting \( \dot{q} \) from (1):

\[
V_{\dot{s}} = \tilde{s}^T \left( -A^{-1} b - A^{-1} g - A^{-1} \Gamma + k \left( \frac{\partial^2 U}{\partial q^2} \right) \dot{q} \right) \] (18)

In order to make the notation more compact, define:

\[
R = -A^{-1} b - A^{-1} g + k \left( \frac{\partial^2 U}{\partial q^2} \right) \dot{q}, \] (19)

leading to the Lyapunov candidate function:

\[
V_{\dot{s}} = \tilde{s}^T (R + A^{-1} \Gamma) \] (20)
C. Control Method

Due to model uncertainty we do not have accurate knowledge of $R$ and $A$ so by using estimates $\hat{R}$ and $\hat{A}$, we can introduce the controller:

$$\Gamma = \hat{A} \left( -\hat{R} - K_{st} \frac{\hat{s}}{||\hat{s}||} \right)$$ (21)

substituting into (20) gives $V < 0$ for:

$$K_{st} > \frac{\|A^{-1}(AR - \hat{A}\hat{R})\|}{\lambda_{\text{min}}(\hat{A})}, \quad \lambda_{\text{min}}(\hat{A}) > 0$$ (22)

where:

$$\hat{A} = A^{-1}\hat{A} + \hat{A}^T A^{-T}$$ (23)

Note that if $\hat{R} = 0$ then the following control scheme results:

$$\Gamma = -K_{st} \frac{\hat{A}\hat{s}}{||\hat{s}||}$$ (24)

Substituting into (20) gives:

$$V_s = \hat{s}^T R - K_{st} \frac{\hat{A}\hat{s}}{||\hat{s}||}$$ (25)

and again, only semi-global stability is guaranteed assuming (22) holds for some compact set and $\hat{R} = 0$. Note that these suggested control laws (21) or (24) do not consider the operational space approach, which shall be discussed next.

D. Velocity Decoupling

In order to implement the above method in operational space, it is necessary to decouple this control action (which forms the posture controller) from the task control action. We achieve decoupling via splitting of the joint space velocities into the abstract quantities of task velocity $\dot{q}_t$ and posture velocity $\dot{q}_p$, each of which is defined by the control action of the respective controller. Together these quantities constitute the overall motion of the system as:

$$\dot{q} = \dot{q}_t + \dot{q}_p$$ (26)

The Jacobian permits derivation of task space velocity from joint space velocity and vice versa. We know that the Cartesian task space velocity satisfies:

$$J\dot{q} = \dot{X}_s, \quad J = \frac{\partial X_s}{\partial q}$$ (27)

The Jacobian may be manipulated to decompose the velocity as in (26),

$$J\dot{q} = \dot{X}_s = J(J^T(JJ^T)^{-1}J)\dot{q} + J(I - J^T(JJ^T)^{-1}J)\dot{q}$$ (28)

Hence, the velocity is composed of velocity contributed by the task controller:

$$\dot{q}_t = (JJ^T)^{-1}J\dot{q}$$ (29)

and the posture controller:

$$\dot{q}_p = (I - (JJ^T)^{-1}J)\dot{q}$$ (30)

For ease of notation these terms may be compactly defined as:

$$B = (I - J^T(JJ^T)^{-1}J), \quad \dot{B} = (J^T(JJ^T)^{-1}J)$$ (31)

$B$ and $\dot{B}$ are symmetric matrices that are analogous to the $N^T$ matrix that Khatib uses for decoupling posture control from task control (where $N^T = I - J^TF$ and $J = A^{-1}J^T(JA^{-1}J)^{-1}$ [3]. The following relationships are satisfied by $B$ and $\dot{B}$

$$BB = B, \quad \dot{BB} = \dot{B}, \quad \dot{BB} = 0, \quad \dot{BB} = 0, \quad B + \dot{B} = I,$$ (32)

Moreover,

$$JB = 0, \quad N^T\dot{B} = 0, \quad \dot{B}A^{-1}N^T = 0,$$ (33)

so that:

$$N^T\dot{q}_t = 0,$$ (34)

thereby justifying again the choice for $\dot{q}_t$ and $\dot{q}_p$. Clearly, the posture controller is unable to regulate the task coordinate velocity $\dot{q}_t$. Hence, the posture controller has to act only on the posture velocity $\dot{q}_p$ only. The suggested sliding surface for steepest descent (15) should now be modified to reflect this:

$$\dot{s} = B\hat{s}$$ (35)

leading to:

$$\dot{s} = B\hat{s} = B\dot{q} + kB \left( \frac{\partial U}{\partial q} \right)^T$$ (36)

Hence, the following choice (instead of (13)) shall be enforced using a sliding mode control approach.

$$q_p = -kB \left( \frac{\partial U}{\partial q} \right)^T$$ (37)

E. Cost Function Revision for Operational Space Analysis

The cost function $U(q) = U(q_p,q_t)$ shall now be interpreted as a function of the task coordinate, $q_t$, and the posture coordinate, $q_p$. The task coordinate $q_t$ is predetermined by the task controller, i.e. $q_t = q_t(t)$:

$$U(q) = U(q_p,q_t(t)) = U(q_p,t),$$ (38)

where $U$ is now a function of time $t$ and the posture coordinate, and is to be minimised in $q_p$. It follows that velocity decoupling may be applied as in (28):

$$U(q) = \frac{\partial U}{\partial q}Bq + \frac{\partial U}{\partial q} \dot{B}q$$ (39)

$$U(q) = \frac{\partial U}{\partial q}Bq + \frac{\partial U}{\partial q} \dot{B}q$$ (40)

Choosing the posture velocity part to be:

$$B\dot{q} = -kB \left( \frac{\partial U}{\partial q} \right)^T$$ (41)

it follows that:

$$U(q) = -k \left\| \frac{\partial U}{\partial q_p}B \right\|^2 + \frac{\partial U}{\partial q}Bq$$ (42)

Thus, the following is an implication of this analysis (the main result of this paper):
**Observation:** Given a compact set $C$ in
\[
\left| \frac{\partial U}{\partial q_i} \dot{B} q_i \right| < K
\] (43)
in $q$ and $U(q_p,t)$ is uniformly convex in $q_p$ so that there is a bounded set $C_u$, for which:
\[
q_p \notin C_u : U = -k \left| \frac{\partial U}{\partial q} \dot{B} q \right|^2 + \frac{\partial U}{\partial q_i} \dot{B} q_i < 0,
\] (44)
then the minimisation of $U(q_p,t)$ is guaranteed up to an accuracy given by the radius of $C_u$.

**Remark:** This is a generic observation known from Lyapunov theory applied to the context of effort minimization in humanoid robotics. Practical results provided later show that cost functions, obtained from our robotic examples, can provide good experimental results.

**F. Sliding Mode Analysis for Task/Posture problem**

To permit control in the operational space it is possible to augment $\dot{s}$ (15) with the splitting term $B$. Hence, $\dot{s} = B\ddot{s}$:
\[
\dot{s} = B \left( \dot{q} + k \frac{\partial U}{\partial q} \right)
\] (45)
The candidate Lyapunov function is:
\[
\dot{V}_s = s^T \dot{s} = \frac{1}{2} \dot{s} \dot{B}^T \dot{B} \ddot{s}
\] (46)
Hence,
\[
\dot{V}_s = s^T \left( B \ddot{q} + k \frac{\partial U}{\partial q} \dot{q} + BB \left( \dot{q} + k \frac{\partial U}{\partial q} \right) \right)
\] (47)

We may now define $\dot{R}$ (which is similar to (19)):
\[
\dot{R} = -A^{-1} \dot{b} - A^{-1} \dot{g} + k \frac{\partial U}{\partial q} \dot{q} + B \left( \dot{q} + k \frac{\partial U}{\partial q} \right)
\] (48)

So $\dot{V}_s$ becomes:
\[
\dot{V}_s = s^T (BA^{-1} \Gamma + B \dot{R})
\] (49)
\[
\dot{V}_s = s^T B (A^{-1} \Gamma + \dot{R})
\] (50)
The control methods proposed in Sections III-C may now be suitably adapted to equation (50) to permit decoupled, operational space control.

**G. Control Implementation**

If $A$ is not assumed, but an estimate, $\hat{A}$, then the following control torque will provide stability as follows:
\[
\Gamma = J^T f + \hat{N}^T \left( -K_d \frac{\hat{A}B \ddot{s}}{||B \ddot{s}||} - \hat{A} \hat{B} \dot{R} \right)
\] (51)
where
\[
\hat{N}^T = (I - J^T (J \hat{A}^{-1} J^T)^{-1} J \hat{A}^{-1})
\] (52)
and the relationship $\hat{N}^T \hat{A} \dot{B} = \hat{A} \dot{B}$ holds. Thus it follows that for $\dot{V}_s$:
\[
\dot{V}_s = -K_d \frac{s^T \ddot{s}}{||B \ddot{s}||} - \hat{A} \hat{B} \dot{R} - J^T f - \hat{A} \dot{B} \dot{R}
\] (53)
\[
\dot{V}_s \leq -K_d \lambda_{\min}(\hat{A}) \frac{1}{2} \frac{s^T \ddot{s}}{||B \ddot{s}||} - \hat{s} \frac{1}{2} \dot{A}^{-1} (\hat{A} \hat{B} \dot{R} - J^T f - \hat{A} \dot{B} \dot{R})
\] (54)

**Fig. 2.** Reaching trajectories for a) De Sapio posture controller (10) and b) sliding mode optimal controller. A line at $x = 0.3m$ has been included to aid visual comparison.

\[
\dot{V}_s \leq -K_d \lambda_{\min}(\hat{A}) \frac{1}{2} \frac{||s||}{||B \ddot{s}||} \left| |A^{-1}(\hat{A} \hat{B} \dot{R} - J^T f - \hat{A} \dot{B} \dot{R})| \right|
\] (55)

This leads also to a stability requirement for $K_d$:
\[
K_d > \frac{2 ||A^{-1}(\hat{A} \hat{B} \dot{R} - J^T f - \hat{A} \dot{B} \dot{R})||}{\lambda_{\min}(\hat{A})}
\] (56)

Note that $\dot{R}$ has to be chosen so that the right hand of (56) is minimal. In case $\hat{A} = A$ and $\dot{R} = \dot{R}$, the right hand of (56) is $|A^{-1} J^T f|$.

**IV. IMPLEMENTATION**

The control scheme was implemented in simulated and practical robot systems. An elbow joint limit was included via modification of the cost function as described in (10).

**A. Simulated Robot**

The effect of the sliding mode optimal posture controller (‘Soc’) on reaching trajectory may be observed in Figure 2. A vertical line has been included to aid visual comparison of the generated trajectories with and without the optimal sliding mode controller. It can be seen that the SOC controller produces a reduced workspace of the original controller given in (10), while maintaining a similar overall trajectory. This trajectory is shown in terms of effort optimisation in Figure 3 where the measure of effort is illustrated over the robot’s workspace. Here it can be seen that the SOC controller is certainly more optimal for the first half of the motion, before $X_0 = 0$.

In the simulation, the most beneficial gains for the SOC controller are $K = 100$ , $\delta = 0.8$ & $K_{slp} = 1000$. For the original controller (10) $K_p = 10$ and $K_d = 8$. The posture space (8) is defined by $K_{q1} = 80$ & $K_{q2} = 100$ with the joint limit is defined $q_{lim} = 90^\circ$, $K_{lim} = 2$, $\delta_L = 0.02$

**B. Physical Robot**

The controller was implemented on an Elumotion anthropomorphic robot arm via a dSpace system [1]. As in the simulated system, only the shoulder and elbow flexors were actuated. Figure 4 shows several trajectories projected onto the posture space of the robot system. Only a portion of the workspace has been illustrated for ease of visual comparison.
Fig. 3. Reaching trajectories from Figure 2 projected onto the Cartesian effort space. The effort space includes one joint limit on the elbow at 90°. An elbow limit has once again been applied at 90° with $K_L = 0.04$, $K_{lim} = 4$, $\delta_{lim} = 0.01$. It is worth noting that the physical robot system has a smaller workspace (particularly with joint limits imposed) than the simulated robot system and as such there is less variation in trajectories. The muscle matrix has been defined by $K_{a1} = 40$ & $K_{a2} = 40$. In Figure 4, areas outside the workspace have been represented by $U(q) < 0$, this includes areas where $q_2 > 120^\circ$, which exceeds the possible motion of the elbow. Note that the practical robot joint limit has a larger $K_{j1}$ value than that for the simulated system (Figure 3). This is necessary for the additional safety requirements of the physical system but leaves less effective workspace for trajectory formulation.

In Figure 4, two instances of the sliding mode optimal controller are shown alongside the original optimal controller from equation (10). For $SOC1$ gains are $K = 2$, $\delta = 0.4$ & $K_{slp} = 150$, while in $SOC2$ gains are $K = 4$, $\delta = 0.4$ & $K_{slp} = 250$. For the original the gains are $K_p = 0.7$ & $K_d = 8$.

The graphical results show that all 3 controllers achieve a trajectory that is close to the optimum. For the low gain, $SOC1$ bears a strong resemblance to the trajectory achieved by the original controller, with a large degree of hysteresis. For higher gains, the sliding optimal controller stays closer to the minimum area of the posture space, as illustrated by $SOC2$.

V. CONCLUSION

A novel control method has been proposed that introduces true steepest descent via sliding mode control to create an optimal posture controller. This controller has been implemented as the posture controller of an operational space control scheme that aims to synthesise human like motion for robotic application. Additional features of the controller preserve the decoupling effect of the posture controller from the task controller.

The controller has been implemented on simulated and practical 2dof planar robots that perform redundant reaching tasks in the vertical direction. In both cases convincing human like motion has been achieved while obeying additional smooth joint limits. Comparisons with a similar optimal controller (10) to that proposed by De Sapio et al [10] have shown the sliding optimal controller to be similar or superior in terms of optimal motion while having the additional benefit of being theoretically sound. Such theoretical confirmation of such a controller is essential for guaranteeing safety in human robot interaction.

REFERENCES