Hamiltonian Exploitation in Underactuated Robot System Inversion

Joel Stephen Short[1], Poo Aun Neow[2], Marcelo H Ang Jr[2], Lai Chow Yin[3], and Tao Pey Yuen[3]

***Abstract*— A new extension of a stable model inversion method is presented with the aim of providing solutions to the feedforward control of underactuated robots performing cyclic tasks. The method uses a boundary value problem framework along with Hamiltonian formalism, representing the dynamic equations of motion, to control the movement between non-equilibrium points in taskspace. The benefits of the method are verified with robot simulation results.**

I. INTRODUCTION

Underactuated robotic systems provide an intriguing problem with regards to control. The inherent nonlinearity present in articulated body systems(robots) is a main contributer to the difficulties in robotics control. Add to this situation a lack of control, due to either no actuation at certain joints or the limitation of torque input, and any effort to accurately control or predict the actions of the robot become exceedingly difficult. These problems are not insurmountable however, for whenever a proper model of the system dynamics can be created there is hope for predicting future behavior.

One of the most common methods of attempting to control, and therefore calculate and predict, the behavior of a robot is by feedforward control. In this methodology the dynamic equations of motion are built up as a set of ordinary differential equations relating the input torque to the joint positions, velocities, and accelerations(within the typical Lagrangian framework). If these equations can be inverted, the desired system behavior can be used to find the necessary torque input trajectories. This is only possible, in a straightforward manner, with linear systems and applying it to nonlinear systems has led to a host of possible linearization techniques and approximation methods. This is where the problem moves from the stable system inversion of robot dynamics and branches into the larger field of stable nonlinear system inversion. See [1] and [2] for references of this wider perspective.

As an important comparison, there are nonlinear robot control methodologies that do not work with an explicit system inversion. One of these areas uses a cost function approach that works to find the overall control system behavior[3]. While others have turned to recent advances in switching methods[4]. Reliable discrete-time control methodologies have also yielded notable results[5].

Though many routes exist in this field, the work of Graichen et al.[6] is of special interest. The method of solving the stable inversion setup was accomplished by using a boundary value problem(BVP) framework where the desired initial and final robot states form the boundary conditions(BCs) and a proper input torque curve can be found. This method for deriving the feedforward control for an underactuated robot comes with its own difficulties, but has been well demonstrated[7] and shows promise for greater expansion and application.

The main limitation of the BVP stable inversion method developed by Graichen et al. is that it can only move an underactuated robot system from one equilibrium point to another. This is problematic becuase an underactuated robot may be required to move to, or between, points in the taskspace that are not equilibrium positions. A cyclic(repeating) pick and place task performed by an underactuated robot, is one example with another seen in a winch based robot[8]. This type of task restricts not only the robot position trajectories but also the input behavior, such that there must be continuity between segments of the task. Fig. 1 shows these restrictions. It is the aim of this paper to present a BVP stable inversion method capable of finding the necessary torque input, for use on underactuated robots, that allows movement between momentary-stop setpoints.

The methodology first discusses the Hamiltonian form with a short comparison to the Lagrangian setup. This leads to the integration of the Hamilton form into the BVP framework such that the feedforward control solutions can be derived via a numerical solver. Lastly, the method is demonstrated with a simulated 2DOF planar robot.

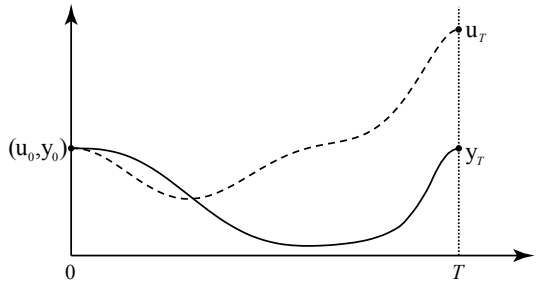

Fig. 1. A transition segment showing possible trajectories between setpoints that must satisfy the output and input boundary conditions at $t = 0$ and $t = T$

This work was supported by the Science and Engineering Research Council, Agency for Science, Technology and Research (A*STAR) under Grant U12-R-051SU / SIMT / 12-410025.

[1]Joel Stephen Short studies at the National University of Singapore A0089327@nus.edu.sg

[2]Poo Aun Neow and Marcelo H Ang Jr are with the Department of Mechanical Engineering, National University of Singapore, 9 Engineering Dr. 1, Singapore 117576 mpepooan@nus.edu.sg; mpeangh@nus.edu.sg;

[3]Lai Chow Yin and Tao Pey Yuen are with the Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research, Singapore 638075 cylai@SIMTech.a-star.edu.sg; pytao@SIMTech.a-start.edu.sg

II. METHOD

The general framework of this method is based around finding the solution to the stable inversion of an underactuated robot's system dynamics, which can then be used to generate the input trajectory for the feedforward control. This goal is accomplished using the BVP setup using the dynamic equations of motion in the Hamiltonian form. While the possibility for wider generalizations is part of ongoing research, the work presented is restricted to application on SISO underactuated robot systems.

A. Hamiltonian Formalism

When the dynamic behavior of a robotic system needs to be mathematically described the most common method is to use the Lagrangian framework. The Lagrangian function(1) is defined as the kinetic energy T of the system minus the potential energy V. The equations of motions are then derived with (2), which is equal to zero for a closed system, or equal to the torque/force input for typical robot systems.

$$L = T - V \tag{1}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \tau_i \tag{2}$$

The equations of motion are defined based on a set of position variables q_i, normally joint angles for a robot, that expands to include the velocity $\dot{q}_i$ and acceleration $\ddot{q}_i$. There are n second order equations of motion ($n =$ system order) that through various steps of manipulation can be divided into sections, one representing the inertia behavior, another the centrifugal and Coriolis affects, with subsequent matrices representing gravity, frictional and other effects. This is known as the joint-space canonical form.

The Hamiltonian formulation is simply a different way of expressing the dynamic behavior of the system. The Hamiltonian formulation uses only two types of variables: the generalized position q_i and the generalized momenta p_i. The Hamiltonian(3) is defined as the sum of the total energy in the system (kinetic plus potential).

$$H = T + V \tag{3}$$

$$H(q_i, p_i) = \sum_{i=1}^{n} p_i \dot{q}_i - L(q_i, \dot{q}_i) \tag{4}$$

$$p_i = \frac{\partial L}{\partial \dot{q}_i} \tag{5}$$

It is sometimes possible to determine the equations of motion directly from (3) but a more versatile route is in (4), which uses a Legendre transform of the Lagrangian along with (5) to define the general momenta p_i. The equations of motion are then derived from H with

$$\dot{p}_i = -\frac{\partial H}{\partial q_i} + \tau_i \tag{6}$$

$$\dot{q}_i = \frac{\partial H}{\partial p_i} \tag{7}$$

Where the input to the robotic system, normally torque or force, is included as τ_i.

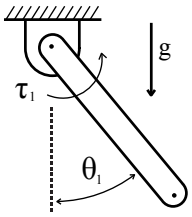

Fig. 2. Single link robot example affected by gravity

The Hamiltonian form renders $2n$ first order equations of motion. As an example, see the 1-link vertical robot, with a single actuator, in Fig. 2. This system is second order with 1 degree of freedom, q_1 is the joint position with p_1 as the angular momentum. The Hamilton equations of motion would consist of

$$\dot{p}_1 = f_1(p_1, q_1) + \tau_1 \tag{8}$$

$$\dot{q}_1 = f_2(p_1, q_1) \tag{9}$$

Using the Hamiltonian with a robotic system, generally the input will always be the force or torque while the output is the general momentum of the corresponding variable. Referring back to (6), the input and output variables can be chosen as

$$y = p_1 \qquad u = \tau_1 \tag{10}$$

This may seem odd but will be justified in section D when the *normal form* is discussed.

The downsides of the Hamiltonian are mainly due to its unfamiliarity and obscurity. Working with a bulky set of first order differential equations can be nonintuitive though with the proper canonical transforms nearly any quantity, variable or aspect of a system is available for direct study and exploitation. Initial appraisal of the system dynamics can be more difficult compared to the Lagrangian, where the equations have a known method of organization, but the more balanced approach in the Hamiltonian system variables can give a wider perspective.

B. Cyclic Coordinates

A cyclic coordinate is a variable that is absent from the Hamiltonian function(and thus the Lagrangian). This means that any change in the coordinate does not change the energy level in the system. These special coordinates are most easily exploited in the Hamiltonian form. For dynamic systems the cyclic coordinates always correspond to a conserved momentum, within the robot dynamics this is typically angular momentum. Referring back to the example of the single link robot, if the base is shifted so that the links motion is planar to the horizon, the coordinate q_1 becomes cyclic(no matter the value of q_1, H will not change) and the first equation of motion becomes $\dot{p}_1 = \tau_1$. If the example system was closed, with no input, $p_1 = \alpha_1$, where α_1 is a constant dependent on the initial energy of the system. Hamiltonian formalism is most common in classical dynamics [9] where the application fields range far wider than robotics.

C. Boundary Problem Formulation

The Hamiltonian form provides an excellent setup when using the BVP framework. The dynamic equations of motion can be represented in general using the Hamiltonian equations of motion,

$$\dot{p}_i = f_1(p_1,\ldots,p_d,q_1,\ldots,q_d) + \tau_i \tag{11}$$
$$\dot{q}_i = f_2(p_1,\ldots,p_d,q_1,\ldots,q_d) \tag{12}$$

Where $i = 1,\ldots,d$ and $d =$ degrees of freedom in the system and τ is the input force or torque. It is important to note that the robot system in focus is underactuated, the torque factor will only exist in the equation that corresponds to the actuated joint. Working with underactuated robot of the SISO type, the input and output are

$$y = p_j \tag{13}$$
$$u = \tau_j \tag{14}$$

with $j =$ actuated degree of freedom.

The problem at hand focuses on the movement between two non-equilibrium(or momentary-stop) setpoints, most commonly seen in a cyclic task. The robot will have two desired setpoints(in taskspace) for the output as a starting example. Though movement between an equilbrium point and a momentary-stop is also covered by the method. Considering each segment of the task individually allows for a simpler formulation. For the first half of the task, moving from the first point(time$= 0$) to the second(time$= T$), the BCs based on the state variables can be written as

$$x(0) = x_0 \quad x(T) = x_T \tag{15}$$

where $x = \{p_1,\ldots,p_i,q_1,\ldots,q_i\}^T$ using the Hamiltonian form. With these basic BCs the stable system inversion has been formulated as a two-point BVP.

Theoretical work[10] states that in general there exists some equation representing the output y with n degrees of freedom(where $n =$ order of the system) that satisfies(15). The output will be approximated by a polynomial in this work, though another type of equation, such as a smooth spline or even a cosine series, may be used. The base equation for the output will be

$$y^*(t) = \sum_{i=0}^{n-1} a_i t^i \tag{16}$$

where $t =$time. Since a polynomial is used, n free parameters can be generated with an equation of order $(n-1)$. In order to include additional BCs, beyond the initial ones seen in(15), the output equation must be expanded an additional free parameter for each new BC.

A key point with this method is that due to the continuity requirements of the cyclic task there also exist restrictions on the input defined with

$$u(0) = \tau_0 \qquad u(T) = \tau_T \tag{17}$$
$$\dot{u}(0) = \dot{\tau}_0 = 0 \quad \dot{u}(T) = \dot{\tau}_T = 0 \tag{18}$$

using (14) to relate them directly to the Hamilton setup. The main hurdle to cross now is to analyze the restrictions put on the input trajectory and relate them through to the output BCs. This is necessary because the BVP formulation requires that the BCs can only contain state variables p_i, q_i, and the unknown parameters a_i, that are built into the output equation(16)

Relating the input restrictions to the output BCs is possible due to the formulation of the Hamiltonian equations of motion. The output approximation is a polynomial, a function easily differentiated. If the the output trajectory seen in (16) is equal to the general momenta, shown in (13), the input restrictions in (17) can be related to the output polynomial by using (11) and the derivative of the output polynomial. The use of the state BCs with the desired starting and ending torque provides BCs

$$\dot{p}_j(0) = f_g(x_0) + \tau_0 \tag{19}$$
$$\dot{p}_j(T) = f_g(x_T) + \tau_T \tag{20}$$

for $\dot{p}_j$ where f_g is a function of position and normally due to gravity. Additional BCs are thus generated using the derivative of the polynomial output.

Transfering the restrictions seen in (18) is a much more difficult problem due to the nonlinear function $f(x)$ that comes from the partial derivative in (6). The related equations show the double derivative of the general momentum

$$\ddot{p}_j(0) = \dot{f}_g(x_0) + \dot{\tau}_0 \tag{21}$$
$$\ddot{p}_j(T) = \dot{f}_g(x_T) + \dot{\tau}_T \tag{22}$$

The following cases represent known conditions where the input restrictions can be transfered through to the double derivative of the output.

Case 1) If the input joint(q_j) is a cyclic coordinate.
Then $-\frac{\partial H}{\partial q_i}$ resolves to zero and $\dot{p}_j = \tau_j$

Case 2) If $f_g(x) = f_g(q_i)$ and $\dot{p}_i|_{0,T} = 0$, for all i
Then $\dot{q}_i|_{0,T} = 0$, for all i and $\ddot{p}_j|_{0,T} = \dot{\tau}_j$

The validity of case 1 should be self evident from (6) while case 2 needs additional reasoning. If the general momenta start and end at zero, so must the joint velocities, where the nonlinear function $y = f_g(q_i)$ is a function of a set of variables q_i and using the total derivative

$$\frac{dy}{dt} = \frac{\partial y}{\partial q_1}\cdot\frac{dq_1}{dt} + \cdots + \frac{\partial y}{\partial q_i}\cdot\frac{dq_i}{dt} \tag{23}$$

If for all i, $\dot{q}_i|_{0,T} = 0$ then $\dot{y}|_{0,T} = 0$ because each multiplication pair contains a zero. Case 2 can occur when p_j is only affected by gravity and $f(x)$ is then only a function of position. Both cases allow (18) to be set to any value but it is typical to restrict them to zero.

Now that it is possible to implement the 4 input restrictions as valid BCs an adjustment to the output polynomial is needed. Instead of only requiring an equation with n free parameters, the polynomial must now have $(n+4)$. The output equation

$$y^*(t) = \sum_{i=0}^{n+3} a_i t^i \tag{24}$$

is a polynomial so will only require an order of $(n+3)$ to achieve $(n+4)$ free parameters. A point of detail to note here is that there are $(2n+4)$ total BCs, while the output polynomial only requires $(n+4)$ free parameters. Expanding the BCs by one adds one more BC, while the base restrictions on the states only require a single free parameter for each 2 BCs.

Next, the output polynomial must be integrated into the dynamic system of equations such that the free parameters used in the solver and given in the solution. With the free parameters for the output polynomial found, the required torque input can be derived due to the relationship seen in (11) and (13) where $i = j$. The best route is to substitute the derivative of the output polynomial $\dot{y}^*$ in the place of the normal $\dot{p}_j$.

With the a set of differential equations representing the system dynamics in Hamiltonian form, the framework for building BCs ready, and the output polynomial substituted, the BVP formulation is now ready for the numerical solver.

D. Normal Form

Previous work in this area is based on using the BVP framework within a standard set of input-output coordinates[6] known as *normal form*. The main aim of this system setup is to restrict the input to a single first order equation while creating a chain of integrators that connect the output to the input. The *relative degree r* is used in the definition and is exactly equal to the number of times the output $y(t)$ must be differentiated at time $t = t^\circ$[11] such that the input u appears in the equation. Though the system is nonlinear and the definition holds for some time t°, a quick examination of a dynamic equations of motion will reveal whether the relative degree will change depending on the state of the system. The input-output coordinates normally require a transformation, via a diffeomorphism, that reworks the differential equations into the following standard setup.

$$y = z_1 \tag{25}$$
$$\dot{z}_1 = z_2 \tag{26}$$
$$\cdots$$
$$\dot{z}_{r-1} = z_r \tag{27}$$
$$\dot{z}_r = b(z) + a(z)u \tag{28}$$
$$\dot{z}_{r+1} = q_{r+1}(z) \tag{29}$$
$$\cdots$$
$$\dot{z}_n = q_n(z) \tag{30}$$

The output of the system is always chosen as the topmost variable z_1, (25) . Once in normal form, the BVP can begin to be set up by moving the BCs into the *normal form*. A quick comparison between the Hamiltonian setup described in (11), (12),and (13) with the *normal form* (28), (29), (25) shows that the Hamiltonian meets the same layout. The SISO Hamiltonian has a relative degree $r = 1$ and due to the straightforward formulation it does not matter which joint of the robot is actuated, the layout will still satisfy the *normal form* requirements.

E. Numerical BVP Solver

Once the formulation is complete a solver must be used to numerically find the solution to the BVP. This can be done by with a standard Runge-Kutta numerical solver and a custom setup to include the use of the free parameters, yet the most convenient solver available is MATLAB's solver bvp4c. This particular solver is pre-built to solve BVP's with the use of free parameters so that once the notation is understood, aided by MATLAB's tutorials, the problem can be directly solved.

It requires three main pieces of information. First, the dynamic equations of motion described as first order differential equations. Second, the BCs, entered in a residual format where the restriction $p_1(0) = 3$ would be entered as $p_1(0) - 3$. The number of BCs, or residuals, that must be specified is equal to the number of free parameters$(n+4)$ added to the number of differential equations that define the system(n), so in effect it always requires$(2n+4)$ BCs. Lastly, an initial guess is needed, both for the output polynomial parameters as well as the likely path of the states during the task. The polynomial parameters can be started at zero, but if the solver fails by hitting a singularity at the Jacobian, this is a good place to finesse the process. The likely paths of the states can be estimated by simply creating a path between the initial and final states.

III. EXAMPLE DEMONSTRATION

Consider the 2DOF, underactuated planar robot seen in Fig. 3. The first joint is actuated while the second joint is passive but contains a radial torsion spring. The passive element within the joint is chosen based on the desired task. The dynamic system has 2 degrees of freedom and the order of the system is $n = 4$. The physical parameters of the simulated robot are seen in Table I while the equations of motions are represented in the Hamiltonian form as

$$\dot{p}_1 = \tau_1 \tag{31}$$
$$\dot{q}_1 = f_1(p_1, p_2, q_2) \tag{32}$$
$$\dot{p}_2 = f_2(p_1, p_2, q_2) - k(q_2 - \text{Offset}) \tag{33}$$
$$\dot{q}_2 = f_3(p_1, p_2, q_2) \tag{34}$$

For further details on the derivation of the dynamic equations of motion, please see the appendix. The functions seen in (32), (33)(which includes the torque generated by the torsion spring), and (34) are nonlinear and derived with the help of a symbolic mathmatics toolbox in MATLAB. The partial derivative portion of (31) reduces to zero since the Hamiltonian equation does not contain the cyclic coordinate q_1. The input and output of the system are defined, respectively, as

$$y = p_1 \quad u = \tau_1 \tag{35}$$

The boundary value problem can then be formulated using the momentary-stop setpoints that the robot needs to move between. For this example the BCs at the state variable level (15) start by restricting the joints to move according to

$$q_1(0) = 0.2 \quad q_1(T) = -0.4 \tag{36}$$
$$q_2(0) = 0.3 \quad q_2(T) = -0.5 \tag{37}$$

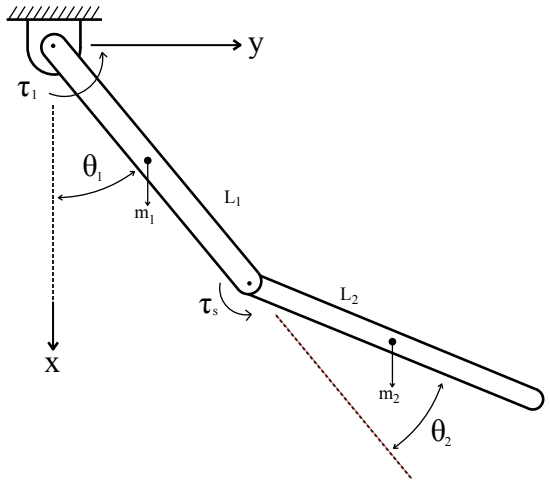

Fig. 3. Demonstration underactuated robot

along with the requirements placed on the general momenta behavior

$$p_1(0) = 0 \quad p_1(T) = 0 \tag{38}$$

$$p_2(0) = 0 \quad p_2(T) = 0 \tag{39}$$

This will ensure that the robot moves from the initial to the final setpoint with an initial and final velocity of zero. The 8 BCs are only based on the system states so the initial output polynomial starts with only 4 free parameters.

Next, the input restrictions must be described, using (17) and (18), in the BC form as defined by the state variables and free parameters. In this setup, the robot in use satisfies Case 1) from the qualifications discussed in the BVP formulation section, allowing the following restrictions

$$\dot{p}_1(0) = \tau_1(0) = -13.16 \quad \dot{p}_1(T) = \tau_1(T) = 11.75 \tag{40}$$

$$\ddot{p}_1(0) = \dot{\tau}_1(0) = 0 \qquad \ddot{p}_1(T) = \dot{\tau}_1(T) = 0 \tag{41}$$

The values in (40) are determined based on the mechanical properties of the robot, where the optimal initial and final torques are used to minimize acceleration at the robot endpoint.

The combination of the state BCs and those derived from the input restrictions are used in the BVP formulation along with the following output polynomial, from (24),

$$y^*(t) = \sum_{i=0}^{4+3} a_i t^i \tag{42}$$

The polynomial used in the BVP example is as follows

$$y^*(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 + a_6 t^6 + a_7 t^7 \tag{43}$$

TABLE I

DEMONSTRATION ROBOT PHYSICAL PROPERTIES

Robot		Passive Joint	
Variable	Value	Variable	Value
L_1	0.341 m	k	6.3 Nm/rad
L_2	0.425 m	Offset	-0.089 rad
m_1	1.278 kg		
m_2	0.752 kg		

TABLE II

RESIDUALS USED IN BVP4C AND OUTPUT SOLUTION

State Based BCs	Input Based BCs	Parameters	Solutions
$p_1(0)$	$\dot{y}^*(0) + 13.16$	a_0	0
$p_1(T)$	$\dot{y}^*(T) - 11.75$	a_1	-10.589
$p_2(0)$	$\ddot{y}^*(0)$	a_2	0
$p_2(T)$	$\ddot{y}^*(T)$	a_3	245.400
$q_1(0) - 0.2$		a_4	-108.964
$q_1(T) + 0.4$		a_5	-5191.260
$q_2(0) - 0.3$		a_6	19301.845
$q_2(T) + 0.5$		a_7	-20821.743

Within this example the torque input is equal to the derivative of p_1, as stated in (35). So in order to integrate the output polynomial into the ordinary differential equations that represent the system dynamics, the polynomial is differentiated once and used to represent the torque input(31)

$$\tau_1 = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4 + 6a_6 t^5 + 7a_7 t^6 \tag{44}$$

It may seem that a_0 is lost as a useful free parameter but it is automatically set equal to zero due to the BC's, ensuring that $p_1(0) = 0$.

Lastly, the transition time, $T = 0.35$ seconds, for the task is chosen with care based on the dynamic capabilities of the robot and the desired speed. Further research into a more generalized method of choosing this transition time is currently underway. This completes the BVP formulation leading to the numerical solver. In order to help describe the process, the actual list of residuals(BCs) used in the MATLAB program is shown in Table II along with the parameter solutions given by the solver. With these parameters the proper input torque trajectory can be found directly using (44) while problems falling under case 2) require using the state variables in the solution to compute the input torque from $\dot{y}$.

The output of the numerical solver is seen in Fig. 4 with the double derivative of both general momenta variables seen in Fig. 5, showing that they both start and end at zero. As can be noted in the graphs, all of the BCs were met, the robot was able to start and end at zero velocity, while the initial and final momenta for both joints was zero. The initial and final positions match the desired. More importantly the initial and final torque, and torque derivatives, followed the prescribed restrictions.

This method has produced a solution that satisfies all the BCs and can be used in each segment of a cyclic task feedforward control problem. In this example another segment, where the robot moves back to the initial starting point, is needed but the method is the same. The initial starting or final stopping motion, from/to a point of rest, can also be calculated using this method.

IV. CONCLUSIONS

This paper has endeavored to present a new extension to the stable inversion method based on the BVP formulation that allows movement between two non-equilibrium points. The benefits and general use of the Hamiltonian form was presented with special attention to its application in the

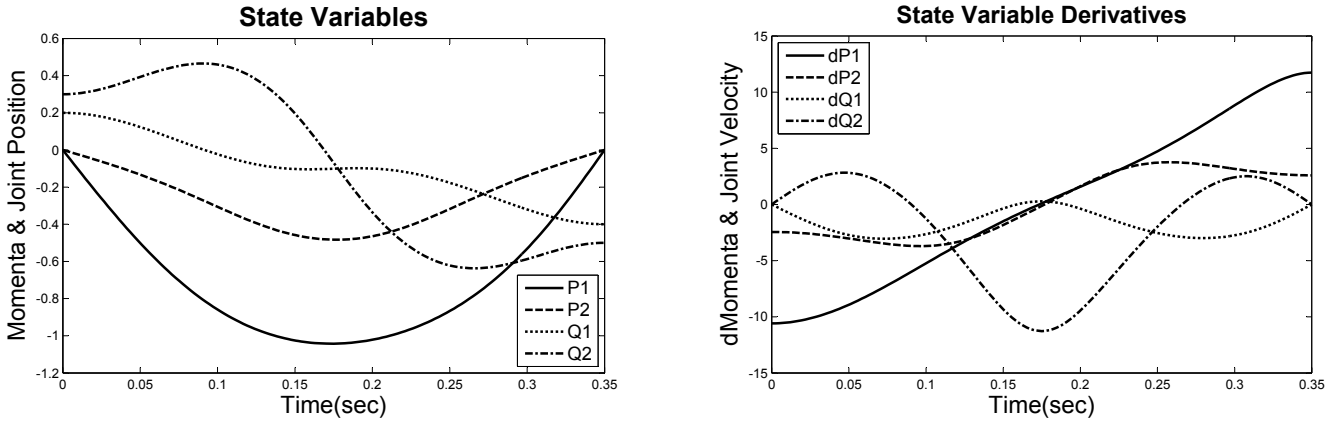

Fig. 4. Continuous state trajectories found by bvp4c using BVP

BVP setup. The main application of the method is in underactuated robots, with a special emphasis on cyclic tasks involving momentary-stop setpoints. These include pick-and-place movement and simple assembly tasks with a high degree of repetition.

Future work will include the search for a more generalized approach that allows the method to be applied on a wider class of robots in addition to the two special cases shown here. Work is also in progress to verify the method with experiments on an actual underactuated robot.

APPENDIX

Derivation of the Hamiltonian equations of motion is most easily done by starting with the traditional Lagrangian equations of motion. The derivation of the Lagrangian for a two link robot is not trivial but due to space constraints will be skipped, see [12] page 98 for a complete treatment. The Lagrangian for the robot seen in Fig. 3 is

$$L = \{\dot{q}_1^2(2L_1^2m_1 + 6L_1^2m_2 + 2L_2^2m_2 + 6L_1L_2m_2\cos(q_2)) + L_2m_2(\dot{q}_1\dot{q}_2(4L_2 + 6L_1\cos(q_2)) + \dot{q}_2^2(2L_2))\}/12 - 0.5k(q_2 - \text{Offset})^2 \quad (45)$$

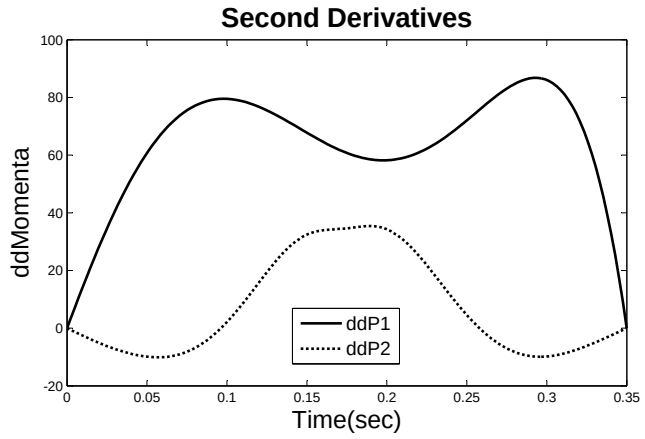

Fig. 5. The double derivative of the general momenta variables

The potential energy in the spring carries through easily because it is only dependent on position and not velocity. The Lagrangian is then used in Legendre transform (4) along with general momenta definition (5) to find the Hamiltonian

$$H = \{p_1^2(6L_2^2m_2) + p_1p_2(-12L_2^2m_2 + 18L_1L_2m_2\cos(q_2)) + p_2^2(18L_1^2m_2 + 6L_2^2m_2 + 6L_1^2m_1 + 18L_1L_2m_2\cos(q_2))\}W^{-1} - 0.5k(q_2 - \text{Offset})^2$$
$$\text{where } W = L_1^2L_2^2m_2(4m_1 - 9m_2\cos(q_2)^2 + 12m_2) \quad (46)$$

Lastly, the dynamics equations of motion are derived from the Hamiltonian using (6) and (7). In this robot configuration th first position coordinate q_1 is cyclic and does not appear in the Lagrangian or the Hamiltonian functions.

REFERENCES

[1] D. Chen and B. Paden, "Stable inversion of nonlinear non-minimum phase systems," *International Journal of Control*, vol. 64, no. 1, pp. 81–97, 1996.

[2] L. R. Hunt and G. Meyer, "Stable inversion for nonlinear systems," *Automatica*, vol. 33, no. 8, pp. 1549–1554, 1997.

[3] J. Nakanishi, K. Rawlik, and S. Vijayakumar, "Stiffness and temporal optimization in periodic movements: An optimal control approach," in *IEEE International Conference on Intelligent Robots and Systems*, 2011, pp. 718–724.

[4] K. Ichida, K. Watanabe, and K. Izumi, "Control of three-link under-actuated manipulators by a logic-based switching method," in *Proceedings of the 2009 IEEE International Conference on Networking, Sensing and Control, ICNSC 2009*, 2009, pp. 108–112.

[5] N. Scherm and B. Heimann, "Dynamics and control of underactuated manipulation systems: A discrete-time approach," *Robotics and Autonomous Systems*, vol. 30, no. 3, pp. 237–248, 2000.

[6] K. Graichen, V. Hagenmeyer, and M. Zeitz, "A new approach to inversion-based feedforward control design for nonlinear systems," *Automatica*, vol. 41, no. 12, pp. 2033–2041, 2005.

[7] K. Graichen, M. Treuer, and M. Zeitz, "Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation," *Automatica*, vol. 43, no. 1, pp. 63–71, 2007.

[8] D. Cunningham and H. H. Asada, "The winch-bot: A cable-suspended, under-actuated robot utilizing parametric self-excitation," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2009, pp. 1844–1850.

[9] H. Goldstein, *Classical Mechanics*. Addison-Wesley, MA, 1965.

[10] H. B. Keller, *Numerical Methods for Two Point Boundary Value Problems*. Blaisdell, 1968.

[11] A. Isidori, *Nonlinear Control Systems*. Springer, 1985.

[12] L. C. Fu K.S., Gonzalez R.C., *Robotics: Control, Sensing, Vision and Intelligence*. McGraw-Hill NY, 1987.