# Energy Efficient Semi-Automatic Control of Loader Cranes using Reinforcement Learning

Amy Rankka[1,2] and Alessandro Dell'Amico[1]

[1]Division of Fluid and Mechatronic Systems, Linköping University, Linköping, Sweden
[2]Hiab AB, Hudiksvall, Sweden
[1]E-mail: amy.rankka@liu.se, alessandro.dellamico@liu.se

## Abstract

Operator assistance functions for working machines such as loader cranes are often developed with the aim of increasing the productivity of an unskilled operator or of increasing the safety of operating the machine. For a kinematically redundant system, such as the boom system of a loader crane where three joints are used to control a motion in a two dimensional plane, the operator assistance functions can also be developed with an energy efficiency objective. In this paper, the semi-automatic function Crane Tip Control (CTC) for loader cranes is considered.
The energy required to perform a CTC motion depends on the pressure levels and flow demand of each active cylinder and how this is translated into pump pressure and flow. Previous research has shown that the current CTC strategy is not optimal with regards to energy efficiency. In this work, CTC controllers for limited working areas are developed where the strategy is decided by a neural network trained by a Reinforcement Learning approach with an energy minimizing objective on a simulation model of the crane.
A dynamic programming (DP) algorithm that can find near-optimal solutions for given trajectories offline has been developed in previous research. The energy performance of the new controllers with neural networks, NN controllers for short, are benchmarked on a number of test cases by comparing the solutions to those of the DP algorithm, denoted DP controllers. The results show that energy consumption of the new controllers are only up to 2.2 % higher than the consumption of the DP controllers. The methodology is thus promising, but future work is required to expand it to cover the complete working envelope of the CTC function.
**Keywords:** loader cranes, energy efficiency, semi-autonomous operation, neural network controller, redundancy resolution

## 1 Introduction

This paper is part of a research project aiming at reducing the energy consumption of loader cranes through automation. The loader crane of consideration here is a kinematically redundant system where in order to follow a given trajectory in its three dimensional working area there can be an infinite amount of solutions to how to actuate the different cylinders. The redundancy makes it possible to optimize the crane movements with regards to different objectives. As a first step of the project, the semi-automatic function Crane Tip Control (CTC) is investigated. With CTC, the operator controls the movement of the crane tip in a two dimensional plane with one lever for horizontal and one for vertical movements.

In a previous study [1], a dynamic programming (DP) offline optimization was developed with the objective of minimizing the energy consumption of CTC operation on given trajectories. The results show that the energy consumption of the crane on a specified drive cycle can be significantly reduced by optimizing the CTC strategy for energy efficiency. The results also show that for some trajectories, the energy optimized CTC outperforms manual operation, increasing the incentive of investigating semi-automatic, or fully automatic, control as a general solution for reducing the energy consumption of crane operation.

The difference in energy consumption between different strategies of moving the cylinders of the crane are mainly dye to three properties of the system; the total flow that needs to be supplied to the cylinders (even though the crane tip distance is the same, the total traveled distance of the cylinders can differ), the load setting pressure (due to the geometry, cylinder dimensions and friction, different cylinder require different pressure levels for lifting the payload in different crane poses) and the losses from simultaneous operation (in a load sensing system with one pump, the flow to all cylinders active

simultaneously is supplied at the same pressure even though the cylinder pressures might differ).

### 1.1 Problem Formulation

For a CTC controller to be of practical use it must be able to calculate a solution to any given position and speed reference in real-time. The aim of this study is to investigate whether a neural network controller trained with reinforcement learning could be part of a good solution for a real-time CTC.

Previously, a neural network CTC controller has successfully been developed with the objective of moving the crane tip as fast as possible, see [2]. The controller has been evaluated on a real crane and can operate within the limit of the normal sample rate of the embedded control system.

### 1.2 System Description

This study is applied to a loader crane, see fig. (1) for a schematic view, with three hydraulic cylinders used for moving the crane tip in the two-dimensional plane with axes $x$ and $y$; the first boom, "1$b$", cylinder, the second boom, "2$b$", cylinder and the extension, "$ext$", cylinder. The crane also has a cylinder for rotating around the base, but it is not considered here due to that CTC only operates in the plane. The first and second boom cylinders control a revolute joints with angles denoted $\alpha_{1b}$ and $\alpha_{2b}$ respectively. The extension cylinder works as a prismatic joint with a position denoted $l_{ext}$. The position of the three joints together with the dimensions of the crane defines the crane pose. The crane tip position is given in Cartesian coordinates in the plane ($[x_{tip}$,$y_{tip}]$).

The cylinders of the crane are controlled by a pressure compensated load sensing hydraulic system connected to a variable displacement pump. The speed commands to each cylinder, denoted $u_{1b}$, $u_{2b}$ and $u_{ext}$, can be directly translated to spool position commands to the directional valve.

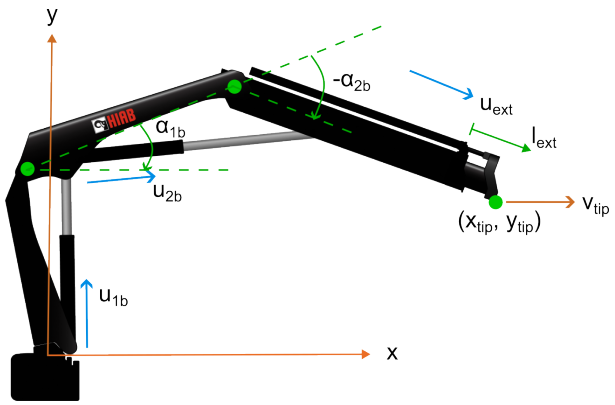


*Figure 1: Crane with cylinders, joints and crane tip position.*

### 1.3 Limitations

For a complete CTC solution, it should be possible to control the motion of the crane within its complete working area (which is limited by the stroke of the cylinders) in real-time. The controller should be able to take in any reference crane tip velocity between its limits, and provide a solution for any given state of the crane in real-time for every payload up to the limit of how much the crane can lift. The state space to sample training starting points will thus need to cover all the crane tip positions of the working area 2D plane, the range of cylinder positions that result in these crane tip positions, the range of velocities and the range of payloads.

In this study, which should be seen as an initial proof of concept, the scope is limited to cover one payload, one velocity and one trajectory at a time.

### 1.4 Prior Art

Solutions that enables control of the end effector in Cartesian coordinates are offered by different manufacturers for their machines. For this study, the Hiab CTC feature for loader cranes, [3], is used as a reference. The strategy for this CTC controller is to maximize the lifting capacity rather than minimizing energy consumption.

A common way of solving the inverse kinematics of the redundant crane boom system is the pseudo-inverse jacobian method, described for example in [4]. The drawback with this approach is that only locally optimal solutions can be found, and in [5] it is shown that the performance is significantly worse than for a dynamic programming algorithm that can find a near-optimal solution offline.

In [6], a real-time controller with an energy minimization objective for a three degree of freedom robotic arm is developed that performs pretty close to the global optimal solutions on trajectories with a length of 10 s. The method is called Predictive Minimization of Kinetic Energy. The motion of the arm is controlled by two tasks, precision tracking by the pseudo-inverse solution and minimization of the kinetic energy by a velocity vector pointing in the direction where the kinetic energy integral is minimized for the chosen trajectory. The optimal control is predicted for a window and continuously updated as the simulation is rolled out. To only minimize the kinetic energy is too much a simplification for the crane with a load sensing hydraulic system, where a significant part of the losses comes simultaneous operation of cylinder with different pressure levels.

As already mentioned, a neural network CTC controller with the objective of moving the crane tip as fast as possible has been developed by reinforcement learning in [2]. The method developed in that work has been applied to the crane used as application in this study with good results. The initial idea was to complement the training of that network by adding an energy minimization term to the objective function. However, the result is not really the desired outcome. When comparing controllers trained with that objective function with different weights on the energy consumption, it became clear that the more the controller cared about the energy consumption, the slower it became.

The issue of lower energy consumption correlating to lower speed can also be in the results of the work in [7], where automatic log collecting of a forwarder is controlled by a neural network trained with reinforcement learning. Compared to a policy trained without an energy objective, the policies trained

with an energy objective show 61 to 78 % decrease in energy consumption and 11 to 36 % increase of cycle time.

In order to avoid this balancing between energy consumption and productivity, in this paper the network is only trained to provide the strategy for CTC, the velocity is controlled by a separate controller.

## 2 Controller Development

In this work, an energy efficient CTC controller is developed in two steps. First, a velocity controller is implemented that calculates the reference velocities for the three boom cylinders required to follow a certain reference crane tip velocity for a given strategy. Then, a neural network is set up and trained as a strategy controller using reinforcement learning in a simulation environment. When later used in comparison with the solutions from the dynamic programming algorithm, a complete new CTC controller (strategy and velocity controllers combined) will be denoted "NN controller" for short, and the solutions from the DP algorithm "DP controller".

### 2.1 Controller Structure

The CTC controller is divided into two parts, a neural network that decides the strategy, denoted "CTC strategy controller", and a conventional feed forward controller that handles the velocity reference following given a certain strategy, denoted "CTC velocity controller".

For a solution deployed on a real crane, the reference velocity is given by the operator. Due to the geometry of the crane, the maximum velocity that the crane tip can achieve varies varies with the crane pose. In order not to command the controller to perform an impossible movement, the reference velocity is limited before it is set as input to the velocity controller. The highest possible crane tip velocity for the current state can be calculated according to eq. (1), where $J$ is the jacobian matrix of the crane tip position evaluated at the current crane pose.

$$v_{tip,max} = J(\alpha_{1b}, \alpha_{2b}, l_{ext}) * \begin{pmatrix} \dot{\alpha}_{1bmax} \\ \dot{\alpha}_{2bmax} \\ \dot{l}_{extmax} \end{pmatrix} \tag{1}$$

With the reference velocity given, the strategy controller only needs to decide how one of the cylinders should be operated in order for the resulting movement to be as good as possible with regards to some objective. The operation of the other two cylinders is then uniquely determined by the reference velocity. In this work, the strategy determines the reference to the extension cylinder with the objective of minimizing the energy consumption.

The network for the strategy controller, which is displayed in a schematic view in fig. (2), has an input layer with LeakyReLU activation, two hidden layers also with LeakyRelu activation and an output layer with hardtanh activation in order to get the output between [-1,1], where -1 corresponds to maximum negative speed command, $u_{ext}$, to the cylinder and 1 to maximum positive speed command.

LeakyReLU stands for leaky rectified linear unitand introduces non-linearity to the network with reduced risk of deactivating neurons compared to simple ReLU, it is defined according to eq. (2), where $alpha$ is set to 0.01. HardTanh stands for hard hyperbolic tangent, is faster to compute than Tanh since it does not have exponentionas, and is defined according to eq. (3).

$$LeakyReLU(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha x, & \text{if } x \leq 0 \end{cases} \tag{2}$$

$$HardTanh(x) = \begin{cases} 1, & \text{if } x > 1 \\ x, & \text{if } -1 \leq x \leq 1 \\ -1, & \text{if } x < -1 \end{cases} \tag{3}$$

The hidden layers have 16 nodes each. The inputs to the network are the positions of the different joints, $\alpha_{1b}$, $\alpha_{2b}$, and $l_{ext}$ (see fig. 1), and the limited velocity reference, $v_{tip,ref}$. The inputs are normalized.

The velocity controller is displayed in fig. 3. It takes in the current state of the crane, and together with the velocity reference and the extension speed command it calculates the 1b and 2b positions required to reach the reference tip position with the selected strategy. The difference between the current and the next state then sets the speed commands for the 1b and 2b cylinders, $u_{1b}$ and $u_{2b}$. If no valid combination of cylinder positions can be found, the cylinder speed commands of all three cylinders are set to zero. The neural network does thus not control the crane directly. When deployed on a real crane, the velocity controller could also function as a safety layer and reduce the speed or even stop the crane in certain situations.

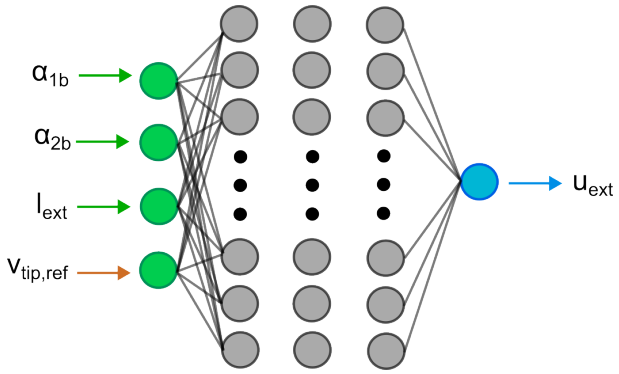


*Figure 2: Neural network layout for the CTC strategy controller*

### 2.2 Training Environment

The neural network is trained using the Pytorch framework in a Python environment where a model of the crane is implemented. The model has a geometry part for simulating the movements of the crane, and an energy consumption part for estimating the energy consumption. The geometry model does not include any dynamics and all movements are ideal, i.e. the commanded cylinder speeds always result in the desired movement. Once deployed on a real crane, an other feedback

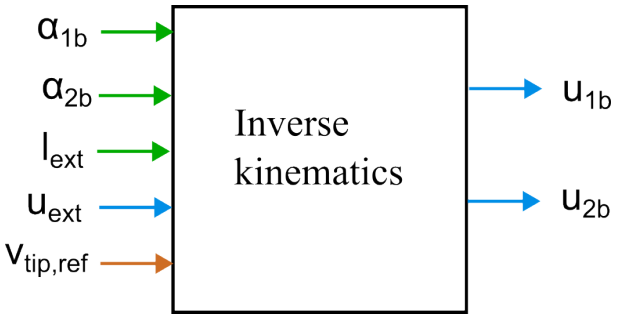


*Figure 3: Layout of the CTC velocity controller*

loop can handle the modeling errors. The energy consumption model has an ideal flow model, i.e. the pump flow directly corresponds the movement of the cylinders, according to eq. (4), where $A_{cyl}$ denotes the active side cylinder areas, $\Delta_{clen}$ the cylinder position increment and $T_s$ the step time. The total flow demand thus differs both with how far the cylinders need to travel and which cylinders that are used, since the areas of the cylinders are different.

The estimation of the pump pressure in the energy consumption model is based on models of the cylinder pressure, fitted to measurement data, together with flow dependent pressure drop models between the valve and the cylinders fitted to measurement data, according to eq. (5), where $p_{func,cyl}$ are the active side cylinder pressures, $\Delta p_{func}$ are the pressure drops between the valve and the cylinders and $\Delta p_{pump}$ is the pump pressure margin. The cylinder pressure level required to move each cylinder depends on the torque of that joint but also from counter pressure and friction in the system.

$$q_{pump} = \frac{\Delta 1b_{clen}}{T_s} * A_{1b,cyl} + \frac{\Delta 2b_{clen}}{T_s} * A_{2b,cyl} + \frac{\Delta ext_{clen}}{T_s} * A_{ext,cyl} \tag{4}$$

$$p_{pump} = \max[p_{1b,cyl} + \Delta p_{1b}, p_{2b,cyl} + \Delta p_{2b}, p_{ext,cyl} + \Delta p_{ext}] + \\ + \Delta p_{pump} \tag{5}$$

During training, a batch of cylinder positions that corresponds to a specified crane tip position is sampled. A simulation with a fixed timestep on a fixed horizon is rolled out on the batch for a number of episodes. The accumulated cost for each trajectory in the batch is calculated and forms the cost function according to eq. (6), where $P_{scaled}$ is the power required during the current step, scaled with the speed of the current step according to eq. (7), $\Delta\dot{x}$ is the error with regards to the reference speed, and $q_{limited}$ is a boolean which is true if more than the maximum flow the pump can deliver is required. The cost is summarized over a fixed horizon, $H$, in order to promote long term gains rather than instant.

The policy of the strategy controller is deterministic so the only randomness comes from the sampling of starting positions in each batch. In future studies it would be of interest to see if a probabilistic policy can learn faster.

$$\sum_{i=1}^{H} C_i = \sum_{i=1}^{H} (P_{scaled,i} * w_e + (\Delta\dot{x}_i)^2 * w_x + q_{limited} * w_{qlim}) \tag{6}$$

$$P_{scaled,i} = \frac{q_{pump} * p_{pump}}{|\dot{x}_i|} \tag{7}$$

Even though the velocity reference following is handled by the velocity controller, the cost function includes the velocity error so that the model learns to avoid singularities and to stop when the outer boundaries of the working area are reached.

After each episode, the mean of the accumulated cost of all trajectories is calculated and set as the loss variable the optimization is set to minimize. The gradients of the loss with respect to the network parameters are computed during a backwards pass and the parameters are then updated by the optimizer in order to minimize the loss. The built in optimizer AdamW of Pytorch is used.

### 2.3 Test Cases

For this initial proof of concept study, two different networks for the strategy controller are trained on two different trajectories, with fixed starting crane tip positions. In each episode, the roll out is simulated for a fixed horizon of 50 samples, that with a sample time of 0.5 s corresponds to 25 s. The value is chosen so as to represent a rather long movement of the crane in a single direction, so that long trajectories can be compared to dynamic programming solutions and benchmarked. In real operation the direction will often change after a shorter time than that, and optimizing over longer trajectories would in many cases be sub-optimal. The horizon might have to be reduced when optimizing a network for the complete working envelop of the crane in future studies. However, a too small horizon might cause the optimizer to only being able to find solutions that are good in a very small, local area. Combining many such locally optimized parts of the controller might not result in a globally optimal solution for longer trajectories.

A fixed payload of 1000 kg and a fixed reference speed of 100 mm/s are used. A reference speed of 100 mm/s does not need to be limited in many areas which makes energy comparisons easier. If a higher reference speed that has to be limited in certain areas is set, a problem is that the optimizer can minimize the energy consumption be lowering the reference speed by finding cylinder configurations that have low maximum speed.

The networks are trained until convergence, which takes about 100 episodes with a learning rate of 0.0005.

The strategy controllers are evaluated together with the velocity controller on four different starting cylinder configurations each, on the same crane tip trajectory as they are trained. Table 1 shows the definition of the four test cases, TC1 to TC4, for the first trajectory, and Table 2 show the same for the second trajectory.

## 3 Results

The four different test cases for each trajectory are also given as input to the dynamic programming optimization in order to

*Table 1: Test Cases for trajectory 1*

| | Tip start pos [x,y] [m] | Cyl start state $[\alpha_{1b}, \alpha_{2b}, l_{ext}]$ [deg, deg, m] |
|---|---|---|
| TC 1 | [1, 0.8] | [54.03, -151.2, 0.8795] |
| TC 2 | [1, 0.8] | [28.99, -144.5, 0.1855] |
| TC 3 | [1, 0.8] | [43.76, -148.2, 0.6110 ] |
| TC 4 | [1, 0.8] | [35.17, -145.9, 0.3686] |

*Table 2: Test Cases for trajectory 2*

| | Tip start pos [x,y] [m] | Cyl start state $[\alpha_{1b}, \alpha_{2b}, l_{ext}]$ [deg, deg, m] |
|---|---|---|
| TC 1 | [3, 2] | [54.03, -151.2, 0.8795] |
| TC 2 | [3, 2] | [28.99, -144.5, 0.1855] |
| TC 3 | [3, 2] | [43.76, -148.2, 0.6110 ] |
| TC 4 | [3, 2] | [35.17, -145.9, 0.3686] |

give an indication of how well, in terms of energy efficiency, the different controllers perform. The dynamic programming algorithm uses the same energy model as the CTC strategy networks are trained on.

Table 3 shows the average power consumption for DP and NN controller respectively, on the four test cases from the trajectory where the first network is trained.

*Table 3: Comparison of average power demand for DP and NN controllers for the four test cases of trajectory 1.*

| Variable | $P_{avg,DP}$ [kW] | $P_{avg,NN}$ [kW] | Diff |
|---|---|---|---|
| TC 1 | 0.7561 | 0.7633 | +1.0% |
| TC 2 | 1.0522 | 1.0685 | +1.6% |
| TC 3 | 0.8349 | 0.8509 | +1.9% |
| TC 4 | 0.9501 | 0.9705 | +2.2% |

The average power consumption is very similar for the two controllers. By looking closer into one of the test cases, illustrated in Figure 4, it is clear that they perform very similar motions of the cylinders. The cylinder commands for the two controllers are displayed in Figure 5. A noticeable difference is that the command to the extension cylinder is not exactly zero for the NN controller since outputting exactly zero is very improbable for a neural network. In situations where the extension is the pump pressure setting function this could be an issue, but during this test case (as well the other three) the first boom is setting the pump pressure for the majority of the trajectory. This gives very similar pump pressures for the two controllers, see Figure 6.

If the second set of controllers, trained and optimized for another trajectory, is considered, another interesting difference regarding the speed command to the extension is highlighted. The average power consumption for the NN controller is actually lower than for the DP controller for all four test cases, see

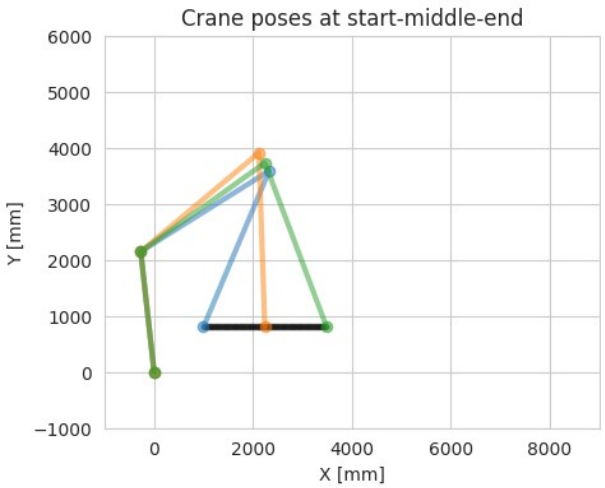


*Figure 4: Crane poses at start, middle and end of test case 2 for the NN controller of trajectory 1.*

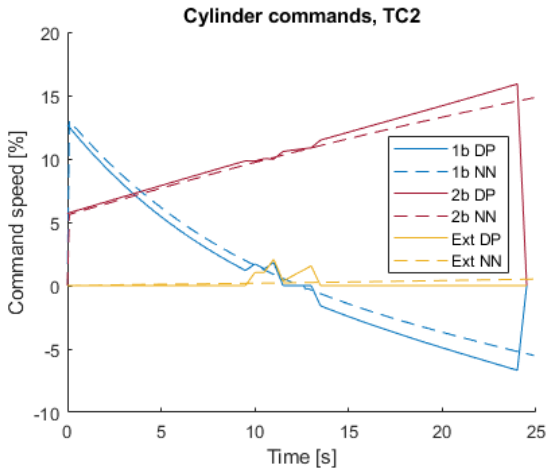


*Figure 5: Cylinder commands for the two different controllers, test case 2, trajectory 1. The commands are very similar, but note that the extension command for the NN controller is not exactly zero.*

Table 4. The reason is that the network finds solutions where the extension is driven at a very low speed, see the cylinder commands of test case 3 (illustrated in Figure 7) in Figure 8. This results in a lower pump flow during the whole trajectory, thanks to that the first and second boom can be operated more slowly. The pump pressure is not affected by operating the extension, the second pump is setting the pump pressure. The solution the network has found is, however, not valid. The speed command to the extension is lower than the lowest non-zero command that can be sent to the valve on the real system. The dynamic programming algorithm is discretized with a grid on the extension speed that has approximately the same resolution as the valve and can thus not find the illegal solution found by the network. The same issue appears for all four test cases.

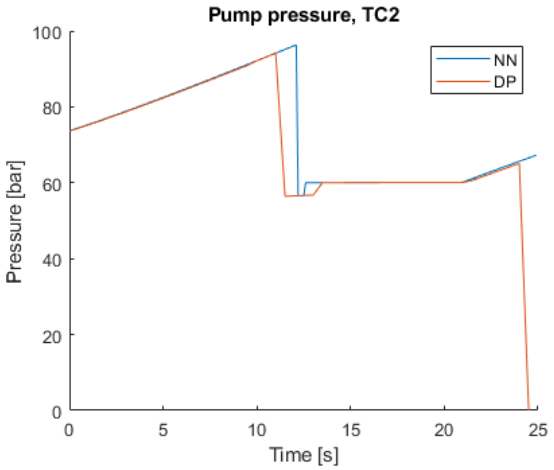


*Figure 6: Pump pressure for the two different controllers, test case 2, trajectory 1.*

*Table 4: Comparison of average power demand for DP and NN controllers for the four test cases of trajectory 2.*

| Variable | $P_{avg,DP}$ [kW] | $P_{avg,NN}$ [kW] | Diff |
|---|---|---|---|
| TC 1 | 1.9493 | 1.8336 | -5.9% |
| TC 2 | 2.2427 | 2.0894 | -6.8% |
| TC 3 | 2.0567 | 1.8830 | -8.4% |
| TC 4 | 2.1417 | 1.9327 | -9.8% |

## 4 Discussion

The results of the small CTC strategy networks trained in this work are promising, but only one step on the way towards a complete energy efficient CTC controller. In [2] four different networks, positive and negative x direction and positive and negative y direction, were used to form a complete CTC controller. For any other direction than horizontal or vertical, linear combinations of two of networks are used. Linear combinations of different network keeps the reference velocity but the energy consumption will not be optimal. Given that losses from simultaneous operation is a significant loss contributor, the energy consumption will be far from optimal since the linear combination rarely will result in only two functions being active at the same time. With the network and loss function setup of this paper it is however possible, at least in theory, to train a single network for covering all possible directions.

The network should also cover all possible payloads, all possible speeds and all possible starting positions. Given that the training time for one of the small networks in this study was about 40 minutes on a desktop computer with NVIDIA RTX 4070 GPU, training of such a network will most likely be very time consuming. It must also be considered that the inference time of a, possibly, much larger network must still fit within the sample time of the embedded system on the crane.

When comparing cylinder commands between the DP and the NN controllers, a general behavior is that the NN controllers are more smooth and unable to capture abrupt changes, that in some cases could allow the DP to find more optimal solutions.

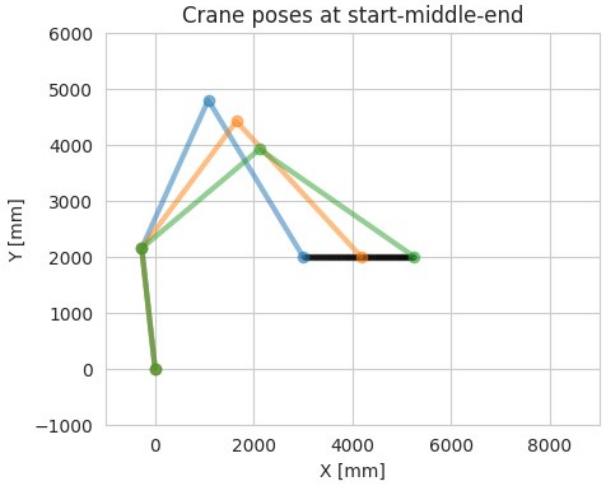


*Figure 7: Crane poses at start, middle and end of test case 3 for the NN controller of trajectory 2.*

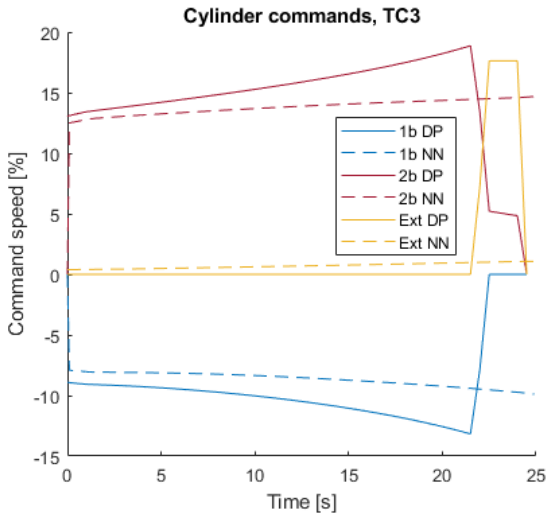


*Figure 8: Cylinder commands for the two different controllers, test case 3, trajectory 2. The extension command for the NN controller is smaller than what the DP controller can find.*

However, it remains to be evaluated whether this more on-off like control of the cylinders is as good when deployed on a real crane. Fast switching of cylinders can lead to more delays and dynamic forces on the crane, causing worse reference following and energy performance.

As seen in the results of the second trajectory, the more abrupt control of the the DP solutions could also be the cause of the discretization of the DP algorithm. The discretization matches the solutions to the resolution of the valve commands on the real system, and forces it to find solutions that are more energy consuming than if a continuous control signal had been possible, which it what the NN controller assumes. There is no straight forward way of introducing such a constraint on the output from the network during training, as it would be a non-differentiable function which would prevent the backwards propagation of gradients to flow properly. If a NN controller is deployed on the real system, the extension command

could be rounded to a legal value, and the commands to the first and second boom re-calculated if necessary to keep the crane tip within the tolerances of the planned trajectory. The energy consumption would be slightly less optimal, but the main task of CTC, driving the crane tip according to a set speed and direction, would still be carried out properly.

## 5 Conclusion

With sufficient training, a neural network for deciding CTC strategies can be taught that, together with a feed forward velocity controller, performs close to the optimal in terms of energy efficiency, on the small part of the working area that it is trained for. On the evaluated test cases, the energy consumption of the CTC controllers developed in this study consumes up to 2.2 % more energy than the near-optimal solutions given by a dynamic programming algorithm. The question remains how well the results will hold for a less generalized network that covers the whole working area of the crane.

## Acknowledgment

## References


[1] Amy Rankka, Marcus Rösth, and Alessandro Dell'amico. Evaluating the performance of semi-autonomous kinematically redundant loader crane operation. GFPS PhD Symposium, accepted for publication, 2024.

[2] Abdolreza Taheri, Amy Rankka, Pelle Gustafsson, Joni Pajarinen, and Reza Ghabcheloo. End-effector cartesian velocity control for redundant loader cranes using reinforcement learning. *IEEE Transactions on Robotics*, 2024.

[3] Hiab Official Webpage. Ctc – crane tip control taking ease of use to a new level, 7 2020.

[4] Rickard Nilsson. Inverse kinematics, 2009.

[5] Jarmo Nurmi and Jouni Mattila. Global energy-optimal redundancy resolution of hydraulic manipulators: Experimental results for a forestry manipulator. *Energies*, 10, 2017.

[6] Alessandro Tringali and Silvio Cocuzza. Finite-horizon kinetic energy optimization of a redundant space manipulator. *Applied Sciences (Switzerland)*, 11:1–15, 3 2021.

[7] Jennifer Andersson, Kenneth Bodin, Daniel Lindmark, Martin Servin, and Erik Wallin. Reinforcement learning control of a forestry crane manipulator. 2021.