A Mapping Approach for Controlling Different Maritime Cranes and Robots Using ANN

F. Sanfilippo [1], L. I. Hatledal [1], H. Zhang [1] and K. Y. Pettersen [2]

[1] Department of Maritime Technology and Operations, Aalesund University College, Postboks 1517, 6025 Aalesund, Norway,
[fisa, laht, hozh]@hials.no

[2] Department of Engineering Cybernetics, Norwegian University of Science and Technology, 7491 Trondheim, Norway,
kristin.y.pettersen@itk.ntnu.no

This work is partly supported by the Research Council of Norway through the Centres of Excellence funding scheme, project number 223254 and the Innovation Programme for Maritime Activities and Offshore Operations, project number 217768

IEEE ICMA 2014

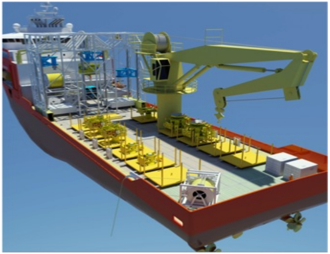
Summary

Current maritime crane control architecture

Low control flexibility and non-standardisation are two crucial issues:

- relatively simple control interfaces;
- array of levers, throttles or buttons are used to operate the crane joint by joint;
- each input device can normally control only one specific crane model.

When considering working efficiency and safety, this kind of control is extremely difficult to manage and extensive experience with high control skill levels is required of the operators [1].

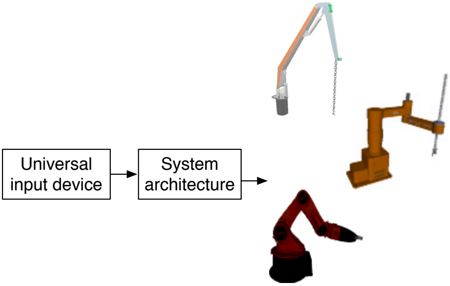


[1] F. Sanfilippo et al., "A universal control architecture for maritime cranes and robots using genetic algorithms as a possible mapping approach", ROBIO, 2013

Currently used input devices

Underlying idea

A general architecture that allows for modelling, simulation and control of different models of maritime cranes and, more generally, robotic arms by using the same universal input device.



Main challenges:

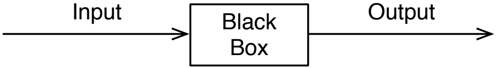
- mapping the fixed degrees of freedom of the universal input device to the variable degrees of freedom of the cranes or robots to be controlled;
- designing and testing different mapping procedures (a case study method based on the use of Artificial Neural Networks (ANNs) is presented).

Machine learning procedures

- A common assumption in the previous literature is that the IK model of the arm to be controlled is a priori knowledge.
- This method is not very flexible, especially when planning to control different arms using a universal input device because several IK models are needed: one for each arm or crane to be controlled.

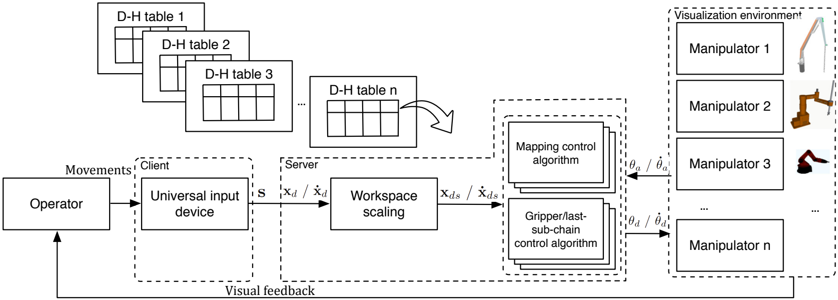
An alternative control method based on machine learning procedures:

- using methods that do not assume a priori knowledge for the IK model of the arm;
- In this way the system would be able to automatically learn the kinematic properties of different arms and new models could also be easily added.



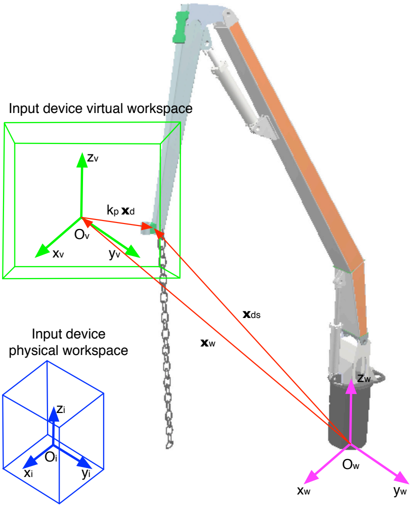
Most of the previous works in this field are only able to learn the control of a specific crane/arm. No common universal input device to control various cranes/arms with different kinematics exists.

The proposed control system architecture, overview



The proposed architecture provides the possibility of controlling the arms in position mode or velocity mode.

The proposed control system architecture, workspace scaling



The proposed architecture allows for expanding and shifting the small-scale physical workspace of the input device to a virtual expanded workspace allowing the robot arm for more accurate and precise movements.

$$\mathbf{x}_{ds} = k_p \mathbf{x}_d + \mathbf{x}_w, \qquad (1)$$

where k_p is the position scaling factor and $\mathbf{x}_w$ is a shifting vector that defines the position of the virtual reference frame with respect to the global reference frame.

$$\dot{\mathbf{x}}_{ds} = k_v \dot{\mathbf{x}}_d, \qquad (2)$$

where, k_v is the velocity scaling factor.

The proposed control system architecture, mapping control algorithm

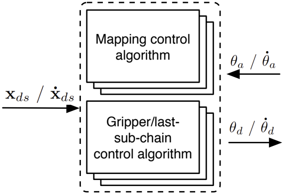
For all the different models to be controlled, the mapping methods have to implement the classic inverse kinematic functions that can be generalised as follows:

$$\theta_d = f_p^{-1}(\mathbf{x}_{ds}), \tag{3}$$

concerning position control, and

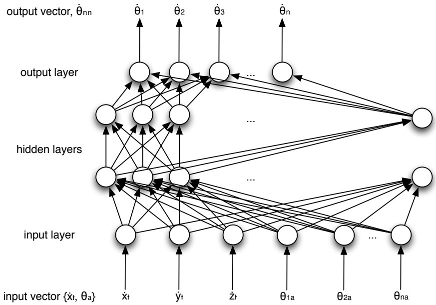
$$\dot{\theta}_d = f_v^{-1}(\theta_a, \dot{\mathbf{x}}_{ds}), \tag{4}$$

for velocity control, where θ_a is the the actual joint angles vector.



The proposed architecture also allows the end-effector to be modelled as a distinct sub-chain that can be controlled separately. In general, a mapping control method may or may not consider the control of the whole manipulator.

The proposed ANN architecture



- The system automatically learns the mapping function, (4), for the different manipulators to be controlled.
- This approach only requires the FK models.

- The unique feature of this method compared to previous works is that the same set-up of the proposed algorithm is adopted independently of which manipulator is being controlled and whether the selected control mode is position or velocity.
- When controlling each specific manipulator and once selecting the particular control mode, the same instance of ANN is continuously used.
- What differs are the semantics and the size of inputs and outputs which are dynamically and automatically set by the system.

The proposed ANN architecture

- The input vector is given by $\{\dot{\mathbf{x}}_t, \theta_a\}$, where $\dot{\mathbf{x}}_t$ is the target velocity whereas θ_a is the actual joint configuration.
- The target velocity, $\dot{\mathbf{x}}_t$, depends on the operation scenario and it is given by:

$$\dot{\mathbf{x}}_t \simeq \mathbf{x}_{ds} - \mathbf{x}_a, \tag{5}$$

  if operating in position control mode, where $\dot{\mathbf{x}}_a$ is the actual end-effector velocity.
- If instead operating in velocity control mode, it is given by:

$$\dot{\mathbf{x}}_t = \dot{\mathbf{x}}_{ds}. \tag{6}$$

- From the top side, the output vector, denoted by $\dot{\theta}_{nn}$, consists of the target joint velocities.
- The number of neurons in the input and output layers changes according to the number of DOFs of the manipulator to be controlled. The number of hidden neurons is experimentally chosen to be equal to 3/2 of the sum of the size of the input layer and the size of the output layer.

Error function, training data and training Process

$$MSE(\dot{\theta}_d, \dot{\theta}_a) = \frac{1}{n}\sum_{i=1}^{n}(\dot{\theta}_d - \dot{\theta}_a)^2. \tag{7}$$

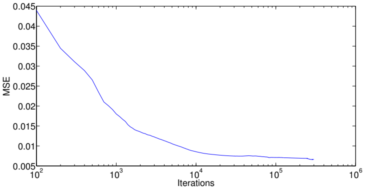
- Training data set: $(\{\dot{\mathbf{x}}_t, \theta_a\}, \dot{\theta}_{nn})$. In this specific case study, the Jacobian matrix is used. First, a set of samples is randomly generated in the Joint space, then the Jacobian matrix is built by using the differential approach. Successively, a set of random joint velocities is created and used to calculate the corresponding velocities in the Cartesian space.
- *Resilient Propagation* (RPROP) learning heuristic [2]. to overcomes the inherent disadvantages of pure gradient-descent. RPROP performs a local adaptation of the weight-updates according to the behaviour of the error function. This leads to an efficient and transparent adaptation process.
- According to the operational scenario, the output is obtained by:

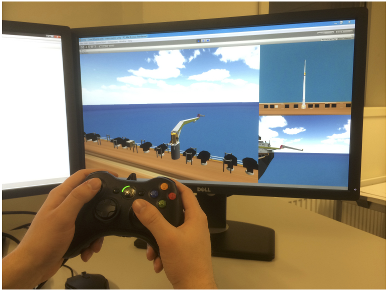
$$\theta_d = \int \dot{\theta}_{nn}\, dt, \tag{8}$$

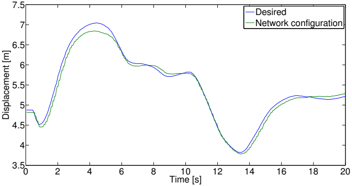
$$\dot{\theta}_d = \dot{\theta}_{nn}. \tag{9}$$

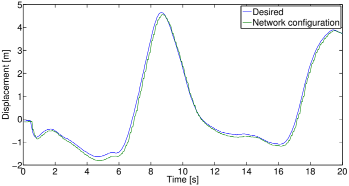
[2] M. Riedmiller et al., "A direct adaptive method for faster backpropagation learning: The RPROP algorithm", 1993

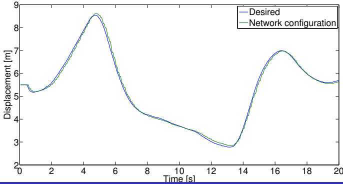
Experimental results











Conclusion and future work

Iteratively learning the IK properties of different arms:

- Using this approach, the system is able to automatically learn the IK properties of different models. Learning is done iteratively based only on observation of input-output relationships, unlike most other control schemes.

Compare different mapping methods:

- As future work, it would be interesting to compare different mapping methods and their corresponding performances.

- In order to do this, a machine learning framework that provides a selection of existing learning approaches and allows for implementing new algorithms has been developed by our research group [3].
- This framework will be used to develop a standard benchmark suite for testing and measuring the effectiveness and accuracy of the compared mapping methods, especially for maritime cranes.

[3] L. I. Hatledal et al., "JIOP: a Java Intelligent Optimisation and Machine Learning Framework", ECMS, 2014

Demo

Thank you for your attention