State Space Constrained Iterative Learning Control for 3DOF Robotic Manipulator

Aleksandar Dubonjac
Ph.D. student
University of Belgrade
Faculty of Mechanical Engineering

Mihailo Lazarević
Full professor
University of Belgrade
Faculty of Mechanical Engineering

In this paper, the trajectory tracking problem of a nonlinear robotic system with 3DOFs under the control signal obtained through nonlinearly constrained state spaceIterative Learning Control (ILC) methods is considered.The focus of this paper is the analysis of different control system parameters on the convergence rate of two constrained state space ILCalgorithms:Bounded Error Algorithm (BEAILC) and Constrained Output algorithm (COILC), as well as the comparison between these two algorithmsthrough simulations. The obtained results have shown that COILC algorithm converges faster than BEAILC algorithm when compared with the same learning and feedback parameters, due to lower trajectory restrictions. Also, it has been shown that an increase in feedback gains can decrease the number of iteration terminations during the learning process, thus allowing for more of the trajectory error information to be learned from during the single iteration. Moreover, simulations have shown that the decrease in learning parameter values will increase the number of iterations required to obtain the desired tracking accuracy.

Keywords: *robot control, ILC, bounded error, state space, constrained output*

1. INTRODUCTION

In recent years, there has been increased interest in the use of various advanced control techniques for engineering and scientific applications. Recently, iterative learning control (ILC) has attracted researchers and scientists' attention as one of the promising fields in intelligent control, [1,2]. In 1984, Arimoto et al. [3] proposed ILC as one of theadvanced learning control strategies for the control of robot manipulators. Namely, some mechatronic systems, such as production machines or industrial robots, often perform the same task repeatedly.

Iterative learning control is an intelligent control method which, through repetition in every iteration, using the information from previous trials, improves the control signal for the next trial in order to decrease (eliminate) the tracking error of a repetitivetask.

This method can be compared to a human skill practice. Rehearsing the movement and observing the error over time the error decreases as the number of repetitions increases.

In industrial applications robotic manipulators are programmed to execute a certain task that is repeated numerous times during its life cycle, making the ILC viable control method to consider, [4-6].

The key purpose of using ILC is to achieve high performance after a few iterations. One advantage of ILC is that there is no requirement for the dynamic model of the controlled system. A typical ILC in the time domain presents a simple off-line feedforward learning control. In terms of how to use the tracking error signal of the previous iteration to form the control signal of the current iteration, the structures of ILC appeared as D-type, P-type, PD-type, and PID-type, [2].

However, there are several critical requirements that limit the applications of ILC, especially to complex nonlinear (robotic) systems. Despite the widespread use of ILC in robot motion control, few attempts have been made to create an integrated design, [7,8].

In reality industrial robots have space boundaries, velocity limits and other saturations which make them constrained state space systems. Thus, all operations performed by these manipulators are contained within the constrained state space, setting particular trajectory planning and tracking requirements near the state space boundaries[8].

Another problem of the standard ILC procedure is the transient error growth. It is possible that in the first several iterations the tracking error grows significantly before it starts to converge to zero [9].

In this paper, constrained state space Iterative Learning Control methods for a nonlinear robotic manipulator with 3 DOFs are suggested. Especially, two constrained state space ILC algorithms are applied: Bounded Error Algorithm (BEAILC) and Constrained Output algorithm (COILC), as well as the comparison between these two algorithms through simulations are done.

2. CONSTRAINED STATE SPACE ILC

The robotic system can be represented with the following equation (obtained from Lagrange's Equations of Motion of second kind)[10]]:

Received: January 2021, Accepted: March 2021
Correspondence to: Dr Mihailo Lazarević
Faculty of Mechanical Engineering,
Kraljice Marije 16, 11120 Belgrade 35, Serbia
E-mail: jsmith@mas.bg.ac.rs
doi: 10.5937/fme2102429D

$$\begin{aligned}&\left[a(q)\right]\ddot{q}+\left[b(q,\dot{q})\right]\dot{q}=Q\\&Q=Q^{a}+Q^{c}+Q^{\beta}+Q^{g}\end{aligned}\tag{1}$$

where $A = [a(q)]$ is inertia matrix consisted of elements $a(q)$ that represent the metric tensor coordinates, $q\in\Re^{n}$, $\dot{q}\in\Re^{n}$ are generalized coordinates and velocities respectively, $Q^{g}\in R^{n}$, $Q^{c}\in R^{n}, Q^{\beta}\in R^{n}$ and $q\in\Re^{n}$ are generalized gravity, elastic, viscous friction and actuator torques (control signals in our case).

The $[b(q,\dot{q})]\in\Re^{n\times n}$ represents the matrix with thefollowing elements

$$b_{\gamma\alpha}(q,\dot{q})=\sum_{\beta=1}^{n}\Gamma_{\alpha\beta,\gamma}\dot{q}^{\beta},\gamma,\alpha=1,2,\ldots,n,\tag{2}$$

where the $\Gamma_{\alpha\beta,\gamma}$ are the Christoffel symbols of the first kind.

Observing the robot's differential equation, some ofthe saturations (constraints) are:

$$\begin{aligned}&Q_{a}\in\left[-Q_{a}^{min},Q_{a}^{max}\right]\\&q_{i}\in\left[Q_{i}^{min},Q_{i}^{max}\right]\\&\dot{q}_{i}\in\left[-\dot{Q}_{i}^{min}\ \ ,\dot{Q}_{i}^{max}\right]\\&i=1,2,\ldots,n.\end{aligned}\tag{3}$$

As a consequence, violation of constraints can cause the failure of the ILC trial (as it will be interrupted prematurely) and could potentially cause damage to the robot or its surroundings. Such violation can occur if the robot is following the desired trajectory that is near to the state space limits (Figure 1a).

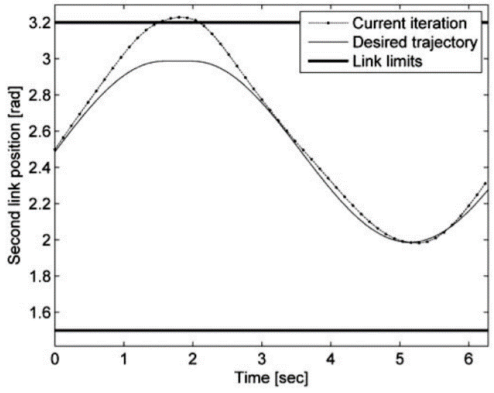

Figure 1a. Violation of generalized coordinates constraints [8]

Research has shown that despite the proven convergence of the standard nonlinear ILC method, during the first few iterations the error grows (sometimes significantly) before it starts to converge to zero potentially violating the operative space boundaries. If the output trajectory is constrained in such a way that generalized coordinates boundaries aren't violated, then it's possible to apply the ILC method to the manipulator. As solutions to this transient error growth problem, two of the following algorithms are proposed:

- Bounded Error Algorithm (BEA)
- Constrained Output Algorithm (CO)

The idea behind both algorithms is that the tracking process is terminated if the generalized coordinates preset boundaries are violated. Both algorithms break the equal trial time duration postulate during the premature termination of the tracking process, but their convergence is proven [9].

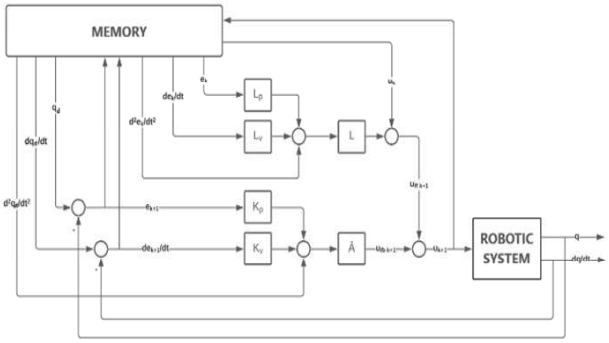

Figure 1b. Block diagram of BEAILC and COILC algorithms

The sufficient condition for both algorithms' convergence, taking into account the corresponding feedback and feedforward update laws (Figure 1b), is:

$$|I-LA^{-1}|\leq\rho<1.\tag{4}$$

For a high convergence rate, the learning operator should be as close as possible to the inertia matrix. As the real inertia matrix is unknown, it's preferable that the learning operator is chosen as estimated inertia matrix $L(q)\equiv\hat{A}(q)$, as it is proven in the previous research[12].

2.1 Bounded Error Algorithm

Bounded Error Algorithm solves the previously mentioned problem by tracking the error norm during the iteration. Iteration is terminated momentarily as the error norm reaches a predetermined limit.A control signal is corrected before the next iteration only with information collected before the interruption of the previous iteration.

In combination with the feedforward control signal generated with BEA ILC algorithm, the following feedback term will be used:

$$\begin{aligned}u_{fb}=\hat{A}(q)\big[&\ddot{q}_{d}(t)+K_{v}\left(\dot{q}_{d}(t)-\dot{q}_{k}(t)\right)+\\&+K_{p}\left(q_{d}(t)-q_{k}(t)\right)\big]\end{aligned}\tag{5}$$

where $\hat{A}(q)$ is the estimated inertia matrix, and K_v and K_p are feedback gains.

Planned desired trajectory inside the generalize coordinates constrains satisfies the inequality:

$$min\left(\min_{t\in[0,T]}\left(Q_{i}^{max}-q_{i}^{d}(t)\right),\ \min_{t\in[0,T]}\left(q_{i}^{d}(t)-Q_{i}^{min}\right)\right)>\mu\tag{6}$$

where $\mu$is the accuracy of the ILC method.

Accordingly, δ can be selected as:

$$\delta = min\left(\min_{t\in[0,T]} \left(Q_i^{max} - q_i^d(t) \right), \min_{t\in[0,T]} \left(q_i^d(t) - Q_i^{min} \right) \right) - \quad (7)$$
$$-\mu < 0$$

and $\varepsilon = \mu + \delta$ for which is guaranteed $q_i^d \in \left[Q_i^{min} + \varepsilon, Q_i^{max} - \varepsilon \right]$. If the control signal in an initial iteration is $u_0(t) \equiv 0, t \in [0,T]$, then the restriction can be applied to the output trajectory at every iteration by applying BEA algorithm, represented through the following steps[13]:

1. Set the initial iteration number k = 0 and begin theiterative procedure
2. Starting from the initial position $q_k(0) = q_d(0)$ the system is tracking the desired trajectory under the control $u(q,t) = u_k(t) + u_{fb}(t)$ while $|q_k(t) - q_d(t)| < \varepsilon$ and $t < T$. When $t = T$ or for the first $T_k : 0 < T_k < T, \left\| q_k(t) - q_d(t) \right\|$, then the tracking process is stopped and T^k is set to the stop time of iteration k.
3. At the end of the current iteration the learning controller updates the input control signals for the next iteration u_{k+1} according to the following learning update law:

$$u_{k+1}(t) = u_k(t) + + \begin{cases} L(q_k(t))[\ddot{q}_d(t) - \ddot{q}_k(t) + L_v(\dot{q}_d(t) - \dot{q}_k(t)) \\ + L_p(q_d(t) - q_k(t))], t \in [0, T_k]; \\ 0, t \in (T_k, T] \end{cases} \quad (8)$$

where L_p and L_v are learning gains.

4. If the overall output error is less than or equal to an acceptable tracking accuracy and T_k equals T, thenthe learning procedure terminates successfully andthe optimal feedforward control signal is u_k. Otherwise, set $k = k + 1$ and go to step 2.

As a result of applying the BEA algorithm, the output trajectory in each iteration lays inside the hypercylinder with a radius of epsilon around the desired trajectory, thus always staying inside of state space constrains (Figure 2).

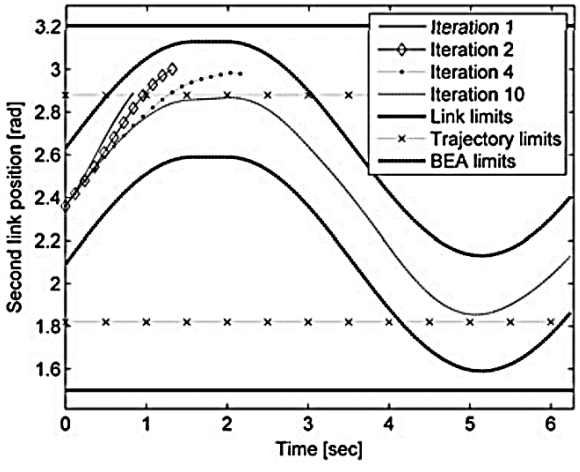

Figure 2. Hypercylinder around the trajectory [9]

This algorithm can speed up the rate of convergence when compared to the standard ILC, where the error accumulation increases the number of iterations required to eliminate said accumulation.

The downside of BEA algorithm is its over restriction in the areas where the trajectory is well far from its generalized coordinates boundaries, leading to higher number of iterations due to more frequent iteration interruptions.

2.2 Constrained Output Algorithm

CO algorithm directly limits the output trajectory maximum value. In the contrary of BEA which forces entire trajectory to stay inside of the hypercylinder with a radius of an epsilon, which is defined as the closest safe distance from the generalized coordinates boundaries, CO limits only the maximum output value allowing deviations from the desired trajectory in areas where it's safe to do so.

CO algorithm can be described through the following steps [9], taking into account the feedback term (5):

1. Set the initial iteration number $k = 0$ and begin theiterative procedure
2. Starting from the initial position $q_k(0) = q_d(0)$ the system is tracking the desired trajectory under the control $u(q,t) = u_k(t) + u_{fb}(t)$ while $Q_i^{min} < q_i^k < Q_i^{max}, i = 1, 2, .., n$ and $t < T$. When $t = T$ or for the first $T_k : 0 < T_k < T, q_i^k = Q_i^m in$, or $q_i^k = Q_i^{min}$ then the tracking process is stopped and T_k is set to the stop time of iteration k.
3. At the end of the current iteration the learning controller updates the input control signals for the next iteration u_{k+1} according to the following learning update law:

$$u_{k+1}(t) = u_k(t) + + \begin{cases} L(q_k(t))[\ddot{q}_d(t) - \ddot{q}_k(t) + L_v(\dot{q}_d(t) - \dot{q}_k(t)) \\ + L_p(q_d(t) - q_k(t))], t \in [0, T_k]; \\ 0, t \in (T_k, T] \end{cases} \quad (9)$$

4. If the overall output error is less than or equal to an acceptable tracking accuracy and T_k equals T, then the learning procedure terminatessuccessfully and the optimal feedforward control signal is u_k. Otherwise, set $k = k + 1$ and go to step 2.

This algorithm should speed up the process of finding the desired control signal, due to its reduced number of iteration terminations caused by lower restrictions compared to BEA algorithm.

3. SIMULATION RESULTS

Trajectory tracking simulations of the 3DOF robotic system (Figure.3) using BEAILC and COILC algorithms are conducted in MATLAB and Simulink environments. Simulink modelled robot differential equations are solved using the Runge-Kutta method (ODE4), where the simulation step was 0.00001.

Mass for each segment is chosen as: 0.15 kg, 0.5 kg, 0.35 kg; and the length of each segment is chosen correspondingly: 0.15 m, 0.5 m, 0.35 m.

The desired trajectories defined in the space of generalized coordinates for joints are:

Figure 3. The adopted robotic system with 3 DOFs

$$\begin{aligned} q_d^1(t) &= 4sin(t), \\ q_d^2(t) &= 2cos(t), \\ q_d^3(t) &= 0.8cos(2t), \\ \forall t &\in [0,T],\ T = 2\pi. \end{aligned} \tag{10}$$

Initial conditions for generalized coordinates and their derivatives are:

$$q^i(0) = q_d^i(0),\ \dot{q}^i(0) = \dot{q}_d^i(0) \tag{11}$$

The learning operator is chosen as the estimated inertia matrix for both algorithms, which is obtained by varying segments' mass and length so that the sufficient condition for convergence is met:

$$\max_{q^i} I - \hat{A}(q)A^{-1}(q) = 0.9282,\ q^i \in [-2\pi, 2\pi] \tag{12}$$

The rest of the control system parameters were manually set and chosen through trial and error.

Generalized coordinates boundaries and control system parameters for both algorithms were set so that the simulation results are comparable.

Hypercylinder radius for BEAILC algorithm is set as:

$$\varepsilon = 0.3 \tag{13}$$

Generalized coordinates limits for COILC algorithm are set as:

$$\begin{bmatrix} q_{\max}^1 & q_{\max}^2 & q_{\max}^2 \\ q_{min}^1 & q_{min}^2 & q_{min}^3 \end{bmatrix} = \begin{bmatrix} 4.3 & 2.3 & 1.1 \\ -4.3 & -2.3 & -1.1 \end{bmatrix}. \tag{14}$$

The desired tracking accuracy that has to be obtained by both algorithms $e_{max}^i < \mu$ is:

$$\mu = 0.005 \tag{15}$$

Simulation results will be shown through the individual joint trajectory tracking with the parameters obtained after the learning process, then the maximum tracking error through iterations will be shown, iteration duration time, as well as the trajectories during the learning process for each joint.

3.1 The first set of parameters – BEA and CO

As suggested in [[13] and [14], parameters are chosen as following diagonal matrices:

$$\begin{aligned} K_p &= 120 * I,\ K_v = 60 * I \\ L_p &= 100 * I,\ L_v = 20 * I \end{aligned} \tag{16}$$

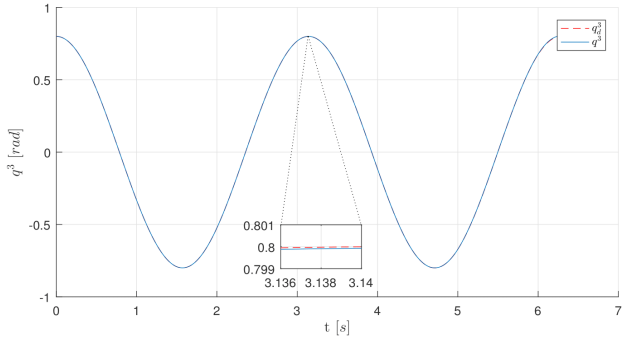

Figure 4.Final trajectory tracking – q^3 - BEA

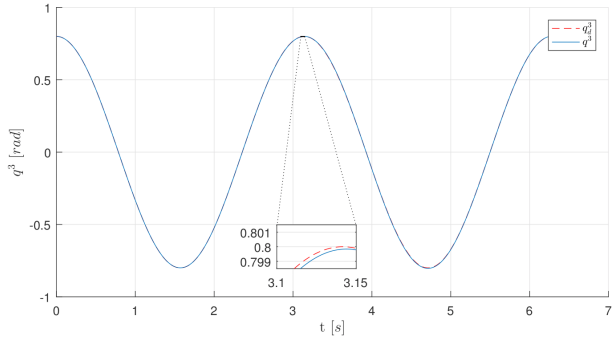

Figure 5.Final trajectory tracking - q^3 - CO

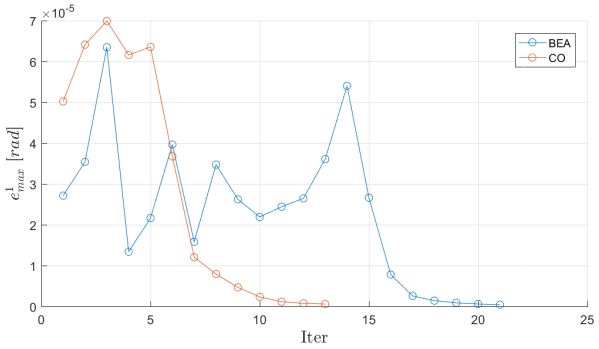

Figure 6. Maximum error norm $\left\| e_{max}^1 \right\|$ through iterations

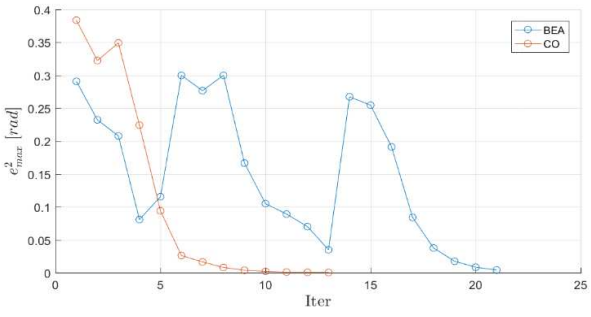

Figure 7. Maximum error norm $\left\| e_{max}^2 \right\|$ through iterations

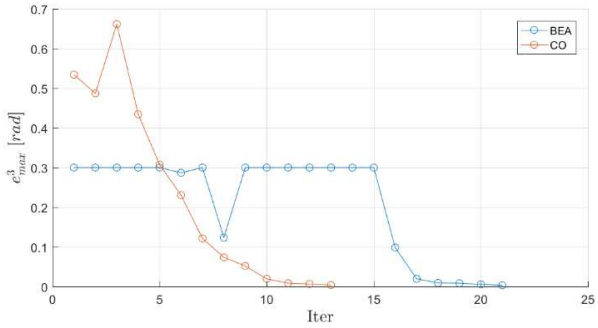

Figure 8. Maximum error norm $\left\|e^3_{max}\right\|$ through iterations

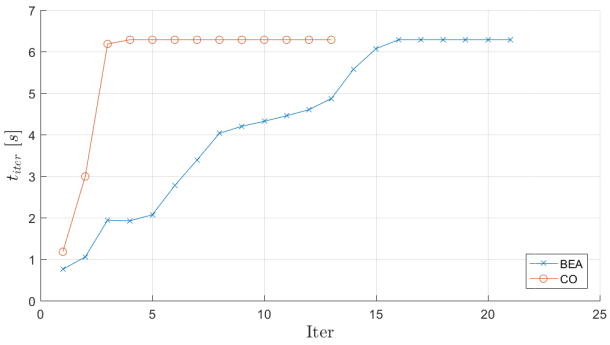

Figure 9. Iteration duration time

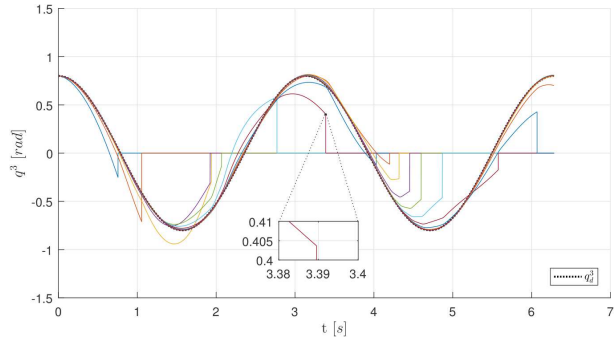

Figure 10.Trajectory tracking q^3 through iterations - BEA

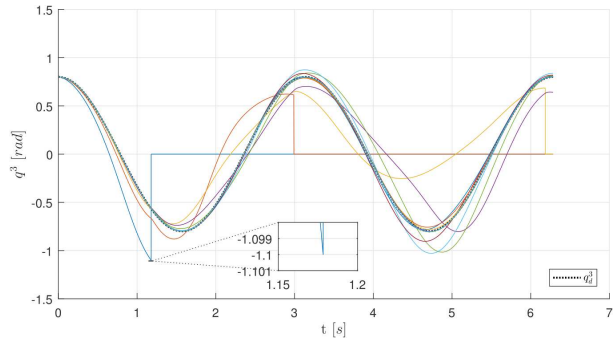

Figure 11.Trajectory tracking q^3 through iterations - CO

Applying the BEA algorithm with (16) parameters, the desired accuracy was obtained after 21 iterations, with maximum tracking errors:

$$e^1_{max} = 4.130688144865502e(-07)$$
$$e^2_{max} = 0.004140314720489$$
$$e^3_{max} = 0.003325527637983$$

From Figures 6, 7 and 8 it can be seen that the iterations were interrupted by the violation of generalized coordinatesconstraints, while on Figure9 duration of iterations can be seen individually. As the iteration duration time increases, the control system successfully decreases the tracking error and incrementally obtains more learning information as the iteration interruptions occur later in the tracking process. Due to high restrictions enforced by hypercylinder, interruptions occur regularly, thus decelerating the learning process and increasing the number of iterations. In Figure 4 final trajectory tracking for the third joint can be seen.

In Figures 7 and 8 it can be seen that maximum tracking errors are limited to the value of $\varepsilon = 0.3$, which means that the algorithm is working properly. Besides that, it can be seen which of the joints caused the iteration interruption as well as that monotonic maximum error convergence can be interrupted as the iteration is interrupted, depending on the position of the joint in the moment of interruption. The trajectory tracking through iterations for the third joint is shown in Figure 10.

Applying the CO algorithm with (16) parameters, the desired accuracy was obtained after 13 iterations, with maximum tracking errors:

$$e^1_{max} = 5.669905331906477e(-07)$$
$$e^2_{max} = 3.983853512188329e(-04)$$
$$e^3_{max} = 0.003912110721542$$

In comparison with BEAILC with same controller parameters and previously mentioned comparable boundaries, it can be seen that CO algorithm requires less iterations to achieve desired tracking accuracy. Due to less restricting constrains, interruptions were less frequent, which resulted in faster convergence. On Figure 5 it can be seen that the final trajectory for the joint 3 almost overlaps the desired trajectory.

In Figure 11 enlarged constrain violation is shown for the third joint where $q^3 = 1.1$. Despite that the absolute value of maximum error was higher in other areas of trajectory, it didn't interrupt the iteration, proving that the algorithm is implemented correctly.

3.2 The second and third set of parameters - BEAILC

Control system parameters can be chosen in a way that the individual joints can be targeted with different gain values.

Second parameter set:

$$K_p = \begin{bmatrix} 90 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 120 \end{bmatrix}, K_v = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 40 & 0 \\ 0 & 0 & 60 \end{bmatrix}$$
$$L_p = \begin{bmatrix} 65 & 0 & 0 \\ 0 & 70 & 0 \\ 0 & 0 & 80 \end{bmatrix}, L_v = \begin{bmatrix} 15 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 45 \end{bmatrix} \quad (17)$$

Third parameter set:

$$K_p = 150 * I \ , \ K_v = 80 * I$$
$$L_p = 70 * I \ , \ L_v = 15 * I \quad (18)$$

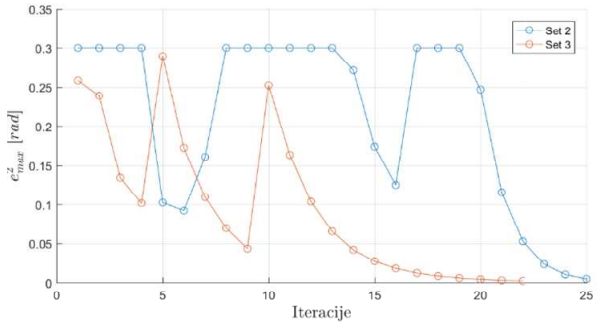

Figure 12. Maximum error norm $\left\|e^2_{max}\right\|$ through iterations

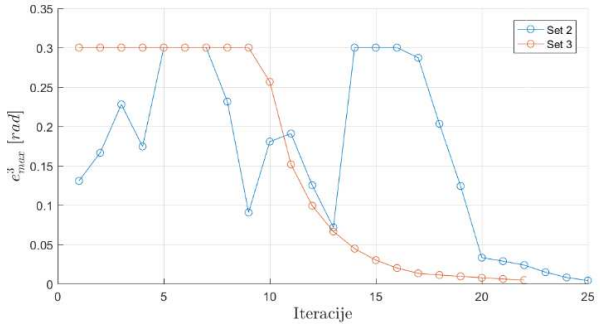

Figure 13. Maximum error norm $\left\|e^3_{max}\right\|$ through iterations

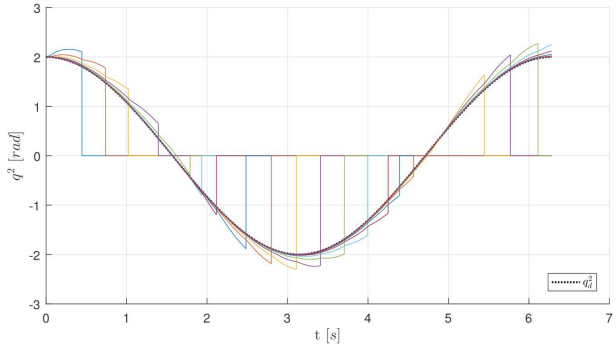

Figure 14. Trajectory tracking q^3 through iterations – set 2

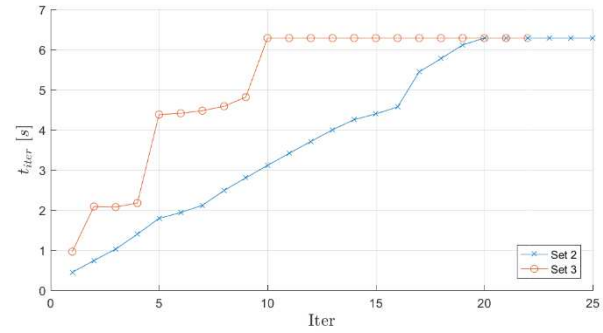

Figure 15. Iteration duration time

Applying the BEA algorithm with (17) parameters, the desired accuracy was obtained after 25 iterations, with maximum tracking errors:

$$e^1_{max} = 2.204710552455858e(-07)$$
$$e^2_{max} = 0.004570354342683$$
$$e^3_{max} = 0.004075814633505$$

In comparison with previous parameter set where the third joint caused most of the iteration terminations (Figure 8), with this parameter set it can be seen that now the second joint caused most of the trial terminations (Figures 12, 13 and 14).

Applying the BEA algorithm with (18) parameters, the desired accuracy was obtained after 22 iterations, with maximum tracking errors:

$$e^1_{max} = 3.769541385700848e(-07)$$
$$e^2_{max} = 0.002220904697019$$
$$e^3_{max} = 0.004660588997101$$

Comparing the iteration duration time of the first set and this current set 3 of parameters (Figure.15), number of iteration increased due to lower learning gains. On the other hand, due to higher feedback gains the number of interruptions has decreased. The interruptions with this set of parameters were caused primarly by the third joint.

3.3 The second and third set of parameters - COILC

With (17) and (18) sets of parameters the COILC simulation results are following:

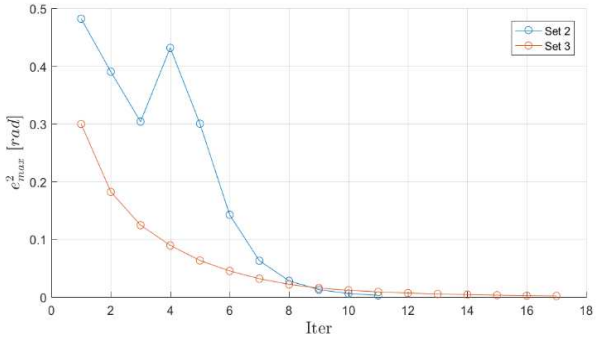

Figure 16. Maximum error norm $\left\|e^2_{max}\right\|$ through iterations

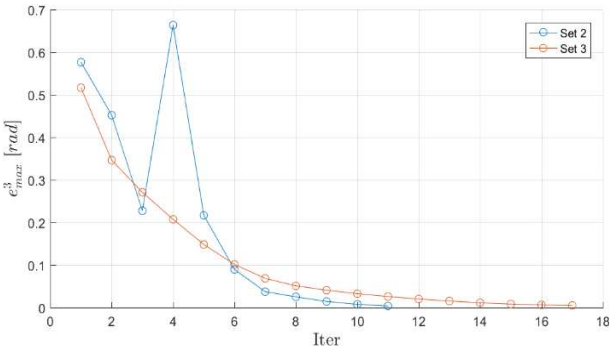

Figure 17. Maximum error norm $\left\|e^3_{max}\right\|$ through iterations

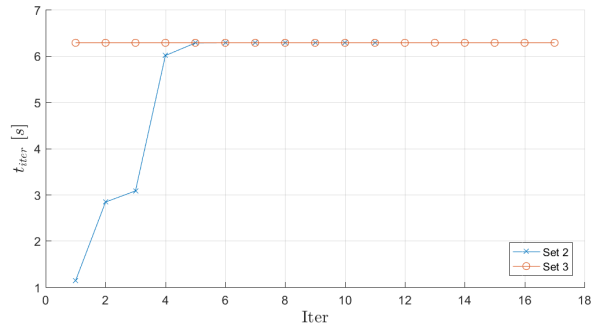

Figure 18. Iteration duration time

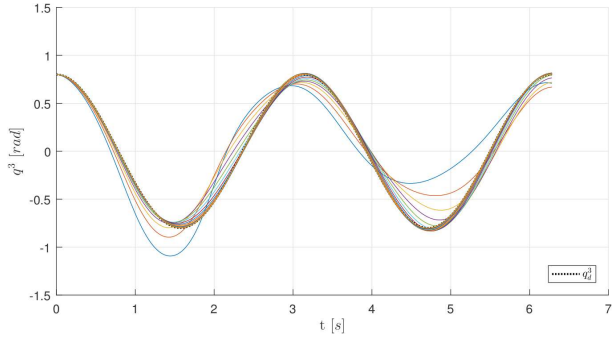

Figure 19. Trajectory tracking q[3] through iterations – set 3

Applying the CO algorithm with (17) parameters, the desired accuracy was obtained after 11 iterations, with maximum tracking errors:

$$e^1_{max} = 9.318638003463775e(-07)$$
$$e^2_{max} = 0.002733375280047$$
$$e^3_{max} = 0.003863099758940$$

As in previous comparison, it can be seen that CO algorithm takes less iterations to obtain the desired tracking accuracy when compared to the BEA algorithm.

Applying the CO algorithm with (18) parameters, the desired accuracy was obtained after 17 iterations, with maximum tracking errors:

$$e^1_{max} = 8.846494474745725e(-07)$$
$$e^2_{max} = 0.001859841362631$$
$$e^3_{max} = 0.004862432948212$$

Like in the case of BEAILC, convergence is slower due to lower learning gains compared to the first parameter set, but thanks to the lower restrictions and higher feedback gains with the third parameter set iterations weren't terminated (Figure18), which means that the output trajectory was inside the CO boundaries, Figure 19. Maximum error norms for joint 3 for both sets can be seen on Figures 16 and 17. More simulation results can be found in [15].

3.4 BEAILC convergence rate influence of radius ε

For convergence rate influence of radius ε demonstration the following parameter will be used:

$$K_p = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 105 & 0 \\ 0 & 0 & 130 \end{bmatrix}, K_v = \begin{bmatrix} 40 & 0 & 0 \\ 0 & 60 & 0 \\ 0 & 0 & 80 \end{bmatrix}$$
$$L_p = \begin{bmatrix} 80 & 0 & 0 \\ 0 & 90 & 0 \\ 0 & 0 & 100 \end{bmatrix}, L_v = \begin{bmatrix} 20 & 0 & 0 \\ 0 & 30 & 0 \\ 0 & 0 & 50 \end{bmatrix} \quad (19)$$

Values for ε are:

$$\varepsilon_1 = 0.2, \varepsilon_2 = 0.3, \varepsilon_3 = 0.4, \varepsilon_4 = 0.5 \quad (20)$$

From Figures 20a, 20b and 20c shown below, it can be seen that with radius ε value increase, BEA algorithm requires less iterations to obtain the desired tracking accuracy (convergence is faster). This effect is more noticeable for BEA than CO, due to higher trajectory restriction.

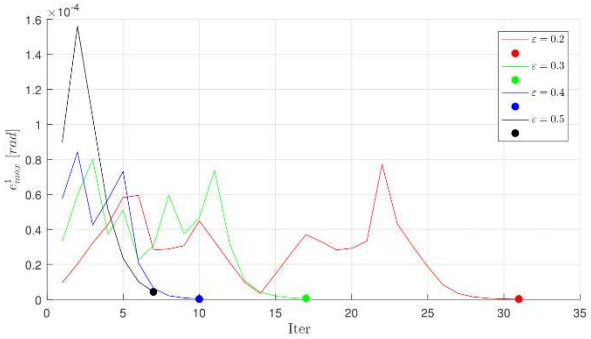

Figure 20a. Trajectory tracking q[1] through iterations

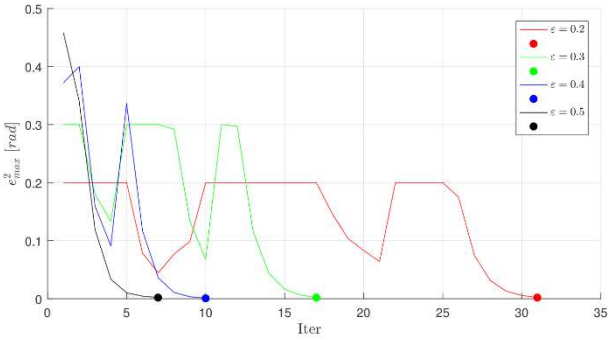

Figure 20b.Trajectory tracking q[2] through iterations

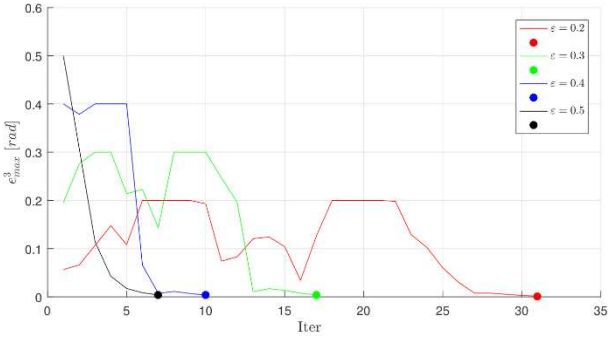

Figure 20. Trajectory tracking q[3] through iterations

4. CONCLUSION

From previously shown simulation results, it can be observed that both algorithms successfully managed to decrease the tracking error under the desired accuracy, with the learning matrix chosen as the estimated inertia matrix so that the convergence condition is met.

It is confirmed that BEAILC algorithm takes more iterations to obtain desired tracking accuracy compared to COILC algorithm with the same parameter set, due to more limiting constrains. The hypercylinder around the desired trajectory enforced by BEA over-constrains the trajectory in the areas where it's physically safe for error to increase slightly in favour of obtaining more learning information from the current trial. On the other hand, COILC algorithm with its maximum and minimum limiting values, allows higher error values as long as the trajectory is inside its constrains, thus obtaining more information from the current iteration resulting in faster convergence. Loosening the hypercylinder radius ε dec-

reases the number of iterations required to obtain the desired tracking accuracy (speeds up the convergence).

ACKNOWLEDGMENT

The presented research was supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia by contract no. 451-03-9/2021-14/200105.

REFERENCES

[1] Ahn HS, Moore K and Chen Y. *Iterative learning control robustness and monotonic convergence for interval systems*. 1st ed. London: Springer-Verlag London Limited, 2007.

[2] Xu JX, Panda SK, Lee TH. *Real-time Iterative Learning Control, Design and Applications*. 1st ed. London: Springer-Verlag, 2009.

[3] Arimoto S, Kawamura S and Miyazaki F. *Bettering operation of robots by learning*. Journal of Robotic Systems, 1984; 2(1):123-140.

[4] Isao T and Hunag PH. *Iterative Learning Control for Trajectory Tracking of Robot Manipulators*. International Journal of Automation and Smart Technology, 2017; 7(3) 133-139.

[5] Lazarević M and Panagiotis T. *Robust second-order PDalpha type iterative learning control for a class of uncertain fractional order singular systems*. Journal of Vibration and Control, Sage Journals, 2016;22(8):2004-2018, DOI: 10.1177/1077546314562241.

[6] Lazarević PM, Mandić P, Cvetković B, et al. *Advanced open-closed-loop PIDD2 /PID type ILC control of a robot arm. In:* Proceedings of the INISTA 2018 conference, Thessaloniki, Greece, 2018,pp.1-8, DOI: 10.1109/INISTA.2018.8466308.

[7] Isao T., Hunag P-H., " *Iterative Learning Control for Trajectory Tracking of Robot Manipulators*", International Journal of Automation and Smart Technology,Vol. 7 No.3,pp.133-139.2017.

[8] Yovchev K., Delchev K., Krastev E., *State Space Constrained Iterative Learning Control for Robotic Manipulators,* Asian Journal of Control, Vol. 20, No. 1, pp. 1–6, DOI: 10.1002/asjc.1680, 2018.

[9] Yovchev K., Delchev K., Krastev E., *Constrained Output Iterative Learning Control*, Faculty of Mathematics and Informatics, Sofia University, 2020.

[10] Lazarević, M., Čović. V., *Robot Mechanics,* Faculty of Mechanical Engineering, Belgrade, 2009, (*in Serbian*).

[11] M. Lazarević, M. Cajić.: *Determination of Joint Reactions in a Rigid Multibody System, Two Different Approaches,* Journal FME Transactions, Faculty of Mechanical Engineering, Belgrade, Vol.44.No2,2016.

[12] Delchev K., Zahariev E., *Computer Simulation-Based Synthesis of Learning-Control Law of Robots*, Institute of Mechanics, Bulgarian Academy of Sciences, Sofia, Bulgaria 2008.

[13] Delchev K., *Iterative learning control for robotic manipulators: A bounded-error algorithm*, Institute of Mechanics, Bulgarian Academy of Sciences, 2013.

[14] Yovchev K., *Finding The Optimal Parameters for Robotic Manipulator Applications of The Bounded Error Algorithm for Iterative Learning Control*, Journal of Theoretical and Applied Mechanics, Sofia, Vol. 47 No. 4 (2017) pp. 3-11 DOI: 10.1515/jtam-2017-0016, 2017.

[15] Dubonjac A., *Control of a robotic system with three degrees of freedoms using ILC algorithms in a constrained state space*. Master thesis, Faculty of Mechanical Engineering, University of Belgrade, 2020, (*in Serbian*).

ИТЕРАТИВНО УПРАВЉАЊЕ УЧЕЊЕМ У ОГРАНИЧЕНОМ ПРОСТОРУ СТАЊА РОБОТСКОГ МАНИПУЛАТОРА СА 3 СТЕПЕНА СЛОБОДЕ

А. Дубоњац, М. Лазаревић

У овом раду је разматрано праћење трајекторије нелинеарног роботског система са 3 степена слободе под дејством управљања добијеног применом нелинеарних метода итеративног управљања учењем (ILC) у ограниченом простору стања. Фокус рада је анализа утицаја различитих параметара управљачког система на брзину конвергенције два ILC алгоритма у ограниченом простору стања: Алгоритма ограничене грешке и алгоритма ограниченог излаза, и међусобно поређење ових алгоритама кроз симулације.Добијени резултати су показали да COILC алгоритам конвергира брже од BEAILC алгоритма, када се упореде са истим вредностима параметара учења и повратне гране, услед слабијих ограничења трајекторије. Такође, показано је да повећање вредности параметара у повратној грани смањује број прекида итерација током процеса учења, тиме омогућавајући више информација за учење из грешке праћења трајекторије у итерацији. Даље, симулације су показале да смањење вредности параметара учења повећава број итерација потребних да се достигне жељена тачност праћења.