

# NASTAVITEV PID Z ALGORITMOM DIFERENCIALNE EVOLUCIJE V PLK NA PRIMERU REGULACIJE PRETOKA ZRAKA

Božidar Bratina, Nenad Muškinja, Milan Rotovnik, Marjan Golob

## Izvleček:

Prispevek predstavlja primer izvedbe naprednih algoritmov v PLK višjenivojskem programskem jeziku SCL za potrebe nastavitve parametrov PID na primeru procesne avtomatizacije. Z razvojem strojne opreme je možno napredne algoritme izvajati tudi direktno v PLK, še posebno zanimive pa so vgradnje algoritmov umetne inteligence in strojnega učenja. V okviru zaključnih del smo s študenti izvedli več prenosov naprednih algoritmov v PLK in jih primerjali s teoretičnimi metodami in praktičnimi orodji v industriji. Na področju uporabe optimizacijskih algoritmov v procesni industriji smo za namen demonstracije nastavitve parametrov PID izvedli z uporabo algoritma diferencialne evolucije na PLK na učnem laboratorijskem modelu regulacije pretoka zraka. Zaradi specifičnosti in omejenosti programskih jezikov so bile rešitve namenoma deloma izvedene tudi s pomočjo orodij generativne umetne inteligence, da se lahko ovrednoti kvaliteta podanih rešitev. Pojav generativnih modelov, kot so chatGPT, Copilot, Gemini, DeepSeek itd., nezadržno vpliva tudi na pedagoški proces in nove priložnosti podajanja učnih vsebin na vseh stopnjah izobraževanja. Izkušnje z različnimi projekti študentov avtomatike in robotike kažejo, da orodja umetne inteligence (UI) še ne zmorejo podati celovitih rešitev, lahko pa služijo kot dobra podpora.

## Ključne besede:

procesna avtomatizacija, algoritmi vodenja, optimizacija parametrov PID, izobraževanje, orodja umetne inteligence

## 1 Uvod

Hitremu razvoju tehnologij in prehodom industrijskih nivojev običajno sledijo tudi prilagoditve učnih vsebin na področju tehniškega izobraževanja. Prehod v industrijo 4.0 je zajel celovito avtomatizacijo procesov z medsebojno povezljivostjo naprav na lokalnem in oblačnem nivoju in postopno vključevanje orodij umetne inteligence (ang. artificial intelligence, AI). Prehod v industrijo 5.0 razširja obstoječi nivo s ključno spremembo, da v ospredje vrača človeka pri odločanju in upravljanju z modernimi orodji (tudi UI). Združevanje izkušenj, spretnosti in človeku prilagojenih naprednih orodij s ciljem dose-

ganja boljšega sodelovanja med stroji in ljudmi. Pri tem so prepoznane ključne tehnologije za prehod, kot so generativna umetna inteligenca, razširjena resničnost, digitalni dvojčki in tehnologije veriženja blokov.

Pojav orodij (generativne) umetne inteligence spreminja način pridobivanja informacij, kot smo ga bili vajeni do nedavnega. Tudi na področju terciarnega izobraževanja že leta tiskane knjige in skripte zamenjujemo s spletnimi vsebinami, interaktivnimi nalogami, kvizi, projekti, alternativnimi metodami učenja, kjer študenti preko različnih metod poučevanja samostojno pridobivajo in iščejo rešitve. Pri pedagoškem delu na nižji stopnji študente elektrotehnike smeri avtomatike in robotike poučujemo vsebine klasične avtomatizacije, kot so: gradniki sistemov vodenja, programiranje programabilnih logičnih krmilnikov (PLK), integracija komponent, komunikacije v avtomatiki, regulacijski algoritmi, vmesniki človek-stroj (HMI), nadzorni (SKADA) sistemi itd. Te vsebine so že desetletja bistven del in se sproti dopolnjujejo glede na razvoj in pojav novih tehnologij. S pridobljenim predznanjem si zato lahko na višji stopnji privoščimo bolj napredne vsebine, ki jih zaradi hitrega razvoja tehnologij in orodij

**Dr. Božidar Bratina**, univ. dipl. inž.,  
**doc. dr. Nenad Muškinja**, univ. dipl. inž.,  
**Milan Rotovnik**, izr. prof. **dr. Marjan Golob**, univ.  
dipl. inž., vsi Univerza v Mariboru, Fakulteta za  
elektrotehniko, računalništvo in informatiko



© The Authors 2025. CC-BY 4.0

<https://doi.org/10.5545/Ventil-31-2025-3.02>

težje direktno vključujemo v posodabljanje vsebin, saj se tehnologije hitro spreminjajo.

Z leti opažamo spremembe v prid vse večje uporabe in zanašanja na informacije iz modelov UI kot pa na iskanje rešitev iz predhodnega znanja, literature, mentorstva itd. Tako smo vplive uporabe orodij UI zaznali tudi pri študentskih projektih ali zaključnih delih, kjer so potrebni teoretično predznanje, integracija vsebin različnih predmetov in celovit pristop k reševanju problema [1, 2]. Uporabo seveda vzpodbujamo kot pripomoček pri raziskovalnem delu študentov na magistrski stopnji elektrotehnike smeri avtomatike in robotike v okviru Laboratorija za procesno avtomatizacijo (LPA) na Fakulteti za elektrotehniko, računalništvo in informatiko v sklopu Univerze v Mariboru. Pri delu so običajno zelo samostojni in iznajdljivi, na UM pa jim je omogočeno vključevanje tako v raziskovalno kot projektno delo v sklopu laboratorijev in inštitutov [3].

V nadaljevanju bo predstavljen en primer takšnega zaključnega dela, kjer je bilo pri delu uporabljeno predhodno znanje študenta, dopolnjeno z orodji umetne inteligence pri razvoju končne rešitve. Za nalogo je bil izveden algoritem diferencialne evolucije v PLK za potrebe nastavitve parametrov PID na učnem laboratorijskem modelu regulacije pretoka zraka. Prenos znanstvenih metod in algoritmov iz simulacijskih okolij, izvajanih na osebnih računalnikih, v industrijsko opremo ni enostaven, zato predstavljena rešitev lahko služi kot dober primer za industrijo 5.0, kjer se vse več naprednih algoritmov integrira v majhne vgrajene naprave.

## 2 Vgradnja diferencialne evolucije v PLK za nastavitev parametrov regulatorja PID

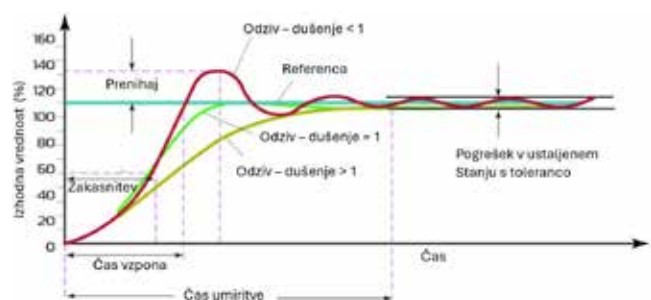
Regulatorji PID (proporcionalno-integrirno-diferencialni) so široko uporabljani v industrijskih regulacijskih sistemih zaradi svoje učinkovitosti in relativne preprostosti. Ključen izziv je določitev dobrih, če že ne optimalnih parametrov ( $K_p$ ,  $T_i$ , in  $T_d$ ), saj neustrezne nastavitve vodijo do nestabilnosti ali slabe regulacije. Enačba 1 predstavlja zapis idealne (paralelne) strukture regulatorja PID.

$$PID(S) = K_p \left( 1 + \frac{1}{T_i S} + T_d S \right) \quad (1)$$

Metode nastavitve parametrov PID v industriji, kot so Ziegler-Nicholsova metoda, Chien-Hrones-Roswick, Cohen-Coon in druge, pogosto ne zagotavljajo optimalnega delovanja pri kompleksnih ali nelinearnih sistemih. Zato se vse pogostejše uporabljajo inteligentni in metahevristični optimizacijski algoritmi, ki omogočajo iskanje optimalnih nastavitve parametrov PID. Optimizacijski algoritmi, od numeričnih, biomimetičnih do evolucijskih, so v praksi najbolj pogosti genetski algoritmi (GA), al-

goritmi na osnovi rojev delcev (PSO), diferencialna evolucija (DE) in v zadnjem času algoritmi na osnovi umetne inteligence (globoke mreže), metode strojnega učenja (učenje z okrepitvijo) itd.

Optimizacijski algoritmi omogočajo iskanje optimalnih parametrov PID s pomočjo cenilne funkcije, ki lahko zajema različne regulacijske zahteve: prenihaj (ang. overshoot), ustalitev (ang. settling time), pogrešek v ustaljenem stanju (ang. steady-state error), čas zakasnitve (ang. delay time), čas vzpona (ang. rise time), čas vrha prenihaja (ang. peak time), in se posamično ali v kombinaciji ovrednotijo preko integralnega kvadratičnega pogreška (ISE), integralnega absolutnega pogreška (IAE) in podobno (slika 1).



Slika 1 : Regulacijske zahteve za zaprtozankni odziv

V zadnjem desetletju se je pojavilo ogromno primerov optimizacije parametrov PID z različnimi algoritmi optimizacije in kombinacijami, redko pa se zaradi kompleksnosti implementirajo v industrijske naprave. [4–6] Na splošno se je uporaba diferencialne evolucije pri optimizaciji parametrov PID izkazala za izjemno uspešno, saj nudi:

- ▶ **stabilno konvergenco:** DE-algoritmi so znani po svoji sposobnosti stabilnega iskanja optimalnih rešitev, tudi v prisotnosti šuma in nestabilnosti sistema;
- ▶ **robustnost:** DE-algoritmi so manj občutljivi na začetne pogoje in lahko učinkovito obravnavajo nelinearnosti in večdimenzionalne iskalne prostore;
- ▶ **prilagodljivost:** Sodobne različice DE-algoritmov vključujejo kombinacije z ostalimi algoritmi ali modifikacije za prilagodljive parametre, ki izboljšajo učinkovitost optimizacije v realnih aplikacijah.

Velika prednost pa je tudi nastavitev parametrov PID brez potrebe po modeliranju sistema.

### 2.1 Diferencialna evolucija

Diferencialna evolucija (DE) je populacijski, metahevristični optimizacijski algoritem. Primeren je za reševanje problemov z neznanimi ali kompleksnimi funkcijami in deluje na principih evolucije, podobno kot genetski algoritmi. Od njih se razlikuje z uporabo diferencialne mutacije za generiranje novih kan-

didatov oziroma rešitev in je bil v osnovi zasnovan za iskanje globalnega optimuma. Deluje brez potrebe po gradnji modela ali izračunavanja odvodov, zato je DE hitro prodrla na številna področja, kot so strojništvo, ekonomija, biomedicina, nastavitve PID, načrtovanje omrežij, načrtovanje poti itd. Delovanje algoritma je predstavljeno z naslednjimi koraki, tudi predstavljenimi s psevdokodo v *tabeli 1*:

- ▶ inicializacija populacije naključnih rešitev znotraj določenih mej;
- ▶ mutacija: za vsakega člana populacije se ustvari mutirana rešitev z uporabo linearne kombinacije razlik med naključnimi člani populacije;
- ▶ križanje (rekombinacija): kombinacija mutirane in obstoječe rešitve za ustvarjanje nove rešitve;
- ▶ izbor: če ima nova rešitev boljšo ocenjeno vrednost, ta nadomesti obstoječo.

Algoritem DE za svoje delovanje uporablja različne matematične funkcije, kot so generator naključnih števil, permutacije, matrični izračuni in podobno. Izvedba algoritma v višjenivojskih jezikih tako ne predstavlja problema, kar pa ne velja za okorno okolje PLK in strojno opremo. Z razvojem vedno močnejših serij PLK pa se umika tudi ta ovira. Tako se je za pripravo algoritma DE v PLK najprej celotna koda pripravila in testirala v Matlab okolju v simulacijah. Sprva se je testiralo delovanje algoritma DE na problemu iskanja minimuma nekaj različnih matematičnih funkcij z vnaprej znanim minimumom (Kowalik, Himmelblau, Rastrigin). Nato pa se je delovanje DE testiralo še na primeru nastavitve parametrov regulatorja PID s pomočjo matematičnih modelov sistema višjega reda z mrtvim časom. Zaprtizančni odzivi so pokazali, da je z ustrezno omejitvijo naključno generiranih parametrov PID z algoritmom DE ta konvergirala proti optimalni rešitvi.

**Tabela 1:** Psevdokoda algoritma diferencialne evolucije

|                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>1. Nastavi parametre algoritma DE:</b>                                                                                                                                                                                                                                                                                                                                                                                  |
| <ul style="list-style-type: none"> <li>▶ velikost populacije (NP),</li> <li>▶ dimenzija problema (število parametrov, 3 za PID),</li> <li>▶ faktor mutacije (npr. 0,5–0,9),</li> <li>▶ stopnja križanja (npr. 0,7–0,9),</li> <li>▶ največje število generacij,</li> <li>▶ spodnje in zgornje meje parametrov.</li> </ul>                                                                                                   |
| <b>2. Inicializiraj populacijo P z NP osebki:</b>                                                                                                                                                                                                                                                                                                                                                                          |
| <ul style="list-style-type: none"> <li>▶ za vsak <math>i \in [1, NP]</math>:</li> <li>▶ <math>x_i</math> = naključni vektor znotraj mej (dimenzije D).</li> </ul>                                                                                                                                                                                                                                                          |
| <b>3. Oцени s cenilno funkcijo <math>f(x_i)</math> za celo populacijo.</b>                                                                                                                                                                                                                                                                                                                                                 |
| <b>4. Izvajaj DE, dokler niso izpolnjeni pogoji za ustavitve</b>                                                                                                                                                                                                                                                                                                                                                           |
| <ul style="list-style-type: none"> <li>▶ za vsak osebki <math>x_i</math> v populaciji: <ul style="list-style-type: none"> <li>a) izberi tri različne posameznike <math>x_{r_1}, x_{r_2}, x_{r_3} \neq x_i</math>,</li> <li>b) izvedi mutacijo in križanje,</li> <li>c) izberi najboljše,</li> <li>d) oceni s cenilno funkcijo nove osebke,</li> <li>e) prestavi najboljše osebke v novo generacijo.</li> </ul> </li> </ul> |
| <b>5. Vrne najboljše rešitev iz končne populacije.</b>                                                                                                                                                                                                                                                                                                                                                                     |

Kot smo omenili, se je v PLK algoritem uporabil za nastavitve parametrov PID na učnem laboratorijskem modelu regulacije pretoka zraka. Pri tem je uporabljen klasični krmilnik Siemens serije S7-1500, sprogramiran z lestvično logiko za vodenje procesa in uporabo osnovnega funkcijskega bloka PID. Funkcija algoritma DE je, da z različnimi generiranimi parametri PID testira in izvaja različne skočne spremembe želene vrednosti in na podlagi zajetega zaprtizančnega odziva ponastavi parametre PID za novo iteracijo. Tako se na podlagi obnašanja sistema z regulatorjem iščejo optimalni parametri PID za doseganje zelenega zaprtizančnega obnašanja sistema oziroma dinamike.

Vključevanje orodij umetne inteligence se je pričelo s pripravo kode algoritma diferencialne evolucije v PLK, ki je bil sprogramiran v programskem jeziku SCL. Prav zaradi okrnjenosti programskih jezikov PLK in specifik programiranja je določene algoritme in funkcije težje izvesti. Problem predstavljajo matematične funkcije, ki so običajno zajete v knjižnicah višjih programskih jezikov (npr. RAND, RANDI, RANDPERM, FLOOR, MOD, TRUNC itd.), v SCL pa jih je običajno potrebno pripraviti lastnoročno. Pri razvoju smo tako identificirali več načinov kako hitro pridobiti parcialne rešitve – od UI orodij, namenskih prevajalnikov ali lastnega kodiranja, a je končno rešitev dalo šele človeško delo.

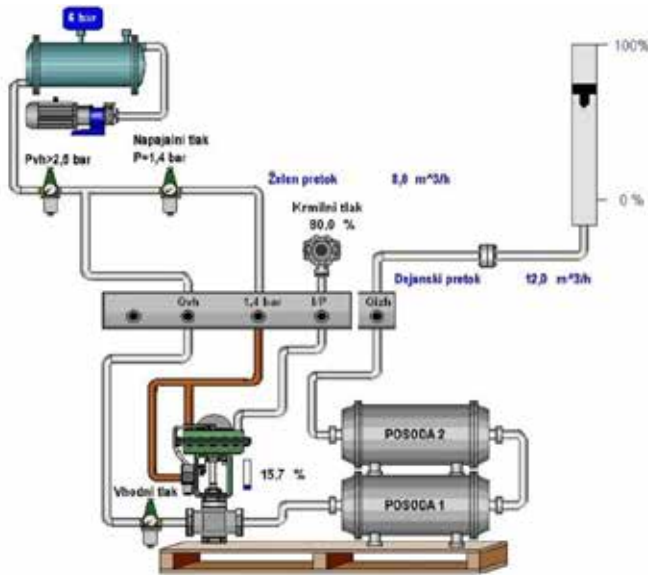
## 2.2 Učni laboratorijski model

Za testiranje smo izbrali učni laboratorijski model z regulacijo pretoka zraka, ki je zaradi uporabljenih pnevmatskih gradnikov nelinearen (sistem višjega reda, pnevmatski ventil, meritev pretoka na podlagi razlike tlakov) in v različnih delovnih točkah zahteva drugačne parametre PID. Proces na *sliki 2* večkrat uporabljamo pri laboratorijskih vajah za demonstracijo različnih vsebin avtomatizacije (gradniki, pretvorbe, programiranje). Regulacija pretoka zraka se izvaja v območju med 0–16 m<sup>3</sup>/h, s pomočjo krmiljenja pnevmatskega ventila v območju med 0,2–1 bar (odprtost med 0–100 %).

S pomočjo tokovnih pretvornikov so vsi signali pretvorjeni v območje med 4–20 mA in v krmilniku nato normirani in skalirani v območje 0–100 %. Običajno študenti problem regulacije in nastavitve parametrov PID rešujejo parcialno z določanjem stabilnih parametrov PID v nekaj delovnih območjih. Nato pa sprogramirajo posodobitev parametrov PID med delovanjem procesa glede na želeno delovno točko. V primeru nastavitve parametrov PID z optimizacijo DE pa se je želelo poiskati parametre avtomatsko v več izbranih delovnih točkah na podlagi zahtevanega stabilnega zaprtizančnega odziva.

## 2.3 Izvedba algoritma DE s funkcijskim blokom v PLK

Celotna koda, napisana v jeziku SCL, je bila predhodno testirana v simulaciji na PLK z razvitim dis-



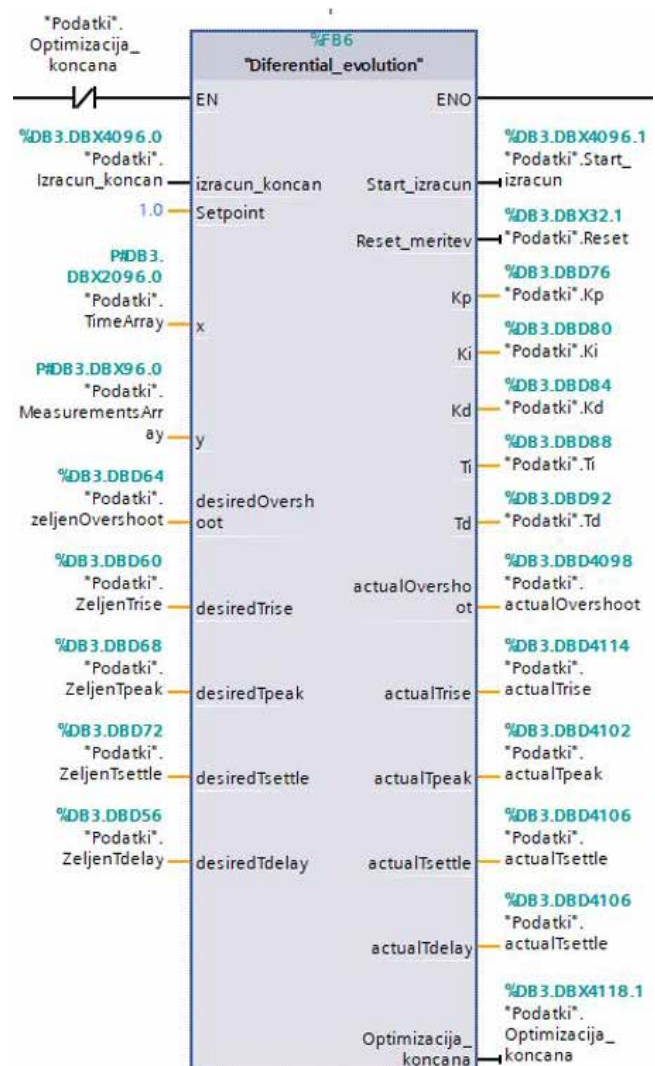
Slika 2 : Laboratorijski učni model regulacije pretoka zraka

kretnim modelom sistema 2. reda z zakasnitvijo v zaprti regulacijski zanki, s podobno dinamiko kot kasnejši realen učni model regulacije pretoka zraka. Slika 3 prikazuje razvit blok algoritma DE [7].

Pred delovanjem je v algoritmu DE potrebno določiti različne parametre: končna vrednost, ki jo algoritem doseže, da se optimizacija zaključi; število parametrov, ki jih iščemo, v našem primeru tri parametre PID; število osebkov oziroma populacije; maksimalno število iteracij; verjetnost za križanje (rekombinacijo) parametrov med različnimi osebki; stopnjo mutacije, ki določa velikost koraka spremembe parametrov; ter spodnje in zgornje meje parametrov populacije. Cenilno funkcijo smo po potrebi omejili zgolj na nekaj regulacijskih zahtev, kot so: vrednost prenihaja, čas umiritve ter minimalno regulacijsko odstopanje v ustaljenem stanju.

Končni parametri regulatorja PID so bili pridobljeni na podlagi skočnih sprememb referenčne vrednosti glede na različne parametre DE (število osebkov, iteracij, cilj, faktorje križanja in mutacije), izvedba v PLC (računanje) pa je zaradi počasnega odziva procesa in zajetnih meritev v vektorjih zahtevala daljše čase obdelave (trajanje nekaj ur). Programska koda in izvedba v jeziku SCL je pravilno delovala, optimizacija pa je vzela kar nekaj časa za doseganje stabilnih rešitev parametrov PID. Pred vsakim generiranim setom parametrov PID se je te preverilo in v testiranje vključilo le tiste, ki so bili smiselni oziroma znotraj dovoljenih mej. S tem se je zagotovilo doseganje stabilnega zaprtozračnega odziva, na podlagi katerega je bilo moč oceniti rezultat.

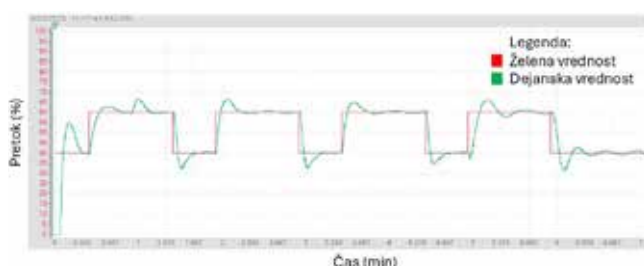
Pri testiranju smo zaradi nelinearnosti procesa izbrali tri območja delovanja procesa: pri spremembi pretoka v območju 10 % in 30 %, 40 % in 60 % in med 70 % in 90 %. Pri vsakem testu novih parametrov PID, ki jih je nastavila DE, se je zajelo meritev krmilne in procesne spremenljivke v vektor 100 se-



Slika 3 : Funkcijski blok algoritma diferencialne evolucije za optimizacijo parametrov regulatorja PID

kund z vzorčenjem 0,1 sekunde. Skočna sprememba navzgor z osveženimi parametri PID je trajala fiksnih 60 sekund, navzdol s stabilnimi parametri PID pa 40 sekund (slika 4).

Parametri algoritma diferencialne evolucije so bili nastavljeni: 3 parametri, 9 osebkov, maksimalno število iteracij 20, končna vrednost cenitve 5, faktor križanja 0,8 in mutacije 0,8. V cenilni funkciji smo upoštevali minimalni pogrešek v ustaljenem stanju 0, vrednost prenihaja 10 % in čas nastavitve 30 sekund (slika 5). Vrednosti so bile izbrane izkustveno glede na dinamiko in nelinearnost procesa. Če je bilo v cenilno funkcijo zajetih preveč kriterijev, algoritem DE ni našel ustreznih parametrov PID, ki bi zadostili vsem kriterijem.



**Slika 4 :** Graf poteka izvajanja skočnih zaprtozančnih odzivov na delovnem območju med 40 % in 60 % vrednosti pretoka, zajetih v portalu TIA

```
//Inicijalizacija
#MaxValue := #y[1];
#Overshoot := 0;
#Trise := 0;
#Tpeak := 0;
#Tsettle := 0;
#Tdelay := 0;
#Tolerance := 0.02 * #Setpoint;

//Iskanje MAX vrednosti in Tpeak
FOR #i := 1 TO 500 DO
  IF #y[#i] > #MaxValue THEN
    #MaxValue := #y[#i];
    #Tpeak := #x[#i];
  END_IF;
END_FOR;

//Izračun Overshoot
#Overshoot := ((#MaxValue - #Setpoint) / #Setpoint) * 100.0;

//Trise (čas, ko signal prvič doseže Setpoint)
FOR #i := 1 TO 500 DO
  IF #y[#i] >= #Setpoint THEN
    #Trise := #x[#i];
    EXIT;
  END_IF;
END_FOR;

//Poišči Tdelay (čas, ko signal prvič doseže 10% Setpointa)
FOR #i := 1 TO 500 DO
  IF #y[#i] >= 0.1 * #Setpoint THEN
    #Tdelay := #x[#i];
    EXIT;
  END_IF;
END_FOR;
```

**Slika 5 :** Izsek kode SCL izračuna cenilne funkcije glede na izbrane zahteve odziva.

## 2.4 Analiza rezultatov

Dobljene parametre PID z algoritmom DE v več delovnih točkah se je primerjalo z nastavitvijo v portalu TIA, ki jo omogoča osnovni funkcijski blok PID. Izkazalo se je, da so nastavljeni parametri v bloku PID vedno izračunali preveliko ojačenje  $K_p$  glede na delovno točko, kar je povzročalo nihanje procesne vrednosti okoli delovne točke. Pri določanju časovnih konstant ni bilo hujših odstopanj. V tabeli 2 so prikazane regulacijske zahteve, v tabeli 3 pa nastavljeni parametri PID pred izvedbo optimizacije in dobljeni z blokom DE in PID. Začetni parametri PID so bili omejeni glede na znanje o dinamiki in nelinearnosti procesa in pridobljeni po izkustveni metodi CHR (Chien-Hrones-Reswick), da so zagotavljali delovanje v delovni točki brez oscilacij. To je bil namreč predpogoj za izračun cenilne funkcije in postopek optimizacije z diferencialno evolucijo.

## 3 Vstop orodij UI v orodja avtomatizacije

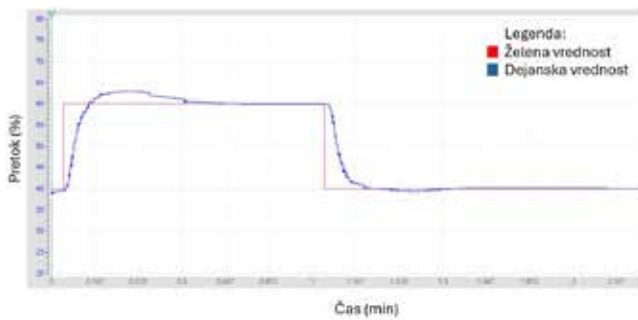
Predstavljeni primer potrjuje možnost dokaj hitre integracije naprednih algoritmov v vgrajene naprave, kot je PLK že danes. Z mnogimi primeri smo s kolegi v okviru zaključnih nalog izvedli še več zanimivih algoritmov seveda tudi s pomočjo orodij UI. Tako smo od konceptov industrije 4.0 in prehodom v industrijo 5.0 na pragu nove revolucije UI. Pristop glavnih proizvajalcev industrijske opreme se že nagiba k vključevanju teh orodij v razvojna okolja za doseganje višje učinkovitosti pri razvoju proizvodnje. S pomočjo generativnih modelov UI se bo dalo lažje avtomatizirati kompleksne naloge na področju vodenja sistemov, analitika, temelječa na podatkih, bo dala boljši vpogled v delovanje in

**Tabela 2 :** Regulacijske zahteve za optimizacijo različnih območij delovanja procesa

| Kriterij         | Območje | Zahteva za odziv | Najboljši odziv |
|------------------|---------|------------------|-----------------|
| Prenihaj (%)     | 10–30 % | 10               | 3,6             |
| Čas umiritve (s) |         | 30               | 31,4            |
| Prenihaj (%)     | 40–60 % | 10               | 8,9             |
| Čas umiritve (s) |         | 30               | 27,5            |
| Prenihaj (%)     | 70–90 % | 10               | 5,3             |
| Čas umiritve (s) |         | 30               | 27,6            |

**Tabela 3 :** Primerjava parametrov PID za različna območja delovanja procesa

| Območje | Parametri PID              | $K_p$ | $T_i$ | $T_d$ |
|---------|----------------------------|-------|-------|-------|
| 10–30 % | začetni stabilni parametri | 0,2   | 2,2   | 0,6   |
|         | diferencialna evolucija    | 0,23  | 3,0   | 0,4   |
|         | samonastavitev PID v TIA   | 0,7   | 2,4   | 0,5   |
| 40–60 % | začetni stabilni parametri | 0,4   | 2,2   | 0,6   |
|         | diferencialna evolucija    | 0,6   | 5,0   | 0,1   |
|         | samonastavitev PID v TIA   | 0,9   | 3,0   | 0,4   |
| 70–90 % | začetni stabilni parametri | 1     | 2,2   | 0,6   |
|         | diferencialna evolucija    | 0,78  | 3,0   | 0,3   |
|         | samonastavitev PID v TIA   | 1,3   | 2,5   | 0,7   |



**Slika 6 :** Končni odziv z dobljenimi parametri PID po DE v delovnem območju med 40-60 % vrednosti pretoka

generirala odločitve, ki vodijo k višji učinkovitosti in natančnosti. Generativna umetna inteligenca lahko obdeluje ogromne količine podatkov, prepozna vzorce in optimizira procese v realnem času, zaradi česar bodo podjetja in panoge, ki želijo z inovacijami ostati konkurenčna, morala vlagati v znanje, orodja UI in kadre, ki znajo uporabljati in izkoristiti njihov potencial. Pri izobraževanju kot podajalci klasičnih in/ali naprednih vsebin študentom s področja avtomatizacije smo primorani slediti trendom in omogočati študentom pridobitev čim več znanja, kompetenc in izkušenj. Soočeni z izzivi kako akademsko pridobljena znanja približati potrebam v industriji poskušamo poleg osnovnih znanj študente spoznavati tudi s koncepti industrije 4.0 in 5.0, ki jih bodo prenesli, uporabljali in razvijali naprej (IIoT, digitalni dvojčki, razširjena resničnost ...).

Vedno več podjetij na trgu že ponuja lastna orodja za razvoj digitalnih dvojčkov za boljšo analizo in optimizacijo delovanja. Po drugi strani se področje programiranja krmilnikov in razvoj kode približujeta kombiniranim konceptom z znanimi programskimi jeziki, kot sta C++ in Python. Tako se z višjega nivoja orodja UI lažje vključujejo v okolja avtomatizacije in programiranje PLK. Večja podjetja, kot npr. Siemens, vključujejo v svoje produkte MS Copilot in s tem posodablajo koncept portala TI. Prav tako podjetje B & R svoje okolje Automation Studio Code nadgrajuje z MS Copilotom, s čimer želi povišati produktivnost in izboljšati timsko delo pri razvoju programske opre-

me v smeri industrije 5.0 s poudarkom na človeških odločitvah. Z združevanjem domenskega znanja avtomatizacije in izkušenj v različnih panogah lahko pričakujemo hitreje obvladovanje strogih zahtev v proizvodnji in avtomatizaciji. Testiranje in spremljanje napredka velikih jezikovnih modelov (chatGPT, Gemini, DeepSeek itd.) pri razvoju različnih algoritmov kaže na hiter razvoj na področju avtomatizacije. Če ta orodja še pred letom dni niso podala konkretnih rešitev, na primer kode SCL diskretnega regulatorja PID, je danes slika že drugačna. Le vprašanje je potrebno pravilno formulirati.

Nazadnje sledi še zahvala vsem študentom zaključnih del, ki so vztrajno razvijali in testirali različne algoritme v kodi SCL [7], ki jih z ustreznimi dopolnitvami že vključujemo v pedagoške vaje na smeri avtomatika in robotika.

## Viri

- [1] Francis, J. N. et al.: Generative AI in Higher Education: Balancing Innovation and Integrity, British Journal of Biomedical Science, 81/ 2024.
- [2] Sengar, S. S. et al.: Generative artificial intelligence: a systematic review and applications, Multimed Tools Appl, 2024.
- [3] Vključevanje študentov v raziskovalno in projektno delo, UM, 2022.
- [4] Fayti, M., Mjahed, M., Ayad, H., & El Kari, A. (2023). "Recent Metaheuristic-Based Optimization for System Modeling and PID Controllers Tuning." Studies in Informatics and Control, 32.
- [5] Hasanien, H. M., & El-Fergany, A. A. (2021). "A novel hybrid GWO-PSO algorithm for optimal PID tuning of nonlinear systems." Applied Soft Computing, 102, 107099.
- [6] Nasir, Mohannad & Saloumi, Majd & Nassif, Ali. (2022). Review of Various Metaheuristics Techniques for Tuning Parameters of PID/FOPID Controllers. ITM Web of Conferences. 43. 01002.
- [7] Grm, A.: Uporaba optimizacijskih algoritmov za samodejno nastavitve PID parametrov v PLK, magistrsko delo. Maribor, 2024.

## PID tuning based on differential evolution algorithm in PLC for airflow control

### Abstract:

The paper presents implementation of advanced algorithms in PLC using the higher-level programming language SCL for PID parameter tuning in process automation. Nowadays advanced algorithms can also be implemented directly in PLCs, whereas implementation of artificial intelligence and machine learning algorithms is particularly interesting. In scope of the master thesis and student projects we implemented several advanced algorithms in PLCs and compared them with theoretical methods and practical industry tools. In the study case we implemented differential evolution algorithm optimization algorithm in the process industry control scheme for PID parameter tuning into PLC for task of airflow control. Due to the specificity and limitations of programming languages, the solutions were partially developed using tools of generative artificial intelligence, to evaluate the quality of obtained solutions. Generative models such as chatGPT, Copilot, Gemini, DeepSeek, etc., also have an impact on the teaching part, bringing new opportunities for teaching content at all levels of education. Experience through various student projects by students of automation and robotics shows that AI tools are not yet able to provide comprehensive solutions, but they can serve as good support.

### Keywords:

process automation, control algorithms, PID parameter optimization, education, artificial intelligence tools.