Univerza v Ljubljani

Fakulteta za elektrotehniko

Bojan Likar

Prediktivno vodenje nelinearnih sistemov na osnovi Gaussovih procesov

Magistrsko delo

Mentor : prof. dr. Juš Kocijan

V Ljubljani, september 2004

Zahvala

Iskreno se zahvaljujem prof. dr. Jušu Kocijanu za vso pomoč v času študija.

Zahvaljujem se tudi Giovanniju Godeni, univ. dipl. inž, za pomoč pri delu v procesnem laboratoriju.

Povzetek

Delo obravnava uporabo teorije Gaussovih procesov za modeliranje, simulacijo in vodenje nelinearnih dinamičnih sistemov. Gaussovi procesi so primer verjetnostnega neparametričnega modeliranja z verjetnostnim napovedovanjem izhodov. Njihova uporaba, ki izvira iz področja statistike, temelji na Bayesovem verjetnostnem modeliranju.

Gaussov proces je naključen večdimenzionalen proces, katerega naključne spremenljivke, določene z vektorjem srednjih vrednosti in kovariančno matriko, so medsebojno porazdeljene po normalnem porazdelitvenem zakonu. Ob dani učni množici, sestavljeni iz parov vhodnih in izhodih točk sistema, predpostavimo, da vsaka izhodna točka določa posamezno naključno spremenljivko, pripadajočo kovariančno matriko večdimenzionalne normalne porazdelitve pa modeliramo s kovariančno funkcijo, ki je funkcija parov vhodnih točk sistema. Vektor srednjih vrednosti večdimenzionalne porazdelitve (normalno je predpostavljena njegova ničelna vrednost) in kovariančna funkcija tako popolnoma določata Gaussov proces. Parametri kovariančne funkcije, imenovani hiperparametri, so določeni (naučeni) na podatkih učne množice. Predikcijo izhoda pri novi vhodni točki je normalna verjetnostna porazdelitev s srednjo vrednostjo in varianco. Velikost variance podaja mero zaupanja v napovedano vrednost izhoda modela.

Pri simulaciji dinamičnih sistemov z Gaussovimi procesi izhajamo iz njihove podobnosti z umetnimi nevronskimi mrežami, kjer pri simulaciji kot vhode v model uporabimo tudi zakasnjene vrednosti vhodnega in izhodnega signala. Opisani so pristopi, kako pri propagiranju zakasnjenih vrednosti izhodov modela upoštevamo tudi informacijo o njihovi verjetnostni porazdelitvi.

Identifikacija dinamičnega procesa z Gaussovimi procesi je prikazana na primeru procesa priprave plina. Prikazana je primerjava rezultatov različnih načinov propagiranja verjetnostnih porazdelitev preteklih izhodov GP modela.

Identificiran model z Gaussovimi procesi je uporabljen pri nelinearnem prediktivnem vodenju tlaka procesa priprave plina. Prediktivno vodenje, temelji na eksplicitni uporabi modela procesa in določitvi optimalnih regulirnih signalov z

minimizacijo kriterijske funkcije razlike želenega in napovedanega odziva procesa v intervalu prihodnosti. Metoda vodenja je popularna predvsem zaradi enostavne razumljivosti njenih osnovnih principov in možnosti vključevanja omejitev v optimizacijski algoritem. Rezultati nelinearnega prediktivnega vodenja s prediktivnim funkcionalnim vodenjem tlaka procesa priprave plina pokažejo, da uporaba modela z Gaussovimi procesi, z vključitvijo omejitev velikosti variance predikcije v optimizacijski algoritem, zagotovi robustno in varno delovanje sistema vodenja v znanem, identificiranem območju. Procesa ne moremo pripeljati v neidentificirano področje, ki ga model ne opisuje ustrezno in kjer bi zato prediktivni algoritem lahko destabiliziral sistem vodenja.

Rezultati identifikacije in vodenja so bili pridobljeni v okolju, ki omogoča uporabniško prijazno eksperimentiranje iz okolja Matlab/Simulink. Sistem vodenja procesnega laboratorija vsebuje dva nivoja: zgornji nivo nadzornega vodenja s Factory Link SCADA nadzorno aplikacijo in spodnji nivo osnovnega in postopkovnega vodenja, realiziran na programabilnih logičnih krmilnikih (PLK) Mitsubishi. Vmesnik za komunikacijo okolja Matlab/Simulink s PLK-ji je bil vključen v sistem postopkovnega vodenja laboratorija z upoštevanjem razvoja sistema vodenja po konceptu življenskega cikla. Tako je omogočeno enostavno eksperimentiranje iz okolja Matlab/Simulink, ob zagotavljanju visoke stopnje varnosti delovanja polindustrijskega procesa, ki jo zagotavlja postopkovno vodenje izvedeno na nivoju programabilnih logičnih krmilnikov in SCADA aplikacije.

Abstract

The thesis addresses the use of a Gaussian processes theory for modeling, simulation and control of non-linear dynamic systems. Gaussian process is an example of a probabilistic, non-parametric model with uncertainty predictions. The theory is based on the Bayesian probabilistic modeling paradigm. Their use originates from the field of the statistics.

A Gaussian process is a collection of random variables which have a joint multivariate Gaussian distribution, defined by the mean values vector and the covariance matrix. Given the training set of the input and output data pairs, each output point is modeled by one normally distributed random variable of o joint multivariate distribution. Covariance matrix is modeled by the covariance function as a function of the input data points. Mean values vector (usually assumed to be zero) and the covariance function fully specify the Gaussian process. Parameters (the term "*hyperparameters*" is used) of the covariance function are learned using the training set of data. Predictive distribution of the output, given the new input, is a normally distributed random variable, where mean is taken to be predicted value and variance indicates the level of confidence in the prediction.

Analogy of Gaussian process models to artificial neural networks is the starting point for the approach to simulation of dynamical systems with Gaussian process models, where delayed values of model inputs and outputs are taken as the new model inputs. Uncertainty information of the previous model outputs is lost, when only mean values are propagated. Different approaches that incorporate also the previous predictions variances, when predicting new outputs, are described.

Dynamic system identification with Gaussian process model is shown on the gas-liquid separator example. Results of different uncertainty propagation approaches are compared.

Pressure of the gas-liquid separator is controlled by the nonlinear predictive control algorithm, with the incorporated GP model of the process. Predictive control is based on the explicit use of controlled process model and the determination of the optimal control signals by minimizing the objective function of the error between desired and predicted process response on the predictive interval in the future. Predictive control is widely used in industrial practice, mainly because of its simple underlying idea and natural incorporation of constraints in the optimization algorithm. Results of the nonlinear predictive control with Predictive Functional Control of the pressure show, that use of a Gaussian process model with included constraints on the prediction variance result in a robust and safe operation in the identified region. Process can not be driven to the unidentified regions, where the Gaussian process model does have the knowledge about the process, and could for that reason destabilize control system.

Real-time identification and control results were pursued in the environment that enables user-friendly experimentation with the process from the Matlab/Simulink application. Supervisory control of the process laboratory encompasses two levels: upper level with Factory Link SCADA application and lower procedural and basic control levels implemented in the Mitsubishi programmable logical controllers (PLCs). Interface, that enables communication of Matlab/Simulink application with the PLC, was incorporated in the procedural control of the process, with the respect to the methodology of development of control systems, based on the concept of the life cycle. High level operating security of semi-industrial process, obtained by procedural control implemented on the PLC and SCADA level, is combined with user-friendly Matlab/Simulink experimentation environment.

Vsebina

1. Uvod

Teorija Gaussovih procesov (GP) izhaja iz področja verjetnosti in statistike, njihova uporaba za identifikacijo nelinearnih dinamičnih sistemov pa predstavlja zelo mlado vejo raziskovanja. Kot osnovo za identifikacijo z GP se uporablja njihova podobnost z umetnimi nevronskimi mrežami. Rezultat predikcije z GP modelom je normalno porazdeljena gostota verjetnosti s srednjo vrednostjo in varianco, kar je tudi njihova prednost, saj tako, v nasprotju z drugimi nelinearnimi modeli, dobimo tudi informacijo o zaupanju v napovedano vrednost izhoda.

Namen naloge je identifikacija nelinearnega procesa priprave plina, ki je del polindustrijskega laboratorija Odseka za sisteme in vodenje na Inštitutu Jožef Stefan, z GP in realizacija nelinearnega prediktivnega vodenja na osnovi identificiranega GP modela. Pri prediktivnem vodenju, ki spada v družino regulacijskih metod na podlagi modela, lahko z uporabo GP modela in upoštevanjem zanesljivosti predikcije izboljšamo kakovost vodenja, ki je neposredno odvisna od kakovosti modela v delovnem območju. Namen naloge je tudi razvoj eksperimentalnega vmesnika za eksperimentiranje in vodenje procesa priprave plina s programskim orodjem Matlab/Simulink in njegova vključitev v obstoječ sistem postopkovnega vodenja procesnega laboratorija.

Delo je sestavljeno iz osmih poglavij. Uvodu v prvem poglavju sledi predstavitev teorije GP v drugem poglavju. Predstavljene so osnove Bayesovega verjetnostnega modeliranja, iz katerega preidemo na predstavitev GP in modeliranje statičnih in dinamičnih procesov z GP modeli. Predstavljeni so različni načini upoštevanja informacije o verjetnostni porazdelitvi preteklih izhodov GP modela pri simulaciji dinamičnih sistemov, kjer kot vhode v model uporabljamo zakasnjene vrednosti vhodnega in izhodnega signala.

Tretje poglavje predstavlja osnovne principe in prednosti uporabe metod prediktivnega vodenja. Prediktivno vodenje temelji na eksplicitni uporabi modela procesa. Podan je tudi opis skupnih značilnosti metod prediktivnega funkcionalnega vodenja, ki se je zaradi svoje enostavnosti in učinkovitosti uveljavilo v inženirski praksi.

V četrtem poglavju so strnjeni opis procesnega laboratorija in posameznih tehnoloških sklopov - procesnih enot, z opisom procesne opreme in različnih možnih načinov zajemanja podatkov in vplivanja na proces prek aktuatorjev. Podrobneje je opisan proces priprave plina, za katerega je bil razvit vmesnik za eksperimentiranje iz okolja Matlab/Simulink. Vmesnik, razvit z upoštevanjem življenskega cikla sistema procesnega vodenja, je bil vključen v obstoječ sistem postopkovnega vodenja laboratorija, realiziranem na nivoju PLK-SCADA. Tako razvito okolje, opisano v petem poglavju, omogoča enostavno eksperimentiranje ob zagotavljanju visoke stopnje varnosti delovanja procesa. Z uporabo vmesnika sta bila realizirana tako identifikacija kot vodenje procesa priprave plina, katerih rezultati so predstavljeni v naslednjih dveh poglavjih.

Šesto poglavje predstavi rezultate identifikacije procesa priprave plina z Gaussovimi procesi. Vključuje opis učenja hiperparametrov kovariančne funkcije GP modela in primerjavo rezultatov simulacij odzivov modela z različnimi načini propagiranja verjetnostnih porazdelitev preteklih izhodov modela.

Na osnovi identificiranega GP modela je bilo realizirano nelinearno prediktivno vodenje tlaka ločevalnika, značilnosti prediktivnega regulatorja s predstavitvijo in komentarji rezultatov vodenja, so predstavljeni v sedmem poglavju.

V zaključku je povzeto predstavljeno delo, podani so glavni rezultati.

Delo je dopolnjeno s tremi podatki. Dodatek A vsebuje opis metodologije načrtovanja sistemov vodenja z upoštevanjem koncepta življenskega cikla, dodatek B vsebuje specifikacije sistema vodenja procesa priprave plina, dopolnjene z delom za vključitev eksperimentalnega vmesnika. Podrobnosti o izvedbi vmesnika pa so opisane v dodatku C.

2. Gaussovi procesi

Poglavje najprej predstavi osnovne pojme iz teorije verjetnostnega modeliranja, na katerih temelji tudi teorija Gaussovih procesov, nato pa v nadaljevanju iz opisa modeliranja statičnih nelinearnih funkcij preide v opis identifikacije in simulacije nelinearnih dinamičnih sistemov z uporabo GP.

2.1 Osnove verjetnostnega modeliranja

2.1.1 Naključne spremenljivke

Naključna spremenljivka je spremenljivka, katere vrednost je odvisna od naključja. O naključnem procesu (ang. *stochastic process*) govorimo, ko imamo več neodvisnih (časovno, prostorsko itd) realizacij naključne spremenljivke. Porazdelitve vrednosti posameznih realizacij naključne spremenljivke so lahko poljubne, opišemo jih s porazdelitvenim zakonom in zalogo vrednosti, s čimer je naključna spremenljivka tudi popolnoma določena. Glede na zalogo vrednosti ločimo diskretne in zvezne naključne spremenljivke, glede na obliko porazdelitvenega zakona pa različne standardne (enakomerna, normalna ali Gaussova, Studentova itd) in nestandardne porazdelitve [20]. Splošna oblika porazdelitvenega zakona za naključno spremenljivko X je porazdelitvena funkcija ali distribucija $F(x)$:

$$F(x) = P(X < x) \quad (-\infty < x < \infty), \tag{2.1}$$

kjer je $P(X<x)$ verjetnost dogodka, da naključna spremenljivka X zavzame vrednost manjšo od x. V primeru zvezne naključne spremenljivke (zaloga vrednosti spremenljivke je zvezna množica) lahko porazdelitveno funkcijo spremenljivke X zapišemo tudi v obliki:

$$F(x) = \int_{-\infty}^{x} p(t)dt \tag{2.2}$$

kjer je *p(x)* gostota verjetnosti ali verjetnostna gostota (ang. *PDF-probability density function*). Verjetnost, da zvezna naključna spremenljivka *X* zavzame vrednost na intervalu [*a, b*] zapišemo:

$$P(a \le x \le b) = F(b) - F(a) = \int_a^b p(x)dx \tag{2.3}$$

Pri analizi porazdelitev naključnih spremenljivk si pomagamo z različnimi številskimi karakteristikami. Najpomembnejše so :

- Matematično upanje ali povprečna vrednost naključne spremenljivke *X*, *E(X)*:

$$E(X) = \int_{-\infty}^{\infty} x\, p(x)dx \tag{2.4}$$

- Varianca ali disperzija slučajne spremenljivke *D(X)* je definirana kot povprečje kvadratov odstopanj spremenljivke *X* od njene povprečne vrednosti *E(X)*:

$$\mathrm{var}(X) = E((X - E(X)^2) = \int_{-\infty}^{\infty} (x\text{-}E(X))^2\, p(x)dx = E(X^2) - (E(X))^2 \tag{2.5}$$

- Standardna deviacija σ_X je definirana kot pozitivni kvadratni koren variance naključne spremenljivke *X*:

$$\sigma_X = \sigma(X) = +\sqrt{\mathrm{var}(X)} \tag{2.6}$$

- Kovarianca slučajnih spremenljivk *X* in *Y* je matematično upanje produkta odklonov spremenljivk od njunih povprečnih vrednosti in je mera za linearno odvisnost spremenljivk:

$$\mathrm{cov}(X,Y) = E((X - E(X))\ (Y - E(Y)) \tag{2.7}$$

- Korelacijski koeficient slučajnih spremenljivk *X* in *Y* je s standardnima deviacijama σ_X in σ_Y normirana kovarianca spremenljivk:

$$r(X,Y)=\frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \tag{2.8}$$

Slučajni spremenljivki sta nekorelirani (linearno neodvisni) natanko takrat, kadar je *r(X,Y)=0*.

Normalno ali Gaussovo porazdeljeno spremenljivko, definirano z gostoto verjetnosti (2.9) na kratko zaznamujemo z $\mathcal{N}(\mu,\sigma)$, kjer je μ srednja vrednost, σ pa standardna deviacija naključne spremenljivke.

$$p(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{2.9}$$

2.1.2 Naključni vektorji

Naključni vektor je vektor, katerega komponente so naključne spremenljivke. Če imamo dve ali več med seboj povezanih naključnih spremenljivk, njihovo realizacijo imenujemo naključni proces (ang. *stochastic process*).

$$\mathbf{X}=(X_1,X_2,...,X_n) \tag{2.10}$$

Podobno kot za naključno spremenljivko, lahko za naključni vektor zapišemo porazdelitveni zakon v obliki:

$$F(x_1,x_2,...,x_n)=P(X_1<x_1,X_2<x_2,...,X_n<x_n) \tag{2.11}$$

Če so komponente naključnega vektorja zvezne naključne spremenljivke, potem velja:

$$F(x_1,\ldots,x_n)=\int_{-\infty}^{x_1}\ldots\int_{-\infty}^{x_n}p(x_1,\ldots x_n)dx_1\ldots dx_n \tag{2.12}$$

kjer je $p(x_1,\ldots x_n)$ n-dimenzionalna gostota verjetnosti (nenegativna zvezna funkcija). Verjetnost, da naključni vektor zavzame vrednost na določenem intervalu imenujemo skupna ali totalna verjetnost in jo zapišemo:

$$P(a_1 \le x_1 \le b_1,\ldots,a_k \le x_n \le b_n) = \int_{a_1}^{b_1} \ldots \int_{a_n}^{b_n} p(x_1,\ldots x_n)dx_1 \ldots dx_n \tag{2.13}$$

Porazdelitve posameznih komponent naključnega vektorja imenujemo tudi robne ali marginalne porazdelitve:

$$p(\infty,\ldots\infty,x_i,\infty,\ldots,\infty) = p_i(x_i) = \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} p(x_1,\ldots x_k)dx_1 \ldots dx_{i-1}dx_{i+1} \ldots dx_n \tag{2.14}$$

Pogojno porazdelitveno funkcijo komponente X_i, ko komponenta X_j zavzame vrednost x_j zapišemo:

$$F(X_i|X_j) = F(X_i|X_j = x_j) = \lim_{h\to 0} \frac{F(X_i,x_j+h) - F(X_i,x_j)}{F(x_j+h) + F(x_j)} \tag{2.15}$$

Enako velja za pogojno verjetnostno gostoto:

$$p(X_i|X_j) = \frac{p(X_i,X_j)}{p(X_j)} \tag{2.16}$$

kjer je $P(X_i, X_j)$ skupna verjetnostna gostota in $P(X_j)$ marginalna verjetnostna gostota spremenljivke (komponente) X_j naključnega vektorja.

Najpomembnejši med številskimi karakteristikami naključnega vektorja sta [20]:

- Matematično upanje ali povprečna vrednost naključnega vektorja **X** je vektor $E(\mathbf{X})$, sestavljen iz povprečnih vrednosti posameznih komponent slučajnega vektorja:

$$E(\mathbf{X}) = (E(X_1),E(X_2),\ldots,E(X_n)) \tag{2.17}$$

- Varianca ali disperzija slučajnega vektorja **X** je definirana s kovariančno matriko $\mathbf{\Sigma}(\mathbf{X})$, sestavljeno iz kovarianc med posameznimi komponentami vektorja:

$$\boldsymbol{\Sigma}(\mathbf{X}) = \begin{vmatrix} K_{11} & K_{12} & \dots & K_{1n} \\ K_{21} & K_{22} & \dots & K_{2n} \\ \dots & \dots & \dots & \dots \\ K_{n1} & K_{n2} & \dots & K_{nn} \end{vmatrix} \tag{2.18}$$

Velja:

1. $K_{ij} = K(X_i, X_j)$.
2. $K_{ij} = K_{ji}$, $\boldsymbol{\Sigma}(\mathbf{X})$ je simetrična.
3. $\mathrm{K_{ii}} = \mathrm{var}(X_i)$, diagonalni elementi $\boldsymbol{\Sigma}(\mathbf{X})$ predstavljajo varianco posamezne komponente.
4. Iz lastnosti variance, ki je vedno nenegativno število, sledi, da je $\boldsymbol{\Sigma}(\mathbf{X})$ nenegativno definitna matrika.

Normalno ali Gaussovo porazdeljen naključni vektor je določen z verjetnostno gostoto:

$$p(\mathbf{x}) = (2\pi)^{-\frac{n}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} e^{\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})\right)} \tag{2.19}$$

kjer je:

n dimenzija naključnega vektorja **X,**

μ vektor srednjih vrednosti naključnega vektorja **X,**

Σ kovariančna matrika naključnega vektorja **X.**

Normalna porazdelitev je funkcija vektorja srednjih vrednosti in kovariančne matrike, zato pogosto uporabljamo skrajšani zapis $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Dvodimenzionalno normalno porazdelitev lahko zapišemo v obliki:

$$p(x_1, x_2) = \frac{1}{2\pi \cdot \sigma_1 \sigma_2 \sqrt{1-r^2}} e^{-\frac{1}{2(1-r^2)}\left(\frac{(x_1-\mu_1)^2}{\sigma_1^2} - \frac{2r(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1 \sigma_2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2}\right)} \tag{2.20}$$

kjer r predstavlja korelacijski koeficient med naključnima spremenljivkama. V tem primeru je:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & r\sigma_1\sigma_2 \\ r\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} \tag{2.21}$$

2.1.3 Bayesovo modeliranje

Pri modeliranju statičnih funkcij, kjer imamo učno množico, sestavljeno iz n vektorjev D-dimenzionalnih neodvisnih vhodnih spremenljivk $\mathbf{X}$ in pripadajoč vektor n izhodnih točk $\boldsymbol{t}$, podanih v obliki $\mathcal{D} = \{\mathbf{X}, \mathbf{t}\} = \{(\boldsymbol{x}_i, t_i), i = 1 \ldots n\}$, želimo najti funkcijo f, ki naj čimbolje opisuje relacijo med vhodno-izhodnimi pari $(\mathbf{x}_i, t_i)$. Z vidika Bayesovega verjetnostnega modeliranja, modeliranje statičnega procesa z nelinearno funkcijo $f = f(\mathbf{x}, \boldsymbol{\omega})$, katere parametre $\boldsymbol{\omega}$ določimo glede na podatke $\mathcal{D}$, lahko predstavimo v obliki Bayesovega teorema [30]:

$$p(\boldsymbol{\omega}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\omega})\, p(\boldsymbol{\omega})}{p(\mathcal{D})}, \tag{2.22}$$

kjer je:

$p(\boldsymbol{\omega})$	"*a-priori*" verjetnostna porazdelitev, ki vsebuje predhodno znanje o parametrih funkcije (ang. *prior*), ki ponavadi predpostavljajo zveznost, frekvenčno razporeditev moči itd,
$p(\mathcal{D}\|\boldsymbol{\omega})$	verjetnostna porazdelitev učne množice pri danih parametrih funkcije (ang. *likelihood*),
$p(\mathcal{D})$	verjetnostna porazdelitev učne množice (ang. *evidence*), v primeru Bayesovega modeliranja služi kot normalizacijska konstanta,
$p(\boldsymbol{\omega}\|\mathcal{D})$	"*a-posteriori*" verjetnostna porazdelitev parametrov $\boldsymbol{\omega}$ pri dani učni množici $\mathcal{D}$ (ang. *posterior*).

Bayesov teorem združuje predhodno znanje o parametrih $p(\boldsymbol{\omega})$ z znanjem, dobljenim v obliki učne množice $\mathcal{D}$. Z upoštevanjem zapisa $\mathcal{D} = \{\mathbf{X}, \mathbf{t}\}$ lahko Bayesov teorem zapišemo v obliki:

$$p(\boldsymbol{\omega}|\mathbf{X},\mathbf{t}) = \frac{p(\mathbf{t}|\boldsymbol{\omega},\mathbf{X})\,p(\boldsymbol{\omega})}{p(\mathbf{t}|\mathbf{X})}, \tag{2.23}$$

Primer določanja parametrov $\boldsymbol{\omega}$ na podlagi verjetnostnega zapisa, pri predpostavkah o Gaussovi porazdelitvi $p(\mathbf{t}|\boldsymbol{\omega},\mathbf{X})$ in $p(\boldsymbol{\omega})$, je s kriterijsko funkcijo:

$$J(\boldsymbol{\omega}) = -\log\left(p(\mathbf{t}|\boldsymbol{\omega},\mathbf{X})\,p(\boldsymbol{\omega})\right) \tag{2.24}$$

Z minimizacijo kriterijske funkcije določimo maksimum pogojne verjetnostne gostote $p(\boldsymbol{\omega}|\mathbf{X},\mathbf{t})$ in s tem oceno najverjetnejše vrednosti parametrov $\boldsymbol{\omega}$ [30].

Za predikcijo porazdelitve izhoda t^* pri vhodu $\mathbf{x}^*$ pri poznavanju $p(\boldsymbol{\omega}|\mathbf{X},\mathbf{t})$ pa je potrebno izračunati integral (marginalizacijo) po parametrih:

$$p\left(t^*\middle|\mathbf{x}^*,\mathbf{X},\mathbf{t}\right) = \int p\left(t^*\middle|\mathbf{x}^*,\mathbf{X},\mathbf{t},\boldsymbol{\omega}\right)p(\boldsymbol{\omega}|\mathbf{X},\mathbf{t})\,d\boldsymbol{\omega} \tag{2.25}$$

2.2 Modeliranje z Gaussovimi procesi

Z neparametričnimi metodami določamo predikcijo izhoda brez eksplicitnega parametriranja funkcije f. Tak primer je tudi modeliranje z Gaussovimi procesi (GP). GP je naključen proces, katerega naključne spremenljivke so medsebojno porazdeljene po normalnem porazdelitvenem zakonu, določen je z vektorjem srednjih $\boldsymbol{\mu}$ vrednosti in kovariančno funkcijo $C(\mathbf{x}_p,\mathbf{x}_q)$, prek katere je definirana njegova kovariančna matrika $\mathbf{K}$ [8][1]. Pri modeliranju z GP brez parametriranja funkcije f predpostavimo n-dimenzionalno normalno porazdelitev $p(\mathbf{y})$:

$$p(\mathbf{y}) = \frac{1}{Z}e^{\left(-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{y}-\boldsymbol{\mu})\right)}, \tag{2.26}$$

kjer je $\mathbf{K}$ kovariančna matrika normalne porazdelitve in Z normalizacijska konstanta. Predpostavimo, da vsak vzorec $(\mathbf{x}_i,t_i)$ učne množice $\mathcal{D} = \{\mathbf{X},\mathbf{t}\} = \{(\mathbf{x}_i,t_i), i = 1\ldots n\}$ določa posamezno naključno spremenljivko y_i večdimenzionalne normalne porazdelitve naključnega vektorja $\mathbf{y} = (y_1,\ldots,y_n) = \mathcal{N}(\boldsymbol{\mu},\mathbf{K})$. Vektor srednjih vrednosti je določen kot funkcija vhodov

učne množice, $\boldsymbol{\mu} = \boldsymbol{\mu}(\mathbf{X})$. Kovariančno matriko $\mathbf{K}$ določa kovariančna funkcija $C = C(\boldsymbol{x}_p, \boldsymbol{x}_q)$, prav tako funkcija vhodov učne množice. V primeru srednje vrednosti vektorja izhodnih točk učne množice $\mu(\mathbf{t}) = 0$, lahko privzamemo tudi GP z vektorjem srednjih vrednosti $\boldsymbol{\mu} = \mathbf{0}$, kar bomo v nadaljevanju zaradi poenostavitev izpeljav vedno predpostavili in primerno normirali vektor $\mathbf{t}$. V tem primeru GP določa samo kovariančna funkcija.

2.2.1 Kovariančna funkcija

Vrednost kovariančne funkcije $C(\mathbf{x}_p, \mathbf{x}_q)$ izraža korelacijo med posameznima izhodoma y_p in y_q modela, obravnavana kot dve medsebojno povezani naključni spremenljivki, glede na vhoda $\mathbf{x}_p$, $\mathbf{x}_q$:

$$\operatorname{cov}(y_p, y_q) = C(\mathbf{x}_p, \mathbf{x}_q) \tag{2.27}$$

V splošnem je lahko kovariančna funkcija poljubna funkcija, ki generira nenegativno kovariančno matriko za poljuben nabor vhodnih vektorjev. S stališča modeliranja je primerna izbira take kovariančne funkcije, ki močneje korelira izhodne točke, ki so si v vhodnem prostoru bližje [33] [2] [34] [37]. Pri predpostavki o stacionarnosti procesa (kovarianca med dvema točkama je odvisna samo od medsebojne razdalje in neodvisna od premika v prostoru), je najpogosteje uporabljena Gaussova kovariančna funkcija:

$$C(\mathbf{x}_p, \mathbf{x}_q) = v\, \mathrm{e}^{-\frac{1}{2}\sum_{d=1}^{\mathrm{D}} w_d \left(\mathbf{x}_p^d - \mathbf{x}_q^d\right)} + v_0 \tag{2.28}$$

Enakovredna vektorska oblika zapisa je:

$$C(\mathbf{x}_p, \mathbf{x}_q) = v\ \mathrm{e}^{-\frac{1}{2}\left(\mathbf{x}_p - \mathbf{x}_q\right)^{\mathrm{T}} \mathbf{W}^{-1} \left(\mathbf{x}_p - \mathbf{x}_q\right)} + v_0 \tag{2.29}$$

kjer je $$\mathbf{W} = \begin{bmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_D \end{bmatrix}$$

Parametre kovariančne funkcije $\boldsymbol{\Theta} = (v, v_0, w_1, ..., w_D)$ imenujemo hiperparametre (uteži). Za zagotovitev pogoja nenegativne definitnosti kovariančne matrike **K** morajo biti njihove vrednosti pozitivne. S parametrom v_0 modeliramo beli šum $\varepsilon \approx N(0, v_0)$ na izhodu sistema, parameter v je skalirni faktor kovariance, parametri w_i pa določajo relativno vlogo (utež) razdalje po posamezni vhodni spremenljivki x_i pri celotni vrednosti kovariance.

Druge oblike kovariančnih funkcij so opisane v literaturi [33], [30], [8] . Termin Gaussovi procesi ni povezan z uporabo Gaussove kovariančne funkcije za njihovo modeliranje.

2.2.2 Določanje hiperparametrov kovariančne funkcije

Najbolj verjetne vrednosti hiperparametrov pri določeni kovariančni funkciji določimo prek "*a-posteriori*" zapisa verjetnostne porazdelitev parametrov:

$$p(\boldsymbol{\Theta}|\mathbf{X},\mathbf{t}) = \frac{p(\mathbf{t}|\boldsymbol{\Theta},\mathbf{X})\,p(\boldsymbol{\Theta})}{p(\mathbf{t}|\mathbf{X})} \tag{2.30}$$

Optimalne vrednosti parametrov določimo z iskanjem največje vrednosti logaritma porazdelitve $p(\mathbf{t}|\boldsymbol{\Theta},\mathbf{X})$, ki je logaritem Gaussovega procesa [8]:

$$log\left(p(\mathbf{t}|\boldsymbol{\Theta},\mathbf{X})\right) = -\frac{1}{2}\log\left(|\mathbf{K}|\right) - \frac{1}{2}\mathbf{t}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{t} \quad - \frac{n}{2}\log(2\pi) \tag{2.31}$$

Parcialni odvod logaritma porazdelitve po hiperparametrih se glasi:

$$\frac{\partial \mathrm{l}}{\partial \theta_\mathrm{i}} = -\frac{1}{2}\mathrm{Tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_\mathrm{i}}\right) + \frac{1}{2}\mathbf{t}^{\mathrm{T}}\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_\mathrm{i}}\mathbf{K}^{-1}\mathbf{t} \tag{2.32}$$

V literaturi se ta metoda imenuje metoda največje podobnosti (ang. *maximum likelihood - ML*). Za iskanje minimuma se lahko uporablja katerakoli metoda. Ena izmed možnih je uporaba metode konjugiranih gradientov zaradi enostavnih analitičnih izračunov parcialnih odvodov (2.32). Predstavljena metoda je občutljiva na začetno izbiro hiperparametrov (padec v lokalni minimum), poleg tega pa je računsko zahtevna, saj vsak korak optimizacije potrebuje izračun inverzne kovariančne matrike dimenzije *n* x *n*, kjer je *n* število podatkov v učni množici.

2.2.3 Predikcija z Gaussovimi procesi

Za predikcijo porazdelitve izhoda y^* pri novem vhodu $\mathbf{x}^*$ velja:

$$p\left(y^*|\mathbf{y}\right)=\frac{p\left(\mathbf{y},y^*\right)}{p\left(\mathbf{y}\right)} \tag{2.33}$$

Ob upoštevanju enačbe (2.26) izpeljemo:

$$p\left(y^*|\mathbf{y}\right)=\frac{Z_n}{Z_{n+1}}e^{\left(-\frac{1}{2}\left(\mathbf{y},y^*\right)^{\mathrm{T}}\mathbf{K}_{n+1}^{-1}\left(\mathbf{y},y^*\right)-\mathbf{y}^{\mathrm{T}}\mathbf{K}_n^{-1}\mathbf{y}\right)} \tag{2.34}$$

V zgornji enačbi je kovariančna matrika $\mathbf{K}_n$ (reda $n\cdot n$) določena na podlagi izbrane kovariančne funkcije $C\left(\mathbf{x}_p,\mathbf{x}_q\right)$ in n vhodnih vektorjev učne množice $\mathcal{D}$, kovariančna matrika $\mathbf{K}_{n+1}$ (reda $(n+1)\cdot(n+1)$), pa na podlagi iste učne množice, razširjene z novim vhodom $\mathbf{x}^*$. Med njima velja relacija:

$$\mathbf{K}_{n+1}=\begin{bmatrix}\begin{bmatrix}&\mathbf{K}_n&\end{bmatrix}&\begin{bmatrix}\mathbf{k}\end{bmatrix}\\ \begin{bmatrix}\mathbf{k}^{\mathrm{T}}\end{bmatrix}&[k]\end{bmatrix} \tag{2.35}$$

Vektor $\mathbf{k}=\left(C\left(\mathbf{x}_1,\mathbf{x}^*\right)\ \ldots\ C\left(\mathbf{x}_n,\mathbf{x}^*\right)\right)$ vsebuje korelacije izhoda y^* z izhodi modela $\mathbf{y}$, skalar $k=C\left(\mathbf{x}^*,\mathbf{x}^*\right)$ pa varianco izhoda y^*. Enačbo (2.34) z upoštevanjem relacij v zapisu (2.35) lahko izrazimo v obliki:

$$p(y^*|\mathbf{y})=\frac{1}{Z}e^{\left(-\frac{\left(y^*-\mu_{y^*}\right)}{2\sigma_{y^*}^2}\right)} \tag{2.36}$$

$$\mu_{y^*}=\mathbf{k}^{\mathrm{T}}\mathbf{K}_n^{-1}\mathbf{t} \tag{2.37}$$

$$\sigma_{y^*}^2=k-\mathbf{k}^{\mathrm{T}}\mathbf{K}_n^{-1}\mathbf{k} \tag{2.38}$$

V prejšnjih enačbah sta podana izraza za izračun srednje vrednosti μ_{y^*} in variance $\sigma^2_{y^*}$ izhoda y^*, ki je normalna porazdelitev, za napovedovanje vrednosti izhoda procesa na podlagi vhodnega vektorja $\mathbf{x}^*$. Pričakovana vrednost izhoda y^* je tako enaka srednji vrednosti porazdelitve μ_{y^*}. Na vektor $\mathbf{k}^{\mathrm{T}}\mathbf{K}_{\mathrm{n}}^{-1}$ lahko gledamo kot na vektor uteži, ki določa uteženost izhodov [30] [31] [38]. Če se nov vhod precej razlikuje od učnih, bo izraz $\mathbf{k}^{\mathrm{T}}\mathbf{K}_{\mathrm{n}}^{-1}\mathbf{k}$ majhen, varianca $\sigma^2_{y^*}$ pa velika.

Zapis za $p(\mathbf{y})$ v obliki Gaussovega procesa (2.26) velja v primeru znanih podatkov $\mathcal{D} = \{\mathbf{X}, \mathbf{t}\}$, znane strukture kovariančne funkcije C in znanih hiperparametrov $\boldsymbol{\Theta}$. V splošnem je ustrezen zapis:

$$p(\mathbf{y}|\mathbf{X}, \mathbf{C}, \boldsymbol{\Theta}) = \frac{1}{Z} e^{\left(-\frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y}\right)} \tag{2.39}$$

Analogen pristop k predikciji porazdelitve y^* pri novem vhodu $\mathbf{x}^*$ pri določeni strukturi kovariančne funkcije in nedoločenih hiperparametrih $\boldsymbol{\Theta}$ predstavlja integriranje (marginalizacija) po hiperparametrih:

$$p\left(y^* \middle| \mathbf{x}^*, \mathcal{D}\right) = \int p\left(y^* \middle| \boldsymbol{x}^*, \mathcal{D}, \boldsymbol{\Theta}\right) p(\boldsymbol{\Theta}|\mathcal{D}) d\boldsymbol{\Theta} \tag{2.40}$$

Ker je zgornji integral običajno analitično neizračunljiv, obstajata dve alternativni rešitvi [30].

Prva predlagana rešitev je z numerično integracijo z metodo MCMC (Markov Chain Monte Carlo) z vključitvijo a-priori znanja o porazdelitvi parametrov [33].:

$$p\left(y^* \middle| \mathbf{x}^*, \mathcal{D}\right) \cong \frac{1}{T}\sum_{t=1}^{T} p\left(y^* \middle| \mathbf{x}^*, \mathcal{D}, \boldsymbol{\Theta}_t\right) \tag{2.41}$$

Vzorci $\boldsymbol{\Theta}_t$ so porazdeljeni po oceni verjetnostne porazdelitve $p(\boldsymbol{\Theta}|\mathcal{D})$.

Druga predlaga rešitev je aproksimacija integrala z uporabo najbolj verjetnih vrednosti hiperparametrov $\boldsymbol{\Theta}_y$:

$$p\left(y^*\middle|\mathbf{x}^*,\mathcal{D}\right)\cong p\left(y^*\middle|\mathbf{x}^*,\mathcal{D},\Theta_v\right) \tag{2.42}$$

Aproksimacijo naredimo s predpostavko, da je porazdelitev $p(\Theta|\mathbf{X},\mathbf{Y})$ izrazito porazdeljena okoli svoje srednje vrednosti. Splošno daje zadovoljive rezultate v primerjavi s pravimi vrednostmi porazdelitve. Rešitev je ekvivalentna predikciji po enačbi (2.36).

2.3 Identifikacija nelinearnih dinamičnih sistemov z Gaussovimi procesi

Teoretično ozadje modeliranja statičnih nelinearnih funkcij z GP je bilo opisano v predhodnih podpoglavjih. Izhodišče za identifikacijo dinamičnih sistemov z GP je njihova podobnost z umetnimi nevronskimi mrežami pri modeliranju statičnih nelinearnosti z uporabo preslikave vhodov v izhode. Za namen modeliranja dinamičnih sistemov z enokoračno predikcijo kot vhode v model uporabimo zakasnjene vrednosti vhodnega in izhodnega signala. Izhod nelinearnega dinamičnega sistema reda *n* tako ob času $t = k$ tako zapišemo z diskretno funkcijo:

$$y(k)=f\left(y(k-1),\ldots,y(k-n),u(k-1),\ldots,u(k-n)\right) \tag{2.43}$$

Vektor $\left(y(k-1),\ldots,y(k-n),u(k-1),\ldots,u(k-n)\right)$ imenujemo vektor regresorjev. Prva predpostavka, ki izhaja iz analogije z identifikacijo z umetnimi nevronskimi mrežami, je uporaba enokoračne predikcije z uporabo predhodnih srednjih vrednosti izhoda modela [16] [17] [27]. Na tak način zavržemo informacijo o porazdelitvi preteklih izhodov modela, s tem pa tudi informacijo o zaupanju v vrednosti preteklih izhodov, ki jih uporabimo kot nove vhode v model [3] [9] [13]. V nadaljevanju je predstavljen način, kako v model dinamičnega nelinearnega sistema vključimo tudi to informacijo. Model tako poleg opisovanja dinamičnih lastnosti nelinearnega sistema daje tudi informacijo o zaupanju v predikcijo [10] [11] [12] [23].

Enačbi (2.37, EQ2.38) za srednjo vrednost in varianco porazdelitve predikcije $p(y^*|\mathbf{y})$ pri vhodu $\mathbf{x}^*$ lahko zapišemo v obliki [11] [3] [8]:

$$\mu_{y^*}=\mu\left(\mathbf{x}^*\right)=\mathbf{k}(\mathbf{x}^*)^{\mathrm{T}}\boldsymbol{\beta}=\sum_{i=1}^{N}\beta_i C\left(\mathbf{x}^*,\mathbf{x}_i\right) \tag{2.44}$$

$$\sigma^2_{y^*} = \sigma^2\left(\mathbf{x}^*\right) = C\left(\mathbf{x}^*,\mathbf{x}^*\right) - \sum_{i,j=1}^{N} K^{-1}_{i,j} C\left(\mathbf{x}^*,\mathbf{x_i}\right) C\left(\mathbf{x}^*,\mathbf{x_j}\right) \tag{2.45}$$

kjer je $\boldsymbol{\beta} = \mathbf{K}^{-1}\mathbf{t}$ in $C\left(\boldsymbol{x}^*,\boldsymbol{x_i}\right)$ kovarianca izhodov y^* in y_i. Za predikcijo $p(y^*|\mathbf{y})$ pri vhodu $\mathbf{x}^*$ kot naključni spremenljivki $p\left(\mathbf{x}^*\right) = \boldsymbol{N}\left(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}\right)$ je potrebno izračunati integral (marginalizacijo) po spremenljivki $\mathbf{x}^*$:

$$p(y^*|\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}}) = \int p(y^*|\boldsymbol{x}^*, \boldsymbol{\mathcal{D}}) p(\mathbf{x}^*) d\mathbf{x}^* \tag{2.46}$$

V splošnem primeru porazdelitve je $p(y^*|\mathbf{x}^*, \boldsymbol{\mathcal{D}})$ nelinearna funkcija spremenljivke $\mathbf{x}^*$, zgornji integral pa zato analitično neizračunljiv. Pri njegovem izračunu si zato pomagamo z numeričnimi ali analitičnimi aproksimacijami.

2.3.1 Numerična aproksimacija z metodo MCMC

Numerična aproksimacija je rešitev z uporabo numerične integracije z metodo MCMC (Markov Chain Monte Carlo):

$$p(y^*|\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}}) \cong \frac{1}{T} \sum_{t=1}^{T} p\left(y^* \middle| \mathbf{x}_t^*, \boldsymbol{\mathcal{D}} \right) \tag{2.47}$$

Vzorci $\mathbf{x}_t^*$ so porazdeljeni po verjetnostni porazdelitvi $p\left(\mathbf{x}^*\right)$ [11]..

2.3.2 Analitična aproksimacija s predpostavko o normalni porazdelitvi predikcije

Analitične aproksimacije temeljijo na predpostavki o normalno porazdeljeni predikciji $p(y^*|\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}})$ tudi v primeru vhoda $\boldsymbol{x}^*$ kot naključni spremenljivki [11]. Ob tej predpostavki nas zanimata samo srednja vrednost in varianca predikcije y^*. Srednjo vrednost $m\left(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}\right)$ kot pričakovano vrednost predikcije y^* izračunamo:

$$m\left(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}\right) = E\left(y^*|\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}}\right) = \int\int y^* p(y^*|\mathbf{x}^*, \boldsymbol{\mathcal{D}}) p(\mathbf{x}^*)\, p(y^*) d\mathbf{x}^* dy^* \tag{2.48}$$

Z upoštevanjem $\int y^* p(y^* | \mathbf{x}^*, \boldsymbol{\mathcal{D}}) dy = \mu(\mathbf{x}^*)$ zapišemo:

$$m(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = E_x(\mu(\mathbf{x}^*)) = E(y^* | \mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}}) = \int\int \mu(\mathbf{x}^*) p(\mathbf{x}^*) d\mathbf{x}^* \tag{2.49}$$

Podobno izračunamo varianco $v(\mathbf{u}, \boldsymbol{\Sigma}_x)$ predikcije y^*:

$$v(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = \mathrm{var}(y^* | \mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}}) = E\left(y^{*2} | \mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}} \right) - \left(E(y^* | \mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}})\right)^2 \tag{2.50}$$

Z upoštevanjem $\int y^{*2} p(y^* | \mathbf{x}^*, \boldsymbol{\mathcal{D}}) dy = \sigma^2(\mathbf{x}^*) + \mu(\mathbf{x}^*)^2$ zapišemo:

$$v(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = \mathrm{var}(y^* | \mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\mathcal{D}}) = \int \sigma^2(\mathbf{x}^*) p(\mathbf{x}^*) d\mathbf{x}^* + \int \mu(\mathbf{x}^*)^2 p(\mathbf{x}^*) d\mathbf{x}^* - m(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}})^2 \tag{2.51}$$

Simboličen zapis variance $v(\mathbf{u}, \boldsymbol{\Sigma}_x)$ z izrazi za izračun srednjih vrednosti je:

$$v(\boldsymbol{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = E_x(\sigma^2(\boldsymbol{x}^*)) + E_x\left(\mu(\boldsymbol{x}^*)^2\right) - \left(E_x(\mu(\boldsymbol{x}^*))\right)^2 \tag{2.52}$$

Po nadomestitvi $\mu(\boldsymbol{x}^*)$ in $\sigma^2(\boldsymbol{x}^*)$ z enačbama (2.44, EQ2.45) sta srednja vrednost $m(\mathbf{u}, \boldsymbol{\Sigma}_x)$in varianca predikcije $v(\mathbf{u}, \boldsymbol{\Sigma}_x)$ porazdelitve izhoda y^* pri naključnem vhodu $\mathbf{x}^*$, $p(\mathbf{x}^*) \approx \boldsymbol{\mathcal{N}}(\mathbf{u}, \boldsymbol{\Sigma}_x)$, določena z izrazi:

$$E_x(\mu(\mathbf{x}^*)) = \sum_{i=1}^{N} \beta_i \int C(\mathbf{x}, \mathbf{x}_i) p(\mathbf{x}) d\mathbf{x} \tag{2.53}$$

$$E_x(\sigma^2(\mathbf{x}^*)) = C(\mathbf{u}, \mathbf{u}) - \sum_{i,j=1}^{N} K_{ij}^{-1} \int C(\mathbf{x}, \mathbf{x}_i) C(\mathbf{x}, \mathbf{x}_j) p(\mathbf{x}) d\mathbf{x} \tag{2.54}$$

$$E_x\left(\mu(\mathbf{x}^*)^2\right) = \sum_{i,j=1}^{N} \beta_i \beta_j \int C(\mathbf{x}, \mathbf{x}_i) C(\mathbf{x}, \mathbf{x}_j) p(\mathbf{x}) d\mathbf{x} \tag{2.55}$$

V prejšnjih enačbah je potreben izračun dveh integralov, označimo ju z I_1, I_2:

$$I_1 = \int C(\mathbf{x}, \mathbf{x}_i) p(\mathbf{x}) d\mathbf{x} \tag{2.56}$$

$$I_2 = \int C(\mathbf{x}, \mathbf{x}_i) C(\mathbf{x}, \mathbf{x}_j) p(\mathbf{x}) d\mathbf{x} \tag{2.57}$$

Zgornja integrala sta analitično rešljiva le za relativno malo oblik kovariančnih funkcij C. V splošnem si pomagamo z aproksimacijami (razvoj kovariančne funkcije v Taylorjevo vrsto itd). V primeru doslej obravnavane in privzete oblike eksponentne kovariančne funkcije za modeliranje z GP, podane z enačbo (2.28), in normalne (Gaussove) porazdelitve naključnega vhoda $p(\mathbf{x}^*)$ sta integrala rešljiva analitično brez aproksimacij kovariančne funkcije (aproksimacija pa je podana že v sami predpostavki o porazdelitvi vhodnega vektorja $p(\mathbf{x}^*)$, ki v splošnem ni Gaussova) .Gaussovo kovariančno funkcijo $C(\mathbf{x}_p, \mathbf{x}_q) = v\, \mathrm{e}^{-\frac{1}{2}(\mathbf{x}_p - \mathbf{x}_q)^{\mathrm{T}} \mathbf{W}^{-1} (\mathbf{x}_p - \mathbf{x}_q)} + v_0$ si lahko predstavljamo tudi kot Gaussovo porazdelitev vektorja $\mathbf{x}_p$ s srednjo vrednostjo $\mathbf{x}_q$:

$$C(\mathbf{x}_p, \mathbf{x}_q) = \mathrm{c} N_{x_p}(\mathbf{x}_q, \mathbf{W}) \tag{2.58}$$

$$c = (2\pi)^{\frac{m}{2}} |\mathbf{W}|^{\frac{1}{2}} v \tag{2.59}$$

V izpeljavi [32] uporabimo tudi lastnost produkta Gaussovih porazdelitev, ki je spet Gaussova porazdelitev:

$$z N_x(\mathbf{c}, \mathbf{C}) = N_x(\mathbf{a}, \mathbf{A}) N_x(\mathbf{b}, \mathbf{B}) \tag{2.60}$$

$$z = N_a(\mathbf{b}, \mathbf{A} + \mathbf{B}) = N_b(\mathbf{a}, \mathbf{A} + \mathbf{B}) \tag{2.61}$$

$$\mathbf{C} = \left(\mathbf{A}^{-1} + \mathbf{B}^{-1}\right)^{-1} \tag{2.62}$$

$$\mathbf{c} = \mathbf{C}\left(\mathbf{A}^{-1}\mathbf{a} + \mathbf{B}^{-1}\mathbf{b}\right)^{-1} \tag{2.63}$$

Rešitvi integralov I_1 in I_2 ob prej navedenih predpostavkah sta:

$$\mathrm{I}_1 = \int N_x(\mathbf{x}_i, \mathbf{W}) N_x(\mathbf{u}, \boldsymbol{\Sigma}_x) d\mathbf{x} = N_u(\mathbf{x}_i, \mathbf{W} + \boldsymbol{\Sigma}_x) \tag{2.64}$$

$$\mathrm{I}_2 = \int N_x(\mathbf{x}_p, \mathbf{W}) N_x(\mathbf{x}_q, \mathbf{W}) N_x(\mathbf{u}, \boldsymbol{\Sigma}_x) d\boldsymbol{x} = N_{x_p}(\mathbf{x}_q, 2\mathbf{W}) N_u\left(\frac{\mathbf{x}_p + \mathbf{x}_q}{2}, \boldsymbol{\Sigma}_x + \frac{\mathbf{W}}{2}\right) \tag{2.65}$$

Rezultat za srednjo vrednost $m(\mathbf{u}, \boldsymbol{\Sigma}_x)$ predikcije y^* je [24].:

$$m(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = \sum_{i=1}^{N} \beta_i C_{\mathrm{mod}1}(\mathbf{u}, \mathbf{x}_i) \tag{2.66}$$

$C_{\mathrm{mod}1}(\mathbf{u}, \mathbf{x}_i)$ je spremenjena (razširjena) kovariančna funkcija, ki vključuje tudi negotovost vhodov:

$$C_{\mathrm{mod}_1}(\mathbf{u}, \mathbf{x}_i) = v \left| \mathbf{I} + \mathbf{W}^{-1} \boldsymbol{\Sigma}_{\boldsymbol{x}} \right|^{-\frac{1}{2}} \mathrm{e}^{-\frac{1}{2}(\mathbf{u}-\mathbf{x}_i)^T (\mathbf{W}+\boldsymbol{\Sigma}_{\boldsymbol{x}})^{-1} (\mathbf{u}-\mathbf{x}_i)} + v_0 \tag{2.67}$$

Po pričakovanjih velja, da je $m(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}})\big|_{\boldsymbol{\Sigma}_{\boldsymbol{x}}=0} = \mu(\mathbf{u})$, kar je enakovredno rezultatu (2.44), kjer je vhodni vektor $\boldsymbol{x}^*$ določena, nenaključna spremenljivka, $p(\mathbf{x}^*) = N(\mathbf{u}, \mathbf{0})$. Podobno se rezultat za varianco $v(\mathbf{u}, \boldsymbol{\Sigma}_x)$ predikcije y^* glasi:

$$v(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) = v + \sum_{i,j=1}^{N} \left(\beta_i \beta_j - K_{i,j}^{-1}\right) C_{\mathrm{mod}2}(\mathbf{x}_i, \mathbf{x}_j) C_{\mathrm{mod}3}\left(\mathbf{u}, \frac{\mathbf{x}_i + \mathbf{x}_j}{2}\right) - m^2(\mathbf{u}, \boldsymbol{\Sigma}_{\boldsymbol{x}}), \tag{2.68}$$

kjer sta:

$$C_{\mathrm{mod}_2}(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{v}_1 2^{-\frac{D}{2}} \mathrm{e}^{-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T \left(\frac{\mathbf{W}}{2}\right)^{-1} (\mathbf{x}_i - \mathbf{x}_j)} \tag{2.69}$$

$$C_{\mathrm{mod}_3}\left(\mathbf{u}, \frac{\mathbf{x}_i + \mathbf{x}_j}{2}\right) = \mathrm{v}_1 \left| \frac{1}{2}\mathbf{I} + \mathbf{W}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{x}} \right| \mathrm{e}^{-\frac{1}{2}\left(\mathbf{u} - \frac{\mathbf{x}_i + \mathbf{x}_j}{2}\right)^T \left(\frac{\mathbf{W}}{2} + \boldsymbol{\Sigma}_{\boldsymbol{x}}\right)^{-1} \left(\mathbf{u} - \frac{\mathbf{x}_i + \mathbf{x}_j}{2}\right)} \tag{2.70}$$

2.3.3 Analitična aproksimacija s predpostavko o normalni porazdelitvi predikcije in razvojem v Taylorjevo vrsto

Prejšnji rezultati za srednjo vrednost $\mu(\mathbf{x}^*)$ in varianco $\sigma^2(\mathbf{x}^*)$ (predpostavljene) normalno porazdeljene predikcije izhoda y^* pri nedoločenem vhodu $\mathbf{x}^*$, z normalno porazdelitvijo $p(\mathbf{x}^*) = N(\mathbf{u}, \boldsymbol{\Sigma}_x)$, v primeru uporabe Gaussove kovariančne funkcije, so izpeljani iz vrednosti $E_x(\mu(\mathbf{x}^*))$, $E_x(\sigma^2(\mathbf{x}^*))$ in $E_x(\mu(\mathbf{x}^*)^2)$ ((2.53), (2.54), (2.55)) dobljenih iz analitičnih rešitev integralov I_1, I_2 ((2.56), (2.57)). Z dodatno aproksimacijo, kjer srednjo vrednost $\mu(\mathbf{x}^*)$ in varianco $\sigma^2(\mathbf{x}^*)$ razvijemo v Taylorjevo vrsto v okolici srednje vrednosti $\mathbf{u}$ nedoločenega vhoda $\mathbf{x}^*$, pridemo do poenostavljenega načina ocenjevanja porazdelitve predikcije y^* [11]. Razvoj v Taylorjevo vrsto prvega reda za $\mu(\mathbf{x}^*)$ v okolici vrednosti $\mathbf{x}^* = \mathbf{u}$ se glasi:

$$\mu(\mathbf{x}^*) = \mu(\mathbf{u}) + (\mathbf{x}\text{-}\mathbf{u})^{\mathrm{T}} \boldsymbol{\mu}'(\mathbf{u}) + O\left(\|\mathbf{x}\text{-}\mathbf{u}\|^2\right) \tag{2.71}$$

kjer je $\boldsymbol{\mu}'(\mathbf{u}) = \frac{\partial \mu(\mathbf{x})}{\partial(\mathbf{x})}\Big|_{\mathbf{x}=\mathbf{u}}$. Sledi ocena $E_x(\mu(\mathbf{x}^*))$ in $m(\mathbf{u}, \boldsymbol{\Sigma}_x)$:

$$E_x(\mu(\mathbf{x}^*)) \approx E_x\left(\mu(\mathbf{u}) + (\mathbf{x}\text{-}\mathbf{u})^{\mathrm{T}} \boldsymbol{\mu}'(\mathbf{u})\right) = \mu(\mathbf{u}) \tag{2.72}$$

$$m(\mathbf{u}, \boldsymbol{\Sigma}_x) \cong \sum_{i=1}^{N} \beta_i C(\mathbf{u}, \mathbf{x}_i) \tag{2.73}$$

Vidimo, da je srednje vrednosti predikcije y^* enaka kot pri metodi, pri kateri za napovedovanje izhoda y^* uporabljamo samo informacijo o srednji vrednosti naključno porazdeljenega vektorja vhodov $\mathbf{x}^*$, brez upoštevanja informacije o njegovi verjetnostni porazdelitvi.

Podobno ocenimo tudi vrednost $E_x(\mu(\mathbf{x}^*)^2)$:

$$E_x\left(\mu(\mathbf{x}^*)^2\right) \approx \boldsymbol{\mu}'(\mathbf{u})^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathbf{x}} \boldsymbol{\mu}'(\mathbf{u}) = Tr\left[\boldsymbol{\mu}'(\mathbf{u}) \boldsymbol{\mu}'(\mathbf{u})^{\mathrm{T}} \boldsymbol{\Sigma}_x\right] \tag{2.74}$$

Razvoj variance $\sigma^2(\mathbf{x}^*)$ v Taylorjevo vrsto drugega reda (varianca je soda funkcija razdalje od vektorja srednje vrednosti $\mathbf{u}$) v okolici točke $\mathbf{x}^* = \mathbf{u}$ se glasi:

$$\sigma^2\left(\mathbf{x}^*\right)=\sigma^2\left(\mathbf{u}\right)+\left(\mathbf{x}^*\text{-}\mathbf{u}\right)^{\mathrm{T}}\boldsymbol{\sigma}^{2'}\left(\mathbf{u}\right)+\frac{1}{2}\left(\mathbf{x}^*\text{-}\mathbf{u}\right)^{\mathrm{T}}\boldsymbol{\sigma}^{2''}\left(\mathbf{x}^*\text{-}\mathbf{u}\right)+O\left(\left\|\mathbf{x}^*\text{-}\mathbf{u}\right\|^3\right) \tag{2.75}$$

kjer je $\sigma^{2'}\left(\mathbf{u}\right)=\left.\frac{\partial\sigma^2\left(\mathbf{x}^*\right)}{\partial\mathbf{x}^*}\right|_{\mathbf{x}^*=\mathbf{u}}$ in $\sigma^{2''}\left(\mathbf{u}\right)=\left.\frac{\partial^2\sigma^2\left(\mathbf{x}^*\right)}{\partial\mathbf{x}^*\partial\mathbf{x}^{*\boldsymbol{T}}}\right|_{\mathbf{x}^*=\mathbf{u}}$. Ocena $E_x\left(\sigma^2\left(\mathbf{x}^*\right)\right)$ je:

$$E_x\left(\sigma^2\left(\mathbf{x}^*\right)\right)\approx\sigma^2\left(\mathbf{u}\right)+\frac{1}{2}Tr\left[\boldsymbol{\sigma}^{2''}\left(\mathbf{u}\right)\boldsymbol{\Sigma}_{\boldsymbol{x}}\right] \tag{2.76}$$

Z vstavitvijo aproksimacij (2.72), (2.74) in (2.76) v (2.52) dobimo oceno za varianco $v\left(\mathbf{u},\boldsymbol{\Sigma}_x\right)$:

$$v\left(\mathbf{u},\boldsymbol{\Sigma}_{\boldsymbol{x}}\right)\cong\sigma^2\left(\mathbf{u}\right)+Tr\left[\boldsymbol{\Sigma}_{\boldsymbol{x}}\left(\frac{1}{2}\boldsymbol{\sigma}^{2''}\left(\mathbf{u}\right)+\boldsymbol{\mu}'\left(\mathbf{u}\right)\boldsymbol{\mu}'\left(\mathbf{u}\right)^{\mathrm{T}}\right)\right] \tag{2.77}$$

Elemente vektorja odvodov srednje vrednosti predikcije $\mu'\left(\mathbf{u}\right)=\left.\frac{\partial\mu\left(\mathbf{x}^*\right)}{\partial\left(\mathbf{x}^*\right)}\right|_{\mathbf{x}^*=\mathbf{u}}$ v okolici točke $\mathbf{x}^*=\mathbf{u}$ določimo z izračunom parcialnih odvodov po posameznih spremenljivkah vhodnega vektorja, $\frac{\partial\mu\left(\mathbf{x}^*\right)}{\partial\left(x_d^*\right)},d=1...D$:

$$\frac{\partial\mu\left(\mathbf{x}^*\right)}{\partial\left(x_d^*\right)}=\frac{\partial\boldsymbol{k}\left(\mathbf{x}^*\right)^{\mathrm{T}}}{\partial\left(x_d^*\right)}\boldsymbol{\beta} \tag{2.78}$$

Podobno določimo elemente vektorja parcialnih odvodov drugega reda variance predikcije $\boldsymbol{\sigma}^{2''}\left(\mathbf{u}\right)=\left.\frac{\partial^2\sigma^2\left(\mathbf{x}^*\right)}{\partial\boldsymbol{x}^*\partial\boldsymbol{x}^{*\boldsymbol{T}}}\right|_{\mathbf{x}^*=\mathbf{u}}$:

$$\frac{\partial^2\sigma^2\left(\mathbf{x}^*\right)}{\partial x_p^*\partial x_q^*}=\frac{\partial^2k\left(\mathbf{x}^*\right)}{\partial x_p^*\partial x_q^*}-2\frac{\partial\mathbf{k}\left(\mathbf{x}^*\right)^{\mathrm{T}}}{\partial x_p^*}\mathbf{K}^{-1}\frac{\partial\mathbf{k}\left(\mathbf{x}^*\right)^{\mathrm{T}}}{\partial x_q^*}-2\frac{\partial^2\mathbf{k}\left(\boldsymbol{x}^*\right)^{\mathrm{T}}}{\partial x_p^*\partial x_q^*}\mathbf{K}^{-1}\mathbf{k}\left(\mathbf{x}^*\right) \tag{2.79}$$

V primeru Gaussove kovariančne funkcije $C(\mathbf{x}_p,\mathbf{x}_q)=v\,e^{-\frac{1}{2}\sum_{d=1}^{D} w_d\left(x_p^d - x_q^d\right)}+v_0$ so vrednosti parcialnih odvodov srednje vrednosti predikcije v okolici srednje vrednosti vhodov enake:

$$\left.\frac{\partial\mu(\mathbf{x}^*)}{\partial(x_d^*)}\right|_{\boldsymbol{x}^*=\boldsymbol{u}} = \left[w_d\,(x_d - u_d)\mathbf{k}(\mathbf{u})\right]^{\mathrm{T}}\boldsymbol{\beta} \tag{2.80}$$

Vrednosti parcialnih odvodov drugega reda variance v okolici srednje vrednosti vhodov pa so:

$$\begin{aligned}\left.\frac{\partial^2\sigma^2(\mathbf{x}^*)}{\partial x_p^*\partial x_q^*}\right|_{\boldsymbol{x}^*=\boldsymbol{u}} = -2w_p w_q\Big(&\left[(x_d - u_d)\mathbf{k}(\mathbf{u})\right]^{\mathrm{T}}\mathbf{K}^{-1}\left[(x_q - u_q)\mathbf{k}(\mathbf{u})\right]\\ &+\left[(x_d - u_d)(x_q - u_q)\mathbf{k}(\mathbf{u})\right]^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{k}(\mathbf{u})\Big)+v\end{aligned} \tag{2.81}$$

2.3.4 Simulacija nelinearnega dinamičnega sistema z Gaussovimi procesi

Simulacijo dinamičnega sistema, modeliranega z GP modelom $y(k)=f\left(y(k-1),\ldots,y(k-n),u(k-1),\ldots,u(k-n)\right)$ in verjetnostno porazdelitvijo vektorja regresorjev $p(\mathbf{x}^*)=N(\mathbf{u},\boldsymbol{\Sigma}_x)$, začnemo s simulacijo ob času $t=k+1$ s poznanimi predhodnimi vrednostmi vhodov in izhodov sistema [11].. Vektor regresorjev $\mathbf{x}(k+1)\sim N\left(\mathbf{u}_{k+1},\boldsymbol{\Sigma}_{\mathbf{x}_{k+1}}\right)$ tako zapišemo:

$$\mathbf{x}(k+1)\sim N\left(\begin{bmatrix} y(k)\\ \vdots\\ y(k-n+1)\\ u(k)\\ \vdots\\ u(k-n+1)\end{bmatrix},\begin{bmatrix} 0 & \ldots & 0 & 0 & \ldots & 0\\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots\\ 0 & \ldots & 0 & 0 & \ldots & 0\\ 0 & \ldots & 0 & 0 & \ldots & 0\\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots\\ 0 & \ldots & 0 & 0 & \ldots & 0\end{bmatrix}\right) \tag{2.82}$$

Na osnovi vektorja regresorjev $\mathbf{x}(k+1)$ napovemo $y(k+1)\sim N\left(m(\mathbf{x}(k+1)),v(\mathbf{x}(k+1))\right)$. Vektor regresorjev v naslednjem diskretnem časovnem trenutku $t=k+2$ generiramo na podlagi nove ocene izhoda $y(k+1)$:

$$\mathbf{x}(k+2) \sim N\left(\begin{bmatrix} m(\mathbf{x}(k+1)) \\ \vdots \\ y(k-n+2) \\ u(k+1) \\ \vdots \\ u(k-n+2) \end{bmatrix}, \begin{bmatrix} v(\mathbf{x}(k+1)) & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix}\right) \tag{2.83}$$

Napoved izhoda $y(k+2) \sim N(m(\mathbf{x}(k+2)), v(\mathbf{x}(k+2)))$ dobimo z uporabo vektorja regresorjev $\mathbf{x}(k+2)$. V splošnem časovnem trenutku $t = k+l, l \geq n$ je vektor regresorjev $\mathbf{x}(k+1) \sim N\left(\mathbf{u}_{k+1}, \boldsymbol{\Sigma}_{\mathbf{x}_{k+1}}\right)$ podan s porazdelitvijo:

$$\mathbf{x}(k+l) \sim N\left(\begin{bmatrix} m(\mathbf{x}(k+l-1)) \\ \vdots \\ m(\mathbf{x}(k+l-n)) \\ u(k+l-1) \\ \vdots \\ u(k+l-n) \end{bmatrix}, \begin{bmatrix} v(\mathbf{x}(k+l-1)) & \dots & \operatorname{cov}\left(y^{k+l-1}, y^{k+l-n}\right) & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \operatorname{cov}\left(y^{k+l-n}, y^{k+l-1}\right) & \dots & v(\mathbf{x}(k+l-n)) & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix}\right) \tag{2.84}$$

V diskretnem časovnem trenutku $t = k+l$ prek vektorja regresorjev $\mathbf{x}(k+l)$ ocenimo porazdelitev izhoda $y(k+1)$. Za naslednji časovni korak to oceno vključimo v vektor regresorjev $\mathbf{x}(k+l+1)$. Za določitev novih vrednosti kovariančne matrike novega vektorja regresorjev potrebujemo izračun kovarianc med izhodom $y(k+1)$ in vektorjem $\mathbf{x}(k+l)$:

$$\operatorname{cov}\left(y_{k+l}, \mathbf{x}_{k+l}\right) = E[y_{k+l}\mathbf{x}_{k+l}] - E[y_{k+l}]E[\mathbf{x}_{k+l}] \tag{2.85}$$

Velja, da je $E[\mathbf{x}_{k+l}] = \mathbf{u}_{k+l}$, oceni vrednosti $E[y_{k+1}]$ in $E[y_{k+l}\mathbf{x}_{k+l}]$ pa sta odvisni od aproksimacij, ki jih uporabljamo pri določanju porazdelitve izhoda modela. V primeru Gaussove kovariančne funkcije in analitičnih izračunov integralov (2.56), (2.57), je $E[y_{k+l}] = m\left(\mathbf{u}_{k+l}, \boldsymbol{\Sigma}_{\mathbf{x}_{k+l}}\right)$ določen z enačbo (2.66), nove elemente kovariančne matrike pa izračunamo po formuli [32]:

$$\operatorname{cov}\left(y_{k+l}, \mathbf{x}_{k+l}\right) = \sum_{i=1}^{N} \beta_i C_{\mathrm{mod}1}\left(\mathbf{u}_{k+l}, \mathbf{x}_i\right)\left(\mathbf{c}_i - \mathbf{u}_{k+l}\right) \tag{2.86}$$

$$\mathbf{c}_i = \left(\mathbf{W}^{-1} + \boldsymbol{\Sigma}^{-1}_{\mathbf{x}_{k+l}} \right)^{-1} \left(\mathbf{W}^{-1} \mathbf{x}_i + \boldsymbol{\Sigma}^{-1}_{\mathbf{x}_{k+l}} \mathbf{u}_{k+l} \right) \tag{2.87}$$

V primeru uporabe Taylorjeve vrste za opis srednje vrednosti $\mu\left(\mathbf{x}^*\right)$ (2.71) in variance $\sigma^2\left(\mathbf{x}^*\right)$ (2.75) za določitev porazdelitve izhoda modela je $E[y_{k+l}] = m\left(\mathbf{u}_{k+l}, \boldsymbol{\Sigma}_{\mathbf{x}_{k+l}}\right)$ določen z enačbo (2.73), novi kovariančni faktorji pa s formulo [11].:

$$\mathbf{cov}\left(y_{k+l}, \mathbf{x}_{k+l}\right) = \frac{\partial \mu\left(\mathbf{x}_{k+l}\right)}{\partial\left(\mathbf{x}_{k+l}\right)}\bigg|_{\mathbf{x}_{k+l} = \mathbf{u}_{k+l}} \boldsymbol{\Sigma}_{\mathbf{x}_{k+l}} \tag{2.88}$$

Ničelni elementi kovariančne matrike $\boldsymbol{\Sigma}_{\mathbf{x}_{k+l}}$ so zapisani ob predpostavki o določenih, nenaključnih vhodih v sistem in nekoreliranosti (zakasnjenih) vhodov in izhodov sistema.

3. Prediktivno vodenje

Prediktivno vodenje (ang. *Model-Based Predictive Control – MPC, MBPC*) je ena redkih naprednejših metod vodenja, ki se je uveljavila v industrijski praksi, predvsem na področju procesnega vodenja. Vzroke gre iskati v enostavnem, intuitivno razumljivem osnovnem principu metode, z enostavnim vključevanjem omejitev regulirnih in reguliranih veličin, enostavnem načrtovanju in uglaševanju vodenja tako univariabilnih kot multivariabilnih, linearnih in nelinearnih procesov itd. Prediktivno vodenje temelji na eksplicitni uporabi modela procesa za napovedovanje (predikcijo) izhoda procesa v prihodnosti, regulirni signal pa je določen na osnovi minimizacije kriterijske funkcije razlike med napovedanim potekom izhodnega signala in potekom referenčne trajektorije za določen horizont v prihodnosti. Metode prediktivnega vodenja tako spadajo v družino regulacijskih metod na podlagi modela (ang. *Model-Based Control*), kamor med drugimi spadata tudi Smithov prediktor in metoda vodenja z notranjim modelom (ang. *Internal Model Control - IMC*). Področje prediktivnega vodenja vključuje veliko različnih algoritmov s podobnimi osnovnimi principi vodenja in različnimi uporabljenimi modeli in mehanizmi minimizacije kriterijske funkcije, izpeljanimi iz konkretne oblike zapisa modela (*Model Algorithmic Control – MAC, Dynamic Matrix Control – DMC, Generalized Predictive Control – GPC, Predictive Functional Control – PFC, Unified Predictive Control – UPC* itd), splošna oznaka pri uporabi nelinearnih modelov pa je nelinearno prediktivno vodenje (ang. *Nonlinear Model-Based Predictive Control – NMPC*).

S stališča načrtovanja vodenja so glavne prednosti uporabe metod prediktivnega vodenja [21]:

- primerne so za vodenje procesov z zahtevnejšo dinamiko,
- primerne so za procese z mrtvim časom in minimalno fazo,
- splošen koncept omogoča vodenje tako univariabilnih kot multivariabilnih procesov,
- omogočajo predkompenzacijo (ang. *feedforward*) merljivih motenj,
- omejitve procesa zajamemo v postopku načrtovanja,
- pri vodenju lahko upoštevamo omejitve v velikosti reguliranega, regulirnega signala in omejitve v hitrosti sprememb regulirnega signala,

- precejšnja prostost pri načrtovanju, pri čemer gledamo na parametre načrtovanja kot na specifikacije,
- prediktivni regulator lahko ob poznavanju prihodnjega poteka referenčne trajektorije vnaprej generira ustrezen regulirni signal,
- metode ne vsebujejo eksplicitnega odvajanja signalov (zato šum meritev ne povzroča problemov),
- metode ne vsebujejo eksplicitnega integriranja (ni problema integralskega pobega),
- osnovni princip je lahko razumljiv.

Največji slabosti uporabe prediktivnih metod pa sta:

- potrebujemo dober model vodenega procesa, saj je kakovost vodenja neposredno odvisna od kakovosti modela,
- računska zahtevnost metod lahko postane problematična pri vodenju hitrejših procesov.

3.1 Osnovni principi prediktivnega vodenja

Osnovni principi prediktivnega vodenja so podani v naslednjih korakih (slika 3.1.):

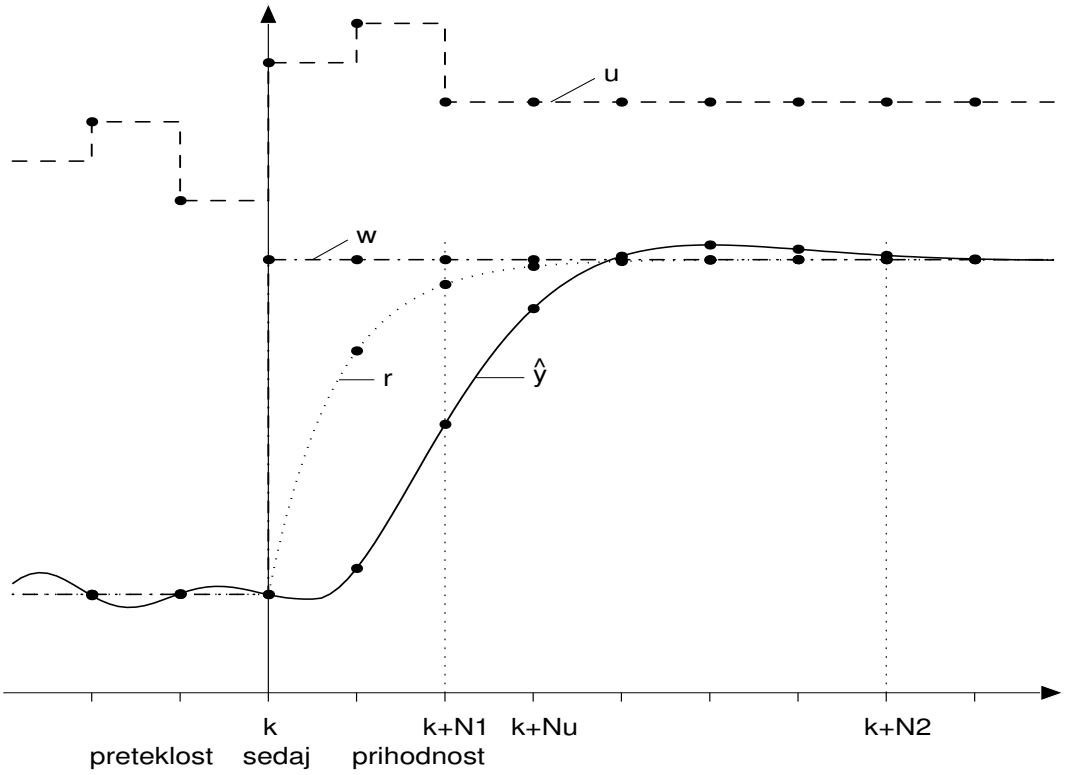
- Napoved (predikcija) izhodnega signala procesa na osnovi modela procesa. Ob vsakem časovnem trenutku *k* izračunamo potek izhodnega signala $y(k+j)$ za horizont v prihodnosti $j=(N_1,\ldots,N_2)$, kjer z N_1 in N_2 označimo spodnjo in zgornjo vrednost predikcijskega horizonta, ki določa horizont ujemanja (ang. *coincidence horizon*), znotraj katerega želimo doseči ujemanje izhodnega signala s predpisanim obnašanjem. Napovedane vrednosti izhodnega signala procesa, izračunane z modelom procesa, označene kot $\hat{y}(k+j|k)$, predstavljajo *j*-koračno napoved modela . Napovedane vrednosti so odvisne od regulirnega scenarija v prihodnosti $u(k+j|k), j=0,\ldots,N_u-1$, ki ga nameravamo uporabiti od trenutka *k* naprej.

- Tvorjenje referenčne trajektorije. Z definiranjem referenčne trajektorije (ang. *setpoint trajectory)* $r(k+j|k), j=N_1,\ldots,N_2$, določimo želen časovni potek procesa od trenutne vrednosti $y(k)$ do predpisane referenčne vrednosti $w(k)$.

- Določitev prihodnjih regulirnih signalov. Vektor prihodnjih regulirnih signalov $u(k+j|k), j=0,\ldots,N_u-1, N_u \le N_2$, izračunamo z minimizacijo ustrezne kriterijske funkcije, prek katere minimiziramo napovedano napako med $r(k+j|k)$ in $\hat{y}(k+j|k)$, $j=N_1,\ldots,N_2$. Določanje prihodnjih signalov temelji na uporabi odprtozančnega kriterija optimalnosti na določenem intervalu v prihodnosti.

- Uporaba prvega elementa vektorja regulirnih signalov za vodenje procesa. Za vodenje uporabimo samo prvi element $u(k|k)$ optimalnega vektorja regulirnih signalov $u(k+j|k), j=0,\ldots,N_u-1$.

V naslednjem časovnem trenutku, ko imamo na voljo nov merjeni izhod procesa, ponovimo celotni postopek. Ta princip se imenuje strategija pomičnega horizonta (ang. *receding horizon strategy*). Za vsak korak je predlaganih veliko različnih rešitev, po katerih se posamezne metode prediktivnega vodenja razlikujejo med seboj. [29], [21], [7]

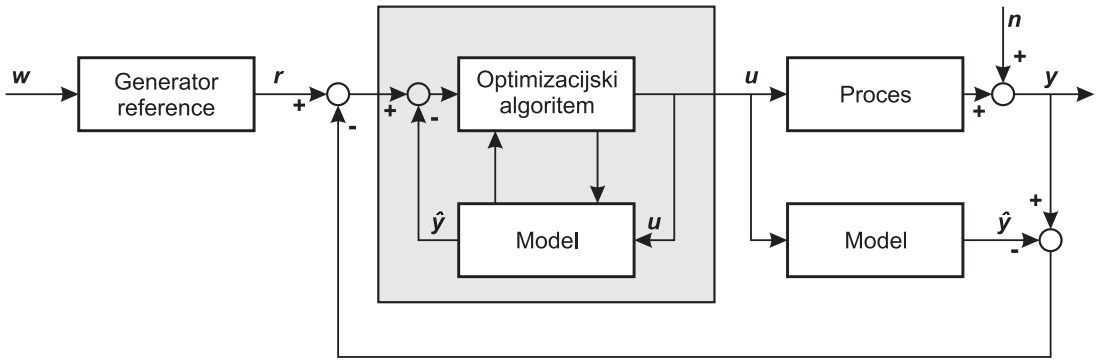


Slika 3.1: Princip prediktivnega vodenja

Osnovno shemo zaprtozančnega prediktivnega sistema vodenja prikazuje slika 3.2. Medtem, ko že osnovni model lahko vključuje tudi model vpliva merljivih motenj, pa je povratna zanka vključena v algoritem z namenom odpravljanja napak enokoračne predikcije modela, nastalih zaradi nemodelirane dinamike in ostalih (nemerljivih) motenj v procesu. Napako $e(k)$ izračunamo kot razliko izhoda procesa in modela ob času vzorčenja $t = k$:

$$e(k) = y(k) - \hat{y}(k) \tag{3.1}$$

Pri napovedovanju izhoda procesa v prihodnosti napako $e(k)$ prištejemo predikciji modela $\hat{y}(k+j|k)$, za vsak $j = N_1, \ldots, N_2$. S tem, ko predpostavimo konstantno napako $e(k)$ za celoten predikcijski horizont, kompenziramo napake modela v ustaljenem stanju in konstantne motnje v procesu. Prediktivni algoritem tako, na podlagi odstopanj med vrednostmi izhoda modela in meritev, oceni prihodnji vpliv nemerljivih motenj, ki delujejo na proces. V idealnem primeru ujemanja modela s procesom in odsotnosti motenj je povratna zanka nedelujoča, vodenje je primer odprtozančnega optimalnega vodenja.



Slika 3.2: Shema prediktivnega vodenja

3.2 Model procesa

Za napoved izhodnega signala procesa je bistvena uporaba modela procesa, s katerim je možno izračunati napoved izhodnega signala več korakov v prihodnost. Načelno lahko uporabimo poljuben linearen ali nelinearen model, v uveljavljenih variantah prediktivnega vodenja pa je najbolj razširjena uporaba linearnih modelov tako pri

vodenju linearnih kot tudi nelinearnih procesov. Glavni vzrok je zmanjšanje računske zahtevnosti pri iskanju optimalnega vektorja regulirnih signalov, ki je v splošnem, z uporabo nelinearnih modelov, določitev minimuma nekonveksnega optimizacijskega problema, za kar je potrebna uporaba računsko zahtevnih iterativnih optimizacijskih algoritmov. Z uporabo linearnih (lineariziranih) modelov pa lahko dobimo konveksen optimizacijski problem in se v primeru odsotnosti omejitev iterativni optimizaciji lahko izognemo. Minimum lahko določimo ali analitično ali pa z enokoračnimi optimizacijskimi metodami, pri vključitvi omejitev pa imamo še vedno konveksen optimizacijski problem, katerega minimum določimo z iterativno optimizacijo. Najpogosteje vključene omejitve so:

- omejitve izhoda procesa,

$$Y_j^{\min} \le y(k+j|k) \le Y_j^{\max}, j = N_1, \ldots, N_2,$$

- omejitve vhodnega (regulirnega) signala,

$$U_j^{\min} \le u(k+j|k) \le U_j^{\max}, j = 0, \ldots, N_u - 1,$$

- omejitve hitrosti sprememb vhodnega signala,

$$D_j^{\min} \le \Delta u(k+j|k) \le D_j^{\max}, j = 0, \ldots, N_u - 1.$$

Najpogosteje uporabljani linearni modeli so [21]:

- impulzni odziv končne dolžine (uporabljen pri metodah *GPC*, *UPC*),
- odziv na stopnico končne dolžine (uporabljen pri metodi *DMC*),
- prenosna funkcija (uporaba pri metodah *GPC*, *UPC*),
- zapis v prostoru stanj (uporaba pri metodi *PFC*).

Pri uporabi nelinearnih modelov za prediktivno vodenje je oblika zapisa normalno neposredno povezana z načinom konstrukcije modela. Modele tako ločimo na:

- Teoretične modele, znane pod imenom modeli v obliki bele škatle (ang. *white box models*). Modele določimo izključno na podlagi določitve vseh fizikalnih mehanizmov in povezav v procesu, kar je pri realnih procesih izjemno zahtevno

in zamudno. Tako dobljene modele zapišemo in simuliramo z diferencialnimi in parcialnimi diferencialnimi enačbami, masnimi in energijskimi ravnotežnimi zapisi itd. Tako dobljeni modeli so običajno zapisani v časovno zvezni obliki.

- Eksperimentalne modele poznamo tudi pod imenom modeli v obliki črne škatle (ang. *black box models*). Konstruiramo jih izključno iz merjenih procesnih veličin. Parametre izbrane strukture zapisa modela določimo s prileganjem merjenim podatkom procesa. Primeri takih opisov procesov so: nevronske mreže, mehki modeli, Gaussovi procesi itd. Tako dobljeni modeli so normalno zapisani v časovno diskretni obliki.

- Kombinirane modele, imenovane modeli v obliki sive škatle (ang. *grey box models*). Pri kombiniranem pristopu uporabimo znanje o procesu na podlagi poznanih fizikalnih mehanizmov in povezav v procesu, preostale neznane parametre pa izračunamo iz meritev procesnih veličin. Dobljeni modeli imajo glede na količino upoštevane informacije o fizikalnem ozdaju različno stopnjo sivine.

Pri uporabi nelinearnih modelov lahko različne prediktivne algoritme z nelinearnim modelom delimo glede na način reševanja nelinearnega regulacijskega problema:

- Neposredni nelinearni pristop zapiše regulacijski problem v obliki problema nelinearnega programiranja in ga rešuje z iterativnimi optimizacijskimi metodami. Tak pristop izhaja direktno iz osnovne ideje uporabe nelinearnega modela pri predikciji, vendar je, zaradi reševanja nekonveksnih optimizacijskih problemov z dodatnimi omejitvami, računsko zahteven [22], [26].

- Linearizacijski pristopi pa, zaradi poenostavitev optimizacijskega problema in uporabe linearnih prediktivnih algoritmov vodenja, proces linearizirajo. Znane, od nelinearnega zapisa modela odvisne, so uporabe povratnozančne linearizacije ali inverzne nelinearne preslikave , sprotne linearizacije modela v delovni točki, uporabe množice lokalnih linearnih modelov, tvorjenje odzivov na stopnico v posameznih delovnih točkah itd [21], [7].

3.3 Tvorjenje referenčne trajektorije

Vse metode prediktivnega vodenja predpostavljajo poznavanje poteka referenčnega signala v prihodnosti, ta predpostavka pa ni vedno izpolnjena. Običajno predpišemo tudi, na kakšen način naj se izhod procesa približuje referenčnemu signalu, kar določimo s tvorjenjem referenčne trajektorije $r(k+j|k), j = N_1,\ldots,N_2$. Referenčno trajektorijo si lahko predstavljamo kot interno referenco prediktivnega regulatorja, ki določa želeno zaprtozančno obnašanje, in na podlagi katere določimo vektor prihodnjih regulirnih signalov. Pri tvorjenju referenčne trajektorije glede na poznavanje referenčnega signala ločimo dve situaciji:

- Referenčni signal $w(k+j)$ je znan vnaprej za vse $j = 1,\ldots,N_2$. Zaradi principa prediktivnega vodenja, ki napoveduje obnašanje v prihodnosti, prediktivni regulator povzroči ustrezno regulirno akcijo še pred dejansko spremembo referenčnega signala. Tako kompenziramo mrtvi čas in velike časovne zakasnitve procesa. Referenčni signal je lahko vnaprej poznan pri robotskih sistemih, sledilnih sistemih, šaržnih procesih itd.

- Referenčni signal $w(k+j)$ ni znan vnaprej. Kot najboljšo možno napoved referenčnega signala vzamemo njegovo trenutno vrednost: $w(k+j) = w(k)$.

Glede na začetno točko (inicializacijo) referenčne trajektorije ločimo dve situaciji:

- Uporabimo trenutno merjeno vrednost izhodnega signala $r(k) = y(k)$. Na ta način vnesemo v sistem dodatno povratno zanko, katere vpliv na celoten zaprtozančni sistem težko ovrednotimo, saj ni rezultat optimizacije kriterijske funkcije. V nekaterih primerih lahko tako celo destabiliziramo sistem.

- Uporabimo trenutno vrednost referenčne trajektorije $r(k)$.

V procesnem vodenju velikokrat želimo približevanje izhoda procesa $y(k)$ referenčni vrednosti $w(k)$ po trajektoriji, določeni s sistemom prvega reda enotinega ojačanja in časovno konstanto T_{ref}. Trenutni pogrešek označimo z $\varepsilon(k)$, definiran je kot:

$$\varepsilon(k) = w(k) - y(k) \tag{3.2}$$

Želeno upadanje pogreška v prihodnosti je določeno z enačbo:

$$\varepsilon(k+j) = e^{-\frac{jT_s}{T_{ref}}}\varepsilon(k) = \lambda^i \varepsilon(k), \qquad j = 1,\ldots,N_2\ , \tag{3.3}$$

kjer je T_s čas vzorčenja, λ pa faktor, ki določa hitrost odziva zaprtozančnega sistema:

$$\lambda = e^{-\frac{T_s}{T_{ref}}}, \qquad 0 \le \lambda < 1\,, \tag{3.4}$$

Referenčna trajektorija, določena ob trenutku $t = k$ (v primeru inicializacije začetne vrednosti referenčne krivulje $r(k) = y(k)$) je tako opisana z enačbo [29]:

$$r(k+j|k) = w(k+j) - \varepsilon(k+j) = w(k+j) - e^{-\frac{jT_s}{T_{ref}}}\varepsilon(k),\ j = 1,\ldots,N_2 \tag{3.5}$$

3.4 Določitev prihodnjih regulirnih signalov

Odprtozančno optimalna določitev odziva sistema v določenem intervalu prihodnosti je glavna lastnost prediktivnega vodenja. V tej točki zasledimo podobnost z linearno kvadratnim (ang. *linear quadratic - LQ*) regulatorjem, vendar slednji uporablja neskončno dolg interval prihodnosti (predikcijski horizont). Izbira ustrezne kriterijske funkcije je prvi korak pri določitvi vektorja prihodnjih regulirnih signalov $u(k+j|k),\ j = 0,\ldots,N_u - 1$. Pri tem težimo k izbiri kriterijskih funkcij, ki so enostavno izračunljive in katerih minimum lahko poiščemo analitično ali pa z optimizacijskimi algoritmi. Ker imamo z referenčno trajektorijo že določeno želeno obnašanje sistema v prihodnosti, je logična izbira kriterijskih funkcij razlike med napovedanim odzivom procesa in referenčno trajektorijo, npr:

$$J = \sum_{j=N_1}^{N_2} \left(r(k+j) - \hat{y}(k+j)\right)^2 \tag{3.6}$$

Parametra N_1 in N_2 določata horizont ujemanja, v katerem želimo, da se napovedani izhod procesa čimbolj ujema z referenčno trajektorijo. Z večanjem

parametra N_1 izpuščamo vpliv napak vodenja v bližnji prihodnosti, kar posebno pri fazno minimalnih sistemih ali sistemih z mrtvim časom vodi k bolj umirjenemu in gladkemu vodenju. Razširjeno obliko kriterijske funkcije (3.5) opisuje enačba:

$$J = \sum_{j=N_1}^{N_2} \alpha_j \left(r(k+j) - \hat{y}(k+j)\right)^2 \tag{3.7}$$

Z vektorjem uteži $\boldsymbol{\alpha} = \lfloor \alpha_{N_1} \quad \ldots \quad \alpha_{N_2} \rfloor$ je možno dodatno vplivanje na pomembnost napak ob posameznih časovnih trenutkih v horizontu ujemanja. Pogosta je tudi vključitev mere za varianco regulirnega signala v kriterijsko funkcijo, prek katere, za ceno povečanja odstopanj med napovedanim izhodom procesa in referenčno trajektorijo, skušamo zmanjšati spreminjanje regulirnega signala:

$$J = \sum_{j=N_1}^{N_2} \left(r(k+j) - \hat{y}(k+j)\right)^2 + \sum_{j=0}^{N_u} \beta \left(\Delta u(k+j)\right)^2 \tag{3.8}$$

Zgornja kriterijska funkcija vsebuje vektor sprememb regulirnega signala $\Delta u(k+j|k), j = 0, \ldots, N_u - 1$. Pri številnih metodah se uporablja določanje optimalnega vektorja sprememb regulirnih signalov $\Delta u(k+j|k), j = 0, \ldots, N_u - 1$ in ne absolutnih vrednosti $u(k+j|k), j = 0, \ldots, N_u - 1$ (npr. metoda DMC, ki temelji na modelu odziva procesa na enotino stopnico in izračunu izhoda modela z diskretno konvolucijsko enačbo, v kateri nastopa vektor sprememb regulirnega signala [21]).

Napovedani signal v prihodnosti $\hat{y}(k+j|k),\ j = 1, \ldots, N_2$ je odvisen od predvidenega vektorja regulirnih signalov v prihodnosti $u(k+j|k), j = 0, \ldots, N_u - 1, N_u \leq N_2$. V splošnem so elementi vektorja poljubni in medsebojno neodvisni, kar pa z večanjem velikosti N_u izjemno poveča računsko zahtevnost in časovno potratnost optimizacije. Regulirni signal lahko tako postane tudi bogat z neželenimi visokimi frekvencami. V praksi se vedno odločimo za strukturiranje vektorja regulirnih signalov z uvedbo relacij med elementi vektorja. Taka odločitev tudi poveča robustnost prediktivnega vodenja. Zaradi principa pomičnega horizonta in uporabe le prvega elementa vektorja regulirnih signalov za vodenje pa dejanski regulirni signal zaradi uvedbe strukturiranja ni omejen. Najpogosteje uporabljene tehnike strukturiranja regulirnega signala [29] so:

- Uvedba regulirnega horizonta po prehodnem pojavu predpostavlja konstanten regulirni signal ob predpostavki o konstantnem referenčnem signalu v prihodnosti. Regulirni horizont $N_u, N_u \leq N_2$, predstavlja časovni trenutek, od katerega naprej bo regulirni signal ostal konstanten. S tem zmanjšamo število spremenljivk optimizacije (elementov vektorja regulirnih signalov). Najenostavnejši in v praksi pogosto uporabljen limitni primer je $N_u = 1$, ki daje dobre rezultate tudi v primeru spreminjajočega se referenčnega signala.

- Tehnika grupiranja, ob predpostavki $N_u = N_2$, razdeli celoten predikcijski horizont na določeno število segmentov, znotraj katerih predpostavi konstanten regulirni signal.

- Razvoj po baznih funkcijah strukturira regulirni signal kot linearno kombinacijo neodvisnih, vnaprej določenih baznih funkcij (linarnih, polinomskih itd). Izbira baznih funkcij je odvisna od procesa in referenčnega signala. Izračun regulirnega signala se tako skrči na izračun parametrov izbranih baznih funkcij.

Po določitvi oblike kriterijske funkcije in strukture vektorja prihodnjih vhodnih signalov sledi določitev vrednosti elementov tega vektorja. Pri uporabi linearnih modelov, brez uvedb omejitev, rešitev poiščemo z določitvijo minimuma konveksnega optimizacijskega problema. Rešitev pogosto lahko določimo analitično (izhod procesa je linearno odvisen od preteklih vrednosti vhodov in izhodov sistema), ali pa z enokoračnimi optimizacijskimi metodami, npr metodo najmanjših kvadratov. Uvedba omejitev pri minimizaciji kriterijske funkcije z uporabo linearnih modelov zahteva iskanje minimuma kriterijske funkcije z uporabo iterativnih optimizacijskih algoritmov, optimizacijski problem pa ostane konveksen. Uporaba nelinearnih modelov v splošnem prinese nekonveksnost optimizacijskega problema, z uporabo časovno in računsko zahtevnih iterativnih algoritmov pa konvergenca k pravi rešitvi ni več zagotovljena znotraj predpisanega časa oziroma števila iteracij optimizacije. Dodatna vključitev omejitev procesnih spremenljivk v optimizacijo pa, pri izbranem optimizacijskem algoritmu, lahko privede celo do nedoločljivosti minimuma kriterijske funkcije ob upoštevanju omejitev (npr. pri nepredvideno velikih motnjah). Obe možnosti sta za izvedbo nelinearnega prediktivnega vodenja na realnih procesih nesprejemljivi. V literaturi je predlaganih več izhodov ob nastopu take situacije, od vodenja v proces predhodne, nespremenjene vrednosti regulirne veličine, do vodenja drugega elementa optimalnega vektorja regulirnih signalov,

določenega v predhodnem koraku itd. Taki optimizacijski postopki (ang. *interior point methods*) [29] pazijo na upoštevanje časovnih in rezultatskih omejitev v vsakem koraku optimizacije in dajo vsaj približno rešitev, če je potrebno zaradi časovne omejitve optimizacijo predčasno prekiniti ali pa optimizacijski postopek ne najde rešitve optimizacijskega problema.

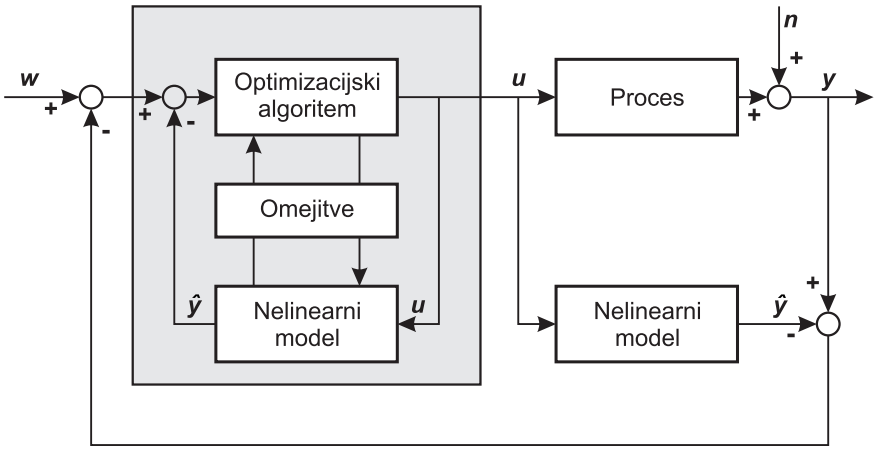
Zanimivo možnost predstavlja "*mehčanje*" omejitev, kjer privzamemo, da omejitve sicer predstavljajo meje, ki naj jih ne bi smeli prestopiti, vendar, prek strukture kriterijske funkcije, uvedemo mehanizem, ki to ob izrednih razmerah dovoli. Pri tem je potrebno definirati dve množici omejitev:

- omejitve, ki jih zaradi fizičnih, varnostnih in podobnih razlogov ne moremo prekoračiti (npr. območje in hitrost spremembe regulirne veličine sta omejeni z uporabljenim aktuatorjem) in jih ne mehčamo,

- ostale omejitve, ki jih lahko mehčamo (npr. v primeru izredne situacije lahko prekoračimo meje, ki določajo sprejemljivo kakovost izdelka).

Mehčanje omejitev realiziramo z razširitvijo uporabljene kriterijske funkcije z dodatno funkcijo mehkih omejitev, ki je neničelna samo v primeru, ko so omejitve prekoračene. Na ta način optimizacijski algoritem omejitve prekorači le v nastopu izredne situacije (npr. nepredvideno velikih motnjah v procesu) [29].

Shemo nelinearnega prediktivnega vodenja z vključenimi omejitvami prikazuje slika 3.3.



Slika 3.3: Shema nelinearnega prediktivnega vodenja z vključenimi omejitvami

Za vodenje uporabimo samo prvi element $u(k|k)$ optimalnega vektorja regulirnih signalov $u(k+j|k), j=0,\ldots,N_u-1$. Zaradi strategije pomičnega horizonta in končne dolžine predikcijskega horizonta odziv regulacijskega sistema v splošnem ni enak optimalnemu odprtozančnemu odzivu, na podlagi katerega je bil določen optimalni vektor regulirnih signalov.

3.5 Prediktivno funkcionalno vodenje

Prediktivno funkcionalno vodenje (ang. *Predictive Functional Control – PFC*) [29], je metoda prediktivnega vodenja, ki se je zaradi svoje enostavnosti in učinkovitosti uveljavila v inženirski praksi. Sama metoda ima veliko različnih variant, ki se razlikujejo po obliki predikcijskega modela, uporabljenih baznih funkcijah za strukturiranje regulirnega signala itd. Skupna značilnost večine PFC metod je skrčenje horizonta ujemanja na nekaj točk, katerega spodnjo in zgornjo mejo označimo s H_s in H_z, v primeru ene same točke pa je horizont ujemanja (tudi predikcijski horizont) , označen s H, $H = H_s = H_z$. V tem primeru govorimo tudi o točki ujemanja P (ang. *coincidence point*). V splošnem PFC vključuje vse glavne splošne principe prediktivnega vodenja, opisane v prejšnjih odstavkih:

- Napoved (predikcija) izhodnega signala procesa na osnovi modela procesa. PFC omogoča uporabo poljubnega modela, s katerim lahko napovedujemo izhod procesa več korakov v prihodnost.

- Tvorjenje referenčne trajektorije. Referenčna trajektorija opisuje gladek prehod od trenutne vrednosti procesa $y(k)$ do želene referenčne vrednosti $w(k)$. Določa želeno zaprtozančno obnašanje sistema. Algoritem predvideva inicializacijo referenčne trajektorije na trenutno merjeno vrednost izhodnega signala, $r(k) = y(k)$, in eksponentno upadanje pogreška v prihodnosti, $\varepsilon(k+j) = \lambda^j \varepsilon(k), j = 1,\ldots,H_z$. Vrednost parametra $\lambda = 0$ določa hiter, agresiven odziv sistema, z večanjem njegove vrednosti proti 1 pa zaprtozančno delovanje upočasnimo.

- Določitev prihodnjih regulirnih signalov na osnovi minimizacije kriterijske funkcije začnemo s strukturiranjem regulirnega signala. Možno obliko poteka regulirnega signala opišemo z baznimi funkcijami:

$$u(k+j|k) = \sum_{c=1}^{C} u_c(k) f_c(j),\ j = 0,\ldots,H_z - 1 \tag{3.9}$$

Najpogostejša je uporaba polinomskih baznih funkcij:

$$u(k+j|k) = \sum_{c=1}^{C} u_c(k) j^{c-1},\ j = 0,\ldots,H_z - 1 \tag{3.10}$$

Število baznih funkcij C je običajno majhno (od 1 do 3), tako da je optimizacija za vsak časovni korak k omejena na določitev nekaj parametrov $u_c(k)$. V primeru analitičnega ali enokoračnega optimizacijskega določanja parametrov $u_c(k)$ mora biti število točk horizonta ujemanja vsaj enako številu parametrov baznih funkcij, ki jih določamo. Pri PFC algoritmih vodenja je najpogosteje predpostavljena samo ena točka ujemanja P na koncu predikcijskega horizonta H, regulirni signal pa je strukturiran tako, da je konstanten na celotnem predikcijskem horizontu H (uporabljena je samo ena, konstantna bazna funkcija). Z uvedbo strukturiranja regulirnega signala po baznih funkcijah tudi odpade določanje dolžine regulirnega horizonta, saj je potek regulirnega signala v prihodnosti določen že s strukturo baznih funkcij do konca predikcijskega horizonta H. Predikcijski horizont je kot nastavitveni parameter regulatorja pomemben za dinamiko, robustnost in stabilnost zaprtozančnega sistema. Z manjšanjem vrednosti H pri konstantno določenem zmanjšanju pogreška $\varepsilon(k+H)$ glede na $\varepsilon(k)$ zaprtozančni sistem pohitrimo in povečamo amplitude regulirnih signalov.

V primeru ene same točke ujemanja P, je kriterijska funkcija določena kot:

$$J = \left(r(k+P) - \hat{y}(k+P)\right)^2 \tag{3.11}$$

Razloga, zakaj v praksi dosežemo zadovoljivo delovanje PFC vodenja z uporabo tako enostavne kriterijske funkcije, ki vsebuje samo eno točko ujemanja in v katero ni vključena mera za varianco regulirnega signala, prek katere bi lahko dinamiko regulirne veličine omejiti, sta [29]:

- zaprtozančno delovanje sistema je v veliki meri določeno že s tvorjenjem referenčne trajektorije, kar implicitno že vsebuje tudi omejitve pri dinamiki regulirne in doseganju regulirane veličine,
- optimizacijski algoritem je, zaradi strukturiranja regulirnega signala z baznimi funkcijami, omejen s številom prostostnih stopenj pri določanju dinamike regulirnega signala.

Implementacija PFC algoritmov vodenja, zaradi zgornjih razlogov, z uporabo kriterijske funkcije (3.11), brez vključene mere za varianco regulirnega signala, rezultira v relativno gladkih potekih regulirnega signala in zadovoljivem zaprtozančnem delovanju.

Določitev optimalnega vektorja prihodnjih regulirnih signalov je odvisna od uporabljenega predikcijskega modela procesa, ki je lahko splošen, in od vključenih omejitev. Optimalne regulirne signale v prihodnosti tako lahko določamo analitično, z enokoračnimi ali iterativnimi optimizacijskimi metodami.

Za vodenje uporabimo samo prvi element določenega optimalnega vektorja regulirnih signalov, v naslednjem časovnem koraku se, po principu pomičnega horizonta, celoten postopek ponovi.

4. Opis procesnega laboratorija

V tem poglavju je predstavljen procesni laboratorij, katerega sestavni del je tudi proces priprave plina, za katerega bo izvedena identifikacija z Gaussovimi procesi in prediktivno vodenje na osnovi GP modela.

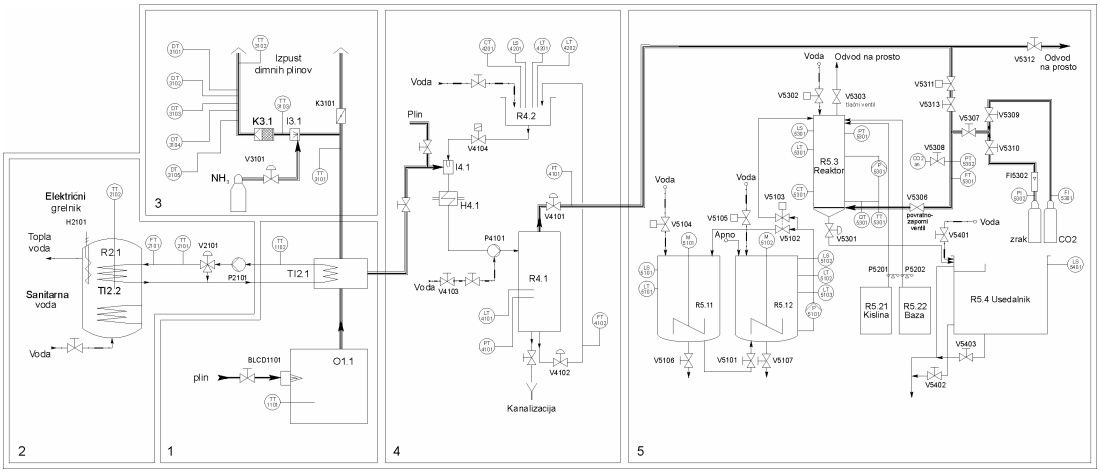
Raziskovalna skupina Odseka za sisteme in vodenje na Inštitutu Jožef Stefan, ki deluje na področju raziskav, razvoja in uporabe znanj s področja tehnologij vodenja sistemov, daje velik poudarek prenosu metod avtomatskega vodenja s teoretične ravni in laboratorijskega okolja na realne industrijske objekte. V ta namen je ob podpori evropskega programa TEMPUS ALIAC (ang. *Active Learning In Automatic Control*) nastal tudi polindustrijski procesni laboratorij s pravo industrijsko procesno opremo in napravami industrijskih razsežnosti.

Procesni laboratorij predstavlja izvor različnih inženirskih nalog in problemov v zvezi z avtomatskim vodenjem procesov ter poligon za praktično preizkušanje različnih metod s področja procesnega vodenja. Laboratorij se dograjuje glede na trenutne potrebe, pri tem pa se uporablja komercialno dostopna profesionalna industrijska procesna oprema, ki omogoča razvoj in preizkus delovanja sodobnih metod avtomatskega vodenja ob upoštevanju vseh omejitev, neidealnosti in tehničnih posebnosti, na katere naletimo v industriji.

Procesni laboratorij je sestavljen iz petih tehnoloških sklopov – procesnih enot, ki predstavljajo pet tipičnih industrijskih procesov (slika 4.1):

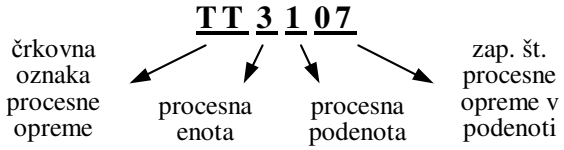
- proces izgorevanja (blok št. 1),
- proces izmenjave toplotne energije (blok št. 2),
- proces selektivne katalitske redukcije dušikovih oksidov iz produktov zgorevanja (blok št. 3),
- proces priprave plina (blok št. 4),
- proces kemijske nevtralizacije (blok št. 5).

Posamezni procesi so zasnovani tako, da lahko delujejo samostojno ali medsebojno povezano, saj je omogočen pretok materiala, energije in informacije med njimi. Medsebojno povezani procesi omogočajo usposabljanje na področju porazdeljenih sistemov vodenja [18], [39].



Slika 4.1: Tehnološka shema procesnega laboratorija

V tehnološki shemi procesnega laboratorija so posamezne naprave označene na sledeči način:



Črkovni oznaki sledi eno mesto za številčno oznako procesne enote, eno mesto za številčno oznako procesne podenote in dve mesti za zaporedno številko procesne opreme v procesni podenoti. Gornja oznaka predstavlja merilnik temperature, ki je nameščen v procesni enoti št. 3, procesni podenoti št. 1 in ima v tej podenoti zaporedno št. 7. Črkovne oznake opreme laboratorija so zbrane v tabeli 4.1:

črkovna oznaka	*angleški (nemški) pomen*	*slovenski pomen*
BLCD	Burner Local Control Device	gorilnik
CT	Conductivity Transmitter	merilnik prevodnosti
DT	Density Transmitter	merilnik koncentracije
FT	Flow Transmitter	merilnik pretoka
H	Heater	grelnik
K	Klappe (*nem.*)	loputa
LS	Level Switch	nivojsko stikalo
LT	Level Transmitter	merilnik nivoja
M	Motor	motor
P	Pump	črpalka
PT	Pressure Transmitter	merilnik tlaka
QT	pH Transmitter	PH-merilnik
TT	Temperature Transmitter	merilnik temperature
V	Valve	ventil

Tabela 4.1 - Črkovne oznake tipov procesne opreme

S procesi lahko upravljamo lokalno preko komandnega pulta (slika 4.2), ali daljinsko preko sistema vodenja.

Instrumentarij na komandnem pultu je razdeljen po tipih. Analogni del komandnega pulta sestavljajo kombinirani analogni in digitalni prikazovalniki vrednosti signalov, dva trikanalna risalnika, šest industrijskih regulatorjev, postaje avtomatsko-ročno (A/M postaje) z dajalniki referenčnih vrednosti in analogna programska plošča, ki omogoča enostavno spreminjanje shem vodenja in dostop do analognih procesnih signalov. Analogni del komandnega pulta je s procesom povezan preko standardnih tokovnih zank 4-20 mA.

Slika 4.2: Komandni pult procesnega laboratorija

Digitalni del sestavljajo indikatorji digitalnih signalov, vklopno-izklopna stikala s preklopom med ročnim in avtomatskim načinom, stikala dodatnih logičnih pogojev in digitalna programska plošča, ki omogoča spreminjanje shem vodenja in dostop do digitalnih procesnih signalov. Digitalni del komandnega pulta je s procesom povezan preko napetostnih signalov 0-24 V. Na komandnem pultu sta še programabilna logična krmilnika (PLK) Mitsubishi za avtomatsko vodenje procesa, ki sta povezana z analognim in digitalnim delom [18], [39].

Sama zasnova laboratorija omogoča različne načine zajemanja signalov senzorjev v procesu in vplivanje na proces prek aktuatorjev. Tako je prek analognih A/M postaj možna izbira za vodenje posameznih aktuatorjev prek 0-10 V napetostnega signala ustreznega potenciometra na analogni stikalni plošči v lokalnem položaju analogne postaje oziroma zunanjega 0-10 V napetostnega signala v daljinskem položaju analogne postaje. Z ustrezno vezavo analogne stikalne plošče je ta zunanji signal lahko signal analognih izhodov Mitsubishi PLK-ja ali pa poljuben zunanji signal (npr. signal digitalno analogne (D/A) pretvorniške kartice osebnega računalnika, prek katerega želimo voditi proces). Zaradi neskladja oznak stanj analognih postaj A/M

(*A – Automatic, M - Manual*) z njihovo osnovno funkcijo in podvojitvijo oznak načina vodenja naprav znotraj samega sistema vodenja, namesto oznak A/M uporabljamo oznaki L/R, lokalno (*L –Local*) in daljinsko (*R – Remote*). Z vezavo analogne stikalne plošče analogne signale (0 – 10 V) senzorjev v procesu lahko vodimo do analognih vhodov Mitsubishi PLK-ja ali pa tudi do poljubne druge naprave (npr. A/D kartice osebnega računalnika). Na analogni stikalni plošči so dostopni tudi analogni signali sistema za vodenje procesa (napetostni signali potenciometrov).

Pri digitalnih aktuatorjih z vklopno-izklopnimi stikali izbiramo med lokalnim in daljinskim režimom delovanja. V lokalnem načinu lahko prek čelne plošče stikala vklopimo ali izklopimo napravo. Na digitalni programski plošči so dosegljivi vsi digitalni signali iz procesa (napetostni nivo 0 – 24 V), dostopni pa so tudi vhodi, na katere vodimo digitalne izhodne signale iz sistema za vodenje procesa (signali vklopno-izklopnih stikal).

Za kontinuirne podprocese v laboratoriju je bilo v preteklosti že razvito osnovno in postopkovno vodenje, realizirano tako na nivoju programabilnih logičnih krmilnikov (Mitsubishi) kot tudi na nivoju nadzornega sistema vodenja (Factory Link). Sistem vodenja je bil razvit z namenom doseganja ciljev vodenja pri zagotavljanju varnega obratovanja procesov [19], [5], [18].

4.1 Proces izgorevanja

Proces izgorevanja je tipičen primer energetskega vira, ki ga redno srečujemo v industriji. Namenjen je proizvodnji toplotne energije, ki jo v laboratoriju prejema proces izmenjave toplotne energije. Dimni plini, ki nastajajo pri zgorevanju goriva – zemeljskega plina, vstopajo v proces selektivne katalitske redukcije dušikovih oksidov. Ker zaradi zakonskih omejitev pri uporabi zemeljskega plina iz mestne plinske napeljave ne smemo posegati v sam gorilnik, nimamo možnosti regulacije pretoka goriva in s tem posredno regulacije izgorevanja.

4.2 Proces izmenjave toplotne energije

Proces izmenjave toplotne energije predstavlja, z merjenjem pretoka in temperature v primarnem in sekundarnem krogu, tipičen primer regulacije toplotnih izmenjevalnikov. Proces lahko uporabljamo v dva namena:

- Za regulacijo temperature dimnih plinov pred vstopom v katalizator. Toplotni izmenjevalnik, ki se nahaja pred katalizatorjem, služi za hlajenje dimnih plinov. Stopnja hlajenja je odvisna od položaja mešalnega ventila, katerega pozicija določa, kolikšen del grelne vode se odvaja v bojler in kolikšen del se vrača v izmenjevalnik pred katalizatorjem. Če temperatura grelne vode naraste preko 90 °C, pride do ustavitve procesa zgorevanja, za kar poskrbi sistem vodenja procesa zgorevanja.

- Za ogrevanje sekundarne vode na določeno temperaturo. Stopnja ogrevanja je odvisna od temperaturne razlike med ogrevano in grelno vodo. Temperatura sekundarne vode pa je odvisna od zahtevane stopnje hlajenja dimnih plinov. Po potrebi je možno sekundarno vodo dodatno ogrevati z električnim grelnikom, ki se nahaja v bojlerju.

4.3 Proces redukcije dušikovih oksidov

Proces selektivne katalitske redukcije dušikovih oksidov iz produktov zgorevanja je tipičen zvezen nelinearen kemijski proces s časovno zakasnitvijo. Namenjen je zmanjševanju količine škodljivih dušikovih oksidov v dimnih plinih, ki nastajajo pri različnih procesih zgorevanja fosilnih goriv. V laboratoriju se z injiciranjem amoniaka kemijsko obdelujejo plini, ki nastajajo v procesu zgorevanja. Delovanje procesa redukcije dušikovih oksidov je tako močno povezano z delovanjem procesa zgorevanja. Proces se lahko zažene le, če je proces zgorevanja v stanju obratovanja, zaustaviti pa ga je potrebno, še preden se ustavi proces zgorevanja, saj bi v nasprotnem primeru onesnaževali okolje z amoniakom.

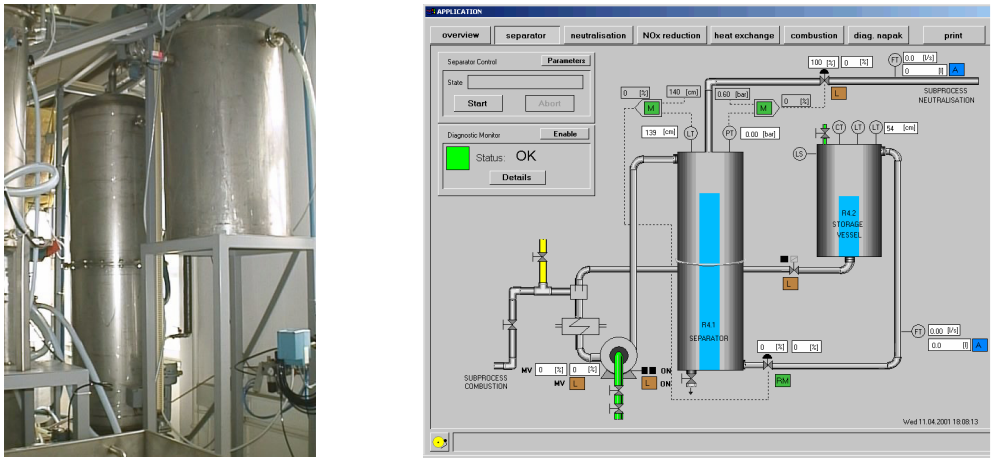
4.4 Proces priprave plina

Proces priprave plina (slika 4.3) je opisan podrobneje, saj je bil zanj realiziran vmesnik za vodenje v realnem času iz okolja Matlab/Simulink. Proces kot samostojni del nima pomembnejše funkcije, v povezavi z ostalimi deli procesnega laboratorija pa je namenjen:

- zajemanju in ohlajanju dimnih plinov, ki vsebujejo CO_2 – pri tem se dimni plini v prirejenem injektorju mešajo s hladilno vodo,

- ločevanju mešanice hladilne vode in dimnih plinov na ponovno uporabljivo vodo in na dimne pline pod tlakom primernim za proces kemijske nevtralizacije bazičnih odplak.

Za vpihovanje v nevtralizacijski reaktor morajo biti dimni plini ohlajeni in pod ustreznim tlakom – približno 0.5 bara. Od tlaka plina je namreč odvisna kvaliteta raztapljanja CO_2 v bazični odplaki in s tem učinkovitost nevtralizacije. Pri samostojnem delovanju procesa priprave plina lahko namesto dimnih plinov, ki nastajajo v procesu zgorevanja, uporabimo CO_2 iz jeklenke.

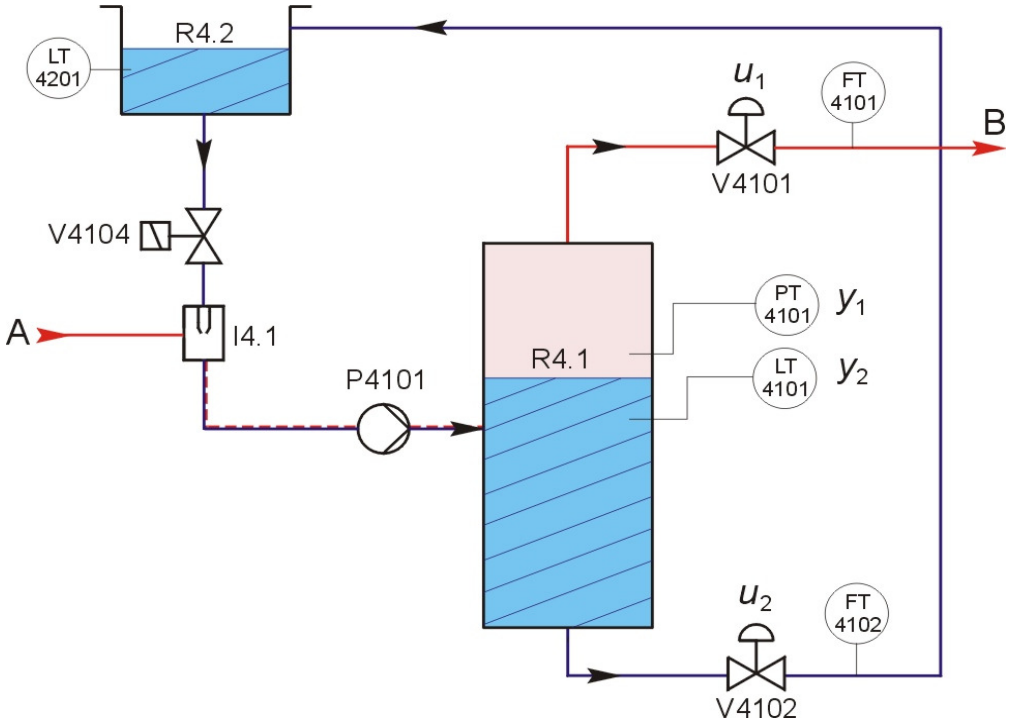


Slika 4.3: Levo: fotografija, desno: SCADA prikaz osnovnega vodenja procesa priprave plina

4.4.1 Procesna oprema enote za pripravo plina

Procesna enota za pripravo plina vsebuje naslednjo procesno opremo (slika 4.4):

- ločevalnik R4.1 opremljen z:
  - zveznim ventilom za plin V4101,
  - merilnikom pretoka plina iz ločevalnika FT4101,
  - zveznim ventilom za vodo V4102,
  - merilnikom pretoka vode iz ločevalnika FT4102,
  - analognim merilnikom nivoja vode LT4101,
  - merilnikom tlaka plina v ločevalniku PT4101,
- shranjevalna posoda R4.2 opremljena z:
  - analognim merilnikom nivoja vode LT4201,
  - nivojskim stikalom za indikacijo maksimalnega nivoja LS4201,
- dvopoložajni ventil V4104,
- injektor I4.1,
- hladilno telo H4.1,
- ročni ventil V4103 za vodo, s katero se podmazuje črpalka,
- črpalka na vodni obroč P4101 s frekvenčnim regulatorjem.



Slika 4.4: Shema procesa priprave plina

4.4.2 Delovanje enote za pripravo plina

Priprava plina temelji na odvzemu dimnih plinov iz dimnika peči s frekvenčno regulirano črpalko P4101. Črpalka P4101 črpa vodo iz shranjevalne posode R4.2 v tlačno posodo R4.1. Voda v injektorju iz vhoda A v cevovod povleče dimne pline, mešanica potuje naprej do tlačne posode, kjer se voda in plin ločita. Zaradi prisiljenega dotoka mešanice vode in plina v posodo in regulirnih ventilov V4101 in V4102 lahko v tlačni posodi dosežemo nadtlak, ki vodo skozi izpust na dnu ločevalnika in ventil V4102 potisne nazaj v zbiralno posodo, s čimer je tokokrog vode sklenjen. Na izhodu B pa iz ločevalnika prek ventila V4101 v nadaljnji proces kemijske nevtralizacije pod tlakom izstopa plin. Tlak plina v zgornjem delu ločevalnika mora biti večji od hidrostatičnega tlaka med nivojem vode v shranjevalni posodi in nivojem vode v ločevalniku. S tem je namreč omogočen odtok vode iz ločevalnika preko regulacijskega ventila V4102 v shranjevalno posodo. Nivo hladilne vode v ločevalniku reguliramo z zveznim ventilom V4102, tlak plina pa z zveznim ventilom V4101 ali s frekvenčno regulirano črpalko P4101. V shranjevalni posodi merimo nivo vode z analognim merilnikom nivoja LT4201. Po potrebi dotočimo vodo preko ročnega ventila, morebitno odvečno vodo pa zaznavamo z nivojskim stikalom LS4201 in jo preko preliva pretočimo v kanalizacijo. Če hladilna voda zelo dolgo kroži po zaključeni poti v sistemu, obstaja nevarnost pregretja vode [18], [39] [6].

4.5 Proces kemijske nevtralizacije

Proces kemijske nevtralizacije bazične tehnološke odplake z ogljikovim dioksidom je primer enostavnega šaržnega procesa. Tehnološka odplaka nastaja v industriji kot stranski produkt, npr. pri proizvodnji cementnih izdelkov. V laboratoriju, kjer ni predhodnega proizvodnega postopka, kjer bi odplaka nastajala, pripravimo vhodno odplako v obliki apnice ročno z doziranjem apna v vodo v zadrževalnih posodah. S tako nastalo bazično odplako polnimo reaktor, kjer se odplaka prepihuje z ogljikovim dioksidom. Pri reakciji nastaja apnenec. Po koncu nevtralizacije izpustimo odplako v usedalnik, kjer se apnenec useda, zato naprava ne obremenjuje okolja. Z napravo pa lahko nevtraliziramo tudi kisle odplake, saj ima za dodatno nevtralizacijo dodani dozirni posodi za bazo in kislino [18], [39].

5. Vmesnik za eksperimentiranje iz okolja Matlab/Simulink

V procesnem laboratoriju Odseka za sisteme in vodenje so se v preteklosti uporabljali različni pristopi k vodenju in eksperimentiranju. Sistem osnovnega in postopkovnega vodenja za zvezne procese, realiziran s programabilnimi logičnimi krmilniki (Mitsubishi) in SCADA nadzorno aplikacijo (Factory Link) na osnovnem vodenju zveznih procesov vsebuje PI regulacijo, postopkovno vodenje pa služi za privedbo sistema v delovno območje in varno avtomatsko zaustavitev sistema ob stanjih naprav in sistema, nevarnih za ljudi ali opremo. Eksperimentiranje na procesih za namene identifikacije ali proučevanja različnih naprednih metod vodenja pa se je običajno izvedlo direktno iz okolja Matlab/Simulink prek A/D in D/A pretvornikov osebnega računalnika. Razlog za to je predvsem fleksibilnost in hitrost tega okolja pri razvoju in spreminjanju algoritmov vodenja ter obdelavi rezultatov. Velika pomanjkljivost pa je bila popolna izključitev postopkovnega sistema vodenja na nivoju PLK-jev in SCADA sistema ob takem eksperimentiranju. Programabilni logični krmilniki zaradi svojih omejitev, predvsem omejenim obsegom pomnilniškega prostora in okornimi programskimi orodji za razvoj matematično zahtevnih algoritmov vodenja, niso ustrezna osnova za hiter razvoj in pogosto spreminjanje naprednih algoritmov vodenja v fazi njihovega razvoja in testiranja. V primerjavi z osebnimi računalniki pa zagotavljajo visoko stopnjo zanesljivosti delovanja, kar je pomembno pri zagotavljanju varnosti realnih procesov, kjer nastopajo visoki tlaki, temperature, strupeni plini itd.

V nadaljevanju je opisan razvoj vmesnika za vodenje podprocesa priprave plina iz okolja Matlab/Simulink v realnem času, ki je vgrajen v sistem postopkovnega vodenja laboratorija. Razvili smo uporabniško prijazno eksperimentalno okolje za vodenje in zajem podatkov v okolju Matlab/Simulink, s preverjanjem stanj naprav in sistema na nivoju programabilnih logičnih krmilnikov in upravljanjem s procesom iz SCADA nadzorne aplikacije. Celotno okolje predstavlja način vključevanja eksperimentiranja iz okolja Matlab/Simulink in preizkušanja različnih algoritmov vodenja, v industrijske procese, vodene s standardno hierarhično strukturo PLK-SCADA, ob zagotavljanju visoke stopnje varnosti delovanja samih procesov.

Metodologija načrtovanja sistema vodenja predhodno obstoječega sistema postopkovnega vodenja kot tudi vključitev eksperimentalnega vmesnika v obstoječ sistem, je temeljila na konceptu življenjskega cikla sistema procesnega vodenja, s poudarkom na fazah izgradnje sistema – specificiranje, načrtovanje in izvedba sistema vodenja. Podrobnosti o načrtovanju sistemov vodenja po konceptu življenskega cikla so opisane v dodatku A.

5.1 Opredelitev zahtev

Glavni cilj je razširitev sistema vodenja podprocesa priprave plina z vmesnikom za eksperimentiranje iz okolja Matlab/Simulink, brez zmanjšanja funkcionalnosti obstoječega sistema vodenja. Vmesnik smo želeli razviti tako, da se na nivoju programabilnih logičnih krmilnikov preverja stanja naprav in procesa, ob morebitnih nevarnih stanjih pa proces varno ustavi in prekine možnost vodenja procesa iz eksperimentalnega okolja na osebnem računalniku. Izbiro delovanja sistema in prikaz stanj opreme in procesa naj bo možen iz SCADA nadzorne aplikacije.

5.1.1 Analiza obstoječega stanja

Sistem vodenja procesnega laboratorija vsebuje naslednjo sistemsko in procesno programsko opremo:

- senzorje in aktuatorje za zajem signalov o procesu in vplivanje na proces,
- dva programirljiva logična krmilnika podjetja Mitsubishi (slika 5.1) s programskim orodjem Melsec Medoc plus verzija 2.40a podjetja Mitsubishi Electric za programiranje Mitsubishijevih PLK-jev:
  - Mitsubishi A2SH-S1 za osnovno vodenje,
  - Mitsubishi Q2AS-S1 za postopkovno vodenje.
- SCADA nadzorno aplikacijo *FactoryLink* verzija 6.6.0 podjetja USDATA za nadzorno vodenje.

Aktuatorje procesa in druge naprave sistema za vodenje (npr. regulatorje) lahko upravljamo na več načinov:

- Lokalno (L, ang. *Local*): v tem načinu upravljamo napravo s komandnega pulta,
- Daljinsko (R, ang. *Remote*): v tem načinu upravljamo napravo daljinsko, pri čemer ločimo med dvema možnostima:
  - Daljinsko avtomatsko (RA, ang. *Remote Automatic*): V tem načinu z napravo upravlja postopek, ki teče na PLK-ju.
  - Daljinsko ročno (RM, ang. *Remote Manual*): V tem načinu z napravo upravlja operater iz SCADA nadzorne aplikacije.

Izbiro med lokalnim in daljinskim delovanjem določamo fizično z analognimi A/M-postajami in vklopno-izklopnimi stikali, načina daljinskega delovanja (RA/RM) pa izberemo v SCADA nadzorni aplikaciji.

Osnovnemu vodenju je namenjen procesno manj zmogljiv PLK Mitsubishi A2SH-S1. Vsebuje algoritme osnovnega vodenja, s senzorji in aktuatorji v procesu je povezan preko analognih in digitalnih vhodov in izhodov. Procesno zmogljivejši PLK Mitsubishi Q2AS-S1 je namenjen postopkovnemu vodenju in vsem algoritmom, ki operirajo z realnimi števili (npr. PI regulator). Postopkovno vodenje vsebuje nižjenivojsko postopkovno vodenje procesa kemijske nevtralizacije, tj. faze, ki so sestavljene iz fazne logike in vmesnika fazne logike, in postopkovno vodenje kontinuirnih podprocesov. Programirljiva logična krmilnika sta med seboj povezana z Mitsubishijevo mrežo NetB. Mreža je izvedena s parico preko ustreznih modulov, hitrost prenosa podatkov med krmilnikoma pa je 1 Mbps.

Slika 5.1: Programirljiva logična krmilnika procesnega laboratorija

SCADA nadzorni sistem Factory Link za osnovno in postopkovno vodenje je v sistemu vodenja procesnega laboratorija namenjen zajemu procesnih podatkov in

podatkov faz iz PLC-ja ter posredovanju podatkov procesnim napravam in fazam, npr. ukazov in parametrov. PLC-ja sta z nadzornim sistemom povezana preko ethernet mreže in ustreznih modulov, za komunikacijo pa skrbi vmesnik Mecom, ki je del Factory Link-a.

Podrobnosti o sistemski in procesni programski opremi procesnega laboratorija so opisane v [19], [5], [18].

5.1.2 Postopkovne zahteve

Postopkovne zahteve za podproces priprave plina so:

- Privedba procesa, ob izpolnjenih pogojih za zagon sistema in ustreznem ukazu operaterja iz nadzorne SCADA aplikacije, iz mirujočega stanja prek vmesnega zagonskega stanja v stanje normalnega obratovanja, v katerem želimo vzdrževanje želenih vrednosti tlaka in nivoja, podanih prek nadzorne aplikacije, v ločevalniku.

- Privedba procesa, ob ukazu operaterja in izpolnjenih pogojih, iz mirujočega stanja prek vmesnega zagonskega stanja v eksperimentalno stanje, v katerem na aktuatorje procesa vplivamo iz okolja Matlab/Simulink.

- Privedba procesa iz stanja normalnega obratovanja ali eksperimentalnega stanja v mirujoče stanje prek vmesnega stanja ustavljanja ob ustreznem ukazu operaterja.

- Privedba procesa iz poljubnega nemirujočega stanja v mirojoče stanje, ob neizpolnjenih pogojih za varno obratovanje procesa.

5.1.3 Zahteve po vmesnikih

Za želen sistem postopkovnega vodenja podprocesa priprave plina potrebujemo naslednje vmesnike:

- proces <-> krmilnik Mitsubushi (že realiziran s senzorji in aktuatorji, prek analognih in digitalnih vhodov in izhodov krmilnika Mitsubishi A2SH-S1),

- krmilnik Mitsubushi A2SH-S1 <-> Mitsubushi Q2AS-S1 (že izveden z Mitsubishijevo NetB mrežo, potrebna je razširitev vmesnika za izvedbo okolja za eksperimentiranje),

- krmilnik Mitsubushi <-> Factory Link SCADA aplikacija (že realiziran prek ethernet mreže in Mecon programskega vmesnika).

- uporabnik <-> Factory Link SCADA nadzorna aplikacija (že realiziran z grafičnimi prikazi procesa, signalizacijo vrednosti procesnih spremenljivk in stanj naprav, alarmov, ukaznimi okni itd, potrebna je dopolnitev aplikacije za izvedbo okolja za eksperimentiranje),

- vmesnik med aplikacijo Matlab/Simulink <-> Mitsubishi krmilnik.

5.2 Specifikacije sistema vodenja

Specifikacije sistema vodenja procesa priprave plina so sestavljene iz:

- specifikacij postopkovnega vodenja, podanih z diagramom prehajanj stanj PSTD,
- naborom vrednosti spremenljivk postopkovnega vodenja,
- specifikacij akcij posameznih stanj postopka in pogojev za prehode med stanji, opisanimi s psevdo jezikom.

Podroben opis specifikacij je podan v dodatku B.

5.3 Izvedba vmesnika za eksperimentiranje iz okolja Matlab/Simulink

Izvedba komunikacijskega vmesnika med okoljem Matlab/Simulink in Mitsubishi krmilniki je bila realizirana na osnovi DDE (ang. *Dynamic Data Exchange*) komunikacije, ki je sistemsko podprta komunikacija v okolju Windows, na osnovi sledečih argumentov:

- Funkcije za DDE komunikacijo so že vključene v osnovni paket Matlab (od verzije 5.2 naprej), kar se za OPC standard, ki je v avtomatizaciji tudi že zelo uveljavljen, obeta šele v prihajajoči verziji Matlab 7.

- Na trgu je veliko število komercialno dostopnih DDE strežniških aplikacij, ki znajo komunicirati s programabilnimi logičnimi krmilniki različnih proizvajalcev po njihovih internih protokolih, z drugimi aplikacijami pa podatke izmenjujejo v standardnih formatih, definiranih znotraj DDE standarda. Tudi večina OPC strežnikov, ki igra podobno vlogo komunikatorja med PLK-ji in okoljem Windows, ima možnost izmenjave podatkov z drugimi aplikacijami tudi po standardu DDE. Za izvedbo vmesnika je bil tako nabavljen paket MelDDE proizvajalca Beijer Electronics, ki podpira komunikacijo z Mitsubishi krmilniki prek serijskega in ethernetnega kanala.

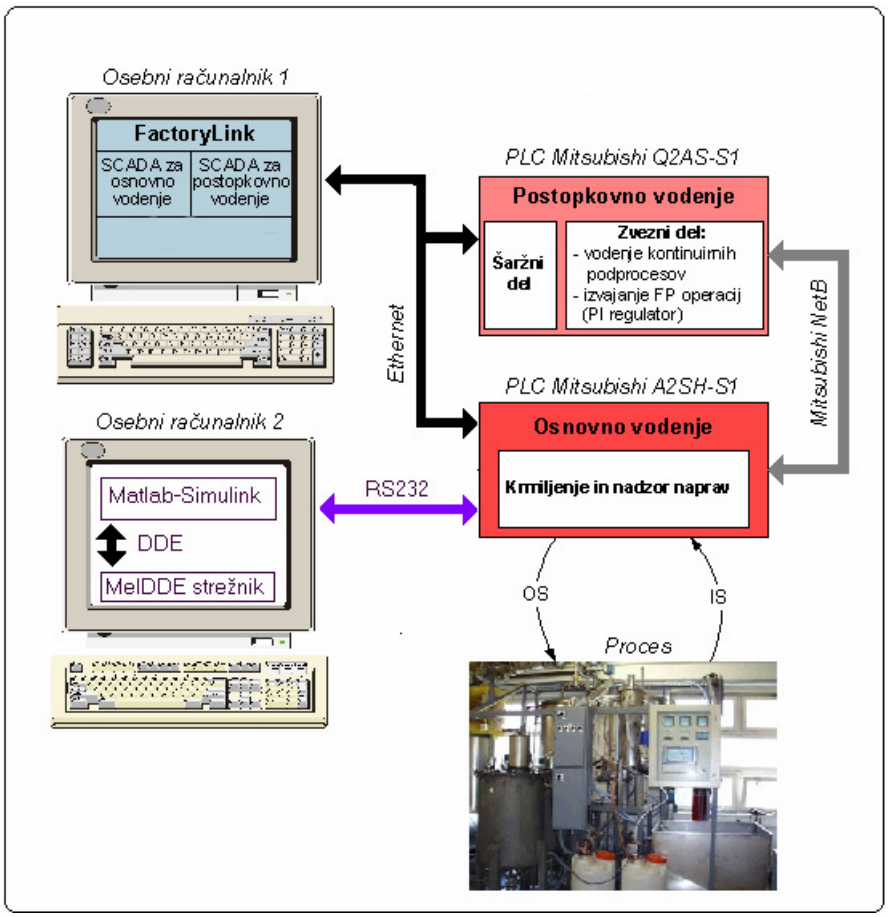
5.3.1 Koncept okolja za eksperimentiranje iz okolja Matlab/Simulink

Realiziran koncept vodenja iz okolja Matlab/Simulink je prikazan na sliki 5.2.

Osnovnemu vodenju je namenjen procesno manj zmogljiv PLC Mitsubishi A2SH-S1. Vsebuje algoritme osnovnega vodenja procesnih naprav, s katerimi je povezan preko analognih in digitalnih vhodov in izhodov.

Procesno zmogljivejši PLC Mitsubishi Q2AS-S1 je namenjen postopkovnemu vodenju in vsem algoritmom, ki operirajo z realnimi števili (npr. PI regulator). Postopkovno vodenje vsebuje nižjenivojsko postopkovno vodenje procesa kemijske nevtralizacije, tj. faze, ki so sestavljene iz fazne logike in vmesnika fazne logike (na sliki označen s kratico PLI), in postopkovno vodenje kontinuirnih podprocesov.

Programirljiva logična krmilnika sta med seboj povezana z Mitsubishijevo mrežo NetB. Mreža je izvedena s parico preko ustreznih modulov, hitrost prenosa podatkov med krmilnikoma pa je 1 Mbps.



Slika 5.2: Koncept vodenja iz okolja Matlab/Simulink

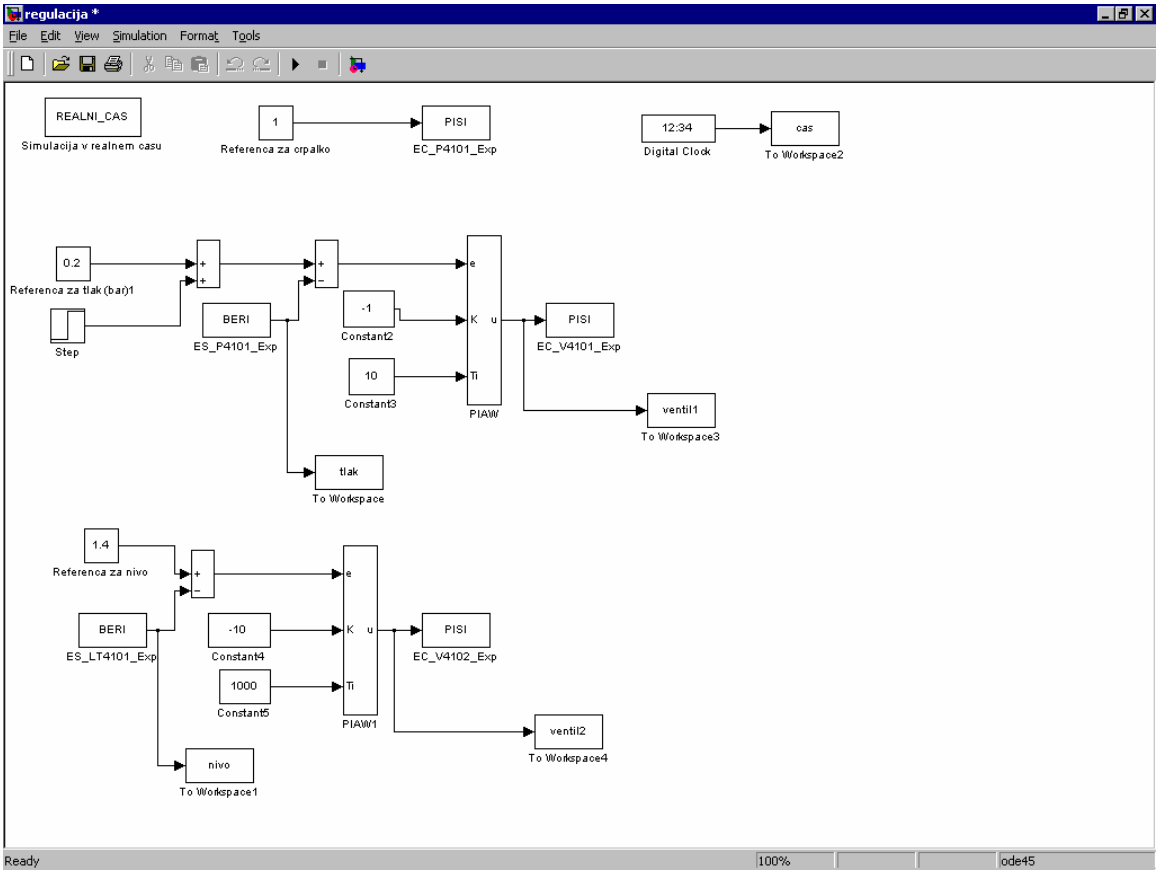
Eden izmed osebnih računalnikov je namenjen implementaciji SCADA nadzornega sistema za osnovno in postopkovno vodenje izdelanega s programskim paketom Factory Link. Sistem za nadzor in vodenje je v sistemu vodenja procesnega laboratorija namenjen zajemu procesnih podatkov in podatkov faz iz PLK-ja (stanja naprav in faz, procesne veličine) ter posredovanje podatkov procesnim napravam in fazam, npr. ukazov in parametrov. PLK-ja sta z nadzornim sistemom povezana preko Etherneta in ustreznih modulov, za komunikacijo pa skrbi vmesnik Mecom, ki je del Factory Link-a.

Na drugem računalniku sta nameščena paketa Matlaba in MelDDE strežnik. Strežnik poženemo iz Matlaba ob inicializaciji DDE komunikacije, s krmilnikom A2SH-S1 pa komunicira prek serijskega izhoda. Ker se na izhode krmilnika ukazi za stanja aktuatorjev, ki jih dobimo iz Matlaba, prepisujejo (zaradi varnega eksperimentiranja) samo v stanju *Experiment Running*, je prvi korak, ki ga moramo storiti, da iz nadzorne aplikacije prek okna postopkovnega vodenja v grafičnem prikazu podprocesa priprave plina zaženemo postopek v omenjeno stanje. Sledi inicializacija DDE komunikacije v aplikaciji Matlab.

Po uspešni inicializaciji DDE komunikacije lahko zaženemo Simulink shemo za vodenje procesa. V shemi lahko uporabljamo poljubne sistemske ali uporabniško razvite (S-funkcije) gradnike simulacijskih shem okolja Simulink, pravilno vključiti moramo le ustrezne Simulink bloke (S-funkcije) za DDE komunikacijo in izvajanje simulacije v realnem času. Primer ustrezne Simulink simulacijske sheme prikazuje slika 5.3.

Vsak čas vzorčenja, podan v datoteki za inicializacijo komunikacije, se iz Simulinka požene datoteka, v kateri je ukaz za branje signalov senzorjev iz krmilnika in ukaz za pošiljanje vrednosti vplivnih veličin aktuatorjev. Preverjamo tudi, ali imamo dovoljenje postopka za eksperimentiranje. Če dovoljenja nimamo, krmilniku pošiljamo ukaze za vrednosti vplivnih veličin, ki v nobenem primeru ne morejo pripeljati sistema v, za človeka ali naprave, nevarno stanje. Slednje je le dodaten previdnostni ukrep, saj že postopkovno vodenje takrat, ko odvzame dovoljenje za eksperimentiranje, onemogoči prepisovanje ukazov, dobljenih iz Matlaba na dejanske izhode krmilnika, hkrati pa samo varno pripelje sistem v ustavljeno stanje

Po končanem eksperimentu prekinemo simulacijo in poženemo datoteko, s katero zapremo DDE komunikacijski kanal in MelDDE strežniško aplikacijo.



Slika 5.3: Primer simulacijske sheme vodenja iz okolja Matlab/Simulink

Pri vodenju sistema iz okolja Matlab/Simulink je pomembno, da med izvajanjem samega vodenja na osebnem računalniku paralelno ne tečejo druge aplikacije, kar ima lahko za posledico časovne zakasnitve pri simulaciji, simulacija ne teče več v realnem času. Minimalni čas vzorčenja, ki ga lahko dosežemo, je odvisen od zmogljivosti računalnika, na katerem izvajamo simulacijo, saj je vsota časa DDE pisanja in branja, ter časa izvajanja simulacije v Simulinku. Tako je bil za zgornji primer simulacijske sheme, ki vsebuje dva PI regulatorja, podana v uporabniško definirani S-funkciji, pri vodenju na manj zmogljivem PC računalniku (P166 Mhz, 64 Mb RAM) minimalni možni čas vzorčenja 270 ms, na bolj zmogljivem PC računalniku (PII 400Mhz, 128 Mb RAM) pa 150 ms.

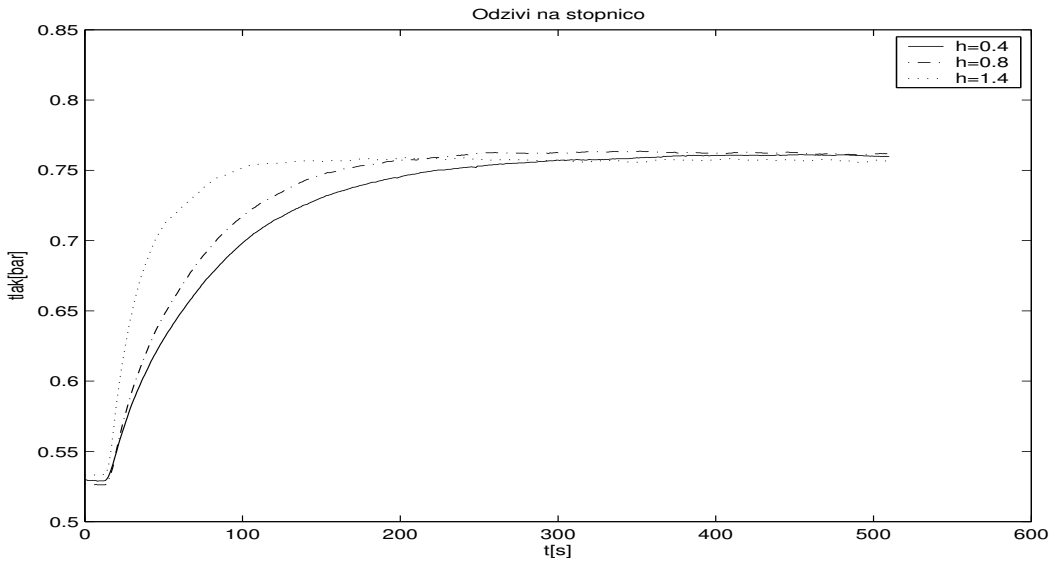
Podrobnosti o izvedbi vmesnika za eksperimentiranje iz okolja Matlab/Simulink na osnovi DDE komunikacije so opisane v dodatku C.

6. Identifikacija procesa priprave plina z Gaussovimi procesi

V tem poglavju bomo predstavili rezultate identifikacije procesa priprave plina z Gaussovimi procesi. Rezultate smo zajemali z vmesnikom za eksperimentiranje iz okolja Matlab/Simulink, opisanim v prejšnjem poglavju.

Proces priprave plina (ločevalnik) je multivariabilen nelinearen proces, kjer kot regulirani veličini y_1 in y_2 nastopata tlak p in nivo vode h v tlačni posodi R4.1 Regulirna signala sta u_1 za odprtost ventila V4101 na izhodu plina iz posode in u_2 za odprtost ventila V4102 na izhodu tekočine iz posode. Glavni viri nelinearnosti procesa so nelinearne odvisnosti pretokov skozi ventile od tlačnih razlik na ventilih in nelinearna odvisnost dinamike tlaka plina od nivoja vode v tlačni posodi. Proces je multivariabilen, z dvema vhodoma in dvema izhodoma, z dinamičnimi križnimi povezavami. Za problem vodenja tlaka p je bil ločevalnik obravnavan kot nelinearen univariabilen sistem z vhodno regulirno veličino u_1 (odprtost ventila V4101). Nivo, voden s PI regulacijsko zanko, je bil upoštevan kot eden izmed merljivih dejavnikov nelinearnosti. Vplivi križnih povezav v procesu so bili obravnavani kot nemerljive motnje, vendar so zaradi velike razlike zaprtozančnih odzivov med glavnima regulacijskima zankama zanemarljivi.

Slika 6.1 prikazuje odprtozančne odzive tlaka na stopničasto spremembo položaja ventila V4101 od $u_1 = 0.51$ do $u_1 = 0.45$ pri različnih vrednostih nivoja.



Slika 6.1 - Odprtozančni odzivi tlaka pri različnih vrednostih nivoja

6.1 Identifikacija z Gaussovimi procesi

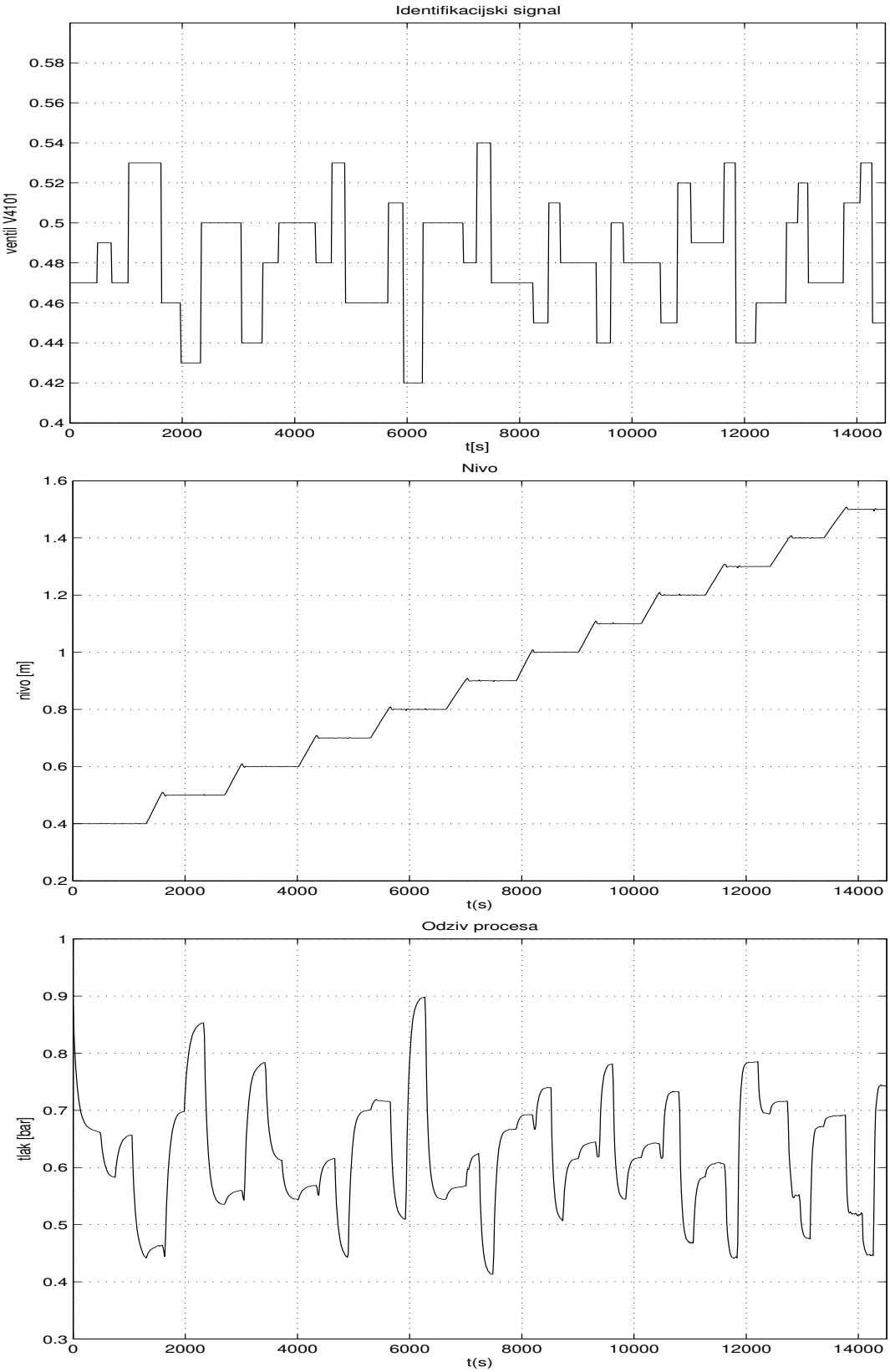
Za nelinearno prediktivno vodenja tlaka ločevalnika je bil najprej identificiran diskretni GP model prvega reda oblike:

$$p(k+1) = f\big(p(k), u_1(k), h(k)\big), \tag{6.1}$$

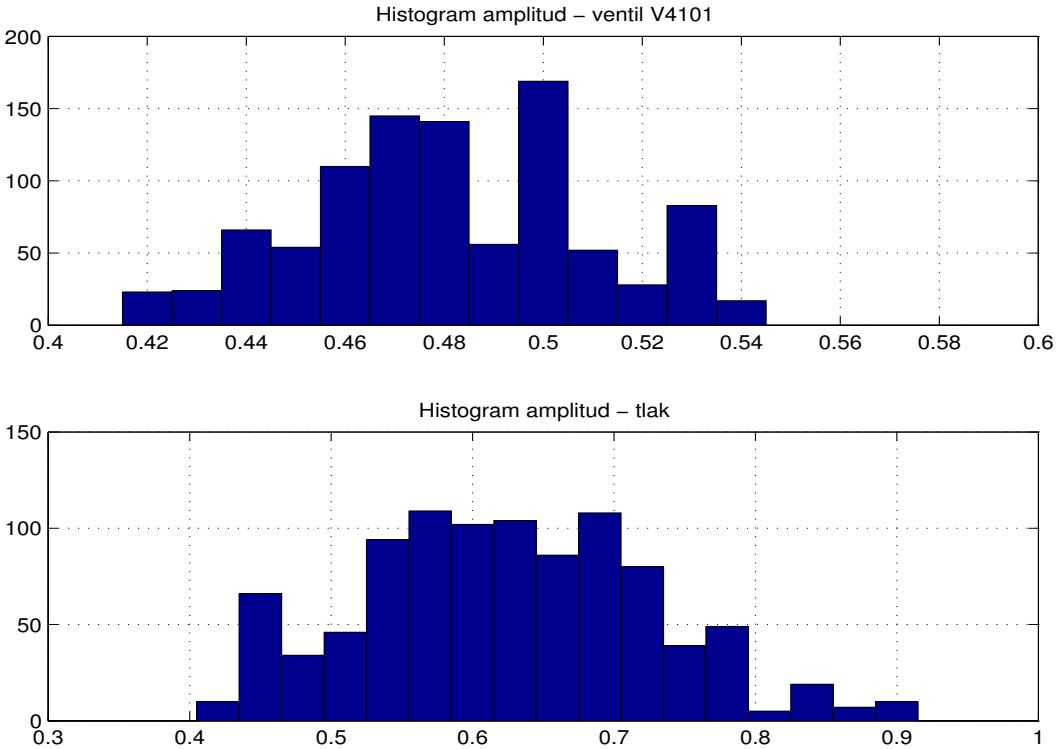
kjer funkcija f predstavlja Gaussov proces, vektor regresorjev $\big(p(k), u_1(k), h(k)\big)$ predstavljajo vrednosti tlaka p, odprtosti regulirnega ventila u_1 in vrednosti nivoja h v diskretnem časovnem trenutku $t = k$, $p(k+1)$ pa napovedano vrednost tlaka v naslednjem diskretnem časovnem trenutku $t = k+1$. Model prvega reda je bil izbran zaradi ujemanja z dejanskim redom sistema (odvisnost tlaka p od odprtosti regulirnega ventila u_1 lahko zapišemo z nelinearno diferencialno enačbo prvega reda [36] [15]).

Identifikacijske signale (položaj ventila V4101 u_1, nivo v ločevalniku h) in odziv procesa (tlak *p* v ločevalniku) prikazuje slika 6.2. Pri odsekoma konstantni vrednosti nivoja smo spreminjali odprtost ventila V4101. Tako smo poskušali na celotnem delovnem področju nivoja ločevalnika zajeti dinamiko tlaka. Pri identifikaciji je potrebno izbrati optimalno dolžino trajanja identifikacije, s katero naraščajo informacije o obnašanju procesa v različnih delovnih točkah, s številom vzorcev identifikacijskega signala se veča tudi dimenzija kovariančne matrike učne množice $\mathbf{K}_n$, s tem pa računska zahtevnost učenja hiperparametrov in kasneje predikcije na osnovi GP modela (zahtevnost računanja inverzne matrike raste s tretjo potenco dimenzije matrike, inverz pa se računa pri vsakem koraku učenja oziroma optimizacije hiperparametrov). Čas identifikacije in čas vzorčenja sta bila izbrana iterativno, ob zahtevi za računsko obvladljivost velikosti kovariančne matrike, za uporabo GP modela pri prediktivnem vodenju v realnem času. Pri izbranem času vzorčenja $T_s = 15s$ in izbranem času identifikacije $T_{ide} = 14505s$ znaša velikost učne množice $\mathcal{D}$ in s tem kovariančne matrike $\mathbf{K}_n$, $n = 967$.

Slika 6.3 prikazuje histograma vhodnega in izhodnega identifikacijskega signala, ki podajata informacijo o identificiranem področju procesa.



Slika 6.2: Identifikacijski signali, odziv procesa



Slika 6.3: Histograma amplitud vhodnega in izhodnega identifikacijskega signala

Pri modeliranju z GP je bila uporabljena kovariančna funkcija oblike:

$$C(\mathbf{x}_p, \mathbf{x}_q) = v\, \mathrm{e}^{-\frac{1}{2}\sum_{d=1}^{3} w_d \left(\mathbf{x}_p^d - \mathbf{x}_q^d\right)} + v_0 \tag{6.2}$$

Pomen hiperparametrov $\boldsymbol{\Theta} = (v_0, v, w_1, w_2, w_3)$, določenih z metodo največje podobnosti (uporabljena je bila optimizacijska metoda konjugiranih gradientov z analitičnim izračunom parcialnih odvodov) je:

- v_0 je varianca belega šuma na izhodu procesa,
- v je skalirni faktor kovariance, določene s kovariančno funkcijo,
- w_1 določa relativno vlogo (utež) razdalje tlaka p pri izračunu kovariance,
- w_2 določa relativno vlogo (utež) razdalje odprtosti regulirnega ventila u_1 pri izračunu kovariance,
- w_3 določa relativno vlogo (utež) razdalje nivoja h pri izračunu kovariance.

Tabela 6.1 prikazuje vrednosti dobljenih optimalnih hiperparametrov za identifikacijske signale, kjer je bila izhodnemu signalu tlaka odšteta srednja vrednost

(pri modeliranju z GP vedno predpostavimo ničelno srednjo vrednost vektorja izhodnih točk učne množice) . Zaradi izognitve lokalnim minimumom smo optimizacijo večkrat ponovili z različnimi začetnimi vrednostmi hiperparametrov.

v_0	v	w_1	w_2	w_3
2.9145e-005	0.1162	20.2759	78.0774	0.1517

Tabela 6.1 – Vrednosti hiperparametrov

Po določitvi hiperparametrov kovariančne funkcije smo z enokoračno predikcijo, z različnimi načini propagiranja verjetnostnih porazdelitev preteklih izhodov modela, simulirali odziv procesa na identifikacijske in validacijske signale. Uporabljene in primerjane so bile tri različne metode:

- t.i. "*eksaktna*" metoda, kjer ob predpostavkah o normalno porazdeljenem izhodu modela pri normalno porazdeljenih vhodih v model, pri uporabljeni obliki kovariančne funkcije (enačba 6.2), analitično določimo srednjo vrednost in varianco izhoda modela (enačbi 2.66, 2.68).

- t.i. "*Taylorjeva aproksimacija*" propagiranja, kjer ob predpostavkah o normalno porazdeljenem izhodu modela pri normalno porazdeljenih vhodih v model, za izračun izhoda in variance modela uporabimo aproksimacijo z razvojem v Taylorjevo vrsto za srednjo vrednost in varianco normalno porazdeljenih vhodov v model (enačbi 2.73, 2.77).

- t.i. "*naivna*" metoda, kjer za predikcijo srednje vrednosti in variance modela uporabimo samo srednje vrednosti predhodnih izhodov modela. Tak način propagiranja izhaja iz analogije umetnih nevronskih mrež in GP (enačbi 2.37, 2.38).

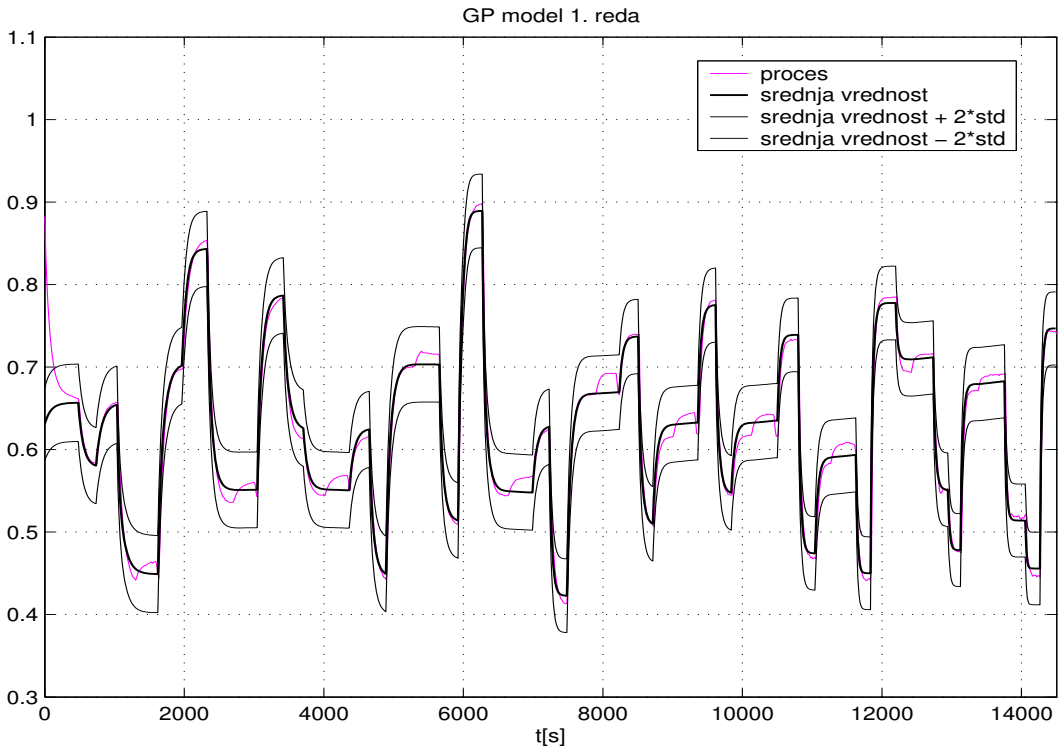
Na slikah 6.4 do 6.12 so prikazani:

- Primerjava odzivov procesa in GP modela na identifikacijski signal, s pasovoma dvakratnih standardnih deviacij modela (varianca modela je vsota variance predikcije in variance pogreška modela), za metode :
  - "*eksaktna*" metoda (slika 6.4, slika 6.5),
  - "*Taylorjeva aproksimacija*" (slika 6.6, slika 6.7),

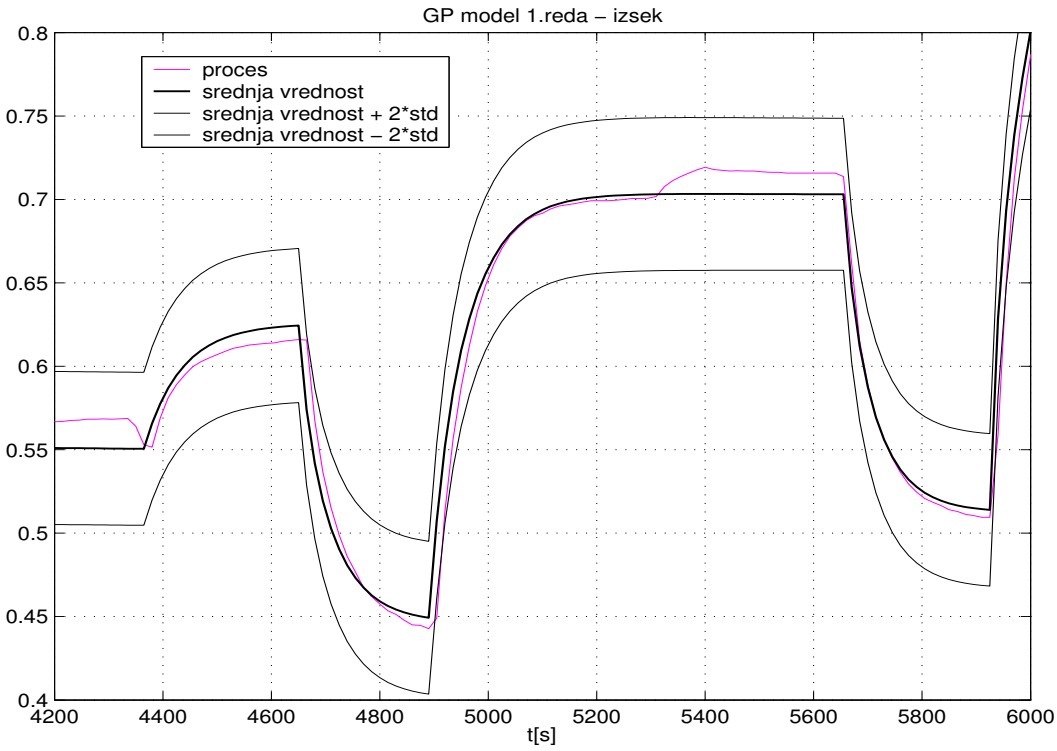
- "*naivna*" metoda (slika 6.8, slika 6.9).

- Pogrešek identifikacije, kot razlika med odzivom procesa in srednjo vrednostjo modela, za metode:
  - "*eksaktna*" metoda (slika 6.10),
  - "*Taylorjeva aproksimacija*", "*naivna*" metoda - srednji vrednosti predikcije po obeh metodah sta enaki, (slika 6.11).

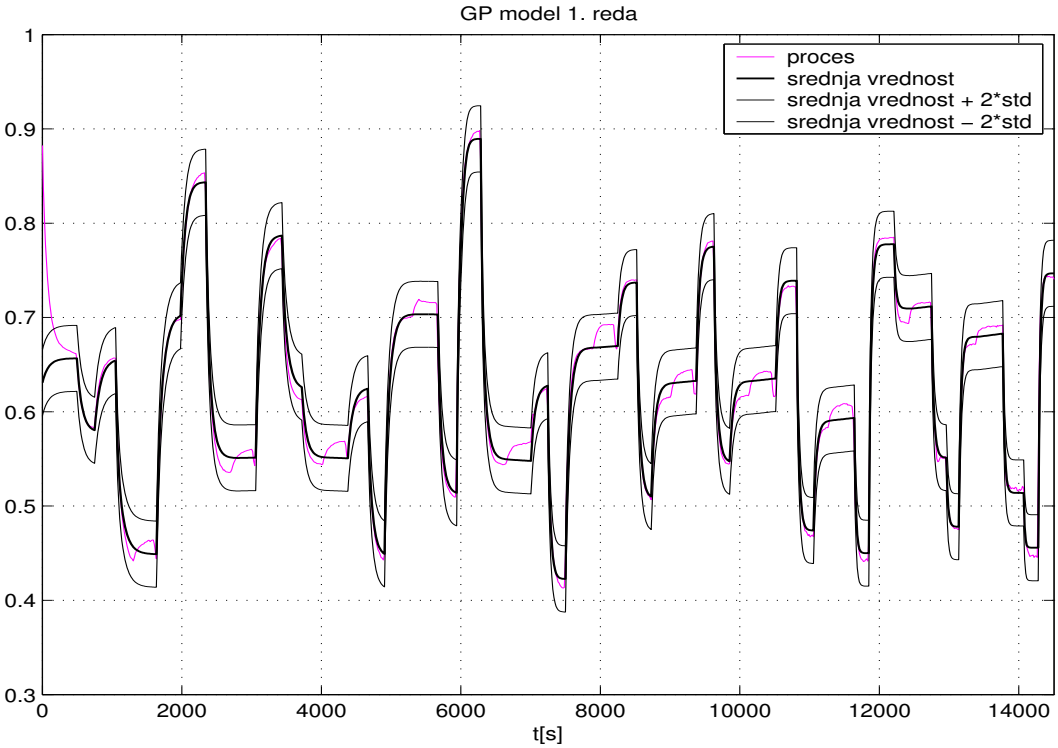
- Primerjava standardnih deviacij predikcij modelov z različnimi načini propagiranja verjetnostnih porazdelitev preteklih izhodov (slika 6.12).



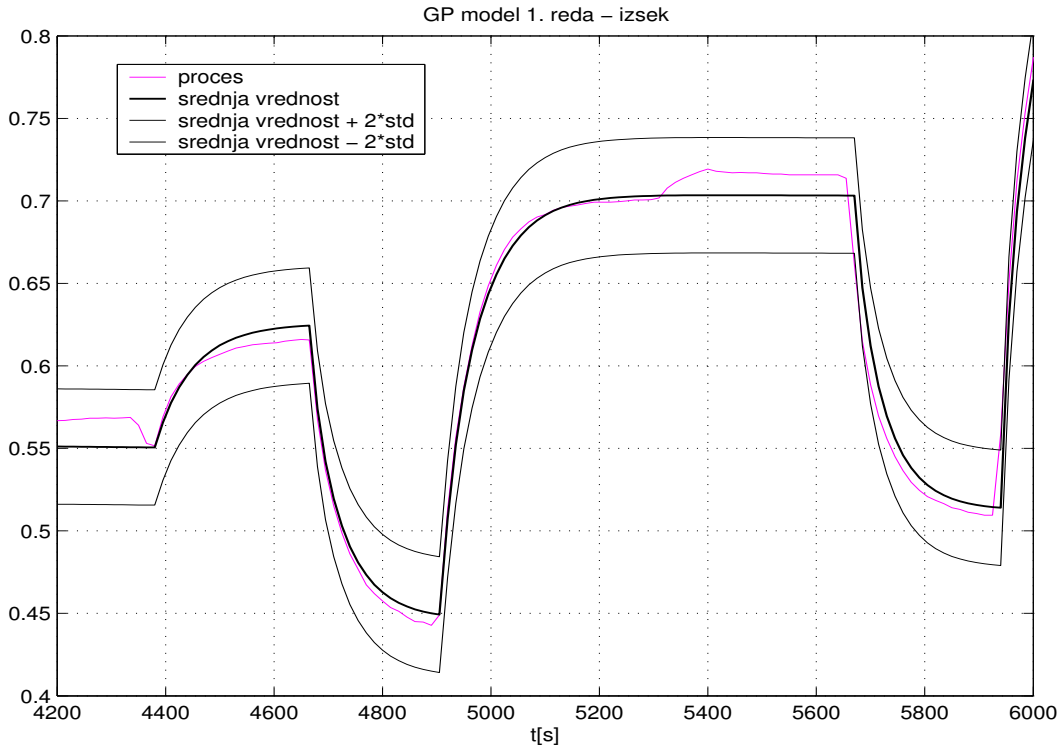
Slika 6.4: Odziv modela ("*eksaktna*" metoda) pri identifikaciji



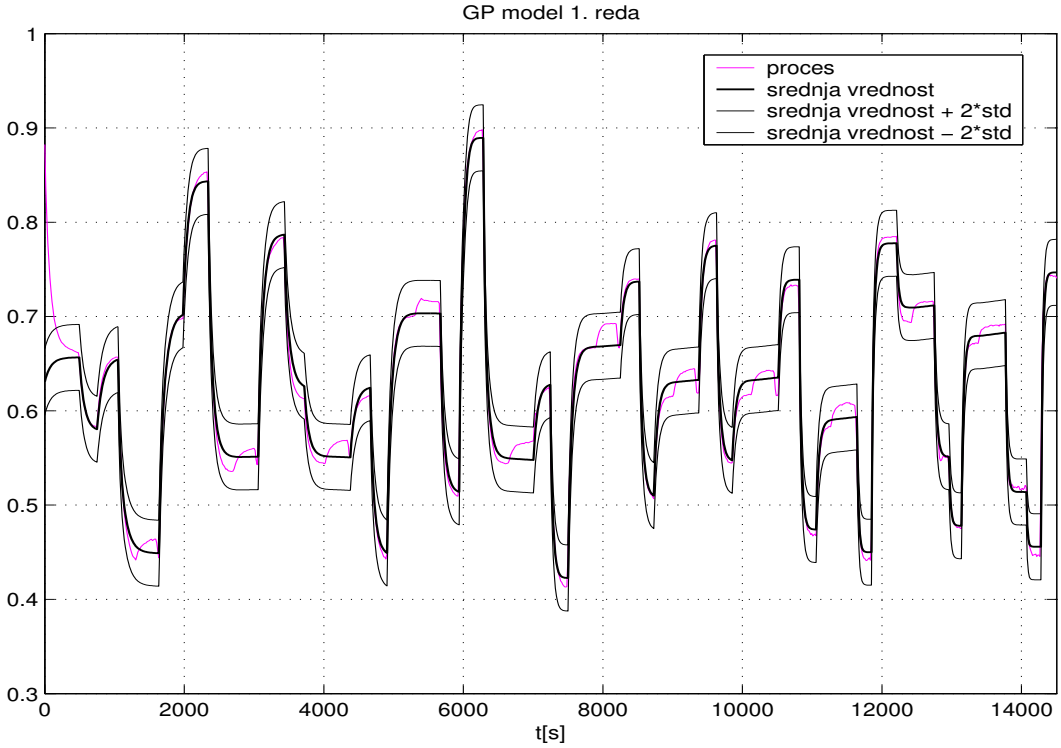
Slika 6.5: Odziv modela ("*eksaktna*" metoda) pri identifikaciji – izsek



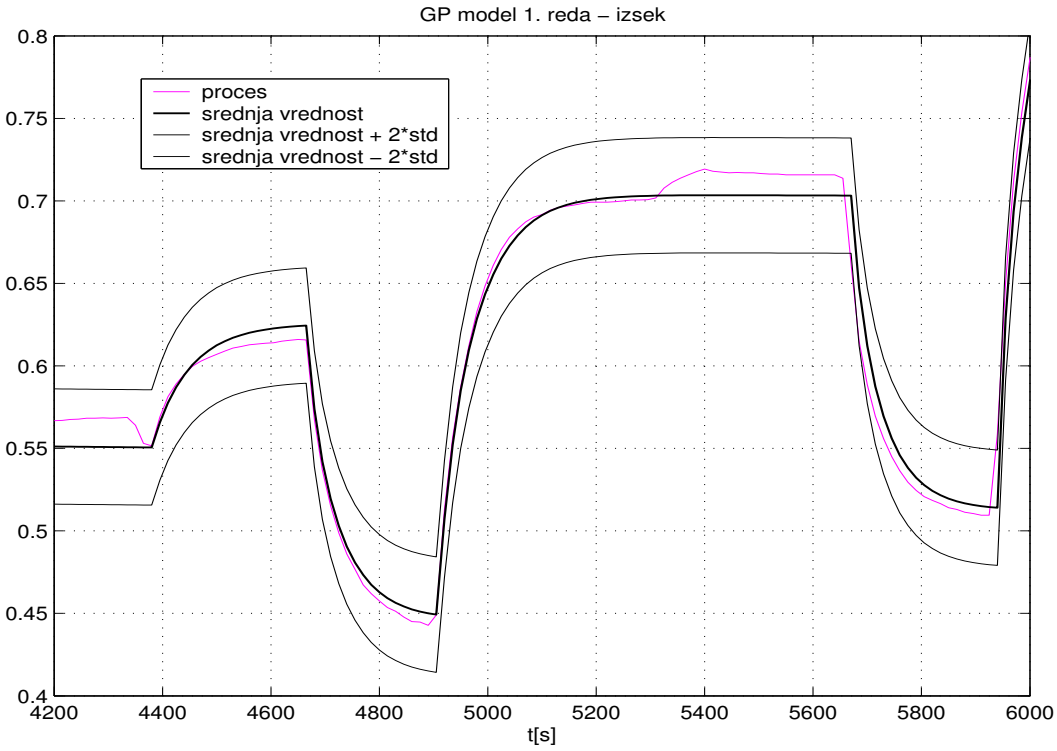
Slika 6.6: Odziv modela ("Taylorjeva aproksimacija") pri identifikaciji



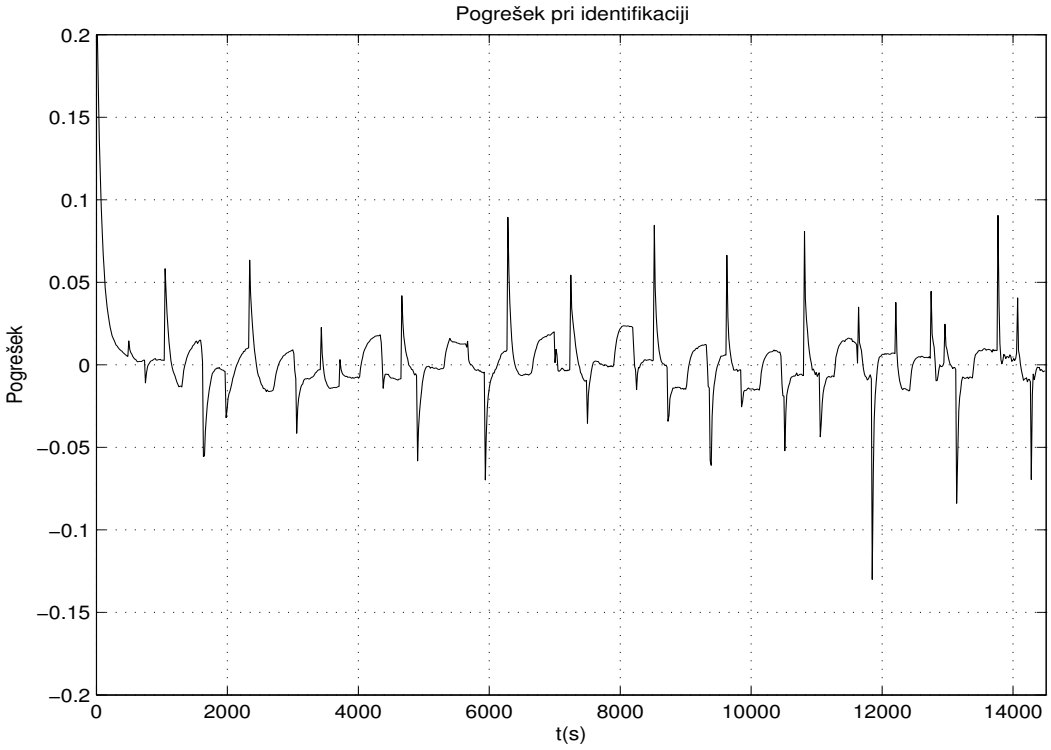
Slika 6.7: Odziv modela (“Taylorjeva aproksimacija”) pri identifikaciji – izsek



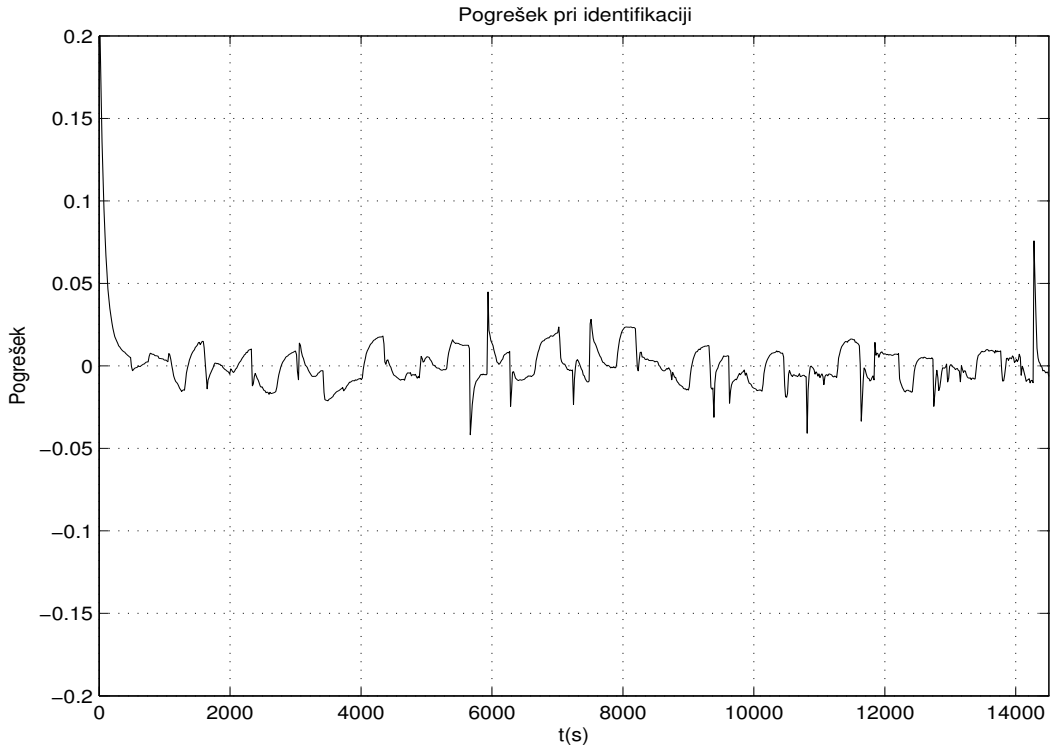
Slika 6.8: Odziv modela (“*naivna*” metoda) pri identifikaciji



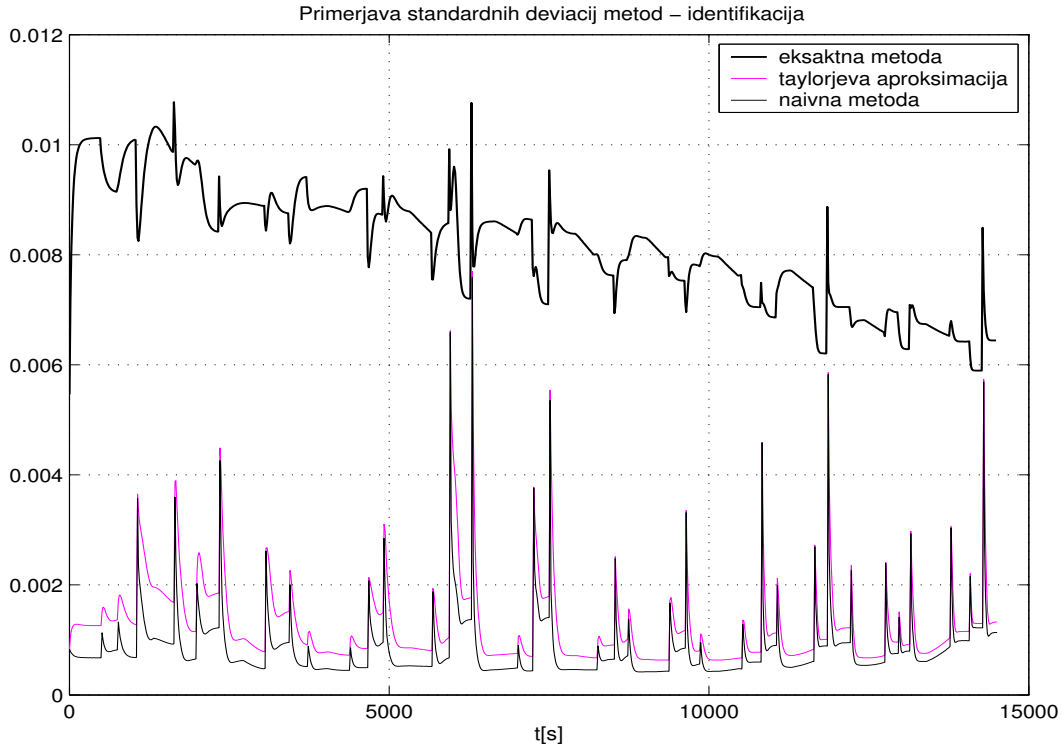
Slika 6.9: Odziv modela (“*naivna*” metoda) pri identifikaciji – izsek



Slika 6.10: Pogrešek identifikacije (“*eksaktna*” metoda)



Slika 6.11: Pogrešek identifikacije (“*Taylorjeva aproksimacija*” , “*naivna*” metoda)



Slika 6.12: Primerjava standardnih deviacij metod pri identifikaciji

GP model je bil preizkušen tudi s testnim, validacijskim signalom (slika 6.13). Histograma vhodnega in izhodnega validacijskega signala, ki podajata informacijo o vrednotenem področju procesa, sta prikazana na sliki 6.14.

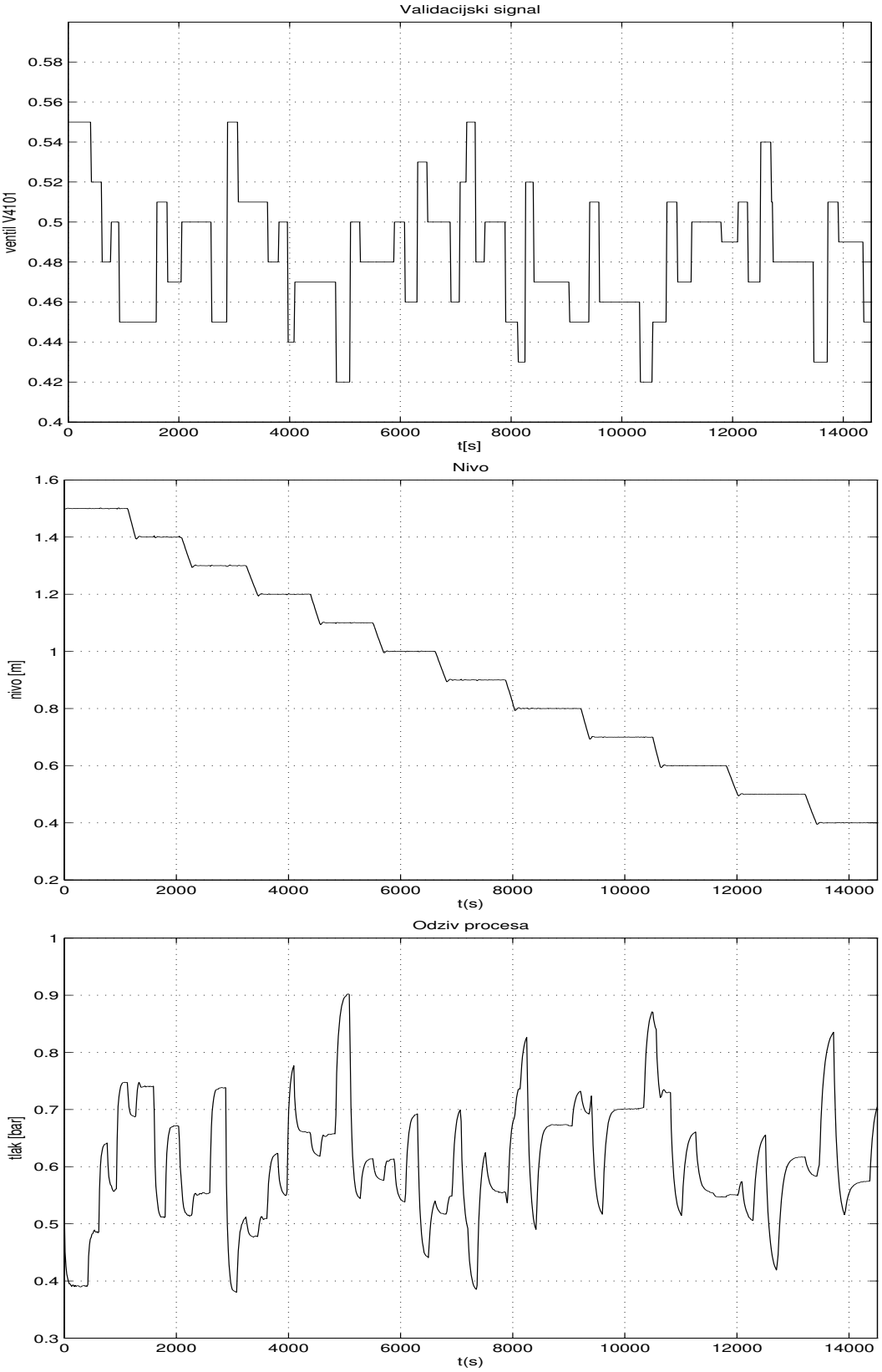
Na slikah 6.15 do 6.25 so prikazani:

- Primerjava odzivov procesa in GP modela na validacijski signal, s pasovoma dvakratnih standardnih deviacij modela (varianca modela je vsota variance predikcije in variance pogreška modela), za metode :
  - "*eksaktna*" metoda (slika 6.15, slika 6.16),
  - "*Taylorjeva aproksimacija*" (slika 6.17, slika 6.18),
  - "*naivna*" metoda (slika 6.19, slika 6.20).

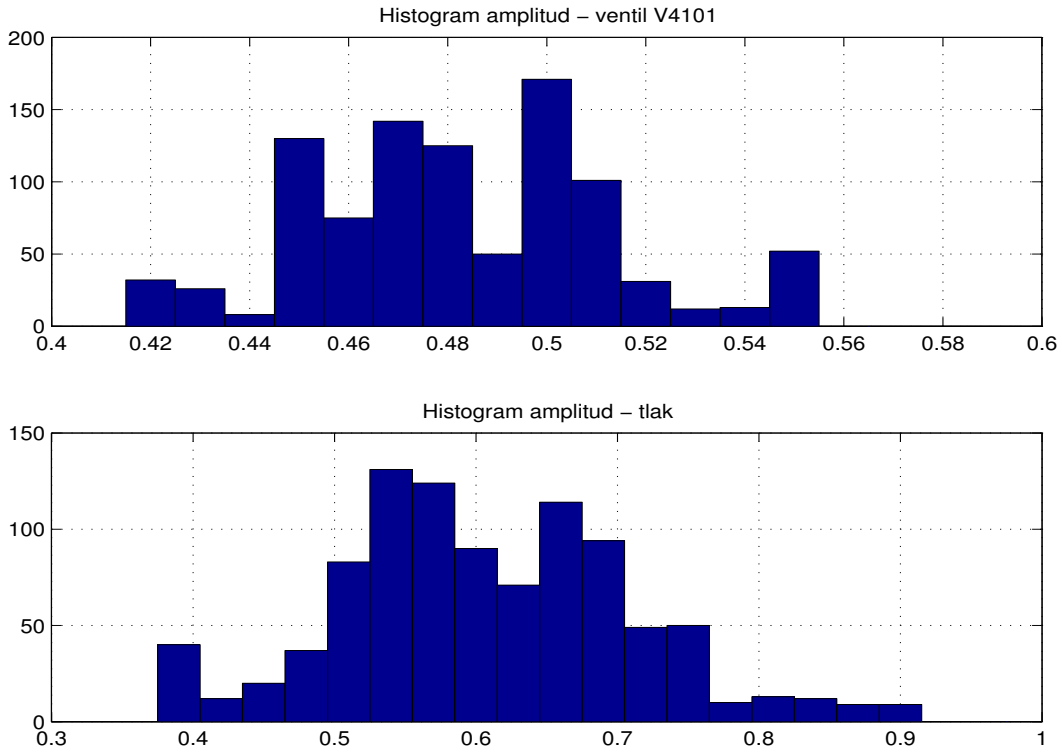
- Pogrešek validacije, kot razlika med odzivom procesa in srednjo vrednostjo modela, za metode:
  - "*eksaktna*" metoda (slika 6.21),
  - "*Taylorjeva aproksimacija*", "*naivna*" metoda (slika 6.22).

- Avtokorelacija Φ_{ee} in križna korelacija pogreška z vhodnim validacijskim signalom Φ_{ue}, kot statistična pokazatelja kvalitete identifikacije, za metode:
  - "*eksaktna*" metoda (slika 6.23),
  - "*Taylorjeva aproksimacija*", "*naivna*" metoda (slika 6.24).

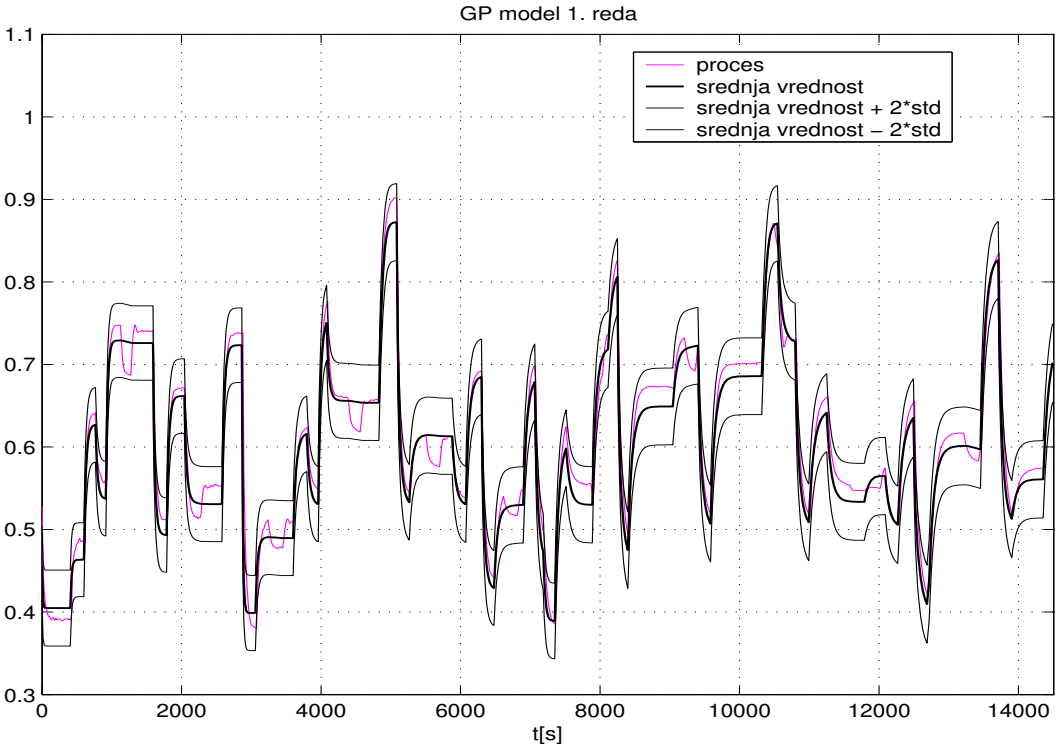
- Primerjava standardnih deviacij predikcij modelov z različnimi načini propagiranja verjetnostnih porazdelitev preteklih izhodov (slika 6.25).



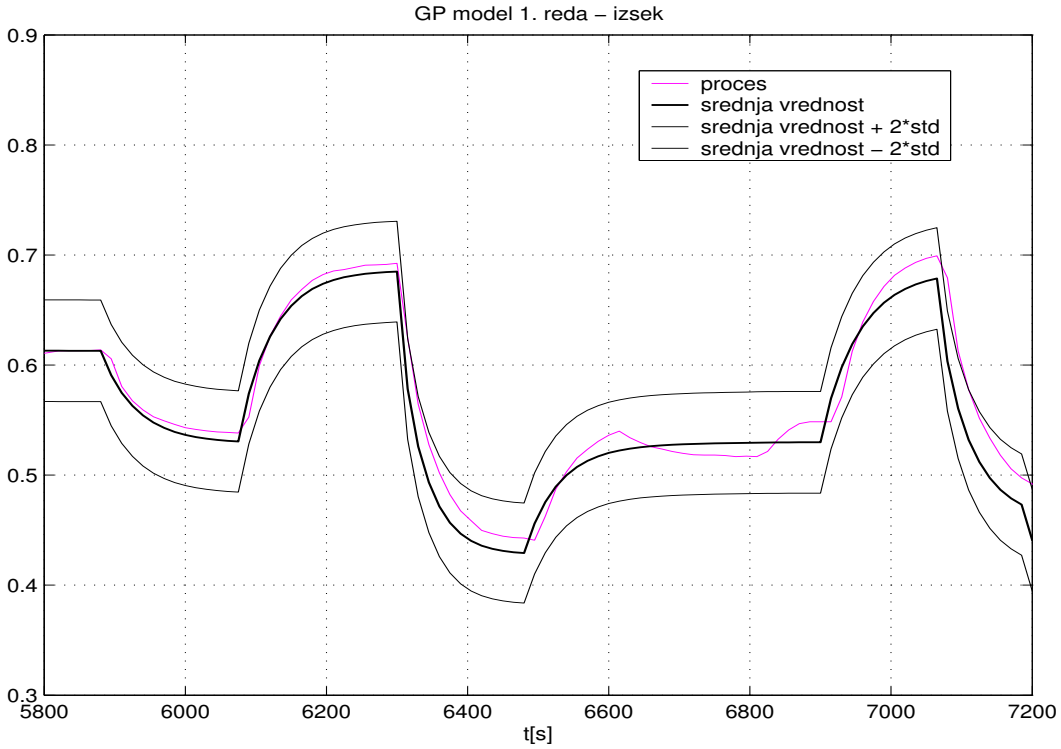
Slika 6.13: Validacijski signali, odziv procesa



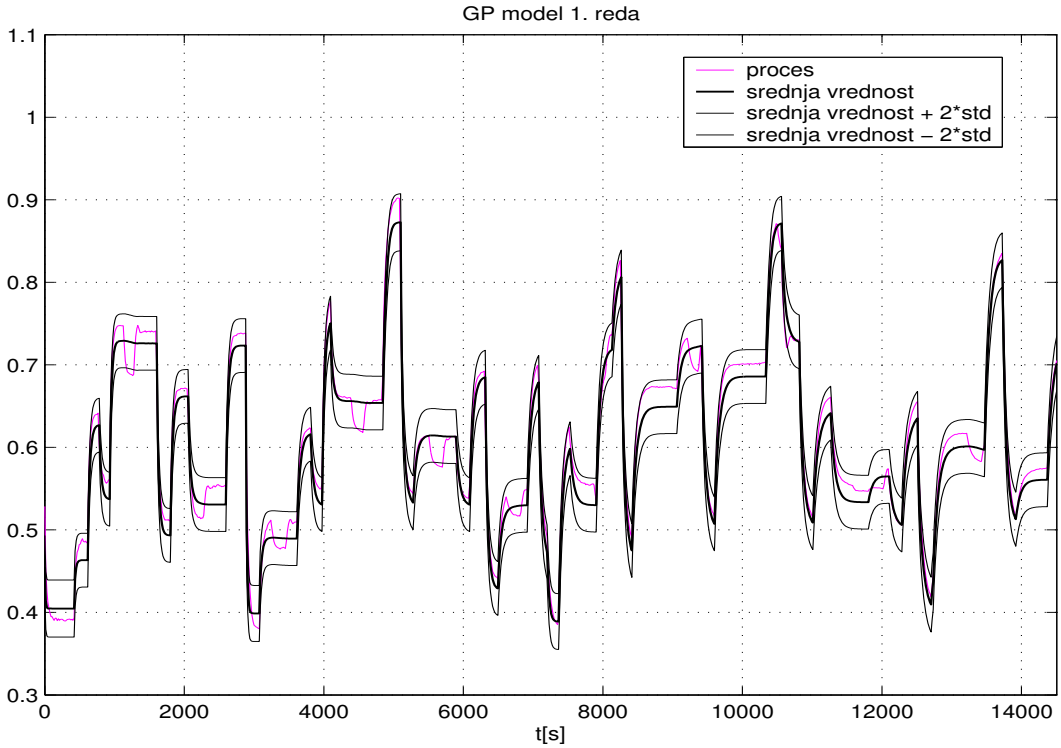
Slika 6.14: Histograma amplitud vhodnega in izhodnega validacijskega signala



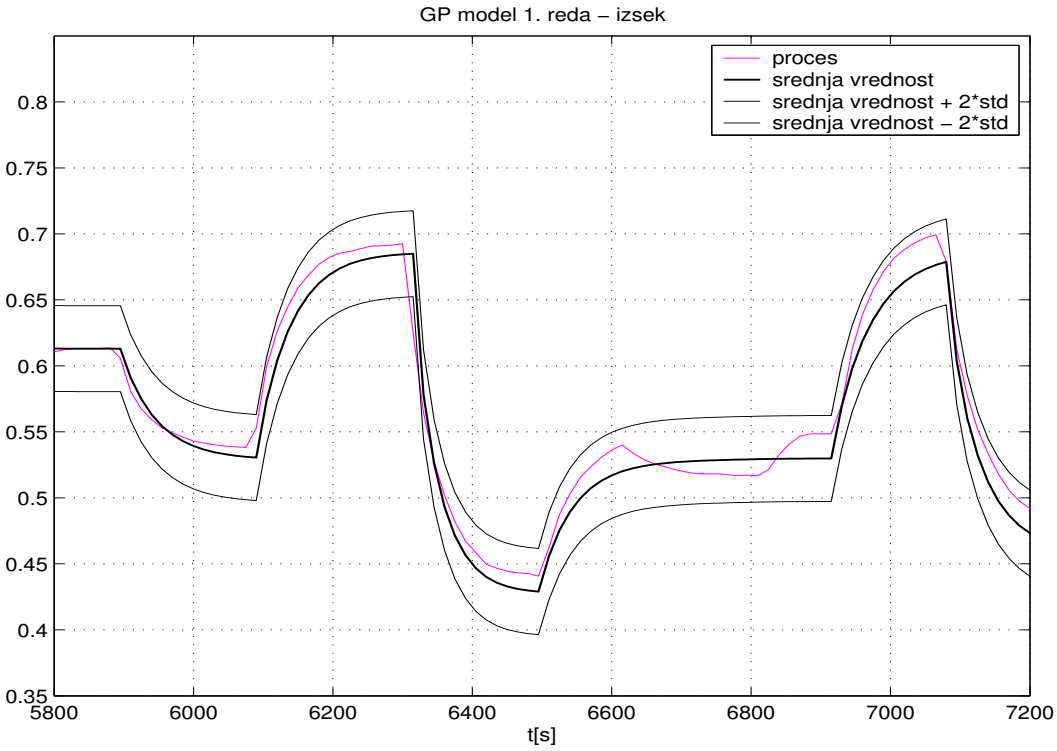
Slika 6.15: Odziv modela ("*eksaktna*" metoda) pri validaciji



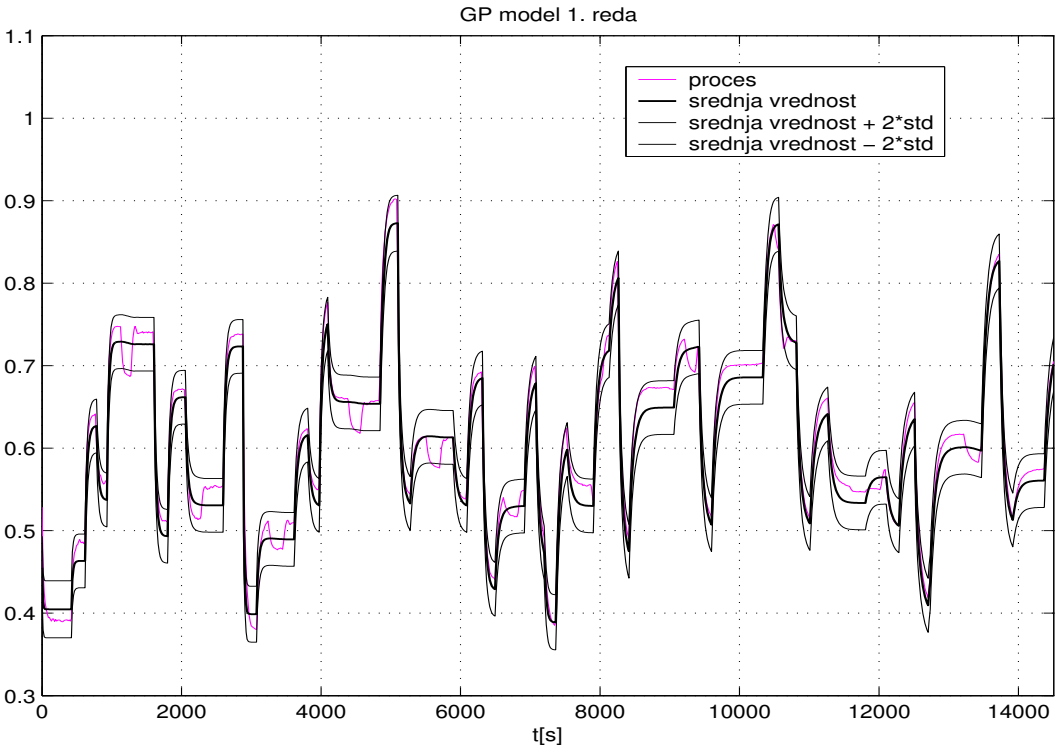
Slika 6.16: Odziv modela ("*eksaktna*" metoda) pri validaciji – izsek



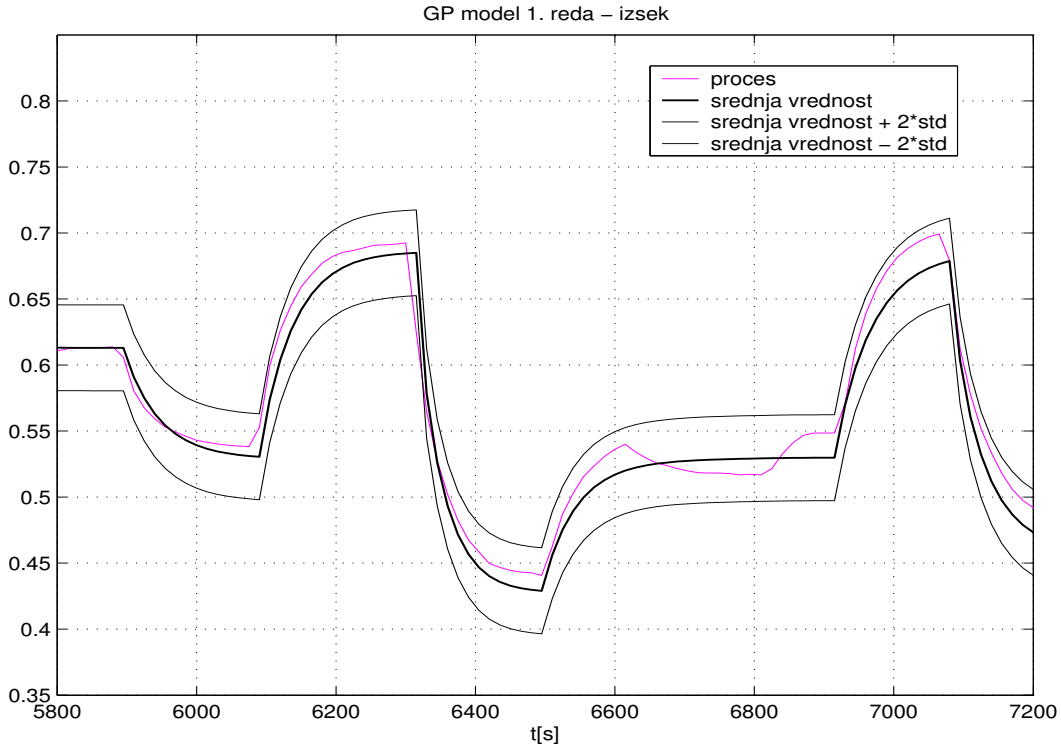
Slika 6.17: Odziv modela ("Taylorjeva aproksimacija") pri validaciji



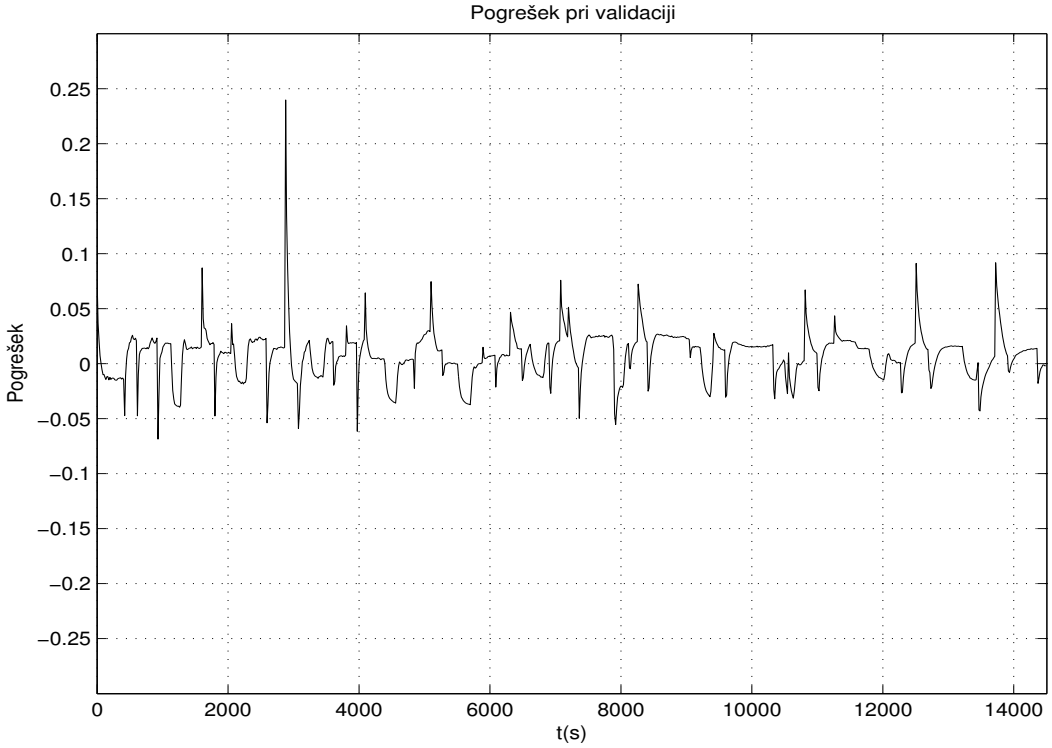
Slika 6.18: Odziv modela (“Taylorjeva aproksimacija”) pri validaciji – izsek



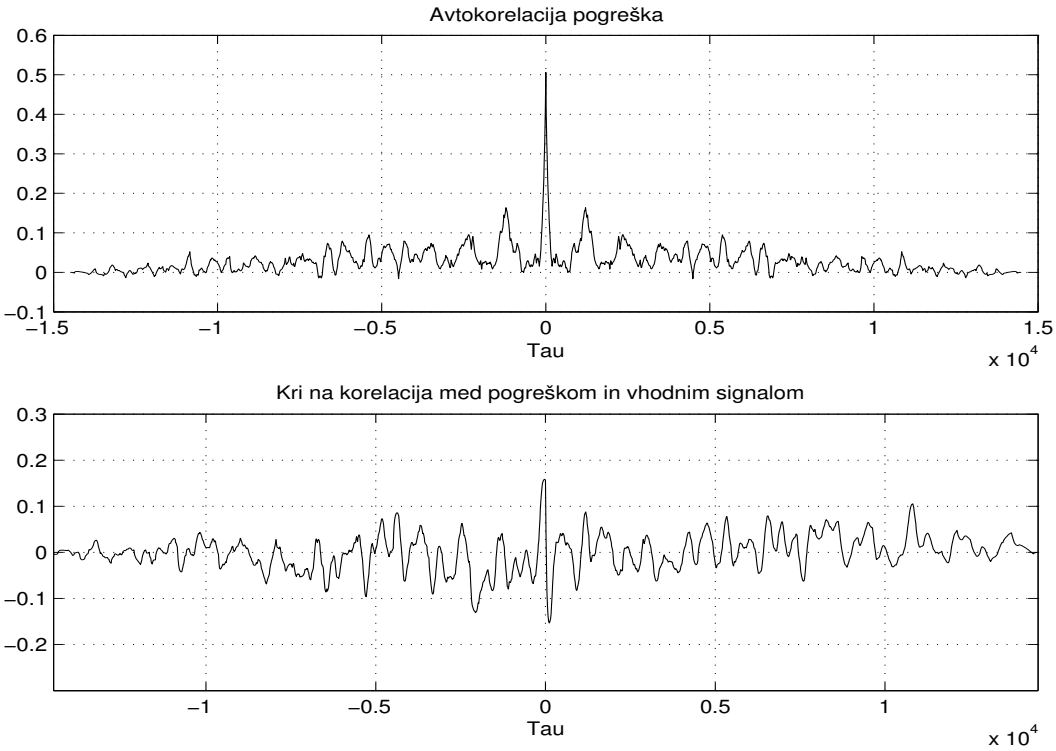
Slika 6.19: Odziv modela (“*naivna*” metoda) pri validaciji



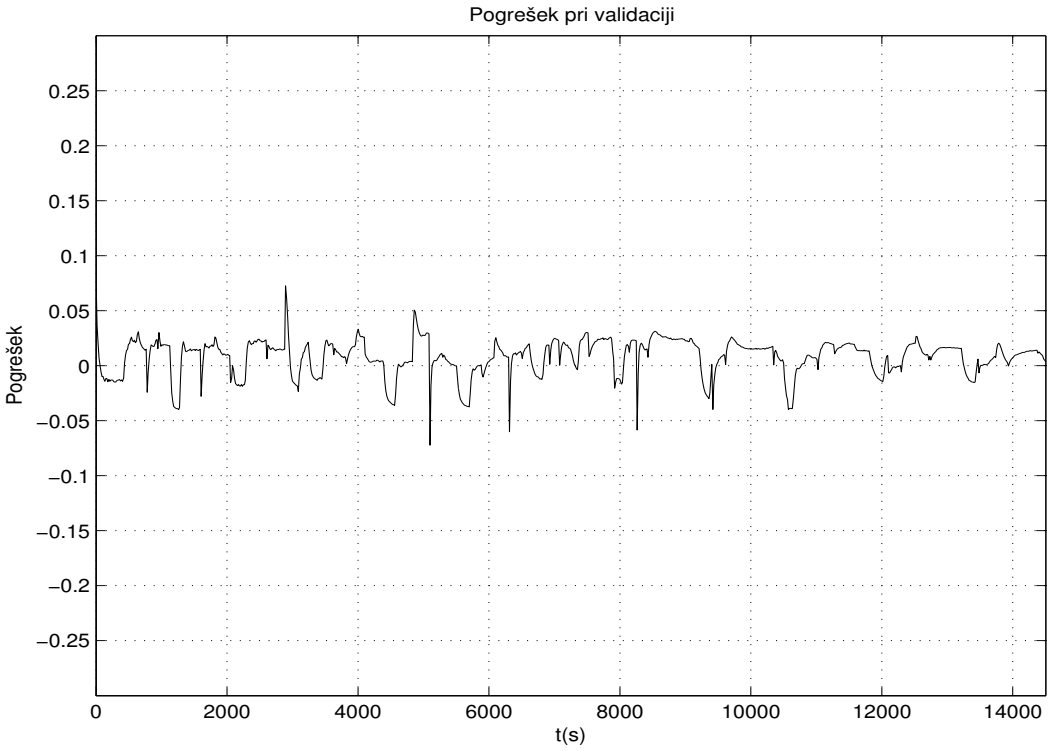
Slika 6.20: Odziv modela (“*naivna*” metoda) pri validaciji – izsek



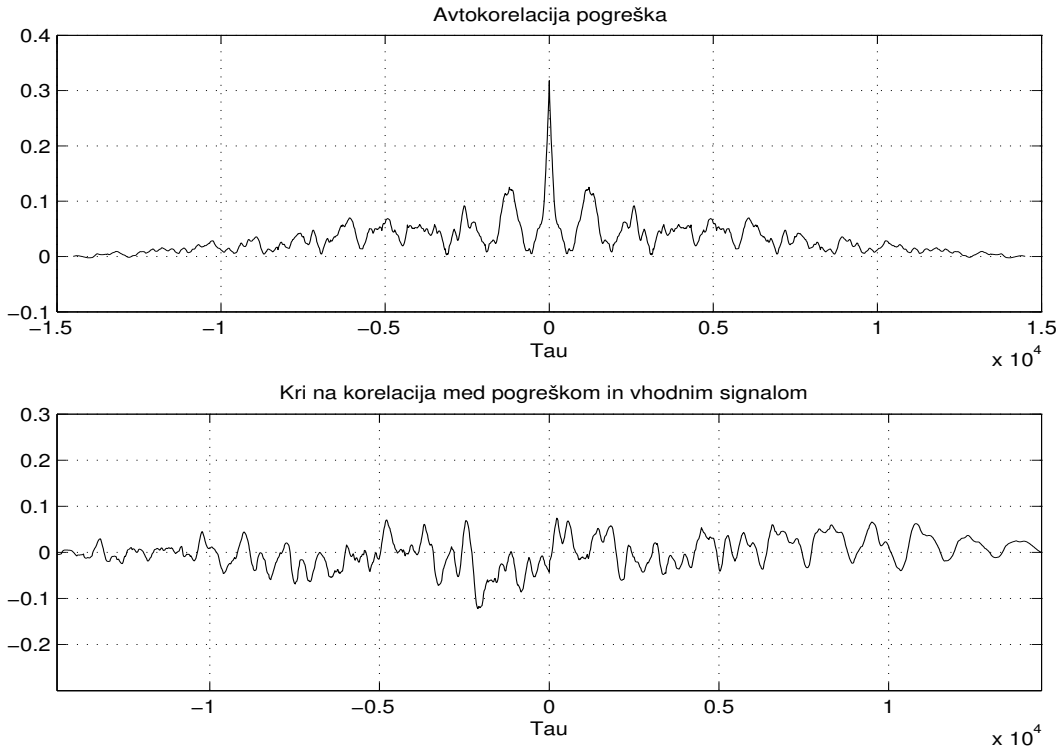
Slika 6.21: Pogrešek validacije (“*eksaktna*” metoda)



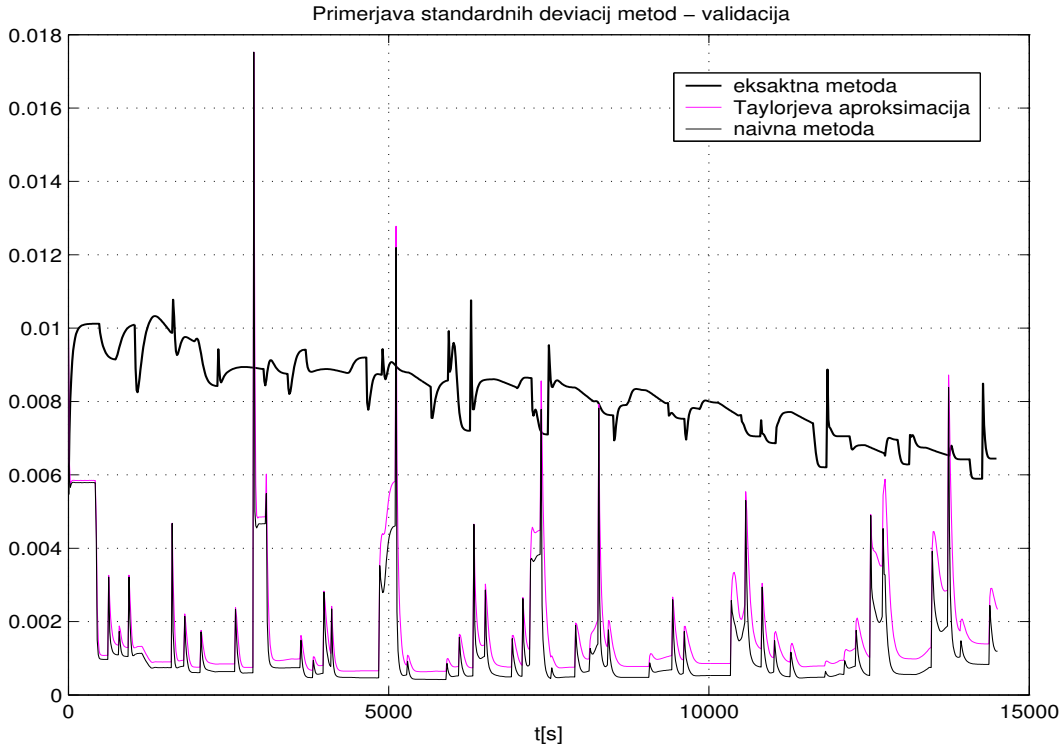
Slika 6.22: Avtokorelacija in križna korelacija pogreška z vhodnim validacijskim signalom ("*eksaktna*" metoda)



Slika 6.23: Pogrešek validacije (*"Taylorjeva aproksimacija"* , "*naivna*" metoda)



Slika 6.24: Avtokorelacija in križna korelacija pogreška z vhodnim validacijskim signalom (*"Taylorjeva aproksimacija"* , "*naivna*" metoda)



Slika 6.25: Primerjava standardnih deviacij metod pri validaciji

Rezultate validacije z GP modelom z različnimi metodami propagiranja smo primerjali tudi z merami za vrednotenje modelov:

- Povprečna absolutna napaka AE (ang. *average absolute error*):

$$AE = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{y}_i - y_i\right|. \tag{6.3}$$

- Povprečna kvadratična napaka SE (ang. *root mean square error*):

$$SE = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i - y_i\right)^2. \tag{6.4}$$

- Logaritem gostote napake LD (ang. *log-predictive density error*):

$$LD = \frac{1}{2N}\sum_{i=1}^{N}\left(\log(2\pi) + \log(var_i) + \frac{\left(\hat{y}_i - y_i\right)^2}{var_i}\right). \tag{6.5}$$

V izračunu mere LD (logaritem gostote napake predikcije) [24] poleg razlike med odzivom procesa in srednjo vrednostjo GP modela nastopa tudi varianca predikcije GP modela, mera LD podaja informacijo o povprečni kvadratični napaki, normirani z vrednostjo varince predikcije.

Tabela 6.2 prikazuje vrednosti številskih karakteristik, dobljenih iz primerjave odzivov procesa in modela po začetnem prehodnem pojavu 500s. Primerjava pokaže ujemanja srednjih vrednosti metod (podobnost AE in SE cenilk) in odstopanja ocenjenih varianc predikcij z *"naivnim"* propagiranjem negotovosti in *"Taylorjevi aproksimaciji"* od *"eksaktne"* metode (LD cenilka).

	AE	**SE**	**LD**
"*eksaktna*" metoda	0.0153	3.2753e-004	-1.3718
"Taylorjeva aproksimacija"	0.0153	3.2744e-004	149.0672
"*naivna*" metoda	0.0153	3.2744e-004	365.7826

Tabela 6.2 – Velikosti mer za vrednotenje modelov

Iz prikazanih grafičnih in številskih rezultatov identifikacije in validacije vidimo, da so razlike med odzivoma (srednjimi vrednostmi) GP modela pri uporabi različnih

metod propagiranja zanemarljive, GP model dobro opisuje nelinearen dinamičen proces. Primerjava varianc predikcije posameznih metod pa pokaže odstopanja med analitičnim izračunom variance (*"eksaktna"* metoda) in izračunom z uporabo Taylorjeve aproksimacije za srednjo vrednost in varianco naključno porazdeljenih vhodov v model. Rezultati metode z uporabo Taylorjeve aproksimacije so bližje rezultatom "*naivne*" metode, kjer pri propagiranju ne upoštevamo podatka o naključno porazdeljenih predhodnih izhodih modela, kot pa analitičnim izračunom z *"eksaktno"* metodo (sliki 6.12, 6.25). Tako kot pri statičnih funkcijah je natančnost uporabe Taylorjeve aproksimacije odvisna od hitrosti spreminjanja modela v delovni točki in razdalje od delovne točke. V splošnem pa vse metode, z relativno velikostjo variance izhoda modela, podajajo informacijo o zaupanju v predikcijo modela, varianca je na področjih, kjer so bili učni (identifikacijski) vzorci gosto razporejeni, majhna, raste pa s približevanjem mejnim področjem učne množice. Na mejnih področjih identifikacijskega signala se poveča tako napaka kot varianca predikcije.

V nadaljevanju so prikazani odzivi procesa in dobljenega GP modela na stopničast vhodni signal odprtosti ventila pri konstantni vrednosti nivoja, kjer je bila vrednost odprtosti ventila spreminjana tudi izven identifikacijskega območja vhodnega signala.

Na slikah 6.26 do 6.32 so prikazani:

- Stopničasto spreminjanje odprtosti ventila pri konstantnem nivoju in odziv procesa (slika 6.26).

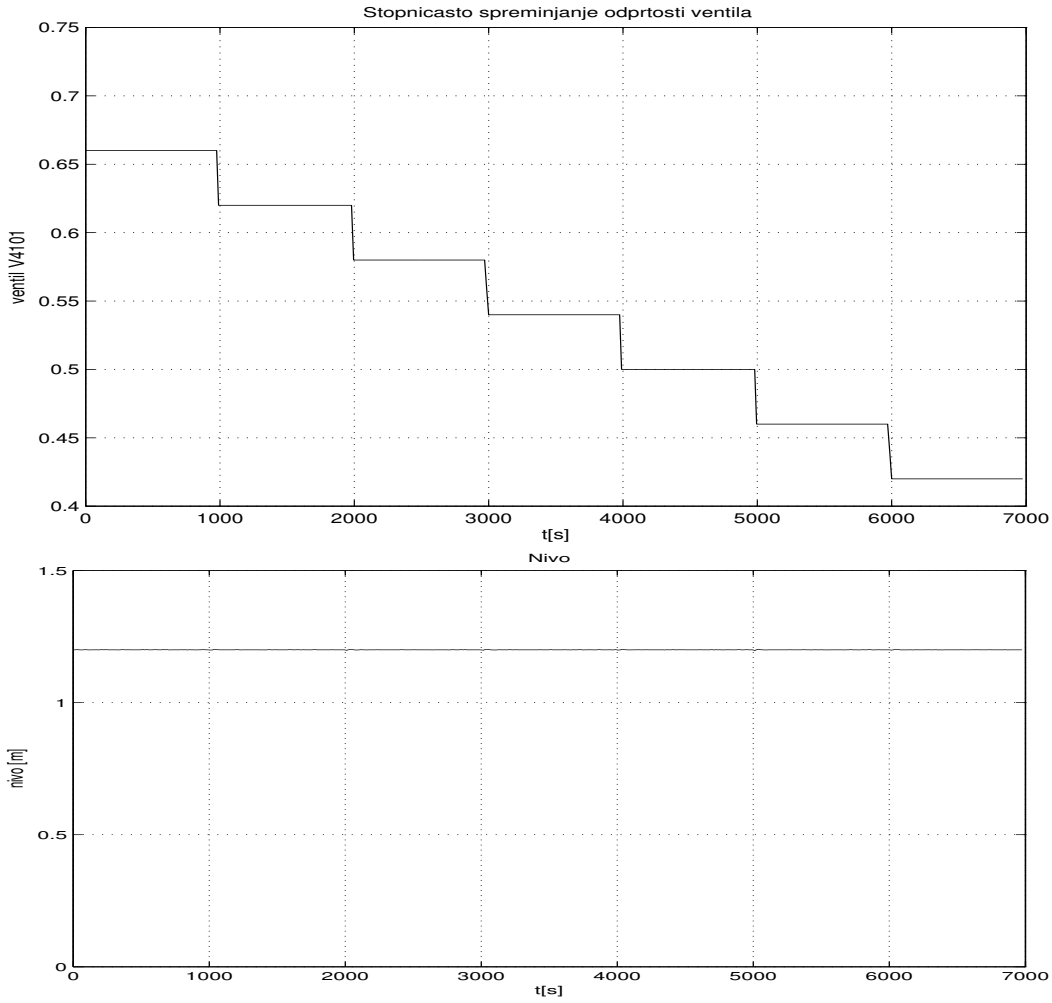
- Primerjava odzivov procesa in GP modela, s pasovoma dvakratnih standardnih deviacij predikcije modela, za metode :
  - "*eksaktna*" metoda (slika 6.27),
  - "*Taylorjeva aproksimacija*" (slika 6.28),
  - "*naivna*" metoda (slika 6.29).

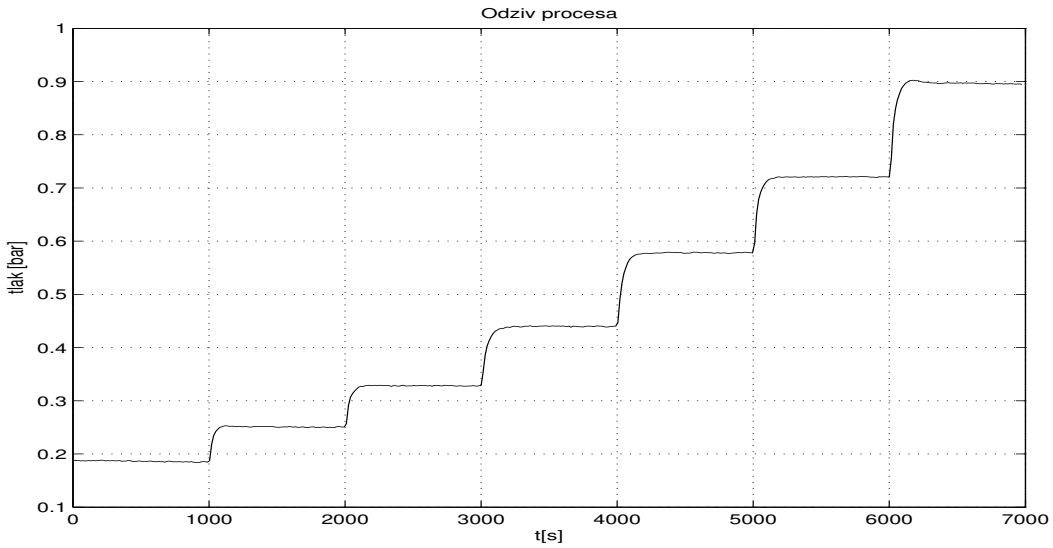
- Primerjava pogreškov modelov (srednjih vrednosti) (slika 6.30).

- Primerjava standardnih deviacij predikcij modelov (slika 6.31).

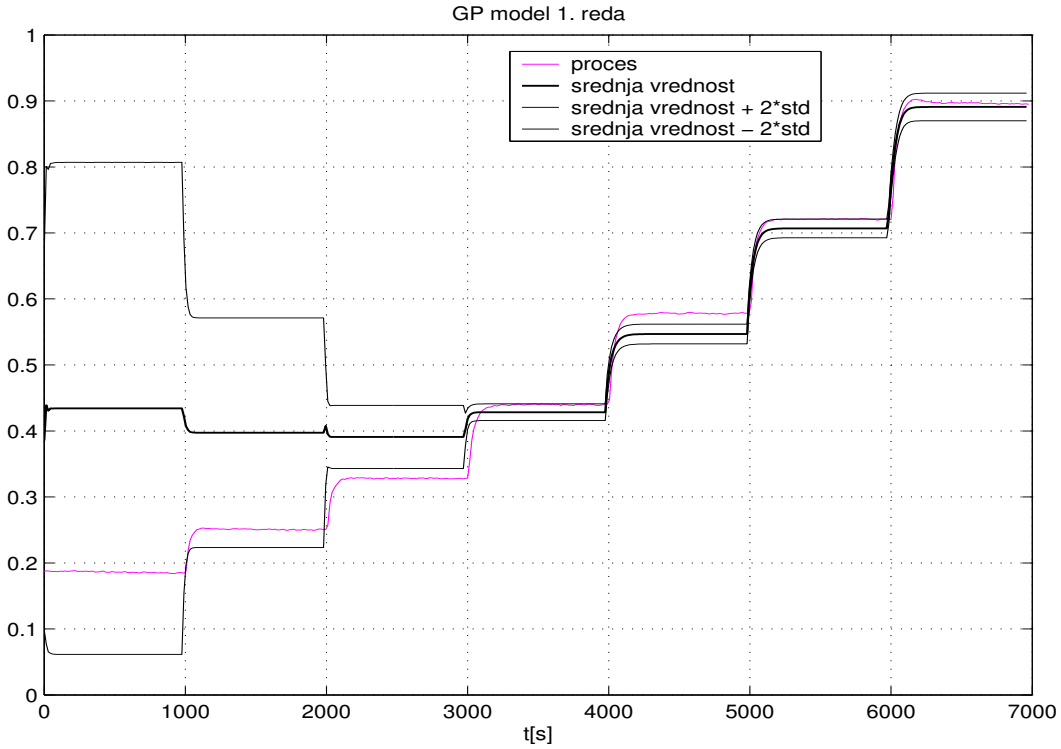
Ne glede na metodo propagiranja vidimo, da GP model zunaj področja identifikacijskih meritev, na katerih so bili naučeni hiperparametri kovariančne

funkcije, neustrezno opisuje proces, na kar pa kaže tudi velikost variance predikcije. Ob prihodu na področje identifikacijskih meritev (učne množice GP modela) se hitro zmanjšata tako napaka kot velikost variance modela.

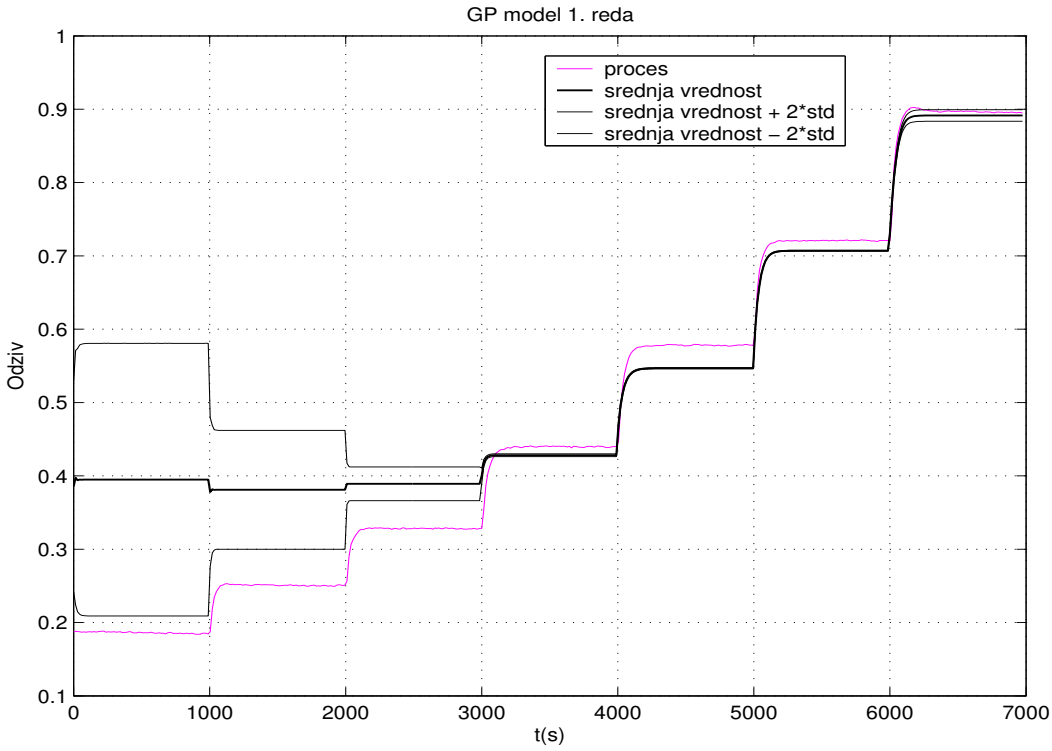




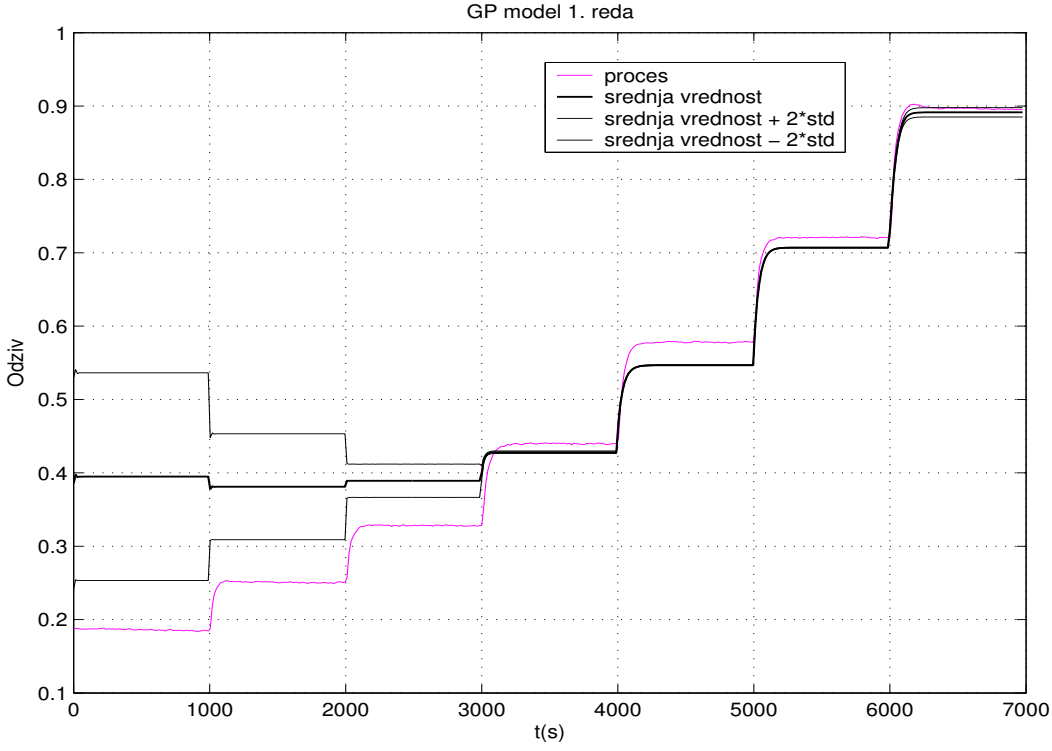
Slika 6.26: Stopničasto spreminjanje odprtosti ventila pri konstantnem nivoju in odziv procesa



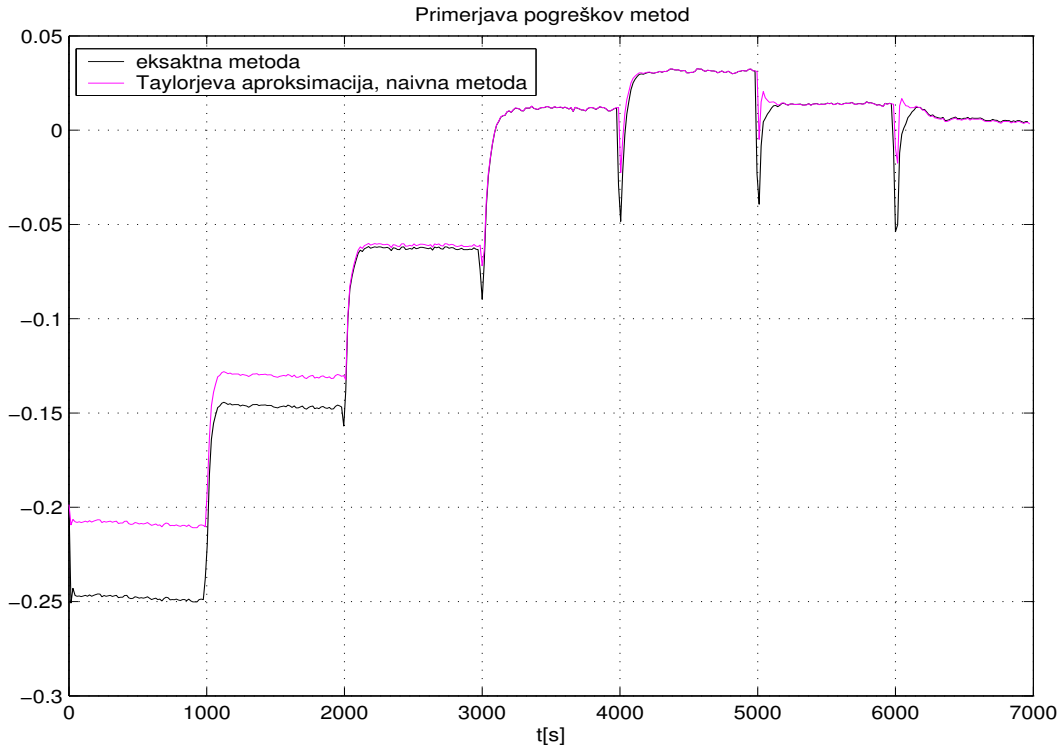
Slika 6.27: Odziv modela (“*eksaktna*” metoda)



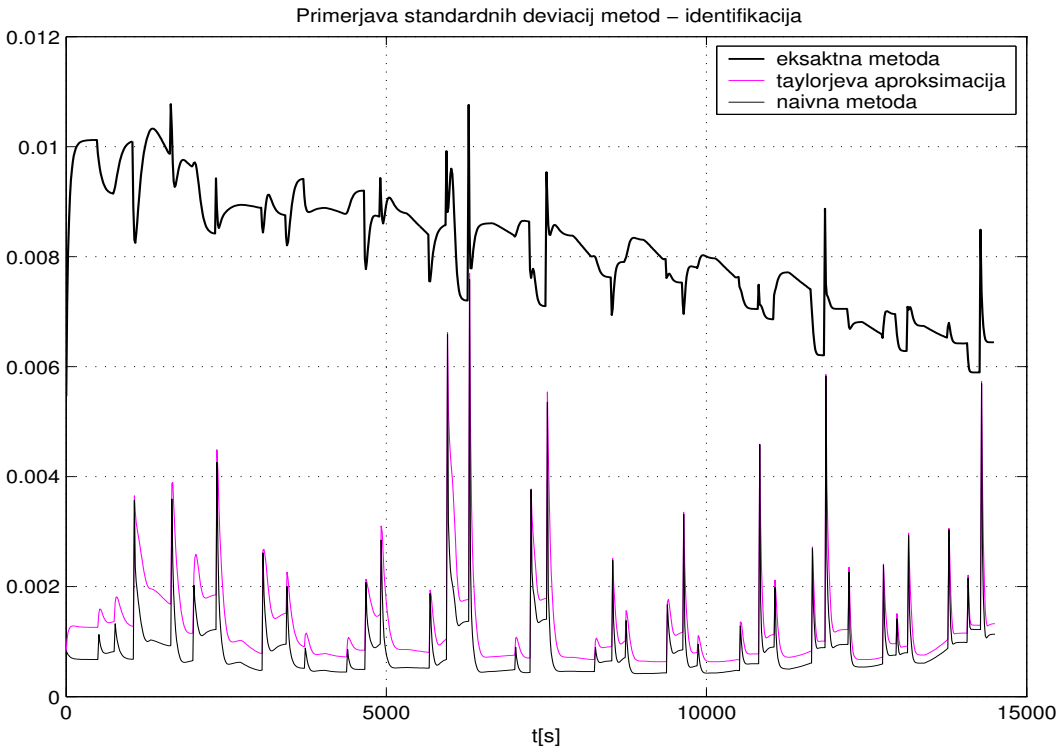
Slika 6.28: Odziv modela ("*Taylorjeva aproksimacija*")



Slika 6.29: Odziv modela ("*naivna metoda*")



Slika 6.30: Primerjava pogreškov metod



Slika 6.31: Primerjava standardnih deviacij metod

V nadaljevanju so, podobno kot v prejšnjem primeru, prikazani odzivi GP modela na linearno naraščajoč signal odprtosti ventila na celotnem področju vhodnega signala, pri konstantni vrednosti nivoja. Ker bi s takimi vrednostmi vhodnega signala velikost tlaka v ločevalniku presegla varnostne omejitve, poskus na realnem procesu ni bil realiziran. Rezultati so bili uporabljeni za informacijo o obliki odzivov, dobljenih z GP modelom, z različnimi metodami propagiranja, na celotnem področju vhodnega signala, in njihovo primerjavo.

Na slikah 6.32 do 6.38 so prikazani:

- Linearno spreminjanje odprtosti ventila pri konstantnem nivoju h=0.6 in odziv procesa (slika 6.32).

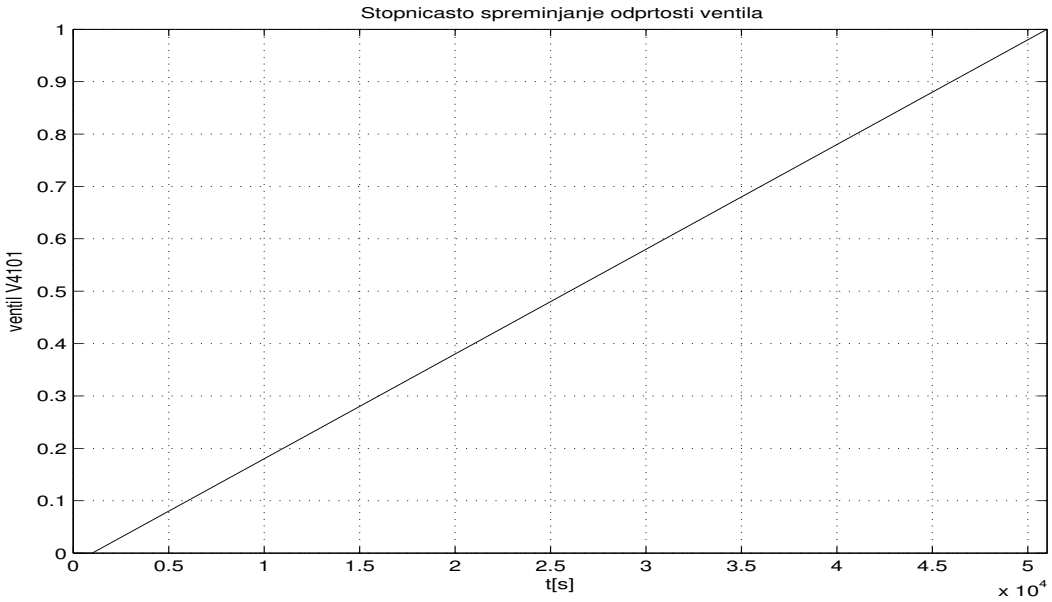
- Odzivi GP modela, s pasovoma dvakratnih standardnih deviacij predikcije, za metode :
  - "*eksaktna*" metoda (slika 6.33),
  - "*Taylorjeva aproksimacija*" (slika 6.34, slika 6.35),
  - "*naivna*" metoda (slika 6.36).

- Primerjava odzivov modelov (srednjih vrednosti) z različnimi načini propagiranja verjetnostnih porazdelitev preteklih izhodov (slika 6.37).

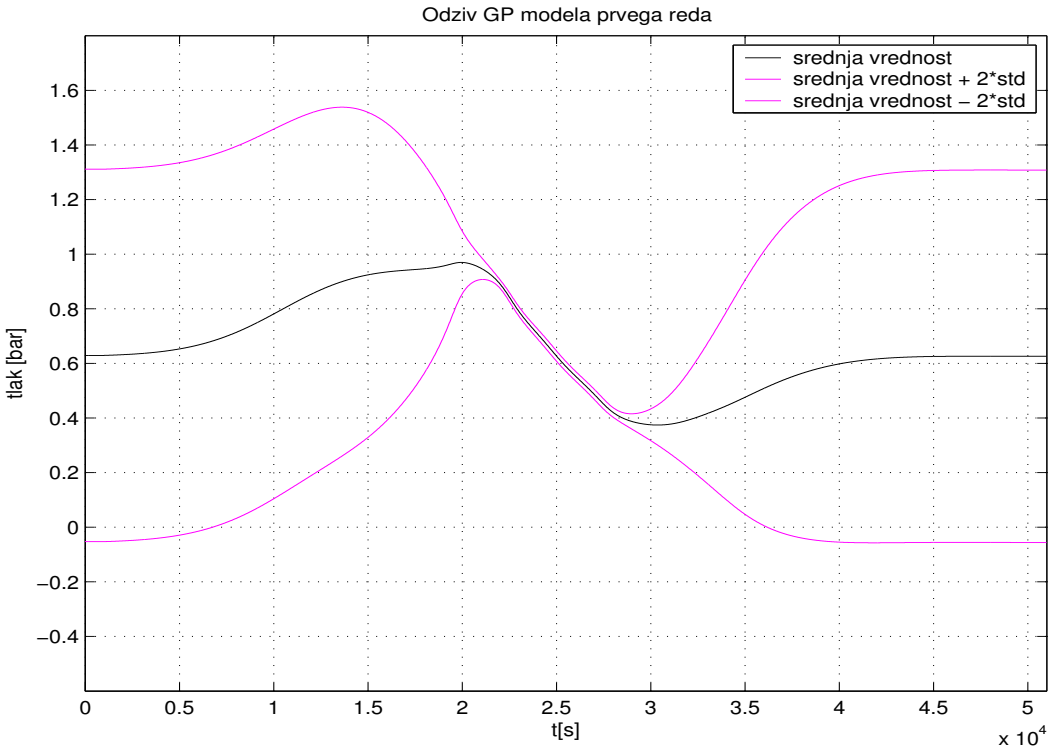
- Primerjava standardnih deviacij predikcij modelov (slika 6.38).

Iz primerjave rezultatov vidimo, da nas GP model z velikostjo variance opozori, da se nahajamo izven področja identifikacijskih meritev. Varianca se hitro zmanjša na področju učne množice GP modela, kjer tudi ni opazne razlike med izhodi (srednjimi vrednostmi) GP modela z različnimi metodami propagiranja. Velikost variance je relativna glede na uporabljeno metodo. Pri opazovanju srednjih vrednosti modelov, z uporabo "*naivne*" metode ali "*Taylorjeve aproksimacije*" opazimo oscilacije izhoda modela (srednje vrednosti) v določenem področju vhodne veličine u_1, ki se nahaja izven idenfifikacijskega območja. Pri metodi "*Taylorjeva aproksimacija*" opazimo tudi nestabilno propagiranje variance izven identifikacijskega področja GP modela. Spet lahko potrdimo ugotovitve, da GP model izven identifikacijskega območja neustrezno opisuje proces, na kar pa nas opozori z veliko vrednostjo variance. Z opazovanjem potekov standardnih deviacij (slika 6.38) lahko hitro ugotovimo, kdaj

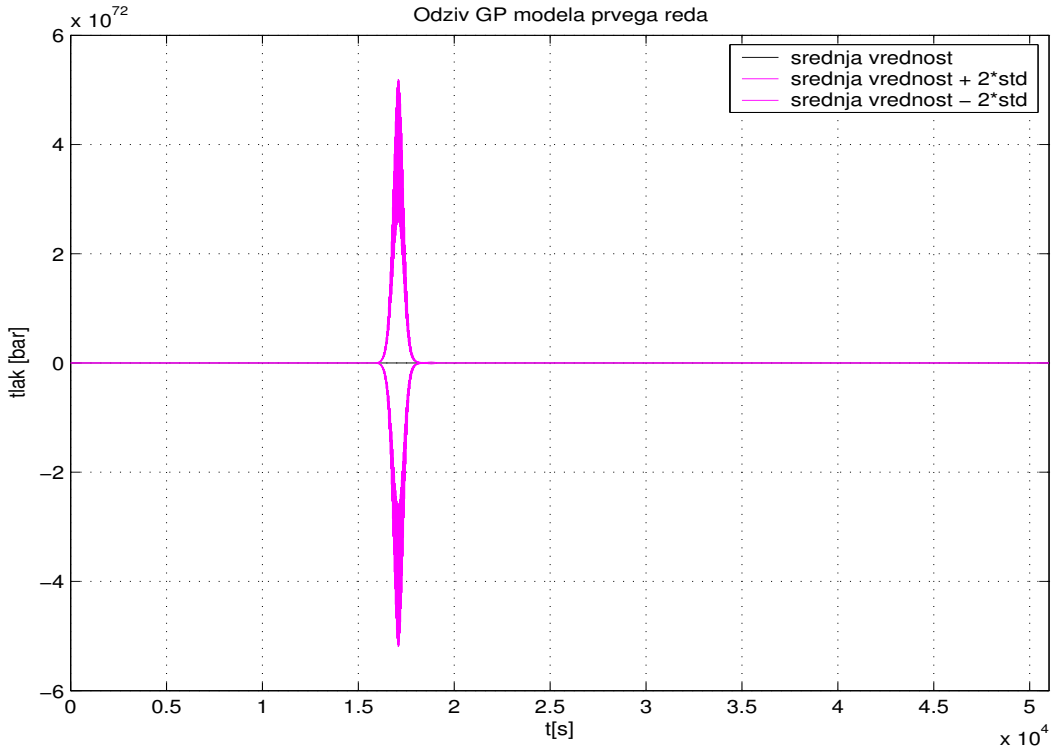
se proces nahaja v področju učne množice GP modela in kdaj to področje zapusti, saj se na mejnem področju vrednost variance zelo hitro spreminja.



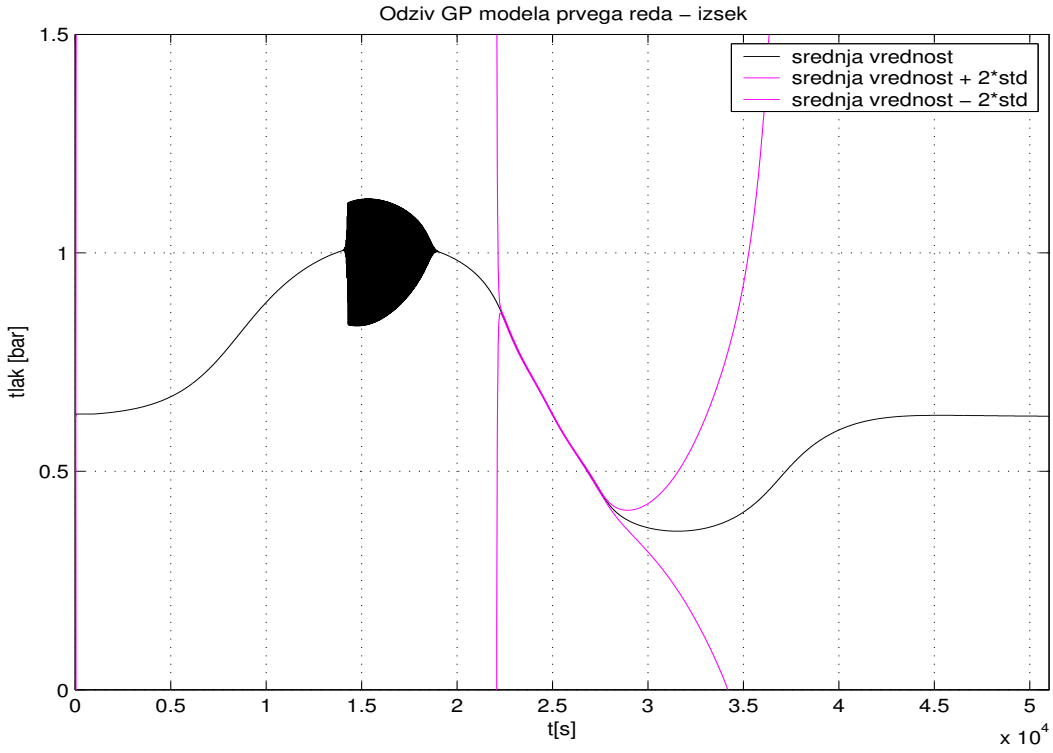
Slika 6.32: Vhodni signal pri konstantnem nivoju h=0.6



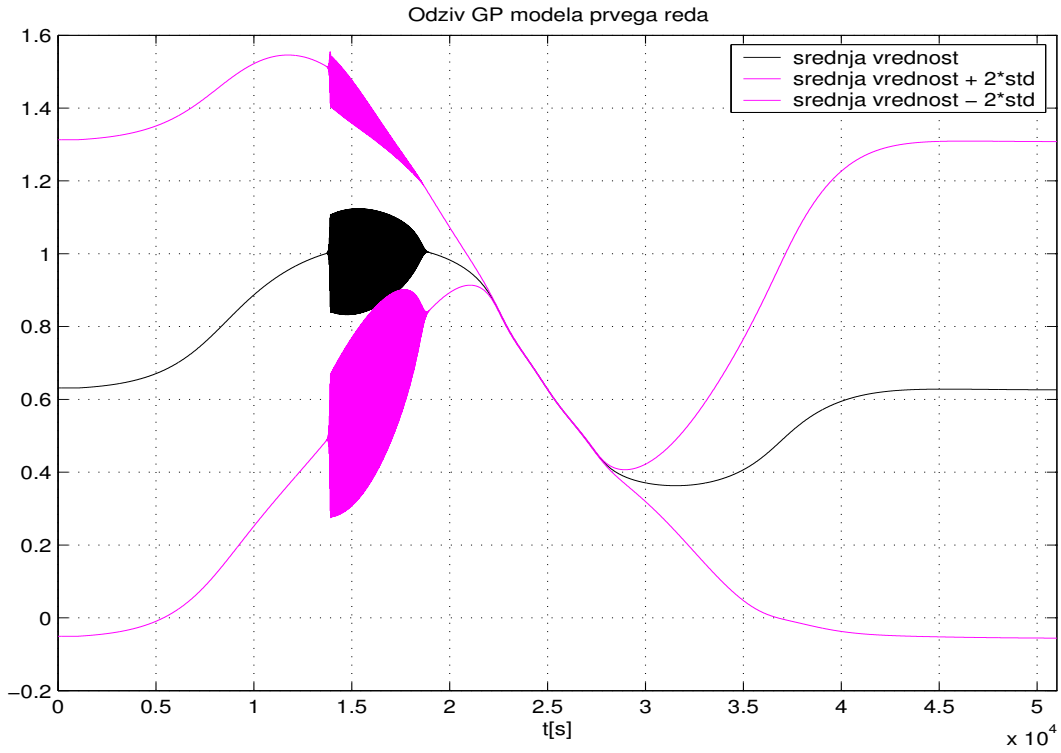
Slika 6.33: Odziv modela (“*eksaktna*” metoda)



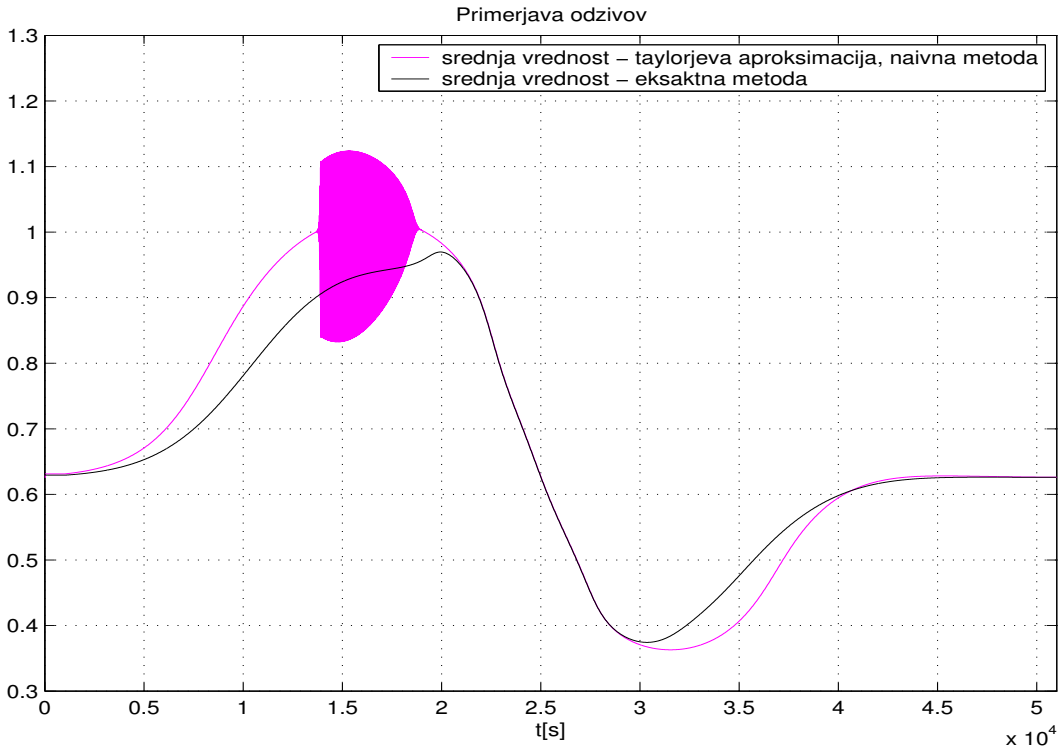
Slika 6.34: Odziv modela ("*Taylorjeva aproksimacija*")



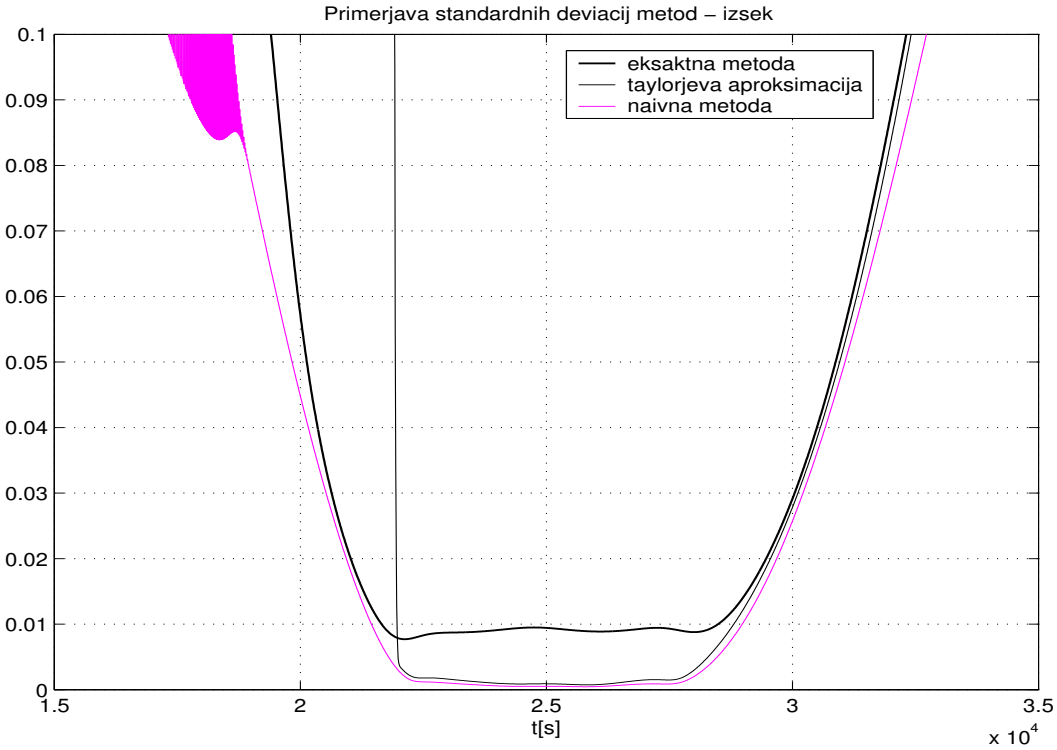
Slika 6.35: Odziv modela ("*Taylorjeva aproksimacija*") –izsek



Slika 6.36: Odziv modela (“naivna metoda”)



Slika 6.37: Primerjava odzivov (srednjih vrednosti) modelov



Slika 6.38: Primerjava standardnih deviacij metod – izsek

7. Prediktivno vodenje procesa priprave plina na osnovi Gaussovih procesov

V tem poglavju bomo predstavili rezultate vodenja tlaka ločevalnika z nelinearnim prediktivnim regulatorjem na osnovi identificiranega GP modela. Tlak p v ločevalniku je bil voden z zveznim ventilom V4101 (regulirna veličina u_1), vrednost nivoja h pa je bila v GP modelu upoštevana kot merljiv dejavnik nelinearnosti.

Nelinearni prediktivni regulator je bil razvit na ideji prediktivnega funkcionalnega vodenja (PFC) [22] [23] [26] z glavnimi značilnostmi:

- Uporaba GP modela za predikcijo izhodnega signala procesa z upoštevanjem tako napovedane srednje vrednosti kot variance izhoda modela.

- Tvorjenje referenčne trajektorije za gladko opisovanje prehoda od trenutne vrednosti procesa $p(k)$ do želene referenčne vrednosti $w_p(k)$, z inicializacijo referenčne trajektorije na trenutno merjeno vrednost izhodnega signala, $r(k) = p(k)$, in želenim eksponentnim upadanjem pogreška v prihodnosti po trajektoriji, določeni s sistemom prvega reda enotinega ojačanja in časovno konstanto T_{ref}.

- Vključitev samo končne točke predikcijskega horizonta H, točke ujemanja P, v kriterijsko funkcijo (7.1), ob predpostavki o konstantnem regulirnem signalu u_1 na celotnem predikcijskem horizontu (uporaba konstantne bazne funkcije pri strukturiranju regulirnega signala).

$$J = \left(r_p(k+P) - \hat{p}(k+P)\right)^2 \tag{7.1}$$

- Uporaba iterativnega optimizacijskega algoritma za minimizacijo kriterijske funkcije (7.1), z omejitvijo velikosti variance v končni točki predikcijskega horizonta:

$$var\left(\hat{p}(k+P)\right) \leq \mathrm{var}_{\max} \tag{7.2}$$

Rezultati simulacije tlaka z identificiranim GP modelom kažejo na neustrezno opisovanje procesa izven identifikacijskega območja, na kar pa nas model opozori z

velikostjo variance predikcije, ki izven tega območja skokovito naraste. Uspešno vodenje na osnovi modela je mogoče samo v območju, kjer model ustrezno opisuje proces. Ob predpostavki o določanju želenih referenčnih vrednosti za tlak ločevalnika v identificiranem območju, je za namen prediktivnega vodenja potrebno tudi optimizacijo omejiti na iskanje najustreznejših vrednosti regulirnih signalov v tem območju, ki je manjše od fizičnega območja regulirnega signala ($0 \le u_1 \le 1$). Pri uporabi nelinearnih GP modelov omejitve regulirne veličine lahko podajamo:

- Eksplicitno z določitvijo spodnje in zgornje meje regulirnega signala, znotraj katerih optimizacijski algoritem lahko išče minimum kriterijske funkcije. Tak način podajanja omejitev je splošen pri uporabi nelinearnih modelov (umetne nevronske mreže, mehki modeli itd).

- Implicitno z določitvijo največje vrednosti variance predikcije modela v končni točki predikcijskega horizonta $var\left(\hat{p}(k+P)\right)$, s čimer optimizacijski algoritem prisilimo na iskanje optimalnih vrednosti regulirnega signala znotraj identificiranega področja GP modela. Tak način podajanja omejitev je mogoč le pri GP modelih, kjer je izhod modela srednja vrednost in varianca predikcije. Določitev velikosti maksimalne vrednosti variance je odvisna od uporabljene metode propagiranja verjetnostnih porazdelitev predhodnih vrednosti izhodov modela.

V nadaljevanju so prikazani rezultati nelinearnega prediktivnega vodenja tlaka ločevalnika z GP modelom. Za zagotovitev vodenja v realnem času, je bila zaradi računske enostavnosti v primerjavi z "*eksaktno* " metodo uporabljena "*Taylorjeva aproksimacija*" propagiranja verjetnostnih porazdelitev preteklih izhodov modela. Uporabljen je bil GP model opisan v poglavju 6 (čas vzorčenja $T_s = 15s$, velikost kovariančne matrike modela $\mathbf{K}_n$, $n = 967$). Dolžina horizonta ujemanja je znašala 8 vzorcev $\left(H = 8T_s = 120s, P = 8\right)$ in je bila določena kot kompromis med predikcijo odziva procesa na zadovoljivem intervalu v prihodnosti in zagotovitvijo delovanja optimizacijskega algoritma v realnem času. Časovna konstanta dinamike referenčne trajektorije je znašala $T_{ref} = 400s$, razen na slikah 7.9 in 7.10, kjer je bila zaradi večje frekvence spreminjanja referenčnega signala zmanjšana na vrednost $T_{ref} = 150s$. Na slikah so prikazani:

- Odziv procesa na stopničasto spreminjanje reference v identificiranem območju GP modela, pri konstantnem nivoju, ter odziv GP modela, potek nivoja,

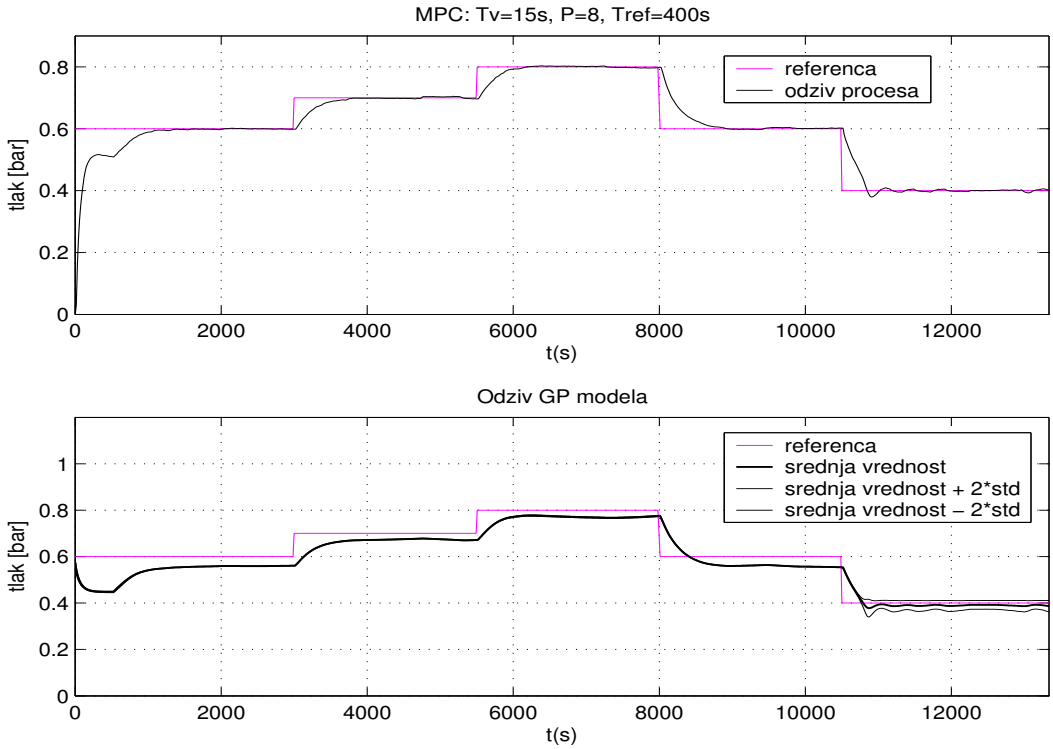
variance GP modela in regulirne veličina u_1, vrednost omejitve variance predikcije modela znaša $\text{var}_{\max} = 10^{-2}$ (slike 7.1, 7.2, 7.3, 7.4).

- Odziv procesa pri konstantni vrednosti reference, v identificiranem območju GP modela, ob spreminjajočem nivoju, ter odziv GP modela, potek nivoja, variance GP modela in regulirne veličina u_1, vrednost omejitve variance predikcije modela smo izbrali $\text{var}_{\max} = 10^{-2}$ (sliki 7.5, 7.6).

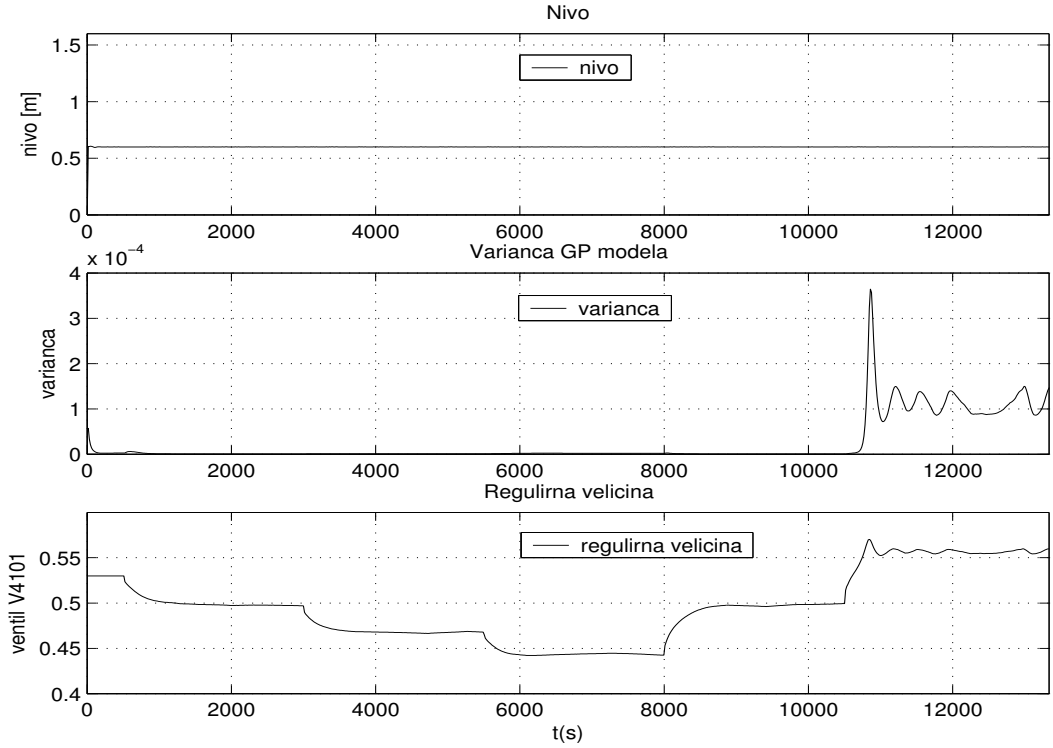
- Odziv procesa na stopničasto spreminjanje reference v identificiranem območju GP modela, pri linearno naraščajoči vrednosti nivoja, ter odziv GP modela, potek nivoja, variance GP modela in regulirne veličina u_1, vrednost omejitve variance predikcije modela smo izbrali $\text{var}_{\max} = 10^{-2}$ (sliki 7.7, 7.8).

- Odziv procesa na stopničasto spreminjanje reference na mejnem področju identifikacijskih podatkov GP modela, pri konstantni vrednosti nivoja, ter odziv GP modela, potek nivoja, variance GP modela in regulirne veličina u_1, vrednost omejitve variance predikcije modela smo izbrali $\text{var}_{\max} = 10^{-2}$ (slike 7.9, 7.10,7.13,7.14).

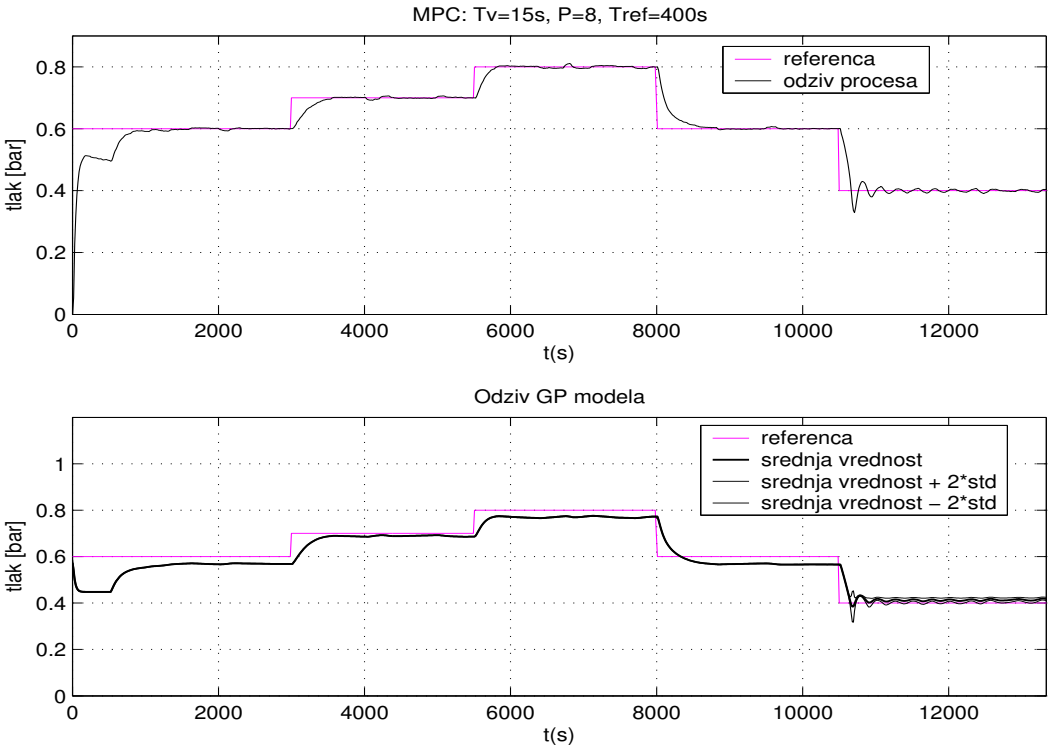
- Odziv procesa na stopničasto spreminjanje reference na mejnem področju identifikacijskih podatkov GP modela, pri konstantni vrednosti nivoja, ter odziv GP modela, potek nivoja, variance GP modela in regulirne veličina u_1, vrednost omejitve variance predikcije modela smo izbrali $\text{var}_{\max} = 1.2 \cdot 10^{-4}$ (slike 7.11, 7.12,7.15,7.16).



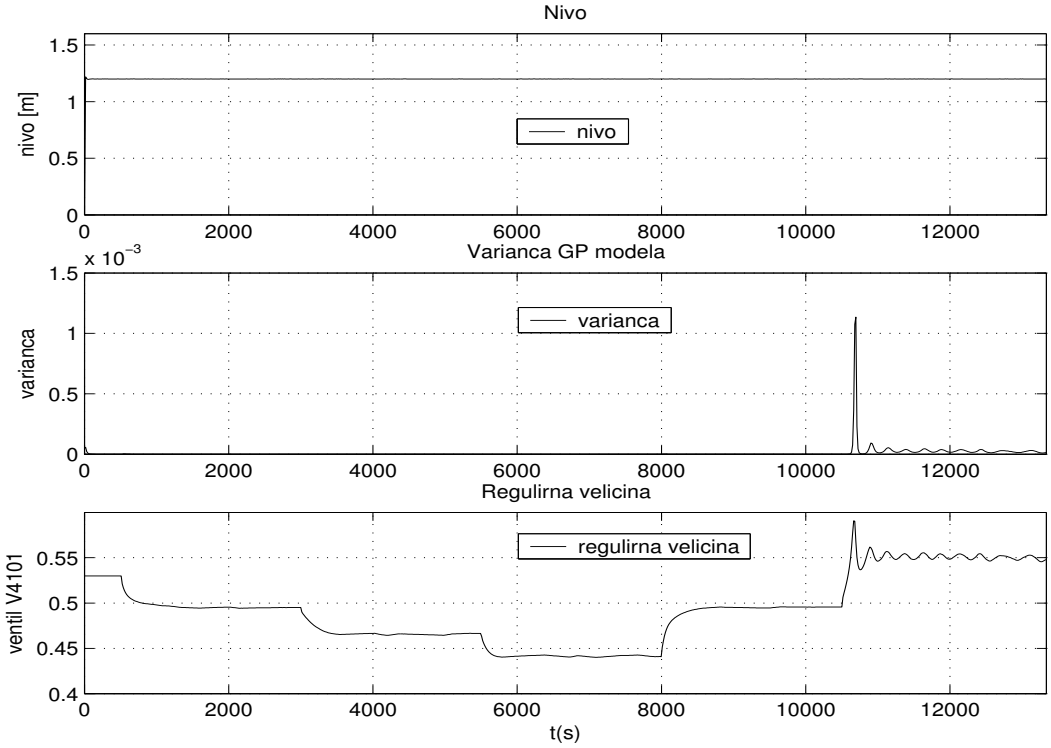
Slika 7.1: Odziv procesa in GP modela



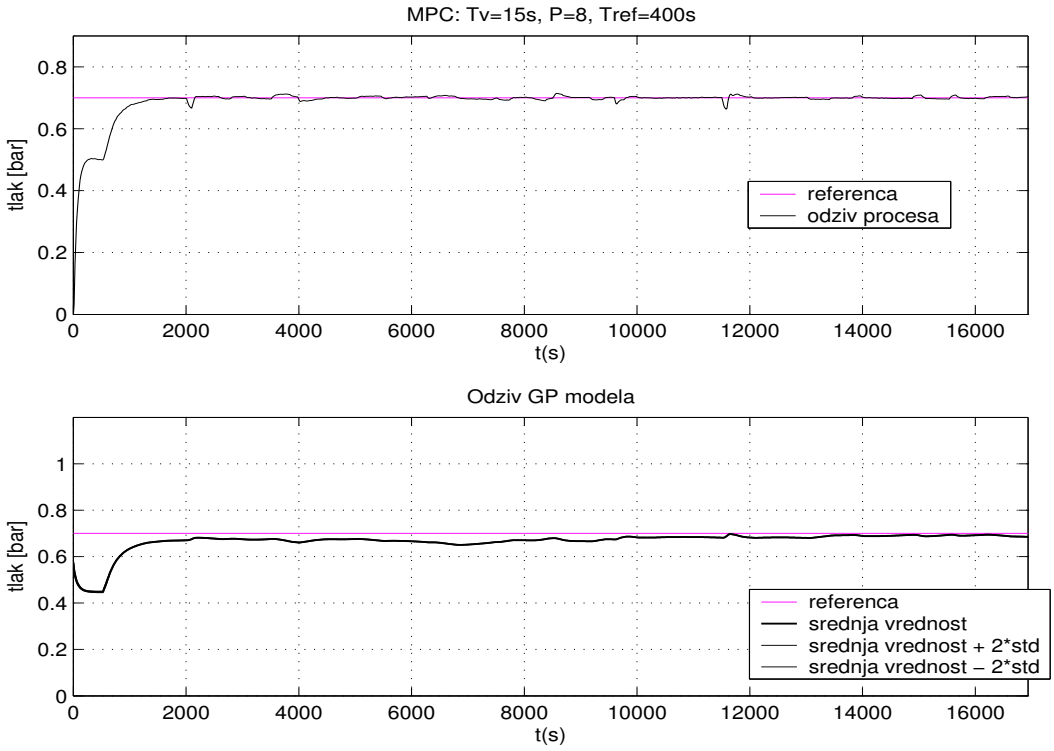
Slika 7.2: Nivo, varianca GP modela in regulirna veličina



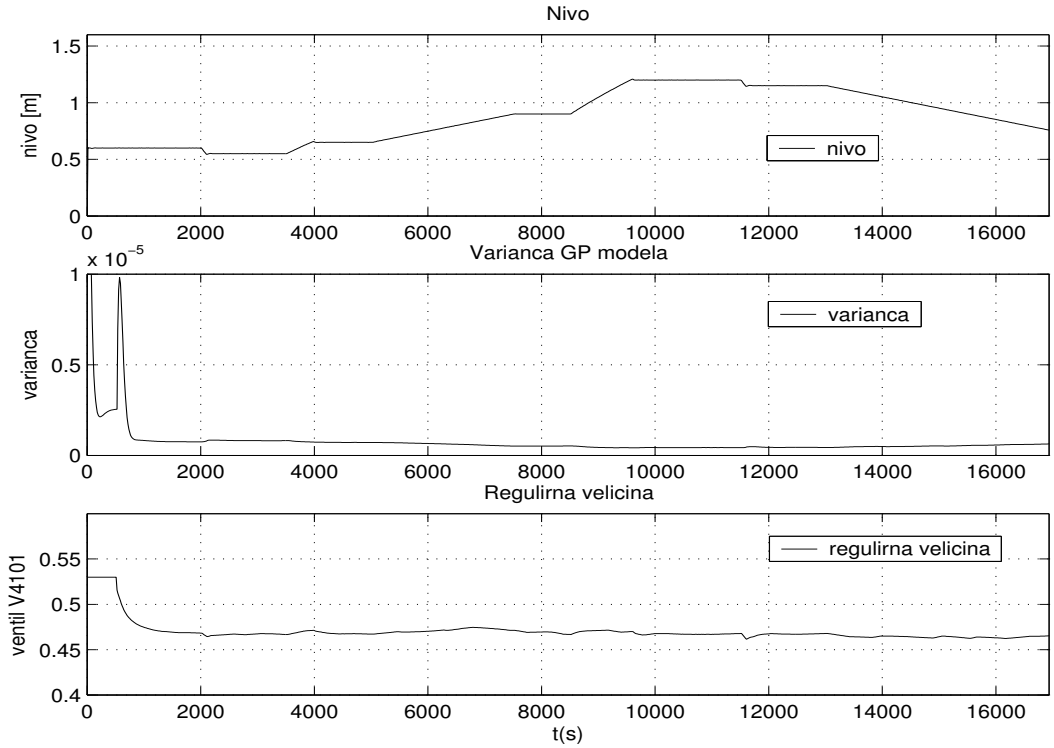
Slika 7.3: Odziv procesa in GP modela



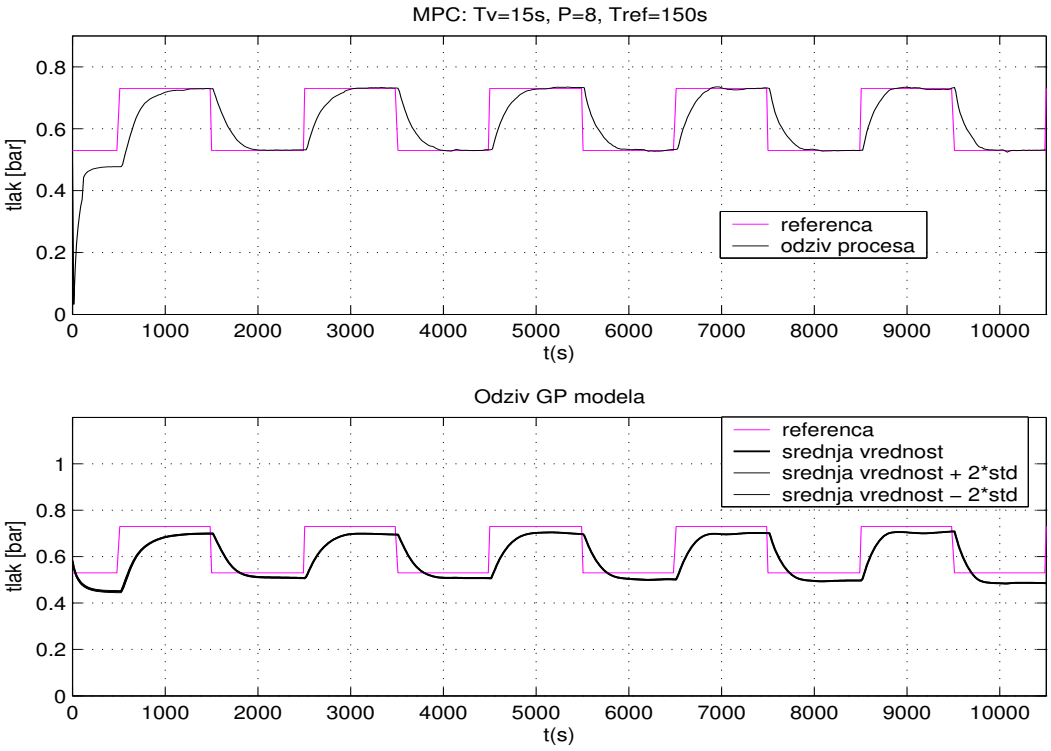
Slika 7.4: Nivo, varianca GP modela in regulirna veličina



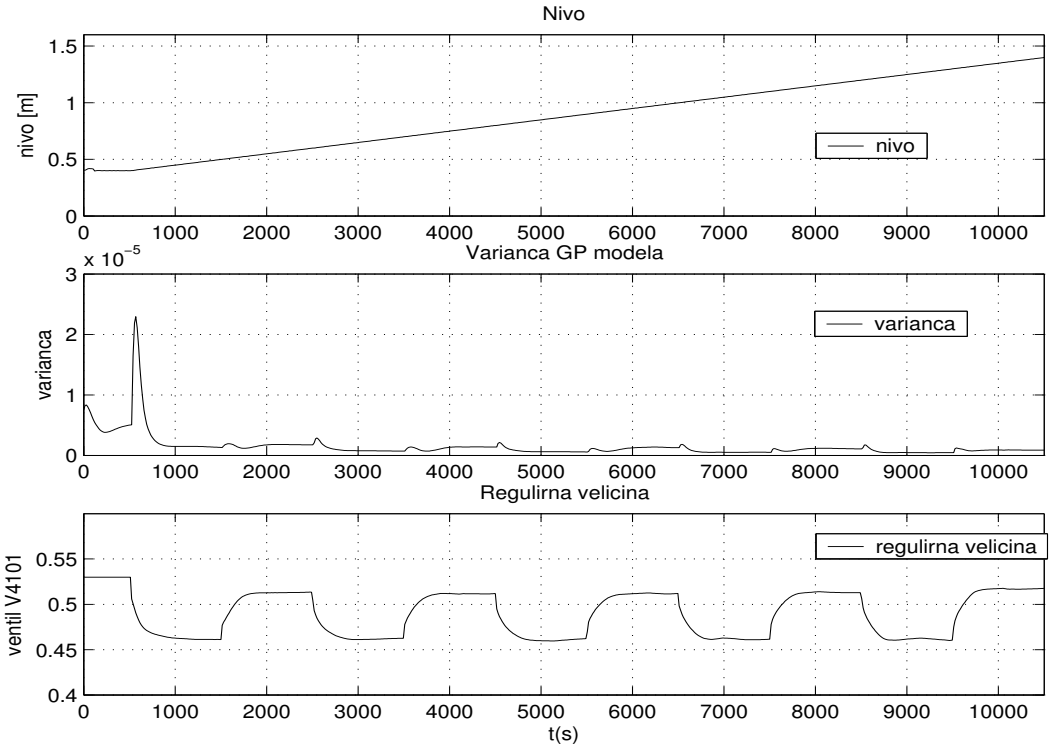
Slika 7.5: Odziv procesa in GP modela



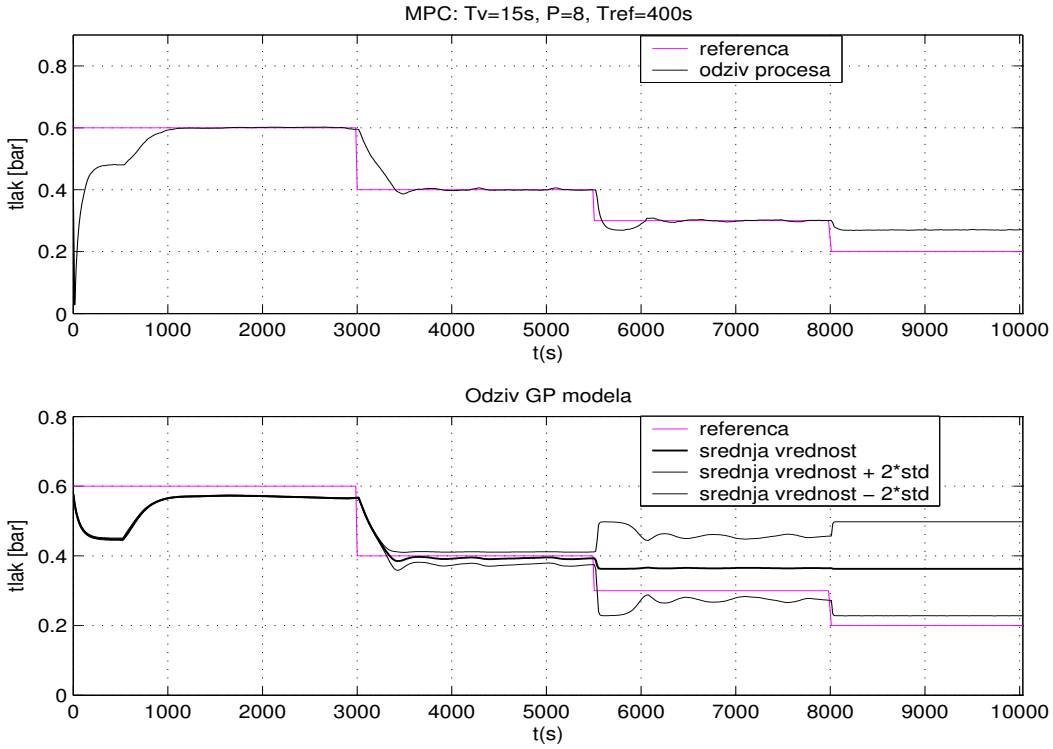
Slika 7.6: Nivo, varianca GP modela in regulirna veličina



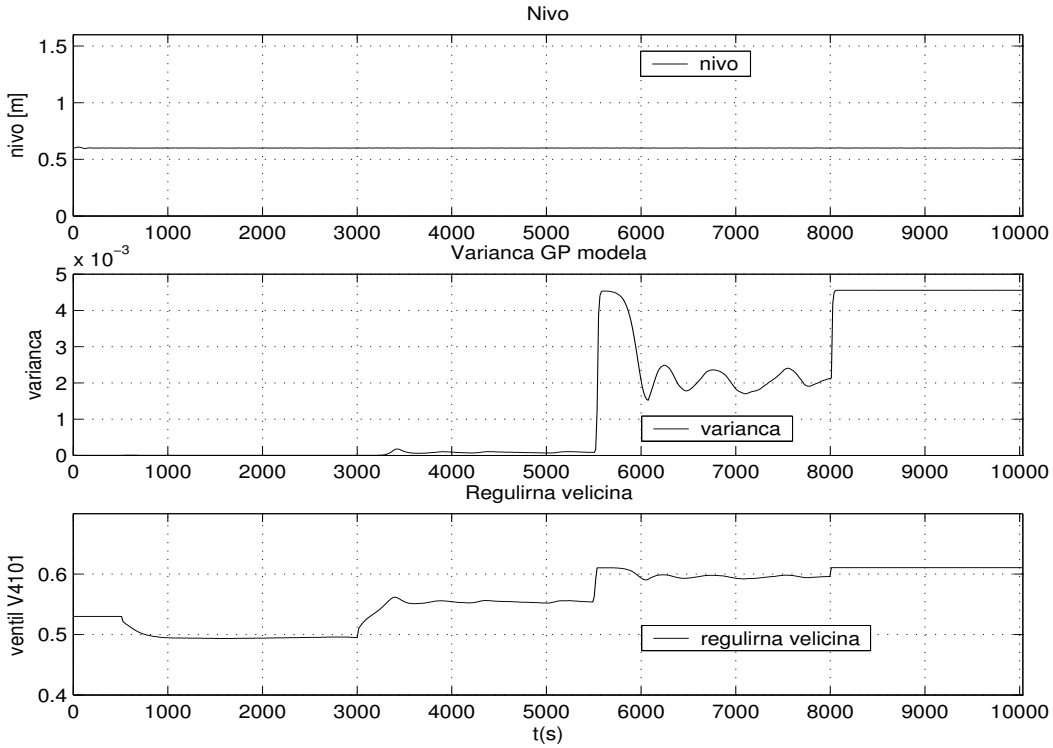
Slika 7.7: Odziv procesa in GP modela



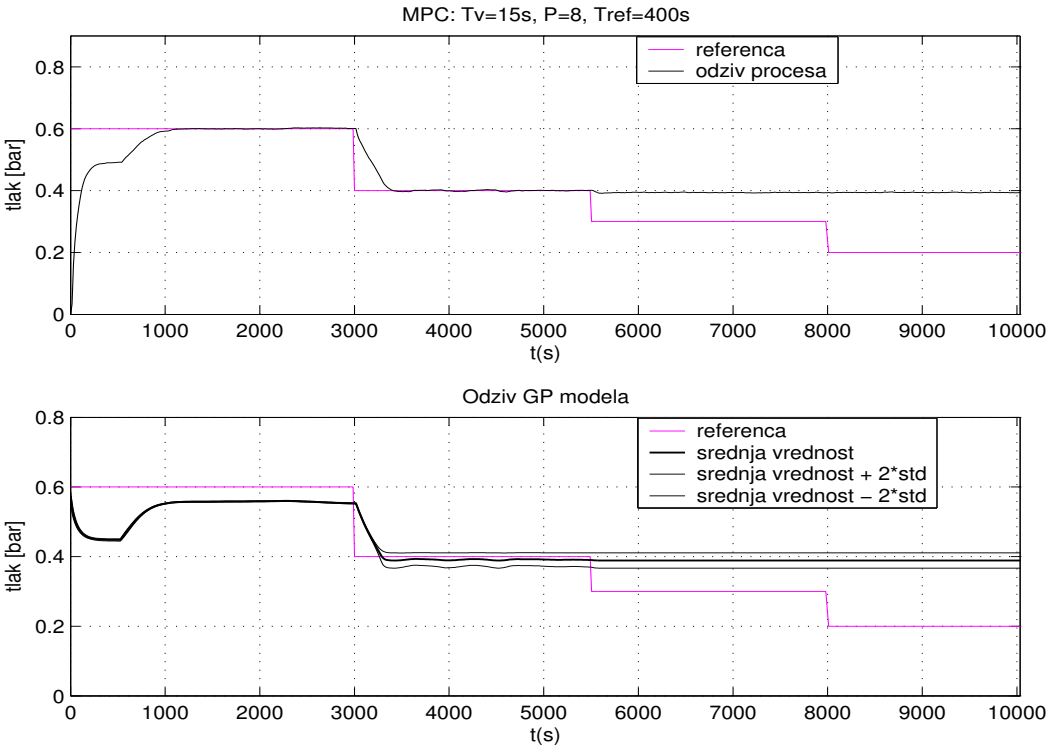
Slika 7.8: Nivo, varianca GP modela in regulirna veličina



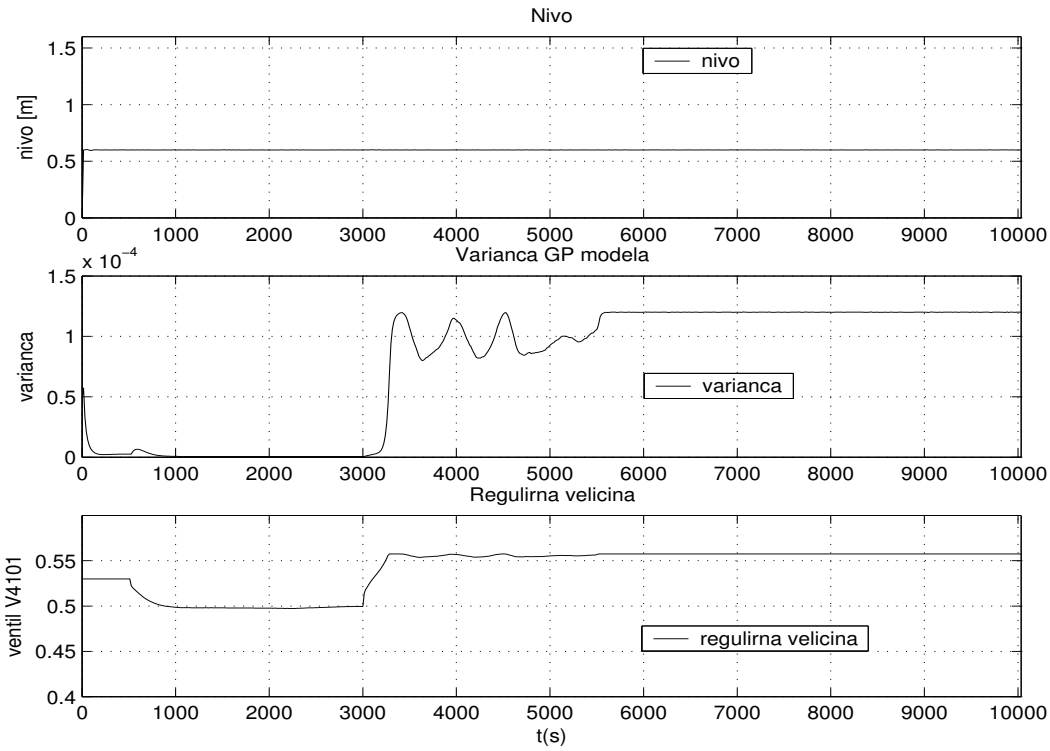
Slika 7.9: Odziv procesa in GP modela



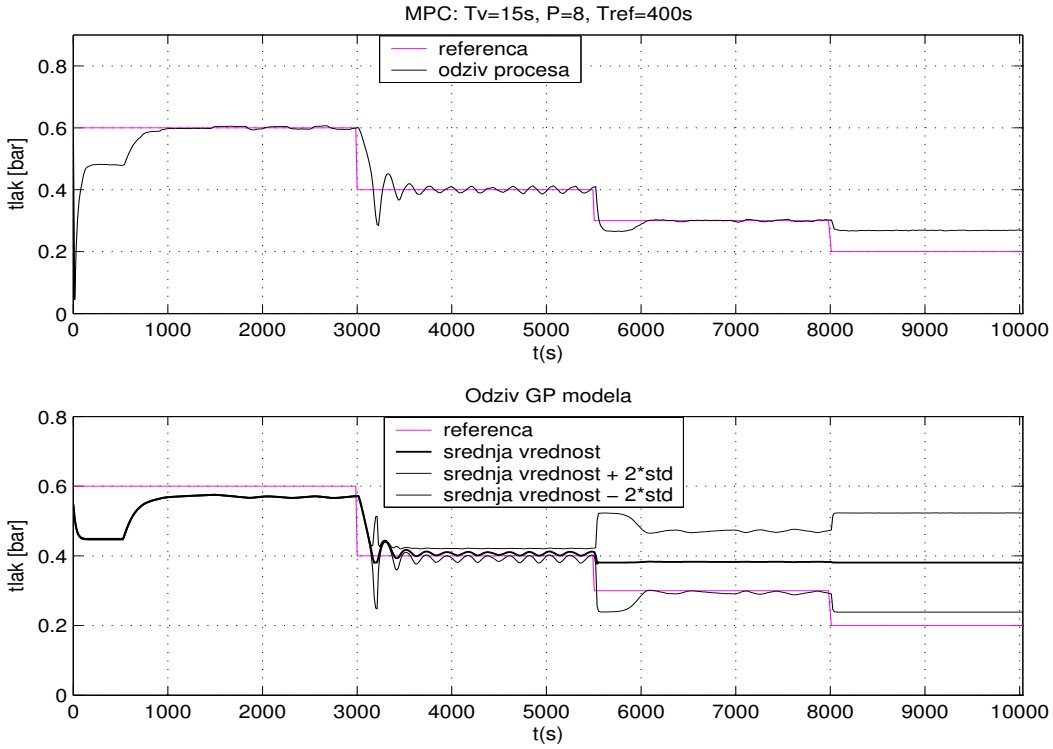
Slika 7.10: Nivo, varianca GP modela in regulirna veličina



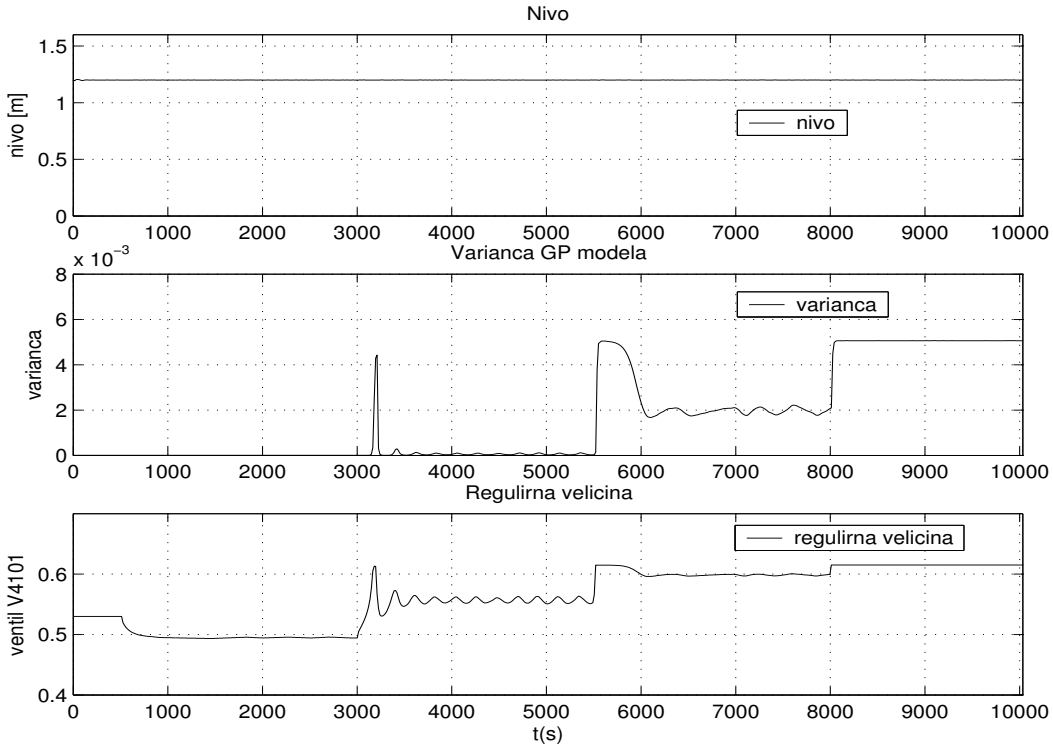
Slika 7.11: Odziv procesa in GP modela



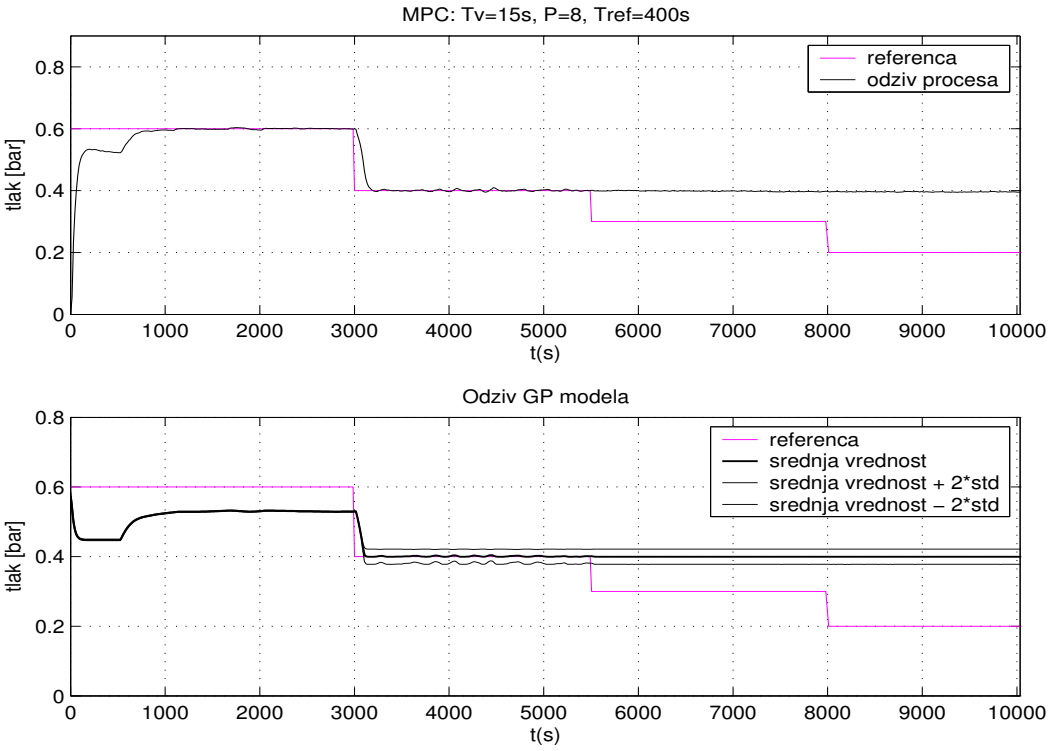
Slika 7.12: Nivo, varianca GP modela in regulirna veličina



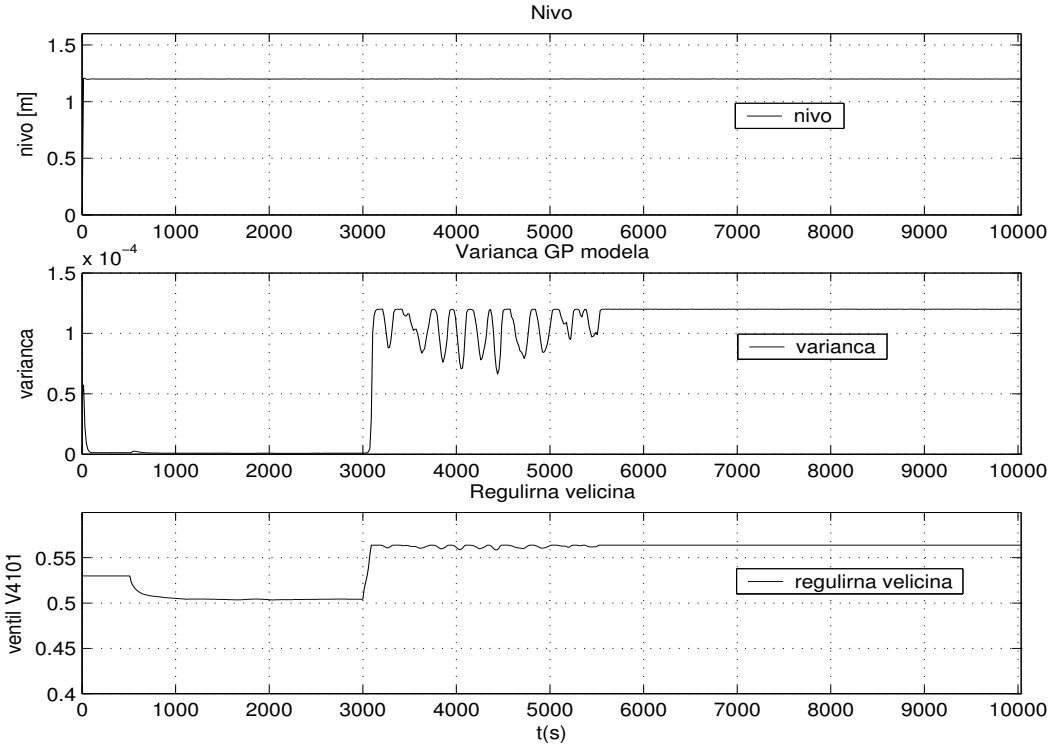
Slika 7.13: Odziv procesa in GP modela



Slika 7.14: Nivo, varianca GP modela in regulirna veličina



Slika 7.15: Odziv procesa in GP modela



Slika 7.16: Nivo, varianca GP modela in regulirna veličina

Predstavljeni rezultati kažejo, da je nelinearno prediktivno vodenje tlaka ločevalnika na osnovi GP modela uspešno v področju identifikacijskih podatkov, tako pri konstantnem kot pri spreminjajočem se nivoju vode v ločevalniku. Primerjava odzivov procesa in modela pokaže manjša odstopanja v ustaljenem stanju, ustaljeno razliko med izhodom procesa in modela pa prediktivni algoritem prek povratne zanke kompenzira (slika 3.2). V primeru določanja reference tlaka na mejnih področjih ali izven področja identifikacijskih podatkov, pa so zaradi prevelikih odstopanj GP modela od procesa rezultati neustrezni, kar je opazno v primerih, kjer ohlapna omejitev variance še dopušča privedbo procesa v neidentificirano območje. Po zmanjšanju meje za velikost dovoljene variance izhoda GP modela pa je iz rezultatov vodenja razvidno, da optimizacijski algoritem ne more več tvoriti regulacijskih signalov izven območja učne množice GP modela. Prediktivno vodenje z izogibanjem področij z veliko vrednostjo variance zagotovi robustnost, stabilnost in varnost vodenja, pri doseganju želenega delovanja znotraj območja identifikacijskih podatkov GP modela, boljši so tudi rezultati vodenja na mejnih področjih identifikacije. Procesa ne moremo pripeljati v področje, kjer GP model nima učnih podatkov in kjer bi zato prediktivni algoritem lahko destabiliziral sistem vodenja ali ga pripeljal v nepredvideno, za ljudi ali opremo, nevarno stanje.

8. Zaključek

Uporaba Gaussovih procesov, kot verjetnostnih neparametričnih modelov, se v zadnjem času iz modeliranja statičnih nelinearnih funkcij širi tudi na področje identifikacije nelinearnih dinamičnih sistemov. Osnovno izhodišče uporabe Gaussovih procesov je Bayesovo verjetnostno modeliranje, kjer na podlagi učne množice, sestavljene iz parov vhodnih in izhodih točk sistema, in predhodnih znanj o sistemu, namesto določanja parametrov funkcije, s katero sistem opisujemo, določimo porazdelitveni zakon nad parametri, predikcijo izhoda modela pa določimo prek poznavanja porazdelitvenega zakona parametrov.

Gaussov proces je naključen večdimenzionalen proces, katerega naključne spremenljivke, določene z vektorjem srednjih vrednosti in kovariančno matriko, so medsebojno porazdeljene po normalnem porazdelitvenem zakonu. Pri modeliranju predpostavimo, da vsaka izhodna točka učne množice določa posamezno naključno spremenljivko, kovariančna matrika pa je določena s kovariančno funkcijo, kot funkcijo parov vhodnih točk učne množice. Pri dani učni množici in določeni strukturi kovariančne funkcije, določimo porazdelitveni zakon nad njenimi parametri (imenovanimi hiperparametri). Predikcijo izhoda pri novi vhodni točki v splošnem lahko izračunamo z upoštevanjem celotnega porazdelitvenega zakona hiperparametrov, zaradi enostavnosti izračuna pa pri predikciji izhoda uporabljamo le predhodno določene najverjetnejše vrednosti hiperparametrov. Predikcija izhoda je normalna verjetnostna porazdelitev s srednjo vrednostjo in varianco, kar je prednost uporabe GP modelov v primerjavi z drugimi metodami nelinearnega modeliranja. Velikost variance, ki je odvisna od območja vhodnega prostora, podaja mero zaupanja v napovedano vrednost izhoda modela. Varianca je majhna na območjih, kjer je bila gostota točk učne množice velika, raste pa s približevanjem področjem, o katerih ima GP model relativno malo informacij.

Podobnost GP modelov z umetnimi nevronskimi mrežami, ki predstavljajo statično preslikavo vhodov v izhode, je izhodišče za simulacijo dinamičnih sistemov z GP modeli, kjer kot vhode v model uporabimo tudi zakasnjene vrednosti vhodnega in izhodnega signala. Ker na tak način uporabimo le predhodne srednje vrednosti izhoda modela, tako zavržemo informacijo o njihovi porazdelitvi in s tem zaupanju v vrednosti preteklih izhodov modela. V delu sta opisana pristopa, ki z različnimi aproksimacijami pri modeliranju dinamičnih sistemov z Gaussovimi procesi

upoštevata tudi informacijo o porazdelitvi preteklih izhodov GP modela. Obe metodi izhajata iz predpostavke o normalno porazdeljenem izhodu modela ob normalno porazdeljenih vhodih v model. Pri '*eksaktni*' metodi lahko, za določene oblike kovariančne funkcije, analitično določimo srednjo vrednost in varianco modela. Pri "*Taylorjevi aproksimaciji*" pa za izračun izhoda in variance modela uporabimo aproksimacijo z razvojem v Taylorjevo vrsto za srednjo vrednost in varianco normalno porazdeljenih vhodov v model. '*Naivna*' metoda pa pri propagiranju prejšnjih izhodov modela upošteva samo srednjo vrednost predikcij.

Na primeru procesa priprave plina je prikazana identifikacija z Gaussovimi procesi, primerjani so rezultati simulacije z različnimi načini propagiranja verjetnostnih porazdelitev preteklih izhodov GP modela. Splošna ugotovitev za vse načine je, da v identificiranem območju dobro opisujejo proces, kar ugotovimo iz majhnih vrednosti pogreškov in varianc predikcije. Izven identificiranega območja pa opazimo hitro naraščanje velikosti pogreškov kot varianc napovedanih izhodov modela.

Prediktivno vodenje, ki temelji na eksplicitni uporabi modela procesa, je v praksi uveljavljena naprednejša metoda vodenja, za katero obstaja veliko različnih variant, ki pa imajo vse podobne osnovne principe. Glavna značilnost je uporaba modela za napovedovanje prihodnjih vrednosti procesa in določitev prihodnjih regulirnih signalov z minimizacijo kriterijske funkcije razlike želenega in napovedanega odziva procesa. Pri nelinearnem prediktivnem vodenju se za napovedovanje izhoda procesa uporablja nelinearni model. Na primeru vodenja tlaka procesa priprave plina vidimo, da uporaba GP modela v nelinearnem prediktivnem vodenju, z vključitvijo omejitev velikosti variance predikcije v optimizacijski algoritem, zagotovi robustno in varno delovanje sistema vodenja v znanem, identificiranem območju. Procesa ne moremo pripeljati v neidentificirano področje, ki ga GP model ne opisuje ustrezno in kjer bi zato prediktivni algoritem lahko destabiliziral sistem vodenja. Izogibanje velikim vrednostim variance pomeni boljše rezultate vodenja na mejnih področjih identifikacijskih podatkov.

Predstavljeni razultati lahko služijo kot izhodišče za uporabo nelinearnega prediktivnega vodenja na osnovi GP modela v industrijskih aplikacijah, za kar bi bilo potrebno predvsem poiskati poenostavitve pri izračunavanju izhodov GP modela in izboljšati optimizacijski algoritem prediktivnega vodenja, za upoštevanje časovnih omejitev in omejitev optimizacijskih rezultatov v vsakem koraku optimizacije.

Obstoječ sistem osnovnega in postopkovnega vodenja procesa priprave plina, realiziran na nivoju programabilnih logičnih krmilnikov in SCADA aplikacije, smo razvili z upoštevanjem koncepta življenskega cikla sistemov vodenja. Sistemsko, ob delovanju sistema vodenja, ni bilo bilo omogočeno eksperimentiranje iz okolja Matlab/Simulink, kar je predstavljalo varnostno pomankljivost, saj postopkovno vodenje, realizirano na nivoju programabilnih logičnih krmilnikov, služi za varno avtomatsko zaustavitev sistema ob stanjih naprav in sistema, nevarnih za ljudi ali opremo. Razvit je bil vmesnik za vodenje procesa priprave plina v realnem času iz okolja Matlab/Simulink, ki omogoča hitro in enostavno eksperimentiranje. Vmesnik je bil, z upoštevanjem razvoja sistema vodenja po konceptu življenskega cikla, vgrajen v sistem postopkovnega vodenja. Okolje predstavlja način vključevanja eksperimentiranja iz okolja Matlab/Simulink za namene identifikacije in testiranja različnih algoritmov vodenja, v industrijske procese, vodene s standardno hierarhično strukturo PLK-SCADA, ob zagotavljanju visoke stopnje varnosti delovanja samih procesov. Z vmesnikom sta bili realizirana tako identifikacija kot vodenje tlaka procesa priprave plina.

Dodatek A

Metodologija načrtovanja sistema vodenja

Sistemi za vodenje postajajo vse bolj kompleksni, kar vodi k potrebi po sistematičnemu pristopu k njihovem razvoju, saj le na ta način lahko zagotovimo obvladljivost, zanesljivost, nadgradljivost sistemov in izpolnjevanje zahtev vodenja. Namen izgradnje računalniškega sistema vodenja procesnega laboratorija je izhajal že iz namena postavitve samega laboratorija – izobraževanje in raziskave na področju vodenja industrijskih procesov [35] [39] [5] .

A.1 Koncept življenskega cikla

Obstoječ sistem vodenja procesnega laboratorija je bil razvit na osnovi koncepta življenjskega cikla, s poudarkom na fazah izgradnje sistema – specificiranje, načrtovanje in izvedba sistema vodenja. Življenjski cikel zajema faze, aktivnosti, ki potekajo znotraj posameznih faz, in vmesne proizvode (rezultate posameznih faz), ki so posledica realizacije aktivnosti v posamezni fazi. Slika A.1 prikazuje linearni model življenjskega cikla, ki velja predvsem za unikatne izdelke, kamor spada tudi večina računalniško podprtih sistemov vodenja. Zaporedne faze izvedbe sistema so tako [14]:

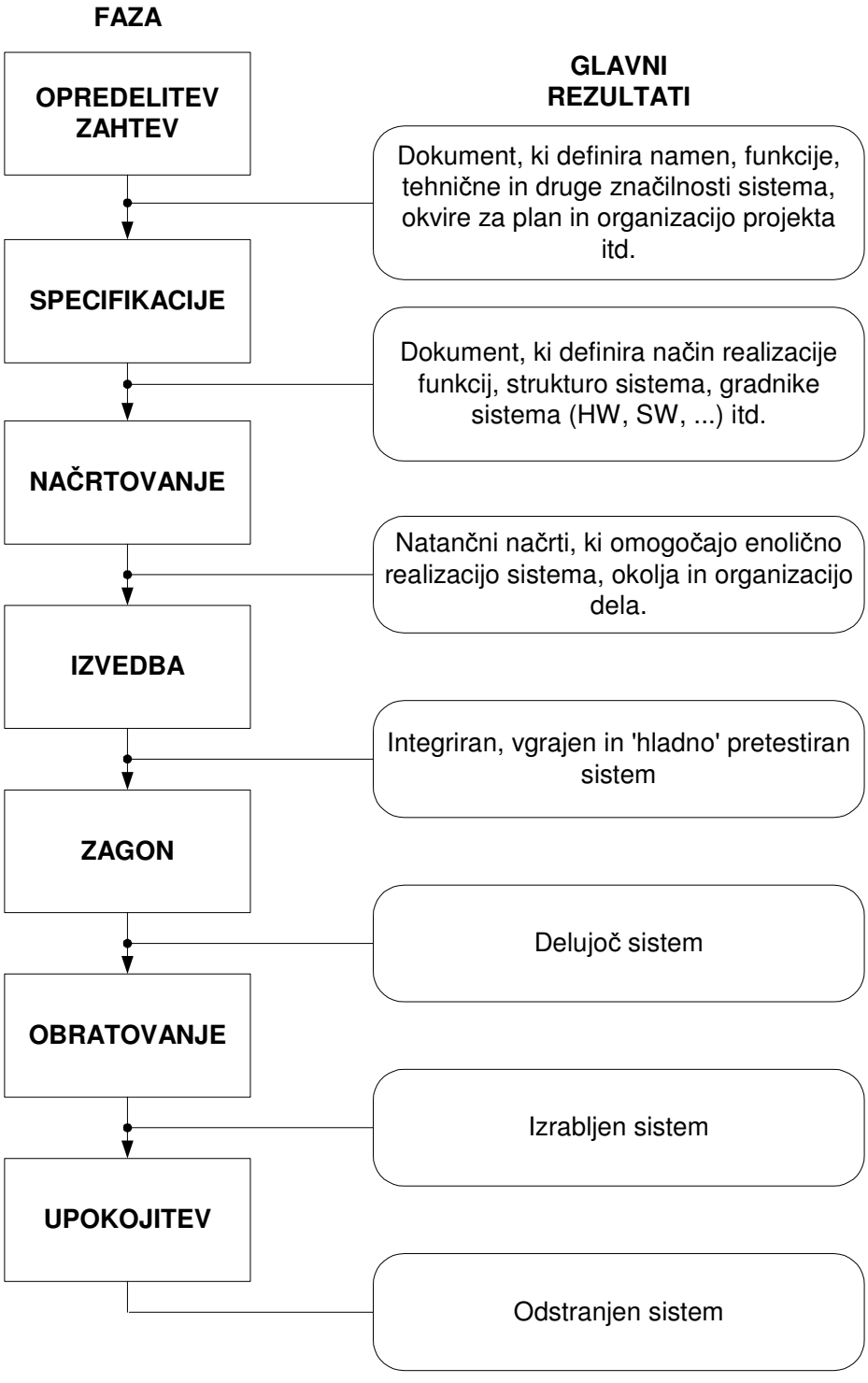
- Opredelitev zahtev (konceptualno načrtovanje), kjer najprej analiziramo potrebe, ki izhajajo iz uporabnikovih problemov in ciljev. V tej fazi ugotovimo dejanske potrebe, nato pa določimo cilje, ki jih želimo doseči. Rezultat aktivnosti te faze je dokument *Opredelitev zahtev*.
- Specifikacije (preliminarno načrtovanje) izhajajo iz zahtev, tu določimo s kakšnimi sredstvi, gradniki in funkcijami bodo izvedene zahteve. Rezultat aktivnosti te faze je dokument, ki vsebuje: specifikacije funkcij, koncept in strukturo sistema, specifikacije materialne in programske opreme itd.
- Načrtovanje (detajlno načrtovanje). V tej fazi se izdelajo natančni načrti za posamezne segmente računalniško podprtega sistema vodenja, ki omogočajo enolično realizacijo sistema, okolja in organizacije dela. V primeru dovolj podrobno podanih specifikacij delovanja sistema se lahko zgodi, da faza

načrtovanja ni potrebna, iz faze specifikacij lahko direktno preidemo na fazo izvedbe.

- Izvedba (izgradnja). Faza vključuje izdelavo in testiranje posameznih podsistemov. Sledi laboratorijska integracija podsistemov, tej pa montaža, integracija in testiranje podsistemov na procesu, ki ga želimo voditi. Rezultat faze je integriran, vgrajen in hladno testiran sistem.
- Zagon zajema dokončno integracijo, hladno in vroče testiranje in zagon sistema, tehnični prevzem, poskusno obratovanje, šolanje uporabnikov, revizijo dokumentacije. Rezultat faze je delujoč sistem.
- Obratovanje je faza življenjskega cikla, v kateri se realizira osnovni namen izgradnje sistema vodenja, vodenje procesa s pomočjo postavljenega sistema vodenja. Pod obratovanjem razumemo, da sistem deluje v obsegu časa in funkcionalnosti tako, kot je bilo predvideno v specifikacijah. Rezultat zaključka te faze je izrabljen sistem.
- Upokojitev sistema vodenja izvedemo, ko po tehničnih, ekonomskih ali drugih kriterijih ne ustreza več svojemu namenu. Rezultat faze Upokojitev pa je odstranjen sistem, s čimer se tudi zaključi življenjski cikel sistema.

Univerzalnega modela življenjskega cikla, ki bi bil uporaben za vsa področja dela, ni. Zato moramo v praksi model življenskega cikla, nabor faz in aktivnosti ter preslikavo aktivnosti v posamezne faze, prirediti konkretnemu problemu, ki ga rešujemo. Različni modeli življenjskega cikla predstavljajo različne načine razvoja in uporabe določenega sistema [35], [18].

V nadaljevanju sta podrobneje predstavljeni prvi dve fazi načrtovanja sistema vodenja, opredelitev zahtev in specificiranje, pri katerih je ustrezen sistematičen pristop najpomembnejši, saj bistveno vplivata na uspešno izvedbo naslednjih faz razvoja sistema vodenja. Za fazo specificiranja so opisani formalizmi za opisovanje specifikacij, ki se uporabljajo za domeno procesnega vodenja.



Slika A.1: Linearni življenjski cikel načrtovanja sistema vodenja

A.2 Opredelitev zahtev

A.2.1 Uvod

V uvodu moramo najprej ugotoviti dejanske potrebe in želje za sistem vodenja s prečiščevanjem zahtev in odpravljanjem njihovih nekonsistentnosti, definirati cilje in merila za vrednotenje stopnje doseganja ciljev.

A.2.2 Analiza obstoječega stanja

Za uvedbo novega sistema vodenja moramo najprej podrobno spoznati obstoječe stanje procesa z morebitnim že obstoječim sistemom vodenja, zajemom procesnih veličin in aktuatorji.

A.2.3 Postopkovne zahteve

Postopkovne zahteve izhajajo iz želenih funkcij sistema vodenja. Pri postopkovnih zahtevah definiramo vrstni red posameznih operacij v procesu, definiramo katere naprave, in v kakšnih načinih delovanja, bomo upravljali v posameznih operacijah, funkcije pa strukturiramo tako, da najprej definiramo vhodne parametre za dano funkcijo, nato potrebne obdelave in na koncu ustrezne učinke, ki jih realiziramo z izhodnimi signali preko ustreznih aktuatorjev. Specificiramo tudi, kako bo postopkovno vodenje procesa vidno s strani operaterja in način upravljanja z operaterske postaje. Sem spadajo uporabljeni operaterski ukazi, prikaz in spreminjanje parametrov postopkovnega vodenja in prikaz stanja, v katerem se postopek nahaja.

A.2.4 Zahteve po vmesnikih

V računalniško podprtih sistemih za vodenje procesov predstavljajo vmesniki ključni element povezovanja računalniškega sistema z okoljem. Vmesniki so prisotni na naslednjih nivojih [14], [39]:

- proces – računalnik (nivo senzorjev, aktuatorjev itd),
- človek – računalnik (naprave za komunikacijo s človekom),
- računalnik – računalnik (v večračunalniškem sistemu),

- uporabniški program – računalnik,
- uporabniški program – uporabniški program.

A.3 Specifikacije

Specifikacije izhajajo iz zahtev, pri čemer so podatki iz zahtev podrobneje razgrajeni. Specifikacije določajo, kako in s čim (s kakšnimi sredstvi, gradniki, funkcijami) izvesti zahteve.

A.3.1 Specifikacije funkcij

Pri specifikacijah funkcij sistema definiramo, kako bomo posamezne funkcije izvedli (algoritmi, postopki), s kakšnimi podatki bodo upravljale in kateri podatki bodo rezultat teh funkcij. Ker je s pisnim opisom težko enolično opisati funkcijske lastnosti in soodvisnosti, se pri specifikacijah poslužujemo modelov, ki predstavljajo formalizem za opisovanje. Zaradi različnih področij dela, z različnimi lastnostmi in zahtevami, modeli niso univerzalni, temveč veljajo samo za določeno domeno (npr. domena sistemov postopkovnega vodenja). Domena vodenja procesov se deli na tri poddomene (slika A.2):

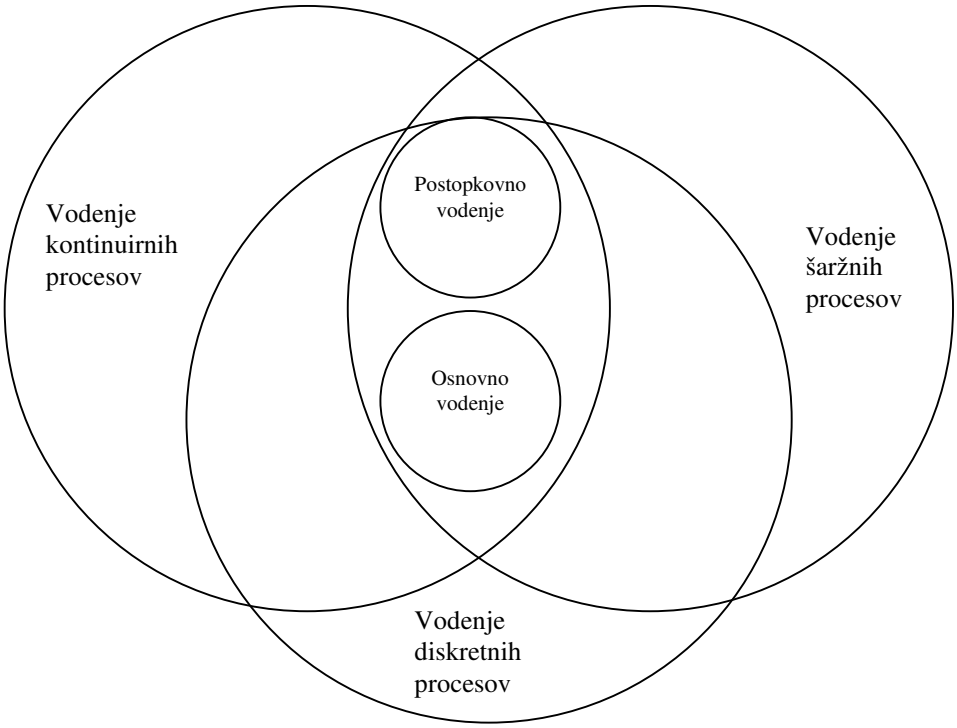
- vodenje kontinuirnih (zveznih) procesov,
- vodenje šaržnih procesov,
- vodenje diskretnih procesov.

Vsako od domen vodenja na t. i. sistemskem nivoju dekomponiramo na:

- postopkovno vodenje (ang. *Procedural control*),
- osnovno vodenje (ang. *Basic control*).

Namen osnovnega vodenja je doseganje in vzdrževanje določenega stanja procesne opreme ali procesa in je običajno za vse poddomene procesnega vodenja enako. Postopkovno vodenje opisuje izvajanje procesno usmerjenih aktivnosti v urejenem vrstnem redu in je ponavadi za vsako od poddomen procesnega vodenja različno.

Namen dekompozicije sistema vodenja na sistemskem nivoju je lažje obvladovanje kompleksnosti analize, načrtovanja, implementacije, uporabe in vzdrževanja sistema.



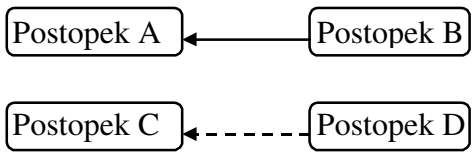
Slika A.2: Poddomene procesnega vodenja

A.3.2 Model postopkovnega vodenja

Primer modela postopkovnega vodenja je notacija razširjenih sočasnih končnih avtomatov (angl. *Extended Concurrent Final State Machine - ECFSM*), ki prikaže obnašanje postopkov in njihovih medsebojnih odvisnosti. Notacijo sestavljajo naslednji diagrami [14]:

- diagram odvisnosti med postopki (*angl. Procedural Dependencies Diagram - PDD*),
- diagram prehajanja stanj postopka (*angl. Procedural State Transition Diagram - PSTD*),
- diagram odvisnosti stanj postopka (*angl. Procedural Dependency State Transition Diagram - PDST*),

ki so opisani s slikami A.3, A.4, A.5. Diagram odvisnosti med postopki PDD se dekomponira na diagram prehajanja stanj postopka PSTD in na diagram odvisnosti stanj postopka PDST, akcije posameznih stanj postopka in pogoje za prehode med stanji v PSTD, pa nadalje opišemo s psevdo jezikom.

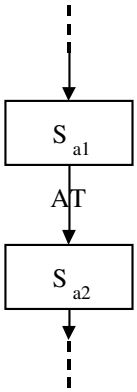


Slika A.3: diagram odvisnosti med postopki - *PDD*

Na diagramu so postopki prikazani s zaobljenimi pravokotniki, odvisnosti med njimi pa z usmerjenimi povezavami. Zgornja relacija na sliki A.3 pomeni, da so nekateri prehodi postopka A odvisni od nekaterih stanj postopka B, spodnja relacija pa govori o tem, da so prehodi stanj postopka C posledica prehodov stanj postopka D, kar se imenuje *propagacija*. Na tem nivoju niso vidna konkretna stanja in konkretni prehodi posameznega postopka ter odvisnosti med prehodi dveh postopkov. Zato se PDD dekomponira na dva diagrama, ki opisujeta relacije na nižjem nivoju. Postopki iz PDD se dekomponirajo na diagram prehajanja stanj postopka PSTD (slika A.4), povezava odvisnosti postopkov iz PDD pa se dekomponira na diagram odvisnosti stanj postopkov PDSTD (slika A.5).

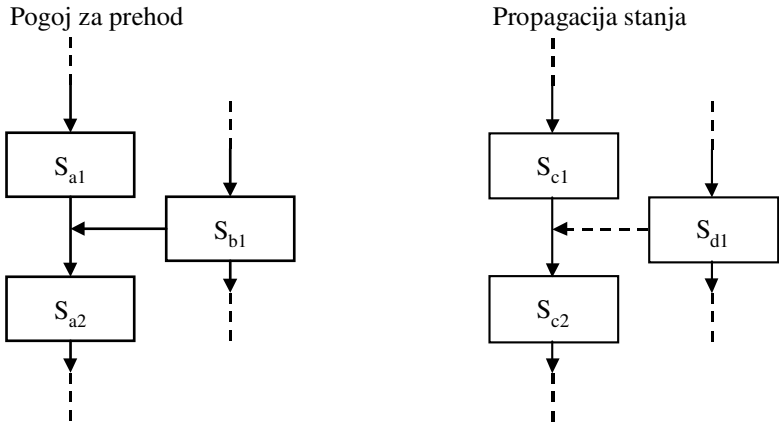
Diagram PSTD prikazuje kako lahko postopek prehaja med stanji. Stanja opisujejo obnašanje elementov postopkovnega modela v določenem trenutku ter možne ukaze za prehod v novo stanje. Število možnih stanj, prehodov med stanji in ukazov, ki sprožijo prehode, je različno za elemente postopkovnega modela. Prehod med stanji in s tem spremembo stanja lahko povzroči notranja logika ali pa zunanji ukazi, ki jih pošilja postopkovno vodenje oz. operater. Vsa stanja so zbrana v tabeli A.1, ukazi pa so zbrani v tabeli A.2.

Na sliki A.4 je prikazan primer, ki določa, da je za prehod postopka A iz stanja S_{a1} v stanje S_{a2} avtomatski (AT, angl. *auto transition*).



Slika A.4: Diagram prehajanja stanj postopka (PSTD)

Diagram PDSTD prikazuje odvisnost prehajanja stanj enega postopka od stanj drugega postopka. Levi primer na sliki A.5 kaže, da je prehod postopka A iz stanja S_{a1} v stanje S_{a2} mogoč le ob stanju S_{b1} postopka B, desni primer na isti sliki pa kaže, da je prehod postopka C iz stanja S_{c1} v stanje S_{c2} propagacija stanja S_{d1} postopka D.



Slika A.5: Diagram odvisnosti stanj dveh postopkov (PDSTD)

Stanje	Pomen stanja
USTAVLJENO ang. *Stopped*	To je začetno stanje, hkrati pa se samodejen prehod v to stanje izvrši po zaključku izvajanja logike stanja *USTAVLJANJE*. Izvajanje postopka se ustavi in oprema se nahaja v varnem stanju.
PREVERJANJE POGOJEV ZA ZAGON ang. *Check Conditions*	V to stanje postopek preide iz stanja USTAVLJENO ob ukazu START. V tem stanju se preverijo pogoji za zagon postopka. Če le-ti niso izpolnjeni, se postopek samodejno vrne v stanje USTAVLJENO.
ZAGANJANJE ang. *Starting*	V tem stanju se izvedejo akcije za zagon postopka.
OBRATOVANJE ang. *Running*	To je stanje normalnega obratovanja.
PAVZIRANJE ang. *Pausing*	Prehod v *PAVZIRANJE* povzroči določeno stanje opreme, ki ne dovoljuje (varnega) normalnega obratovanja. To stanje se v kompleksnejših postopkih razdeli na stanji ZAKASNITEV (angl. D*elay*) in SEKVENCO (angl. S*equence*).
PAVZIRANO ang. *Paused*	Prehod v stanje je samodejen po zaključku izvajanja logike stanja *PAVZIRANJE*. V tem stanju postopek čaka, da se zopet vzpostavijo pogoji, ki omogočajo vnovično normalno obratovanje.
OBIČAJNO USTAVLJANJE ang. *Regular Stopping*	Operater z ukazom *Stop* sproži brezpogojen začetek izvajanja logike ustavljanja z namenom ustavitve izvajanja postopka. Postopek po izvršenem ustavljanju preide v stanje *USTAVLJENO*. To stanje se v kompleksnejših postopkih razdeli na stanji ZAKASNITEV (angl. D*elay*) in SEKVENCO (angl. S*equence*).
HITRO USTAVLJANJE ang. *Fast Stopping*	Ukaz *Prekini* sproži brezpogojen začetek izvajanja logike ustavljanja (npr. zaradi hujše okvare na opremi), kadar ni mogoče izvesti regularne sekvence ustavljanja. Sekvenca *HITRO USTAVLJANJE* se vedno izvede do konca in postopek preide v stanje *USTAVLJENO*.

Tabela A.1 - Stanja postopkovnega vodenja zveznih procesov

Ukaz		Pomen ukaza
START	ang. *Start*	Zaženi postopek.
STOP	ang. *Stop*	Ustavi izvajanje postopkovnega elementa in opremo postavi v varno stanje.
PREKINI	ang. *Abort*	Brezpogojno prekini izvajanje postopkovnega elementa in opremo postavi v varno stanje.

Tabela A.2 - Ukazi in njihov pomen

Stanja elementov postopkovnega modela se glede na možne prehode in pogoje za prehode delijo na mirujoča in prehodna stanja.

Mirujoča stanja (ang. *quiescent state*) so tista stanja, v katerih postopki sicer mirujejo, vendar je možno nadaljevanje postopka v tistem koraku, v katerem je bil postopek zadržan. Mirujoči stanji sta USTAVLJENO in PAVZIRANO. Pred končnim ali mirujočim stanjem je vedno vsaj eno prehodno stanje. Ostala stanja iz tabele A.1 so prehodna stanja (ang. *transient state*). Prehode v prehodna stanja povzročijo ukazi iz tabele A.2. Prehodna stanja so PREVERJANJE POGOJEV, ZAGANJANJE, OBRATOVANJE, PAVZIRANJE, HITRO USTAVLJANJE in REGULARNO USTAVLJANJE. Vsak element postopkovnega modela se lahko nahaja samo v enem od naštetih stanj in nikoli v dveh hkrati.

Stanja so prikazana s pravokotnimi elementi, usmerjene povezave pa prikazujejo prehode med njimi. Oznake ob povezavah označujejo tip prehoda. V tabeli A.3 so opisani tipi prehodov, njihove oznake in pogoji.

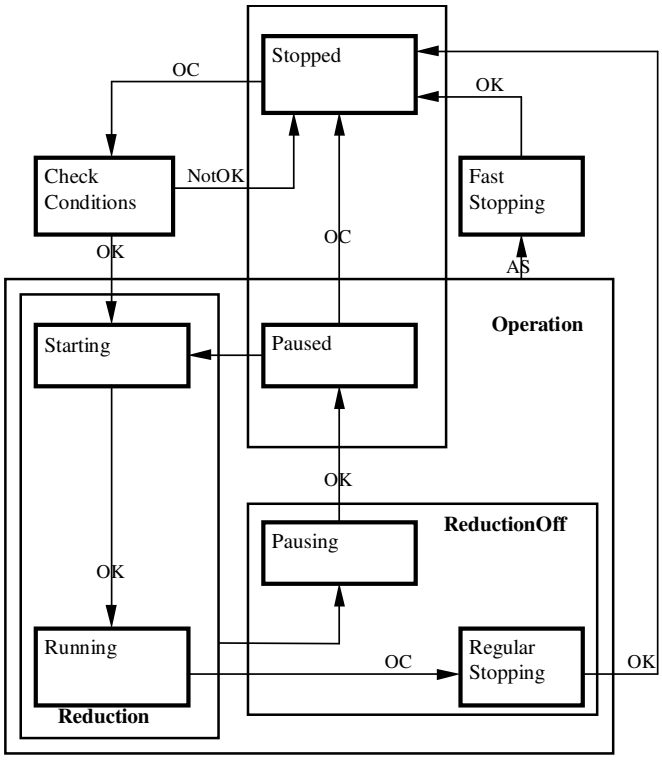
Prehod	Oznaka	Pogoji za prehod
POGOJ IZPOLNJEN	OK	Prehod se zgodi, ko se izvršijo vsa opravila v trenutnem stanju oziroma ko so izpolnjeni pogoji za prehod v novo stanje (npr. iz stanja PREVERJANJE POGOJEV v stanje STARTANJE).
POGOJ NI IZPOLNJEN	NotOK	Prehod se zgodi, ko niso izpolnjeni pogoji za prehod v novo stanje (npr. iz stanja PREVERJANJE POGOJEV se ob neizpolnjenih pogojih proces vrne v staro stanje USTAVLJENO).
AVTOMATSKI PREHOD ang. *Automatic transition*	AT	Pogoj za ta prehod je določeno stanje procesne opreme oziroma stanje drugega procesa, ki je vezan na prvega.
AVTOMATSKO USTAVLJANJE ang. *Automatic stopping*	AS	Ta prehod je različica AVTOMATSKEGA PREHODA in se zgodi zaradi hujše napake na procesni opremi oziroma zaradi operatorjevega ukaza PREKINI. Pri tem proces vedno preide v stanje HITRO USTAVLJANJE.
UKAZ OPERATERJA ang. *Operator command*	OC	Prehod se zgodi, ko operater pošlje vodenju ustrezen ukaz.
PREHOD ZARADI ODVISNOSTI OD DRUGEGA POSTOPKA	DP	Prehod se zgodi zaradi propagacije stanja drugega postopka ali zaradi izpolnjenih pogojev za prehod, ki se nanašajo na drugi postopek.

Tabela A.3 – Opis prehodov

V diagramih PSTD in PDSTD lahko poleg stanj nastopajo tudi nadstanja, ki jih definiramo zaradi poenostavitev diagramov ali pa zato, ker bi z njimi radi prikazali sorodnost med različnimi stanji. Vsebujejo dve ali več stanj enega postopka, ki imajo določene akcije ali pogoje za prehod v novo stanje skupne, zato te skupne elemente v vsebovanih stanjih izpustimo in jih umestimo v nadstanje. Nadstanja prikažemo s pravokotniki okoli stanj, ki jih nadstanja vsebujejo. Nadstanja, ki jih definiramo za poenostavitev diagramov PSTD, vsebujejo prehode oziroma akcije, ki jih prav tako kot za elementarna stanja opisujemo s psevdo kodo.

Na sliki A.6 je primer obeh vrst nadstanj. Stanji *Stopped* in *Paused* sta združeni v nadstanje za poenostavitev diagrama PDSTD, stanja *Starting*, *Running*, *Pausing*,

Paused in *Regular stopping* so združena v nadstanje *Operation* zaradi skupnih vzrokov za prehod v stanje *Fast stopping*, kar poenostavi diagram PSTD. Na sliki je tudi primer gnezdenja nadstanj; stanji *Starting* in *Running* sta obenem še v nadstanju *Reduction*, ker imata skupni vzrok za prehod v stanje *Pausing* (ta prehod je odvisen od drugega postopka). Primer vsebuje tudi poseben primer nadstanja – *ReductionOff*. To vsebuje stanji *Pausing* in *Regular stopping*, definirano pa je zaradi identičnih akcij v obeh stanjih (stanji sicer nimata skupnih povezav) [5].



Slika A.6 - Primer diagrama PSTD z nadstanji

A.3.3 Oznake spremenljivk

Zaradi množice spremenljivk in signalov, ki nastopajo v sistemu vodenja, je smiselno uvesti formalizacijo za njihovo označevanje. Tako jih razdelimo glede na njihov izvor in tip informacije. Oznake in razlage vhodno izhodnih informacij se nahajajo v tabeli A.4.

Postopkovno vodenje prejema ukaze OC od operaterja, ukaze IC od drugih postopkov (medpostopkovni ukazi) in signalizacijo ES od osnovnega vodenja.

Operaterju pošilja signalizacijo postopkov vodenja PS, osnovnemu vodenju pa ukaze EC. Na nivoju postopkovnega vodenja imamo tako opravka le s signalizacijo in ukazi, ne pa tudi s fizičnimi vhodnimi in izhodnimi signali, tj. z električnimi analognimi in/ali digitalnimi signali povezanimi z napravami (ventil, nivojsko stikalo, črpalka). Ti so zajeti v osnovnem vodenju.

Zaloga vrednosti signalizacije osnovnega vodenja ES oz. postopkovnega vodenja PS je odvisna od števila možnih stanj, ki jih lahko zavzame določen gradnik osnovnega oz. postopkovnega vodenja. Podobno kot signalizacija tudi ukazi vsebujejo informacijo na višjem nivoju abstrakcije, ki se v osnovnem vodenju spremeni v ustrezno vrednost izhodnega signala. Zaloga vrednosti ukazov postopkovnega vodenja je odvisna od števila ukazov, ki jih določen gradnik postopkovnega vodenja lahko prepozna.

Parametri EP in PP so podatki, ki jih operater vnaša v sistem vodenja, npr. maksimalno dovoljeno odstopanje regulirane veličine od reference, želeno vrednost tlaka v ločevalniku in podobno.

V/I informacije	**Naprava**	**Postopkovno vodenje**	**Opis**
Signalizacija	**ES** (ang. *Equipment Signalisation*)	**PS** (ang. *Procedural Signalisation*)	Stanje elementa
Ukaz	**EC** (ang. *Equipment Command*)	**OC** (ang. *Operator Command*) **IC** (*angl. Internal Command)*	Ukazi napravam in postopkom. Lahko so izhod drugega postopka ali pa izvirajo iz zunanjega vira (operater).
Parameter	**EP** (ang. *Equipment Parameter*)	**PP** (ang. *Procedural Parameter*)	Parameter naprave ali postopka.

Tabela A.4 – Oznake ukazov in signalizacije

A.3.4 Nabor spremenljivk postopkovnega vodenja

Postopkovno vodenje zveznih procesov vsebuje štiri spremenljivke, s katerimi se zagotovi pravilno izvajanje postopka in nadzor nad njimi (tabela A.5).

Spremenljivka	Tip	Pomen
Stanje Sistema	REAL	Signalizacija stanja podprocesa
OC	REAL	Ukaz operaterja postopku
VzrokUstavitve	WORD	Signalizacija vzroka ustavitve
Sporočilo	WORD	Signalizacija poteka izvajanja postopka

Tabela A.5 – Oznake ukazov in signalizacije

A.3.5 Psevdo jezik

Pri specifikacijah izhajamo iz zadanih ciljev vodenja, ki jih moramo v tej fazi enoumno opisati in tako definirati potrebna zaporedja dogodkov. Samo s pisnim opisom pogosto ne moremo enolično opisati zahtevanega zaporedja dogodkov, hkrati pa predstavlja pisni opis tudi slabo izhodišče za programiranje PLC-jev. Zato je smiselno uvesti nek formalizem – psevdo jezik, ki upošteva določene zakonitosti. Na osnovi pisnega opisa bomo specificirali zahteve procesnega vodenja v psevdo kodi. Tako dobljena psevdo koda ni odvisna od tipa PLK-ja in predstavlja osnovo za programiranje kateregakoli PLK-ja.

Psevdo jezik, ki je uporabljen za specificiranje zahtev osnovnega in postopkovnega vodenja procesnega laboratorija, temelji na simbolih. Ukazi psevdo jezika (tabela A.6) se definirajo in dopolnjujejo glede na potrebe načrtovanja, pri tem pa je potrebno upoštevati naslednje zahteve:

- nedvoumnost ukazov,
- enostavno sintakso,
- jasna pravila preslikave in enostavnost preslikave v kodo PLK-ja,
- konsistentnost pri uporabi.

1. **Oznake ukazov**	
Oznaka	***Pomen***
↑	Vklop pogona (motorja)
↓	Izklop pogona (motorja)
□	Odpiranje ON-OFF lopute/ventila
⦿	Zapiranje ON-OFF lopute/ventila
E	Usposobi (Enable)

D	Onesposobi (Disable)
2. Oznake stanj	
Oznaka	***Pomen***
↑	Pogon (motor) deluje
↓	Pogon (motor) stoji
↻	Vrtenje
E	Napaka
○	ON-OFF-loputa/ventil odprt
●	ON-OFF-loputa/ventil zaprt
Off	Digitalni vhod je enak 0
On	Digitalni vhod je enak 1
3. Oznake procesnih spremenljivk – združeno stanje in ukaz	
Oznaka	***Pomen***
◐	Stanje zvezne lopute/ventila (preverjanje stanja: ◐=x; nastavitev stanja ◐:=x)
4. Oznake načinov	
Oznaka	***Pomen***
L	Lokalno
R	Daljinsko
RA	Daljinsko avtomatsko
RM	Daljinsko ročno
5. Operatorji	
Oznaka	***Pomen***
= , > , <	Enostavni relacijski operatorji
≥ , ≤	Sestavljeni relacijski operatorji
¬	Negacija (logični operator NOT)
&	Konjunkcija (logični operator AND)
V	Disjunkcija (logični operator OR)
⇒	Implikacija (iz resničnosti leve strani sledi resničnost desne strani)
△(PP)	Zakasnitev v trajanju PP
∇(pogoj)	Čakanje na pogoj
∇_L(pogoj, PP)	Časovno omejeno (PP) čakanje na pogoj
T	Prehod stanja
DURATION(pogoj, PP)	Funkcija, ki vrne TRUE, če je pogoj TRUE v trajanju PP
6. Krmiljenje toka izvajanja aktivnosti	
Oznaka	***Pomen***
IF $pogoj_1$ THEN $akcija_1$ ELSEIF $pogoj_2$ THEN $akcija_2$ ..... ELSEIF $pogoj_m$ THEN $akcija_m$	Izvedba ene od m-akcij glede na vrednost pripadajočih pogojev, če noben od pripadajočih pogojev ni izpolnjen, se izvede $akcija_n$

ELSEIF pogoj$_n$ ENDIF;	

Tabela A.6 - Nabor ukazov psevdo jezika

A.3.6 Označevanje spremenljivk v psevdo kodi

V psevdo kodi vhodne parametre ali ukaze podamo tako, da je iz sintakse razvidno, za kakšen podatek gre in kateri napravi procesa ali kateremu procesu pripada. Predpone in jedro so med seboj ločene z znakom \, vrste predpon pa si sledijo v tem vrstnem redu:

- predpona po tabeli A.4, ki določa vrsto elementa, npr. EC, OC, ES, PS, PP,
- jedro, ki se tvori po enem izmed naslednjih načinov:
  - po imenu konkretne naprave procesnega laboratorija, npr. PC4101
  - po imenu postopkovnega elementa, npr. SC za postopek podprocesa priprave plina,
- pripona, ki je potrebna za ločevanje več podatkov, ki imajo isto predpono in jedro, npr. EC\LC4101\TrVal in EC\LC4101\SPRmt.

Primeri psevdo kode so:

- spremenljivki EC_PC4101_SPTR priredi vrednost Tracking:

  `EC\PC4101\SPTR:=Tracking`
- vnesi zakasnitev za čas PP_SC_TransTime:

  $\triangle$(`PP\SC\TransTime`)
- ob ukazu STOP preidi v stanje STOPPING:

  `OC\SC=Stop` $\Rightarrow$ T: `Stopping`
- postopek čaka v času PP_SC_StopMaxDuration, da vrednost signala ES\PT4101 pade pod vrednost parametra PP_SC_R41StopPressure:

  ∇_L(`ES\PT4101<PP\SC\R41StopPressure,PP\SC\StopMaxDuration`)
- spremenljivki EP_PC4101_SPRemote priredi vrednost parametra PP_PC4101_SPRemote:

```
EC\PC4101\SPRemote:= PP\PC4101\SPRemote
```

- zapri ventil V4101:

  `EC\V4104[⦿]`

- vključi črpalko P4101:

  `EC\P4101[⇞]`

Dodatek B

Specifikacije sistema vodenja procesa priprave plina

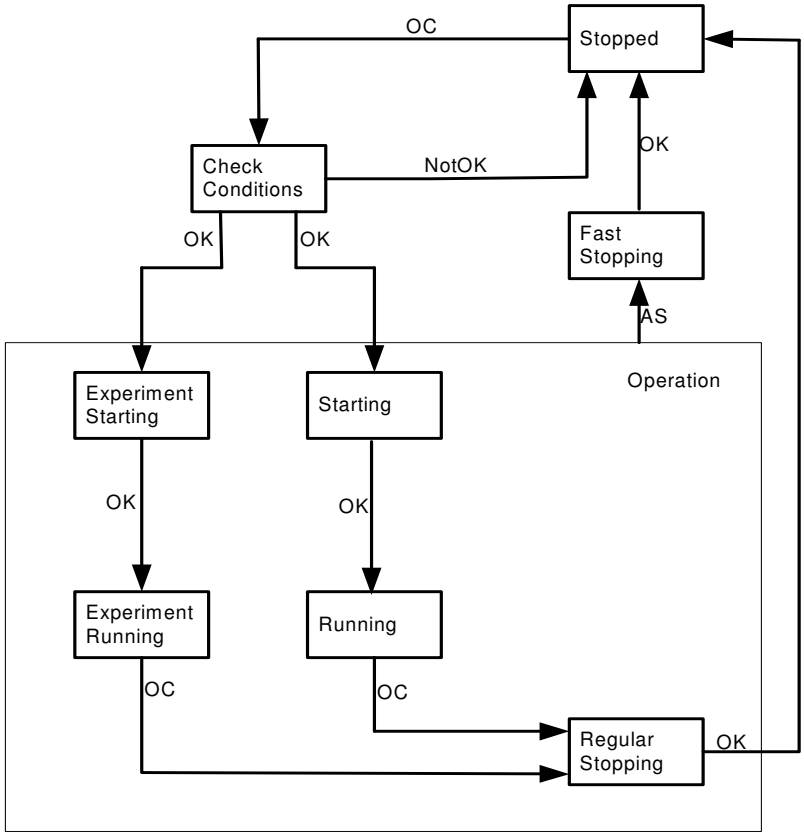
Specifikacije sistema vodenja procesa priprave plina so sestavljene iz specifikacij postopkovnega vodenja, podanih z diagramom prehajanj stanj PSTD, specifikacij akcij posameznih stanj postopka in pogojev za prehode med stanji, opisanimi s psevdo jezikom in naborom vrednosti spremenljivk postopkovnega vodenja [28].

B.1 Diagram prehajanja stanj PSTD

Proces priprave plina ni postopkovno odvisen od drugih procesov laboratorija, zato za opis postopka nista potrebna diagram odvisnosti med postopki PDD in diagram odvisnosti stanj postopka PDSTD. Proces postopkovnega vodenja zato opišemo samo z diagramom prehajanja stanj PSTD (slika B.2).

Postopek podprocesa priprave plina vsebuje mirujoče stanje *Stopped*, ostala stanja pa so prehodna. Prav tako vsebuje nadstanje *Operation*, ki združuje stanja *Starting*, *Running* in *Regular Stopping* ter za namen varnega eksperimentiranja iz okolja Matlab dodatni stanji *Experiment Starting* in *Experiment Running* . Nadstanje *Operation* je bilo definirano zaradi skupnih vzrokov za *AS* prehod iz stanj, ki jih vsebuje, v stanje *Fast Stopping*.

Za opis postopka najprej predpostavimo, da je le-ta na začetku v stanju *Stopped*. Ob operaterjevem ukazu postopek preide v stanje *Check Conditions*, v katerem se preverijo pogoji za nadaljevanja izvajanja; če so le-ti izpolnjeni, postopek preide ali v stanje *Starting* ali *Experiment Starting* (odvisno od operaterjevega ukaza), drugače pa se vrne v stanje *Stopped*. Po izvršitvi akcij v stanju *Starting* postopek preide v stanje *Running* ali *Experiment Running*. Iz stanja *Running* ali *Experiment Running*, z ustreznim ukazom operaterja, preide v stanje *Regular stopping*, po izvršitvi akcij pa v stanje *Stopped*. Ko je postopek v katerem od stanj, ki jih vsebuje nadstanje *Operation*, lahko ob morebitni napaki opreme oziroma ob operaterjevem ukazu Prekini, postopek preide v stanje *Fast stopping*, po izvršitvi akcij tega stanja pa v stanje *Stopped*.



Slika B.2: - Diagram prehajanja stanj podprocesa priprave plina

B.2 Nabor vrednosti spremenljivk postopkovnega vodenja

Specifikacije za postopkovno vodenje procesa vsebujejo pet spremenljivk, s katerimi se zagotovi pravilno izvajanje postopka in nadzor nad njim. Prikazuje jih tabela B.1:

Spremenljivka	Tip	Pomen
State	WORD	Signalizacija stanja podprocesa
OC	WORD	Ukaz operaterja postopku
CauseOfStop	WORD	Signalizacija vzroka ustavitve
Message	WORD	Signalizacija poteka izvajanja postopka
StepCounter	WORD	Števec korakov

Tabela B.1 - Nabor spremenljivk postopkovnega vodenja

Stanja postopkovnega vodenja

Spremenljivka *State* signalizira stanje v katerem je postopek. Vsakemu stanju tako pripada določena vrednost spremenljivke. Vrednosti so zbrane v tabeli B.2:

Vrednost	Stanje
80	Stopped
90	Check conditions
100	Starting
50	Running
25	Pausing delay
20	Pausing / Pausing sequence
130	Paused
35	Regular stopping delay
30	Regular stopping / Regular stopping sequence
10	Fast stopping
91	Check experiment conditions
110	Experiment starting
60	Experiment running

Tabela B.2 - Zaloga vrednosti spremenljivke *State*

Pri kompleksnejših postopkih oziroma postopkih, ki so odvisni od drugih postopkov se stanje *Pausing* oziroma *Regular stopping* razdeli na dve stanji - *delay* in *sequence*, pri čemer vrednost osnovnega stanja prevzame *sequence*. Tako je iz tabele razvidno, da pripadata vrednostim 20 in 30 spremenljivke *State* po dve stanji, vendar pa ti dve stanji v nobenem postopku nista istočasno definirani.

Ukazi postopkovnega vodenja

Ukazi so zbrani v tabeli B.3. Poda jih operater s pritiskom na gumb v oknu Factory Link-a, le-ta pa priredi spremenljivki *OC* vrednost, ki ustreza ukazu. Ko postopek detektira ukaz, spremenljivki *OC* priredi vrednost 0.

Ukaz	Vrednost	Pomen
Start	100	Zaženi postopek.
Stop	30	Ustavi izvajanje postopkovnega elementa in opremo postavi v varno stanje.
Abort	10	Brezpogojno prekini izvajanje postopkovnega elementa in opremo postavi v varno stanje.
Start Exp.	110	Zaženi postopek v eksperimentalnem načinu

Tabela B.3 - Ukazi postopkovnemu vodenju

Sporočila postopkovnega vodenja

Sporočila o delovanju postopka podaja spremenljivka *Message* (tabela B.4), sporočila o vzroku ustavitve pa spremenljivka *CauseOfStop* (tabela B.5). Tudi v ti spremenljivki se vpisujejo le vrednosti, ki pomenijo določeno sporočilo, v oknu Factory Link-a pa se te vrednosti pretvorijo v tekstovni prikaz sporočila.

Vrednost	Izpis v Factory Link-u
0	ni sporočila
1	Waiting for Start command
2	Setting initial conditions
3	Waiting for pressure to rise
4	Separator is operating
5	Stopping separator
6	Setting experiment initial conditions
7	Separator is in experimental mode

Tabela B.4 - Sporočila o stanju postopkovnega vodenja

Vrednost	Izpis v Factory Link-u
0	ni sporočila
1	Not enough water in the storage vessel
2	Too much water in the system
3	Maximum pressure exceeded
4	Stop command
5	Abort command
6	Water pump stoped during operation
7	Water pump stoped during operation
8	Storage vessel is full
9	Equipment not in Automatic mode

Tabela B.5 - Sporočila o vzroku ustavitve

B.3 Psevdo koda stanj in prehodov

Psevdo koda vsebuje specifikacije akcij posameznih stanj postopka in pogojev za prehode med stanji.

SC Stopped

Transition:
OC\SC=Start ⇒T:CheckConditions

SC CheckConditions

Action:
Volume:=ES\LT4201*PP\SC\R42Area+ES\LT4101*PP\SC\R41Area

Transitions:
ES\LT4201≥PP\SC\LT4201Min & Volume≤PP\SC\MaxVolume & ES\V4101\AM=A
&
ES\V4102\AM=A & ES\V4104\AM=A & ES\P4101\AM_mv=A & ES\V4101\AM_on=A &
ES\V4101\LR=R & ES\V4102\LR=R & ES\V4104\LR=R & ES\P4101\LR_mv=R &
ES\V4101\LR_on=R & ES\PC4104\AM=A & ES\PC4104\LR=R & ES\LC4104\AM=A
&
ES\LC4104\LR=R ⇒T: Starting
ES\LT4201<PP\SC\LT4201Min ∨ Volume>PP\SC\MaxVolume ∨ ES\V4101\AM=M
∨

ES\V4102\AM=M ∨ ES\V4104\AM=M ∨ ES\P4101\AM_mv=M ∨ ES\V4101\AM_on=M
∨
ES\V4101\LR=L ∨ ES\V4102\LR=L ∨ ES\V4104\LR=L ∨ ES\P4101\LR_mv=L ∨
ES\V4101\LR_on=L ∨ ES\PC4104\AM=M ∨ ES\PC4104\LR=L ∨ ES\LC4104\AM=M
∨
ES\LC4104\LR=L ⇒T: Stopped

SC Operation

Transitions:

AS: ES\PT4101>PP\SC\R41MaxPressure ∨ OC\SC=Abort ⇒ T: FastStopping

SC Starting

Actions:

EC\PC4101\TrValue:=0
EC\LC4101\TrValue:=4000
EC\PC4101\SPTr:=Tracking
EC\LC4101\SPTr:=Tracking
EC\V4104[©]
Δ(PP\SC\ValveSettingDelay)
EC\P4101\MV:=4000
EC\P4101[↑]
∇(ES\PT4101>PP\SC\R41OperPressure)
EC\PC4101\SPRemote:=PP\PC4101\SPRemote
EC\LC4101\SPRemote:=PP\LC4101\SPRemote
EC\PC4101\SPTr:=Control
EC\LC4101\SPTr:=Control
Δ(PP\SC\RegulationSettlingDelay)

SC Running

Actions:

EC\PC4101\SPRemote:=PP\PC4101\SPRemote
EC\LC4101\SPRemote:=PP\LC4101\SPRemote
PS\SC\Pressure\AL:=|ES\PT4101-ES\PC4101\SPActual|>PP\SC\MaxPressDev
PS\SC\Level\AL:=|ES\LT4101-ES\LC4101\SPActual|>PP\SC\MaxLevelDev
Volume:=ES\LT4201*PP\SC\R42Area+ES\LT4101*PP\SC\R41Area

Transition:

OC\SC=Stop∨(ES\FT4101\AI*10+ES\FT4102\AI)<PP\SC\R41FlowMin∨ES\LS420
1=on∨ ES\FT4102\AI<PP\SC\WaterFlowMin∨Volume≥PP\SC\MaxVolume ∨
ES\LT4201<PP\SC\LT4201Min ∨ ES\V4101\AM=M ∨
ES\V4102\AM=M ∨ ES\V4104\AM=M ∨ ES\P4101\AM_mv=M ∨ ES\V4101\AM_on=M
∨
ES\V4101\LR=L ∨ ES\V4102\LR=L ∨ ES\V4104\LR=L ∨ ES\P4101\LR_mv=L ∨
ES\V4101\LR_on=L ∨ ES\PC4104\AM=M ∨ ES\PC4104\LR=L ∨ ES\LC4104\AM=M
∨

ES\LC4104\LR=L ⇒T: RegularStopping

SC RegularStopping

Actions:

PS\SC\ExpEnable:=False

EC\LC4101\TrValue:=EC\V4102\A
EC\PC4101\TrValue:=4000
EC\LC4101\SPTr:=Tracking
EC\PC4101\SPTr:=Tracking
∇_L(ES\PT4101<PP\SC\R41StopPressure,PP\SC\StopMaxDuration)
EC\P4101[⇓]
Δ(PP\SC\PumpStoppingDelay)
EC\V4104[◉]
EC\LC4101\TrValue:=0

SC FastStopping

Actions:

PS\SC\ExpEnable:=False

EC\LC4101\TrValue:=EC\V4102\A
EC\PC4101\TrValue:=4000
EC\LC4101\SPTr:=Tracking
EC\PC4101\SPTr:=Tracking
EC\P4101[⇓]
EC\V4104[◉]
EC\LC4101\TrValue:=0

SC Experiment Starting

Actions:
EC\PC4101\TrValue:=4000
EC\LC4101\TrValue:=4000
EC\PC4101\SPTr:=Tracking
EC\LC4101\SPTr:=Tracking

EC\P4101\MV:=4000
EC\P4101[⇑]
EC\V4104[◎]
Δ(PP\SC\ExperimentSettingDelay)

EC\V4101\EXP :=4000
EC\V4102\EXP :=4000
EC\P4101\EXP :=0

SC Experiment Running

Actions:

PS\SC\ExpEnable:=True

ES\PT4101\EXP:=ES\PT4101
ES\LT4101\EXP:=ES\LT4101
ES\LT4102\EXP:=ES\LT4102
ES\FT4101\EXP:=ES\FT4101\AI
ES\FT4102\EXP:=ES\FT4102\AI
ES\LS4201\EXP:=ES\LS4201
ES\V4104\EXP:=ES\V4104\EC

EC\P4101\A_act:= EC\P4101\EXP
EC\V4101\A_act:= EC\V4101\EXP
EC\V4102\A_act:= EC\V4101\EXP

Transition:

OC\SC=Stop ∨ ES\LS4201=on ∨ Volume≥PP\SC\MaxVolume ∨
ES\LT4201<PP\SC\LT4201Min ∨ ES\V4101\AM=M ∨
ES\V4102\AM=M ∨ ES\V4104\AM=M ∨ ES\P4101\AM_mv=M ∨ ES\V4101\AM_on=M
∨
ES\V4101\LR=L ∨ ES\V4102\LR=L ∨ ES\V4104\LR=L ∨ ES\P4101\LR_mv=L ∨
ES\V4101\LR_on=L ∨ ES\PC4104\AM=M ∨ ES\PC4104\LR=L ∨ ES\LC4104\AM=M
∨
ES\LC4104\LR=L ⇒T: RegularStopping

Dodatek C

Izvedba vmesnika za eksperimentiranje iz okolja Matlab/Simulink

C.1 DDE komunikacijski standard

DDE je komunikacijski protokol tipa strežnik/odjemalec za medprocesno komunikacijo (ang. *interprocess communication - IPC*) v okolju Windows. Služi izmenjavi podatkov med različnimi aplikacijami prek deljenega pomnilnika. DDE strežniška aplikacija od DDE odjemalskih aplikacij sprejema zahteve in jim posreduje odgovore. Aplikacije v okolju Windows, ki podpirajo DDE komunikacijo lahko nastopajo kot DDE odjemalci (Matlab), DDE strežniki (MelDDE) ali pa hkrati kot strežniki in odjemalci (MS Excel, Word, Access, itd). Komponente in knjižnice za enostavno implementacijo DDE komunikacij imajo danes tudi vsa orodja za hiter razvoj aplikacij v okolju Windows (VB, Delphi, Visual C++, C++ Builder itd).

DDE standard je bil eden prvih standardov v Windows okolju za medprocesno komunikacijo, njegovi začetki segajo v začetek 90-ih. Kot tak je bil hitro sprejet tudi v avtomatizaciji, kjer je obstajala in še vedno obstaja velika potreba po standardni, od strojne in programske opreme čimbolj neodvisni povezljivosti različnih aplikacij. Pojavile so se celo različne izboljšave osnovnega DDE protokola, prikrojene specifičnim zahtevam v avtomatizaciji (kratki odzivni časi, zmožnost dovolj velike hitrosti komunikacij za delovanja v realnem času, itd):

- FastDDE (Wonderware) omogoča paketni prenos podatkov in s tem večjo hitrost prenosa (do 7000 podatkov/s), razvit pa je bil za komunikacijo Wonderwarove InTouch Scade s krmilniki in vhodno/izhodnimi moduli.

- NetDDE (Wonderware) je razširitev DDE protokola za medprocesno komunikacijo od enega računalnika na celotno računalniško mrežo. Protokol NetDDE je neodvisen od mrežne platforme, podpira različne mrežne protokole (TCP/IP, Arcnet, DecNET, itd), tako je možna povezava med aplikacijami, ki delujejo v različnih operacijskih sistemih (Windows, Linux itd). Licence za

NetDDE protokol je kasneje odkupil Microsoft, danes je integralni del operacijskih sistemov Windows.

- AdvenceDDE (Rockwell Software).

C.2 DDE komunikacija v Matlabu

Matlab v DDE komunikaciji nastopa kot odjemalec. Odjemalec je aplikacija, ki sproži zahtevo za vzpostavitev DDE komunikacijskega kanala, zahteva podatke od DDE strežniške aplikacije in tej tudi pošilja ukaze za nove vrednosti spremenljivk.

C.2.1 Vzpostavitev DDE komunikacije

Prva zahteva, ki jo odjemalec postavi strežniku, je po vzpostavitvi DDE komunikacijskega kanala in vsebuje:

- ime DDE strežniške aplikacije (ang. *Application name*), s katero vzpostavljamo DDE kanal,
- predmet komunikacije (ang. *Topic name*) v DDE strežniški aplikaciji.

V primeru zahteve za povezavo iz Matlaba, v katerem v zahtevi za odprtje komunikacijskega kanala podamo dva parametra, je oblika ukaza:

```
channel = DDEINIT ('application name','topic name').
```

Funkcija ob uspešni vzpostavitvi DDE komunikacije vrne ročico (ang. *handle*) kanala, ki je večja od 0, uporabljamo pa jo pri drugih DDE funkcijah.

Konkretno za MelDDE strežnik ima ukaz obliko:

```
channel = DDEINIT('meldde','konfiguracijska_datoteka').
```

S prvim parametrom povemo ime aplikacije, z drugim pa ime konfiguracijske datoteke (ustvarimo jo z CfgDDE modulom MelDDE paketa), v kateri so nastavitve za komunikacijo MelDDE strežnika z Mitsubishi krmilnikom (tip Cpu-ja, način komunikacije, hitrost komunikacije itd).

Opomba: Pred vzpostavljanjem DDE komunikacije iz Matlaba mora biti aplikacija, s katero vzpostavljamo komunikacijo že pognana, drugače Matlab kot rezultat ukaza DDEINIT vrne vrednost 0, ki indicira neuspešno odprt komunikacijski kanal. Načelno bi torej iz Matlaba najprej pognali MelDDE aplikacijo z ukazom:

```
!meldde ,
```

nato pa bi šele uporabili ukaz `DDEINIT`. Vendar ob zagonu MelDDE strežnika iz Matlaba MelDDE aplikacija čaka na ukaz in Matlabu ne vrne povratne informacije, da se je res že zagnala, sprejemanje ukazov prek komandne vrstice (ali prek tekstovne datoteke) je ustavljeno, dokler MelDDE strežnika ne zapremo. Prva rešitev problema je, da MelDDE aplikacijo poženemo ročno izven Matlaba, druga pa, da iz Matlaba poženemo program, ki sam zažene MelDDE strežnik. Tako je bil v Visual Basicu napisan program DDE_EXP.EXE, ki se po zagonu Meldde aplikacije zapre. Tako je pravo zaporedje ukazov pri odpiranju DDE komunikacije naslednje:

```
!DDE_EXP ,
channel = DDEINIT('meldde','konfiguracijska_datoteka').
```

DDE komunikacijski kanal zapremo z ukazom `OK = DDETERM(channel),` kjer s parametrom `channel` podamo ročico DDE komunikacijskega kanala, ki ga zapiramo. Funkcija v primeru uspešnega zaprtja kanala vrne vrednost 1, ob napaki pa vrednost 0.

C.2.2 DDE branje v Matlabu

Branje podatkov iz DDE strežniške aplikacije v Matlab najbolj enostavno realizirano z ukazom `DDEREQ`:

```
data = DDEREQ(channel,item,format,timeout),
```

kjer s parametri določimo:

- S parametrom `channel` podamo ročico DDE komunikacijskega kanala, prek katerega naj se izvrši branje spremenljivke DDE strežnika.

- S parametrom `item` podamo točko komunikacije s strežnikom - definiramo naslov, tip in velikost (skalar, vektor) spremenljivke, katere vrednost želimo zajeti.

- Parameter `format` je opcijski, in je dvodimenzionalno polje, s katerim določimo format branih spremenljivk. Prvi element polja pove, v katerem Windows Clipboard formatu naj se zajame spremenljivka (trenutno je mogoč samo format CF_TEXT, vrednost elementa je 1) . Drugi element polja pa pove, ali je spremenljivka numerična (privzeto, vrednost elementa je 0) ali tekstovna (vrednost elementa je 1). Prvizeta vrednost parametra format je [1 0].

- Parameter `timeout`, ki je opcijski, določa največji čas čakanja v ms na sprejem podatkov po izdanem ukazu. Privzeta vrednost parametra je 3 s.

Funkcija vrne prejeto vrednost v spremenljivki '`data`'. V primeru neuspešnega branja dobimo prazno spremenljivko.

Primer 1:

DDE kanal z MelDDE strežnikom je odprt, njegova ročica je v spremenljivki `channel1`, zanima nas vrednost registra D100 v pomnilniku Mitsubishi krmilnika, s katerim je povezan MelDDE strežnik, zapišemo jo v Matlabovo spremenljivko D100. Ustrezen ukaz se glasi:

```
D100 = ddereq(channel1, 'D100').
```

Primer 2:

DDE kanal z MelDDE strežnikom je odprt, njegova ročica je v spremenljivki `channel1`, zanima nas vrednost polja treh zaporednih števil formata *long integer*, ki se začne na naslovu D100 (D100 do D105) v pomnilniku Mitsubishi krmilnika, s

katerim je povezan MelDDE strežnik, zapišemo ga v Matlabovo spremenljivko D100. Ustrezen ukaz se glasi:

```
D100 = ddereq(channel1, 'D100,3*1,ld').
```

Druga možnost za branje podatkov je prek tople (warm link) ali vroče (hot link) DDE povezave s strežniško aplikacijo.

O topli povezavi za posamezno spremenljivko govorimo takrat, ko nas DDE strežniška aplikacija prek DDE kanala obvesti, da se je vrednost spremenljivke, za katero topla povezava obstaja, spremenila, ne dobimo pa nove vrednosti spremenljivke. O vroči povezavi pa govorimo takrat, ko ob spremembi vrednosti spremenljivke avtomatsko dobimo tudi njeno novo vrednost. Toplo (in vročo) povezavo vzpostavimo z ukazom:

```
OK = DDEADV(channel,item,callback,upmtx,format,timeout),
```

kjer s parametri ukaza `DDEADV` določimo:

- S parametrom `channel` podamo ročico DDE komunikacijskega kanala, prek katerega želimo vzpostaviti toplo ali vročo povezavo.

- S parametrom `item` podamo točko komunikacije s strežnikom - definiramo naslov, tip in velikost (skalar, vektor) spremenljivke, za katero vzpostavljamo toplo (vročo) povezavo.

- S parametrom `callback` določimo zaporedje ukazov, ki jih Matlab izvrši z ukazom `eval`, ko je obveščen o spremembi vrednosti spremenljivke, za katero ima vzpostavljeno toplo (vročo) povezavo z DDE strežniško aplikacijo.

- Parameter `upmtx` je opcijski, če ga ne določimo, imamo toplo, drugače pa vročo povezavo, saj z njim definiramo, v katero Matlabovo spremenljivko se zapiše nova vrednost v DDE strežniški aplikaciji spremenjene spremenljivke, določene s parametrom `item`.

- Parameter `format` je opcijski, in je dvodimenzionalno polje, s katerim določimo format branih spremenljivk. Prvi element polja pove, v katerem Windows

Clipboard formatu naj se zajame spremenljivka (trenutno je mogoč samo format CF_TEXT, vrednost elementa je 1) . Drugi element polja pa pove, ali je spremenljivka numerična (privzeto, vrednost elementa je 0) ali tekstovna (vrednost elementa je 1). Privzeta vrednost parametra format je [1 0].

- Parameter `timeout`, ki je opcijski, določa maksimalen čas čakanja v ms na potrditev sprejema podatkov od DDE strežniške aplikacije. Privzeta vrednost parametra je 3 s.

Funkcija v primeru uspešnega posredovanja ukaza vrne vrednost 1, ob napaki pa vrednost 0.

Primer 3:

DDE kanal z MelDDE strežnikom je odprt, njegova ročica je v spremenljivki `channel1`, vzpostavimo vročo povezavo za register D200 v pomnilniku Mitsubishi krmilnika, s katerim je povezan MelDDE strežnik. Ob vsaki spremembi vrednosti tega registra novo vrednost zapišemo v Matlabovo spremenljivko `hotlink`, in jo tudi izpišemo. Ukaz ima obliko:

```
OK = ddeadv(channel1, 'D200','disp(hotlink)', 'hotlink').
```

Toplo ali vročo povezavo z DDE strežniško aplikacijo zapremo z ukazom

```
OK = DDEUNADV(channel,item,format,timeout),
```

kjer s parametri ukaza `DDEUNADV` določimo:

- S parametrom `channel` podamo ročico DDE komunikacijskega kanala, prek katerega imamo vzpostavljeno toplo ali vročo povezavo, ki ji zapiramo.

- S parametrom `item` podamo točko komunikacije s strežnikom - definiramo naslov, tip in velikost (skalar, vektor) spremenljivke, za katero zapiramo toplo (vročo) povezavo. Parameter mora biti enak, kot je bil pri odpiranju tople (vroče) povezave z ukazom `DDEADV`.

- Parameter `format` je opcijski, in je dvodimenzionalno polje, s katerim določimo format branih spremenljivk. Prvi element polja pove, v katerem Windows Clipboard formatu naj se zajame spremenljivka (trenutno je mogoč samo format CF_TEXT, vrednost elementa je 1) . Drugi element polja pa pove, ali je spremenljivka numerična (privzeto, vrednost elementa je 0) ali tekstovna (vrednost elementa je 1). Prvizeta vrednost parametra format je [1 0]. Parameter mora biti enak, kot je bil pri odpiranju tople (vroče) povezave z ukazom `DDEADV`.

- Parameter `timeout`, ki je opcijski, določa maksimalen čas čakanja v ms na potrditev sprejema podatkov od DDE strežniške aplikacije. Privzeta vrednost parametra je 3 s.

Funkcija v primeru uspešnega posredovanja podatkov vrne vrednost 1, ob napaki pa vrednost 0.

Primer 4:

Prekiniti želimo toplo (vročo) povezavo za register D200 v pomnilniku Mitsubishi krmilnika, s katerim je povezan MelDDE strežnik, vzpostavljeno prek DDE kanala z ročico v spremenljivki `channel1`. Ukaz ima obliko:

```
OK = ddeunadv(channel1, 'D200').
```

C.2.3 DDE pisanje iz Matlaba

Pisanje podatkov v DDE strežniško oplikacijo iz Matlaba realiziramo z ukazom `DDEPOKE`:

```
OK = DDEPOKE(channel,item,data,format,timeout),
```

kjer s parametri ukaza `DDEPOKE` določimo:

- S parametrom `channel` podamo ročico DDE komunikacijskega kanala, prek katerega naj se do DDE strežniške aplikacije pošlje vrednost spremenljivke.

- S parametrom `item` podamo točko komunikacije s strežnikom - definiramo naslov, tip in velikost (skalar, vektor) spremenljivke, ki jo pošiljamo.

- S parametrom `data` podamo vrednost(i), ki jih želimo poslati.

- Parameter `format` je opcijski, in pove, v katerem Windows Clipboard formatu naj se spremenljivka pošlje (trenutno je mogoč samo format CF_TEXT, vrednost parametra je 1) .

- Parameter `timeout`, ki je opcijski, določa maksimalen čas čakanja v ms na potrditev sprejema podatkov od DDE strežniške aplikacije. Privzeta vrednost parametra je 3 s.

Funkcija v primeru uspešnega posredovanja podatkov vrne vrednost 1, ob napaki pa vrednost 0.

Primer 5:

DDE kanal z MelDDE strežnikom je odprt, njegova ročica je v spremenljivki `channel1`, v registre D200 do D203 v pomnilniku Mitsubishi krmilnika, s katerim je povezan MelDDE strežnik, želimo zapisati vrednost spremenljivke `referenca`, v obliki 32-bitnega *integer* števila (oznaka *ld - long decimal*). Ker so vse spremenljivke v Matlabu tipa *float*, je potrebna pretvorba v ustrezno obliko. Ustrezen ukaz se glasi:

```
OK = ddepoke(channel1, 'D200,ld',round(referenca)).
```

Primer 6:

DDE kanal z MelDDE strežnikom je odprt, njegova ročica je v spremenljivki `channel1`, v registre D200 do D205 v pomnilniku Mitsubishi krmilnika, s katerim je povezan MelDDE strežnik, želimo zapisati vrednost vektorja `referenca`, v obliki treh (32-bitih) realnih števil (oznaka *f - float*).

Ustrezen ukaz se glasi:

```
OK = ddepoke(channel1, 'D200,3*1,f',referenca).
```

Komentar: Pri pošiljanju (in pretvarjanju formatov) spremenljivk iz Matlaba v Mitsubishijeve krmilnike je potrebno upoštevati:

- Pri pošiljanju spremeljivk tipa *float*, (običajna oblika v Matlabu) na naslove krmilnika, kjer so definirane spremenljivke tipa *float*, ni potrebna pretvorba, seveda, če ima krmilnik aritmetiko s plavajočo vejico (serija AxSH je nima, QxAS jo ima).

  Primer 7: `OK = ddepoke(channel1, 'D200,f',referenca)`

- Pri pošiljanju spremenljivk iz Matlaba (ki so v *float* obliki) na naslove krmilnika, kjer so definirane spremenljivke tipa *integer*, je potrebna pretvorba, za kar so v Matlabu ustrezni naslednji ukazi:

  - Z ukazom `I = int16(X)` pretvorimo *float* spremenljivko X v predznačeno 16 bitno celoštevilsko spremenljivko. Pretvorba brez zaokroževanja vzame celoštevilski del vrednosti spremenljivke. Obseg pretvorbe je od –32768 do 32767. Spremenljivke izven tega območja se pretvorijo v najbližjo mejno vrednost. Spremenljivke v Mitsubishijevih krmilnikih tipa INT (16 bitov) imajo enak obseg.

    Primer 8: `OK = ddepoke(channel1, 'D200',int16(referenca))`

  - Z ukazom `I = int32(X)` pretvorimo *float* spremenljivko X v predznačeno 32 bitno celoštevilsko spremenljivko. Pretvorba brez zaokroževanja vzame celoštevilski del vrednosti spremenljivke. Obseg pretvorbe je od-2147483648 do 2147483647. Spremenljivke izven tega območja se pretvorijo v najbližjo mejno vrednost. Spremenljivke v Mitsubishijevih krmilnikih tipa DINT (32 bitov) imajo sicer enak obseg, vendar iz neznanega razloga MelDDE strežnik javi napako, ko pošiljamo spremenljivko v obliki, dobljeni z ukazom INT32. Drugi način je z ukazom `round`.

  - Z ukazom `I = round(X)` zaokrožimo *float* spremenljivko X v najbližjo celoštevilsko spremenljivko. V Matlabu ni dokumentacije, v kakšno obliko se spremenljivka pretvori, zanimivo pa je, da lahko spremenljivke, pretvorjene z ukazom `round` pravilno pišemo na naslove spremenljivk v Mitsubishijevih krmilnikih tipa INT in DINT, le paziti moramo, da ne prekoračimo njihovega

obsega (realne spremenljivke v Matlabu imajo obseg od velikostnega razreda 10^{-308} do 10^{+30}), sicer MelDDE strežnik javi napako.

Primer 9:

Primer za INT spremenljivko v pomnilniku Mitsubishi krmilnika:

```
OK = ddepoke(channel1, 'D200',round(referenca))
```

Primer 10:

Primer za DINT spremenljivko v pomnilniku Mitsubishi krmilnika:

```
OK = ddepoke(channel1, 'D300,1*1,ld',round(referenca))
```

- Z ukazoma `I = UINT16(X)` ali `I = UINT32(X)` pretvorimo *float* spremenljivko X v nepredznačeno 16 ali 32 bitno celoštevilsko spremenljivko. Pretvorba brez zaokroževanja vzame celoštevilski del vrednosti spremenljivke. Obseg pretvorbe je od 0 do 65535 (0 do 4.294.967.295). Spremenljivke izven tega območja se pretvorijo v najbližjo mejno vrednost. Spremenljivke v Mitsubishijevih krmilnikih tipa WORD (16 bitov) in DWORD (32 bitov) imajo sicer enak obseg, vendar MELDDE strežnik strežnik iz neznanega vzroka javi napako, ko pošiljamo spremenljivke v obliki, dobljeni z omenjenima ukazoma. Druga možnost je z ukazom `int16` ali `round`, vendar moramo biti pozorni na predznak spremenljivk, ki jih pretvarjamo. Če po pretvorbi dobimo negativne vrednosti, MelDDE sicer ne javi napake, saj se spremenljivke tipa WORD in DWORD tretirajo enostavno kot zaporedje bitov, krmilnik pa bo dobljeno vrednost obravnaval napačno kot nepredznačeno število.

Najenostavnejše je torej pretvarjanje z ukazom `round`, paziti moramo le, da spremenljivka ni izven obsega tipa, v katerega jo pretvarjamo.

- Pri pošiljanju spremenljivk na bitne naslove v Mitsubishijevem krmilniku mora imeti spremenljivka v Matlabu vrednost 0 ali 1 (ni potrebno pretvarjanje), sicer MelDDE strežnik javi napako.

C.2.4 Pošiljanje ukazov DDE strežniški aplikaciji iz Matlaba

Iz okolja Matlab je možno tudi pošiljanje posebnih ukazov DDE strežniški aplikaciji v obliki:

```
OK = DDEEXEC(channel,command,item,timeout),
```

kjer s parametri ukaza `DDEEXEC` določimo:

- S parametrom `channel` podamo ročico DDE komunikacijskega kanala, prek katerega naj se izvrši branje spremenljivke DDE strežnika.

- V parametru `command` podamo ukaz, ki naj ga DDE strežniška aplikacija izvrši.

- S parametrom `item`, ki je opcijski, podamo točko komunikacije s strežnikom - definiramo naslov, tip in velikost (skalar, vektor) spremenljivke, na katero se nanaša ukaz, posredovan v parametru `command`.

- Parameter `timeout`, ki je opcijski, določa maksimalen čas čakanja v ms na sprejem podatkov po izdanem ukazu. Privzeta vrednost parametra je 3 s.

Funkcija v primeru uspešnega posredovanja ukaza vrne vrednost 1, ob napaki pa vrednost 0.

Primer 11:

DDE kanal z MelDDE strežnikom je odprt, njegova ročica je v spremenljivki `channel1`, pošljemo mu ukaz, ki njegovo okno postavi v ospredje (seznam vseh ukazov za MelDDE aplikacijo je v [4]). Ustrezen ukaz se glasi:

```
OK = DDEEXEC(channel1,'show').
```

C.2.5 DDE komunikacija med aplikacijo Matlab in Mitsubishi PLK-ji

Komunikacija med okoljem Matlab in Mitsubishi PLK-ji poteka prek DDE strežnika MelDDE, ki služi kot komunikacijski vmesnik [4]. V procesnem laboratoriju odseka E2 se nahajata dva krmilnika:

- Mitsubishi A2SH-S1 za osnovno vodenje (brez aritmetike z realnimi števili).
- Mitsubishi Q2AS-S1 za postopkovno vodenje (z aritmetiko z realnimi števili).

Krmilnika sta prek komunikacijskih modulov A1SJ71AT21B povezana v Mitsubishijevo mrežo Melsecnet/B. Krmilnik A2SH-S1 ima Ethernet komunikacijski modul A1SJ71E71-B2-S3, krmilnik Q2AS-S1 pa Ethernet A1SJ71QE71-B2 komunikacijski modul. Modula omogočata istočasno komunikacijo prek 8-ih kanalov po TCP/IP ali UDP protokolu. Trenutno je prvi kanal vsakega izmed modulov uporabljen za komunikacijo s Factory Link nadzorno aplikacijo laboratorija.

DDE komunikacija s krmilnikom A2SH-S1 je možna ali prek serijskega RS232 kanala na CPU modulu ali prek Ethernet modula (na enem ali več kanalih istočasno). DDE komunikacija s krmilnikom Q2AS-S1 pa je možna prek Ethernet komunikacijskega modula. Tako je na krmilniku A2SH-S1 možna istočasna DDE komunikacija prek 9-ih, na Q2AS-S1 krmilniku pa prek 8-ih DDE kanalih.

Komunikacijske parametre določimo z orodjem CfgDDE paketa MelDDE. Za serijsko komunikacijo ni potrebno pisati dodatne logike na PLC-ju, saj prek serijskega kanala CPU modula dostopamo direktno do pomnilnika krmilnika. Za komunikacijo prek Ethernet modula posameznega krmilnika pa je potreben poseben program na PLC-ju, ki skrbi za povezavo med Ethernet modulom in glavnim pomnilnikom krmilnika.

Glede hitrosti prenašanja podatkov se je pri testiranju za hitrejšo izkazala komunikacija prek serijskega vhoda krmilnika, prek katerega je možna komunikacija s hitrostjo 9600 bit/s. Kljub več kot 1000-krat hitrejši komunikaciji (10Mbit/s) prek etherneta, je pri testiranju dejanska hitrost prenašanja podatkov zaostajala v primerjavi s komunikacijo prek serijskega vhoda. Čas branja ali pisanja spremenljivke prek serijskega kanala je velikostnega reda 50 ms, prek etherneta pa je večji za faktor 2. Glavni vzrok temu na prvi pogled nelogičnemu rezultatu je

enostaven: S serijsko komunikacijo, ki v prvi vrsti služi programiranju krmilnika, dostopamo direktno do pomnilniškega prostora krmilnika, medtem ko pa je za komunikacijo prek Ethernet modula potrebna posebna programska logika v krmilniku, ki se, tako kot ostala programska koda, izvršuje ciklično (minimalni cikel krmilnika je 10 ms, dejanski 80 ms). Sama hitrost prenašanja podatkov je tako omejena z dolžino cikla krmilnika, ki je glede na velikost komunikacijskih paketov in hitrost komunikacijskega medija izrazito prevelika. Ethernet komunikacija bi se izkazala za učinkovitejšo šele ob morebitni potrebi za prenašanje velikega števila podatkov, kar pa za namen eksperimentiranja iz okolja Matlab ni potrebno.

C.3 Gradniki za vodenje v realnem času v okolju Matlab/Simulink

Vodenje procesov iz okolja Matlab/Simulink z DDE komunikacijo z Mitsubishijevim krmilnikom je realizirano s tremi tekstovnimi datotekami Matlabovih ukazov in s tremi gradniki Simulink simulacijske sheme.

Tekstovne datoteke Matlabovih ukazov služijo za:

- Datoteka za inicializacijo DDE komunikacije služi za vzpostavitev DDE komunikacijskega kanala med Matlab-om in MelDDE strežnikom in definicijo spremenljivk, ki se uporabljajo v Simulink shemi. Spremenljivke Simulink simulacijske sheme, ki jih želimo brati ali spreminjati iz Matlaba v realnem času med samo simulacijo, moramo definirati kot globalne. Ob začetku simulacije se namreč iz Matlabovega pomnilniškega prostora (ang. *workspace*) preberejo spremenljivke, ki jih potrebujemo za simulacijo. Če so spremenljivke lokalne, se celotna simulacija izvede z vrednostmi spremenljivk, ki so jih te imele ob štartu. Če pa so spremenljivke definirane kot globalne, se vsak simulacijski korak njihove vrednosti na novo preberejo. Podobno je s spremenljivkami, v katere se zapisujejo rezultati simulacije. Če te niso definirane kot globalne, dobimo v Matlabu rezultat šele po koncu simulacije, drugače pa po vsakem simulacijskem koraku.

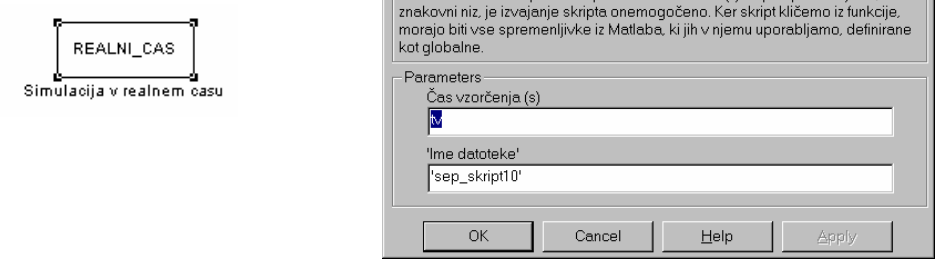
- Datoteka za DDE komunikacijo med simulacijo vsebuje DDE ukaze za branje in pisanje spremenljivk, potrebnih za vodenje, ter njihove morebitne obdelave,

pretvorbe, skaliranja, normiranja, ipd. To datoteko ob konstantnih časovnih intervalih iz okolja Simulink poganja poseben blok (S funkcija), ki služi tudi za to, da se simulacija izvaja v realnem času. Ker datoteko kličemo iz bloka v Simulinku, morajo biti, iz istega razloga kot pri inicializacijski datoteki, vse spremenljivke, ki jih v njej uporabljamo, definirane kot globalne.

- Datoteka za končanje DDE komunikacije po zaključeni simulaciji zapre odprt DDE komunikacijski kanal.

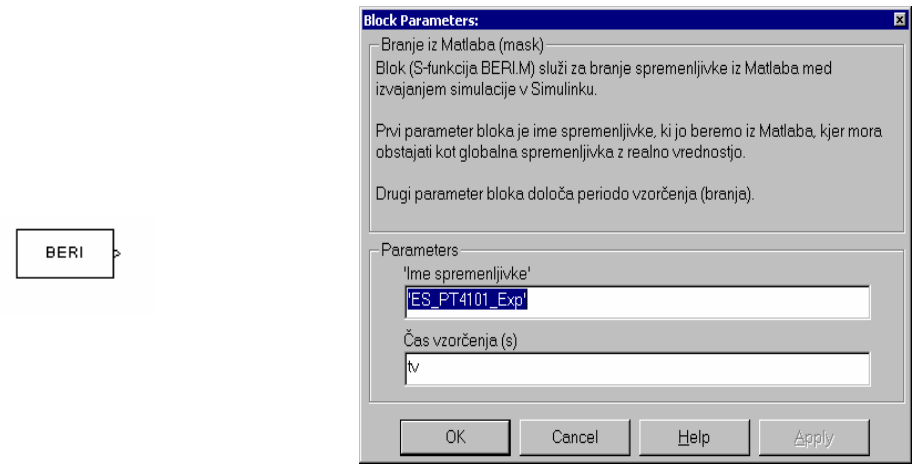
V Simulink simulacijski shemi se (poleg standardnih blokov) za vodenje v realnem času uporabljajo trije bloki (S-funkcije):

- Blok "REALNI ČAS" je maskirana S-funkcija REALNI_CAS.M, ki omogoča simulacijo v realnem času in poganjanje Matlab datoteke za DDE komunikacijo med simulacijo ob konstantnih časovnih intervalih. S-funkcija brez vhodov in izhodov ima samo eno diskretno stanje, katerega vrednost se ob začetku simulacije inicializira na vrednost interne ure računalnika. Ob vsakem simulacijskem koraku, ko se kličejo vsi bloki (S-funkcije), se diskretno stanje, povečano za dolžino časovnega intervala *tv*, ki določa periodo časovno diskretnega vodenja, primerja s trenutno vrednostjo ure računalnika. Izvajanje simulacije se zaradi prazne *while* zanke v S-funkciji zaustavi toliko časa, dokler ura računalnika ni enaka diskretnemu stanju, povečanem za interval *tv*, kar pa je tudi nova vrednost stanja, ki ga S-funkcija vrne. Na ta način se simulacijo v Simulinku prisili, da teče v realnem času. Če pa je primerjana vrednost ure večja od diskretnega stanja, povečanega za *tv*, se izpiše opozorilo, da je pri simulaciji v realnem času prišlo do zamude, funkcija takoj vrne novo vrednost stanja, ki je dobljeno stanje, povečano za interval *tv* (slika C.3).



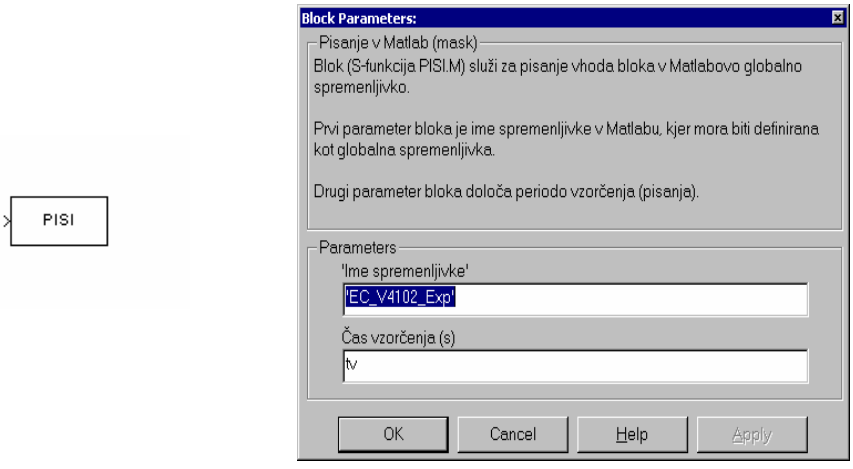
Slika C.3: Levo: Simulink blok, desno: uporabniški vmesnik funkcije REALNI_CAS

- Blok "BERI" je maskirana S-funkcija BERI.M, ki omogoča branje globalnih spremenljivk iz Matlaba med simulacijo. Funkcija ima samo en izhod, katerega vrednost je med simulacijo enaka vrednosti globalne spremenljivke v Matlabu, ki jo podamo kot parameter, spremenljivka pa se potem kot globalna definira tudi v S-funkciji (slika C.4). V bloku lahko nastopa poljubna spremenljivka, ki je kot globalna definirana v Matlabu, blok pa je namenjen branju spremenljivk iz Matlabovega pomnilniškega prostora, katerih vrednost prek DDE komunikacije dobimo iz pomnilnika Mitsubishi krmilnika.



Slika C.4: Levo: Simulink blok, desno: uporabniški vmesnik funkcije BERI

- Blok "PISI" je maskirana S-funkcija PISI.M, ki omogoča pisanje rezultatov simulacije v globalne spremenljivke v Matlabu med simulacijo. Funkcija ima samo en vhod, katerega vrednost se med simulacijo piše v globalno spremenljivko v Matlabu, ki jo podamo kot parameter bloka (slika C.5). Spremenljivka se kot globalna definira tudi v S-funkciji. V bloku lahko nastopa poljubna spremenljivka, ki je kot globalna definirana v Matlabu, blok pa je namenjen pisanju spremenljivk v Matlabov pomnilniški prostor, katerih vrednost prek DDE komunikacije naprej pošiljamo v pomnilnik Mitsubishi krmilnika.



Slika C.5: Levo: Simulink blok, desno: uporabniški vmesnik funkcije PISI

C.4 Dopolnitev aplikacije vodenja Mitsubishi PLK-jev

Na podlagi specifikacij podanih v obliki diagrama prehajanja stanj PSTD in psevdo kode stanj in prehodov, je bila dopolnjena obstoječa aplikacija vodenja Mitsubishi PLK-jev.

Vodenje procesov iz okolja Matlab/Simulink prek DDE komunikacije z Mitsubishi PLK-jem je bilo realizirano s serijsko komunikacijo prek serijskega vhoda krmilnika Mitsubishi A2SH-S1, ki služi za osnovno vodenje. Zaradi varnosti se ukazi za stanje naprav, ki jih dobivamo iz Matlaba, dejansko prepišejo na izhode krmilnika le v stanju *Experiment Running*. V tem stanju postopek tudi preverja stanja naprav in sistema ter ob morebitnih prekoračitvah za ljudi ali naprave varnih mej procesnih veličin preide v stanje *Regular Stopping* ali *Fast Stopping* in varno ustavi podproces ter onemogoči dejansko vodenje naprav iz Matlaba. Ker je postopkovno vodenje realizirano na krmilniku Q2AS-S1, DDE komunikacija pa se vzpostavi s krmilnikom A2SH-S1, je bila definirana bitna mrežna spremenljivka PS_SC_ExpEnable (naslov B1FF), ki z logično vrednostjo 1 ta krmilnik obvesti, da se nahajamo v stanju *Experiment Running*. V tem stanju se na izhode krmilnika prepišejo vrednosti spremenljivk, ki jih prek DDE komunikacije določamo iz Matlaba. V nasprotnem primeru pa izhode krmilnika lahko določa postopkovno vodenje (direktno ali prek PI regulatorja), PI regulator ali operater prek Factory Link nadzornega sistema. Tako je bilo celotno stanje *Experiment Running* realizirano na obeh krmilnikih.

Zaradi hitrejše (vektorske) DDE komunikacije so naslovi spremenljivk Mitsubishi krmilnika A2SH-S1, v katere se prepisujejo signali senzorjev in naslovi spremenljivk, v katere se iz Matlaba sporočajo ukazi za naprave podprocesa, zaporedni (tabela C.6, tabela C.7).

Ker ima krmilnik A2SH-S1 samo celoštevilsko aritmetiko, se analogne vrednosti signalov po A/D pretvorbi pretvorijo v celoštevilsko območje (normalno od 0 do 4000). Ko v Matlabu prek DDE komunikacije preberemo celoštevilske vrednosti, dobimo po množenju z vrednostjo pripadajočih pretvorbenih faktorjev vrednost signalov senzorjev v podanih enotah (tabela C.6).

Ime Spremenljivke	Naslov spremenljivke	Tip spremenljivke	Pretvorbeni faktor	Enota spremenljivke po pretvorbi
PS_SC_Enable_Exp	D500	Integer	/	[0/1]
ES_PT4101_Exp	D501	Integer	0.00025	[bar]
ES_LT4101_Exp	D502	Integer	0.0005	[m]
ES_LT4102_Exp	D503	Integer	0.00025	[m]
ES_FT4101_Exp	D504	Integer	0.005	[l/s]
ES_FT4102_Exp	D505	Integer	0.000125	[l/s]
ES_LS4201_Exp	D506	Integer	/	[0/1]
ES_V4104_Exp	D507	Integer	/	[0/1]

Tabela C.6 - Naslovi spremenljivk signalov senzorjev in stanj namenjenih za posredovanje v okolje Matlab

Ime spremenljivke	Naslov spremenljivke	Tip spremenljivke	Območje spremenljivke (0-100%)
EC_V4101_Exp	D508	Integer	0-1000
EC_V4102_Exp	D509	Integer	0-1000
EC_P4101_Exp	D510	Integer	0-4000

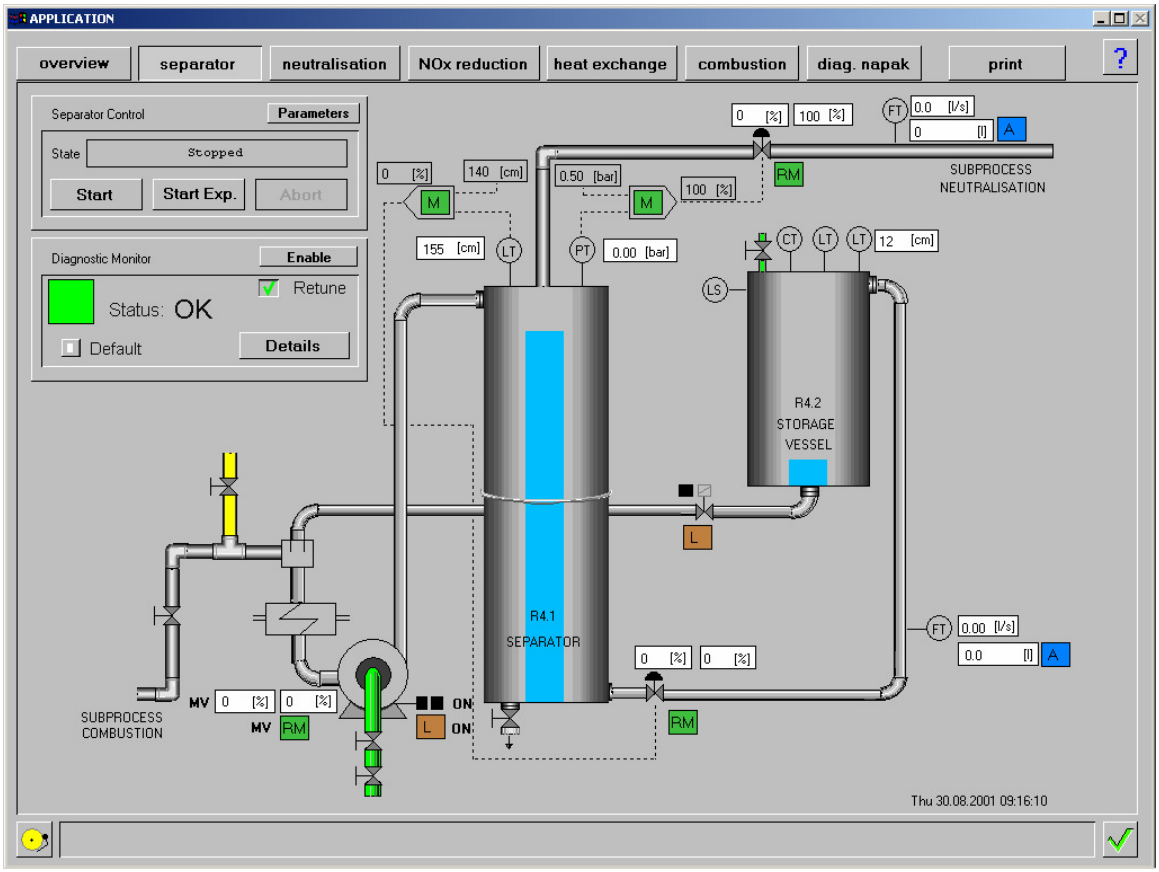
Tabela C.7 - Naslovi spremenljivk, namenjenih sprejemanju ukazov za naprave iz Matlaba

Pri pošiljanju ukazov iz Matlaba (in pri samem vodenju) moramo biti pozorni na celoštevilsko območje v Mitsubishi krmilniku, ki (prek D/A pretvorbe in posredovanju signala aktuatorju) ustreza območju vplivne veličine v procesu.

Podrobnosti o programski aplikaciji vodenja procesnega laboratorija so opisane v [28], [5], [18] in [19].

C.5 Dopolnitev aplikacije nadzornega vodenja

V aplikaciji nadzornega sistema vodenja Factory Link je bil, glede na nabor vrednosti spremenljivk postopkovnega vodenja, podanih v specifikacijah, grafični prikaz tehnološke sheme podprocesa priprave plina (slika C.6.) dopolnjen z ukazom za zagon postopka v eksperimentalnem načinu in z dodatnimi sporočili o trenutnem stanju, delovanju in vzroku ustavitve.

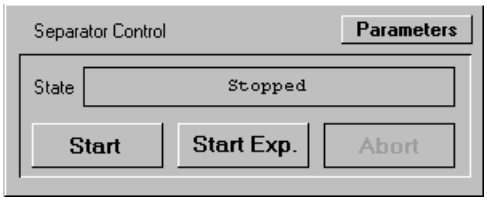


Slika C.6: - Grafični prikaz podprocesa priprave plina

C.5.1 Okno postopkovnega vodenja

Okno postopkovnega vodenja (slika C.7) zaseda prostor v samem oknu podprocesa za pripravo plina. Vsebuje polje, v katerem se izpisuje trenutno stanje postopka, tri gumbe za ukaze postopkovnemu vodenju (pomen gumbov je odvisen od trenutnega stanja), in gumb *Parameters*, ki aktivira okno s parametri postopkovnega vodenja.

Gumb *Start* je namenjen zagonu postopka v stanje *Running*, Gumb *Start Exp.* pa zagonu postopka v stanje *Experiment Running*.



Slika C.7: - Grafični prikaz podprocesa priprave plina

Klik na izpisno polje stanja postopka aktivira okno, prikazano na sliki 3.5. To okno vsebuje dve izpisni polji. Prvo vsebuje trenutno akcijo (korak) postopkovnega vodenja, drugo pa razlog zaustavitve delovanja.

Slika C.8: - Okno s sporočili postopkovnega vodenja

Podrobnosti o aplikaciji nadzornega vodenja procesnega laboratorija so opisane v [28] in [5].

Literatura

[1] Akcoglu M.A., Baxter J.R., Ha D.M., Jones R.L., Approximation of L2-Processes by Gaussian Processes, *New York Journal of Mathematics*, Volume 4, 75-82, 1998.

[2] Ažman, K., *Identifikacija dinamičnih sistemov z Gaussovimi procesi*, seminar, Univerza v Ljubljani, Fakulteta za elektrotehniko, Ljubljana, 2004.

[3] Banko B., Kocijan J., Uporaba Gaussovih procesov za identifikacijo nelinearnih sistemov = Use of Gaussian processes for non-linear systems identification, *Zbornik enajste mednarodne Elektrotehniške in računalniške konference ERK 2002, 23.-25. september 2002*, IEEE Region 8, Slovenska sekcija IEEE, zv. A, str. 323-326, Portorož, Slovenija, 2002.

[4] Beijer Electronics, MelDDE version 1.4, Manual, G & L Beijer Electronics AB, 2000.

[5] Burkeljc B., Godena G., Sistem vodenja kontinuirnega dela procesnega laboratorija, verzija 1.0, IJS delovno poročilo DP-8298/IJS, Institut Jožef Stefan, 2001.

[6] Gerkšič S., Navodila za eksperimentiranje s podprocesom ločevanja plina in tekočine v Procesnem laboratoriju za nevarne tehnologije Odseka za računalniško avtomatizacijo in regulacije IJS, Institut Jožef Stefan, 1999.

[7] Gerkšič, S., *Obravnava motenj v prediktivnem vodenju*, doktorska disertacija, Univerza v Ljubljani, Fakulteta za elektrotehniko, Ljubljana, 2000.

[8] Gibbs, M.N., J., Bayesian Gaussian Processes for Regression and Classification, Ph.D. Disertation, Cambridge University, 1997.

[9] Girard A., Murray-Smith R., Learning a Gaussian Process Model with Uncertain Inputs, Technical report, TR-2003-144, University of Glasgow, Glasgow, 2003.

[10] Girard A., Rasmussen C., Quinonero Candela J., Murray-Smith R., Multiple-step ahead prediction for non linear dynamic systems - A Gaussian Process treatment wih propagation of the uncertainty, *Advances in Neural Information Processing Systems*, volume 15, 545-552, MIT Press, 2002.

[11] Girard A., Rasmussen C.E., Murray-Smith R., Gaussian Process priors with Uncertain Inputs: Multiple-Step-Ahead prediction, Technical report, DCS TR-2002-119, University of Glasgow, Glasgow, 2002.

[12] Girard A., Rasmussen C.E., Quinonero Candela J., Murray-Smith R., Gaussian Process Priors With Uncertain Inputs & Application to Multiple-Step-Ahead Time Series Forecasting, NIPS 15, Vancouver, Canada, MIT Press, 2003.

[13] Girard, A. Modelling Non Linear Dynamic Systems with Gaussian Processes, First year report, University of Glasgow, U.K., (http://www.dcs.gla.ac.uk/~agathe/reports.html), 2001.

[14] Godena G., Model Based Software Engineering in the Domain of Process Control (internal report), IJS delovno poročilo DP-8032, Institut Jožef Stefan, Ljubljana, 1999.

[15] Grancharova, A., Johansen, T. A., Kocijan J., Explicit model predictive control of gas-liquid separation plant, *European Control Conference,* Cambridge, UK, 2003.

[16] Gregorčič G., Lightboy G., Internal model control based on a Gaussian process prior model, Proceedings of ACC'2003, Denver, CO, 4981-4986, 2003.

[17] Gregorčič G., Lightboy G., Gaussian Processes for Modelling of Dynamic Non-linear Systems, Irish Signals and Systems Conference, 2002.

[18] Hauptman B., *Specificiranje zahtev in izvedba procesnega vodenja na programabilnih logičnih krmilnikih*, magistrska naloga, Univerza v Ljubljani, Fakulteta za elektrotehniko, Ljubljana, 2000.

[19] Hauptman B., Godena G., Kandare G., Programska oprema sistema vodenja procesnega laboratorija, verzija 1.0, IJS delovno poročilo DP-8280/IJS, Institut Jožef Stefan, 2000.

[20] Jamnik, R., *Verjetnostni račun in statistika*, Društvo matematikov, fizikov in astronomov , Zveza organizacij za tehnično kulturo Slovenije, 1986.

[21] Kavšek-Biasizzo K., *Prediktivno vodenje nelinearnih sistemov,* doktorska disertacija, Univerza v Ljubljani, Fakulteta za elektrotehniko, Ljubljana, 1998.

[22] Kocijan J., Murray-Smith R., Rasmussen C., Girard A., Gaussian Process Model Based Predictive Control, American Control Conference, Boston, 2004.

[23] Kocijan J., Girard A., Banko B., Murray-Smith R., Dynamic Systems Identification with Gaussian Processes, Proceedings of 4th Mathmod, Vienna, 776-784, 2003.

[24] Kocijan J., Girard A., Leith D.J., Incorporating Linear Local Models in Gaussian Process Model, IJS delovno poročilo DP-8895, Inštitut Jožef Stefan, Ljubljana, 2003.

[25] Kocijan J., Likar B., Banko B., Girard A., Murray-Smith R., Rasmussen C.E., A case based comparisoin of identification with neural network and Gaussian process models, Preprints of IFAC ICONS Conference, Faro, 137-142, 2003.

[26] Kocijan J., Murray-Smith R., Rasmussen C.E., Likar B., Predictive control with Gaussian process, *IEEE Region 8 EUROCON 2003 : computer as a tool: 22-24. September 2003, Faculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia : proceedings*, zv. A, str. 352-356, 2003.

[27] Kocijan, J., Likar B., Banko B., Girard A., Murray-Smith R., Rasmussen C. E., Identification of pH neutralization process with neural networks and Gaussian process models, Institute Jožef Stefan, report DP-8575, Ljubljana, 2002.

[28] Likar, B., Godena G., Kocijan J., Eksperimentalno okolje za vodenje podprocesa priprave plina v realnem času iz okolja MATLAB-SIMULINK, IJS delovno poročilo DP-8463, Inštitut Jožef Stefan, Ljubljana, 2001.

[29] Maciejowski J.M., *Predictive control with constraints*, Pearson Education Limited, Harlow, 2002.

[30] Mackay, D.J.C., *Information theory, inference, and learning algorithms*, Cambridge University Press, Cambridge, 2003.

[31] Murray-Smith R., Girard A., Gaussian Process priors with ARMA noise models, Irish Signals and Systems Conference, Maynooth, 147-152, 2001.

[32] Quinonero Candela J., Girard A., Rasmussen C., Prediction at an Uncertain Input for Gaussian Processes and Relevance Vector Machines Application to Multiple-Step Ahead Time-Series Forecasting, Technical report, Informatics and Mathematical Modelling, Technical Univesity of Denmark, 2002.

[33] Rasmussen C.E., Evaluation of Gaussian Processes and other Methods for Non-Linear Regression, Ph.D. Disertation, Graduate department of Computer Science, University of Toronto, Toronto, 1996.

[34] Sollich, P., Learning curves for Gaussian processes. *Advances in Neural Information Processing Systems*, Vol. 11, 344-350, MIT Press, Cambridge, U.K., 1999.

[35] Strmčnik S., urednik, *Celostni pristop k računalniškemu vodenju procesov*, Založba FE Ljubljana, 1998.

[36] Vrančić, D., Juričić, Đ., Petrovčič, J., Measurements and mathematical modeling of a semi-industrial liquid gas separator for the purpose of a fault diagnosis. Delovno poročilo DP-7260, Institut Jožef Stefan, 1995.

[37] Williams, C.K.I., Prediction with Gaussian Processes: From Linear Regression to Linear Prediction and Beyond, *Learning and Inference in Graphical Models*, Kluwer Academic Press, 1998.

[38] Williams, C.K.I., Regression with Gaussian Processes, *Mathematics of Neural Networks: Models, Algorithms and Applications*, Kluwer Academic Publishers, 1997.

[39] Zorzut S., *Izvedba sistema vodenja podprocesa priprave plina z uporabo orodja Siemens Simatic PCS7*, Diploma, Univerza v Ljubljani, Fakulteta za elektrotehniko, Ljubljana, 2001.

Izjava

Izjavljam, da sem magistrsko nalogo izdelal samostojno pod vodstvom prof. dr. Juša Kocijana. Izkazano pomoč ostalih sodelavcev sem v celoti navedel v zahvali.

Bojan Likar