UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

LQG REGULACE V REÁLNÉM ČASE V PROSTŘEDÍ MATLAB

Bc. Jan Hudský

Diplomová práce

2023

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2021/2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Jan Hudský**
Osobní číslo:	**I20191**
Studijní program:	**N0714A150005 Automatické řízení**
Téma práce:	**LQG regulace v reálném čase v prostředí MATLAB**
Zadávající katedra:	**Katedra řízení procesů**

Zásady pro vypracování

Cíl: Vytvořte program v prostředí MATLAB, který realizuje LQ řízení lineárního SISO systému se známým modelem připojeným pomocí Arduino Due

Teoretická část:

a) návrh LQ regulátoru se sledováním žádané bez trvalé regulační odchylky
b) návrh estimátoru stavu
c) MATLAB: grafické uživatelské rozhraní programu a komunikace přes sériovou linku

Praktická část:

a) komunikační protokol Arduino Due a návrh řešení komunikace využívající možnosti sériové komunikace v MATLABu
b) simulační ověření návrhu LQG regulátoru
c) návrh GUI programu
d) vytvoření programu a implementace LQG regulátoru

Rozsah pracovní zprávy: **50 – 60 stran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

BALÁTĚ, Jaroslav, 2004. *Automatické řízení*. 2., přeprac. vyd. Praha: BEN –technická literatura. ISBN 80-730-0148-9.
OGATA, Katsuhiko. *Discrete-time control systems*. 2nd ed. Englewood Cliffs, N.J.: Prentice Hall, c1995, xi, 745 p. ISBN 01-303-4281-5.
Obsah dokumentace MATLAB. MathWorks. *MATLAB –mathematical computing software* [online]. 2021 [cit. 2020-09-20]. Dostupné z: MATLAB Documentation (mathworks.com)

Vedoucí diplomové práce: **doc. Ing. František Dušek, CSc.**
Katedra řízení procesů

Datum zadání diplomové práce: **8. listopadu 2021**
Termín odevzdání diplomové práce: **20. května 2022**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Daniel Honc, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 16. listopadu 2021

Prohlášení autora

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 25. 8. 2023

Bc. Jan Hudský

Poděkování

Rád bych vyjádřil svou vděčnost vedoucímu diplomové práce doc. Ing. Františkovi Duškovi, Ph.D. za jeho vedení, trpělivost a cenné rady při zpracování mé práce. Mé díky patří také mé rodině a blízkým za neustálou podporu během mého studia.

V Pardubicích dne 25. 8. 2023

Bc. Jan Hudský

ANOTACE

Diplomová práce se zabývá návrhem a implementací řídicího systému pro LQ řízení lineárního SISO systému v MATLABu s komunikací přes Arduino Due. V teoretické části jsou zkoumány metody získání modelu a principy LQ regulace. V praktické části je představen matematický model tohoto systému a důraz je kladen na grafické uživatelské rozhraní (GUI) v MATLABu, rozdělené do tří režimů: základního, návrhového a řídicího. Efektivita a správnost návrhu byla ověřena sérií testů a měření.

KLÍČOVÁ SLOVA

LQ, Stavová regulace, Arduino Due, Matlab GUI

TITLE

Real time LQG control in MATLAB environment

ANNOTATION

The master's thesis addresses the design and implementation of a control system for LQ control of a linear SISO system in MATLAB, communicating through Arduino Due. The theoretical section explores methods for model acquisition and principles of LQ regulation. The practical section introduces the mathematical model of this system, with a focus on the graphical user interface (GUI) in MATLAB, divided into three modes: basic, design, and control. The effectiveness and accuracy of the design were verified through a series of tests and measurements.

KEYWORDS

LQ, State control, Arduino Due, Matlab GUI

OBSAH

SEZNAM ZKRATEK A ZNAČEK

ARM	Acorn RISC Machine
ASCII	American Standard Code for Information Interchange
CAN	Controller Area Network
CMSIS	Common Microcontroller Software Interface Standard
GPS	Global Positioning Systém
GUI	Graphical User Interface
LQ	Lineárně kvadratický
LTI	Linear Time-Invariant
MATLAB	MATrix LABoratory
MIMO	Systém s více vstupy více výstupy
MRAC	Model Reference Adaptive Control
PC	Personal Computer
SISO	Systém s jedním vstupem jedením výstupem
URO	Uzavřený regulační obvod
USB	Universal Serial Bus

SEZNAM SYMBOLŮ PROMĚNNÝCH VELIČIN A FUNKCÍ

A	matice systému
$\mathbf{A}_{aug}$	rozšířená matice systému
$\mathbf{A}_{E}$	matice estimátoru
B	matice řízení
$\mathbf{B}_{aug}$	rozšířená matice řízení
$\mathbf{B}_{E}$	matice buzení estimátoru
C	matice výstupu
$\mathbf{C}_{aug}$	rozšířená matice výstupu
D	přímá přenosová matice
e(k)	regulační odchylka
He	matice estimátoru zpětné vazby
i	vektory proudů i_{1-4}
i_{1-4}	proudy jednotlivých větví obvodu, A
i_{A-Z}	smyčkové proudy, A
J	kvadratická účelová funkce
k	diskrétní čas
P	řešení Riccatiho rovnice
Q	váhová matice
$\boldsymbol{s_e}(k)$	suma regulačních odchylek
t	čas
$t0$	počáteční čas
tf	horizont řízení ve spojité oblasti
Ts	perioda vzorkování
U	elektrické napětí, V
u(k)	akční zásah (regulační zákon)
u(t)	akční zásah (spojitá oblast)
u_0	počáteční akční zásah
$\mathbf{x}_0$	vektor počátečních podmínek
x_{aug} (k)	rozšířený vektor stavových veličin

SEZNAM ILUSTRACÍ

SEZNAM TABULEK

ÚVOD

V dnešní době je automatizace procesů a systémů považována za jeden z klíčových faktorů v mnoha průmyslových, výzkumných a komerčních aplikacích. Pro dosažení optimálních a spolehlivých výsledků je vyžadováno sofistikované metody a techniky v oblasti řízení a kontroly těchto procesů. LQ (Linear Quadratic) regulace je považována za jednu z osvědčených metod řízení.

V rámci této diplomové práce byl stanoven hlavní cíl ve vytvoření programu v prostředí MATLAB, který je určen k realizaci LQ řízení lineárního SISO (Single Input Single Output) systému. Klíčovým aspektem tohoto projektu je zapojení a komunikace s hardwarem Arduino Due, což umožňuje aplikaci teoretických modelů a algoritmů v reálném čase na fyzickém systému.

V teoretické části je kladen důraz na detailní popis systému, metody získání jeho modelu a hlubší zkoumání problematiky LQ regulace. Různé metody návrhu LQ regulátoru jsou diskutovány, včetně diferencí mezi spojitým a diskrétním modelem a různými horizonty regulace. Pozornost je věnována návrhu estimátoru stavu, zejména Luenbergova deterministického estimátoru.

V praktické části je prezentována aplikace teoretických metod v konkrétních reálných podmínkách. popis je poskytnut pro regulovanou soustavu, matematický model a interakci mezi MATLABem a Arduinem Due. Dále zde je verifikace modelu a přehled několika simulací, aby bylo zjištěno, jak parametry LQ regulátoru ovlivňují finální výsledek. Zvláštní důraz je kladen na grafické uživatelské rozhraní (GUI) v MATLABu, sloužící jako vizuální platforma pro ovládání Arduina a provedení aktuálního řízení. Toto GUI umožňuje uživatelům realizovat návrh estimátoru, LQ regulátoru a provádět řízení v reálném čase.

V závěru byla pozornost soustředěna na porovnání simulací s experimenty provedenými na skutečné soustavě.

1 TEORETICKÁ ČÁST

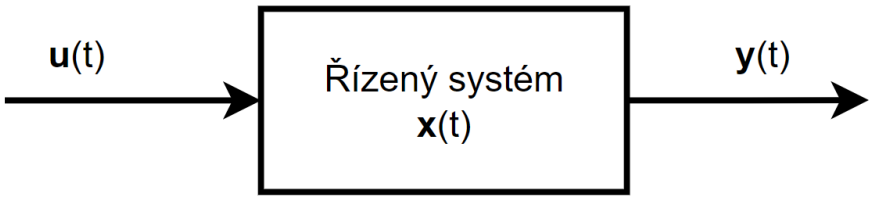
Teoretická část je věnována stručnému popisu systému a přístupu k modelování. Dále zde byly představeny hlavní rozdíly mezi spojitým a diskrétním modelem. Následující kapitoly byly zaměřeny na pochopení základu vlastností LQ (Lineárně-kvadratického) regulátoru a následně jsou zde připomenuty problémy, které mohou nastat při návrhu LQ regulátoru a estimátoru stavů. K závěru teoretické části je návrh LQ regulátoru se sledováním žádané hodnoty bez trvalé regulační odchylky.

1.1 POPIS SYSTÉMU

Popis systému je považován za jeden z základních kroků v procesu modelování a identifikace. Tímto popisem jsou charakterizovány informace o vstupech a výstupech systému, jeho dynamických vlastnostech a způsoby, jakými tyto vlastnosti jsou ovlivňovány, jsou určeny chováním systému. V řídicích systémech jsou běžně používány dva základní termíny, SISO a MIMO, pro lepší pochopení toho, o jaký systém se jedná.

SISO (Single Input, Single Output) je považován za termín, kterým jsou popsány systémy s jedním vstupem a výstupem. Tyto systémy jsou považovány za nejjednodušší a nejčastěji se vyskytující v praxi. MIMO (Multiple Input, Multiple Output) je používán jako termín, kterým jsou popsány systémy s více vstupy a výstupy. Tyto systémy jsou vnímány jako složitější a umožňují pokročilejší řízení a regulaci.

Lineární MIMO dynamický systém, jehož vstupem je vektor vstupních veličin $\mathbf{u}(t)$, stavové veličiny jsou ve vektoru $\mathbf{x}(t)$ a výstupem je vektor výstupních veličin $\mathbf{y}(t)$. Jednorozměrný systém (viz obrázek 1.1) je charakterizován tím, že oba vektory, jak vstupní, tak výstupní, obsahují pouze jednu složku. Počet prvků stavového vektoru $\mathbf{x}(t)$ je určen řádem systému (Balátě, 2004).



Obrázek 1.1 – Systém s jedním vstup a jedním výstupem pro stavový popis

Pro lepší pochopení fungování systému je často využíván tzv. matematický model. Matematický model je popisován jako matematická struktura, což je soubor veličin, které jsou vzájemně propojovány relačními operátory a funkčními vztahy. Tyto funkční vztahy jsou vnímány jako reprezentace fyzikálních vazeb mezi veličinami v reprezentovaném reálném systému. Matematický model může být rozdělen na vnější a vnitřní popis systému. (Balátě, 2004. Vnější popis, 2006).

Vnější popis systému (také označovaný jako černá skříňka) je zaměřen na vztah mezi vstupy a výstupy systému, aniž by bylo zkoumáno, co se děje uvnitř systému. Tento popis je často využíván, když detaily vnitřní struktury systému jsou neznámé, považovány za nepodstatné nebo jsou příliš složité. Je zdůrazňována pouze reakce systému na vstupní signály. (Balátě, 2004. Vnější popis, 2006).

Na druhé straně je vnitřní popis zaměřen na vnitřní strukturu a fungování systému. Tento popis je často využíván, když je potřeba porozumět, jak systém funguje, nebo když má být systém navržen nebo optimalizován. Dynamické vlastnosti reakcí mezi vstupem $\mathbf{u}(t)$, vnitřním stavem $\mathbf{x}(t)$ a výstupem systému $\mathbf{y}(t)$ jsou v systému zachyceny, což vede k vytvoření stavového modelu systému (Balátě, 2004).

1.1.1 Způsoby získání modelu

Model je vyvíjen prostřednictvím procesu modelování. Je považováno za nezbytné, aby model odpovídal kvantitativním a kvalitativním požadavkům původního řízeného systému. Významné vlastnosti z hlediska zkoumaného řízeného systému musí být v matematickém modelu vyjádřeny. Matematický model je zjednodušován, aby nebyl příliš složitý, což je dosaženo vyloučením pomocných nebo nadbytečných atributů, které by jinak zvýšily jeho složitost. V důsledku toho je při modelování hledán vhodný kompromis mezi složitostí a přesností (Balátě, 2004. Modelování, 2006).

Identifikace je chápána jako proces, v němž jsou model a vyšetřovaný objekt ztotožňovány. Jsou rozlišovány dva primární postupy při modelování a identifikaci. Konkrétně je to analytický postup a empirický postup. Oba postupy jsou charakterizovány svými výhodami a nevýhodami, a proto jsou v závislosti na konkrétní situaci uplatňovány. Existují také hybridní postupy, které kombinují výhody obou těchto základních postupů (Strejc, 1978. Modelování, 2006)

Analytický postup je založen na materiálové a energetické bilanci zařízení a na pochopení fyzikálních, chemických dílčích procesů probíhajících v zařízení a jejich

matematického popisu. Na základě těchto poznatků může být vypracován analytický model daného zařízení. Model je charakterizován svými vnitřními stavovými veličinami a vzájemnými vztahy mezi nimi. Každá parametrická a stavová proměnná modelu je spojena s konkrétním fyzikálním významem. Struktura modelovaného objektu je tímto modelem reprezentována. Model získaný tímto analytickým způsobem je považován za vnitřní popis systému (Strejc, 1978. Modelování, 2006)

Pro empirický postup je vyžadována existence reálného objektu pro měření. Experimentální model může být hodnocen jako přesnější než model, který byl získán analytickým způsobem, protože výsledky měření zahrnují aspekty reality, které nemohou být zachyceny nebo jsou při analytickém modelování zanedbány. Nicméně, model je platný pouze v té oblasti, kde bylo měření prováděno, a pro jinou oblast změn vstupních veličin musí být vytvořen jiný model.

Modely, které jsou tímto způsobem vytvářeny, ať už analyticky nebo empiricky, jsou považovány za základní nástroj pro porozumění dynamickým systémům a jejich chování. Je umožněno otestování a ověření různých scénářů a strategií bez potřeby implementace nebo zásahu do skutečného systému. Toto je považováno za obzvláště cenné v situacích, kdy experimenty s reálnými systémy jsou hodnoceny jako nákladné, nebezpečné nebo nemožné.

Díky modelování může být dosaženo hlubšího porozumění dynamice systému, což může vést k lepším rozhodnutím a strategiím. Bez ohledu na to, zda je model vytvořen analyticky nebo experimentálně, jeho cílem je vždy poskytnout užitečný nástroj pro pochopení, předpověď a řízení chování systému. (Modelování, 2006).

1.1.2 Spojitý stavový model

Spojitý stavový model je matematická reprezentace dynamického systému, v níž jsou definovány vztahy mezi vstupními **x**(*t*), výstupními **y**(*t*) a stavovými proměnnými systému **x**(*t*). To je realizováno prostřednictvím souboru diferenciálních rovnic prvního řádu (Švarc, 2003).

$$\boldsymbol{x}'(t) = \boldsymbol{f}[\boldsymbol{x}(t), \boldsymbol{u}(t), t], \qquad (1.1)$$

kde t – čas, s,

$\boldsymbol{u}$ – vektor vstupních veličin,

$\boldsymbol{x}$ –vektor stavových veličin.

Výstupy systému pak lze určit výstupní rovnicí

$$\boldsymbol{y}(t) = \boldsymbol{g}[\boldsymbol{x}(t), \boldsymbol{u}(t), t] \qquad (1.2)$$

kde $\boldsymbol{y}$ – je vektor výstupů

Pokud je systém lineární a časově invariantní, jsou funkce f a g lineární s konstantními koeficienty. V takovém případě mohou být rovnice (*1.1*) a (*1.2*) převedeny na vektorově-maticový formát:

$$\boldsymbol{x}'(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \qquad (1.3)$$
$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t), \qquad (1.4)$$

kde t – čas, s,

A – matice systému,

B – matice buzení systému,

C – výstupní matice systému,

D – přímá přenosová matice,

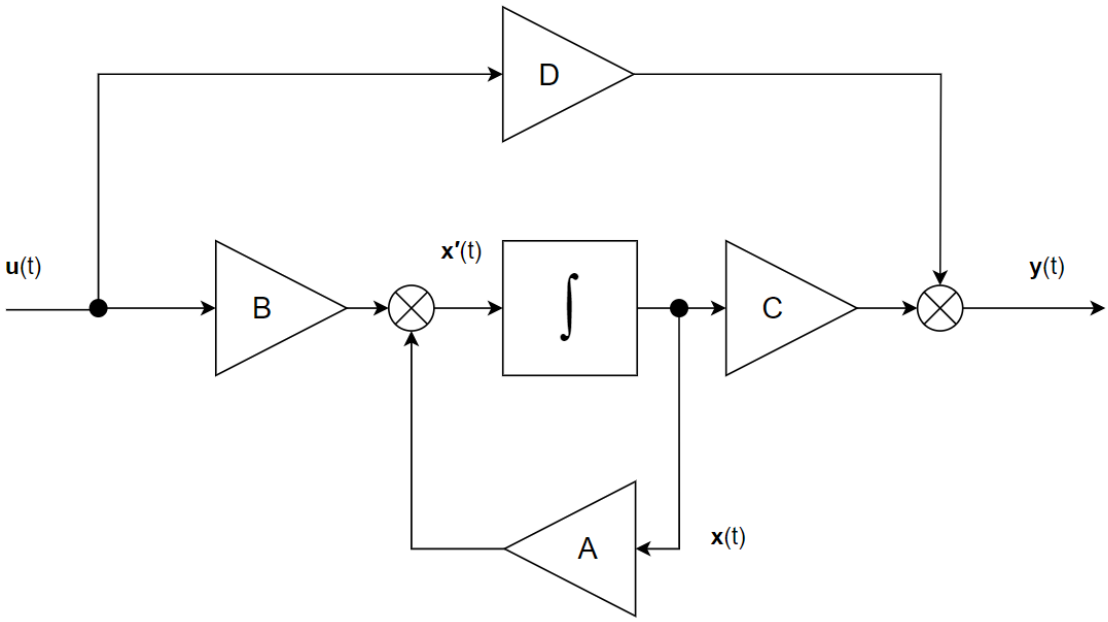
$\boldsymbol{x}$ –vektor stavových veličin,

$\boldsymbol{u}$ – vektor vstupních veličin,

$\boldsymbol{y}$ – vektor výstupních veličin,

Rovnice *1.3* a *1.4* jsou označovány jako lineární stavové rovnice nebo také jako rovnice lineárního stavového modelu dynamického systému. Prvky matic **A**, **B**, **C**, **D**, jež jsou konstantní v čase, jsou považovány za kompletní vnitřní popis systému. (Švarc, 2003).

U systémů, u nichž není určena přímá souvislost mezi vstupem a výstupem, je matice **D** stanovena jako nulová. Jak bylo zmíněno, rovnice *1.3* a *1.4* jsou považovány za spojitý stavový popis a odpovídající blokové schéma je zobrazeno na obrázku 1.2 (Balátě, 2004).



Obrázek 1.2 – Blokové schéma dle stavového spojitého popisu

1.1.3 Diskrétní stavový model

Diskrétní stavový model, podobně jako jeho spojitý protějšek, je vnímán jako matematická reprezentace dynamického systému, ve které jsou vztahy mezi vstupními $\mathbf{x}(k)$ $\mathbf{x}(t_k)$, výstupními $\mathbf{y}(k)$ a stavovými proměnnými $\mathbf{x}(k)$ systému definovány pouze v diskrétních časových okamžicích $t_k=t_0+kT$ $k=0,1,\ldots$. V diskrétním modelu, na rozdíl od spojitého modelu s kontinuálním časem, je čas reprezentován diskrétními kroky k.

Dále se budeme zabývat pouze lineárním časově invariantním systémem *1.3, 1.4* tj. s konstantními maticemi **A**, **B**, **C** a **D**. Vzorkování systému T musí být provedeno s dostatečnou frekvencí, aby byly zachyceny všechny klíčové dynamické charakteristiky. Při transformaci spojitých rovnic na diskrétní v rámci diskrétního stavového modelu jsou často využívány metody vzorkování a držení vzorku nebo z-transformace. Existuje několik metod diskretizace, ale jejich výběr závisí na specifikách aplikace a požadované přesnosti. Rovnice následně lze přepsat pro diskrétní stavový model následovně:

$$\boldsymbol{x}(k+1) = \mathbf{f}[\boldsymbol{x}(k), \boldsymbol{u}(k), k], \quad (1.5)$$

$$\boldsymbol{y}(k) = \mathbf{g}[\boldsymbol{x}(k), \boldsymbol{u}(k), k], \quad (1.6)$$

$$\boldsymbol{x}'(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \quad (1.7)$$

$$\boldsymbol{y}(\mathrm{k}) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(\mathrm{k}), \quad (1.8)$$

Kde k – diskrétní čas,

$\mathbf{u}(k)$ – vektor vstupních veličin,

$\mathbf{x}(k)$ – vektor stavových veličin,

$\mathbf{x}(k+1)$ – vektor stavových veličin v následujícím kroku,

$\mathbf{y}(k)$ – je vektor výstupů,

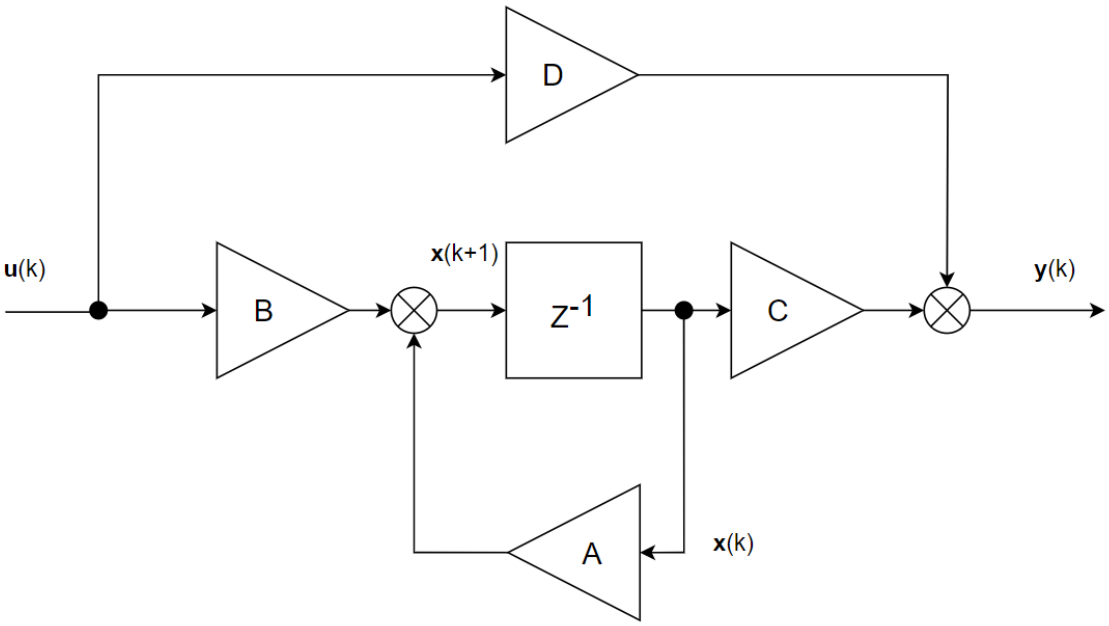
A – matice systému,

B – matice buzení systému,

C – výstupní matice systému,

D – přímá přenosová matice.

Pro systémy, které nemají přímou souvislost mezi vstupem a výstupem, je matice **D** nulová. Vztahy (*1.7*) a (*1.8*) pak reprezentují diskrétní stavový popis, jehož odpovídající blokové schéma je znázorněno na obrázku 1.3 (Balátě, 2004).



Obrázek 1.3 – Blokové schéma dle stavového diskrétního popisu

1.2 LQ REGULÁTOR

LQ regulátor je příkladem optimální metody regulace. Hlavním cílem LQ regulátoru je určit časový průběh akčního zásahu $\mathbf{u}(t)$ tak, aby kvadratické kritérium (*1.11*) bylo minimalizováno. Slovo "optimální" zde znamená dosažení nejlepšího možného výsledku vzhledem k zadanému kritériu. Ačkoliv je spojitý LQ regulátor při splnění předpokladů optimální a garantuje stabilitu URO je třeba si uvědomit jeho omezení v kontextu robustnosti a stability, zejména v situacích s rušením a nejistotami.

V základní podobě LQ regulátor zajišťuje z hlediska kritéria optimální přechod systému z počátečního nenulového stavu do koncového nulového stavu. Volitelné váhové matice umožňují uživateli ovlivnit výslednou regulaci podle potřeby. Pro sledování konkrétní cílové hodnoty může být potřeba upravit regulátor.

1.2.1 Problematika návrhu LQ regulátoru

Při návrhu LQ regulátoru musí být splněny určité předpoklady. Na tyto konkrétní problémy se podíváme níže.

Řiditelnost

Pojem řiditelnost je považován za základní kámen teorie řízení a je charakterizován možností ovlivnění systému prostřednictvím externího vstupu. Pokud je systém považován za řiditelný, může být převeden pomocí adekvátního řídicího vstupu z libovolného počátečního stavu do jakéhokoli konečného stavu v omezeném časovém rámci. Zejména u lineárních systémů, které jsou často modelovány v rámci stavového prostoru, je koncept řiditelnosti považován za velmi významný. Testování řiditelnosti u lineárních časově invariantních systémů je prováděno prostřednictvím matice řiditelnosti. Matice řiditelnosti je konstruována ze systémových matic daného systému. Pokud je systém popisován stavovými rovnicemi (*1.3*) a (*1.4*) (Strejc, 1978).

$$\mathbf{R} = [\mathbf{A^{-1}B}\ \ \mathbf{A^{-2}}\ \ \ldots\ \ \mathbf{A^{-n}B}] \qquad (1.9)$$

Pokud systém není řiditelný, znamená to, že externím vstupem nemohou být ovlivněny některé stavy systému. V důsledku toho může být schopnost regulátoru efektivně řídit systém a dosáhnout požadovaného výstupu výrazně omezena. Jako klíčový koncept při

navrhování systémů řízení je řiditelnost nezbytnou podmínkou pro efektivní řízení a regulaci většiny systémů (Kubík, 1982).

Dosažitelnost

Dosažitelnost a řiditelnost jsou považovány za kritické koncepty v teorii řízení. Na rozdíl od řiditelnosti, která se týká schopnosti ovládat systém a převést ho z jakéhokoli počátečního stavu do nulového koncového stavu, je dosažitelnost širší pojem, který se zabývá možnostmi stavového prostoru systému. Stav je považován za dosažitelný, pokud lze systém pomocí určité sekvence vstupů převést do tohoto stavu v omezeném časovém období od daného počátečního stavu. Dosažitelná množina daného systému je tvořena množinou všech takových dosažitelných stavů. Dosažitelnost má zásadní význam pro návrh a analýzu dynamických systémů. Pokud je cílový stav mimo dosažitelnou množinu, žádný řídicí zásah nemůže dosáhnout jeho realizace. To může mít významné důsledky pro návrh řízení, protože omezuje možnosti vývoje regulátoru a může vést k neschopnosti systému dosáhnout požadovaného výkonu. Dosažitelnost lze ověřit z následujícího vztahu (Kubík, 1982).

$$\mathbf{P} = [\mathbf{C} \quad \mathbf{A}^T\mathbf{C}^T \quad \dots \quad (\mathbf{A}^T)^{\mathbf{n-1}}\mathbf{C}^T] \qquad (1.10)$$

U lineárních systémů je obvykle dosažitelnost spojena s řiditelností. Pokud je systém řiditelný, každý stav je dosažitelný. Nicméně, u nelineárních systémů nejsou řiditelnost a dosažitelnost nutně totožné a analýza dosažitelnosti může být složitější.

1.2.2 Návrh LQ regulátoru spojitá oblast konečný horizont

Spojitý stavový model

Pro návrh LQ regulátoru ve spojité oblasti bylo považováno za nezbytné mít detailní znalosti o spojitém stavovém modelu, o kterém bylo hovořeno v kapitole 1.1.2. Rovnicemi *1.3* a (*1.4*) je ve spojité oblasti reprezentován stavový model.

Kvadratická účelová funkce na konečném horizontu řízení

Při konstrukci LQ regulátoru bylo považováno za esenciální zahrnout kvadratickou účelovou funkci. K definování přístupu k regulaci bylo použito kritérium, jehož cílem byla minimalizace kvadratické účelové funkce v průběhu určitého časového intervalu od

startovního bodu t_0 až po konečný bod t_f. Toto kritérium může být matematicky formulováno následovně (Strejc. 1978):

$$\mathrm{J(u)} = \mathbf{x}^T(tf)\mathbf{F}\mathbf{x}(tf) + \int_{t_0}^{t_f} [\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}\mathbf{u}(t)dt], \qquad (1.11)$$

Kde t – čas,

t_f – horizont řízení,

x –vektor stavových veličin,

u – vektor vstupních veličin,

F – pozitivně semidefinitní váhová matice,

Q – pozitivně semidefinitní váhová matice,

R – pozitivně definitní váhová matice.

V řízení byla klíčová role připisována váhovým maticím **Q** a **R**. Velikostí akčních zásahů bylo ovlivňováno hodnotami matice **R**, přičemž nižší hodnoty této matice umožňovaly větší zásahy. Změnou hodnot v matici **Q** byla ovlivněna rychlost, s jakou se regulovaná veličina ustálila. Volba mezi rychlým řízením s většími akčními zásahy a pomalejším řízením, které je šetrnější k řídicím prvkům, může být ovlivněna projektanty regulátoru tímto způsobem (Balátě, 2004. Strejc, 1978).

Riccatiho rovnice

Minimalizace kvadratického kritéria na konečném horizontu (*1.11*) s lineární dynamickou soustavou vede na řešení diferenciální maticové Riccatiho rovnice (Strejc, 1978)

$$-\frac{d\mathbf{P}}{dt} = \mathbf{A}^T\mathbf{P}(t) + \mathbf{P}(t)\mathbf{A} - \mathbf{P}(t)\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{P}(t) + \mathbf{Q} \qquad (1.12)$$

s okrajovou podmínkou

$$\mathbf{P}(t_f) = \mathbf{F} \qquad (1.13)$$

Regulační zákon s proměnným zesílením

Regulační zákon (*1.16*) je zpětná vazba od aktuálního stavu systému **x**(*t*) s proměnným zesílením **L**(*t*) – časový průběh zesílení je závislý na matici řešení Riccatiho rovnice **P**(*t*). Pro regulační zákon platí:

$$\boldsymbol{u}(t) = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{P}(t)\mathbf{x}(\boldsymbol{t}). \quad (1.14)$$

Pro zjednodušení je zaveden vektor zesílení $\mathbf{L}(t)$

$$\mathbf{L}(t) = \mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{P}(t). \quad (1.15)$$

Výsledná regulační zákon s proměnným zesílením je ve tvaru

$$\mathbf{u}(t) = -\mathbf{L}(t)\mathbf{x}(t), \quad (1.16)$$

kde **L** – zesílení zpětné vazby.

1.2.3 Návrh LQ regulátoru spojitá oblast nekonečný horizont

Řešení LQ regulátoru na nekonečném horizontu (*1.17*) vede na ustálené řešení Riccatiho rovnice (*1.18*), tj. matice **P** je konstantní. Potom i zesílení **L** regulačního zákona (*1.19*) je konstantní, což vede na velmi jednoduchou implementaci LQ regulátoru.

Kvadratická účelová funkce na nekonečném horizontu řízení

Pro nekonečný horizont řízení je kvadratická účelová funkce zjednodušena následovně:

$$\mathrm{J(u)} = \int_{\boldsymbol{t_0}}^{\infty} [\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u}]dt. \quad (1.17)$$

Riccatiho rovnice

Pro nekonečný horizont řízení je Riccatiho rovnice transformována do algebraické maticové formy. V této formě je řešením konstantní matice. Díky tomu je zesílení řídicího zákona také konstantní a nezávisí na čase

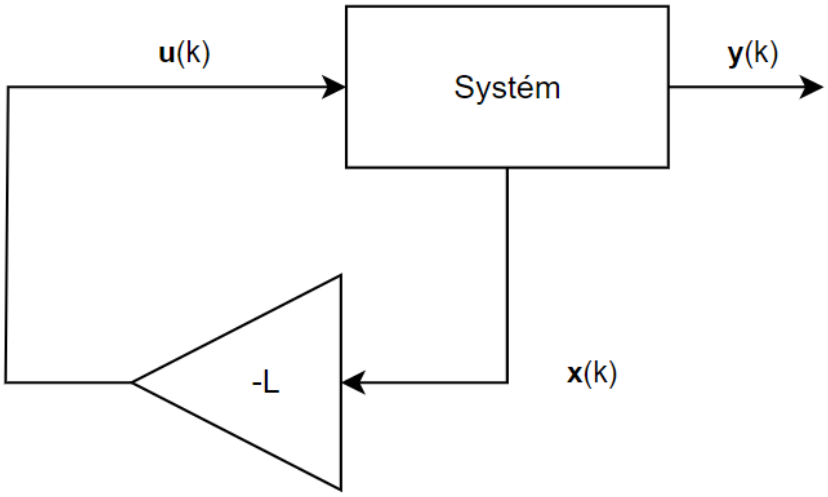
$$0 = \mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} - \mathbf{P}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P} + \mathbf{Q}. \quad (1.18)$$

Regulační zákon s konstantním zesílením

Výsledný vztah pro regulační zákon je stanoven následovně

$$\mathbf{u}(t) = -\mathbf{L}\mathbf{x}(t). \quad (1.19)$$

V moderní době byly vyvinuty nástroje, které usnadňují výpočet vektoru zesílení. Jako příklad lze uvést MATLAB, v němž je pomocí funkce `lqr` možné vypočítat vektor zesílení **L**, když jsou poskytnuty matice systému **A** a **B** a parametry pro regulátor **Q** a **R**. Blokové schéma zapojení LQ regulátoru je zobrazeno na obrázku 1.4.



Obrázek 1.4 – Blokové schéma LQ regulátoru

1.2.4 Návrhu LQ regulátoru diskrétní oblast konečný horizont

Diskrétní stavový model

Znalost diskrétního stavového modelu je nezbytná pro navržení LQ regulátoru v diskrétní oblasti. V kapitole 1.1.3 je tento model popsán, konkrétně pomocí rovnic (*1.7*) a (*1.8*).

Kvadratická účelová funkce na konečném horizontu řízení

Základní tvar účelové funkce pro konečný horizont řízení je určen následovně.

$$J = \mathbf{x}^T(N)Fx(N) + \sum_{k=0}^{N-1} \mathbf{x}^T(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}^T(k)\mathbf{R}\mathbf{u}(k), \qquad (1.20)$$

kde N – horizont řízení.

Riccatiho rovnice

Minimalizace účelové funkce (*1.20*) se soustavou (*1.7*) vede na řešení diferenční Riccatiho rovnice (Ogata, 1995)

$$\mathbf{P}(k) = \mathbf{Q} + \mathbf{A}^T\mathbf{P}(k+1)\mathbf{A} - \mathbf{A}^T\boldsymbol{P}(k+1)\mathbf{B}(\mathbf{R} + \mathbf{B}^T\mathbf{P}(k+1)\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}(k+1)\mathbf{A}. \quad (1.21)$$

Regulační zákon s proměnným zesílením

Pro stanovení regulačního zákona je vztah (Ogata, 1995),

$$\mathbf{u}(k) = -(\mathbf{R} + \mathbf{B}^T\mathbf{P}(k+1)\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}(k+1)\mathbf{A}\mathbf{x}(k), \quad (1.22)$$

vektor zesílení zpětné vazby $\mathbf{L}(k)$ je vyjádřen,

$$\mathbf{L}(k) = (\mathbf{R} + \mathbf{B}^T\mathbf{P}(k+1)\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}(k+1)\mathbf{A}. \quad (1.23)$$

Výsledný vztah pro regulační zákon je

$$\mathbf{u}(k) = -\mathbf{L}(k)\mathbf{x}(k). \quad (1.24)$$

1.2.5 Návrh LQ regulátoru diskrétní oblast nekonečný horizont

Při adaptaci na nekonečný horizont byla kvadratická účelová funkce upravena. V rámci tohoto přístupu byla využita algebraická Riccatiho rovnice k získání ideálního zesílení pro zpětnovazební řízení.

Kvadratická účelová funkce na nekonečném horizontu řízení

$$J = \sum_{k}^{\infty} \mathbf{x}^{\mathbf{T}}(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}^T(k)\mathbf{R}\mathbf{u}(k). \quad (1.25)$$

Riccatiho rovnice

Pro nekonečný horizont řízení je člen **P** považován za konstantní a Riccatiho diferenční rovnice je převedena do algebraické formy

$$\mathbf{P} = \mathbf{Q} + \mathbf{A}^T\mathbf{P}\mathbf{A} - \mathbf{A}^T\mathbf{P}\mathbf{B}[\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B}]^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A}. \quad (1.26)$$

Regulační zákon s konstantním zesílením

Pro stanovení regulačního zákona je využíván následující vztah

$$\mathbf{u}(k) = -(\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A}\mathbf{x}(k), \quad (1.27)$$

vektor zesílení zpětné vazby **L** je zaveden,

$$\mathbf{L} = (\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A}. \quad (1.28)$$

Vztahem je dán regulační zákon s konstantním zesílením

$$\mathbf{u}(k) = -\mathbf{L}\mathbf{x}(k). \quad (1.29)$$

Ve vývojovém prostředí MATLAB je pro výpočet vektoru zesílení **L** v diskrétní oblasti poskytována funkce `dlqr`. K jejímu použití jsou vyžadovány matice systému **A, B** a parametry pro regulátor **Q** a **R**.

1.2.6 Návrh LQ regulátoru se sledováním žádané hodnoty bez trvalé regulační odchylky

LQ regulátor v základní podobě (*1.19*) nebo (*1.29*) zajišťuje přechod řízené soustavy do nulového stavu, a tedy nulové hodnoty regulované veličiny. Při praktickém použití se ale nejčastěji požaduje regulace na nenulovou hodnotu regulované veličiny. Proto je potřeba základní řešení rozšířit o sledování žádané hodnoty regulované veličiny.

Jedno z možných rozšíření spočívá v zahrnutí integrálu regulační odchylky mezi stavové veličiny a tím i do kritéria. Tím se získá regulátor s integračním charakterem, který umožňuje dosažení nulové regulační odchylky v ustáleném stavu i při neshodě chování modelu a reálné soustavy a při působení poruch.

Regulační odchylka je definována vztahem (*1.30*)

$$\boldsymbol{e}(k) = \boldsymbol{w}(k) - \boldsymbol{y}(k), \quad (1.30)$$

agregace regulační odchylky je definována vztahem,

$$\boldsymbol{S_e}(k+1) = \boldsymbol{S_e}(k) + \overbrace{\boldsymbol{w}(k) - \boldsymbol{y}(k)}^{\boldsymbol{e}(k)}, \quad (1.31)$$

kde $\boldsymbol{S_e}(k)$ – suma regulačních odchylek.

Po dosazení za $\mathbf{y}(k)$ je provedena následující úprava

$$\boldsymbol{S_e}(k+1) = \boldsymbol{S_e}(k) + \overbrace{\boldsymbol{w}(k) - \mathbf{C}\mathbf{x}(k)}^{\boldsymbol{e}(k)}. \tag{1.32}$$

Rozšířené stavy lze rozepsat následovné

$$\begin{aligned} \boldsymbol{x_{aug}}(k+1) &= \begin{bmatrix} \boldsymbol{x}(k+1) \\ \boldsymbol{S_e}(k+1) \end{bmatrix}, \quad \mathbf{A_{aug}} = \begin{bmatrix} \mathbf{A} & 0 \\ -\mathbf{C} & 1 \end{bmatrix}, \\ \boldsymbol{x_{aug}} &= \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{S_e}(k) \end{bmatrix}, \quad \mathbf{B_{aug}} = \begin{bmatrix} \boldsymbol{B} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{C_{aug}} = [\mathbf{C} \quad 0]. \end{aligned} \tag{1.33}$$

Rozšířený model vypadá pak následovně

$$\overbrace{\begin{bmatrix} \boldsymbol{x}(k+1) \\ \boldsymbol{S_e}(k+1) \end{bmatrix}}^{\boldsymbol{x_{aug}(k+1)}} = \overbrace{\begin{bmatrix} \boldsymbol{A} & 0 \\ -\boldsymbol{C} & 1 \end{bmatrix}}^{\mathbf{A_{aug}}} \overbrace{\begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{S_e}(k) \end{bmatrix}}^{\boldsymbol{x_{aug}(k)}} + \overbrace{\begin{bmatrix} \boldsymbol{B} \\ 0 \end{bmatrix}}^{\mathrm{B_{aug}}} \boldsymbol{u}(k) + \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix} \boldsymbol{w}(k), \tag{1.34}$$

výstupní rovnice byla pak prezentována následovně

$$\boldsymbol{y}(k) = \overbrace{[\mathbf{C} \quad 0]}^{\mathbf{C_{aug}}} \overbrace{\begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{S_e}(k) \end{bmatrix}}^{\boldsymbol{x_{aug}}(k)}. \tag{1.35}$$

Rovnice pak lze přepsat následujíc

$$\boldsymbol{x_{aug}}(k+1) = \mathbf{A_{aug}}\boldsymbol{x_{aug}}(k) + \mathbf{B_{aug}}\boldsymbol{u}(k) + \mathbf{B_w}\boldsymbol{w}(k), \tag{1.36}$$

$$\boldsymbol{y}(k) = \mathbf{C_{aug}}\boldsymbol{x_{aug}}(k), \tag{1.37}$$

Kde $\mathbf{A_{aug}}$ – rozšířená matice systému **A**,

$\mathbf{B_{aug}}$ – rozšířená matice řízení **B**,

$\mathbf{C_{aug}}$ – rozšířená matice výstupu **C**.

Kvadratická účelová funkce bude upravena a konkrétní změna spočívá v použití rozšířených stavů.

$$J = \sum_{k}^{\infty} x_{aug}^{T}(k)\mathbf{Q_{aug}}\mathbf{x_{aug}}(k) + \boldsymbol{u}^{\boldsymbol{T}}(k)\mathbf{R}\mathbf{u}(k). \tag{1.38}$$

Po začlenění rozšířeného stavu do regulačního zákona bylo dosaženo nové formy

$$\mathbf{u}(k) = -\mathbf{L}_{\mathbf{aug}}\boldsymbol{x}_{aug}(k) = -[\boldsymbol{L}_x \quad \boldsymbol{L}_W]\begin{bmatrix}\boldsymbol{x}(k)\\ \boldsymbol{S}_e(k)\end{bmatrix}, \tag{1.39}$$

$$\mathbf{u}(k) = -\mathbf{L}_x\boldsymbol{x}(k) - \boldsymbol{L}_w\boldsymbol{S}_e(k), \tag{1.40}$$

kde $\mathbf{u}(k)$ – řídicí signál v čase k,

$\mathbf{L}_x$ – matice zesílení pro rozšířené stavy,

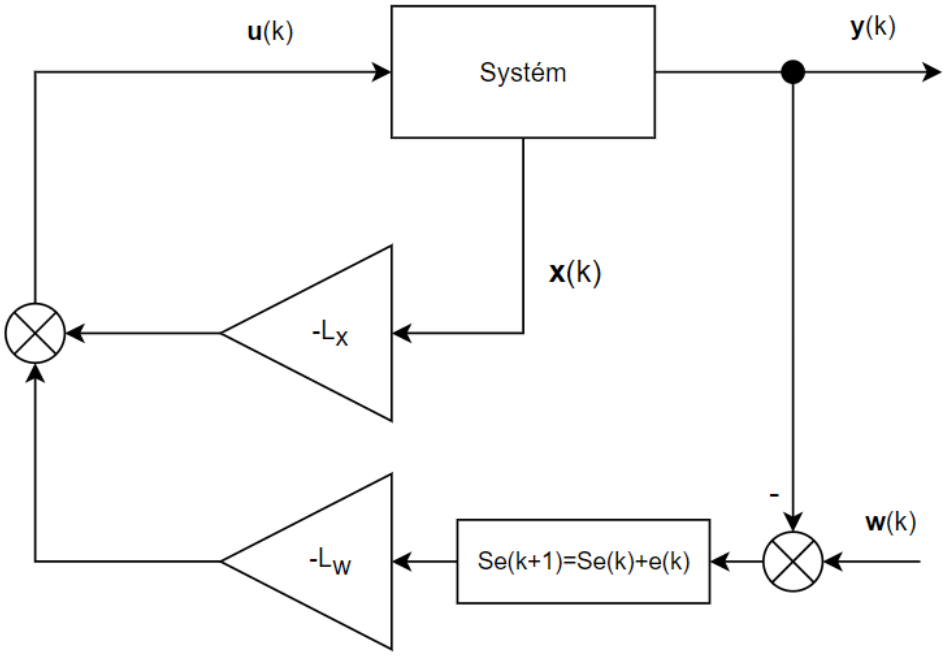
$\mathbf{x}(k)$ – Stavový vektor v čase k,

$\mathbf{L}_w$ – zesílení pro kompenzaci sumy regulačních odchylek,

$\boldsymbol{S}_e(k)$ – Suma regulačních odchylek v čase k.

S uvedenými rovnicemi byl formulován rozšířený stavový model, ve kterém je zahrnuta suma regulačních odchylek. Pro návrh LQ regulátoru jsou tyto rovnice považovány za klíčové. Díky zahrnutí této sumy může být dosaženo lepšího sledování žádané hodnoty s kompenzací trvalých regulačních odchylek. Pro výpočet je opět možné použití MATLABu.

URO s LQ regulátorem, který sleduje žádanou hodnotu a integruje regulační odchylku, lze znázornit ve formě blokového schématu na obrázku 1.5.



Obrázek 1.5 – LQ regulátor rozšířený o sumou regulačních odchylek

1.3 NÁVRH ESTIMÁTORU STAVU

Estimátor stavu je klíčovým prvkem mnoha moderních řídících systémů, zejména těch využívajících stavové řízení, má za úkol odhadovat vnitřní, nepřímo měřitelné stavy systému. Tyto odhady jsou poté použity pro řízení systému. Průběžné odhady hodnot stavového vektoru systému jsou prováděny estimátorem na základě poznání vstupů a výstupů systému a jeho modelu. Mnohdy jsou používány i termíny jako rekonstruktor nebo stavový pozorovatel. Pro implementaci estimátoru je nutné mít znalost stavového popisu systému (Strejc, 1978).

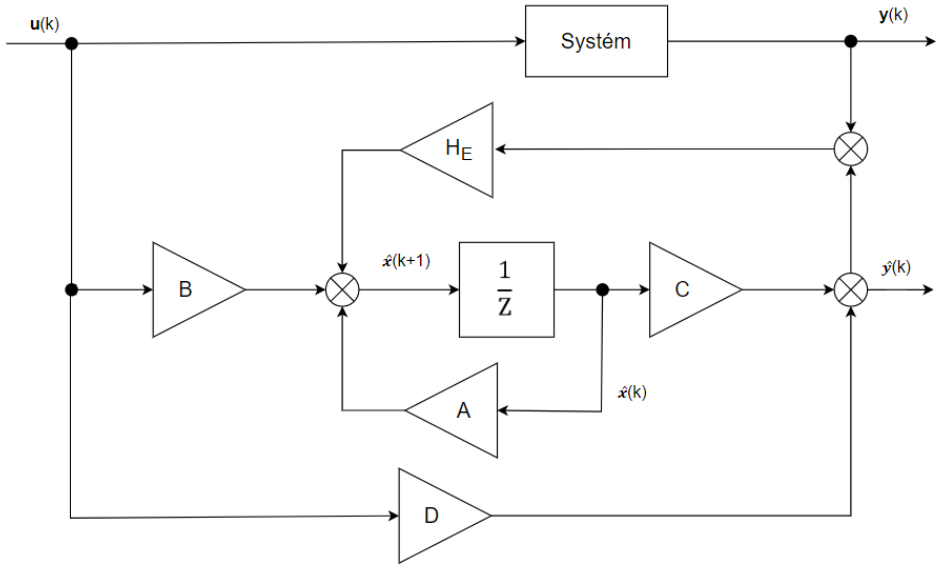
Estimátory stavu nacházejí uplatnění v široké škále aplikací, mezi které patří robotika, letectví, navigace, procesní řízení a mnoho dalších. Tyto aplikace vyžadují odhad stavů systému, které nemohou být přímo měřeny, ale jsou kritické pro řízení nebo monitorování systému.

Hlavní výhodou použití estimátoru stavu je možnost odhadnout vnitřní stavy systému, které nejsou přímo měřitelné. Nicméně, i u estimátorů stavu lze narazit na několik problémů. První je požadavek na přesný model systému pro odhad stavů. Pokud model není dostatečně přesný, odhady stavů mohou být rovněž nepřesné, což může vést k chybám v řízení (Strejc, 1978).

Druhým problémem je, že některé typy estimátorů stavu, jako je Kálmánův filtr, vyžadují znalost šumových vlastností systému, které nemusí být vždy přesně známy. To může také vést k chybám v odhadech stavů.

1.3.1 Luenbergův deterministický estimátor

Luenbergerův estimátor byl oceňován pro svou srozumitelnost a účinnost, což ho činí vhodným pro praktické využití. Na rozdíl od Kálmánova filtru u něj není nutné mít znalosti o statistických charakteristikách šumu. Popis chování systému je založen na stavových rovnicích (*1.7*) a (*1.8*). Schéma tohoto estimátoru byl prezentován na obrázku 1.6. Aby byl výklad jasnější, vektory a matice v rámci estimátoru byly označeny znakem "^".



Obrázek 1.6 – Blokové schéma zapojení estimátoru

Ze znalosti schématu lze sestavit rovnice

$$\hat{\boldsymbol{x}}(k+1) = \mathbf{A}\hat{\boldsymbol{x}}(k) + \mathbf{B}\boldsymbol{u}(k) + \mathbf{H_E}\,[\boldsymbol{y}(k) - \hat{\boldsymbol{y}}(k)], \qquad (1.41)$$

$$\hat{\boldsymbol{y}}(k) = \boldsymbol{C}\hat{\boldsymbol{x}}(k) + \boldsymbol{D}\boldsymbol{u}(k), \qquad (1.42)$$

kde $\boldsymbol{x}$ – vektor stavových veličin,

$\mathbf{A}$ – matice systému,

$\mathbf{B}$ – matice buzení systému,

$\mathbf{C}$ – výstupní matice systému,

$\mathbf{D}$ – přímá přenosová matice,

$\mathbf{H_E}$ – matice estimátoru zpětné vazby,

$\boldsymbol{y}$ – vektor výstupních veličin.

Dosazením rovnice (*1.42*) do rovnice (*1.41*) a následné matematické úpravě lze získat

$$\hat{\boldsymbol{x}}(k+1) = \underbrace{(\mathbf{A} - \mathbf{H_E C})}_{\mathbf{A_E}}\hat{\boldsymbol{x}}(k) + \underbrace{(\mathbf{B} + \mathbf{H_E D})}_{\mathbf{B_E}}\boldsymbol{u}(k) + \mathbf{H_E}\boldsymbol{y}(k), \qquad (1.43)$$

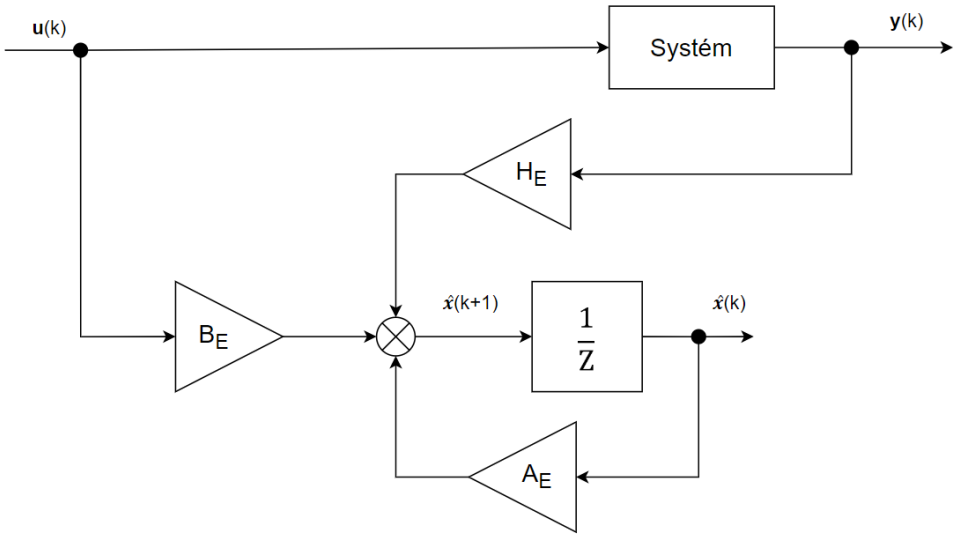
kde $\mathbf{A_E}$ – matice estimátoru,

$\mathbf{B_E}$ – matice buzení estimátoru.

Rovnici (*1.43*) jde přepsat do přehlednějšího tvaru

$$\hat{\boldsymbol{x}}(k+1) = \mathbf{A_E}\hat{\boldsymbol{x}}(k) + \mathbf{B_E}\boldsymbol{u}(k) + \mathbf{H_E}\boldsymbol{y}(k) \qquad (1.44)$$

Ze vztahu (*1.44*) lze udělat modifikované schéma zapojení regulátoru toto schéma je zobrazeno na obrázku 1.7.



Obrázek 1.7 – Modifikované schéma estimátoru

Matice estimátoru lze určit ze vztahu

$$\mathbf{A_E} = \mathbf{A} - \mathbf{H_E}\mathbf{C}, \qquad (1.45)$$

$$\mathbf{B_E} = \mathbf{B} + \mathbf{H_E}\mathbf{D}, \qquad (1.46)$$

vektor $\mathbf{H_E}$, je zvolen tak, aby vlastní čísla matice $\mathbf{A_E}$ byla umístěna uvnitř jednotkové kružnice

$$det(\lambda\mathbf{I} - \mathbf{A_E}) = det(\lambda\boldsymbol{E} - \mathbf{A} + \mathbf{H_E}\mathbf{C}) = (\lambda - \lambda_1)(\lambda - \lambda_2)\ldots, \qquad (1.47)$$

kde $\mathbf{I}$ – jednotková matice.

Pro zajištění stability diskrétního Luenbergerova estimátoru je nezbytné, aby vlastní čísla matice zesílení estimatoru byly uvnitř jednotkové kružnice, což zajišťuje jejich asymptotickou stabilitu.

Návrh estimátoru lze brát jako duální úlohu k návrhu LQ regulátoru popsaného v kapitole 1.2, z důvodu matematických a strukturních podobností mezi těmito dvěma procesy.

Návrh estimátoru je pojat jako duální úloha k návrhu LQ regulátoru a je odvozen na základě minimalizace kvadratického kritéria (*1.48*) s ohledem na **y**(*k*) z pohledu estimátoru vstupu.

$$J = \sum_{k}^{\infty} [\hat{\mathbf{x}}^{\mathbf{T}}(k)\mathbf{Q}(k)\hat{\mathbf{x}}(k) + \mathbf{y}^{T}(k)\mathbf{R}(k)\mathbf{y}(k)], \qquad (1.48)$$

kde $\hat{\mathbf{x}}$ – estimovaný vektor stavových veličin,

Q, R – váhové matice kritéria,

y – vektor výstupních veličin.

Řešení je opět prezentováno maticovou Riccatiho rovnici dle (*1.26*). S využitím Riccatiho rovnice můžeme následně odvodit:

$$\mathbf{H_E} = (\mathbf{A}^{T}\mathbf{PC})^{T}(\mathbf{C}^{T}\mathbf{PC} + \mathbf{R})^{-1}, \qquad (1.49)$$

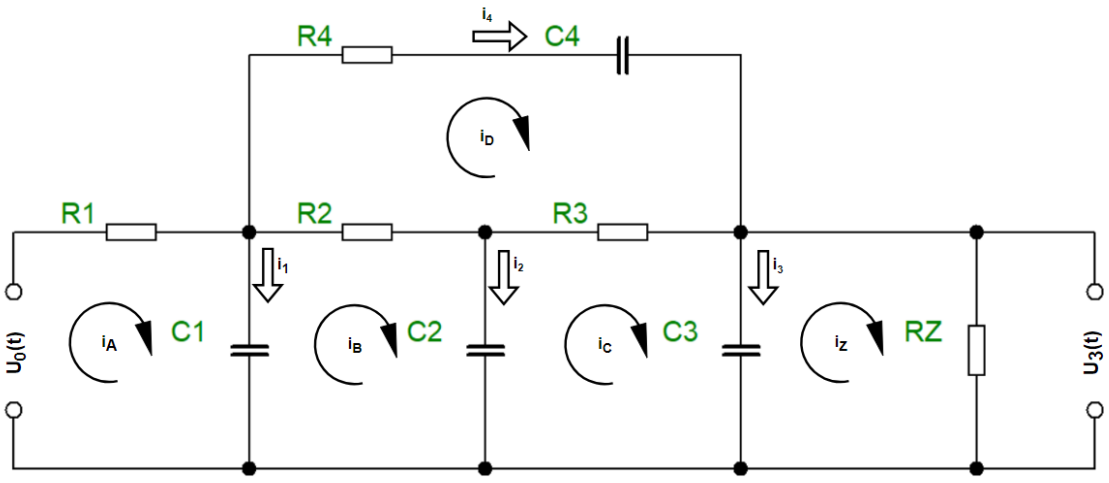
kde **P** – řešení diskrétní Riccatiho rovnice pro daný systém.

2 PRAKTICKÁ ČÁST

Praktická část se věnuje uplatnění teoretických principů v reálném prostředí. Regulovaná soustava je popsána společně s jejím matematickým modelem. Hardwarové platformě Arduino Due je věnována pozornost, neboť je využívána pro implementaci. MATLAB je prezentován jako esenciální softwarový nástroj, přičemž speciální důraz je kladen na jeho grafické uživatelské rozhraní a komunikaci. Dále je provedena řada simulací, aby bylo zjištěno, jaký vliv mají parametry LQ regulátoru na výslednou regulaci. V závěru praktické části je demonstrován návrh GUI programu a následné zhodnocení experimentů na reálné soustavě s porovnáním simulací.

2.1 POPIS REGULOVANÉ SOUSTAVY A MATEMATICKÝ MODEL

Regulovaná soustava je tvořena z pasivních součástek rezistorů a kondenzátorů, které jsou osazeny na nepájivém poli. Systém disponuje jedním vstupem a výstupem, což ho klasifikuje jako SISO typ. Vizualizace zapojení je na obrázku 2.1. Struktura zapojení vychází ze čtyřnásobného RC článku. Vzhledem k tomu, že se v zapojení vyskytují pouze pasivní prvky, není vyžadováno dodatečné napájení pro tyto komponenty.



Obrázek 2.1 – Schéma zapojení regulované soustavy

Pro získání spojitého dynamického matematického modelu v podobě diferenciálních rovnic bylo použito analytického přístupu. Pomocí fyzikálních zákonů, které se standardně využívají v elektrotechnice. Konkrétně se jedná o Kirhoffovy zákony a metodu smyčkových

proudů. Matematický model byl již odvozen v předešlých prací – např. (Jelínek, 2023), a proto je uvedena jen zkrácená verze.

Odvození matematického modelu

$$
\begin{aligned}
u_0 &= R_1 i_A + \frac{1}{C_1}\int_0^t (i_A - i_B)dt, \\
0 &= R_2(i_B - i_D) + \frac{1}{C_2}\int_0^t (i_B - i_C)dt + \frac{1}{C_2}\int_0^t (i_B - i_A)dt\,, \\
0 &= R_3(i_C - i_D) + \frac{1}{C_3}\int_0^t (i_C - i_Z)dt + \frac{1}{C_2}\int_0^t (i_C - i_B)dt\,, \\
0 &= R_4 i_D + \frac{1}{C_4}\int_0^t i_D dt + R_2(i_D - i_B) + R_3(i_D - i_C)\,, \\
0 &= R_Z i_Z + \frac{1}{C_3}\int_0^t (i_Z - i_C),
\end{aligned}
\qquad (2.1)
$$

kde R – odpor, Ω,

C – kapacita, F,

u_0 – napájecí napětí, V,

$i_{A\text{-}Z}$ – smyčkové proudy, A,

t – spojitý čas, s.

Obecné vztahy

$$
u = \frac{1}{C}\int i\,dt \;\; a \;\; i = C\frac{du}{dt}, \qquad (2.2)
$$

kde C – kapacita, F,

u – napětí v kondenzátorech, V,

i – proud protékající kondenzátorem, A.

Vztah mezi smyčkovými proudy a skutečnými proudy kondenzátorů viz Obrázek 2.1

$$\begin{aligned} i_1 &= i_A - i_B \;\rightarrow\; i_A = i_1 + i_2 + i_3 + i_Z, \\ i_2 &= i_B - i_C \rightarrow\; i_B = i_2 + i_3 + i_Z, \\ i_3 &= i_C - i_Z \;\rightarrow\; i_C = i_3 + i_Z, \\ i_4 &= i_D \;\rightarrow\; i_D = i_4 \\ i_Z &= \frac{1}{R_Z} u_3, \end{aligned} \qquad (2.3)$$

kde $i_{1\text{-}4}$ – proudy v různých větvích, A,

$i_{\text{A-Z}}$ – smyčkové proudy, A,

u_3 – napětí na zátěži, V,

R_Z – odpor zátěže, Ω.

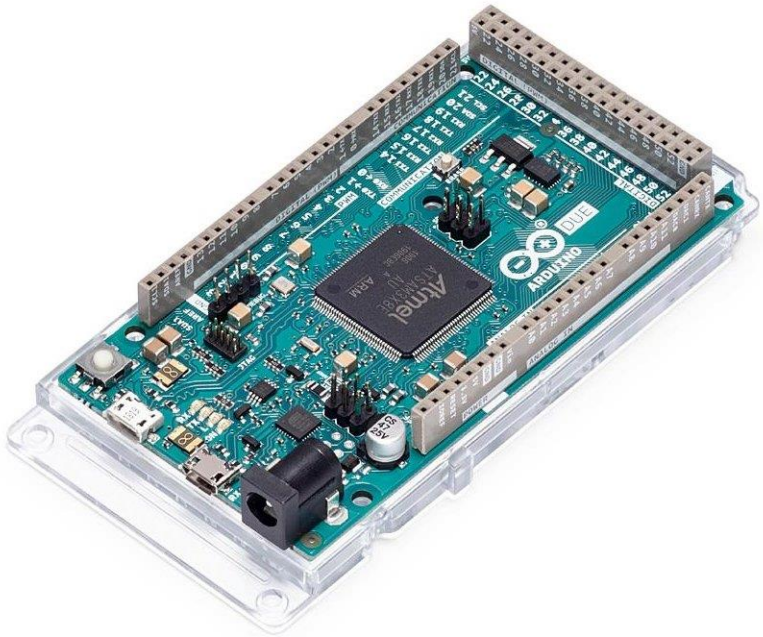
Matice modelu budou stanoveny následovně

$$A = \begin{bmatrix} -\frac{1}{C_1}\left(\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_4}\right) & \frac{1}{C_1 R_2} & \frac{1}{C_1 R_4} & \frac{1}{C_1 R_4} \\ \frac{1}{C_2 R_2} & -\frac{1}{C_2}\left(\frac{1}{R_2}+\frac{1}{R_3}\right) & -\frac{1}{C_1 R_3} & 0 \\ \frac{1}{C_3 R_4} & \frac{1}{C_3 R_3} & -\frac{1}{C_3}\left(\frac{1}{R_3}+\frac{1}{R_4}+\frac{1}{R_Z}\right) & -\frac{1}{C_4 R_4} \\ \frac{1}{C_4 R_4} & 0 & -\frac{1}{C_4 R_4} & -\frac{1}{C_4 R_4} \end{bmatrix} \qquad (2.4)$$

$$\boldsymbol{B} = \begin{bmatrix} \frac{1}{C_1 R_1} \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad C = [0 \quad 0 \quad 1 \quad 0] \qquad D = 0 \qquad (2.5)$$

2.2 ARDUINO DUE

Arduino Due je deska s mikrokontrolerem, která byla navržena, aby poskytovala flexibilní a výkonnou platformu pro velkou škálu aplikací. Je založena na procesoru Atmel AM3X8E ARM Cortex-M3, který disponuje taktovací frekvencí 84 MHz a podporuje velkou škálu komunikačních protokolů například Ethernet, CAN (Controller Area Network) a USB. Na desce je umístěno 54 digitálních vstupně-výstupních pinů, 12 analogových vstupů a 2 analogové výstupy, společně s řadou dalších prvků, včetně USB konektoru a napájecího konektoru. Díky těmto prvkům je Arduino DUE považováno za ideální pro projekty sahající od automatizace přes robotiku až po záznam dat. Mezi klíčové vlastnosti Arduina DUE patří kompatibilita s celou řadou programovacích jazyků a softwarových nástrojů. Deska je podporována programovacím prostředím Arduino IDE, které poskytuje přehledné uživatelské rozhraní založené na jazyce C/C++, kromě toho dalšími programovacími prostředími jako je třeba Atmel studio a knihovny ARM CMSIS. Arduino je schopno díky frekvenci procesoru 84MHz a 512 kB Flash paměti zpracovávat data i v reálném čase např. rozpoznávání obrazu atd. Díky těm to vlastnostem je Arduino ideální volbou pro pokročilejší uživatelé, kteří vyvíjejí složitější projekty. Arduino Due je zobrazeno na Obrázek 2.2 (Arduino, 2023).



Obrázek 2.2 – Arduino Due (ARDUINO DUE, 2023)

2.3 MATLAB

MATLAB byl vytvořen jako programovací jazyk a softwarová platforma v roce 1984 firmou The Mathworks, Inc. ve Spojených státech. Je to robustní nástroj určený pro inženýrské a vědecké výpočty a vizualizaci dat. Uživatelské prostředí MATLABu, které je přívětivé, zvládá maticové výpočty, grafiku a mnohé další. Díky schopnostem řešit širokou škálu problémů, zejména v oblasti matematiky, fyziky, měření a zpracování dat, je MATLAB vyhledáván. Jeho rozsah může být rozšířen díky toolboxům, pomocí kterých by bylo přidáno mnoho dalších užitečných funkcí, jako je například řízení v reálném čase nebo komunikace s Arduinem (Matlab, 2023).

Silnou stránkou MATLABu je jeho kompatibilita s dalšími programovacími jazyky a nástroji, včetně C++ a Java. Uživatelům je tím umožněno kombinovat výhody jednotlivých jazyků a tak jsou jejich projekty vytvořeny efektivně a elegantně.

Navíc, MATLAB je podporován velkou a aktivní základnou uživatelů a vývojářů. Bohaté zdroje, včetně dokumentace a uživatelských fór, jsou poskytovány touto komunitou, což může novým uživatelům pomoci rychleji zvládnout práci s MATLABem

2.3.1 Grafické uživatelské rozhraní

Interaktivní aplikace mohou být vytvořeny pomocí GUI (Graphical User Interface) v MATLABu, čímž je eliminována nutnost psaní velkého množství kódu. Díky MATLAB GUI je uživatelům umožněna snadná adaptace na nástroj, a zároveň je jim poskytnuto rozhraní pro vizualizaci a interakci s daty.

MATLAB GUI je tvořen mnoha komponentami, mezi kterými jsou ovládací prvky jako tlačítka, posuvníky, zaškrtávací políčka, textové pole a další. Uživatelům je tedy umožněno interagovat s daty a algoritmy, které jsou v aplikaci implementovány, pomocí těchto ovládacích prvků (Matlab, 2023).

Existují dva hlavní přístupy, jak může být vytvořeno GUI v MATLABu. První z nich zahrnuje programování pomocí kódu, při kterém jsou prvky GUI vytvářeny a manipulovány pomocí funkcí a skriptů MATLABu. Tento způsob je vhodný pro ty, kteří jsou seznámeni s MATLABem a jeho programovacím jazykem.

Druhý způsob využívá nástroj App Designer jako vizuální prostředí pro vytváření a úpravu grafických uživatelských rozhraní (GUI). Ovládací prvky mohou být snadno přetahovány na plátno a jejich vlastnosti mohou být intuitivně upravovány pomocí funkce

"drag and drop". Rychlé a jednoduché vytváření GUI může být tímto způsobem značně usnadněno, zejména pro ty, kteří nejsou zkušenými programátory. Zvláštní pozornost by měla být věnována pečlivému plánování a kvalitnímu návrhu rozhraní (Matlab, 2023).

2.3.2 Komunikace přes sériovou linku

Jednou z možností MATLABu je podpora komunikace po sériové lince, která umožňuje přímý kontakt s různými zařízeními. Ta mohou obsahovat hardwarové prvky, jako jsou modemy, jednotky GPS a mikrokontrolery. V technických a vědeckých oborech se tato funkce často využívá pro sběr dat, ovládání zařízení a další účely.

Sériová komunikace je oblíbeným způsobem přenosu dat u zařízení s omezenými zdroji, jako je Arduino. Tento přístup, který posílá data bit po bitu přes sériové připojení, je méně náročný na hardware než paralelní přenos, kde se bitů přenáší více najednou.

Existují různé způsoby připojení MATLABu k Arduinu využívající sériovou komunikaci. Jeden z těchto přístupů, MATLAB Support Package for Arduino Hardware, umožňuje jednoduše číst a zapisovat data do Arduina prostřednictvím sériového připojení.

Uživatelé MATLABu mohou s využitím tohoto balíčku programovat Arduino přímo z MATLABu. To nabízí dynamické vývojové prostředí, které usnadňuje experimentování a řešení problémů. Díky této metodě není nutné nahrávat do Arduina vlastní program, protože o vše se postará MATLAB (Matlab, 2023).

V případě potřeby sofistikovanějších funkcí, jako je například knihovna DueTimer pro ovládání hardwarových časovačů na Arduino Due, je však třeba do Arduina nahrát vlastní software. Poté se posílají zprávy za účelem komunikace. V MATLABu je třeba nejprve vytvořit a otevřít objekt sériového připojení, teprve potom přes něj může probíhat jakákoli komunikace. Poté je možné číst a zapisovat data.

Komunikace s měřící jednotkou je realizována přes virtuální sériový port emulovaný nad USB rozhraním. V Arduinu je nahrán program, který zajišťuje:

- Obousměrnou sériovou komunikaci přes USB linku využívající zprávy s jednoduchým ASCII protokolem
- Periodické měření až 8 napětí s volitelnou periodou
- Nastavení 2 výstupních napětí po příjmu příslušné zprávy
- Odeslání zprávy s aktuálně změřenými hodnotami napětí jako odpověď na příjem příslušné zprávy
- Volitelně periodické posílání zpráv s měřenými napětími

Jako vzor komunikace s programem v Arduinu mi byly poskytnuty funkce pro realizaci komunikace v MATLAB skriptu:

- `ArIniM`: inicializaci komunikace s Arduinem přes sériový port. Objekt sériového portu s určenými parametry je vytvořen, poté je získáváno aktuální nastavení programu v Arduina, (verze programu, režim, perioda měření a počet měřených kanálů). Dále je počáteční hodnota výstupního napětí a jsou vyčteny aktuální hodnoty analogových vstupů. Dále aktivuje obsluhu události příjmu zprávy ze sériové linky, která umožňuje zpracování přijaté zprávy na pozadí.
- `ArParSet`: nastavení parametrů pro Arduino. Umožňuje nastavit interval vzorkování a počet měřených kanálů. Po odeslání příkazů na Arduino je kontrola odpovědí zařízení, aby bylo zajištěno správné nastavení.
- `ArAOSetM`: nastavení výstupních napětí na dvou analogových výstupech. Po odeslání příkazů na Arduino je odpověď zařízení kontrolována, aby bylo zajištěno správné nastavení.
- `ArAIGetM`: vyčtení posledního měření analogových vstupů (AI) z Arduina v jednom ze dvou režimů. V režimu jednorázového čtení je do Arduina odeslán požadavek na měřená data a čeká se na odpověď. V režimu cyklického čtení jsou měřená napětí periodicky vysílána Arduinem s nastaveným intervalem a jsou v MATLABu na pozadí přijímána. Při požadavku na měřená napětí jsou okamžitě vyčtena posledně přijatá data (pokud jsou k dispozici) nebo se čeká (s kontrolou na překročení nastaveného intervalu – timeout) na příjem dalších dat. Režim cyklického čtení umožňuje synchronizaci provádění programu v MATLABu s příjmem zpráv z Arduina.

2.4 VERIFIKACE MATEMATICKÉHO MODELU

Nejprve bylo nutné provést měření průběhu reakce reálné soustavy, aby mohlo být umožněno srovnání mezi matematickým modelem popsaným v kapitole 2.1 a skutečným zařízením. Tato úloha byla realizována prostřednictvím samostatného programu.

Nejprve byla na příslušném portu inicializována komunikace s Arduinem. Poté byly stanoveny základní parametry experimentu, jako je vzorkovací doba, reprezentující čas mezi jednotlivými měřeními, a počet měřených kanálů.

Po dokončení této přípravné fáze bylo připojení k Arduinu zprovozněno a základní parametry byly nastaveny. Pomocí funkcí, které jsou popsány v kapitole 2.3.2.

```
s = ArIniM(port);
% Nastavení vzorkovací frekvence a počtu kanálů
ArParSetM(s, Ts*1e6, P_ka);
% Počáteční hodnoty
u0 = 1; u1 = 1;
ArAOSetM(s, [u0, 1]);
```

V hlavní části programu byla provedena série skokových změn napětí. Tyto změny byly navrženy a následně byly generovány prostřednictvím funkce.

```
% Funkce pro generování skokových změn napětí
function U_Napeti = G_V_Krok(Napeti, Vzo)
    U_Napeti = [];
    for i = 1:length(Napeti)
        U_Napeti = [U_Napeti, ones(1, Vzo)*Napeti(i)];
    end
end
```

Na Arduino byla každá skoková změna napětí aplikována a reakce zařízení byla systematicky zaznamenávána programem až do doby, kdy byl experiment ukončen.

Po ukončení všech měření byla vizualizace a analýza získaných dat. Získané reakce byly graficky znázorněny, což umožnilo vizuální vyhodnocení celého experimentu. K zachování dat pro další analýzy byly informace uloženy do souboru.

2.4.1 Zpřesnění matematického modelu

V rámci zdokonalení simulací a návrhu Estimátoru s LQ regulátorem bylo rozhodnuto o zpřesnění existujícího modelu systému.

Proces, kdy se dohledávají parametry modelu tak, aby se průběh výstupu modelu co nejvíce shodoval s naměřeným průběhem výstupu (pro stejný průběh vstupu) se nazývá experimentální identifikace. K tomuto účelu byl využit MATLAB.

Po rozšíření MATLABu o Control System Toolbox jsou poskytovány funkce *ss, tf* a *zpk*. Pomocí těchto funkcí mohou být vytvořeny objekty popisující chování lineárního dynamického systému s časově neproměnnými parametry (LTI). Systém může být definován různými parametry v závislosti na funkci. Pro *ss* je systém charakterizován maticemi stavového popisu, pro *tf* polynomy čitatele a jmenovatele přenosu a pro *zpk* zesílením, nulami a póly přenosu. V našem případě bude použita funkce *ss*, jelikož máme matematický model popsán rovnicemi (*2.4*) a (*2.5*).

Na začátku procesu bylo pracovní prostředí programu vyčištěno a všechna předchozí data byla odstraněna. Data s informacemi o napětí, měřených hodnotách a časovém průběhu byla poté načtena z externího souboru, která byla naměřena v předešlém programu.

Nominální matice pro model LTI jsou definovány rovnicemi (*2.4*) a (*2.5*). Na základě toho byl vytvořen LTI objekt a podle daného vztahu byly stanoveny počáteční podmínky (*2.6*). Pro provedení měření by měla být soustava v ustáleném stavu. Následně je, s využitím nominálních hodnot původních matic, výstup ideálního modelu vypočítán příkazem `lsim`, který zahrnuje počáteční podmínky (Opršal, 2021. Jelínek, 2023).

$$\mathbf{x}_0 = \mathbf{A}^{-1}\mathbf{B}u_0, \tag{2.6}$$

kde $\mathbf{x}_0$ – vektor počátečních podmínek,

$\mathbf{A}$ – matice systému,

$\mathbf{B}$ – matice buzení systému,

u_0 – akční veličina.

Pro optimalizaci s cílem určit korekci devíti parametrů (hodnot komponent R1 až Rz a C1 až C4) tyto parametry ovlivňují konečné matice systému. Kvůli značným rozdílům v hodnotách komponent byl zaveden korekční vektor kor. Tento vektor, skládající se z devíti prvků, je použit k násobení hodnot součástek, což usnadňuje sjednocení řádu parametrů během optimalizace (Opršal, 2021. Jelínek, 2023)

Tento parametr byl původně nastaven na hodnotu 1 pro všechny prvky modelu. Pro kalibraci modelu k dosažení lepší shody mezi simulovaným a skutečným chováním byla použita optimalizační funkce `fminsearch`. Funkce se zaměřila na hledání minima funkce chyby mezi simulací a měřenými daty, což bylo realizováno metodou nejmenších čtverců. Kritérium pro tuto funkci je rovnice (*2.7*). Cílem této operace bylo nastavit hodnoty parametru kor tak, aby simulovaný průběh co nejvíce odpovídal reálně naměřeným datům (Opršal, 2021. Jelínek, 2023).

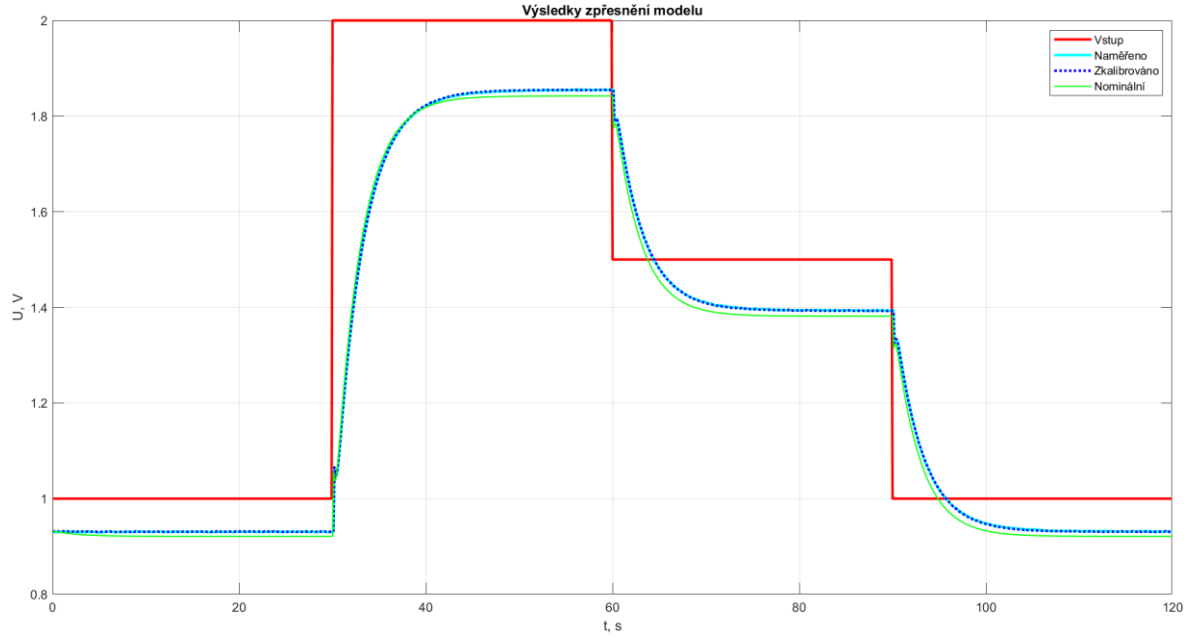
$$J = \left(\mathbf{y}_m - \mathbf{y}(\mathbf{kor})\right)^T \left(\mathbf{y}_m - \mathbf{y}(\mathbf{kor})\right), \tag{2.7}$$

kde $\mathbf{y_m}$ – měřený vektor,

$\mathbf{y}_{\mathrm{kor}}$ – výstup ideálního modelu.

Po úspěšné kalibraci byly upravené matice získány a následně byla provedena nová simulace s těmito zkorigovanými hodnotami. Tato simulace následně lépe odpovídala skutečným datům (Opršal, 2021. Jelínek, 2023).

Finální krok celého procesu představovalo vizuální srovnání naměřených dat a dvou simulací – původní a upravené – v jednom grafu. Díky tomu byla rychle zhodnocena úspěšnost kalibrace a byla vyhodnocena přesnost upraveného modelu. Srovnání naměřených dat a simulací je zobrazeno na obrázku 2.3. Bylo zjištěno, že model byl již celkem přesný, a byla provedena jen menší úprava.



Obrázek 2.3 – Výsledky zpřesnění modelu

V tabulce 2.2 byly zobrazeny hodnoty vektoru kor, které byly vypočítány pomocí experimentální identifikace popsané výše. Identifikace byla prováděna s intervalem měření vzorků 0,1 s.

Tabulka 2.1 – Nominální hodnoty součástek

R1[kΩ]	R2[kΩ]	R3[kΩ]	R4[Ω]	Rz[kΩ]	C_1[µF]	C_2[µF]	C_3[µF]	C_4[µF]
1,6	1,6	1,6	47	56	47	820	47	47

Tabulka 2.2 – Hodnoty vektoru kor

$kor_{(1)}$	$kor_{(2)}$	$kor_{(3)}$	$kor_{(4)}$	$kor_{(5)}$	$kor_{(6)}$	$kor_{(7)}$	$kor_{(8)}$	$kor_{(9)}$
1,6	1,6	1,6	47	56	47	820	47	47

Po roznásobení nominálních hodnot součástek z tabulky 2.1 s příslušným vektorem kor byly získány upravené parametry součástek viz tabulka 2.3. Ačkoliv v MATLABu se pracuje s více desetinnými místy, hodnoty kor i výsledné hodnoty součástek byly zde zaokrouhleny pro lepší přehlednost.

Tabulka 2.3 – Korigované parametry součástek

R1[kΩ]	R2[kΩ]	R3[kΩ]	R4[Ω]	Rz[kΩ]	C_1[µF]	C_2[µF]	C_3[µF]	C_4[µF]
1,564	2,666	2,096	44,95	39,16	20,70	1004	56,59	46,28

2.5 SIMULACE A REGULACE

Před aplikací LQ regulátoru v kombinaci s deterministickým estimátorem na skutečný systém byla v MATLABu provedena simulace. Tato simulace kombinuje tři klíčové komponenty: Luenbergův deterministický estimátor, LQ regulátor a integrační složku.

Luenbergův deterministický estimátor využívá dvou klíčových rovnic k odhadu stavů systému z měřeného výstupu a řídicího signálu. Konkrétně byly aplikovány rovnice (*1.41*) pro aktualizaci odhadu stavu na základě aktuálního odhadu stavu, vstupního řídicího signálu a rozdílu mezi skutečným a odhadovaným výstupem. Rovnice (*1.42*) pak popisuje, jak je výstup systému odhadnut z aktuálního odhadu stavu a vstupního řídicího signálu.

Co se týče rozšíření matic pro integrační složku, matice systému byly rozšířeny dle rovnice (*1.33*) tak, aby zahrnovaly integrátor, který akumuluje regulační odchylku. Díky tomuto rozšíření je možné integrovat regulační odchylku v průběhu času, jak je popsáno rovnicí (*1.34*).

K výpočtu vektoru zesílení **K**, který je nezbytný pro následné určení akční veličiny **u**(*k*), byla v MATLABu využita funkce `dlqr`. Tato funkce vrací optimální zesílení na základě matic diskrétního stavového popisu rozšířeného systému a váhových matic.

```
[K,S,e] = dlqr(Aaug, Baug, Q2, R);     % Výpočet LQ regulátoru
Ki = K(1:nx);                          % Rozdělení vektoru zesílení
Ki_aug = K(end);                       % Rozdělení vektoru zesílení
```

Pomocí funkce `dlqr` byl získán rozšířený vektor zesílení **K** rovnice (*1.39*), proto bylo nutné tento vektor rozdělit dle rovnice (*1.40*) pro jeho následnou aplikaci. Při návrhu LQ regulátoru byly využity hodnoty **Q** a **R** zobrazeny níže.

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad \mathbf{R} = [0{,}3].$$

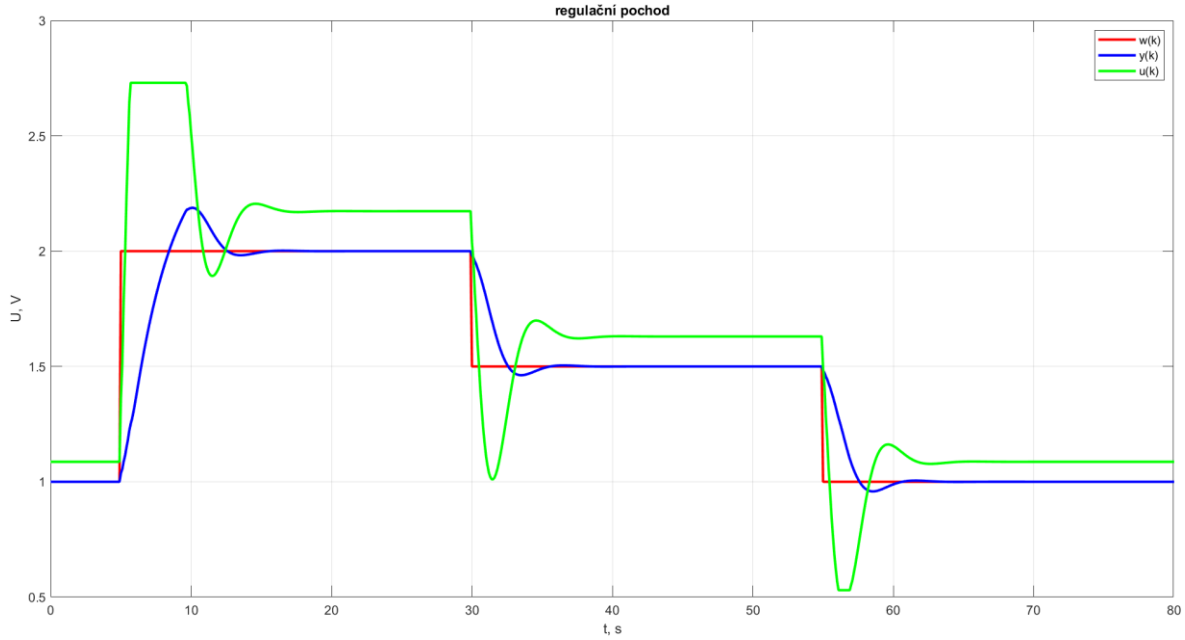
Hodnoty vektoru zesílení **K** jsou následovné

$$\mathbf{K} = \left[\underbrace{0{,}3355 \quad 1{,}8424 \quad 0{,}0952 \quad 0{,}1757}_{K_i} \; \underbrace{-0{,}0606}_{K_{iaug}} \right]$$

Po dokončení všech potřebných kroků bylo přistoupeno k samotné simulaci. V hlavní simulační smyčce byl akční zásah regulátoru vypočítán programem na základě odhadovaných stavů a regulační odchylky. Akční zásah byl omezen v rozsahu 0,53 až 2,73V, což jsou limity Arduina, v nichž může být akční zásah nastaven.

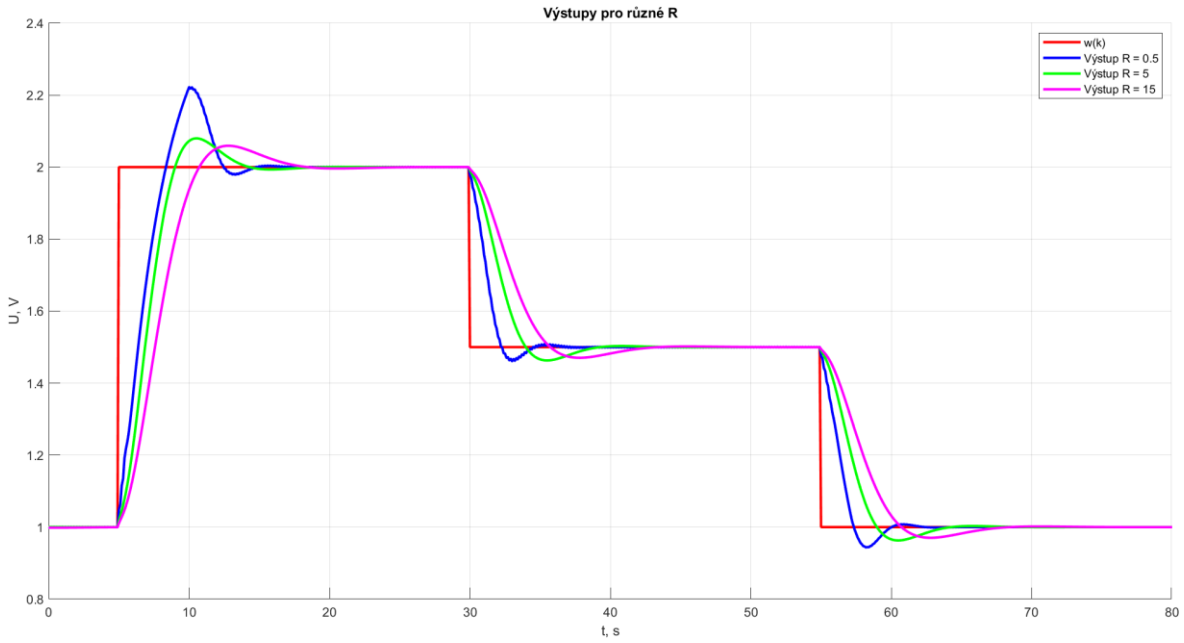
```
% Simulace LQ
for k = 1:n
    % Výpočet regulační odchylky
    e = ref(k) - y;
    % Kumulativní součet regulačních odchylek
    e_int = e_int + e;
    % Výpočet akčního zásahu s využitím estimátoru
    u = -Ki * x_est - Ki_aug * e_int;
    % Omezení akčního zásahu
    u = min(max(u, 0.53), 2.73);
    % Simulace chování systému
    x = Aaug * x + Baug * u;  % Výpočet nového stavu systému
    y = Caug * x;             % Výpočet nového výstupu systému
    % Estimace
    x_est = Ae * x_est + Bd * u + He' * (y(1) - Cd * x_est);
    % Ukládání
    outputs(k) = y;
    controls(k) = u;
    errors(k) = e;
end
```

Po dokončení programu byla prezentována vizualizace ukazující cílovou hodnotu **w**(*k*), intervenci **u**(*k*) a odpověď systému **y**(*k*). Graf s časovým krokem 0,1s je znázorněn na obrázku 2.4

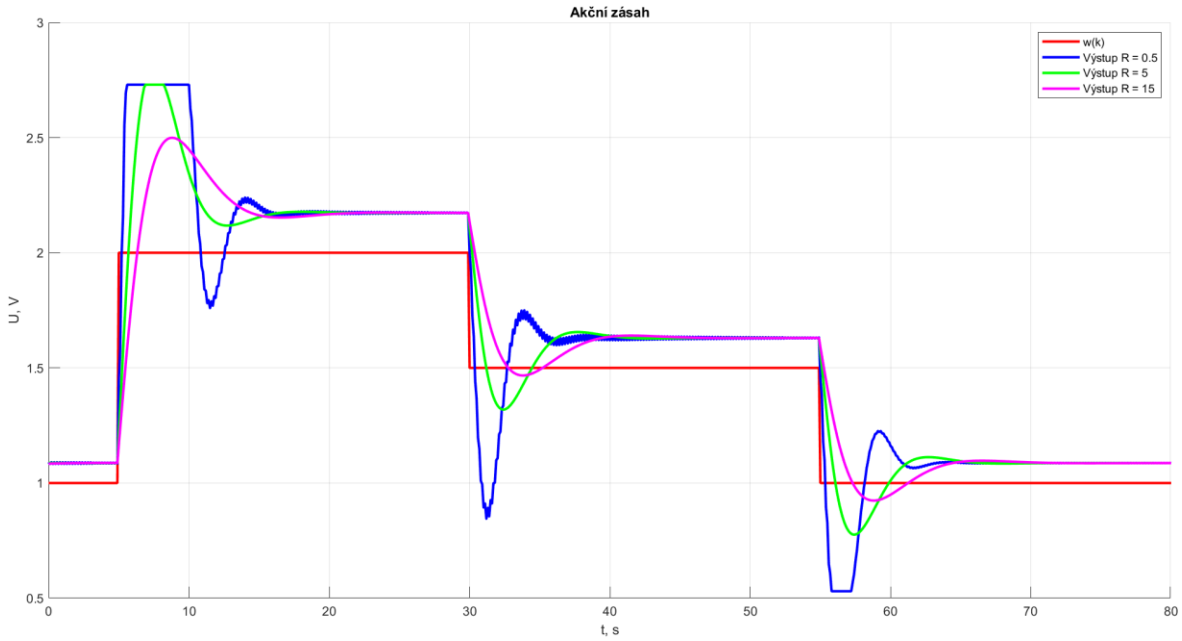


Obrázek 2.4 – Simulace regulačního pochodu

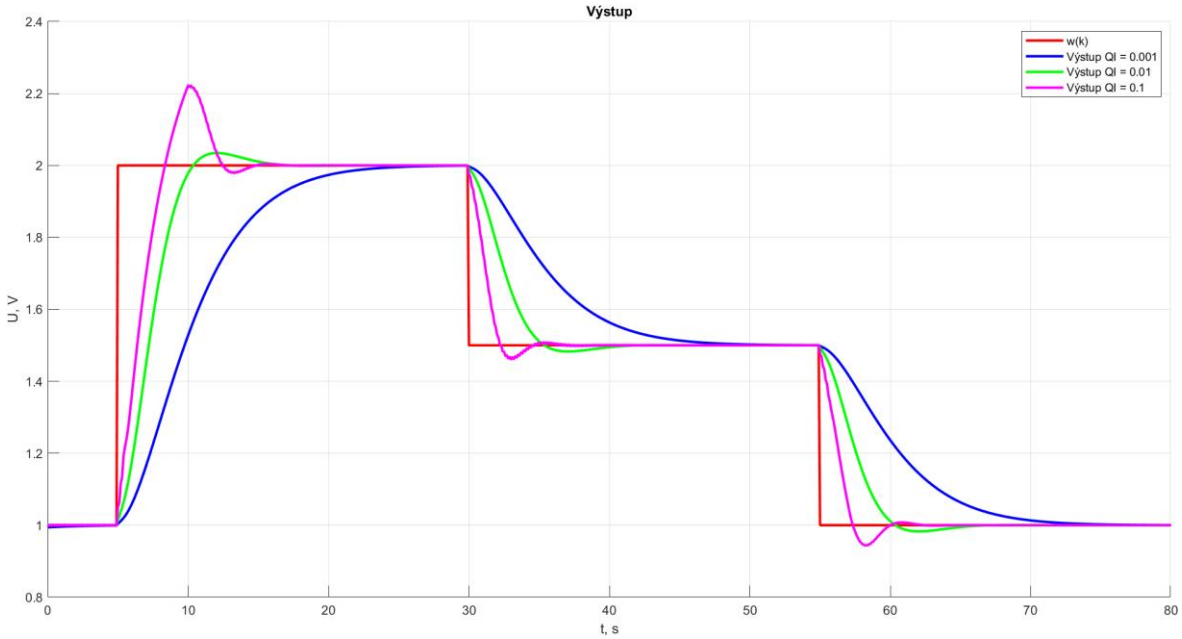
Byly provedeny simulace s různými nastaveními parametrů, aby bylo zjištěno, jak na tyto změny systém reaguje. Na obrázku 2.5 je ukázáno, jak se výstup mění v závislosti na změně hodnoty **R** při konstantní **Q**. Na obrázku 2.6 je zobrazen akční zásah pro průběhy které jsou zobrazeny na obrázku 2.5. Na obrázku 2.7 je zobrazen graf s vlivem členu matice **Q** odpovídajícího integrálu regulační odchylky. Interval vzorkování je nastaven na 0,1s.



Obrázek 2.5 – Průběhy výstupní veličiny pro rozdílné R s konstantní Q



Obrázek 2.6 – Průběhy akční veličiny pro rozdílné R s konstantní Q



Obrázek 2.7 – Vliv členu matice Q odpovídajícího integrálu regulační odchylky

Během analýzy simulací LQ regulátoru v MATLABu, kde sledování žádaných hodnot bylo kombinováno s integrací regulační odchylky, byly identifikovány klíčové vlivy.

Zásadní vliv hodnoty **R** na regulační charakteristiky byl zaznamenán. Jak je vidět z grafu na obrázku 2.6, s narůstajícím **R** je akční zásah regulátoru považován za mírnější.

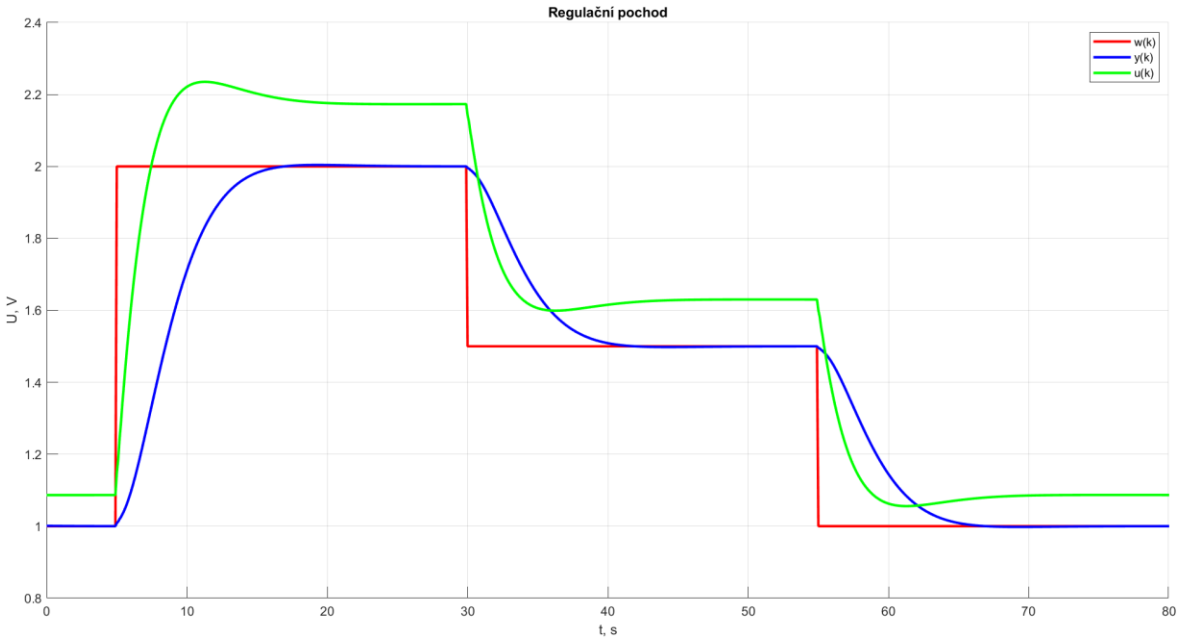
Ukazuje se, že při nízkých hodnotách **R** je reakce systému rychlá, avšak doprovázena větším překmitem, zatímco při vyšších hodnotách **R** jsou reakce považovány za mírnější bez výrazného překmitu.

Dále bylo pozorováno, jakým způsobem ovlivňuje integrační složka regulační pochod. Integrační složka je volena členem v matici **Q**. Bylo zjištěno, že když je tento člen nadměrně velký, může být způsobena nežádoucí oscilace, protože se od regulátoru očekává rychlá eliminace odchylek. Naopak, když je tento člen považován za příliš nízký, odchylka je eliminována pomalu a ačkoli systém neosciluje, regulační čas je považován za neúměrně dlouhý, jak je zobrazeno na obrázku 2.7.

Z těchto pozorování vyplývá, že nalezení vyváženého kompromisu mezi hodnotou **R** a integrační složkou matice **Q** je považováno za nezbytné pro dosažení optimálního výkonu regulátoru a splnění požadavků na dynamiku systému.

Na základě několika simulací byly určeny následující hodnoty jako doporučené pro LQ regulátor s vzorkovací frekvencí 0,1s. Výsledný průběh je zobrazen na obrázku 2.8.

$$\mathbf{Q}_{aug} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0{,}002 \end{bmatrix}; \qquad \mathbf{R} = [0{,}3].$$



Obrázek 2.8 – Regulační pochod s doporučenými parametry

2.6 NÁVRH GUI PROGRAMU

Základní doporučení při návrhu GUI

Pro ideální GUI v MATLABu by měl být základní layout navrhován tak, aby odrážel jednoduchost a intuitivnost. Doporučuje se vyvarovat se přeplnění mnoha prvky, což by uživatelské rozhraní učinilo nepřehledným. Mělo by být umožněno, aby uživatelé mohli rychle identifikovat a používat hlavní funkce bez zbytečného hledání nebo zmatku.

Při návrhu GUI by měl být kladen důraz na výběr barev, které by měly být snadno rozpoznatelné a neměly by způsobovat únavu očí. Měl by být navrhován kontrast mezi textem a pozadím tak, aby byla zajištěna dobrá čitelnost. Barvy by měly být vybírány s ohledem na jejich funkčnost, například červená by mohla signalizovat chybu nebo zastavení. Doporučuje se, aby všechny ovládací prvky v GUI byly plánovány s jasnými popisky. Když uživatel narazí na prvek s nejasným účelem, měl by být schopen rychle zjistit jeho funkci díky jeho popisku. Tato doporučení by měla snížit potřebu hledání nápovědy nebo čtení manuálů.

Pro posílení uživatelské zkušenosti by měla být v GUI začleněna interaktivita. Když je provedena nějaká akce, mělo by být poskytnuto zpětné vazby uživateli, ať už prostřednictvím zprávy, změny barvy prvku, nebo jiného vizuálního signálu. Tato zpětná vazba by měla uživatelům pomáhat pochopit, co se v aplikaci děje, a zda byla jejich akce úspěšná.

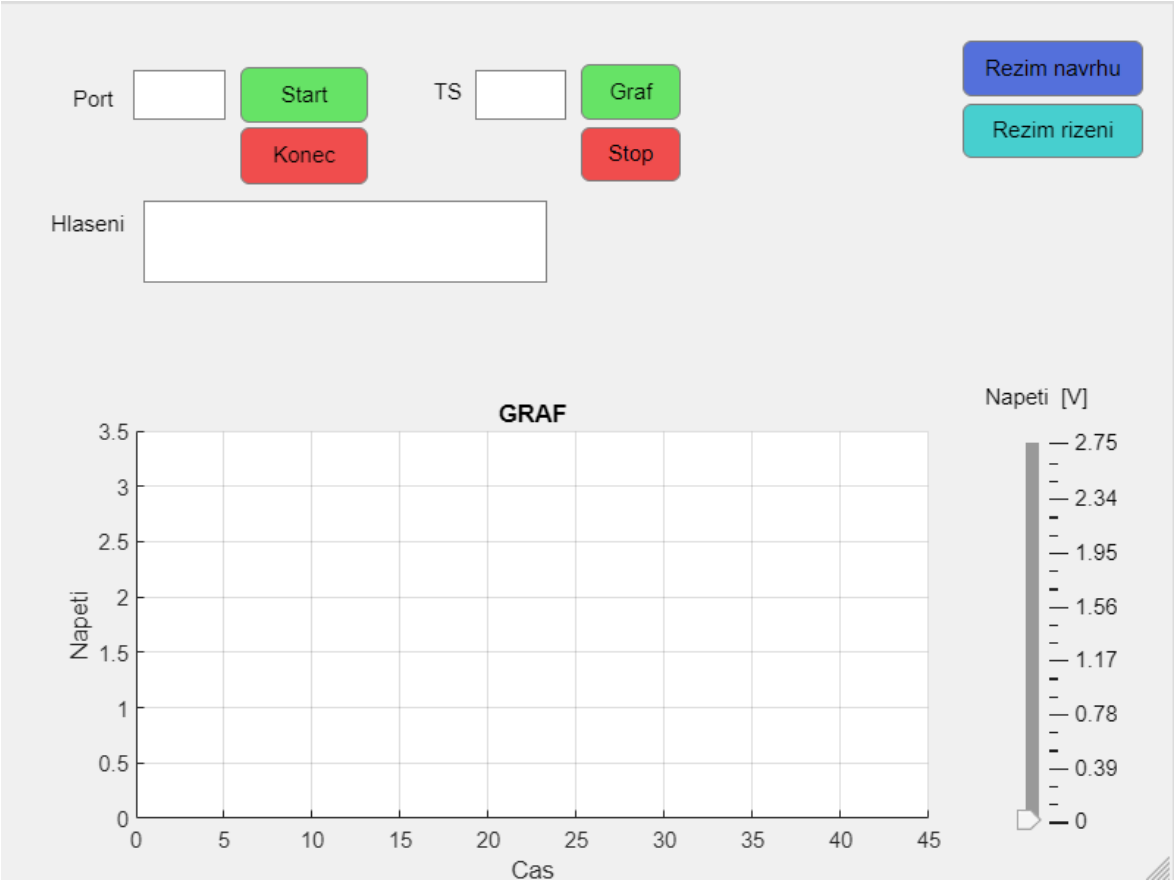
Kritéria na funkcionalitu GUI

- Uživateli bude nabídnuta možnost výběru komunikačního portu. Bude zajištěn přístup k tlačítkům pro zahájení a ukončení komunikace s Arduinem. Reakce Arduina na změny napětí bude možné ověřovat, což bude vizuálně zobrazeno.
- Bude umožněn návrh LQ regulátoru a estimátoru pro specifický model s volbou parametrů, jako jsou matice **Q** a **R**. Po dokončení návrhu bude uživateli prezentováno vektorové zesílení.
- Uživatel bude moci ověřit funkčnost navrženého LQ regulátoru a estimátoru s možností nastavení žádané hodnoty. V testovacím režimu budou zobrazeny aktuální a žádané hodnoty, a uživatel bude moci monitorovat reakce systému na různé zásahy.

Na základě stanovených kritérií bylo rozhodnuto o rozdělení GUI do tří samostatných obrazovek: základní režim, režim návrhu a režim řízení. V každé odrážce je reprezentováno, co by měla daná obrazovka umožňovat. Toto rozdělení bylo provedeno za účelem zachování jednoduchosti a přehlednosti jak obrazovky, tak kódu. Detaily jednotlivých GUI budou níže prezentovány.

2.6.1 GUI základní režim

Bylo představeno, jaké základní funkce by mělo GUI zvládat. V následující sekci bude pozornost věnována grafickému rozvržení a interakci GUI. Na obrázku 2.9 je základní režim GUI zobrazen, který lze rozdělit do čtyř pomyslných částí. První část byla určena k navázání komunikace, druhá k inicializaci vykreslování dat v grafu. Ve třetí části je umístěn graf s posuvníkem a čtvrtá část umožňuje přesun do dalších obrazovek.



Obrázek 2.9 – GUI základní režim

První oblast je určena pro navázání komunikace s Arduinem a případné ukončení. K tomuto účelu je využíváno textové pole s názvem "Port" a tlačítka "Start" a "Konec". Uživatelem je do textového pole zadán port, například COM3. Po zadání portu a stisknutí tlačítka "Start" se pokouší vytvořit sériový port emulovaný nad USB rozhraním. V případě úspěchu se v textovém poli "Hlaseni" zobrazí verze programu nahraná v Arduinu, v opačném případě je vypsána chybová zpráva.

Druhá část se zaměřuje na zobrazování dat v grafu. Jakmile uživatel nastaví periodu vzorkování hodnotu v textovém poli "Ts" a stiskne tlačítko "Graf", aktivuje se časovač. Úkolem tohoto časovače je pravidelně volat funkci UpdatePlot.

Když je funkce UpdatePlot spuštěna, prvním krokem je čtení dat z pinů A0 a A1 na Arduinu. Tyto zaznamenané hodnoty jsou poté zpracovány a zobrazeny v grafu. Ale nejde o klasické vykreslování grafu. Namísto toho jsou stávající křivky v grafu pouze aktualizovány. Toto řešení je efektivnější, protože aktualizace křivek je mnohem méně náročná na výpočetní kapacity než kompletní vykreslení grafu. Takto je zajištěno, že graf je průběžně aktualizován bez zbytečné zátěže na výpočetní výkon.

Bylo objasněno, že funkce UpdatePlot je pravidelně spouštěna pomocí časovače. Na obrazovce je uživatelům ukázán graf, jenž je průběžně aktualizován díky novým datům přijatým z Arduina a časovači. V aplikaci je tak aplikován princip synchronizace v reálném čase. Koordinace činností, od sběru dat po jejich vizualizaci, je řízena časovačem a funkcí UpdatePlot.

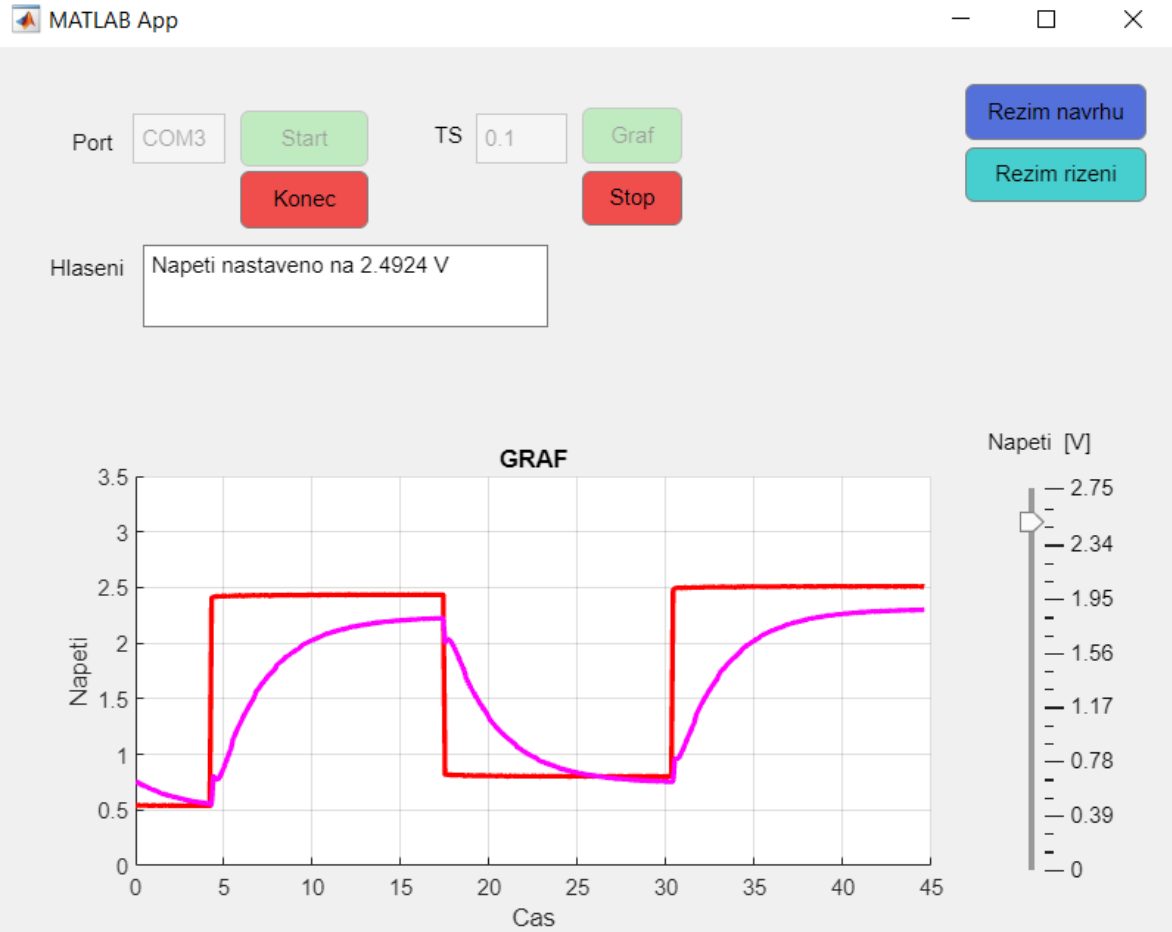
Pro zajištění, aby data nezůstávala uložena do nekonečna, byl zaveden limit, díky kterému se ukládají pouze informace z nejnovějších 45 sekund. Délka závisí na zvolené frekvenci vzorkování.

Ve třetí části je prezentován graf doplněný o posuvník. Je nezbytné mít na paměti, že arduino komunikuje s binárními hodnotami v rozmezí od 0 do 4095, což vyžaduje přepočet dat při jejich zápisu i čtení z Arduina. Díky posuvníku může být napětí nastaveno uživateli, a jakmile je hodnota na posuvníku vybrána, je tato hodnota aplikována na DAC1. Po změně napětí je v grafu zaznamenána odpovídající reakce. Na obrázku 2.10 byly prezentovány reakce na změny uskutečněné pomocí posuvníku. Zvolené napětí bylo ilustrováno červenou čarou, zatímco růžová čára reprezentovala výstupní hodnotu systému.

Nastavování hodnot a čtení dat z Arduina je realizováno prostřednictvím posílání zpráv, podobně jak bylo popsáno v kapitole 2.3.2. Hlavní rozdíl spočívá v tom, že funkce byly výrazně zjednodušeny. Kromě toho, kód pro Arduino byl upraven pro jednodušší

implementaci a místo programing portu byl využit nativní port, díky kterému je možné dosáhnout vyšších rychlostí přenosu dat.

Ve čtvrté části GUI v základním režimu byla umístěna dvě tlačítka, konkrétně "Rezim navrhu" a "Rezim rizeni". Pomocí těchto tlačítek je umožněn přechod na další obrazovky. Pro nastavování hodnot na Arduinu a čtení dat z něj byly použity metody zvané "gettery" a "settery". Díky těmto metodám je komunikace s Arduinem z dalších oken GUI umožněna.



Obrázek 2.10 – GUI základní režim reakce soustavy bez regulátoru

2.6.2 GUI režim návrhu

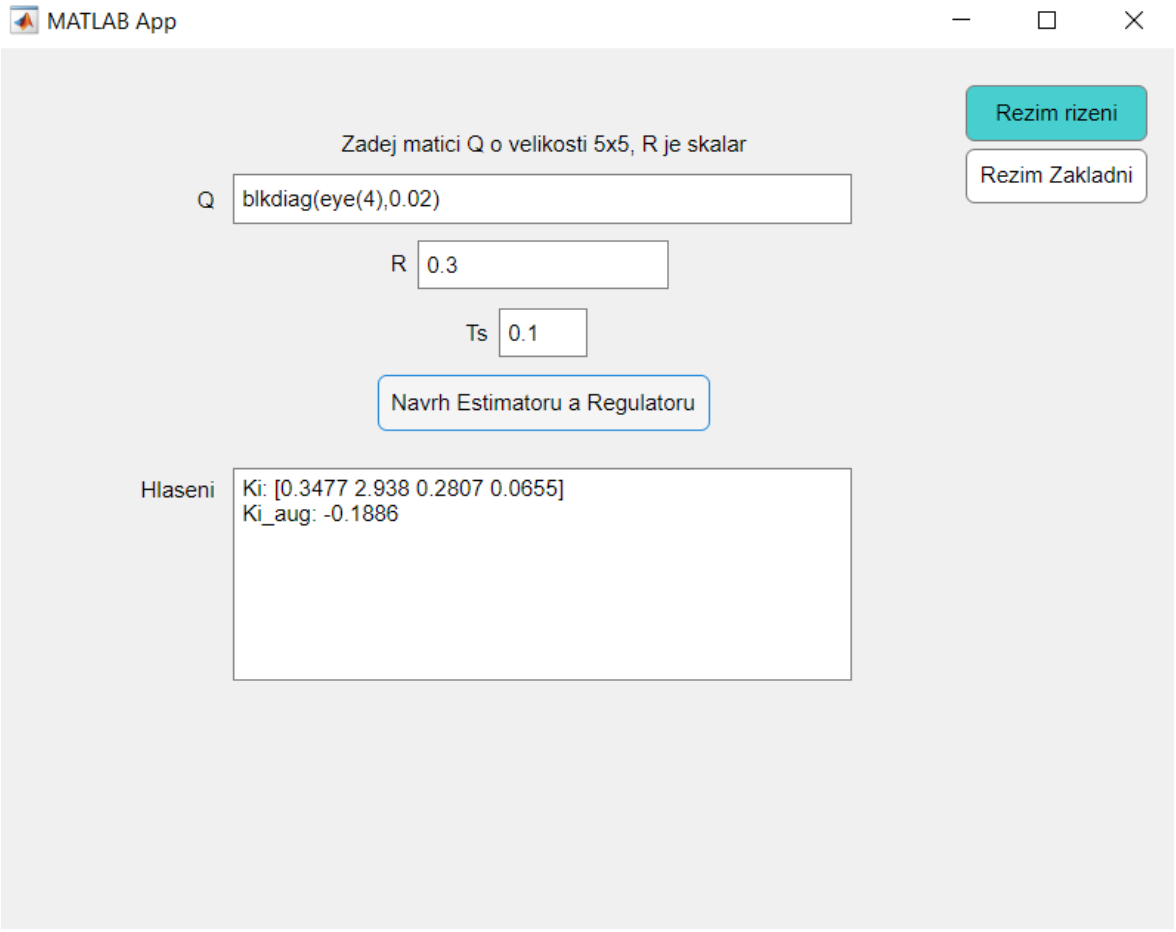
Ve funkci na výpočet estimátoru a LQ regulátoru bylo provedeno několik důležitých kroků. Na začátku byly data načtena ze souboru MAT.mat, kde byly uloženy matice pro spojitý model systému (*1.3*) a (*1.4*). Pro návrh diskrétního řízení byla tato spojitá

reprezentace převedena na diskrétní model (*1.7*) a (*1.8*) s využitím funkce `c2d` a intervalem vzorkování *k*=Ts_T.

Po získání diskrétního modelu byl vypočítán estimátor za pomoci funkce `dlqr`, přestože její primární účel je návrh LQ regulátoru. Zle tuto funkci použít i pro návrh estimátoru jak je blíže popsáno v kapitole 1.3.1.

V následující fázi byl navržen LQ regulátor s integrací regulační odchylky. K tomuto účelu byly systémové matice rozšířeny o integrační složku, což bylo reprezentováno rovnicí (*1.33*). S těmito rozšířenými maticemi byla opětovně aplikována funkce `dlqr` za účelem výpočtu zesílení **K**, které minimalizuje kvadratickou funkci dle rovnice (*1.38*). Následně bylo zesílení **K** rozděleno podle rovnice (*1.39*). Všechny výsledky z těchto operací byly následně uloženy do souboru Est_LQ.mat.

Na obrázku 2.11 je znázorněna obrazovka určená pro návrh estimátoru a LQ regulátoru. Jak lze vidět, obsahuje čtyři textová pole. Do prvního se zadává matice **Q** s rozměry 5x5. Jde o rozšířenou matici s integrační složkou. Je nezbytné pečlivě zvolit poslední hodnotu na diagonále, jelikož má výrazný dopad na konečný výsledek regulace. Skalární hodnota **R** je vkládána do druhého pole. Ve třetím poli, označeném jako Ts, je požadována frekvence vzorkování, podle které je navrhován LQ regulátor. Ve čtvrtém, posledním textovém poli, jsou zobrazovány vypočítané hodnoty zesílení. Výpočet je proveden po kliknutí na tlačítko "Navrh Estimatoru a Regulatoru".



Obrázek 2.11 – GUI režim návrhu

2.6.3 GUI režim řízení

Tohle GUI je v mnohém podobné základnímu režimu GUI. Jak název napovídá, umožňuje řízení soustavy k žádané hodnotě. Hlavním rozdílem je začlenění LQ regulátoru, který tuto funkci řízení nabízí.

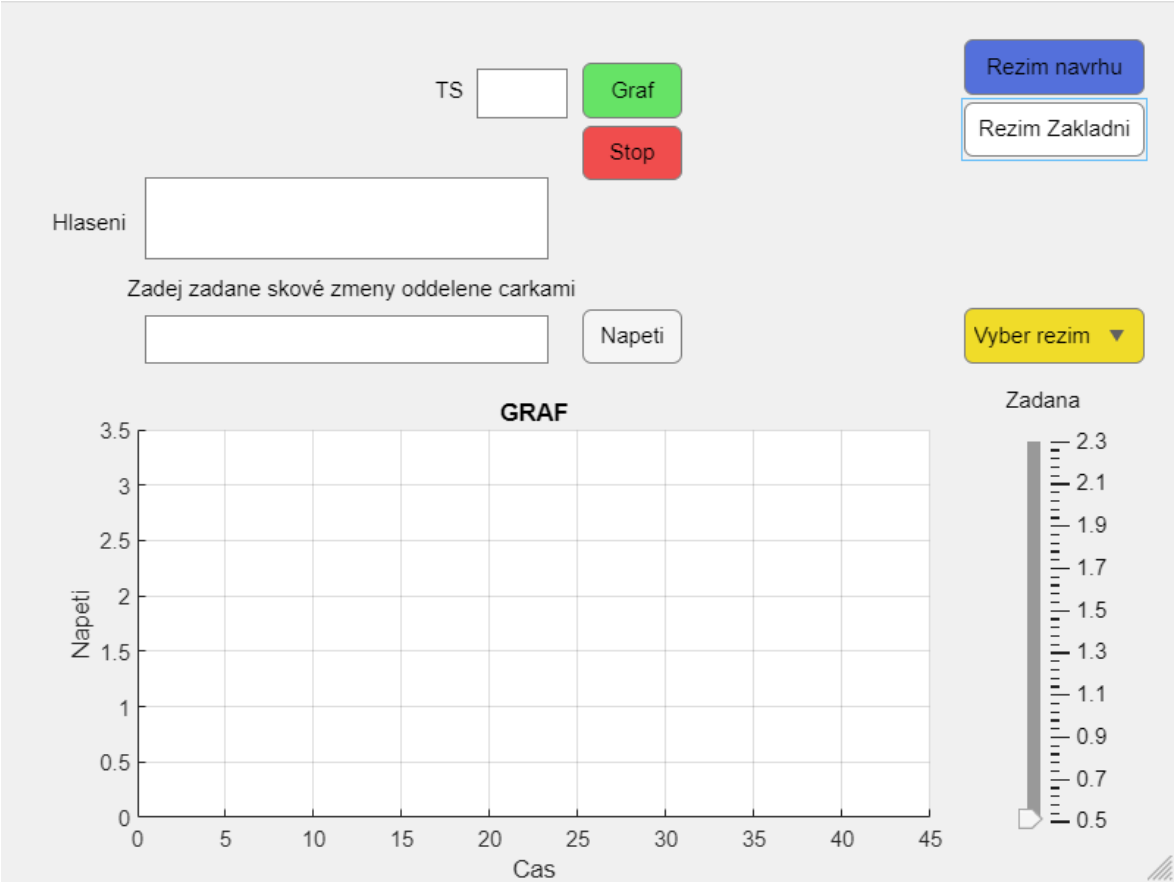
LQ regulátor je implementovaný ve funkci UpdatePlot tím je zajištěno že vypočet akční veličiny bude probíhat v pravidelných intervalech.

Pro správné fungování LQ regulátoru je potřeba načíst příslušná data, která byla vypočítána v GUI režim návrhu. Tato data se načítají stiskem tlačítka "Graf".

Ohledně výpočtu akční veličiny a jejího nastavení na Arduino. Odhad stavu byl nejprve vykonán, po němž následoval výpočet chyby a její integrace. Na základě těchto informací byla určena akční veličina. Tato veličina byla omezena, aby hodnoty nepřesáhly rozsah od 0.53 do 2,73V, což jsou limity arduina. Tato akční veličina byla poté nastavena na Arduino.

V aplikaci byla přidána komponenta "Drop down", skrze kterou může uživatel zvolit jeden z režimů řízení – buď manuální nebo předdefinovaný. V manuálním režimu je žádaná hodnota určena ručně pomocí posuvníku. Po zadání této hodnoty je řízení předáno LQ regulátoru, který se o následné kroky postará. V předdefinovaném režimu je postup složitější. Referenční hodnoty jsou načítány ze vstupního pole, kde jsou odděleny čárkou. Na základě těchto hodnot je vypočítán odhad doby trvání experimentu, přičemž každý krok má předpokládanou dobu 20 sekund. Tato odhadovaná doba je uživateli prezentována v textovém poli. Po spuštění experimentu se automaticky mění referenční hodnoty v pravidelných intervalech 20 sekund. Jakmile jsou všechny referenční hodnoty aplikovány, časovač je zastaven a uživatel je informován o ukončení experimentu prostřednictvím textového pole. Po ukončení experimentu zůstává poslední nastavená hodnota z experimentu. Po experimentu je možné změnit jaký režim řízení opět chceme.

Na obrázku 2.12 bylo zobrazeno, že rozsah posuvníku byl zmenšen. Toto opatření bylo přijato, aby bylo možné dosáhnout žádané hodnoty, pokud by žádaná hodnota byla blízko maximálního napětí, které Arduino může poskytnout. V takovém případě by dosažení žádané hodnoty nebylo možné.



Obrázek 2.12 – GUI režim řízení

2.6.4 Blokovací logika

Byla implementována blokovací logika, aby bylo zabráněno uživatelům v nastavení nebo vyvolání nepřípustných situací. V první fázi byly identifikovány klíčové problémy vyžadující řešení:

1. Když je aktivní režim řízení (viz obrázek 2.13) s běžící regulací, musí být zablokována možnost měnit napětí v hlavním GUI (viz obrázek 2.9). Tím se zamezuje rušení běžící regulace. Současně by mělo být zakázáno vykreslování grafu v hlavním GUI.
2. Pokud je regulátor navržen pro určitou frekvenci vzorkování a pokusí se jej použít pro jinou frekvenci, může dojít k nesprávné regulaci. Tento jev je třeba blokovat.
3. Přechod do režimu řízení by měl být možný pouze za podmínek, kdy jsou dostupné parametry pro estimátor a LQ regulátor a je otevřený sériový port.
4. Ihned po spuštění aplikace by některé prvky měly být neaktivní, aby se předešlo nežádoucímu chování. Na konci běhu aplikace je zase nutné provést některé úkony, např. ukončení komunikace s Arduinem.

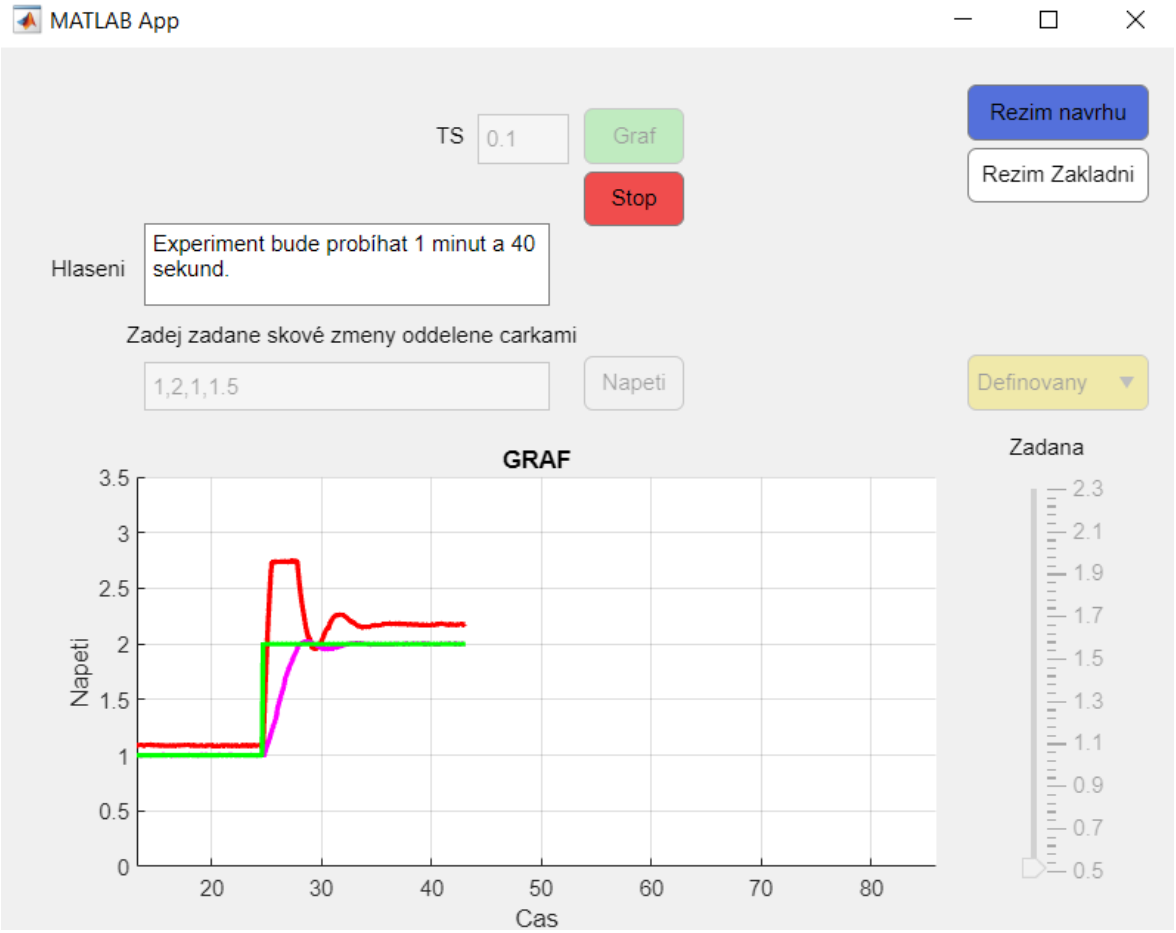
Řešení jednotlivých problémů:

1. Deaktivací konkrétních komponent v hlavní obrazovce, konkrétně posuvníku, pole Ts a tlačítka "graf", bylo dosaženo zamezení nežádoucím interakcím. Vzhledem k tomu, že tato deaktivace byla ovlivněna událostmi v jiném GUI, tedy "režim řízení", byly využity settery a gettery pro správnou manipulaci s těmito komponentami.
2. V obrazovce "režim řízení" byla po stisknutí tlačítka "graf" data pro estimátor a LQ regulátor načtena ze souboru. V těchto datech byla obsažena hodnota Ts_T, která reprezentuje vzorkovací periodu, pro kterou byl navržen LQ regulátor. Pokud se tato hodnota shodovala s aktuálně zadanou hodnotou v textovém poli Ts, spustil se časovač a data se začaly vykreslovat do grafu. V případě neshody byla v textovém poli hlášení zobrazena zpráva, upozorňující na potřebu navrhnout nový regulátor v "režimu návrhu" nebo změnit vzorkovací periodu, přičemž byla uvedena hodnota vzorkovací periody, pro kterou byl navržen aktuální regulátor.

3. Pro tento problém je použita podmínka, která ověřuje, zda je port otevřen a zda je současně dostupný soubor s názvem Est_LQ.mat. Tento soubor je vytvářen v návrhovém režimu po úspěšném dokončení návrhu. V podstatě je zde uplatněn logický AND.
4. Tento problém je řešen funkcemi startupFcn a closeRequest. Jak název napovídá, funkce startupFcn je automaticky spouštěna při spuštění (otevření) aplikace. Na druhou stranu, když se uživatel pokusí aplikaci zavřít, je vyvolána funkce closeRequest. Díky tomuto postupu mohou být definovány specifické akce, které se mají vykonat, než je aplikace skutečně uzavřena.

V GUI bylo několik blokací, které byly obvykle řešeny triviálním způsobem. Když bylo stisknuto tlačítko, konkrétní komponenta byla často zneaktivněna nebo byla provedena kontrola velikostí matic v návrhovém režimu atd. Tato ošetření nejsou v tomto textu podrobněji popisována.

Pro lepší pochopení zneaktivnění komponent je zobrazeno na obrázku 2.13 GUI režim řízení při probíhajícím experimentu s předdefinovaným průběhem.

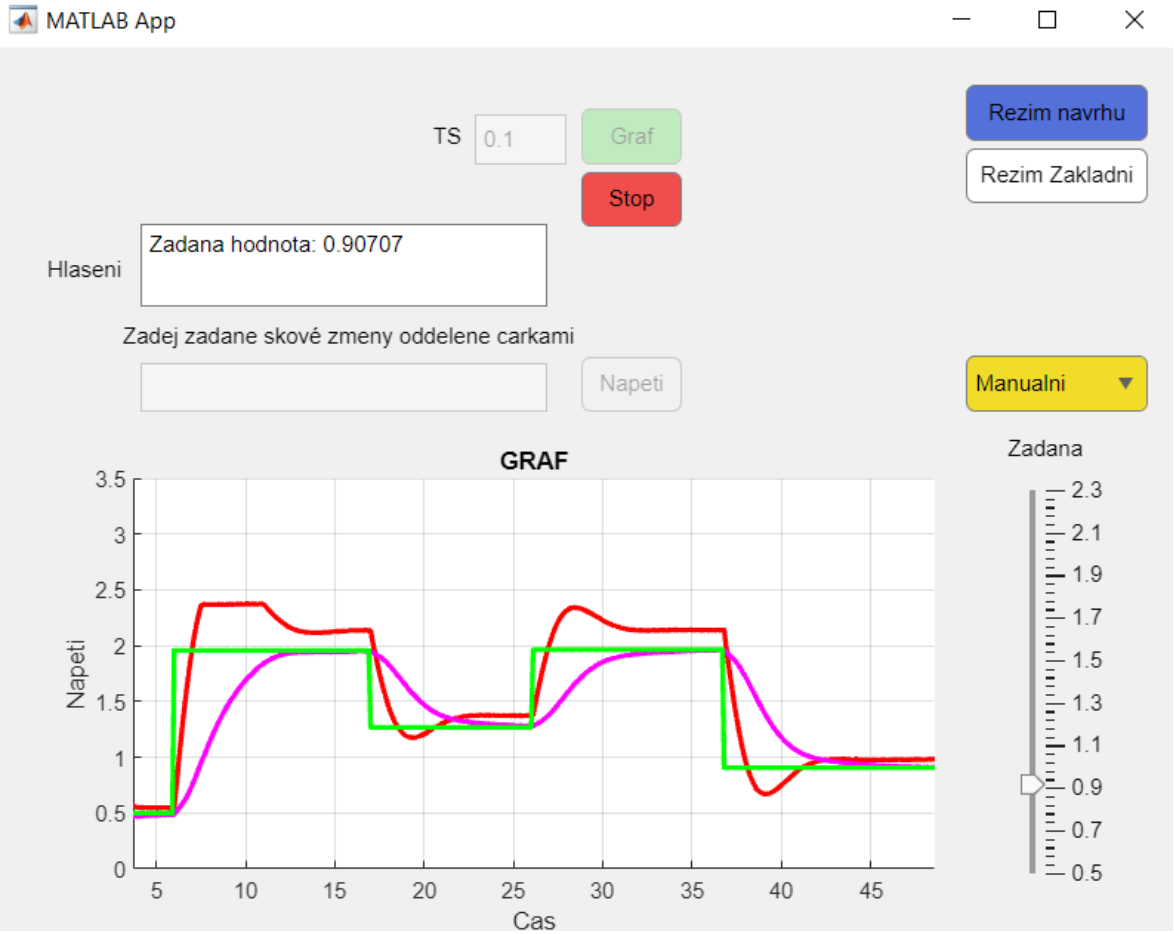


Obrázek 2.13 – GUI režim řízení při experimentu s předdefinovanou žádanou hodnotou

2.7 POROVNÁNÍ SIMULACÍ A REÁLNÉ SOUSTAVY

Pro ověření funkčnosti navrhovaného GUI pro řízení byly realizovány různé experimenty a simulace. Klíčovým cílem bylo potvrdit správnou funkčnost LQ regulátoru s estimátorem a schopnost odstranění trvalé regulační odchylky. Experimenty na reálném zařízení s manuálním řízením a několika skokovými změnami jsou znázorněny na obrázku 2.14. Dále se stejnými parametry byl proveden experiment s předdefinovanými parametry viz obrázek 2.15.

Poté byla vytvořena simulace s parametry shodnými s prvním experimentem. Cílem těchto testů bylo ověřit, jestli výsledky simulace odpovídají skutečnému měření a zda se matematický model chová stejně jako skutečný systém.

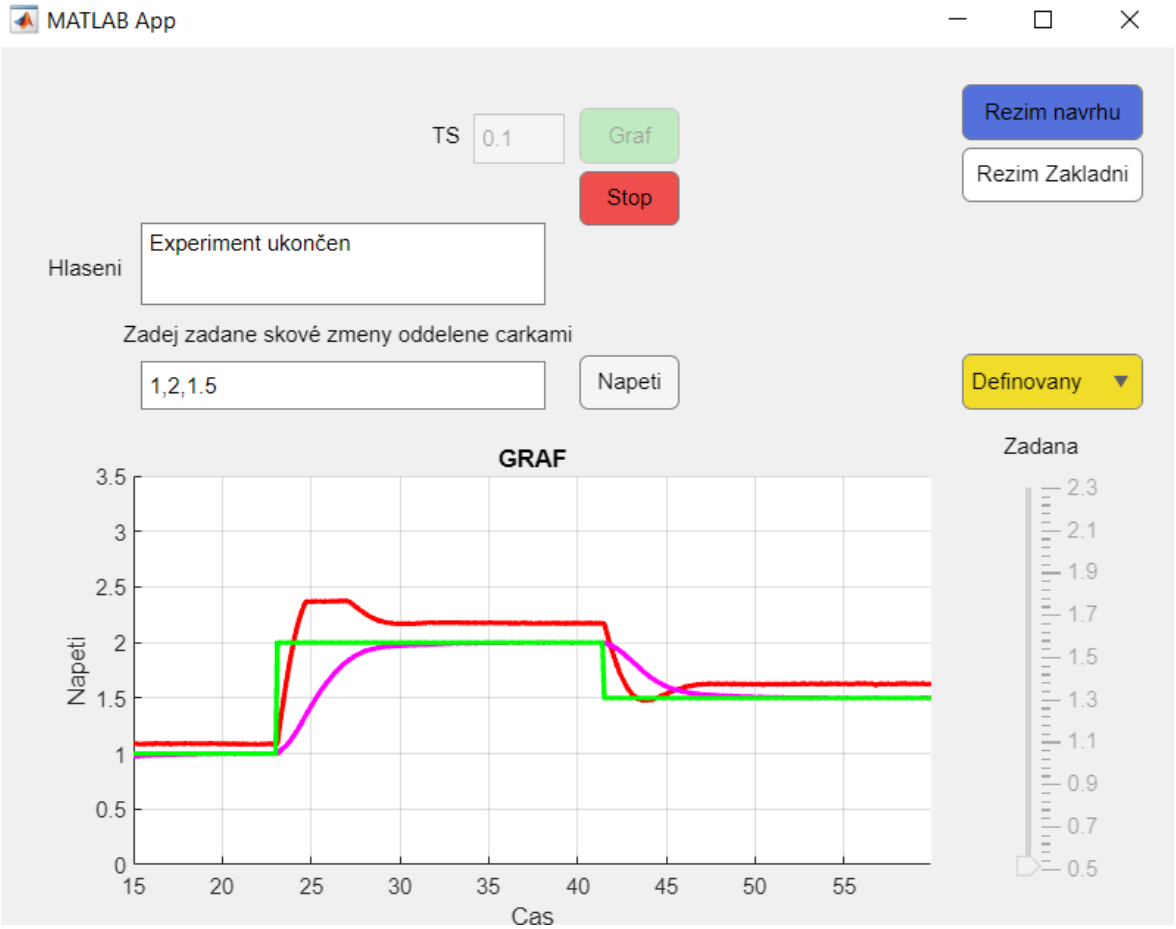


Obrázek 2.14 – Regulační pochod reálná soustava žádávána manuálně Ts=0,1s

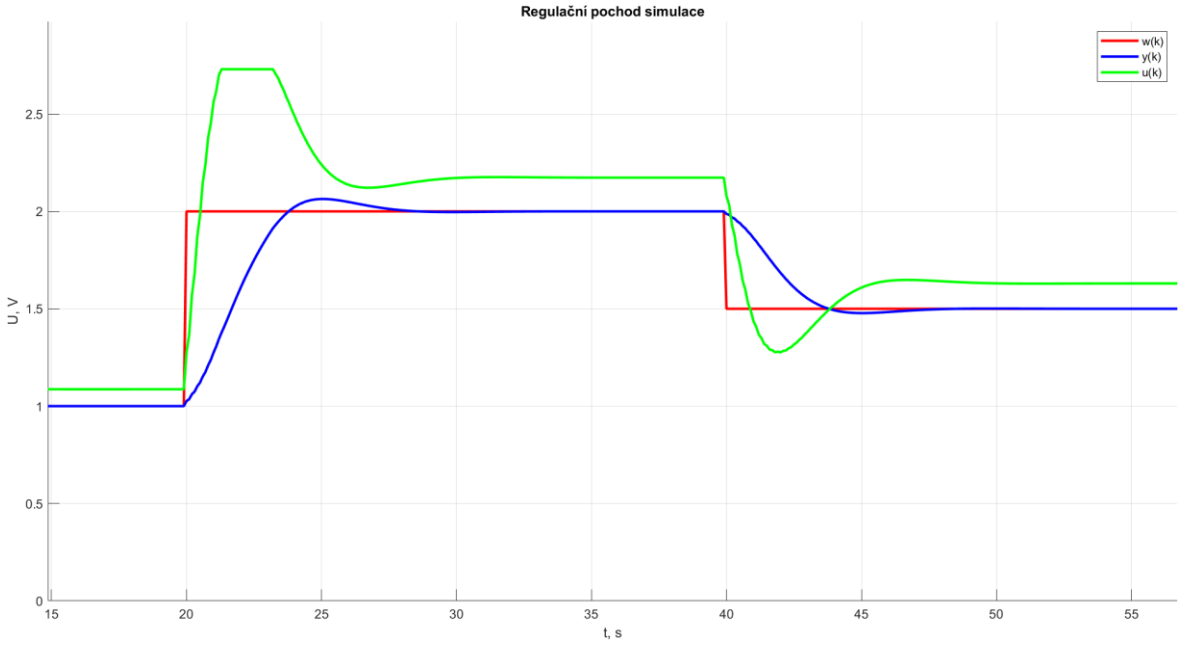
Použité parametry pro tento regulační pochod byly následovné Ts = 0,1s matice **R** a $\mathbf{Q_{aug}}$ jsou zobrazené níže

$$\mathbf{Q_{aug}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0{,}02 \end{bmatrix}; \qquad \mathbf{R} = [0{,}3].$$

Se stejnými parametry ale pro předdefinovaný průběh na reálné soustavě



Obrázek 2.15 – Regulační pochod předdefinovaný průběh Ts=0,1s

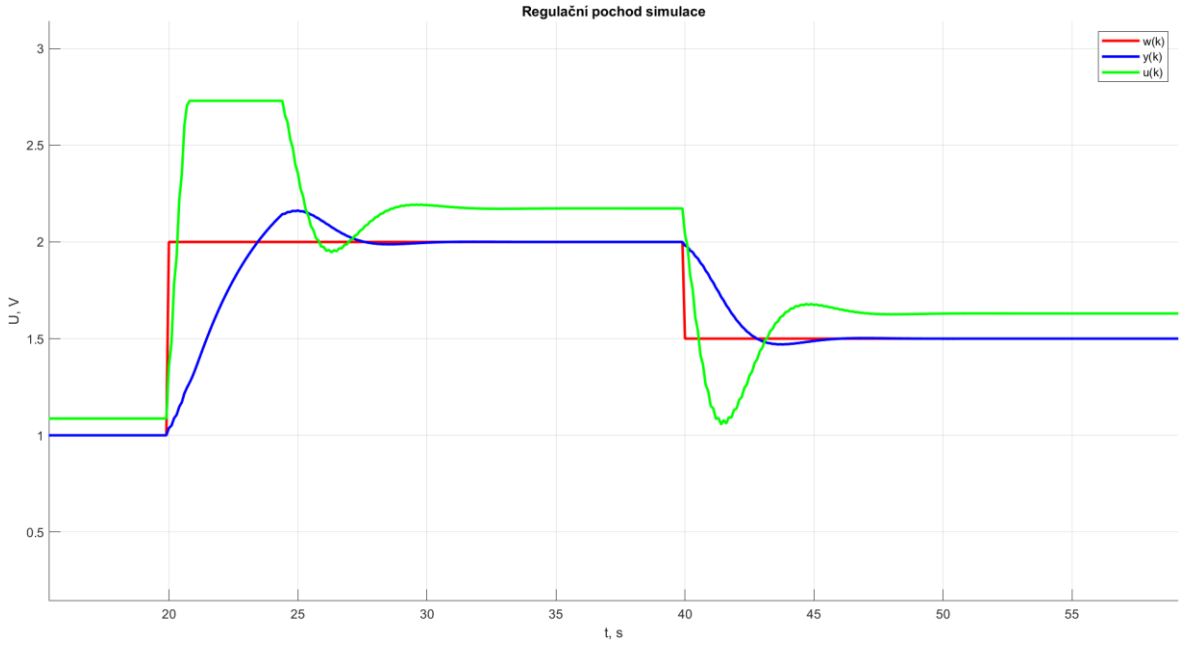


Obrázek 2.16 – Simulace se stejnými parametry jako obrázky 2.14 a 2.15

Nyní budou nastaveny tyto parametry (parametry 2) níže s Ts= 0.1s

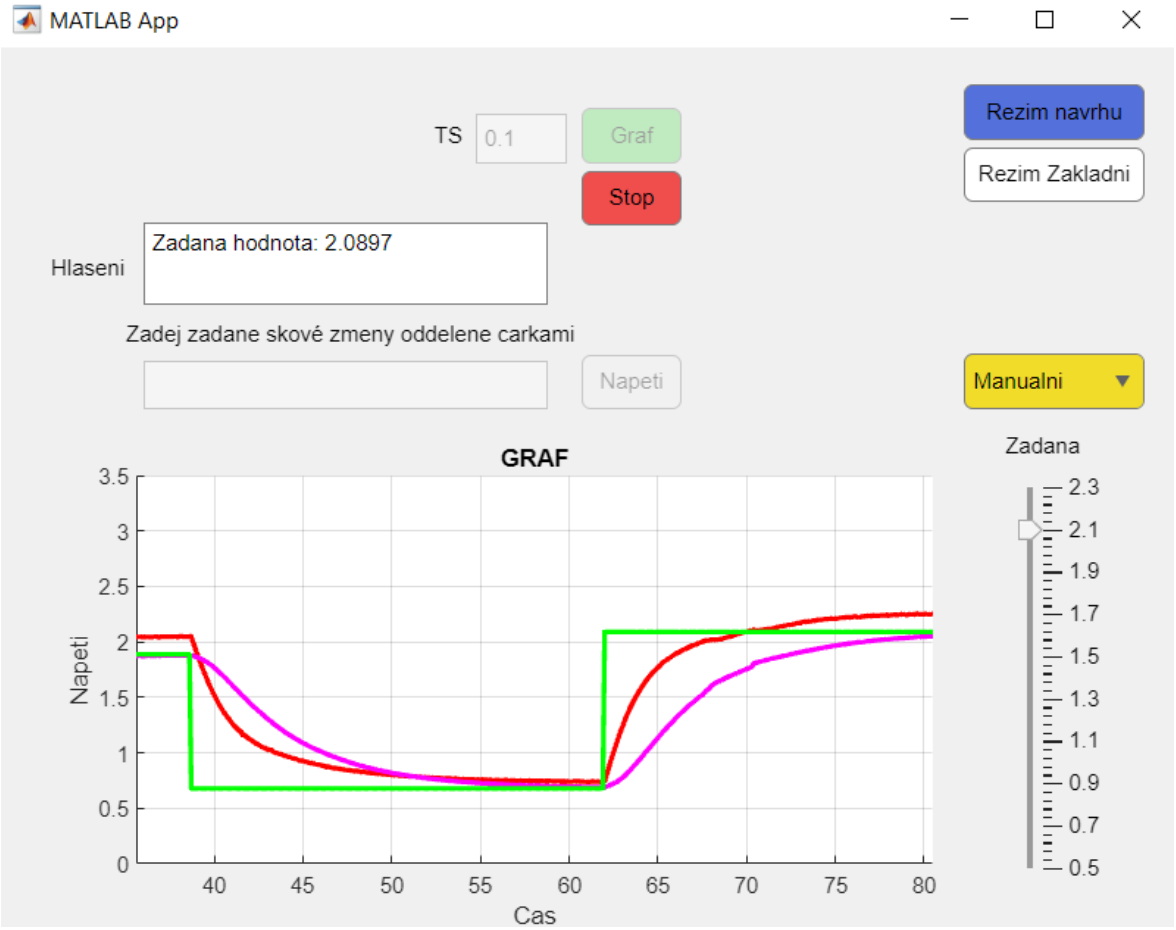
$$\mathbf{Q}_{aug} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0{,}05 \end{bmatrix}; \qquad \mathbf{R} = [0{,}4].$$

Obrázek 2.17 – Regulační pochod reálná soustava parametry 2 s Ts=0,1s



Obrázek 2.18 – Simulace s parametry 2

Na závěr testování byli aplikovány doporučené parametry ze simulace v kapitole 2.5.



Obrázek 2.19 – Regulační průběh s doporučenými parametry z kapitoly 2.5

Z obrázků výše týkajících se reálné soustavy bylo zjištěno, že dochází k eliminaci trvalé regulační odchylky. Tím bylo ověřeno správné fungování LQ regulátoru.

Pokud srovnáme výsledky simulace s reálným měřením, zjistíme, že se výsledky liší, ačkoli ne výrazně. Tyto rozdíly mohou být způsobeny nepřesností matematického modelu. Dále je třeba si uvědomit, že všechny stavy jsou odhadovány, a protože regulátor byl navržen jako duální úloha, odhady stavů nemusí být vždy zcela přesné. Přesto se průběhy obou měření alespoň částečně podobají.

Z analýz simulací a měření na reálné soustavě bylo zjištěno, že reálná soustava toleruje agresivnější nastavení regulátoru než simulace. Po prozkoumání obrázku 2.15 bylo zaznamenáno, že ustálení na reálné soustavě nastává přibližně po 10 vteřinách. V simulaci, jak je zobrazeno na obrázku 2.16, bylo ustálení dosaženo rychleji, konkrétně po 8 vteřinách. Přesto těmto rozdílům bylo fungování regulace na reálné soustavě hodnoceno jako správné.

V kapitole 2.5 prezentované doporučené parametry způsobily velmi pomalé ustálení na reálné soustavě, což bylo způsobeno nízkou hodnotou **Qi**. Jak bylo uvedeno, reálná soustava může snášet agresivnější nastavení regulátoru. Příslušný průběh byl zobrazen na obrázku 2.19.

Pro správnou regulaci reálné soustavy musely být správně vybrány parametry pro LQ regulátor. Jak již bylo zmíněno, reálná soustava lépe toleruje agresivnější nastavení regulátoru než simulace. Simulace tedy byla použita primárně k orientačním účelům, poskytujíc nám základní přehled.

ZÁVĚR

Tato diplomová práce se zaměřila na problematiku LQ regulátoru a na tvorbu grafického uživatelského rozhraní (GUI) v prostředí MATLAB, které je schopno komunikovat s Arduinem Due.

V rámci výzkumu byl v prostředí MATLAB vytvořen program, který realizuje LQ řízení lineárního SISO systému za pomoci Arduino Due. Hlavním cílem bylo nabídnout nástroj, který je nejen efektivní, ale také uživatelsky přívětivý, pro regulaci a odhad stavu zkoumaného systému.

V teoretické části byla provedena detailní analýza spojitých a diskrétních stavových modelů. LQ regulátor byl zkoumán z různých perspektiv, včetně různých metod jeho návrhu v spojitém i diskrétním kontextu. Zvláštní pozornost byla věnována návrhu LQ regulátoru s cílem dosáhnout sledování žádané hodnoty bez trvalé regulační odchylky.

V praktické části byly simulace zpracovány, což nám umožnilo ověřit, jak nastavený regulátor ovlivňoval výstup soustavy, a dalo nám představu o správném nastavení regulátoru pro reálnou soustavu. Hlavní důraz byl kladen na GUI, protože skrze ně probíhala většina interakcí a nastavení. Toto rozhraní bylo navrženo s ohledem na snadné ovládání a komunikaci s Arduinem a řízením soustavy.

Výsledkem této diplomové práce bylo GUI v prostředí MATLAB, které mohlo regulovat soustavu s konkrétním matematickým modelem pomocí LQ regulátoru. Fungování bylo ověřeno experimenty s různými nastaveními LQ regulátoru.

Pro další vývoj a rozšíření práce by mělo smysl zkoumat adaptivní algoritmy, zejména Model Reference Adaptive Control (MRAC) a adaptivní řízení založené na Lyapunovově teorii stability. V kontextu našeho systému, kde je možné měřit tři ze čtyř stavů, by mohl být výhodný přístup k adaptivním algoritmům, který by poskytl vylepšený odhad pro čtvrtý, nezměřený stav. Tyto metody by mohly dynamicky aktualizovat parametry v závislosti na měřených datech a modelech, čímž by byla zvýšena robustnost a citlivost systému v různých podmínkách. Dalším přístupem, který by mohl být prozkoumán, je rozšířený Kalmanův filtr, jenž kombinuje měřená data s modely pro odhad stavů.

Shrnutím výše uvedeného lze konstatovat, že tato diplomová práce úspěšně dosáhla svých cílů, poskytla pevný základ v oblasti automatizace a řízení, s potenciálem pro další výzkum a inovace v oblasti adaptivního řízení.

POUŽITÁ LITERATURA

Arduino - Home [online], 2023. [cit. 2023-6-6]. Dostupné z: https://www.arduino.cc/

ARDUINO DUE. In: *Hwkitchen.cz* [online],2023 [cit. 2023-08-22]. Dostupné z: https://www.hwkitchen.cz/arduino-due/?gclid=Cj0KCQjwuZGnBhD1ARIsACxbAVgBSW5vcj4Smp-vFM8Az6wVGS3VI3Gg7frWLjlQyFVHo6u3-mlA_kcaAtETEALw_wc

BBALÁTĚ, J. 2004. Automatické řízení. 2., přeprac. vyd. Praha: BEN – technická literatura. ISBN 80-7300-148-9.

DUŠEK, František, 2021. *Spojité řízení.* Přednášky. Fakulta elektrotechniky a informatiky. Univerzita Pardubice: Osobní sdělení.

DUŠEK, František, 2023. *Osobní konzultace*. Fakulta elektrotechniky a informatiky. Univerzita Pardubice: Osobní sdělení.

JELÍNEK, Josef, 2023. *Experimentální identifikace diskrétního stavového modelu*. Fakulta elektrotechniky a informatiky Univerzita Pardubice. Dostupné také z: https://dk.upce.cz/bitstream/handle/10195/81555/JelinekJ_ExperimentalniIdentifikace_FD_2023.pdf?sequence=1&isAllowed=y. Diplomová práce. Univerzita Pardubice. Vedoucí práce František Dušek.

KUBÍK, Stanislav, Zdeněk KOTEK, Vladimír STREJC a Jan ŠTECHA, 1982. *Teorie automatického řízení I: Lineární a nelineární*. 1. Praha: Státní nakladatelství technické literatury, 522 s.

KUPKA, Libor, 2020. *Diskrétní řízení*. Přednášky. Fakulta elektrotechniky a informatiky. Univerzita Pardubice: Osobní sdělení.

MACHEK, Ondřej, 2019. *LQ regulace systému motor generátor*. Pardubice. Dostupné také z: https://dk.upce.cz/bitstream/handle/10195/73107/MachekO_LQRegulace_FD_2019.pdf?sequence=1&isAllowed=y. Diplomová práce. Univerzita Pardubice. Vedoucí práce F. Dušek.

MATLAB Documentation, 2023. *MATLAB: Documentation* [online]. United States: The MathWorks [cit. 2023-06-21]. Dostupné z: https://www.mathworks.com/help/?s_tid=gn_supp

Modelování a identifikace řízeného systému, 2006. Řízení technologických procesů [online]. Zlín: Univerzita Tomáše Bati ve Zlíně Fakulta aplikované informatiky [cit. 2023-06-06]. Dostupné z: http://rtp.webzdarma.cz/model1.php

OGATA, Katsuhiko, 1995. *Discrete-time control systems*. 2nd ed. Englewood Cliffs, N.: J.: Prentice Hall. ISBN 01-303-4281-5.

OPRŠAL, Daniel, 2021. *IMPLEMENTACE DISKRÉTNÍHO LQG REGULÁTORU S VYUŽITÍM SCADA SYSTÉMU PROMOTIC*. Fakulta elektrotechniky a informatiky Univerzita Pardubice. Dostupné také z: https://dk.upce.cz/bitstream/handle/10195/77954/OprsalD_ImplementaceDiskretniho_FD_2021.pdf?sequence=1&isAllowed=y. Diplomová práce. Univerzita Pardubice. Vedoucí práce František Dušek.

STREJC, Vladimír, 1978. Stavová teorie lineárního diskrétního řízení. Praha: Academia.

ŠVARC, I. Teorie automatického řízení. [online]. 2003 [cit. 2023-06-06]. VUT FSI. Dostupné z: http://www.fsiforum.cz/upload/soubory/knihy/Automatizace/Teorie.automatickeho.rizeni_Svarc_2003.pdf

Vnější popis a vlastnosti lineárních spojitých dynamických systémů 2006. Řízení technologických procesů [online]. Zlín: Univerzita Tomáše Bati ve Zlíně Fakulta aplikované informatiky [cit. 2023-06-06]. Dostupné z: http://rtp.webzdarma.cz/vlastnosti1.php

PŘÍLOHY

A – CD

Příloha A

Příloha k diplomové práce

LQG REGULACE V REÁLNÉM ČASE V PROSTŘEDÍ MATLAB

Bc. Jan Hudský

CD

Obsah