

# Detaljni opis projektnog zadatka

Prijediplomski studij Računarstva i Elektrotehnike i informacijske tehnologije

Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva

**Naziv teme:** Implementacija nelinearnog modelskog prediktivnog upravljanja dinamičkim sustavom

**Mentor:** izv. prof. dr. sc. Branimir Novoselnik

**Akadska godina:** 2025./2026.

## Članovi projektnog tima

Student	Zadatak / odgovornost	Kontakt (e-mail)
Mateo Gnjdicić	Matematičko modeliranje i implementacija dinamičkog modela sustava	mateo.gnjidic@fer.hr
Luka Kordić	Implementacija NMPC algoritma	luka.kordic@fer.hr
Marko Maslač	Simulacija i vizualizacija ponašanja sustava	marko.maslac@fer.hr

## Opis zajedničke teme

Cilj projekta je razvoj i implementacija algoritma nelinearnog modelskog prediktivnog upravljanja (NMPC) za sustav kolica s njihalom u MATLAB okruženju. Projekt obuhvaća matematičko modeliranje sustava, formulaciju i implementaciju NMPC algoritma za stabilizaciju njihala u uspravnom položaju te simulaciju i vizualizaciju ponašanja sustava u zatvorenoj petlji.

Kroz projekt će studenti razviti modularan MATLAB kod organiziran u klase (ili strukture s pripadnim metodama), čime se omogućuje jasno razdvajanje funkcionalnih cjelina i jednostavna nadogradnja modela i regulatora.

## 1. Matematičko modeliranje i implementacija dinamičkog modela sustava

Student je zadužen za matematičko modeliranje sustava kolica s njihalom i implementaciju pripadajućih funkcija u MATLAB-u. Potrebno je:

- Izvesti nelinearni model sustava u obliku diferencijalnih jednadžbi stanja.
- Definirati vektor stanja  $x = [x, \dot{x}, \theta, \dot{\theta}]^T$  i upravljačku veličinu  $u$  (sila koja djeluje na kolica).
- Implementirati MATLAB klasu `CartPendulumModel.m` koja uključuje:
  - **Svojstva:** parametre sustava (mase, duljina poluge, trenje, gravitacija, broj stanja, broj ulaza, broj izlaza, itd.),
  - **Metode (primjeri):**
    - \* `stateFcn(x,u)` – funkcija koja vraća derivacije stanja,
    - \* `outputFcn(x,u)` – funkcija koja vraća izlazne varijable,
    - \* `simulateStep(x,u,Ts)` – numerička integracija pomoću metode Euler, Runge–Kutta 4, `ode45` i sl. uz opcionalnu linearizaciju sustava.
    - \* ...
- Validirati model kroz simulacije slobodnog gibanja i usporedba s teorijskim očekivanjima.

**Ishod:** MATLAB klasa dinamičkog modela koja omogućuje simulaciju sustava i korištenje u NMPC algoritmu.

## 2. Implementacija algoritma nelinearnog modelskog prediktivnog upravljanja (NMPC)

Student je zadužen za razvoj NMPC algoritma koristeći MATLAB-ove optimizacijske i MPC alate. Potrebno je:

- Formulirati NMPC problem u diskretnom obliku:

$$\min_{u(0), \dots, u(N-1)} \sum_{k=0}^{N-1} \ell(x(k), u(k)) + \ell_f(x(N))$$

uz ograničenja:

$$x(k+1) = f(x(k), u(k)), \quad u_{\min} \leq u(k) \leq u_{\max}$$

- Implementirati MATLAB klasu `NMPCController.m` koja:

- prima model i parametre algoritma (horizont, težine, ograničenja),
- koristi `Optimization Toolbox` ili `nlmpc` objekt za rješavanje optimizacijskog problema,
- u svakoj iteraciji poziva solver (`fmincon`, `quadprog`, `nlmpc`),
- vraća optimalnu upravljačku akciju  $u_{\text{opt}}$  za trenutno stanje.
- Implementirati mehanizme za “warm start” i predikciju budućih stanja.
- Validirati algoritam na scenariju swing-up i stabilizacije.

**Ishod:** funkcionalna NMPC implementacija koja generira optimalne upravljačke signale u svakom diskretnom koraku.

### 3. Simulacija zatvorene petlje i vizualizacija ponašanja sustava

Student je zadužen za razvoj simulacijskog okvira koji integrira model i regulator te omogućuje vizualnu i analitičku analizu ponašanja sustava. Potrebno je:

- Implementirati MATLAB klasu `SimulationEnvironment.m` koja omogućuje:
  - definiranje simulacijskih parametara (vremenski korak, ukupno vrijeme, početni uvjeti, poremećaji, itd.),
  - izvođenje simulacije zatvorene petlje (petlja:  $x_{k+1} = f(x_k, u_k)$ ),
  - spremanje rezultata u strukture ili `timeseries` objekte.
- Izraditi funkcije za prikaz rezultata:
  - grafovi stanja i upravljačkog signala kroz vrijeme,
  - graf troška kroz iteracije optimizacije (ako je primjenjivo).
- Implementirati animaciju gibanja kolica i njihala u 2D (npr. pomoću `plot`, `rectangle`, `line`, `drawnow`).
- Omogućiti jednostavno mijenjanje scenarija (npr. swing-up, poremećaji, različiti horizonti, ograničenja).

**Ishod:** potpuno funkcionalan simulacijski okvir s mogućnošću animacije i analize odziva sustava u zatvorenoj petlji.

### Napomena o zajedničkom radu

Studenti trebaju koristiti zajednički `GitLab` ili `GitHub` repozitorij, definirati zajedničke konvencije za imenovanje datoteka i funkcija te dokumentirati kod pomoću MATLAB-ovih komentara u formatu `help` (`%%, help function_name`).

Primjer strukture repozitorija:

```
src/  
  model/  
    CartPendulumModel.m  
  controller/  
    NMPCController.m  
  simulation/  
    SimulationEnvironment.m  
results/  
  simulation_results.mat  
  figures/  
  animations/  
docs/  
  report.pdf  
main.m
```

---

*Zagreb, listopad 2025.*