Kunnskap for en bedre verden

Department of Engineering Cybernetics

TTK4550 - Specialization Project

Design and Control of a Spring-actuated Jumping Quadruped in Earth Gravity

Author:
Johannes Ihle
Daniel Rosmæl Skauge

Supervisor:
Prof. Dr. Kostas Alexis

Co-supervisor:
Jørgen Anker Olsen

Date

Abstract

This project report presents our specialization project, which is the design and control of a quadruped, spring-actuated, etc.

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
API	Application Programming Interface
CPU	Central Processing Unit
DRL	Deep Reinforcement Learning
EKF	Extended Kalman Filter
ESKF	Error State Kalman Filter
GNC	Guidance, Navigation, and Control
INS	Inertial Navigation System
ML	Machine Learning
MOOS	Mission Oriented Operating Suite
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
USV	Unmanned Surface Vehicle

1 Introduction

1.1 Motivation

1.2 Scope

1.2.1 Problem Description

1.2.2 Related Work

[1] is an important paper.

1.3 Report Outline

2 Crazy Matrix

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \end{bmatrix} \tag{1}$$

3 Theory

3.1 Reinforcement Learning

3.1.1 The Reinforcement Learning Problem Setting

3.1.2 Value-Based and Policy-Based Methods

3.1.3 Policy Gradient Methods

3.2 Artificial Neural Networks

3.3 Electric Motor Dynamics

3.3.1 Motor Modeling

3.3.2 Torque Speed Models

3.4 Spring-Damper Systems

3.5 Kinematics, Jacobians, and Virtual Work

3.5.1 Robot Kinematics

Consider a robotic link arm existing in $\mathbb{R}^2$ consisting of n links, each with a length l_i and a joint angle θ_i. The position of the end-effector is given by the vector $\mathbf{x} = [x, y]^T$, where x and y are the coordinates of the end-effector in the global coordinate system. Using simple trigonometry, the position of the end-effector can be expressed as a function of the joint angles and link lengths as seen in equation 2.

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} l_i \cos(\theta_i) \\ \sum_{i=1}^{n} l_i \sin(\theta_i) \end{bmatrix} \tag{2}$$

3.5.2 Jacobian Matrix

As described in section 3.5.1, the position of the end-effector can be expressed as a function of the joint angles and link lengths. In this section we define the Jacobian, as described in equation 3, which relates infitesimal changes in the joint angles to the resulting change in the end-effector position.

Now I am testing how quick this is. Testing again. So much testing.

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} & \cdots & \frac{\partial x}{\partial \theta_n} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \cdots & \frac{\partial y}{\partial \theta_n} \end{bmatrix} \tag{3}$$

3.5.3 Virtual Work and Force/Torque Mapping

Example citation [1]

3.6 Dynamical Systems and Contact Dynamics

3.6.1 Rigid-Body Dynamics

3.6.2 Contact Modeling and Impact Dynamics

3.6.3 Friction Models

4 Modeling and Simulation

4.1 Simscape

Her forklares Simscape og hvordan vi lagde en simscape simulering. Her forklares *ikke* hvordan vi valgte ut link-lengths, spring, motor, etc. Altså dimensjonering forklares ikke her. Kun hvordan vi lagde simuleringen som etter hverdt skal brukes til dimensjonering. Det at den brukes til dimensjonering, forklares trolig i hardware/design seksjonen...

4.2 IsaacLab

5 Robot Design

This is where we explain "overall" design. It's motivation, etc. Also where we cover dimensioning, link-lengths, etc. We explain here our parameter sweep to find optimal design. We also explain here our kinematics-script that we used to derive the required stall-torque for our motor given springs and link-lengths.

5.1

5.2

6 Robot Hardware

This is where we discuss actual hardware, ie. component selection, materials, CAD, etc.

So here we specify motors and how they match specs, while referring to the design section to explain why we want these motors. We should here have a list of existing motors. The design section can link to this list to motivate its own choice. It will be circular, but that's okay.

7 Robot Control

7.1 RL Problem Description?

This is not where we explain the goal of the thesis, ie. "we want to jump". This is just where we explain the RL problem. That we want to * jump * land * etc. is something we describe earlier, for instance in Introduction/Scope/Problem Description.

8 Results

9 Discussion

10 Conclusion

Bibliography

[1] John Schulman et al. *Proximal Policy Optimization Algorithms*. en. arXiv:1707.06347 [cs]. Aug. 2017. URL: http://arxiv.org/abs/1707.06347 (visited on 12th Nov. 2024).

Appendix

A Hello World Example

```
int main {
    // This is a comment
    std::cout << "Hello World from C++!" << std::endl;
    std::cout << "I am using the default style to print this code in beautiful colors. Since the t
    return 0;
}
```

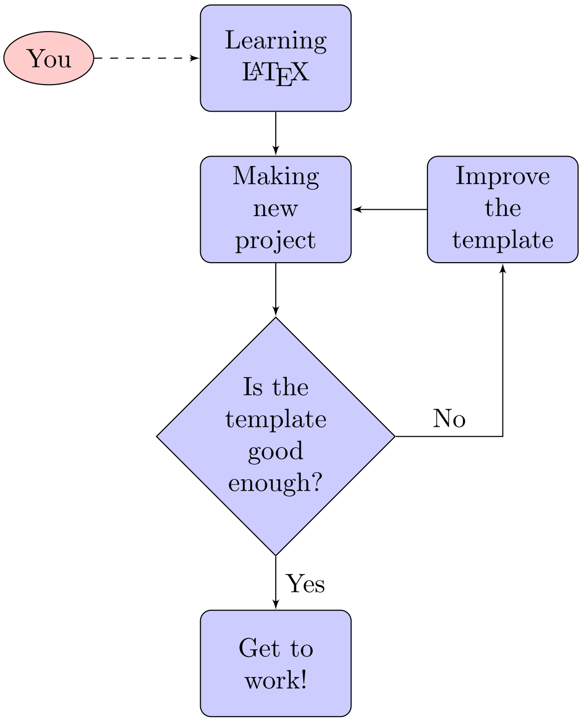
⋮

```
# This is a comment
print('Hello world from Python!')
print('I am using the "rrt" style to print this code in beautiful colors')
```

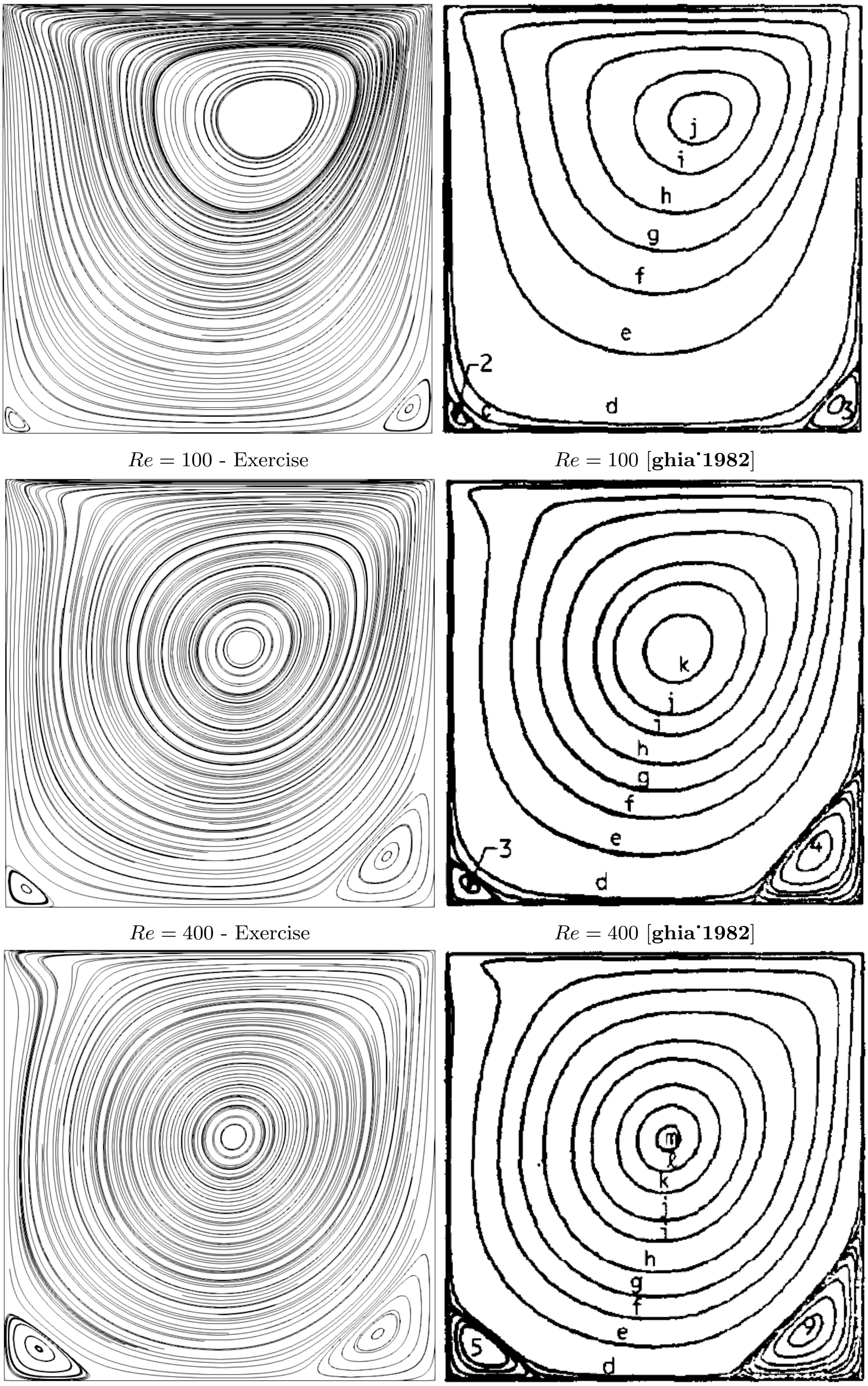
⋮

```
% Content of HelloWorld.m
disp('Hello World from Matlab!')
```

B Flow Chart Example



C Sub-figures Example



Re = 100 - Exercise

Re = 100 [**ghia˙1982**]

Re = 400 - Exercise

Re = 400 [**ghia˙1982**]

Re = 1000 - Exercise

Re = 1000 [**ghia˙1982**]

Figure 1: Streamlines for the problem of a lid-driven cavity.