

Introdução

Segundo [Nise, 2017](#), um sistema de controle consiste em um conjunto de subsistemas e processos (ou plantas) com o propósito de se obter uma saída especificada, dado uma entrada especificada.

Dito posto, o mesmo autor ressalta que esse conceito se faz muito presente no dia a dia. Tendo seu estudo uma grande importância.

Como o controle de sistemas reais exige uma estrutura considerável. O uso de simuladores é uma excelente alternativa para fins acadêmicos e de projeto.

O simulador [iDynamic](#) desenvolvido pela UFRN é bastante completo. Contando com simulação gráfica, interativa, 2D, 3D e com vários exemplos de sistemas diferentes. Pecando apenas na documentação rasa.

Contudo, sua principal funcionalidade é a possibilidade do usuário inserir o próprio controlador no sistema.

Objetivo

Esse projeto se trata de uma aplicação em que é possível trocar dados com o iDynamic, com a possibilidade de se inserir várias curvas de entrada e plotar as saídas do simulador. Concentrando-se no exemplo de massa-mola complexo.

Comunicação

O iDynamic foi desenhado para funcionar com [websockets](#). Isto é, uma comunicação TCP-IP, onde a aplicação é o servidor e o cliente é o próprio simulador.

Baseado nisso, a primeira comunicação foi feita através do [exemplo](#) dado na documentação do iDynamic. Contudo, por comodidade, tentou-se estabelecer uma comunicação através de [socket](#). Já que na conversação TCP-IP pouco importaria se seria utilizado websockets ou socket.

Entretanto, o iDynamic se mostrou apenas compatível com o websockets, encerrando qualquer comunicação via socket rapidamente. Portanto, adaptou-se um servidor websockets a arquitetura do projeto.

Arquitetura

O projeto é estruturado com [threading](#). Uma thread faz a comunicação com o iDynamic e outra controla a interface gráfica. Essa escolha foi tomada devido ao delay que seria demasiado no caso de um processamento single thread.

Como o websockets é baseado em [asyncio](#) que é uma biblioteca pensada em processos concorrentes em single thread, sua implementação com múltiplas threads foi um tanto custosa.

Por fim, a comunicação com o iDynamic foi feita com websockets, rodando na thread principal. Uma thread secundária roda a interface gráfica. A comunicação entre as duas threads é feita através de uma [queue](#).

Interface gráfica

A interface gráfica foi desenvolvida com [PyQt6](#), baseando-se no [tutorial](#). Já os gráficos foram desenvolvidos com [PyQtGraph](#), baseando-se no [passo-a-passo](#).

Setup inicial

```
$ python3 -m venv env
$ source env/bin/activate
$ pip install -r requirements.txt
```

Rodar o servidor

```
$ python3 test.py
```

Em seguida, abra o [simulador](#). Clique em *Control* e em *Play*.

Selecione a forma de onda de entrada e altere a amplitude.

