UNIVERSIDADE NOVA DE LISBOA
FCT - Departamento de Engenharia Electrotécnica e de Computador
Secção de Robótica e Manufatura Integrada

Integração de Sistemas Ciber-Físicos

2021 / 2022

Trabalho Laboratorial 1

Desenvolvimento de serviços RESTful, JSON e Aplicações Web

Duração: 5 aulas acompanhadas por docente

Entrega: 22 de Abril de 2022

1. Apresentação do Problema

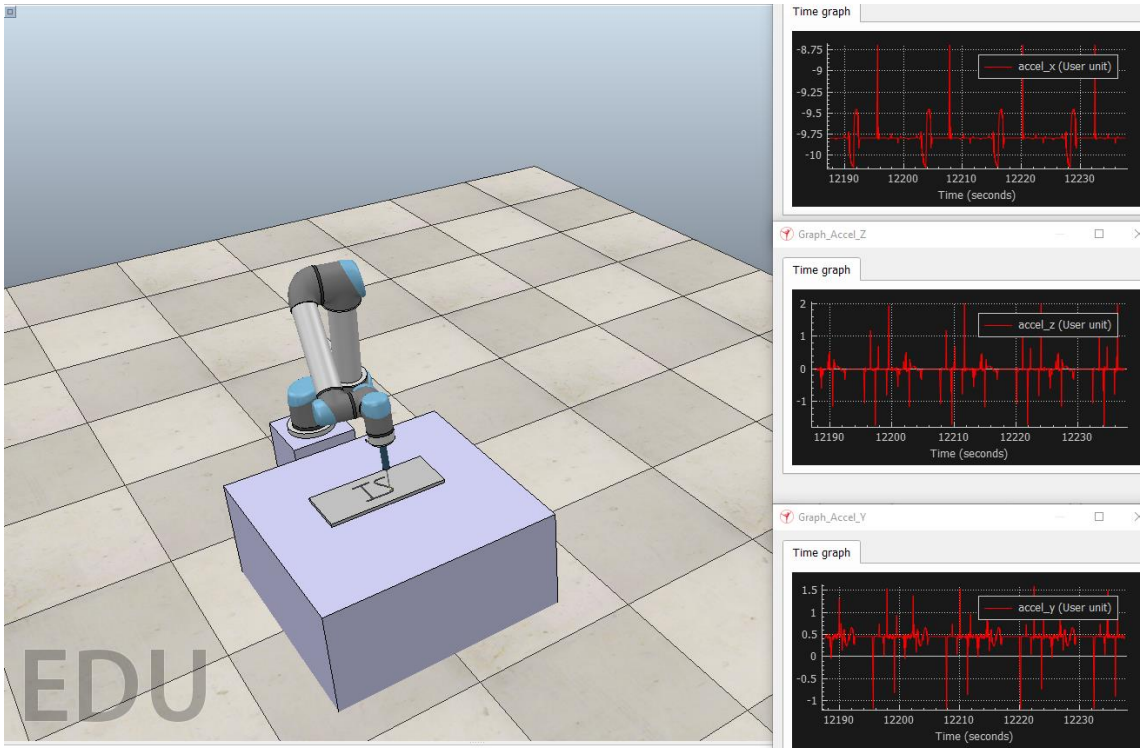

Figura 1 - Ambiente de simulação no CoppeliaSim

Pretende-se desenvolver uma solução capaz de monitorizar dados de um acelerómetro (em x, y e z) instalado num UR5 simulado no CoppeliaSim. Para este efeito, irá ser implementado um script em Python como **ferramenta de integração responsável por extrair os dados e guardá-los numa base de dados na cloud** (Heroku Postgres).

Posteriormente, deverá ser ainda desenvolvida uma ***web application*** (numa linguagem/framework escolhida pelo grupo, por exemplo utilizando Dash, React, Angular, Vue.js, etc) **que permita a interação com o utilizador** em duas vertentes. Em primeiro lugar, esta aplicação deverá possibilitar a visualização dos dados do acelerómetro guardados no Heroku PostgreSQL. Adicionalmente, o utilzador deverá ter

a possibilidade de actualizar o intervalo a que os dados são extraídos e enviados para a cloud (definido inicialmente como 1 segundo).

Infraestrutura

O trabalho encontra-se dividido em duas partes. Durante a primeira aula, cada grupo deverá desenvolver uma solução local, focando-se apenas na ferramenta de integração (módulo *Data Collection* na Figura 2) para a aquisição dos dados, como representado na Figura 2.

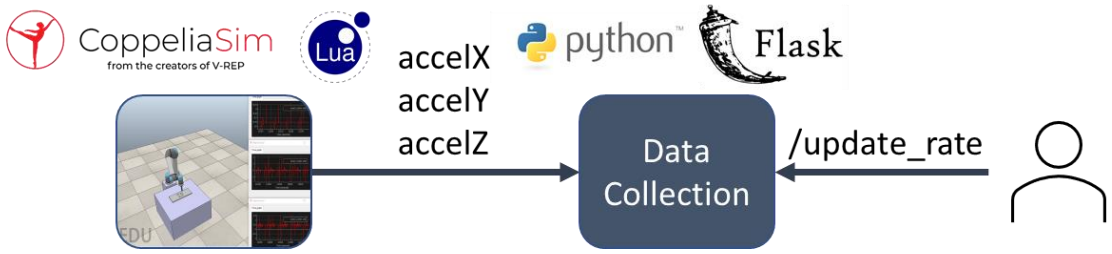

Figura 2 - Esquemático para a primeira fase do trabalho

Nesta fase, pretende-se apenas que a ferramenta de integração consiga extrair os dados da simulação, imprimindo os respectivos valores na consola, dando simultaneamente a possibilidade ao utilizador de actualizar o intervalo a que os dados são extraídos através de uma API RESTful.

Numa segunda fase, os dados deverão ser guardados numa base de dados na cloud (Heroku), podendo ser visualizados através de uma aplicação web (*dashboard*), como ilustrado na Figura 3.

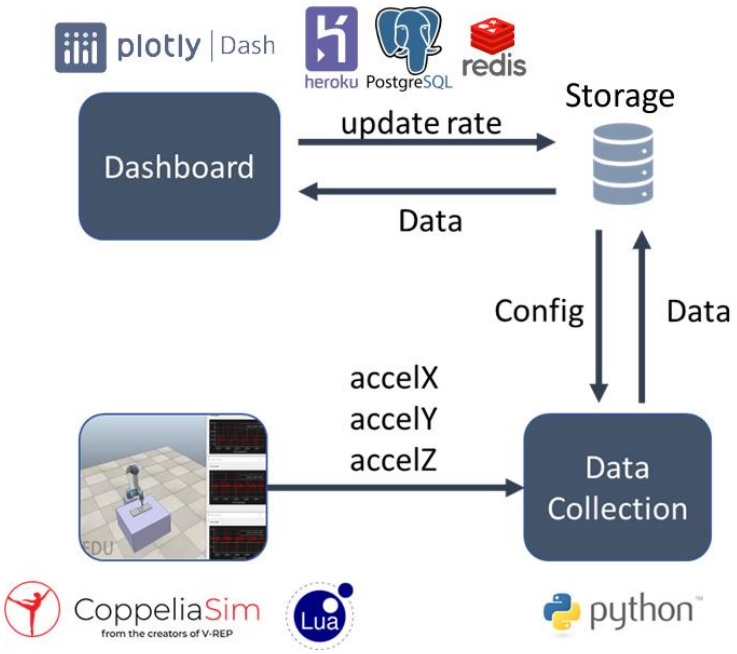

Figura 3 - Esquemático para a segunda fase do trabalho

Desta forma, após extrair os dados da simulação, a ferramenta de integração deverá enviá-los para a base de dados na cloud, sendo estes disponibilizados para visualização no *dashboard*.

A cloud deve ainda fornecer a configuração desejada para a ferramenta de integração, nomeadamente referente ao intervalo de aquisição de dados, podendo esta configuração ser actualizada dinamicamente através do *dashboard*.

2. Implementação

Na implementação pedida será utilizado o seguinte material:

- Linguagem Python 3.7 (recomenda-se a instalação através do Anaconda); https://www.anaconda.com/products/individual
- VSCode ou semelhante;
- CoppeliaSim Edu version 4.0.0; https://coppeliarobotics.com/files/CoppeliaSim_Edu_V4_0_0_Setup.exe
- Código fornecido pelo corpo docente;

Packages de Python necessários:

- Flask: https://anaconda.org/anaconda/flask
- Flask Restful: https://anaconda.org/conda-forge/flask-restful
- Flask Caching: https://anaconda.org/conda-forge/flask-caching
- Requests: https://anaconda.org/anaconda/requests
- Psycopg2: https://www.psycopg.org/docs/

Para instalar as dependências necessárias basta utilizar o ficheiro *requirements.txt* fornecido juntamento com o código base disponibilizado no CLIP, correndo o comando seguinte na directoria do projecto:

```
pip install -r requirements.txt
```

3. Planeamento das Aulas

Aula 1 – Configuração inicial e Desenvolvimento da Ferramenta de Integração

1. Siga as instruções fornecidas pelos docentes e instale o Anaconda Environment com Python 3.7. Instale também o IDE Pycharm 2019.1 ou o VS Code.
2. Estude o código fornecido pelos docentes

   Leia atentamente todo o código fornecido, familiarizando-se com as funcionalidades implementadas e as sugestões que se encontram em comentário.

3. Implemente o código necessário em Python para extrair os dados do CoppeliaSim numa daemon thread, utilizando a base fornecida pelo corpo docente.

4. Adapte o script anterior para conseguir modificar dinamicamente o intervalo de aquisição dos dados através de uma API REST.

Como referência, pode utilizar a documentação Quickstart:
https://flask-restful.readthedocs.io/en/latest/quickstart.html
Um exemplo:
https://blog.miguelgrinberg.com/post/designing-a-restful-api-using-flask-restful
Documentação de Flask-Caching:
https://flask-caching.readthedocs.io/en/latest/

Aula 2 –Integração com Heroku PostgreSQL

1. Implemente as funções necessárias para a interação com as soluções de gestão de dados na plataforma Heroku. Estas deverão possibilitar, no minimo:
    - Adicionar valores de cada eixo do acelerómetro com um identificador único e respectivo timestamp na base de dados;
    - Actualizar o valor da *config* que controla o intervalo de extração dos valores;
    - Ler o valor da *config* que controla o intervalo de extração dos valores;

2. Teste as funções implementadas e adapte o código da ferramenta de integração de forma a manter o valor do intervalo de extração dos valores actualizado com o valor da *config* armazenada na cloud.

Aulas 3 e 4 – Desenvolvimento do Front-End para Visualização

1. Implemente o *Front-End* com o dashboard de monitorização.

Alguns exemplos possiveis:

- **Dash:** **https://dash.plotly.com/**
- **React:** **https://reactjs.org/docs/getting-started.html**
- **Angular:** **https://angular.io/docs**
- **Vue.js:** **https://vuejs.org/v2/guide/**

O *front-end* deve possibilitar ao utilizador, **no mínimo**:
- Monitorizar continuamente através de um gráfico os valores do acelerómetro extraídos do CoppeliaSim;
- Actualizar o intervalo de extração dos dados (por exemplo de 1 em 1 segundo para 2 em 2 segundos) através de *input* do utilizador;

Aulas 5 – Finalização do Trabalho e Funcionalidades Adicionais

- Deployment da web application no Heroku

Funcionalidades adicionais

Algumas sugestões:

- Implementar autenticação na web application;
- Estender a monitorização para gerar alarmes caso os dados ultrapassem determinado valor limite;
- Implementar uma funcionalidade de previsão dos dados;
- Etc.

4. Avaliação

A avaliação do trabalho tem a seguinte ponderação:

- Correta implementação e demonstração de funcionamento do trabalho previsto para as aulas 1 e 2:
  - 12 valores
- Correta implementação da aplicação web (dashboard):
  - 6 valores
- Correta implementação e demonstração de funcionalidades adicionais relevantes:
  - 2 valores.

Docentes:

Ricardo Peres ra.peres@campus.fct.unl.pt

José Barata jab@uninova.pt