Visualization of a robotic arms working area

Flückiger Quentin (flucq1@bfh.ch)

November 13, 2018

Contents

1 Introduction

2 Project requirements

2.1 Vision

2.2 Goals

2.3 System context

2.4 Legend and additional information

2.5 Table

2.6 Description

2.7 Stakeholder descriptions

3 Phase 1

3.1 Environment setup / Modeling

In the first step of phase one came all the setup of the different software, API, system and models to be used during the rest of the project. I decided to use Unity to develop the application for the following reasons. I am already a little bit familiar with it, it is cross platform, the documentation is abundant, and it is rather easy and intuitive to take in hand. As the use of GitLab was suggested by Mr. Fuhrer I went with this solution to share my work and progress. But for Git to recognize the change in Unity files we need a special .gitignore. This .gitignore can be added when we create a repository with GitHub, so I decided to create the repository with GitHub and mirror it on GitLab. For this to work, we need to do change two settings in Unity editor settings. A step by step guide on how to setup github with unity can be found here https://www.studica.com/blog/how-to-setup-github-with-unity-step-by-step-instructions . The use of LaTex was strongly recommended as well, as it allows to see exactly what has been change in the documentation file, contrary to a word file for example which will erase the old one and push the new one every time a commit is done. To be able to use LaTex I had to instal Visual Studio Code, LaTex Compiler and Texlive. I had to solve a little issue where I was trying to compile with pdfTex when I had LuaLaTex. Afterward I looked for a free, open source 3D model of a Kuka arm robot. I came up with three solutions.

3.2 ADD IMAGES HERE !

I chose the number 1 as it was already rigged and therefore the easiest to animate at this stage. Before I could animate the 3D model, I decided to "purify" it as there was a lot of useless part, such as screw, bolt, cylinder. Doing so I reduced the size of the 3D model file by approximately the half. I used Blender to view and modify the file.

3.3 Animating

Once all the Software, model setup was done I could start to animate the model. To do so I used once more Blender, this time to animate the model we refined. I managed to do a few animations after a long try out, and suddenly a part of the arm was a few pixels away from the location it should be. This occurred on every animation and even the base refined model. I had to revert all changes done to the model, so all the animations were lost. I looked for another solution rather than continuing with Blender and maybe having the same issue again later. I chose to use Unity's built in animator as a solution. It worked way better than I expected, was faster to develop than with Blender as well. I created a set of three times four different animations which pick up or drop an imaginary object on the ground, in mid-air or high in the air.

- blender, unity animation
- script

4 Phase 2

4.1 Voxel world

5 Conclusion and future work

To be filled later