Software Engineering Report Skeleton

January 5, 2017

1 Introduction

1.1 Purpose of the project

This project is the course project of software engineering at Shanghai Jiao Tong University. The purpose of this project is to build a working software for image inpainting. In this project specification, we will mainly give the requirements specification, which includes 1. functional requirements 2. non-functional requirements 3. domain requirements; the software design, which includes 1. software model 2. software development tools 3. architectural design 4. object identification; testing, which includes 1. test plan 2. test-design specification 3. test-case specification 4. test-procedure specification; as well as conclusion and references.

1.2 Scope of the project

1.2.1 Project goal and task

The goal and purpose of this project is shown as follows:

1. Implement image in-painting algorithm. When it is given an image and a chosen area, it can remove the object in the area and inpaint the image.

2. Both implement the project in a traditional manner and with a probabilistic graphical model, these two models should exhibit their benefits in different situations.

3. Finally translate the prototype program into a GUI application with feasible user interaface, so that users can easily inpaint a target image.

1.2.2 Project cost

In this project, there is no too much money cost needed for the exemplar-based method. However, in order to train that probabilitic graphical model, i.e. Markov random field model, we need some number of dataset. As we searched on the internet, there are not large dataset for image inpainting and the only existing one is a dataset provided by UC Berkeley with 300 inpainted images and we will use this dataset to train the model.

1.2.3 Schedule for this project

The concreate timeline for this project is shown in the Table 1, the deadline of this project is Jan 7, 2017.

1.2.4 Hardware and software resources

In this project, we have to deal with hardware and software resources since in a software development, these resources are critical.

Hardware:

Major Mission	Date	Finished or Not
Paper research, UML design, Requirement analysis	9/15 - 11/16	finished
Prototype design, paper proposal	11/17 - 11/30	finished
Backend development of exemplar-based algorithm	12/1 - 12/11	finished
Backend development of Markov random field	12/11 - 12/25	finished
Frontend development and user interface	12/25 - 12/28	finished
Test and refactoring	12/28 - 1/3	finished
final presentation	1/4 - 1/4	finished
final report and paper writing	1/4 - 1/6	finished

Table 1: Time schedule of this project

1. Lab cluster to train MRF model
2. 3 laptops for software development

Software:

1. use Python/MATLAB as backend programming language
2. use Python Tkinter cross platform GUI framework and MATLAB builtin GUI utility to build user interface

1.3 Overview

In this final report, we will first talk about the requirement of image inpainting. Image inpainting itself is as old as human art and it is really a topic worth hard working. Actally, the image inpainting is quite useful for ancient art museums.

Next, we will discuss the architecuture of this project in a software enginnering manner. We will discuss why we design the image inpainting system in that way and how we overcome the difficulties during developing this system. We will also present some UML diagrams to illustrate our design formally.

Also, we will talk about how our users can use this image inpainting software and what is the common work flow to use our image inpaiting software. Although our user interface is quite user-friendly, users may still be confused by the software since the technique we used to implement image inpainting is quite complex.

Moreover, test specification is also given. Any modern reliable software should be tested over multiple test cases and only in this way can our software present good reliability.

At last, we will conclude this project and give several topics for future work since we find there might be more things we can discover in this research field and we will give some references of this project.

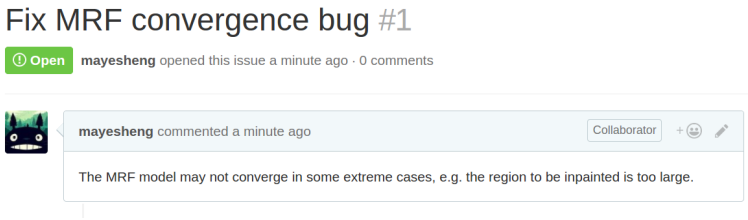

Figure 1: A sample issue tracker

1.4 Development environment & teamwork integration

1.4.1 Development environment

We implemented two different algorithms: exemplar-based algorithm and Markov random field in two platforms. We implemented the exemplar-based algorithm on Windows 10 operating system and the MRF-based algorithm is implemented on Ubuntu 16.04 Linux OS.

The proramming languages we use are carefully chosed. We use Python for the exemplar-based algorithm since Python code is easy to write and there are useful libraries like Tkinter, PIL, numpy, and scipy. We implement the Markov random field algorithm in MATLAB since it is a machine learning, to be specific a probabilistic graphical model, problem and we can use MATLAB to easily implement matrix operations and some optimization algorithms like SGD(stochastic gradient descent).

1.5 Teamwork integration

Since this is a quite large programming task and we work in a team to contribute to this project, we decide to use Git version control system to help us develop our project and actually we find we can further save our source code on cloud for convenience and safety.

The most convenient component of GitHub might be the issue tracker. Once a contributor discovers an issue, he can write a issue and another contributor can work on this issue and fix that bug. A sample issue is shown as follows: Altogether we make about 50 commits and roughly 15000 lines of code.

Cite a paper[1]

References

[1] M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In J. R. Brown and K. Akeley, editors, *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2000, New Orleans, LA, USA, July 23-28, 2000*, pages 417–424. ACM, 2000.