PyGraphics Final Report

Facilitating Multimedia Graphics Programming in the Introductory Computer Science Curriculum

Leo Alexandre Kaliazine

August 14, 2007

Abstract

At the beginning of this term the plan for PyGraphics was to create a multi-platform installer, implement Media Tools and expand the unit test coverage. As the project progressed several serious problems came to light. The projects dependency on Numeric and pygame libraries caused serious installer issues with Macintoshes and Python 2.5. This problem was caused by the fact that not all packages of Numeric and pygame worked with all version of Python. Another major problem was the inability to launch basic functions like "show" from outside a Python session (e.g. from Wing101 Python terminal).
Also, the dependencies on many different libraries caused inherent installation complexities. Installing required six different packages: Python 2.5, pygame, Numeric, tkSnack, PIL, PyGraphics. It was decided that the best way to fix these problems was to remove dependencies on pygame, Numeric and tkSnack. To make this task feasible, Turtle, Sound and Movie parts of PyGraphics were removed. All of PyGraphics' existing picture manipulation functionality was redone using only PIL. All the tasks set out in the Analysis & Estimation report were completed successfully. Limitations/Remaining tasks and Post-mortem evaluation sections are included and provide future teams information on possible expansions and improvements to the project.

Table of Contents

1. DEFINITIONS 3

2. PROJECT OVERVIEW 4

3. ACKNOWLEDGEMENTS 4

4. COMPLETED TASKS 5

4.1 TICKETS 5

5. LIMITATIONS/REMAINING TASKS 7

5.1 TURTLE, SOUND AND MOVIE LIBRARIES 7
5.2 MAC OS PICTURE TOOL THREADING 7
5.3 MULTI-THREADING 7
5.4 INSTALLER AUTOMATION 7

6. POST-MORTEM EVALUATION 9

6.1 PROBLEM: PYGRAPHICS' DEPENDENCY ON PYGAME 9
6.2 POSITIVE: PROJECT PROGRESS MEETINGS 9

7. CONCLUSION 9

REFERENCES 10

1. Definitions

Jython	An implementation of Python written in Java, running atop the JVM
CPython	The standard Python implementation written in C
pygame[8]	A Python game library that supports images, sounds, and much more
PIL[9]	A Python imaging library that provides a variety of image operations on a wide range of image file formats
`media.py` (Jython)	The Python multimedia library that makes use of multimedia capabilities in Java through Jython.
Unit Tests (Jython)	Unit tests for the multimedia libraries (currently dependent on Jython)
`media.py`[4] **(CPython)**	A version of the multimedia library implemented in CPython, with dependencies on pygame, PIL and Numeric. It is currently incomplete. (This is the pre-summer term PyGraphics implementation, saved in a separate branch)
Docstring[10]	A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. Such a docstring becomes the `__doc__` special attribute of that object.[10]
Distutils[11]	Distribution Utilities ('Distutils') exists to make the distribution and installation of Python modules, extensions and applications easier and standardized.
`picture.py` (CPython)	The branch of PyGraphics that depends only on the PIL library. This is where all changes described in this document are implemented.

2. Project Overview

The state of the project at the start of the summer term was as follows:

> PyGraphics provides a CPython, pygame[5], PIL[6], and Numeric based multimedia library that allows manipulation of images, sounds and movies. This library is based on the one developed using Jython under the supervision of Mark Guzdial[2]. The features still not implemented are Media Explorer tools, multi-platform installer and the removal of the project's dependency on pygame and Tkinter libraries.

Currently, PyGraphics provides a CPython based multimedia library with dependencies solely on PIL and Tkinter. However, the library provides only the Picture manipulation and Picture Tool functionalities; the Sound, Turtle and Movie parts of the library have been removed. From current tests, pixel access speed is much faster than it was previously and is slightly faster than Mark Guzdial's Jython implementation of the library. All but one of the previously existing unit tests pass; the one that fails has OS path related problems. (All tests pass when run in Windows).

All tasks scheduled in the Analysis and Estimation document have been completed successfully.

3. Acknowledgements

I would like to thank Paul Gries for giving a chance to contribute to the PyGraphics project and for guiding me throughout. His advice and good cheer made this project the most enjoyable experience I have had at the University of Toronto. I would also like to thank Danny Heap and the TerraToons team for all the great project discussions; they helped keep me on track. Finally, I would like to thank Jennifer Campbell for helping me test and debug the project; thanks to her PyGraphics plays well with Macs.

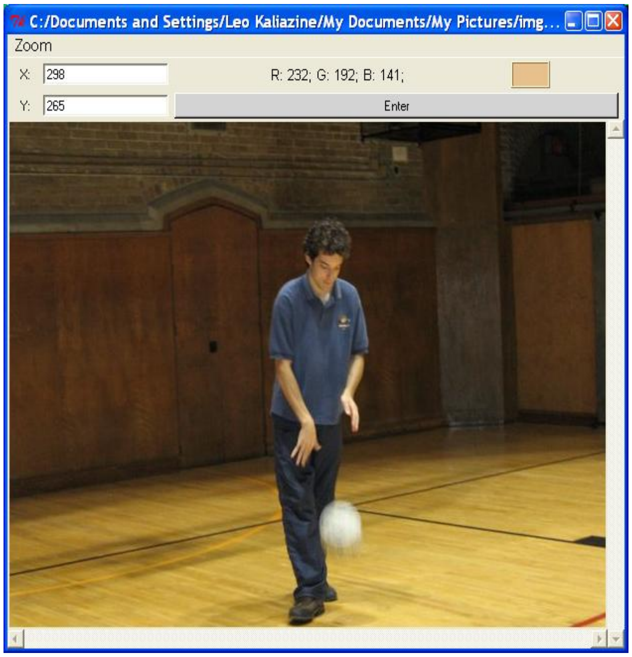
4. Completed Tasks

Task 1
Created a Google code[9] project website for the PyGraphics project. Transferred all data, wikipages and tickets currently located on the DrProject website.

Added new wikipages, Downloads and Tickets to the website, as work on project progressed.

Task 2
Implemented the Picture Explorer tool as `open_picture_tool()`. This task covers tickets #14 and #15.

The Picture tool was created as part of the warm up. It has been integrated into `picture.py`. Threading support has been added under Windows and Linux OS for this tool.


C:/Documents and Settings/Leo Kaliazine/My Documents/My Pictures/img...

X: 298

Y: 265

R: 232; G: 192; B: 141;

Figure 1: Picture Tool in action

4.1 Tickets

Ticket #14
Completed, see task 2 above.

Ticket #15
Completed, see task 2 above.

Ticket #22 – Priority High
Create a platform agnostic Distutils installer.

This was implemented in order to make it much easier to distribute the PyGraphics library. Three separate installers were created:

1. Standalone PyGraphics Distutils installer.
2. Source based, combined PIL and PyGraphics Distutils installer.
3. Windows executable PyGraphics installer.

Although the initial idea was to create a standalone platform agnostic installer for PyGraphics and Python libraries it depends on, it was found that the Distutils PIL installer is source based, adding complications the installation process. Therefore a standalone Distutils PyGraphics installer was created. The windows installer was created to make the installation process smoother for Windows users.

Ticket #25 – Priority Critical

Switch to 0-based indexing.

Picture coordinates were 1-based in `media.py` (`picture.py`). This was hard to work with. All code in `picture.py` was modified to be 0-based.

Ticket #26 – Priority Medium

Thread Support

Added thread support for `show()` and `open_picture_tool()` functions. Thread used with `open_picture_tool()` are deactivated under Mac OS due to threading problems under that OS.

Ticket #27 – Priority High

Add docstrings to top-level functions.

Added docstings to all the global methods, allowing the use of the `help()` function on all available global library methods. This makes it much simpler to learn the purpose of, as well as how to use, all global picture functions.

Ticket #28 – Priority Medium

Remove sound, turtle and movie libraries.

In order to prepare PyGraphics for the removal of all pygame and Numeric dependencies it was decided to remove all sound, turtle and movie libraries. The library was branched and moved to a new file called `picture.py`. All further work on the project was done in `picture.py`.

Ticket #29 – Priority Medium

Switch to pothole_case.

All method and variable names in `picture.py` changed to pothole case. This was done in accordance with Python style conventions.

Ticket #30 – Priority Medium

Remove pygame and Numeric dependencies

All dependencies on pygame and Numeric libraries have been removed. Currently `picture.py` depends on only PIL and Tkinter libraries.

Ticket #31 – Priority Medium

Modify UnitTests to comply with pygame removal

Fixed all existing unit tests of PyGraphics' image manipulation functions. Many fixes were needed in order for the tests to comply with the removal of pygame and Numeric dependencies.

5. Limitations/Remaining Tasks

All the tasks set out for this project were completed successfully. However, there still remain several tasks that can be completed by future teams working on this project.

5.1 Turtle, Sound and Movie libraries

The Turtle, Sound and Movie libraries were removed from the current PyGraphics implementation. This was done in order to make getting rid of dependencies on pygame and Numeric libraries feasible. The old implementation of the libraries was made into a branch and is available for any future teams to work on.

5.2 Mac OS Picture Tool Threading

Threading for the Picture Tool still needs to be implemented on Mac OS platforms. Currently, threading for the Picture Tool is done only when running the library on a Linux or Windows OS. Mac OS seems to implement threads in a unique way. Thus, what works for other operating systems does not work on a Mac OS. In the current implementation the Picture Tool will launch in a separate thread on all operating systems except the Mac OS.

5.3 Multi-threading

Threading used in the "`show()`" method needs to be fixed. In order to do this the parent thread should wait for the start of the child process. Currently a 0.1 second delay is built-in to allow the external image viewer application to launch before the parent process exits.

5.4 Installer Automation

The process for creating the PyGraphics installers should be automated. Currently there are three installers for PyGraphics. The following describes the creation processes for the three installers:

Note: For installers 1 and 3 update the existing installers as per instructions.

1. Multi-platform distutils installer. (PIL not included)
    a) Place picture.py in a directory named PyGraphics
    b) Update the setup.py file to reflect the current version number
    c) Compress the two files as a tar.gz archive
2. Windows installer. (PIL not included)
    a) Place picture.py in a directory named PyGraphics
    b) Update the setup.py file to reflect the current version number
    c) Execute the command: python setup win32

3. Multi-platform source bundled with PIL

   Note: The PIL source bundle is difficult to install, since the library is compiled from source. The removal of this installer can be considered in the future.

   a) Place picture.py in a directory named PyGraphics, which should be located in the root of the PIL library source.
   b) Update the setup.py file to reflect the current version number. Only modify the part of setup.py that pertains to PyGraphics.
   c) Compress as a tar.gz archive

6. Post-Mortem Evaluation

Work on the PyGraphics project this term was a continuation of work done by a previous team. There was only one real problem encountered that could not be solved directly: the media library's inability to run within an IDE. This problem should be kept in mind when working on future projects. Also, the weekly progress meetings were a positive aspect to the project and should be continued for the benefit of future teams.

6.1 Problem: PyGraphics' dependency on pygame

The main problem encountered was the inability of the media library to run outside of python and, thus, within any IDE. This was a major problem because PyGraphics is meant to be used for teaching first year students from within the Wing IDE 101. This problem was fixed by removing dependencies on pygame and Numeric Python libraries; however this meant that the Turtle, Sound and Movie parts of PyGraphics had to be removed. This unexpected problem could not be avoided this term. The lesson to be learned from this is that better testing by developers needs to be done at the earliest stages of any project. Specifically, testing on the eventual release platform (e.g. Wing IDE 101) should be done at the earliest stages of any project.

6.2 Positive: Project Progress Meetings

The weekly progress meetings were very useful; continuing these meetings would greatly benefit groups working on any future projects. Meeting with groups working on other projects was useful for keeping on track with project work. It also provided a valuable perspective by showing how other projects were progressing throughout the term. Finally it was interesting to learn about other projects being done, tools used by them, problems they encountered, and how these problems were solved.

7. Conclusion

Work on the PyGraphics project this term ended with all the tasks set out in the Analysis and Estimation report being completed successfully. During the course of the project many valuable lessons were learned. The most important one was that major problems can and will be encountered when working on any project. Thus any plans made in the early stages of the project must allow for such eventualities by being malleable and by having a time buffer built into the schedule.

References

1 Media Comp at Georgia Institute of Technology
http://coweb.cc.gatech.edu/mediaComp-plan

2 Mark Guzdial
http://www.cc.gatech.edu/gvu/people/Faculty/Mark.Guzdial.html,
guzdial@cc.gatech.edu;

3 Mark Guzdial, Introduction to Computing and Programming with Python. A Multimedia Approach
Prentice Hall; ISBN 978-0131176553 (December 27, 2004)

4 `media.py` (CPython) development page
http://coweb.cc.gatech.edu/mediaComp-plan/117

5 pygame
http://www.pygame.org/

6 PIL
http://www.pythonware.com/products/pil/

7 Docstring
http://www.python.org/dev/peps/pep-0257/#what-is-a-docstring

8 Distutils
http://www.python.org/community/sigs/current/distutils-sig/

9 Google code
http://code.google.com/
Google code - project website
http://code.google.com/p/pygraphics/