Python and the CBC v2.0
Scott Miller
Southern Illinois University Edwardsville
scottmil@gmail.com, www.robotgeek.net

Python and the CBC v2.0

1. Introduction

KIPR's CBC, or CBC Botball Controller, has proven to be a robust, easy to use robot controller that was originally designed for use in the Botball Educational Robotics Program [1]. In an effort to increase the CBC's versatility and potential for use in projects outside of Botball, a package for installing the Python programming language on to the CBC has been developed. This paper presents a brief description of the Python language, installation package instructions, and outlines potential future work.

DISCLAIMER: Modifying the CBC in anyway, physically or via software can cause irreplaceable damage to your CBC, and may void your warranty. Though any and every precaution has been taken to ensure the safety of your device when following the installation process described below, we are not responsible for any damage that may result from the use of the information presented in this paper.

2. The Python Programming Language

Python is a free open-source high-level programming language that incorporates many useful features as part of its standard library [2]. The language was developed in order to solve many common programming challenges with a simpler faster means, while placing a large emphasis on code readability. Python is prominently an object oriented language, though the OO paradigm is not strictly enforced.

It is beyond the scope of this paper to present the Python language in detail. However, there are several extremely helpful online resources which do. These resources include: "Dive Into Python" which provides a free comprehensive introduction to the Python language directed toward any individual with some programming experience [3]; "Python.org", the official website of the Python language [2]; and the official Python tutorial [4].

3. Installing Python on the CBC v2.0

Installation of Python on the CBC requires placing additional files into the CBC's operating system partition, "/mnt/kiss/". More specifically, it requires files to be placed into the "/mnt/kiss/bin" and "/mnt/kiss/lib/" directories. In order to simplify the process an installation package has been created, and should be included on your GCER CD. This package works in a manner similar to the Norman/Nease CBC Mod Installer [5].

Installation of Python v2.6 onto a CBC requires the following steps:

1) *Place installation package on a USB flash drive*

Unzip the CBC_Python files to the root directory, or top most directory, of a USB flash drive. This may take awhile depending on the speed of your device. Your flash drive should now contain a folder labeled "CBC_Python" which contains two folders "bin" and "lib", and a file labeled "python_installer.c" (note: you may or may not see the .c file extension).

2) *Prep the CBC for installation*

Place the USB flash drive in the USB port on the back of the CBC. From the CBC's main menu press the "Programs" button. Next, select the "File Manager" button. Finally press the "Mount USB" button in the right hand column. You should now see an additional folder in the list on the left hand side labeled "usb." If you DO NOT see this folder try removing and reinserting your USB flash drive, or rebooting your CBC and repeat this step.

3) *Running the Python installer*

Open the "usb" folder by pressing on it once followed by pressing the "Open" button in the right hand column. You are now viewing a list of files and folders on the USB flash drive. This list should contain the "CBC_Python" folder, and the "python_installer.c" program file. Select the "python_installer.c" program file by pressing on it once, and then press the "Compile" button. Once the program is finished compiling, press the "Run python_insta…" button at the top of the screen, this will begin the Python installation process.

4) *Installation Process*

The installation process should now be running. Press "A" after reading on screen messages when prompted. When the installation finishes two lines of "*" characters will appear with a message between them. If the message reads "Python 2.6.2" the installation was successful. If any other messages appear, the installation did not complete successfully. You may retry the installation or follow the instructions on the screen for more assistance.

4. Using Python on the CBC

KISS-C does not directly support Python and therefore cannot directly execute `.py` Python program files. In order to execute a Python file or a Python command the `system()` function will need to be used. As an example, to see what version of Python your CBC is running you can create and run a `.c` program file (typically with KISS-C) that includes the following function call:

```
system("/mnt/kiss/bin/python -V");
```

This function call requests that the operating system execute Python, which is located at "/mnt/kiss/bin/python" and is sending Python a parameter of `-V`, which stands for version. When the `.c` program file you have created runs you should see a message similar to "Python 2.6.2" displayed to the screen.

In order to execute a `.py` Python program you have written, you will need to first place the `.py` file on a USB flash drive or upload it into the "/mnt/user/code/" directory on your CBC. Uploading files directly to the CBC is not covered in this paper, but is discussed by Rand et al. [6]. If you placed your `.py` file onto your flash drive simply place the flash drive into the back of the CBC and mount the drive using the instructions from Step 2 of the installation process listed above.

To execute your `.py` file you will need to create and run a `.c` program file (typically with KISS-C) that contains the following function call:

```
system("/mnt/kiss/bin/python /mnt/browser/usb/filename.py");
```

Where `filename.py` is the name of the python program file saved onto your USB flash drive. Similarly, if you uploaded your program directly to the CBC the command would look something like:

```
system("/mnt/kiss/bin/python /mnt/user/code/filename.py");
```

This function call requests that the operating system executes Python, sending it a parameter that is the file location of a `.py` python program. Python then executes the Python code inside of the `.py` file.

5. Future Work

Future work with Python and the CBC should include C source header extensions that would allow users to operate a robot via the Python programming language. These extensions would provide Python access to function calls such as `mav()`, `ao()`, and `wait_for_light()`. The creation of these headers should be a relatively straight forward process.

The installation of the NumPy [7] and SciPy [8] packages would provide libraries supporting a vast array of scientific computations. These include linear algebra, N-dimensional arrays, and Fourier transforms. The inclusion of these libraries would allow the CBC to more easily support several features such as the use of scanning lasers and complex mapping procedures.

Finally, serial communication with devices connected to USB ports via Python would allow the CBC to utilize a wide variety of devices making it a truly dynamic robot controller. Though serial communication can be accomplished in other programming languages, doing so in Python in collaboration with NumPy and or SciPy would make data processing far simpler and faster.

References

[1] KIPR. Botball robotics education. http://www.botball.org, 2009.

[2] Python Programming Language. http://www.python.org, 2010.

[3] Dive Into Python. http://diveintopython.org, 2004.

[4] The Python Tutorial. http://docs.python.org/tutorial/, 2010.

[5] Rand, Thompson, and McDorman.: Hacking the CBC Botball Controller: Because It Wouldn't Be a Botball Controller if It Couldn't Be Hacked Part 1. *Global Conference on Education Robotics*, on CD, KISS Institute for Practical Robotics, 2009

[6] Rand, Thompson, and McDorman.: Hacking the CBC Botball Controller: Because It Wouldn't Be a Botball Controller if It Couldn't Be Hacked Part 2. *Global Conference on Education Robotics*, , on CD, KISS Institute for Practical Robotics, 2009

[7] NumPy. http://numpy.scipy.org/, 2009.

[8] SciPy. http://scipy.org/, 2010.