Project Proposal

Jiecao Chen

December 13, 2012

Contents

1 Project Overview

This will be a one-year software project that aims to create an environment of programmable graphical robots, which is initially designed as a tool for early programming education, but could be easily extended as a game engine or a physical simulation illustrator, etc.

1.1 Background

Nowadays, a increasing number of parents want their kid to learn basic knowledge of how the computer works. However, it is difficult to provide formal computer science education to kids because they are too young. Our goal here is to develop a kind of software that could help children learn what is programming and how to control the computer through writing code.

A significant limitation of most software is poor extensibility. For example, if a game is designed without extensibility, game players will soon be tired of it because they have become familiar with every detail.

In this sense, one of the most powerful software is the compiler or interpreter : one with enough talent and knowledge could create any program he like with the help of a compiler. One of the reasons that explain why a compiler or an interpreter could be so powerful is that, it provides a way to users to create new thing rather than operating something. Some famous software such as ***emacs*** and ***mathematica*** build their own interpreter and thus gain amazing extensibility.

By providing a build-in interpreter, our project could also be extended to be applied in areas such as physical simulation, or be adapted into a game engine.

1.2 Project Goal

We are planning to develop an environment of graphical robots with a build-in interpreter. By providing a special programming language, users of this environment could write code to control the movement of the robots (and other objects). This system could used as an platform of Early Programming Education for children. As the result of the code is vivid and immediate, children's passion and creativity will be greatly encouraged.

To be more specific, the final goal of this project is to create an environment

with 3D robots and obstacles. The robots may have legs, arms, heads, eyes, and some other characteristics, also the build-in interpreter will support a large number of programming features, such as for-loop, while-loop,variables, function, recursion, OOP, etc.

Another good news is, with slight modification, it could be applied in the area of Physical Simulation, or be a game engine.

2 Project Objectives

We will describe specific features that will be developed in our project in this section, they would be further explained in section Timeline of Deliverables. In our final deliverable iteration, there will be several controllable robots in the graphical screen, they will be look like this (by providing parameters, their appearance can be changed).

They are all 3D robots and their facial expression and their movement could be controlled by code. We will create a specific programming language to control the movement of robots. A sample is shown in figure 1.

Figure 1: Sample Robot

There will also be a target and several obstacles in the screen, children can write code in a text file, e.g cmd.txt, and the program will read commands from

that file and execute them thus control the movement of robots. Kid should use their code to make the robots avoid obstacles and reach the goal.

Also, children can also use this specific programming language to create an interesting animation. I strongly believe that every kids will love this game/tool.

3 Timeline of deliverables

We separate the whole project goal into 4 iterations, each iteration will be described as below.

3.1 Iteration 1

Time: Sept 10, 2012 - Oct 10
Following concrete objects will be realized.

- Create an graphic window using openGL
- Basic shape of the robots (2D)
- Robots could be controlled by keypress

3.2 Iteration 2

Time: Oct 10, 2012 - Dec 10
Following new features will be added.

- A basic interpreter dealing with a limited programming language.
- Add obstacles, target into the environment.
- Robot can be controlled by keypress and corresponding commands will be output
- a small tool called "convert2readable" will be created to make automated-output commands more readable. Child can learn how to write code by imitate those code

Description of the Programming Language:
Following feature will be supported:

- for-loop
- a set of commands, including
    1. stepDirection(float)
    2. moveForward(float)
    3. moveBackward(float)
    4. stepSpeed(float)

A sample code may look like this:

```
moveBackward(10)
stepSpeed(10)
for 0 to 10 do
   moveForward(3.0)
   stepDirection(0.2)
endfor
moveForward(10)
```

3.3 Iteration 3

Time: Dec 10, 2012 - June 10, 2013
Following new features will be added.

- The interpretor will be improved
- Robots will be extended to 3D, with controllable legs, arms, head
- New atomic commands will be added into the programming language
    1. raiseArm(left,right, angle)
    2. raiseLeg(left,right, angle)
    3. moveHead(angle)
    4. jump(height)
    5. run(speed, sec)
    6. changeObservePoint(x,y,z, direction)

- New language feature will be added
    1. recursion
    2. define function
    3. function nesting

3.4 Iteration 4

Time: June 10, 2013 - Sept 10, 2013
Following new features will be added.

- The interpretor will continue to be improved
- Robots will have face expressions
- New atomic commands will be added into the programming language
    1. moveEyes(...)
    2. moveNose(...)
    3. moveEyebrow(...)
    4. etc
- The Programming Language now supports variables, micro.
- The appearance of robot could be controlled by parameters.

4 Overview of Current State of System

4.1 Snapshot of Current system

Our current state of system has already met most requirement in iteration 2. The following is a snapshot of the graphic screen.

4.2 How to Control

There two ways to control to robot, the first one is by keypress:

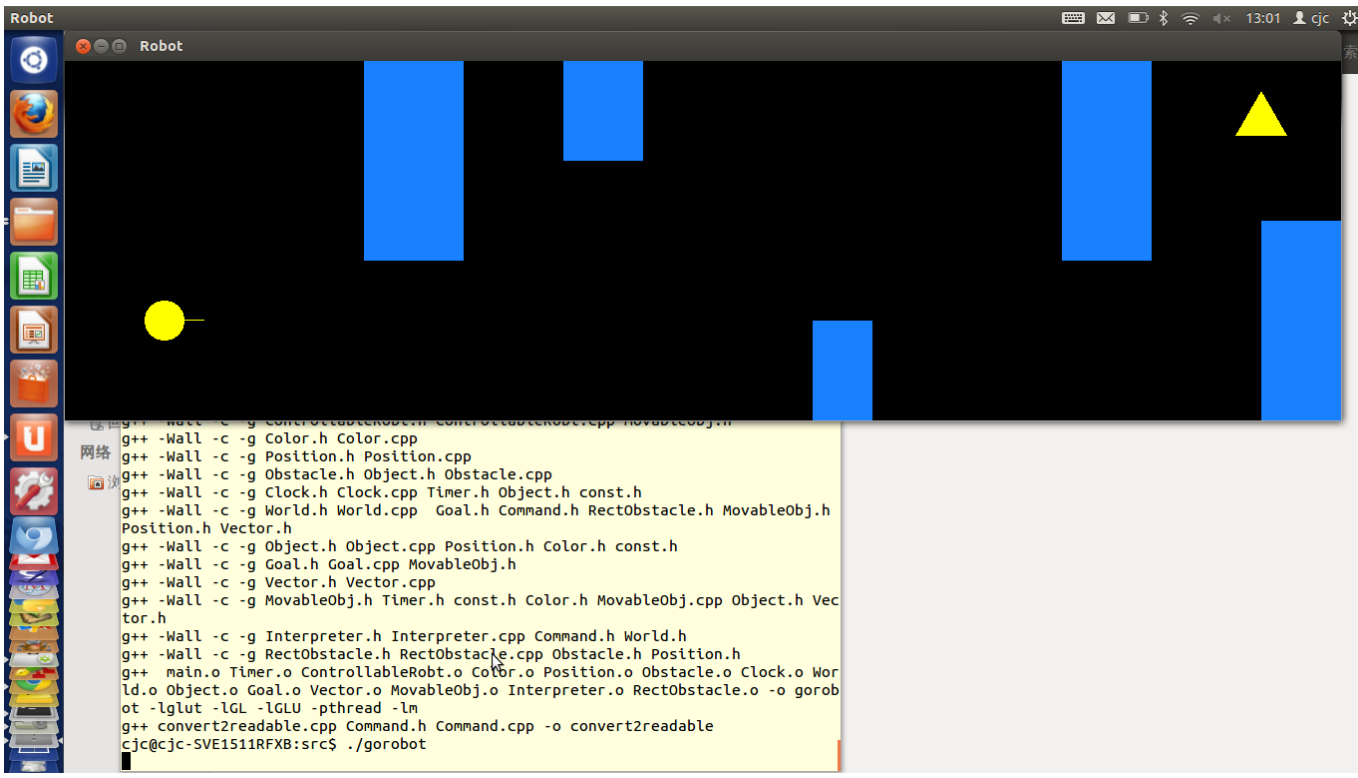

Figure 2: Snapshot of current system : iteration 2

After running the command "**./gorobot -k > cmd**" in terminal, as showed in the snapshot, there is a robot on the graphic screen, and also there are some obstacles. Users could control the direction and the speed of the robot with the key up, down, left, right. The detail is listed below:

- F1 : move forward
- up key : increase the speed of robot with a fixed rate
- down key : decrease speed of the robot with a fixed rate
- left key : change direction of robot, counter-clockwise
- right key : change direction of robot, clockwise

When the robot is controlled by keypress, corresponding code will be automatically printed into a text file, in this case, it is "cmd". Users can use the small tool "**convert2readable**" to make those code more readable, by running "*./convert2readable cmd*". Children can learn how to write code through those sample.

The second way to control the robot is using source code file directly. For example, if we have a source code file named ”cmd”, we can run ”**./gorobot** < **cmd**” and press F2 to execute code in file ”cmd”.

4.3 UML of Current System

Figure 3 is an UML of current system

4.4 Improve from Current System

By adding more language feature and make the appearance of robot more complex and vivid, the current system will basically meet the requirements of iteration 3. And by adding more language features and providing more commands for robots, adding facial expression supports, we will finally meet the goal of iteration 4.

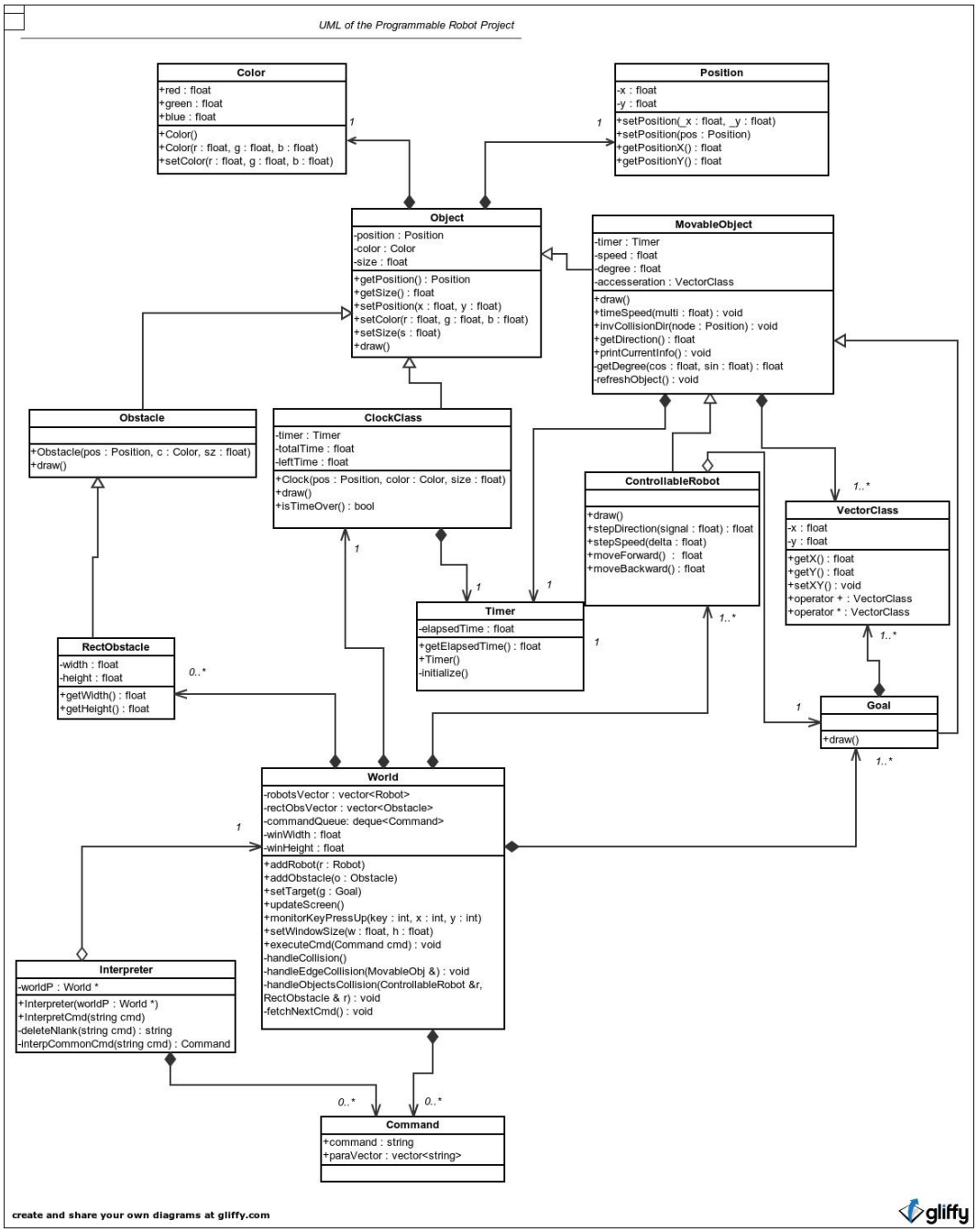

Figure 3: UML of Current System : iteration 4