Pocket Mole

Team: Little Cutie

Team members: Yihan Li, Gaomeizhu Qu,Wenqian Ma

ABSTRACT

Team little Cutie has designed an intuitive, visually appealing pocket game called Pocket Mole, which can improve the player's reactive potency and relax the player's mind and body. A Gumstix Verdex board is used to communicate with the computer program as well as the the Samsung 4.3'' touchscreen LCD panel. This board's function is to transfer the various pictures(mole) from computer to the touchscreen so that the player can select which mole he hates most to beat down. The project required writing various use level c and c++ programs and the development of a QT based GUI. Although the project was success and met the initial expectations, there is room for improvement in terms of game's complexity and ornamental degree.

INTRODUCTION

Topic:

The goal of Team Little Cutie's EC535 final project was to create a visually appealing pocket game which is easy and flexible to attract people from all age groups to play. Besides, this game muse be funny and amusing in order that the players can relax themselves from the heavy pressure of work or in life. Based on this idea, we have reminded the pocket game Whac-a-mole, a game invented in 1976 by Aaron Fechter of Creative Engineering, Inc. Since we have gained enormous pleasure in this game in childhood, each member in our team believed it will inspire others' interests and arouse their memory of childhood.

Motivation:

The motivation for this project drew fundamentally from our team's interest in pocket games and the potential of implementing a feature-full device.

From an academic standpoint, designing, organizing, and implementing the Gumstix and QT software provided a number of useful lessons that enlarged skills we

have already learned through EC535 homework and labs. For example, this game needs a time countdown function which is similar to the lab2 and lab3's requirement we have accomplished. Furthermore, the implementation of our project's goal required working with outdate hardware and outdates software, which can expand our skill of embedded system and help us to understand this course deeply.

Actually, each team member wanted to realize a physical object from a familiar pocket game such as Tetris or Flying Bird. We all were longed to create a game to call the the players' memory of their childhood because all of us were deeply missing the interests gained when we were children. Obviously the game Pocket Mole is the best choice for us as well as helping us to explore several interesting product design and prototyping problems.

The Big Picture:

After the initially learning of QT software and considering the group's skill level, time limitations, and the requirements of the final project, Team Little Cutie created the following design overview for the Pocket Mole:

The game has three elements: One, a visually attractive and intuitive user interface including the page of picking mole, the page of randomly showing the mole and the page of ending; two, the time countdown module, which countdown the time from 30s to 0s; three, the score counting module: incrementing the score when the player beats the right mole.

Accomplishments:

We were capable of accomplishing most of our state goals, including let the player pick the mole and making the mole randomly show in the screen, and the counting score speed can timely follow the beating mole speed.

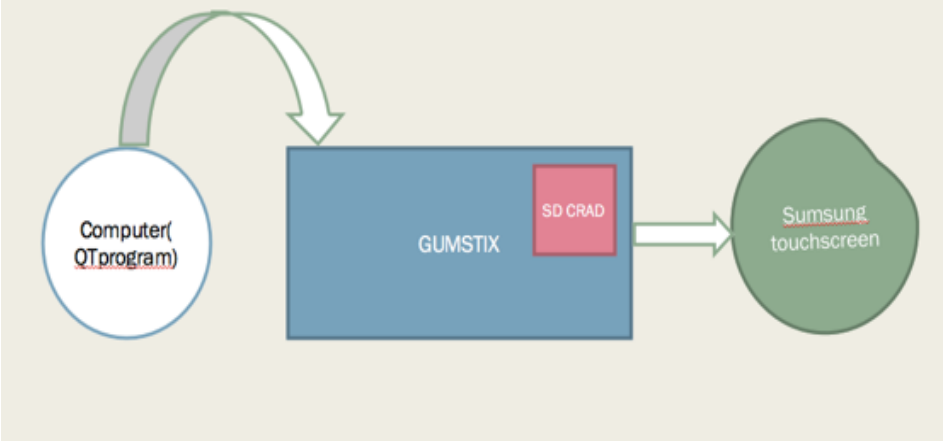

Figure 1: Hardware flow chart

Design Overview

Gaomeizhu Qu has written the picking mole page and ending page. She has loaded three moles into the game to be selected by the players, and she makes the text 'Gamever Over, You score is:' and the play again button pump out when the game ends. Besides, she has calculated the mole's position to let them show in the hole of the scene. Also she implemented the timer function and the frame of the whole program.

Yihan Li has set the window title to show the text 'Pocket-Mole time: score:' so the the player can see the time countdown and the score they have gained timely when they are playing the game. Besides, he has set the cursor as a hammer and added the whole scene into the game, which can promise the mole will show in the right hole.

Gaomeizhu Qu and Yihan Li cooperated with each other to write the mole randomly show module: Dividing the screen into 18 parts and the mole will randomly show in one of the parts. Besides, they have set the mole's sad picture to show when it is beaten.

Wenqian Ma has written the time countdown and score counting module: setting the time to 30s and countdown it to 0s; the score will increment when the mole is beaten.

Project Details:

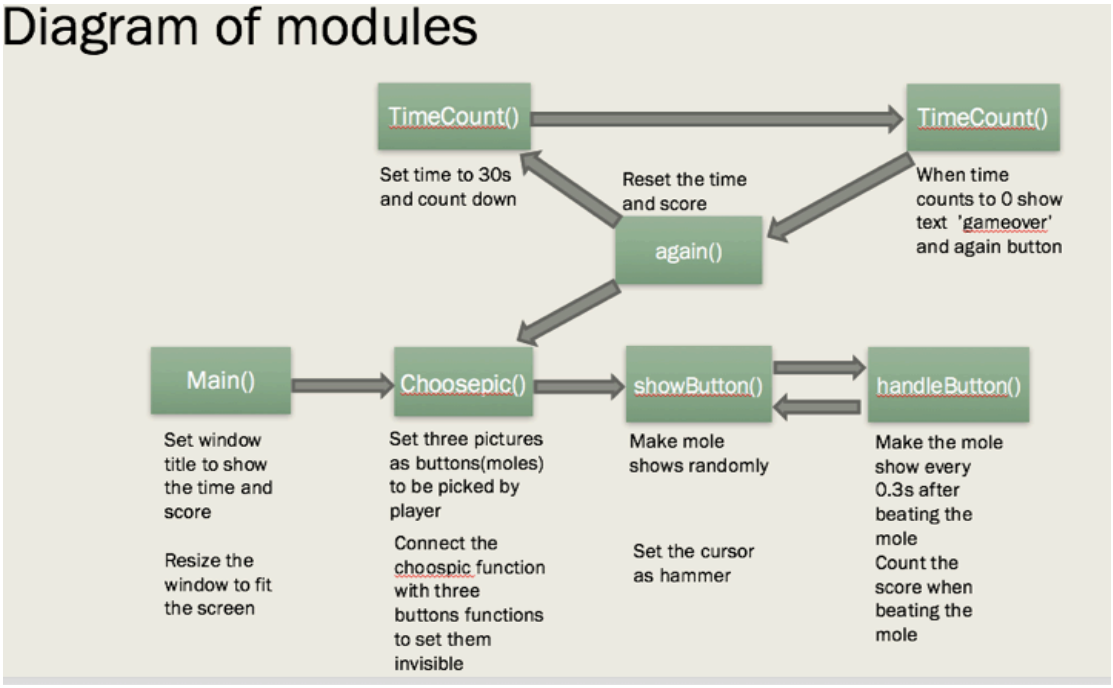

Figure 2: Diagram of modules

Preliminary research:

Most of the preliminary research for this project revolved around the Gumstix board and the QT software. As we have completed the lab work, we are familiar with the Gumstix command including how to compile the file, how to load the file and images, and how to run the file. But there were something confusing us in QT software: the image could not show in the screen which has been loaded in Gumstix, we have tired some libraries such as QPainter, QPixmap, and QLabel, and we finally found it need three levels in the program if we wanted the picture to show in the screen: main.cpp->scene->game.cpp, every background or items should be added in the scene if we want them to show, since the Gumstix will be confused if let it just directly read the image or the module from the game.cpp.

After that, we continue to do the next work: Creating different modules for different functions, such as defining the QTimer to make time countdown module or defining the QPushButton to make the mole's randomly showing module. By the way, the reason why we use QT Creator to create an original GUI rather than modify existing code is that Yihan developed an executable using QT Creator capable of visualizing on the LCD.

Emulating Modules in QT Creator:

Page1

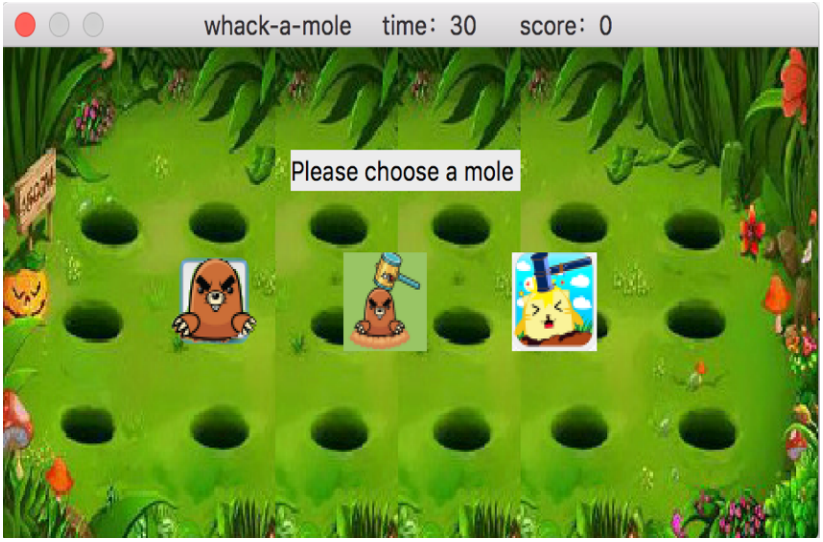

Figure3: Starting page(Main()&&Choosepic())

Main(): this module's function is to set the window title: showing the the text 'whack-a-mole time:30 score:0' to inform the player the game will start, another function is to resize the screen, because if the screen size is to large the 4.3'' touchscreen may not be able to show the whole background.

Choosepic(): this module's function is to show three moles in the screen to be picked by the player, when then player has selected the mole, the game will start.

Page2

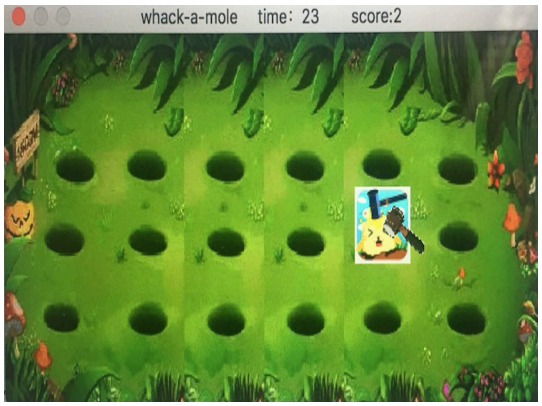

Figure4: Playing page(showButton(), handleButton(),TimeCount())

showButton(): this module's function is to make the mole randomly show in the screen. First, we used the code

```
static int position_x[18] = {25,100,175,250,315,385, 25,100,175,250,315,385,
25,100,175,250,315,385};
static int position_y[9] = {55,100,155,55,100,155,55,100,155};
```

to set define the mole's position as well as to divide the screen into 18 parts(the screen size is 480*240); then we use the code

```
int x = qrand()%18; int y = qrand()%9
```

to gain two random number x,y; finally we use the code

```
mybutton -> setGeometry(position_x[x],position_y[y],50,50)
```

to make the mole randomly show in the screen(the mole has been set as button). Besides, another function of this module is to set the cursor as hammer.

handleButton(): This module has three functions. One, when the flat 'hit' is 1, which means the mole has been beaten, the score will increment 1; two, let the next mole show after the previous mole be beaten 0.3s; three, change the mole's face when it is beaten, which means to use another button to replace the previous button.

TimeCount(): this module's function is to set time to 30s when the game starts and then countdown to 0s, and to send the text 'Game Over, You Score is:' and play again command to again() module. Besides, it will refresh the window title every second.

Page 3

Figure5: Ending page(again)

again(): This module's function is to receive the command from TimeCount() module so as to show the text and 'AGAIN' button when the game ends.

Experiment in touchscreen:

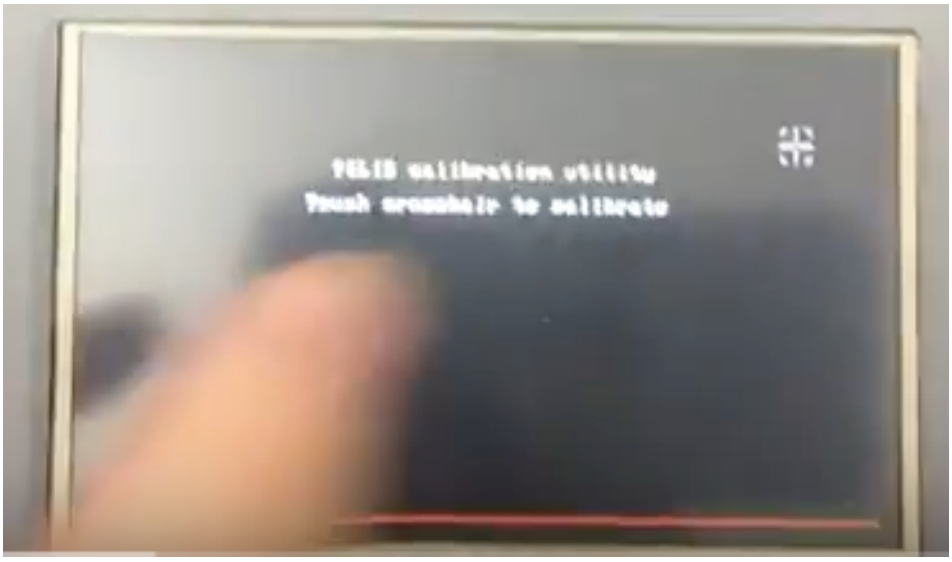

Figure6: Calibration

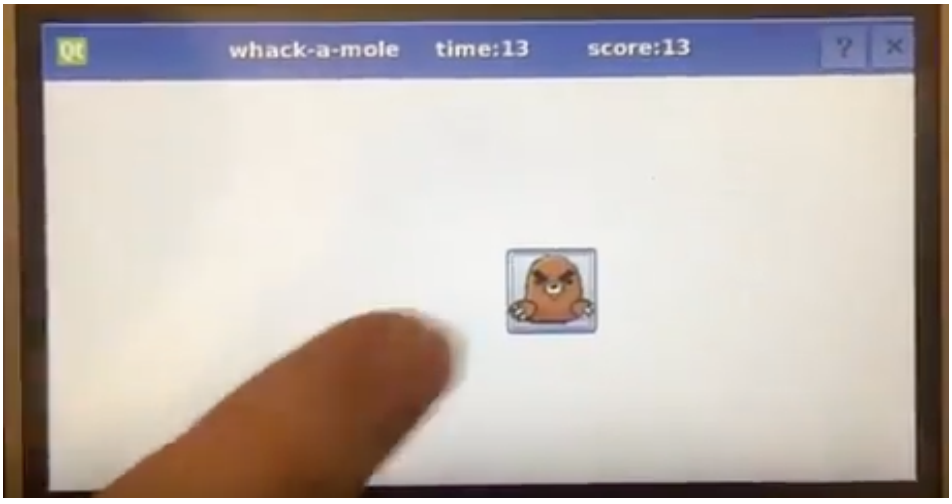

Figure7: Experiment

As the picture shows, the functions of the game works well, the only difference between the emulation and experiment is that the background is not shown in the touchscreen, that's mainly because we have not added the scene into the program, so every item is added in the game not in the scene, which makes the Gumstix confusing to read which picture as scene.

Summary:

Although the Pocket Mole is only a proof of principle game, we feel that the features we have incorporated into it make it extremely marketable. The goal of our project was to create an intuitive, visually appealing game to attract people from all age groups to play. The final "Pocket Mole" achieved most of out goal, it loads the images by Gumstix from computer; makes the mole randomly show in the screen;

shows the score timely when the mole is beaten. I'm sure it will gain the great interests of the players and recall their honey memory of their childhood.

However, there still remains some defeats in the game. For example, the background can't show in the touchscreen, the mole should automatically show in despite of being beaten or not. Besides, there is more that can be done: we can add some music and sound effect in the game.

Reference:

1. Qt Documentation: http://doc.qt.io/qt-5/qtexamplesandtutorials.html
2. QT Game tutorial: https://github.com/MeLikeyCode/QtGameTutorial