Wifi based Multiplayer Game

Guanghan Ning , Lingshuang Wu, Yi Shang

Computer Science, University of Missouri-Columbia
gnxr9@mail.missouri.edu
Lw2mf@mail.missouri.edu
shangy@missouri.edu

Abstract—*As the smart phones running Android OS become more and more popular, the limitation of the network connection comes out to be a problem. With the idea of adhoc network, we decided to develop a multiplayer game Aero Plane Chess based on wifi to test the performance of the wifi network. The game is implemented by two part: the game part and the wifi network part. For the game part, we implemented a turn based game which allows users to roll the dice and choose one of four planes of his to play. Besides, we implement some rules such as jumping and killing others; for the wifi network part, we used centralized mode which makes synchronization simple. Besides, TCP was utilized on top of wifi. According to the game performance, we found that wifi has high speed to support phones' communication and adhoc network could support multiplayer applications.*

Index Terms—*multiplayer, game, wifi, adhoc network, android*

1. Introduction

With the blooming popularity in mobile market, smart phone dramatically change people's life style. The combination of portable device and powerful server services enable smart phone perform a variety of tasks. Though the cloud and client approach is widely applied, such services are limited by network connection. Despite the fact that Ad-hoc network through wifi and bluetooth is supported by the Android lower level system, smart phone's capabilities of setting up ad-hoc network that allows peer communication are not utilized.

Previous REU students have developed a game based on Blue tooth. However, it is limited to two players and Blue tooth has low speed. So, we would implement a wifi based multiplayer game – Aero Plane Chess – which has high speed communication.

2. Game rules

Basically, this is a turn based multiplayer game.

2.1. Basic rules

There are four different colors on the map as shown in Fig 1. and one player represents one color. Each player controls four planes with the same color as his. The goal is to have all four planes reach the destination. Each player has a turn to roll the dice and choose one of his four planes to move.

2.2. Specific rules

The player can only roll the dice and choose a plane in his turn. In his turn, the player can only roll dice when it is in Roll dice mode and the player can only choose a plane when it is in Choose plane mode.

2.3. Advanced rules

At the plane's first stop, it will check the color of the block, if the block's color is the same as the plane's color, the plane will move forward to next block which has this color; If two or more planes with different colors arrive at the same block, the previous ones will be sent to home(More specific rule: if this happens at the plane's first stop where it is supposed to "fly", it won't "fly" any more)

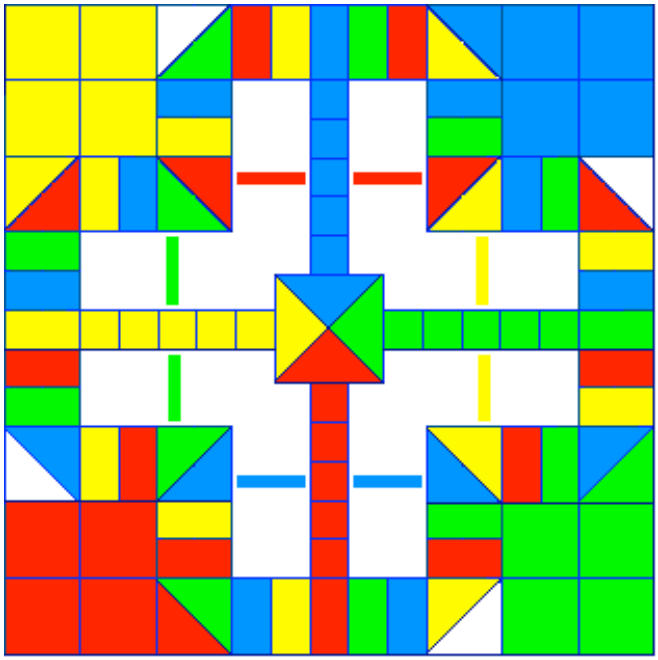

Fig. 1. Game map

3. Game design

For the whole game, we have three cycles working together. They are game life cycle, plane life cycle, and dice life cycle.

3.1. Game life cycle

As shown in Fig 2, for the whole game loop, we have designed three statuses which include Roll Dice, Choose Plane, and Start Moving. When the game is started, the status is Roll Dice. At this point, the game will wait for the user to roll the dice. Whenever the game captures the event that the right user has clicked on the dice, it changes the status to

Choose Plane and wait for the user to click on one plane. As long as the user clicks on a legal plane, the game status will change to Start Moving which means the chosen plane is moving now. Whenever the running plane finishes its turn, the game status will change back to Roll Dice and wait for next user to click on the dice. The loop keeps working until the game is over.

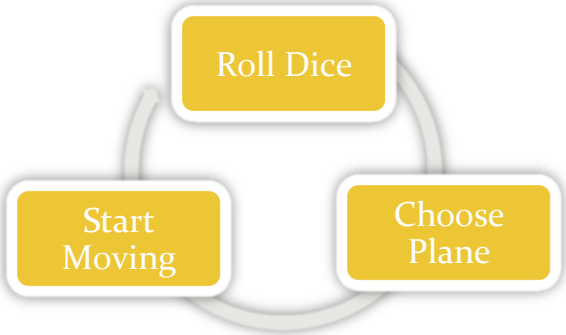

Fig. 2. Game life cycle

3.2. Plane life cycle

As shown in Fig 3, for each individual plane, it has three statuses which are Start Moving, First Stop, and Finish Turn. When the plane is not activated, the status is Finish Turn. When it receives the event message saying that game status changes to Start Moving, it will change the plane status from Finish Turn to Start Moving, and at the meantime, the plane starts moving on the map. Whenever the plane stops in this situation, it changes the status to First Stop and at this point, the plane checks if others with different color from its are on the block it steps on. If some are, it kills them (send those planes back to home). Besides checking planes, it also checks the color of the block. If the color of the block is the same as its color, it will move forward to the next block which has the same color as its. If at the first stop, the plane cannot move forward, it will change the status to Finish Turn itself. Otherwise, it will keep moving until it stops again and changes the status to Finish Turn. When the status is at Finish Turn, the plane will check other planes' locations and the color of the block again and deal with those situations.

.

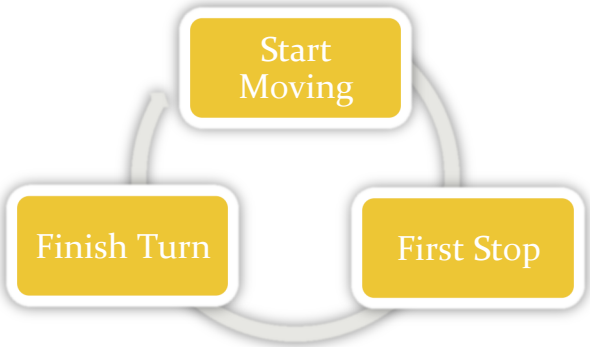

Fig. 3. Plane life cycle

3.3. Dice life cycle

As shown in Fig 4, for the dice, it has three statuses which are Display "?" picture, Display randomly, and Display the player's dice number. As the name indicates, the first status will display a question mark on the screen, the second status will display ten random number one by one as if the dice is rolling, and the third status will display the number generated by the user as the moving steps for the plane. At the very beginning of the game, the status is at Display "?" picture. When the user clicks on the dice, it generates a random number and turns the status from Display "?" picture to Display randomly. Then it will display ten random dice pictures. After that, the status changes to Display the player's dice number. Whenever the running plane finishes its turn, the status will change back to Display "?" picture again. This process will keep working until the game is over.

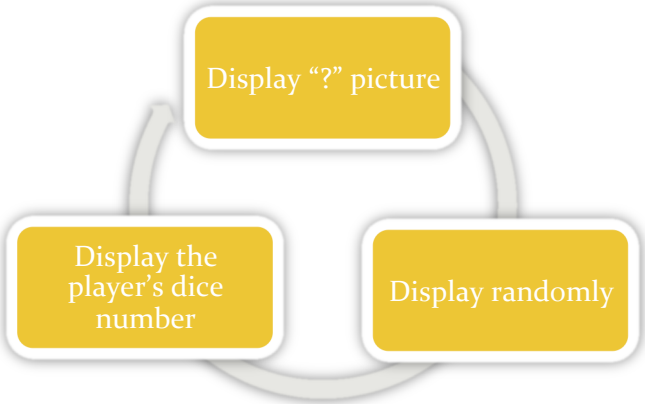

Fig. 4. Dice life cycle

4. WIFI DESIGN

Basically the wifi can be implemented in centralized mode or distributed mode. We will compare these two modes here.

4.1. Centralized mode

In the centralized mode, the first player is supposed to be a sever setting up the wifi network. Other phones will join the network by setting up the connection between itself and the sever, which means they turn out to be clients. When it is the sever player's turn, it will send all the necessary data to clients. When it is the client player's turn, it will send all the necessary data to the sever first and the sever will send the data to other clients.

As it is shown in Fig 5, the sever is the centre of the whole wifi network controlling the communications between each clients.

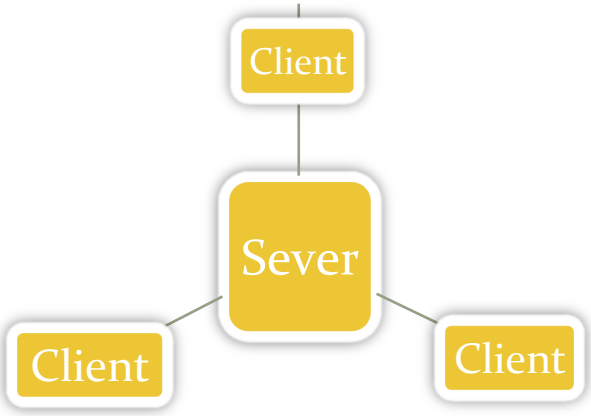

Fig. 5. Centralized mode

4.2. Distributed mode

Unlike the centralized mode, in the distributed mode, each player is supposed to be a sever as well as a client. When it is a player's turn, this phone will perform as a sever sending all the necessary data to other phones. When it is other phones' turns, this phone will perform as a client receiving message from others. By this way, all the phones are connect equally.

As it is shown in Fig 6, there is no centre in the whole wifi network. All the phones are equally connected.

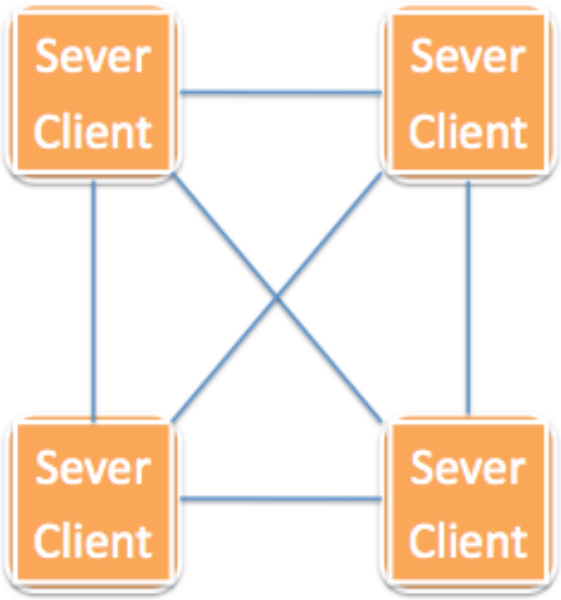

Fig. 6. Distributed mode

4.3. Comparison of centralized and distributed mode

Centralized mode is easy to be implemented and synchronization is good. In the centralized mode, the minimum connections between four phones can be three which is simple to work on. Besides, only one sever to manage communication makes synchronization good. However, the disadvantage is that in the whole network, all the phones are not completely equal which is a little against to the idea of adhoc network.

The distributed mode overcomes this shortcoming, which has every phone equally exist in the network. But it needs more connections between phones. Besides, since each phone will send data to other phones independently, the synchronization is difficult to manage.

Comparing the advantages and disadvantages of these two wifi modes, finally we choose to use centralized mode to implement the network.

4.4. TCP implementation

Based on wifi API, we use TCP to implement data communication. TCP, as a mature way in network connection, is easy to be implemented in Java and has low rate to lose data.

4.5. Data communication

The data communication is very simple in this game. Only user ID, running plane number, and dice number are supposed to send to other phones to finish one turn.

After one player generates his dice number and chooses his running plane, the phone sends user ID, dice number, and running plane number to sever, and sever will send the message to all clients.

No matter client or sever, there is always a thread trying to receive message. Once it receives not null message, it will store them for further use.

5. Experimental Results and Analysis

Based on all the theories above, we implemented a 2 player Aero Plane Chess on wifi adhoc network. The performance shows that there is almost no latency in the communication. As long as one phone sends all the necessary data to the other one, the other phone does correct actions immediately.

6. Conclusion

According to the game performance, we found that multiplayer game can be implemented based on wifi network on Android phones and TCP works very well on adhoc network as well. The wifi speed is high enough to support general turn based games.

7. What We Learned

Throughout this small project, we learned the basic techniques to design and implement a game on Android OS. Besides, this learning experience enhanced our understanding of algorithms behind adhoc network. This project has also cultivated our research spirits in exploring unknown world with limited knowledge and time constriait.

8. Future Work

Building up on our current implementation, we would like to improve the game interface. Besides, we also want to implement some other specific game rules and expand the game to 3+ players. Distributed connection mode is also a valuable way to work on.

REFERENCES

[1] Zechner, Mario. *Beginning Android Games*, pp. 185-227. Springer Science+Business Media. 2011: New York.

[2] "Sockets programming in Java" , 1996 http://www.javaworld.com/javaworld/jw-12-1996/jw-12-sockets.html

[3] Tiancheng, Zhuang. "Wifi single hop networking java library documentation", unpublished.

[4] "Android Game programming" http://www.edu4java.com/androidgame/androidgame.html