Complex Games and Palm Computers

Pieter Spronck and Jaap van den Herik
Universiteit Maastricht/IKAT

Abstract: Today's Palm computers face practical restrictions when playing complex games. This article deals with both the functional as well as the hardware limitations of a Palm computer and their impact on game design. It is observed that the designers of complex games refuse to take sufficiently into account the imposed limitations. Seven design problems are identified and examples of these problems in actual games are given, including ways of solving them. Consequently we propose a list of five design rules which may alleviate the problems. Finally, we conclude that now the time is ripe for one-person Palm game businesses. We expect this to last only a few years.

Key words: computer games, game design, palm computers

1. COMPUTER GAME DEVELOPMENT

Originally computers were installed for business purposes. However, with the introduction of home computers games became a major application. The first games were produced by solitary home-based game programmers. Later on, companies dedicated to game programming were founded, and the first game development teams were constituted (Levy 1994). While those teams in the late eighties consisted of five to ten people, in the second half of the nineties game development teams encompassed sometimes hundreds of people, all with specialized jobs. Nowadays, the total budget invested in the development of computer and video games is on a par with the budget invested in Hollywood movies. Just like aspiring amateur movie makers cannot hope to create a box-office hit with their meagre means, amateur programmers can safely assume that it is impossible for them to create a commercially successful computer game. That is, until recently...

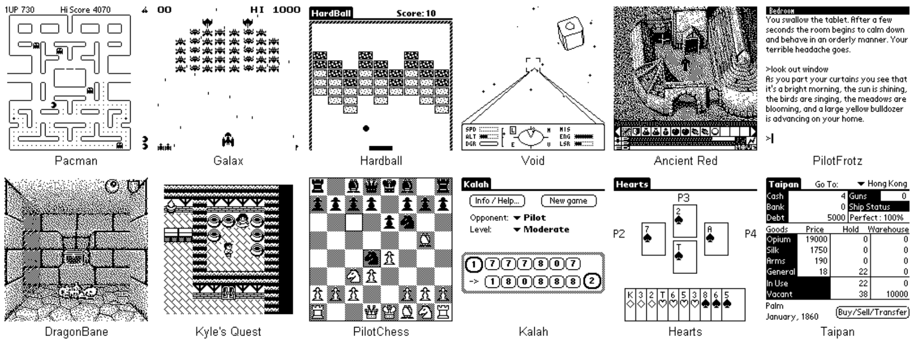
Figure 1: The Palm III

After the release of the Palm III (see figure 1) in 1998, handheld computers have become all the rage (Williams 1999). A Palm is a small, programmable computer of which the main input device is a so-called “stylus”, a pen by which characters can be written on a special pad and that can be used to “tap” control areas on the screen. The Palm was neither the first nor is it the last type of handheld computer, but it is, so far, the most successful one. Starting out as a replacement for the electronic agenda, newly developed applications turned Palms into calculators, web browsers, notebooks, email-managers, translators, e-books, databases, barcode-scanners and even remote controls. Of course, games were not left out. There are thousands of Palm games available, often as shareware or freeware. It is noteworthy that most Palm games have been developed by either a single programmer or a team of at most three people. The limited hardware capabilities of the Palm enforce a simple, concise, sparse game design, which a solitary programmer can implement just as well as a professional team. For the design of a new game one can be inspired by the multitude of games that have seen the light of day since the first release of PONG. Therefore the current Palm gaming world is an ideal place for an aspiring game programmer to leave his mark. It is even possible to start a small one-person business in Palm games.

In this paper we observe that while it is easy to design a simple game for a Palm, complex games are prone to design problems. Our objective is to formulate a set of design rules, which can be used to design well-behaving complex Palm games.

2. PALM GAMES

Investigating the domain of current Palm games, we see games of many types and qualities (see figure 2). Most games are rather simple, finding their inspiration in games from the early eighties. Recently, especially since the advent of colour Palms, more complex games have entered the Palm scene. The evolution of games on Palms goes fast, and developers try to catch up with even the latest PC games. Most Palm games have a simple design. This simplicity is caused by two factors: the Palm’s hardware limitations and its functional limitations. The hardware limitations come in three forms:



Bedroom
You swallow the tablet. After a few seconds the room begins to calm down and behave in an orderly manner. Your terrible headache goes.

>look out window
As you part your curtains you see that it's a bright morning, the sun is shining, the birds are singing, the meadows are blooming, and a large yellow bulldozer is advancing on your home.

>|

Taipan Go To: ▼ Hong Kong

Cash	4	Guns	0
Bank	0	Ship Status	
Debt	5000	Perfect : 100%	

Goods	Price	Hold	Warehouse
Opium	19000	0	0
Silk	1750	0	0
Arms	190	0	0
General	18	22	0
In Use		22	0
Vacant		38	10000

Palm
January, 1860
Buy/Sell/Transfer

Figure 2: Several Palm Games

1. The Palm screen has a low resolution and supports, for a majority of the Palms in use, only greyscales.
2. The Palm has a small amount of memory that must be used not only for running but also for storing programs when they are not in operation.
3. The Palm has limited input devices, namely a stylus and six hardware buttons (shown at the bottom of the device in figure 1). The main function of the hardware buttons is to switch to other applications and therefore they should not be reprogrammed light-heartedly. This leaves the stylus as the major input device.

The functional limitations of the Palm are the result of the way Palms are used. The main goal of Palm computers is executing quick, simple tasks, anytime, anywhere. One of the reasons the Palm is far more successful than its derivatives is that the Palm hardware, as well as the Palm OS, is geared towards such quick tasks (Rhodes and McKeehan 1999). Because of this general Palm philosophy clarity of the interface and ease-of-use are important requirements for Palm games. It is easy to design a simple game adhering to these requirements, but for a complex game it is a difficult task.

A survey of the few complex Palm games that currently exist shows that most of them contain design problems. The basic cause for these problems is the struggle designers have to combine complex-game interactions with the enforced simplicity of the Palm. Knowledge of often-encountered design problems and of ways to avoid them will help designers to create technically well-behaving complex Palm games.

3. DESIGN PROBLEMS

This section identifies seven different design problems encountered in Palm games. Examples and possible solutions are provided too.

3.1 Too Detailed

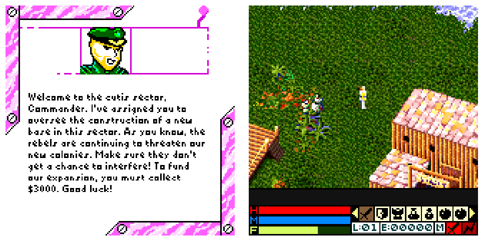

Welcome to the cutis sector, Commander. I've assigned you to oversee the construction of a new base in this sector. As you know, the rebels are continuing to threaten our new colonies. Make sure they don't get a chance to interfere! To fund our expansion, you must collect $3000. Good luck!

Figure 3: Two screenshots

Complex games usually contain many functionalities and have to provide much information. Designers of complex games often find it difficult to cram all that information into the small Palm screen. Consequently, they revert to miniaturizing things which are already small to begin with. For example, figure 3 shows screenshots from the games GALACTIC REALMS (figure 3, left) and ANCIENT RED (figure 3, right). GALACTIC REALMS uses a non-standard font that is only four pixels high and therefore barely readable. ANCIENT RED presents an isometric view on the game world, which forces objects to be smaller than a grid-based view would require. This makes it very difficult to “tap” an object with the stylus, especially a moving one, which unfortunately is something that constitutes the main form of interaction with this game. The solution here consists of simplifying the interface. Even if it means losing desired functionalities, a designer should acknowledge that the Palm screen can only hold a limited amount of information without becoming difficult to use.

3.2 Too Many Screens

From the previous subsection the conclusion may be derived that the number of functionalities in one screen should be restricted. An obvious solution would seem to be simply increasing the number of game screens to store all desired functionalities. This, however, should be avoided. A game that contains too many screens requires many screen-switches to accomplish basic game functions. It is standard design practice, in accordance with the Palm philosophy, that basic functions should be executed with one stylus “tap” (3Com 1999). If basic functions are distributed over several screens, too many taps are needed. An example of a game that errs in this respect is SPACE TRADER. The basic functionalities of this space trading/strategy game require no less than thirteen major and several minor screens. Similar functionalities should be integrated in one screen. It is the designer’s job to make sure that the integration is done without losing clarity.

3.3 Excessive Memory Overhead

Memory is a valuable resource for the Palm. Not only is it used to run applications; it is also used to store them and their associated databases when they are not in operation. This means the user is severely restricted in the number and size of installed applications. Games, especially the larger ones, are not a priority and will be quickly deleted when there is a lack of memory. Complex games tend to be larger and are therefore commonly first to be removed. It should be noted that colour images usually take large amounts of memory. Having them as an optional feature could alleviate the size problems. In some instances, dividing a game into modules could be a relief.

3.4 Hard to Control

When porting a game from a PC to the Palm, designers often try to map the PC interface as faithfully as possible to the Palm. Hardware devices like the joystick and the mouse are mapped to the Palm hardware buttons (shown at the bottom of the Palm in figure 1). Next to the fact that the process of reprogramming the buttons in itself is a questionable technique (Spronck 2001), it should be noted that an interface combining the use of the hardware buttons with the use of the stylus is hard to control. For example, in the 3D space combat game VOID (see figure 2), which is derived from the popular eighties game ELITE, the player navigates a spaceship. The hardware buttons are used to emulate ELITE's joystick functions, while keyboard commands have been mapped to control areas on the screen. So, for playing VOID the user needs to handle six different Palm hardware buttons and at the same time he must tap the screen with the stylus. This is virtually impossible. A better interface design would use the stylus for most of the control functions. Although it may have some impact on the game functionalities, playability, and therefore interface-design, should always be the first issue.

3.5 Need of Additional Materials

Complex games are often in need of additional materials besides the game itself. For example, the game DRAGONBANE (see figure 2) is situated in a large 3D maze. Since it is necessary for the player to make a map, access to pen and paper is required. This prohibits the player from delving into the game in places where pen and paper cannot easily be used. It also makes it difficult for the player to pause the game and resume it at a later time, since he probably would not keep the annotations together with the Palm. The obvious solution to this problem is that the game itself provides everything the player needs to play. DRAGONBANE, for instance, should contain an auto-map feature. All games should at least provide a status report which is accessible at any point of the game, to allow the player to review the current situation and continue from thereon.

3.6 Non-interruptible

The Palm is mainly a business tool. Games are of secondary importance. All Palm applications, and especially games, should allow the user to interrupt without punishment. That means a game should quickly and automatically save the game's state whenever the user switches to another application. Many games do not support this. For instance, VOID only saves the game's state when the player is docked at a space station, causing the player to lose minutes of game progress when he interrupts the game while flying towards another station. The solution is obvious. A game's state-saving mechanism should encompass all possible game situations.

3.7 Need of a Manual

Commonly a complex game needs a manual to explain all possibilities of the game. Of course one cannot expect a player to carry a bulky separate manual wherever the Palm is carried. Still, it is not an acceptable solution to provide a manual in the game in the form of help texts. The Palm screen is too small to read long texts easily. The solution is that a game should be designed in such a way that a long manual is simply not needed at all. This means that all the game's functionalities should be placed in a clear and orderly fashion in logical places in the game's interface.

4. DESIGN RULES FOR COMPLEX GAMES

In this section we present five design rules which are derived from the solutions offered to the design problems identified in the previous section. Using these rules as guidelines for designing complex games ensures the design of technically well-behaving games.

4.1 Limitations Drive the Design

A computer game's interface is typically designed after all functionalities have been determined. In contrast, a complex Palm game is severely restricted by the resolution of the Palm screen and the Palm memory. Therefore, during all stages of the design process, the limitations of the Palm should be taken into account. This encompasses the hardware limitations as well as the less obvious functional limitations. Being guided by the Palm's limitations may cause the designer to lose some desired functionalities. However, for a game it is better to be simple than unplayable. This rule alleviates the problem of a game being *too detailed*.

4.2 Memory Overhead Is Kept as Small as Possible

A complex game may be bigger than an average game. However, a game that is too big will antagonise a large part of the potential player-base. Optional modules and moderate or optional graphics are a transparent way of handling the problem. This rule alleviates the problem of a game having *excessive memory overhead.*

4.3 Game Control Is by Parsimonious Use of the Stylus

When porting a PC game to the Palm, it is a misapprehension to map the input devices of the PC to the Palm's input devices. The main Palm input device is the stylus, and the game's interface should be designed to work with it. On top of that, the number of stylus taps needed to play a game should be minimized by allowing the main functions to be executed with as few taps as possible. This rule alleviates the problems of a game having *too many screens* and being *hard to control.*

4.4 Interruptions and Continuations Are Unrestricted

A game that cannot be interrupted at any time is a game that disturbs the normal functioning of the Palm. A game should be able to save its state at any moment. Furthermore, since an interrupted game should allow the player to continue effortless at another time, it should provide all important state information. This rule alleviates the problems of a game being in *need of additional materials* and being *non-interruptible.*

4.5 Games Do Not Need a Manual

The Palm itself is not suitable for reading a manual and a separate manual is obviously not a realistic option. A clear, orderly and natural user-interface may reduce the need for a manual significantly. This rule alleviates the problem of a game being in *need of a manual.*

5. THE FUTURE OF PALM GAMING

Palms have dropped in price and have become popular as a tool for business users and as a "cool gadget" for everyone else. The Palm Corporation seems to find this sufficient reason to forego their philosophy of focussing on simplicity and starts to push the Palm as a games machine. From now on we will see the release of Palms enhanced specifically for gamers. Palms without these extras will, as long as the game-specific Palms are clumsier than regular Palms, still be available for business purposes. Just like what happened with PCs, the gamers will use the most advanced and most expensive machines.

The consequence is that game programmers must decide whether they will build a game that runs on all Palms, or whether they will produce a game for a “GamePalm”. The first option restricts the game's features, the second the potential player-base. Since the enhanced Palm offers the opportunity to produce visually enticing games and appeals to users that are likely to pay for games, professional programming teams will mainly produce games for the GamePalm, leaving the regular Palm as a playing ground for amateur game programmers. When the GamePalms have completely replaced the regular Palms, solitary game programmers are forced to compete with the professional game development teams. This is a battle they will most likely lose. Therefore, the statement at the start of this article that the Palm gives an ideal opportunity to start a small business in game development has an expiration date. The time to grab that opportunity is now. It will have passed in a few years.

6. CONCLUSION

In section 3 we identified seven design problems that complex Palm games face. These problems are caused by both hardware and functional limitations of the Palm computer. To alleviate these problems section 4 proposes a set of five design rules. Adhering to these rules will positively influence the design of a complex Palm game to be technically feasible. For the near future we expect that, whereas currently solitary programmers can still produce commercially successful Palm games, dedicated game development companies will take over this position.

7. REFERENCES

3Com Corporation, *Palm OS Programmer's Companion.* Part of the Palm OS Software Development Kit, 1999.

Levy, S. *Hackers: Heroes of the Computer Revolution*. Penguin, 1994.

Rhodes N. and McKeehan, J. *Palm Programming: The Developer's Guide*. O'Reilly & Associates, 1999.

Spronck, P. “Palm Game Design.” In *GAME-ON 2001 2nd International Conference on Intelligent Games and Simulation* (eds. Quasim Mehdi, Norman Gough and David Al-Dabass), pp. 95-99. SCS Europe, Ghent, Belgium, 2001.

Williams J. *Games to Go: The PalmPilot Series*. Available at: http://www.gamasutra.com. GamaSutra, 1999.