

Appendix A

Sample Game Design Documents

- Detailed design discussions
- Initial treatments and sample designs
- Technical specifications
- Code review forms
- Test scripts

Detailed Design Discussions

These documents describe the overall working architecture for a project. Our example for this section is the design overview of *Balls!*

1. *Balls!* Introduction

Balls! is a level-based, quasi-isometric, 3D puzzle game, designed to be intuitive and addictive to play. It is based on the interactions between balls of different capabilities on a block-based play area.

This design has been carefully thought out and refined by taking a reasoned comparison of other successful puzzle games. It is important to note that the game was not *designed* based on this premise, but that certain refinements were made and questions were asked based on the analysis. Where necessary, modifications were made. These design specifications are all subject to change pending further investigation.

2. Overview of Gameplay

The two main modes of gameplay are *strategic* and *action*.

Level success conditions are the same for both types of game and usually involve getting a certain number of balls to the exit.

The strategic game is more thought based and requires traditional reasoning skills within a fixed time limit. In the strategic game, each level is split into two phases. A timer runs through both phases, although for some levels it may be restarted at the beginning of the second phase.

The first half of the first phase allows you to look at the level and the balls and blocks you are provided with to complete the level. The second half of this phase allows you to place the balls and blocks and tweak the starting parameters.

The second phase begins by clicking the Go button and plays out the level as per your starting conditions. You will know in advance where and when a ball is going to fall, so you have time to arrange to receive it by arranging blocks. As the difficulty progresses, this warning time will be less, and it *may* contain less information (for example, you may not know exactly where and/or when the ball will appear).

If the level success conditions are reached within the time limit, the level is successful, and you are given the chance to replay or continue. If you fail, you are given the option of retrying the level.

Every now and then, both in the strategic game and the action game, you will get a special ball. Getting this ball to the exit will earn a huge bonus. This special ball will in most cases act like a normal ball, but it is possible that it will also have properties of one or more of the other types of ball (for example, a hot balloon).

It is possible that both types of action level will integrate well. Certainly, the latter design features are a subset of the former.

A third level-design concept for the action game is *continuous play*, which is closer to *Tetris* in concept.

This type of level starts in a similar fashion to the second type, but layers of blocks will be added or removed as you progress. Initially, things progress slowly, but, as you get further into the game, more diverse ball types and blocks will be added to spice things up.

Block layers are added after a time interval (decreasing as you get further) or by losing a ball. There are no levels in this action mode. The game just gets progressively harder and faster, rather like *Tetris*.

Because players like to have a sense of achievement, the game will have conceptual level numbers, based upon the number of balls saved and the score. When a level is increased, the game becomes harder and faster. After some time, the floor size can also be decreased to make things more difficult.

Care will have to be taken to ensure that this progression is smooth, as it was in the Game Boy *Tetris*.

With the continuous game, there is a definite possibility of adding two (or more?) player functionality via DirectPlay. The extent of network message traffic would be negligible, and the response would not need to be instant, so Internet latency would not appear to be a problem.

3. Platforms

Initial target platforms are Windows 95/98/NT, with the possibility of a no-frills (or few-frills) implementation on Windows CE. Two graphics engines are being developed. The initial prototype engine will be sprite based, and the advanced engine will be Direct3D based. The two engines will be developed in parallel to some extent, and will probably both be in the final game. The sprite-based system would be suitable for low-end machines or puzzle-game purists, and the Direct3D-based system would appeal to whizzy 3D card owners. It should be emphasized here that the focus of the game is not whizzy graphics. The graphics will be attractive and functional, but not over the top (like those, for example, in *Sentinel II*).

Because of its simplicity, this design has a lot of potential for other platforms such as the Sony PlayStation, Nintendo 64, and even the Color Game Boy, although we would rather wait for the next-generation handheld. The development for these alternative

platforms would be outsourced as required. The design is such that all the platform-specific code is separated from the game logic, so conversion work would be easier than usual, because a large proportion of the code could be directly reused with only minimal modification. Doing so would help keep costs down.

The PC distribution will be on CD-ROM, and the others on whatever format they natively support.

Target Audience

The target audience for this game is effectively everybody. It is intended to have an appeal similar to *Tetris*. While we feel that this is a very difficult task (and near impossible to achieve to any great degree by design), the basic design concepts are pure enough to allow a fair attempt.

This game is abstract enough to appeal to a wider audience than something such as *Puzzle Bobble*, but may be perceived as too “computery” to be the next *Tetris*. The advantage that *Tetris* had was that it was simple enough to fit the design on one page and that you didn’t feel as if it were a computer game you were playing. The interface to the hardware was forgotten and was irrelevant.

It is important to try to emulate this simplicity with *Balls!*. This will be made more difficult by the fact that it is a 3D game—which automatically labels it as a computer-oriented game, as opposed to *Tetris*, which was just a game that happened to be on a computer. It is interesting to note that the 3D versions of *Tetris* did not achieve anything like the same level of success as the original.

This may have been because a simple idea was overcomplicated before its time. However, this does not mean that *Balls!* is an inherently bad idea or that it is too complex. It just means that the emphasis of design needs to be on hiding the complexity behind a simple interface and presenting it in a completely natural feeling way. This, we think, has been achieved in the design.

4. Time Scales

Given sufficient free time, a basic, no-frills implementation could be put together in six months, although this implementation would possibly contain only one of the game-play modes and no multiplayer mode. This would be a functional prototype.

If the structure of the application was carefully thought out, then additional effects could be plugged in. The majority of time will be spent on level design.

The full product could be completed in under one year, given one artist and two or three programmers, one of these with some art skills, though this may not include the full set of enhanced features.

Given 18 months, the product would be entirely complete on the PC platform.

If time were running short, the enhancements discussed later in this document could be pared down or omitted entirely. The enhancements add very little to the game itself, but they do give a slightly more polished appearance and enhance the sense of achievement when a player reaches a certain level.

A fair amount of groundwork has already been put in place in developing resource access and graphics and sound libraries that have been used successfully in other projects.

Basic Concepts

The fundamental idea behind *Balls!* is that it will be a simple and addictive game of the *Tetris* ilk.

Every element of gameplay has been examined and analyzed. There are no cases of “Wouldn’t it be cool if...” or “This would be a neat feature—it would look amazing,” that seem to be prevalent in most games today, which increasingly seem to be a triumph of style over content.

The graphics in *Balls!* are pleasant but functional, and any embellishments are provided as rewards for progress. Most of the effort has been spent on defining the gameplay.

5. Why Puzzle Games Aren’t as Good as They Used to Be

The games industry has made several attempts to follow in the footsteps of *Tetris*; not many of them, however, have made the grade. Usually, they repeatedly fall into the same traps inadvertently set up by marketing men who have dollar signs in their eyes, and completely overlook playability.

A notable example of this was the release of a puzzle game on 14 or so different platforms with a huge and expensive marketing campaign and lots of magazine coverage.

The marketing man concerned was convinced that he had a sure-fire winner, which would match *Tetris* in sales. Unfortunately for him, the game was too complicated. A game of that nature must be intuitive. Some aspects of the game were not suitable for the style of game that they wanted to achieve.

Without delving too deeply into the whys and wherefores of the failure of that particular game, the main point here is that the game failed because it started with a marketing concept and ended with a product. This is the wrong order. Start with a well-designed product, and then plan the marketing.

6. Puzzle Game Appeal

For puzzle games to have a wide appeal across the sexes, a few simple guidelines should be followed. These do not guarantee success, but it is possible that they could help. Of course, there are exceptions to every rule.

Don't Be Too Cute

Sickening cuteness doesn't work. Contrary to industry opinion, girls are not interested in sickly cuddly things, and neither are boys. However, an example of where this has worked well is *Lemmings* (the first release only—people got a bit fed up with them after a few releases), but this was offset by the humor within the game, which had an altogether darker edge to it.

The cuteness was not overdone, unlike—for an example of where this has failed—*Fury of the Furries*.

Avoid "Serious" Violence

This is quite an abstract rule. For example, *Zoop* fell foul of this rule by using shooting. The player shot colored shapes to remove their color. However, *Lemmings*, although violent in places, worked well because it was cartoon violence.

This worked to counterbalance *Lemmings*' cuteness, although players do not want to see their favorite cute pet die horribly. In *Lemmings*, players do not get the opportunity to become attached to individual characters, because the lemmings themselves are

anonymous, which also reduces their cuteness. In the later *Lemmings* games, the graphics for the lemmings were improved, which some players felt actually reduced the appeal of the game.

In short, if you have to be violent, make it funny.

Keep it Abstract, but not too Abstract

Tetris is a good example. It is abstract, but it can be mentally related to sliding-block puzzles.

Some very successful games on the ZX Spectrum were enhanced sliding-block puzzles, an example being *Confuzion*, which involved sliding segments of fuse wire around to dispose of the sparks traveling along it. *Lemmings* is abstract to an extent: The individual lemmings are not very detailed, and character development is not attempted. *Zoop* was too abstract, as was *Endorfun*. They bore no relationship to any other experience. Players have to have something to mentally relate to. For the wide appeal, you have to suspend the players' belief that it is a computer game and allow them to imagine that the game could be actually happening for real.

Give Rewards for Progress

In *Pac-Man*, the main reward apart from gaining a high score is a set of animated sequences that occur every fifth level.

Similarly, in *Lemmings*, the reward is gained by reaching new and visually different levels. Past a certain level of Game Boy *Tetris*, a reward animation is played. This was a surprise, and the element of surprise is important. People appreciate rewards more if they do not expect them.

7. Why *Balls!* Would Be Good

The game involves positioning a set of balls and/or blocks with varying characteristics onto an array of blocks that affect them in different ways.

In the strategic game, the player positions the balls so that their eventual paths and interactions achieve some arbitrary level goal. The goal could be to get a certain number of balls into an exit location or into a certain pattern, with or without a time limit.

There are some constraints on placing the balls, such as positioning, direction of initial travel, and a time limit.

Once the balls are placed, they are set in motion, and the player watches the results. This is very much a thinking puzzle game.

In both types of action game, the players are able to influence the behavior of the balls after release, by dropping blocks to affect their path.

The three game types can be used together to allow people to play the type of game they prefer. The three types of games are aimed at three different types of player or player mood.

The first type, the *strategic* game, is designed to have long-term appeal to the deep-thinking puzzler. Think of it as the “crossword” mode. If they cannot figure out a level, they can come back to it later with fresh inspiration, or they can retry the level in *action* mode.

The second type of game, the *level-based action* game, is again designed for long-term appeal, but this time the emphasis is on quick reflexes and fast thinking. Think of this as the “speed chess” mode. Again, if the player cannot get past a level in this mode, they can retry it in the strategic mode. Of course, they get more points if they complete it in both modes. Note that the player gets points for completing a level only once per mode: You cannot repeatedly play the same level in the same mode to bump up your score.

The third type, the *continuous action* game, is designed for when the player fancies a quick blast of *Tetris*-type adrenaline. It is designed to allow the player to dip in for a quick game, and then hook them into that “just one more go until I get to level 13” feeling, before the player realizes that they have been playing for half the night. Think of this as the “*Ballstris*” mode. There are no separately designed levels here; the game just gets algorithmically more difficult as gameplay progresses.

This mode has no long-term goals. It is just a quick fix, hopefully with a killer addictive hook. This mode is the only one that provides a multiplayer game.

A fundamental point of the game is that the balls cannot gain in potential energy; that is, they cannot gain height to get a block higher than the one they are on. Individual balls will never change speed, only direction. The system is based on much-simplified “real” physics, with some liberties taken for the purposes of gameplay.

Balls! would appeal because it can be readily related to common phenomena such as rolling balls or marbles. Even though the laws of physics in the game are slightly different from in the real world, they are close enough to be readily accepted. This is a strong advantage.

The player is given a visual reward for completing some (or all) of the levels. The movement of the balls could be designed to trigger a sequence of actions that cause some event. Some of these could have two outcomes—nice and nasty—depending on how you completed the level. These outcomes could be integrated into the level. This is discussed in more detail later. (Note that the in-house level editor is being produced to release quality, so we can include it as part of the package and increase replay value.)

Look and Feel

The look and feel of the game is very important. It should be simple and not an obstacle to the game (that is, giving virtually instant access to the game itself).

A consistent theme should run throughout, and the interface should be simple and easy to understand. Although the standard windows interface is dull, a similar but more harmonious approach could be used to provide familiarity. The other approach is to base the interface on a familiar piece of everyday equipment. This approach has been used reasonably successfully in a number of games (usually driving games), but it has the disadvantage of having to find a bit of everyday equipment that everybody is familiar with and that fits the theme of the game.

The former approach is the most likely here, and it is possible that a design based on a game level and using the game engine may be a good idea. As well as acting as a good testbed for the dynamics, it will also add consistency. This will be fine, as long as it does not appear too contrived and get in the way of what the interface was trying to achieve in the first place. Figure A.1 shows an overview of the program structure.

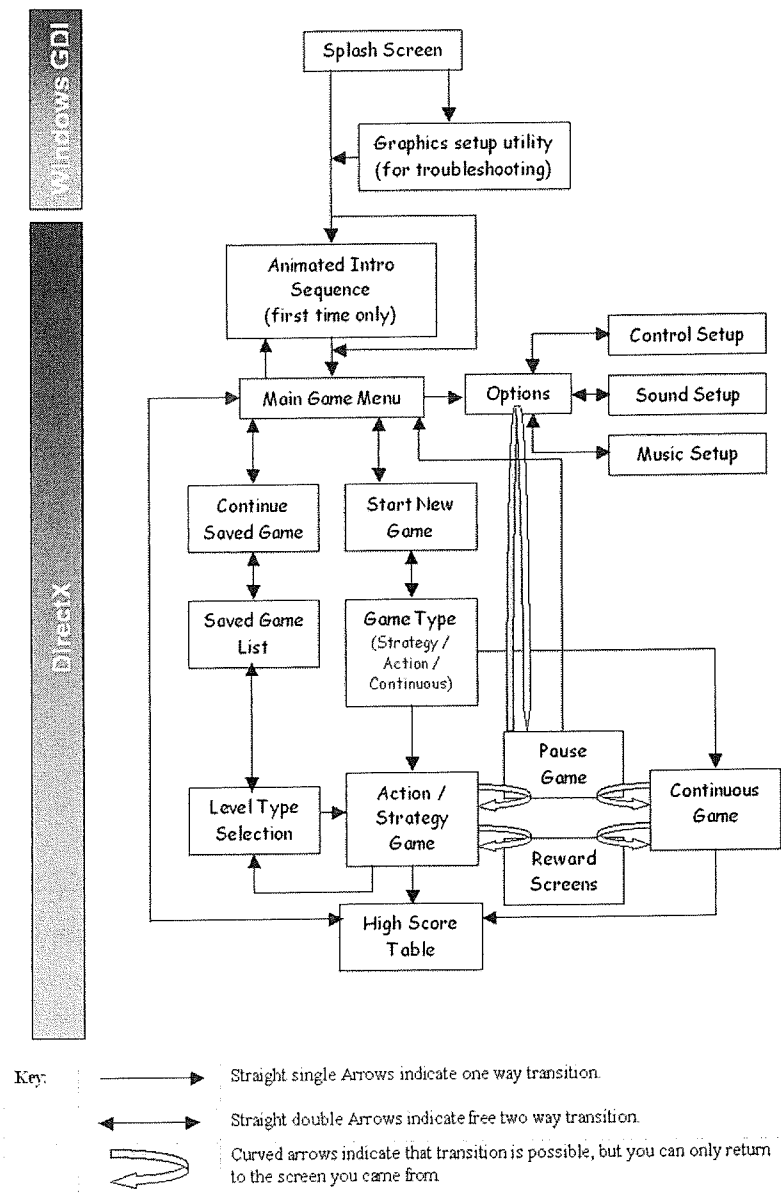


Figure A.1 *Balls!* Program structure.

8. Game Design: User Interface Elements

The user interface is split into two parts: the “game configuration” interface and the “in-game” interface. The idea is to make these as seamless as possible.

This does not include the Windows-based graphics troubleshooting application, which will use a standard Windows interface in case the in-game graphics are configured badly, so that the screen cannot be viewed.

Game Configuration Interface

The game configuration user interface will be based on elements of the game engine. For example, the configuration of blocks shown in Figure A.2 represents a slider bar.

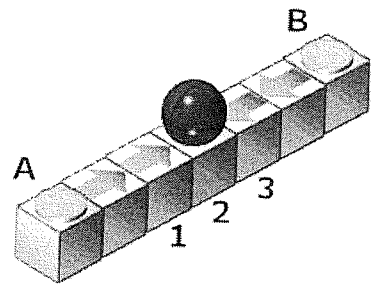


Figure A.2 Slider bar.

The player operates the slider bar by clicking on either of the buttons.

Block A is a push-button switch. Blocks 1 and 3 are directional blocks, and block 2 is a sticky block that holds the ball.

Clicking on button A causes block 1 to become a sticky block, and block 2 to become a directional block of the same type as block 3. This will cause the ball to move towards block 1 because block 2 is now a directional block propelling it towards button A. The ball will then stop on block 1, which is the new sticky block.

Clicking on button B will have the reverse effect.

This particular slider bar has six settings, increasing from A to B.

Note that a slider bar could also be implemented using a magnetic block with a metal ball instead of the sticky block.

The two types of switch block are push-button and toggle switch. The push button is in the ON position only for as long as there is pressure on it. The toggle switch is toggled from the ON position to the OFF position and vice versa every time it is pressed (see Figure A.3).

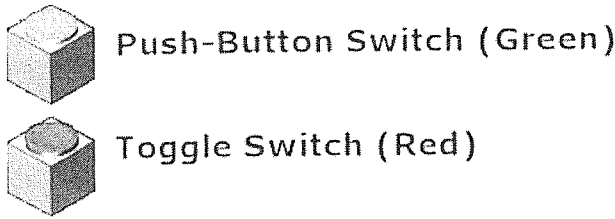


Figure A.3 Toggle and push switches.

Level Design (For Level-Based Game Modes)

All level types derive from a base level template shown in Figure A.4.

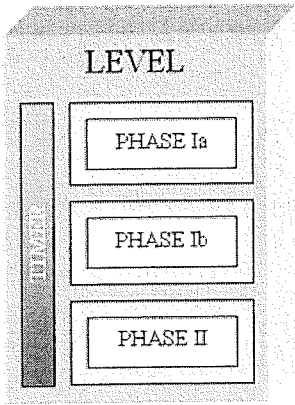


Figure A.4 Level template.

The level template has two phases and a timer running through both. The timer object is quite flexible: it can run, as shown, through all phases, or it can be deactivated for any or all phases. The timer can also be reset at the beginning of phase II, if desired.

All levels must have at least one phase. This represents the general format of all levels, including the strategic type.

The designer is free to implement any level he or she likes, with the emphasis on action, strategy, or any combination of both. The *Balls!* system is flexible enough to accommodate both types.

Strategic Game The strategic game is similar in mental approach to a chess problem. You are presented with a configuration of blocks and a set of balls to place on them. Some balls may already be in place. You may also be provided with a limited set of

blocks, which can be used to modify the configuration already in place. Various other parameters such as release times of blocks and balls and initial direction of movement may also be configurable at the behest of the level designer.

This configuration process may have a time limit, which is equivalent to phase “Ia” and “Ib” in the level template. This timer can either be allowed to run through both phases or restarted at the beginning of phase II. Phase II is initiated by clicking on a Go button.

The player then watches the results of the phase I setup. If the success conditions are met before the phase II time limit runs out, then the level is completed; otherwise, the player is allowed to retry the level.

Action Game (Level Based) The action game will progress into a fast and frenetic mayhem zone. The action game could have two types of levels, but, until playtesting, we are unsure which sort of level type will be more suitable. It is possible they will both be fine. Fortunately, the issue is only one of level design and not of programming. If one level type isn’t suitable, it can be redesigned.

The basic idea is that blocks fall from above and must be steered and rotated into positions to enable the balls to reach the target squares. The balls can either already be on the level or they can fall from above.

If balls fall from above, the player gets some form of warning that they are coming and where they will fall. The level designer decides the format of the information. The player cannot steer falling balls.

Level Type I The first level type is similar to the thinking game, in which there is a complex configuration of blocks with some balls already stationed on them. This level can be split into two phases, the lengths of which are set by the designer using the timer.

In the first phase, the player gets the opportunity to examine the block-and-ball configuration. He does not know which blocks will be falling, but he may be informed if any other balls apart from those already on the level will be falling.

These levels usually have two phases, but the first phase is optional for the level designer.

Level Type II The second type is closer to *Tetris*, in that you start with a simple floor of blocks and blocks and balls fall from above. As in the first level type, the balls cannot be steered and you get some sort of warning as to when and where they will arrive, but the emphasis here is on building your own structures to get the ball to the exit block, which may not appear immediately in the level and fall in only later.

Action Game (Continuous) The continuous action game is closer to *Tetris* in concept, in that there is continuous action with no respite. It is derived from *action level type II*, in that you start with a flat, one-layer-thick floor of blocks. Blocks and balls fall from above, and, as before, only the blocks can be steered. You will still get a warning as to when and where balls will appear, although this will be algorithmically decreased as you progress.

The difficulty increases algorithmically as time progresses. For example, new layers will be added every so often. The time intervals between layer additions will decrease as the game progresses, probably bottoming out at around 10 seconds (subject to playtesting). Also, as the game progresses, the layers may get smaller—starting out at, say, 12x12 and ending at 6x6 (again subject to playtesting). With this, there may need to be some way to allow re-extension of the floor space, in order to balance out the gameplay.

Losing a ball causes a layer to be added, and saving a ball causes a layer to be removed.

Also, the interval between the arrival of new balls steadily decreases, and the frequency of different ball types appearing increases.

In the continuous game, the target block appears at the beginning. Layers are added from below, so the target block always appears on the top layer. It is impossible for the lowest layer to be destroyed. Blocks may be discarded by dropping them over the edge of the level or through the target block, but doing this too many times will incur an added layer. It should be possible to get some sort of bonus multiplier to remove more than one layer at once.

For the two-player game, an additional block type is required, the wormhole block. This allows blocks and balls to be sent to the other player. The wormhole block is used in a similar fashion to the target block, but it sends the block or ball to the other player. Sending a ball through the wormhole block will not remove a layer, and there is no penalty for dropping blocks through it. It can be used only once before reverting to a normal block. If a layer that is destroyed contains any wormhole blocks, then one extra layer per wormhole block is added to the other player's game.

The two play modes are *cooperative* and *competitive*. The fundamental difference between the two is in the attitude of the players. The wormhole block behavior is the same, but the emphasis in the cooperative mode is in using them to help the other player, rather than hinder them as in the competitive mode. For the purposes of the high-score table, the scores of each player in cooperative mode are summed, and both names (and a team name) are entered into the high-score tables on both machines.

Level Progression

The level design will be implemented to gradually introduce the player to all the features of the game.

The first ten or so levels will be tutorial levels to familiarize the player with game features and the basic ways they can be manipulated. In this set of levels, simple ball-and-block types will be introduced.

The difficulty of the continuous action game will be algorithmically increased, probably based on a logarithmic scale.

9. Physics of Balls!

Figure A.6 shows the timing of ball movements. Note that all balls reach the center of the blocks simultaneously.

Ball types are shown in Table A.1.

Table A.1 Ball types.

Ball Types	Image	Properties
Normal Ball	 Normal	The default ball. The time that this ball takes to get from the center of one square to the next is one game “turn.”
Metal Ball	 Metal	This ball will stick to a magnetic block if it is adjacent to the magnet. If it collides with a hot block, it will become a hot ball.
Mimic Ball	 Mimic	A mimic ball takes on the characteristics of the ball that it last struck. Its uninitialized characteristics are that of a normal ball. It can also have a limit on the number of transitions it can make before it sticks in one form. It will always look slightly different from the ball type it is mimicking.
Balloon Ball	 Balloon	It will float across single gaps in blocks, and burst on contact with spikes. It can be blown off course by fan blocks, and does not affect switch blocks. It does not stop on sticky blocks.
Glass Ball	 Glass	If it drops any distance it will break, unless it lands on a balloon ball or a soft block, and is resistant to pools of acid on top of acid blocks.
Snow Ball	 Snow	It will melt after a short period of time or on coming into contact with a hot block. If it collides with an ice block, it will take longer to melt. (The melt timer will be reset to new.)
Explosive Ball	 Explosive	It will explode on dropping any distance, destroying the block it lands on, except the soft block.
Hot Ball	 Hot	This ball is a hot metal ball. If it collides with a snow ball, the snow ball will melt, and the hot ball will cool to become a normal metal ball. The same applies for collision with an ice block. If it collides with a magnetic block, the magnetic block becomes a normal block.

Balls cannot change speed, except to stop. Empirically, their speed is either full speed or zero speed. Of course, some animations (for example, rolling halfway up a slope) may give the impression that the ball is slower, but—as far as the game engine is concerned—all balls will reach the center of a square simultaneously as shown in Figure A.5.

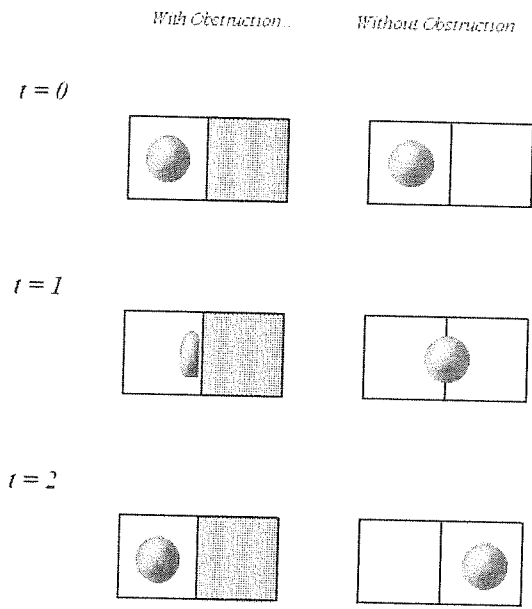


Figure A.5 Simple ball-block collisions.

A comparable system also applies for ball-ball collisions and 90-degree ball deflections, as shown in Figure A.6.

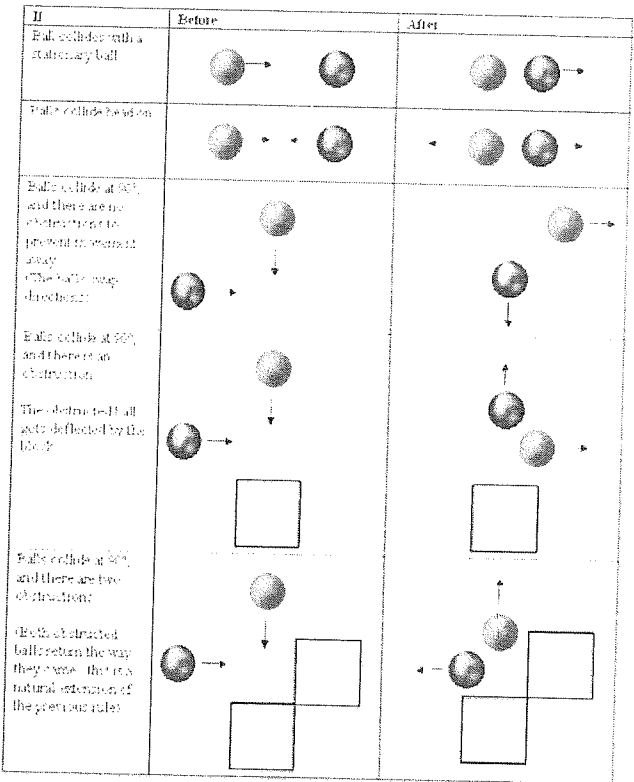


Figure A.6 More complex collisions.

The game is timed on the presumption that all balls reach the center of a block at the same time, no matter what action they are taking at the time. This presumption ensures that the simplified physics system deals only with the situations that are covered in this document.

When a ball collides with a block, it should take the same amount of time to travel from the center of the square, rebound, and return to the center of the square as it would to travel to the center of the next square if there had been no obstruction.

As far as the physics system is concerned, no matter what size the balls appear visually, a ball is large enough to be able to collide with a ball on an adjacent block when both balls are in the center of their blocks. This implies that the balls' diameter is the same as the length of the side of the block. This will be considered true for collisions. An exception to this rule is when the collision occurs from above (a ball dropping on another ball); otherwise, the described behavior wouldn't occur.

Another exception is required when balls are traveling through archways in blocks. For these purposes, the balls will be considered to have a diameter of approximately three-fourths of the length of a block's side. These minor inconsistencies will not be noticed.

Note that no one complained that in *Tetris* you could rotate pieces in gaps that would strictly be too small to allow such rotation. The game uses simplified baby physics. It is deterministic, meaning that there can be no randomness.

NOTE

The complex ball-block collisions above follow naturally from, and can be derived from, the simple ball-ball collisions in Figure A.6, and the ball-block collision in Figure A.5.

Figure A.7 shows a ball falling from a block.

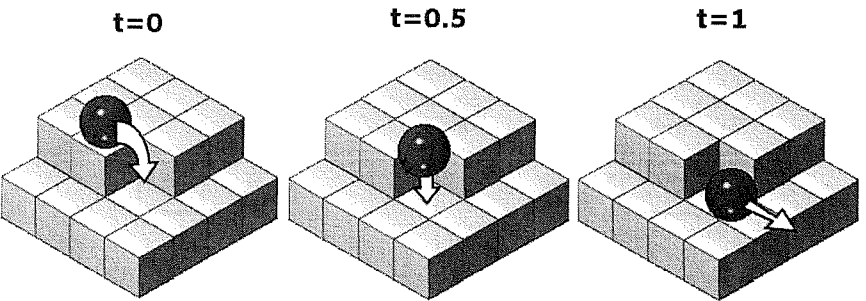


Figure A.7 Ball falling from a block.

In Figure A.8, the ball falls into the redirection block, which deforms to give it room. It is then sprung out in the new direction by the redirection block.

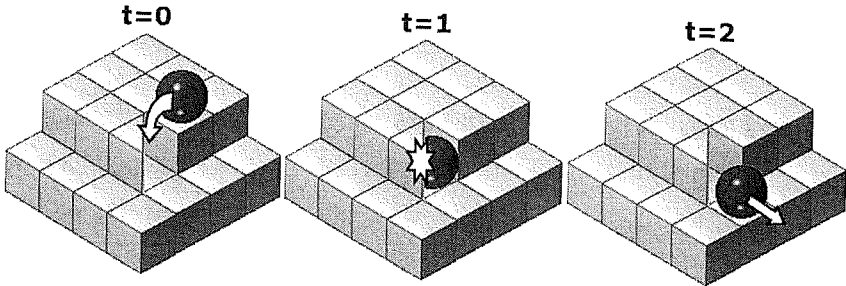


Figure A.8 Ball falling from a block into a redirection block.

In Figure A.9, the balloon ball floats over the single gap to land on the other block. Note that, with this block configuration, it will then fall off the edge and land on the X, without touching any other blocks.

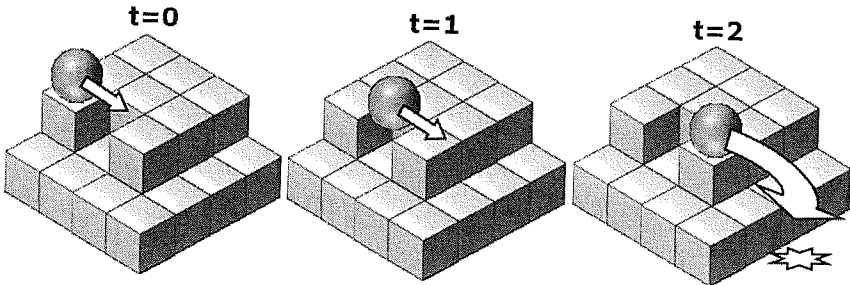


Figure A.9 Balloon ball floating over a gap.

In Figure A.10, a ball falls from a block and lands on another ball, then the balls carry on.

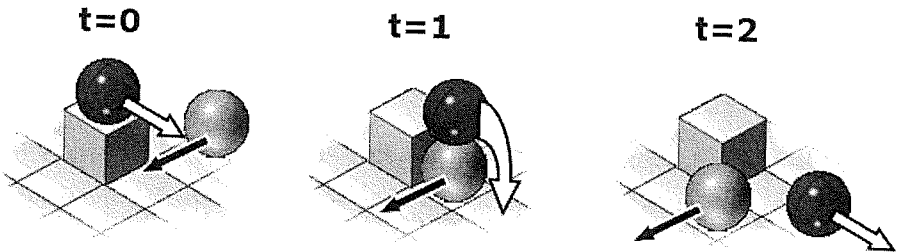


Figure A.10 Ball-ball collisions from above.

Figure A.11 shows a similar situation as in A.10, except that the lower ball is a balloon ball. The ball is able to jump the gap by bouncing on the balloon ball. If the gap is two or more units wide, the ball is unable to jump it, unless the gap is completely bridged by balls.

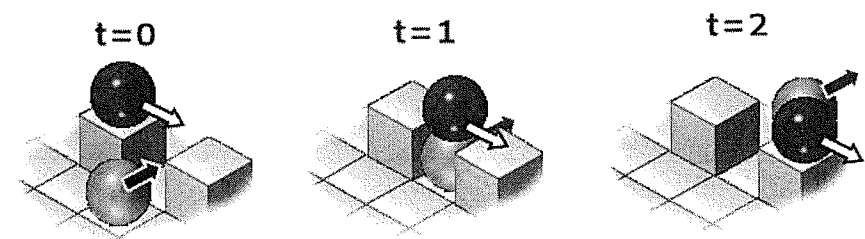


Figure A.11 A similar collision with a ball underneath.

In Figure A.12, the ball rolls halfway up the slope, stops, and rolls back down.

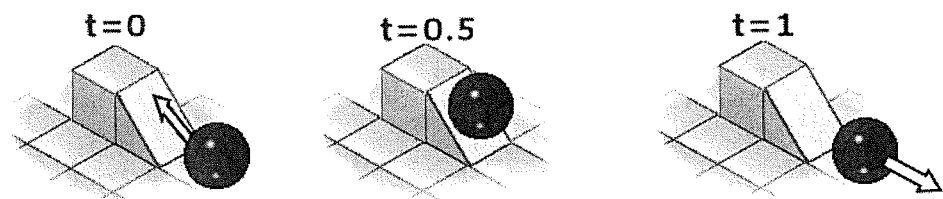


Figure A.12 Ball rolling up a slope.

In Figure A.13, one ball approaches down a slope, one ball up a slope. The collision occurs on the slope; then the balls bounce back the way they came.

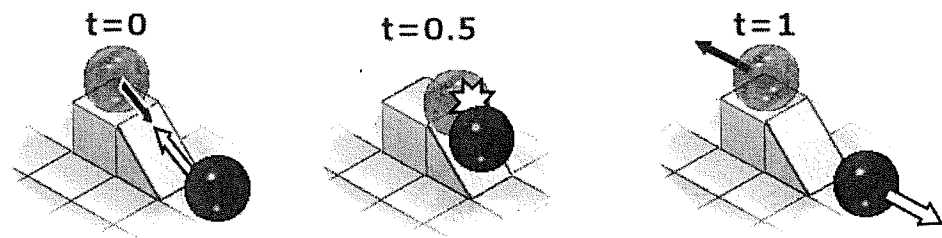


Figure A.13 A direct collision on a slope.

The situations shown in Figures A.12 and A.13 are the only exception to the rule that balls do not gain potential energy. It is resolved by defining only integral potential energy (PE). Hence, the ball does not lose any PE until it reaches the bottom of the slope (strictly past halfway), so it can legitimately roll back the way it came.

Many more complicated situations could be defined, but they emerge naturally from these basic rules.

10. Blocks

Tables A.2, A.3, and A.4 describe the types of blocks present in the game.

Table A.2 Types of blocks—Part 1

Block Types	Properties
Normal Blocks	If a ball hits a flat side of a block, and no other rules prevail, then it rebounds back the way it came.
Redirection Blocks	Deflects a ball by 90 degrees As the ball hits the block, it deforms to allow the ball to get to the center of the square, and give the impression of springing the ball away. Any block landing on top treats this as a solid block. How a ball falls depends on the direction of approach.
Archway Blocks	Allows a ball to go through a block. If balls go through at right angles, collision rules are followed as in Figure A.6.
Spiky Blocks	Will burst the balloon ball.
Ramp Blocks	Balls go down but not up ramps. If they try they get halfway up and then roll back down. If a ball hits any other side of a ramp, it rebounds as if it had hit a solid block. If a cubic block is at the base of the ramp, preventing a ball from rolling, the ball will become stuck (unless any other rules apply). If any block falls onto a ramp, it will destroy the ramp, and replace it, unless the falling block is destroyed as well.
Magnetic Blocks	If the metal ball passes adjacent to the magnetic side, it will stick until another ball collides with it. The magnet will become a normal block on contact with the hot ball or hot block.
Starting Blocks	Starting blocks may have directional arrows indicating the allowed start directions. The starting position may appear on any flat-topped cubic block.

Table A.3 Types of blocks—Part 2

Block Types	Properties
Switch Blocks	Switch blocks have a number of effects including removing blocks, opening archways, removing spikes, controlling directional blocks, etc. Illuminated is ON. Switches can either be toggled or push buttons. Any block falling here apart from the crumbling block will keep the switch permanently on, or toggle it in the case of a toggle switch. The crumbling block releases the switch when it crumbles, if it is a push-button switch. Green switches are push-button switches, and red switches are toggle switches.
Soft Blocks	This pinker-than-pink block cushions the landing of the glass ball and the explosive ball, so that they are not destroyed.
Hot Blocks	This block is the same as a normal block except that it is hot. It will destroy any ball that comes in contact with it except the metal ball, which becomes the hot ball. If the ball is the explosive ball, it will destroy the block. If the ball is the snow ball, it will neutralize the block, causing it to become a normal block. If the ice block collides with this block, it will melt, and the hot block will become a normal block. If the magnetic block collides with this, the magnetic block will lose its magnetism.
Ice Blocks	This block is solid ice. If a hot ball comes into contact with it, the block will melt, any block on top of it will fall, and the hot ball will cool into a metal ball. If the snow ball comes into contact with it, it will be able to stay frozen longer. If a hot block collides with this, the hot block becomes a normal block, and the ice block will melt.

Table A.4 Types of blocks—Part 3

Block Types	Properties
Fan Blocks	This will blow the balloon ball away from the fan if it passes. If its new path is blocked by an obstruction, the balloon ball will lodge in place until knocked out by another ball. The radius affected is one square.
Acid Blocks	This block will dissolve any ball that crosses it, except the glass ball. Any block landing here will dissolve in the same amount of time a crumbling block takes to crumble.
Target Blocks	The object of the game is to get the balls here. This block cannot be destroyed. If a block lands here, it will disappear after a few seconds, taking the same amount of time it takes for a crumbling block to crumble.
Directional Blocks	The directional block allows travel in only one direction, and will propel the ball in that direction, indicated by an arrow. A timer can allow cycling through the arrows. If a block falls here, it will remain here and not be propelled in the direction of the arrow.

Block Types	Properties
Crumbling Blocks	Will disintegrate on contact with any ball. It allows one ball to travel over it before disintegrating after a short delay. If this block falls, it will disintegrate, even if it is falling in as part of the game. In this way it can be used to trigger switches without covering them.
Sticky Blocks	This block has a ball-shaped indentation. Any ball apart from the balloon ball will stop dead when it hits this block. It can only be dislodged by a collision with another ball.
Wormhole Blocks	This is a multiplayer-only block, which acts in a similar fashion to the target block, except that items sent through it will appear in the other player's level. Anything disposed of in this fashion is not counted as going through an exit block, or as being lost. It simply goes to the other player.

Figure A.14 describes the ball-block interactions.

	GLASS	EXPLOSIVE	SNOW	METAL	STEEL	WORMHOLE	WORMHOLE	WORMHOLE
GLASS					Ball breaks on Fall		Destroy ball and block; it falls on	
EXPLOSIVE					Ball breaks on Fall		Destroy ball and block; it falls on	
SNOW					Ball breaks on Fall		Destroy ball and block; it falls on	
METAL				Push Ball	Ball breaks on Fall		Destroy ball and spikes; it falls on	
STEEL					Ball breaks on Fall		Destroy ball and block; it falls on	
WORMHOLE		Ball sticks to block			Ball breaks on Fall		Destroy ball and block; it falls on	Wormhole Block
WORMHOLE					Ball breaks on Fall		Destroy ball and block; it falls on	
WORMHOLE	Press Switch	Press Switch	Press Switch		Ball breaks on Fall; Destroy Switch	Press Switch	Destroy ball and block; it falls on	Press Switch
WORMHOLE	Toggle Switch	Toggle Switch	Toggle Switch		Ball breaks on Fall; Toggle Switch	Toggle Switch	Destroy ball and block; it falls on	Toggle Switch
WORMHOLE	Ball moves	Ball moves	Ball moves	Ball moves	Ball breaks on Fall; Ball moves	Ball moves	Destroy ball and block; it falls on	Ball moves
WORMHOLE	Destroy Block	Destroy Block	Destroy Block		Ball breaks on Fall; Destroy Block	Destroy Block	Destroy ball and block; it falls on	Destroy Block
WORMHOLE	Ball goes through exit	Ball goes through exit	Ball goes through exit	Ball goes through exit	Ball goes through exit	Ball goes through exit	Ball goes through exit	Ball goes through exit
WORMHOLE	Destroy Ball	Destroy Ball	Destroy Ball	Destroy Ball	Destroy Ball	Destroy Ball	Destroy Ball	Destroy Ball
WORMHOLE					Breaks on Fall; Destroy Ball	Destroy Ball	Destroy ball and block; it falls on	Destroy Ball
WORMHOLE				Ball moves	Breaks on Fall		Destroy ball and block; it falls on	
WORMHOLE	Destroy Ball	Destroy Ball	Destroy Ball	Destroy Ball	Breaks on Fall	Destroy Ball	Destroy ball and block; it falls on	Destroy Ball
WORMHOLE	Slope Ball	Slope Ball	Slope Ball		Slope Ball	Slope Ball	Slope Ball	Slope Ball
WORMHOLE								
WORMHOLE	Ball goes to other player	Ball goes to other player	Ball goes to other player	Ball goes to other player	Ball goes to other player	Ball goes to other player	Ball goes to other player	Ball goes to other player

Figure A.14 Ball/block interaction table.

11. Special Case Block-Block Collisions

Note that all block destruction takes one game turn.

Figure A.15 shows that the hot block cools to a normal block. The ice block is destroyed.

Before	After
	
	X

Figure A.15 Hot block and ice block.

Figure A.16 shows that the hot block remains hot. The magnetic block cools to a normal block.

Before	After
	
	

Figure A.16 Hot block and magnetic block.

Figure A.17 shows that the target block remains the same. Any other block will vanish through the exit.

Before	After
	
	X

Figure A.17 Any block falling on target block.

Figure A.18 shows that the wormhole block becomes a normal block. Any other block will vanish through to the other player.

Before	After
	
	To other player

Figure A.18 Any block falling on wormhole block.

Figure A.19 shows that the acid block remains the same. The other block dissolves in the same amount of time that it takes the crumbling block to crumble.

Before	After
	
	X

Figure A.19 Any block falling on acid block.

Figure A.20 shows that any block that isn't destroyed by the fall activates the push-button switch and leaves it activated.

Before	After
 OFF	 ON
	

Figure A.20 Any block (except crumbling block) falling on push-button block.

Figure A.21 shows that the crumbling block keeps the push-button switch pressed until it disintegrates.

Before	During	After
 OFF	 ON	 OFF
 	 	X

Figure A.21 Crumbling blocks falling on push-button block.

Figure A.22 shows that the ramp is destroyed by the falling block.

Before	After
 	
 	X

Figure A.22 Any block falling on ramp block.

If any block falls on a ball, then the ball is destroyed in that game tick. In the following game tick, the block/block interaction is resolved.

12. Playing the Game

This section discusses the gameplay of both the strategic and the action gameplay modes.

Strategic Game

In the 5x5x5 level shown in Figure A.23, the player has three balls to play: a normal ball, a balloon ball, and a glass ball.

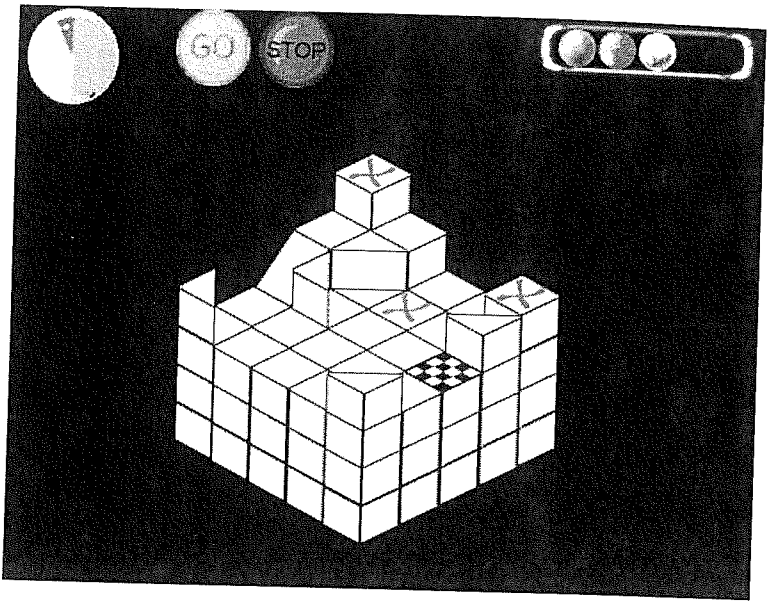


Figure A.23 A conceptual screen shot.

The objective of the level is to get all three balls in the exit. This screenshot shows only the format of the levels; it is not representative of in-game graphics. This is similar to one of the views in the level editor.

While you think about the ball placements a timer ticks down, limiting your thinking time. Clicking on a ball will take it, and you can then place it on a starting cross. Arrows will appear for the starting directions available (if any). Choose one with the mouse. You can change your mind about the placement of any ball by clicking on it again. You can also delay the release of a ball using a timer attached to the starting square.

Once you have placed all the balls, click on the Go button to start the balls rolling.

The level finishes when the level objectives are complete, or, in the case of fuzzy objectives, when the last ball has gone, the Stop button is pressed, or a timer runs out.

When the level has finished, you get the option to retry or go onto the next one if you completed the level objectives.

Examples of objectives are:

- Get all the balls to the exit.
- Get at least six balls to the exit. This could be timed.
- Get the balloon ball to the exit.
- Get the balls to the exit in a certain order.

Level-Based Action Game

The incoming balls in the level shown in Figure A.24 are a hot ball and a snow ball.

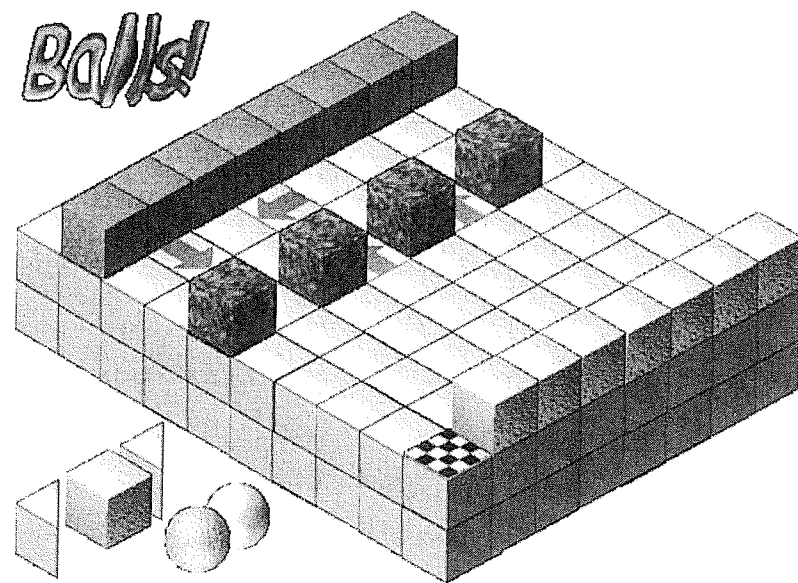


Figure A.24 A conceptual screen shot.

The incoming blocks are a redirection block, an ice block, and another redirection block.

The objective for this level is to get both balls to the exit. Any other result is a failure condition.

The level starts when a hot ball falls in at arrow A.

The hot ball is propelled around by the arrows and strikes the ice block, B, which melts.

The ball bounces back from the melted block and is reflected back the same way by the arrow block C. If this is allowed to continue, it will fall off the edge.

While this is happening, a redirection block falls in, which must be placed where block B used to be and oriented so as to deflect the incoming hot ball into the target zone.

Next, a snow ball drops at arrow D, and an ice block falls in. The ice block must be used to quickly neutralize hot block E before the snow ball reaches it.

Finally, a redirection block falls, which must be used to redirect the snow ball towards arrow A.

This is a fairly simple level that would appear quite early on, after the introduction and tutorial levels. Each ball uses only one of the incoming blocks, and each is used only once.

13. Further Embellishments

Although *Balls!* is designed to have a straightforward implementation on as many platforms as possible, various ideas could be applied to the game to spruce up the appearance.

The decision whether to do this depends on the sort of impression the game is meant to give. The design, as described so far, is aimed at producing a *Tetris*-style game—that is, abstract but recognizable. This perfectly valid approach has advantages as well as drawbacks. The key advantage is that there is nothing getting in the way of the gameplay, but it means that the gameplay has to be perfect. The key disadvantage is that it may be considered too spartan for magazine reviewers. (Most game players probably wouldn't care, but, unfortunately, the state of the industry is that reviewers mainly ignore gameplay if the product isn't graphically and sonically showy, and good reviews mean good sales no matter how well it plays.) Increasingly, however, graphics also have to be damn good, especially with the advent of 3D cards. There really is no excuse any more.

Lemmings is an example of a game in which embellishments worked well. We think that a similar approach should be taken here.

Every fifth level could be a special level that will provide light relief.

We have two ideas for this. The first is that additional tasks have to be completed in order to get a graphical reward. This is the more complicated option. A simpler option would be to replace the balls with special objects, such as small furry animals, cars, sports balls, and so on.

As an example for the first option, some embellishments could be:

- Growing some flowers
- Turning an Archimedian screw
- Boots kicking something
- Hammers hitting something
- Sending a message to aliens
- Building space rockets and launching them
- Rescuing a kitten from a tree
- Setting off a flock of birds
- Disturbing a hive of bees
- Making a sandwich
- Building a dog kennel for a puppy
- Causing or preventing nuclear holocaust
- Baking a cake
- Choosing between Heaven and Hell for a departed soul
- Milking a cow
- Ringing a bell
- Opening or closing a book
- Lighting a candle
- Exorcising a ghost
- Rescuing a test beagle

- Squashing a goose-stepping dictator
- Releasing some balloons
- Hatching an egg
- Setting off fireworks

If an event requires more than one action to trigger it, then the previous levels should give the player a warm-up for what is expected.

For example, the case of an exorcism requires a bell, a book, and a candle, and the levels could progressively build up to this exorcism in the following manner:

- Level (n-3) Ring a bell.
- Level (n-2) Open a book.
- Level (n-i) Light a candle.
- Level (n) Exorcise a ghost.

We'll briefly explain how some of these ideas could be implemented.

The First Option

This is the more detailed option, with special bonus functionality built into the levels. Here are some examples.

- *Contacting aliens*—There would be five Tannoy-style loudspeakers arrayed around the screen, with switches linked to them. The object of the level is to get the balls to activate the switches in the correct order to play the five-note alien contact message from *Close Encounters*, and get them all into the exit. If the message is correct, a UFO flies down and buzzes around the screen before landing. The occupant (a gray) will get out, wave at you, and then get back in and fly off.
- *Baking a cake*—This level presents a system of delivering ingredients (custard, sugar, chocolate, butter, and cream) down a ramp into a cooking bowl. The ingredients are placed on top of crumbling blocks, and, when the blocks are destroyed, the ingredients fall onto directional blocks that propel them into the bowl. The catch is to avoid losing balls in the same way. If you succeed in baking a cake without losing any balls then you get to see the cake rise.

- *Rescuing a kitten*—A kitten crawls around the level chasing after the balls. Avoid crushing the kitten and keep it safe. If possible, lead it to a cushion so it can fall asleep.

The Second Option

For the second option, replacement graphics for the balls could be rewards for every tenth level. This would require much less work than the first option and would be more suitable for multiplatform release.

Some examples could be:

- Sports balls
- Small furry animals
- Alcohol (bottles and cans)
- Cars
- Cakes
- Vegetables
- Insects
- Planets
- Ghosts and monsters
- Fruit
- Lemmings
- Wild West
- Robots
- 1950s Sci-Fi
- Fish
- Types of cheese
- Livestock
- Rodents in “hamster-balls”