FROM START TO FINISH:

THE DIARY OF A GAME.

"Andrew Braybrook is the guy responsible for such Commodore 64 classics as Paradroid, Uridium and Alleykat. He was also one of the team behind Firebird's superb conversion of Rainbow Islands. His next game is more of a labour of love than a job, as he is converting Paradroid to the Amiga. In the first of a bi-monthly series, we follow the game's progress, as he begins an exclusive diary of a game."

The story so far...

I started writing 68000 in June 1988, and wrote a text font plotter and some sprite plotting routines first, as Graftgold didn't have any sprite-based code at the time. Our only other 68000 coder, Dominic Robinson, had been writing 3D vector routines and some fractal programs before progressing onto writing our Object Orientated Programming System (or O.O.P.S!) Kernel. After a brief dabble in the world of fractals myself, I had a go at writing a scrolling background system for Paradroid. This work was shelved in September 1988 when we took on the conversion of Rainbow Islands for Telecomsoft. That work was completed at the end of June 1989 when we were contracted to produce Paradroid on the ST and Amiga by Hewson. We are contracted to finish in April 1990, not before, so provided they do their publication properly taking the three months that they always tell us they always need, then don't expect to see the game on the shelves before July 1990.

I began the project by throwing away most of the earlier program as most of the coding was well out of date. We tend to write very re-usable code now, being heavily data-driven, and we have got a solid core of code for controlling sprite-based objects. I started by isolating all of this code from the Rainbow Islands-specific code and making some planned improvements, removing seldom-used bits to streamline the code, and adding some new features that I thought would be useful.

I'll be referring to my AMP system quite a lot, so I'll start by explaining what it is. AMP stands for Alien Manoeuvre Program, which was pioneered by Dominic Robinson and John Cumming some time ago before they joined us - although they claim I started it all off in Uridium way back in 1986. It's a method of controlling, originally, alien movements in shoot'em-ups. It's a sort of interpreted language, although when I used it, it was just for the Uridium ships getting movement instructions from a list. John and Dominic took it much further, controlling grouping of multiple objects, animation and collision controls, with looping constructs and suchlike. I have built a lower level version of this system, as the 68000 is the first CPU with enough muscle to be able to cope.

Anyway, an AMP is therefore a sub-program exclusive to an individual object telling it what to do, and can be used for just about anything, from running the title sequence, to moving a cursor around the screen, to making the doors open and shut, and controlling the player. I'll also be referring to sprites quite a lot, and usually I mean blitter objects rather than hardware sprites, but I can't quite bring myself to call them blobs! I'll start the diary from when I began full-time work on Paradroid, as there was a hazy area in the middle when I had to return to Amiga Rainbow Islands briefly to carry on the upgrades. Up to this point I had mostly been just doing the graphics for the game.

Monday 31st July

In order to test some of the new features in the core of the AMP system, I decided to design the title sequence. I first designed the logo. I had intended to split the title 'Paradroid 90' into separate letters. Paradroid is a great word for compacting, as three letters occur twice, thus saving space. I wanted each letter to arrive in turn, each casting a shadow onto the background. The logo turned out to be quite large and involves multiple colour changes on each line.

I also redrew the old font from the 64 version, as I thought it would be nice for it to make a re-appearance. This didn't take very long, although separately saving each letter as a brush soon got tedious - we must write a font editor some time! I use my own format for fonts, I wouldn't use Amiga format fonts if you paid me. The only editor I have for them only works with KickStart 1.1 anyway.

Tuesday 1st August

Continuing the title sequence, I drew some different coloured hexagons as a backdrop to the logo and credits and wrote the routines to tessellate them on the screen. I also wanted the credits to fade in, letter by letter, so I had to get the plot routines to talk to the text interpreter. The letters fade through four or five colours, but look a little flat on the screen in one colour. My solution to this was to run a colour fade through the text, which makes the letters look more interesting, but ruins the fade effect somewhat. A solution is not forthcoming at this time. Never mind; plan B: do some graphics.

Wednesday 2nd August

The logo is split into letters and my system can cope with them arriving one at a time, but moving them all at once causes major headaches for the system; there's a lot of plotting going on. Plotting eleven 32-pixel-wide sprites onto a non-blank background, with shadows, as well as running colour changes is too much. I can only print four lines of text with colour-bars and music running before the frame-rate drops and things really start juddering.

After much tinkering, here it is - the Paradroid 90 logo in all its glory!

Thursday 3rd August

If I split the logo into 32-pixel-wide chunks instead of letters, I can get the logo into seven sprites instead of eleven, and by using an enormous frig to fiddle the amount of background restored to a minimum, I can run it faster, but I'm still not happy with the overall look.

I split the logo into 32-pixel-wide chunks as my plot routines all need either 16 or 32-pixel-wide data. This simplifies clipping off the side of the screen and is adequate for most things. Larger sprites are displayed by ganging-up images next to each other. I want the logo to move as a whole and am prepared to drop the letter-by-letter arrival.

Friday 4th August

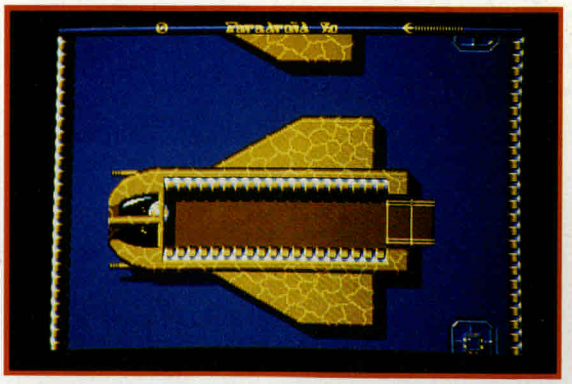
Nailing down the colour palette is always difficult, but does need to be done early as all the graphics must be drawn with respect to the palette. I use DPaint II to sketch things out and experiment, but I use Art Studio to actually create and store my sprites. I write my code on a 1040ST with the Tempus editor and assemble with the HiSoft assembler, and link the object together with the Metacompco linker. I assemble most of my graphics into the code itself, so ultimately I need them on a GEM disk. We have various means of getting graphics to and from the Amiga, mainly involving a lot of time and pain, so I try to avoid it as much as possible. I have something of a love/hate relationship with the Amiga: I love what it does, but I hate the way it does it!

Back to the palette. I've decided on sixteen basic colours, split into four sets: lone miscellaneous set of four colours, and three sets of four related colours, ie. the same basic colour but four different brightnesses. This gives me four greys, four yellow/oranges and four floating colours for the different ships. The miscellaneous colours are black, a colour for space (probably very dark blue) and variable colours for the alert status display.

The observant among you may have noticed that sixteen colours is less than the Amiga is capable of, but is as many as is practical. Running 32-colour graphics costs 25% more space for the screens, 20% more space for the graphics and an effective 16% loss in CPU and blitter time due to extra screen DMA. I would ideally like this game to run at 50 frames a second, but this is not possible as the amount of screen plotting that I require, even running at 25 frames is going to be pushing it. No, the hardware sprites aren't going to help because I need the robots and bullets to interact more with the background, going under door lintels and lift controllers. I'm much more concerned about producing a good, playable game than putting something on screen that's more amazing than the best demos you have seen, but has no lasting interest. Trust me, I know what I'm doing!

Monday 7th - Friday 11th August

I spent this week preparing the graphics for the doors, energisers, alert blocks and various combinations of walls. I also converted the old AMP routines for running these objects to the new system and tested them. I will be using a map editor to create the decks for the ships, but this has not yet been written. John will convert the Rainbow Islands mapper to support a larger character set, over 1000 characters instead of 256, 16 decks plus a scratchpad instead of 4 rounds, and any sized deck instead of 256 X 40 fixed. Until this is done, I will use Rainbow Islands round filled with my new graphics - it'll be good enough to sort out the scrolling system.



Extra features are being added to the Amiga version. Seen here is a shuttle naturally enough - a deck's shuttle bay: "a nice set piece to have."

Monday 14th - Friday 18th August

Started the week by designing the graphics for the influence device which the player controls. Being on screen all of the time, this should be graphically superb, as well as being small so that the controlled robot underneath can be seen and identified. I want to have very few gauges and words on the control panel, mainly because the actual screen playing area is so large, and so the player's graphics should show the different modes that the influence device can be in. Anyway, what does an influence device look like? Answers on a postcard to...

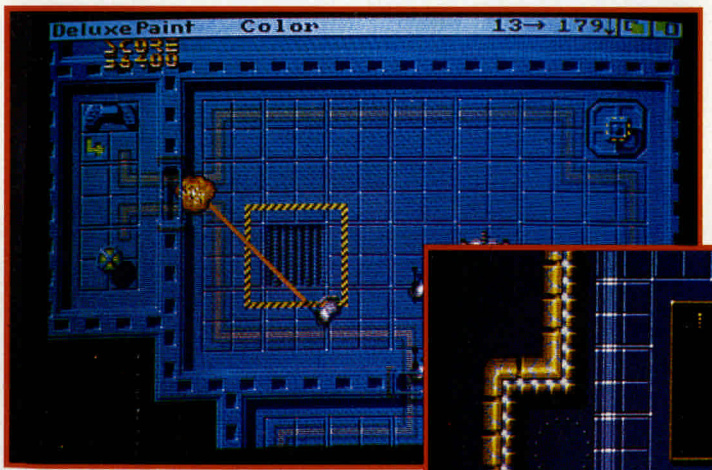
The week's proceedings were rudely interrupted by Rainbow Islands being officially tested at Commodore's Hall of Fame, where they have one of every version of the Amiga ever available. There were quite a few. The upshot of it is that some of the Gary chips do not allow blitter-friendly mode, which is quite a blow, as I used that extensively. We therefore had to think of a new strategy for blitting objects. Fortunately (or unfortunately), we know two people with such Amigas so we can test the new version without going all the way to Commodore. This is all relevant to Paradroid, because we will be using many of the plot routines from Rainbow. Had to fix my 1040ST which was refusing to read disks. Took it apart, poked it about with a big stick, put it together again, and, voila, it works again. Marvellous thing, technology.

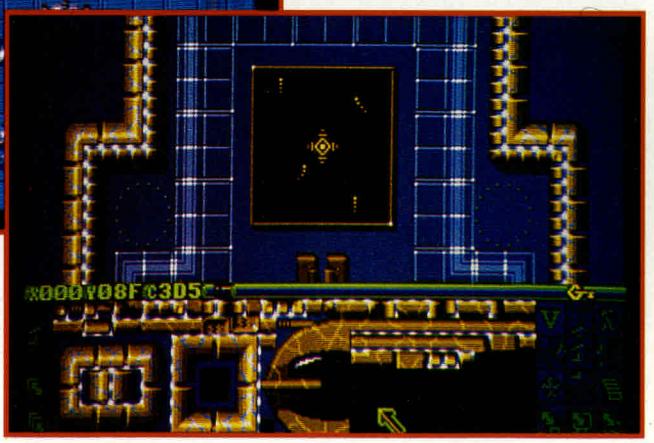
Tuesday 22nd August

At last, I'm back to Paradroid. Today, I'm redesigning the 'cell system' that we invented for Rainbow. The background will be broken up into characters, just like the C64 version, which makes collision detection robots disappear if they can't be seen by the player. The cell system allows us to put the characters on top of existing ones, on or off screen, singly or in sets. These cells can be piled on top of each other and were used to do the Rainbows themselves. Now I wish to use them to do the doors and other destructible blocks on the decks. I need to be able to plot or unplot one of the many cells, and hide them. The cell system could also be used for laser fire. What the cell system allows me to do is put objects on the screen that don't move against the background, and leave them there until such time that they are removed. It cuts down tremendously on the plotting overheads. Plotting objects is one of the biggest spenders of time in the Amiga, the blitter is fast but at the expense of stopping the CPU except when the blitter pauses for breath.

Thursday 24th August

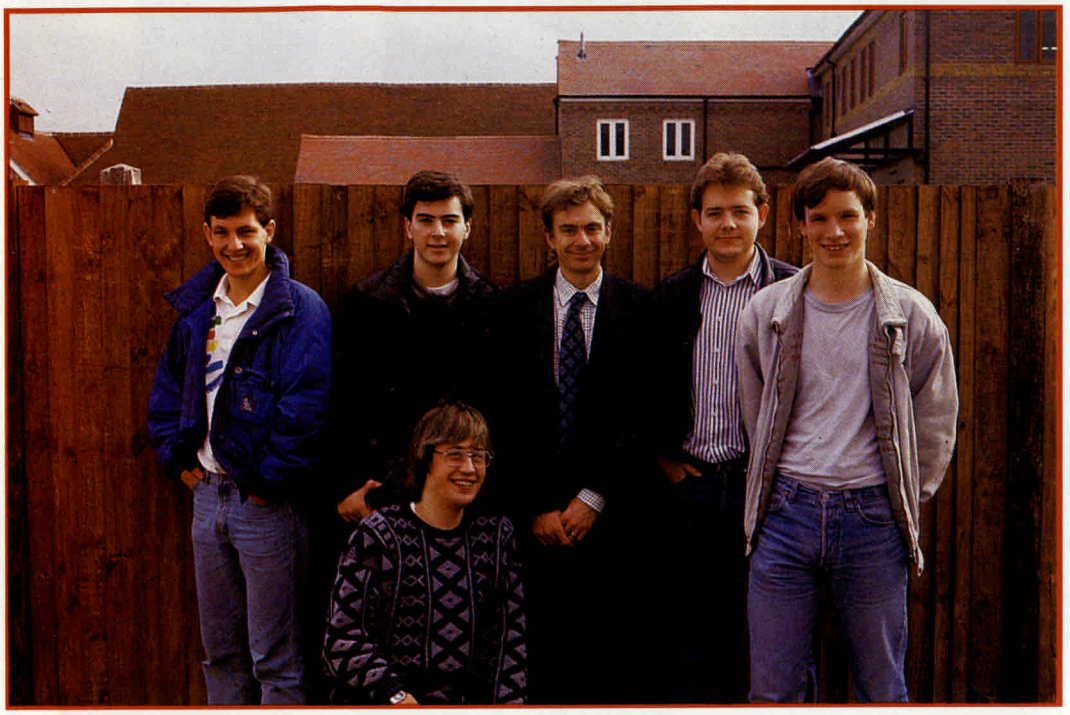
It's time to implement that cell system; there's no putting it off. It actually turned out to be a lot easier than I was expecting. It just looks messy when you have to link up a doubly-linked list in one place and three singly-linked lists in another. The system is pretty flexible though, as I can get from any part of the system to any other quickly. Whether I can use the system to its full potential depends on how many things I can think up for it to do.





Early decks shots. The picture on the left is an experiment using DPAint II, whilst the second set of pictures on the right are the finalised graphics - for now, anyway!

Graftgold themselves. From left to right: Gary Foreham (programmer on other projects), Jason Page (sound and graphics), Steve Turner (the boss!), Dominic Robinson (O.O.P.S kernel), David O'Connor (prohrammer on other projects). In front, is Andrew himself (diarist extraordinaire and self-proclaimed mega programmer!).



Friday 25th August

Put the doors in, and got them working. The doorway is effectively open, but the cells are plastered on top to show a closed door. When approached, the cells go invisible and a sprite takes over to show the door opening, and remains there until no robots are near. It then shuts, the sprite disappears, the cells reappear, and the door appears to be closed. The beneficial side-effect of all this is that I can leave a door lintel there when the door is open and the robots will go beneath it. I shall also use the cell system to do shadows for other static objects suspended above the deck, like the energisers.

Tuesday 29th August

Made up two RS-232 leads today, which we use for the download of the game from ST to Amiga. We are one short anyway, and we also need another lead to get two PCs talking to each other. I did serious damage to my fingers when I stood on the soldering iron lead and the iron slipped through my fingers, melting quite a lot of my skin. "Gosh, that hurts!", I said (Yeah, I bet! - Ed), and got to some cold water fast. I expect it'll heal eventually.

Wednesday 30th August

Drew the energiser, the lift head and an alert block, and implemented them; shadows courtesy of the cell plotter. I want all these destructible so that players spending all their time on the energiser will find it shot from over them. Sitting under the energiser lights on the ground below to spin up to speed and energy is supplied.

Thursday 31st August

Changed the design of the influence device and started work on the control mode. It's a nasty one this because all the robots can be controlled will have different characteristics, as well as graphics. Also, their weapons will be different, if they carry weapons at all, and the one control mode has to cope with all this. Got the lone Influence Device skating around, which has its own shadow casting on the floor below. Rather than a black shadow or stippled blob, the shadow darkens the colours it covers.

Friday 1st September

Spent the day adjusting the speed and acceleration variables for the player and adding in some weapons to fire. Unlike the C64 version, which featured linear speeds, I am now using a polar system. The main difference is in the diagonals which, in a polar system, a short cut is to apply full X and Y speed in diagonals resulting in a faster diagonal speed.

Monday 4th September

John has been working on the deck mapper, which is written using STOS with some 68000 routines for graphics support and... (sound the trumpets!) the A.I Mega Compressor. This marvellous piece of software takes a set of decks and looks for similar patterns in them, and substitutes 'macro codes'. It does up to 8 passes of the maps, either horizontally or vertically looking for repeated patterns and finally removes all the spaces resulting usually in the removal of 75% of all the data.

Designed by David O'Connor and written by myself, it's probably one of the most complex routines I've ever written, not least of all because debugging it was really tricky as it needed to run under STOS. We also had to write the decompressor to get the original data back, so if it went wrong we couldn't be sure whether the compressor or the decompressor (or both) were wrong. The bad news is that it also needs to be changed to allow 16 decks of varying sizes instead of 4. This didn't turn out to be too tricky, and it worked second time, so now I can't use the deck I created in Rainbow Islands format. So it's time to design a working deck with all the combinations of elements for thorough checking.

Up to now I've had a full 25-line screen scrolling with an overlaid score and energy bar. With the aid of a border colour change I can see that this is taking a very large chunk of the available time without even running any robots around. I have therefore decided to cut the screen down to 24 lines and put the score and energy on the blank line above the main screen. I can then relegate the score update process to a lower priority task in our multi-tasking system so that it only gets updates when there is time available. The only disadvantage of this method is that, if there is a lot going on, the score might not update for a few frames. Not a heavy price to pay when so much time is reclaimed and available for the main tasks, scrolling and plotting sprites. Maybe later I will use the hardware sprites for an overlaid score, but they aren't very wide and there are heavy colour restrictions. If it looks prettier in more colours, I'll leave it where it is.

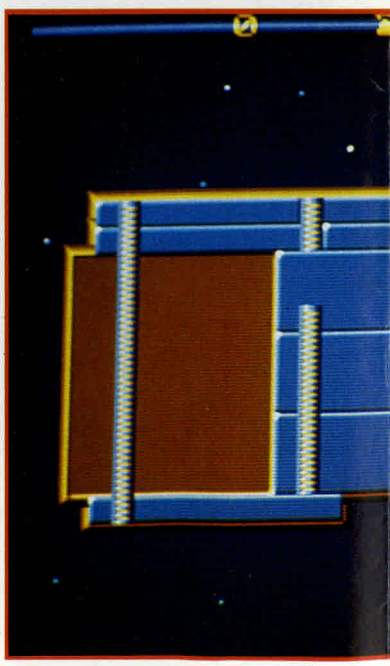
Wednesday 6th - Friday 8th September

Adjusting the screen size means changing the sprite clipping window size, ie, the area inside the map in which sprites may be plotted. This also applies to my colour-bar system to stop colour changes occurring above or below this window. Also, the O.O.P.S Kernel needs to know what size of screen I require and where any full palette changes are to occur. This is the tricky bit as the very mention of object-orientated programming makes me forget everything I know about coding.

Spent the rest of the week putting in the sprite/background collision detection to stop the robots and the player driving through the walls. This time it'll be idiot-proof. First of all, find me an idiot.

Monday 11th - Friday 15th September

Put in the first robot and a weapon for it to fire, a flame-thrower, which must also collide with and stop at walls. One of the problems about using real robot images rather



than markers is that they need to be displayed in all directions, with the lighting right for all of them, so I can't reflect them. This is going to be very expensive on memory. I have settled on 8 different images for the robots at 45 degree angles. Some robots will have independently swivelling heads, which being smaller can have 16 images at 22 degree angles for really smooth rotating.

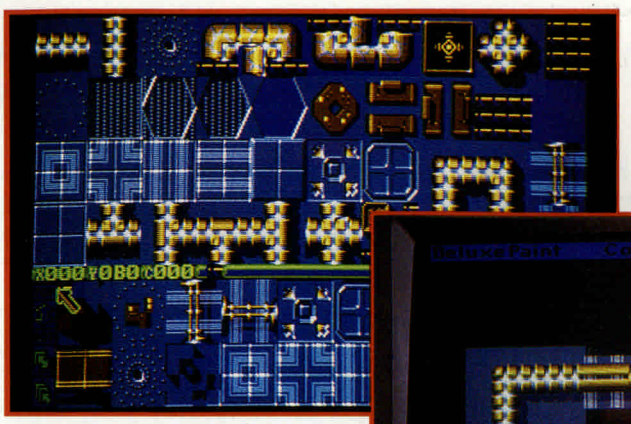
Monday 18th - Friday 22nd September

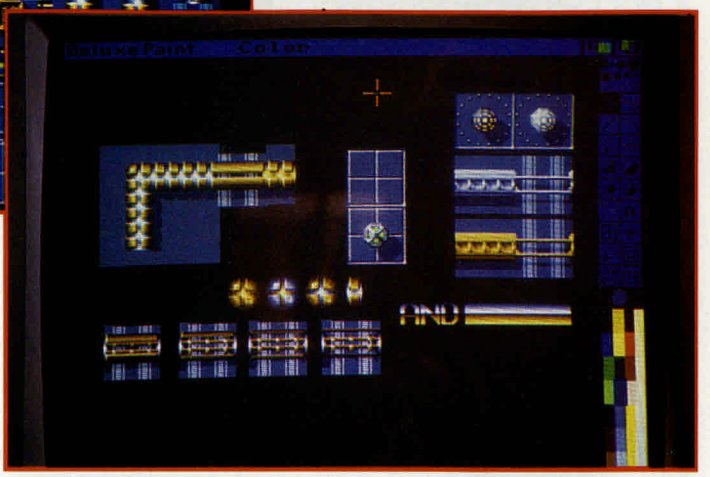
Drew an opening and closing hexagonal hatch to put on the floor as a sort of airlock. It sits there open until approached, when it closes to allow robots to walk on it. It operates in a similar way to the doors, but in reverse, so it didn't take too long to implement.

The thought of drawing up to 24 different robots in 8 different directions doesn't fill me with enthusiasm. John had this idea of designing the robots on a 3D CAD system and then we can display them from any angle. Also, this allows us to use large images of the robots for the console look-up system. John built one and did a mock-up screen showing the robot from 4 angles and it looked very impressive, so this looks like the way to do it. I dug up the C64 version and picked 21 of the original robots as a starting point for John to work with.

Tuesday 26th - Friday 29th September

All of the robots in the C64 version were moving round predefined patrol routes. I wanted most of the new ones to do that as it makes them out to be more intelligent than just blundering about crashing into things. I put in a homing system so that the robots select their next target point and head for it. Unfortunately, inaccuracies in the homing system are causing them to keep banging into door posts as the doors are quite a tight fit. This needs a rethink. The rest of the week is disrupted by the PC Show.





Using handy Graftgold's useful map construction kit, decks for Paradroid are literally pieced together like a jigsaw.

Monday 2nd October

My first robot is now patrolling round quite happily, selecting new routes at the patrol points and generally enjoying life.

Tuesday 3rd - Wednesday 4th October

Now it's time to put a second deck in and see if I can get between them using the lifts. As with the C64 version, if you get in the lift and just exit on the same deck, then all the robots and bullets will be in exactly the same place, otherwise it could be used as a cheat to reset the robots. On a fresh deck the robots are effectively unpacked and restarted as it keeps a full track of only the current deck. Of course, destroyed robots never reappear, but the others are reset to their original starting positions.

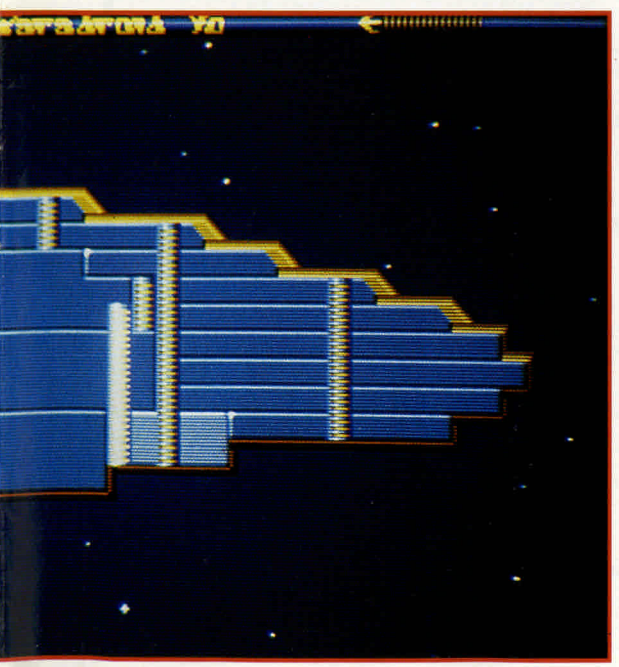
I experimented on DPaint II at home that night with the new palette and came up with a new style for the walls. Most of the other graphics need only be remapped on to the new palette. The doors need changing completely to fit in and are one of the most difficult to draw as the door shadow has to work properly and the sprites have to match the background and the cell system. This time I actually got some work done and got it saved. For some reason, the version of DPaint that I have got is very unreliable and crashes at the most inopportune moments, often when I am running short of memory. Sometimes it boots up with a screen corruption and it's easy to spot when it's going to crash; other times it waits until I have created something worth saving before it goes.

Thursday 5th - Friday the 6th October

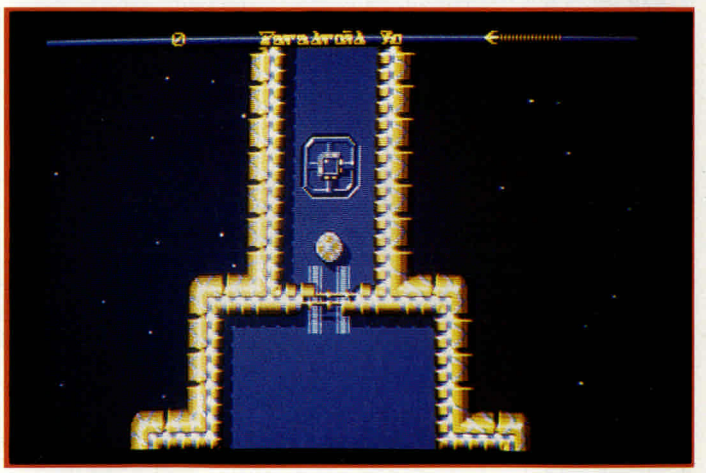
Redrew all of the graphics to date. Only the logo remains the same, as it is drawn in the two colours reserved for in-flight colour changes. In my rush to put all the pieces together, the only thing I forgot to do was actually change the colours in the palette set-up, so all the new graphics came out in the old colours. Fixed that: now it all looks much better. I've coded the routine to check whether the robot can be seen from the player, what I call 'hidden robot removal' which again uses one of Dominic's routine - this time an adaptation of a line-drawing algorithm. That also works a treat but eats more time.

A second deck has now been drawn to test out the lifts. This routine will drive an animated display, but for now I just want it to set up a new deck and work out my position on that deck. Got the scale a bit muddled up as one routine was working in characters, and the other in pixels, which caused some rapid scrolling off the bottom of the deck into oblivion.

To Be Continued...



Every time your Influence device changes levels, an exterior map of the space station is shown. At presnt, only two levels are finshed, but when the game is released you will be able to move freely between them all.



The Influence Device takes a trundle through an unfinished deck. Notice that so far the floor graphics haven't been added.

FROM START TO FINISH:

THE DIARY OF A GAME PART 2

"In the second part of our bi-monthly look at the development of a game, Andrew Braybrook talks us through the trials and tribulations that a programmer encounters whilst writing a game as he continues with Paradroid 90."

The story so far...

Work has been going well, despite a few interruptions from the still-unreleased Rainbow Islands. There are two decks up and running, with the player`s sprite able to make his way between the two. A few graphics are still to be changed, and the rest of the decks need to be entered. Since the last diary, from the 10th of October onwards, Andrew worked on the robot-to-robot collision detection and tinkered with the maps and decks. We pick up the story on the 23rd, as Andrew battles on with the intricacies of Paradroid `90.

Graftgold`s resident gurner, John Cummings, poses for the camera whilst playing Anco`s Rally Cross - what a handsome chap, eh, readers...

Monday 23rd October.

I`d been having trouble with the robot-to-robot collisions. Instead of bumping into each other and then swivelling away, the robots kept locked together and appeared to be dancing until they finally drained each other of energy and blow up. The cure for this, as Dominic pointed out, was to back out the move that got them to collide properly. I was only backing out the pixel part of the move, whereas the full co-ordinates are stored in pixels and sub-pixels, so sometimes the full move wasn`t being backed out, and the robots still touched afterwards, result: much embarrassment to programmer who ought to know better. I also populated two more decks of my ship with consoles and a couple of robots to check out the packing away of robots on other decks. All seems well.

Monday 30th October

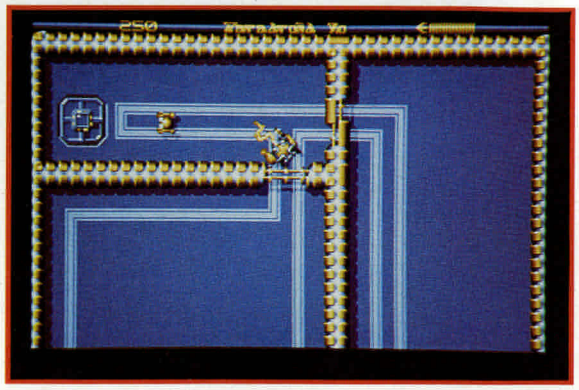
Our resident musical super-hero, Jason Page, has been beavering away writing the title sequence music using Soundtracker, although the game will run our own music routines. So far, he`s used rather a lot of memory on sampled sound, drums and things, but still no Rickenbacker bass. We`ll resample the larger instruments when we have a clearer idea of how much memory is available.

Inter-robot collision is working most of the time now, it still can`t fully cope with a three-way simultaneous collision, but with careful design of the ship layouts the liklihood of such an event can be reduced. I started drawing my first walking robot. I have budgeted for about half the robots being rotated in eight graphics frames, some of which will have three animation frames per rotation, which works out at 15K per robot body. The head being smaller, I can afford sixteen frames of rotation for a smoother effect. The heads will be independently controlled from the body to give a more realistic look to them. The internal machinations of the game will be working to 256 possible rotations, and the nearest available rotation frame will be displayed. Most weapons will still fire in 256 directions, which will look a little odd at times but I`m prepared to put up with that. A super rotating-shrinking sprite chip would be useful here, as now featured on such arcade machines as Assault, but at least by drawing the frames by hand we can get the lighting correct on all the images rather than just rotate them on DPaint.

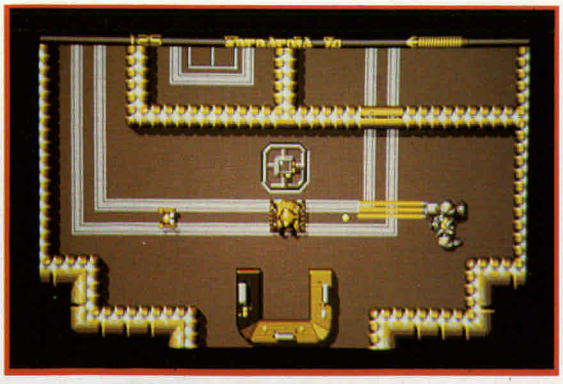
We weren`t supposed to print this one, but how can we resist printing this piccy of diarist extraordinaire, Mr Andrew `rock`n`roll is my life` Braybrook!

Tuesday 31st October - Wednesday 1st November.

Drew some more background details for inclusion in the background character set, including an airlock and some warning stripes. Started to decorate more decks and got bored fairly quickly. There`s such a large area to cover, even on one ship, let alone five. Finished drawing and animating the first walking robot, with head in place, in direction at least. He`s toting a rifle under his right arm and animates reasonable for three



Two robots battle it out as they approach the lift entrance.



The droids can now 'see' and 'hear' each other, and can react accordingly. As seen here, as a tank is attacked by a fellow droid.

frames. He`s intended for the marauding pirates. I handed the graphics over to John to separate the head from the body and then spin them and relight them correctly. He`ll be using Cyberpaint II for this.

Thursday 2nd November

As one in a series of `nice touches`, I wanted the shuttle window to be tinted and darken any robots that move under it. I was using a special shadow plotter of rectangular shape below the actual window of the shuttle to change the colours below. This effect turned out to be either unnoticeable on some sprites or a mess on others, or polarised them, so subtle shading suddenly looks horrible. So that idea was scrapped. Instead, I drew some flames for the shuttle`s engines which will be activated when any robot approaches the shuttle cockpit. These will behave as bullets or explosions and will roast any robot standing by the engines. Great for the player, unless another robot switches them on as he goes past. That has always been one of Paradroid`s strong points: anything that the robots can do, so can the player, and vice versa.

Friday 3rd November

Drew some NASA rocket engines for the shuttle to make it clearer where the flames would come from. I also redrew the bridge radar screen as no-one knew what it was before. As the lift display now takes a long time to build, I changed the display so that it shows the outside of the ship and then adds the decks one at a time. This gives a lesser delay before something appears on the screen and creates movement on the screen where there was none. The display probably takes a little longer overall to draw as the screen swapping must be synchronised with the raster output on the monitor, but is much more interesting.

Monday 6th November

Gary Foreman will be starting another project on 16-bit shortly, so I spent a while explaining how the Paradroid AMP system works, and how the game interacts with the kernel. We are looking at alternate development systems using our PCs, so a bit of conversion may be necessary. We already have a number of PCs around from the 8-bit development systems, but up to now we have been using STs. Drew a smaller messenger droid in eight frames, for what was the 302 robot.

Tuesday 7th November

Got the pirate back from John in eight rotations. After some manipulation I got all the images into my sprite format. The helmetted head looked a bit messy as he had designed it in CAD and then spun and reduced it, which resulted in fairly fuzzy images. The reduction process was not very clean, as it didn`t seem to be using a very good averaging system to pick the colour for each pixel. I spent an hour drawing all the heads by hand and produced a far more pleasing result.

Wednesday 8th November

I`ve been worrying about object allocation numbers for quite some time now. If I allow a maximum of, say, eighty objects on screen, then with this sort of game, sooner or later someone will manage to want eighty-one. At present, if an AMP tries to initiate a new object and no sprite blocks are available, a return-code is passed back and the AMP can instigate plan B, if it has one. Paradroid will be running two sets of AMP, one for the main game, the other for the transfer game, during which all the current game AMPs are preserved so that the game can continue from exactly the right place. What is needed is a dynamic system whereby any number of AMPs may run, limited only by the memory available in the machine. Chris Hinsley is already doing this in Onslaught, and it works well.

The kernel`s dynamic memory allocator is fairly fast - not quite as fast as having the sprite blocks ready and waiting, but the extra freedom of having lots more sprites is well worth going for. Having a 1MEG machine will help a lot to unburden the memory allocator, but the game must still function as well in 512K, so I still have to be careful... but I`ll code in contingencies if the memory does run out. A second problem with this approach is memory fragmentation, where different-sized blocks are allocated and freed willy-nilly, and one large block of memory becomes many small blocks. It is therefore essential to release all of the blocks of memory periodically so that the memory allocator can rebuild the large blocks again.

I put in all the changes and the program just refuses to run: it keeps leaping off into the middle of the kernel and grinding to a halt. This smells of corruption, which is a notoriously difficult beast to locate. So.... after much tearing of hair I decided to abandon that version, restore from back-up and start again. A corruption is often caused by accidentally moving or deleting a line of code, and almost impossible to spot. After putting all the changes in again, I was most irritated to discover that the game crashed in exactly the same way! Since it ended up in the kernel code, Dominic had a forage around on the stack and worked out what had been scribbled on, and what with. A whole stream of 6-byte table entries not entirely dissimilar to those in the background restore system, also being updated at the time and nothing to do with dynamic memory allocation at all.

Thursday 9th November

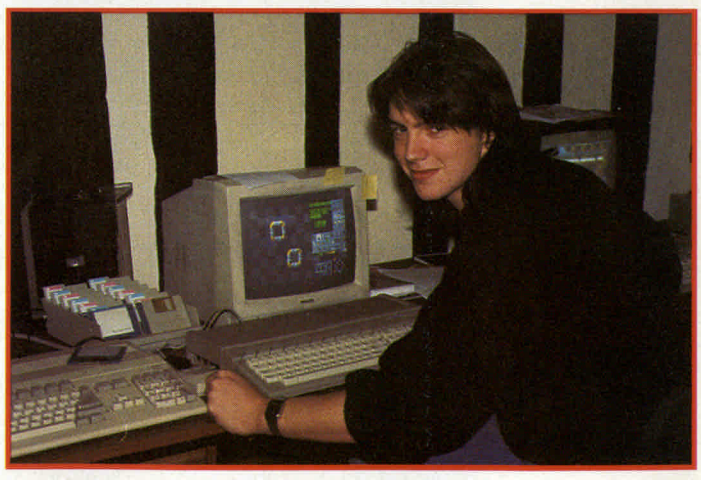
Implemented the AMP for the small messenger droid. This basically runs around minding its own business. Being a small sprite rather than a large one involves adjusting some patrolling routines as the patrol routes are measured from the top left corner of the sprites, which is fine for most of them, but not the little ones. Fixed an occasional bug that used to cause a slight screen corruption on the title page. This was yet another truncation of co-ordinates, and won`t happen again... `till the next time!

Friday 10th November

Put in a triple-laser weapon using my cell system which allows static objects to become part of the background temporarily. The laser beam is made out of a series of lines which sit around for a while in a chain and disappear one by one. The only disadvantage of this is that images have to be character-aligned.

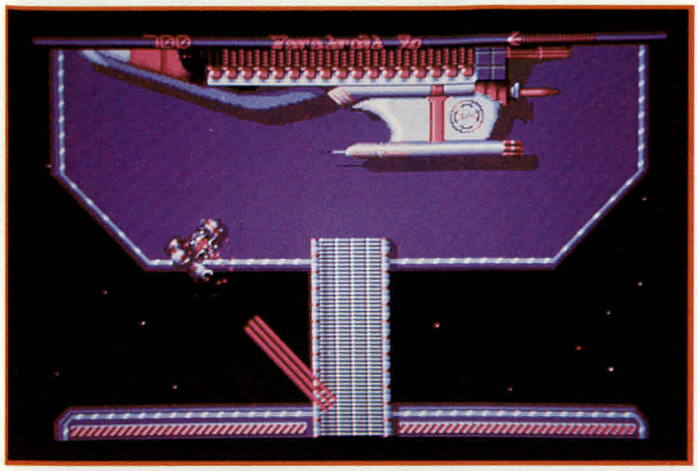
As part of the Apparent Intelligence system, I coded up the routines to allow robots to `see` and `hear` the player, as well as radar detection. Seeing involves the body of a robot asking the head which way it is facing, and calculating whether the player is visible in a sight-cone from the head. The angle of this cone is definable for each robot; some may have tunnel vision, some will have wide-angle vision. The distance each robot sees is also definable, so that some can be very short-sighted. Some robots will not have independent heads, so a second variant of the routine is required. The hearing system involves robots listening for the player`s gun to fire. In order, that fair play is maintained, they will not react to other robots weapons. Thus a robot can now stand as a sentry and watch the player, not firing. As soon as the player opens fire, the robot will return fire.

To clarify my own internal definitions, I have now designated a robot as being a composite object, consisting of a body doing all of the moving and generally being in charge, and a head which has its own program, but taking orders from its body. Additionally the body can get extra information from the head.



Graphics chap, Michael Field, takes a break from redrawing Andrew`s graphics to have his picture taken.

A deadly tri-laser spews death across the screen whilst guarding the shuttle.



The body carries the weapon and will decide when to fire based on this information.

Other types of object are: tanks, where the turret (or head) carries the weapon and does the firing, and dumb objects having no independent head. These might be simple cleaning devices or security orbs.

Saturday 11th November

Implemented a small particle explosion for when bullets hit solid objects, as in the C64 version. Each particle is running under its own AMP and has its own shadow. Generating about a dozen particles is enough, three at a time. An unexpected bonus is that since the bullet has hit a wall to stop in the first place, and since the particles are generated in random directions, those that head further into the wall are instantly purged, so only those moving back from the wall appear. I was wondering how that could be done.

On a 1 Meg test machine it took a while to spot, but `The Creeping Death` exists deep in the bowels of fast memory. After generating a few particles I decided to check on how available memory was doing, and something was eating it up, slowly but surely. I wrote a `sniffer` routine to execute after each object and spot when some memory goes astray, it`s still looking.

Sunday 12th November

Getting really worried about this memory loss. What was the question again? Oh, yes, `The Creeping Death`, well there are only a few places where memory is requested, and even less where it`s given back, so it didn`t take long to narrow the culprit down to the background task which is responsible for looking along the leading edge of the scroll window for new objects to activate. This marvellous piece of code says: "Ok boys, new object to create, get some memory, great, now... oops, that object is already alive so I don`t need to generate it, let`s exit" which consequently forgets about the memory it has just been allocated and it`s lost forever. A quick re-arrangement of the code and everything is alright again. Now I can sleep at night.

Monday 13th November

New graphics artist starting today, one Michael Field, whose first task will be to draw my robots and backgrounds properly. We`ve decided to double-light the objects, from a major source from top left, and a minor one from bottom-right. Using the new palette we should be able to get some excellent looking robots.

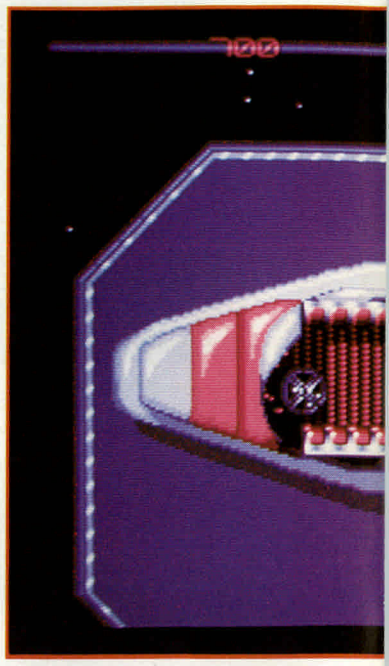
Tuesday 14th November

Implemented a proximity mine-firing weapon. The mine is flung across the room, lands on the floor, waits for about five seconds and then explodes. It also explodes if a robot touches it during that period. Again, this uses the cell system and suffers from the mine `snapping` onto character boundaries as it stops moving. Don`t like it at all. Some problems were also observed as the mine skids across the floor. It must stop at the consoles and other low objects which can normally be fired over. The biggest discrepancy is still that firing a mine across an open area of space will cast a shadow on space, destroying the whole space continuum with it. I`ll have to think about this one.

Thursday 16th November

Up to now the flame-thrower weapon has been drawn in sprites and can only fire in eight directions. This severely limits the firer of this weapon, so I redesigned the weapon to fire a trail of particles which look remarkably like the original flames, but can fire in any direction. It takes a while to run it as each particle of the flame is individually animated and drifts slowly outwards. It does save a lot of graphics memory and plot time so I`m happy with it, though.

Started laying the foundations for the disruptor and designed a new directable scatter-fire weapon which builds in front of the player before firing seven bullets in an arc. I pressed my FanFire routine into action,



previously seen releasing bats from Big Dracula on Monster Island.

Friday 17th November

Tried to design a new Influence device and put in a new mine design which does not use the cell system as it is animated, which looks a little better but still can`t be fired across space. It`s not so much the shadow, I think I could get around that, but firing it across an open airlock you would expect it to drift off into space, not just sit there in limbo. More thinking needed.

Monday 20th November

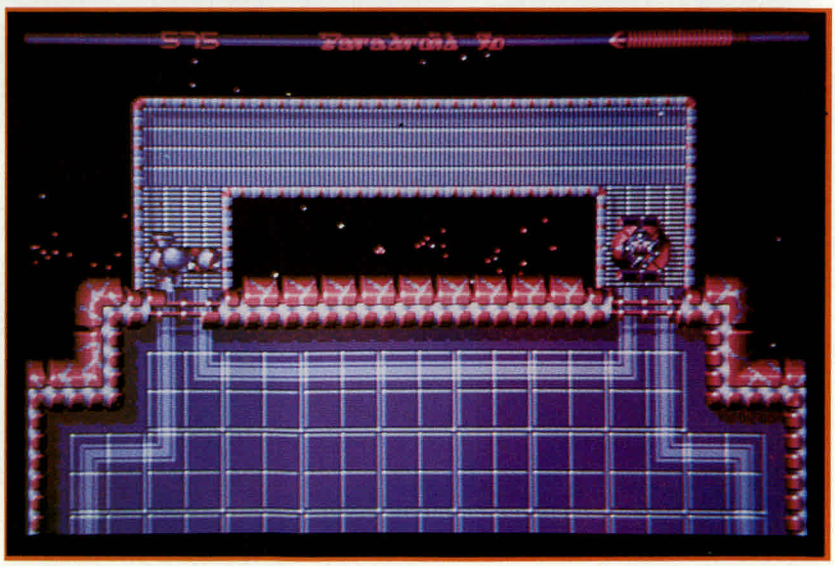
Due to a monumental error introduced into the robot`s AMPs, the robots keep killing themselves and each other before I can get to see them. As soon as a robot hits a patrol point, it dies. I was using the wrong manoeuvre error vector, so a quick wave of the magic mouse mat and bingo, all the robots are happy again, except the meesenger, which refuses to be killed at all.

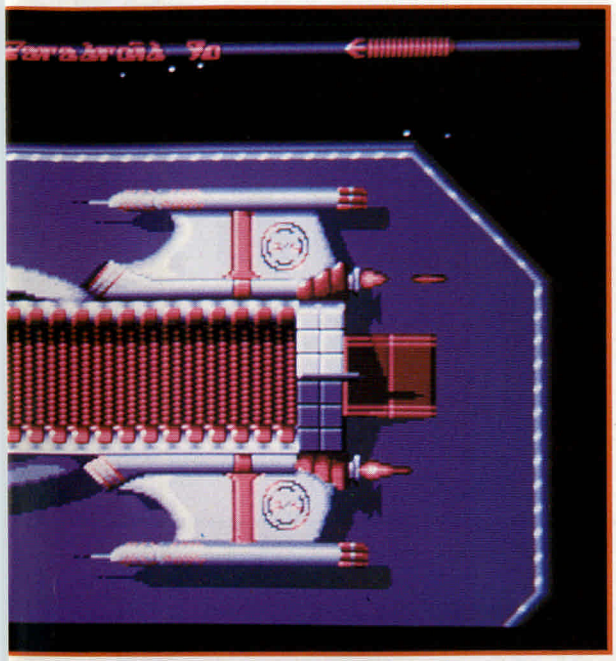
Tuesday 21st November

Implemented the mini battle tank. I have decided that this vehicle will not rotate. I shall use them in one of the cargo bays to block the player`s path. They will have a lot of energy but a fast decay rate so the player will not be stuck in one for long.

David O`Conner suggested when robots die they lose their heads and career about madly for a while before blowing up. This was not too tricky to do and adds to the fun, as getting too close to one of these is really dangerous, they fire madly and crash into walls before exploding.

The Influence Device plays a deadly cat and mouse game as one of the higher-ranked robots pursues it. Andrew hopes to have up to eighty different robots in all, ranging from deadly war-machines to menial droids that simply deliver messages and tidy the decks.





Seen here in all its glory - the NASA shuttle. As droids pass over an adjacent square, they can ignite the retros, killing any unfortunates in the burner's path!

Wednesday 22nd November

Designed some larger consoles and put them into mapper, which makes them look more like the correct scale. Put in the new Influence Device that Michael has drawn, which will allow a controlled robot to be seen through the device as it splits into four segments. The materialisation sequence looks a lot better too.

Thursday 23rd November

Put in the disruptor for the first time, as a hybrid missile type which globally collides with everything. As with the C64 version it is necessary to know who fired the disruptor so that points can be awarded for player kills but not for robots killing robots. The player must also be immune from his own disruptor. All disruptor-firing robots will, naturally, be disruptor-proof, and some other robots will also have this feature. Michael is busy doing a laser-toting robot built like an American Footballer. He looks real mean, the footballer, not Michael.

Up to now the player control AMP has been something of a dishevelled mess, and badly needs rewriting to incorporate the ability to transfer.

Friday 24th November

Rewrote the player's AMP which is all working bar the fact that the Device is not releasing its slave body properly when destroyed. Our regular critic, Mr. O'Conner, has been heard complaining that robot explosions are visible even through a robot that wasn't blown up, so this has been sorted out.

Monday 27th November

After lengthy discussions with David and John we have decided on a new format for all of our graphics, which incorporates an X offset so that an animated sprite can appear bigger than any of its components without having to move the object. We will still save the graphics from Art Studio but with all leading blank lines present, and David's new sprite stripper will remove all unnecessary data and build 12 byte headers for each image. This utility will run on a PC, which can happily read three-and-a-half inch disks and provides a hard disk for storage of graphics, long since overdue, and they won't then clog up the hard disk on my development machine.

I had to re-write the front-end of my sprite plotters to cope with the new headers, which detail whether the sprites are sixteen or thirty-two pixels wide, so each sprite knows its width rather than the AMP having to know. This is an anomaly that has been needling me for some time.

Tuesday 28th November

Implemented the new plot routines, some of which haven't actually been used yet, and the changeover has been completed fairly smoothly. There is no going back now. Having all the graphics together also means that I can load the graphics from disk at run time instead of assembling them into the game. That means the intermediate file is about half its previous size and takes much less time to link and write out to disk.

Wednesday 29th/Thursday 30th November

Michael has finished his battle droid, and has certainly gone to town on the size of it. I originally intended the robots to be about 24x24 pixels big, but this one is 32x32 and with the extra offsets he'll end even bigger. That will make it quite a squeeze through the doors.

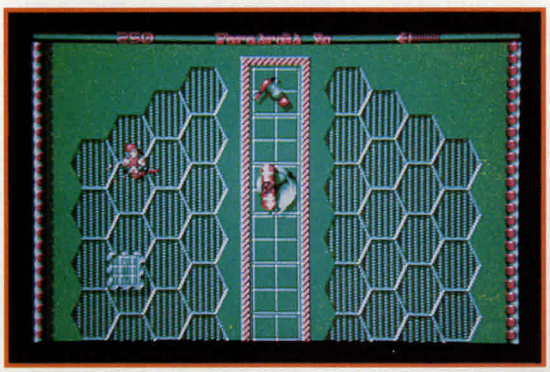
Implementing the new robot as the laser he fires is limited to eight directions, so if he stands and keeps firing and missing he looks pretty stupid. Thus I made him fire once and resume patrolling. This makes him more of a menace because he still may not shoot at the player, but in conjunction with other robots he can block the player's escape. Keeping on the move should allow him to hit the player sooner or later.

Delved inside the Amiga again to stop screen break-up as mentioned before. A quick reseating of the chips seems to do the trick.

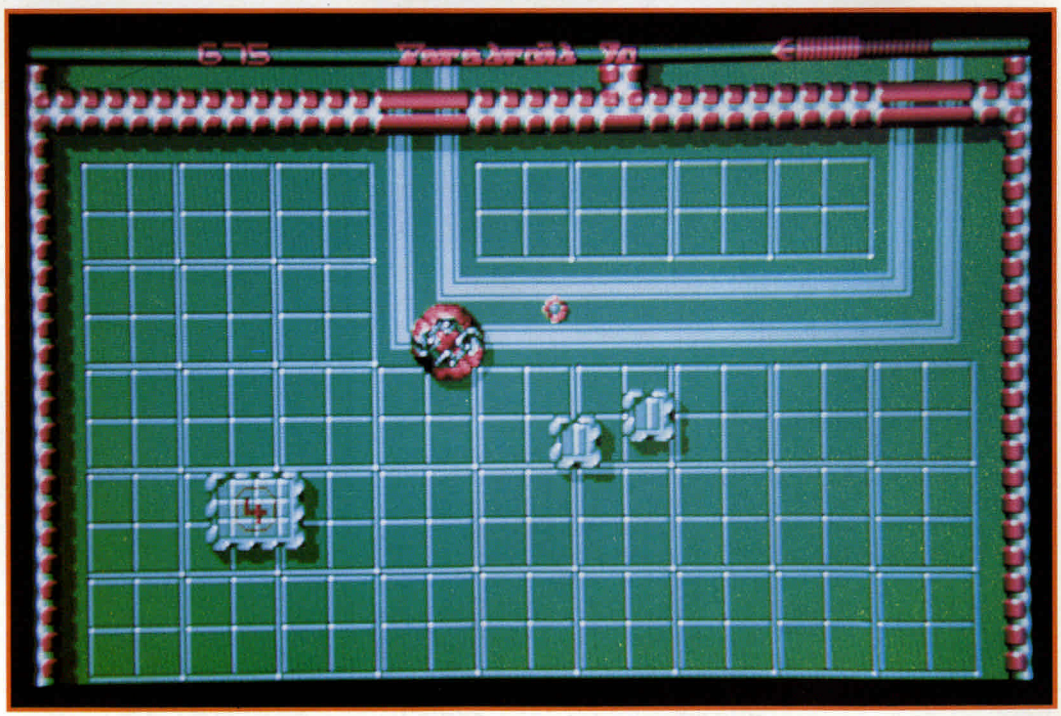
Tuesday 5th December

Michael has been drawing a new security droid with two-head mounted guns. This looks suitable to fire my particle-scatter weapon, except that my game system doesn't currently support the release of two independent bullets from two separate places. I'll either have to add some routines, or cut one of the guns off. Having spent ages rotating the head in eight directions, Michael may not be too impressed at changing it.

To be continued...



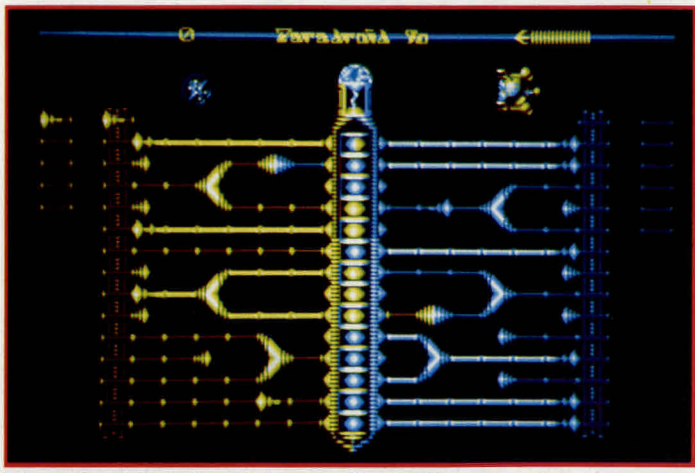
A robot, looking suspiciously like an American footballer, goes walkies through one of the four existing decks.



The influence device makes its way over one of the sketchily-drawn decks. Expect more detail in the final version.

FROM START TO FINISH:

THE DIARY OF A GAME.



The all-important transfer game. Controlling a small block to the side of the screen, your basic aim is to light up more of the dots in the centre than your robot enemy.

Tuesday 2nd January

I've now worked out all the robot patrol routes for the first ship. I don't have an editor for these and so I have to work them out by hand. This makes it important to get the layout right the first time, as making adjustments is really difficult. It's usually better to throw the whole layout away and start again. We have a patrol route editor for another game which nearly does what I need, but I can put patrol layouts in fairly quickly, and it cuts out having loads of little data files around to lose or mix up.

The control mode I've been using up to now has been fairly simple and, after reading the C64 code, is missing some of the niceties of its forerunner. It was a little too easy to get into transfer mode, so now you have to centre the joystick for a short time with the button down to get the transfer spark out. Fixing this has the beneficial side-effect of allowing continuous auto-fire provided the joystick is not centered for too long at a time - very useful for the machine gun-toting sentinel.

Wednesday 3rd January

A busy day, today, it is. Got the verb in there eventually. After the Influence Device blows up, I wanted the robots to continue on their patrol routes as normal, so it requires that the robots can no longer 'see' the I.D. so they won't react to it. Unfortunately, the mechanism to achieve this also handles the hidden robot removal, i.e. the non-displaying of robots obscured by walls. Thus as the I.D. disappears, all the other robots do too. This is a little untidy so I've persuaded the system to continue to display the robots that were last viewable, but not those that weren't. I didn't want previously unseen robots to suddenly appear.

Thursday 4th January

Started work on the transfer game. Steve Turner, the boss-man, says if it only took me a week to write on the C64 it should take less on 16-bit. I reckon it took nearer two weeks to design and implement on the C64 so I ought to be able to impliment a superior one in about that time, given that the graphics will be a lot better. After one day its putting up a display of yellow and blue wires, but on the wrong sides of the screen.

Friday 5th January

Day two on the transfer game and it's putting the colours on the correct sides and I'm teaching each of the elements of the game how to pass power along to the next element in the chain. Wires are easy, they just pass the power along, as it is, but splitters and joiners are a little more complex. The auto-pulsers are going to be animated this time, and I want to tie that in with power passing, preferably using no bytes at all.

Tuesday 9th January

Spent yesterday implementing the anti-copy protection on Rainbow Islands and some of today testing it. Back to the transfer game, day three, and I've been explaining to it how to set up the game, putting different elements together to produce a valid working arrangement. Some patterns of splitters are invalid and must be avoided. I didn't like the way the C64 version did it, and it also relied on looking at screen character codes, which are not available on the new version.

I fiddled the random element selector to force lots of splitters and joiners so that in a short time it would run through all possible combinations. It's still getting one particular arrangement wrong, putting a dead end on an already existing wire. There are some complex rules for positioning a joiner.

Wednesday 10th January

Day four, and I really need some proper graphics to get the absolute positioning right. Up to now I've been plotting some line-drawn sprites just to get the idea of what's going on.

Sat down with Art Studio and had a puggle about with some pixels and came up with a style I could live with. Drawing the various elements in that style proved a little trickier as the elements are sixteen pixels wide for storage and plotting convenience, but they'd really like to be an odd number of pixels wide. I've done the wires like fluorescent tubes which light up and stay on while power flows, and then fade down. The pulsers and auto-pulsers have 'pumps' on to generate power, and the colour switcher is a combination of both colours. I think the elements are a lot more meaningful than the coloured blocks used in the C64 version.

The central indicator bar now features a built-in timer which fills up while you choose sides, then empties out during play. This avoids the need for integrating a digital counter at the top, thus integrating the design. I never used to have time to look at the counter anyway.

Friday 12th January

Day six, and I can now select which side to play in transfer game, and the waiting pulsers are being swapped from side to side. This type of display is a pain, being double-buffered, as two copies of the screen are required and must be kept identical at this stage. The solution to this is to painstakingly update the currently hidden screen, then swap the displays over so the hidden one becomes the visible one, and then block-copy this new screen back over the hidden one. Our font plotter is so slow that this copy is quicker than doing the changes again. This is due to the colour remapping facility which is very useful but takes an age.

Monday 15th January

Day eight of the transfer game and it's time to put in the control routine for the robot-controlled opponent. The C64 version was as daft as a brush and just picked one of the lines at random to fire down, went there, and tried to fire, often firing down dead-ends and colour-switchers. This just won't

Before the transfer game begins, you are treated to a picture of the droid you are about to link with for supremacy. Mike Field drew each of the droids using Cyberpaint 2, and started of with a simple line drawing of each of them. Each one is hand drawn, starting as a rough sketch, and gradually being filled in and polished. After that, the picture is added to a 3D grid and extras views of the droid are added.

do for the Amiga. What I've written is a line evaluator which examines the side it has been given prior to play. It follows each line in turn, awarding points for good elements, minus points for bad ones. If it reaches a splitter it follows both routes in turn to the centre. It seemed like an ideal oppurtunity to write my second-ever recursive routine.

After evaluation, a list of points per line is left. It then reads through the list to find the line with the highest point value that doesn't currently have a pulser on it. It moves to that point and fires. By juggling with the points for each element I can rig it so that lines with a colour switcher end up with negative points and will not be fired down at all. Just as a final addition, it also gives one extra point for a line leading to an enemy colour in the middle, so it favours firing down lines that can capture a colour.

What this all means is that the opponent now displays apparent intelligence and can be relied on not to make mistakes. It must be beaten by choosing sides carefully and waiting until the right moment to fire each pulser. I think I have resolved all anomalies of the C64 version, like the situation where two auto-pulsers are firing at each other. The C64 version has to let one side win; the 16-bit version allows a colour bar in the centre to be split and awards half points to each side.

Michael has managed to lose half the background character set but fortunately we had a recent backup. Our 1 meg Amiga has died again. All it does is go green.

Tuesday 16th - Wednesday 17th January

Maintenance days, fixing slightly annoying things in the game, like the fact that pirates can't fire diagonally up and left. Their bullets are emerging slightly too close to them and think they have hit something so they blow up.

The I.D. once entered a lift and emerged in the middle of deep space far below the deck, due to its energy being zero but having transferred simultaneously with being blown up. This confused the system completely.

Poking the chips about in the broken Amiga while it's on (not recommended, really) highlights the big square beastie as the one causing the trouble, Fat Angus. Its restraining harness is not enough to stop it leaping out of its holder, doing 3 back somersaults and not quite getting back into place before Andy Pandy and Teddy get back. Well, we're on to your game, Fatty.

Thursday 18th January

I've noticed that my main block of code assembles in 518k on my 1040ST with only 538k available for assembling, what with the CLI and `make' in there. It's time to shuffle things about and move some completed routines into another file to just link-in rather than re-assemble them every time. Brief aside: Why do most programmers not use linkers?

Jason is now working on the sound effects which I have specified. There are about 70 including 16 instruments for the music. I want the sound effects in the game to be in stereo, so sounds generated on the left-hand side of the screen come out of the left-hand speaker. I will use two channels for this, put background sounds on the third channel, and miscellaneous sounds on the fourth.

It has been suggested that each robot should have its own sound when moving and get louder and quieter depending on its distance from the player, as well as being in stereo. This is an example of a great idea in theory, but not in practice. With only four channels, more than one robot on-screen will overload the sound channel with requests for sound, let alone when they all start firing. I will settle for a different sound for each weapon, sounds for robots being hit, walls being hit and the ubiquitous explosions. The 999 cyborg may well have its own volume-adjusted droning sound as it is one of a kind - ideas are seldom completely discarded.

Friday 19th January

Michael has finished decorating the first ship, and he's used many characters in ways I'd never thought of. The A.I. Mega-compressor took 95 minutes to compress the backgrounds, removing around 65% of the data. This leaves about 28k of compressed data, more than I had anticipated. Put in a `Game On' screen to try out the colour-bar processor. I've also shortened the eyesight of one of the robots, so that his laser fires to the same length as he can see. There's no point in it spotting you, stopping, and firing at you without success. I also had to make the mines keep the doors open when dropped, otherwise the door slams on the mine and embarrasses the sprite display priority system.

This is a screen beginners can expect to see a lot - the death screen! Should the enemy droids prove too much, your death will be signalled by a blaze of static, and this fearful message.

Tuesday 23rd January

I'm not happy with the I.D. bouncing off robots and being swung round to face its new direction. This makes ramming things difficult and also if you fire at a robot but it keeps on going and bumps into you, it is the I.D. which immediately flips round and you can't shoot until you've rotated back. I need to separate the current moving direction from the current facing direction, which up to now has been one and the same. This means adding a new field to my sprite control block and changing all of the routines which use the current moving direction field to hit both fields as required.

Friday 26th January

Inputting the new sprite-facing field, which incidentally has improved the control mode no end.

I also introduced a number of new bugs, the main one being that collisions with other robots aren't handled even as badly as before, i.e. not at all.

Due to a limited palette, I have decided not to darken the deck when all of the robots have been disposed of. Instead, a message and sound effect appear when the last robot is removed, and on entry to an empty deck. A similar message is generated when no robots remain on the whole ship, indicating that it is time to leave.

Tuesday 30th January

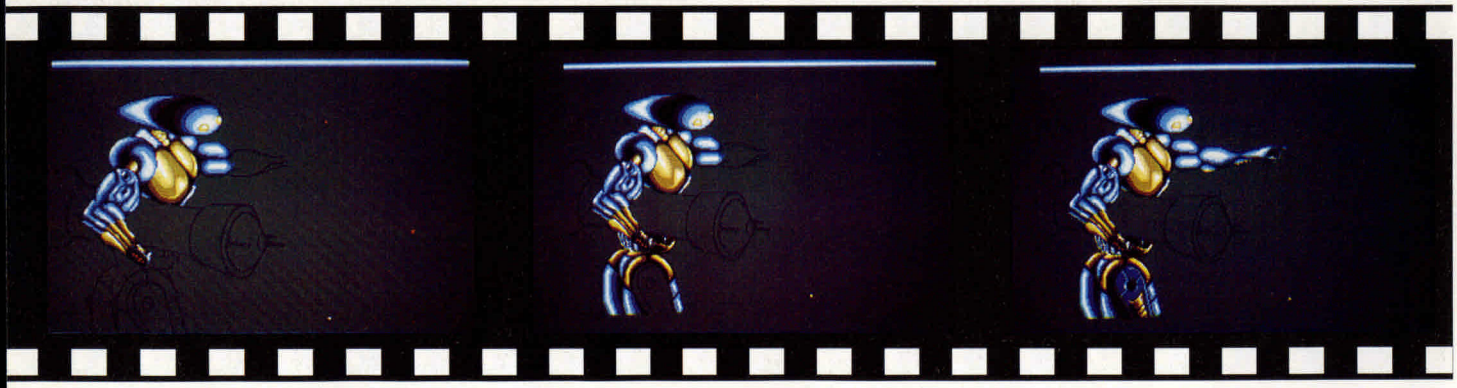
I had been putting in my sound assignments as absolute channel numbers, but in order to cater for all eventualities, such as the new Atari STE having its YM sound chip hooked up in stereo, I want to refer to the sound channels by name, then I only have to change one place to re-assign the sound effects to different channels. I could then get the channels deliberately mixed up so you have to face the monitor away from you and play watching in a mirror to get the sound effects right!

Michael has redrawn my rather scrawny messenger robot to make it bigger. It certainly looks better now.

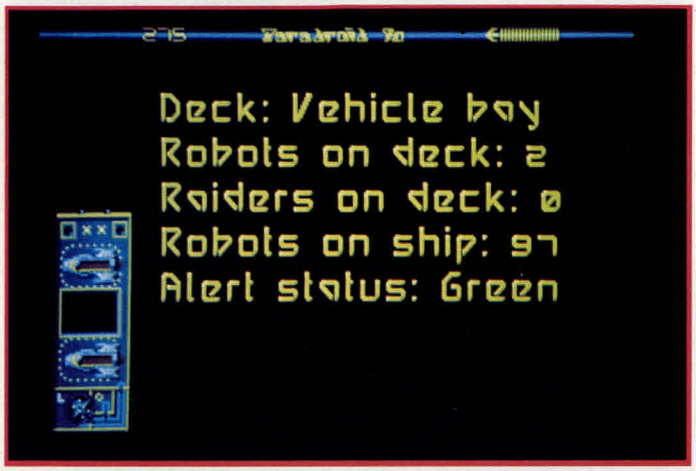
Wednesday 31st January

Finished working on the `transmission terminated' game over screen which follows the fabled `interference' screen, which appears as you lose contact with the I.D. I did the latter using four updated characters on the C64, nice and cheap, but this time it is going to be more expensive.

Rather than start the game in



This useful information screen gives indication of how many robots are left and which part of the ship you are on.



one or two preset places, I want to start on any one of the transmat positions already defined. I've clustered six such points together so that'll be the most likely place to start, but there are other, more dangerous places to start. I don't want any impossible start positions but it's nice to show off the meaner robots earlier on. It's just a case of knowing when to run away.

We've changed one of the security droids into a throbbing, wobbling droid as two of the security droids looked very similar.

Friday 2nd February

Put in the new drinks server and disruptor-toting security droid. The drinks server seems to be carrying a couple of cups and a bowl, while the disruptor droid is a shiny black mean-looking droid if ever I saw one.

Got the interference screen done using four small sprites showing similar wiggles, placed randomly over the screen. My coup-de-grace is a sync-line which wipes quickly down the screen while the interference drifts slowly up the screen.

Monday 5th February

Until now the robots have preset start positions and do not start to move until they are near the screen. This saves time as I'm not running more robots than necessary. The side-effect of this is that the robots are very predictable after a while. In order to alleviate this I've made some robots only pseudo-random. I can choose roughly what sort of order a robot will be, but not exactly. Others can be exactly fixed by type, so even with an unlucky setup I can guarantee some easy robots. In order to further randomise the deck, all of the robots are activated at the start and are free to roam about. The overheads of running a robot off screen are not too great as it has fast get-outs of all lengthy routines if it is not near to the player.

Tuesday 6th February

In order to encourage faster gameplay, raiders will attack the ship after a preset time, and get more and more frequent. Raiders have `jaunting devices' which allow them to beam directly to any part of the ship. They then wander about for a while before `jaunting` out. There will be a master countdown for each ship. No pirates will appear for about twenty minutes on the large freighter, enough time for good players to clear it. After that, pirates will begin to jaunt in, not very often at first. After a while there will be a few of them around all of the time, more on later ships.

Complaints from the lunch-time play-testing department include the fact that the transfer game is quite tricky and often it's not possible to transfer to a lowly hoover. I obliged by suppling the player with an extra pulser to satisfy hero syndrome. This crops up a lot, to compensate for the one-against-many situation and to give the player an extra feeling of power and achievement. Hero syndrome also crops up in the collision routines: the player takes about half the damage of the other robots. Hero's syndrome is present in all games, mostly in the form of the player having a more powerful spaceship, extra weaponry and smart bombs. In Paradroid 90 the player actually BECOMES one of the enemy, and as such would have no advantage over others of the same type. A small dose of hero syndrome makes the game so much more playable and therefore enjoyable.

Wednesday 7th February

It's Michael's birthday. Hoorah! Hoorah! Hang out the flags. To celebrate, he delivered a couple of new droids for inclusion in the game. I also caught the game out, putting the newly generated robots down facing in the wrong direction. This mainly affected the non-walking sentinels used to specific points. It's no good them facing the wall.

Michael had a beer or two after work.

Thursday 8th February

It's Steve's birthday. Hoorah! Hoorah! Don't put the flags away yet. I put some more robots on the first ship to complete its population of 95 droids. Had wander around as a casual observer. I can use the monitor to switch off various features, like natural energy decay, collision damage and the robots` ability to sense the player. Then it is easy to watch the robots wander about. I spotted some dragging along walls trying to reach patrol points that they shouldn't be. A quick look at the data usually locates the problem.

I've also been adjusting the pirate timer. It's not too pleasant when they start appearing, so I want it to expire only if the player is progressing quite slowly. They're actually not too much trouble to deal with: they have weak armour and not much energy, so they're easy to blow away. Serves them right, really.

Friday 9th February

Started coding the console access system where the player can log on to any console and find out more information about various aspects of the game. There's a side view of the ship, showing it inside and outside, detailing which decks still contain robots. The plan view shows a small-scale representation of the current deck, and such information as the number of robots and raiders on the deck and ship. The droid library will show large pictures of all of the droids less powerful than the player's current robot, and finally some in-game statistics.

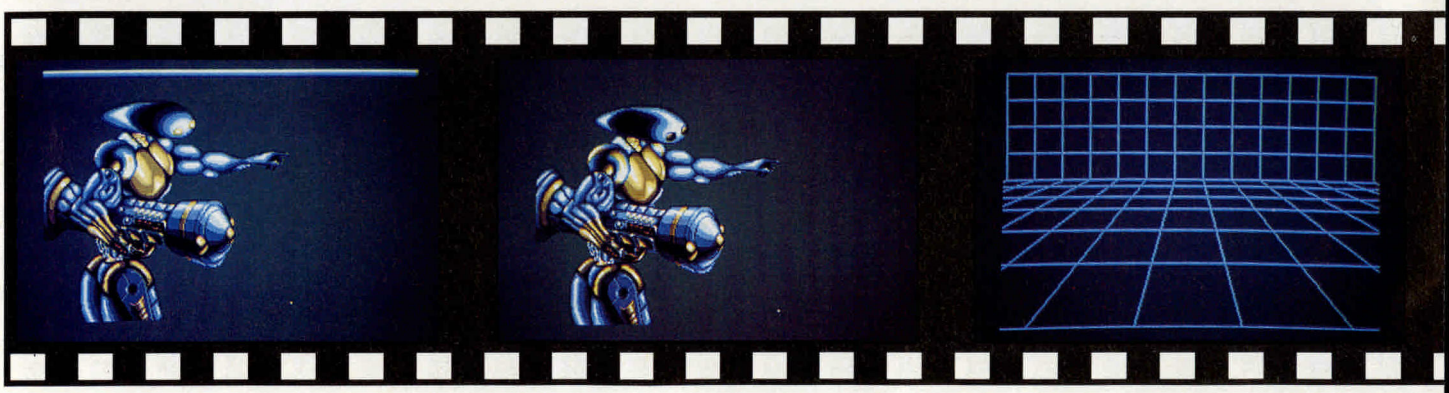
Designed the second freighter, the USF Odyssey, over the weekend.

Monday 12th February

All bar one of the firing droids has its own weapon type. The 999 cyborg is nearly complete and it would be nice for it to have its own type of weapon. I tried a Ghostbusters-style electric spark with very little success, but got an interesting fat smudgey (is that a word?) sweep by accident. Then I hit upon the idea of concentrating the beam into one sharp line, not unlike my original sprite-drawn flame which I threw away. The downfall of this weapon is that it draws a long line out of individual dots, each under its own control with animation running - quite a processing overhead. I eventually settled on plotting half as many dots with twice the interval between them. To make the weapon that bit more interesting, Michael drew an expanding star of light which forms at the gun of the firer before releasing its power.

Tuesday 13th February

Taken delivery of the 999 cyborg, floating around on his hover base. One mean dude he looks, too. The animation frames that would have been used for his walking are being used for the recoil of his gun. We tried a couple of



variations on these frames to get it just right. There's a bit of tidying up to do about the star build-up position before he fires, but other than that, it's great! The weapon gets a double sound-effect, once for the start of its build-up, and once when it is released.

Wednesday 14th February

The cyborg hover-base is a little bit too long. When rotating in a doorway he protrudes quite a lot into the walls. Michael shaved a few pixels off his base and now it looks a lot more comfy.

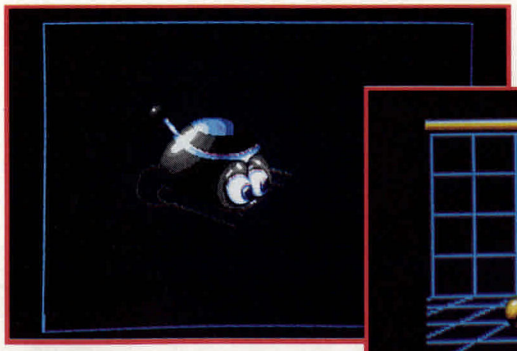
Implemented the statistics information. The game has been counting up this data for some time, it's only now when I get to display this data that I'll find out whether it's really working. I'm only using 16-bit unsigned numbers so I'm hoping that they won't round. I've worked out that you'd have to fire continuously with the fastest reloading weapon for over three hours to get near to worrying the system. Anyone daft enough to try that deserves all the divide-by-zero errors they get. Actually, that particular case will be detected, for anyone worried about the integrity of my code.

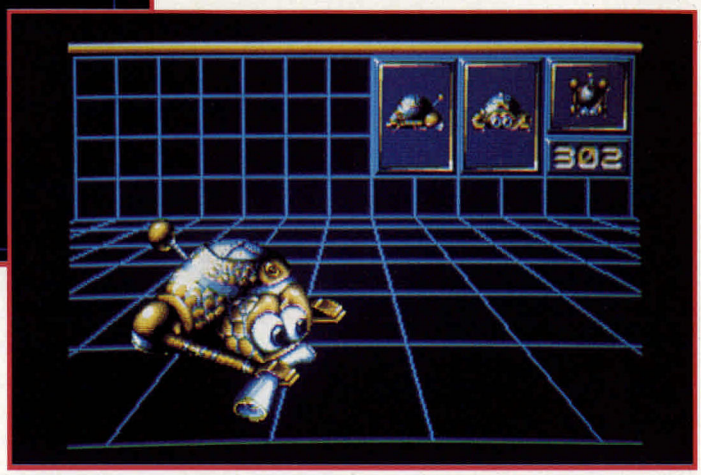
Friday 16th February

Started doing the high score entry display. I've been drooling over Dominic's Anarchy high score table on the old speccy for years, check it out if you've still got one. I've known how to do it for some time but not yet had an excuse to try it out, so here goes, but with a new variation.

Dominic's score table had colour bars passing vertically down above and below the high score entry, the upper bar casting a shadow on the letters. I've changed that slightly by having my colour bars pass down over the letters casting a shadow, then looping back under the letters going upwards, before looping back round at the top and starting again. What do you think, Dom?

Slight error with the statistics keeper as I cleared the first freighter for the first time, and I destroyed one more robot than existed on the ship to begin with. Clever! The Sentinels and other walking robots have a number of different ways of behaving when destroyed. It turned out that one particular method resulted in a Sentinel not ticking himself off as dead when he went, so he got re-incarnated next time I passed through that deck.





Another droid shown in stages of preparation, this time a humble 302. This droid is used to ferry messages to and fro and is essentially a menial.

Monday 19th February

I'm now running desperately short of memory in 512k. I remember discussing with Steve Turner two years ago that if it took us nine months to fill a 64k Commodore 64 with a game, it would take years to write a game that filled 512k. Well, here we are, two years later, seven months of game-writing on and I'm short of memory. Basically what we hadn't considered was the fact that we'd need two 32k screens, a third scrolling buffer of 40k, 68000 machine-code is about double the size code for code, and graphics are about twenty times more memory consuming. I would also venture to say that one's coding has greater depth and is considerably more complex. I'm now getting things to happen that I really want, not having to write something that vaguely does what I want only much simpler. To that end, Paradroid 90 is much more what I would have liked to have coded on the C64 five years ago.

Tuesday 20th February

We've rationalised the transfer game indicator as it was originally supposed to show not only who's winning, but by how much. Unfortunately, the light display was vaguest when the transfer game was nearly drawn, when you most needed to see who was winning. We tried a number of graphical ideas, but the small size and large requirement meant that we didn't come up with anything. We (Michael and I) have decided to use a colour spark, tying in with the spark emitted by the I.D. in transfer mode, to indicate who's winning - no spark means a draw. The top of the indicator also glows to emphasize the colour.

Wednesday 21st February

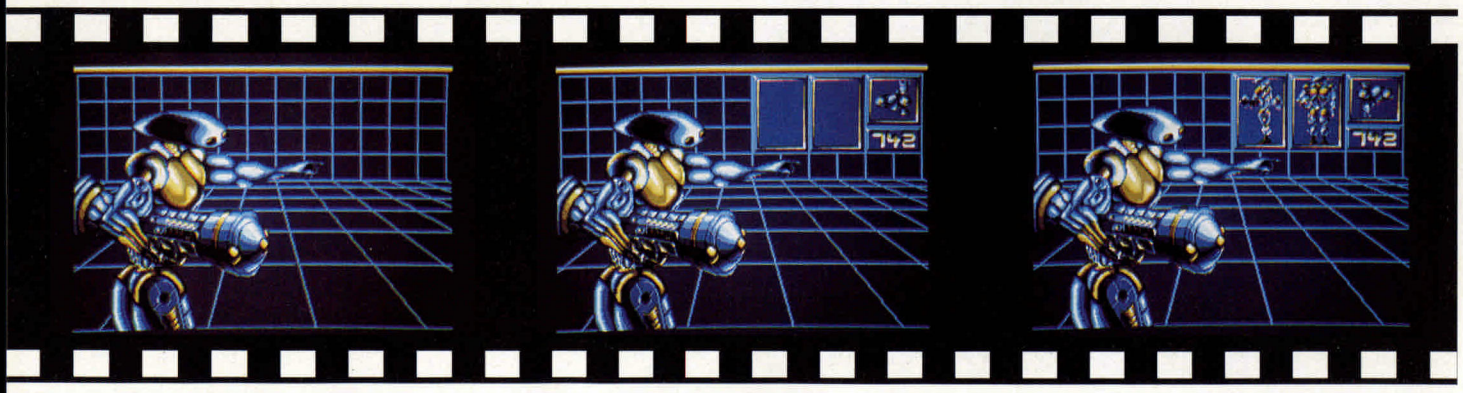
Put in a game options screen, called up from the title screen, allowing selection of one of either joystick, mouse, keyboard controls and the ship to start on. I'll allow players to start on either of the first two ships to start with, up to any of the first four once they have been reached. I would have also saved this number to disk with the high score table but for the fact that in this day and age, viruses abound and I wouldn't recommend anyone to run around with a write-enabled original game disks. Our disk filing system requires that one of our disks be used to write on so supplying a blank disk would be fruitless and wasteful. 'Tis a sad day indeed when a man can't Scribble on his disk in the privacy of his own castle.

Thursday 22nd February

Having a bit of difficulty with the robots patrolling again. Every now and again one of them smashes into a door post instead of going through it. This, it turns out, is due to slight inaccuracies in our vector homing routines. In order to make it easier for these routines, I deliberately move a robot onto the exact position of the patrol point when he gets close to it, reducing inaccuracies. This is, however, an illegal thing to do as sometimes this unofficial move can cause a collision with another robot that cannot be correctly backed out, so two robots get locked together, twisting and turning for evermore. The solution to this was that, instead of physically moving the robot to the exact spot when it got near, I just set up its next move to the exactly the distance required to get it to the exact position. Thus it makes that next move legally itself and can back it out if it causes a collision.

This doesn't actually make a lot of difference to the original problem as the inaccuracies in the system still cause the robots to thump into door posts. They're not even getting the exact horizontal and vertical cases right. What I actually have to do is round the answer I get from the angle calculator and rely on that, since all doors are vertical or horizontal anyway.

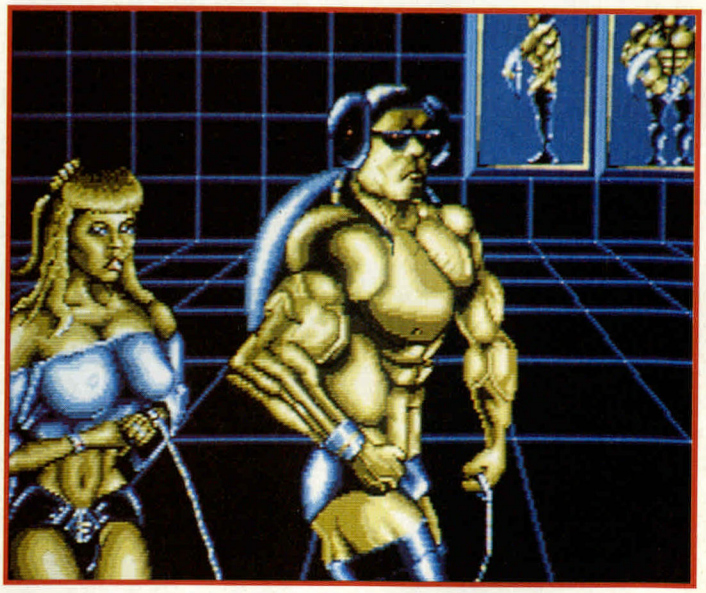
Michael has just about completed work on the first large picture for the droid library. It's the battle droid that was his first sprite for the game. A magnificent example of robotics engineering.



FROM START TO FINISH:

THE DIARY OF A GAME

Some ten months after Andrew Braybrook and the Graftgold team started work on Amiga Paradroid '90, the game is now finally nearing completion. Most of the decks are in, as are the rogue droids, and all that remains is to tie up the loose ends along with a few other problems. So, in this the last part of Andrew's Diary of a Game, read on as he gradually starts to reach the light at the end of the very long tunnel.



Another of the many on-baord rebellious droids, shown here as you prepare to enter the transfer game. I wonder why he needs the... er, well developed accomplice?

Tuesday 6th March.

I have just completed the robot enquiry system and input all the text about the robots. All the texts are drawn from a dictionary of about 200 words so some of the sentences are a little bit odd, having to re-use certain words rather than put in new ones.The dictionary in the game is currently limited to 256 words overall, a number which I thought would be plenty when I first wrote the text printer. I've started populating the 3rd ship with robots and other sundry items like crates and cups of coffee. This involves looking through each deck layout in turn, deciding where the robots should start, and where they can all move.

Wednesday 7th March.

One of the new large decks that I've just done has seven lifts on it, one more than anticipated in the system. A quick fiddle of the way the lifts work alleviates this so that lifts not near the screen get removed from the system and re-allocated when they are nearby. I thought it was already doing that anyway, it is with most other objects. Still, you live and learn. I've also completed the side view of the third ship - huge, it is, with over 130 resident robots.

Thursday 8th March.

The third ship was getting rather large: it takes nearly three hours of computer time to compress the decks, to around 20% of their original size. In order to save a bit of memory, I removed a small deck from the layouts, which ultimately saved about 2K.

Meantime, Michael has been drawing full-screen pictures of some more robots. These are also compressed, by one of Dominic's wee routines this time. After bolting the compressed pictures together, the game loads them all in at once and caches them in slow memory. Each one is then decompressed as required onto the screen. This all worked nearly magnificently except that the black bits on the picture came out bright green. Two errors caused that: I set up the wrong palette and Michael didn't remap the black colour. One all, replay next Wednesday evening, seven-thirty kick off.

Made up the first ever 512K master for testing to see how much memory is actually spare, and to see if it still works. Took it home to test.

Friday 9th March.

A black day indeed. There I was, thinking all was going really well; played a whole ship all the way through, got the end of ship bonuses, it started to load in the second ship... and ground to a halt. I then realised that the load ship screen was using some allocated object blocks and had thus fragmented my dynamic memory. I start off with a vast field of memory and begin to carve it up, always taking the big chunks out first. Trouble was I took a couple of little chunks out, then gave the big lump back and asked for a bigger lump. Although I had enough free memory available, it wasn't all in one place. Result: system panic error. AmigaDos has the same sort of hassle. Quite often DPaint II complains on a 512K Amiga for similar reasons, as would any software using this design philosophy. It means that you can run more software at once if they all share the resources, but if they squeeze too hard, something will break.

Monday 12th March.

Further testing of the 512K version has revealed a major error in the game. It spontaneously combusts sometimes when I enter a lift. The whole game just stops. This I find odd, as I haven't played with the lift routine for some time, and nine times out of ten if a bug appears, it is to be found in the most recently altered code.

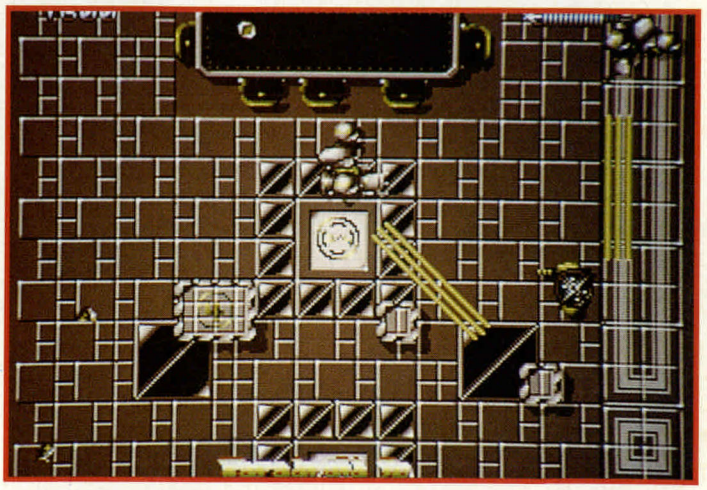
I've decided to ditch the 'deck cleared' message. It uses the font plotter and slows the game down as it appears and is totally superfluous now that the deck lights darken when the last robot is destroyed. I'll keep the 'ship cleared' message, it doesn't happen as often so it doesn't matter if things slow down a little then.

Tuesday 13th March.

Still looking for the bug. It's a really good one: the interrupts all get stitched so the machine is totally dead. Just to continue showing progress I'm continuing to populate another ship, and worrying a lot. It doesn't normally take this long to sort out a bug. I rarely lose a night's sleep these days by having an unsolved bug to think about.

Wednesday 14th March.

Jason was playtesting the game when it decided to leave the



Patrolling one of the detailed competed decks, the wandering Influence Device is attacked by a tri-laser-wielding droid - you have two choices: destroy him with your weaponry or merge with him for a bout of transference.

loading ship music running during the game. Since I go to great lengths to switch this off I was quite surprised. Yet another problem to think about.

The main breakthrough with 'The Bug' came just before lunch time. I was tracing the game through and it went into the sound reset routine, which basically tells the sound manager to be quiet and then waits for its interrupt server to do so. This never happened. Of course, looking through the sound manager code it was obvious that it could not fail to complete its task.

Following rule 15 paragraph B of the bug-fixers guide: "If the code can't fail, it ain't executing it at all" it transpired that not only had the sound manager been shut down, but all interrupts had stopped. Now, I didn't get where I am today by shutting down the interrupts on the 68000, especially as I run my code in user mode most of the time so it's tricky anyway, then it can't be my fault.

This bug only occurs on a 512K machine, not on a 1 Megabyte beast so it's either a corruption or maybe a memory allocation fault. On a whim I decided to trace the memory allocator, especially its strategy when unable to pass the required memory back, i.e. failure on a 512K machine. Suddenly it dawned on me, the memory allocator switches off interrupts while it is playing with doubly-linked lists, and doesn't restart them when it fails. That's one to Dominic and his Kernel.

Spent the rest of the week on cloud 9 having fixed a bug in the Kernel and generally pottering about with ship layouts.

Monday 19th - Wednesday 21st March.

Still quietly decorating and populating ships. The stages in the life of a new freighter are as follows:

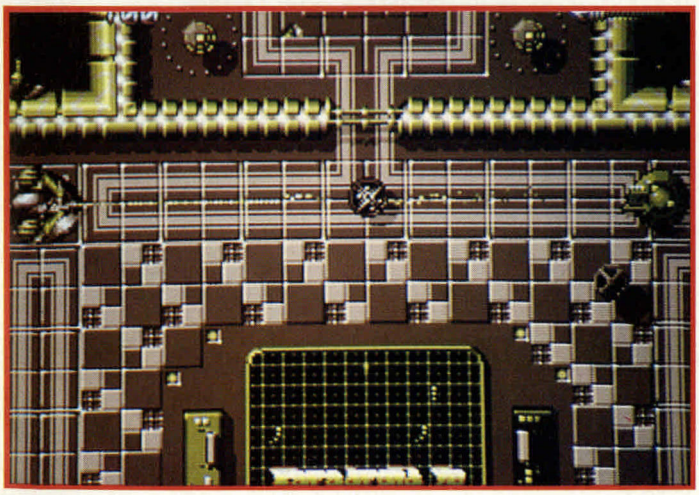
1) design freighter on paper, showing walls, lifts and general ideas for decks.
2) give design to John Lilley to set up on the mapper program.
3) give shell of the ship to Michael to put floor designs and detailed console layouts on ship, the really creative bit.
4) assign doors, lifts, energisers, consoles etc. to map.
5) put in side view of decks and shafts.
6) test integrity of lift shafts and positioning of doors etc.
7) draw patrol routes, with co-ordinates on original paper map.
8) input patrol routes and assign robots to patrol points.
9) assign sundry objects, e.g. crates and alert blocks to decks.
10) thoroughly test layouts.

This whole process takes the best part of two man weeks altogether.

I decided to add a feature I'd been thinking about for some time, to make some robot's heads fly off when destroyed. On a random chance, the head may spin off, casting a shadow on the ground, and fall to the ground before exploding separately. Quite gruesome. All it needs now is a squirt of 20/50 multigrade and...

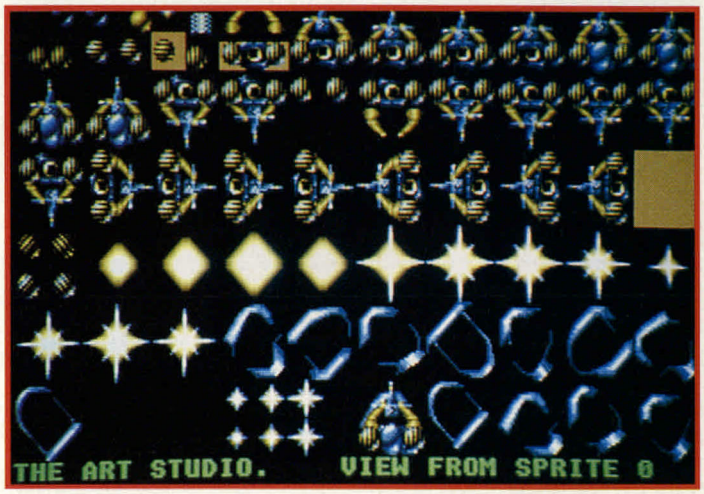
Thursday 22nd - Friday 23rd March.

Found a long-lost bug in the tank turret AMP (Alien Manoeuvre Program, remember them?) that meant that when fired upon it fired three shots back and then stopped. I'd got so used to it doing this that I had forgotten to look for that one. As soon as I fixed it and the tanks fired back continuously the lunch-time testing team of Gary and Jason have complained non-stop for its re-instatement. Now I would never put a bug back into a program, but...they're getting violent, so I'll re-program the tanks to fire three shots back and then stop. OK lads, put the monkey wrench away, now.



Most of the decks are now complete, and all that remains is to pick out the few troublesome bugs that ruin the game's continuity and polish.

A humble ST, complete with Rainbird's trusty OCP Art Studio is used to develop the sprites, but they are then ported over to the Amiga where extra colours from the machine's better pallette are used.



I have a version of the pirate/raiders that are resident, for locking away in the brig and so forth. On meeting them in enclosed corridors I was surprised to be quite successful against them. Then I discovered why. They were not defined up as an enemy, so their bullets did not inherit that enemyness, and they passed right through me - not a scratch. Better fix that one. I'm not going to tell Gary and Jason, though.

Monday 26th March.

Every now and then a couple of wandering robots will seemingly grab hold of each other and dance, ad infinitum. The only way to stop them is to fire at them. I never intended that to happen at all. They have a random waiting period and are then supposed to both turn round and walk away. Fortunately, my inter-robot collision routine passes back some important information, like what direction a robot was hit from. By looking at that, a robot can determine whether it is already moving away from the collision and thus do nothing. Only the robot or robots that caused the collision will turn round. Since doing that, not one pair of robots has been seen doing the waltz at all.

Thursday 29th March - Friday 30th March.

Having got all the plot routines working and clearly defined, that is, I am unlikely to want to change them, I must now persuade the blitter to do the work instead of the 68000. I hate working out minterms, both the methods in the hardware reference manual strike me as being artificial. What I really want to know is: how does the blitter deal with minterms?

Anyway, the shadow plot routine was a doddle, so much so that the first minterm plotted everything back to front, so I had black squares running about with robot-shaped holes cut through them, bleah!

The particle plotter is best done with straight code, you still need to plot about five or six raster lines per blit to make up the overhead of setting up the blitter in the first place. After that it is much faster. I have one plot routine that I haven't used yet, so it's about time to lose that one. Slash, tear, rip.

Monday 2nd April.

Spent the day at the European Trade Fair. Ocean kindly picked up our award for Rainbow Islands being voted the Best Conversion. I'm sure they'll send it along to us soon. Many thanks to all who voted for it.

Thursday 5th - Friday 6th April.

In order to test Paradroid on a stand-alone basis, free of the parallel link from the ST where all the source code is kept and where remote debugging is done from, it is necessary to put the game and all loaded data files onto a bootable Amiga disk. This is not quite as simple as it might seem as both of our Amigas are deficient in some way. Let me elaborate: our 1 Meg Amiga is required for debugging as it keeps extra symbols in memory for the debugger, the game is too big for me to develop on a half Meg machine. However, the RS232 download software won't work with the 1 Meg Amiga, so to make up a disk I have to switch to the other Amiga.

The half Meg Amiga won't boot from any disk with the parallel cable connected, so I have to unplug the lead, boot up, download the disk-maker software, and finally unplug the

Any relevant info on the deck you are about to enter is called up between stages, and details the classes of robots you are likely to encounter.

lead again to persuade the RS232 to work. Marvellous, this technology stuff.

Monday 9th April.

Well, Paradroid '90 is now essentially booting itself off the disk and running in some sort of fashion. Unfortunately, one of the ship data files has mysteriously been cut short so one of the ships won't work; the screen just appears to be a jumbled-up mess. All of the screen drivers that use the blitter are fine, now that I've logically inverted the mask on all of the graphics. The blitter is a strange beast, for the first and last word masks to work correctly, the data mask has to be the opposite way round from the way the 68000 would like it.

Tuesday 10th April.

Remade the boot disk and this time all the files are complete. Must have been cosmic interference from the Graftgold geo-stationary satellite, which for convenience is placed on the end of a big stick attached to the roof. I've been pondering over what to do with the hardware sprites for some time. There aren't really enough of them to handle the multitude of moving objects in the game, most of which go under and over bits of background scenery anyway. I have plumped for pressing them into action as an overlaid score and energy bar. Using them in three-colour mode in conjunction with a few copper instructions to change the colours every line should be enough to get the desired effect. How difficult can it be?

Wednesday 11th April.

Well, I can't find the sprites. I've got all eight of them positioned on the screen, not necessarily quite where the hardware reference manual would have me put them, but they're supposed to be on the visible screen; sprite DMA is switched on and the DMA pointers are reset. Where are they?

I hate looking for things in Dominic's Kernel. He said I'd enjoy trying to get sprites working. Now I've found out why. He's switching off the sprite DMA every frame. No wonder I can't see them. Gary nobbled the Kernel so now my sprites will appear. But no, they're still not there.

Thursday 12th April.

Back on with the miners' helmets and we're off into the dark dank depths of the Kernel once more. What's this? A weeny little copper fragment that switches the sprites into hide-behind-the-playfields mode. I'm pretty suspicious of the palette handling too, I think it's changing the sprites to all black every frame, I just can't prove it. More sticking-plaster and the Kernel is back on its knees again, and yes, my sprites are here at last.

Steve, the boss, has been playing the game for the first time in ages and had a small gripe regarding being able to spot the Influence Device quickly when it beams in. Some sort of attention-drawing display is required. I knocked up some AMPs to cause a whole heap of particles to zoom in towards the player during the beam-in sequence. It looked quite pretty but I calculated that it was running at least three hundred particles to do it, more than a half Meg machine would allow, and the game grinds to a halt while it's doing it. I'll have to calm it down a bit.

Tuesday 17th April.

Visit to Hewson to show them the latest version of the game and tell them how wonderful it all is. Meantime, Michael has been beavering away on the giant pictures of the droids. More than half of them are done now, all the difficult ones, so he says, just the seven-stone weakling ones to go.

Wednesday 18th April.

Having got the game nearly finished, I can now work out what sound effects I want. I know what sort of overall effect I am going for, and I have to scour the game looking for events that will produce a sound. I am directing most in-game sound effects to the left or right-hand speaker, depending on where on screen the effect is coming from, which takes two channels. One of the remaining channels is for background effects, like the deck burbling noises and the alert klaxon, and the other channel is for miscellaneous other effects of importance. Counting up, over sixty effects are required, plus thirty-two instruments for the music, which should keep Jason busy for a while.

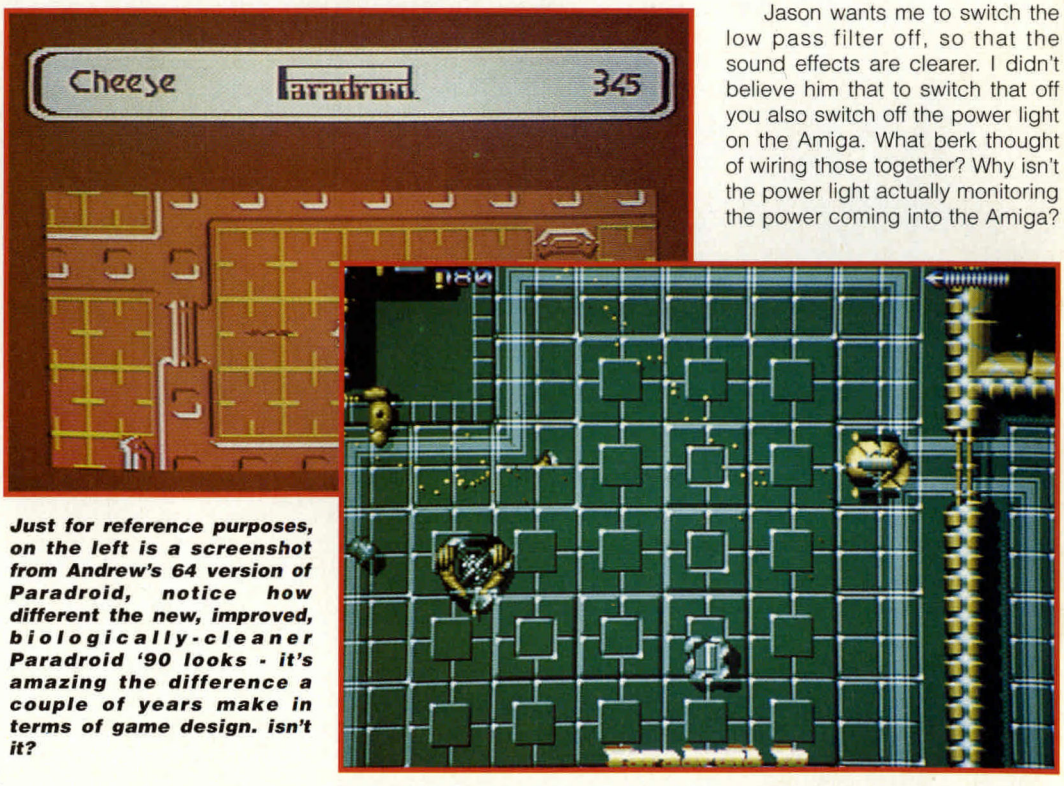
An addendum to the saga of the Amigas not talking to the parallel and serial ports properly is that neither sound sampler that we have will work with either of our Amigas. I'm beginning to suspect that we've broken something in there.

Thursday 19th April.

Jason and Gary each took a sampler home to try on their own Amigas. Both of them worked. Jason has therefore decided to obtain the required samples at home. One that always causes trouble is the white noise used for explosions. In Rainbow Islands we actually sampled a C64 doing white noise! Our friends at Rainbow Arts, Julian and Boris, tell us that they sampled an atomic bomb going off, but we haven't got one of those.

We have an algorithmic sound effects generator, which means that we don't have to just sample everything raw, we can make effects from simple samples just like the C64s SID chip uses simple waveforms. This gives us great flexibilty with the sounds and doesn't burn up lots of valuable chip memory with huge samples. Of course we can just play a raw sample if required and we will be doing so for the main Paradroid 90 theme.

Jason wants me to switch the low pass filter off, so that the sound effects are clearer. I didn't believe him that to switch that off you also switch off the power light on the Amiga. What berk thought of wiring those together? Why isn't the power light actually monitoring the power coming into the Amiga?



Just for reference purposes, on the left is a screenshot from Andrew's 64 version of Paradroid, notice how different the new, improved, biologically-cleaner Paradroid '90 looks - it's amazing the difference a couple of years make in terms of game design. isn't it?

Isn't it rather dangerous that you actually have no real indication that power is gushing through the machine? Come the glorious day, brothers, I know one hardware designer who'll be first up against the wall.

Friday 20th April.

A niggling thought about widening the door overlay has finally got to me. There are two droids which cause minor embarrassment to my system and I when they rotate in a doorway. Their arms or other protuberances (ooo-er) overlap the door edges and show through. By widening the door lintel overlay I can cover their, and my, embarrassment.

Jason has almost finished the sound effects, and so I can switch out the dummy sound routine and run the new one, still with the Rainbow Islands tunes in for the moment. Now that's what I call odd.

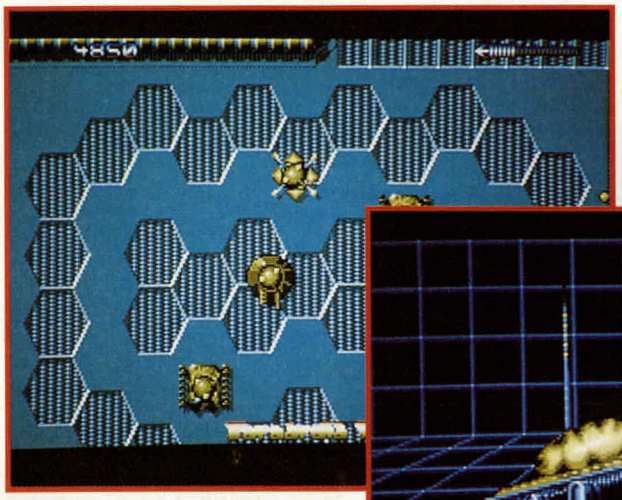
Monday 23rd April.

We can prioritise each sound effect so that the important ones get heard. It's always a problem trying to balance the sound, what with over sixty effects fighting over four channels, not all at the same time, fortunately, but you get the idea. All the weapons firing are fairly low, and the explosion effects are higher so they interrupt the former. More important effects are normally short so they don't just hog the channel.

The big meanie cyborg has its own drone sound at four different volumes so that as it approaches the player it gets louder. It's also stereo panned so you can hear what side of the deck it's on.

Tuesday 24th April.

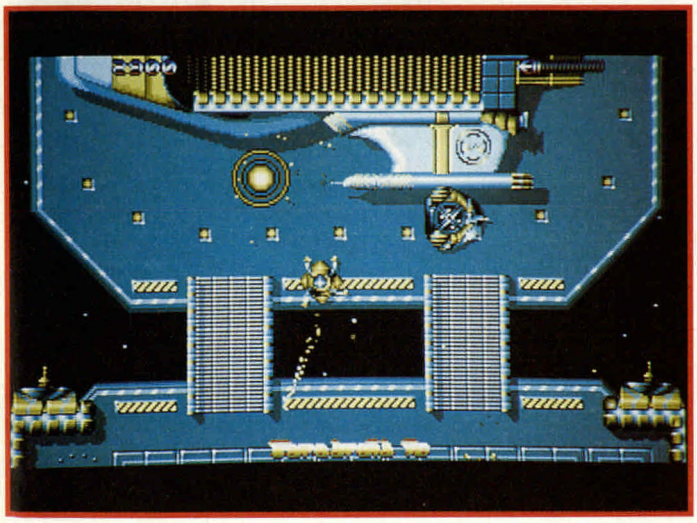
David spotted a minor discrepancy in the system that no-one else had. He wanted to pummel a sentinel on emerging from a lift so he stood facing in the required direction, operated the lift, and was most disgruntled to discover that his facing direction had been randomised. I hadn't bothered to preserve it as I didn't think it would matter. It has taken eight months to spot it so I nearly got away with it. Still, it's in now. Advanced players can now use this facility, and in fact I may position droids so that on later ships it is necessary to do so.



This is a 729 Armoured Tank - one of the deadliest around. Ask yourself this: do you feel lucky, punk?' One of the more powerful rogue droids shown both in detail and in action - well worth beating in the transfer game, if only for the weaponry.

Wednesday 25th April.

Put on the rubber gloves again and dived into the Kernel to fix its sample switching. Now that we have longer samples for the instruments such as drums, we had noticed that the samples were not always switching between instruments and some sound effects. The audio DMA must be switched off for a short, fairly undefined, time to shut it down properly. The hardware reference guide was rather vague about how long exactly, I don't think it really knows. It actually depends on the playback rate, so I've shut the channel down for one call to the sound manager, or one hundredth of a second. Using a software delay loop could be disastrous if someone has a 68030 turbo-nutter CPU.



The robots mill about outside Andrew's set piece - the impressive retro-firing shuttle.

Thursday 26th April.

Jason's still complaining about the sound routine. He'd quite like a total rewrite but he'll settle for his bass drum being a bit louder. Our software enveloping is a little slow to start so the sample is already fading out before the envelope has even got to full attack. Being a full bass drum sample it's entitled to supply its own attack envelope in the sample so the player now wallops the volume up to full from the word go, rather like Jason and Michael's portable CD players.

Friday 27th April.

Jason has had the tunes on SoundTracker for some time, so it didn't take him long to convert them to our format and try them out. So far he has done one three-voice tune for the high score entry, leaving one for the letter entered sound effect, and four four-voice tunes for intermediary stages in the game. Due to a chronic shortage of chip memory I have decided to load in the samples for Jason's super-atmospheric tune separately. On a boring old 512K machine this would mean switching out the currently loaded ship, a real pain every time you return to the titles sequence, so we have an alternate tune for that. For those people with 1 Meg or more, the samples will be loaded once off and kept. Also, all the ships can be in memory at once without reloading. Such are the perks available to those people with upgraded machines.

I've started writing out the game instructions, always difficult when you already know the game inside out, as it's difficult to gauge what people need to know just to get started. I also want to convey the complexity of what is going on in the game, not because you need to know it, but because if you do know it then it makes the game more understandable. Once in possession of all the facts you can plan winning strategies against the game.

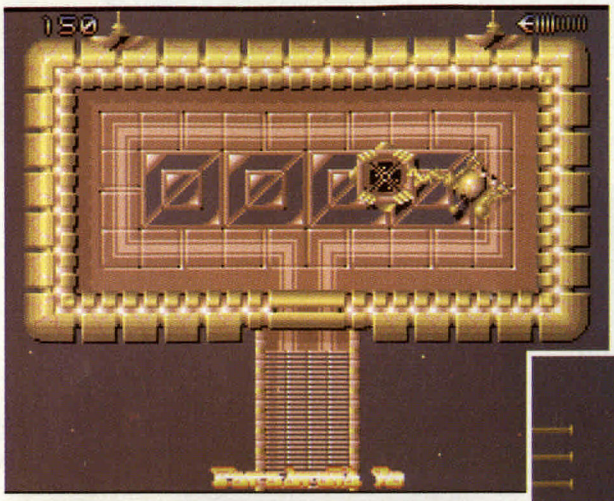
Monday 30th April.

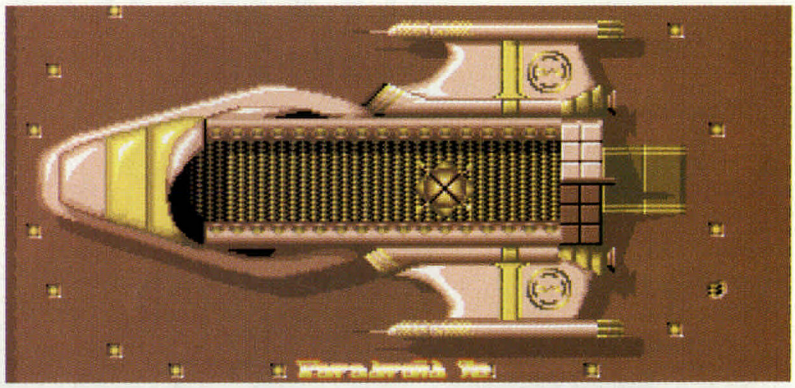
Officially the last day. Michael has delivered the last of the large robot pictures, and Jason is happy with the music. In fact, I have one ship left to populate with robots, which is about two days work if it all goes smoothly. The game is working well, there is just about enough free memory in the machine to cope, despite a last minute scare that the disk system needs about 24K before it'll consider doing anything useful.

The instructions are just about finished and we'll be producing some support material in the form of ship layouts and pictures, and I still have to produce a shop demo version, which may also be suitable for a demo cover disk. I hope that someone somewhere will enjoy the game. I've tried to put the emphasis on gameplay in the true tradition of great games but still do graphic and sonic justice to the Amiga.

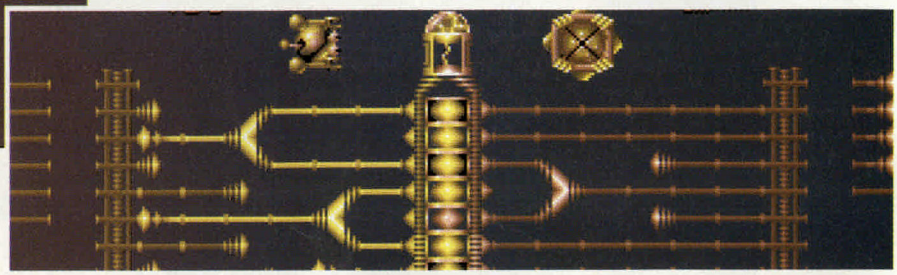
Thanks very much to Andrew Braybrook and all the guys at Graftgold and Hewson for their help in making this diary. Look out for a review of the game, hopefully, next month.

(Right) You come out of the lift and find yourself in the docking bay. Although this may look a very nice place it certainly isn`t, and not only do you have to watch out for the patrolling robots, but you also have to avoid the retros of the parked ships.

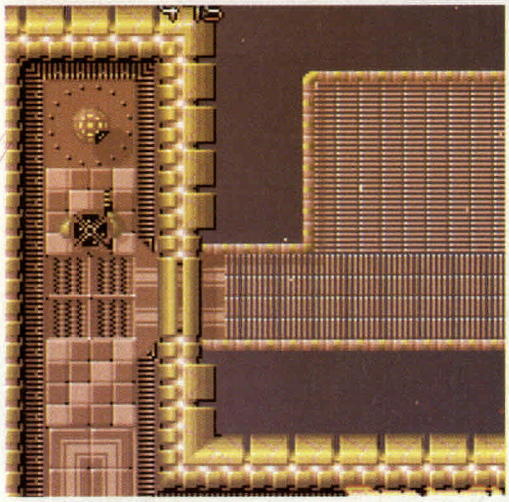




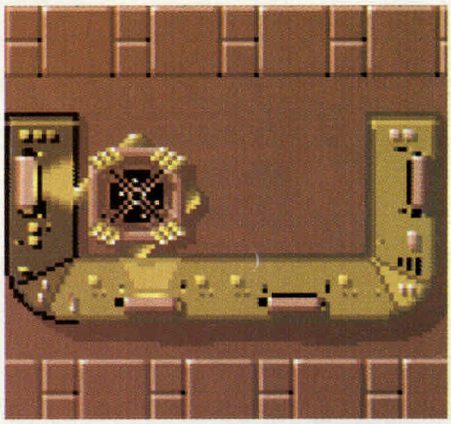
(Left & below) Spotting a rather tasty looking robot you decide to transfer with him and get to have a go with his huge gun. Once connected it`s a fight to the death to try and gain control of the circuit board.



(Below) The computers come in very handy for finding out what`s going on, so regular visits are essential. Once you`ve logged on you can access all sorts of things, but the droid library is by far the best.



(Above) You're bound to run low on energy every now and then, so knowing where the energy pods are is very handy. A robot can only go on for so long and eventually you`ll have to grab another in order to reach your goal.



HEWSON £24.99

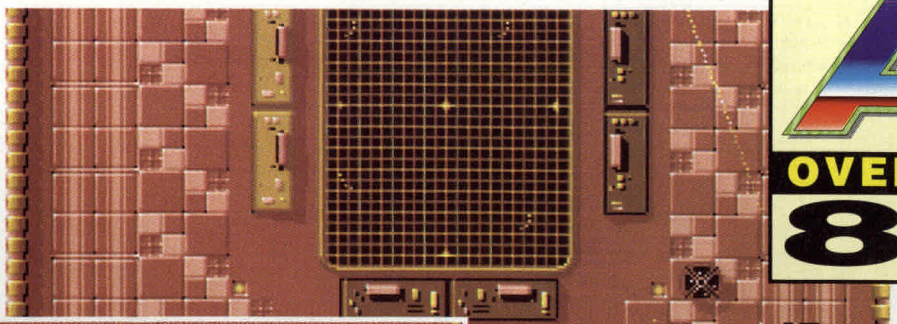
PARADROID 90'

Although not a lot of people know it, Hewson have been around for a very long time. They first made their appearance when the home computer market was making its first boom, and were responsible for such classics as Gribbly`s day out and Paradroid, the original 8-bit version upon which this is based. The game received a whole selection of different accolades, and one magazine actually gave it 100% for presentation (which I think was a bit dodgy). But when it came down to it, Hewson`s games were very good, and way ahead of their time in both capabilities and playability.

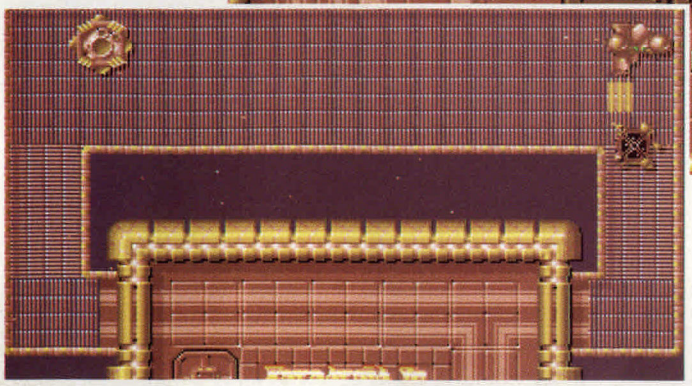
Within the game you control a small droid known as the influence device, whose soul intention is to rid the space craft that you are on of any other droid presence. Disposing of the other droids can be done in two different ways; the first and most straight forward way is to shoot them with your laser until their energy runs out and they explode; Method two is Transference. This is done by over-riding the other droids circuitry and then connecting to him, thus being able to use his weapons and protection. When you begin a transfer sequence, a circuit board will appear on the screen. The goal is to change the components in the centre of the board to your colour by sending power to them with the pulsers at your disposal. Depending on which droid you in control of, you

DOUG

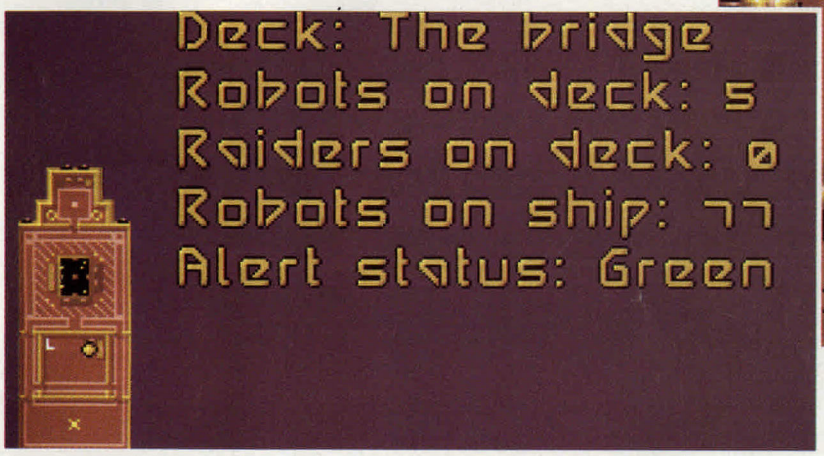
It was five years ago that Paradroid was brought out, and I remember when I saw it for the first time. It was a brilliant game then and even with all of the new ideas that have appeared during those five years Paradroid 90 still plays exactly the same as the original and a lot better than most of todays games. Paradroid 90 is the definitive shoot`em-up and should not be missed by anyone. If you remember the original then you`ll already be getting your money out. If you haven`t then put Paradroid 90 at the top of your shopping list - it`s a must.

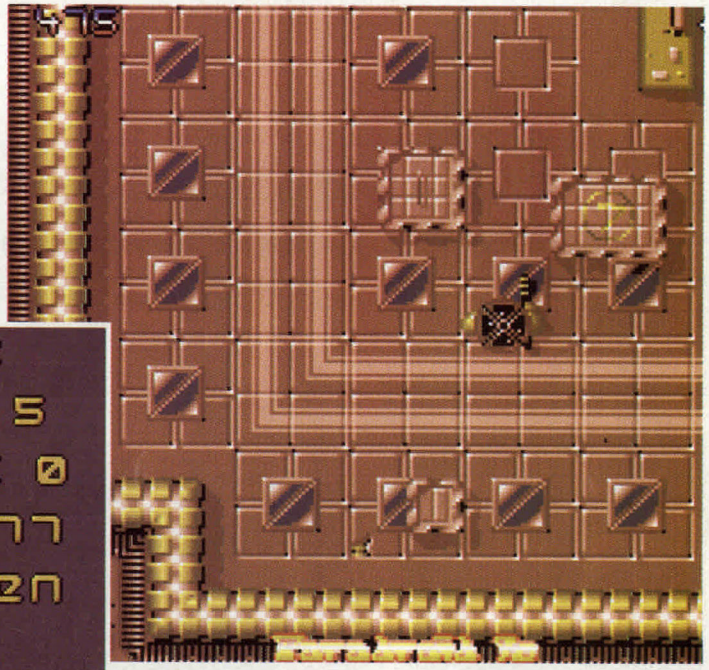


(Right) The most powerful droid is the 999, so gaining control of him is obviously hard. Although this robot is the best, his energy runs out quickly and no sooner have you got him, you'll probably have to look for something else!



(Below) With three lasers blasting away, not many robots are going to stand in your way. However, the gun doesn`t reload too quickly and if you get stuck in a corner you could find yourself in trouble.





(Above) You log on to the computer and take a quick look at the deck to see where the robots might be hiding. The question is, will they still be there by the time you get there.

will have a certain amount of pulsers; the more powerful your robot the more pulsers you will have. You only have a certain amount of time to accomplish this task and failure means the destruction of both the droid you are trying to transfer with and yourself.

The ship comprises of a selection of different decks, that are connected by a network of lifts. Each of the decks will also have a computer console. Stored in each of these computers is information on the alert the ship is presently on; a map of the present deck and whether there is any robots still patrolling the area; and information on each of the robots that you have transfered with.

Something that was not on the original is the Raiders. At any point in the game the Raiders will teleport onto the ship. The ship`s onboard computer will sound a klaxon as the Raiders ship approaches and then they will beam aboard and start wrecking the ship and stealing anything important. These robots should be destroyed as quickly as possible. However, they don`t need to be killed to clear the ship and advance to the next level.

AMIGA ACTION

OVERALL RATING 85%

STEVE

I really enjoyed Paradroid when it was released back in 1985, but this 16-bit version is simply incredible. The graphics and sound are top class and the game plays like a dream. The game is really simple and to look at it you would think that it would become a bit repetetive. However, this is not so and once played you can`t put the game down. Paradroid 90 is a stunning shoot`em-up and rates as one of the best I`ve ever played. I would definitely recommend it as a worthwhile addition to your software collection.

GRAPHICS

Although the graphics still retain the 8-bit design they`ve been upgraded well and look brilliant.

87%

SOUND

Again the sound is based on the C64 version, but they still sound great, and help add atmosphere the game.

81%

ALEX

I remember in the distant past playing a game on the C64 that was far superior to that of any other around at the time - that game was Paradroid. This classic has now made it onto the more powerful Amiga, and what a conversion it is. The game is highly polished, and the action is fast and yet thought must be used. The transfer section makes a change from the exploration screens, and that can only be for the better. All of this adds up to one hell of a game. If you owned this game on the 64, it is well worth getting for the Amiga. I have to admit that the conversion was definitely worth the wait. If you haven't seen Paradroid, buy it now - you won't be disappointed with the result.