ADAPTABILIDADE DE IMAGENS PARA JOGOS MÓVEIS:
O processo de produção e adaptação das imagens em pixel art para as telas de telefones móveis

Allyson Silva André Neves Diego Credidio

Universidade Federal de Pernambuco, Depto. de Design, Brasil

Resumo

As informações compiladas neste artigo relatam o processo geral na produção de imagens em *pixel art*, adotado pela indústria pernambucana de jogos para aparelhos móveis. Engloba desde o momento da concepção artística às revisões finais e posteriores correções dos desenhos em baixa resolução, onde serão destacados problemas de adaptabilidade destes para os diferentes tipos de tela nos celulares.

Palavras-chave: jogo móvel, celular, adaptabilidade, *pixel art*, processo.

1. Introdução

Para um jogo móvel se firmar no mercado em termos de inovação, é preciso atribuí-lo certas qualidades. Dentre estas, talvez a mais relevante corresponda ao nível dos gráficos elaborados. São eles que primeiro atraem a atenção do usuário. Os atuais dispositivos móveis possuem o mesmo poder encontrado nos videogames de 16 e 32 bits da época de ouro da Nintendo e Sega no que diz respeito à configuração das imagens. Não é a toa que alguns *games* considerados sucesso nesses consoles estão sendo adaptados para telas cada vez menores.

Existem duas formas de se trabalhar a produção de imagens nesse tipo de aplicativo: uma é bidimensional, através da *pixel art*, e a outra através da composição de gráficos tridimensionais. O primeiro método, apesar de suas limitações, ainda é o recurso mais utilizado no desenvolvimento de jogos móveis devido a sua viabilidade no maior número possível de aparelhos.

Contudo, entre os aparelhos móveis há uma variedade de dimensões e propriedades gráficas das telas. Quando os gráficos de um *game* são concluídos, eles ainda precisam ser adaptados para atender ao maior número possível de aparelhos. Assim, uma imagem, depois de planejada e produzida, deve ser redimensionada e tratada para ser aplicada em outro grupo de aparelhos através da síntese e arte-finalização. Para se ter uma idéia, alguns elementos perdem tanto sua qualidade visual na redução ou ampliação, que se faz necessário redesenhá-los.

Nosso trabalho permite o entendimento mais claro da adaptabilidade de imagens em aparelhos móveis. A partir desse projeto, pretendemos propor um conjunto de ferramentas computacionais para auxiliar esse processo, tornando-o o mais prático possível.

O artigo está distribuído em 6 seções, sendo a seção 2 um apanhado sobre a divisão em grupos dos aparelhos comuns no mercado pernambucano. Serão consiedrados neste momento, os níveis de resolução do visor e capacidade de memória dos celulares. Na seção 3, ocorrerá uma descrição relativa ao desenvolvimento de *mobile games*, ressaltando o impacto causado pela estimativa de arte no prazo geral desta atividade, onde serão abordadas características importantes sobre *pixel art* e os arquivos pertinentes a este tipo de jogo. Em seguida, é relatado o processo de produção completo das imagens na seção 4 e posterior detalhamento, na seção 5, da etapa de *porting*. Por fim, a seção 6 aponta os pricipais fatores que corroboram com a necessecidade de sanar certos problemas relacionados à adaptação dos elementos gráficos às telas dos telefones, fomentando assim, estudos futuros.

2. Os celulares e os jogos móveis

Quando a inovação se torna um instrumento indispensável para preservar determinado produto num mercado competitivo, principalmente no que se refere a *games*, os celulares surgem como um forte aliado. Isto se deve não só a mobilidade e conectividade que tais dispositivos proporcionam, como também à incrível capacidade de configuração dos personagens e belos cenários em suas reduzidas telas [Madan 2002]. Para compreender melhor este terceiro atributo, é preciso tomar nota de três fatores essenciais: *quantidade de cores, memória heap* e *resolução.*

2.1) Quantidade de cores

Atualmente os celulares possuem plataformas de configuração de imagens que permitem a leitura de diversas cores no visor. Em todos eles, a quantidade de cores processadas pelo aparelho pode chegar a mais de 10 milhões. No entanto, as telas só configuram cerca de 200 mil cores distintas. Em telefones mais avançados, este limite é ultrapassado.

Este recurso se deve aos **pixels**, minúsculos quadrados definidos como a menor unidade da imagem digital. Cada pixel tem uma cor específica de modo a compor, juntamente aos outros pixels adjacentes,

nuances cromáticas bastante interessantes [Davies 2004], quando visualizadas na tela do aparelho. Porém, se tais desenhos forem ampliados, perceber-se-á que são formados por uma colorida malha quadriculada onde cada módulo corresponde a um dos vários pixels da imagem.

2.2) Memória *heap* e resolução

A memória executável de um celular é conhecida tecnicamente como ***heap***. Trata-se da capacidade (em bytes) volátil do celular, responsável por armazenar o conteúdo de uma aplicação: imagens, textos, código e sons. É similar à memória RAM de um computador comum, porém, muito mais limitada.

Já o termo **resolução**, refere-se ao número de detalhes apreendidos por uma imagem [Image 2006], neste caso, a imagem digital em *pixel art*. Isso significa que, quanto menor a resolução de uma imagem, menos nítida está será e vice versa, pois ela captura uma pequena quantidade de pixels por área no visor do celular. Geralmente a resolução é discriminada pela relação entre largura e altura, cujas medidas estão em pixel (px). Por exemplo: a tela de 128x117 possui 128 pixels de largura por 117 de altura, proporcionando uma área de 14.976 pixels, onde poderão ser dispostos elementos configurados dentro dos limites desta resolução.

A memória *heap* guarda uma relação direta com a resolução das imagens usadas no jogo. Imagens em alta resolução são mais pesadas, devido a grande quantidade de bytes necessária para chegar a tal configuração, ocupando mais espaço na memória, o que interfere de forma relevante na adaptação dos mesmos gráficos a serem utilizados em diversos tipos de celular. Por tal motivo, o jogo começa a ser estimado para as telas mais limitadas, facilitando a sua posterior adaptação para as versões menos restritas.

2.3) Divisão dos aparelhos por resolução e memória *heap*

Devido às variações de resolução de tela e capacidade de memória encontradas entre um aparelho e outro, um dos cuidados que os desenvolvedores devem ter ao trabalhar num jogo móvel é com o suporte das imagens. Isto causa impacto significativo na organização visual dos elementos empregados no *game*. Para melhor ilustrar tais diferenças, propõe-se dividir os diversos modelos de celular em quatro grupos distintos, cada um com suas particularidades:

1) **Low end** (128x117, 128x127, 128x128, 130x130): possuem resolução baixa, memória *heap* reduzida e pouco espaço de tela. Neste grupo, a perda de detalhes visuais nas imagens é acentuada.
2) **Middle end** (128x149, 128x160, 132x176): têm a mesma resolução dos aparelhos em *low end*, porém ganham mais espaço de tela referente à altura ou, em alguns casos, nas duas dimensões.
3) **High end** (176x205, 176x208, 176x220, 208x208): com uma resolução maior, apresentam gráficos mais ricos em detalhes.
4) **Super end** (240x294, 240x320): apresentam uma configuração visual superior aos aparelhos em *high end*. Memória *heap,* gráficos e espaço de tela são bem vantajosos.

Todo celular possui uma plataforma responsável por processar os dados dos aplicativos nestes aparelhos. As resoluções citadas entre parênteses nos grupos, correspondem a plataforma JAVA, e serão estes os modelos considerados aqui, por motivos de delimitação do presente estudo e pricimpalmente por ser unânime a sua aplicação nos telefones comercializados em Pernambuco.

3. O desenvolvimento geral de um jogo móvel

No mercado em questão, cada jogo, tratado profissionalmente como "projeto" durante seu processo de produção, precisa de um certo tempo para ser elaborado, variando de 3 a 6 meses de atividade intensa. No entanto, certos *games*, devido a sua complexidade, ultrapassam este prazo para serem concluídos podendo chegar até um ano de produção, desde que haja um consentimento entre desenvolvedores e clientes, decorrente dos motivos relevantes à extensão do tempo.

Estão envolvidas aí todas as áreas de atuação de uma empresa de jogos como gerência, game-design, arte, programação e teste. Trata-se de um trabalho conjunto. Um time onde a função de cada integrante relaciona-se à do outro. Se alguém dessa equipe atrasar sua tarefa, provavelmente outras atividades sofrerão impacto de estimativa. Bethke [2003] chega a mencionar que, hoje em dia, tanto artistas quanto programadores e *game-designers* exercem certa influência sobre a concepção dos gráficos de um jogo. Assim, analisar o processo de arte é de extrema importância devido ao efeito em termos de tempo causado por este trabalho no andamento geral do projeto.

3.1) O tipo de arte aplicado num jogo

A arte de um jogo pode ser trabalhada tanto bi quanto tridimensionalmente. Embora o segundo tipo de configuração seja mais avançado, o bidimensional em *pixel art* é mais viável a níveis mercadológicos, uma vez que a maioria dos aparelhos comercializados, não processam jogos tridimensionais.

A ***pixel art*** é uma modalidade de arte digital onde as imagens são desenhadas minuciosamente pixel a

pixel em softwares apropriados para edição de imagens, como o MS Paint, Adobe Photoshop ou Macromedia Fireworks. Em qualquer um desses programas a ferramenta básica para se desenhar é a mesma, o *lápis, a*lém de outras que aceleram a execução das imagens como a *lata de tinta,* empregada nos preenchimentos de cor, e as *linhas,* usadas para criar contornos mais rapidamente [Pixel 2006].

Trabalhar num jogo móvel em *pixel art*, significa desenhar em baixa resolução, o que exige uma certa habilidade, em termos de síntese, por parte do artista. Para se ter uma idéia, certas imagens chegam a ter uma configuração de 16x16 de resolução com o uso de poucas cores. Os tipos de arquivo trabalhados podem ser em 2 formatos [Arte 2006]:

1) ***Bitmap*** – imagem formada por mapa de bits, interpretada bit a bit e pintada na tela. São exemplos de *bitmap*: JPG, GIF, PNG, BMP e TIFF entre outros. Quanto mais cores forem adicionadas no desenho, mais carregado ele ficará, consequentemente, ocupando mais espaço de *heap*. Por isso os *bitmaps* geralmente são imagens mais pesadas [Raster 2006].

2) **Vetorial** – imagem formada por coordenadas e cálculos matemáticos. Aparecem na tela através do resultado desses cálculos. São imagens normalmente mais "leves" já que a quantidade de informação armazenada tende a ser menor. Exemplos: AI, CDR e SVG. Aqui, os arquivos gerados são utilizados apenas para composição do layout das telas existentes nos jogos, onde são posicionados os elementos já desenhados digitalmente.

Devido às limitações de memória executável já citadas, os *bitmaps* decodificados nos aparelhos são convertidos em PNG, pois o nível de compactação neste formato é maior. Outro motivo para um enfoque maior nesta extensão, é por ela ser o tipo de arquivo mais freqüentemente processado pela plataforma em questão (JAVA).

4. Produção das imagens

Para conceber os PNGs utilizados no decorrer do aplicativo com eficácia , é preciso seguir uma série de atividades que englobam todos os elementos empregados desde a abertura *(splash screem)* às tipografias e gráficos de interface. Tal processo pode ser dividido em 10 etapas:

1. **Pesquisa formal:** baseadas no conceito geral estabelecido para o jogo, pesquisas são feitas em textos, fotos, desenhos e aplicativos similares a fim de se buscar referências relevantes para a concepção dos desenhos que formarão o primeiro conjunto de idéias criadas para o *game*.

Figura 1: Imagem usada como referência para o cenário de um *game.*

2. **Arte conceitual:** com base nas pesquisas feitas a respeito do jogo, os personagens e demais elementos do jogo, como itens e cenários, são definidos através de esboços no papel ou no computador por meio de softwares de edição de imagens.

Figura 2: Arte conceitual para um personagem de jogo móvel.

3. **Storyboard:** engloba a concepção, num papel apropriado, das cenas que o jogo provavelmente terá. Geralmente são definidas as seqüências *in game* (ações dentro do jogo), telas de abertura, menu principal e seus respectivos itens, como por exemplo, JOGAR, RECORDES, AJUDA, SONS, SAIR, etc.

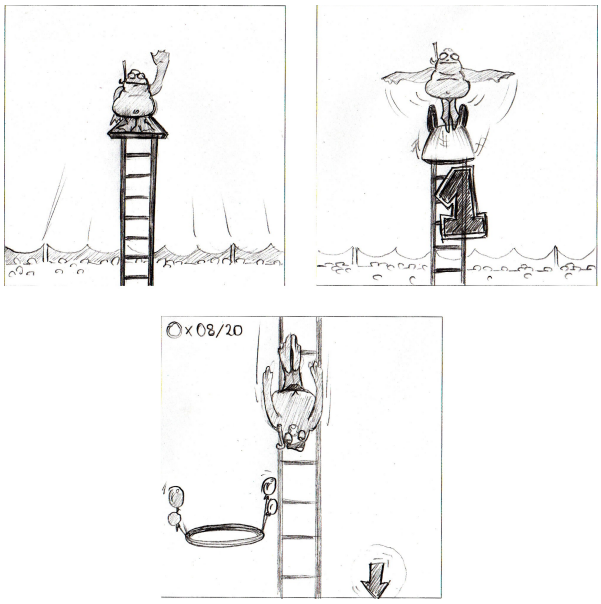

Figura 3: Seqüência de cenas do jogo (*in game*).

Tanto nesta quanto na fase anterior de conceituação artística, o objetivo principal é adquirir mais liberdade quanto a criação das imagens para o jogo. Os *stoyboards* não necessitam ser produzidos unicamente pelos artitas, pois os *game-desginers* também podem participar desta etapa [Bethke, 2003].

4. **Listagem das imagens:** neste momento é estabelecida a quantidade aproximada de imagens que o jogo irá conter, levando em consideração o cálculo de memória *heap* total do celular. São estimados, portanto, não só o tempo gasto para execução dos *sprites* (elementos de jogo) como também o tamanho, em bytes, de cada um deles, visando uma otimização no processamento do *game*.

#	Nome Arquivo	Jar (bytes)	Largura (pixels)	Altura (pixels)	Heap (bytes)
1	frog_dizzy.png	979	112	27	6116
2	frog_greeting.png	1160	93	36	6764
3	frog_jump1.png	954	56	30	3428
4	frog_jump2.png	1500	110	36	7988
5	frog_fall.png	1015	50	35	3568
6	frog_hit_obstacle.png	554	37	25	1918
7	frog_hit_ground1.png	575	40	27	2228
8	frog_hit_ground2.png	274	18	21	824
9	dust.png	281	57	17	2006
10	bizarro_head.png	1000	45	67	6098
11	bizarro_faces.png	484	50	29	2968
12	circus_board.png	326	64	14	1860
13	indication.png	367	28	15	908
14	confette.png	162	28	15	908
15	people.png	562	44	15	1388
16	stair.png	42	42	20	1748
17	jump_plataform.png	427	116	25	5868
20	cont_basin1.png	578	29	31	1866
21	cont_basin2.png	317	14	31	936
22	Cont_bucket.png	359	13	18	536
23	cont_cup.png	175	11	9	266
25	box_tiles.png	161	44	10	948
27	score_target.png	202	16	10	388
28	ring_png	243	74	6	956

Figura 4: Tabela de estimativa em *heap*.

A listagem é uma forma de se calcular a estimativa tanto de tempo quanto de *heap*, porém não é o único caminho a ser seguido. Existem outras formas de se estimar a produção das imagens, como por exemplo, através de softwares específicos controladores desse processo.

5. **Revisão e simplificação:** se numa lista de estimativa o limite de *heap* direcionado a todas as imagens for ultrapassado, ou mesmo for possível uma síntese mais precisa no número de elementos, alguns recursos são utilizados para se reduzir o espaço gasto em memória como:

- *mudança de cores:* numa mesma imagem, as cores podem ser substituídas por outras, acarretando numa economia de memória considerável, pois essa alteração aproveita um mesmo desenho para gerar outros itens distintos. Neste caso, o artista prepara um documento onde ele especifica que cor será trocada e qual a sua subtituta.

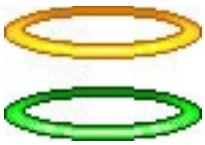

Figura 5: A mudança nas cores forma duas ou mais imagens usando um mesmo desenho.

- *imagens em código:* formas básicas podem ser trocadas por imagens geradas por código como círculos, retângulos e em alguns casos elipses. Tais figuras são incomparavelmente mais leves do que imagens *bitmap*.

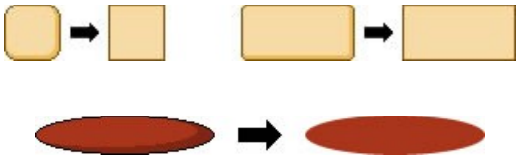

Figura 6: *Bitmaps* geométricos são substituídos por figuras pintadas em código.

- *eliminação de supérfluos:* detalhes como texturas e elementos decorativos nos cenários, são eliminados, sobrando assim espaço suficiente para os elementos essenciais ao *game*.

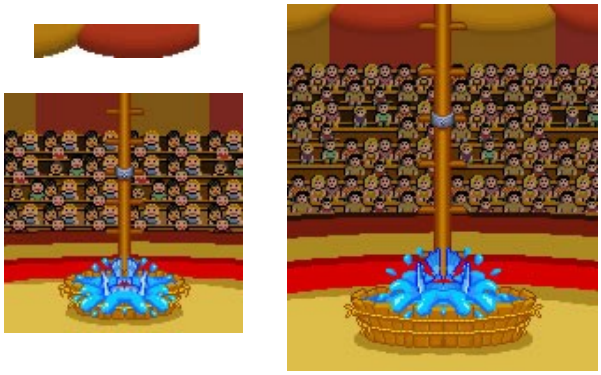

Figura 7: A borda acima é excluída na série menor.

Esse método é empregado mais especificamente nos aparelhos das versões em *low* e *middle end,* devido às suas limitações já citadas. Quanto maior for a resolução do visor, em *high* e *super end*, maior será a necessidade de atribuir elementos gráficos com a finalidade de enriquecer a qualidade visual do jogo. Isso acontece devido à maior visibilidade que tais telas proporcionam.

- *tile-set*: são pequenas imagens que, repetidas lado a lado, compõem outra maior. É uma espécie de quebra-cabeças, normalmente adotado na composição de cenários e seus elementos. O celular carrega apenas uma imagem contendo todas as partes necessárias para formação de outra. Depois, ele apenas pinta esta imagem, já carregada, quantas

vezes forem necessárias para formar um *background*., por exemplo. As possibilidades são inúmeras.

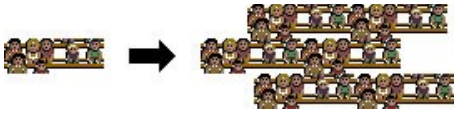

Figura 8: Exemplo de *tile-set* como elemento de cenário.

- *imagens em flip*: são figuras rotacionadas (*flipadas*) vertical ou horizontalmente, às vezes nos 2 sentidos, e em seguida reutilizadas para representar outra imagem. Podem vir acompanhadas de alteração de cor ou não, além de algumas vezes servirem de *tile-set*. Tem-se, portanto, mais um caso em que o mesmo elemento pode gerar outros.

Figura 9: A borda do circo, além de flipada, teve sua paleta de cores alterada.

6. **Produção final:** trata-se da produção final, já em baixa resolução, das imagens listadas para o jogo. Estas são adequadamente desenhadas no computador, em *pixel art*, e divide-se em três etapas:

- *mockups*: "esboços digitais" das imagens finais. Suas características técnicas são definidas (resolução, dimensões e quantidade de cores aproximadas), restando apenas a arte-finalização. São confeccionados a partir da versão *super end* e em seguida, são reduzidos proporcionalmente para as de *high*, *middle* e *low end*. Estes elementos (*em low*) são destinados à equipe de programadores, quando são analisados os primeiros testes de jogabilidade no celular. Enquanto isso, os artistas empenham-se no aprimoramento dos *mockups*.
- *arte-finalização*: finalização dos *mockups*. É o momento para aperfeiçoar as imagens através da suavização nos traços e melhoria na distribuição e quantidade de cores (acréscimo ou redução), procurando uma aproximação ainda maior desta nova configuração com a arte conceitual da mesma. Em seguida, estas imagens substituem os *mockups*.

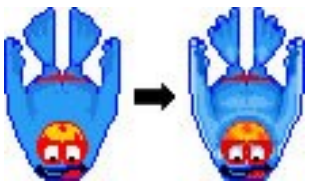

Figura 10: Um *mockup* e sua respectiva arte-finalização.

- *porting*: após serem finalizadas na maior versão, as imagens são portadas paras os outras três resoluções de tela através da redução. Para a versão em *high* e em seguida para *middle* e *low end*. Quando reduzidas, precisam ser novamente arte-finalizadas uma a uma. Este caminho (maior – menor resolução) é recomendável devido a sua precisão. Se o *porting* iniciasse ao contrário (menor – maior resolução), o trabalho seria mais desgastante, pois imagens diminuídas são menos distorcidas que as ampliadas.

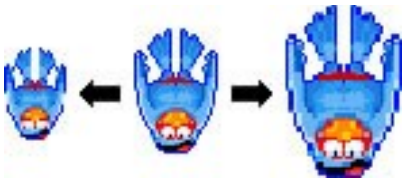

Figura 11: Ao centro, uma imegem em *high end* finalizada e nas extremidades sua redução em *low* e ampliação em *super*.

Quando reduzida, a imagem já é parcialmente sintetizada, enquanto qua na ampliação, alguns pixels são duplicados, dificultando o processo de suavização no contorno e distribuição das cores.

7) **Compactação:** todas as imagens, depois de desenhadas, são exportadas no formato PNG de 8 bits. Em seguida, são rapidamente compactadas em determinados softwares livres como o Gimp a fim de haver uma maior economia referente à memória e processamento do celular. Aqui, os *sprites* têm sua resolução e dimensões inalteradas, enquanto que seu tamanho em bytes é reduzido.

8) **Diagramação:** são especificadas as posições (x,y) dos elementos *in game*, *splash screen* e interface. Uma ferramenta muito interessante têm sido empregada nessa tarefa: um software vetorial que arquiva automaticamente as coordenadas dos principais elementos do jogo permitindo que esses valores sejam interpretados no próprio celular, cabendo ao designer estabelecer apenas os valores que faltam.

Figura 12: Arquivos finais separados e usados em cena de abertura.

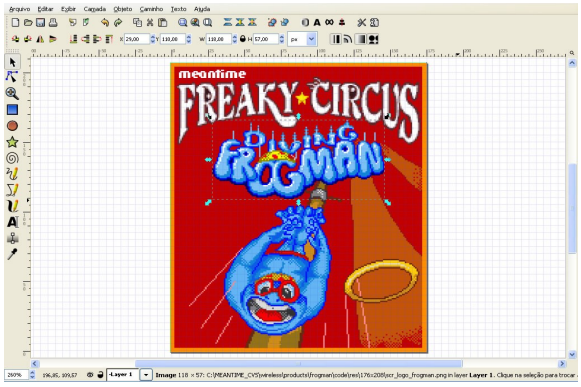

Figura 13: Tela já montada no software vetorial utilizado para diagramar o jogo.

9) **Revisão de arte:** são as avaliações feitas não só pela equipe de testadores como também pelos artistas que conceberam os gráficos do jogo. Vale ressaltar que todas as falhas referentes à arte devem ser percebidas o quanto antes, pois qualquer adiamento desnecessário para a conclusão do projeto pode custar caro.

10) **Correções:** todos os erros encontrados sejam nos *sprites* ou nas coordenadas de diagramação são revistos. Enfim, resolve-se todos os problemas ligados ao visual final do jogo, desde as melhorias nas próprias imagens como na supressão ainda maior do espaço usado em memória pelo jogo no celular.

5. Detalhando a adaptabilidade das imagens

Os elementos definitivos são todos desenhados e arte-finalizdos, neste caso, no Macromedia Fireworks. Dentro do programa, cada *sprite* é trabalhado separadamente numa tela que vai gerar um único arquivo. Em seguida, ele é copiado para outra tela que originará mais um arquivo e assim por diante. Esses arquivos serão reduzidos separada e proporcionalmente para suas respectivas versões de aparelhos.

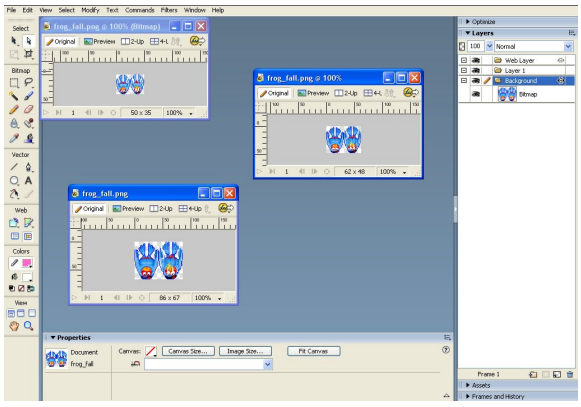

Figura 14: Interface do Macromedia Fireworks.

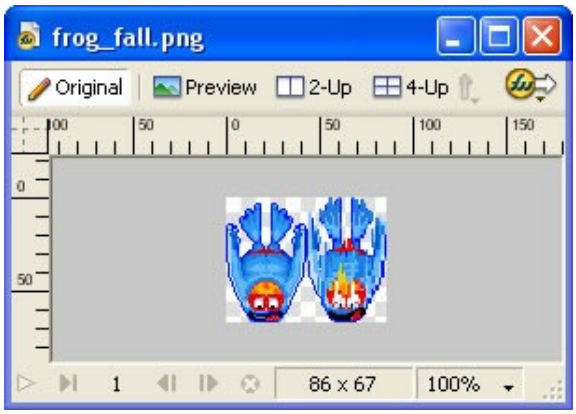

Figura 15: Tela de um desenho arte-finalizado em *super end*.

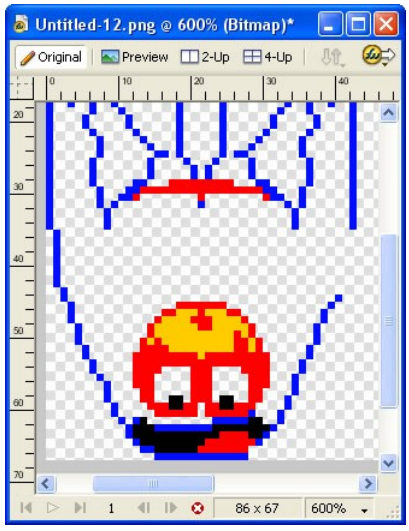

Figura 16: Ampliação do mesmo desenho em fase de produção.

Aqui, as reduções ocorrem no mesmo software onde são desenhadas as imagens em *pixel art*, através de funções específicas.

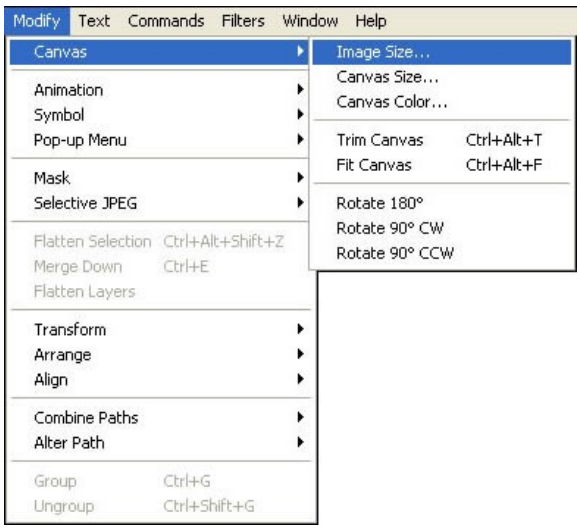

Figura 17: Caminho usado na barra de menus do software para se chegar a caixa de redução das imagens.

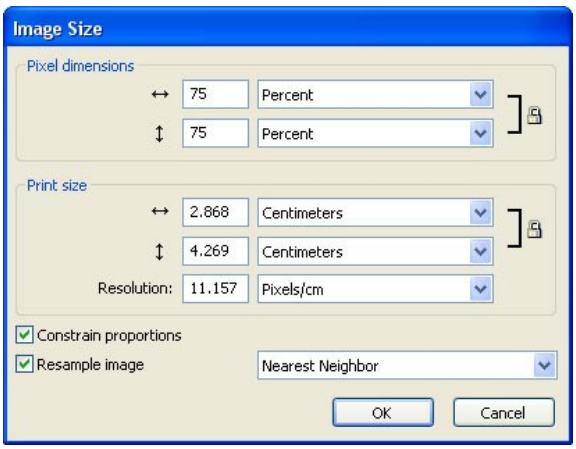

Figura 18: Caixa onde é alterada a proporção da imagem em %.

Na figura 18 , que ilustra a caixa onde é alterado o tamanho da imagem, o acionamento de 3 opções é necessário:

a) seleção *constrain proportions*: matem as proporções do elemento na sua redução;

b) seleção *resample image*: estabelece um posicionamento proporcional da imagem, quando reduzida, relativa à posição desta no tamanho anterior dentro de sua tela;

c) função *nearest neighbor*: permite ao *bitmap* ser comprimido sem perdas ou acréscimos na sua quantidade de cores.

Perecebe-se, na mesma caixa, uma seção chamada *pixel dimension*. É aí onde é colocado o valor, em porcentagem, da redução pela qual a imagem passará.

A seguir, a ordem detalhada na produção das imagens de um jogo móvel. Ela enfatiza a fase de *porting*, uma vez que esta etapa é considerada, pela maioria dos artistas, como a mais desgastante no processo geral.

1) Confecção dos *mockups* em *super end*;
2) redução dos *mockups* de *super* para *high end*. A redução ocorrida é de aproximadamente 30%;
3) redução dos *mockups* de *high* para *middle* e *low end*. As mesmas imagens são utilizadas nessas duas versões menores. Redução de 25%;

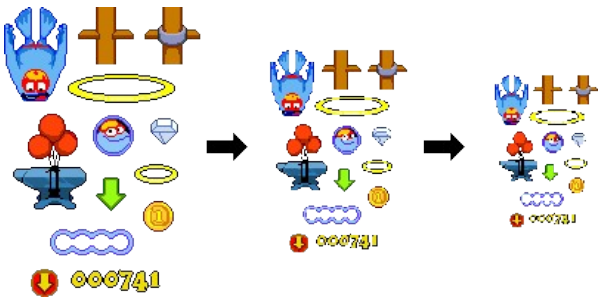

Figura 19: Passos 1, 2 e 3 na confecção dos *mockups*.

4) arte-finalização dos *mockups* em *super end*. Há uma certa liberdade quanto ao limite de cores usadas nesta versão;

Figura 20: Finalização das imagens em *super end*.

5) redução das imagens finais em *super* para *high end*. Redução: 30%;

Figura 21: Redução de *super* para *high end*.

6) finalização das imagens em *high end*. Diminui-se a quantidade de cores a serem usadas nas imagens;

Figura 22: Finalização das imagens em *high end*.

7) redução das images de *high* para *middle/low end*. Redução: 25%;

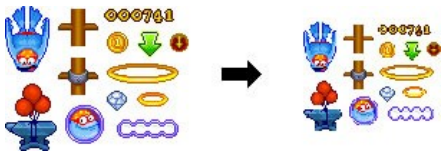

Figura 23: Redução de *high* para *middle/low end*.

8) finalização das imagens em *low/middle end*. Reduz-se ainda mais o número de cores utilizadas;

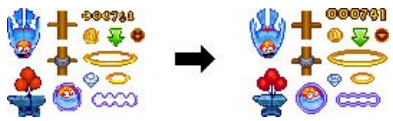

Figura 24: Finalização das imagens em *middle/low end*.

9) compactação das imagens;

10) diagramação de todos os tamanhos de tela, visando um layout comum nas quatro versões.

Na maioria das vezes, esse passo é feito paralelamente ao processo de confecção das imagens.

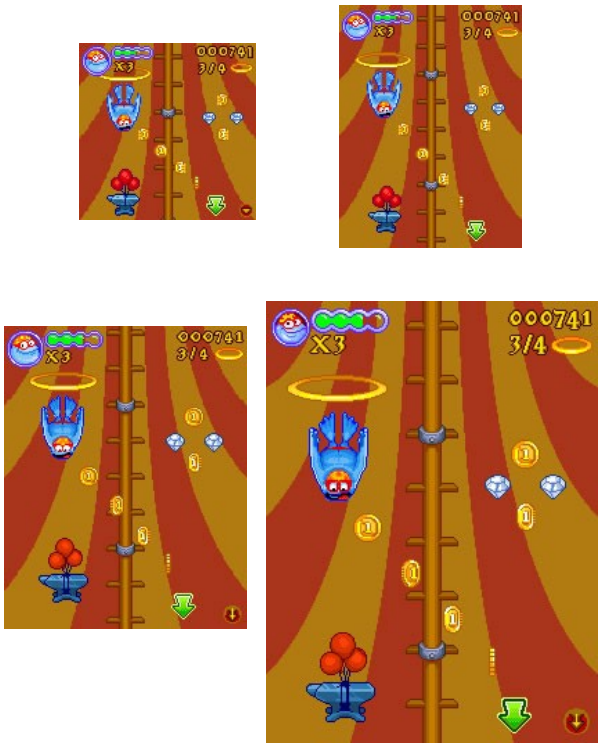

Figura 25: Exemplos de telas montadas nas 4 versões: Acima, *low* (128x128) e *middle* (132x176) *end,* respectivamente. Em seguida, *high* (176x220) e *super end* (240x320).

As porcentagens das reduções (30% e 25%) são usadas na maioria dos jogos. Porém, dependendo do projeto, esses números podem ser alterados para melhor atender às necessidades gráficas do jogo, seja em reduções ou ampliações.

6. Conclusão

De acordo com os dados coletados nesta análise, fica explícito o quanto custa para o trabalho de arte ser finalizado. E quando se trata de *pixel art*, o esforço é ainda maior.

Logo, dentre as etapas descritas no processo, a de produção final, e em especial a fase de *porting*, destacam-se como as mais trabalhosas. Como se não bastasse desenhar todos os elementos do jogo pixel a pixel, é preciso, depois de concluí-los em *super end*, adaptá-los aos outros dois tipos de resolução menor (*high* e *middle/low*), o que requer mais tempo além do prazo definido para a execução das outras atividades.

O objetivo primordial do relato é, enfim, sugerir a criação e implementação de ferramentas ou softwares que permitam um processo de *porting* mais rápido, viabilizando um melhor uso do tempo estabelecido para elaboração das imagens, direcionando uma maior parcela do prazo para as fases de pesquisa, conceituação e arte-finalização das mesmas. O efeito desta otimização seria percebido principalmente na qualidade dos gráficos projetados para o *game.*

Agradecimentos

Gostariamos de agradecer aos profissionais da Meantime Mobile Creations que contribuiram de alguma forma com este artigo, fornecendo não só as informações sobre o processo de produção geral dos jogos criados por esta empresa, como também as imagens de tais produtos, que, aqui, ilustram o corpo do texto.

Referências bibliográficas

ARTE digital. In: Wikipédia, a enciclopedia livre. Disponível em: http://pt.wikipedia.org/wiki/Arte_digital. Acessado em: 26 ago. 2006.

BETHKE, Eric. Game Development and Production. United States of America: Wordware Publishing, Inc, 2003.

DAVIES, Rhys. *The comlpete guide of Isometric Pixel Art*. Disponível em: http://rhysd.syntesis.org/tutorial/chapter1point1.html. Acessado em: 21 ago. 2006.

IMAGE resolution. In: Wikipedia, the free encyclopedia. Disponível em: http://en.wikipedia.org/wiki/Image_resolution. Acessado em: 24 ago. 2006.

MADAN, Hemant. Art of Mobile Gaming. *Forum NOKIA, APAC*. Novembro, 2002.

PIXEL art. In: Wikipedia, the free encyclopedia. Disponível em: http://en.wikipedia.org/wiki/Pixel_art . Acessado em: 23 ago. 2006.

RASTER graphics. In: Wikipedia, the free encyclopedia. Disponível em: http://en.wikipedia.org/wiki/Raster_graphics . Acessado em: 24 ago. 2006.