REPÚBLICA FEDERATIVA DO BRASIL
MINISTÉRIO DA INDÚSTRIA, COMÉRCIO EXTERIOR E SERVIÇOS
INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL

CARTA PATENTE Nº PI 0412013-2

O INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL concede a presente PATENTE DE INVENÇÃO, que outorga ao seu titular a propriedade da invenção caracterizada neste título, em todo o território nacional, garantindo os direitos dela decorrentes, previstos na legislação em vigor.

(21) Número do Depósito: PI 0412013-2

(22) Data do Depósito: 06/05/2004

(43) Data da Publicação do Pedido: 20/01/2005

(51) Classificação Internacional: H04N 19/93; H04N 21/488

(52) Classificação CPC: H04N 19/93,H04N 21/4884

(30) Prioridade Unionista: EP 03014970.2 de 01/07/2003

(54) Título: PROCESSO PARA CODIFICAR POR COMPRIMENTO DE SÉRIE DE UM FLUXO DE DADOS DE MAPA DE BITS

(73) Titular: THOMSON LICENSING, Sociedade Francesa. Endereço: 46, Quai A. Le Gallo, F-92100 Boulogne-Billancourt, FRANÇA(FR)

(72) Inventor: DIRK ADOLPH; JOBST HÖRENTRUP; AXEL KOCHALE; RALF OSTERMANN; HARTMUT PETERS

Prazo de Validade: 10 (dez) anos contados a partir de 03/04/2018, observadas as condições legais

Expedida em: 03/04/2018

Assinado digitalmente por:
Júlio César Castelo Branco Reis Moreira
Diretor de Patente

"PROCESSO PARA CODIFICAR POR COMPRIMENTO DE SÉRIE DE UM FLUXO DE DADOS DE MAPA DE BITS"

CAMPO DA INVENÇÃO

Esta invenção se refere a um processo para codificar um fluxo de dados, particularmente, um fluxo de dados de legendação codificados por mapa de bits.

ANTECEDENTES

Meios de difusão ou exclusivos de leitura contendo dados de vídeo também podem compreender fluxos de dados de subimagens, contendo informações textuais ou gráficas necessárias para proporcionar legendas, glifos ou animação para qualquer fim particular, por exemplo, botões de menu. Uma vez que a exibição dessas informações pode ser habilitada ou desabilitada usualmente, são sobrepostas na imagem de vídeo associada como uma camada adicional, e são implementadas como uma ou mais áreas retangulares chamadas regiões. Essa região tem um conjunto de atributos especificado, como, por exemplo, tamanho da área, posição da área ou cor de fundo. Devido à região sendo sobreposta na imagem de vídeo, o seu fundo é freqüentemente definido como sendo transparente, de modo que a imagem de vídeo pode ser vista, ou camadas de subimagens múltiplas podem ser sobrepostas. Além disso, uma região de legenda pode ser mais ampla do que a imagem associada, de modo que apenas uma parte da região de legenda é visível, e a parte visível da região é deslocada, por exemplo, da direita para a esquerda por toda a área de legenda, que se assemelha como se as legendas estivessem deslocadas pelo visor. Esse processo de produzir legendas baseadas em

pixels é descrito no pedido de patente européia EP 02025474.4 e é chamado recorte.

As legendas foram mencionadas originalmente como um suporte para pessoal sofrendo algum tipo de incapacidade, ou para economizar custos para traduzir um filme em linguagens raramente usadas, e, portanto, para texto de legenda puro, seria suficiente se o fluxo de dados de legendas contivesse, por exemplo, caracteres codificados em ASCII. Mas as legendas contêm hoje também outros elementos, até imagens de alta resolução, glifos ou objetos gráficos animados. O tratamento desses elementos é mais fácil, se o fluxo de produção de legendas for codificado em formato de mapa de bits, com as linhas de uma área e os pixels dentro de uma linha sendo codificados e decodificados sucessivamente. Esse formato contém mais redundância, por exemplo, quando pixels sucessivos têm o mesmo valor de cor. Essa redundância pode ser reduzida por vários processos de codificação, por exemplo, codificação por comprimento de série (RLE). RLE é freqüentemente usada quando seqüências de dados têm o mesmo valor, e as suas idéias básicas são codificar o comprimento da seqüência e o valor separadamente, e codificar as palavras dos códigos mais freqüentes como as mais curtas possíveis.

Particularmente, quando codificando a camada de legendas para vídeo de alta definição (HDTV) de 1.920 x 1.280 pixels, um algoritmo de codificação, que é otimizado para esse fim, é necessário para reduzir a quantidade de dados necessária.

SUMÁRIO DA INVENÇÃO

A finalidade da invenção é descrever um processo para codificação otimizada de camadas de legendas ou subimagens para vídeo de alta resolução, tal como HDTV, sendo representadas como áreas formatadas de mapas de bits, que podem ser muito mais amplas do que o quadro de vídeo visível.

Esse processo é descrito na reivindicação 1. Um aparelho para codificação, que utiliza o processo, é descrito na reivindicação 7. Um aparelho para decodificação, que utiliza o processo, é descrito na reivindicação 8.

De acordo com a invenção, a codificação por comprimento de série (RLE) de quatro estágios é usada para essa finalidade, com uso das palavras de código mais curtas para pixels únicos tendo valores de cores individuais diferentes de transparente, as palavras de código mais curtas secundárias sendo usadas para seqüências mais curtas de pixels transparentes, as palavras de código mais curtas terciárias sendo usadas para seqüências mais longas de pixels transparentes e seqüências mais curtas de pixels de cores iguais distintas de transparente, e as palavras de código mais curtas quaternárias sendo usadas para seqüências mais longas de pixels de cores iguais distintas de transparente. Usualmente, a maior parte dos pixels dentro da camada de legendas é transparente. Diferentemente da RLE convencional, na qual os dados mais freqüentes usam as palavras de código mais curtas, esse processo compreende o uso das palavras de código mais curtas secundárias para seqüências mais curtas da cor mais freqüente, e as palavras de código mais curtas terciá-

rias para as seqüências mais longas da cor mais freqüente e também as seqüências mais curtas de outras cores. As palavras de código mais curtas são reservadas para pixels únicos de outra cor distinta da mais freqüente. Isso é vantajoso, quando os pixels da cor mais freqüente quase sempre aparecem em seqüências, como sendo o caso para pixels transparentes na camada de legendas, enquanto que os pixels únicos de cor individual são mais prováveis de não serem transparentes.

Vantajosamente, um código de acordo com o processo inventivo incorpora apenas poucas palavras de código redundantes, que são definidas como estando entre as palavras de código mais longas. Por exemplo, um único pixel de qualquer cor diferente de transparente é codificado idealmente com uma palavra de código do tipo mais curto, mas uma palavra de código do terceiro tipo mais curto pode ser também usada, com o comprimento da seqüência sendo um. Apesar dessa última possibilidade não ser utilizada usualmente para essa finalidade, essas palavras de código não utilizadas, ou intervalos no espaço da palavra de código, podem ser usadas para transporte de outras informações. Um exemplo é o das informações de fim de linha, que podem ser usadas para ressincronização. De acordo com a invenção, a palavra de código redundante mais curta é usada para codificar essas informações.

Como outra vantagem, o processo descrito reduz a quantidade de dados requeridos, comprimindo, desse modo, o fluxo de dados de legendas, com o fator de compressão dependendo do conteúdo do fluxo de dados. Fatores de compressão altos particulares são obtidos para combinações de dados que

aparecem muito freqüentemente em fluxos de legendação típicos. Essas são seqüências de comprimento mais curto do que, por exemplo, 64 pixels que têm o mesmo valor de cor, mas também seqüências de pixels transparentes tendo qualquer comprimento e pixels únicos tendo valores de cor individuais. O primeiro desses grupos é freqüentemente usado em caracteres ou glifos, o segundo desses grupos é usado antes, entre e depois dos elementos exibidos do fluxo de legendação, e o terceiro desses grupos é usado em imagens, ou áreas com ligeira variação de cor. Uma vez que os pixels transparentes dificilmente aparecem em seqüências muito curtas, por exemplo, de menos de três pixels, é suficiente codificá-los não com as palavras de código mais curtas, mas apenas com as palavras de código mais curtas secundárias.

Simultaneamente, o processo inventivo pode tratar eficientemente seqüências que são mais longas que 1.920 pixels, e, por exemplo, podem ser até de um comprimento de 16.383 pixels, possibilitando, desse modo, áreas de legendação muito ampla.

Além disso, o processo de codificação gera um valor único, representando o fim de uma linha e, portanto, no caso de perda de sincronização, é possível ressincronizar cada linha.

Vantajosamente, o processo inventivo é otimizado para codificar essa combinação de vários recursos sendo típicos para fluxos de legendação.

Portanto, a quantidade de dados necessária para o fluxo de legendação pode ser reduzida, o que provoca uma me-

lhor utilização da largura de banda de transmissão no caso de difusão, ou a uma freqüência de salto de captação reduzida, no caso de meios de armazenamento nos quais um único captador lê múltiplos fluxos de dados, como, por exemplo, em tecnologia de disco Blu-ray (BD). Além disso, quanto melhor o mapa de bits de legendação for comprimido, mais alta a capacidade em termos de taxa de bits que vai ser deixada para fluxos de áudio e vídeo, aumentando a qualidade da imagem ou áudio.

As modalidades vantajosas da invenção são descritas nas reivindicações dependentes, e na descrição e figuras a seguir.

BREVE DESCRIÇÃO DOS DESENHOS

As modalidades exemplificativas da invenção são descritas com referência aos desenhos em anexo, que mostram na:

Figura 1, recorte de uma área de legenda em um quadro de vídeo;

Figura 2, uma seqüência de pixels em uma área de legenda;

Figura 3, uma tabela de codificação para legendação, incluindo texto e gráficos; e

Figura 4, uma tabela com uma sintaxe exemplificativa de um segmento de dados de objeto estendido para o padrão pré-gravado Blu-ray.

DESCRIÇÃO DETALHADA DA INVENÇÃO

Ainda que a legendação em material audiovisual (AV) pré-produzido, para discos de difusão ou de filmes, seja basicamente otimizada para representar informações textu-

ais estáticas simples, por exemplo, Legenda Fechada, Teletexto ou Legenda DVB, o progresso em desenvolvimento multimídia para apresentação e animação de informações textuais e gráficas, adequadas para novos formatos HDTV, requer uma adaptação avançada para codificação de mapa de bits. A Figura 1 mostra um quadro de vídeo TV e uma área de legendas SUB contendo texto e elementos gráficos G, com a área de legendas SUB sendo codificada por mapa de bits. O tamanho da área de legendas SUB pode exceder as dimensões do quadro de vídeo, como, por exemplo, os mapas de bits de formato de Disco Blu-ray Pré-Gravado (BDP) são deixados ter uma dimensão maior do que a do quadro de vídeo. Depois, as linhas são recortadas antes de serem exibidas, isto é, uma parte comparada com a dimensão do quadro respectivo é cortada da linha virtual e exibida, sobrepondo-se à imagem de vídeo. Na Figura 1, a área de legendas SUB de largura B_{SUB} é recortada, de modo que apenas uma parte da largura B_{TV} é visível. Para HDTV padrão, como usado, por exemplo, para BDP, B_{TV} é 1.920 pixels, enquanto que B_{SUB} pode ser muito mais.

Devido à forma retangular da área de legendas SUB, mais pixels nessa área são transparentes. Isso é mostrado em uma escala ampliada mostrada na Figura 2, em uma maneira simplificada, uma vez que usualmente uma linha SL1, SL2 em uma TV de tela HDTV deve ser de vários pixels de largura, para que seja claramente visível. Uma linha é aqui entendida como uma estrutura horizontal. Cada linha de dados de legendas contém usualmente uma ou mais seqüências de pixels de mesma cor. A Figura 2 mostra uma parte de uma linha de le-

genda SL1, contendo as seqüências transparentes PS1, PS5, mas também pixels visíveis únicos PS4, linhas visíveis mais curtas PS2 e linhas visíveis mais longas PS3. A maior parte dos pixels dentro de uma linha é transparente. Esse é o caso entre os caracteres, mas também no início e no final das linhas de legendação. De qualquer maneira, uma vez que as linhas começam e terminam com seções transparentes, cada linha contém uma seção mais transparente que colorida. Mas as seções transparentes PS1, PS5 são usualmente mais longas, enquanto para seqüências de pixels diferentes de transparente, usadas, por exemplo, para caracteres, o caso mais freqüente é um comprimento de seqüência igual ou inferior a 64. Isso pode ser reconhecido de uma estimativa grosseira, considerando que pelo menos 25 caracteres são exibidos simultaneamente, e que o espaço entre os caracteres tem aproximadamente um quarto da largura de um caracter, de modo que um único caracter pode usar mais do que 1.920/25*(8/10) = 62 pixels dentro de uma linha. Freqüentemente, uma linha SL2 contém apenas muito poucos pixels visíveis e, portanto, apenas poucas seqüências transparentes são muito longas.

Um código representando uma modalidade preferida da invenção é listado na Figura 3. É um código de comprimento da série, compreendendo palavras de código de comprimentos variando de 1 byte a 4 bytes, com 8 bits por byte. É capaz de codificar 256 cores diferentes, com uma cor preferida. A cor preferida é neste exemplo "transparente", mas pode ser qualquer outra cor, se adequada. Uma tabela de consulta de cores (CLUT) pode transformar os valores de cores decodi-

ficadas na cor de exibição efetiva. Além disso, as seqüências de pixels de mesma cor podem ser codificadas em duas faixas, com a faixa mais curta sendo de até 63 pixels e a faixa mais longa sendo de até 16.383 pixels.

As palavras de código mais curtas de comprimento de 1 byte são usadas para codificar um único pixel tendo qualquer outra cor individual diferente da cor preferida, que é aqui transparente. O valor da cor CCCCCCCC pode variar de 1 a 255 e pode representar uma cor direta ou indiretamente. Pode, por exemplo, representar uma entrada em uma tabela de consulta de cores (CLUT), que contém o código de cor efetivo. Um dos valores de 8 bits, contendo apenas zeros (00000000), serve como uma seqüência de escape, indicando que os seguintes bits têm que ser considerados como parte da mesma palavra de código. Nesse caso, a árvore de palavras de código tem quatro possíveis ramificações, marcadas pelos dois bits seguintes.

Na primeira ramificação, indicada pelos bits seguintes sendo 00, as palavras de código têm dois bytes, e uma seqüência mais curta de até 63 pixels é codificada tendo a cor preferida, por exemplo, transparente. A única palavra de código inválida nessa ramificação é aquela que compreende apenas 0s, uma vez que 0 não representa qualquer comprimento de seqüência válida. Essa palavra de código "000000000 00000000" pode ser usada para outros fins. De acordo com a invenção, é usada para indicar o fim de uma linha, desde que seja a palavra de código redundante mais curta.

Na segunda ramificação, indicada pelos bits se-

guintes sendo 01_b, a palavra de código compreende outro byte, e os catorze bits L são usados para codificar o comprimento de uma seqüência de pixels da cor preferida, por exemplo, transparente. Desse modo, o comprimento da seqüência pode ser de até 2^{14} - L = 16.383. As palavras de código nas quais os bits L têm um valor abaixo de 64 são redundantes, e podem ser usadas para outros fins.

Na terceira ramificação, indicada pelos bits seguintes sendo 10_b, as palavras de código compreendem um byte adicional, e os seis bits l do segundo byte representam o comprimento de uma seqüência mais curta de até 63 pixels, que apresentam uma cor diferente da preferida. A cor efetiva é representa, direta ou indiretamente, pelo valor CCCCCCCC do terceiro byte. As palavras de código com um comprimento de seqüência LLLLLL abaixo de três são redundantes, uma vez que uma seqüência de um ou dois pixels dessa cor pode ser codificada mais barata usando um byte por pixel, como descrito acima, e um comprimento de seqüência de zero é inválido. Essas palavras de código podem ser usadas para outros fins.

Na quarta ramificação, indicada pelos bits seguintes sendo 11_b, as palavras de código compreendem dois bytes adicionais, em que os seis bits remanescentes do segundo byte e do terceiro byte fornecem o comprimento de uma seqüência mais longa de 64 a 16383 pixels, e o valor de cor CCCCCCCC do quarto byte fornece a cor, direta ou indiretamente, e não sendo a cor preferida. As palavras de código com um comprimento de seqüência abaixo de 64 são redundan-

tes, uma vez que essas seqüências podem ser codificadas mais baratas usando a terceira ramificação. Essas palavras de código podem ser usadas para outros fins.

As palavras de código redundantes mencionadas acima podem ser usadas para estender o código, por exemplo, adicionar somas internas de verificação ou outras informações.

A tabela de codificação por comprimento de série estendida mostrada na Figura 3 e descrita acima proporciona, basicamente, duas vantagens. Primeiro, propicia a codificação mais compacta de fluxos de legendas típicos, incluindo áreas transparentes, pequenos objetos gráficos e texto de legenda normal. Pixels únicos de qualquer cor, como os usados para pequenos gráficos coloridos, são codificados com um único byte. A cor dominante, por exemplo, transparente para legendação BDP, é sempre codificada juntamente com um comprimento de série. Os códigos de comprimentos de série estão disponíveis em dois tamanhos diferentes, ou duas quantidades de pixels. Em uma primeira etapa, comprimentos de série de até 63 pixels estão disponíveis como palavras de código de 2 bytes para a cor dominante, e como palavras de código de 3 bytes para as outras cores. Em uma segunda etapa, os comprimentos de série de até 16.383 pixels estão disponíveis como palavras de código de 3 bytes para a cor dominante, e como palavras de código de 4 bytes para as outras cores. O código da cadeia do fim de pixels, ou código de fim de linha, é um código único de 2 bytes, que pode ser usado para ressincronização. Em segundo lugar, a disponibilidade de seqüências

mais longas para a área de legendação, de até 16.383 pixels por palavra de código, significa uma redução da redundância e, portanto, da quantidade de dados. Isso significa que para aplicações com fluxos de dados separados partilhando um canal, por exemplo, múltiplos fluxos de dados em um meio de armazenamento óptico partilhando o mesmo fonocaptor, partes maiores do fluxo de legendação podem ser carregados com a mesma quantidade de dados, reduzindo, desse modo, a freqüência de acesso para o fluxo de subtítulos.

Outro aspecto da invenção é uma otimização posterior do fluxo de dados para transporte usando os pacotes de transporte, por exemplo, em um fluxo elementar compactado (PES). Devido ao tamanho de arquivo grande dos mapas de bits, a compactação desses dados, por exemplo, em segmentos de dados de objetos (ODS), é um problema. Freqüentemente, o tamanho máximo de ODS é limitado por outros fatores, por exemplo, tamanho do pacote PES. Para ajustar grandes mapas de bits nesses pacotes, seria necessário cortar os mapas de bits em pequenos pedaços de mapas de bits antes da codificação, o que reduz a eficiência da compressão. Para superar essa divisão de mapas de bits, um novo segmento de dados de objeto estendido (ExODS) para BDP ou aplicações comparáveis é descrito, como mostrado na Figura 4. ExODS é uma estrutura de dados representando cada um dos fragmentos nos quais um ODS é cortado para ajustá-lo em uma seqüência de segmentos de tamanho limitado e pacotes PES. O ODS completo pode ser reconstruído por concatenação da seqüência de pedaços individuais de ExODSs consecutivos.

O início e o fim de uma seqüência de ExODS são indicados por sinalizadores separados, first_in_sequence e last_in_sequence. Quando o sinalizador first_in_sequence é 1, uma nova seqüência é iniciada. Um ExODS tendo ajustado o sinalizador first_in_sequence a 1 também indica o tamanho do mapa de bits descomprimido, por conter as suas dimensões object_width e object_height. A vantagem da indicação da dimensão do mapa de bits é o suporte da alocação da memória alvo, antes do início da descompressão. Outra vantagem é que as dimensões do mapa de bits indicadas podem ser também usadas durante a decodificação para uma verificação cruzada das dimensões do mapa de bits. Quando o sinalizador last_in_sequence é ajustado em 1, os últimos ExODS de um ODS completo é indicado. Pode haver ExODS não tendo ajustado nem o sinalizador first_in_sequence como também o last_in_sequence. Há pedaços de ExODS na parte intermediária de uma seqüência. Também, o caso de ter ajustado ambos, o sinalizador first_in_sequence e o sinalizador last_in_sequence, é possível se o ODS puder ser transportado dentro de um único ExODS. Para superar a limitação em tamanho disponível para um único ODS pelo tamanho do pacote PES dentro da legendação, o tipo descrito de ExODS pode ser introduzido como um recipiente para pedaços de um ODS, por exemplo, para compactação de grande ODS para aplicação HDTV. Além dos pedaços ODS, o ExODS também transporta sinalizadores indicando se está transportando o primeiro pedaço, o último pedaço, um pedaço intermediário ou um pedaço apenas completo de uma seqüência ExODS. Além do mais, se o primeiro

pedaço na seqüência do ExODS é transmitido, as dimensões do ODS resultante, isto é, altura e largura do mapa de bits codificado, fica contido no segmento. As dimensões do mapa de bits indicado também podem ser usadas para uma verificação cruzada de decodificação.

O processo inventivo pode ser usado para compressão de fluxos de dados de mapas de bits contendo, por exemplo, dados de texto, de imagens ou gráficos para animação, menus, navegação, logos, anúncio, encaminhamento de mensagens ou outros, em aplicações, tais como, por exemplo, discos Pré-Gravados Blu-Ray (BDP) ou, geralmente, gravações ou difusão de vídeo de alta definição (HDTV).

REIVINDICAÇÕES

1. Processo de codificação por comprimento de série de um fluxo de dados, o fluxo de dados compreendendo dados de legendas ou menu formatados por mapa de bits para apresentação de vídeo em um visor, em que os dados de legendas ou menu incluem gráficos (G) ou texto (TEXT) ou ambos, e onde palavras de código iniciam com uma indicação de seus comprimentos de palavras de código, **CARACTERIZADO** pelo fato de que compreende as etapas de:

definir uma cor preferida (cor 0);

codificar os pixels da cor preferida para as palavras de código primárias com dois ou três bytes, e em que as ditas palavras de código primárias compreendem um valor de comprimento de série, e onde o valor de comprimento de série compreendido nas palavras de código primárias tendo três bytes podem exceder a largura do visor (B_{TV});

codificar os pixels de cor diferente da preferida para as palavras de códigos secundárias com um, três ou quatro bytes, em que as palavras de código secundárias compreendem um valor de cor (C...C), e em que as palavras de código secundárias, tendo três ou quatro bytes, compreendem um valor de comprimento de série, e em que o valor de comprimento da série compreendido nas palavras de código secundárias tendo quatro bytes pode exceder a largura do visor (B_{TV}).

2. Processo, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que as palavras de códigos com três bytes contêm dentro da indicação de seu comprimento da

palavra de código, uma indicação indicando se eles pertencem às ditas palavras de código primária ou secundária.

3. Processo, de acordo com a reivindicação 1 ou 2, **CARACTERIZADO** pelo fato de que os ditos valores de código e a cor preferida são mapeados com uma tabela de consulta para cores do visor.

4. Processo, de acordo com a reivindicação 1, 2 ou 3, **CARACTERIZADO** pelo fato de que a menor palavra de código redundante é usada para sincronização de linha.

5. Processo de decodificação por comprimento de série de um fluxo de dados codificados para uma apresentação de vídeo em um visor, **CARACTERIZADO** pelo fato de que compreende as etapas de:

determinar o primeiro byte de uma palavra de código;

se o dito primeiro byte não tiver um primeiro valor definido (00000000), decodificar o dito primeiro byte para um pixel único tendo cor individual definida pelo valor (C...C) do dito primeiro byte, a cor sendo diferente de uma cor primária definida (cor 0);

se o dito primeiro byte tiver o primeiro valor definido (00000000), determinar os primeiro e segundo bits do byte seguinte sendo o segundo byte;

se os primeiro e segundo bits do segundo byte tiverem um primeiro valor (00), decodificar os bits remanescentes do segundo byte para uma seqüência de pixels da cor primária definida, em que os ditos bits remanescentes do segundo byte definem o comprimento da seqüência;

se os primeiro e segundo bits do segundo byte tiverem um segundo valor (01), decodificar os ditos bits remanescentes do segundo byte, juntamente com o terceiro byte seguinte para uma seqüência de pixels da cor primária definida, em que os ditos bits remanescentes do segundo byte e do dito terceiro byte definem o comprimento da seqüência, e onde o comprimento da seqüência pode exceder a largura do visor (B_{TV});

se os primeiro e segundo bits do segundo byte tiverem um terceiro valor (10), decodificar os ditos bits remanescentes do segundo byte, juntamente com o terceiro byte a uma seqüência de pixels, em que os ditos bits remanescentes do segundo byte definem o comprimento da seqüência e o terceiro byte define a cor dos pixels; e

se os primeiro e segundo bits do segundo byte têm um quarto valor (11), decodificar os ditos bits remanescentes do segundo byte, juntamente com o terceiro e o quarto byte seguinte, em que os ditos bits remanescentes do segundo byte e do terceiro byte definem o comprimento da seqüência e o quarto byte define a cor do pixel, e em que o dito comprimento da seqüência pode exceder a largura do visor (B_{TV}).

6. Processo, de acordo com a reivindicação 5, **CARACTERIZADO** pelo fato de que a dita definição de uma cor do pixel do primeiro, terceiro ou quarto byte do dito primeiro valor compreende o uso de uma tabela de consulta.

7. Processo, de acordo com reivindicação 5 ou 6, **CARACTERIZADO** pelo fato de que o fluxo de dados codificados para uma apresentação de vídeo é uma camada separada, sobre-

pondo-se aos outros dados de vídeo no visor, adicionalmente compreendendo a etapa de selecionar uma parte da dita camada separada para exibição.

8. Aparelho para codificação por comprimento de série de um fluxo de dados compreendendo dados de legendas ou menu formatados por mapa de bits, para uma apresentação visual em um visor, em que os dados de legendas ou menu incluem gráficos (G) ou texto (TEXT), ou ambos, e em que as palavras de código iniciam com uma indicação do comprimento da palavra de código do mesmo, **CARACTERIZADO** pelo fato de que compreende:

um dispositivo para definir uma primeira cor (cor 0);

um dispcsitivo para codificar pixels da primeira cor em primeiras palavras de código com dois ou três bytes, em que as ditas primeiras palavras de código compreendem um valor de comprimento de série, e em que o valor de comprimento de série compreendido nas primeiras palavras de código tendo três bytes pode exceder a largura do visor (B_{TV});

um dispositivo para codificar pixels de cor diferente da primeira cor para segundas palavras de código com um, três ou quatro bytes, em que as segundas palavras de código compreendem um valor de código (C...C), e em que as segundas palavras de código tendo três ou quatro bytes compreendem um valor de comprimento de série, e em que o valor de comprimento de série compreendido nas segundas palavras de código tendo quatro bytes pode exceder a largura do visor (B_{TV}).

9. Aparelho para decodificação por comprimento de série de um fluxo de dados codificado compreendendo dados de legendas ou menu formatados por mapa de bits para aplicação de vídeo, **CARACTERIZADO** pelo fato de que compreende:

um dispositivo para determinar o comprimento da palavra de código, em que o primeiro byte de uma palavra de código é avaliado, e em que se o dito primeiro byte tiver outro valor diferente de um primeiro valor definido (00000000), então o dito comprimento da palavra de código é determinado para ser um byte, e de outro modo os primeiro e segundo bits do segundo byte seguinte são avaliados, e dependendo dos ditos primeiro e segundo bits, o comprimento da palavra de código é determinado como sendo dois, três ou quatro bytes, respectivamente;

um dispositivo para decodificar palavras de código determinadas como sendo de comprimento de um byte para pixels únicos tendo uma cor definida pelo dito um byte, a cor sendo diferente de uma primeira cor definida (cor 0);

um dispositivo para decodificar palavras de código determinadas como sendo de comprimento de dois bytes para seqüências de pixels da primeira cor definida, em que o comprimento da seqüência é definido pelos bits remanescentes do segundo byte da palavra de código;

um dispositivo para decodificar as palavras de código determinadas como sendo de comprimento de três bytes ou para seqüências de pixels da primeira cor definida, em que o comprimento da seqüência pode exceder a largura do visor de vídeo (B_{TV}) e é definida pelo terceiro byte e bits remanes-

centes do segundo byte, ou para seqüências de pixels de mesma cor diferente da primeira cor definida, em que o comprimento da seqüência é definido pelos bits remanescentes do segundo byte; e

um dispositivo para decodificar palavras de código determinadas como sendo de comprimento de quatro bytes para seqüências de pixels de mesma cor diferente da primeira cor definida, em que o comprimento da seqüência pode exceder a largura do visor de vídeo.

10. Aparelho, de acordo com a reivindicação 9, **CARACTERIZADO** pelo fato de que ainda compreende um dispositivo para decodificar palavras de código que possuem dois bytes, e incluem um comprimento de seqüência de zero como uma palavra de código de sincronização, para fim de linha.

11. Aparelho, de acordo com qualquer uma das reivindicações de 8 a 10, **CARACTERIZADO** pelo fato de que compreende ainda um dispositivo de tabela de consulta, para mapear entre os ditos valores de cores, incluindo a dita primeira cor definida, e as cores do visor.

12. Aparelho, de acordo com qualquer uma das reivindicações de 8 a 11, **CARACTERIZADO** pelo fato de que o dito fluxo de dados codificados é distribuído a pacotes de transporte múltiplos.

13. Disco ótico contendo um fluxo de dados codificado por comprimento de série, o fluxo de dados compreendendo dados de legendas ou menu formatados por mapa de bits para apresentação de vídeo em um visor, em que os dados de legendas ou menu incluem gráficos (G) ou texto (TEXT), ou am-

bos, **CARACTERIZADO** pelo fato de que o fluxo de dados codificados por comprimento de série compreende:

primeiro dado para definir uma cor preferida;

primeiras palavras de código com dois ou três bytes para codificar pixels de uma cor preferida, em que as ditas primeiras palavras de código compreendem um valor de comprimento de série e em que o valor de comprimento de série compreendido em primeiras palavras de código tendo três bytes pode exceder a largura do visor (B_{TV});

segundas palavras de código com um, três ou quatro bytes para codificar pixels de uma cor diferente da cor preferida, em que as ditas segundas palavras de códigos compreendem um valor de cor (C...C), e em que as segundas palavras de código possuindo três ou quatro bytes compreendem um valor de comprimento de série, e em que o valor de comprimento de série compreendido em segundas palavras de código tendo quatro bytes pode exceder a largura do visor (B_{TV}).

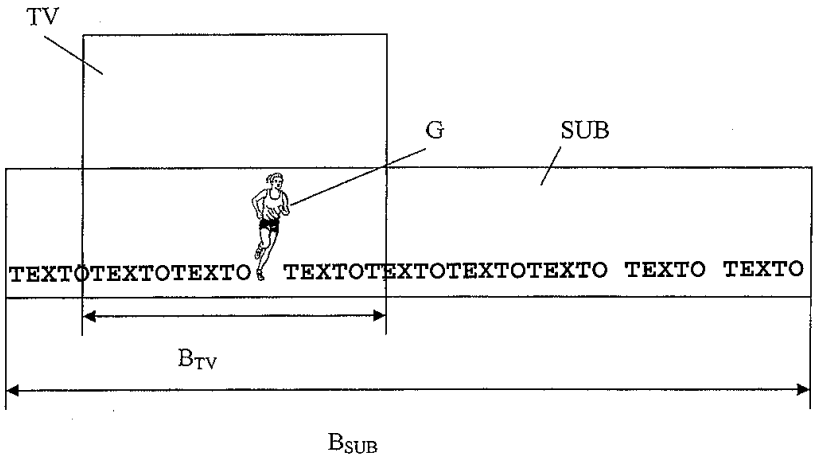
TV
G
SUB
TEXTOTEXTOTEXTO TEXTOTEXTOTEXTOTEXTO TEXTO TEXTO
B_{TV}
B_{SUB}

Fig.1

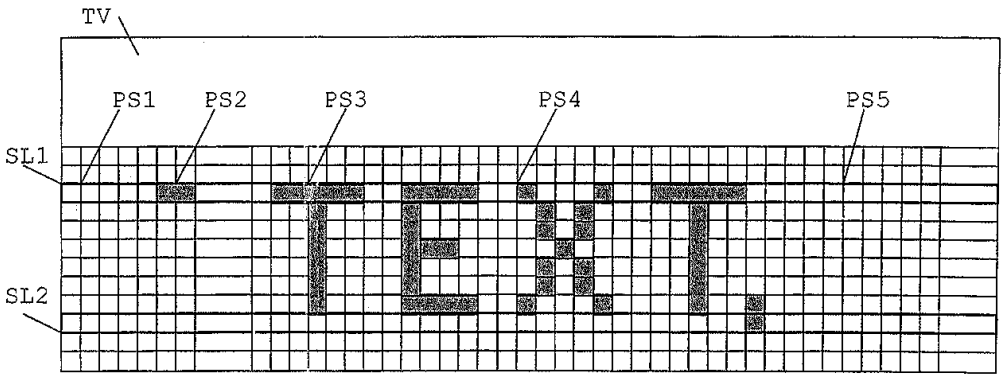
TV
PS1
PS2
PS3
PS4
PS5
SL1
SL2

Fig.2

Código	Significado
CCCCCCCC	**Um pixel em cor C** (1 ≤ C ≤ 255)
00000000 00LLLLLL	**L pixels em cor 0** (1 ≤ L ≤ 63)
00000000 01LLLLLL LLLLLLLL	**L pixels em cor 0** (64 ≤ L ≤ 16383)
00000000 10LLLLLL CCCCCCCC	**L pixel em cor C** (3 ≤ L ≤ 63, 1 ≤ C ≤ 255)
00000000 11LLLLLL LLLLLLLL CCCCCCCC	**L pixel em cor C** (64 ≤ L ≤ 16383, 1 ≤ C ≤ 255)
00000000 00000000	**Fim de linha**
Possíveis extensões	
00000000 10000000 X 00000000 10000001 X 00000000 10000010 X 00000000 01000000 00000000 X 00000000 01000000 … 00000000 01000000 00011111 X	

Fig. 3

Sintaxe	Tamanho [bit]	Tipo
extended_object_data_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
first_in_sequence	1	bslbf
last_in_sequence	1	bslbf
reserved	6	bslbf
if (first_in_sequence == '1') {		
object_width	16	uimsbf
object_height	16	uimsbf
}		
object_data_piece()		
}		

Fig. 4