ALEX REIS DOS SANTOS

MEDIÇÃO DE AUDIÊNCIA DE TELEVISÃO EM TEMPO REAL PELO RECONHECIMENTO DE LOGOS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

São Paulo
2007

ALEX REIS DOS SANTOS

MEDIÇÃO DE AUDIÊNCIA DE TELEVISÃO EM TEMPO REAL PELO RECONHECIMENTO DE LOGOS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

Área de Concentração:

Sistemas Eletrônicos

Orientador:

Prof. Dr. Hae Yong Kim

São Paulo
2007

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, de dezembro de 2007.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Santos, Alex Reis dos
Medição de audiência em tempo real pelo reconhecimento de logos / A.R. dos Santos. -- ed.rev. -- São Paulo, 2007.
110 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1.Medição de audiência de televisão 2.Logo 3.DSP I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II.t.

À Marcia, minha esposa, com amor e gratidão pela sua compreensão e incansável apoio ao longo do período de elaboração deste trabalho.

AGRADECIMENTOS

Gostaria de expressar os meus agradecimentos, de forma geral, aos amigos e colegas pelo apoio e incentivo para a realização deste trabalho e em particular:

- Aos meus pais, pelo apoio ao longo da elaboração do trabalho;

- Ao Prof. Dr. Hae Yong Kim pela orientação, pelas discussões e pelo incentivo sem o qual esta dissertação não seria possível;

Ao Diretor Luiz Motta e Gerente Márcio Alves de Engenharia do grupo Ibope pelo incentivo a pesquisa durante todo o período.

RESUMO

Os logos de televisão são uma das mais importantes estratégias criadas e registradas pelas emissoras de televisão para proteger o conteúdo produzido e distribuído por elas. Cada logo é único, gerando robustez e segurança ao processo de medição de audiência de televisão. Estes logos podem ser considerados uma marca d'água, que em alguns casos identificam não só a emissora, mas também o tipo de conteúdo que está sendo veiculado. Por exemplo, alguns canais mudam o logo de semitransparente para opaco quando há uma transmissão ao vivo. No reconhecimento de logos em tempo real utilizando-se sistemas embarcados, torna-se necessário o uso de técnicas que reduzam o processamento e o armazenamento de dados. Neste trabalho estudamos os principais métodos envolvidos em reconhecimento de imagens encontrados na literatura. Verificamos o uso do logo em outras aplicações, e desenvolvemos uma solução viável, técnica e economicamente. Aplicamos a técnica proposta em dados previamente gravados, e também em situações em tempo real, onde não se tem o controle do tipo de vídeo ao qual será veiculado. Avaliamos o novo método proposto e sua melhoria ao longo do processo, demonstrando a sua viabilidade. Apresentamos resultados comparativos entre o primeiro paper publicado e os novos métodos.

ABSTRACT

Television logo is one of the most important strategies used by the broadcasting companies to claim and protect the contents created and broadcasted by them. Each logo is unique, yielding robustness and security to the measurement of television audience. These logos can be considered as visible watermarks, and in some cases can identify the kind of broadcasted content, besides identifying the broadcasting company. For instance, some broadcasting company changes the logo's type from semitransparent to opaque to identify a live broadcast. For real time logo recognition using embedded systems, it is necessary to reduce the amount of processing and the memory storage. In this work, we describe the methods involved in logo recognition found in the literature. We verify the use of logos in other applications, and developed a technically and economically viable solution to recognize television logos in real time.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

O reconhecimento de logos em tempo real é uma tarefa com grande aplicação em áreas de monitoração, veiculação, publicidade, etc. Uma aplicação para o reconhecimento de logos é a detecção de comerciais televisivos, [(Albiol, 2004), (Yel, 2005)], onde a falta de um logo indica que está sendo veiculado um conteúdo não produzido pela emissora, no caso um comercial televisivo. (Meisinger, 2005) propôs um método para a remoção automática de logos de um pacote de vídeo para que seja possível o reuso de um conteúdo televisivo por uma outra emissora. Na mesma linha de raciocínio, (Yan, 2002) propôs um método para remover logos e gerar um maior prazer ao telespectador, por acreditarem que os logos atrapalham a imagem. (Kovár, 2002) implementou um método para a detecção de logomarcas de publicidade em conteúdos esportivos veiculados pela televisão. (Seeber, 2007) trata o reconhecimento de logos em tempo real, como a detecção de uma marca registrada, não se restringindo a uma determinada aplicação, ficando a caráter da aplicação utilizar em uma aplicação específica.

A detecção de logos exige um alto desempenho do hardware, pois do ponto de vista computacional, grandes matrizes são trabalhadas. Estes logos podem ser considerados como marcas d'água, pois, embora não exista uma norma técnica que defina a sua produção, segundo as normas da ABNT, os logos são registrados para deter direitos da marca às emissoras. Além de identificar a emissora responsável pelo conteúdo veiculado no momento, em alguns casos, os logos indicam o tipo de transmissão que está ocorrendo. Por exemplo, na figura 1.1, a imagem (c) apresenta o logo da emissora Record quando a mesma está transmitindo um conteúdo previamente gravado. Na mesma figura agora olhando a imagem (d), a emissora Record mudou o seu logo para informar que há um conteúdo sendo transmitido ao vivo. Esta prática é comum em emissoras brasileiras e também acontece em emissoras estrangeiras. Pensando em uma aplicação para o mercado brasileiro, o sistema PAL-M de recepção do sistema analógico de televisão foi pré-requisito no desenvolvimento.

Os logos de televisão são marcas registradas pelas emissoras que não detém uma norma ABNT para sua construção, ficando a caráter publicitário e de aceitação pública, a sua

estratégia de criação. Hoje em dia é comum aliar vários detalhes de imagem e animações ao logo para se demonstrar inovação e capacidade tecnológica.

Existem três tipos de logos: opaco, semitransparente e animado. Exemplos são dados na figura 1.1.

Figura 1.1 - Exemplos de logos capturados de vídeos veiculados na televisão brasileira. O logo da emissora Discovery, (a) e (b) é parcialmente animado (o globo terrestre gira em torno de seu eixo).

O processo se divide basicamente em três fases:

- cadastrar os logos para criar um banco de emissoras que serão monitoradas;
- detectar o logo frente a um banco de logos pré-cadastrados, e;
- posteriormente, confirmar a existência deste logo, ao longo do tempo, nos pacotes de vídeo. Nos dois processos é aproveitado o fato de haver uma grande correlação entre quadros consecutivos.

Para a implementação em um processador digital de sinais, foi utilizado um kit de desenvolvimento da Analog Devices (EZ Kit-Lite BF533), dedicada a aplicações multimídia por possuir:

- codificador e decodificador de vídeo para sistemas NTSC, PAL e SECAM;
- capacidade de armazenamento volátil (memória SDRAM) de 32MB (trinta e dois mega bytes) e não-volátil (memória Flash) de 2MB (dois mega bytes). Estes números são adequados ao processo, uma vez que, em média, cada logo possui 2500 (dois mil e quinhentos) pixels, com 8 (oito) bits por pixel. Uma operadora de distribuição de televisão a cabo, hoje, distribui não mais que 200 (duzentos) canais. Este hardware suportaria um total de até 600 (canais). O que previne uma expansão considerável no número de canais.

1.1 Objetivos

O primeiro objetivo deste trabalho de mestrado é estudar e produzir um método para a detecção de logos em vídeos previamente gravados e que, portanto não precisam necessariamente estar preocupados com desempenho e performance de processamento. Em particular, analisar os desafios existentes quando se quer detectar com grande eficiência e robustez logos semitransparentes nos mais diversos fundos. O segundo objetivo é adotar um método eficaz e implementá-lo em um hardware dedicado para o funcionamento em tempo real gerando audiência de televisão.

1.2 Contribuições

A medição de audiência de televisão é importante em seu aspecto social informando o comportamento do povo em geral, além de ser uma fonte muito utilizada pelo setor publicitário para a veiculação de comerciais e pelas emissoras para estabelecerem o custo de um comercial em determinado horário. Com a convergência digital fica cada vez mais difícil efetuar a medição por métodos tradicionais, como por freqüência do sintonizador.

Este estudo, publicado no Simpósio Internacional I2TS (Santos, 2006), comprova a eficiência e robustez da utilização de detectores de logos trabalhando com bordas em nível de cinza, no qual três tipos de logos: semitransparentes, opacos ou parcialmente animados são detectados corretamente, gerando a possibilidade de utilização do processo na medição de audiência de televisão do conteúdo recebido em tempo real. Com esse algoritmo, mostramos que é plenamente viável a elaboração de um produto para a medição de audiência em situações complexas como em televisão digital, através da saída de vídeo componente ou A/V existente em todos os "set-top-boxes" brasileiros, ou mesmo em aparelhos portáteis como o telefone celular onde só é possível a medição de audiência pelo áudio ou pelo vídeo recebidos.

1.3 Organização

No segundo capítulo apresentamos um resumo dos princípios básicos de processamento digital de imagens, televisão e os conceitos gerais envolvidos no reconhecimento de padrões.

O terceiro capítulo descreve o algoritmo gerado, os resultados obtidos de experimentos em tempo real pela aquisição de sinais através de uma fonte de vídeo obtidos, que serviram como motivação para a geração de novas versões do mesmo algoritmo.

O quarto capítulo apresenta as plataformas de desenvolvimento com maior profundidade, utilizada para implementar o algoritmo.

Por último apresentamos as conclusões finais.

2 PRINCÍPIOS FUNDAMENTAIS APLICÁVEIS AO PROCESSO

Neste capítulo apresentaremos alguns princípios básicos de processamento de imagens, além de conceitos de televisão utilizados no processo. A maioria das técnicas abordadas é vastamente conhecida na literatura.

2.1 Imagem Digital

2.1.1 Representação de uma Imagem Digital

(Gonzalez, 2002) define uma imagem monocromática, como sendo uma função $f(x, y)$, de duas dimensões, da intensidade da luz, onde x e y representam a coordenada espacial, e o valor f em qualquer ponto, é proporcional ao brilho, ou nível de cinza da imagem em qualquer ponto. Uma imagem digital é uma imagem $f(x, y)$ que foi discretizada em sua coordenada espacial e em seu brilho. Tipicamente temos 256 (duzentos e cinqüenta e seis) níveis de cinza, 8 (oito) bits para representar um pixel. O *pixel* é o nome dado ao menor elemento de imagem, também chamado de *detalhe de imagem*.

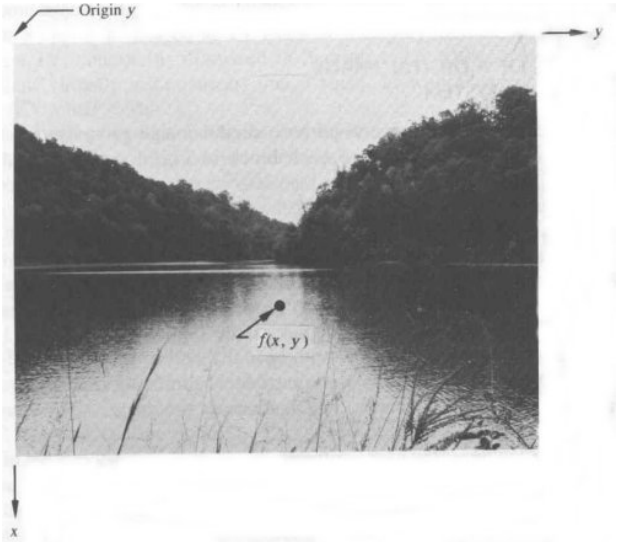


Figura 2.1 – Representação de uma imagem digital. Figura retirada de (Gonzalez, 2002).

2.1.2 Vizinhança

A imagem digital é composta por pixels, como foi explicado nas seções anteriores. Espacialmente, um pixel p localizado nas coordenadas (x, y), tem quatro pixels vizinhos nas posições verticais e horizontais. Estes pixels podem ser endereçados da seguinte maneira, (Gonzalez, 2002):

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1) \tag{2.1}$$

Os vizinhos horizontais e verticais formam um conjunto de pixels ao redor de (x, y) denominado 4-vizinhos. Uma característica importante nesta vizinhança é a distância unitária que cada pixel vizinho tem em relação ao pixel p.

Pensando em uma distância de um pixel, pode-se ter também vizinhos diagonais ao pixel p cujas coordenadas serão, (Gonzalez, 2002):

$$(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1) \tag{2.2}$$

Esta vizinhança em conjunto com 4-vizinhos forma a vizinhança 8-vizinhos do pixel p.

(Golay, 1969) utilizou uma estrutura hexagonal para definir a vizinhança. Com esta estrutura resolvem-se alguns problemas relacionados à conectividade em malhas retangulares. Dois problemas têm feito com que esta técnica não seja vastamente utilizada. Primeiro, a dificuldade em se utilizar scanners com esta técnica. Segundo, a malha hexagonal não é bem definida quanto a algumas operações como convolução e transformada de Fourier.

2.1.3 Conectividade

Conectividade é um conceito importante e muito utilizado nas áreas de filtragem, segmentação de imagens, codificação, análise de movimento, reconhecimento de padrões, etc. A conectividade define as fronteiras entre objetos ou regiões de uma imagem. (Gonzalez, 2002) diz que para se determinar se dois pixels estão conectados, deve-se determinar se eles são adjacentes em algum sentido, como por exemplo, com 4-vizinhos e, se a sua amplitude, como nível de cinza, respeita algum critério de similaridade. Ou seja, se temos por exemplo uma imagem binária, onde os pixels podem ter os valores 0 (zero) ou 1 (um), define-se um modelo, como por exemplo 8-vizinhos, se estamos em um pixel p em (x, y). Analisaremos a redondeza especificada e caso algum pixel dentro de:

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1),\ (x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1) \quad (2.3)$$

ter o mesmo valor será considerado conectado ao pixel p.

(Braga-Neto, 2004) defini um conceito de conectividade que mostra o quanto um objeto está mais ou menos conectado a outro dependendo da escala utilizada. Para demonstrar o conceito a figura 2.2 ilustra a conectividade de objetos em função da variação da escala dos objetos medidos.

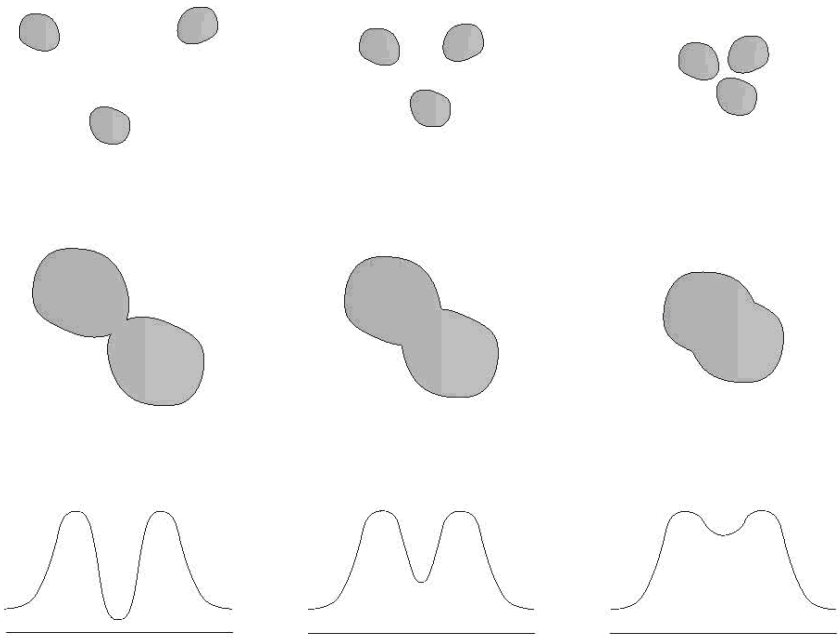


Figura 2.2 – A figura mostra um aumento de conectividade da esquerda para a direita para os objetos contidos em cada linha. Os objetos são os mesmos, porém com diferentes escalas. Figura retirada de (Braga-Neto, 2004).

2.2 Vídeo

A palavra vídeo, que vem do latim "eu vejo", é uma tecnologia de processamento de sinais eletrônicos analógicos ou digitais para representar imagens em movimento. A aplicação mais difundida do vídeo é a televisão. Uma animação composta em sua grande maioria de fotos seqüenciais, quadro-a-quadro também é chamada de vídeo.

Existem vários formatos de vídeo analógicos e digitais. Como exemplos de formatos analógicos podemos citar: NTSC, PAL, SECAM, etc, (Nince, 1991); e como exemplos de formatos de vídeo digitais podemos citar: AVI compactado ou não compactado (MICROSOFT, AVI File Reference), MPEG versões 1, 2 ou 4 (Symes, 2001), DIVX, etc.

A criação do vídeo só foi possível devido a um fenômeno existente no olho humano denominado *persistência da visão* (GROB, 1979), que corresponde a um tempo da ordem de 50 (cinqüenta) milisegundos em que a imagem permanece no cérebro após ser retirada da retina. Da mesma forma, uma imagem projetada na retina gasta um tempo, da mesma ordem de grandeza do tempo de persistência, para atingir o cérebro (tempo de sensibilização).

Levando-se em consideração o fenômeno da persistência da visão, uma cena móvel pode perfeitamente ser representada por um número finito de imagens estáticas, desde que a troca de imagens se faça em um tempo menor que o tempo de persistência da visão para que se tenha a sensação de continuidade dos movimentos. O olho, neste caso, funciona como elemento integrador, proporcionando uma percepção contínua de movimentos a partir de uma apresentação descontínua de imagens estáticas.

O número de amostras na unidade de tempo denomina-se *freqüência de amostragem*, que no caso do cinema foi padronizada em 24 (vinte e quatro) quadros por segundo. Este valor corresponde a um compromisso estatístico entre o olho normal médio e os movimentos mais prováveis. A freqüência de amostragem de 24 (vinte e quatro) quadros por segundo resolve completamente o efeito da continuidade da imagem, entretanto, não resolve o problema da variação da intensidade luminosa, que é percebida pelo olho como uma cintilação da iluminação da tela. As experiências demonstraram que uma freqüência de cintilação equivalente a 48 (quarenta e oito) quadros por segundo não é perceptível pelo olho humano. Para resolver este problema sem aumentar o número de quadros, as máquinas de cinema reproduzem o mesmo quadro duas vezes para obter a freqüência de cintilação de 48 (quarenta e oito) quadros por segundo com freqüência de amostragem de 24 (vinte e quatro) quadros por segundo. Na televisão a freqüência de cintilação e de amostragem varia de sistema para sistema.

2.2.1 O Sistema de Televisão Brasileira

Como mencionado em (Nince, 1991), os menores elementos na televisão são reproduzidos nas linhas. No Brasil o padrão de televisão adotado é o PAL-M (Nince, 1991).

O sistema PAL (phase alternating line), que significa fase alternada linha-a-linha, foi desenvolvido a partir do NTSC (Nince, 1991) pelo Dr. Walter Bruch, da Telefunken Alemã com dois objetivos:

- Evitar que erros de fase provoquem mudanças na cor da imagem reproduzida, e;

- Reduzir, consideravelmente, os erros de cromaticidades produzidas na modulação em quadratura pela redução de faixa do sinal Q.

No padrão M, adotado no Brasil, o número total de linhas horizontais é 525 (quinhentos e vinte e cinco), destas, 31,5 (trinta e um e meio) são apagadas para permitir o retorno da varredura vertical, sobrando um total de 493,5 (quatrocentos e noventa e três e meio) linhas visíveis na tela. Essas linhas correspondem aos menores elementos da imagem na televisão. A altura do menor elemento de imagem corresponde exatamente à espessura de uma linha horizontal.

Pode-se achar estranho o fato de aparecer 0,5 (meia) linha, mas, isto têm uma explicação técnica, o padrão M foi estabelecido para 525 (quinhentos e vinte e cinco) linhas com tempos de retorno horizontal e vertical respectivamente de 16% (dezesseis porcento) e 6% (seis porcento) dos períodos de freqüência horizontal e freqüência vertical, por isso o número de linhas apagadas pode ser calculado por (Nince, 1991):

$$N_o = 525 \times 0{,}06 = 31{,}5 \text{ linhas.} \tag{2.4}$$

Enquanto que o número de linhas visíveis pode ser calculado por (Nince, 1991):

$$N = 525(1 - 0{,}06) = 493{,}5 \text{ linhas.} \tag{2.5}$$

A reprodução das linhas é feita de uma maneira intercalada, em que as 525 (quinhentos e vinte e cinco) linhas são divididas em dois campos iguais, pares e ímpares, intercalados, isto para reduzir o efeito da cintilação da imagem. Cada campo possui 262,5 (duzentos e sessenta e dois e meio) linhas, significando que as linhas 1, 3, 5, 7,..., 525 pertencem ao primeiro campo e as linhas 2,4,6,8,..., 524 pertencem ao segundo campo. Pela reprodução das linhas serem feitas de maneira intercalada, alguns fabricantes de equipamentos de televisão veiculam a denominação 525i para afirmar que os quadros são formados por linhas intercaladas e 525p, quando o quadro é formado continuamente, exemplo o DVD.

Para o Brasil, padrão PAL-M, a freqüência de quadros (campos) é de 60Hz (sessenta Hertz), ou seja, temos 60 (sessenta) campos por segundo, que produzem um total de 30 (trinta) quadros.

No início da década de 50, nos EUA, o comitê National Television System Committee estabeleceu o padrão para TV colorida conhecido como NTSC. Como já existia uma grande base de aparelhos P&B (preto e branco) funcionando, para manter a compatibilidade com eles o novo sistema deveria ter suas imagens captadas sem problemas por esses aparelhos também. Desenvolver um sistema completamente novo faria com que todos esses televisores tornassem-se obsoletos da noite para o dia, exigindo assim sua troca, e a coexistência de dois sistemas não era algo prático nem viável.

A solução encontrada foi embutir os sinais de cores dentro do sinal já existente P&B (preto e branco), de modo que televisores antigos simplesmente o ignorassem. Para isso, através da técnica de multiplexação (que, de maneira simplificada, nesse caso significa ter diferentes tipos ondas eletromagnéticas (chamadas subportadoras) "montadas" sobre uma mesma onda principal (chamada portadora) onde se varia a freqüência dessas ondas subportadoras) foi incluída uma nova onda no sinal, correspondendo a sua parte de cor. Como os aparelhos antigos não "liam" essa freqüência nova, estava revolvido o problema.

Os engenheiros perceberam, no entanto, que essa nova freqüência de onda incluída causava interferências no sinal de áudio do sistema, que possuía freqüência próxima da freqüência do novo sinal de cor. Para minimizar o problema de poder ocorrer essa interferência, algumas modificações no padrão original tiveram que ser feitas, sempre de modo a não interferir na recepção dos sinais P&B (preto e branco) pelos antigos televisores. E uma dessas modificações foi reduzir ligeiramente o “frame rate” (taxa de quadros por segundo) do sinal, em uma taxa de 0,1% (mais precisamente o resultado da divisão de 1000/1001, ou seja, 0,999000999000999000999000999...).

Com isso, ao invés de se ter 30 quadros por segundo, passou a ter-se 30000/1001 = 29,97 quadros por segundo (ou, mais precisamente, 29,970029970029970029970029970... quadros por segundo). Esse fato levou a criação de técnicas, para opcionalmente ajustar a contagem de tempo do vídeo com a do tempo real. Como o vídeo "corre" mais lentamente do que o tempo real (em uma taxa de 0,1%, como visto acima), ao término de cada minuto (exceto os terminados em "0") a numeração de contagem dos quadros avança 2 (dois) quadros, para compensar essa diferença da taxa de quadros do sistema NTSC com o tempo real.

2.2.2 O Padrão ITU-656

O padrão ITU-656, também chamado de Recomendação BT.656 descreve um simples protocolo de vídeo digital para padrões de sinais descomprimidos PAL ou NTSC (525 ou 625) para TV. Este padrão foi criado pela ITU-R (União Internacional de Telecomunicação, Setor de Radiocomunicação), responsável por definir a alocação das rádios freqüências, também reduzir a interferência entre estações de rádio de vários países, além de outras atribuições.

A construção do protocolo foi baseada na codificação 4:2:2 de crominância e luminância, baseada na Recomendação BT.601, que provê:

- dados de vídeos entrelaçados, ou seja, as linhas são transmitidas com um espaço de uma linha entre cada; com isso 2 (dois) campos, par e ímpar, são transmitidos separadamente;
- padrão de cor YCrCb;
- freqüência de amostragem de 13,5MHz (treze e meio mega Hertz) por pixel.

A sub-amostragem do sinal de croma, utilizado na codificação 4:2:2 é uma prática muito utilizada para se implementar uma maior resolução para a luminância em comparação com a cor. Esta prática também é utilizada no padrão de codificação do JPEG (International Telecommunication Union, 2006).

O formato padrão ITU-656 é uma seqüência de 8-bit (oito bits) ou 10-bit (dez bits) por palavra, transmitida em uma taxa de 27MHz (vinte e sete mega Hertz). O comprimento horizontal de uma linha é delimitado pelas palavras de 4 (quatro) bytes SAV (início de vídeo ativo) e EAV (final de vídeo ativo) formando um código. A palavra SAV também contém alguns bits que informam a posição da linha no campo ou no quadro. Os pixels são transmitidos no formato YCrCb. Depois da palavra SAV são transmitidos 8 (oito) bits de luminância (Y), 8 (oito) bits de crominância (Cb), seguidos por mais 8 (oito) bits de crominância (Cr), (ITU-656, 2006).

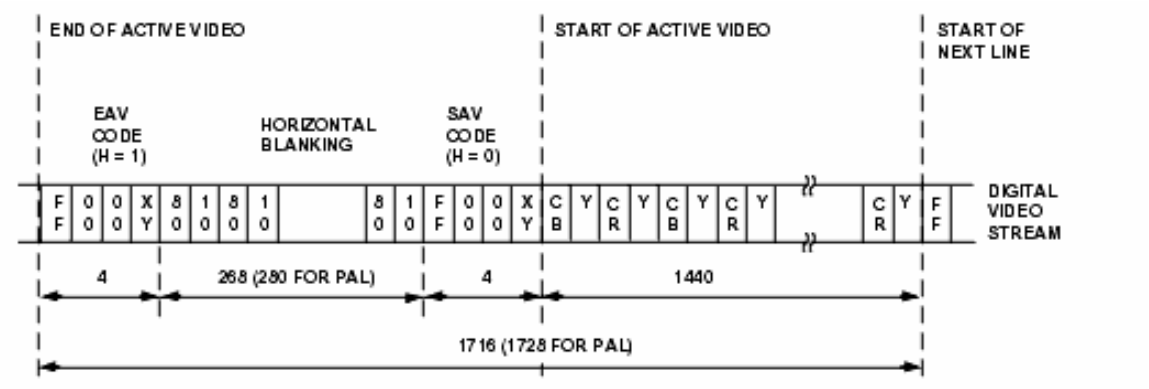


Figura 2.3 – Protocolo de vídeo digital no padrão ITU-656. Figura retirada de (Analog Devices, 2003).

2.2.3 Audiência de Televisão

A medição de audiência de televisão reporta o comportamento da população. No Brasil, a medição é feita desde a década de 50, (IBOPE). No Brasil há duas metodologias de medição de audiência: os Peoplemeters e os Cadernos.

O peoplemeter é um aparelho responsável por medir as emissoras e sua programação é registrada automaticamente e transmitidas ao escritório central para futuro processamento e entrega dos dados aos clientes, (IBOPE). Cada indivíduo do domicílio é identificado, permitindo assim a formação de levantamentos de comportamento por idade, faixa etária, etc.

Uma das formas mais tradicionais de medição de audiência é o caderno, este consiste no preenchimento por parte do indivíduo da programação que ele assistiu durante o dia em intervalos de 15 (quinze) minutos. Os cadernos são recolhidos no domicílio a cada duas semanas e suas informações são armazenadas em banco de dados através de um sistema de leitura ótica com entrega mensal dos dados, (IBOPE).

No Brasil as informações de audiência são registradas a cada minuto e enviadas a central de duas maneiras: minuto a minuto, em um sistema chamado de “Real-Time”, e durante a madrugada chamado de “Overnight”, (IBOPE).

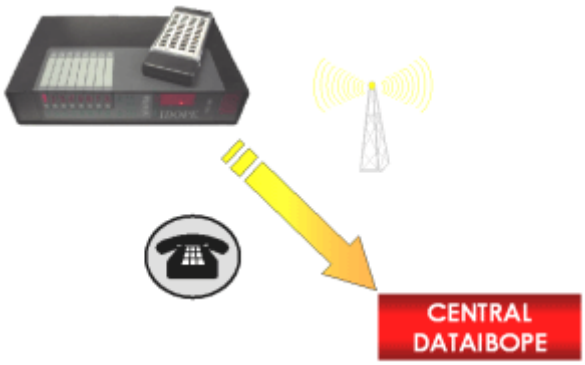


Figura 2.4 – Exemplo de medição de audiência no Brasil, (IBOPE).

A medição eletrônica de audiência no Brasil começou em 1970 com o equipamento eletromecânico chamado Tvêmetro I lançado em São Paulo. Ele media um televisor por domicílio através da perfuração de uma fita de papel toda vez que o colaborador trocava de canal através do sintonizador eletromecânico disponível nos televisores na época. A coleta era manual, ou seja, de tempos e tempos uma pessoa do Ibope passava no domicílio para efetuar a troca da fita. Já nesta época havia a comunicação via rede elétrica entre os aparelhos instalados na casa. A audiência era revelada após duas semanas, (IBOPE).

Em 1973 foi lançado o Tvêmetro II para a cidade do Rio de Janeiro com as mesmas características do equipamento anterior, (IBOPE).

No ano de 1985 foi lançado o primeiro medidor de audiência eletrônico, o Tevetron. Instalado nas cidades de São Paulo, Rio de Janeiro e interior de São Paulo efetuava a medição de um domicílio por televisor e registrava os dados em uma fita magnética. A coleta dos dados era semanal feita por um funcionário. A audiência era entregue após duas semanas, (IBOPE).

Em 1989 foi lançado o primeiro peoplemeter Dib2, instalado nas praças de São Paulo, Rio de Janeiro e Porto Alegre. Media até quatro televisores por domicílio e coletava as informações por telefone e radiofreqüência. Em São Paulo a medição era feita em tempo real e de modo "overnight" nas demais praças. Marcou a expansão das operações do Ibope na América Latina, (IBOPE).

A coleta por Internet celular foi iniciada em 1996 com o lançamento do Dib4. As maneiras anteriores de coleta: telefone e radiofreqüência também eram utilizados, (IBOPE).

Em 2007 foi lançada uma amostra piloto em São Paulo de uma tecnologia preparada para medir audiência de novas mídias como: celular, internet e rádio. Neste equipamento há reconhecimento de conteúdo por meio de áudio, (IBOPE).

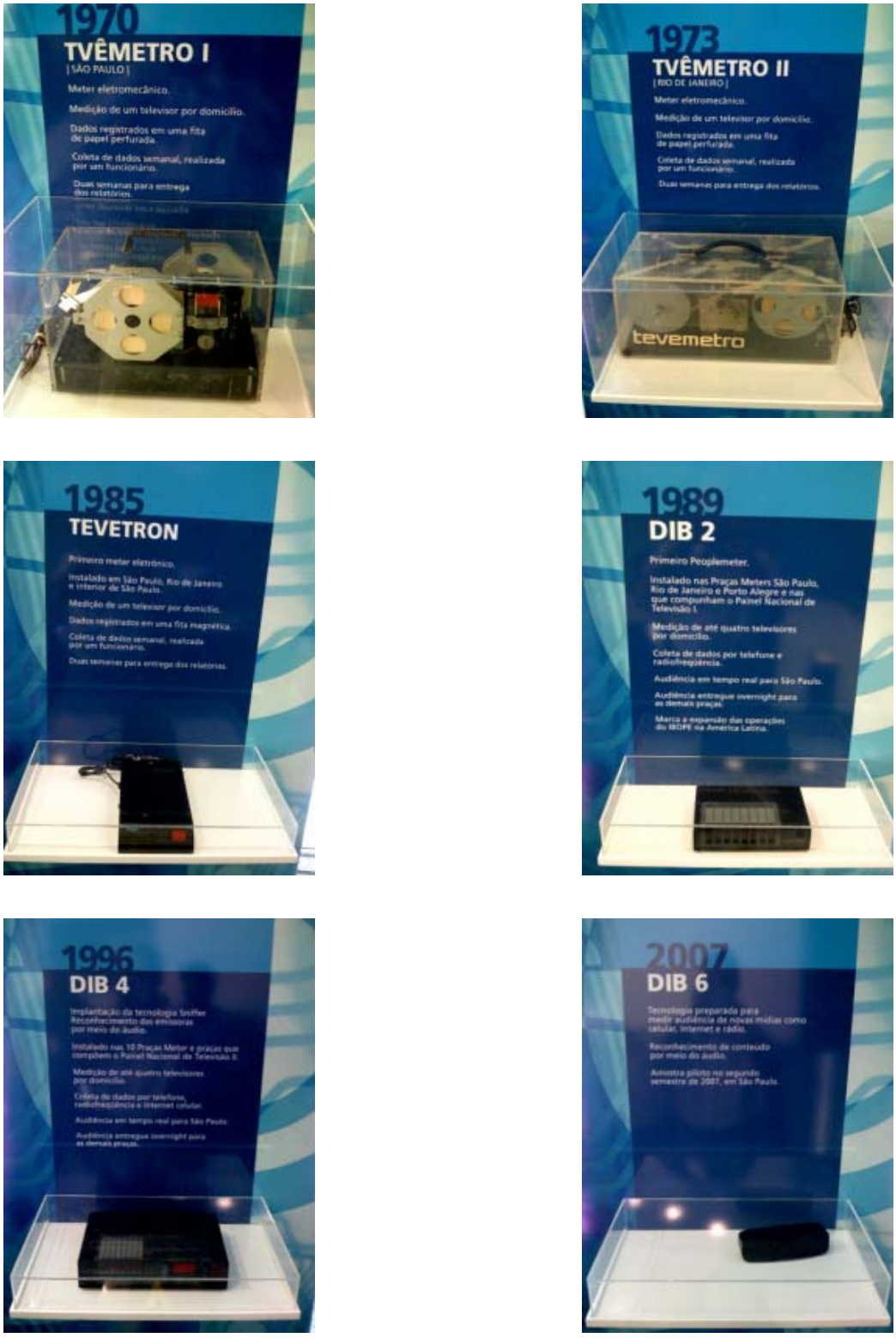


Figura 2.5 – História dos medidores de audiência brasileiros. Fonte: Ibope.

O IBOPE constrói a amostra sobre os hábitos televisivos com base nos dados do censo demográfico brasileiro, realizado pelo Instituo Brasileiro de Geografia e Estatística (IBGE) e nos estudos sociodemográficos do próprio IBOPE.

Essa montagem ocorre a partir de análises estatísticas que permitem coletar e fazer a ponderação dos dados para representar a mesma proporção de segmentos da população que o conjunto inteiro dos brasileiros, (IBOPE).

Supondo que, a partir dos dados do censo, a população total se dividisse em 15% nas classes A e B, 35% na classe C e 50% nas classes D e E, por exemplo, o IBOPE montaria uma amostra aleatória com uma divisão idêntica à da população total, distribuindo diferentes pesos aos elementos da amostra.

2.2.4 Média Móvel Temporal "Time-Averaging"

A função de média móvel temporal, "*time-averaging"* é um método utilizado para melhorar a razão sinal/ruído de imagens, filtrando as variações abruptas dos pixels. Ou seja, após algumas imagens com um determinado padrão, seja ele um objeto qualquer, que, espacialmente se mantém imóvel em várias imagens consecutivas. Sendo exposto a função de média móvel temporal, uma imagem com este padrão mais realçado aparecerá, enquanto que em outras áreas da mesma imagem permanecerão borradas. Para que o padrão existente nesta imagem de saída da função desapareça, um determinado número de imagens sem o padrão deverá ser exposto a função. O tempo de persistência dos pixels é determinado pelo fator σ da equação 2.6. Ou seja, todo padrão de imagem, seja ele um logo ou um outro objeto qualquer, permanecerá visível em uma imagem gerada de saída pela aplicação desta função, mesmo que uma outra imagem qualquer sem este objeto seja aplicado a entrada da função de média móvel temporal, por um tempo inferior ao tempo de persistência dos pixels.

Matematicamente temos, (OPPENHEIM, 1975):

$$F_n(j,k) = \frac{\sigma - 1}{\sigma} F_{n-1}(j,k) + \frac{1}{\sigma} F(j,k) \qquad (2.6),$$

onde $F_n(j,k)$ é a nova imagem após processado uma média móvel temporal; $F_{n-1}(j,k)$ é a última imagem gerada pelo processo; $F(j,k)$ é a imagem do novo quadro capturado; e σ é o fator de média móvel temporal escolhido pelo usuário.

2.3 Reconhecimento de Imagens

Nesta seção, alguns conceitos utilizados no desenvolvimento do estudo em relação à questão de reconhecimento dos logos são explanados.

2.3.1 Convolução

Convolução é uma maneira matemática de se combinar dois sinais para formar um terceiro sinal (Smith, 1999). É uma das mais importantes técnicas no Processamento Digital de Sinais. (Lathi, 1968) diz que o teorema da convolução é, talvez, uma das ferramentas mais eficazes na análise em freqüência.

Dadas duas funções $f_1(t)$ e $f_2(t)$ no domínio contínuo do tempo, a equação abaixo formula a convolução destas duas funções:

$$f(t) = \int_{-\infty}^{+\infty} f_1(\tau) f_2(t-\tau) d\tau \qquad (2.7).$$

Uma outra maneira de se denotar uma convolução entre duas funções é utilizar a seguinte notação:

$$f(t) = f_1(t) * f_2(t) \qquad (2.8).$$

Existem dois teoremas importantes sobre a convolução, um se diz respeito ao domínio do tempo e outro sobre o domínio da freqüência.

Teorema da convolução no tempo diz que:

$$f_1(t) \longleftrightarrow F_1(w) \qquad (2.9)$$

$$f_2(t) \longleftrightarrow F_2(w) \tag{2.10}$$

então,

$$\int_{-\infty}^{+\infty} f_1(\tau) f_2(t-\tau) d\tau \longleftrightarrow F_1(w) F_2(w) \tag{2.11}$$

ou seja,

$$f_1(t) * f_2(t) \longleftrightarrow F_1(w) F_2(w) \tag{2.12}.$$

Teorema da convolução na freqüência diz que:

$$f_1(t) \longleftrightarrow F_1(w) \tag{2.13}$$

$$f_2(t) \longleftrightarrow F_2(w) \tag{2.14}$$

então,

$$f_1(t) f_2(t) \longleftrightarrow \frac{1}{2\pi} \int_{-\infty}^{+\infty} F_1(u) F_2(w-u) du \tag{2.15},$$

ou seja,

$$f_1(t) f_2(t) \longleftrightarrow \frac{1}{2\pi} \left[F_1(w) * F_2(w) \right] \tag{2.16}.$$

Com isso concluímos que a convolução de duas funções no domínio do tempo é equivalente à multiplicação dos seus espectros no domínio da freqüência, e que a multiplicação de duas funções no domínio do tempo é equivalente a convolução dos seus espectros no domínio da freqüência. Esta conclusão é muito importante no aspecto performance do sistema, pois às vezes, dependendo do tamanho das imagens, a realização do cálculo de FFT seguida pela multiplicação dos espectros e a inversa de FFT tornam-se mais rápida. (Brigham, 1974) diz que para um vetor de uma dimensão, o cálculo de FFT é viável se o número de pontos for superior a 32(trinta e dois).

Existem algumas propriedades importantes que são aplicadas a álgebra de convolução de acordo com diretrizes semelhantes às da multiplicação:

1. Lei comutativa:

$$f_1(t) * f_2(t) = f_2(t) * f_1(t) \tag{2.17};$$

2. Lei distributiva:

$$f_1(t) * \left[f_2(t) + f_3(t)\right] = f_1(t) * f_2(t) + f_1(t) * f_3(t) \tag{2.18};$$

3. Lei associativa:

$$f_1(t) * \left[f_2(t) * f_3(t)\right] = \left[f_1(t) * f_2(t)\right] * f_3(t) \tag{2.19}.$$

Vendo as propriedades, podemos pensar que a convolução é uma operação matemática muito parecida com a multiplicação, mas a multiplicação envolve dois números para se gerar um terceiro, já a convolução envolve dois sinais para gerar um terceiro sinal.

No domínio discreto, pensamos na convolução da mesma maneira. Primeiramente, todo sinal de entrada de um sistema pode ser decomposto em uma série de impulsos, na qual cada impulso é visto como uma versão deslocada e escalada da função delta. Segundo, a saída do sistema pode ser visto também como vários impulsos também escalados e deslocados da resposta ao impulso do sistema. Portanto, conhecendo a resposta ao impulso de um sistema podemos calcular a resposta a um sinal de entrada. Matematicamente:

$$x[n] * h[n] = y[n] \tag{2.20},$$

ou, pode ser escrita da seguinte maneira:

$$y[n] = \sum_{m=0}^{n} h[n-m]x[m] \tag{2.21},$$

e,

$$y[n] = \sum_{m=0}^{n} h[m]x[n-m] \qquad (2.22).$$

2.3.2 Coeficiente de correlação

A *correlação* é uma operação matemática muito parecida com a convolução. Assim como a convolução, a correlação usa dois sinais para gerar um terceiro sinal. Caso os sinais de entrada sejam provenientes de duas fontes independentes, esta correlação ganha o nome de *correlação cruzada*. Agora, se os sinais de entrada são os mesmos, está se efetuando uma *autocorrelação*.

Pode-se perguntar para que usaríamos a autocorrelação, mas ela se justifica para o caso onde se quer encontrar padrões que sejam repetidos em um sinal, conseguindo até medir a sua periodicidade.

[(Gonzalez, 2002), (Castleman, 1996)] definem o coeficiente de correlação cruzado entre duas imagens:

$$R(m,n) = \sum_{j}\sum_{k} \widetilde{G}(j,k)\widetilde{L}\ (j+m,k+n) \qquad (2.23),$$

onde $\widetilde{G}(j,k)$ e $\widetilde{L}(j,k)$ são as imagens em nível de cinza com seus valores médios retirados, ou seja (Russ, 1994):

$$\widetilde{G}(j,k) = G(j,k) - \overline{G}(j,k) \text{ e } \widetilde{L}\ (j,k) = L\ (j,k) - \overline{L}\ (j,k), \quad 1 \le i \le n \qquad (2.24),$$

onde $\overline{G}(j,k)$ e $\overline{L}\ (j,k)$ são os níveis médios das imagens em nível de cinza e $G(j,k)$ e $L(j,k)$ são as imagens em nível de cinza sem correção do fator CC.

O maior valor encontrado representa o ponto (*m*, *n*) no qual existe a maior semelhança entre as duas imagens. Como este valor varia em função da distribuição em nível de cinza das imagens analisadas, é difícil determinar um valor ótimo que sirva como limiar de decisão para concluir se determinado logo analisado foi encontrado ou não. Para resolver este problema,

normalizamos a correlação cruzada, como definido em [(Gonzalez, 2002), (Castleman, 1996), (Russ, 1994)].

A normalização é feita, utilizando-se o comprimento dos vetores $\widetilde{G}$ e $\widetilde{L}$:

$$\gamma(m,n) = \frac{\sum_j \sum_k \widetilde{G}(j,k)\widetilde{L}_i(j+m,k+n)}{\left\{\sum_j \sum_k \left[\widetilde{G}(j,k)\right]^2 \sum_j \sum_k \left[\widetilde{L}_i(j+m,k+n)\right]^2\right\}^{1/2}} \quad (2.25).$$

O coeficiente de correlação encontrado $\gamma(m,n)$ pode variar entre -1 e 1.

Utilizando o Teorema da Convolução pode-se chegar a uma fórmula para cálculo de correlação mais rápida dependendo do comprimento das imagens analisadas, (Brigham, 1974). Esta fórmula calcula a correlação utilizando o espectro das imagens. Para fins de notação, o espectro da imagem G será denotado como G, e o espectro da imagem L será denotado como L. Então, matematicamente, (Ifeachor, 2002) define:

$$\gamma(j) = \frac{1}{N} F_D^{-1}\left[\mathrm{G}^*(k)\mathrm{L}(k)\right] \quad (2.26),$$

onde F_D^{-1} denota a Transformada Discreta Inversa de Fourier, G^* é o espectro da imagem G complexa conjugada.

Ao se utilizar à equação (2.26) é necessário calcular dois FFT's e uma inversa de FFT, por este motivo, a utilização de tal método não é aplicado a qualquer tamanho de vetor.

2.3.3 Detecção de Bordas

A detecção de descontinuidades em imagens em tons de cinza é de grande uso em muitas aplicações práticas, (Gonzalez, 2002). A razão para isso é que, tentar resolver o problema pensando em detecção de formas, como linhas ou pontos, é um pensamento não generalista para aplicações práticas.

Uma borda pode ser definida como um limite entre duas regiões com relativa distinção de tons de cinza.

A maioria das técnicas de detecção de bordas é baseada em um operador de derivada local. Isto porque o perfil na transição das regiões, bordas, seria modelado por uma rampa com uma inclinação. Então a primeira derivada desta função forneceria uma constante em uma região e, outra constante diferente na região de transição. Caso se efetue uma segunda derivada, todas as regiões exceto nas bordas teriam valor 0. Com o uso destas duas derivadas é possível determinar a presença da borda com a magnitude da primeira derivada, e com o auxílio do sinal da segunda derivada pode-se determinar as regiões interiores e exteriores do objeto.

O gradiente de uma imagem $f(x,y)$ localizado em (x,y) é definido como um vetor bi-dimensional:

$$\mathbf{G}[f(x,y)] = \begin{bmatrix} \mathbf{G}_x \\ \mathbf{G}_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{2.27}$$

Para detecção de bordas estamos interessados na magnitude do vetor, chamado de *gradiente*, assim:

$$\mathbf{G}[f(x,y)] = [G_x^2 + G_y^2]^{1/2} \tag{2.28}$$

(Gonzalez, 2002) aproxima o gradiente pelos valores absolutos:

$$\mathbf{G}[f(x,y)] \approx |G_x| + |G_y| \tag{2.29}$$

Com esta definição pode-se montar uma máscara que servirá como base para os demais operadores quando utilizada uma vizinhança 3x3 (três linhas por três colunas):

x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9

(2.30).

Para uma vizinhança 3x3 (três linhas por três colunas), em um ponto (x, y), os componentes dos gradientes para o operador Sobel podem ser definidos, utilizando a equação 2.30, da seguinte maneira:

$$\begin{aligned} \mathbf{G}_{\mathbf{x}} &= (x_7 + 2x_8 + x_9) - (x_1 + 2x_2 + x_3) \\ \mathbf{G}_{\mathbf{y}} &= (x_3 + 2x_6 + x_9) - (x_1 + 2x_4 + x_7) \end{aligned} \qquad (2.31).$$

Como exemplo, a máscara utilizada para os operadores de Prewitt e Sobel, em uma vizinhança 3x3 (três linhas por três colunas) é fornecida pela tabela (2.1). Cada ponto deve ser convoluído com as duas máscaras, pois uma maximiza o gradiente na direção vertical e a outra máscara maximiza o gradiente na direção horizontal.

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Operador Prewitt

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Operador Sobel

Tabela 2.1 – Máscara de operadores para detecção de bordas

Seguindo as definições apresentadas em operadores com gradientes, abaixo o operador local de Roberts, (Castleman, 1996):

$$g(x,y)] = \left\{ \left[\sqrt{f(x,y)} - \sqrt{f(x+1),(y+1)} \right]^2 + \left[\sqrt{f(x+1,y)} - \sqrt{f(x,y+1)} \right]^2 \right\}^{1/2} \quad (2.32),$$

onde $f(x, y)$ é a imagem de entrada nas coordenadas do pixel (x, y), e $g(x, y)$ corresponde a imagem de saída nas coordenadas do pixel (x, y).

Há um outro operador, chamado de Laplaciano ao qual utiliza a segunda derivada da função bi-dimensional para efetuar a detecção. É definido como, (Castleman, 1996):

$$\nabla^2 f(x,y) = \frac{\partial^2}{\partial x^2} f(x,y) + \frac{\partial^2}{\partial y^2} f(x,y) \qquad (2.33).$$

Assim como foi feito para os demais operadores, pode-se trabalhar com máscaras que maximizam o laplaciano vertical e horizontalmente. Abaixo, as máscaras utilizadas para o detector Laplaciano.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & 1 \\ -1 & 8 & -1 \\ -1 & -1 & 1 \end{bmatrix} \tag{2.34}$$

2.3.4 Casamento de Gabarito "Template Matching"

A técnica de casamento de gabarito "*template matching"* é uma técnica utilizada para verificar se uma réplica do objeto que se procura é semelhante a outros objetos em uma imagem qualquer.

O padrão armazenado é verificado pixel a pixel contra uma imagem qualquer e uma distância entre o padrão e a imagem formada naquele pixel é medida. Diz-se que há semelhança "matching" quando a distância é menor que um determinado limiar. Esta técnica é vastamente utilizada na literatura.

Uma medida de distância que se pode utilizar é a correlação cruzada normalizada, pois raramente um objeto reconhecido é igual ao padrão armazenado, devido a ruídos na imagem, efeitos de quantização e diferenças espaciais.

2.4 O Processador Digital de Sinais

As raízes do Processamento Digital de Sinais estão nas décadas de 1960 e 1970, quando os computadores digitais se tornaram disponíveis. Naquela época os computadores eram muito caros, e encontravam-se poucas restritas aplicações de Processamento Digital de Sinais, como os radares e sonares, (SMITH, 1999). A revolução dos computadores pessoal, nas décadas de

1980 e 1990, fez com que várias aplicações de DSP surgissem. Além das aplicações militares e necessidades governamentais, o Processamento Digital de Sinais foi inserido nas aplicações comerciais.

A velocidade de execução da maioria dos algoritmos de Processamento Digital de Sinais é muitas vezes limitada pela capacidade de multiplicações e adições requeridas. Por exemplo, a figura 2.6 mostra a implementação de um filtro FIR, muito comum em técnicas de DSP. Utilizando a notação padrão, o sinal de entrada é referido pelo *x[ ]*, enquanto que o sinal de saída é denotado pelo *y[ ]*. A tarefa é calcular a amostra na localização *n* no sinal de saída *y[n]*. Um filtro FIR realiza esta tarefa pela multiplicação, apropriada, das amostras do sinal de entrada por um grupo de coeficientes, denotados por: a_1, a_2, a_3, ..., e então adiciona os produtos. Na forma de equação, *y[n]* é dado por:

$$y[n] = a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + a_3 x[n-3] + a_4 x[n-4] + \ldots \tag{2.34}$$

Esta equação mostra simplesmente que, o sinal de entrada está sendo convoluído com os coeficientes do filtro. Dependendo da aplicação, pode haver poucos coeficientes, ou alguns milhares. Enquanto pode haver algum tipo de transferência de dados, ou testes de igualdade neste algoritmo, as operações matemáticas dominam o tempo de execução do algoritmo.

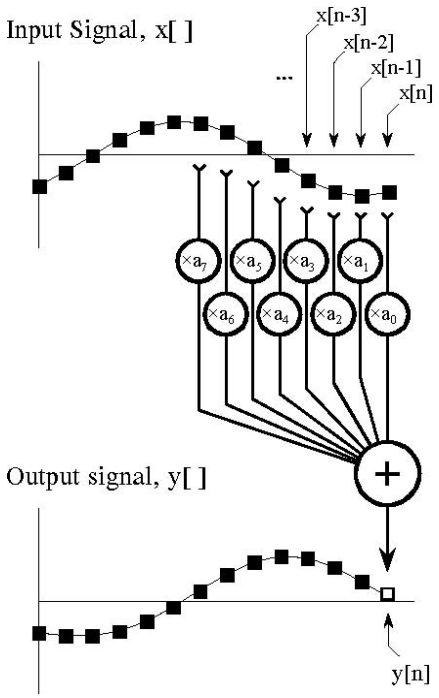


Figura 2.6 – Ilustração de um filtro FIR. Cada amostra de um sinal de saída y[n] é encontrado pela multiplicação das amostras de do sinal de entrada, x[n], x[n-1], x[n-2],..., pelos coeficientes, a_0, a_1, a_2, a_3,..., e somando os produtos. Figura retirada de (SMITH, 1999).

Com o propósito de se implementar, de uma maneira eficiente, os sistemas de Processamento Digital de Sinais, ao qual, caracterizam-se por possuir operações em tempo-real com ênfase na alta taxa de transferência de dados e o intensivo uso de operações aritméticas, (IFEACHOR, 2002), foi criado um processador, especialmente dedicado a este tipo de tarefa, chamado de Processador Digital de Sinais.

O Processador Digital de Sinais é um microprocessador especialmente desenhado para realizar tarefas de processamento digital de sinais, (SMITH, 1999). Estes processadores incluem algumas características como descrito abaixo:

- multiplicadores construídos em hardware, que permitem multiplicações rápidas. Alguns novos processadores incorporam instruções que realizam em um único ciclo as funções de multiplicar e acumular, enquanto há vários multiplicadores trabalhando em paralelo;

- barramentos separados para memória de programa e dados, conhecidos na literatura como arquitetura Harvard, ao qual permitem uma sobreposição de busca e execução;

- redução no consumo de ciclos para desvios e repetições;
- o uso de “pipelining”, técnica de se dividir a tarefa a ser realizada em tarefas menores, que reduz o tempo de execução de uma instrução, aumentando a velocidade.

Alguns processadores também incluem capacidades aritméticas em ponto flutuante, além de incorporar características encontradas em microprocessadores padrões, como porta serial, espaço de memória estendida e temporizadores.

Buscando uma maior eficiência na implementação de algoritmos de DSP, a Analog Devices criou o que foi chamado de *Arquitetura Super Harvard*. Esta arquitetura foi criada com o pensamento que geralmente os algoritmos gastam a maior parte do tempo em voltas, ou seja, executando o mesmo conjunto de instruções repetidamente. Isto significa que um conjunto de instruções irão continuamente passar da memória de programa a CPU. A arquitetura Super Harvard leva vantagem neste contexto por incluir uma cache de instruções na CPU. Esta cache é constituída de uma pequena memória de aproximadamente 32 (trinta e dois) mais recentes instruções de programas. Na primeira vez que uma volta é executada, as instruções de programa devem passar sobre o barramento de memória de programa. Isto resulta em uma operação mais lenta devido aos conflitos com os coeficientes que devem ser carregados com este caminho. Entretanto, em execuções futuras da mesma volta, as instruções de programa podem ser retiradas da cache de instruções, ou seja, toda a transferência de informação entre a memória e a CPU pode ser realizada em um único ciclo, pensando na execução de um filtro FIR: as amostras do sinal de entrada originam-se do barramento de memória, os coeficientes são carregados do barramento de programa, e as instruções vêm da cache de instruções.

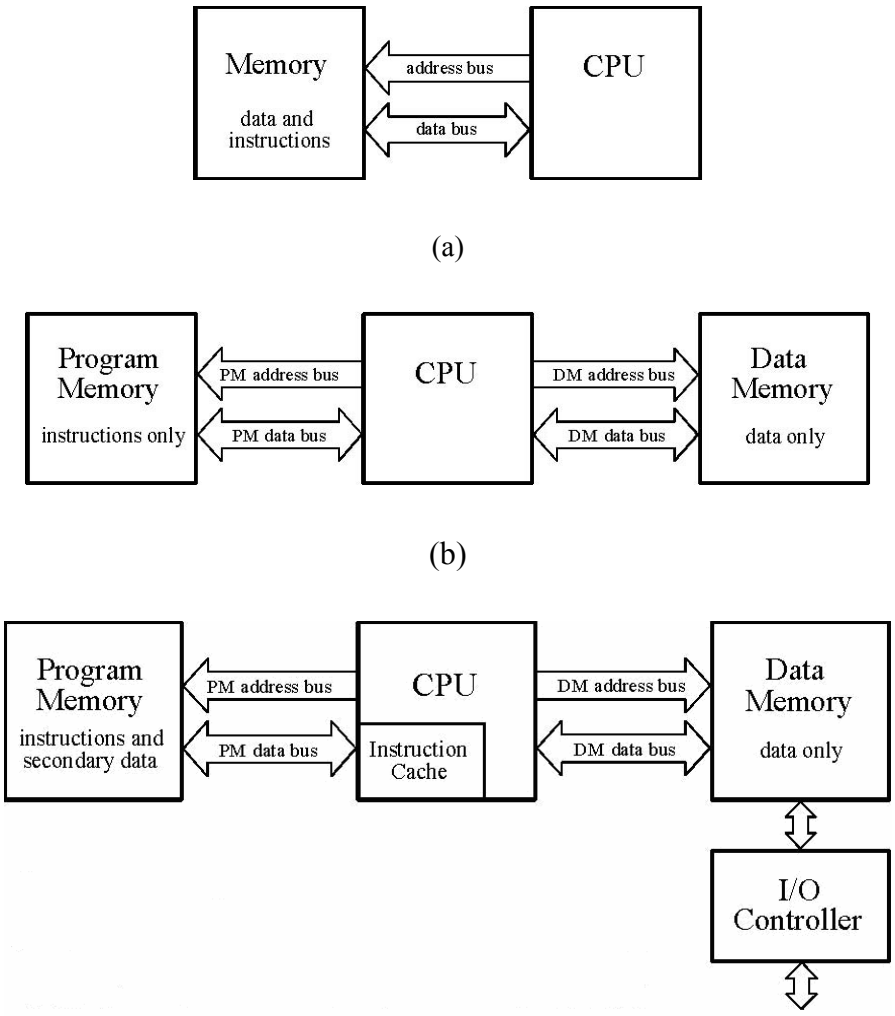


Figura 2.7 – Arquitetura de um microprocessador. Na figura (a) vemos a arquitetura Von Neumann, na figura (b) vemos a arquitetura Harvard, já na figura (c) temos uma arquitetura chamada de Super Harvard utilizada em DSP's Sharc da Analog Devices. Figura retirada de (SMITH, 1999).

Olhando dentro da CPU de um Processador Digital de Sinais, encontram-se os Geradores de Endereçamento de Dados (DAG), um para cada memória. Estes controlam os endereços enviados as memórias de programa e dados, especificando onde as informações devem lidas ou escritas. Em microprocessadores padrões, esta tarefa é inerente ao seqüenciador de programa, e é transparente ao programador. Entretanto, DSP's são desenhados para operarem com buffers circulares, e se beneficiam do hardware extra para o gerenciamento de forma mais eficiente. Assim evitando a necessidade de usar previamente, ciclos de instruções extras para manter a pista de como os dados são armazenados. Os DAG's também, em muitos DSP's

são desenhados para trabalhar em endereçamentos em bit-reverso, ferramenta utilizada para o cálculo de algoritmos de FFT.

3 ALGORITMO DE DETECÇÃO DE LOGOS

A medição de audiência de televisão é importante em seu aspecto social informando o comportamento do povo em geral, além de ser uma fonte muito utilizada pelo setor publicitário para a veiculação de comerciais e pelas emissoras para estabelecerem o custo de um comercial em determinado horário. Com a convergência digital, fica cada vez mais difícil efetuar a medição por métodos tradicionais como, por exemplo, freqüência. A geração de um processo que efetua a medição do conteúdo através de uma marca registrada como o logo, (Santos, 2006), facilita a medição de audiência em situações complexas como em televisão digital ou mesmo em aparelhos portáteis como o telefone celular. Neste capítulo apresentaremos o algoritmo gerado, assim como resultados práticos obtidos.

A primeira versão do algoritmo gerou a publicação do paper "Real-Time Opaque and Semi-Transparent TV Logos Detection" no Congresso Internacional *I2TS'06 vol. 1 pg. 156, Cuiabá – Brazil*. A segunda versão do algoritmo foi criada com o intuito de diminuir o alto índice de falso negativo encontrado. Como objetivo principal queremos expor a aplicação das técnicas apresentadas na seção 2 para a resolução de um problema, até então pouco estudado, mas com possibilidades de implantação imediata pela indústria. Como resultado final, a geração de audiência confiável, robusta a ataques, que minimize os falsos negativos e esteja dentro dos padrões de medição de audiência determinada pelos órgãos internacionais para que o processo seja utilizado. A geração da segunda versão do algoritmo gerou uma questão: por que não se utilizar um algoritmo com a mescla dos dois primeiros gerados? Devido a esta indagação, uma terceira versão do algoritmo foi produzida e avaliada para o texto final de dissertação.

3.1 O Problema de Detecção de Logo

O objetivo principal deste algoritmo é detectar logos, sejam eles opacos, semitransparentes ou animados, com o método de média móvel temporal, “template macthing”, pelas bordas em nível de cinza. Obter um algoritmo econômico em uso de memória e processamento para gerar uma aplicação em hardware dedicado com um baixo custo que trabalhe em tempo real. Visamos provar que é possível e viável utilizar tal método para essa aplicação. O tratamento de logos animados pode ser considerado igual a logos opacos, pois o reconhecimento é feito pela sua parte imóvel, o que torna o logo animado um logo opaco ou ainda semitransparente dependendo da emissora.

3.1.1 Formação de uma imagem

O decodificador de vídeo por nós utilizado é o ADV7183 da Analog Devices, o que tem como característica a entrega de 720 (setecentos e vinte colunas) colunas para qualquer padrão encontrado. Portanto, a faixa ativa, visível capturada e entregue na forma de um protocolo para o sistema de televisão analógica brasileira (PAL-M) são 493 (quatrocentos e noventa e três) linhas por 720 (setecentos e vinte colunas) colunas ao todo.

Os quadros recebidos pelo decodificador de vídeo estão separados em 2(dois) campos, par e ímpar. Assim cabe a aplicação formar o quadro entrelaçando os dois campos. A figura 3.1 ilustra uma imagem adquirida pelo kit de desenvolvimento e a mesma figura após o entrelaçamento.

(a)

(b)

Figura 3.1 – A imagem (a) apresenta os dois campos separadamente; imagem (b) apresenta os dois campos da figura (a) entrelaçados para formar um quadro. Imagens adquiridas em programação real da emissora Universal.

Estatisticamente, (Meisinger, 2005) observou que os logos de televisão de uma maneira geral, independente do tipo são posicionados nos quatro cantos da imagem. Utilizando esta informação nós criamos uma imagem *F*, formada pelos quatro cantos do quadro entrelaçado

(F_1, F_2, F_3 e F_4). Cada retângulo é composto por 100 (cem) linhas e 150 (cento e cinqüenta) colunas, totalizando 15000 (quinze mil) pixels por retângulo. Com isto um quadro que tinha 525×720(quinhentos e vinte e cinco multiplicado por setecentos e vinte) pixels, terá agora 200×300 (duzentos multiplicado por trezentos) pixels, isto implica em uma redução de memória consumida e tempo em processamento desses pixels que não serão relevantes para o processo de reconhecimento, uma vez que a probabilidade de existir um logo em qualquer outro pixel fora desse limite é pequena. A figura abaixo ilustra os quatro cantos aplicados na imagem 3.1(b). Estas posições seriam diferentes caso a implementação fosse feita para sistemas NTSC ou SECAM, por exemplo, do mesmo modo, HDTV deverá ter posições ligeiramente diferentes, devido a diferentes quantidades de linhas para cada sistema.

Figura 3.2 – Os quatro cantos (F_1, F_2, F_3 e F_4) são ilustrados.

Empiricamente, nós observamos que alguns logos podem aparecer em até 2 (dois) retângulos diferentes, dependendo da faixa horário ou tipo de programação veiculada no momento. Para resolver tal problema cada logo quando é incorporado ao banco de dados, recebe uma informação dizendo em quais cantos o mesmo deva ser procurado. Esta implementação gera uma redução no processamento, uma vez que cantos improváveis para determinados logos não serão pesquisados. Como exemplo podemos citar as emissoras CNN e SporTV. A emissora CNN disponibiliza o seu logo sempre no canto superior esquerdo F_1, enquanto a emissora SporTV disponibiliza o seu logo no canto superior direito F_2 durante a parte da

manhã, e na parte da tarde o mesmo logo aparece no canto inferior direito F_4, até o anoitecer, quando o logo volta a posição F_2. Na figura 3.3 é possível verificar a emissora CNN com a imagem formada sendo resultante do agrupamento dos quatro cantos, como explicado, além do ocorrido com a emissora SporTV também citada como exemplo no contexto.

(a)

(b) (c)

Figura 3.3 – Os quatro cantos foram agrupados para formar a imagem; na figura (a) temos a emissora CNN capturada de uma transmissão real; as imagens (b) e (c) ilustram o fato da emissora SporTV modificar o local onde imprime seu logo, as imagens foram capturadas de uma transmissão real.

A informação de cor das imagens é útil apenas para logos coloridos, como pode ser visto pela figura 3.3(a), onde a tonalidade vermelha poderia ser uma informação importante para a definição do logo da emissora CNN. Como o objetivo do trabalho é obter um único algoritmo para resolver qualquer tipo de logo existente atualmente, a informação de cor é descartada e

somente o nível de cinza das imagens é considerada informação relevante, pois quando temos logos semitransparentes a variação de cor do fundo da imagem forma o próprio logo, fazendo com que não haja uma tonalidade predominante no mesmo. Na figura 3.4, podemos observar esta ocorrência em logos semitransparentes.

(Yan, 2005) utiliza as informações de cores, além da forma do logo para formar o seu vetor de características e pesquisar por logo nos quadros em sua detecção Bayesiana.

(a) (b)

Figura 3.4 – Imagens da emissora Globo com logos semitransparentes. A cor do logo depende do fundo da imagem. Imagens capturadas de transmissão real.

3.1.2 O Banco de Logos

As emissoras a serem pesquisadas devem conter a sua marca registrada, logo, armazenada para eventual pesquisa. Para isso, algumas informações são armazenadas conjuntamente, tais como: cantos onde este logo pode aparecer, código da emissora para fins de audiência, nome da emissora, e dependendo da versão do algoritmo gerado, necessita-se de uma linha e coluna como sementes iniciais para a pesquisa.

Pixels que não pertencem às bordas dos logos, ou mesmo ao próprio logo, pois uma imagem é retangular e podemos ter logos com formas circulares, elípticas, etc, devem ser pintados de

preto, ou seja, ter intensidade igual a zero. Isso é feito para que estes pixels não influenciem o resultado da comparação com quadros capturados, seja positivamente, ou negativamente.

O mecanismo utilizado para carga de logos é a porta serial, onde se conectam um microcomputador e o nosso hardware dedicado, comunicando-se via RS-232.

As versões que necessitam de uma semente para iniciar a pesquisa, verificam se para cada canto configurado há tal semente ou um código de semente inexistente. Caso um código de semente inexistente foi configurado, a semente adotada será a média dos 3 (três) primeiros pontos encontrados para tal logo, criando assim um mecanismo de cadastro automático para início da pesquisa.

3.1.3 Pesquisa por Logos

Na literatura encontram-se vários métodos de reconhecimento de padrões que podem ser utilizados no trabalho de processamento de imagens.

(Yan, 2005) forma um espaço de características através de informações extraídas das cores e formas de um logo, para então treinar uma rede neural com o intuito de conseguir da mesma a probabilidade a posteriore de que determinado logo esteja em uma posição.

(Seeber, 2007) utiliza a distância Euclidiana, além de algumas heurísticas para detectar logos semitransparentes.

Em nosso trabalho, foi utilizado o conceito de procura através de um gabarito, também chamado de "Template Matching" por alguns especialistas da área, [(Gonzalez, 2002), (Castleman, 1996), (Russ, 1994)].

Para se reconhecer os logos nas imagens processadas é necessário ter os logos, L_1, L_2,..., L_n, no qual se necessita pesquisar além das informações sobre em quais cantos determinado logo pode aparecer, habilitando assim a pesquisa neste canto da imagem G. Com os logos gravados, dado um vídeo PAL-M, a cada Δt quadros é gerado uma imagem $\overline{F}$. Dependendo da versão do algoritmo, esta imagem pode ser passada pelo operador de Prewitt gerando a imagem G que é passada pela média móvel temporal (time-averaging), ou entrar direto no

processo de média móvel temporal (time-averaging). A imagem resultante é utilizada para a pesquisa dos logos contidos no banco de dados.

Temos que correlacionar o frame e o logo corrigidos do fator DC, então matematicamente temos:

$$\widetilde{G}(j,k) = G(j,k) - \overline{G}(j,k) \text{ e } \widetilde{L}_i(j,k) = L_i(j,k) - \overline{L}_i(j,k), \quad 1 \le i \le n \qquad (3.1),$$

onde $\overline{G}(j,k)$ e $\overline{L}_i(j,k)$ são os níveis médios das imagens em nível de cinza da imagem *G*(quadro no instante *t*) e o logo da emissora a ser pesquisada L_i.

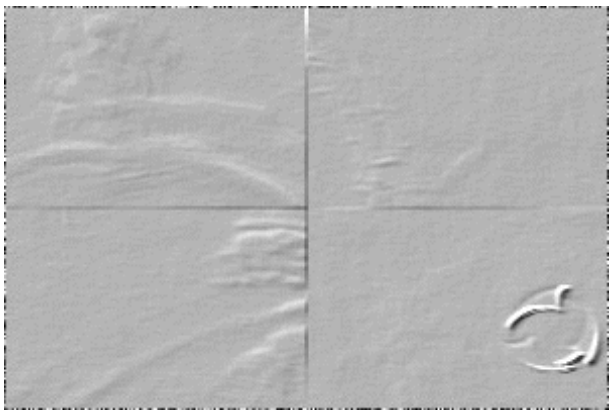
O maior valor encontrado representa o ponto (m, n) no qual existe a maior semelhança entre o logo e o quadro no instante *t*. Como este valor varia em função da distribuição em nível de cinza das imagens analisadas, é difícil determinar um valor ótimo que sirva como limiar de decisão para concluir se determinado logo analisado foi encontrado ou não. Para resolver este problema, trabalhamos com a correlação cruzada normalizada.

A normalização é feita, utilizando-se o comprimento dos vetores $\widetilde{G}$ e $\widetilde{L}$ como definido na equação (2.25). O coeficiente de correlação encontrado $\gamma(m,n)$ pode variar entre -1 e 1.

Para uma questão de monitoração do processo, pode-se gerar uma imagem de saída *H* que será a convolução da imagem *G* com o logo L_n. Matematicamente:

$$H = G * L_n \qquad (3.2).$$

Se a mascará for semelhante, um ponto de máxima aparecerá, localizado no centro do logo existente na imagem *G*. A figura 3.5 exemplifica o processo com uma imagem processada da emissora Record contra um logo da mesma emissora. É visualmente perceptível o ponto onde o logo se encontra na imagem convolucionada.



(a)

(b)

(c)

Figura 3.5 – Convolução: (a) imagem com bordas enfatizadas; (b) logo da emissora a ser pesquisada; (c) imagem resultante da convolução entre (a) e (b).

O método aplicado no processo em tempo real é o de correlação cruzada, onde o coeficiente de correlação normalizado é calculado.

Existe um banco de dados para os logos, que guarda além do próprio logo, informações sobre em quais cantos deve ser feita à pesquisa, o código da emissora representada por este logo no sistema de audiência, além do nome desse logo. Então, cada logo é pesquisado na área total onde pode ser encontrado.

3.2 Resultados Experimentais

Apresentamos os resultados experimentais obtidos aplicando o algoritmo desenvolvido para os três tipos de logos. Um objetivo comum às três versões do algoritmo criado é testar o método proposto com vídeos off-line, e utilizando uma implementação com DSP. O teste do algoritmo em vídeos off-line utilizando o Proeikon permite avaliar problema, enquanto em uma implementação com DSP dá sustentação ao método, uma vez que trabalhamos em tempo real com qualquer tipo de imagem. Utilizamos as características de operação do sistema de medição de audiência para nos informar se os resultados obtidos estão dentro de um padrão de qualidade para um sistema de medição de audiência em tempo real.

Para qualquer algoritmo testado, sempre teremos quatro situações distintas: verdadeiro positivo, falso positivo, falso negativo e falsa identificação, lembrando que os resultados que serão demonstrados quantificam a taxa de quadros verificados.

1. **Verdadeiro positivo:** quando um logo foi corretamente detectado no quadro testado;
2. **Falso positivo:** quando um logo foi incorretamente detectado no quadro testado, pois não havia qualquer logo a ser identificado na ocasião;
3. **Falso negativo:** há um logo no vídeo, porém o algoritmo não o detectou.
4. **Falsa identificação:** quando houve a detecção de um logo no quadro verificado, porém, a emissora encontrada não corresponde ao logo do vídeo em questão;

Existem dois fatores críticos para um Sistema de Medição de Audiência de Televisão:

- falsa identificação, pois gera uma falsa audiência, e;
- o canal deve ser identificado dentro do período de 1 (um) minuto. Limite de tempo máximo para se detectar um canal no Sistema de Tempo Real, atualmente, (IBOPE, 2006).

3.3 Versão 1 do Algoritmo

3.3.1 Filtragem

Conforme demonstrado na figura 3.4, a variação do fundo em transmissões onde a emissora utiliza um logo semitransparente como marca d'água é o fator que dificulta o processo de reconhecimento.

(Hargrove, 2006) utiliza uma técnica de filtragem, na qual o cálculo do mínimo da imagem digital é considerado. Ele se baseia na hipótese de que os pixels contidos dentro do logo são mais claros que os pixels fora do logo. Esta idéia não pode ser considerada como um padrão de geração dos logos semitransparentes, uma vez que cada emissora detém uma técnica para melhor fixar a sua marca perante o expectador, não sendo ela obrigada a seguir alguma regra pré-estabelecida, ou norma ABNT.

Embora o fundo esteja variando, há informações que na maioria das vezes estão destacadas no logo. Estas informações são as bordas que sofrem um tipo de tratamento para poder se realçar e imprimir na imagem a marca da emissora. Cada emissora tem uma forma de ressaltar as bordas do seu logo.

(Albiol, 2004) utilizou o conceito de média móvel temporal para suavizar mudanças abruptas na imagem, ao mesmo tempo em que ressalta pixels que não variam ou há uma pequena variação no tempo. Utilizamos este conceito em nosso trabalho, implementando uma média móvel temporal de peso *0,5*. Matematicamente temos:

$$\overline{F}(j,k,t) = \frac{1}{2}\left[\overline{F}(j,k,t-\Delta t) + F(j,k,t)\right],\ t \geq 0 \qquad (3.3).$$

Na equação (3.3) verificamos que a taxa de execução da média móvel temporal depende de Δt. Para este algoritmo Δt vale 30 (trinta) quadros, ou 1(um) segundo. $\overline{F}(j,k,t)$ é o pixel (j, k) do pacote de vídeo no instante *t*, que sofreu o processo de média móvel temporal. Note que o quadro resultante no instante *t* com peso 0,5, é fruto da média ponderada *t*-Δ*t* com peso 0,25, o quadro *t-2*Δ*t* com peso 0,125 e desta maneira, a progressão contínua. A idéia é que a cada vez que o processo é feito, mais objetos vão se tornando borrados, deixando somente as

bordas do logo ressaltadas, melhorando assim o processo seguinte de detecção de bordas e por conseqüência dando maior robustez e segurança a pesquisa de logos.

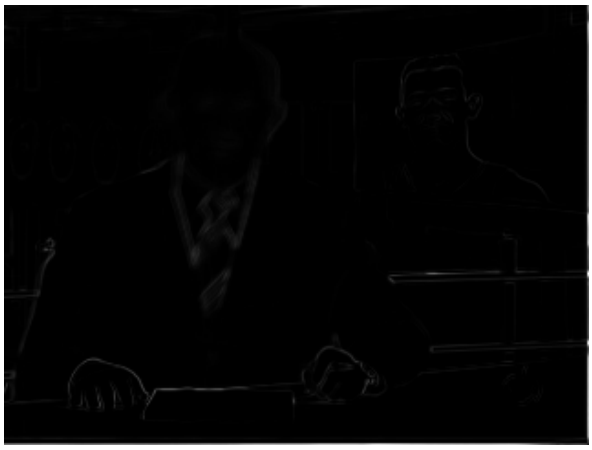
3.3.2 Detecção de Bordas

Há inúmeros exemplos de algoritmos utilizados para detecção de bordas na literatura, (Gonzalez, 2002). Em nosso estudo, testamos alguns algoritmos, os quais foram expostos nas secções anteriores, avaliando a capacidade de detecção de bordas. Para analisar as imagens geradas pelos detectores analisados, veja a figura 3.6. Nesta figura é possível verificar alguns algoritmos como Prewitt e Sobel oferecem um bom detector. Porém, de uma maneira geral, analisados diversos vídeos, o algoritmo Prewitt se mostrou mais eficaz, gerando maior enfatização em detalhes suaves da imagem, pois alguns logos não apresentam variações abruptas de nível de cinza, o que torna a detecção de bordas uma tarefa mais complexa.

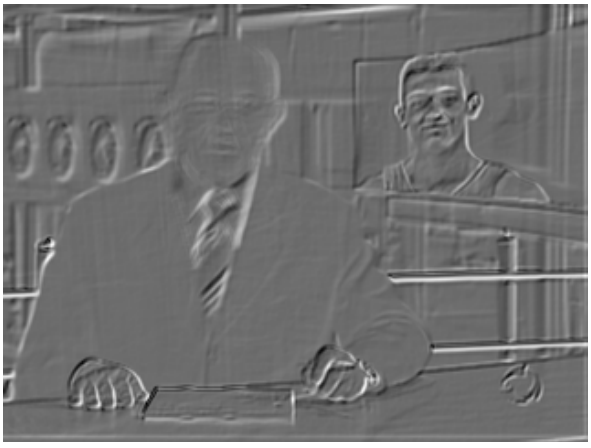
A detecção de borda pode ser aplicada a magnitude da imagem $\overline{F}$ passada pelo processo de média móvel temporal, ou quadro a quadro, sendo que o processo de média móvel temporal pode ser aplicado em uma imagem resultante do detector de borda. Para reduzirmos o processamento exigido, utilizamos a primeira idéia, ou seja, após o processo de média móvel temporal, o filtro detector de bordas é aplicado a esta imagem. Matematicamente temos que:

$$G(x,y) = \left\| \nabla \overline{F}(x,y) \right\| = \left\{ \left[\frac{\partial \overline{F}(x,y)}{\partial x} \right]^2 + \left[\frac{\partial \overline{F}(x,y)}{\partial y} \right]^2 \right\}^{1/2} \qquad (3.4).$$

(a)



(b) (c)



(d) (e)

Figura 3.6 – (a) imagem gerada pelo processo de time-averaging; (b) detector de bordas tipo Prewitt; (c) detector de bordas tipo Roberts; (d) detector de bordas tipo Sobel; (e) detector de bordas tipo Laplaciano.

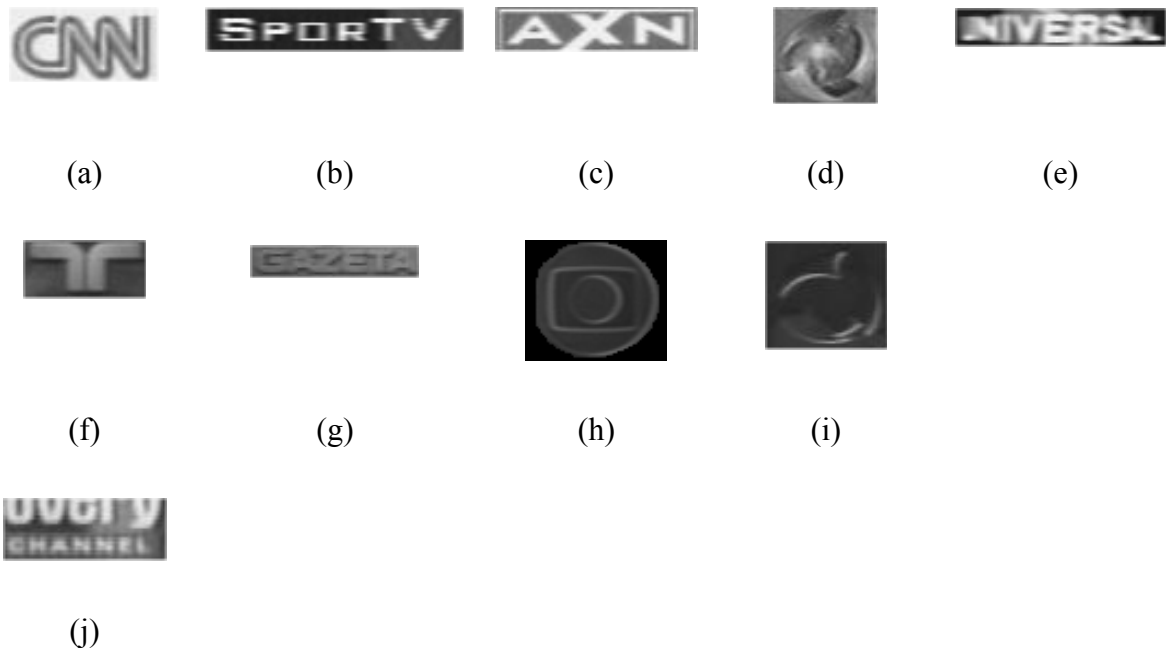
3.3.3 Pesquisa por Logos

Empiricamente nós estimamos o valor do limiar de match para 0,73 que não produziria falsos alarmes, critério importante em medição de audiência de televisão; e gera reconhecimento para todos os logos testados.

3.3.4 Resultados Experimentais – Emissoras avaliadas

O banco de logos continha 10(dez) emissoras, que são mostradas na figura 3.7, sendo:

- 5(cinco) logos opacos das emissoras: CNN, SporTV, AXN, Record e Universal;
- 4(quatro) logos semitransparentes das emissoras: Cultura, Gazeta, Globo e Record;
- e, 1(um) logo animado da emissora Discovery Channel.



(a) (b) (c) (d) (e)

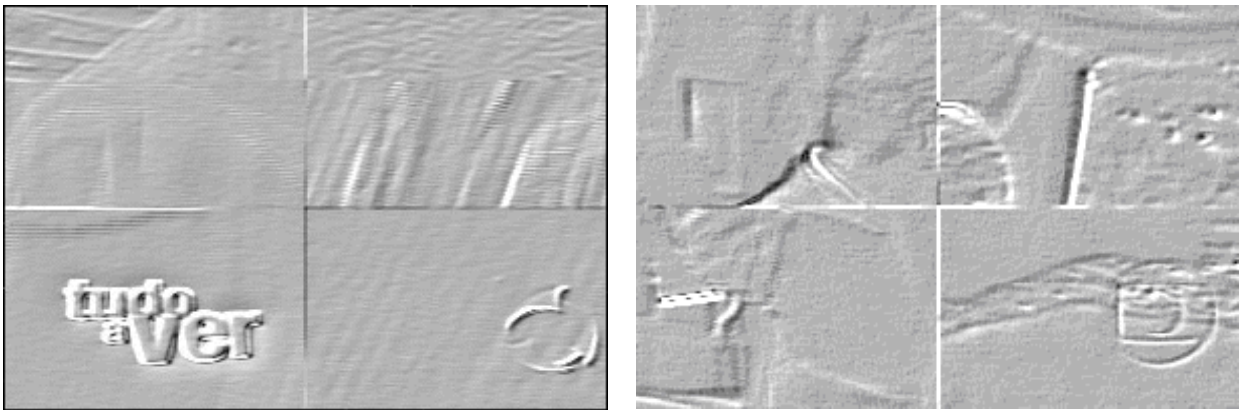
(f) (g) (h) (i)

(j)

Figura 3.7 – Imagens de (a) a (e) representam os logos opacos; imagens de (f) a (i) representam os logos semitransparentes e a imagem (j) representa o logo animado em sua parte opaca.

É visualmente perceptível a grande diferença existente entre logos opacos e semitransparentes. A evidência está bem ressaltada para o caso da emissora Record, vide figura 3.7, onde a imagem (d) representa o logo da emissora em uma transmissão ao vivo, enquanto a imagem (i) representa a mesma emissora, porém em uma transmissão previamente gravada.

A figura 3.8 mostra uma imagem capturada após a detecção da borda ser aplicada, pode-se ver que objetos estáticos foram bem enfatizados enquanto pixels que variavam ao longo do tempo foram borrados. O processo é falho, já que em vídeos com objetos estáticos, ou que variem pouco com o tempo, podem dificultar o reconhecimento ou até mesmo impossibilitar o mesmo. Veja exemplo na figura 3.8(b), onde o logo da emissora Globo contém um ruído devido a pixels que variam pouco com o tempo.



(a) (b)

Figura 3.8 – Na imagem (a) vemos o logo da emissora Record; na imagem (b) vemos o logo da emissora Globo. Imagens obtidas de conteúdos capturados por uma fonte de vídeo.

3.3.5 Resultados Experimentais - Testes em Tempo Real

A tabela 3.1 apresenta a taxa de quadros testados para cada tipo de teste, lembrando que são quatro: verdadeiro positivo, falso positivo, falso negativo e falsa identificação. Os testes foram realizados com a aquisição de vídeos veiculados durante mais de 24 (vinte e quatro) horas, o que significa mais de 86400 (oitenta e seis mil e quatrocentos) quadros testados.

	Verdadeiro Positivo	Falsa Identificação	Falso Positivo	Falso Negativo
Percentual (%)	65%	0%	0%	35%

Tabela 3.1– Métricas de Desempenho

Os resultados encontrados nos forçaram a gerar uma nova versão para tentar reduzir a alta taxa de falsos negativos, muito embora para a audiência a cada 1 (um) minuto não foi encontrada uma taxa significativa, pois a mesma ficou em torno de 0,1%.

3.4 Versão 2 do Algoritmo

O objetivo da criação deste algoritmo é buscar reduzir a taxa de erros no critério falso negativo, sem prejudicar os outros critérios, além de melhorar a performance. Continuamos com a visão de ter um único processo que detecte os 3 (três) tipos de logos: opaco, semitransparente e parcialmente animado.

As informações de cores continuam sendo descartadas, e o enfoque principal deste novo estudo é melhorar a filtragem, com o intuito de descartar as alterações do fundo da imagem pelo melhor uso da técnica de média móvel temporal. Estudou-se melhor a pesquisa dos logos, com isso o tempo gasto para realizar esta tarefa foi reduzido.

3.4.1 Formação de uma imagem

Empiricamente, concluímos que o tamanho dos cantos criados na versão 1, descrito acima, poderia ser diminuído. Com isso a intenção é reduzirmos o uso de memória e o processamento envolvido no tratamento da imagem. Nesse novo algoritmo cada canto tem 64 (sessenta e quatro) linhas por 128 (cento e vinte e oito) colunas. Continuamos com quatro cantos por imagem para pesquisar o logo. A figura 3.9 ilustra uma imagem formada por este novo tamanho dos cantos.

Figura 3.9 – Novo tamanho de imagem gerada. Imagem adquirida de veiculação da emissora Globo.

3.4.2 Filtragem

Na primeira versão do algoritmo o peso aplicado a média móvel temporal era 0,5 (meio), e o intervalo entre quadros (Δt) era de 30 (trinta) quadros. Como no sistema brasileiro de televisão temos 30 (trinta) quadros por segundo, o quadro de 4 (quatro) segundos atrás teria o peso de apenas 0,0625 (seiscentos e vinte e cinco milésimos), mostrando que variações abruptas de imagem sobrepondo a área do logos fariam com que um falso negativo ocorresse.

A idéia desta versão é utilizar uma melhor relação entre o peso da média móvel temporal e o intervalo entre quadros para que não seja necessário utilizar um detector de bordas como foi feito na primeira versão do algoritmo, resultando em um melhor desempenho do sistema. A figura 3.10 demonstra os efeitos da alteração no peso para o mesmo vídeo, onde o Δt é de 10(dez) quadros por segundo.

3.4.3 Pesquisa por logo

O método de casamento de gabarito, “template matching”, utilizado na primeira versão do algoritmo continua sendo o método de pesquisa de um logo nesta versão.

A correlação cruzada é uma tarefa que consome grande parte do processamento da máquina. Para reduzir o tempo gasto em tal tarefa, há duas possibilidades: utilizar FFT para o cálculo, ou diminuir o número de pontos máximos a serem verificados.

O uso de FFT é válido quando o número de elementos no vetor é grande, ou quando se quer avaliar toda a área, no caso o canto inteiro. Então partirmos para avaliar a variação espacial de um logo em tempo real. Empiricamente verificamos que o logo pode variar em uma distância de 2 (dois) pixels em torno de um pixel central. Esta conclusão nos fez reduzir a área de pesquisa para apenas 25 (vinte e cinco) pixels, inviabilizando o uso de FFT e ao mesmo tempo reduzindo drasticamente o tempo de pesquisa para um logo.

3.4.4 Resultados Experimentais – Emissoras avaliadas

Esta versão do algoritmo, como descrito nas seções anteriores, leva em conta o fato de pixels pertencentes às bordas do logo variam pouco ou nada com o tempo. Portanto, dependendo do peso de um quadro anterior, pode-se tornar o logo semitransparente imune, ou bem resistível ao ataque de duração finita. (Albiol, 2004) utilizou a mesma técnica aplicada a detecção de comerciais pelos logos de televisão.

Na figura 3.10, há uma demonstração da robustez que se pode alcançar variando o peso dos quadros anteriores. Um vídeo da emissora Globo com logo semitransparente de aproximadamente 3 (três) minutos e 17 (dezessete) segundos foi gravado. Foram geradas algumas imagens de saída com diferentes fatores de média móvel temporal (time-averaging). O programa para tal demonstração foi construído utilizando a biblioteca Proeikon (Kim, 2006).

A implementação de um peso grande influencia o tempo para se reconhecer o primeiro logo, uma vez que para o filtro conseguir estabilizar a imagem, muitos quadros devem ser adquiridos e ruídos devido a objetos que fiquem presentes durante um grande período de tempo podem ocorrer. Como é o exemplo dado por alguns quadros adquiridos de uma transmissão da emissora Record de seu noticiário local, mostrado na figura 3.11. Há uma mesa que se torna parte presente do logo durante um grande espaço de tempo, em situações como essa, grandes fatores de média móvel temporal agem de duas maneiras:

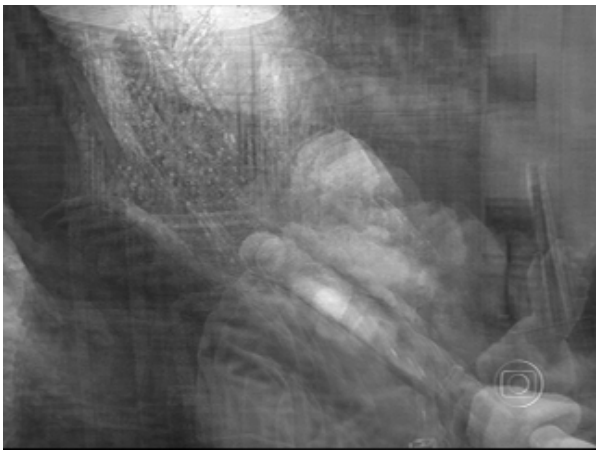
- se previamente o logo já havia se estabilizado, o ruído gerado por este objeto não irá atrapalhar o reconhecimento;

- caso o processo de estabilização não tenha sido finalizado, o reconhecimento será prejudicado.

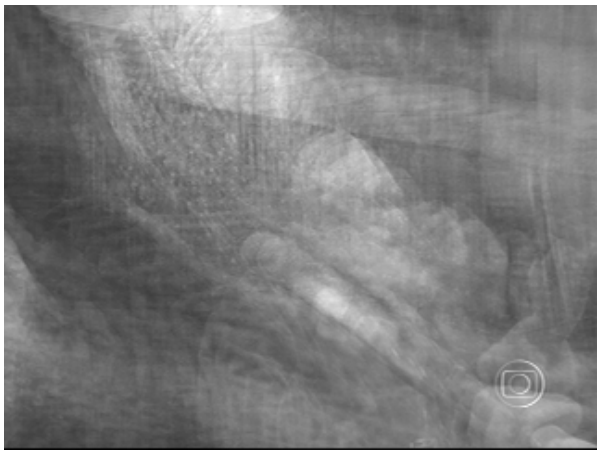
Com este algoritmo tentamos reduzir os altos índices de erro encontrado no critério Falso Negativo do algoritmo anterior.

(a)

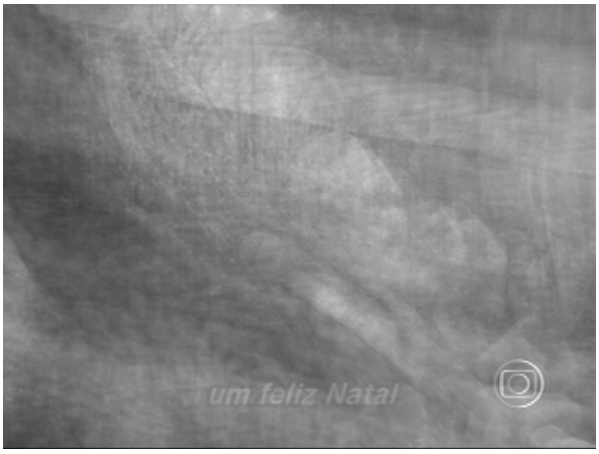
(b)



(c)



(d)



(e)

(f)

(g)

Figura 3.10 – A imagem (a) foi extraída com um fator de média móvel temporal de 5/6; a imagem (b) foi extraída com um fator de média móvel temporal de 7/8; a imagem (c) foi extraída com um fator de média móvel temporal de 15/16; a imagem (d) foi extraída com um fator de média móvel temporal de 31/32; a imagem (e) foi extraída com um fator de média móvel temporal de 63/64; a imagem (f) foi extraída com um fator de média móvel temporal de 127/128 e a imagem (g) foi extraída com um fator de média móvel temporal de 255/256.

(a)

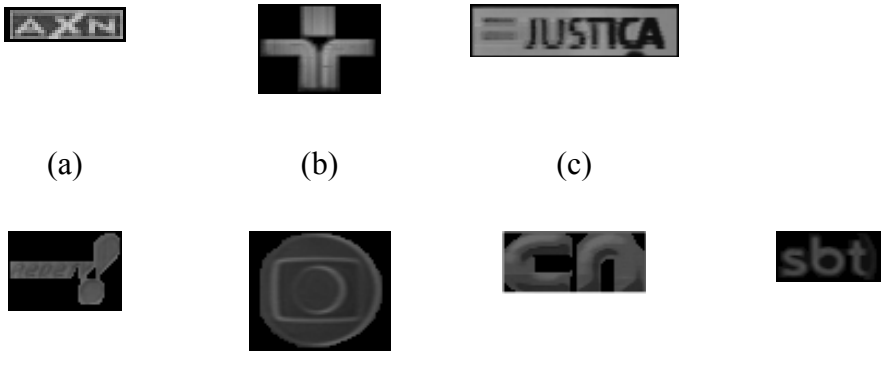
(b)

(c) (d)

Figura 3.11 – Quadros extraídos de um noticiário da emissora Record de televisão

O banco de logos continha 8 (oito) emissoras, que são mostradas na figura 3.12, sendo:

- 4 (quatro) logos opacos das emissoras: CNN, AXN, Record e Universal;
- 4 (quatro) logos semitransparentes das emissoras: Cultura, Gazeta, Globo e Record;
- e, 1 (um) logo animado da emissora Discovery Channel.



(a) (b) (c)

(d) (e) (f) (g)

(h)

Figura 3.12 – Imagens de (a) a (c) representam os logos opacos; imagens de (d) a (g) representam os logos semitransparentes e a imagem (h) representa o logo animado em sua parte opaca.

3.4.5 Resultados Experimentais - Testes em Tempo Real

A tabela 3.2 apresenta a taxa encontrada para o fator falso negativo, motivo pelo qual este algoritmo foi gerado. Os testes foram realizados com a aquisição de vídeos veiculados durante mais de 16 (dezesseis) horas, o que significa mais de 345600 (trezentos e quarenta e cinco mil e seiscentos) quadros testados. Para cada emissora o teste foi realizado durante 2 (duas) horas. Para este teste o peso dos quadros utilizado na média móvel temporal era $\sigma = 127/128$.

Tipo	Emissora	Falso Negativo em %
Opaco	*AXN*	0
	Cultura	1,01
	JUSTIÇA	7,63
Semitransparente	*RedeTV*	0,56
	Globo	2,53
	CNU	12,07
	SBT	0,41
Animado	*Discovery Channel*	0,02
Geral	*Não se aplica*	3,029

Tabela 3.2 – Desempenho da versão 2 do algoritmo para o fator de falso negativo

Para os demais fatores, verdadeiro positivo, falsa identificação e falso positivo, não encontramos mudanças significativas, como pode ser visto pela tabela 3.3. Somente o verdadeiro positivo, pois é inversamente proporcional ao fator falso negativo teve mudanças significativas.

Observando os dados coletados, as emissoras JUSTIÇA e CNU saltam aos olhos pelo alto índice de falso negativo encontrado, em relação aos demais, o fato ocorrido é que estas duas

emissoras contêm imagens que cintilam ao longo do tempo, gerando um ligeiro borrão e alterando a semente de pesquisa para as mesmas. O fato ocorre devido à baixa qualidade dos dois canais, uma vez que são de domínio público sem muita preocupação em sua parte técnica, e sim no conteúdo programático transmitido. Pode-se melhorar este índice aumentando a quantidade de pixels verificados para emissoras onde se sabe que tal fato pode ocorrer.

	Verdadeiro Positivo	Falsa Identificação	Falso Positivo	Falso Negativo
Percentual (%)	96,971%	0%	0%	3,029%

Tabela 3.3 – Métricas de desempenho

Em termos práticos a redução de 35% da versão 1 para 3,029% da versão 2 reduz a zero a não geração de audiência no intervalo de 1 (um) minuto, assim, não gerando buracos no reconhecimento de canais.

3.4.6 Resultados Experimentais - Testes em Tempo Real com variação do σ

O objetivo da criação de outro algoritmo se deve pelo alto índice de falsos negativos encontrados no algoritmo anterior. Como a solução encontrada está ligado ao fator de média móvel temporal aplicado, um novo teste foi criado para verificar a relação entre σ e falso negativo. Para isso foi escolhido um logo semitransparente, que é o pior caso para reconhecimento de logos, a emissora escolhida foi o SBT. Então alguns fatores σ foram testados para esta emissora. Cada teste tem a duração de 2 (duas) horas e relaciona o índice de falso negativo encontrado e a relação de tempo médio em que uma transição de logo não encontrado para logo encontrado ou vice-versa ocorre. Os valores encontrados podem ser observados na tabela 3.4.

O fator σ da média móvel temporal representa o nível de variação abrupta aceita. Quanto mais próximo de 1 (um) for este índice, menor é o peso atribuído a um quadro recém captado. Portanto o tempo para se reconhecer este logo está ligado a este fator, além dos detalhes de imagem também.

Fator de Média Móvel Temporal (σ)	Falso Negativo	Tempo de Reconhecimento
31/32	6,85%	Entre 1 e 2 segundos
63/64	7,98%	Entre 4 a 5 segundos
127/128	3,71%	No mínimo 8 segundos
255/256	0,41%	No mínimo 10 segundos

Tabela 3.4 – Comparação entre fatores σ.

A figura 3.13 demonstra a variação do coeficiente de correlação em duas situações: quando há o mesmo logo no vídeo e na comparação efetuada, e quando não há logo, ou o logo existente não é da emissora que está sendo comparada. Com isso efetuamos um histograma geral dos dados gerados.

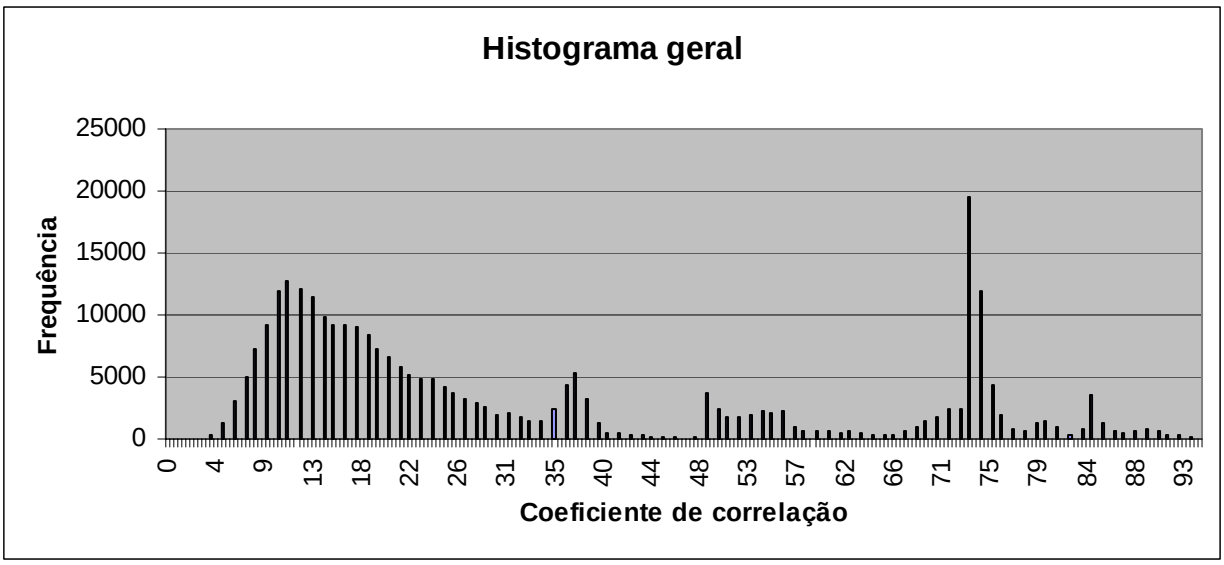


Figura 3.13 - Demonstra o histograma montado com os dados de verdadeiro positivo e falso positivo em conjunto entre todas as médias móveis computadas. Os valores do coeficiente de correlação podem variar de 0 a 100.

Como exemplo, a figura 3.14 traz o histograma individualizado de algumas emissoras testadas para a situação de verdadeiro positivo e falso positivo.

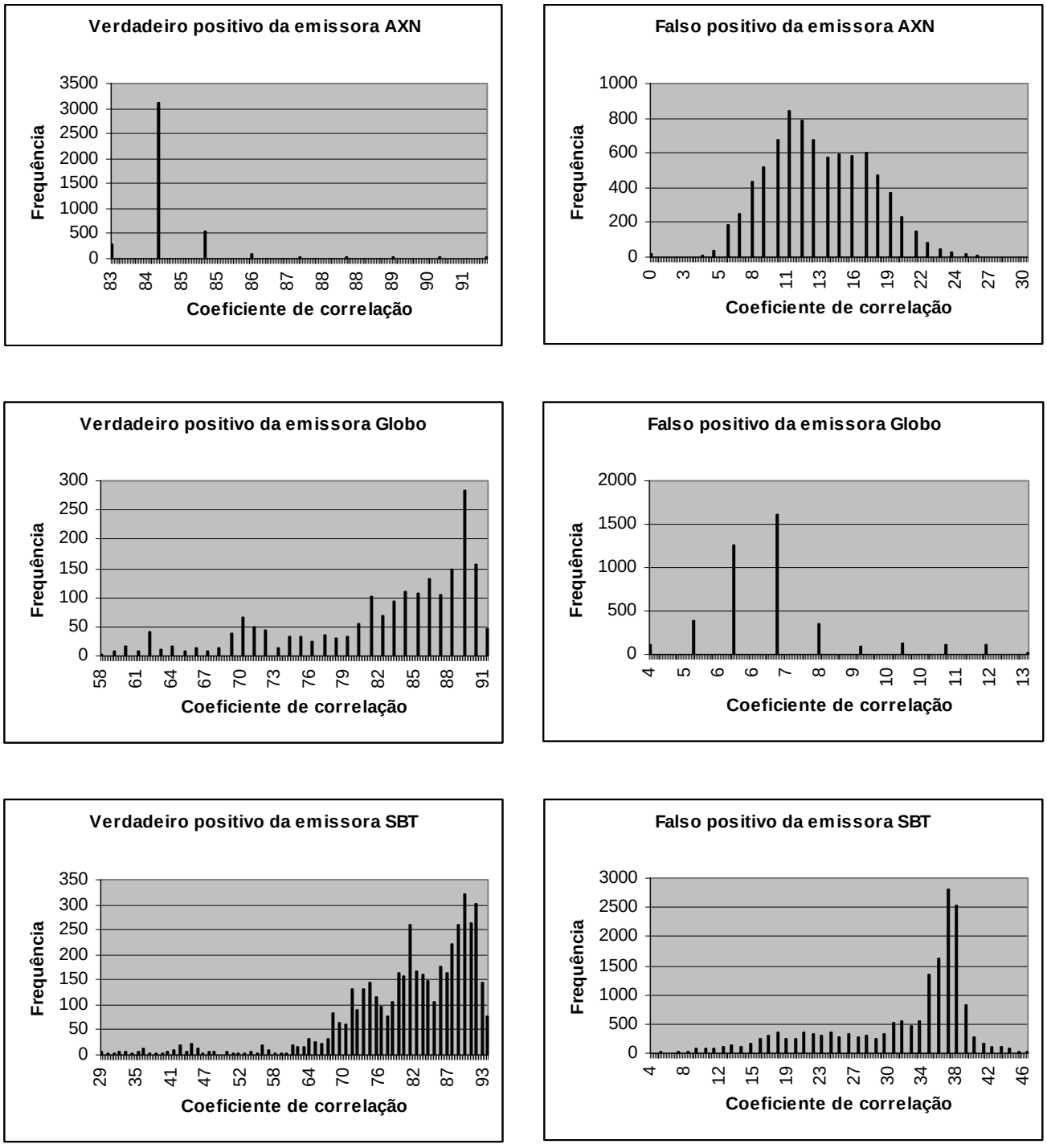


Figura 3.14 - As figuras a esquerda demonstram a variação da correlação quando o logo analisado aparecia no vídeo, enquanto a coluna da direita mostra a mesma variação para situações onde não havia o logo da emissora analisada. Os valores do coeficiente de correlação podem variar de 0 a 100.

Com esta versão do algoritmo, houve uma drástica redução na taxa de falsos negativos, vide tabela 3.1, assim, fica uma pergunta, o que aconteceria se mesclássemos as duas versões do algoritmo? Teríamos algo mais robusto a ataques? Estas perguntas serão respondidas com uma nova versão do algoritmo, onde mesclaremos a detecção de bordas com um fator elevado de média móvel temporal.

3.5 Versão 3 do Algoritmo

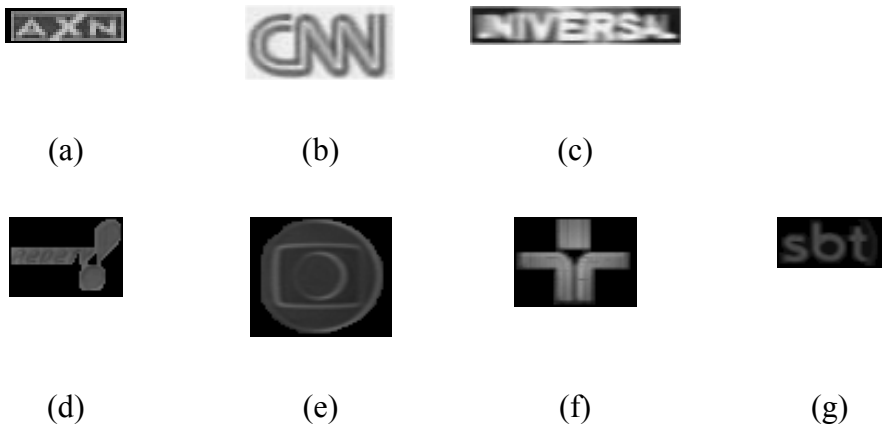
O objetivo desta versão é efetuarmos uma junção das duas versões anteriores do algoritmo e avaliarmos os resultados, sempre pensando em uma visão comparativa para respondermos as indagações geradas com a criação da segunda versão.

Para isso, os dois fatores principais de cada versão: detector de bordas e a variação da média móvel temporal, respectivamente, foram modificados e avaliados.

3.5.1 Resultados Experimentais – Emissoras avaliadas

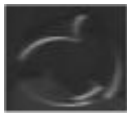
O banco de logos continha 11 (onze) emissoras, que são mostradas na figura 3.15, sendo:

- 3 (três) emissoras com logos opacos: AXN, CNN e Universal;
- 7 (sete) emissoras com logos semitransparentes: Cultura, Gazeta, Globo, Record, RedeTV e SBT, além de;
- 1 (um) logo animado.



(a) (b) (c)

(d) (e) (f) (g)



(h) (i)

(j)

Figura 3.15 - Imagens de (a) a (c) representam os logos opacos; imagens de (d) a (i) representam os logos semitransparentes e a imagem (j) representa o logo animado em sua parte opaca.

3.5.2 Resultados Experimentais - Testes em Tempo Real

Foram realizados testes nas 11 (onze) emissoras mostradas na figura 3.15, com um tempo de aproximadamente 3 (três) horas para cada emissora. As médias móveis temporais utilizadas (σ) foram: 7/8, 15/16, 31/32, 63/64 e 127/128. A média móvel 3/4 foi descartada por ter sido utilizada na primeira versão do algoritmo.

Com a intenção de se gerar um limiar de separação entre duas situações distintas de uma maneira empírica, os dados capturados por mais de 33 (trinta e três horas) para verdadeiro positivo e falso negativo foi gerado um histograma, onde se verifica a existência de duas regiões distintas, o que mostra por dados reais, uma possível região de litígio e um limiar de separação entre o verdadeiro positivo e o falso positivo que minimiza a taxa de quadros não reconhecidos (falso negativo). A figura 3.16 ilustra o histograma comentado.

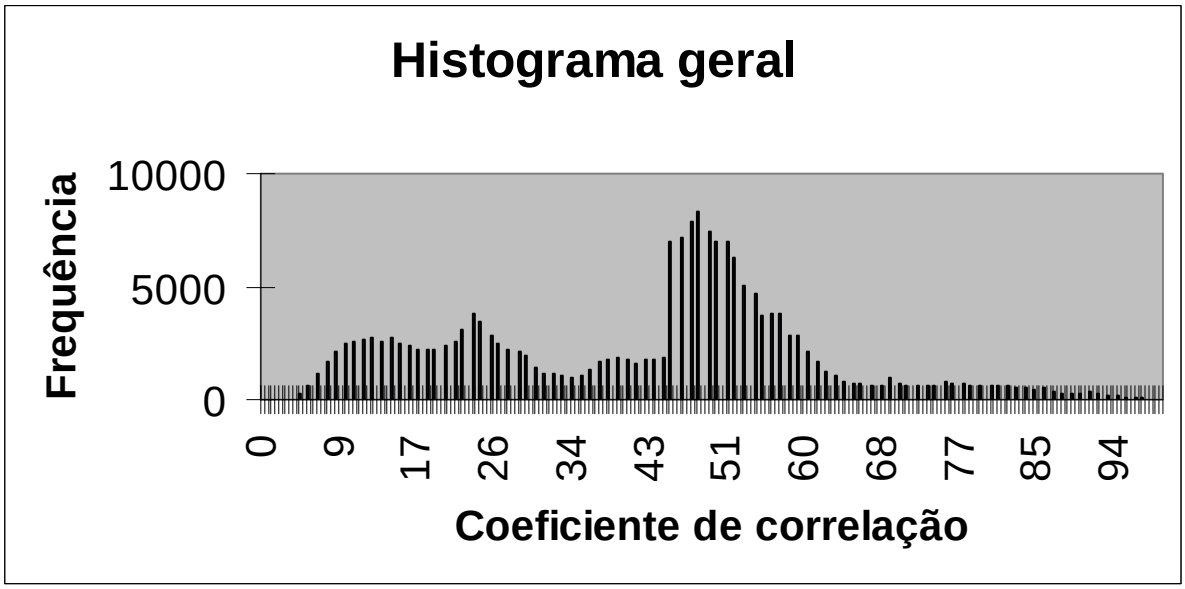


Figura 3.16 – Demonstra o histograma montado com os dados de verdadeiro positivo e falso positivo em conjunto entre todas as médias móveis computadas. Os valores do coeficiente de correlação podem variar de 0 a 100.

Escolhendo um limiar de 34% obtêm-se a tabela 3.5 para o desempenho da versão no critério falso negativo, fator que se tenta minimizar. Os fatores são médios entre todos os coeficientes de média móvel analisados.

Tipo	**Emissora**	**Falso Negativo em %**
Opaco	*AXN*	0
	CNN	0,22
	Universal	2,87
Semitransparente	*Cultura*	18,27
	Gazeta	3,75
	Globo	5,98
	Record	22,31
	RedeTV	10,28
	SBT	22,39
	TeleCine Premium	0,53
Animado	*Discovery Channel*	0,08
Geral	*Não se aplica*	7,88

Tabela 3.5 – Desempenho médio para o critério falso negativo utilizando um limiar de 34%

Pode-se otimizar os resultados pela análise do coeficiente de correlação encontrado ao longo do tempo, pois, há grande variedade no histograma encontrado para cada emissora. As figuras 3.17, 3.18, 3.19 e 3.20 ilustram o histograma encontrado para as 11 (onze) emissoras em dois

contextos: onde havia o logo da emissora no momento do teste; e, em outro contexto onde ou não havia o logo da emissora, ou havia algum outro logo qualquer. Para a comparação, o histograma foi gerado com a mesma média móvel temporal, 127/128.

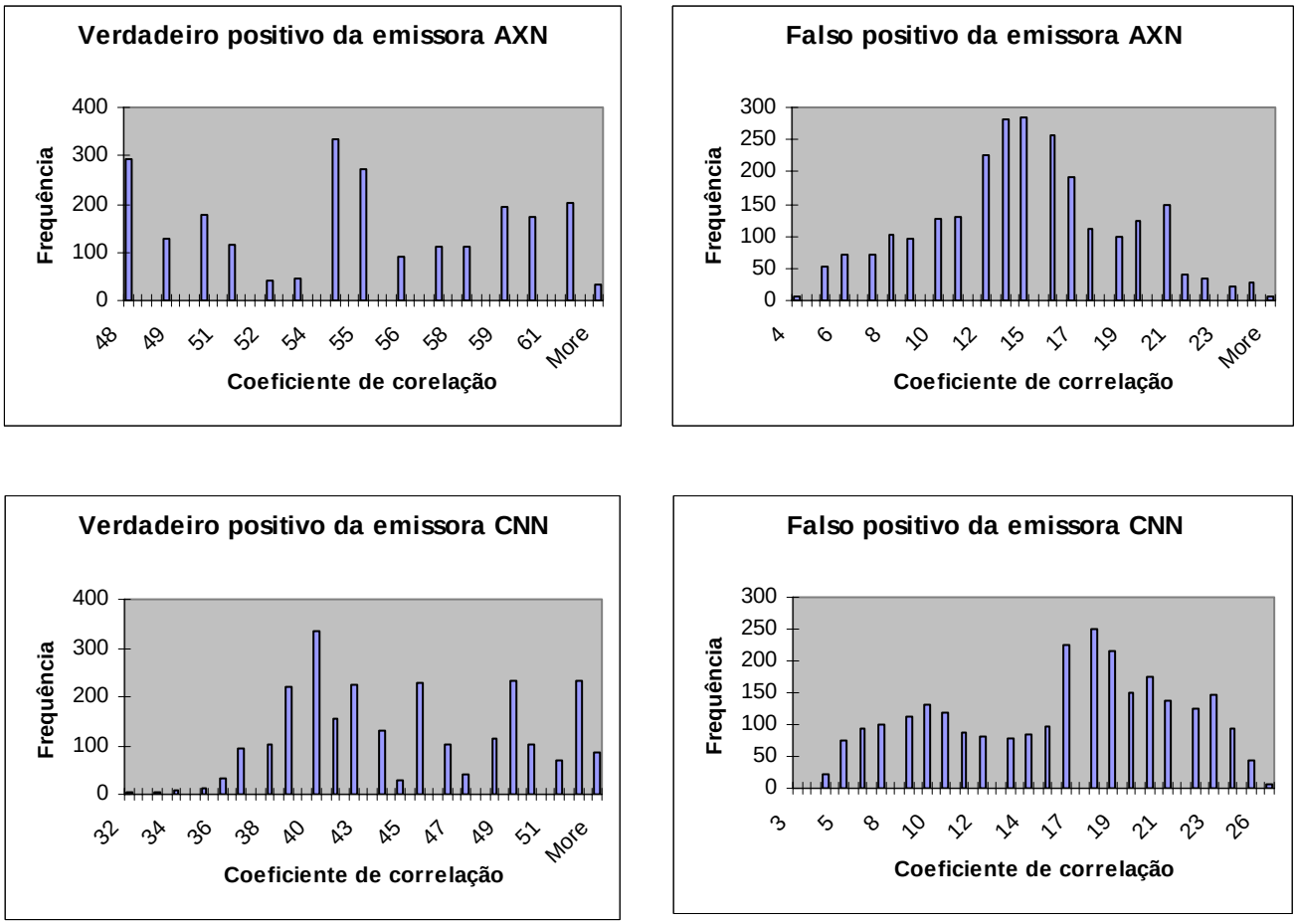


Figura 3.17 – As figuras a esquerda demonstram a variação da correlação quando o logo analisado aparecia no vídeo, enquanto a coluna da direita mostra a mesma variação para situações onde não havia o logo da emissora analisada. Os valores do coeficiente de correlação podem variar de 0 a 100.

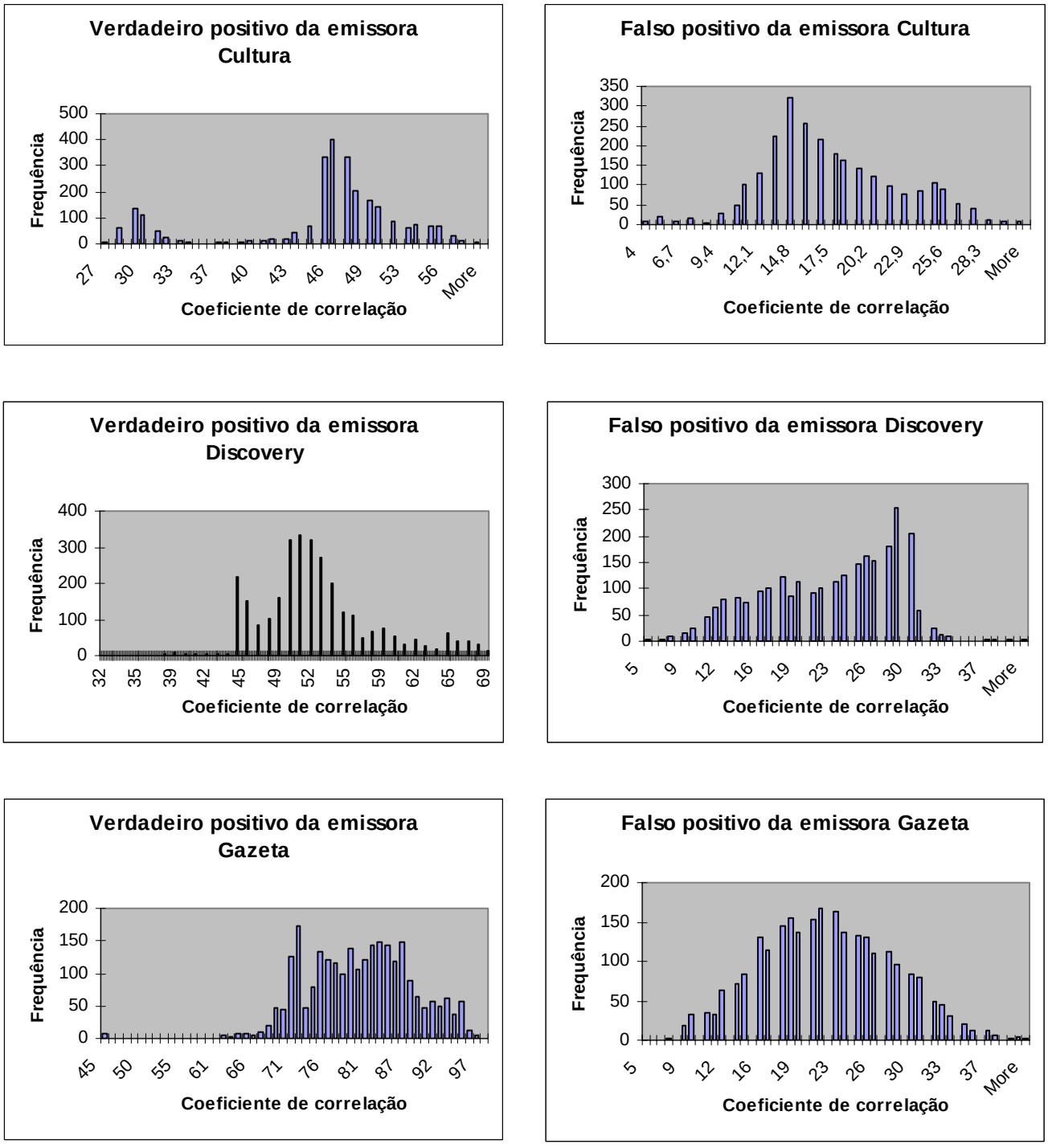


Figura 3.18 – As figuras a esquerda demonstram a variação da correlação quando o logo analisado aparecia no vídeo, enquanto a coluna da direita mostra a mesma variação para situações onde não havia o logo da emissora analisada. Os valores do coeficiente de correlação podem variar de 0 a 100.

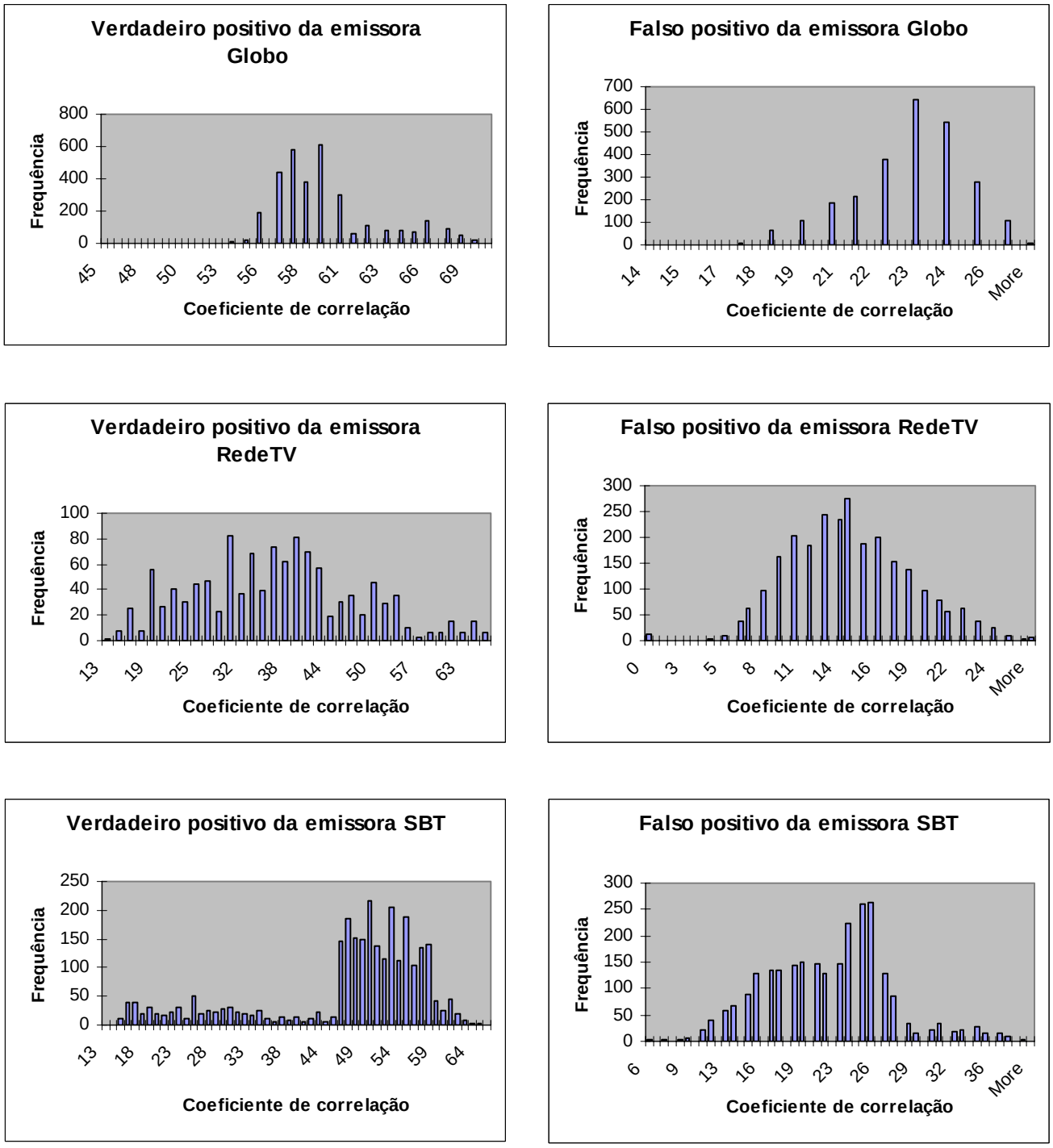


Figura 3.19 – As figuras a esquerda demonstram a variação da correlação quando o logo analisado aparecia no vídeo, enquanto a coluna da direita mostra a mesma variação para situações onde não havia o logo da emissora analisada. Os valores do coeficiente de correlação podem variar de 0 a 100.

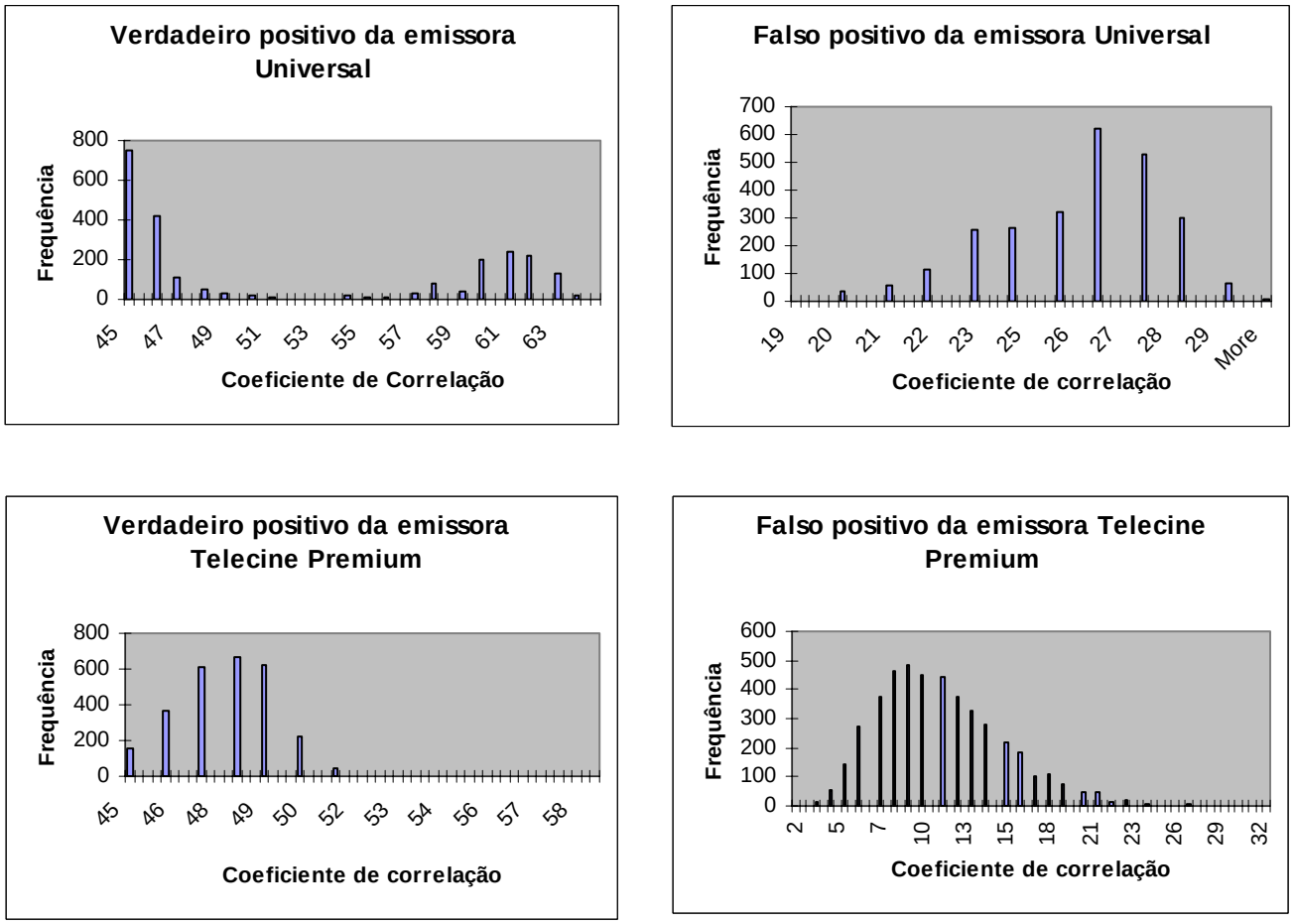


Figura 3.20 – As figuras a esquerda demonstram a variação da correlação quando o logo analisado aparecia no vídeo, enquanto a coluna da direita mostra a mesma variação para situações onde não havia o logo da emissora analisada. Os valores do coeficiente de correlação podem variar de 0 a 100.

A análise de tempo de reconhecimento efetuado na versão anterior, importante aspecto que mede o tempo de resposta ao estímulo (aparecimento do logo), também foi feita. Pelos dados coletados, verificasse, na tabela 3.6, que há uma resposta mais rápida, mesmo com médias móveis temporais mais elevadas.

Fator de Média Móvel Temporal (σ)	Tempo de Reconhecimento
7/8	1 segundo
15/16	1,36 segundos
31/32	2,18 segundos
63/64	3,72 segundos
127/128	5,5 segundos

Tabela 3.6 – Tempo de resposta ao primeiro logo, versus a média móvel temporal.

Podemos concluir, ainda que de uma maneira empírica, a eficácia desta versão, muito embora se a análise for feita encima do fator de falso negativo, veremos que a versão anterior possui uma melhor performance nesse quesito. Porém, como o processamento da audiência em tempo real no Brasil é computado de minuto a minuto (IBOPE), podemos utilizar esta versão, uma vez que a mesma tem a resposta mais rápida aliada a uma boa robustez.

A tabela 3.7 apresenta as taxas para os fatores, verdadeiro positivo, falsa identificação, falso positivo e falso negativo para uma análise global, continuamos não encontramos problemas nos erros de falsa identificação e falso positivo para o limiar adotado, como pode ser visto pela tabela 3.7, além de continuarmos com um erro de audiência muito baixo, aproximadamente 0,01% em 1 (um) minuto para o reconhecimento.

	Verdadeiro Positivo	Falsa Identificação	Falso Positivo	Falso Negativo
Percentual (%)	92,12%	0%	0%	7,88%

Tabela 3.7 – Métricas de desempenho para a versão 3.

3.6 Levantamentos Estatísticos

Até o presente momento, todos os limiares de separação entre imagens que possuam logos de outras que não os possuem, além de provas de funcionamento do algoritmo foram retiradas de maneira empírica. A fim de comprovarmos estatisticamente a eficácia do algoritmo, além de estimarmos um valor eficiente para o coeficiente de correlação, iremos efetuar algumas provas, como a curva de ROC e o teste de hipótese t-Student.

3.6.1 A Curva ROC

Uma curva ROC é um gráfico de sensibilidade versus especificidade quando algum parâmetro do método de detecção varia (METZ, 1978), em nosso caso, o parâmetro utilizado é o coeficiente de correlação cruzada. As curvas ROC foram desenvolvidas durante a segunda guerra mundial para determinar o rendimento de radares, a capacidade de diferenciar entre sinais verdadeiros e falsos.

A curva ROC demonstra vários aspectos importantes:

1. Mostra a relação entre sensibilidade e especificidade;
2. Quanto mais perto da borda superior, mais preciso é o teste;
3. Quanto mais próximo da diagonal de 45º (quarenta e cinco graus) do espaço da curva ROC, menos preciso é o teste.

Com os dados gerados pelas versões 2 e 3 do algoritmo foi possível gerar a curva ROC da figura 3.18, utilizando os coeficientes de correlação cruzada para os pontos de corte situados em: 32%, 34%, 40%, 51%, 75% e 94%.

Analisando a curva, e a tabela 3.8, onde a área sob a curva foi calculada, vemos que a mesma comprova os dados percentuais levantados empiricamente para as duas versões, mostrando a maior confiabilidade encontrada para a versão 2 em comparação com a versão 3, devido a sua maior área em relação a diagonal principal da curva. Pode-se notar que, um limiar de separação entre verdadeiro positivo e falso positivo que minimize os falsos negativos pode ser definido sem prejudicar a audiência gerada.

O limiar de decisão pode ser escolhido levando-se em conta a taxa de falsos negativos que serão gerados, por exemplo, para a curva da versão 2, sob análise na figura 3.22, formada pela região com a concentração de pontos da figura 3.21 ampliada, pode-se verificar que, se admitirmos um valor para o limiar de decisão do coeficiente de correlação em 75%, garante-se uma taxa de falso positivo inferior a 0,1%, porém, aproximadamente 35% dos quadros serão considerados sem logo. Como para o algoritmo gerado analisa-se em média 20 quadros

e a medição de audiência brasileira é feita com intervalos de 1 (um) em 1 (um) minuto, pode-se escolher este limiar, minimizando o fator mais importante em um sistema de medição de audiência, que é a falsa identificação e ainda assim, garantindo a correta geração da audiência no intervalo requerido.

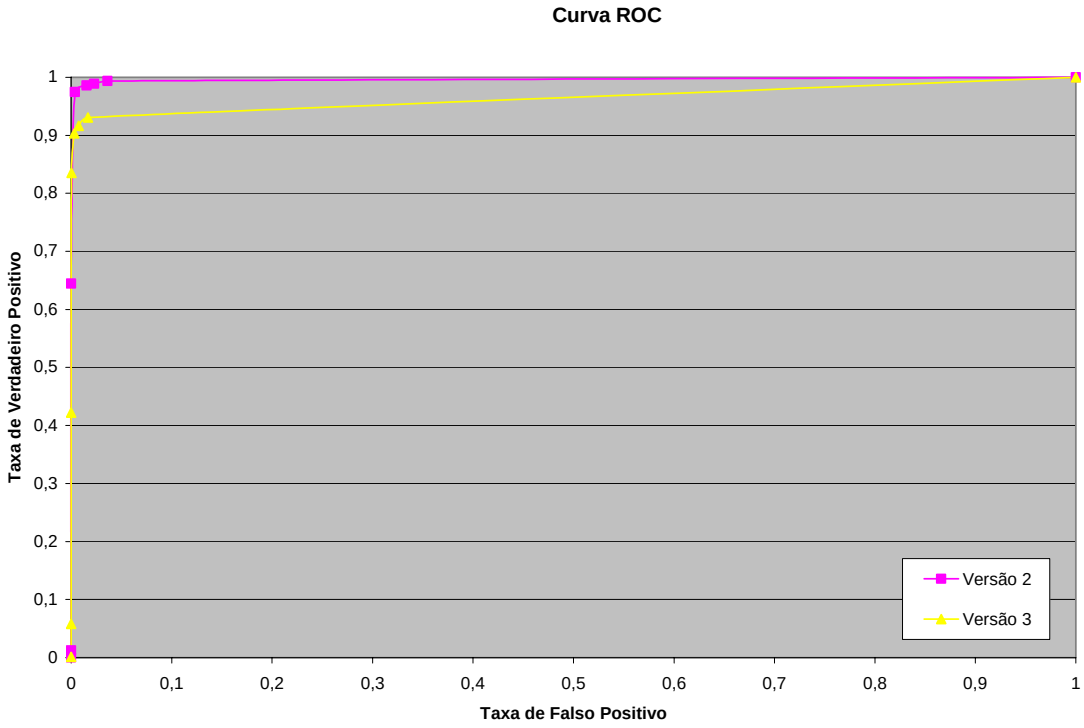


Figura 3.21 – Curva ROC de comparação entre as versões 2 e 3 do algoritmo.

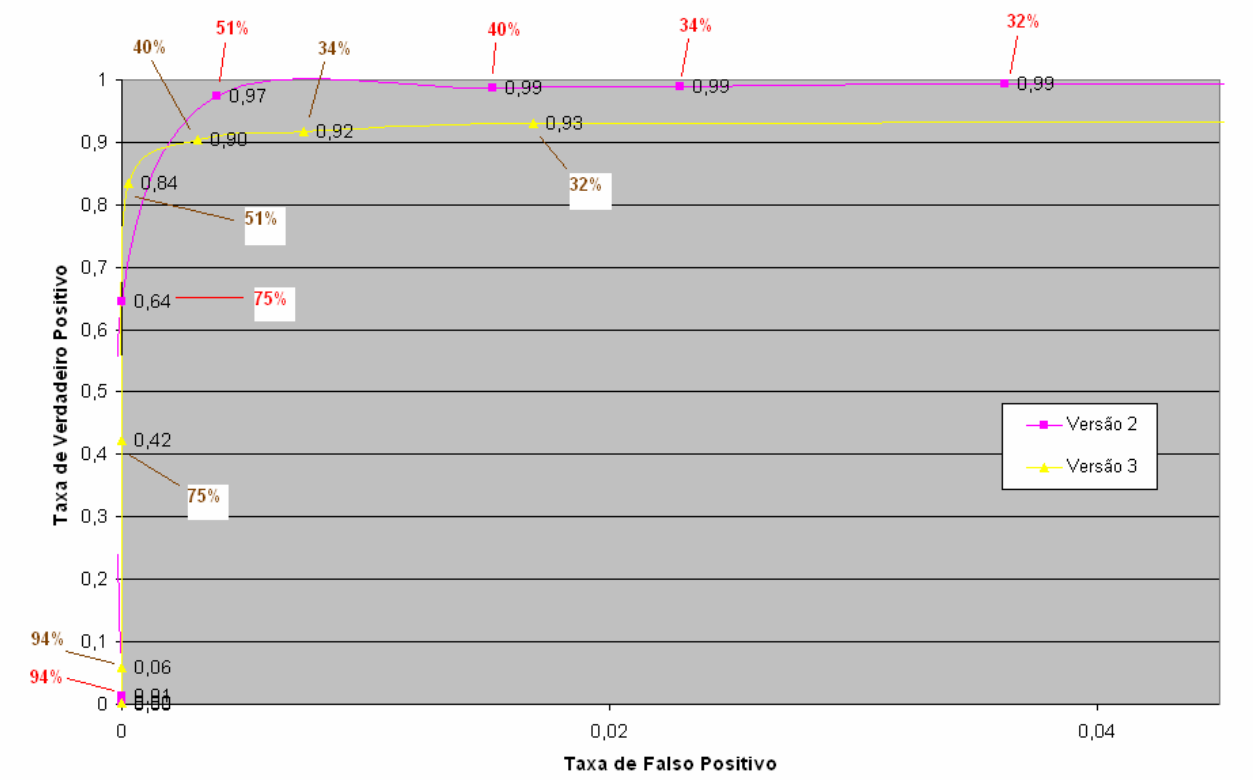


Figura 3.22 – Curva ROC da figura 3.21 com área de maior interesse ampliada

	Área da curva
Versão 2	0,964751
Versão 3	0,881971

Tabela 3.8 – Área sob as curvas ROC da figura 3.21.

3.4.2 Teste de Hipótese t-Student

O fator mais crítico em um sistema de medição de audiência é a falsa detecção, ou seja, dar audiência erroneamente a uma emissora em lugar de outra. Por isso, a comprovação estatística da robustez ao processo, reforça os dados empíricos se, por exemplo, o sistema for auditado.

Para se assinalar que uma imagem contém o logo de uma emissora, o coeficiente de correlação cruzado é comparado contra um limiar de decisão, caso este coeficiente esteja acima de um determinado patamar, admitimos que o canal do logo analisado foi encontrado. Ou seja, temos que verificar se para o máximo coeficiente de correlação (ρ_{max}) entre uma imagem qualquer que não contenha um logo existente na base de dados e vários logos, obtemos um valor superior a λ que determine uma verdadeira identificação, ou não.

A fim de se testar a hipótese, o coeficiente de correlação máximo (ρ_{max}), encontrado pelo uso da equação 2.25, para os 25 (vinte e cinco) pixels em torno da semente inicial entre a imagem $\widetilde{G}$ e o logo $\widetilde{L}_i$ é convertido em uma estatística τ t-Student. Então a hipótese "não existe um logo $\widetilde{L}_i$ localizado na imagem na imagem $\widetilde{G}$ no pixel (m,n), ambos definidos na equação (3.1)", pode ser estatisticamente testada para o ponto (m,n) onde foi encontrado a maior semelhança entre as imagens analisadas.

Os pixels contidos na imagem $\widetilde{G}$ são colocados em um vetor de uma dimensão Y, igualmente, os pixels do logo $\widetilde{L}_i$ são também vetorizados da mesma maneira de forma a criar um vetor unidimensional X. Então, o objetivo é estimar o parâmetro β que minimiza o erro quadrático ε^2 na seguinte equação:

$$\begin{bmatrix} Y_1 \\ \vdots \\ Y_N \end{bmatrix} = \begin{bmatrix} X_1 \\ \vdots \\ X_N \end{bmatrix} \beta + \begin{bmatrix} \varepsilon^2{}_1 \\ \vdots \\ \varepsilon^2{}_N \end{bmatrix} \tag{3.5}.$$

A equação 3.5 também é usualmente denotada em sua forma matricial, como:

$$Y = \beta X + \varepsilon^2 \tag{3.6}.$$

O parâmetro β que minimiza o erro quadrático ε^2 pode ser estimado pelo procedimento dos mínimos quadrados:

$$\beta = \frac{XY}{X^2} \tag{3.7}.$$

Transforma-se o parâmetro β em uma estatística τ t-Student computando-se:

$$\tau = \frac{\beta}{\sqrt{\frac{\varepsilon^2}{X^2(n-1)}}} \tag{3.8}.$$

Para grandes valores de *n*, acima de 120 (Kazmier, 1982) pode-se aproximar a estatística t-Student pela estatística normal, com isso encontrar a probabilidade *p* dado o valor encontrado da estatística τ, diretamente com o auxílio da tabela de distribuição de probabilidade de uma curva normal (Kazmier, 1982). A estatística τ obtida é utilizada para se efetuar o teste de hipótese.

Assumindo que a hipótese nula H_0 indica a **não** correlação entre Y (imagem $\widetilde{G}$) e X (a imagem do logo $\widetilde{L}_i$, transladado a posição (m,n)), nós gostaríamos de saber quanto precisa é a nossa medida τ. O teste de hipótese nos permite analisar uma comparação entre o valor τ obtido e o valor τ_α, correspondente ao nível de significância α selecionado, assim, aceitando ou rejeitando a hipótese nula se $\tau < \tau_\alpha$ ou $\tau \geq \tau_\alpha$, respectivamente.

A conclusão sobre a rejeição ou aceitação da hipótese é feito pela comparação da probabilidade *p* para a estatística τ contra o nível de significância α admitido, caso o valor *p* encontrado for menor que α pode-se rejeitar a hipótese H_0.

Para testar a teoria acima, utilizamos três grupos de imagens, com 13 (treze) exemplos em cada um, gerando um total de 39 (trinta e nove) testes efetuados. Os três tipos são:

1. *Verdadeira identificação:* logos da imagem e do banco são da mesma emissora (linhas de 1 a 13 na tabela 3.9);

2. *Falsa identificação:* logos da imagem e do banco são de emissoras diferentes (linhas de 14 a 26 na tabela 3.9), e;

3. *Falso positivo:* imagens onde cada pixel foi gerado utilizando distribuição gaussiana, testados contra logos contidos no banco (linhas de 27 a 39 na tabela 3.9).

Retiramos o coeficiente de correlação máxima entre as imagens analisadas contra os logos do banco para os pixels em torno da semente de cada logo, assim como é feito no algoritmo em tempo real. O nível de significância α admitido para os testes foi de 0,001, ou seja, 0,1%. A tabela 3.9 ilustra os parâmetros discutidos para as três situações respectivamente, indicando a cada teste se a hipótese H_0 foi aceita ou rejeitada.

Linha	Imagem da emissora	Logo	β	ε^2	τ	$\rho_{(max)}$	p	Aceita H_0
1	SBT imagem 1	SBT	0,910905	5,924625	13,889966	0,925708	< 0,0001	*não*
2	Globo imagem 1	Globo	0,982694	52,293325	2,881812	0,983559	0,0045	*não*
3	CNN	CNN	1,066694	194,304231	2,210462	0,920465	0,0286	*não*
4	Discovery Channel imagem 1	Discovery	0,819337	104,426853	2,612244	0,922814	0,0099	*não*
5	Gazeta	Gazeta	0,999221	0,002190	23892,433086	0,999720	< 0,0001	*não*
6	Cultura imagem 1	Cultura	0,779382	136,671779	1,061007	0,942440	0,2904	*sim*
7	SBT imagem2	SBT	0,817267	25,114237	2,939905	0,814589	0,0038	*não*
8	TV Justiça	Justiça	1,399280	33,902403	8,514539	0,797568	< 0,0001	*não*
9	CNU	CNU	0,207920	38,854727	1,340388	0,448684	0,1821	*sim*
10	SPORTV	SPORTV	0,612760	16,994722	9,87939	0,774152	< 0,0001	*não*
11	Globo imagem 2	Globo	0,736727	20,408954	5,535789	0,827563	< 0,0001	*não*
12	Discovery Channel imagem 2	Discovery	0,274946	22,588876	4,052444	0,846866	< 0,0001	*não*
13	Cultura imagem 2	Cultura	0,870159	127,632820	1,268478	0,927044	0.2066	*sim*
14	CNU	Globo	0,05629	61,944516	0,139355	0,0574	0.8804	*sim*
15	Globo imagem 1	SBT	-0,17587	91,759832	-0,173148	0,3741	0.8628	*sim*
16	Globo com ruído sal-pimenta	CNN	0,21748	701,25089	0,124874	0,06968	0,9008	*sim*
17	Globo com ruído sal-pimenta	Globo	-0,3849	1177,9334	-0,05011	0,04051	0,9601	*sim*
18	AXN	Discovery	0,03174	233,79573	0,012264	0,26128	0,9902	*sim*
19	CNN	AXN	0,03166	23,773639	0,443409	0,11435	0,6581	*sim*
20	Cultura imagem 1	Gazeta	-0,20853	7,382915	-1,479385	0,1501	0.1411	*sim*
21	SBT imagem 2	Justiça	-0,21604	20,463203	-2,177937	0,25412	0.3011	*sim*
22	TV Justiça	CNU	-0,59851	241,82159	-0,619949	0,34034	0,5362	*sim*
23	Cultura imagem 1	SPORTV	-0,0668	13,883336	-1,318347	0,14593	0,1894	*sim*
24	Discovery channel imagem 2	Cultura	0,25054	94,872202	0,491348	0,3706	0,6239	*sim*
25	SPORTV	SBT	0,03174	233795728	0,012264	0,26128	0.9902	*sim*
26	CNN	Discovery	0,00538	31,535391	0,056763	0,07335	0.9548	*sim*
27	Sem emissora	SBT	-0,06192	7,989913	-0,700143	0,09272	0,4849	*sim*
28	Sem emissora	Globo	-0,0938	59,297758	-0,242585	0,04172	0,8087	*sim*
29	Sem emissora	CNN	0,06948	80,223441	0,348716	0,06539	0,7278	*sim*
30	Sem emissora	Discovery	-0,12748	350,72817	-0,121011	0,06347	0.9038	*sim*
31	Sem emissora	Gazeta	-0,51174	103,08886	-0,259998	0,09921	0,7952	*sim*
32	Sem emissora	Cultura	-0,14004	112,59377	-0,231411	0,0924	0,8173	*sim*
33	Sem emissora	SBT	-0,3849	316,63807	-0,109819	0,09017	0,9127	*sim*
34	Sem emissora	Justiça	0,02269	8,176567	0,572477	0,04362	0,5679	*sim*
35	Sem emissora	CNU	-0,1619	269,00903	-0,150752	0,06603	0,8804	*sim*
36	Sem emissora	SPORTV	-0,10751	92,219917	-0,319443	0,09173	0,7498	*sim*
37	Sem emissora	Globo	-0,35047	685,2026	-0,078438	0,04622	0,9376	*sim*
38	Sem emissora	Discovery	0,22673	370,53259	0,203728	0,1095	0,8388	*sim*
39	Sem emissora	Cultura	-0,27796	496,74208	-0,104112	0,08751	0,9172	*sim*

Tabela 3.9 – Valores encontrados para o teste de hipótese. Aceita-se a hipótese H_0, caso a probabilidade *p* para o teste for maior que o nível de significância admitido de 0,001. As linhas 1 a 13 indicam o resultado encontrado para o teste onde as imagens contêm o logo da mesma emissora; as linhas 14 a 26 indicam o resultado encontrado para o teste onde as imagens contêm logos de emissoras diferentes, e; as linhas 27 a 39 indicam o resultado encontrado para o teste onde as imagens das emissoras foram geradas artificialmente de maneira uniforme com desvio padrão escolhido aleatoriamente.

Analisando a tabela verifica-se que para o índice mais crítico de falsa identificação ou falso positivo, não encontramos dados que rejeitem a hipótese H_0, nos dando um suporte estatístico quanto à eficácia do processo.

4 IMPLEMENTAÇÃO

Esta seção retrata os dois hardwares utilizados para implementar o algoritmo descrito nas seções anteriores. Dando detalhes que afetam a customização do processo.

4.1 Processamento Off-Line

A intenção do estudo é permitir o reconhecimento de canais em tempo real, com a utilização de um hardware dedicado. Embora a utilização de vídeos gravados, não sirva para homologar este estudo, o mesmo foi muito útil para avaliar os problemas que seriam encontrados em uma implementação em tempo real. Para este teste, não há preocupação quanto a tempo de processamento.

Os vídeos foram gravados utilizando-se um microcomputador (PC computer) com uma placa de TV tipo Play TV Pro Ultra, de fabricação da empresa PC View. A mesma pode gravar vídeos no formato *AVI,* recebidos por PAL-M ou NTSC de emissoras abertas ou a cabo. A fim de reduzir o consumo em armazenamento pelos vídeos produzidos, o mesmo tem a resolução de 320x240 (320 linhas por 240 colunas) colorido, formato RGB.

Foram gerados dois tipos de vídeos para a mesma emissora:

1. O primeiro onde a intenção era a geração do logo, e;
2. Um segundo vídeo onde o a intenção seria de reconhecimento do canal pela verificação do logo gerado.

Quando queremos gerar um logo semitransparente, o vídeo gravado deve ter uma variedade grande de imagens e fundos de imagens, “backgrounds”, para que a média dos quadros não tenda a um contraste específico. Para isso, alguns vídeos menores foram criados e agrupados para gerar um vídeo maior com grande variedade de situações, tanto em cores como em contraste. Caso o logo na qual se quer gerar seja opaco ou animado, não há preocupação

quanto a diferenças de contraste geradas pelo fundo da imagem, uma vez que o fundo da imagem não se torna parte do logo.

Todas implementações dos algoritmos mencionados durante o texto foram desenvolvidas com o auxílio da biblioteca Proeikon, desenvolvida por Hae Yong Kim em linguagem C++ (Kim, PROEIKON). Biblioteca composta por vários algoritmos de processamento de imagens em diversas áreas de pesquisa, como: aprendizagem de máquina, marca d'água, etc.

4.2 Implementação em Tempo Real em DSP

Como objetivo maior de nosso estudo, a implementação de um algoritmo em hardware embarcado para trabalhar em tempo real na tarefa de medição de audiência de televisão, nós decidimos por implementá-lo em um Processador Digital de Sinais (DSP). O DSP escolhido para tal tarefa foi o Blackfin BF533 da companhia Analog Devices. A plataforma de desenvolvimento para o DSP escolhido, chamada de EZ KIT LITE BF533 foi a nossa base. A figura 4.1 ilustra o ambiente de desenvolvimento que é composto por um computador para programação e captação de resultados, um televisor, VCR ou DVD, para geração de sinais PAL-M, além da placa de desenvolvimento.

Figura 4.1 – Ilustração do ambiente de desenvolvimento composto de um televisor para geração do sinal PAL-M além da placa de desenvolvimento da Analog Devices EZ KIT LITE BF533.

4.2.1 O Processador Digital de Sinais Blackfin

Os processadores digitais de sinais se caracterizam por possuir seu ISA (arquitetura do conjunto de instruções, “instruction set architeture”) ligeiramente modificado a fim de acelerar o seu desempenho. Com o processador Blackfin não é diferente, como será explicado em detalhes a seguir.

A linha de DSP’s Blackfin, da Analog Devices, possuem grande performance aliada a um baixo consumo de energia e custo. Com um duplo MAC, “multiply” (multiplica) e “acumulate” (acumula), funções vastamente utilizadas em algoritmos de processamento de sinais, um ortogonal conjunto de instruções RISC, arquitetura SIMD (vários dados única instrução, “single instruction multiple data”), e características de aplicações multimídia, o Blackfin é um ótimo DSP para aplicações de vídeo como neste caso, (Analog Devices, 2003).

O termo SIMD é membro de uma taxonomia definida por (Flynn, 1972) e que é baseada no fluxo de informação. A máquina onde existe apenas uma unidade de controle e todos os processadores executam a mesma instrução de maneira síncrona, dá-se o nome de SIMD (vários dados única instrução, “single instruction multiple data”). Em arquiteturas SIMD, o paralelismo é explorado aplicando-se simultâneas operações em grandes quantidades de dados, (Ifeachor, 2002).

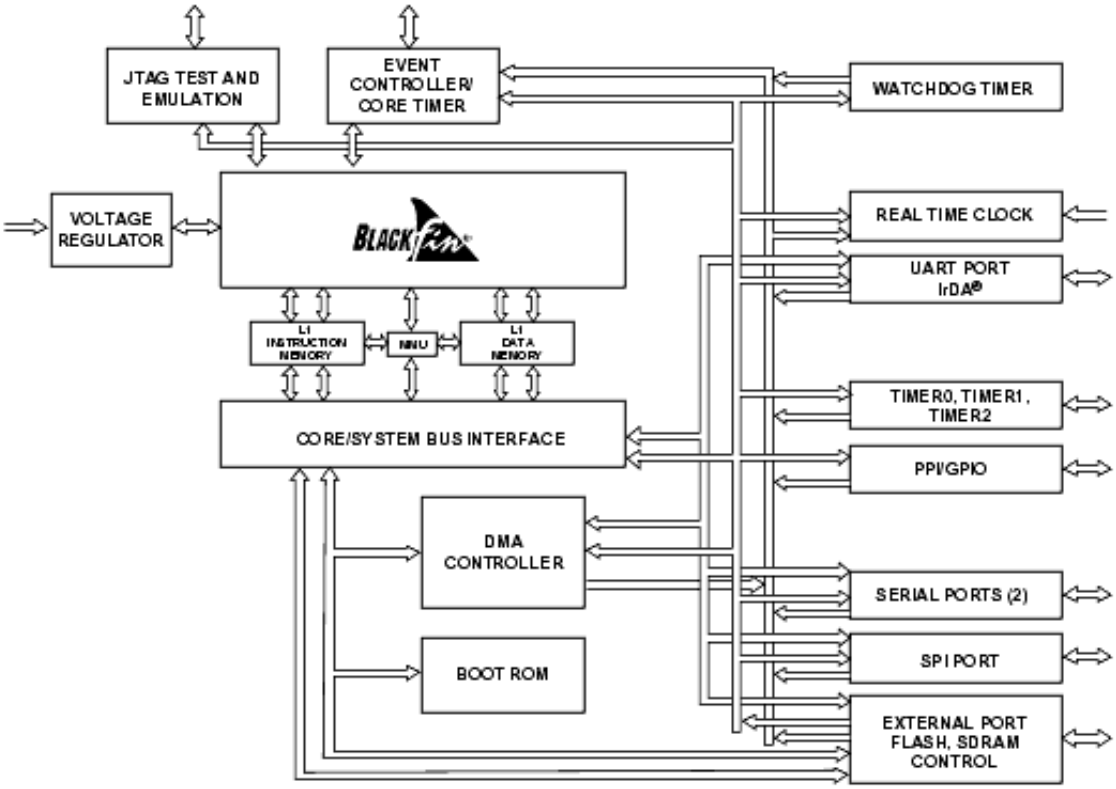


Figura 4.2 - Arquitetura do DSP Blackfin, figura retirada de (Analog Devices, 2003).

O núcleo do processador contém dois multiplicadores de 16 (dezesseis) bits, dois acumuladores de 40 (quarenta) bits, duas unidades lógicas aritméticas de 40 (quarenta) bits (ALUs), quatro ALUs de 8 (oito) bits para vídeo e um deslocador de 40 (quarenta) bits. A unidade de processamento computacional tem a capacidade de processar dados de 8 (oito), 16 (dezesseis) e 32 (trinta e dois) bits dos registradores.

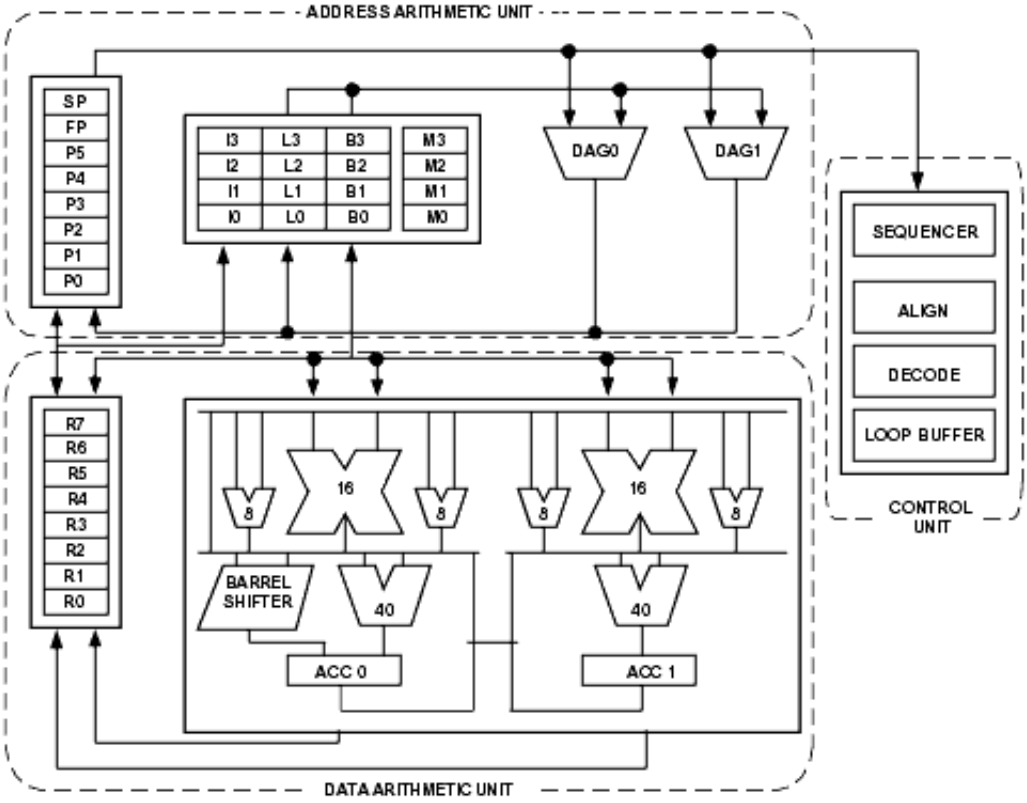


Figura 4.3 - Organização computacional do Blackfin, figura retirada de (Analog Devices, 2003).

Cada MAC pode realizar multiplicações de 16-por-16 (dezesseis por dezesseis) bits em um ciclo, acumulando o resultado em um acumulador de 40 (quarenta) bits.

As ALUs além de conter operações aritméticas tradicionais, também realizam algumas operações especiais a fim de acelerar o processo computacional de algoritmos específicos a processamento de sinais. Estas operações incluem: extração de campo, multiplicações com módulo 2^{32}, divisões, saturação e arredondamento e detecção de sinal/expoente. As instruções de vídeo incluem: alinhamento e agrupamento, soma de 16 (dezesseis) bits e 8 (oito) bits com "clipping", média de 8 (oito) bits e operações de somar/subtrair valores absolutos. Também são realizadas instruções de pesquisa de vetores como comparar/selecionar. Utilizando a vantagem de uma máquina SIMD, em algumas operações, pares de registradores podem ser processados paralelamente, ou ainda, usando a segunda ALU, quatro operações de 16 (dezesseis) bits podem ser executadas no mesmo ciclo. O deslocador de 40 (quarenta) bits pode depositar dados, realizar deslocamentos, rotacionar, normalizar e realizar operações de extração de dados.

Um Seqüenciador controla o fluxo de execução do programa, realizando alinhamento e decodificação. Para o controle de fluxo do programa, o seqüenciador suporta desvios relativos e indiretos, com predição estática, e chamada de sub-rotinas. Para acelerar o processo, implementações em hardware são realizadas para tal fim.

A Unidade de Endereçamento detém dois endereçadores para, simultaneamente, efetuar o acesso aos dados da memória. Uma estrutura de buffer circular é oferecida, além de 8 (oito) ponteiros de 32 (trinta e dois) bits.

O processador Blackfin contém memória L1, que opera na velocidade do núcleo, trabalhando com instruções e dados e podendo também ser configurada para trabalhar como memória cache, utilizando-se a Unidade de Gerenciamento de Memória.

O conjunto de instruções é otimizado, para que micro-códigos de 16 (dezesseis) bits sejam utilizados com mais freqüência, deixando que operações complexas, apenas utilizem micro-códigos de 32 (trinta e dois) bits. Há suporte de um limitado conjunto de operações que podem realizar em paralelo operações de 32 (trinta e dois) e 16 (dezesseis) bits.

4.2.1.1 Suporte a DMA (acesso direto à memória)

O processador tem múltiplos controladores independentes de DMA que suportam, automaticamente, realizar operações de transferência de dados com um mínimo desperdício de processamento para o núcleo, (Analog Devices, 2003). Assim como, transferências entre periféricos e memória, interna e externa, são permitidas, acelerando o processo de aquisição de dados da porta paralela como é o caso em questão.

O controlador de DMA suporta transferência de dados em uma dimensão, ou em duas dimensões, ideal para processamento de imagens. Além de permitir interrupções ao longo do processo ou ao final do mesmo. Processos de auto-inicialização, para transferência contínua, ou com parada no final do processo, ou ainda com a utilização de descritores, que permitem transferências de memórias com fontes e destinos diferentes, muito utilizados em processos que detenham protocolos de comunicação são permitidos.

4.2.1.2 Interface Periférica Paralela (PPI)

A conexão com o decodificador de vídeo é feita através de uma interface paralela de até 16(dezesseis) bits, PPI (interface periférica paralela), half-duplex. Tem um relógio dedicado ao sistema, três pinos multiplexados de sincronismo de quadro e quatro pinos dedicados a dados.

Apesar da comunicação de dados poder ser efetuada utilizando-se palavras de 16 (dezesseis) bits, a melhor performance do sistema é alcançada quando a comunicação é feita com palavras de 8 (oito) bits de comprimento, pois 2 (dois) bytes podem ser agrupados em uma palavra e transferidos, (Analog Devices, 2003). Para a aplicação em questão, devido à freqüência exigida pelo decodificador de vídeo ser de 27MHz (vinte e sete mega Hertz), a freqüência da PPI deverá ser também de 27MHz (vinte e sete mega Hertz), sincronizada com o sinal de vídeo fornecido pelo decodificador de vídeo.

O DMA é a maneira mais adequada de efetuar comunicações com esta porta, assim utilizamos este modo para fazer as aquisições de quadros vindos do decodificador de vídeo e transferir estes quadros diretamente para a memória externa SDRAM sem a intervenção do núcleo do DSP, deixando como obrigação do mesmo, apenas processar este quadro quando o DMA avisar que uma transferência foi concluída.

O canal de comunicação de DMA pode ser configurado para recepção ou transmissão de dados, neste caso é utilizado como recepção de dados. Um recurso muito útil para aplicações de vídeo existentes no Blackfin é o recurso de DMA 2D, ou seja, a memória é montada em uma estrutura de linha e coluna assim como uma imagem é formatada, dando a opção de controle das informações por linha, por quantidade de linhas ou ainda pelo quadro inteiro. Em nossa implementação, o controle é feito por quadro, ou seja, configuramos quantas linhas existem no quadro, e o DMA avisará quando a comunicação terminar. A figura abaixo ilustra o registrador de controle da porta PPI.

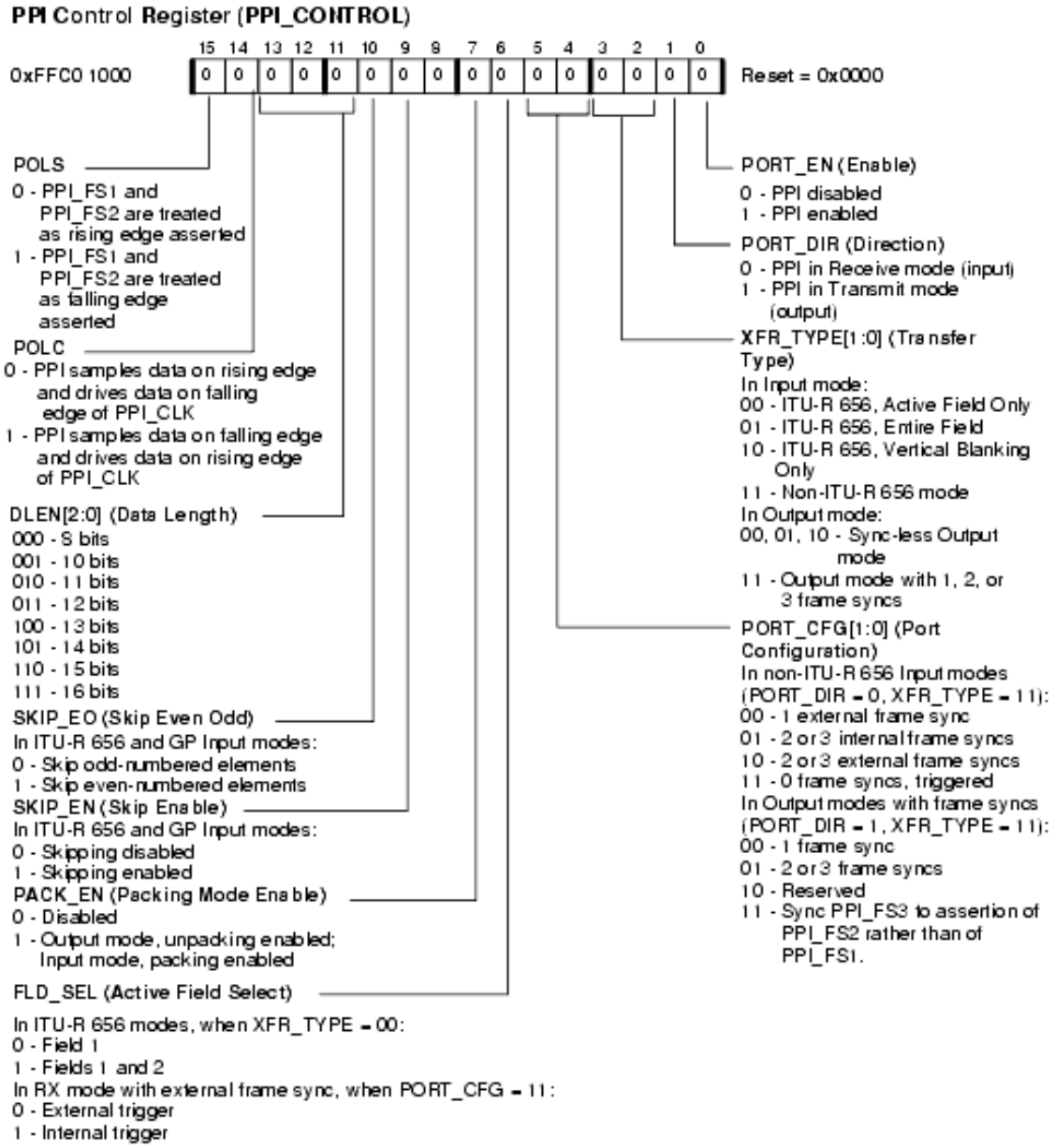


Figura 4.4 – Registrador de controle da porta PPI. Figura retirada de (Analog Devices, 2003).

A configuração feita para o registrador PPI_CONTROL foi a:

- DLEN de 8 (oito) bits;
- FLD_SEL para aquisição dos campos pares e ímpares;
- habilitado o agrupamento de luma com croma para formar uma palavra de 16 (dezesseis) bits;

- porta PPI habilitada (PORT_EN igual a 1 (um)).

4.2.1.3 Memória

Um grande problema atualmente que reduz a performance dos processadores é o "memory wall", (Mckee, 1995). No qual a velocidade de integração entre processador e memória não cresce na mesma proporção. A fim de se reduzir este problema, os processadores estão incluindo memórias internas ao chip que trabalham em nível 1 (L1).

O DSP Blackfin trabalha nesta linha provendo memória SRAM interna diminuindo a latência, assim aumentando a performance. A adição de caches para dados e instruções (SRAMs com controle de cache por hardware) provê alta performance e um simples modelo de programação. Caches eliminam a necessidade do gerenciamento de movimentação de dados na memória L1.

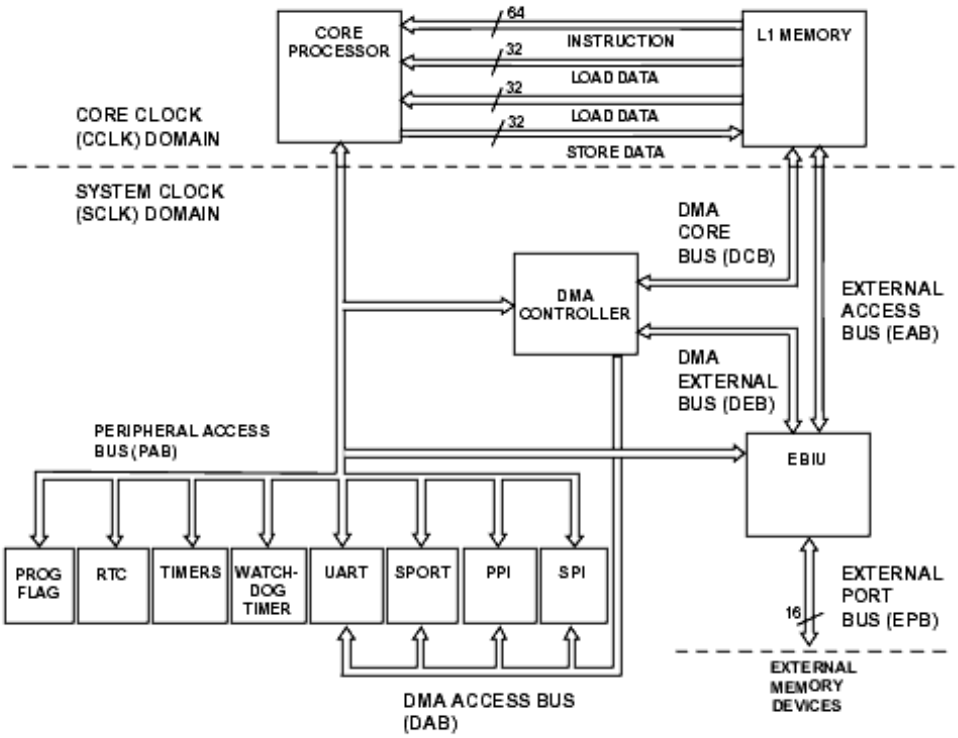


Figura 4.5 – Fluxo de dados e instruções. Figura retirada de (Analog Devices, 2003).

Existe um dedicado banco de memória de rascunho de 4 (quatro) Kbytes de memória SRAM. Esta memória é independente da configuração de outros bancos de memória L1, e não é possível o seu uso para cache ou DMA. Aplicações que tem a velocidade como máxima, deverão utilizar este recurso. Um exemplo de aplicação seria utilizá-la em interrupções onde a rápida troca de contexto é desejada.

4.2.1.4 Cache de Instruções

A memória L1 SRAM pode ser configurada para trabalhar com cache de dados ou instruções. Trabalhando com instruções em cache, esta pode conter 4-grupo 16K (dezesseis quilo) bytes de cache associativa. Para melhorar a média de latência de acesso, para seções de código que são críticas, cada grupo de linhas de cache pode ser bloqueado independentemente. Quando a memória é configurada como cache, nenhum acesso direto é permitido.

Como mostrado na figura 4.6, a cache consiste de uma coleção de linhas de cache. Cada linha de cache é definida por um componente de identificação e um componente de dados:

- o componente de identificação incorpora um endereço de identificação de 20(vinte) bits, os bits menos recentemente usados (LRU), um bit válido, e um bit de trava linha;
- o componente de dados é feito de quatro palavras de 64 (sessenta e quatro) bits de dados de instruções;
- os componentes de identificação e dados das linhas de cache são armazenadas na identificação e nos vetores de memória de dados, respectivamente.

A identificação do endereço consiste dos 18 (dezoito) bits mais significativos adicionados dos bits 11 (onze) e 10 (dez) do endereço físico. Os bits 12 (doze) e 13 (treze) do endereço físico não fazem parte da identificação do endereço. Ao contrário, estes bits são usados para identificar os 4K (quatro quilo) bytes de memória reservado para o acesso.

Os bits do LRU são parte do algoritmo LRU (menos usada recentemente) (Smith, 1982), utilizado para determinar quais linhas de cache devem ser sobrescritas se uma perda de coerência ocorrer. O bit válido indica o estado de cada linha de cache.

Linhas de cache inválidas contém seu tem seu bit válido em estado zero lógico, indicando que a linha deve ser ignorada durante uma operação de comparação de identificação de endereço.

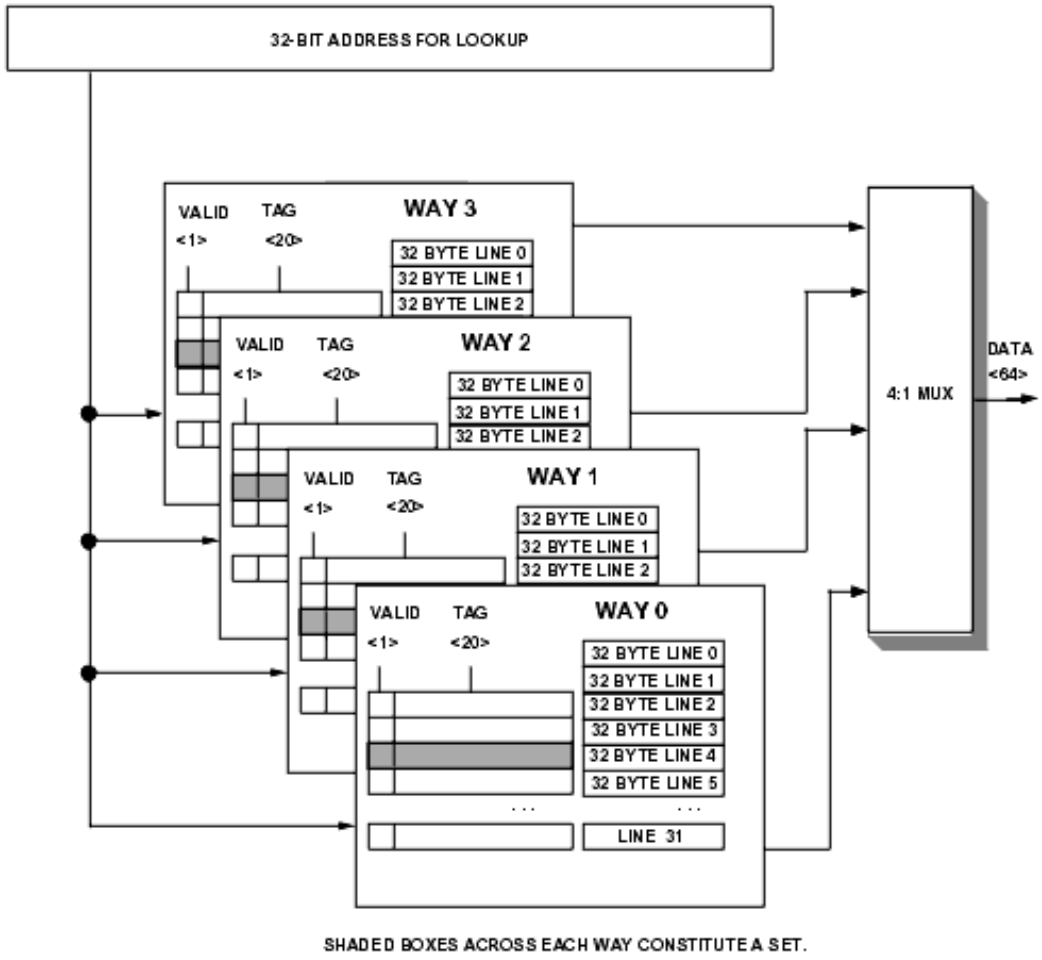


Figura 4.6 – Cache de instrução. Figura retirada de (Analog Devices, 2003).

Não utilizamos cache de instruções em nossa aplicação.

4.2.1.5 Cache de Dados

Dependendo do processador da família Blackfin, a quantidade máxima de memória que pode ser definida como cache varia, por exemplo: no ADSP-BF533 é possível habilitar os bancos A e B de 16K (dezesseis quilo) bytes cada, enquanto que no ADSP-BF531 apenas o banco A de

16 K (dezesseis quilo) bytes pode ser habilitada. Ao contrário da cache de instruções que é de 4 (quatro) grupos, a cache de dados é de 2 (dois) grupos associativos. Quando ambos os bancos são habilitados como cache, configurações adicionais são criadas.

Acessos a cache não colidem, a menos que eles estejam no mesmo 4K (quatro quilo) bytes de sub-bancos, o mesmo meio-banco.

O processador define 3 (três) instruções para controle de cache de dados: PREFETCH, FLUSH e FLHUSHINV, a fim de garantir a coerência de cache, (Analog Devices, 2003).

O controlador de acesso a cache testa o endereço das DAGs contra os bits identificadores. Se o endereço lógico está presente na cache L1, um encontro a cache ocorre, e os dados são acessados na L1. Se o endereço lógico não está presente, uma perda de cache ocorre, e uma transação de memória é passada para o próximo nível de memória via o sistema de interface. O índice da linha e a política de recolocação para o controlador de cache determinam o identificador de cache e o espaço dos dados que são alocados para os dados vindos da memória externa.

A identificação da cache de dados pode estar em um dos três estados: inválido, exclusivo (válido e limpo) e modificado (válido e sujo). Se um dado válido já ocupa uma linha alocada e a cache está configurada para armazenamento “write-back”, o controlador checa o estado da linha da cache e a trata da seguinte maneira:

- se o estado da linha é exclusivo (limpo), o novo identificador e os dados sobrescrevem a linha antiga;
- se o estado da linha é modificado (sujo), então a cache contém somente uma cópia válida dos dados;
- se a linha está suja, o conteúdo corrente da cache é copiada de volta para a memória externa antes dos novos dados serem escritos na cache.

Operações de escrita na memória podem ser implementadas usando ou o método de “write-through” ou “write-back”:

- para cada operação, o método “write-through” inicia a memória externa imediatamente a escrita na cache;

- o método “write-back” não escreve na memória externa até que a linha seja recarregada pela operação que necessita da linha.

Não utilizamos cache de dados em nossa aplicação.

4.2.1.6 Economia de Energia

Uma das principais características do Blackfin é o DPCM (gerenciamento dinâmico de controle de potência) que trabalha em conjunto com a PLL permitindo ao usuário que dinamicamente altere a performance do processador para reduzir o consumo de energia. Esta redução se faz através:

- múltiplos modos de operação, até 4 (quatro), sendo: full on, active, sleep e deep sleep;
- os relógios dos periféricos são desabilitados quando os mesmos não estão sendo utilizados;
- controle de tensão através de um controlador de tensão dentro do próprio processador.

4.2.2 A Plataforma de Desenvolvimento

A Analog Devices detém uma plataforma de desenvolvimento para processadores Blackfin BF533 dedicada a aplicações multimídia chamada EZ KIT LITE BF533, (EZ KIT LITE BF533, 2006).

Esta placa contém todos os elementos necessários para uma implementação real: 2MB de memória não volátil tipo Flash divididos em dois bancos de 1MB cada, responsável por armazenar o programa, além do banco de logos; 128MB (cento e vinte e oito mega byte) de memória volátil tipo SDRAM para acesso rápido, com clock de até 133MHz (cento e trinta e três mega Hertz) padrão PC133, para dados de um modo geral; uma porta serial padrão RS-232 para prover uma comunicação com o PC; o DSP Blackfin modelo BF533, o processador principal da máquina; um decodificador de vídeo capaz de detectar automaticamente sinais nos padrões PAL, NTSC ou SECAM e gerar pacotes de vídeo no padrão ITU-656, com

palavras com resoluções de 8 (oito) ou 10 (dez) bits configuráveis, (ANALOG DEVICES, 2003).

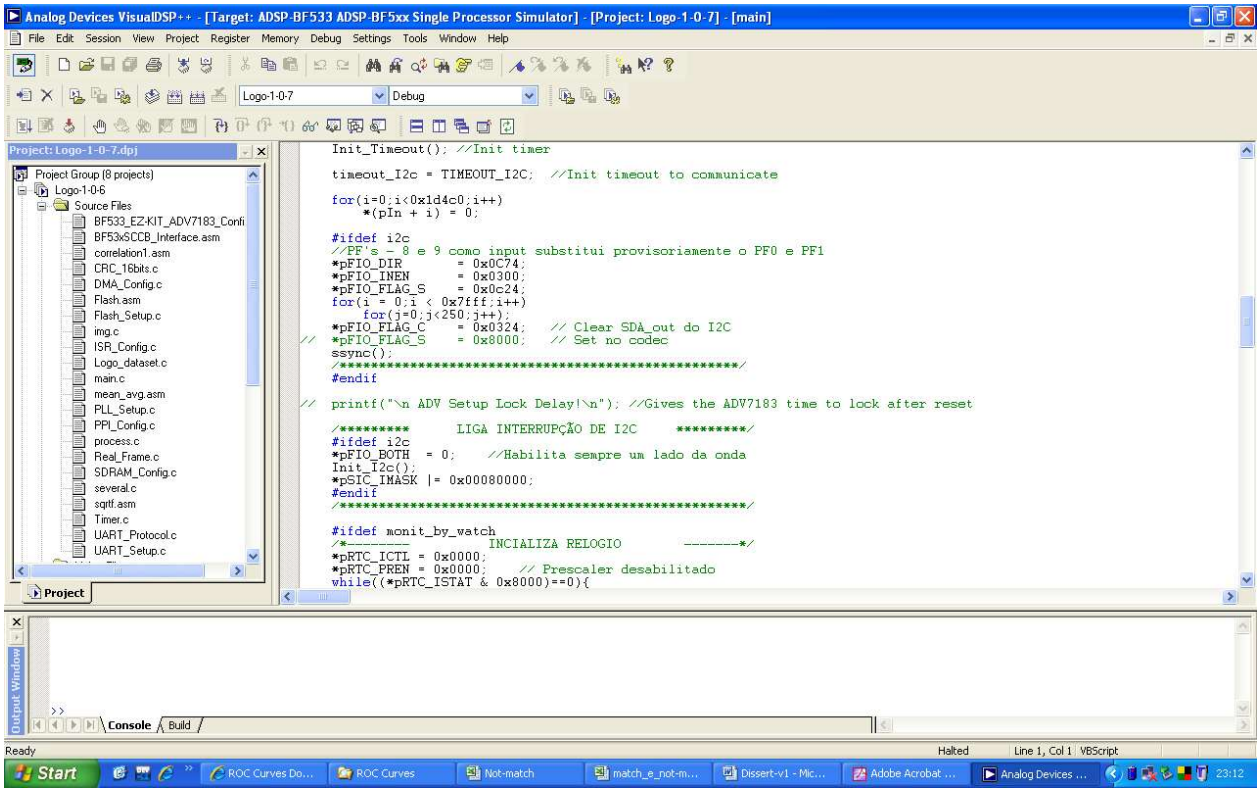
4.2.3 O Ambiente de Desenvolvimento

A ferramenta de desenvolvimento de aplicações para o processador de sinais Blackfin disponibilizada pela Analog Devices se chama Visual DSP++. É um completo ambiente de desenvolvimento e depuração para toda a linha de processadores Blackfin, com ferramentas gráficas para depuração e análise de dados, além de contadores de ciclos para medições de eficiência, visualização de registradores, ponteirs, "stack pointer", e memória, ferramenta para salvar o conteúdo da memória em arquivo texto, visualizador gráfico de buffers com informação de imagens em vários formatos, com possibilidade de se gravar estas imagens em arquivo bitmap ou jpeg, dentre outras funções. Existe uma função de depuração que indica a porcentagem de processamento da máquina na qual cada função existente no código está ocupando no momento, tudo em tempo real no próprio código escrito, sem a necessidade de criar funções de avaliação de desempenho. O ambiente comporta o desenvolvimento de aplicações nas linguagens: assembly, C e C++.

Uma ferramenta muito utilizada neste desenvolvimento foi à visualização das imagens processadas e carregadas, além dos recursos de depuração da memória interna e externa, agilizando o desenvolvimento da aplicação.

A Analog Devices disponibiliza várias funções de biblioteca escritas para o Blackfin, o uso das mesmas gera uma maior rapidez no desenvolvimento da aplicação e o uso eficiente da máquina.

Pode-se desenvolver aplicações utilizando os recursos de depuração já comentados, e posteriormente, com a ajuda de um aplicativo fornecido pela Analog Devices, carregar a aplicação em memória não-volátil. A figura 4.7 ilustra o ambiente de desenvolvimento de aplicações.




```
    Init_Timeout(); //Init timer

    timeout_I2c = TIMEOUT_I2C;  //Init timeout to communicate

    for(i=0;i<0x1d4c0;i++)
        *(pIn + i) = 0;

    #ifdef i2c
    //PF's - 8 e 9 como input substitui provisoriamente o PF0 e PF1
    *pFIO_DIR       = 0x0C74;
    *pFIO_INEN      = 0x0300;
    *pFIO_FLAG_S    = 0x0c24;
    for(i = 0;i < 0x7fff;i++)
        for(j=0;j<250;j++);
    *pFIO_FLAG_C    = 0x0324;   // Clear SDA_out do I2C
//  *pFIO_FLAG_S    = 0x8000;   // Set no codec
    ssync();
    /*************************************************/
    #endif

//  printf("\n ADV Setup Lock Delay!\n"); //Gives the ADV7183 time to lock after reset

    /*********      LIGA INTERRUPÇÃO DE I2C     *********/
    #ifdef i2c
    *pFIO_BOTH  = 0;    //Habilita sempre um lado da onda
    Init_I2c();
    *pSIC_IMASK |= 0x00080000;
    #endif
    /*************************************************/

    #ifdef monit_by_watch
    /*--------          INCIALIZA RELOGIO        -------*/
    *pRTC_ICTL = 0x0000;
    *pRTC_PREN = 0x0000;    // Prescaler desabilitado
    while((*pRTC_ISTAT & 0x8000)==0){
```

Figura 4.7 – Ilustração do ambiente de desenvolvimento para Blackfin.

Para ilustrar algumas ferramentas utilizadas no desenvolvimento da aplicação, a figura 4.8 traz alguns instrumentos de depuração existentes no ambiente de desenvolvimento VisualDSP++, como a visualização de variáveis em memória interna ou externa, na ilustração a variável vista está alocada em memória externa SDRAM, também se verifica três imagens, a maior, localizada na parte central da imagem constitui a saída da média móvel temporal, e a imagem ao lado na parte inferior, corresponde ao logo da emissora Globo armazenada em memória Flash.

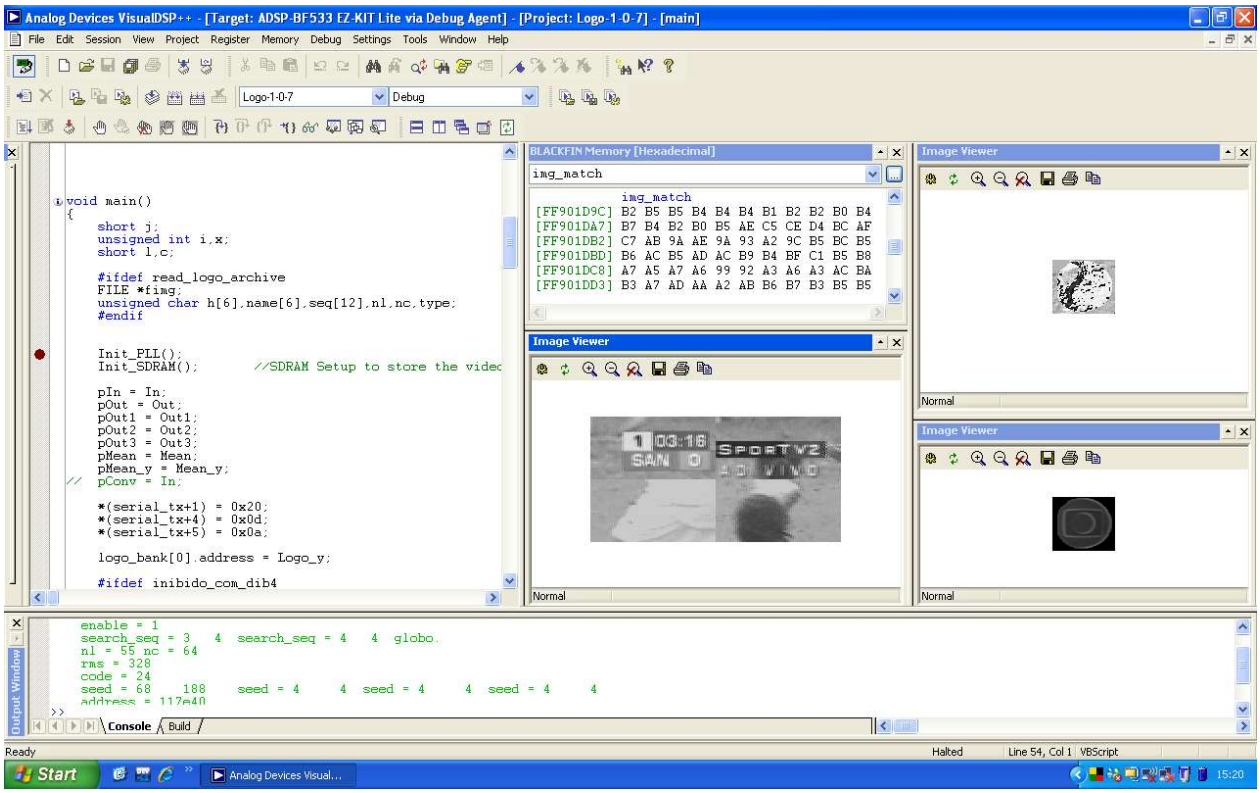


Figura 4.8 – Ilustração do ambiente de desenvolvimento com a aplicação. É possível verificar as imagens que estavam sendo depuradas, além de um buffer em memória SDRAM.

4.2.4 Arquitetura da Aplicação Desenvolvida

Existem dois clocks separados na aplicação, um chamado de “core clock”, e o segundo chamado de “system clock”. O core clock é o relógio que controla o núcleo do DSP e foi configurado para atuar a uma velocidade de 594MHz (quinhentos e noventa e quatro mega Hertz). O system clock é o relógio que controla a velocidade dos periféricos, memória SDRAM, etc; este foi configurado para atuar a uma velocidade de 126MHz (cento e vinte e seis mega Hertz). Os relógios são configurados mediante a alterações em prescalers que atuam em cima de um cristal. Este cristal, para o kit de desenvolvimento utilizado é de 27MHz (vinte e sete mega Hertz).

Deve-se tomar alguns cuidados na calibração dos relógios e levar em consideração a velocidade limite para alguns periféricos, como a SDRAM no nosso caso que suporta velocidades de até 133 MHz (cento e trinta e três mega Hertz) padrão PCI-133, além do

próprio DSP, que para o Blackfin BF533 admite uma velocidade máxima de 600MHz (seiscentos mega Hertz).

A escolha de um cristal de 27MHz (vinte e sete mega Hertz) é justificada pelos periféricos de vídeo, decodificador de vídeo ADV7183 e codificador de vídeo ADV7171, que exigem um clock externo nessa freqüência.

Com o intuito de efetuarmos uma melhor utilização das memórias, internas e externas, algumas funcionalidades são executadas na memória interna, L1, e outras são executadas na memória externa, SDRAM. Não houve a necessidade de se utilizar cache de dados, ou instruções, ficando como opção em uma continuação deste trabalho se houver interesse.

Rotinas de inicialização, carga de logos, monitoração, ou seja, que são pouco ou, na maioria das vezes, não utilizadas no processo de geração de audiência em tempo real, são executadas diretamente da SDRAM. As demais rotinas que demandam uma alta performance são executadas ma memória interna L1.

Pode-se fazer uso dos dados gerados sobre a audiência de duas maneiras:

1. Enviando os dados a uma central ou equipamento mestre no domicílio responsável pela geração da audiência, modo “Real-Time” (IBOPE), minuto a minuto;

2. Fazendo o envio das informações coletadas durante o dia no período noturno, “Overnight” (IBOPE), sendo que para isso é necessário que se tenha um buffer de armazenamento das audiências minuto a minuto.

A implementação do algoritmo foi feita utilizando as linguagens C e Assembly, com algoritmos em ponto fixo, onde obtém-se a maior capacidade de processamento para o Blackfin, utilizando o recurso da manipulação de dados em ponto fracionário (1.15 e 1.31), sem o uso de “threads”, efetuando o controle de tempo Δt, de controle dos quadros utilizados para o cálculo mediante a captura do mesmo e sinalização da interrupção de DMA associada ao periférico PPI de comunicação paralela com o DSP. Portanto, a cada quadro recebido e transferido para a memória SDRAM via DMA, uma interrupção é gerada. Através de um simples contador, o controle de quadros válidos para o cálculo de média móvel temporal foi efetuado.

Uma vez que o quadro foi escolhido para ser pesquisado, o mesmo passa pelo processo de média móvel temporal e um aviso para que o programa principal efetue os demais cálculos, formação das quatro imagens de cantos, detector de bordas, extrator de nível DC, e a pesquisa por logos é feita. O processo se repete indefinidamente, gerando como saída o canal encontrado no minuto corrente.

O sistema possui as seguintes interrupções:

- por recebimento de quadros;
- por comunicação I^2C, e serial;
- por tempo.

O processador Blackfin contém um registrador para configuração da prioridade com que as interrupções serão tratadas, **IFIVG**. Para o nosso sistema, a interrupção com maior prioridade é a de recepção de quadros via PPI, assim melhoramos a performance do sistema.

As figuras 4.9, 4.10 e 4.11 demonstram um fluxograma macro para as três versões do algoritmo, há algumas variações dependendo da versão, embora elas utilizem uma mesma estrutura. Estas figuras têm como objetivo ilustrar alguns blocos básicos que existem nas três versões.

Os logos a serem pesquisados são recebidos de maneira serial e guardados em memória flash, compondo um banco de logos. Este banco de logos é composto por uma estrutura de dados que informa todas as características necessárias para a rotina de pesquisa, são elas:

- número de linhas do logo;
- número de colunas do logo;
- cantos onde o logo pode aparecer, portanto, informam as áreas onde as pesquisas serão realizadas;
- linha semente para inicio da pesquisa;
- coluna semente para inicio da pesquisa;

- código do logo que será informado na audiência;
- nome do logo para depuração e informe ao operador do sistema.

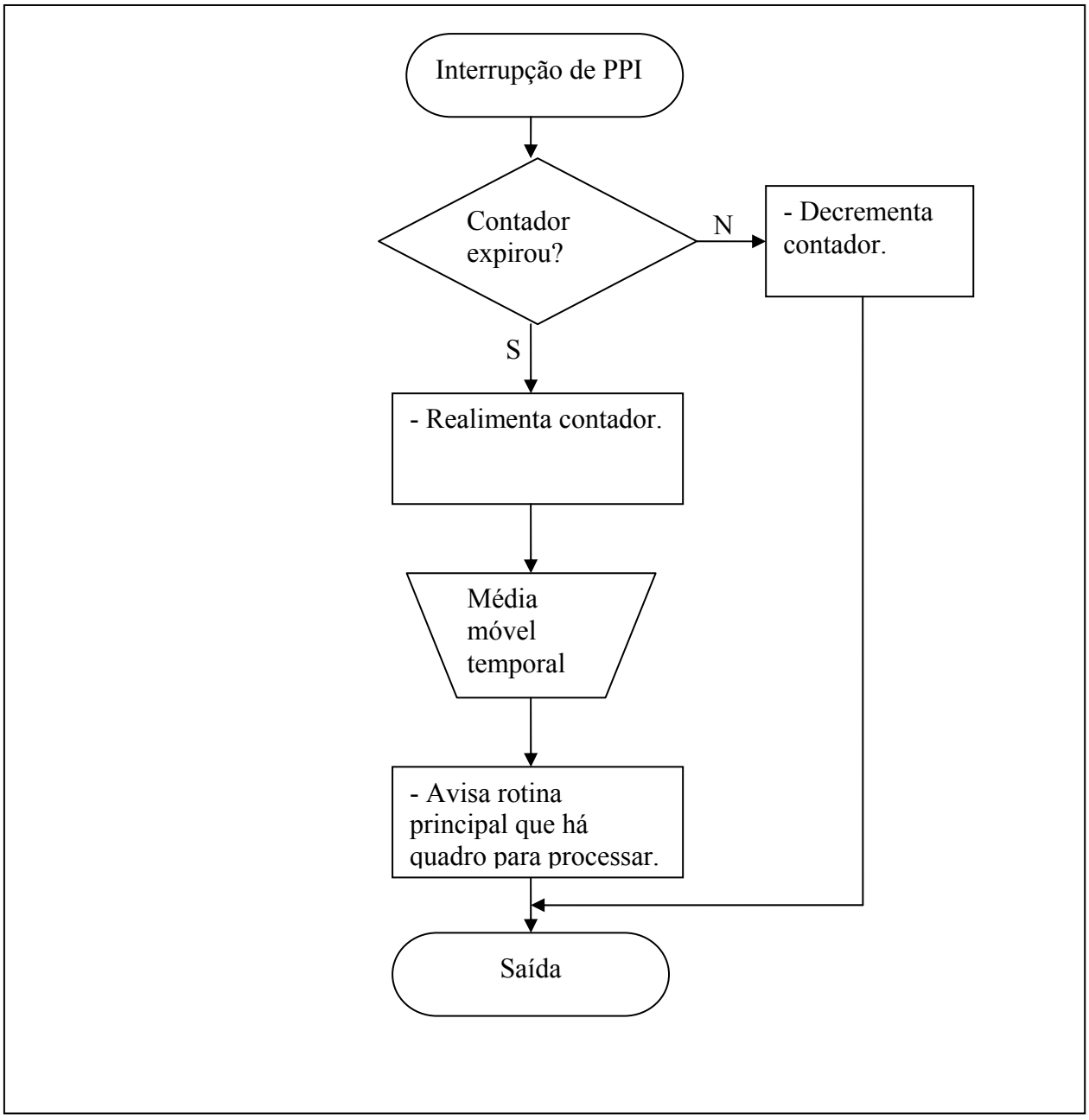


Figura 4.9 – Fluxograma macro da interrupção de PPI, responsável por receber os quadros oriundos do decodificador de vídeo.

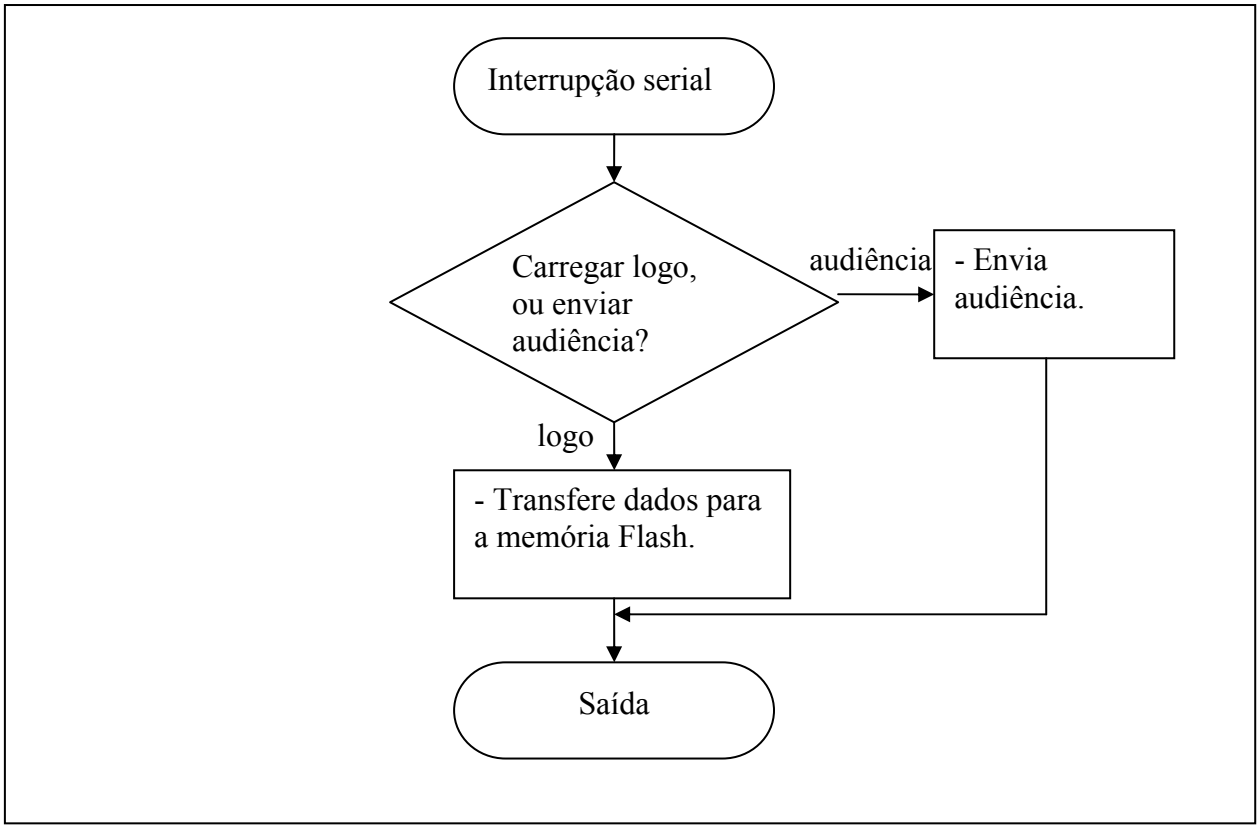


Figura 4.10 – Fluxograma macro do processo de comunicação serial.

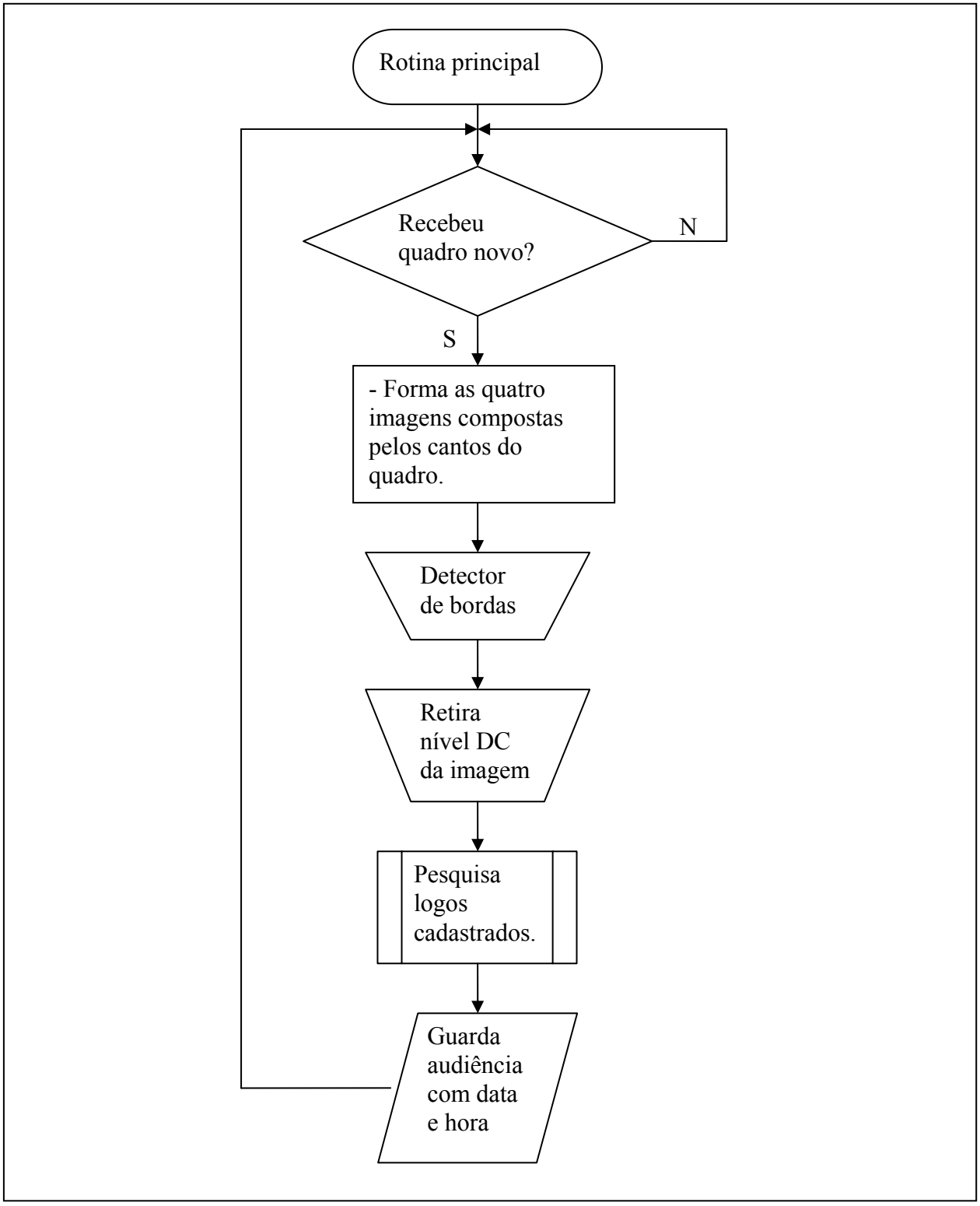


Figura 4.11 – Ilustração do fluxograma macro do bloco da rotina principal.

Efetuamos também um levantamento aproximado do custo de fabricação de uma placa para tal fim. Para isso levamos em conta um lote de 100 (peças) para termos um custo reduzido neste protótipos. Concluímos que pode ser feita uma placa utilizando a estrutura do kit de desenvolvimento a um valor de aproximadamente U$100,00 (cem dólares americanos).

5 CONCLUSÕES

O processo de medição de audiência de televisão está sempre em constante pesquisa por processos que identifiquem os canais em diferentes meios. Com o advento da televisão digital e a introdução de set-top-boxes digitais, cada vez mais as pesquisas dos conteúdos de áudio e vídeo serão realizadas, pois, meios vastamente utilizados até então como a medição de freqüência de sintonia do "tunner" se torna ineficaz, ou até impossível em alguns casos.

Esta pesquisa teve como objetivo principal demonstrar a eficácia de se utilizar uma informação disponibilizada pelas emissoras de televisão para registrar a sua marca, o logo.

As emissoras de televisão não utilizam regras para a criação de um logo, sendo uma função apenas publicitária a criação do design e efeitos para atrair a visão do telespectador. Isso implica em formas, cores e animações diversas.

Através dos estudos demonstrados ficou comprovado a eficiência do algoritmo criado para qualquer tipo de logo, independente de sua forma ou cor. Pode-se utilizar qualquer uma das versões 2 ou 3 criadas, pois, todas as duas reconhecem o canal dentro do limite de 1 (um) minuto estipulado pelo IBOPE em seu sistema de tempo real, muito embora a versão 2 (dois) se mostrou mais robusta a ataques, onde se pode verificar tal constatação pela taxa de falsos negativos demonstrados, e também é um algoritmo com uma resposta rápida na detecção do primeiro logo para a emissora, podendo trabalhar em um sistema em tempo real brasileiro, "real-time" (IBOPE), ou com reporte das informações em uma faixa horária pré-estabelecida, "overnight" (IBOPE).

Pode-se ainda fazer a junção de versões criadas para, por exemplo, ter uma combinação de robustez a ataques e rapidez na detecção de canais, para isso, a versão 2 aliada a uma taxa de média móvel de 63/64 pode ser utilizada, com um limiar de decisão de 73 (setenta e três) a 75% (setenta e cinco porcento).

6 REFERÊNCIAS

ALBIOL, A; FULLÀ, M. J. C.; ALBIOL A.; TORRES, L., "Detection of TV commercials," *Proc. ICASSP'04,* vol. III, pp. 541-544, May 2004.

ANALOG DEVICES, "ADSP-BF533 Blackfin Processor Hardware Reference," revision 1.0, 2003.

ANALOG DEVICES, "Blackfin Processor Instruction Set Reference," revision 2.0, 2003.

BRIGHAM, E. O., "The Fast Fourier Transform," Prentice Hall, 1974.

BRAGA-NETO, U. M.; GOUTSIAS, J., "An Axiomatic Approach to Multiscale Connectivity in Image Analysis" in Technical Report JHU/ECE, 2004.

CASTLEMAN, K. R., "Digital Image Processing," Prentice-Hall, 1996.

CHEN, J.; LEUNG, M. K. H.; GAO, Y., "New Approach for Logo Recognition," *Optical Pattern Recognition XI, Proc. of SPIE,* vol. 4043, pp. 272-279, March 2000.

INTERNATIONAL TELECOMMUNICATION UNION - ITU. Disponível em http://www.itu.int/net/. Acesso em: 05 de Agosto de 2006.

DUDA, R. O.; HART, P. E.; STORK, D. G., "Pattern Classification", second edition, Wiley, 2001.

DUNCAN, Ralph, "A Survey of Parallel Computer Architectures," in *IEEE Computer*, February 1990, pp. 5-16.

EKIN, A.; BRASPENNING, R., "Spatial detection of TV Channel Logos as outliers from the content," *Visual Communications and Image Processing 2006, Proc. SPIE,* vol. 6077.

EZ KITE LITE BF533. Disponível em: http://www.analog.com/processors/platforms. Acesso em: 11 de Agosto de 2006.

FLYNN, M., "Some Computer Organizations and Their Effectiveness," in *IEEE Trans. Comput.*, vol. C-21, pp. 948, 1972.

GOLAY, M. J. E., "Hexagonal Pattern Transformation", IEEE Trans. Computers, C-18, 733-740, 1969.

GONZALEZ, R. C.; WOODS, R. E., "Digital Image Processing," second edition, Prentice-Hill, 2002.

GROB, B.; "Televisão Básica", Guanabara Dois, Rio de Janeiro, 1979.

HARGROVE, T., "Logo Detection in Digital Video," in *http://thomashargrove.com/logo-detection/* acessado em 03/03/2006.

HAYKIN, S., "Redes Neurais: Princípios e prática," segunda edição, Bookman, 2004.

IBOPE. Disponível em: www.ibope.com.br. Acesso em: 18 de Junho de 2007.

IFEACHOR, E. C.; JERVIS, B. W., "Digital Signal Processing: A Practical Approach," second edition, Prentice Hall, 2002.

ITU-656. Disponível em: International Telecommunication Union - http://www.itu.int/rec/R-REC-BT.656-4-199802-I/en . Acesso em: 05 de Junho de 2007.

JAISHIMA, M. Y., "Wavelet features for similarity-based retrieval of logo images," *Document Recognition III, Proc. SPIE,* vol. 2660, pp. 89-100, March 1996.

KAZMIER, L. J., "Estatística Aplicada a Economia e Administração," Schaum McGraw-Hill, 1982.

KIM, H. Y. PROEIKON. Disponível em: http://www.lps.usp.br/~hae/software. Acesso em: 26 de Setembro de 2006.

KOVAR, B.; HANJALIC, A., "Logo Detection and Classification in a Sport Video: Video Indexing for Sponsorship Revenue Control," *Storage and Retrieval for Media Databases, Proc. SPIE*, vol. 4676, p. 183-193, 2001.

MCKEE, S. A., "Hitting the Memory Wall: Implications of the Obvious," in *ACM SIGARCH Architecture News, 23*, ISSN: 0163-5964, march 1995.

MEISINGER, K.; TROEGER, T.; ZELLER, M.; KAUP, A., "Automatic TV Logo Removal Using Statistical Based Logo Detection and Frequency Selective Inpainting" *Proc. European Signal Processing Conference*, September 2005.

METZ, C., "Basic principle of ROC analysis" *Nuclear Medicine*, VIII, n. 4, p.283-298, 1978.

MICROSOFT, AVI REFERENCE. Disponível em: Microsoft – AVI File Reference - http://msdn2.microsoft.com/en-us/library/ms779636.aspx. Acesso em: 11 de dezembro de 2007.

NINCE, U. S., "Sistemas de Televisão e Vídeo," segunda edição, Livros Técnicos e Científicos, 1991.

OPPENHEIM, A. V.; SCHAFER R. W.; BUCK J. R., "Discrete-Time Signal Processing," second edition, Prentice Hall, 1975.

PHAN, T., "Applications of geostatistics and Markov models for logo recognition," *Document Recognition and Retrieval XI, Proc. SPIE,* vol. 5010, pp. 20-27, January 2003.

PIXELVIEW. Disponível em: http://www.pixelview.com.br/. Acesso em: 02 de Agosto de 2006.

PROAKIS, J. G.; MANOLAKIS, D. G., “Digital Signal Processing: Principles, Algorithms, and Applications,” third edition, Prentice Hall, 1996.

SANTOS, A. R.; KIM, H. Y., “Robust Edge-based Template Matching,” in *Proc. Int. Information and Telecommunication Technologies Symposium* (*I2TS)*, vol. 1, pp. 156-162, 2006.

SEEBER, B.; YAGER, N.; AMIN, A., “Real-time Detection of Semi-transparent Watermarks in Decompressed Video,” in *Proc. WACV*, 2007.

SMITH, A. J., “Cache Memories,” in *ACM Computing Surveys*, vol. 14, p. 473-530, 1982.

SMITH, W. S., “The Scientist and Engineer’s Guide to Digital Signal Processing,” second edition, California Technical Publishing, 1999.

SYMES, P.; “Video Compression Desmystified”, Mc-Graw Hill, 2001.

RUSS, J. C., “The Image Processing Handbook,” second edition, CRC Press, 1994.

WANG, J.; DUAN, L.; LIU, Z. Li; LU, H.; JIN, J. S., “A Robust Method for TV Logo Tracking in Video Streams” in *Proc. IEEE Int. Conf. on Multimedia and Expo (ICME)*, pp. 1041-1044, 2006.

YAN, W. Q.; KANKANHALLI, M. S., “Erasing Video Logos Based on Image Inpainting,” in Proc. *Int. Conf. on Multimedia and Expo (ICME)*, Switzerland, vol.2, pp. 521-524, Aug. 2002.

YAN, W. Q.; WANG, J.; KANKANHALLI, M. S., “Automatic Video Logo Detection and Removal” *Multimedia Systems,* 10(5), pp. 379-391, July 2005.

YEH, J.–H.; CHEN, J.–C.; KUO, J.–H.; WU, J.-L., “TV Commercial Detection in News Program Video,” *Proc. Int. Sym. Circuits and Systems (ISCAS)*, vol.5, pp. 4594-4597, May 2005.