Cálculo de Modelo de Fundo com Média de Frames em Assembly MIPS-32

Tobias Rocha
RA: 2714027
Curso: Ciência da Computação
Disciplina: Organização de Computadores
Universidade Tecnológica Federal do Paraná (UTFPR)
Ponta Grossa, Brasil

***Resumo*—O cálculo de modelo de fundo é uma técnica fundamental em visão computacional para a detecção de objetos em movimento. Este trabalho descreve a implementação de um algoritmo de subtração de fundo baseado na média aritmética de um conjunto de quadros (frames) de vídeo. A solução foi desenvolvida em Assembly para a arquitetura MIPS-32, utilizando o simulador MARS. O projeto aborda a manipulação de arquivos de imagem no formato PGM (escala de cinza) como requisito base, e sua extensão para o formato PPM (colorido) como funcionalidade extra. São discutidos os desafios de implementação em baixo nível, como o gerenciamento de memória e as limitações do ambiente de simulação, bem como as estratégias adotadas para superá-los.**

***Index Terms*—Organização de Computadores, Arquitetura de Computadores, Assembly MIPS-32, Modelo de Fundo, Processamento de Imagens, PGM, PPM.**

I. Introdução

A separação entre objetos de interesse (primeiro plano) e o cenário estático (plano de fundo) é uma das tarefas primordiais em sistemas de visão computacional, servindo como base para aplicações de vigilância, contagem de objetos e análise de tráfego [6]. Diversas técnicas foram propostas na literatura para realizar essa tarefa, variando em complexidade e robustez [2], [8].

Uma das abordagens mais diretas e eficazes para cenas com câmera estática é o cálculo da média temporal dos pixels [5]. Este método assume que o plano de fundo é o que aparece com maior frequência em uma sequência de imagens. Assim, ao calcular a média do valor de cada pixel ao longo do tempo, os objetos em movimento, que são transitórios, tendem a ser suavizados, revelando o fundo estático.

O objetivo deste trabalho é implementar este algoritmo em Assembly para a arquitetura MIPS-32. O programa deve ser capaz de ler uma sequência de arquivos de imagem, processar os dados de pixel em baixo nível e gerar uma nova imagem contendo o modelo de fundo da cena.

II. Desenvolvimento

A solução foi desenvolvida no simulador MARS, seguindo a arquitetura MIPS-32. O projeto foi dividido em duas fases: a implementação base para imagens em escala de cinza (PGM) e a extensão para imagens coloridas (PPM), que constitui o ponto extra.

A. Implementação Base (PGM)

Para o formato PGM (P5), cada pixel é representado por um único byte. O algoritmo implementado segue os seguintes passos:

1) Alocação de um buffer acumulador em memória, com tamanho de uma *word* (4 bytes) por pixel da imagem, para evitar *overflow* durante a soma.
2) Um laço principal itera sobre cada frame de entrada. Para cada frame, os dados de pixel são lidos.
3) Os valores de cada pixel do frame atual são somados às posições correspondentes no buffer acumulador.
4) Após processar todos os frames, um segundo laço percorre o buffer acumulador, dividindo cada valor pelo número total de quadros para obter a média.
5) O resultado (o pixel médio) é armazenado em um buffer final de bytes, que é então gravado em um novo arquivo PGM, precedido pelo cabeçalho apropriado.

B. Extensão para Imagens Coloridas (PPM)

A extensão para o formato PPM (P6) introduziu novos desafios. Neste formato, cada pixel é composto por três bytes: Vermelho (R), Verde (G) e Azul (B). Isso implicou em:

- **Processamento por Canal:** A lógica de soma e média foi triplicada para ser executada independentemente para cada um dos três canais de cor.
- **Aumento da Memória:** A quantidade de memória necessária para os buffers aumentou em três vezes. O buffer acumulador, em particular, tornou-se significativamente grande (largura $\times$ altura $\times$ 3 canais $\times$ 4 bytes/canal).

C. Superando Limites do Simulador

Durante a implementação para PPM, o programa encontrou um limite intransponível de alocação de memória no simulador MARS, mesmo utilizando alocação dinâmica no *heap* via *syscall* 9 (*sbrk*). A solicitação de memória para armazenar metade de um frame colorido já excedia a capacidade do simulador.

Para contornar essa restrição, uma nova estratégia de processamento foi adotada: a imagem foi dividida e processada em duas metades (superior e inferior). O algoritmo foi modificado para:

1) Processar todos os frames de entrada, mas lendo e acumulando apenas os dados da primeira metade da imagem.
2) Calcular a média da primeira metade e escrevê-la no arquivo de saída.
3) Repetir todo o processo para a segunda metade, desta vez pulando os dados da primeira metade de cada arquivo de entrada antes de realizar a leitura.

Essa abordagem reduziu o uso de memória a um nível suportado pelo simulador, ao custo de um aumento no número de operações de leitura de arquivo, mas viabilizou a execução do projeto.

III. Resultados

O programa foi executado com sucesso para ambos os formatos de imagem. As figuras a seguir ilustram os resultados obtidos.

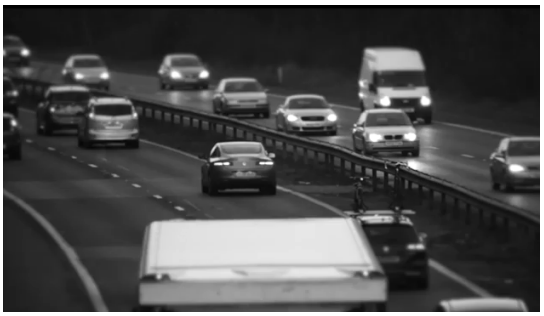

Figura 1. Quadro de entrada (PGM) da sequência de vídeo.

Figura 2. Modelo de fundo (PGM) gerado a partir de 400 quadros.

Para a implementação base, um conjunto de 400 frames em escala de cinza foi utilizado. A Figura 1 mostra um quadro típico da sequência. A Figura 2 exibe o resultado final, onde o fundo estático (pista e barreiras) está nítido, e os veículos em movimento são apenas rastros semitransparentes, o que é o comportamento esperado do algoritmo da média. A execução do algoritmo no simulador MARS para 400 frames em escala de cinza levou aproximadamente 5 minutos.

IV. Ponto Extra

Para a obtenção do ponto extra, o algoritmo foi estendido para o formato PPM, conforme descrito na seção de desenvolvimento. A Figura 3 mostra um frame colorido da sequência de entrada. O resultado final, processado com 400 frames, é visto na Figura 4. A execução do algoritmo no simulador MARS para 400 frames em escala de colorida levou aproximadamente 15 minutos, demonstrando a intensidade computacional do processamento de uma imagem 3 vezes maior e abrindo o dobro de arquivos. A superação dos desafios de memória do simulador através da estratégia de processamento em metades foi crucial para o sucesso desta etapa.

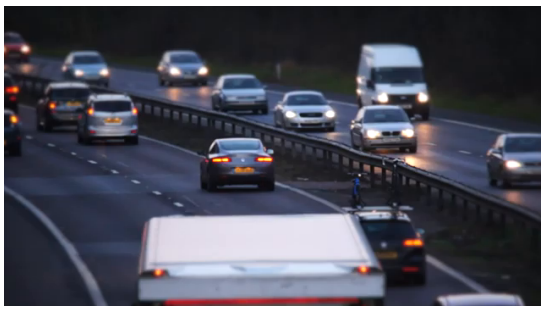

Figura 3. Quadro de entrada (PPM) da sequência de vídeo colorida.

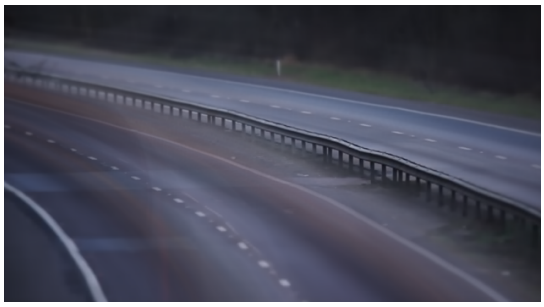

Figura 4. Modelo de fundo (PPM) colorido gerado a partir de 400 quadros.

Referências

[1] O. Barnich and M. Van Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Transactions on Image processing*, vol. 20, no. 6, pp. 1709–1724, 2011.

[2] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *European conference on computer vision*. Springer, 2000, pp. 751–767.

[3] B. Henderson, "Netpbm home page," Electronic document at http://netpbm.sourceforge.net/, 2008.

[4] D.-S. Lee, "Effective gaussian mixture learning for video background subtraction," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 5, pp. 827–832, 2005.

[5] B. Lo and S. Velastin, "Automatic congestion detection system for underground platforms," in *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*. IEEE, 2001, pp. 158–161.

[6] M. Piccardi, "Background subtraction techniques: a review," in *2004 IEEE International Conference on Systems, Man and Cybernetics*, vol. 4. IEEE, 2004, pp. 3099–3104.

[7] S. C. Sen-ching and C. Kamath, "Robust techniques for background subtraction in urban traffic video," in *Visual Communications and Image Processing 2004*, vol. 5308. International Society for Optics and Photonics, 2004, pp. 881–893.

[8] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *null*. IEEE, 2004, pp. 28–31.

[9] Bouwmans, T. Traditional and recent advances in background modeling for moving object detection: A review. *Computer Science Review* 11 (2014), 31–66.