2
Subtração de fundo

Este capítulo apresenta a subtração de fundo, tarefa para qual foi reservado um grande esforço inicial devido a sua enorme importância para a boa funcionalidade das outras etapas.

A subtração de fundo visa diferenciar, em uma seqüência de vídeo, os objetos dinâmicos (em movimento) dos estáticos (parados). Quanto maior a qualidade com que os objetos em movimento forem extraídos, menor será o custo computacional nas etapas seguintes. Essas técnicas vêm sendo estudadas há mais de 25 anos e ainda não existe uma abordagem definitiva para resolver este problema de forma genérica. As soluções existentes consistem na resolução desse problema em condições específicas relacionadas à aplicação que se deseja criar.

O maior desafio de um algoritmo de subtração de fundo é ser robusto em relação a: variações na iluminação (posição e intensidade), presença de sombras, objetos em movimento que sofrem camuflagem, regiões com superfícies espelhadas, mudanças na movimentação (oscilação da câmera e objetos de alta freqüência), mudanças na geometria do fundo. Além disso, neste trabalho é buscada uma solução em tempo real para a vigilância.

Algoritmos de subtração de fundo estão bastante populares na literatura. São citados aqui alguns dos mais populares. Stauffer et al. [7] apresentaram uma proposta que lida com múltiplos modelos de fundo conhecido como *MOG* (*Mixture of Gaussians*), onde cada pixel da imagem pode ser modelado por uma mistura de 3 a 5 gaussianas. O algoritmo Wallflower de Toyama et al. [8] emprega o filtro linear de Wiener (um modelo simplificado do filtro de Kalman) para aprendizagem e previsão de eventuais mudanças no cenário de fundo. Kim et al. [9] apresentaram o chamado *Codebook* (*CB*). Esse algoritmo permitiu a construção de um modelo de fundo a partir de longas seqüências de vídeo.

Haritaoglu et al. [1] propuseram o W^4, um algoritmo de subtração de fundo baseado em estatísticas de uma distribuição normal. Esse trabalho é provido de ótimos resultados publicados, porém ele descarta a informação que a cor fornece e

trabalha somente com a luminância. Tal fato faz com que o algoritmo falhe ao segmentar objetos escuros ou de luminância similar à do modelo de fundo.

Kampel e Wildenauer [10] apresentam um modelo de espaço de cor modificado, chamado de IHSL, e realizam a subtração de fundo através de métricas de distância. Os seus resultados foram apenas medianos, apresentando muito ruído, devido à falta de um modelo estatístico mais apurado.

A idéia do algoritmo deste trabalho é justamente unir os trabalhos de Haritaoglu et al. [1] e Kampel e Wildenauer [10], realizando algumas modificações para que um modelo de fundo apresente características estatísticas para uma espaço de cor sofisticado que auxilie na subtração de fundo. O algoritmo foi projetado para trabalhar com vídeos de rodovias e, a partir disso, algumas considerações foram feitas:

1. O ambiente de trabalho é somente o ambiente externo durante a fase diurna.
2. O modelo de fundo é baseado no plano estático do vídeo, sendo sua principal representante a rodovia.
3. Os objetos em movimento são todos os que passam ao longo da rodovia, como veículos e pessoas.
4. Os veículos que, por um motivo qualquer, se tornarem estacionários ao longo da cena, não são incorporados ao modelo de fundo.
5. As únicas alterações que podem ocorrer no cenário são as que forem causadas pela iluminação, tanto devido a mudanças climáticas quanto a mudanças temporais e estas, neste trabalho, não são incorporadas ao modelo de fundo.

A opção de termos optado pelo espaço ISHV é que este espaço de cor apresenta algumas vantagens sobre outros espaços, pois este representa bem a percepção humana de cor, devido à separação da cromacidade (saturação e tonalidade) da intensidade (valor). A vantagem disso é que, além de ser mais fácil para a modelagem, uma variação da intensidade luminosa na cena é irrelevante para a informação de cromacidade da imagem. O mesmo não acontece com o tradicional espaço RGB, onde a distância entre duas cores não é simétrica. O espaço IHSV e os espaços de cores similares (HSL, HSI, HSV, etc.) também

oferecem outras vantagens: lidam melhor com ruídos, têm maior exatidão de valores médios de suas componentes e maior facilidade de caracterizar regiões em sombra. Além disso, o IHSL retira a normalização da saturação pela luminância trazendo mais duas vantagens em relação aos seus similares: a saturação dos pixels acromáticos é sempre baixa e é independente da função de luminância usada [10].

Kampel e Wildenauer [10] propuseram a seguinte transformação de RGB para IHSV:

$$
\begin{aligned}
s &= \max(R,G,B) - \min(R,G,B) \\
y &= 0.215R + 0.7154G + 0.0721B \\
cr_x &= R - \tfrac{G+B}{2},\ \ cr_y = R - \tfrac{\sqrt{3}}{2}(B-G) \\
cr &= \sqrt{cr_x^2 + cr_y^2} \\
\theta^H &= \begin{cases} não\ definido, & se\ cr = 0 \\ \cos^{-1}\left(\frac{cr_x}{cr}\right), & se\ cr_y \le 0 \\ 360° - \cos^{-1}\left(\frac{cr_x}{cr}\right), & caso\ contrário \end{cases}
\end{aligned}
\tag{1}
$$

onde s, y e θ^H são os valores encontrados para a saturação, da luminância e da matiz, respectivamente.

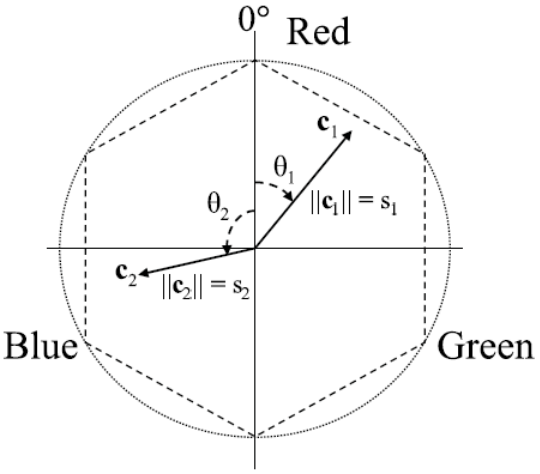


Figura 2: Plano cromático do espaço IHSL. Retirado de [10].

O algoritmo desenvolvido neste trabalho segue a organização do modelo de Haritaoglu et al. [1], possuindo três etapas bem definidas: treinamento, classificação e pós-processamento. A primeira fase, a de treinamento, tem o objetivo de realizar a estimativa inicial do modelo de fundo. Essa etapa tem duração de 200 quadros em seqüência do trecho inicial do vídeo. Para todos os pixels de cada quadro do treinamento é realizado o filtro da mediana da mesma maneira que Haritaoglu et al. [1] sugerem.

O filtro da mediana cria uma estimativa inicial para identificar quais pixels são estacionários e quais são dinâmicos. A Equação 2 demonstra esse filtro onde $I^t(x)$ é a intensidade do pixel x em um quadro t qualquer, $\lambda(x)$ é a mediana e $\sigma(x)$ é o desvio padrão do pixel x considerando todos os quadros do treinamento.

$$\left|I^t(x) - \lambda(x)\right| < 2\sigma(x) \tag{2}$$

Apenas os pixels que satisfazem essa equação são utilizados de fato na montagem do representante do modelo de fundo. Esta operação é realizada somente para o canal da luminância y.

Haritaoglu et al. [1] sugerem, também, que cada pixel filtrado deve ser modelado através de um vetor de fundo $B(x)$ que armazenaria para cada pixel x, o maior valor $m(x)$, o menor valor $n(x)$ e a maior diferença entre dois quadros consecutivos $d(x)$, na fase de treinamento para o canal da luminância.

Neste trabalho, propõe-se a inclusão de um quarto campo, chamado $f(x)$, que representa a moda de x, ou seja, valor mais freqüente para um determinado pixel x. O vetor do modelo de fundo final é representado pela Equação 4. Essa modificação no formato original visa ajustar os valores de $m(x)$ e $n(x)$ quando a variação de valores do pixel for muito grande. Esse caso acontece quando, durante todo o treinamento, a rodovia está engarrafada com veículos, sobrando pouco espaço para capturar a informação predominante da pista. Assim, essa correção é sugerida segundo a Equação 3, onde α é um limiar que vale 0,1. Admite-se

também que os pixels da imagem são temporalmente coerentes ao longo de todo o vídeo.

$$\text{Se,} \quad d(x) > 2 * \sigma(x)$$

$$\text{Então,} \quad \begin{cases} m(x) = (1-\alpha)\, f(x) \\ n(x) = (1+\alpha)\, f(x) \end{cases} \tag{3}$$

No caso da saturação e da matiz, a modelagem do fundo segue a mesma sugerida por Kampel e Wildenauer [10], que é encontrar o vetor de crominância médio $\bar{c}_n$ para cada pixel da imagem, sendo calculado segundo a Equação 4, onde *n* é o número de quadros do treinamento.

$$\begin{aligned} C_S &= \sum_{i=1}^{n} s_i \cos\theta_i^H \\ S_S &= \sum_{i=1}^{n} s_i \sin\theta_i^H \\ \bar{c}_n &= \left(\frac{C_S}{n}, \frac{S_S}{n}\right)^T \end{aligned} \tag{4}$$

Assim, a fase de treinamento chega ao seu fim com toda a informação do modelo de fundo B(x) já calculada, conforme a Equação 5.

$$\mathrm{B}(x) = \begin{bmatrix} m_y(x) \\ n_y(x) \\ d_y(x) \\ f_y(x) \\ \bar{c}_n(x) \end{bmatrix} = \begin{bmatrix} \min_n\{I^t(x)\} \\ \max_n\{I^t(x)\} \\ \max_n\{|I^t(x) - I^{t-1}(x)|\} \\ moda(x) \\ \left(\frac{C_S}{n}, \frac{S_S}{n}\right)^T \end{bmatrix} \tag{5}$$

2.1 Classificação

A segunda etapa do algoritmo é a fase de classificação. Ela utiliza o vetor do modelo de fundo para decidir quais pixels fazem parte dos objetos em movimento (representado por 1) e quais fazem parte do fundo (representado por 0).

Segundo Haritaoglu et al. [1], a classificação deve ser de acordo com a Equação 6, onde cada pixel *x* de um novo quadro de entrada é representado por $I^t(x)$, em que *t* é o número do quadro no treinamento, *k* é empiricamente 2 e d_u é a mediana das maiores diferenças absolutas de todos os pixels do quadro atual.

$$C(x) = \begin{cases} 0 & \left(I^t(x) - m(x)\right) < kd_\mu \; ou \; \left(I^t(x) - n(x)\right) < kd_\mu \\ 1 & caso\ contrário \end{cases} \tag{6}$$

No caso do algoritmo de Kampel e Wildenauer [10], a classificação é representada pela Equação 7, na qual y_0 é o valor da luminância observada; s_0,a saturação; h_0, a matiz; σ_y o desvio padrão da luminância no treinamento; μ, a média e σ_D é o desvio padrão de D, que por sua vez é a distância euclidiana do plano cromático calculado no treinamento segundo a Equação 8.

$$C(x) = \begin{cases} 1 & \left|(y_0 - \mu_y)\right| > \alpha\sigma_y ou \; \|\bar{c}_n - s_0 h_0\| > \alpha\sigma_D \\ 0 & Caso\ contrário \end{cases} \tag{7}$$

$$D = \sqrt{(\bar{c}_n - c_0)^T \, (\bar{c}_n - c_0)} \tag{8}$$

A equação de classificação utilizada neste trabalho explora um pouco das duas idéias sugeridas pelas equações 6 e 7, aproveitando a parte da classificação em relação à luminância do W^4 e à cromacidade de Kampel e Wildenauer [10], resultando na classificação final proposta que é representada pela Equação 9.

$$C(x) = \begin{cases} 0 \begin{cases} ((y^t(x) > m(x) - k) \\ e\ (y^t(x) < m(x) + k)) \\ ou\ \|\bar{c}_n - s_0 h_0\| > \alpha\sigma_D \end{cases} \\ 1 \quad caso\ contrário \end{cases} \tag{9}$$

Com o intuito de definir melhor o formato das regiões segmentadas, ou *blobs*, encontradas pela subtração de fundo é realizada uma etapa de pós-processamento. Essa etapa é composta por uma seqüência de algoritmos clássicos: aplicação de filtros morfológicos, aplicação de filtros de suavização, detecção e suavização de silhuetas ou contornos. A Figura 3 ilustra a imagem original seguida do resultado encontrado pela subtração de fundo, a seguir o pós-processamento e por fim a detecção das silhuetas.

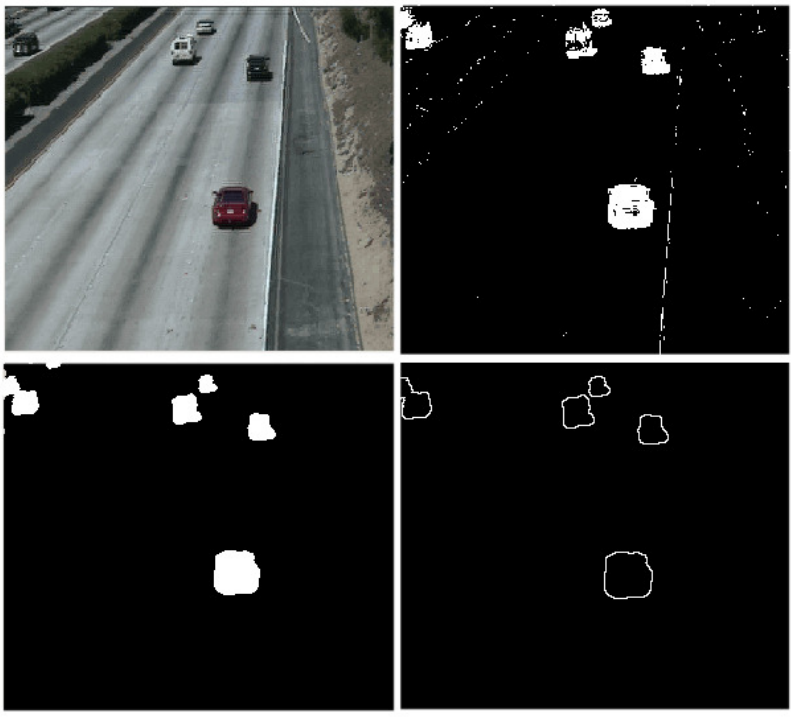


Figura 3: Resultado da etapa de pós-processamento da subtração de fundo.

2.2
Resultados parciais

Para avaliar o desempenho do algoritmo de subtração de fundo foram utilizados quatro vídeos ilustrados pela Figura 4 em ordem de dificuldade, chamados de (a) "*Highway* 1", (b) "*Highway* 2", (c) "Passarela 1" e (d) "Passarela 2". Foram considerados nesta avaliação dois quesitos: qualidade da segmentação e o desempenho computacional. Alguns algoritmos da literatura foram selecionados para realizar uma comparação com o algoritmo proposto. São eles: IHSL de Kampel e Wildenauer [10], W^4 de Haritaoglu et al. [1], W^4 modificado (somente com a inclusão do campo *f* no modelo de fundo), Kaewtrakulpong e Bowden [11], KDE [12] e Li et al. [13].

O primeiro quesito utiliza métricas baseadas no "*ground-truth*" do vídeo para definir a precisão dos resultados de um algoritmo de subtração de fundo. Karaman et al. [14] sugerem alguns quesitos para realizar esta avaliação:

1. A razão dos positivos VPr, também chamada de precisão P, é dada por $\frac{\mathrm{VP}}{\mathrm{VP+FN}}$, onde VP são os verdadeiros-positivos e FN os falsos-negativos, e fornece o número de pixels em movimento detectados corretamente.

2. A razão VNr dada por $\frac{\mathrm{VN}}{\mathrm{VN+FP}}$, onde VN são os verdadeiros-negativos e FP os falsos-positivos, fornece o número de pixels de fundo detectados corretamente.

3. A razão FPr, dada por $\frac{\mathrm{FP}}{\mathrm{FP+VN}}$, fornece o número de pixels de fundo detectados equivocadamente como movimento.

4. A razão FNr, dada por $\frac{\mathrm{FN}}{\mathrm{VP+FN}}$, fornece o número de pixels de fundo detectados equivocadamente como fundo.

5. A razão R, dada por $\frac{\mathrm{VP}}{\mathrm{VP+FP}}$, mede o quão preciso é o resultado dos verdadeiros-positivos.

6. A razão F combina as precisões R e P para avaliar com mais eficiência os algoritmos. Essa proporção é dada por: $\frac{2\mathrm{PR}}{\mathrm{P+R}}$.

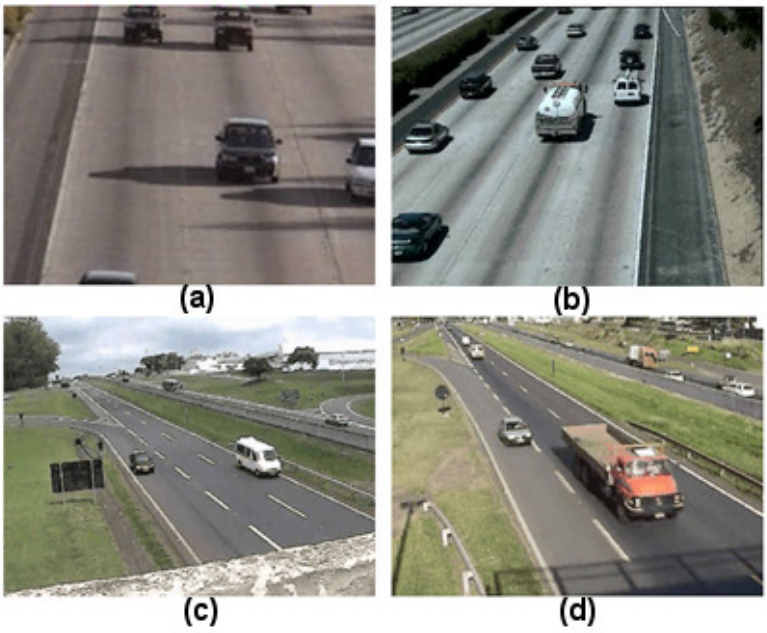


Figura 4: Os quatro vídeos selecionados para os testes da subtração de fundo: (a) Highway1, (b) Highway2, (c) Passarela 1 e (d) Passarela 2

A Figura 5 ilustra o resultado da segmentação de alguns destes algoritmos em relação ao "*ground-truth*" que foi obtido manualmente para um quadro do vídeo "*Highway* 1". As cores da segmentação representam cada uma destas métricas apresentadas por Karaman et al. [14], onde os VP são os da cor verde, os VN da cor branca, FN da cor vermelha e FP da cor azul.

O valor de F foi considerado o critério mais importante para a avaliação da segmentação, e a partir de sua análise observou-se que os resultados dos algoritmos variam bastante de um vídeo para o outro. Os resultados encontram-se no Apêndice A, e observando-os conclui-se que os algoritmos que mais ocilaram no valor de F foram: Kaewtrakulpong e Bowden [11] com resultados de F entre 50 e 89%, KDE [12] entre 61 e 89%, o algoritmo proposto, 64 e 80%, entre 68 e 89% com o pós-processamento, W^4 modificado entre 50 e 75%. Com os algoritmos restantes obteve-se resultados mais contantes de um vídeo para outro, são eles W^4 com a variação de 61 a 68% e 57 a 70% com o pós-processamento, Li et al. [13] com 65 a 75% e por fim, IHSL com 54 a 64% e 58 a 68% com o seu pós-processamento.

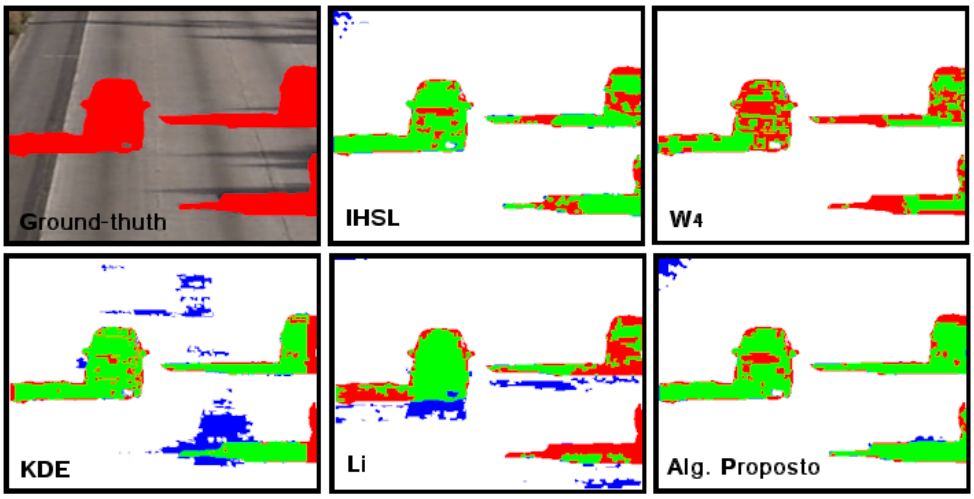
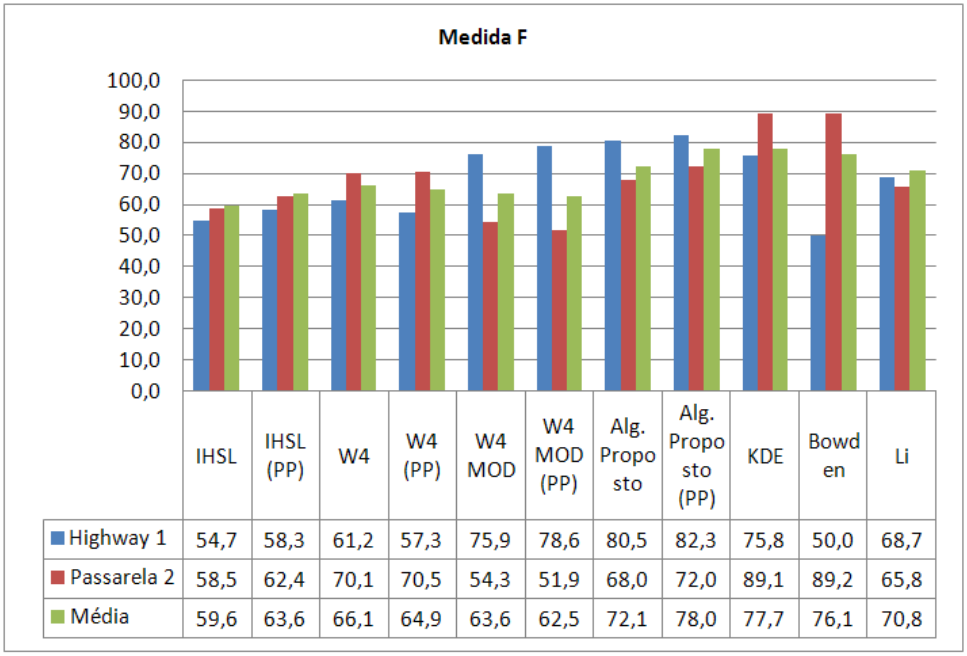


Figura 5: Resultados comparativos de alguns dos algoritmos de subtração de fundo selecionados. Legenda: VP – Verde, VN – Braco, FN – Vermelho e FP – Azul.

Um mesmo método que obteve bons resultados em um vídeo obteve outros não tão satisfatórios num vídeo diferente. O gráfico da Figura 6 ilustra justamente este contraste entre dois desses vídeos e a comparação com a média final de todos os quatro vídeos. Observando este gráfico, o algoritmo de Kaewtrakulpong e Bowden [11] obtiveram resultados próximos de 50% para o vídeo "*Highway* 1", enquanto obteve excelentes resultados (89%) para a "Passarela 2". A tabela completa com todos os resultados encontra-se no Apêndice A no final do documento.

Estes resultados mostram que ainda não existe uma abordagem de subtração de fundo que resolva bem todos os casos apresentados, mas o algoritmo proposto e o KDE [12] foram os que obtiveram os melhores resultados encontrados, com uma pequena vantagem do algoritmo proposto na média.



	IHSL	IHSL (PP)	W4	W4 (PP)	W4 MOD	W4 MOD (PP)	Alg. Proposto	Alg. Proposto (PP)	KDE	Bowden	Li
Highway 1	54,7	58,3	61,2	57,3	75,9	78,6	80,5	82,3	75,8	50,0	68,7
Passarela 2	58,5	62,4	70,1	70,5	54,3	51,9	68,0	72,0	89,1	89,2	65,8
Média	59,6	63,6	66,1	64,9	63,6	62,5	72,1	78,0	77,7	76,1	70,8

Figura 6: Resultado da medida F para os testes da subtração de fundo dos algoritmos selecionados.

Ao avaliar a média final dos quatro vídeos da razão R para estes dois algoritmos, KDE [12] obteve o valor de 94% contra 98% do algoritmo proposto. Isso mostra que o KDE tem uma tendência de desegmentar partes dos veículos (grande número de falsos-negativos) maior do que a do algoritmo proposto. Porém, KDE teve vantagem em relação à média da medida P com 66% contra 66.0% do proposto. Essa medida avalia que o algoritmo desenvolvido neste trabalho tem a tendência de segmentar mais que o necessário (grande número de falsos-positivos) em relação ao KDE. Como avaliação final da segmentação, o valor de F foi decisivo para demonstrar a pequena superioridade do algoritmo proposto com 78% de acerto contra os 78% do KDE.

O último quesito é o desempenho computacional dos algoritmos escolhidos. Essa informação é de extrema importância para as aplicações que buscam um desempenho em tempo real, um dos objetivos deste trabalho. A máquina utilizada para este teste de desempenho foi um Intel Core 2 @ 1.87GHZ com 2GB RAM e todos os vídeos em questão possuem a resolução de 320x240. Essa avaliação foi dividida em duas partes: quadros por segundo e memória.

Em relação ao número de quadros por segundo, foi avaliado a média dos W^4 e a sua versão modificada foram os mais rápidos com 50 e 54 fps, respectivamente. Os algoritmos IHSL e o algoritmo proposto obtiveram, assim como W^4, ambos com resultados satisfatórios para aplicações em tempo real, com 28 fps. Entretanto, Li et al. [13], Kaewtrakulpong e Bowden [11] e KDE [12] obtiveram resultados um pouco inferiores com 20, 14 e 3 fps respectivamente. A Figura 7 ilustra a comparação de FPS entre os métodos.

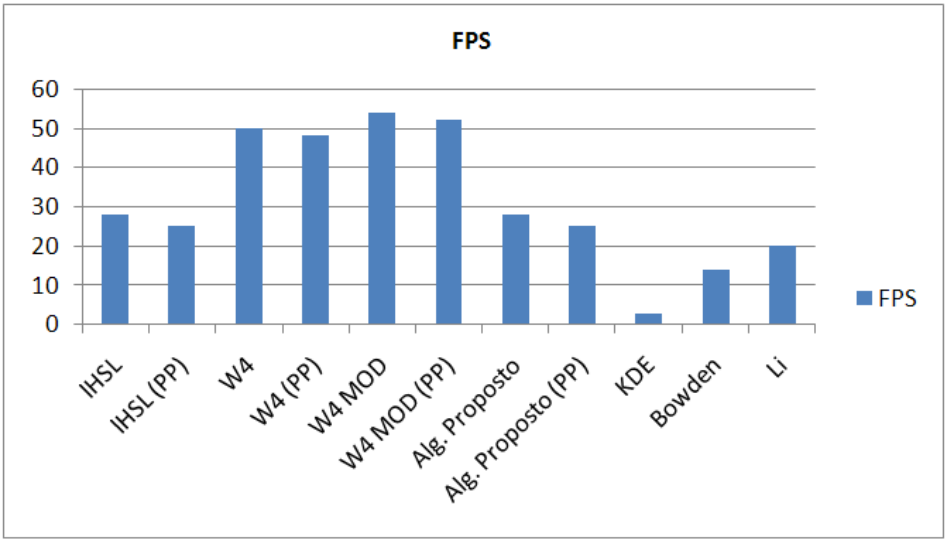


Figura 7: Comparação de FPS dos algotirmos de subtração de fundo

Considerando o consumo de memória, foi levada em consideração somente a quantidade de memória necessária para armazenar o modelo de fundo de cada algoritmo. Os que mais consumiram foram os algoritmos Li et al. [13] com 8500Kb e Kaewtrakulpong e Bowden [11] com 7500Kb. W^4 [1] e KDE [12] estão entre os que menos consumiram com 1000Kb e 800Kb, respectivamente. O algoritmo proposto utiliza 3000Kb para armazenar o seu modelo de fundo. A Figura 8 ilustra um gráfico comparativo entre os algoritmos de subtração de fundo.

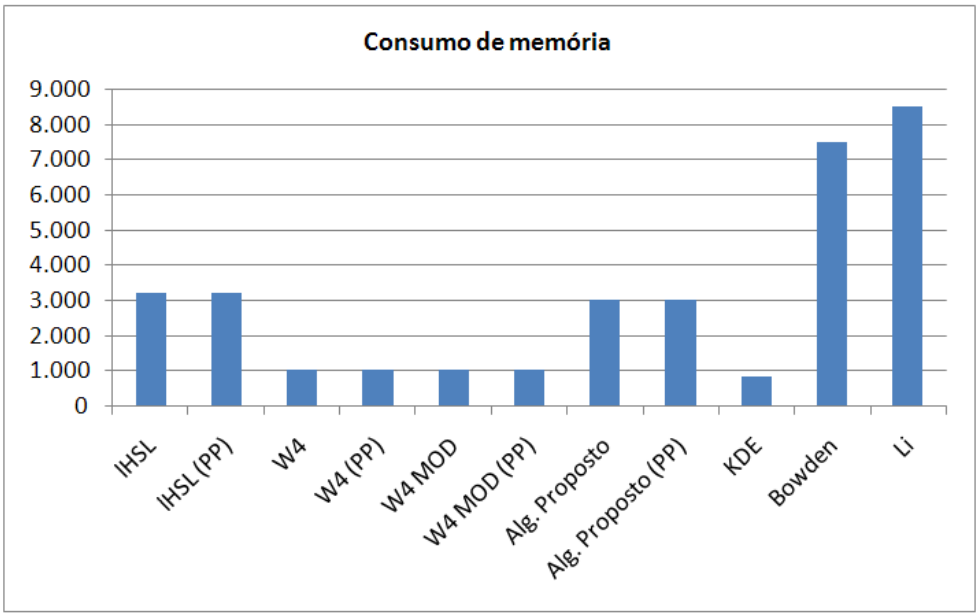


Figura 8: Gráfico comparativo do consumo de memória dos algoritmos de subtração de fundo

2.3 Conclusões parciais

Como conclusão dos resultados, o algoritmo proposto revelou-se bastante eficiente para os vídeos "*Highway* 1" e "*Highway* 2", mas apresentou certa dificuldade na segmentação dos veículos dos vídeos "Passarela 1" e "Passarela 2". Isso ocorreu devido à grande variação de iluminação da cena dado que este algoritmo não apresenta um modelo que se adapte a isso, o que mostra a ineficiência do treinamento inicial. Mesmo assim, o algoritmo proposto obteve os mais altos valores de F pelas métricas de Karaman e Goldmann [14] e, além disso, seu desempenho foi satisfatória para uma aplicação em tempo real, ao contrário dos seus maiores concorrentes como o KDE [12].