SISTEMA DE ESTABILIZAÇÃO DIGITAL DE VÍDEO BASEADO EM ALGORITMO TSS E FILTRO ROBUSTO BDU PARA SISTEMAS ROBÓTICOS

Ayrton G. B. Florêncio*, Marco H. Terra*, Roberto Santos Inoue†, Valdir Grassi Jr*

*Laboratório de Sistemas Inteligentes (LASI), Departamento de Engenharia Elétrica e da Computação, Universidade de São Paulo (USP)
Av. Trabalhador São-Carlense, 400, São Carlos – SP, CEP 13566-590

†Departamento de Engenharia Elétrica, Centro de Ciências Exatas e Tecnologia (CCET), Universidade Federal de São Carlos (UFSCar)
Rodovia Washington Luís, km 235 - SP-310, São Carlos – SP, CEP 13565-905

Emails: ayrtonflorencio@usp.br, terra@sc.usp.br, rsinoue@ufscar.br, vgrassi@usp.br

Abstract - This paper presents a video stabilization system based on digital images of a camera embedded on moving vehicle and unreliable plataform. It is performed by three-step search (TSS) algorithm for estimating the global motion vector. The main contribution of this paper is the application of a robust filter for stabilizating image sequences. A comparative study between the robust filter and the Kalman filter is done for evaluating advantages of using a robust approach.

Keywords - Computer Vision, Three-step search, Extended Kalman filter, BDU filter, Robotic systems.

Resumo - Esse trabalho tem por finalidade apresentar um sistema de estabilização digital de vídeo para uso em plataformas instáveis e veículos em movimento. Isso é realizado através do algoritmo *busca por três etapas* para detectar o movimento global. A principal contribuição deste artigo é a aplicação de um filtro robusto para estabilizar a imagem da câmera. Um estudo comparativo entre o filtro robusto e o filtro de Kalman é realizado para analizar a vantagem de se usar uma abordagem robusta.

Palavras-chave - Visão Computacional, Busca por três etapas, Filtro de Kalman estendido, Filtro BDU, Sistemas robóticos.

1 Introdução

A câmera é um sensor bastante utilizado em sistemas robóticos devido a seu baixo custo, fácil implementação e alta flexibilidade de aplicações, como: *dead-reckogning* (Roggen et al, 2007) baseado em fluxo óptico (Ballard & Brown, 1982); SLAM (*Simultaneous Localization and Mapping*) com multi-câmeras (Bonato, 2008); veículos autônomos guiados por sinais na trajetória (Jia et al, 2009); câmeras embarcadas em aeronaves não tripuladas para capturar características em ambientes rurais (He et al, 2012); entre outros. O objetivo principal deste artigo é a implementação de um sistema para eliminar, ou minimizar, as pertubações contidas em ambientes dinâmicos, tornando a câmera uma fonte de informação mais confiável.

Estabilização de vídeo é um recurso que tem sido buscado para preservar a qualidade de sensoriamento quando a câmera é colocada sobre plataformas instáveis, como é o caso de câmeras embarcadas em veículos em movimento. Os sistemas de estabilização podem ser mecânico, óptico ou digital. Atualmente, os sistemas de estabilização digital são muito procurados devido aos avanços tecnológicos de processadores fazendo este sistema ter baixo custo, boa resposta em tempo real, compacto e alta qualidade.

Para explicar como os sistemas de estabilização digital de vídeo (*Digital Video Stabilization* - DVS) funcionam, é preciso entender os conceitos de movimentos presentes na captura de vídeo. Quando se trabalha em ambientes dinâmicos é possível observar três tipos de movimentos capturados pela câmera: movimento de objetos presentes no campo visual, movimento intencional e não-intencional da câmera. O sistema DVS faz a estimativa desses três movimentos como um só, que é chamado de vetor de movimento global (GMV) e, em seguida, são separados devendo eliminar o movimento não-intencional, este chamado de vetor de movimento indesejável (*hand-shake motion vector* - HMV).

Os sistemas de estabilização digital de vídeo (DVS) trabalham capturando quadros (*frames*) em sequência devendo deslocar o *frame* decorrente para a posição mais adequada, tomando o *frame* anterior como referência. O deslocamento é realizado no plano da imagem através de procedimentos de filtragem para eliminar o HMV (ver Seção 4).

Outro objetivo deste trabalho consiste na utilização de métodos de baixo custo computacional para serem utilizados em sistemas robóticos, em tempo real. Nesse contexto, o algoritmo escolhido para estimar o GMV foi o *three-step search* (TSS) (Kim & Shoi, 1998) em virtude de sua simplicidade de implementação. Em seguida foram utilizados o filtro de Kalman extendido (FKE) (Ge & Lewis, 2006) e o filtro robusto *Bounded Data Uncertainties* (BDU) (Sayed, 2001) para implementar duas versões do sistema DVS. O FKE fornece ótimo desempenho devido sua capacidade de lidar com sistemas imersos em sinais ruidosos e por ter um algoritmo simples sem a necessidade

de armazenamento de grandes bancos de dados. Além dessas características do FKE, o filtro BDU apresenta também robustez às incertezas paramétricas.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta a arquitetura do sistema DVS; a Seção 3 explica o funcionamento do algoritmo TSS; a Seção 4 explica o funcionamento e a aplicabilidade dos filtros; a Seção 5 mostra e comenta alguns resultados experimentais do sistema completo.

2 Arquitetura do sistema DVS

Seguem abaixo o fluxograma (Figura 1) e uma breve explicação das etapas do sistema DVS proposto:

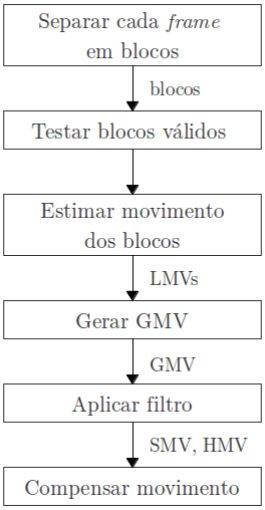


Figura 1: Fluxograma da arquitetura do sistema DVS proposto.

A primeira etapa consiste na divisão do *frame* atual em vários blocos menores, no qual cada um contém 16×16 *pixels*. Essa técnica é utilizada para obter um GMV mais preciso através do vetor de movimento de bloco, popularmente chamado de vetor de movimento local (*local motion vector* - LMV). Alguns blocos não são usados para gerar o GMV devido ao alto grau de incerteza da estimativa do LMV. Tais blocos são classificados como alta suavidade ou alta rugosidade, que serão detalhados na Seção 3. Em seguida os LMV's são calculados através da estimativa de movimento de cada bloco válido, que é realizado pelo algoritmo TSS. O GMV é gerado através de uma função associada aos LMV's estimados, que pode ser visualizado em (4) na Seção 3. Em seguida aplica-se a filtragem com a intenção de encontrar um movimento suave (*smooth motion vector* - SMV) a partir do GMV. Para esse sistema DVS adota-se o SMV para representar a estimativa do movimento intencional da câmera (ver Seção 4). Com o HMV produzido através do SMV e do GMV, a compensação de movimento é realizada empregando uma função do HMV para deslocar cada *frame* em sequência no vídeo (Seção 4.3).

3 *Three-step Search* (TSS)

Existem vários métodos eficazes que fazem boa estimativa do GMV, entre eles podemos citar o *three-step search* (TSS) (Kim & Shoi, 1998), *new three-step search* (Li et al, 1994), *gray-coded* (Ko et al, 1999), *diamond search* (DS) (Tham et al, 1998), entre outros. Nesse trabalho foi utilizado o algoritmo TSS por causa da simplicidade de implementação.

3.1 Metodologia

Primeiro, para validar os blocos é preciso testar se estes contêm alta rugosidade ou alta suavidade, que é definido pelo cálculo da diferença absoluta entre *pixels* adjacentes contidos no bloco. As equações (1) e (2) mostram uma alternativa para o cálculo, onde n é o número de linhas e colunas por bloco, $b^{(i,j)}$ corresponde ao *pixel* de linha i e coluna j no mesmo bloco, *th1* e *th2* são *thresholds* pré-definidos que delimitam as condições de alta suavidade e alta rugosidade, respectivamente.

$$\sqrt{\sum_{i=2}^{n} |b^{(i,j)} - b^{(i-1,j)}| + \sum_{j=2}^{n} |b^{(i,j)} - b^{(i,j-1)}|} < th_1 \tag{1}$$

$$\sqrt{\sum_{i=2}^{n} |b^{(i,j)} - b^{(i-1,j)}| + \sum_{j=2}^{n} |b^{(i,j)} - b^{(i,j-1)}|} > th_2 \tag{2}$$

Se o bloco é válido, estima-se o LMV procurando um bloco dentro de uma região de busca que assemelha-se com o bloco do *frame* anterior. Essa procura é realizada pelo algoritmo TSS, que executa uma busca eficiente onde só é necessário vinte e cinco locais de comparação dentro de uma região de busca de $(2w+n) \times (2w+n)$ *pixels*. Esse w representa o deslocamento máximo do bloco entre *frames* consecutivos. A análise comparativa dessas regiões apontadas pelo TSS e o bloco do instante anterior é medida pelo grau de associação entre eles através da expressão (3):

$$\rho_{k,k-1} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(b_k^{(i,j)} - \bar{b}_k)(b_{k-1}^{(i,j)} - \bar{b}_{k-1})}{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(b_k^{(i,j)} - \bar{b}_k)^2}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(b_{k-1}^{(i,j)} - \bar{b}_{k-1})^2}} \tag{3}$$

onde $\rho_{k,k-1}$ é o coeficiente de correlação, $b_k^{(i,j)}$ representa *pixels* do bloco do *frame* atual, $b_{k-1}^{(i,j)}$ representa *pixels* do bloco do *frame* anterior, $\bar{b}$ é a média dos valores dos *pixels* do bloco em tratamento e k é o número do *frame* que está sendo analisado. Assim, o algoritmo TSS evolui procurando o maior valor de $\rho_{k,k-1}$. A Figura 2(b) mostra um exemplo da aplicação do algoritmo TSS, que é executado em três etapas descritas abaixo:

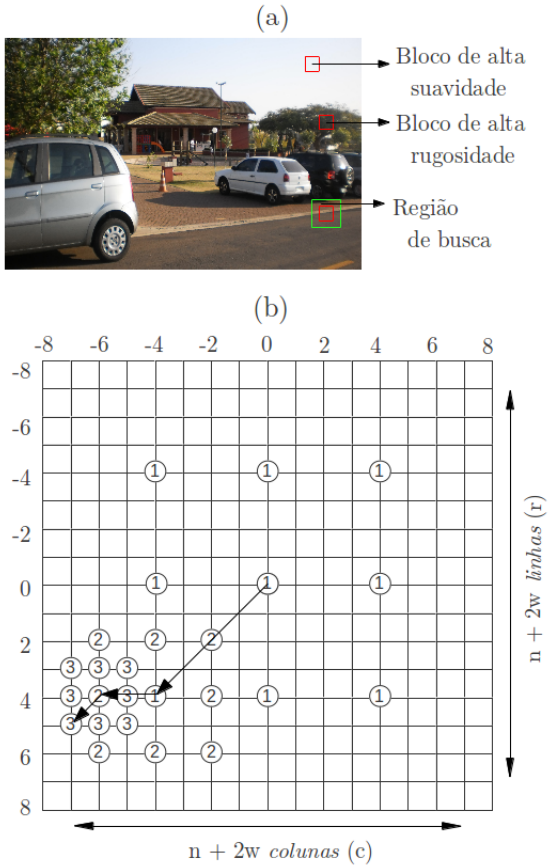


Figura 2: Em (a) blocos de alta suavidade e alta rugosidade são exemplificados em uma imagem retirada de um vídeo. Em (b) mostra a região de busca de um bloco e o procedimento que determina o LMV para $w = 7$. O maior $\rho_{k,k-1}$ foi encontrado na posição $(r, c) = (5, -7)$, ou seja, o bloco do *frame* atual foi deslocado cinco linhas pra baixo e sete colunas pra esquerda com relação ao bloco de posição $(r, c) = (0, 0)$ do *frame* anterior. Portanto, $(r, c) = (5, -7)$ é o vetor de deslocamento.

1 Marcar 9 pontos espaçados de $\lceil w/2 \rceil$ entre si, sendo um deles a posição inicial do bloco $(r, c) = (0, 0)$. Busca-se a região com o maior $\rho_{k,k-1}$ através de (3);

2 Marcar 8 pontos espaçados de $\lceil w/4 \rceil$ entre si ao redor do ponto escolhido na primeira etapa. Novamente é buscada a região com o maior $\rho_{k,k-1}$ através de (3);

3 Mesmo processo que a etapa 2, sendo que a marcação dos pontos são espaçados de $\lceil w/8 \rceil$. Encerrar busca.

Esse procedimento é feito para todos os blocos válidos contidos no *frame* decorrente. O LMV é definido pelo vetor de deslocamento do bloco associado ao coeficiente de correlação: $\begin{bmatrix} r_k & c_k & \rho_{k,k-1} \end{bmatrix}^T$, no qual o armazenamento do coeficiente de correlação ajuda a melhorar a estimativa do GMV com o cálculo do valor esperado, visto em (4), que minimiza a percepção da movimentação de objetos na cena, resultando somente o movimento intencional e não-intencional da câmera.

$$GMV_k = \frac{\sum_{i=1}^{N} \rho_{k,k-1} LMV_k^{(r_k,c_k)}}{\sum_{i=1}^{N} \rho_{k,k-1}} \qquad (4)$$

Em (4), $LMV^{(r,c)}$ representa o vetor de deslocamento do bloco e N é o número de blocos válidos.

4 Filtragem

O movimento intencional da câmera é calculado pelo processo de filtragem aplicado no GMV. O resultado da filtragem deve eliminar movimentos bruscos provenientes dos LMV's com alta magnitude e calcular a posição mais adequada que reproduz o movimento intencional da câmera.

Há vários métodos para obter esses resultados, dentre eles podemos citar o filtro IIR adaptativo (Yang & Jheng, 2006), transformada de Fourier discreta (DFT) (Erturk & Dennis, 2000), filtro de Kalman (Brown, 1996), etc. Este artigo foca na implentação de filtros baseados no filtro de Kalman buscando melhor desempenho de acordo com o ambiente de trabalho. Assim, foi feito um estudo comparativo do filtro de Kalman estendido (FKE) e do Filtro robusto *Bounded Data Uncertainties* (BDU).

4.1 Filtro de Kalman estendido (FKE)

O FKE é uma modificação do algoritmo do filtro de Kalman para aplicações em sistemas não lineares, que pode ser implementado pelo seguinte algoritmo

$$\hat{x}_{k+1|k} = F_k \hat{x}_{k|k}$$

$$P_{k+1|k} = F_k P_{k|k} F_k^T + G_k Q_k G_k^T$$

$$K_{k+1} = P_{k+1|k} H_{k+1}^T \left[H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1} \right]^{-1}$$

$$P_{k+1|k+1} = \left[I - K_{k+1} H_{k+1} \right] P_{k+1|k}$$

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1} \left[z_{k+1} - h(\hat{x}_{k+1|k}) \right]$$

onde $\hat{x}_{k+1|k}$ é a variável de estado, $P_{k+1|k}$ é a covariância de estado, K_{k+1} é o ganho do filtro, z é a variável de medida e k é o índice decorrente, ou seja, o *frame* que está sendo analisado. Já F_k, G_k, Q_k, H_k e R_k são matrizes constantes, ver apêndice. Para maiores detalhes sobre o FKE veja Ge & Lewis, 2006.

4.2 *Filtro BDU*

Este filtro considera incertezas de modelagem (Sayed, 2001). Essas incertezas afetam F_k, G_k, Q_k, R_k e $P_{k+1|k}$. As componentes fundamentais do algoritmo BDU são as seguintes:

$$\hat{R}_{k+1} = R_k - \hat{\lambda}_k^{-1} H_{k+1} M_k M_k^T H_{k+1}^T$$

$$\hat{P}_{k|k} = \left(P_{k|k}^{-1} + \hat{\lambda}_k N_{F_k}^T N_{F_k} \right)^{-1}$$

$$\hat{G}_k = G_k - \hat{\lambda}_k F_k \hat{P}_{k|k} N_{F_k}^T N_{G_k}$$

$$\hat{Q}_k^{-1} = Q_k^{-1} + \hat{\lambda}_k N_{G_k}^T \left[I + \hat{\lambda}_k N_{F_k} P_{k|k} N_{F_k}^T \right]^{-1} N_{G_k}$$

$$\hat{F}_k = (F_k - \hat{\lambda}_k \hat{G}_k \hat{Q}_k N_{G_k}^T N_{F_k})(I - \hat{\lambda}_k \hat{P}_{k|k} N_{F_k}^T N_{F_k})$$

onde

$$\hat{\lambda}_k = (1 + \alpha_f).\|M_k^T H_{k+1}^T R_{k+1}^{-1} H_{k+1} M_k\|,$$

N_F, N_G, M e α_f são modelados pelo projetista. Veja todos os detalhes deste algoritmo em Sayed, 2001, tabela 1.

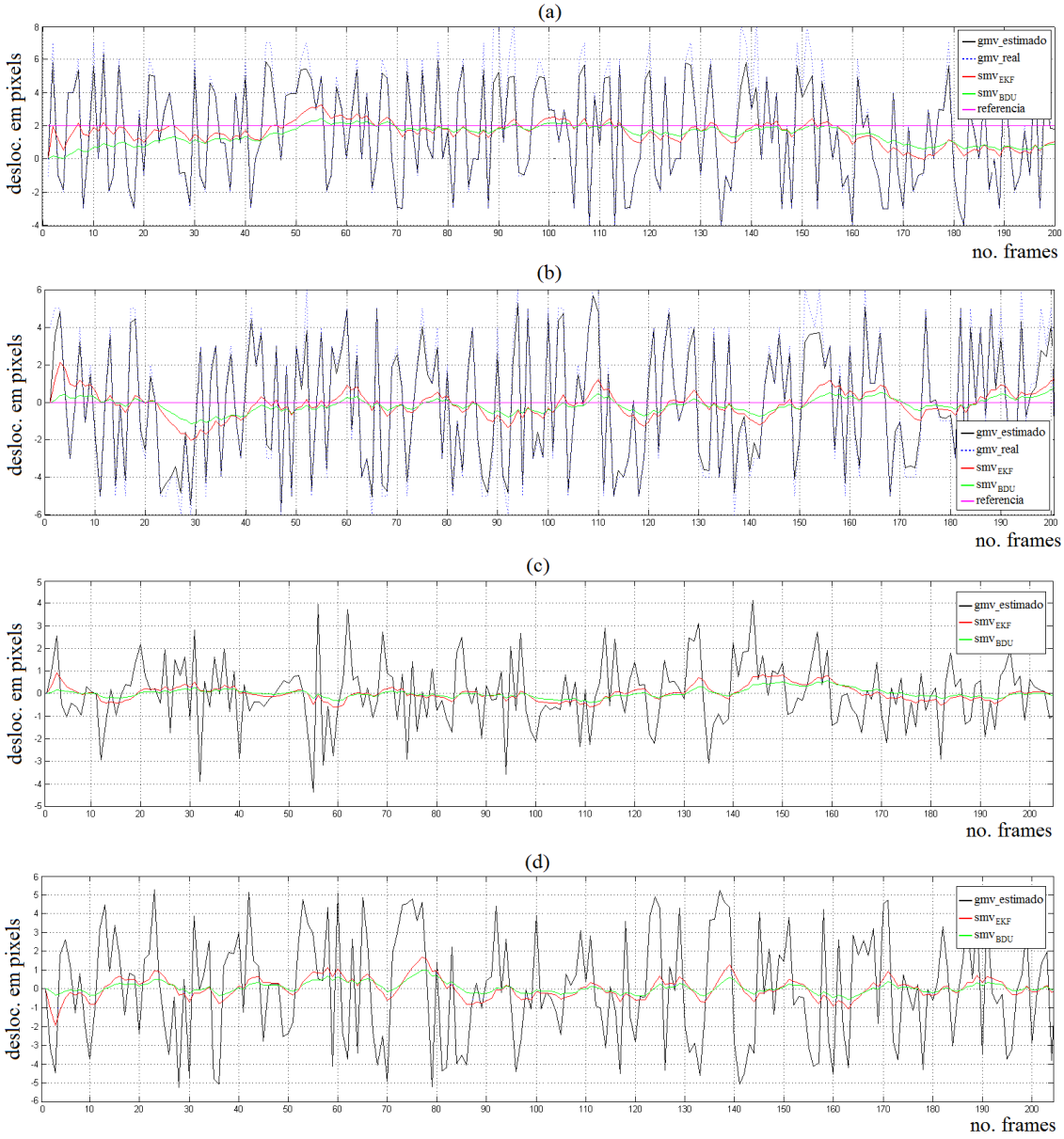


Figura 3: Os gráficos acima mostram o desempenho do algoritmo TSS para estimar o GMV e o desempenho do filtro de Kalman estendido e do filtro robusto BDU no cálculo do SMV. Os gráficos (a) e (b) são referentes aos movimentos vertical (em linhas de pixels) e horizontal (em colunas de pixels) da câmera, respectivamente, ao longo da sequência de 200 *frames* do vídeo da vista da cidade de novo hamburgo (Figura 4(a)). Já os gráficos (c) e (d) mostram os mesmos dados com relação ao vídeo de uma câmera dentro de um carro em movimento (Figura 4(b)).

4.3 Compensação de movimento

Os filtros retornam o movimento suavizado (SMV) da câmera. Os resultados dessas filtragens podem ser vistos nos gráficos da Figura 3.

Por fim, o HMV é separado do GMV pela subtração do SMV, conforme abaixo.

$$HMV_k = GMV_k - SMV_k \tag{5}$$

O vídeo está sujeito a erro cumulativo por não fazer compensação mecânica de movimento, ou seja, não há uma força aplicada na câmera para corrigir o movimento não-intencional real. Portanto, o HMV é acumulado (AMV) a cada *frame* na sequência do vídeo e deve-se corrigir o movimento no instante decorrente com o mesmo AMV, como demonstrado em (7).

$$AMV_k = \sum_{i=2}^{k} HMV_i \tag{6}$$

$$I_k'^{(i,j)} = I_k^{(i-AMV(1),j-AMV(2))} \tag{7}$$

Em (7), I representa o *frame* capturado pela câmera e I' representa o *frame* estabilizado, $AMV(1)$ e $AMV(2)$ são as componentes vertical e horizontal, respectivamente, do AMV que são aplicados em todos os *pixels* de I.

5 Resultados experimentais

A fim de avaliar os algoritmos do sistema DVS, um dos experimentos foi baseado na produção de um vídeo feito artificialmente a partir de uma foto de alta resolução aplicando deslocamento constante junto a um sinal ruidoso onde ambos são conhecidos. Tal procedimento é útilizado como referência para testar a qualidade dos algoritmos propostos.

Os experimentos foram feitos em dois ambientes diferentes: sendo o primeiro uma visão da cidade de Novo Hamburgo (Figura 4(a)) e o segundo dentro de um carro em movimento (Figura 4(b)).

O primeiro vídeo foi produzido artificialmente pelo software Matlab em um computador e o segundo vídeo foi gravado por uma câmera digital colocada em um suporte sobre o painel de um carro. Os dois vídeos possuem resolução de 640×480 *pixels* e taxa de 30 *frames* por segundo. Todos os algoritmos foram compilados no g++ com o auxílio da biblioteca OpenCV (Open Source Computer Vision Library).

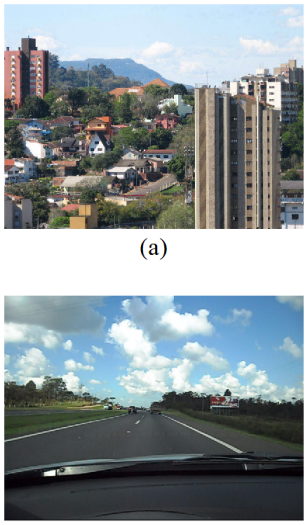
Enquanto o vídeo da cidade causa fluxo óptico translacional, o vídeo do carro resulta em fluxo óptico divergente e, para os dois casos, o LMV e o respectivo GMV foram calculados e ocasionaram boa estimativa dos mesmos, como pode ser verificado na Figura 3(a) e (b) mostram a comparação entre o GMV real que foi aplicado artificialmente e o GMV estimado pelo algoritmo TSS.

Em alguns momentos é possível observar movimentos rotacionais no vídeo gravado no carro. Logo, dependendo da aplicação, é necessário um algoritmo capaz de compensar os tipos de movimentos presentes causados pelas pertubações do ambiente.

A comparação entre o FKE e o filtro BDU foi realizada calculando-se a variância do erro de estimativa (ver Figura 5) utilizando a seguinte aproximação:

$$\varepsilon\|x_k - \hat{x}_k\| \approx \frac{1}{T}\sum_{j=1}^{T} \|x_k^{(j)} - \hat{x}_k^{(j)}\| \tag{8}$$

onde $\|.\|$ é a norma euclidiana, x é o movimento intencional real, $\hat{x}$ é o SMV para cada filtro e T é o número de experimentos.



(a)

(b)

Figura 4: Vídeo construído artificialmente a partir de uma foto de alta resolução da cidade de novo hamburgo (a) e um vídeo mostrando o percurso de um carro em uma rodovia (b).

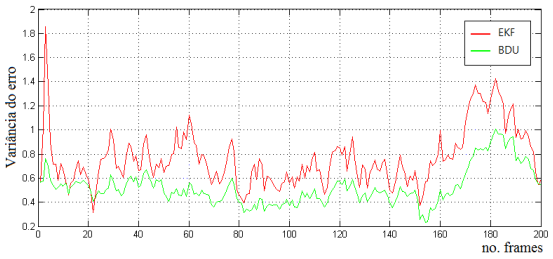


Figura 5: Variância do erro dos filtros. Quatro experimentos são considerados para cada filtro ($T = 4$).

De acordo com os gráficos da Figura 3 há diferenças nos resultados das filtragens. O filtro robusto foi menos sensível do que o FKE aplicado e, a partir da Figura 5, o filtro robusto teve menor variância de erro aproximando-se ao movimento intencional de referência (desejado) com maior destreza, comparado ao FKE.

6 Conclusão e trabalho futuro

Este artigo teve como objetivo mostrar a eficiência dos algoritmos descritos na Seção 3 e Seção 4. Esse sistema DVS mostra qualidade e confiabilidade para trabalhar com sensoriamento para aplicações que dependem das pertubações do ambiente.

Os experimentos realizados neste trabalho mostram que o comportamento do filtro BDU é mais confiável para restaurar o movimento intencional da câmera.

Como mencionado na Seção 1, em ambientes dinâmicos há movimentação de objetos que são capturados pela câmera. Se essa movimentação ocupa boa parte do campo visual da câmera, pode haver erros graves na estimativa do GMV. Dessa forma será colocada uma IMU (*Inertial Measurement Unit*) junto à câmera para resolver o problema mencionado acima e fazer adaptações e melhorias no algoritmo de estimativa de movimento buscando robustez para empregá-lo em diversos ambiente.

Agradecimentos

Ao conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo auxílio financeiro.

Referências

Roggen, D., Jenny R, de la Hamette, P., & Troster, G. (2007). Mapping by seeing: wearable vision-based dead-reckoning, and closing the loop. EuroSSC 2007, lecture notes in computer science, pp. 29-45.

Ballard, D. H. & Brown, C. M. (1982). Computer Vision. Prentice Hall.

Bonato, V. (2008). Proposta de uma arquitetura de hardware em FPGA implementada para o SLAM com multi-câmeras aplicada à robótica móvel. Tese (Doutorado), ICMC, Universidade de São Paulo, São Carlos.

Jia, Z., Balasuriya, A., & Challa, S. (2009). Vision based target tracking for autonomous land vehicles navigation: a brief survey. Academic Journal - United Technologies Research Center, vol. 2, no. 1, pp. 32-42.

He, Q., Chu, C. H., & Camargo, A. (2012). Architectural Building Detection and Tracking under Rural Environment in Video Sequences Taken by Unmanned Aircraft System (UAS). *Worldcomp*'12.

Kim, J. N. & Shoi, T. S. (1998). A fast three-step search algorithm with minimun checking points using unimodal error surface assumption. *IEEE Trans. on Consumer Electronics*, vol. 44, No. 3, pp. 638-648.

Ge, S. S. & Lewis, F. L. (2006). *Autonomous Mobile Robos Sensing, Control, Decision Making and Applications*. Taylor & Francis Group.

Sayed, A. H. (2001). A framework for state-space estimation with uncertain models. *IEEE Transaction on Automatic Control*, vol. 46, pp. 998-1013.

Li, R., Aeng, B., & Liou, M. L. (1994). A new three-step search algorithm for block motion estimation. *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 438-442.

Ko, S.-J., Lee, S.-H., Jeon, S.-W., & Kang, E.-S. (1999). Fast digital image stabilizer based on gray-coded bit-plane matching. *IEEE Transactions on Consumer Electronics*, vol. 45, no. 3, pp. 598-603.

Tham, J. Y., Ranganath, S., Ranganath, M., & and Kassim, A. A. (1998). A novel unrestricted center-biased diamond search algorithm for block motion estimation. *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 369-377.

Yang, S. H. & Jheng, F. M. (2006). An adaptive image stabilization technique. In Proceedings of the *IEEE International Conference on Systems, Man, and Cybernetics* (SMC2006), pp. 1968-1973.

Erturk, S. & Dennis, T. J. (2000). Image sequence stabilisation based on DFT filtering. *IEE-Proc. Vision, Image and Signal Processing*, vol. 147, no. 2, pp. 95-102.

Brown, R. G. (1996). Introduction to Random Signals and Applied Kalman Filtering. Wiley, 3 edition, 1996.

Apêndice

Abaixo estão os valores utilizados nas matrizes constantes dos filtros:

$$F_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ G_k = Q_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$H_k = \begin{bmatrix} 1 & -1 \end{bmatrix},\ R_k = \begin{bmatrix} 64 & 0 \\ 0 & 64 \end{bmatrix},$$

$$M = 1.67,\ \alpha_f = 0.5,\ N_F = 0.1,\ N_G = 0.1.$$