Kỷ yếu Hội nghị Quốc gia lần thứ X về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR), Đà Nẵng, ngày 17-18/08/2017
DOI: 10.15625/vap.2017.00067

MỘT PHƯƠNG PHÁP KHỬ NHIỄU VIDEO DỰA TRÊN PHÉP LẶP BREGMAN

Đặng Ngọc Hoàng Thành[1], Nguyễn Hoàng Hải[2]

[1]Trường Cao đẳng Công nghiệp Huế – 70 Nguyễn Huệ, TP. Huế.

[2]Trường Đại học Sư phạm Đà Nẵng - 459 Tôn Đức Thắng, Q. Liên Chiểu, TP. Đà Nẵng.

dnhthanh@hueic.edu.vn, hoanghai@ued.udn.vn

TÓM TẮT: *Bài báo này nhằm đề xuất một phương pháp khử nhiễu video dựa trên việc áp dụng phép lặp Bregman cho mô hình ROF cải tiến. Mô hình ROF truyền thống chỉ khử nhiễu trên ảnh kĩ thuật số dạng xám (grayscale). Thông qua việc cải tiến mô hình ROF, chúng tôi đã tạo ra một phiên bản khác có thể khử nhiễu cho dữ liệu video (bao gồm cả video màu và video không màu). Bên cạnh đó, phép lặp Bregman là một phương pháp hiệu quả để giải bài toán tối ưu lồi. Tính đặc thù của các phương pháp lặp dựa trên phép lặp Bregman là tốc độ thực thi nhanh hơn so với các phương pháp lặp truyền thống. Do đó, nó thích hợp để thực thi trên dữ liệu video vốn yêu cầu khối lượng tính toán lớn. Phương pháp khử nhiễu video được đề xuất có thể được nâng cấp trong tương lai để khử nhiễu trên các video thời gian thực.*

Từ khóa*: khử nhiễu video, phép lặp Bregman, nhiễu số, mô hình ROF.*

I. GIỚI THIỆU

Trong lĩnh vực xử lý ảnh và video thì bài toán khôi phục ảnh/video (image/video restoration) là một trong những bài toán quan trọng và có ứng dụng rộng rãi trong các lĩnh vực khoa học, kĩ thuật, y sinh v.v. Nhóm bài toán khôi phục ảnh/video có thể được phân ra làm các dạng: bài toán khử nhiễu (image/video denoising), bài toán khử nhòa (image/video deblurring), bài toán khôi phục thông tin bị mất (image/video inpainting). Một số dạng bài toán như tăng/giảm độ phân giải cho ảnh/video để hiển thị trên các kích thước màn hình khác nhau (upsampling, downsampling), phóng đại ảnh/video (image/video zooming) v.v. cũng có thể được xếp vào dạng bài toán khôi phục bởi trong quá trình thực thi các tác vụ phóng to, thu nhỏ, tăng/giảm độ phân giải sẽ làm cho cấu trúc thông tin lưu trữ trên ảnh/video bị thay đổi hoặc bị mất.

Trong thực tế, các ảnh kĩ thuật số cũng như video kĩ thuật số được tạo ra thường chứa nhiễu. Có khá nhiều nguyên nhân tạo ra nhiễu trong trường hợp này: do điều kiện thiếu sáng, hiệu chỉnh thông số ISO quá cao, do chất lượng của các cảm biến, do nhiệt độ môi trường cao v.v. Trong quá trình truyền tải thông tin, các ảnh và video được truyền tải trên một đường truyền không đảm bảo cũng sẽ làm cho một phần thông tin trên ảnh hoặc video bị mất và tạo ra nhiễu. Trường hợp này được gọi là nhiễu do tính hiệu đường truyền. Trong bất kì trường hợp nào, nhiễu cũng sẽ làm giảm chất lượng của ảnh và video kĩ thuật số. Do đó, cần thiết phải loại bỏ nó.

Bài toán khử nhiễu là một bài toán quan trọng và được nhiều nhà khoa học tập trung nghiên cứu. Để giải quyết bài toán khử nhiễu này, có khá nhiều hướng tiếp cận được áp dụng, nhưng có thể được chia ra làm hai nhóm cơ bản: nhóm hướng tiếp cận dựa trên tối ưu rời rạc và nhóm hướng tiếp cận dựa trên tối ưu liên tục. Trong bài báo này, chúng tôi sẽ sử dụng hướng tiếp cận biến phân (là một dạng của tối ưu liên tục) kết hợp với phép lặp Bregman để nâng cao tốc độ thực thi của quá trình khử nhiễu.

Hướng tiếp cận biến phân được áp dụng rộng rãi trong xử lý ảnh và video. Với bài toán khử nhiễu thì cũng không phải là ngoại lệ. Người đầu tiên đã áp dụng thành công hướng tiếp cận này cho bài toán khử nhiễu trên ảnh kĩ thuật số là Rudin. Ông cùng các đồng nghiệp của mình đã tạo ra một mô hình mang tên ROF (Rudin-Osher-Fatemi) [1-3]. Mô hình ROF dù đã ra đời nhiều thập kỉ, nhưng tính hiệu quả trong việc khử nhiễu mà nó mang lại vẫn được đánh giá rất cao.

Từ mô hình ROF, nhiều công trình nghiên cứu khác cũng đã được thực hiện để cải tiến mô hình ROF. Bản thân mô hình ROF được xây dựng tập trung vào loại bỏ nhiễu Gauss trên ảnh kĩ thuật số. Đây là một dạng nhiễu phổ biến trên hầu hết các camera kĩ thuật số. Tuy nhiên, nhiễu trên ảnh X-quang trong y khoa lại là nhiễu Poisson. Dù rằng mô hình ROF vẫn làm việc hiệu quả trong trường hợp này, nhưng tồn tại nhược điểm: các đường biên sẽ không được bảo toàn tốt. Triet Le và các đồng nghiệp đã tạo ra một mô hình cải tiến để nâng cao hiệu quả khử nhiễu Poisson [4]. Một dạng nhiễu khác cũng khá phổ biến trong ảnh y sinh là tổ hợp nhiễu Gauss và Poisson. Để loại bỏ loại nhiễu này, Dang N.H. Thanh và Dvoenko Sergey đã đề xuất một mô hình kết hợp [5-7]. Mô hình này làm việc hiệu quả cho cả trường hợp tổ hợp nhiễu và các nhiễu riêng biệt.

Tuy làm việc hiệu quả, nhưng mỗi phương pháp khử nhiễu nêu trên đều thể hiện nhược điểm của nó. Nhược điểm này thể hiện ở giải thuật được thiết kế để giải quyết mô hình. Phần lớn các thuật toán được đề xuất để giải quyết các mô hình nêu trên đều được xây dựng dựa trên kĩ thuật sai phân vốn có tốc độ thực thi không đủ nhanh. Điều này rất khó áp dụng trên dữ liệu video (vốn yêu cầu khối lượng tính toán lớn). Nếu giải thuật đề xuất không đủ nhanh thì sẽ mất rất nhiều thời gian để khử nhiễu. Đặc biệt, nếu muốn khử nhiễu video thời gian thực, thì thuật toán cần phải xử lý

đủ nhanh. Với các video thời gian thực có độ phân giải cao như HD 1080, 2K, 4K v.v. thì thuật toán cần phải thực thi cực kì nhanh mới đảm bảo quá trình xử lý không có độ trễ.

Trong công trình nghiên cứu của mình, Rudin cùng các đồng nghiệp đã đề xuất mô hình ROF để khử nhiễu trên ảnh kĩ thuật số dạng grayscale (ảnh xám) và phương pháp để giải mô hình ROF được đề xuất dựa trên kĩ thuật sai phân nâng bậc. Nhược điểm lớn của kĩ thuật sai phân nâng bậc cũng như các kĩ thuật sai phân thông thường chính là ở tốc độ tính toán. Với khối lượng tính toán rất lớn như dữ liệu video, thì kĩ thuật sai phân tỏ ra không thực sự hiệu quả.

Trong bài báo này, chúng tôi sẽ tiến hành cải tiến mô hình ROF để nó có thể hoạt động tốt trên dữ liệu video (cả video dạng xám grayscale và video màu) và đề xuất một giải thuật khử nhiễu trên video để nâng cao tốc độ tính toán dựa trên phép lặp Bregman [8] – là một phương pháp được đề xuất để giải bài toán tối ưu lồi. Phương pháp này sẽ được áp dụng cho dữ liệu video trong thư viện của Matlab. Một phiên bản cải tiến đủ nhanh để áp dụng cho video thời gian thực dựa trên sự kết hợp của phép lặp Bregman và kĩ thuật tính toán song song sẽ được chúng tôi nghiên cứu trong tương lai.

Để đánh giá tính hiệu quả của mô hình, chúng tôi tập trung vào đánh giá chất lượng của việc khử nhiễu trên video thông qua hai chỉ số định lượng PSNR [7] và định tính SSIM [7]. Đối với thị giác người, thì chỉ số SSIM là quan trọng hơn. Tuy nhiên, chỉ số PSNR lại cho chúng ta cái nhìn toàn cảnh từ góc độ của sai số toán học. Do đó, nó cũng là một chỉ số nên được khai thác. Chỉ số PSNR càng lớn thì chất lượng ảnh/video càng gần với chất lượng của ảnh/video gốc. Giá trị SSIM càng lớn thì chất lượng khôi phục ảnh/video càng cao. Bên cạnh đánh giá chất lượng khôi phục, chúng tôi cũng sẽ thảo luận đến tốc độ tính toán và khả năng áp dụng để xử lý video thời gian thực.

II. PHƯƠNG PHÁP KHỬ NHIỄU TRÊN VIDEO

A. Mô hình khử nhiễu trên video

1. Mô hình ROF khử nhiễu trên ảnh kĩ thuật số

Mô hình ROF [1-3] được Rudin và các đồng nghiệp đề xuất dựa trên biến phân đầy đủ có dạng sau đây:

$$\min_u \left(\int_\Omega |\nabla u| dxdy + \frac{\lambda}{2} \int_\Omega (v-u)^2 dxdy \right),$$

Trong đó, $u, v \in \mathbb{R}^2$ – lần lượt là ảnh lí tưởng không chứa nhiễu và ảnh thực nhận có chứa nhiễu. Ω – là một miền bị chặn trong $\mathbb{R}^2$, toán tử ∇u – vector gradient, chuẩn $|\nabla u| = \sqrt{\left(\partial u/\partial x\right)^2 + \left(\partial u/\partial y\right)^2}$, tham số λ – là tham số cân bằng giữa độ mượt (theo biến phân đầy đủ) và độ khử nhiễu.

Bản thân mô hình hình ROF được đề xuất để khử nhiễu Gauss trên ảnh kĩ thuật số dạng grayscale. Để mô hình ROF có thể làm việc với dữ liệu vào là video (bao gồm cả video màu và video không màu) thì chúng tôi sẽ tiến hành cải tiến mô hình ROF.

2. Mô hình ROF cải tiến để khử nhiễu trên video

Giả sử, trong không gian $\mathbb{R}^2$, cho trước một video lí tưởng $\boldsymbol{u}$ (không nhiễu, không bị phá hủy bởi bất kì tác nhân ngoại cảnh nào) và một video thực nhận $\boldsymbol{v}$ đã bị làm nhiễu bởi các yếu tố ngoại cảnh. Các video này là một dãy tuần tự của các khung hình liên tiếp nhau:

$$\boldsymbol{u}(x,y) = \{u_1(x,y), u_2(x,y), \dots, u_n(x,y)\},$$
$$\boldsymbol{v}(x,y) = \{v_1(x,y), v_2(x,y), \dots, v_n(x,y)\},$$

Trong đó, n là số khung hình của video. Trong thực tế, mỗi video thường có tốc độ chuẩn là 24fps (frames per second). Con số này xuất phát từ khả năng lưu ảnh của mắt người. Điều đó có nghĩa là tổng số khung hình của video chính là tốc độ chuẩn nhân với thời gian trình chiếu (tính theo giây). Tùy vào độ mượt của video, người ta có thể tăng tốc độ chuẩn lên. Các video thông dụng hiện nay thường sử dụng tốc độ chuẩn là 30fps. Một số video cần trình chiếu ở chế độ quay chậm (slow motion) thì tốc độ chuẩn này có thể được tăng lên, ví dụ 60fps. Một số camera hỗ trợ trên các thiết bị di động ngày nay cũng đã hỗ trợ đến tốc độ 60fps.

Giá trị x, y là chỉ số theo chiều ngang và theo chiều dọc của ma trận điểm ảnh. Với mỗi khung hình k, giá trị $u_k(x,y)$ hay $v_k(x,y)$ phản ánh giá trị màu tại điểm ảnh có tọa độ (x,y). Trong mô hình khử nhiễu mà chúng tôi nghiên cứu, mô hình màu được sử dụng là RGB. Do đó, mỗi giá trị $u_k(x,y)$ có thể được phân tách theo ba kênh màu riêng biệt là:

$$u_k(x,y) = \left(u_k^R(x,y), u_k^G(x,y), u_k^B(x,y)\right),$$

với R, G, B lần lượt là các chỉ số tương ứng với với ba kênh màu Red, Green và Blue.

Không làm giảm tính tổng quát, ta có thể xét từng kênh màu riêng biệt của từng khung hình riêng biệt, nghĩa là:

$$\boldsymbol{u} = u_k^c(x,y), \boldsymbol{v} = v_k^c(x,y), k = 1,\dots,n; c = \{R,G,B\}.$$

Một mô hình khử nhiễu theo biến phân đầy đủ (Total Variation) sẽ có dạng [2-3]:

$$\min_{\boldsymbol{u}} J(\boldsymbol{u}) + \frac{\lambda}{2}\|\boldsymbol{v}-\boldsymbol{u}\|_X^2.$$

Trong đó, J là một phiến hàm bị chặn theo biến phân đầy đủ, X là một không gian Hilbert. Trong thực tế, bài toán khử nhiễu sẽ cho kết quả tốt nhất nếu X là không gian L_2. Tham số λ – là tham số cân bằng giữa độ mượt (theo biến phân đầy đủ) và độ khử nhiễu.

Phiến hàm J được sử dụng trong mô hình ROF là phiến hàm dựa trên biến phân đầy đủ. Do đó, chúng tôi cũng sẽ sử dụng phiến hàm này cho trường hợp khử nhiễu trên video:

$$J(u) = \|\boldsymbol{u}\|_{TV} = \int_\Omega |\nabla \boldsymbol{u}| dxdy.$$

Nếu sử dụng kết hợp chuẩn theo TV và chuẩn theo L_2 thì mô hình khử nhiễu trên video theo biến phân đầy đủ có thể viết lại một cách tường minh dưới dạng sau đây:

$$\min_{u_k^c}\left(\int_\Omega |\nabla u_k^c| dxdy + \frac{\lambda}{2}\int_\Omega (u_k^c - v_k^c)^2 dxdy\right);\ k = 1,\dots,n; c = \{R,G,B\}, \quad (1)$$

Trong đó, Ω – là một miền bị chặn trong $\mathbb{R}^2$, toán tử ∇u_k^c – vector gradient, chuẩn $|\nabla u_k^c| = \sqrt{\left(\partial u_k^c/\partial x\right)^2 + \left(\partial u_k^c/\partial y\right)^2}$, $k = 1,\dots,n; c = \{R,G,B\}$.

3. Một số đánh giá về mô hình khử nhiễu trên video

Trong mô hình (1) ở trên, các hàm u_k^c và v_k^c được xét trên mỗi khung hình và trên mỗi kênh màu riêng biệt, với chỉ số k cho khung hình và chỉ số c cho các kênh màu. Điều đó có nghĩa là ta cần phải giải một hệ gồm có $3n$ bài toán tối ưu bộ phận. Bởi vì giá trị xám của các điểm ảnh là độc lập trên mỗi khung hình và độc lập trên mỗi kênh màu, cho nên khi tiến hành giải, chúng tôi chỉ đề xuất thuật toán cho một bài toán tối ưu bộ phận. Các thuật toán giải cho các bài toán tối ưu bộ phận còn lại hoàn toàn tương tự như thuật toán giải cho bài toán tối ưu bộ phận đã đề xuất.

Trọng số λ được sử dụng để tạo ra sự cân bằng giữa hai phần: phần làm mượt (phần chứa biến phân đầy đủ và là hạng tử đầu tiên trong mô hình (1)) và phần khử nhiễu (phần chứa độ lệch giữa hai khung hình tương ứng của video gốc và video bị nhiễu hay hạng tử thứ hai trong mô hình (1)). Nếu giá trị của λ quá nhỏ thì các khung hình sẽ bị làm mượt nhiều và dẫn đến một số chi tiết nhỏ sẽ bị mất. Nếu giá trị λ quá lớn, thì biên giữa các vùng màu sẽ không được bảo toàn khi thực hiện quá trình khử nhiễu. Giá trị này sẽ được tính toán trong quá trình lặp để đảm bảo có sự cân bằng giữa hai yếu tố mượt và khử nhiễu.

Bài toán tối ưu mô hình (1) sẽ tồn tại nghiệm và nghiệm đó là duy nhất. Điều này đã được chứng minh trong các công trình [1-3] về tính tồn tại và duy nhất nghiệm của bài toán chuẩn hóa theo biến phân đầy đủ (TV regularization). Tính chất này sẽ đảm bảo cho các phép lặp dựa trên các lược đồ sai phân luôn hội tụ.

B. Phương pháp lặp Bregman cho mô hình khử nhiễu trên video

1. Phép lặp Bregman cho mô hình khử nhiễu trên video

Ta sẽ khảo sát phần biến phân đầy đủ trong mô hình (1) trước tiên. Phần biến phân đầy đủ này có thể được rời rạc hóa như sau:

$$\|\boldsymbol{u}\|_{TV} = \int_\Omega |\nabla \boldsymbol{u}| dxdy \approx \sum_{i=0}^{N-1}\sum_{j=0}^{M-1} |\nabla \boldsymbol{u}_{ij}|,$$

trong đó, N, M lần lượt là số điểm ảnh theo chiều ngang và theo chiều dọc của mỗi khung hình trên video. Kết hợp với phần khử nhiễu trong mô hình (1), ta sẽ thu được mô hình rời rạc như sau:

$$\min_{d,\boldsymbol{u}}\left(\sum_{i,j}|d_{ij}| + \frac{\lambda}{2}\sum_{i,j}(\boldsymbol{v}_{ij} - \boldsymbol{u}_{ij})^2\right),$$

với ràng buộc:

$$d = \nabla \boldsymbol{u}.$$

Bằng cách sử dụng thêm phần tử d ta sẽ hạn chế sự phụ thuộc trực tiếp giữa hai phần làm mượt và khử nhiễu trong mô hình (1). Điều này là cần thiết để thực thi phép lặp Bregman.

Giải thuật 1. Phép lặp Bregman chính tắc cho mô hình (1).

Khởi gán với $\boldsymbol{u}^{(0)} \in \mathbb{R}^2, b^{(0)} = 0$.

Lặp theo $q = 0, 1, ...$

$$\boldsymbol{u}^q = \text{argmin}_{d,u}\left(\sum_{i,j}(|d_{ij}|) + \frac{\lambda}{2}\sum_{i,j}(\boldsymbol{v}_{ij} - \boldsymbol{u}_{ij})^2 + \frac{\gamma}{2}\sum_{i,j}|d_{ij} - \nabla\boldsymbol{u}_{ij} - b_{ij}^q|^2\right), \quad (2)$$

$$b^{q+1} = b^q + \nabla\boldsymbol{u} - d.$$

Trong đó, $\gamma > 0$ là hệ số chắn (penalty parameter) để đảm bảo tính hội tụ của phép lặp.

Trong công trình nghiên cứu [8], Goldstein và Osher đã chứng minh rằng, khi áp dụng phép lặp Bregman cho phiến hàm dạng:

$$\min_{\boldsymbol{u}}\big(E(u) + \lambda H(u)\big),$$

với E và H là các phiến hàm lồi trên $\mathbb{R}^2$, phiến hàm H là khả vi, thì khi đó phép lặp Bregman cho bài toán trên là hội tụ. Trong công trình [1], Rudin cũng đã chỉ ra rằng phiến hàm theo TV là lồi. Ta cũng dễ dàng chứng minh được hạng tử thứ hai trong (1) cũng là phiến hàm lồi và khả vi. Điều đó có nghĩa là phép lặp Bregman cho mô hình (1) sẽ luôn hội tụ. Với bài toán chuyển đổi (2), khi bổ sung vào tham số chắn, thì bài toán (2) hội tụ với mọi $\gamma > 0$.

Với mỗi bước lặp trong giải thuật 1 ta cần giải một bài toán tối ưu (2). Điều này có thể được giải quyết bởi phép phân chia Bregman. Ý tưởng của phép phân chia Bregman:

+ Phương 1. Giải bài toán con theo d với $\boldsymbol{u}$ cố định:

$$\underset{d}{\text{argmin}}\left(\sum_{i,j}|d_{ij}| + \frac{\gamma}{2}\sum_{i,j}|d_{ij} - \nabla\boldsymbol{u}_{ij} - b_{ij}|^2\right).$$

Ta nhận được nghiệm:

$$d_{ij} = \frac{\nabla\boldsymbol{u}_{ij} + b_{ij}}{|\nabla\boldsymbol{u}_{ij} + b_{ij}|}\max\left\{|\nabla\boldsymbol{u}_{ij} + b_{ij}| - \frac{1}{\gamma}, 0\right\}.$$

+ Phương 2. Giải bài toán con theo u với d cố định:

$$\underset{u}{\text{argmin}}\left(\frac{\lambda}{2}\sum_{i,j}(\boldsymbol{u}_{ij} - \boldsymbol{v}_{ij})^2 + \frac{\gamma}{2}\sum_{i,j}|\nabla\boldsymbol{u}_{ij} - d_{ij} + b_{ij}|^2\right).$$

Ta nhận được phương trình sau:

$$\lambda(\boldsymbol{u} - \boldsymbol{v}) + \gamma\nabla^*(\nabla\boldsymbol{u} - d + b) = 0, \quad (3)$$

Trong đó $\nabla^*(\boldsymbol{u}_n) = \boldsymbol{u}_{n-1} - \boldsymbol{u}_n$.

Theo kết quả nghiên cứu [8], bài toán theo phương 2 được giải quyết hoàn toàn bằng phương pháp lặp được đề xuất bởi Goldstein và Osher. Khi đó, giải thuật 1 có thể được viết lại dưới dạng sau đây:

Giải thuật 2. Phép lặp Bregman đầy đủ cho mô hình (1)

Khởi gán $\boldsymbol{u} = 0, d = 0, b = 0$.

Lặp nếu $\|\boldsymbol{u}_{hiện\ tại} - \boldsymbol{u}_{trước}\| > \epsilon$ (với ϵ – sai số cho trước).

Giải bài toán theo phương 1.

Giải bài toán theo phương 2.

$b = b + \nabla\boldsymbol{u} - d.$

2. Lựa chọn tham số γ

Trong sơ đồ lặp để giải bài toán theo phương 2, ta nhận được điều kiện để lựa chọn giá trị của tham số γ. Tốc độ hội tụ của các bài toán con theo hai phương cũng như tốc độ hội tụ của toàn bộ phép lặp sẽ phụ thuộc vào giá trị của tham số γ này. Trong các thực nghiệm, giá trị γ được lựa chọn là giá trị cố định $\gamma = 5$ thì phép lặp Bregman sẽ hội tụ đủ nhanh.

3. Lựa chọn tham số λ

Việc lựa chọn tham số λ tương tự như cách mà Rudin đã thực thi trên mô hình ROF. Đó là dựa vào phương sai của nhiễu Gauss σ^2. Điều đó có nghĩa là việc tìm tham số λ sẽ được thực hiện thông qua giải bài toán tối ưu:

$$\min_{\boldsymbol{u}} \int_{\Omega} |\nabla \boldsymbol{u}| dxdy \text{ , với ràng buộc } \int_{\Omega} (\boldsymbol{v}-\boldsymbol{u})^2 dxdy = \sigma^2 |\Omega|.$$

Bài toán này đã được Chambolle đề xuất thuật toán giải [9] như sau:

Giải thuật 3. Tìm tham số λ.

Lặp

Giải bài toán $\boldsymbol{u} = \text{argmin}_u \int_{\Omega} |\nabla \boldsymbol{u}| dxdy + \frac{\lambda}{2} \int_{\Omega} (\boldsymbol{v}-\boldsymbol{u})^2 dxdy$,

$\lambda^{(k+1)} = \frac{\lambda^{(k)}}{\sigma} \sqrt{\int_{\Omega} (\boldsymbol{v}-\boldsymbol{u})^2 dxdy}$.

Điều đó có nghĩa là tiến trình lặp để tìm λ sẽ được tiến hành song song với các bước lặp trong phép lặp Bregman. Để tìm ra giá trị λ tối ưu, sẽ mất thêm một khoảng thời gian tính toán. Do đó, trong thực tế, chúng tôi cũng lựa chọn giá trị cho tham số $\lambda = 2$. Giá trị này là phù hợp và làm cân bằng giữa hai thành phần khử nhiễu và làm mượt. Việc cố định giá trị λ sẽ giúp thuật toán thực thi nhanh hơn và phù hợp hơn với khối lượng tính toán lớn trên dữ liệu video.

III. ĐÁNH GIÁ CHẤT LƯỢNG VIDEO

Để đánh giá chất lượng của video sau khi khử nhiễu, ta sẽ tiến hành so sánh chỉ số PSNR [7] và SSIM [7] của video sau khi khử nhiễu với video gốc và của video nhiễu với video gốc. Việc so sánh sẽ tiến hành trên một số khung hình được lựa chọn và thông qua giá trị trung bình của toàn bộ các khung hình.

Tiêu chuẩn PSNR được xác định theo công thức sau:

$$PSNR = 10 \lg \left(\frac{NML^2}{\sum_{i=1}^{N} \sum_{j=1}^{M} (\boldsymbol{v}_{ij} - \boldsymbol{u}_{ij})^2} \right),$$

trong đó, L – mức xám của các pixel trên ảnh hoặc video, thông thường với mức xám 8 bit, thì giá trị này là 255. Giá trị PSNR càng lớn thì chất lượng của ảnh/video càng cao. Thông thường giá trị này trên 30 thì chất lượng ảnh/video được đánh giá là tốt.

Tiêu chuẩn SSIM được xác định theo công thức sau:

$$SSIM = \frac{(2\overline{\boldsymbol{u}}\overline{\boldsymbol{v}} + C_1)(2\sigma_{\boldsymbol{uv}} + C_2)}{(\overline{\boldsymbol{u}}^2 + \overline{\boldsymbol{v}}^2 + C_1)(\sigma_{\boldsymbol{u}}^2 + \sigma_{\boldsymbol{v}}^2 + C_2)},$$

Trong đó,

$$\overline{\boldsymbol{u}} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \boldsymbol{u}_{ij}, \overline{\boldsymbol{v}} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \boldsymbol{v}_{ij}, \sigma_{\boldsymbol{u}}^2 = \frac{1}{NM-1} \sum_{i=1}^{N} \sum_{j=1}^{M} (\boldsymbol{u}_{ij} - \overline{\boldsymbol{u}})^2, \sigma_{\boldsymbol{v}}^2 = \frac{1}{NM-1} \sum_{i=1}^{N} \sum_{j=1}^{M} (\boldsymbol{v}_{ij} - \overline{\boldsymbol{v}})^2,$$

$$\sigma_{\boldsymbol{uv}} = \frac{1}{NM-1} \sum_{i=1}^{N} \sum_{j=1}^{M} (\boldsymbol{u}_{ij} - \overline{\boldsymbol{u}})(\boldsymbol{v}_{ij} - \overline{\boldsymbol{v}}), C_1 = (K_1 L)^2, C_2 = (K_2 L)^2, K_1 \ll 1, K_2 \ll 1, K_1 = K_2 = 10^{-6}.$$

Giá trị SSIM thường dùng để đánh giá chất lượng ảnh hoặc video thông qua việc so sánh tính tương tự của hai ảnh hoặc các khung hình tương ứng của hai video. Cách thức đánh giá của tiêu chuẩn SSIM tương tự với cách cảm nhận của thị giác người. Do đó, phương pháp này được đánh giá cao. Thông thường, giá trị SSIM nằm trong khoảng từ -1 đến 1. Giá trị này càng lớn, thì chất lượng ảnh/video càng cao.

IV. THỰC NGHIỆM

Trong phần thực nghiệm tính toán, trước tiên chúng tôi sẽ áp dụng giải thuật nêu trên đối với video *shuttle.avi* trong thư viện của Matlab. Đây là video màu, có kích thước khung hình là 512x288 pixel và có 121 khung hình. Tốc độ trình chiếu là 30fps.

Đầu tiên, chúng tôi sẽ bổ sung nhiễu Gauss vào video với giá trị $\sigma = 0{,}05$ đồng đều cho cả ba kênh màu R, G và B. Tiếp theo, chúng tôi sẽ tiến hành khử nhiễu trên video thu được. Kết quả khử nhiễu trên một số khung hình được đưa ra ở hình 1. Trong bảng 1 chúng tôi đưa ra các giá trị theo đánh giá PSNR và SSIM trên một số khung hình.

Từ kết quả đưa ra ở bảng 1, ta có thể thấy rằng kết quả khử nhiễu trên các khung hình của video thu được là rất tốt thể hiện qua trị số PSNR và SSIM vượt xa so với trị số PSNR và SSIM của video nhiễu. Giá trị trung bình của trị số PSNR và SSIM của video nhiễu và video sau khi khử nhiễu trên toàn bộ 121 khung hình lần lượt là: 26,0623, 0.6579, 35.2964 và 0.9662.

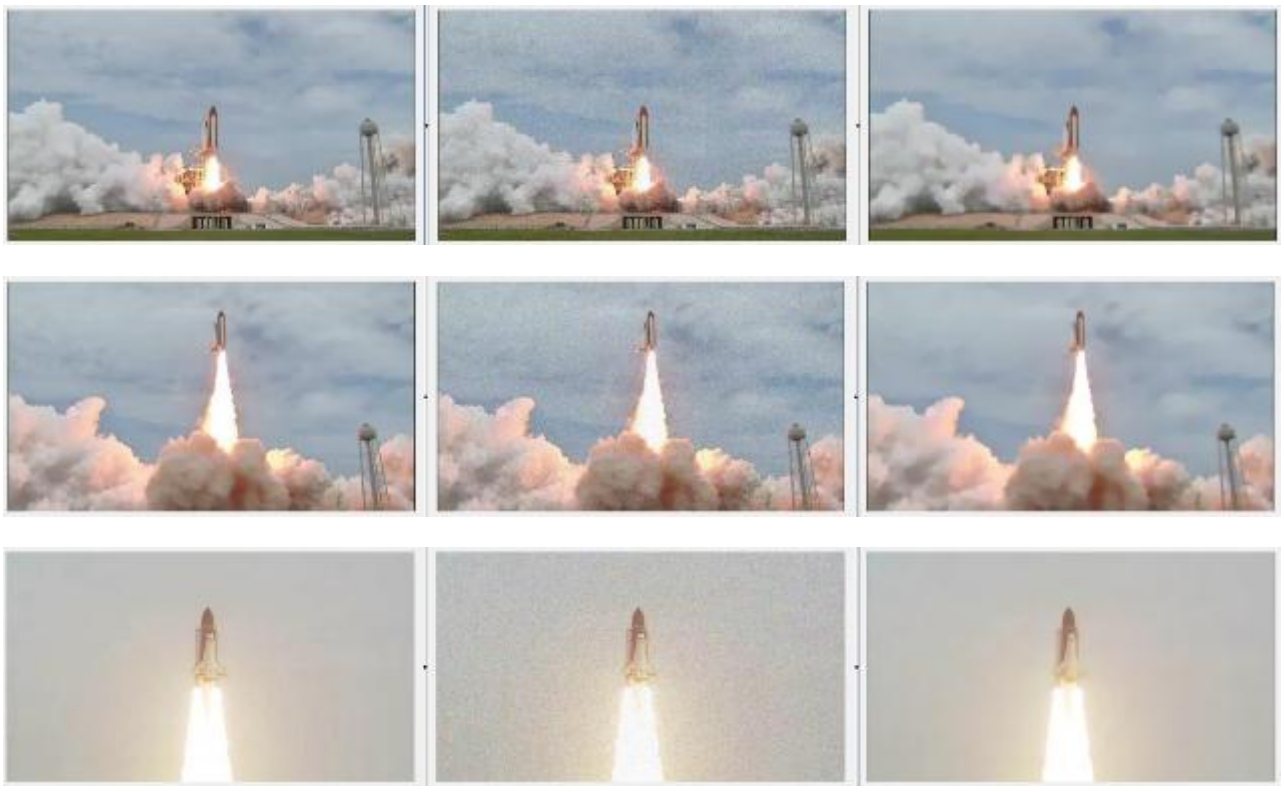
Bảng 1. Bảng trị số PSNR và SSIM trên một số khung hình

Tiêu chuẩn \ Khung hình		1	10	20	50	100	120
Nhiễu	**PSNR**	26.0493	26.0281	26.0450	26.0548	26.0818	26.1192
	SSIM	0.6690	0.6720	0.6752	0.6843	0.6351	0.4380
Khử nhiễu	**PSNR**	34.0036	33.9397	34.0171	34.2436	36.8901	42.1138
	SSIM	0.9966	0.9577	0.9584	0.9612	0.9757	0.9860

Kết quả thực nghiệm trên được cài đặt trên hệ điều hành Windows 10, Chip Intel core i5, tốc độ 1.6Ghz và 2.29Ghz và RAM 4Gb. Thời gian chạy thuật toán khử nhiễu mất 15 giây. Ta có thể thấy rằng, với khoảng thời gian này thì thuật toán chưa thể áp dụng trong thời gian thực, bởi thời gian xử lý dài gấp ba lần so với độ dài trình chiếu của video (khoảng 5 giây). Nếu áp dụng thuật toán downsampling để giảm độ phân giải của video xuống HQ 360 thì thời chạy chương trình giảm xuống còn 4 giây. Với chất lượng video dạng này thì có thể áp dụng khử nhiễu trong thời gian thực. Tuy nhiên, chất lượng video HQ 360 là thấp so với các tiêu chuẩn mới nhất về chất lượng video hiện nay (đạt đến 4K).

Thực nghiệm thứ hai chúng tôi tiến hành trên video *rhinos.avi*. Đây cũng là video nằm trong thư viện của Matlab. Nó cũng là video màu, có kích thước khung hình là 320x240 pixel (gần HQ 360). Tốc độ khung hình là 15fps. Tổng số khung hình là 114. Các thông số nhiễu được bổ sung hoàn toàn tương tự như trường hợp 1.

Kết quả khử nhiễu trên video *rhinos.avi* được đưa ra trong hình 2. Trong trường hợp này, chúng tôi chỉ đưa ra giá trị trung bình của PSNR và SSIM cho toàn bộ 114 khung hình mà không đưa ra bảng thống kê trên một số khung hình chọn lọc. Đối với video bị làm nhiễu, giá trị PSNR và SSIM trung bình lần lượt là: 26.2911 và 0.5651. Đối với video sau khi khử nhiễu, giá trị PSNR và SSIM trung bình lần lượt là: 33.3089 và 0.9195.



Hình 1. Kết quả khử nhiễu trên các khung hình 1, 91 và 119 của video *shuttle.avi* (tương ứng với các dòng 1, 2, 3 từ trên xuống dưới); theo thứ tự từ trái sang phải: video gốc, video bị làm nhiễu, video sau khi khử nhiễu.

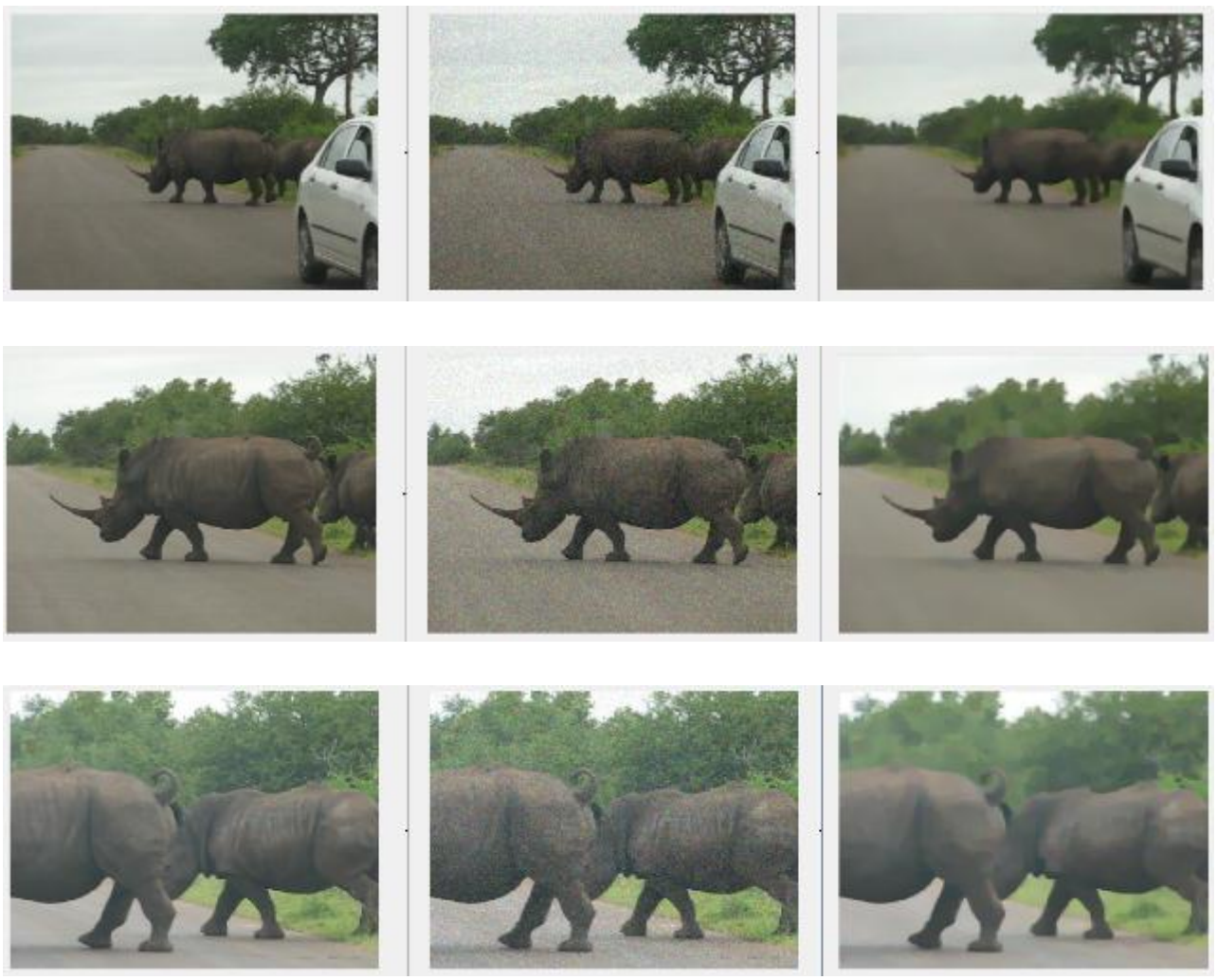
Đối với trường hợp này, chúng tôi sử dụng cùng cấu hình máy tính ở trên để cài đặt thuật toán. Thời gian thực thi chương trình là 6 giây. Nếu so sánh với độ dài trình chiếu của video là 7 giây thì với chất lượng video dạng này (gần HQ 360), thuật toán có thể áp dụng xử lý trong thời gian thực. Cũng cần lưu ý rằng, tốc độ trình chiếu của video *rhinos.avi* là 15fps chỉ bằng ½ so với tốc độ trình chiếu của video *shuttle.avi* (30fps).

Bảng 2. Bảng trị số PSNR trung bình (APSNR) và SSIM trung bình (ASSIM) của toàn bộ các khung hình trên một số video

Tên video		***fil_cat.avi***	***shaky_car.avi***	***viplanedeparture.avi***	***vipmen.avi***	***vipmosaicking.avi***
Độ phân giải		240x176	320x240	360x240	160x120	320x240
Tốc độ fps		24	30	30	30	15
Số khung hình		241	132	337	283	71
Độ dài (giây)		10	4	11	9	4
Nhiễu	**APSNR**	30.0078	26.3066	26.2780	26.2704	26.3299
	ASSIM	0.9319	0.4722	0.6025	0.8349	0.7777
Khử nhiễu	**APSNR**	34.9685	32.1933	35.8093	31.9454	33.2224
	ASSIM	0.9763	0.8548	0.9678	0.9542	0.9516
Thời gian xử lý (giây)		8	4	11	4	3

Thực nghiệm thứ ba chúng tôi tiến hành trên năm video được lựa chọn từ thư viện của Matlab. Trong số các video này, chỉ có video *shaky_car.avi* là video dạng xám grayscale. Bốn video còn lại đều là các video màu. Các thông số của về độ phân giải, tốc độ trình chiếu, số khung hình, độ dài trình chiếu cũng như kết quả đánh giá chất lượng khử nhiễu theo trị số PSNR trung bình và SSIM trung bình trên toàn bộ các khung hình của mỗi video được đưa ra trong bảng 2. Các thông số nhiễu được bổ sung hoàn toàn tương tự như hai trường hợp nêu trên. Cấu hình máy tính để cài đặt thuật toán cũng không có sự thay đổi.

Từ các thực nghiệm nêu trên, ta có thể thấy rằng tuy thuật toán có tốc độ thực thi đủ nhanh để xử lý dữ liệu video, nhưng vẫn còn chưa đủ nhanh để khử nhiễu video thời gian thực. Thuật toán chỉ mới đủ khả năng xử lý video thời gian thực với chất lượng từ HQ 360 trở xuống thì khi đó mới không có độ trễ. Với chất lượng cao hơn, cần thiết phải tăng cấu hình máy tính cài đặt thuật toán cũng như cần thiết phải áp dụng kĩ thuật tính toán song song.



Hình 2. Kết quả khử nhiễu trên các khung hình 1, 11 và 31 của video *rhinos.avi* (tương ứng với các dòng 1, 2, 3 từ trên xuống dưới); theo thứ tự từ trái sang phải: video gốc, video bị làm nhiễu, video sau khi khử nhiễu.

V. KẾT LUẬN

Phương pháp được đề xuất để khử nhiễu trên video cho kết quả tốt về mức độ khử nhiễu trên dữ liệu video. Thông qua giá trị PSNR và SSIM ta có thể thấy kết quả thu được tốt về cả mặt định lượng (qua tiêu chí PSNR) và cả định tính (qua tiêu chí SSIM).

Tốc độ thực thi của giải thuật đủ nhanh để có thể áp dụng cho dữ liệu video vốn yêu cầu khối lượng tính toán tương đối lớn. Tuy nhiên, phương pháp đề xuất vẫn chưa đạt tốc độ đủ nhanh để có thể khử nhiễu video thời gian thực chất lượng cao hơn HQ 360.

Thông qua kết quả thu được, chúng tôi cũng đưa ra một số biện pháp để nâng cao hơn nữa thời gian thực thi: kết hợp với kĩ thuật xử lý song song, tăng cấu hình cho các máy tính tham gia tính toán. Chúng tôi kì vọng với những biện pháp nêu trên, phương pháp được đề xuất này có thể áp dụng để khử nhiễu cho video thời gian thực chuẩn HD 720.

TÀI LIỆU THAM KHẢO

[1] Rudin L. I., Osher S., Fatemi E., "Nonlinear total variation based noise removal algorithms", Physica D, vol. 60, pp. 259-268, 1992.

[2] Chang T. F, Shen J., Image processing and analysis: Variational, PDE, Wavelet, and stochastic method, SIAM, 2005.

[3] Chen K., "Introduction to variational image processing models and application", International Journal of Computer mathematics, vol. 90, no. 1, pp. 1-8, 2013.

[4] Triet Le, Chartrand R. Asaki T. J., "A variational approach to reconstructing images corrupted by Poisson noise", Journal of mathematical imaging and vision, vol. 27, is. 3, pp. 257-263, 2007.

[5] Thanh D. N. H., Dvoenko S. D., "A method of Total variation to remove the mixed Poisson-Gaussian noise", Pattern Recognition and Image Analysis, vol. 26, is. 2, pp. 285-293, 2016.

[6] Thanh D. and Dvoenko S., "A Variational Method to Remove the Combination of Poisson and Gaussian Noises", Proceedings of the 5th International Workshop on Image Mining. Theory and Applications (IMTA-5-2015), pp. 38-45, Berlin-2015.

[7] Thanh D. N. H., Dvoenko S. D., Sang D. V., "A Mixed noise removal method based on Total variation", Informatica, vol. 40, is. 2, pp. 159-167, 2016.

[8] Tom Goldstein, Stanley Osher, "The split Bregman method for L1-Regularized problem", SIAM Journal Imaging sciences, vol. 2, no. 2, pp. 323-343, 2009.

[9] A. Chambolle, "An Algorithm for Total Variation Minimization and Applications", Journal of Mathematical Imaging and Vision, vol. 20, is. 1, pp. 89-97, 2004.

VIDEO DENOISING METHOD BASED ON THE BREGMAN ITERATION

Dang Ngoc Hoang Thanh, Nguyen Hoang Hai

ABSTRACT: *In this paper, we propose a video denoising method based on the Bregman iteration and the modified ROF model. The traditional ROF model can only remove noise on grayscale digital images. By modifying the ROF model, we created a new version of the traditional ROF model that can be applied to remove noise on video (include colorful and colorless videos). Otherwise, the Bregman iteration is an effective method to solve the convex optimization problems. The characteristic of these iterations types is faster than other traditional iterations. Hence, this proposed method is suitable for removing noise on video data. This data type always requires a vast number of calculations. The proposed method can be upgraded to denoise on real-time videos.*