Tăng tốc độ phát hiện dị thường trên ảnh đa phổ và siêu phổ ứng dụng trong tìm kiếm cứu nạn

Nguyễn Văn Phương, Đào Khánh Hoài, Tống Minh Đức
Học viện Kỹ thuật Quân sự, Hà Nội

Tác giả liên hệ: Nguyễn Văn Phương, phuongnv@mta.edu.vn
Ngày nhận bài: 14/06/2019, ngày sửa chữa: 27/10/2019, ngày duyệt đăng: 27/10/2019
Định danh DOI: 10.32913/mic-ict-research-vn.v2019.n2.866
Biên tập lĩnh vực điều phối phản biện và quyết định nhận đăng: TS. Phan Anh Huy

Tóm tắt: Máy dò dị thường do Reed và Yu đề xuất được công nhận là máy chuẩn để phát hiện dị thường trên ảnh đa phổ và siêu phổ. Tuy nhiên, máy này có một số hạn chế: dữ liệu ảnh phải tuân theo mô hình Gauss đa biến, tính toán nghịch đảo của ma trận hiệp phương sai rất phức tạp khi ảnh nền có kích thước lớn, hoạt động thiếu ổn định, đôi khi có tỉ lệ báo động giả cao, thiếu mối liên hệ không gian giữa các điểm ảnh. Quy tắc quyết định Neyman-Pearson thường được sử dụng dựa trên việc tính toán hàm mật độ xác suất phi tham số của dữ liệu nền để nâng cao hiệu suất và độ tin cậy, nhưng lại có độ phức tạp tính toán cao. Để giảm độ phức tạp tính toán và thời gian tính toán, nhiều phương pháp đã được sử dụng, như: biến đổi Fourier nhanh, biến đổi Gauss nhanh, lập trình đa luồng trên bộ xử lý trung tâm (CPU), song song trên bộ xử lý đồ họa (GPU). Bài báo này trình bày một phương pháp ước lượng nhanh hàm mật độ xác suất bằng cách phân nhóm các điểm ảnh trên miền giá trị và tổ chức dữ liệu trên cây Kd-tree. Kết quả kiểm nghiệm cho thấy phương pháp đề xuất vượt trội các phương pháp khác và có thể ứng dụng trong thực tế.

Từ khóa: *Tăng tốc độ phát hiện dị thường, Kd-tree, ước lượng mật độ phi tham số.*

Title: **Acceleration of Anomaly Detection in Multispectral and Hyperspectral Images for Search and Rescue Situations**

Abstract: Reed-Yu detector is recognized as a standard algorithm for detecting anomalies on multispectral and hyperspectral images. However, this detector has several limitations: image data must follow the multivariate Gaussian model, calculation of the covariance matrix inverse is complex for large size background images is a complex, lack of robustness, high false alarm rates sometimes, lack of spatial correlation among pixels. The Neyman-Pearson detection criterion is often applied on the nonparametric probability density function of the background data for effectiveness and reliability, at the expense of high computational complexity. To reduce the computational complexity, various methods can be applied, such as: fast Fourier transform, fast Gaussian transform, multi-threaded programming on CPU, parallel on GPU. This paper proposes a method for fast estimation of the density by grouping pixels based on the range of pixels and organizing the data using the Kd-tree. The experimental results show that the proposed method outperforms the state-of-the-art methods and can be applied in practice.

Keywords: *Acceleration of anomaly detection, Kd-tree, non-parametric density estimation.*

I. MỞ ĐẦU

Hoạt động tìm kiếm và cứu nạn bao gồm việc tìm kiếm và giải cứu người, phương tiện bị mắc kẹt trong các tình huống khó khăn hoặc báo nạn. Một cách tiếp cận đang ngày càng được sử dụng nhiều trong tìm kiếm cứu nạn là sử dụng ảnh đa phổ hay siêu phổ có độ phân giải cao được thu từ các bộ cảm biến gắn trên máy bay hoặc vệ tinh. Tuy nhiên, các ảnh hưởng bất lợi gây ra bởi đặc trưng của địa hình, điều kiện thời tiết khắc nghiệt làm cho tọa độ báo nạn có dung sai lớn. Các thiết bị cảm biến thu dữ liệu phải quét trên một diện rộng và dung lượng dữ liệu lớn là một rào cản đối với việc tìm kiếm thủ công bằng mắt thường. Các kỹ thuật tiền xử lý dữ liệu và các thuật toán tìm kiếm tự động là giải pháp phù hợp giúp người quan sát nâng cao hiệu suất và tốc độ tìm kiếm.

Tự động phát hiện mục tiêu dựa trên các đặc trưng hình học có thể được sử dụng để tiếp cận vấn đề này. Tuy nhiên, các đặc trưng hình học của các đối tượng quan tâm không được xác định rõ trong hầu hết các tình huống tìm kiếm cứu nạn. Mặc dù trực tiếp tìm ra người đang gặp nạn sẽ là lý tưởng, nhưng trong một số trường hợp các đồ vật đi kèm như quần áo, vật dụng cá nhân, mảnh vỡ phương tiện, v.v. có thể cung cấp một số thông tin hữu ích. Vì vậy, phát hiện

dị thường sẽ cung cấp một cách tiếp cận phù hợp hơn cho vấn đề này. Dị thường trên ảnh đa phổ và siêu phổ được xác định là những điểm ảnh hoặc cụm điểm ảnh có phổ nổi bật hoặc khác biệt nhiều so với những điểm ảnh lân cận. Những điểm ảnh này thường là thưa thớt và hiếm khi đại diện cho ảnh [1]. Nói chung, các dấu hiệu dị thường là rất nhỏ về mặt không gian và tồn tại với xác suất thấp trong một cảnh ảnh.

Trong hơn 20 năm qua, cộng đồng nghiên cứu trên thế giới đã xây dựng rất nhiều bộ dò dị thường để phát hiện các điểm ảnh dị thường trên ảnh đa phổ, siêu phổ. Dựa trên các kỹ thuật khác nhau của các máy dò, dựa trên bốn nhóm giải pháp chính: thống kê, hạt nhân, không gian đặc trưng và phân đoạn [2]. Máy dò dị thường do Reed và Yu xây dựng vào năm 1990 [3] là một trong những máy dò dị thường dựa trên thống kê và được gọi là máy dò RX (RXD). RXD đã khơi nguồn cho rất nhiều thuật toán được phát triển sau này [2] và nó được coi là máy phát hiện dị thường chuẩn cho hình ảnh đa phổ, siêu phổ [4]. Hiệu quả của RXD trong việc phát hiện dị thường từ các ảnh đa phổ và siêu phổ đã được kiểm chứng [1, 3–9]. Mặc dù vậy, RXD có những hạn chế nhất định. Thứ nhất, việc ước lượng nghịch đảo của ma trận hiệp phương sai của dữ liệu nền với kích thước chiều dữ liệu lớn thường rất phức tạp và hoạt động không ổn định [10, 11] dẫn đến làm suy yếu thuật toán. Thứ hai, đôi khi RXD gây ra tỷ lệ báo động giả cao (ví dụ, một cái cây đơn lẻ trong đồng cỏ được phát hiện là dị thường cục bộ ngay cả khi toàn bộ ảnh có cả một khu rừng) [11–14]. Thứ ba, RXD giả định dữ liệu nền tuân theo mô hình Gauss đa biến, nhưng có nhiều trường hợp giả định này có thể không đầy đủ vì trong thực tế các cảnh ảnh rất đa dạng và chứa nhiều lớp đối tượng khác nhau [11, 14, 15]. Thứ tư, RXD thiếu mối liên hệ về không gian, mỗi điểm ảnh được đánh giá riêng lẻ và không quan tâm đến những điểm ảnh xung quanh nó.

Để giảm những hạn chế của RXD, trong một vài năm gần đây các nhà khoa học đã áp dụng quy tắc ra quyết định dựa trên kiểm nghiệm tỷ lệ khả năng (LRT) dựa trên hàm mật độ xác suất (PDF) của dữ liệu nền để phát hiện các dị thường trên ảnh đa phổ và ảnh siêu phổ. Cụ thể, năm 2011 trong nghiên cứu [16] của Veracini và các cộng sự, phương pháp đề xuất sử dụng Parzen Widnow (PW) để ước tính PDF nền đã cho kết quả đáng tin cậy. Sau khi PDF nền được xấp xỉ thông qua PW, nó được dùng làm đầu vào để phát hiện các dấu hiệu dị thường trên ảnh dựa trên kiểm nghiệm tỷ lệ khả năng. Năm 2012, trong nghiên cứu [1], Bolukbasi và cộng sự đã xây dựng kiểm nghiệm giả thuyết nhị phân cho phát hiện dị thường và sử dụng thuật toán KNN để tìm k láng giềng gần nhất để tính hàm mật độ xác suất phi tham số cho điểm ảnh đang xét. Kết quả thu được đã vượt so với RXD. Năm 2014, trong nghiên cứu [17], Matteoli và nhóm tác giả đã đưa ra chiến lược để quyết định một điểm ảnh có phải là dị thường hay là nền dựa trên định lý Neyman-Pearson sử dụng các hàm PDF. Trong đó các tác giả đã kiểm nghiệm trên ba hàm nhân PDF: hạt nhân Gauss cố định băng thông, hạt nhân Gauss không cố định băng thông (VKDE) và tìm kiếm k láng giềng gần nhất, để ước lượng hàm mật độ giống như trong [1]. Kết quả là cả ba hàm nhân PDF trên đều cho ra hiệu suất phát hiện dị thường cao hơn RXD. Năm 2017, trong nghiên cứu [18] của Zhao và các cộng sự, sự kết hợp của các phương pháp ước lượng mật độ phi tham số và phát hiện dựa trên biểu diễn mối quan hệ tương quan (CRD), cho thấy hiệu suất phát hiện dị thường khá cao và đã vượt RXD.

Tuy nhiên, độ phức tạp tính toán của kỹ thuật phi tham số trong ước lượng hàm mật độ xác suất là $O(kn^2)$, trong đó n là số lượng điểm ảnh và k là số kênh phổ, làm cho việc tính toán tốn kém thời gian (trong phần thực nghiệm của bài báo, một ảnh màu ba kênh RGB, kích thước 3396×3349 pixel tốn gần 21 ngày để tính toán) dẫn đến khả năng ứng dụng vào thực tế rất hạn chế, đặc biệt là ứng dụng trong công tác tìm kiếm cứu nạn. Để tăng tốc độ tính toán, giảm thời gian xử lý, một số kỹ thuật gần đúng đã được đề xuất. Đầu tiên, đó là đề xuất của Silverman trong nghiên cứu [19] sử dụng biến đổi Fourier nhanh (FFT) để ước lượng mật độ. Nó làm giảm đáng kể yêu cầu tính toán của phương pháp ước tính mật độ, đã giảm độ phức tạp tính toán từ $O(N^2)$ xuống còn $O(N \log N)$. Một phương pháp khác là áp dụng biến đổi Gauss nhanh (FGT) được Elgammal và các cộng sự đề xuất trong nghiên cứu [20]. Phương pháp này đã giảm độ phức tạp tính toán từ $O(NM)$ xuống còn $O(N + M)$. Trong đó, $N = kn$ là kích thước dữ liệu, và M là số lượng mục tiêu cần tính PDF. Mặc dù cả hai phương pháp FFT và FGT đã giảm độ phức tạp tính toán PDF nhưng đổi lại, việc tính toán gần đúng giảm hiệu suất phát hiện dị thường của thuật toán.

Ngoài ra, một cách tiếp cận khác để giảm thời gian tính toán là song song hóa quá trình ước tính mật độ hàm hạt nhân trên mạng máy tính, trên CPU hoặc GPU. Trong nghiên cứu [21], Lukasik đã đề xuất sử dụng thư viện giao thức truyền thông điệp (MPI) để song song hóa việc ước lượng hàm mật độ xác suất. Năm 2013, Michailidis và Margaritis đã song song hóa ước lượng mật độ hàm hạt nhân trên các khung lập trình khác nhau như Pthreads, OpenMP, Intel Cilk ++, Intel TBB và SWARM [22]. Cũng trong năm 2013, họ tiếp tục song song hóa ước lượng hàm mật độ hạt nhân trên nền tảng GPU CUDA [23]. Ưu điểm của các phương pháp này là không làm thay đổi hiệu suất phát hiện dị thường của các thuật toán. Tuy nhiên, độ phức tạp tính toán PDF không thay đổi, vẫn là $O(kn^2)$; thời gian tính toán giảm do các phương pháp này đã chia tổng khối lượng công việc làm nhiều phần và tính toán đồng thời.

Qua quá trình nghiên cứu chúng tôi thấy rằng, trong công thức tính mật độ xác suất, việc tìm những điểm ảnh trong phạm vi băng thông để hàm hạt nhân khác 0 tiêu tốn rất nhiều thời gian. Vì vậy, để giảm được thời gian tính toán, chúng tôi phân các điểm ảnh về các nhóm cùng giá trị. Mục đích là làm giảm số lượng dữ liệu cần tính toán, thay vì phải tính toán toàn bộ n điểm ảnh, chúng ta chỉ phải tính toán trên m nhóm các điểm ảnh, với m nhỏ hơn rất nhiều so với n. Trong tự nhiên, lớp phủ thực địa luôn có tính chất phân lớp đối tượng, lớp phủ càng đồng nhất số lượng nhóm càng ít. Bởi vậy, bước phân nhóm các điểm ảnh về cơ bản làm giảm đáng kể số lượng điểm dữ liệu cần xét đến. Bước tiếp theo, chúng tôi tổ chức dữ liệu theo cây Kd-tree đối với dữ liệu chưa phân nhóm và dữ liệu sau phân nhóm để tăng tốc độ tìm kiếm các điểm dữ liệu trong phạm vi băng thông thỏa mãn hàm hạt nhân khác 0.

Phần tiếp theo của bài báo được cấu trúc như sau. Phần II trình bày lý thuyết ước lượng mật độ phi tham số và thuật toán để thực hiện việc ước lượng này. Phần III trình bày phương pháp phân nhóm dữ liệu, xây dựng, tìm kiếm trên cây Kd-tree và thuật toán để tính toán PDF khi dữ liệu đã được nhóm và tổ chức vào cây Kd-tree. Phần IV trình bày kết quả thực nghiệm trên ba loại ảnh (ảnh đa phổ 3 kênh phổ, ảnh đa phổ 8 kênh và ảnh siêu phổ 224 kênh). Cuối cùng là kết luận và tài liệu tham khảo.

II. ƯỚC LƯỢNG MẬT ĐỘ HẠT NHÂN

Phương pháp ước lượng mật độ xác suất phi tham số trong đó công cụ chính là ước lượng mật độ hạt nhân (KDE) đã được Rosenblatt công bố vào năm 1956 [24] và sau đó được Parzen phát triển, công bố vào năm 1962 [25].

Bảng I
MỘT SỐ HÀM NHÂN ĐIỂN HÌNH [27]

Tên hàm nhân	$K(u)$	Điều kiện
Uniform	$\frac{1}{2}$	$\lvert u\rvert \le 1$
Hypercube	1	$\lvert u\rvert \le \frac{1}{2}$
Triangular	$1 - \lvert u\rvert$	$\lvert u\rvert \le 1$
Epanechnikov	$\frac{3}{4\sqrt{5}} - \frac{3u^2}{20\sqrt{5}}$	$\lvert u\rvert \le \sqrt{5}$
Quartic	$\frac{15}{16}(1-u^2)^2$	$\lvert u\rvert \le 1$
Triweight	$\frac{35}{32}(1-u^2)^3$	$\lvert u\rvert \le 1$
Tricube	$\frac{70}{81}(1-\lvert u\rvert^3)^3$	$\lvert u\rvert \le 1$
Gaussian	$\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$	
Cosine	$\frac{\pi}{4} cos\left(\frac{\pi}{2}u\right)$	$\lvert u\rvert \le 1$

Đối với dữ liệu một chiều, xét vector ngẫu nhiên $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ của biến ngẫu nhiên $\mathbf{x}$ có n phần tử. Điều này có nghĩa rằng có n quan sát của biến ngẫu nhiên $\mathbf{x}$ và x_i là quan sát thứ i của biến ngẫu nhiên $\mathbf{x}$. Khi đó, mật độ hạt nhân của biến ngẫu nhiên $\mathbf{x}$ được ước lượng như sau:

$$\hat{f}(x_i) = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{h_j} K\left(\frac{x_i - x_j}{h_j}\right), \quad i = 1, 2, \ldots, n, \quad (1)$$

trong đó $\hat{f}(\cdot)$ được gọi là hàm mật độ xác suất (PDF), $K(u)$ được gọi là hàm nhân thỏa mãn điều kiện $\int_{-\infty}^{\infty} K(u)d(u) = 1$ và h_j là hệ số tỷ lệ quyết định "khoảng rộng" của hàm nhân hay còn gọi là băng thông. Thảo luận mở rộng về các thuộc tính thống kê của $\hat{f}(\cdot)$ có thể được tìm thấy trong [26], $K(u)$ có thể là các hàm nhân điển hình do Hardle trình bày trong [27] và được thể hiện trong bảng I.

Trong trường hợp dữ liệu có k chiều, quan sát thứ i của $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)^T$ là $\mathbf{x}_i = (x_i^1, x_i^2, \ldots, x_i^k)^T$, $i = 1, \ldots, n$, và công thức ước tính mật độ hạt nhân của dữ liệu đa biến được định nghĩa trong [27] là:

$$\hat{f}(\mathbf{x}_i) = \frac{1}{n}\sum_{j=1}^{n} \left\{ \prod_{d=1}^{k} \frac{1}{h_d} K\left(\frac{x_i^d - x_j^d}{h_d}\right)\right\}, \quad i = 1, 2, \ldots, n. \quad (2)$$

Đối với ảnh đa phổ và siêu phổ, dữ liệu thuộc dạng đa biến, chúng tôi sử dụng công thức (2) để cài đặt thuật toán. Không làm mất tính tổng quát, chúng tôi cố định băng thông, đặt $h = h_1 = h_2 = \cdots = h_d$ với $d = 1, 2, \ldots, k$. Thuật toán 1 được viết giả lập theo ngôn ngữ lập trình C để ước tính mật độ của dữ liệu đa biến theo phương pháp tuần tự trên CPU, đây là thuật toán do Lukasik [21], Michailidis và Margaritis [22, 23] xây dựng.

Trong thuật toán 1, $\mathbf{X}$ là dữ liệu ảnh đa phổ hoặc siêu phổ được tổ chức thành một ma trận hai chiều từ nhiều vector, chỉ số của chiều thứ nhất tương ứng với vị trí trong không gian của các điểm ảnh, chiều thứ hai chứa dữ liệu của các kênh ảnh tại vị trí đó, n là tổng số điểm ảnh, k là số kênh phổ, h là băng thông của hàm ước lượng mật độ, *pdf* là vector lưu trữ mật độ xác suất của các điểm ảnh. Trong thuật toán 1, hàm Kernel được thiết kế riêng bởi những thuật toán phía sau đều phải sử dụng đến nó. Trong hàm Kernel, $\mathbf{x}_i$ là vector giá trị của điểm ảnh cần tính mật độ, $\mathbf{x}_j$ là vector giá trị của điểm ảnh bất kỳ nằm trong băng thông, $K(u)$ là một trong những hàm đã nêu trong bảng I. Thuật toán 1 có độ phức tạp tính toán là $O(kn^2)$.

III. TĂNG TỐC ĐỘ ƯỚC LƯỢNG HÀM MẬT ĐỘ

Như phân tích trong phần II, thuật toán 1 có độ phức tạp tính toán là $O(kn^2)$. Đây là độ phức tạp tính toán theo hàm số mũ. Trong phần thực nghiệm của nghiên cứu [20], tác giả sử dụng 100.000 điểm dữ liệu để kiểm nghiệm và thời gian tính toán là 4 ngày. Trên thực tế, thời gian chúng tôi tính toán PDF cho một ảnh màu RGB 11.373.204 điểm

Thuật toán 1: Thuật toán ước lượng mật độ hạt nhân [21–23]

input: Ma trận các điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h
output: Mật độ xác suất của các điểm ảnh *pdf*

for $i \leftarrow 0$ **to** $n-1$ **do**
  $sum\_ker \leftarrow 0$;
  for $j \leftarrow 0$ **to** $n-1$ **do**
    $sum\_ker \leftarrow sum\_ker + Kernel(x_i, x_j, k, h)$;
  end
  $pdf[i] \leftarrow \dfrac{sum\_ker}{n}$;
end
return *pdf*;

Function Kernel

Input: điểm ảnh đang xét x_i, điểm ảnh kiểm tra x_j, số kênh phổ k, băng thông h
Output: Giá trị của $K(u)$
$mul\_ker \leftarrow 1$;
for $d \leftarrow 0$ **to** $k-1$ **do**
  $mul\_ker \leftarrow mul\_ker \times \dfrac{1}{h} \times K\left(\dfrac{x_i^d - x_j^d}{h}\right)$;
end
return *mul_ker*;

ảnh là 21 ngày; thời gian tính toán PDF cho một ảnh 8 kênh phổ 710.613 điểm ảnh là hơn 3 giờ. Do đó, rất khó áp dụng phương pháp này trong những ứng dụng thực tế, nhất là trong công tác tìm kiếm cứu nạn do nó đòi hỏi cao về thời gian đưa ra quyết định.

Quan sát công thức (2) cho thấy chỉ những điểm ảnh làm cho $K(u) \neq 0$ mới có ý nghĩa, những điểm ảnh còn lại không làm tăng giá trị của $\hat{f}(\mathbf{x}_i)$. Vì vậy, chúng ta chỉ đi tìm tập hợp những điểm ảnh thỏa mãn điều kiện $K(u) \neq 0$. Qua quá trình nghiên cứu, chúng tôi nhận thấy rằng, công đoạn để tìm những điểm ảnh thỏa mãn $K(u) \neq 0$ tiêu tốn nhiều thời gian nhất. Vì vậy, phương pháp đầu tiên chúng tôi nghĩ đến là làm thế nào để giảm bớt dữ liệu tính toán mà không làm thay đổi kết quả đầu ra. Đối với ảnh đa phổ, khả năng các điểm ảnh có vector phổ giống nhau tương đối cao, nhất là ảnh màu RGB. Do đó, chúng tôi chia những điểm ảnh này thành m nhóm có cùng giá trị. Như vậy, thay vì chúng ta phải tính toán PDF cho n điểm ảnh chúng ta chỉ phải tính toán PDF cho m nhóm điểm ảnh, các điểm ảnh trong cùng một nhóm sẽ có giá trị mật độ xác suất giống nhau. Khi đó, m càng nhỏ thì thời gian tính toán càng nhanh.

Tiếp theo, chúng tôi tổ chức dữ liệu theo cấu trúc của cây Kd-tree [28]. Về bản chất cây Kd-tree là một cây nhị phân bởi mỗi nút có nhiều nhất là hai con. Nút lá chứa điểm dữ liệu k chiều, những nút không phải là nút lá tạo ra một siêu phẳng tách (lát cắt) để phân chia không gian thành hai phần, được gọi là nửa không gian. Các điểm ở bên trái của siêu phẳng này được biểu thị bằng cây con bên trái của nút đó và các điểm ở bên phải của siêu phẳng được thể hiện bằng cây con bên phải. Những điểm ảnh $\mathbf{x}_j$ thỏa mãn $K(u) \neq 0$ phải là những điểm ảnh láng giềng gần nhất của $\mathbf{x}_i$. Nói cách khác những điểm ảnh này phải nằm trong hình siêu cầu có bán kính r cho trước, tâm là $\mathbf{x}_i$. Thông thường chúng ta phải duyệt hết toàn bộ dữ liệu mới tìm ra được tập hợp đó. Mục đích tổ chức dữ liệu theo cấu trúc cây Kd-tree là để nhanh chóng tìm được tập hợp các điểm ảnh làm cho $K(u) \neq 0$. Do tính chất của cây Kd-tree, mỗi nút không phải là lá sẽ chia không gian thành hai phần nên bắt đầu xét từ nút gốc, nếu điểm $\mathbf{x}_i$ nhỏ hơn gốc một khoảng r thì rõ ràng những điểm ảnh đáp ứng điều kiện $K(u) \neq 0$ phải nằm nhánh bên trái của nút gốc, chúng ta chỉ phải đi tìm những điểm dữ liệu nằm nhánh bên trái của gốc mà không cần quan tâm đến những những nút dữ liệu nằm nhánh bên phải của nút gốc. Và ngược lại, chúng ta chỉ phải đi tìm những những điểm ảnh nằm nhánh bên phải của gốc mà không cần quan tâm đến những điểm ảnh nằm bên nhánh bên trái của nút gốc. Vì vậy, việc áp dụng cây Kd-tree đã giảm được thời gian tính toán hàm tính tổng trong công thức (2).

Những bước ở trên là quá trình tiền xử lý dữ liệu trước khi dữ liệu được dùng để ước lượng hàm mật độ xác suất. Dưới đây, chúng tôi trình bày chi tiết hai bước tiền xử lý và việc ước lượng hàm mật độ xác suất.

1. Nhóm các điểm ảnh có cùng giá trị phổ

Đối với ảnh đa phổ hoặc ảnh siêu phổ, quá trình tìm kiếm những điểm ảnh có phổ trùng nhau mất rất nhiều thời gian, với độ phức tạp tính toán là $O(kmn)$, trong đó m là số nhóm các điểm ảnh có phổ trùng nhau, n là số điểm ảnh và k là số kênh ảnh. Để giảm độ phức tạp tính toán, ý tưởng nhóm các điểm ảnh cùng giá trị là xây dựng một mảng hai chiều, được gọi là mảng **A**. Kích thước chiều thứ nhất của **A** là số lượng tổ hợp màu của các kênh ảnh. Ví dụ, với ảnh màu RGB 24 bit, chiều thứ nhất của mảng **A** sẽ có kích thước là 16.777.216. Kích thước chiều thứ hai của **A** sẽ được cấp phát linh động để lưu trữ vị trí không gian trên ảnh của các điểm ảnh thuộc về nhóm đó.

Thuật toán nhóm chạy qua lần lượt các điểm ảnh, tính giá trị tổ hợp màu của điểm ảnh đó theo công thức sau:

$$index_i = \sum_{d=0}^{k-1} Max^d \times x_i^d, \quad i = 1, 2, \ldots, n, \tag{3}$$

trong đó Max là giá trị lớn nhất của các kênh ảnh của tất cả các điểm ảnh (thông thường, đối với ảnh lưu trữ 8 bit/kênh ta có $Max = 256$, 10 bit/kênh thì $Max = 1024, \ldots$), $index_i$ tương ứng với chỉ số trên chiều thứ nhất của mảng **A**. Chiều thứ hai của mảng **A** sẽ tự động tăng thêm một ô nhớ để lưu trữ vị trí không gian của điểm ảnh đó.

Thuật toán nhóm được trình bày cụ thể tại thuật toán 2. Độ phức tạp của thuật toán là $O(kn)$. Trong thuật toán, đầu tiên phải xây dựng được một cấu trúc để lưu trữ thông tin của nhóm các điểm ảnh. Trong cấu trúc nhóm điểm ảnh phải lưu trữ được giá trị phổ của điểm ảnh và chỉ số của các điểm ảnh thuộc nhóm. Công việc tiếp theo là đi tìm giá trị Max để có cơ sở tính toán kích thước chiều thứ nhất của mảng **A** (tính toán $size = Max^k$), khởi tạo mảng **A** với chiều thứ nhất có kích thước bằng $size$, chiều thứ hai bằng $\varnothing$. Trong vòng lặp chính, tính giá trị tổ hợp màu của điểm ảnh thứ i và gán cho $index$, tại chỉ số $index$ của mảng **A**, điểm ảnh thứ i sẽ được thêm vào nhóm. Cuối cùng, loại bỏ những nhóm điểm ảnh không chứa bất kỳ điểm ảnh nào ta sẽ được các nhóm điểm ảnh.

Để ước lượng mật độ xác suất cho các điểm ảnh, công thức (2) được biến đổi thành:

$$\hat{f}_G(\mathbf{g}_i) = \frac{1}{n}\sum_{j=1}^{m}\left\{\left[\prod_{d=1}^{k}\frac{1}{h_d}K\left(\frac{g_i^d - g_j^d}{h_d}\right)\right] \times |M_j|\right\}, \quad (4)$$

trong đó $\hat{f}_G(g_i)$ là hàm ước tính mật đô xác suất của nhóm thứ i, khi đó, tất cả các điểm ảnh trong nhóm thứ i sẽ có mật độ xác suất bằng nhau, $i = 1, 2, \ldots, m$, m là tổng số nhóm, $\mathbf{g}_i$ là vector chứa giá trị các kênh phổ của nhóm thứ i, M_j là tập hợp các điểm ảnh trong nhóm thứ j, và $|M_j|$ là kích thước của tập hợp M_j.

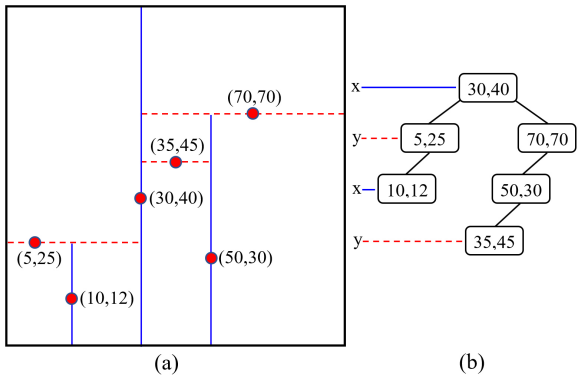
2. Xây dựng và tìm kiếm trên Kd-tree

Cây Kd-tree được phát triển và công bố bởi Bentley [28] vào năm 1975. Về bản chất, nó là một cây nhị phân (do mỗi nút của cây có tối đa 2 nhánh con), trong đó mỗi nút biểu diễn một phân vùng không gian k chiều. Nút gốc đại diện cho toàn bộ không gian, các nút lá đại diện cho không gian con chứa một tập con độc nhất của tập dữ liệu đầu vào. Điểm đặc biệt của cây Kd-tree là các đỉnh của cây là những điểm phân chia không gian thành hai phần. Việc phân chia không gian như vậy sẽ thuận tiện cho tìm kiếm những điểm trong cây gần nhất với một điểm nào đó trong vùng không gian. Điều này có nghĩa rằng, việc tìm những điểm thuộc cây gần với một điểm nào đó trong không gian dựa trên một số phép phân hoạch không gian để loại bỏ các vùng không gian không cần thiết, như vậy sẽ thu hẹp được không gian tìm kiếm.

Thuật toán 2: Thuật toán nhóm các điểm ảnh (CreateGroup)

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k
output: vector các nhóm điểm ảnh *groups*

```
Max ← phần tử có giá trị lớn nhất của ma trận x;
size ← Max^k;
Khởi tạo ma trận A, chiều thứ nhất có kích thước bằng size;
for i ← 0 to size − 1 do
    A[i]← {∅}
end
for i ← 0 to n − 1 do
    index ← 0;
    for d ← 0 to k − 1 do
        index ← index + Max^d × x_i^d;
    end
    A[index]← A[index] ∪ {i}
end
groups ← {∅};
for i ← 0 to size − 1 do
    if A[i]≠ ∅ then
        groups ← groups ∪ {A[i]}
    end
end
return groups;
```



Hình 1. a) Minh họa phân chia miền không gian, b) Minh họa cây Kd-tree đã được xây dựng từ dữ liệu đã cho.

Để hiểu rõ hơn về cây Kd-tree, chúng ta xét một ví dụ xây dựng cây Kd-tree từ bộ dữ liệu 2 chiều $(30, 40)$, $(5, 25)$, $(10, 12)$, $(70, 70)$, $(50, 30)$, $(35, 45)$, chi tiết được thể hiện trong hình 1. Trong đó, hình 1(a) thể hiện các vùng không gian đã được chia, những đường thẳng liền nét là những đường chia không gian dữ liệu theo chiều thứ nhất (chiều x), những đường thẳng nét đứt chia không gian dữ liệu theo chiều thứ hai (chiều y). Hình 1(b) thể hiện cây Kd-tree được xây dựng từ bộ dữ liệu trên.

Quy tắc xây dựng cây và phân chia không gian như sau. Chọn một điểm dữ liệu làm gốc, tại gốc chia toàn bộ không gian dữ liệu theo chiều thứ nhất làm hai phần (trong ví dụ, điểm gốc được chọn là $(30, 40)$, điểm này chia toàn bộ không gian dữ liệu theo chiều x thành 2 phần, phần bên trái là những điểm dữ liệu có chiều x nhỏ hơn hoặc bằng 30, phần bên phải là những điểm dữ liệu có chiều x lớn hơn hoặc bằng 30). Tiếp theo, xét lần lượt từng điểm dữ liệu, so sánh điểm dữ liệu này với các nút trên cây, bắt đầu từ gốc. Quy tắc so sánh như sau. Giả sử dữ liệu của chúng ta có k chiều (quy định bắt đầu từ 0 đến $k-1$), lấy bậc của nút cần so sánh chia cho k được phần dư, phần dư này chính là chiều dữ liệu được dùng để so sánh. Nếu điểm dữ liệu nhỏ hơn hoặc bằng với nút so sánh thì điểm dữ liệu sẽ nằm bên trái nút so sánh, ngược lại nằm bên phải. Tiếp tục so sánh đến khi gặp lá thì thêm điểm dữ liệu vào cây.

Thuật toán 3 xây dựng cây Kd-tree từ các điểm ảnh của ảnh đầu vào. Trong thuật toán này, đầu tiên phải xây dựng cấu trúc một nút của cây Kd-tree để lưu trữ dữ liệu và một số thuộc tính khác nữa của node, sau đó xây dựng hàm InsertNode để chèn một nút vào cây, cuối cùng xây dựng hàm CreateKDTree để xây dựng thành một cây hoàn chỉnh. Độ phức tạp tính toán xây dựng cây Kd-tree là $O(n \log n)$ [29].

Sau khi xây dựng cây Kd-tree (theo thuật toán 3), mục tiêu của chúng ta là sử dụng cây này để tìm tất cả những điểm ảnh đáp ứng điều kiện $K(u) \neq 0$. Trong bảng I, chúng ta thấy rằng ngoại trừ hàm nhân Gauss là không có điều kiện, các nhân còn lại đều có điều kiện

$$|u| = \left| \frac{x_i^d - x_j^d}{h} \right| \leq \epsilon,$$

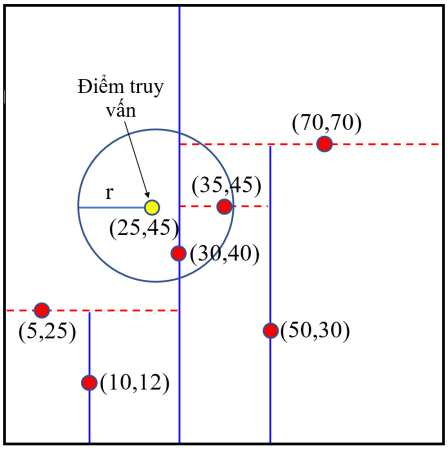
với $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$ và $d = 1, 2, \ldots, k$ thì $K(u)$ mới có giá trị, ngược lại $K(u) = 0$. Tùy thuộc vào hạt nhân cụ thể mà ϵ sẽ nhận các giá trị khác nhau. Đặt $r = h \times \epsilon$, khi đó để hạt nhân $K(u) \neq 0$ thì $|x_i^d - x_j^d| \leq r$. Điều này có ý nghĩa rằng với điểm x_i đang được xem xét, những điểm x_j nằm trong hình siêu cầu bán kính r (sử dụng thước đo khoảng cách Chebyshev [30] để đo khoảng cách từ điểm x_i tới điểm x_j) sẽ là những điểm được chọn để tính $K(u)$. Xem minh họa trong hình 2, trong đó điểm dữ liệu cần tính toán PDF là (25,45) với $r = 10$ thì những điểm dữ liệu (35,45) và $(30, 40)$ nằm trong hình tròn bán kính r sẽ thỏa mãn điệu kiện để $K(u) \neq 0$. Vì vậy, những điểm dữ liệu này được tham gia tính toán PDF cho điểm dữ liệu đang xét.

Thuật toán 4 tìm kiếm những điểm ảnh nằm trong hình siêu cầu có bán kính r, có tâm là x_i. Thuật toán sử dụng phương pháp đệ quy để tìm kiếm danh sách các điểm dữ liệu nằm trong hình siêu cầu bán kính r có tâm là điểm đang xét. Những điểm dữ liệu thỏa mãn yêu cầu được lưu trong

Thuật toán 3: Tạo cây Kd-tree (Create Kd-tree)

Function `InsertNode()`
- **input:** nút gốc *root*, điểm dữ liệu *point*, số chiều dữ liệu k, mức của cây *level*;
- **output:** điểm dữ liệu được thêm vào cây;
- Tìm chiều của không gian dữ liệu: $axis \leftarrow level \,\%k$;
- **if** $root = \emptyset$ **then**
  - - Khởi tạo một nút mới;
  - - Gán nút mới khởi tạo cho *root*;
- **else**
  - **if** *point[axis]<root->data[axis]* **then**
    - $InsertNode(root \rightarrow left, point, k, level+1)$
  - **else**
    - $InsertNode(root \rightarrow right, point, k, level+1)$
  - **end**
- **end**

Function `CreateKDTree()`
- **input:** Ma trận điểm ảnh $\mathbf{X}$, số điểm ảnh n, số kênh phổ k;
- **output:** Cây Kd-tree hoàn chỉnh;
- KDNode*$root \leftarrow \emptyset$;
- **for** $i \leftarrow 0$ **to** $n-1$ **do**
  - $InsertNode(root, \mathbf{x}_i, k, 0)$;
- **end**
- **return** *root*;



Hình 2. Minh họa những điểm được chọn để tính $K(u)$.

danh sách *list*. Đầu tiên, thuật toán kiểm tra nút gốc *root*, nếu nó rỗng thì thoát khỏi thuật toán. Tiếp đến tính toán khoảng cách từ điểm đang xét đến *root* (sử dụng phương pháp tính khoảng cách Chebyshev), tìm chiều dữ liệu để so sánh. Kiểm tra nếu *root* nằm trong hình siêu cầu đó thì thêm *root* vào *list*. So sánh điểm đang xét với *root*, nếu

Thuật toán 4: Search [27]

input: Node gốc *root*, điểm cần tính PDF *query*, số chiều dữ liệu k, bán kính siêu cầu r;
output: Danh sách các điểm được tìm thấy *list*;

if *root*= $\emptyset$ **then**
  return
end
Tính khoảng cách từ *query* đến *root*, gán vào d;
Tìm chiều dữ liệu so sánh: $axis \leftarrow root.level\%k$;
if $d \leq r$ **then**
  $list \leftarrow list \cup \{root\}$;
end
if $query[axis] < root.data[axis]$ **then**
  if $root.left \neq \emptyset$ **then**
    $Search(root.left, query, r, list)$;
  end
  if $root.right \neq \emptyset$ **then**
    if $|query[axis] - root.data[axis]| \leq r$ **then**
      $Search(root.right, query, r, list)$;
    end
  end
else
  if $root.right \neq \emptyset$ **then**
    $Search(root.right, query, r, list)$;
  end
  if $root.left \neq \emptyset$ **then**
    if $|query[axis] - root.data[axis]| \leq r$ **then**
      $Search(root.left, query, r, list)$;
    end
  end
end

nhỏ hơn *root* tìm nhánh bên trái của *root*, ngược lại tìm nhánh bên phải của *root*. Chỉ xét riêng trên một chiều dữ liệu so sánh, nếu khoảng cách từ điểm đang xét đến *root* trên chiều dữ liệu đó mà nhỏ hơn hoặc bằng r bắt buộc phải tìm cả nhánh bên trái và nhánh bên phải của *root*. Ví dụ trên hình 2, rõ ràng điểm truy vấn nằm bên phần không gian bên trái của nút gốc là $(30, 40)$, thông thường chúng ta chỉ tìm những điểm nằm trên nhánh trái của *root* sẽ bỏ qua những điểm dữ liệu thỏa mãn yêu cầu của hàm nhân $K(u) \neq 0$.

Trong nghiên cứu của Kakde [29] tác giả đã chứng minh rằng, độ phức tạp của thuật toán xây dựng cây là $O(n \log n)$, độ phức tạp thuật toán tìm kiếm một vùng không gian trên Kd-tree là $O(\sqrt{n} + c)$, trong đó c là số điểm ảnh được tìm thấy trong thuật toán 4. Khi đó, để ước lượng mật độ xác suất cho các điểm ảnh, công thức (2) trở thành:

$$\hat{f}(\mathbf{x}_i) = \frac{1}{n} \sum_{j=1}^{c_i} \left\{ \prod_{d=1}^{k} \frac{1}{h_d} K\left(\frac{x_i^d - p_{ij}^d}{h_d} \right) \right\}, \quad i = 1, 2, \ldots, n, \quad (5)$$

Thuật toán 5: Thuật toán tính PDF khi dữ liệu đầu vào là GRP

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h;
output: Mật độ xác suất của các điểm ảnh *pdf*;

$groups \leftarrow CreateGroup(\mathbf{X}, n, k)$;
$m \leftarrow |groups|$;
for $i \leftarrow 0$ **to** $m - 1$ **do**
  $sum\_ker \leftarrow 0$;
  for $j \leftarrow 0$ **to** $m - 1$ **do**
    $sum\_ker \leftarrow sum\_ker + Kernel(groups_i, groups_j, k, h) \times |groups_j|$;
  end
  for $j \leftarrow 0$ **to** $|groups_i| - 1$ **do**
    $pdf[groups[i].indexes[j]] \leftarrow \frac{sum\_ker}{n}$;
  end
end
return *pdf*;

trong đó c_i là tổng số điểm ảnh nằm trong hình siêu cầu bán kính r có tâm x_i đã tìm được trong thuật toán 4, p_{ij} là vector chứa giá trị của các kênh phổ của điểm ảnh thứ j trong danh sách các điểm ảnh được tìm thấy trong thuật toán này.

Với việc sắp xếp các nhóm điểm ảnh trên Kd-tree công thức ước lượng mật độ xác suất của các điểm ảnh được viết lại như sau:

$$\hat{f}_G(\mathbf{g}_i) = \frac{1}{n} \sum_{j=1}^{mc_i} \left\{ \left[\prod_{d=1}^{k} \frac{1}{h_d} K\left(\frac{g_i^d - p_{ij}^d}{h_d} \right) \right] \times |C_j^i| \right\}, \quad (6)$$

với $i = 1, 2, \ldots, m$, trong đó m là số nhóm các điểm ảnh có các giá trị phổ trùng nhau, mc_i là tổng số nhóm điểm ảnh nằm trong hình siêu cầu bán kính r và tâm g_i tìm được trong thuật toán 4 (gọi là danh sách thứ i), C_j^i là tập hợp các điểm ảnh trong nhóm thứ j của danh sách thứ i.

3. Tính toán PDF

Các phần III.1 và III.2 giải quyết khâu tiền xử lý dữ liệu, việc tiếp theo là tính toán PDF. Chúng ta có ba kiểu cấu trúc dữ liệu, bao gồm: cấu trúc dữ liệu mà từ dữ liệu ảnh gốc sau khi được nhóm, gọi tắt là GRP, cấu trúc dữ liệu sau khi dữ liệu ảnh gốc đã được tổ chức vào Kd-tree, gọi tắt là KDT, cấu trúc dữ liệu mà dữ liệu ảnh gốc sau khi được nhóm sẽ tiếp tục được tổ chức lại vào Kd-tree, gọi tắt là GGP-KDT.

Trong phần này, chúng tôi xây dựng các thuật toán khác nhau để giải bài toán ước lượng mật độ xác suất của các điểm ảnh khi dữ liệu đầu vào là các cấu trúc dữ liệu kể trên. Thuật toán đầu tiên (Thuật toán 5) tính toán PDF theo công thức (4) khi dữ liệu đầu vào là GRP. Độ phức tạp tính

Thuật toán 6: Thuật toán tính PDF khi dữ liệu đầu vào là KDT

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h
output: Mật độ xác suất của các điểm ảnh *pdf*

$root \leftarrow CreateKDTree(\mathbf{X}, n, k)$;
$r \leftarrow h \times \epsilon$;
for $i \leftarrow 0$ **to** $n-1$ **do**
  $sum\_ker \leftarrow 0$;
  $list \leftarrow \varnothing$;
  $Search(root, x_i, r, list)$;
  for $j \leftarrow 0$ **to** $|list|-1$ **do**
    $sum\_ker \leftarrow sum\_ker + Kernel(x_i, list_j, h)$;
  end
  $pdf[i] \leftarrow \frac{sum\_ker}{n}$;
end
return *pdf*;

toán của thuật toán 5 là $O(km^2)$, trong đó m là số nhóm các điểm ảnh. Hàm *Kernel* đã được trình bày trong phần II.

Thuật toán thứ hai (Thuật toán 6) tính toán PDF theo công thức (5) khi dữ liệu đầu vào là KDT. Cây Kd-tree đóng vai trò là cây tìm kiếm những nút thỏa mãn điều kiện để nhân $K(u) \neq 0$. Thuật toán này có độ phức tạp là $O(kn(\sqrt{n}+c_i))$, trong đó c_i là số điểm ảnh được tìm thấy trong Thuật toán 4, với $i = 1, 2, \ldots, n$.

Thuật toán thứ ba (Thuật toán 7) tính toán PDF theo công thức (6) với dữ liệu đầu vào là GRP-KDT, Kd-tree đóng vai trò là cây tìm kiếm để tìm các nhóm điểm ảnh thỏa mãn điều kiện của hạt nhân $K(u) \neq 0$. Độ phức tạp của thuật toán là $O(km(\sqrt{m}+cm_i))$, m là số nhóm của các điểm ảnh, mc_i là số nhóm điểm ảnh được tìm thấy trong thuật toán 4, với $i = 1, 2, \ldots, m$.

Thuật toán 7: Thuật toán tính PDF khi dữ liệu đầu vào là GRP-KDT

input: Ma trận điểm ảnh x, số điểm ảnh n, số kênh phổ k, băng thông h;
output: Mật độ xác suất của các điểm ảnh *pdf*;

$groups \leftarrow CreateGroup(x, n, k)$;
$m \leftarrow |groups|$;
$root \leftarrow CreateKDTree(groups, m, k)$;
$r \leftarrow h \times \epsilon$;
for $i \leftarrow 0$ **to** $m-1$ **do**
  $sum\_ker \leftarrow 0$;
  $list \leftarrow \varnothing$;
  $Search(root, group_i, r, list)$;
  for $j \leftarrow 0$ **to** $|list|-1$ **do**
    $sum\_ker \leftarrow sum\_ker + Kernel(group_i, list_j, h)) \times |group[list[j].index]|$;
  end
  for $j \leftarrow 0$ **to** $|group_i|-1$ **do**
    $pdf\,[group[i].indexes[j]] \leftarrow \frac{sum\_ker}{n}$;
  end
end
return *pdf*;

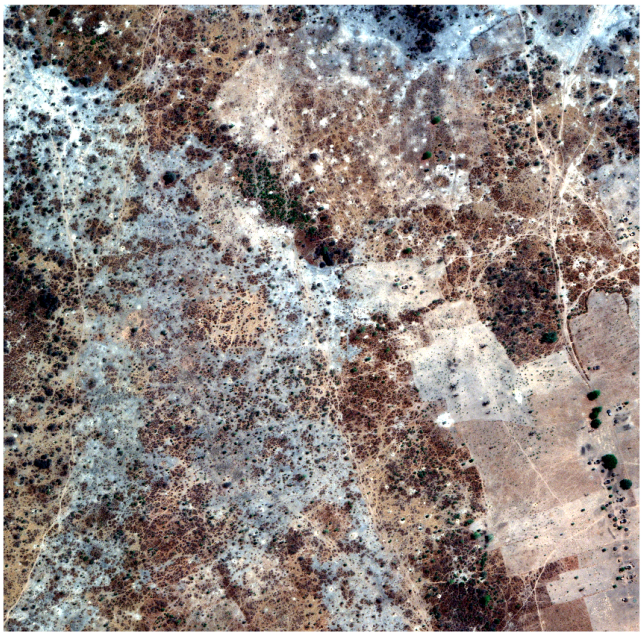
IV. THỰC NGHIỆM

1. Kịch bản thử nghiệm

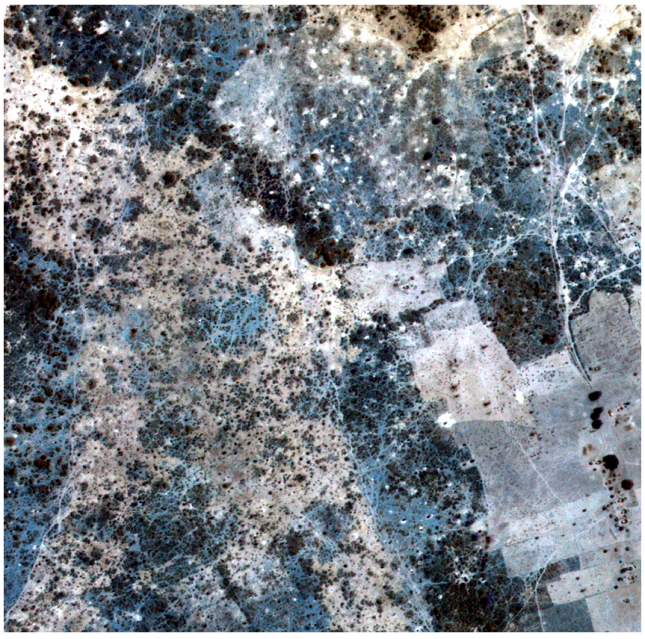
Chúng tôi thử nghiệm trên 3 loại ảnh khác nhau: ảnh màu có 3 kênh phổ (RGB), ảnh đa phổ có 8 kênh phổ, và ảnh siêu phổ có 224 kênh phổ. Tương ứng với mỗi một ảnh như vậy, chúng tôi chạy với thuật toán khi dữ liệu chưa qua giai đoạn tiền xử lý (Thuật toán 1). Sau đó, tương ứng với mỗi loại ảnh, chúng tôi chuyển dữ liệu ảnh đó qua giai đoạn tiền xử lý để có được dữ liệu theo ba cấu trúc GRP, KDT và GRP-KDT. Dữ liệu này sẽ chạy với các thuật toán 5, 6 và 7 tương ứng.

Ngoài ra, chúng tôi còn chạy thử nghiệm 3 ảnh trên với hai phương pháp: song song hóa việc ước lượng hàm mật độ trên khung lập trình Intel TBB (gọi tắt là Intel TBB), tốt nhất theo kết luận trong [22], và song song hóa tính toán PDF trên nên tảng GPU CUDA (gọi tắt là GPU CUDA), đề xuất trong công bố [23]. Việc này nhằm mục đích so sánh thời gian chạy và có cái nhìn khách quan hơn về phương pháp chúng tôi đề xuất. Lưu ý rằng phương pháp của chúng tôi đề xuất là giảm độ phức tạp tính toán dẫn đến giảm thời gian tính toán, trong khi Intel TBB và GPU CUDA không giảm độ phức tạp tính toán mà giảm thời gian tính toán bằng việc phân nhỏ tổng khối lượng công việc và tính toán đồng thời. Điểm giống nhau là các phương pháp này đều giảm thời gian tính toán mà không làm thay đổi kết quả tính toán PDF.

Trong thực tế, dữ liệu ảnh đa phổ hoặc siêu phổ thu chụp trong các tình huống tìm kiếm cứu nạn còn khan hiếm và cơ bản là không được phát hành công khai. Vì vậy, chúng tôi lựa chọn cách tiếp cận theo cách sử dụng các thư viện ảnh đã được công bố phù hợp với mục đích bài toán phát hiện dị thường trên ảnh. Đầu tiên, chúng tôi sử dụng ảnh 3 kênh phổ và ảnh 8 kênh phổ tại [32]. Đây là những ảnh được dùng trong cuộc thi "Dstl Satellite Imagery Feature Detection" do Phòng thí nghiệm khoa học và công nghệ quốc phòng (Dstl)- Vương quốc Anh cung cấp. Ảnh 3 kênh phổ có mã là 6010_1_2 (gọi tắt là ảnh 3 kênh phổ) và ảnh 8 kênh phổ có mã là 6010_1_2_M (gọi tắt là ảnh 8 kênh phổ). Hai ảnh này được thu từ bộ cảm biến WorldView 3 tại cùng một địa điểm trong phạm vi 1 km^2, kích thước (1 km $\times$ 1 km).



Hình 3. Ảnh 3 kênh phổ có mã là 6010_1_2.

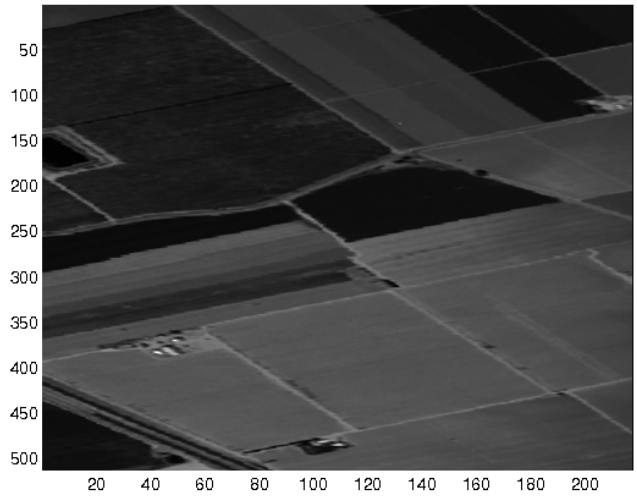


Hình 4. Ảnh hiển thị tổ hợp 3 kênh phổ (kênh 1, 2 và 3) từ ảnh 8 kênh phổ có mã là 6010_1_2_M.

Ảnh 3 kênh phổ (Hình 3) là ảnh màu RGB với độ phân giải mặt đất là $0,31$ m, kích thước ảnh là 3396×3349 điểm ảnh. Ảnh 8 kênh phổ (Hình 4) có độ phân giải mặt đất là $1,24$ m, kích thước ảnh là 849×837 điểm ảnh.

Hình 5 hiển thị những điểm ảnh dị thường (màu trắng là những công trình nhân tạo hỗn hợp trên thực địa) và những điểm ảnh không dị thường (màu đen) của hai ảnh 6010_1_2 và 6010_1_2_M.

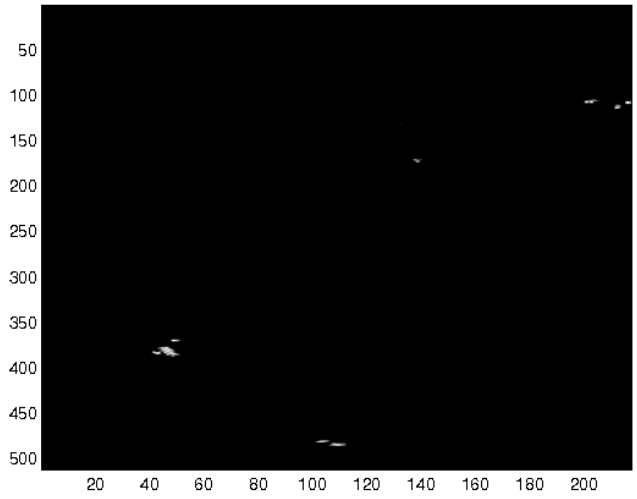
Hình 5. Những điểm ảnh dị thường (màu trắng).



Hình 6. Kênh thứ 220 của ảnh siêu phổ Salinas 224 kênh phổ.

Ảnh siêu phổ 224 kênh phổ là ảnh miễn phí được cung cấp tại [33] (Hình 6). Cảnh ảnh này được thu tại thung lũng Salinas, California bởi bộ cảm biến AVIRIS với 224 kênh phổ, độ phân giải không gian là $3,7$ m, kích thước ảnh là 512×217 điểm ảnh (gọi tắt là ảnh Salinas). Hình 7 thể hiện những điểm ảnh dị thường (màu trắng là những công trình nhân tạo được bao quanh bởi cánh đồng trồng các loại thực vật) và những điểm ảnh không dị thường (màu đen).

Bước tiếp theo, chúng tôi sử dụng hàm nhân Hypercube (Bảng I) với băng thông cố định $h = 10$ để kiểm nghiệm các thuật toán trên ba ảnh này. Cấu hình máy tính dùng để chạy các thuật toán tính PDF là như sau.



Hình 7. Những điểm ảnh dị thương (màu trắng).

Bảng II
THỜI GIAN CHẠY CỦA CÁC THUẬT TOÁN TRÊN BA ẢNH (TÍNH BẰNG GIÂY)

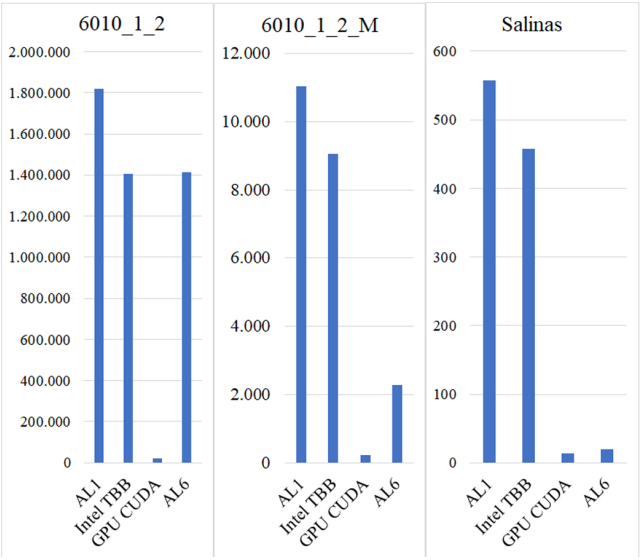
Thuật toán	6010_1_2	6010_1_2_M	Salinas
Thuật toán 1 (AL1)	1.819.712	11.038	557
Intel TBB	1.404.590	9.052	458
GPU CUDA	21.609	**253**	**13,36**
Thuật toán 6 (AL6)	1.414.826	2.269	20
Thuật toán 5 (AL5)	230	n/a	n/a
Thuật toán 7 (AL7)	**21,19**	n/a	n/a

- **CPU:** Intel Core i5-7400 3.00 GHz (4 core, 8 thread);
- **Mainboard:** MSI B150M MORTAR ARCTIC;
- **RAM:** DDR4 16 GB;
- **HDD:** SDD BIOSTAR S100 - 240 GB;
- **Graphic:** NVIDIA GeForce GTX 1070 Ti (2432 core, 1683 MHz, 8 GB RAM).

2. Đánh giá thời gian chạy của các thuật toán

Thời gian thực thi của các thuật toán trên cả ba ảnh như đã mô tả trong Phần IV.1 được thể hiện trên Bảng II. Thuật toán Intel TBB thực thi trên cả 4 nhân và 8 luồng của CPU, CPU chạy hết công suất 100%. Thuật toán GPU CUDA thực thi trên card màn hình NVIDIA GeForce GTX 1070 Ti, 2432 nhân CUDA, 8GB RAM, tốc độ xử lý 1683 MHz. Các thuật toán AL1, AL5, AL6 và AL7 chạy trên một nhân và một luồng của CPU.

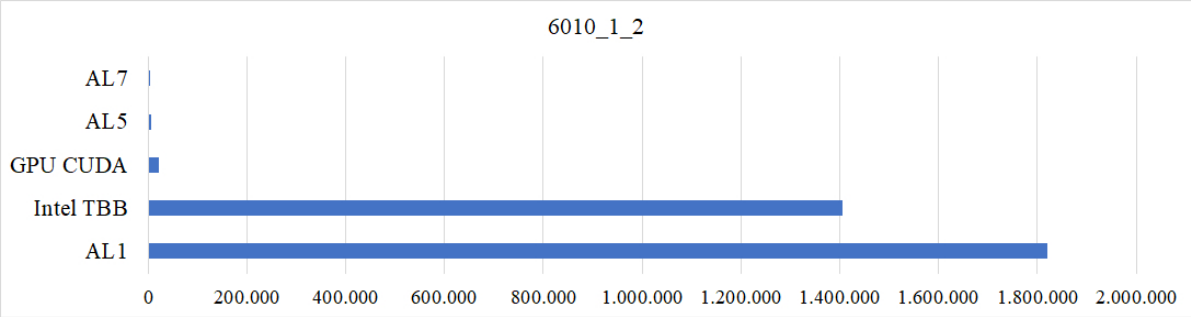
Trong trường hợp dữ liệu đầu vào của các thuật toán là KDT, kết quả được thể hiện trên hình 8. Rõ ràng rằng, khi dữ liệu được tổ chức vào cây Kd-tree, thời gian tính toán PDF đã giảm đi đáng kể so với trường hợp tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1). Cụ thể: trên ảnh 3 kênh phổ thời gian đã giảm đi 404.886s tương đương với việc giảm đi 22,25% thời gian tính toán; trên



Hình 8. Biểu đồ hiển thị thời gian chạy của thuật toán tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1), tính toán song song trên CPU (Intel TBB), tính toán song song trên GPU (GPU CUDA) và dữ liệu đã được tổ chức vào cây Kd-tree (AL6).

ảnh 8 kênh phổ, thời gian đã giảm đi 8.769s tương đương với việc giảm đi 79,44% thời gian tính toán; trên ảnh 224 kênh phổ, thời gian tính toán đã giảm đi 537s tương đương với việc giảm đi 96,41% thời gian tính toán. Trên ảnh ba kênh phổ, thuật toán AL6 có thời gian tính toán lớn hơn thuật toán Intel TB 10.236s, tương đương với thời gian tính toán của LA6 nhiều hơn Intel TBB 0,73%. Trên hai ảnh 8 kênh phổ và 224 kênh phổ thời gian tính toán của AL6 đã nhanh hơn Intel TBB lần lượt là: 6.783s và 438s, tương đương với việc giảm đi 74,93% và 95,63% thời gian tính toán. Trong trường hợp so sánh về thời gian tính toán giữa AL6 và GPU CUDA thì AL6 đều có thời gian tính chậm hơn GPU CUDA trên cả ba ảnh. Tuy nhiên có sự khác biệt, AL6 tính toán chậm hơn GPU CUDA rõ rệt trên ảnh ba kênh phổ, trên ảnh 8 kênh phổ khoảng cách đã được thu hẹp, trên ảnh 224 kênh phổ chênh lệch không nhiều (AL6 chỉ chậm hơn GPU CUDA 6,64s). Như vậy, áp dụng cây Kd-tree để quản lý dữ liệu trong giai đoạn tiền xử lý trước khi tính toán PDF sẽ hiệu quả hơn đối với những ảnh có số kênh phổ lớn.

Đối với trường hợp nhóm các điểm ảnh có phổ trùng nhau, chúng tôi chỉ áp dụng cho ảnh 3 kênh phổ. Không áp dụng cho loại ảnh 8 kênh phổ và 224 kênh phổ bởi tổ hợp màu của loại ảnh 8 hoặc 224 kênh phổ là một số rất lớn vượt khỏi khả năng quản lý của máy tính và không thể áp dụng được cho thuật toán 2. Nếu sử dụng thuật toán nhóm thông thường sẽ tốn rất nhiều thời gian do đó không khả thi. Nhìn vào bảng II và hình 9 ta thấy, thời gian tính toán PDF của ảnh 3 kênh phổ khi dữ liệu đã được



Hình 9. Biểu đồ hiển thị thời gian chạy của thuật toán tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1), tính toán song song trên CPU (Intel TBB), tính toán song song trên GPU (GPU CUDA), dữ liệu đã được nhóm (AL5) và dữ liệu đã được nhóm sau đó tổ chức trên cây Kd-tree (AL7).

nhóm đã giảm tới 1.819.482s tương đương với việc giảm đi 99,98% thời gian tính toán so với trường hợp tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (AL1). Lý do giảm thời gian tính toán PDF là quá trình nhóm các điểm ảnh có phổ trùng nhau đã làm giảm dữ liệu cần tính toán. Cụ thể, việc này đã giảm số lượng dữ liệu cần tính toán từ $3396 \times 3349 = 11.373.204$ điểm ảnh xuống còn 65.607 nhóm điểm ảnh, dẫn đến giảm được 99,4% khối lượng dữ liệu cần tính toán. Thời gian tính toán của thuật toán AL5 nhanh hơn thuật toán Intel TBB 6.107 lần, nhanh hơn thuật toán GPU CUDA 94 lần.

Dữ liệu sau khi nhóm tiếp tục đưa vào cây Kd-tree để quản lý, thời gian tính toán PDF trên ảnh 3 kênh phổ chỉ còn là 21,19s. So sánh với trường hợp tính toán PDF khi dữ liệu chưa qua giai đoạn tiền xử lý (thời gian tính toán là 1.819.712s, tương đương với hơn 21 ngày tính toán) dữ liệu GRP-KDT đưa vào tính toán PDF này đã giảm đi 99,999% thời gian tính toán. Nhìn vào bảng II và hình 9 ta thấy AL7 hoàn toàn vượt trội so với thuật toán Intel TBB và GPU CUDA, cụ thể, AL7 đã nhanh hơn Intel TBB 66.285 lần và nhanh hơn GPU CUDA 1.020 lần.

3. Đánh giá về độ phức tạp tính toán

Độ phức tạp tính toán của thuật toán AL1 là $O(kn^2)$, của thuật toán AL5 là $O(km^2)$, của thuật toán AL6 là $O(kn(\sqrt{n}+c_i))$ và của thuật toán AL7 là $O(km(\sqrt{m}+mc_i))$. Rõ ràng độ phức tạp tính toán của thuật toán AL6 và AL7 luôn luôn nhỏ hơn thuật toán AL1. Đối với thuật toán AL5, trong trường hợp xấu nhất $m = n$ (không có bất kỳ điểm ảnh nào có phổ trùng nhau) thì hai thuật toán này có độ phức tạp tính toán tương đương nhau. Tuy nhiên, điều này rất khó xảy ra ngoài thực tế bởi các ảnh chụp trong tự nhiên các lớp phủ thực địa luôn có tính chất phân lớp đối tượng, lớp phủ càng đồng nhất (chụp trên biển, trên rừng, hoang mạc, sa mạc, v.v.) thì m càng nhỏ dẫn đến độ phức tạp tính toán của AL5 nhỏ hơn AL1.

Đối với phương pháp biến đổi Fourier nhanh do Silverman để xuất [19], độ phức tạp tính toán là $O(N \log N)$, phương pháp biến đổi Gauss nhanh do Elgamall và các cộng sự đề xuất [20] có độ phức tạp tính toán là $O(N+M)$, trong đó $N = kn$ là kích thước dữ liệu và M là số lượng điểm dữ liệu cần tính PDF. Ứng dụng trong trường hợp tính toán PDF cho các điểm ảnh thì $M = N$. Rõ ràng thuật toán AL6 có độ phức tạp tính toán lớn hơn cả hai phương pháp biến đổi nhanh này do $O(kn(\sqrt{n} + c_i)) > O(N \log N) > O(N + M)$. Cả hai phương pháp biến đổi Fourier nhanh và biến đổi Gauss nhanh có độ phức tạp tính toán phụ thuộc hoàn toàn vào kích thước dữ liệu, thuật toán AL5 và AL7 có độ phức tạp tính toán phụ thuộc vào cấu trúc, nội dung của ảnh. Trong phần thực nghiệm với ảnh ba kênh phổ ở trên cho ta thấy, trong khi $n = 11.373.204$ thì m chỉ bằng 65.607, có nghĩa là $m \approx 0.006n$, dẫn đến độ phức tạp tính toán của AL5 và AL7 nhỏ hơn rất nhiều so với phương pháp biến đổi Fourier nhanh và biến đổi Gauss nhanh.

V. KẾT LUẬN

Trong công tác tìm kiếm cứu nạn, thời gian phản ứng mang ý nghĩa hết sức quan trọng. Việc rút ngắn thời gian xử lý dữ liệu và ra quyết định đồng nghĩa với việc giảm phí tổn tài chính, sức lực, tinh thần và nâng cao khả năng sống sót của nạn nhân. Trong nghiên cứu này, chúng tôi đề xuất phương pháp để làm giảm thời gian phát hiện các điểm dị thường trên ảnh siêu phổ và đa phổ (những điểm ảnh dị thường này có thể là mục tiêu cần tìm kiếm hoặc những dấu hiệu phục vụ cho công tác tìm kiếm cứu nạn). Đầu tiên là giai đoạn tiền xử lý dữ liệu với mục đích làm giảm số lượng dữ liệu cần tính toán (sử dụng nhóm các điểm ảnh có các kênh phổ trùng nhau) và tổ chức lại dữ liệu một cách hợp lý để phục vụ cho quá trình tính toán PDF (tổ chức dữ liệu vào cây Kd-tree). Sau đó, 3 cấu trúc dữ liệu trong giai đoạn tiền xử lý GRP, KDT và GRP-KDT được sử dụng để tính toán PDF.

Độ phức tạp về tính toán trong giai đoạn tiền xử lý dữ liệu khi nhóm những điểm ảnh có các kênh phổ trùng nhau là $O(kn)$, xây dựng Kd-tree là $O(n \log n)$. Trong giai đoạn tính toán PDF, độ phức tạp tính toán của các hàm PDF khi dữ liệu đã được nhóm là $O(km^2)$, trong đó m là số nhóm các điểm ảnh có các kênh phổ trùng nhau. Trong rất nhiều trường hợp, m nhỏ hơn rất nhiều so với n nên thời gian tính toán PDF cũng giảm đi tương ứng. Độ phức tạp tính toán khi dữ liệu đầu vào đã được quản lý bởi Kd-tree là $O(kn(\sqrt{n} + c_i))$, do c_i nhỏ hơn n rất nhiều nên thời gian tính các hàm PDF cũng giảm đi tương ứng. Độ phức tạp tính toán các hàm PDF khi dữ liệu đã được nhóm và tổ chức trên cây Kd-tree là $O(km(\sqrt{m} + mc_i))$.

Kết quả kiểm nghiệm trên ba loại ảnh (ảnh đa phổ 3 kênh, ảnh đa phổ 8 kênh và ảnh siêu phổ 224 kênh) đã vượt ngoài mong đợi của nhóm tác giả. Đặc biệt là đối với ảnh màu, đây là những ảnh thường được thu chụp từ thiết bị bay không người lái hoặc có người lái và được ứng dụng rộng rãi trong công tác tìm kiếm cứu nạn.

LỜI CẢM ƠN

Nghiên cứu này được tài trợ kinh phí bởi đề tài nghiên cứu khoa học cấp quốc gia mã số VT-UD.04/16-20 thuộc Chương trình KHCN vũ trụ của Bộ khoa học và công nghệ Việt Nam. Nhóm tác giả trân trọng cảm ơn sự ủng hộ và đồng hành của Ban chủ nhiệm Chương trình KHCN vũ trụ.

TÀI LIỆU THAM KHẢO

[1] T. Bolukbasi, P. Tran, "Outline Color Identification For Search And Rescue," Technical Report of Department of Electrical and Computer Engineering, Boston University, no. ECE-2012-07, 2012.

[2] M. B. Salem, K. S. Ettabaa, M. A. Hamdi, "Anomaly detection in hyperspectral imagery: An overview," in *International Image Processing, Applications and Systems Conference*, pp. 1–6, 2015.

[3] I. S. Reed and X. Yu, "Adaptive Multiple-Band CFAR Detection of an Optical Pattern with Unknown Spectral Distribution," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 10, pp. 1760–1770, 1990.

[4] T. E. Smetek, K. W. Bauer, "Finding hyperspectral anomalies using multivariate outlier detection," *IEEE Aerospace Conference*, pp. 1–24, 2007.

[5] D. Manolakis, D. Marden, G. A. Shaw, "Hyperspectral image processing for automatic target detection applications," *Lincoln Laboratory Jour.*, vol. 14, no. 1, pp. 79–116, 2003.

[6] D. C. Borghys, V. Achard, S. R. Rotman, N. Gorelik, C. Perneel, E. Scwheicher, "Hyperspectral anomaly detection: a comparative evaluation of methods," *XXXth URSI General Assembly and Scientific Symp.*, pp. 1–4, 2011.

[7] T. Marshall, L. N. Perkins, "Color Outline Detection For Search And Rescue," Technical Report of Department of Electrical and Computer Engineering, Boston University, no. ECE-2015-01, 2015.

[8] M. Ramachandran, W. Moik, *Outline Color Identification For Search And Rescue*, Technical Report of Department of Electrical and Computer Engineering, Boston University, No. ECE-2013-03, 2013.

[9] D. K. Hoai, N. V. Phuong, "Anomaly Color Detection on UAV Images for Search and Rescue works," in *2017 9th International Conference on Knowledge and Systems Engineering*, pp. 287–291, 2017.

[10] S. Khazai, S. Homayouni, A. Safari, and B. Mojaradi, "Anomaly Detection in Hyperspectral Images Based on an Adaptive Support Vector Method," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 4, pp. 646–650, 2011.

[11] A. Banerjee, P. Burlina, and C. Diehl, "A support vector method for anomaly detection in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 8, pp. 2282–2291, Aug. 2006.

[12] D. W. J. Stein, S. G. Beaven, L. E. Ho, E. M. Winter, A. P. Schaum, and A. D. Stocker, "Anomaly detection from hyperspectral imagery," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 58–69, 2002.

[13] S. Matteoli, T. Veracini, M. Diani, and G. Corsini, "Models and Methods for Automated Background Density Estimation in Hyperspectral Anomaly Detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 5, pp. 2837–2852, 2013.

[14] P. Gurram and H. Kwon, "Support-Vector-Based Hyperspectral Anomaly Detection Using Optimized Kernel Parameters," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, pp. 1060–1064, 2011.

[15] C.-I. Chang and S.-S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sensing*, vol. 40, no. 6, pp. 1314–1325, 2002.

[16] T. Veracini, S. Matteoli, M. Diani, and G. Corsini, "Nonparametric Framework for Detecting Spectral Anomalies in Hyperspectral Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 4, pp. 666–670, 2011.

[17] S. Matteoli, T. Veracini, M. Diani and G. Corsini, "Background Density Nonparametric Estimation With Data-Adaptive Bandwidths for the Detection of Anomalies in Multi-Hyperspectral Imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, pp. 163–167, 2014.

[18] C. Zhao, X. Wang, and G. Zhao, "Detection of hyperspectral anomalies using density estimation and collaborative representation," *Remote Sensing Letters*, vol. 8, no. 11, pp. 1025–1033, 2017.

[19] B. Silverman, "Algorithm AS 176: Kernel density estimation using the fast Fourier transform," *Applied Statistics*, vol. 31, no. 1, pp. 93–99, 1982.

[20] A. Elgammal, R. Duraiswami and L.S. Davis, "Efficient Kernel density estimation using the Fast Gauss Transform with applications to color modeling and tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1499–1504, 2003.

[21] S. Lukasik, "Parallel computing of kernel density estimates with MPI," in *7th International Conference on Computational Science*, pp. 726–733, 2007.

[22] P. D. Michailidis, and K. G. Margaritis, "Parallel Computing of Kernel Density Estimation with Different Multi-core Programming Models," in *21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pp. 77–85, 2013.

[23] P. D. Michailidis, K. G. Margaritis, "Accelerating Kernel Density Estimation on the GPU Using the CUDA Framework," *Applied Mathematical Sciences*, vol. 7, no. 30, pp. 1447–1476, 2013.

[24] M. Rosenblatt, "Remarks on Some Nonparametric Estimates of a Density Function," *Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 832–837, 1956.

[25] E. Parzen, "On Estimation of a Probability Density Function and Mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

[26] L. Devroye and L. Gyorfi, *Nonparametric Density Estimation: The L1 View*, Wiley, New York, 1985.
[27] W. Hardle, A. Werwatz, M. Muller and S. Sperlich, *Nonparametric Density Estimation, In: Nonparametric and Semiparametric Models*, Springer Series in Statistics, pp. 39-83, 2004.
[28] J. L. Bentley, "Multidimensional Binary Search Trees Used for Associative Searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509-517, 1975.
[29] H. M. Kakde, "Range Searching using Kd Tree," 2005. References, Aug. 12, 2019. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.122.5818.
[30] P. Trebuňa, J. Halčinová, "Experimental Modelling of the Cluster Analysis Processes," *Procedia Engineering 48*, pp. 673–678, 2012.
[31] M. Harris, "Optimizing parallel reduction in CUDA," *Nvidia developer technology 2*, no. 4, p. 70, 2007. [Online]. Available: https://developer.download.nvidia.com/assets/cuda/files/reduction.pdf.
[32] Dstl Satellite Imagery Feature Detection. [Online]. Available: https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection. [Accessed: Oct 25, 2019].
[33] Hyperspectral Remote Sensing Scenes. [Online]. Available: http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes. [Accessed: Oct 25, 2019].

Nguyễn Văn Phương tốt nghiệp Đại học và Thạc sĩ tại Học viện Kỹ thuật Quân sự năm 2003 và 2009. Hiện tại là nghiên cứu sinh tại Khoa Công nghệ Thông tin cũng tại Học viện Kỹ thuật Quân sự. Lĩnh vực nghiên cứu bao gồm: GIS, xử lý ảnh viễn thám quang học.

Đào Khánh Hoài nhận học vị Tiến sĩ năm 2005. Hiện công tác tại Học viện Kỹ thuật Quân sự. Lĩnh vực nghiên cứu bao gồm: GIS, xử lý ảnh vệ tinh, UAV, đo ảnh và thị giác máy tính.

Tống Minh Đức tốt nghiệp Đại học tại Học viện Kỹ thuật Quân sự năm 2000, nhận học vị Tiến sĩ tại Trường Đại học Tổng hợp Kỹ thuật Điện (LETI), Nga năm 2007. Hiện là giảng viên tại Khoa Công nghệ Thông tin của Học viện Kỹ thuật Quân sự. Lĩnh vực nghiên cứu bao gồm: xử lý ảnh, nhận dạng đối tượng, an toàn bảo mật thông tin.