**ỨNG DỤNG PHƯƠNG TRÌNH POISSON TRONG VIỆC TẠO CÁC HIỆU ỨNG ẢNH**

**Họ và tên SV1, Họ và Tên giảng viên2**

1 Trường Đại học Sư phạm Hà Nội, [email@gmail.com](mailto:email@gmail.com)

2 Trường Đại học Sư phạm Hà Nội, [email@hnue.edu.vn](mailto:email@hnue.edu.vn)

**TÓM TẮT:** Việc ghép nối các ảnh với nhau có thể được thực hiện dễ dàng bởi các công cụ chỉnh sửa ảnh, tuy nhiên, rất dễ để người xem phát hiện ra vùng ảnh ghép khi các vùng biên có sự chênh lệch về cường độ độ sáng và mức độ gradient. Bài báo trình bày một phương pháp nhằm hỗ trợ việc ghép các vùng ảnh được liền mạch với nhau sao cho người xem khó phát hiện. Ý tưởng chính của kỹ thuật này dựa trên việc giải phương trình Poisson. Bài báo sẽ trình bày chi tiết nội dung lý thuyết và cách rời rạc hóa phương trình để ứng dụng vào lĩnh vực xử lý ảnh, cụ thể là bài toán ghép ảnh. Sau khi rời rạc hóa, phương trình Poisson được chuyển thành một hệ phương trình tuyến tính và được giải theo phương pháp Jacobi. Một loạt các kết quả thực nghiệm đã được tiến hành với một chuỗi các ảnh tự nhiên khác nhau. Các kết quả thu được cho thấy sự hiệu quả vượt trội so với việc ghép ảnh thủ công thông thường. Đây được xem là một hướng đi đầy tiềm năng trong lĩnh vực xử lý ảnh trong tương lai.

**Từ khóa—** poisson, image editing, gradient,…

**1. GIỚI THIỆU**

Đối với những khán giả đã được trải nghiệm bộ phim “Rampage” – một trong những bộ phim được đánh giá cao về chất lượng hình ảnh, đã bao giờ bạn tự hỏi vì sao các diễn viên trong phim có thể diễn cùng với những con khỉ đột to lớn mà không bị tấn công chưa? Lý do rất đơn giản, nhà sản xuất phim sử dụng các kỹ xảo điện ảnh để ghép những con khỉ ảo vào các bức ảnh trong phim. Nhưng vấn đề ở đây là tại sao những con khỉ ảo đó lại có thể thật và sống động đến vậy? Câu hỏi này đã thúc đẩy chúng tôi đưa ra quyết định tìm hiểu sâu và kĩ hơn về kỹ thuật ghép ảnh trong công nghệ “Chỉnh sửa ảnh”.



**Hình 1‑1 Ảnh bên tay trái: người đóng mô phỏng hành vi con khỉ đột, ảnh bên tay phải: con khỉ đột được ghép vào cảnh một cách cẩn thận và khéo léo**

Công nghệ “Chỉnh sửa ảnh” là một trong những công nghệ đang được chú ý nhiều nhất hiện nay. Nó không chỉ giúp chúng ta khắc phục những bức ảnh có độ sáng kém, góc chụp bị che khuất,… mà còn có thể khôi phục lại những tấm hình xưa cũ, thêm và xóa bớt chi tiết trong hình, hoặc đơn giản là phục vụ nhu cầu “sống ảo” của giới trẻ hiện đại.

Các tác vụ chỉnh sửa hình ảnh hoặc liên quan đến các thay đổi toàn cục (chỉnh sửa màu, cường độ, các bộ lọc, …) hoặc thay đổi cục bộ giới hạn trong một lựa chọn. Trong bài báo này, chúng tôi quan tâm đến việc đạt được những thay đổi cục bộ, những thay đổi được giới hạn trong một vùng được chọn thủ công, theo cách liền mạch. Mức độ thay đổi dao động từ biến dạng nhẹ đến hoàn toàn bị thay thế bằng nội dung mới. Với công cụ chỉnh ảnh cổ điển, để đạt được điều đó, nó cần có các bộ lọc hình ảnh giới hạn trong một lựa chọn (dành cho những thay đổi nhỏ) và công cụ cắt-và-dán sẽ tương tác với các công cụ nhân bản để thay thế bức hình hoàn toàn. Tuy nhiên, khi cắt-dán một chi tiết nào đó từ hình ảnh nguồn đến hình ảnh đích, chúng ta dễ dàng nhận thấy những đường biên xung quanh đối tượng được ghép và hình ảnh chứa nó, điều này làm cho bức ảnh trở nên thiếu tự nhiên, dẫn đến sự nghi ngờ của người quan sát ảnh.

|  |  |
| --- | --- |
| (a) | (b) |

**Hình 1-2 Hai hình ảnh trước khi đưa vào chỉnh sửa. Ảnh (a): ảnh nguồn, ảnh (b): ảnh đích**



**Hình 1-3 Hình ảnh được ghép bằng công cụ cổ điển “Paint”. Ta có thể thấy rõ những đường biên xung quanh người nhảy dù được ghép vào**

Ở đây, một thuật toán chung được tập trung nghiên cứu mà từ đó các công cụ khác nhau có thể chỉnh sửa liền mạch và nhân bản một vùng được lựa chọn của ảnh nguồn để ghép vào ảnh đích. Phương trình Poisson được sử dụng để vi phân từng phần với các điều kiện biên Dirichlet [2][4]. Nó chỉ định Laplacian [3] của một hàm không xác định trên miền quan tâm (vùng ảnh mà ta lựa chọn để cắt-dán, chỉnh sửa), cùng với các giá trị hàm không xác định trên ranh giới của miền. Phương trình này sẽ được giải thích cụ thể trong Phần 2.

Sau khi cho chạy thử nghiệm chương trình, kết quả cho thấy người dùng có thể loại bỏ và thêm các đối tượng một cách liền mạch và khó bị nhận ra hơn, bằng cách pha trộn gradient [1] (hay còn gọi là độ chênh lệch giữa điểm ảnh đang xét và các điểm ảnh hàng xóm của nó) của hình ảnh nguồn và hình ảnh đích. Hơn nữa, các đối tượng có đường viền phức tạp có thể được thêm tự động mà không cần phải tốn công cắt tỉ mỉ, điều này cũng sẽ được nói rõ hơn ở Phần 2. Phần 3 sẽ trình bày các kết quả thực nghiệm, đánh giá hiệu quả qua các ví dụ thực tế và so sánh với một phương pháp ghép ảnh tự động khác có tên là mô hình Kim tự tháp. Cuối cùng, bài báo kết thúc bằng phần kết luận và các hướng nghiên cứu trong tương lai.

**2. Giải Pháp poisson cho nội suy ĐƯỢC ĐỊNH HƯỚNG**

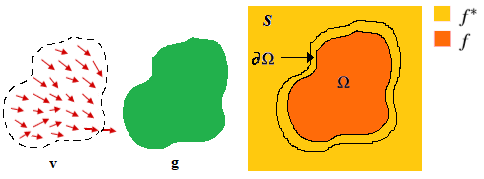
**2.1 Nội suy được định hướng**

Trong phần này, trường vector hướng dẫn được sử dụng để giải quyết nội suy cho từng thành phần màu riêng biệt và chỉ quan tâm đến các chức năng hình ảnh vô hướng [4]. Hình 2-1 minh họa các ký hiệu: cho *S*, một tập hợp con đóng của *R2*, là miền xác định hình ảnh và cho Ω là một tập hợp con khép kín của *S* với biên ∂Ω. Gọi *f\** là một hàm vô hướng đã biết được xác định trên *S* trừ phần bên trong của Ω và cho *f* là một hàm vô hướng chưa xác định ở bên trong Ω. Cuối cùng, *v* là một trường vector được xác định trên Ω.

Nội suy *f* đơn giản nhất của *f\** trên Ω là nội suy màng, được xác định là giải pháp của tối thiểu hóa vấn đề [4][5]:

, (1)

Trong đó, . = là toán tử gradient [3]. Với điều kiện là *f* và  *f\** có điểm trùng nhau và là điểm nằm trong biên ∂Ω.

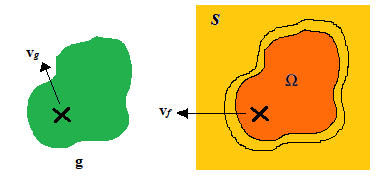


**Hình 2‑1 Chú thích nội suy hướng dẫn. Hàm không xác định *f* nội suy trong miền** ***Ω*, hàm đích *f\**, dưới sự định hướng của trường vector *v*, cái mà có thể là hoặc không phải là trường gradient của hàm nguồn *g***

Trường hợp mở rộng hơn của (1) là sử dụng trường vector định hướng v:

, (2)

Trong đó, = , như ta đã biết, vector định hướng hay còn gọi là gradient là độ chênh lệch giữa điểm ảnh đang xét và hàng xóm của nó [1], vì vậy, mục đích của phương trình (2) chính là tìm ra độ chênh lệch nhỏ nhất giữa điểm ảnh *vf* đang xét và điểm ảnh *vg* sao cho khi chèn Ω vào *S* thì mắt thường không thể nhận biết độ chênh lệch về màu sắc của Ω trong *S.*



**Hình 2-2** **Vùng được chọn để ghép vào *g* có vector *vg*và vùng được ghép vào trong ảnh đích có vector *vf***

Phương trình (2) phải thỏa mãn với điều kiện biên Dirichlet:

, (3)

Trong đó *divv* = là sự phân kì của *v = (u,v).* Đây là phương trình cơ bản cho việc chỉnh sửa hình ảnh màu: ba phương trình Poisson của biểu mẫu (3) được giải quyết độc lập trên ba kênh màu của không gian đã chọn. Tất cả các kết quả được báo cáo trong bài báo này đều là trong không gian màu RGB và sẽ được giới thiệu rõ hơn ở Phần 3.

**2.2 Rời rạc hóa và giải phương trình Poisson**

Vấn đề biến đổi (2), và phương trình Poisson liên quan với các điều kiện biên Dirichlet (3), có thể được phân loại và giải quyết theo một số cách.

Đối với hình ảnh rời rạc, vấn đề có thể được chia ra bằng cách sử dụng lưới pixel riêng biệt cơ sở. Để không mất tính tổng quan, những ký hiệu ở phần 2.1 sẽ được sử dụng lại cho các phương trình dưới đây: *S,* Ω bây giờ trở thành các tập hợp hữu hạn được xác định trên một mạng lưới rời rạc vô hạn. Lưu ý rằng *S* có thể bao gồm tất cả các pixel của một hình ảnh hoặc chỉ một tập con của chúng. Đối với mỗi pixel *p* trong *S,* hãy để *Np* là tập hợp các bộ “4 - hàng xóm” của nó trong *S*, và để cho <p,q> biểu thị một cặp pixel sao cho *q Np*. Biên của Ω bây giờ là . Gọi *fp* là giá trị của *f* tại *p*. Nhiệm vụ là tính toán tập các cường độ *f|*Ω *= {fp, q* Ω*}*.

Đối với các điều kiện biên Dirichlet được xác định trên một đường biên của hình dạng tùy ý, tốt nhất là chúng ta nên cụ thể hóa vấn đề biến đổi (2) trực tiếp, thay vì sử dụng phương trình Poisson (3). Sự khác biệt rời rạc hữu hạn của (2) mang lại sự rời rạc và bài toán tối ưu bậc hai sau:

, (4)

Trong đó *vpq* là phép chiếu trên cạnh định hướng [p,q], tức là, *vpq* = . Giải pháp của nó thỏa mãn các phương trình tuyến tính đồng thời sau đây:

, (5)

Từ phương trình (5), ta có hai trường hợp xảy ra:

|  |  |
| --- | --- |
| (a) | (b) |

**Hình 2-3** **Hình (a): Trường hợp điểm ảnh ta muốn thay đổi giá trị là điểm biên thuộc , hình (b): Trường hợp điểm ảnh mà ta muốn thay đổi giá trị nằm trong** Ω

* Trường hợp 1: chính là phương trình (5). Là khi điểm ảnh mà ta muốn thay đổi giá trị là điểm biên thuộc .
* Trường hợp 2:

Với trường hợp này, điểm ảnh mà ta muốn thay đổi giá trị nằm ở bên trong Ω nhưng không thuộc đường biên. Chính vì vậy, các hàng xóm của nó cũng thuộc Ω. Ta có công thức sau:

, (6)

Phương trình (5) tạo thành hệ thống cổ điển, thưa thớt (dải), đối xứng, xác định dương. Bởi vì có hình dạng tùy ý, các giải pháp lặp nổi tiếng đã được sử dụng. Các kết quả được trình bày trong bài báo này đã được tính toán bằng cách sử dụng lặp Jacobi [5] với sự thừa kế liên tiếp hoặc đa chu trình V - chu kỳ. Cả hai phương pháp đều đủ nhanh để chỉnh sửa tương tác các vùng hình ảnh có kích thước trung bình mà sẽ được nêu ở phần dưới đây.

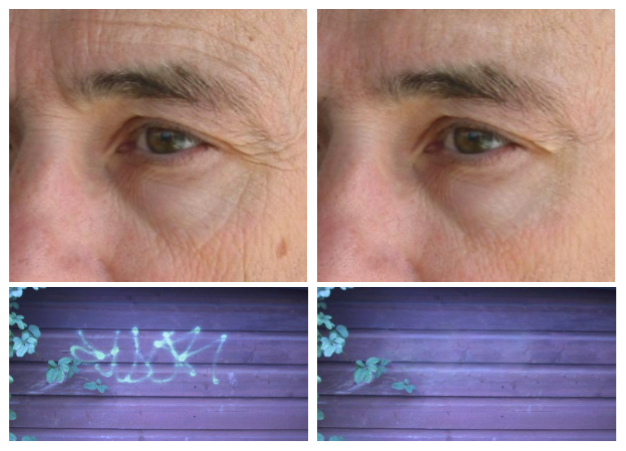
**2.3 Ứng dụng phương trình Poisson và trộn Gradient để ghép ảnh**

Lựa chọn cơ sở cho trường hướng dẫn *v* là trường gradient được lấy trực tiếp từ một hình ảnh nguồn. Biểu thị bằng *g* – hình ảnh nguồn, nội suy được thực hiện theo sự hướng dẫn của:

*v = ,* (7)

và bây giờ (4) sẽ đọc phương trình sau:

, (8)



**Hình 2-4** **Che giấu**

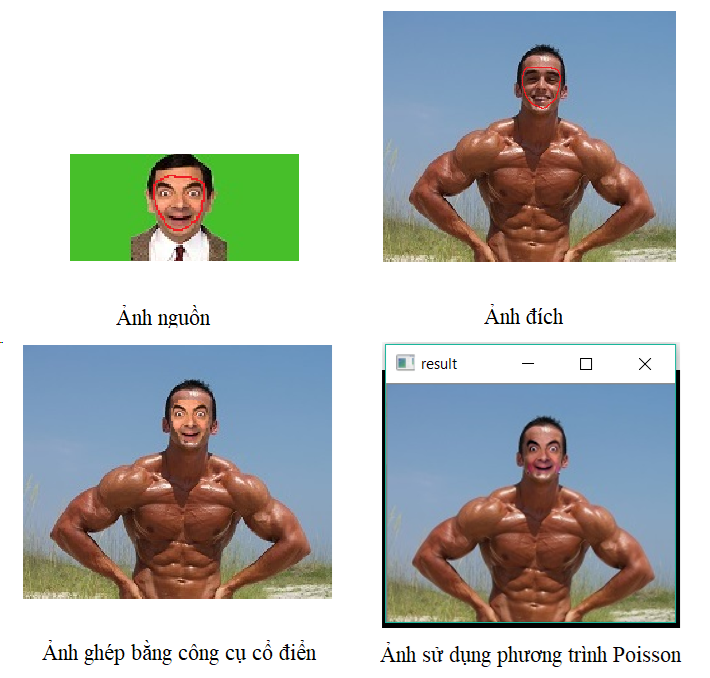
Bằng cách nhập liền mạch một phần của hình ảnh nền có sẵn, đối tượng hoàn chỉnh, các phần của các đối tượng và các tạo phẩm không mong muốn có thể dễ dàng bị ẩn. Trong cả hai ví dụ, nhiều nét (không được hiển thị) đã được sử dụng. [4]

Đối với việc thực hiện bằng số, (7) được chuyển thành:

, (9)

được gán vào phương trình (7).

Các công cụ nhân bản liền mạch có thể được sử dụng để che giấu các chi tiết hình ảnh không mong muốn hoặc chèn các phần tử mới vào một hình ảnh, nhưng linh hoạt và dễ dàng hơn nhiều so với nhân bản thông thường [1][3][5], như được minh họa trong các Hình 2-3 và 2-4. Từ góc nhìn của người dùng, hầu hết các tác vụ sẽ là các lựa chọn yêu cầu khoanh vùng tùy ý, như được biểu diễn trong Hình 2-4. [1][2]



**Hình 2-5** **Sự chèn. Sức mạnh của phương pháp được thể hiện đầy đủ khi chèn đối tượng với các hình dạng phức tạp vào một hình ảnh mới. Do sự khác biệt lớn giữa ảnh nguồn và ảnh đích, việc sao chép hình ảnh chuẩn không thể được sử dụng trong trường hợp này**

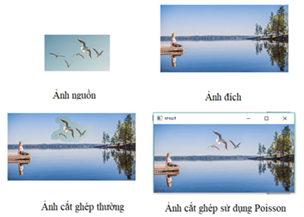
**3. KẾT QUẢ THỰC NGHIỆM**

Trong phần này, chúng tôi đánh giá hiệu quả ứng dụng của thuật toán Poisson để thực hiện ghép ảnh tự động thông qua 2 phần là kết quả cài đặt chương trình và đánh giá hiệu suất thuật toán.

**3.1 Kết quả ghép ảnh**

Chúng tôi sử dụng ngôn ngữ C++, bộ thư viện OpenCV 2.4.13 [6], Visual Studio 2017 để cài đặt bài toán trên tập hợp các ảnh tự nhiên với nội dung khác nhau. Đầu tiên, hình ảnh nguồn, đích và hình ảnh có khoanh vùng được chọn được đưa vào. Tiếp đó, giải thuật Poisson sẽ thao tác ghép vùng được chọn từ ảnh gốc để dán vào ảnh đích và thực hiện tính toán các hệ phương trình dựa vào số điểm ảnh của ảnh. Cuối cùng, kết quả trả về sẽ là một bức ảnh được ghép tự động hoàn chỉnh. Chúng tôi đã áp dụng cho 3 cặp ảnh, định dạng JPG và PNG, ảnh tự nhiên được lấy từ Google Hình Ảnh với các từ khóa: Muscle man, Mr. Bean, skydive, birds, lake criminal, .v.v... Sau đây là một số ví dụ minh họa:

****

****

**Hình 3-1 Ảnh cắt ghép minh họa cho phương trình**

* 1. **Đánh giá hiệu suất**

Để đánh giá hiệu suất ứng dụng giải thuật Poisson cho ghép ảnh tự động, chúng tôi dựa trên các điểm đặc trưng của từng bức ảnh. Bảng 3-1 đánh giá hiệu suất của thuật toán cho từng ảnh khác nhau.

**Bảng 3-1: Bảng đánh giá hiệu suất ứng dụng giải thuật Poisson**

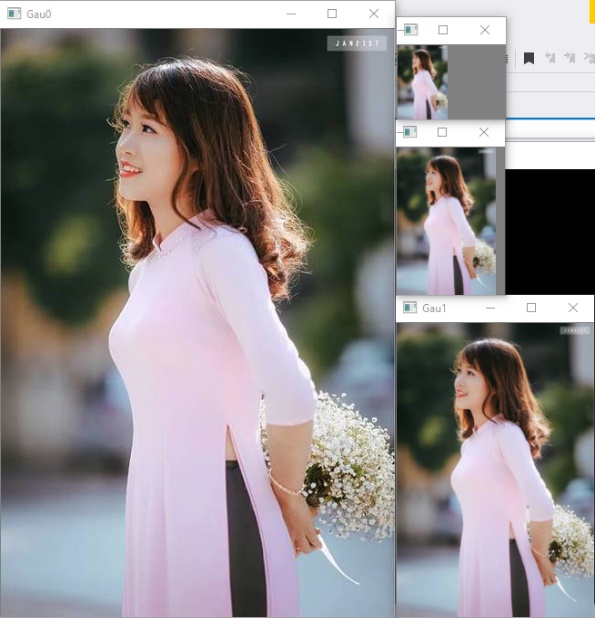
|  |  |  |  |
| --- | --- | --- | --- |
| Kích thước ảnh | Tên ảnh | Loại ảnh | Thời gian |
| 100x131 | 1\_lisa và 1\_face | png | 0.576s |
| 200x263 | 5\_face và 8\_body | jpg | 498.065s |
| 300x394 | 6\_skydive và 6\_umbrella | jpg | 1139.6s |

Từ bảng này ta thấy thời gian xử lý tăng rõ rệt, có sự chênh lệch lớn khi thực hiện trên các kích thước ảnh khác nhau. Đối với hình ảnh có kích thước nhỏ, kết quả trả về rất nhanh, nhưng đối với những hình ảnh có kích thước lớn (từ 200x300 pixel trở lên) thuật toán này sẽ cho ra kết quả trong thời gian chờ đợi rất lâu, có thể gây chán nản cho người dùng. Mặc dù có chút bất lợi về thời gian, chất lượng hình ảnh trả về khi dùng thuật toán vẫn được giữ nguyên.

Để có được kết quả trên, chúng tôi đã sử dụng phương pháp 4 hàng xóm và lặp Jacobi để thực hiện giải hệ phương trình và tạo ra ảnh kết quả.

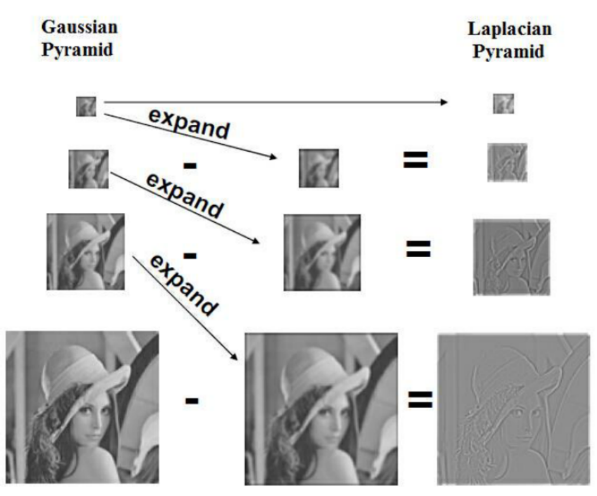
* 1. **So sánh với phương pháp mô hình Kim tự tháp**

Đối với phương pháp Kim tự tháp, đầu tiên chúng ta phải thu nhỏ hình ảnh để lọc ra những chi tiết không quan trọng. Mỗi lần thu nhỏ, kích thước hình ảnh mới bằng một nửa hình ảnh cũ, quá trình này tạo ra một hình dạng kim tự tháp được gọi là Gaussian.

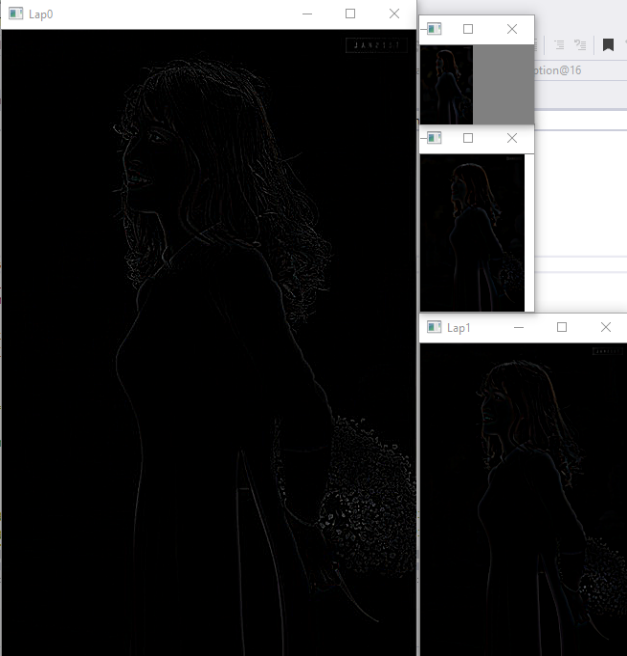


**Hình 3-2 Hình ảnh kim tự tháp Gaussian**

Và tương tự với phương trình Poisson, phương pháp Kim tự tháp này cũng cần phải trải qua giai đoạn lọc để làm nổi các đường biên bằng lọc Laplacian. Trước tiên, để lấy được các đường biên, chúng ta cần phải phóng to những bức ảnh trong trong Kim tự tháp Gaussian theo chiều ngược lại. Sau đó lấy các bức ảnh trong Kim tự tháp gốc trừ cho ảnh tương ứng trong Kim tự tháp mới phóng to ra, chúng ta được kim tự tháp Laplacian.

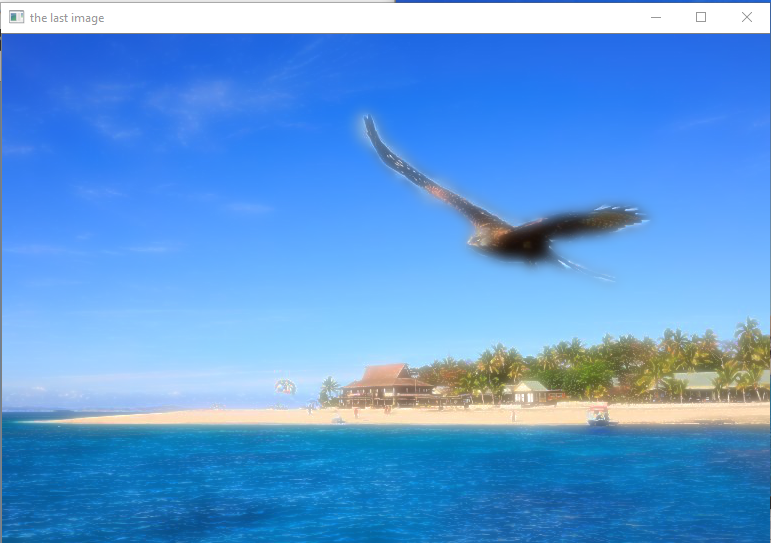


**Hình 3-3 Hình ảnh với 4 cấp độ của tháp Gaussian và Lapliacian**

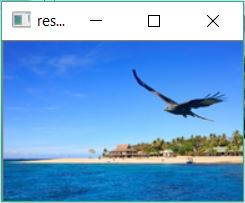


**Hình 3-4 Kết quả kim tự tháp Laplacian của Hình 3-2**

Cuối cùng, áp dụng các bước trên vào hai hình ảnh, những tấm ảnh ghép được ra đời. Tuy nhiên, hãy cùng nhìn qua kết quả trả về của bức ảnh được thực hiện bởi phương trình Poisson và phương pháp mô hình Kim tự tháp:



**Hình 3-5 Ảnh ghép sử dụng phương pháp mô hình Kim tự tháp**



**Hình 3-6 Ảnh ghép sử dụng phương trình Poisson**

Rõ ràng, chúng ta có thể thấy độ chân thực chênh lệch giữa hai bức ảnh trả về của hai phương pháp là rất lớn. Mặc dù khi nhìn tổng quan cả hai bức ảnh, chúng ta đều có thể nghĩ rằng đây không phải là ảnh ghép. Nhưng đối với bức ảnh sử dụng Kim tự tháp, bộ não của chúng ta vẫn có sự nghi ngờ về việc bức ảnh đã được chỉnh sửa, cụ thể ở đây là sử dụng bộ lọc làm mịn (là hệ quả của việc thu nhỏ ảnh và phóng to ảnh đã thu nhỏ). Chính vì điều đó mà bức ảnh trở nên thiếu tự nhiên đối với người xem. Ngược lại, ảnh sử dụng phương trình Poisson lại một lần nữa khẳng định được khả năng “đánh lừa” bộ não của con người trong việc xác định độ chân thực của bức hình, cho thấy triển vọng lớn về ứng dụng của phương trình trong xử lý ảnh.

**4. kết luận**

Song song cùng với sự phát triển của công nghệ, các kỹ thuật xử lý ảnh ngày càng trở lên phức tạp và tinh vi hơn. Trong bài báo này, chúng tôi đã sử dụng giải thuật Poisson cho việc thực hiện ghép ảnh tự động. Dựa trên giải thuật Poisson, các vùng ảnh ghép lộ biên đã được chỉnh sửa kỹ càng hơn để có một bức ảnh tự nhiên nhất có thể. Các kết quả thí nghiệm đã chứng tỏ giải thuật Poisson được đề xuất là phù hợp để chỉnh sửa hình ảnh tự động, không tốn nhiều công sức và độ chính xác cao hơn.

Không những vậy, thông qua việc so sánh với phương pháp Kim tự tháp sử dụng lọc Gaussian và Laplacian, sức mạnh của Poisson trong tự động chỉnh sửa ảnh được minh chứng mãnh liệt hơn nữa.

Mặc dù giải thuật được chứng minh là đúng đắn trong việc chỉnh sửa hình ảnh tự động, phương pháp này vẫn còn hạn chế với một số hình ảnh có sự khác biệt về khung cảnh quá lớn. Vì vậy, nó có thể được xem là hướng nghiên cứu quan trọng trong tương lai về lĩnh vực xử lý ảnh số.

**TÀI LIỆU THAM KHẢO**

1. AGRAWAL, A., AND RASKAR, R., Gradient Domain Manipulation Techniques in Vision and Graphics, 2007.
2. ELDER, J. H., AND GOLDBERG, R. M., Image editing in the contour domain, IEEE Trans. Pattern Anal. Mach. Intell. 23, 3, 2001, pp. 291-296.
3. GRADY, L., TASDIZEN, T., AND WHITAKER, R. T., A geometric multigrid approach to solving the 2d inhomogeneous laplace equation with internal dirichlet boundary conditions, In ICIP (2), 2005, pp. 642-645.
4. Patrick Pérez, Michel Gangnet, Andrew Blake., Poisson image editing, ACM, New York, NY, USA.: In ACM SIGGRAPH 2003 Papers (SIGGRAPH '03), 2003, pp. 313-318.
5. SAAD, Y., Iterative Methods for Sparse Linear Systems, 2nd ed, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA., 2003.
6. Gloria Bueno García, Oscar Deniz Suarez, José Luis Espinosa Aranda., Learning Image Processing with Open CV, March 2015, Chapter 6: Computation Photography – Section 3: Seamless cloning.