**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN II**

-----🙞🙜🕮🙞🙜-----



**BÁO CÁO THỰC TẬP**

**Đề tài:**

**NÉN VIDEO THEO MỘT SỐ GIẢI THUẬT UMHEXAGONS**

**Giảng viên hướng dẫn : TS. GVC. VÕ XUÂN THỂ**

**Sinh viên thực hiện : ĐỖ HOÀNG LONG**

**Mã số sinh viên : N14DCCN300**

**Lớp : D14CQCP01-N**

**Khoá** **: 2014**

**Hệ** **: ĐẠI HỌC CHÍNH QUY**

# 

# MỤC LỤC

[MỤC LỤC 1](#_Toc521367762)

[DANH MỤC CÁC BẢNG BIỂU 4](#_Toc521367763)

[LỜI MỞ ĐẦU 5](#_Toc521367764)

[CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 6](#_Toc521367765)

[1.1 Lý do chọn đề tài 6](#_Toc521367766)

[1.2 Mục tiêu đề tài 6](#_Toc521367767)

[1.3 Đối tượng và phạm vi nghiên cứu 6](#_Toc521367768)

[1.3.1 Đối tượng nghiên cứu 6](#_Toc521367769)

[1.3.2 Phạm vi nghiên cứu 7](#_Toc521367770)

[1.4 Phương pháp nghiên cứu 7](#_Toc521367771)

[1.5 Cấu trúc báo cáo 7](#_Toc521367772)

[CHƯƠNG 2: LÝ THUYẾT VỀ NÉN VIDEO 8](#_Toc521367773)

[2.1 Dữ liệu video 8](#_Toc521367774)

[2.1.1 Các khái niệm 8](#_Toc521367775)

[2.1.2 Dữ liệu ảnh tĩnh 12](#_Toc521367776)

[2.1.3 Phương pháp đánh giá chất lượng video 13](#_Toc521367777)

[2.2 Thuật toán ước lượng chuyển động 15](#_Toc521367778)

[2.2.1 Các khái niệm cơ bản 15](#_Toc521367779)

[2.2.2 Full Search 16](#_Toc521367780)

[2.2.3 Umhexagon Search 17](#_Toc521367781)

[2.3 Kỹ thuật bù chuyển động 21](#_Toc521367782)

[2.3.1 Bù chuyển động toàn phần 21](#_Toc521367783)

[2.3.2 Bù chuyển động theo khối 22](#_Toc521367784)

[CHƯƠNG 3: PHÂN TÍCH VÀ ĐANH GIÁ HIỆU QUẢ CỦA BỘ NÉN VIDEO 23](#_Toc521367785)

[3.1 Phân tích 23](#_Toc521367786)

[3.1.1 Ý tưởng 23](#_Toc521367787)

[3.1.2 Phương pháp 24](#_Toc521367788)

[3.2 Cài đặt các bộ nén 26](#_Toc521367789)

[3.3 Kết quả thực nghiệm 28](#_Toc521367790)

[CHƯƠNG 4: TỔNG KẾT 38](#_Toc521367791)

[TÀI LIỆU THAM KHẢO 39](#_Toc521367792)

[PHỤ LỤC 42](#_Toc521367793)

**DANH SÁCH CÁC HÌNH VẼ**

[Hình 2.1: Khung hình interlaced 8](#_Toc520603022)

[Hình 2.2: Sơ đồ minh họa video encoder 10](#_Toc520603023)

[Hình 2.3: Minh họa các khái niệm cơ bản trong kỹ thuật tìm kiếm 16](#_Toc520603024)

[Hình 2.4: Cấu trúc 4 mẫu tìm kiếm được sử dụng trong kỹ thuật UMHexagon Search 19](#_Toc520603025)

[Hình 2.5: Minh họa thuật toán tìm kiếm UM Hexagon 20](#_Toc520603026)

[Hình 3.1: Hai mẫu tìm kiếm mới: a) Bát giác 12 điểm và b) Thập giác 12 điểm 25](#_Toc520603027)

[Hình 3.2: Quá trình tìm kiếm của giải thuật do Chiến và Cảnh đề xuất 26](#_Toc520603028)

[Hình 3.3: Kết quả sau khi chạy phần mềm thực nghiệm 28](#_Toc520603030)

[Hình 3.4: Biểu đồ so sánh giữa các bộ nén video 35](#_Toc520603031)

[Hình 3.5: Biểu đồ so sánh bitrate giữa các bộ nén video 35](#_Toc520603032)

# 

# DANH MỤC CÁC BẢNG BIỂU

[Bảng 3.1: Cấu hình phần mềm JM Reference Software 27](#_Toc520601260)

[Bảng 3.2: Cấu hình máy tính chạy phần mềm thực nghiệm 28](#_Toc520601261)

[Bảng 3.3: Các đoạn video mẫu 29](#_Toc520601262)

[Bảng 3.4: Kết quả thực nghiệm các bộ nén với video mẫu Akiyo 30](#_Toc520601263)

[Bảng 3.5: Kết quả thực nghiệm các bộ nén với video mẫu Coastguard 31](#_Toc520601264)

[Bảng 3.6: Kết quả thực nghiệm các bộ nén với video mẫu Stefan 32](#_Toc520601265)

[Bảng 3.7: Kết quả thực nghiệm các bộ nén với video mẫu Container 33](#_Toc520601266)

[Bảng 3.8: Kết quả thực nghiệm các bộ nén với video mẫu Foreman 33](#_Toc520601267)

[Bảng 3.9: Kết quả thực nghiệm các bộ nén với video mẫu News 34](#_Toc520601268)

[Bảng 3.10: Bảng so sánh kết quả giữa ba bộ nén mới và UMHexagonS cải tiến so với UMHexagonS 37](#_Toc520601269)

# LỜI MỞ ĐẦU

Em xin gửi lời cảm ơn chân thành và sâu sắc tới các thầy cô trong khoa Công nghệ thông tin - trường Học viện Bưu chính Viễn Thông, những người đã tận tình giảng dạy và truyền đạt những kiến thức cần thiết, những kinh nghiệm quý báu cho chúng em trong suốt những năm tháng học tại trường Học viện Bưu chính Viễn Thông để em có thể tự tin khi thực hiện đồ án này.

Đặc biệt em xin chân thành cảm ơn giảng viên NGƯỜI HƯỚNG DẪN là người đã tận tình hướng dẫn, chỉ bảo, động viên và hỗ trợ em trong suốt quá trình thực hiện đề tài này. Trong thời gian làm đồ án cùng NGƯỜI HƯỚNG DẪN, em không những học hỏi được những kiến thức mà còn học hỏi được khả năng làm việc nghiêm túc, độc lập và có trách nhiệm với công việc của mình.

Xin cảm ơn các bạn lớp HỌC CÙNG- những người bạn đồng hành đã nhiệt tình động viên, ủng hộ, giúp đỡ mình trong suốt thời gian học tập cũng như thời gian thực hiện đồ án này.

Cuối cùng, em xin bày tỏ lòng biết ơn vô hạn đối với cha mẹ và gia đình những người thân xung quanh đã luôn động viên, khích lệ và tạo điều kiện tốt nhất cho chúng em trong quá trình học tập.

Mặc dù em đã có cố gắng hoàn thiện đồ án trong phạm vi và khả năng cho phép nhưng chắc chắn không tránh khỏi những thiếu sót. Em kính mong nhận được sự cảm thông và góp ý của quý thầy cô và các bạn.

Em xin chân thành cảm ơn!

*TP.HCM, tháng năm 2018*

Sinh viên

Đỗ Hoàng Long

# CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

## Lý do chọn đề tài

Ngày nay, với sự phát triển của công nghệ thông tin, hàng loạt cách thức trao đổi thông tin liên lạc giữa con người với nhau cũng trở nên đa dạng hơn. Không chỉ bằng những văn bản hoặc giọng nói đơn thuần, con người đã chú trọng hơn đến những dữ liệu đa phương tiện với sự kết hợp của hình ảnh và âm thanh. Trong đó, video chính là một trong những dữ liệu được nhiều người xem, chia sẻ và ứng dụng.

Nhu cầu thưởng thức và ứng dụng video trong cuộc sống của con người cũng ngày một nhiều hơn như: thưởng thức các video có độ phân giải cao trên TV online, sử dụng các ứng dụng video chat, hội nghị truyền hình trực tuyến với hình ảnh sắc nét và trung thực... trong khi tốc độ phát triển của băng thông và phần cứng không theo kịp so với nhu cầu. Do đó, việc nén dữ liệu video được xem như là một giai đoạn bắt buộc nhằm phục vụ quá trình truyền tải video đến người dùng.

Theo thời gian, các nhà nghiên cứu đã đề xuất nhiều thuật toán và phương pháp mới để nâng cao hiệu suất của quá trình nén video. Một trong những bước tiến quan trọng trong kỹ thuật nén video là sử dụng ước lượng chuyển động để nén thông tin hiệu quả hơn.

Từ những lýdo trên, em đã chọn đề tài ***“Tìm hiểu một số bộ nén video dựa trên thuật toán ước lượng chuyển động”***.

## Mục tiêu đề tài

* Nghiên cứu một số thuật toán liên quan đến bài toán nén và giải nén video.
* Tìm hiểu về cài đặt thuật toán ước lượng chuyển động mới trên phần mềm tham khảo JM Reference Software nhằm kiểm nghiệm hiệu quả của bộ nén video.
* Đánh giá hiệu quả của một số bộ nén video bằng thực nghiệm.

## Đối tượng và phạm vi nghiên cứu

### Đối tượng nghiên cứu

* Một số thuật toán liên quan đến bài toán nén và giải nén video.
* Thuật toán ước lượng chuyển động
* Phần mềm tham khảo JM Reference Software

### Phạm vi nghiên cứu

Cài đặt thuật toán ước lượng chuyển động trên phần mềm tham khảo JM Reference Software phiên bản 19 nhằm kiểm nghiệm hiệu quả của bộ nén video.

## Phương pháp nghiên cứu

Nghiên cứu lý thuyết, kết hợp thực nghiệm.

## Cấu trúc báo cáo

Nội dung báo cáo đồ án gồm bốn chương:

Chương 1: GIỚI THIỆU

Chương 2: LÝ THUYẾT VỀ NÉN VIDEO

Chương 3: PHÂN TÍCH VÀ ĐÁNH GIÁ HIỆU QUẢ CỦA MỘT SỐ BỘ NÉN VIDEO

Chương 4: KẾT LUẬN VÀ ĐỀ XUẤT HƯỚNG LỰA CHỌN BỘ NÉN VIDEO PHÙ HỢP

# CHƯƠNG 2: LÝ THUYẾT VỀ NÉN VIDEO

## 2.1 Dữ liệu video

### 2.1.1 Các khái niệm

#### **2.1.1.1 Bitrate**

Bitrate là chỉ số dùng để biểu thị cho lượng dữ liệu mà video được truyền hoặc phát trong khoảng thời gian một giây.

Đơn vị đo: bits per second (bps) hoặc các đơn vị lớn hơn như kilobits per second (kbps), megabits per second (Mbps), gigabits per second (Gbps) và terabits per second (Tbps).

Mục tiêu của việc thiết kế các bộ nén video: làm cho chỉ số bitrate thấp mà chất lượng của video vẫn ở mức chấp nhận được.

#### **2.1.1.2 Frame rate**

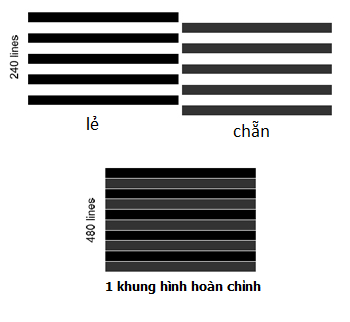
Frame rate là hệ số cho biết có bao nhiêu khung hình trong video được hiển thị trong khoảng thời gian một giây. Số frame rate càng cao thì video đó càng mượt và dung lượng của video đó cũng càng lớn.

Đơn vị đo: frame per second.

Một số frame rate chuẩn thường gặp là 24fps, 25fps, 30fps và 60fps.

#### **2.1.1.3 Interlaced và Progressive**

Interlaced video là các video mà mỗi khung hình được chia thành hai phần ảnh khác nhau, một phần bao gồm các dòng ảnh lẻ và một phần bao gồm các dòng ảnh chẵn, mỗi phần ảnh là dữ liệu được chụp tại hai thời điểm khác nhau. Khi được trình chiếu, phần ảnh lẻ và phần ảnh chẵn sẽ được chiếu xen kẽ nhau.



***Hình 2.1: Khung hình interlaced***

Progressive video là các video mà mỗi khung hình chứa dữ liệu ảnh của một thời gian nào đó. Khi trình chiếu, toàn bộ khung hình sẽ được hiển thị tại một thời điểm.

Lợi ích của interlaced video: giảm băng thông truyền tải hoặc nâng cao frame rate so với progressive video vì chỉ một nửa dữ liệu của toàn khung hình được xử lý tại một thời điểm.

#### **2.1.1.4 Codec**

Codec là viết tắt của coder/decoder dùng để chỉ một phần mềm có khả năng mã hóa và giải mã các dữ liệu video. Codec không phải là định dạng hay chuẩn video vì nó đề cập đến các phần mềm làm việc trên video, nhưng trên thực tế thường được sử dụng để chỉ các chuẩn hay định dạng video.

Có thể chia Codec thành hai loại phổ biến như sau:

* Loại đuôi tập tin sẽ chỉ ra loại codec file đó sẽ sử dụng, bao gồm: .WMV .RM .RMVB .MPG .MPEG .MOV… ;
* Loại đuôi tập tin không chỉ chính xác ra loại codec file đó sẽ sử dụng, mà chỉ thể hiện đó là file media: .AVI .MKV .OGM .MP4…

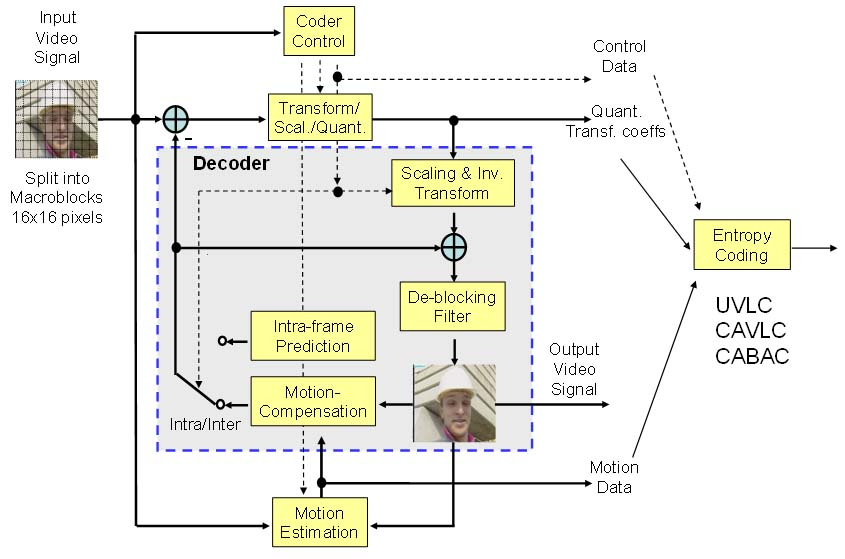
Đặc điểm của hai loại trên:

* Chỉ là các container để chứa nội dung.
* Việc nội dung đó như thế nào, cách ghi ra sao chỉ khi trình media player mở ra và đọc nó sẽ biết file đó sử dụng codec gì. Nói cách khác, các file media đều thuộc loại thứ hai. Đuôi tập tin chỉ nói lên chúng là file media mà thôi.

Encoder là một chương trình xử lý dùng để chuyển đổi dữ liệu từ một định dạng video sang một định dạng khác nhằm mục đích chuẩn hóa hoặc nén nhằm tiết kiệm dung lượng. Có hai loại encoder trong xử lý video:

* Lossless encoder: không làm mất dữ liệu
* Lossy encoder có làm mất dữ liệu.
* Hai encoder của cùng một chuẩn video có thể xuất ra những đoạn video với chất lượng khác nhau. Một đoạn video khi đưa vào encoder sẽ trải qua nhiều quá trình khác nhau để xuất ra kết quả là một tập tin video đã được nén. Hình 2.2 minh họa sơ đồ hoạt động của một video encoder tổng quát.

Ngược lại với encoder, Decoder là giải mã các thông tin đã được mã hóa bởi encoder nhằm khôi phục các thông tin dữ liệu gốc. Trong thực tế, các chuẩn video khác nhau sẽ đặc tả đầy đủ thông tin dùng để giải mã các video được nén theo chuẩn đó. Vì thế, cùng một chuẩn video có thể có nhiều encoder cũng như decoder khác nhau, miễn sao đoạn video được tạo ra thỏa mãn các đặc tả của chuẩn video đó. Decoder thường sẽ có các thành phần tương tự như encoder, chỉ có điều hoạt động theo các bước ngược lại so với khi mã hóa.



***Hình 2.2: Sơ đồ minh họa video encoder***

#### **2.1.1.5 Một số chuẩn nén video**

MPEG­1 là một trong những mô hình nén đầu tiên được phát triển bởi tổ chức MPEG và vẫn còn được sử dụng cho đến ngày hôm nay trong một số CD-ROM hay các chương trình phát nhạc Windows Media. MPEG­1 sử dụng kỹ thuật Biến đổi Cosin rời rạc để chuyển đổi tín hiệu sang miền tần số, sau đó xử lý các giá trị tần số này để thực hiện việc nén video. Định dạng âm thanh MP3, một trong những định dạng phổ biến nhất hiện nay, về thực chất cũng là một phần của chuẩn nén MPEG­1.

MPEG­2 là chuẩn nén này được phát triển để đáp ứng nhu cầu nén các đoạn video chất lượng cao. Bitrate của các video được nén bằng MPEG­2 thường dao động trong khoảng từ 5 đến 8Mbps. Nguyên tắc nén của MPEG­2 cũng gần giống với MPEG­1 khi sử dụng biến đổi cosin rời rạc để xử lý, nhưng có hỗ trợ interlaced video. Ngày nay, MPEG­2 được sử dụng trong các DVD video, truyền thông số qua vệ tinh hoặc cáp, và trong các hệ thống truyền tải truyền hình độ nét cao (High Definition Television).

MPEG­3, đây từng là một dự án của hội đồng MPEG khi phát triển một chuẩn video mới hỗ trợ truyền hình độ nét cao. Tuy nhiên, sau đó người ta nhận thấy rằng chỉ cần chỉnh sửa một chút trong chuẩn MPEG­2 thì đã có thể hỗ trợ HDTV cho nên MPEG­3 không được tiếp tục phát triển.

MPEG­4 không đơn thuần là một chuẩn nén và giải nén video. MPEG­4 được thiết kế để thực hiện nhiều công việc khác nhau như thu hình, chỉnh sửa, mã hóa, chuyển phát và lưu trữ. Nó là bộ khung chứa cho các đối tượng media ngoài video và audio, ví dụ như văn bản, hình ảnh tĩnh, ảnh động, liên kết web… Mục tiêu lớn nhất của chuẩn MPEG­4 là cố gắng giải quyết hai vấn đề tồn đọng trong việc truyền video: gửi video qua các đường truyền băng thông thấp như Internet hoặc các thiết bị di động; đạt hiệu quả nén cao hơn so với MPEG­2 đối với các tín hiệu truyền thông. Bitrate của MPEG­4 thường giao động trong khoảng 64kbps đến 1,800Mbps. Tuy nhiên, hiệu quả MPEG­4 đạt được đối với các tín hiệu truyền thông lại không cao, do đó MPEG­2 vẫn tiếp tục được sử dụng.

Sau thất bại của MPEG­4 trong việc thay đổi ngành công nghiệp truyền thông, hai tổ chức ITU­T Video Coding Experts Group (VCEG) và ISO/IEC Moving Picture Experts Group (MPEG) đã phát triển một chuẩn video mới có hiệu suất cao hơn MPEG­2 để có thể áp dụng rộng rãi trong tương lai, đó là chuẩn H.264/AVC. Nó còn được gọi là chuẩn MPEG­4 Part 10 AVC (Advanced Video Compression), mặc dù không hề liên quan gì đến chuẩn MPEG­4. Mục tiêu chính của H.264/AVC là đạt hiệu suất nén cao rõ rệt trong các ứng dụng truyền video qua mạng, kể cả các ứng dụng hội nghị truyền hình (video conferecing) hoặc các ứng dụng không hội thoại (lưu trữ, truyền thông hoặc streaming).

H.265 hay HEVC (High Efficiency Video Coding) được Ủy ban Viễn thông Quốc tế ITU­T thông qua và bắt đầu được các nhà phát triển đưa vào sản phẩm thương mại. H.265 hứa hẹn mang lại khả năng nén cao gấp đôi so với người tiền nhiệm H.264/AVC (Advance Video Coding) hiện đang được dùng rất phổ biến. H.265/HEVC mở rộng cách H.264/AVC làm việc. Đầu tiên, nó sẽ kiểm tra các khung hình để xem giữa các khung hình có gì thay đổi không. Hầu hết, các khung hình trong một cảnh video không có nhiều thay đổi. Ví dụ, một cảnh biên tập viên đọc tin, các ảnh nền sẽ không thay đổi nhiều trong các khung hình khác nhau, ngay cả các điểm ảnh trên khuôn mặt của biên tập viên này cũng sẽ không thay đổi qua các khung hình ngoại trừ khẩu hình. Vậy thay vì mã hóa tất cả các điểm ảnh từ mỗi khung hình, khung ban đầu được mã hóa và sau đó chỉ những gì thay đổi trên khung tiếp theo mới được mã hóa. HEVC mở rộng kích thước của khu vực để dựa vào đó tìm ra những điểm ảnh thay đổi.

### 2.1.2 Dữ liệu ảnh tĩnh

Về bản chất, video là tập hợp các hình ảnh tĩnh được trình chiếu liên tiếp nhau tạo cho người xem một cảm giác đó là những hình ảnh chuyển động. Mỗi hình ảnh số là một ma trận có kích thước các điểm ảnh, với là chiều rộng và là chiều cao của bức ảnh. Tại mỗi điểm ảnh sẽ là các giá trị được lưu trữ nhằm quy định màu sắc của điểm ảnh đó. Tùy theo hệ màu của ảnh mà cách thức lưu trữ tại mỗi điểm ảnh sẽ khác nhau.

* Hệ màu RGB

Trong hệ màu RGB, mỗi điểm ảnh được lưu trữ bằng một bộ ba giá trị màu cơ bản là: đỏ (red), xanh lá cây (green) và xanh dương (blue). Khi đó, bất cứ màu sắc nào mà mắt người cảm nhận được có thể được xem là sự kết hợp tuyến tính của ba màu cơ bản này. Hệ màu RGB thường được sử dụng trong các ứng dụng chụp và hiển thị ảnh. Trong khi đó, những thao tác nén ảnh và video thường được thao tác trên hệ màu luminance­chrominance vì nó hiệu quả hơn.

* Hệ màu YUV

Trong hệ màu này, Y là thể hiện cho độ sáng của màu (thành phần luminance), còn U và V thể hiện cho hai thành phần màu (chrominance). Các giá trị của Y, U và V có thể được tính từ ba thành phần R, G, B của màu đó như sau:

*Y = 0.299R + 0.587G + 0.114B*

*U = 0.492(B ‒ Y )*

*V = 0.877(R ‒ B)*

Trong ba thành phần Y, U, V thì thành phần Y là quan trọng nhất đối với một bức ảnh vì nó quy định các chi tiết trên một bức ảnh là như thế nào. Còn hai thành phần U, V có nhiệm vụ bổ sung màu sắc cho bức ảnh đó. Bức ảnh chỉ chứa thành phần Y được gọi là ảnh xám (grayscale). Do đó, trong các ứng dụng nén ảnh và video, người ta thường chỉ thao tác trên thành phần Y của bức ảnh.

* Hệ màu YCBCR

Từ các công thức tính giá trị U và V trong hệ màu YUV, ta thấy rằng hai số này có thể mang giá trị âm. Do đó, để cho chúng không mang giá trị âm thì ba thành phần màu Y, U và V được dịch chuyển trên trục giá trị. Các giá trị mới thuộc hệ màu YCbCr, được sử dụng trong các chuẩn quốc tế như JPEG hay MPEG. Công thức chuyển đổi giá trị từ hệ màu RGB sang YCbCr như sau:

### 2.1.3 Phương pháp đánh giá chất lượng video

Để so sánh hai phương pháp nén video khác nhau, người ta cần phải đánh giá chất lượng video sau khi nén. Nếu như chất lượng của hai video như nhau thì bộ nén nào tạo ra dữ liệu có kích thước nhỏ hơn sẽ được xem là bộ nén tốt hơn. Ngược lại, nếu như hai bộ nén tạo ra cùng một lượng dữ liệu như nhau, bộ nén nào cho video chất lượng hơn thì sẽ là bộ nén tốt hơn.

Người ta thường sử dụng hai phương pháp đánh giá chất lượng video:

* Đánh giá chủ quan (dựa trên sự quan sát của con người);
* Đánh giá khách quan (dựa trên các tính toán số học).
* Đánh giá chủ quan

Đây là phương pháp đánh giá bằng công thức tính toán dựa trên dữ liệu của video sau khi nén. Phương pháp đánh giá được sử dụng rộng rãi hiện nay là tỉ lệ tín hiệu tới nhiễu (SNR ­ Signal to Noise Ratio).

Xét là hình ảnh đầu vào của hệ thống nén, là hình ảnh xuất bởi hệ thống này. Để đánh giá chất lượng của , hàm lỗi được định nghĩa như sau:

*e(x, y) = f(x, y) ‒ g(x, y)*

Sai số bình phương trung bình, *Ems*, được định nghĩa là:

*Ems =*

với M và N là chiều rộng và chiều cao của bức ảnh.

Khi đó, chỉ số SNR được tính như sau:

*SNRms =*

Trong nén ảnh và video, có một chỉ số khác gần giống như chỉ số SNR được sử dụng rộng rãi, đó là chỉ số PSNR (Peak Signal to Noise Ratio), được định nghĩa như sau:

*PSNR =*

Dựa theo cách tính toán trên, ta có thể thấy rằng chỉ số PSNR (hoặc SNR) càng lớn thì chất lượng của bức ảnh đã nén càng cao. Nhưng trên thực tế, điều này có thể không chính xác vì hệ thống nhận biết hình ảnh của con người còn phụ thuộc vào rất nhiều thành phần hoặc chi tiết trên bức ảnh. Đôi khi những bức ảnh có chỉ số PSNR cao lại là những bức ảnh được đánh giá có chất lượng thấp khi sử dụng phương pháp đánh giá chủ quan.

Chỉ số PSNR hoặc SNR có thể không luôn luôn chính xác, nhưng tốc độ và sự đơn giản của nó thì vượt trội hoàn toàn so với phương pháp đánh giá chủ quan. Vì vậy, phương pháp này vẫn được sử dụng rất phổ biến trong các nghiên cứu để đánh giá hiệu quả các các phương pháp nén.

* Đánh giá khách quan

Theo phương pháp này, đoạn video sau khi nén sẽ được trình chiếu cho một nhóm người thử nghiệm. Mỗi nhóm người sẽ được phát các thang điểm để đánh giá chất lượng của video đó. Sau đó sẽ tổng hợp kết quả của tất cả những người tham gia để tính ra giá trị đánh giá chất lượng trung bình của video. Một trong những thang điểm chuẩn được khuyên dùng bởi tổ chức ITU­R là như sau:

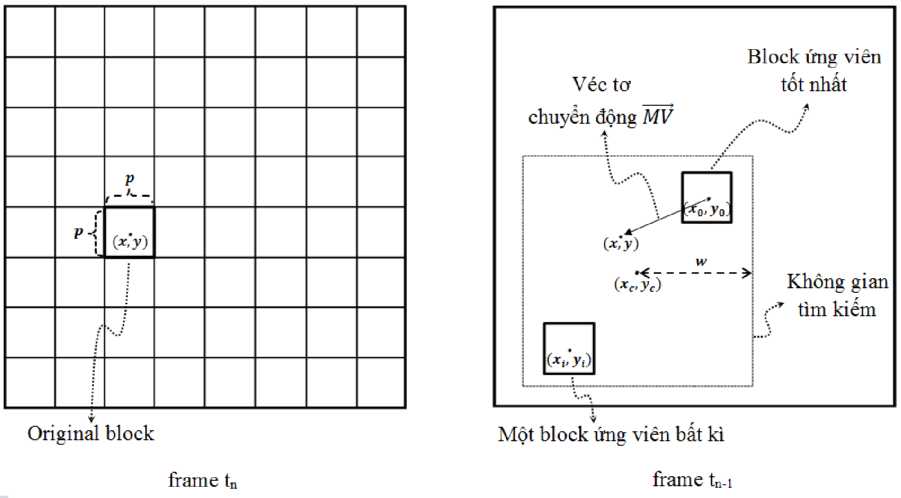
* Không tìm thấy các khiếm khuyết.
* Các khiếm khuyết khó nhận thấy.
* Các khiếm khuyết có thể nhận thấy được, nhưng không gây khó chịu.
* Các khiếm khuyết gây khó chịu.
* Các khiếm khuyết cực kỳ gây khó chịu.

Nhược điểm của phương pháp này đó là tốn nhiều thời gian và công sức của người quan sát để đánh giá chất lượng video.

## 2.2 Thuật toán ước lượng chuyển động

### Các khái niệm cơ bản

* **Block:** Được hiểu là một khối ảnh, bao gồm các điểm ảnh nằm liền nhau, có hình dạng và kích thước xác định. Vị trí của block được hiểu là tọa độ của pixel trung tâm của block đó. Ta có vị trí của original block trong hình 2.3 là ;
* **Original block:** Một block đang xử lý thuộc frame . Trong các kĩ thuật tìm kiếm phổ biến hiện nay, original block có dạng hình vuông và có kích thước hoặc ;
* **Vùng tìm kiếm:** Việc tìm kiếm trên toàn bộ frame (frame liền trước của frame ) là tốn nhiều thời gian và không cần thiết. Để tăng tốc độ tính toán, người ta chỉ thực hiện việc tìm kiếm trong một khu vực tìm kiếm xác định, được đọi là vùng tìm kiếm. Thông thường vùng tìm kiếm này có dạng hình vuông. Phạm vi tìm kiếm được hiểu là khoảng cách từ trung tâm của vùng tìm kiếm đến biên của vùng tìm kiếm như hình 2.16;
* **Block ứng viên:** Là một block bất kì có vị trí nằm trong vùng tìm kiếm, có cùng kích thước với original block. Một vùng tìm kiếm có phạm vi tìm kiếm thì sẽ chứa block ứng viên. Block ứng viên tốt nhất là block được đánh giá là tương đồng với original block nhất;
* **Giá trị BDM (Block Distortion Measure) của block ứng viên(ngầm hiểu là so với original block):** Là chỉ số để dánh giá sự tương đồng của block ứng viên so với original block. Chỉ số này có thể là một trong các chỉ MAD, MSE, SAD, SSE. Chi tiết về phương pháp đánh giá đã được trình bày trong *chương II, mục 5.1*;
* Véc tơ chuyển động được xác định bằng công thức . Với lần lượt là vị trí của original block và block ứng viên tốt nhất.



***Hình 2.3: Minh họa các khái niệm cơ bản trong kỹ thuật tìm kiếm***

Với mỗi original block nằm trong frame tn, nhiệm vụ của kỹ thuật tìm kiếm là xác định trong frame một block ứng viên tốt nhất. Dựa trên tọa độ của của original block và block ứng viên tốt nhất vừa tìm được, tính toán và trả về giá trị của vectơ chuyển động . Giá trị vectơ sẽ được sử dụng làm thông tin cho việc nén video theo phương pháp bù chuyển động.

### 2.2.2 Full Search

Full search là kỹ thuật tìm kiếm ra đời đầu tiên và cũng là cơ bản nhất.

Ý tưởng: lần lượt duyệt qua tất cả các block ứng viên nằm trong vùng tìm kiếm, tính giá trị BDM của từng block ứng viên. Từ đó chọn ra ứng viên có giá trị BDM nhỏ nhất. Thuật toán này đảm bảo luôn tìm ra block ứng viên tốt nhất.

Nhược điểm duy nhất của nó số lượng phép tính cần thực hiện quá lớn. Chính nguyên nhân này đã thúc đẩy sự ra đời các thuật toán tìm kiếm nhanh (Fast search) nhằm cải thiện tốc độ tìm kiếm.

Đặc điểm của các giải thuật tìm kiếm nhanh là không thực hiện tìm kiếm trên tất cả các ứng viên, mà chỉ thực hiện tìm kiếm trên một số ứng viên nhất đinh. Do đó, nhược điểm là không đảm bảo được là luôn tìm được block ứng viên tốt nhất trong mọi trường hợp. Ưu điểm của giải thuật tìm kiếm nhanh là cải thiện được tốc độ tìm kiếm một cách đáng kể.

### 2.2.3 Umhexagon Search

UMHexagon Search là một kỹ thuật tìm kiếm hiện đại, được giới thiệu năm 2004 bởi nhóm tác giả Zhibo Chen, Jianfeng Xu, Yun He và Junli Zheng [2, 5]. Đây là một kỹ thuật phức tạp, cho tốc độ tìm kiếm nhanh, chất lượng tìm kiếm tốt và đang được sử dụng phổ biến trong việc nén video theo phương pháp bù chuyển động.

**Vùng tìm kiếm:** Trung tâm của vùng tìm kiếm trong kỹ thuật UM Hexagon được xây dựng theo các bước sau:

* Xác định vectơ chuyển động dự đoán của original block. được xác định dựa trên các thông tin của các block xung quanh nó, block ở cùng vị trí và các block xung quanh trong frame liền trước, hoặc sử dụng phương pháp dự đoán dựa trên việc chia nhỏ và tính toán các thông tin nội tại của block.
* Vị trí của trung tâm vùng tìm kiếm được xác định bằng cách dịch chuyển vị trí của original block theo vec tơ chuyển động dự đoán ;
* Dựng vùng tìm kiếm có phạm vi với trung tâm được xác định ở bước trên.

**Điều kiện kiểm tra việc kết thúc sớm:** Trong quá trình tìm kiếm ứng viên tốt nhất, thuật toán tìm kiếm sẽ liên tục thực hiện việc kiểm tra điều kiện kết thúc sớm. Việc kết thúc sớm nhằm mục đích đẩy nhanh tốc độ tìm kiếm, ngay sau khi tìm được một ứng viên tốt (giá trị BDM của ứng viên ở dưới ngưỡng cho phép) thuật toán sẽ dừng tìm kiếm trên phạm vi toàn cục mà chuyển sang giai đoạn tìm kiếm cục bộ xung quang ứng viên tốt. Việc kiểm tra điều kiện kết thúc sớm được thực hiện như sau:

* Kiểm tra điều kiện :
* Nếu đúng, dến bước 4.2;
* Nếu sai, Kiểm tra điều kiện CURRENT\_MIN < BDMT1.
* Nếu đúng, dến bước 4.1;
* Nếu sai, tiếp tục.

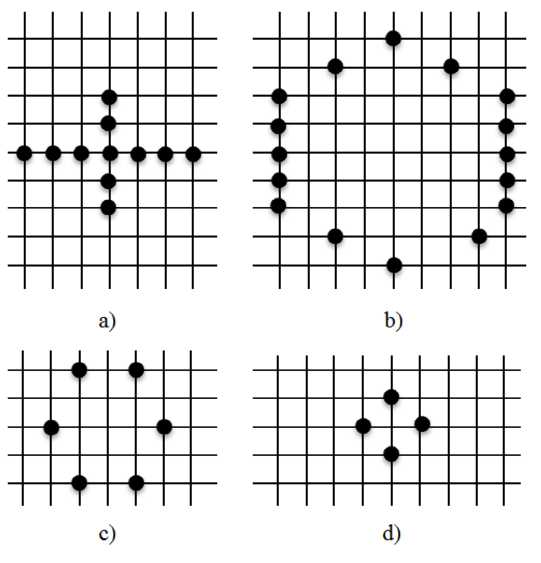
Với và là hai giá trị ngưỡng cho phép thực hiện việc kết thúc sớm mà không làm ảnh hưởng lớn đến chất lượng tìm kiếm. và được tính theo các công thức sau:

Trong đó:

* : Giá trị BDM dự đoán. Tương tự với cách xác định véc tơ chuyển động dự đoán , cũng được tính toán dựa trên các block cung quanh, block ở cùng vị trí và các block xung quanh trong frame liền trước, thông tin nội tại của block;
* và : Là hai hằng số, nếu và có giá trị càng cao thì tốc độ tìm càng cao, nhưng chất lượng tìm kiếm sẽ giảm và ngược lại.

Phương pháp tính , và tương đối phức tạp và được trình bày chi tiết trong tài liệu [2, 5].

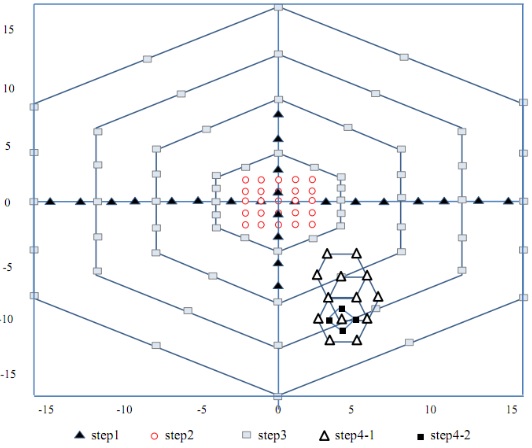
**Mẫu tìm kiếm:** Kỹ thuật UM Hexagone sử dụng 4 mẫu tìm kiếm có cấu trúc như hình 2.4.

**

***Hình 2.4: Cấu trúc 4 mẫu tìm kiếm được sử dụng trong kỹ thuật UMHexagon Search***

Thuật toán tìm kiếm:

* Bước 1: Tìm kiếm trên mẫu tìm kiếm hình chữ thập không cân đối­hình 2.4a. Một trong những đặc điểm chuyển động của các đối tượng trong video là các đối tượng có xu hướng chuyển động theo phương ngang phổ biến hơn so với các đối tượng theo phương thẳng đứng. Do đó việc tìm kiếm sẽ được ưu tiên tìm kiếm theo phương ngang. Tính giá trị BDM của các block ứng viên trên mẫu tìm kiếm. Gán CURRENT\_MIN bằng giá trị BDM nhỏ nhất tìm được. Kiếm tra việc kết thúc sớm;
* Bước 2: Tìm kiếm theo thuật toán Full Search với phạm vi tìm kiếm . Tính giá trị BDM của các block ứng viên trên mẫu tìm kiếm. Gán LOCAL\_MIN bằng giá trị BDM nhỏ nhất tìm được. Nếu LOCAL\_MIN < CURRENT\_MIN thì gán giá trị CURRENT\_MIN = LOCAL\_MIN. Kiểm tra việc kết thúc sớm;
* Bước 3: Tìm kiếm trên lưới bao gồm nhiều mẫu tìm kiếm hình lục giác lớn­hình 2.4b đồng dạng có kích thước khác nhau. Tính giá trị BDM của các block ứng viên trên lưới tìm kiếm. Gán LOCAL\_MIN bằng giá trị BDM nhỏ nhất tìm được. Nếu LOCAL\_MIN < CURRENT\_MIN thì gán giá trị CURRENT\_MIN = LOCAL\_MIN. Kiểm tra việc kết thúc sớm;
* Bước 4:
* Bước 4.1: Tạo một mẫu tìm kiếm lục giác nhỏ­hình 2.4c có tâm trùng với CURRENT\_MIN. Tính giá trị BDM của các block ứng viên trên mẫu tìm kiếm. Gán CURRENT\_MIN bằng giá trị BDM nhỏ nhất tìm được.
* Nếu vị trí CURRENT\_MIN trùng với tâm mẫu tìm kiếm. Đến bước 4.2;
* Nếu không, quay lại bước 4.1.
* Bước 4.2: Tạo một mẫu tìm kiếm Diamond­hình 2.4d có tâm trùng với vị trí của CURRENT\_MIN. Tính giá trị BDM của các block ứng viên trên mẫu tìm kiếm. Gán CURRENT\_MIN bằng giá trị BDM nhỏ nhất tìm được
* Nếu vị trí CURRENT\_MIN trùng với tâm mẫu tìm kiếm. Dừng lại;
* Nếu không, quay lại bước 4.2.
* Bước 5: Kết thúc thuật toán, trả về véc tơ chuyển động dựa trên vị trí hiện tại của CURRENT\_MIN.



***Hình 2.5: Minh họa thuật toán tìm kiếm UM Hexagon***

## 2.3 Kỹ thuật bù chuyển động

Trong một đoạn video, sự thay đổi giá trị pixel giữa các khung hình thường có mối quan hệ với nhau tùy thuộc vào chuyển động của các vật thể trên video đó. Do vậy, thay vì nén một khung hình riêng lẻ sử dụng thông tin nội tại từ video đó, ta có thể mã hóa thông tin một khung hình bằng thông tin của khung hình phía trước đó.

Kỹ thuật bù chuyển động giả định rằng sự thay đổi của các điểm ảnh trong video chẳng qua là sự thay đổi tịnh tiến của những đối tượng trên video đó. Do vậy, thay vì nén tín hiệu của các điểm ảnh trên khung hình hiện tại dựa trên điểm ảnh tại cùng vị trí trên khung hình trước, thì ta thực hiện việc ước lượng xem vị trí tương đối của điểm ảnh đó trên khung hình trước là ở đâu. Sau đó, giá trị khác biệt giữa điểm ảnh này và điểm ảnh ước lượng trên khung hình trước sẽ được mã hóa. Phương pháp này tận dụng thông tin chuyển động của vật thể trong video để mã hóa thông tin hiệu và tỏ ra hiệu quả hơn so với phương pháp làm đầy khung hình.

### 2.3.1 Bù chuyển động toàn phần

Kỹ thuật bù chuyển động toàn phần (GMC ­ Global Motion Compensation) giả định các đối tượng trong một khung hình có thể biểu diễn bằng một vectơ chuyển động duy nhất so với khung hình trước đó. Điều này sẽ có ích trong các trường hợp cảnh trong khung hình dịch chuyển tuyến tính theo một chiều nào đó do sự thay đổi của camera như xoay, phóng to hoặc di chuyển.

Mỗi khung hình trong phương pháp bù chuyển động toàn phần sẽ được ước lượng chuyển động bằng các điểm đánh dấu (Warp Point). Các chuẩn nén video bù chuyển động toàn phần thường sử dụng 1 điểm đánh dấu (trong phần mềm DivX) hoặc 3 điểm đánh dấu (trong XviD). Sử dụng 1 điểm đánh dấu chỉ cho phép mô tả các chuyển động tịnh tiến, trong khi nếu sử dụng 3 điểm đánh dấu thì có thể mô tả được các thay đổi khác trên khung hình như xoay hoặc phóng to.

GMC không cải thiện nhiều hiệu quả của phương pháp bù chuyển động mà lại làm tăng độ phức tạp trong quá trình tính toán, cho nên nó đã bị loại bỏ trong đặc tả của chuẩn nén H.264/AVC. Hiện tại, hầu hết các phần cứng đều không hỗ trợ bù chuyển động toàn phần.

### 2.3.2 Bù chuyển động theo khối

Trong phương pháp này, mỗi khung hình sẽ được chia nhỏ thành các khối ảnh hình chữ nhật gọi là block. Sau đó, việc ước lượng chuyển động sẽ được thực hiện trên những khối ảnh này thay vì ước lượng chuyển động của toàn bộ khung hình. Trong trường hợp các đối tượng trong khung hình có sự biến đổi đặc biệt hoặc các đối tượng đó có hình dạng bất kỳ, phương pháp bù chuyển động theo khối vẫn tỏ ra hiệu quả hơn nhờ việc xấp xỉ các thay đổi bất kỳ trên đối tượng thành những thay đổi tịnh tiến trên các khối ảnh nhỏ. Sau đó, vectơ chuyển động của mỗi khối ảnh này cũng sẽ được mã hóa đi kèm với dữ liệu ảnh.

# CHƯƠNG 3: PHÂN TÍCH VÀ ĐANH GIÁ HIỆU QUẢ CỦA BỘ NÉN VIDEO

## 3.1 Phân tích

Thuật toán UMHexagonS là thuật toán chính dùng để ước lượng chuyển động các khối ảnh. Thuật toán này chạy khá nhanh và cho ra vectơ chuyển động tốt trên cả những video có hình ảnh chuyển động chậm và nhanh. Năm 2009, Lai-Man Po đã giới thiệu thuật toán FDGDS nhằm giảm thiểu thời gian ước lượng chuyển động so với thuật toán UMHexagonS và giữ cho bitrate cũng như chỉ số PSNR của video không thay đổi nhiều. Vấn đề là thuật toán FDGDS thường làm tăng bitrate của video lên nhiều hơn UMHexagonS trong những video có nhiều cảnh chuyển động nhanh.

### 3.1.1 Ý tưởng

Giải thuật UMHexagonS gồm hai bước tìm kiếm chữ thập không đối xứng và tìm kiếm Full Search trong khu vực có kích thước được dùng để tìm kiếm sơ bộ điểm tối ưu. Để giảm thời gian thực thi mà vẫn cho ra kết quả gần chính xác như ban đầu, năm 2012, nhóm Trần Xuân Chiến và Phan Công Cảnh (nhóm Chiến và Cảnh) đã đề xuất sử dụng giải thuật FDGDS nhằm tận dụng khả năng tìm kiếm điểm tối ưu trong thời gian ngắn của thuật toán [1].

Tại bước tìm kiếm theo lưới lục giác, UMHexagonS sử dụng một mẫu tìm kiếm có dạng hình lục giác với 16 điểm trên đó. Vì đa số các video có xu hướng chuyển động ngang nên các điểm tập trung nhiều về hai phía theo chiều ngang. Li Hong-ye đã đề xuất một mẫu tìm kiếm mới có dạng lục giác 8 điểm nhằm tăng tốc độ thực thi bằng cách giảm một nửa số lượng điểm cần tìm trên mỗi lượt [12]. Trong bài báo cáo năm 2012, nhóm Chiến và Cảnh đã cải tiến khác đi và sử dụng hai mẫu tìm kiếm dạng bát giác 12 điểm và thập giác 12 điểm. Hai mẫu tìm kiếm này được sử dụng xen kẽ nhau giữa các lượt tìm kiếm nhằm tìm cách phủ rộng các điểm được xét trên toàn bộ vùng tìm kiếm [1].

Dựa trên hai ý tưởng này, nhóm Chiến và Cảnh đã xây dựng một giải thuật mới từ giải thuật UMHexagonS giúp giảm thiểu thời gian tìm kiếm, trong khi đó các chỉ số bitrate và PSNR không có sự thay đổi đáng kể, thậm chí trên cả những video có hình ảnh chuyển động nhanh.

### 3.1.2 Phương pháp

#### **3.1.2.1 Tích hợp thuật toán FDGDS**

Như đã trình bày, một trong những bước tìm kiếm đầu tiên của giải thuật UMHexagonS là tìm kiếm chữ thập bất đối xứng và tìm kiếm Full Search trong khu vực **.** Sau khi thực hiện các bước này, UMHexagonS kiểm tra điều kiện để xem có thể kết thúc nhanh chóng việc tìm kiếm hay không. Do bước kiểm tra điều kiện kết thúc là một trong những bước rất quan trọng trong UMHexagonS, cho nên nhóm Chiến và Cảnh đã tích hợp bước kiểm tra này vào ngay trong giải thuật FDGDS để tăng tốc độ tìm kiếm hơn nữa. Giải thuật tìm kiếm FDGDS mới sẽ như sau:

Bước 1: Tính giá trị sai lệch tại điểm bắt đầu tìm kiếm và gán nó là CURRENT\_MIN;

Bước 2: Với mỗi hướng tìm kiếm trong tám hướng xung quanh điểm CURRENT\_MIN:

* Thực hiện chiến lược OTS trên hướng đó;
* Gán giá trị sai lệch nhỏ nhất tìm được trên hướng đó là DIRECTIONAL\_MIN;
* Nếu tìm thấy giá trị DIRECTIONAL\_MIN, *kiểm tra điều kiện kết thúc nhanh và chuyển sang bước 5 nếu thỏa.* Nếu điều kiện không thỏa thì tính tỉ lệ nhiễu tương đối (RDR) của hướng đó. Nếu RDR < T, cập nhật CURRENT\_MIN với giá trị DIRECTIONAL\_MIN này và lặp lại bước 2 (bỏ qua các hướng tìm kiếm còn lại).

Bước 3: Nếu không tìm thấy DIRECTIONAL\_MIN nào cả thì chuyển sang bước 5, ngược lại chuyển sang bước 4.

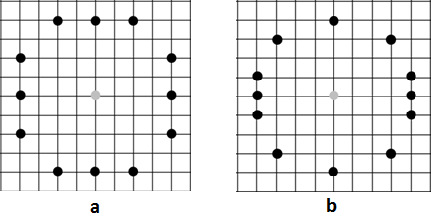
Bước 4: Các DIRECTIONAL\_MIN của các hướng được so sánh với nhau. Giá trị nhỏ nhất được chọn là CURRENT\_MIN. Kết thúc một lượt tìm kiếm, chuyển sang bước 2 với giá trị CURRENT\_MIN và vị trí tương ứng của nó;

Bước 5: Thuật toán kết thúc. Trả về vectơ chuyển động tìm được dựa trên vị trí của CURRENT\_MIN.

Một số ưu điểm của việc dùng giải thuật FDGDS thay thế cho hai bước tìm kiếm chữ thập không đối xứng và tìm kiếm toàn bộ là số lượng điểm cần tìm trong giải thuật FDGDS không cố định như phương pháp cũ mà sẽ nhiều hơn hoặc ít hơn tùy thuộc vào mức độ chuyển động của khối ảnh. Nhưng trong hầu hết các trường hợp được thử nghiệm thì số lượng điểm của giải thuật FDGDS sẽ ít hơn số điểm cần kiểm tra trong phương pháp cũ, đặc biệt khi mức độ chuyển động của các đối tượng trong video không cao.

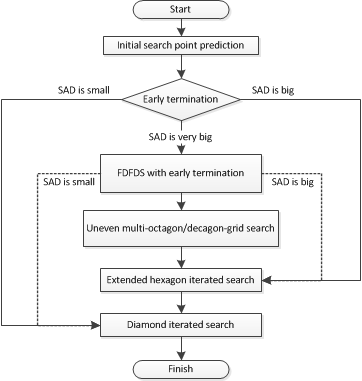
#### **3.1.2.2 Sử dụng mẫu tìm kiếm mới**

Trong giải thuật mới, nhóm Chiến và Cảnh đã sử dụng hai mẫu tìm kiếm mới là dạng lục giác 12 điểm và thập giác 12 điểm. Rõ ràng việc sử dụng hai mẫu tìm kiếm mới đã giảm việc tìm kiếm đi 4 điểm ở mỗi lượt. Mặc khác, hai mẫu tìm kiếm này được sử dụng xen kẽ nhau đảm bảo rằng các điểm tìm kiếm sẽ trải đều trên toàn bộ vùng tìm kiếm giống như mẫu lục giác 16 điểm. Hình 3.1 thể hiện hai mẫu tìm kiếm được sử dụng trong giải thuật mới. Không giống như mẫu tìm kiếm 8 điểm trong [12], hai mẫu tìm kiếm mới sẽ vẫn giữ được độ chính xác gần giống như mẫu tìm kiếm lục giác 16 điểm mà vẫn cải thiện được tốc độ tìm kiếm.



***Hình 3.1: Hai mẫu tìm kiếm mới: a) Bát giác 12 điểm và b) Thập giác 12 điểm***

Toàn bộ quá trình tìm kiếm của giải thuật ước lượng chuyển động mới được minh họa trong hình 3.2.



***Hình 3.2: Quá trình tìm kiếm của giải thuật do Chiến và Cảnh đề xuất***

## 3.2 Cài đặt các bộ nén

Các bộ nén được cài đặt trên phần mềm JM Reference Software phiên bản 19. Đây là phần mềm được phát triển bởi JVT chuyên dùng cho các nhà nghiên cứu trong lĩnh vực nén video để cài đặt và so sánh các giải thuật mới dựa trên nền tảng là chuẩn H.264.

Cấu hình phần mềm JM Reference Software khi thực hiện chạy các giải thuật tìm kiếm được lưu trong một tập tin có phần mở rộng là “cfg”, một số cấu hình được sử dụng khi chạy thực nghiệm giải thuật cải tiến được trình bày như bảng 3.1.

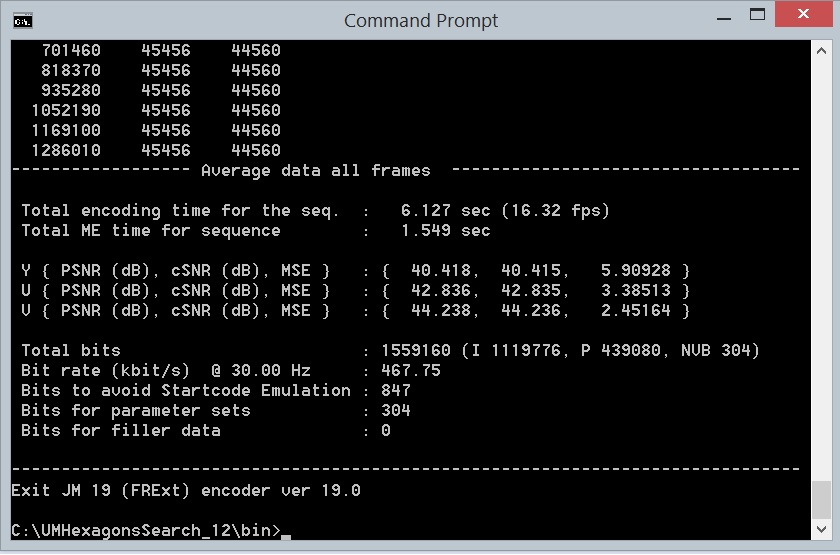
|  |  |  |
| --- | --- | --- |
| **Cấu hình** | **Giá trị** | **Giải thích** |
| FramesToBeEncoded | 100 | Số lượng khung hình sẽ được nén |
| FrameRate | 30 | 30 khung hình/giây |
| SourceWidth | 352 | Độ rộng khung hình của video nguồn |
| SourceHeight | 288 | Độ cao khung hình của video nguồn |
| IntraPeriod | 4 | Trong 4 khung hình liên tiếp sẽ có một khung hình I |
| DisableSubpelME | 1 | Tắt chế độ ước lượng chuyển động Subpel |
| SearchRange | 16 | Kích thước vùng tìm kiếm |
| MEDistortionFPel | 0 (SAD) | Chọn SAD là chỉ số đánh giá sự tương đồng |
| NumberReferenceFrames | 1 | Số lượng khung hình tối đa được tham chiếu trong khung hình P |
| NumberBFrames | 0 | Không dùng khung hình B |
| UseWeightedReferenceME | 0 | Không sử dụng WeightedReferenceME |
| RDOptimization | 0 | Tắt chế độ tối ưu Rate-Distortion |
| HMEEnable | 0 | Tắt chế độ ước lượng chuyển động có cấu trúc |
| RateControlEnable | 0 | Tắt chế độ Rate-Control |

Bảng 3.1: Cấu hình phần mềm JM Reference Software

Phần mềm được thực thi thông qua giao diện Command Line của Windows bằng cách thực thi lệnh với cú pháp như sau: *lencod.exe –f “<địa chỉ file cấu hình>” –p InputFile=“<địa chỉ file video>”* *[–p SearchMode=<mã thuật toán tìm kiếm>]*. Với mã thuật toán tìm kiếm như bên dưới. Mặc định :

* 1 = UMHexagon Search;
* 2 = UMHexagon Search eight directions;
* 3 = UMHexagon Search twelve directions type 1;
* 4 = UMHexagon Search twelve directions type 2;
* 5 = UMHexagon Search sixteen directions.

Sau khi thực thi với giao diện trên, JM Reference Software sẽ xuất ra thông tin chi tiết về kết quả của quá trình nén, các thông số đạt được như Bitrate, PSNR, thời gian ước lượng chuyển động… như mô tả trong hình 3.3.



***Hình 3.3: Kết quả sau khi chạy phần mềm thực nghiệm***

Máy tính dùng để chạy phần mềm thực nghiệm có cấu hình như trong bảng 3.2.

|  |  |
| --- | --- |
| Bộ nhớ RAM | 8GB DDR3 1600 MHz |
| CPU | Intel Core i5 3320M (2.6GHz, Turbo Boost 3.3GHz, 3MB Cache) |
| Hệ điều hành | Windows 8.1 Pro 64-bit |
| Graphics | Intel HD 4000 |

Bảng 3.2: Cấu hình máy tính chạy phần mềm thực nghiệm

## 3.3 Kết quả thực nghiệm

Các video mẫu dùng để kiểm tra bộ nén được download tại [4], đây là những đoạn video gốc chưa qua xử lý và thường xuyên được các nhà nghiên cứu về video đem ra kiểm nghiệm kết quả nghiên cứu. Trong bài báo cáo này, em tìm hiểu kết quả thực nghiệm từ các đoạn video có kích thước , định dạng YUV với mức độ chuyển động trong từng video được mô tả trong bảng 3.3.

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên video** | **Chuyển động** |
| 1 | Akiyo | Chậm |
| 2 | Coastguard | Trung bình |
| 3 | Stefan | Nhanh |
| 4 | Container | Chậm |
| 5 | Foreman | Nhanh |
| 6 | News | Trung bình |

Bảng 3.3: Các đoạn video mẫu

Có ba tiêu chí dùng để so sánh hiệu quả của ba bộ nén mới so với UMHexagonS và UMHexagonS cải tiến, đó là:

* Chỉ số đánh giá chất lượng PSNR;
* Bitrate của video sau khi nén. Đơn vị tính bằng kbit/s;
* Thời gian thực thi việc ước lượng chuyển động. Đơn vị tính bằng giây (s).

Kết quả so sánh được thể hiện trong bảng 3.4, 3.5, 3.6, 3.7, 3.8, 3.9 và được minh họa qua biểu đồ trong hình 3.6, 3.7 như bên dưới.

Kết quả thực nghiệm được thực hiện lần lược với các video mẫu như bảng 3.3 ở trên.

Vì thời gian ước lượng chuyển động (MET) của thuật toán trên từng máy tính khác nhau sẽ khác nhau, thời gian của mỗi lần nén cũng khác nhau và còn tùy thuộc vào cấu hình của máy tính thực nghiệm. Vì vậy, để việc tính MET được khách quan thì tính hiệu quả của mỗi bộ nén bằng cách tính thời gian trung bình () của 5 lần nén liên tục. Cách tính trung bình của như sau:

* **Akio**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Akiyo** | **UMHexagonS** | **UMHexagonS**  **cải tiến** | **UMHexagonS**  **12 directions**  **loại 1** | **UMHexagonS**  **12 directions**  **loại 2** | **UMHexagonS**  **16 directions** |
| MET | 1.514  1.549  1.521  1.519  1.514 | 1.519  1.501  1.498  1.483  1.487 | 1.499  1.494  1.488  1.499  1.492 | 1.491  1.492  1.504  1.485  1.493 | 1.506  1.512  1.511  1.513  1.511 |
|  | 1.5234 | 1.7746 | 1.4944 | 1.493 | 1.5106 |
| PSNR (Y) | 40.418 | 40.418 | 40.418 | 40.418 | 40.418 |
| PSNR (U) | 42.836 | 42.836 | 42.836 | 42.836 | 42.836 |
| PSNR (V) | 44.238 | 44.239 | 44.239 | 44.239 | 44.238 |
| Bitrate | 467.75 | 467.80 | 467.78 | 467.79 | 467.82 |

Bảng 3.4: Kết quả thực nghiệm các bộ nén với video mẫu Akiyo

* **Coastguard**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Akiyo** | **UMHexagonS** | **UMHexagonS**  **cải tiến** | **UMHexagonS**  **12 directions**  **loại 1** | **UMHexagonS**  **12 directions**  **loại 2** | **UMHexagonS**  **16 directions** |
| MET | 3.929  3.854  3.856  3.866  3.846 | 3.206  3.208  3.216  3.201  3.290 | 3.286  3.281  3.282  3.282  3.281 | 3.281  3.291  3.278  3.289  3.270 | 3.349  3.358  3.440  3.366  3.350 |
|  | 3.8702 | 3.2242 | 3.2824 | 3.2818 | 3.3726 |
| PSNR (Y) | 35.357 | 35.357 | 35.357 | 35.357 | 35.357 |
| PSNR (U) | 43.859 | 43.858 | 43.857 | 43.858 | 43.858 |
| PSNR (V) | 45.015 | 45.015 | 45.014 | 45.011 | 45.014 |
| Bitrate | 2204.74 | 2205.03 | 2205.01 | 2205.07 | 2204.53 |

Bảng 3.5: Kết quả thực nghiệm các bộ nén với video mẫu Coastguard

* **Stefan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Akiyo** | **UMHexagonS** | **UMHexagonS**  **cải tiến** | **UMHexagonS**  **12 directions**  **loại 1** | **UMHexagonS**  **12 directions**  **loại 2** | **UMHexagonS**  **16 directions** |
| MET | 3.177  3.186  3.183  3.210  3.314 | 2.707  2.709  2.712  2.710  2.697 | 2.782  2.767  2.774  2.787  2.763 | 2.791  2.782  2.810  2.766  2.763 | 2.833  2.853  2.835  2.841  2.995 |
|  | 3.214 | 2.707 | 2.7746 | 2.7824 | 2.8714 |
| PSNR (Y) | 36.576 | 36.577 | 36.576 | 36.577 | 36.577 |
| PSNR (U) | 38.650 | 38.649 | 38.650 | 38.647 | 38.646 |
| PSNR (V) | 38.698 | 38.697 | 38.693 | 38.695 | 38.696 |
| Bitrate | 2926.31 | 2932.01 | 2931.75 | 2932.57 | 2931.41 |

Bảng 3.6: Kết quả thực nghiệm các bộ nén với video mẫu Stefan

* **Container**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Akiyo** | **UMHexagonS** | **UMHexagonS**  **cải tiến** | **UMHexagonS**  **12 directions**  **loại 1** | **UMHexagonS**  **12 directions**  **loại 2** | **UMHexagonS**  **16 directions** |
| MET | 1.700  1.703  1.694  1.696  1.698 | 1.628  1.631  1.623  1.622  1.624 | 1.646  1.647  1.651  1.653  1.645 | 1.648  1.646  1.668  1.647  1.662 | 1.671  1.721  1.684  1.689  1.680 |
|  | 1.6982 | 1.6256 | 1.6484 | 1.6542 | 1.689 |
| PSNR (Y) | 37.092 | 37.091 | 37.091 | 37.091 | 37.091 |
| PSNR (U) | 42.192 | 42.192 | 42.192 | 42.192 | 42.192 |
| PSNR (V) | 42.280 | 42.281 | 42.281 | 42.281 | 42.281 |
| Bitrate | 994.18 | 994.16 | 994.14 | 994.13 | 994.13 |

Bảng 3.7: Kết quả thực nghiệm các bộ nén với video mẫu Container

* **Foreman**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Akiyo** | **UMHexagonS** | **UMHexagonS**  **cải tiến** | **UMHexagonS**  **12 directions**  **loại 1** | **UMHexagonS**  **12 directions**  **loại 2** | **UMHexagonS**  **16 directions** |
| MET | 2.717  2.719  2.718  2.719  2.706 | 2.346  2.346  2.357  2.361  2.349 | 2.427  2.423  2.421  2.428  2.429 | 2.416  2.413  2.429  2.422  2.421 | 2.491  2.493  2.506  2.510  2.498 |
|  | 2.7158 | 2.3518 | 2.4256 | 2.4202 | 2.4996 |
| PSNR (Y) | 37.403 | 37.402 | 37.403 | 37.404 | 37.403 |
| PSNR (U) | 41.288 | 41.292 | 41.294 | 41.293 | 41.291 |
| PSNR (V) | 43.979 | 43.976 | 43.979 | 43.977 | 43.979 |
| Bitrate | 1097.53 | 1102.26 | 1097.49 | 1096.91 | 1096.73 |

Bảng 3.8: Kết quả thực nghiệm các bộ nén với video mẫu Foreman

* **News**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Akiyo** | **UMHexagonS** | **UMHexagonS**  **cải tiến** | **UMHexagonS**  **12 directions**  **loại 1** | **UMHexagonS**  **12 directions**  **loại 2** | **UMHexagonS**  **16 directions** |
| MET | 1.768  1.767  1.813  1.759  1.766 | 1.660  1.668  1.663  1.660  1.667 | 1.684  1.679  1.682  1.684  1.685 | 1.691  1.690  1.692  1.675  1.688 | 1.706  1.713  1.710  1.716  1.702 |
|  | 1.7746 | 1.6636 | 1.6828 | 1.6872 | 1.7094 |
| PSNR (Y) | 39.003 | 39.002 | 39.003 | 39.002 | 39.002 |
| PSNR (U) | 41.493 | 41.493 | 41.496 | 41.495 | 41.496 |
| PSNR (V) | 42.507 | 42.506 | 42.507 | 42.508 | 42.508 |
| Bitrate | 809.90 | 809.93 | 809.89 | 809.94 | 809.95 |

Bảng 3.9: Kết quả thực nghiệm các bộ nén với video mẫu News

***Hình 3.4: Biểu đồ so sánh giữa các bộ nén video***

***Hình 3.5: Biểu đồ so sánh bitrate giữa các bộ nén video***

Lần lượt gọi các bộ nén *UMHexagonS cải tiến, UMHexagonS 12 directions loại 1, UMHexagonS 12 directions loại 1, UMHexagonS 16 directions* là UMHexagonS [1, 2, 3, 4]. Và .

Ta có, độ giảm thời gian được tính như sau:

Và độ tăng bitrate của bộ nén mới cũng được tính tương tự như sau:

Qua các bảng kết quả từ 3.4 đến 3.10 và biểu đồ 3.6, 3.7, ta có thể thấy được hiệu quả của các bộ mới so với bộ nén UMHexagonS về mặt thời gian ước lượng chuyển động. Đặc biệt, các đoạn video chứa nhiều chuyển động nhanh như Foreman, Stefan hay Coastguard cho kết quả rõ rệt hơn so với bộ nén UMHexagonS với thời gian chạy nhanh hơn đến gần 17%. Trong khi đó, những đoạn video chứa ít chuyển động thì tốc độ có thay đổi nhưng không nhiều như các đoạn video khác, chỉ khoảng 4.3%. Mặt khác, chỉ số PSNR có chiều hướng tăng khi số hướng tìm kiếm tăng lên. Bộ nén UMHexagonS 12 directions loại 1 có hiệu quả nhất so với các bộ nén còn lại với bitrate đều giảm ở tất cả các video và đều nhỏ hơn các bộ nén còn lại. Bitrate tăng lớn nhất là 0.07% đối với bộ nén UMHexagonS 16 directions so với bộ nén UMHexagonS khi nén video Foreman, thậm chí bitrate còn có xu hướng giảm ở một số video như Coastguard, Stefan hoặc Foreman đối với bộ nén UMHexagonS cải tiến.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **UMHexagonS 1** | **UMHexagonS 2** | **UMHexagonS 3** | **UMHexagonS 4** |
| Akiyo | Độ tăng bitrate (kbit/s) | 0.01% | 0.01% | 0.01% | 0.01% |
| Tốc độ cải thiện (s) | -16.49% | 1.90% | 2.00% | 0.84% |
| Coastguard | Độ tăng bitrate (kbit/s) | -0.01% | -0.01% | -0.01% | 0.01% |
| Tốc độ cải thiện (s) | 16.69% | 15.19% | 15.20% | 12.86% |
| Stefan | Độ tăng bitrate (kbit/s) | -0.19% | -0.19% | -0.21% | -0.17% |
| Tốc độ cải thiện (s) | 15.77% | 13.67% | 13.43% | 10.66% |
| Container | Độ tăng bitrate (kbit/s) | 0.00% | 0.00% | 0.01% | 0.01% |
| Tốc độ cải thiện (s) | 4.28% | 2.93% | 2.59% | 0.54% |
| Foreman | Độ tăng bitrate (kbit/s) | -0.43% | 0.00% | 0.06% | 0.07% |
| Tốc độ cải thiện (s) | 13.40% | 10.69% | 10.88% | 7.96% |
| News | Độ tăng bitrate (kbit/s) | 0.00% | 0.00% | 0.00% | -0.01% |
| Tốc độ cải thiện (s) | 6.25% | 5.17% | 4.93% | 3.67% |

Bảng 3.10: Bảng so sánh kết quả giữa ba bộ nén mới và UMHexagonS cải tiến so với UMHexagonS

# CHƯƠNG 4: TỔNG KẾT

Bài báo cáo đã trình bày một số khái niệm về dữ liệu video, thuật toán ước lượng chuyển động.

Tìm hiểu những phân tích đánh giá về các giải thuật ước lượng chuyển động theo khối và tham khảo các tài liệu về những thuật toán của ba bộ nén mới từ bộ nén UMHexagonS cải tiến, một bộ nén được cải tiến từ giải thuật UMHexagonS.

Ba bộ nén mới được cài đặt thử nghiệm trên phần mềm JM Reference Software phiên bản 19, một phần mềm thường được sử dụng bởi các chuyên gia trong lĩnh vực nén video nhằm kiểm tra các bộ nén mới. Kết quả của ba bộ nén đã được so sánh và đánh giá so với bộ nén UMHexagonS và UMHexagonS cải tiến. Ba bộ nén mới và UMHexagonS cải tiến đạt được hiệu quả rõ ràng về mặt thời gian. Riêng bộ nén UMHexagonS 12 directions loại 1 cho hiệu quả nén tối ưu hơn các bộ nén còn lại, còn bộ nén UMHexagonS cải tiến lại có tốc độ nén nhanh nhất so với các bộ nén còn lại.

# TÀI LIỆU THAM KHẢO

1. Trần Xuân Chiến và Phan Công Cảnh (2012), “Nghiên cứu và cải tiến thuật toán ước lượng chuyển động trong nén video”, *trường Đại học Công Nghệ Thông Tin*, Đại học Quốc Gia thành phố Hồ Chí Minh, tháng 7 năm 2012.
2. (2006), “A Novel Search Algorithm for Variable Block Size Motion Estimation of H.264/AVC”, March 2006.
3. Adan A. Mustafa and Mark A.Ganter (1994), “Efficient Image Registration by Intensity Combinatorial Minimization”, University of Washington Seattle, Washington 98195 in 1994.
4. Arizona State University (2015), *YUV video sequences*, website: <http://trace.eas.asu.edu/yuv/>.
5. Chunjiang Duanmu and Yu Zhang, The Limited Company of Planning and Design of Post and Telecommunications (2012), “A New Fast Block Motion Algorithm Based on Octagon and Triangle Search Patterns for H.264/AVC”, *International Journal of Digital Content Technology and its Applications (JDCTA)* Volume 6, Number 10, June 2012.
6. Cisco system (2012), “Forecast and Methodology, 2011­2016”, *Cisco Visual Networking Index,* this document is Cisco Public May 30, 2012.
7. Jeffrey P. Kern and Marios S. Pattichis (2007), “Robust Multispectral Image Registration Using Mutual-Information Models”, IEEE Transactions on Geoscience and Remote Sensing, VOL. 45, NO. 5, May 2007.
8. Ka-Ho Ng, Lai-Man Po, Ka-Man Wong, and Chi-Wang Ting (2009), “Multiple Block-Size Search Algorithm for Fast Block Motion Estimation”, *Department of Electronic Engineering, City University of Hong Kong*, Hong Kong SAR, China.
9. Koga (1981), T.K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing” *in* Proc.NTC81**,** New Orleans, LA, in 1981.
10. Lai-Man Po, Ka-Ho Ng, Kwok-Wai Cheung, Ka-Man Wong, Yusuf Md. Salah Uddin, and Chi-Wang Ting (2009), “Novel Directional Gradient Descent Searches for Fast Block Motion Estimation”, *IEEE Transactions On Circuits And Systems For Video Technology, Vol. 19, No. 8*, August 2009.
11. Lai-Man Po and Wing-Chung Ma (1996), “A novel four step search algorithm for fast block motion estimation”, *IEEE Trans. Circuits Syst. Video Technol*., vol. 6, No. 3, pp. 313-317, Jun 1996.
12. LI Hong-ye, LIU Ming-jun, and ZHANG Zhi-qiang (2009), “A New Fast Motion Estimation Algorithm Based on H.264” *Intenational Conference on Multimedia Information Networking and Security*, School of Information Science and Engineering University of Jinan, Jinan, China in 2009.
13. Pawan Lathwal (2015), “Line Searches for Fast Block Motion Estimation”, *Proceedings of National Conference on Innovative Trends in Computer Science Engineering (ITCSE-2015) held at BRCMCET*, Bahal on 4th April 2015.
14. Pengyu Liu and Kebin Jia (2011), “Research and Optimization of Low-Complexity Motion Estimation Method Based on Visual Perception”, *Journal of Information Hiding and Multimedia Signal Processing*, Ubiquitous International Volume 2, Number 3, July 2011.
15. N. S. Jayant (1973), “Adaptive Quantization With a One-Word Memory”, *The Bell System Technical Journal*, Vol. 52, No. 7. September, 1973.
16. Scott Firestone, Thiya Ramalingam, and Steve Fry (2007), “Voice and Video Conferencing Fundamentals”, Cisco Press: 800 East 96th Street Indianapolis, IN 46240 USA, 2007.
17. S. Immanuel Alex Pandian and Dr.G. Josemin Bala and Becky Alma George (2011), “A study on block matching algorithms for motion estimation,” *International Journal on Computer Science and Engineering,* vol. 3, pp. 34-44, Jan 2011.
18. Yao Wang, Joern Ostermann, and Ya-Qin Zhang (2002), “Video processing and communication”, *Prentice-Hall,* ISBN 0-13-017547-1 in 2002, updated 3/17/2015.
19. Yuan Gao, Peng-yu Liu and Ke-bin Jia (2013), “A Fast Motion Estimation Algorithm Based on Motion Vector Distribution Prediction”, journal of software, vol. 8, no. 11, november 2013.
20. Yung-Lyul Lee, Yung-Ki Lee and HyunWook Park (2006), “A Fast Motion Vector Search Algorithm for Variable Blocks”, J. Blanc-Talon et al. (Eds.): ACIVS 2006, LNCS 4179, pp. 311-322, in 2006.
21. Yun Q. Shi and Huifang Sun (2008), “fundamentals, algorithms, and standards”, *Image and video compression for multimedia engineering,* New Jersey Institute of Technology Newark, New Jersey and Mitsubishi Electric Research LaboratorieCambridge, Massachusetts in USA, 2008.

# 

# PHỤ LỤC